IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Kweku Danquah Otchere
Email: otcherekweku@gmail.com

August 1, 2023

Outline

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

Executive Summary

- Summary of methodologies
  a) SpaceX Data Collection using the SpaceX API
  b) SpaceX Data Collection using Web Scraping
  c) SpaceX Data Wrangling
  d) SpaceX Exploratory Data Analysis using SQL
  e) SpaceX Exploratory Data Analysis (Data visualization using Pandas and Matplotlib)
  f) SpaceX Interactive Visual Analytics and Dashboard using Folium and Plotly Dash
  g) SpaceX Predictive Analysis (Classification)

- Summary of all results
  a) Exploratory Data Analysis results
  b) Interactive Visual Analytics and Dashboards
  c) Machine learning (Predictive Analysis)

Introduction

- ## Project background and context

With the commercial space age upon us, companies seek to make space travel more affordable for everyone. SpaceX is arguably the most successful of all the companies.

Other companies provides space travel services include:

  - Virgin Galactic, a company providing suborbital spaceflights
  - Rocket Lab, a company that is into the provision of small satellites
  - Blue Origin manufactures sub-orbital and orbital reusable rockets

SpaceX accomplishments include:

  - Sending spacecraft to the International Space Station
  - Starlink, a satellite internet constellation providing satellite internet access
  - Sending manned missions to space

One reason SpaceX is able to do this is the rocket launches are relatively inexpensive. SpaceX makes a record $100 million savings as compared to its competitors in the launches of its Falcon 9 rocket as advertised on its website. This is attributed to the reusability of the first stage. A determination that the first stage will land successfully can provide an indication of the cost at launch, which can be useful for any company that wants to bid against SpaceX for a rocket launch.

SpaceX's Falcon 9 launch is like that of regular rockets. The first stage of the rocket launch does most of the work and is much larger than the second stage. Unlike other competitors, Falcon9 is able to recover the first stage.

The second stage, helps bring the payload to orbit, but most of the work is done by the first stage.

- ## Problems you want to find answers

  - This projects seeks to predict the likelihood that the first stage of the Falcon 9 first stage will land successfully using data from previous launches and determine the impact of related factors on such successes.
  - The project also seeks to determine the price of each launch.
  - Another determination is to know if SpaceX will reuse the first stage. This will be done by training a machine learning model and usage of public information to predict if SpaceX will reuse the first stage.

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

How was SpaceX Falcon9 data collected?

1. SpaceX API

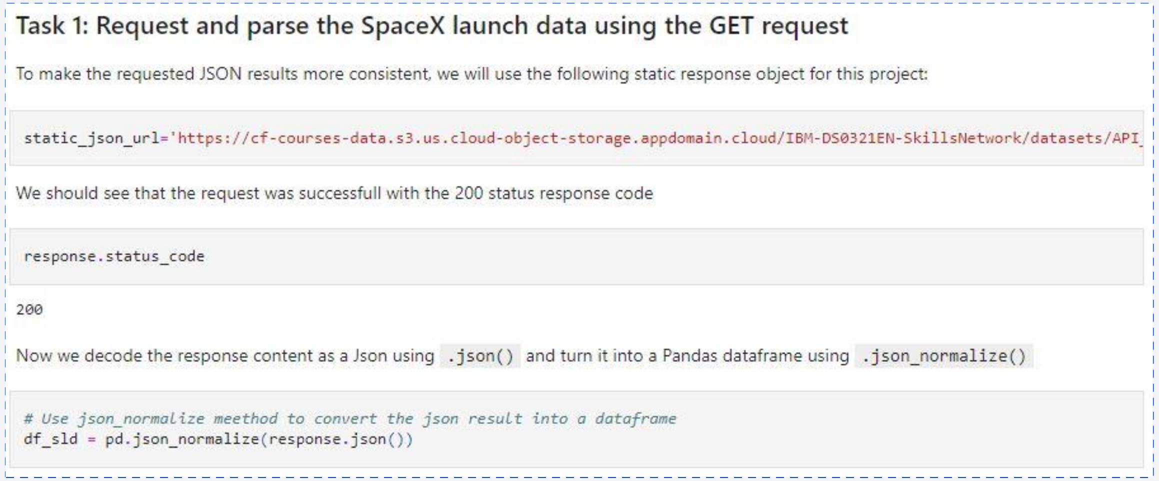
Data was collected by making a get request to the Restful API, SpaceX API. The response from the API call provide massive information about SpaceX launches. Data retrieved from the API call includes information on the rocket, payloads, launchpad, date, flight number among others

2. Web scraping

From a page titled “List of Falcon 9 and Falcon Heavy launches” on Wikipedia, extraction of Falcon 9 historic launch records was made using web scraping. One helpful library, BeautifulSoup, was utilized in this exercise of extracting data from the web page.

Data Collection – SpaceX API

- A get request was made to the SpaceX API via the url: (“https://api.spacexdata.com/v4/launches/past”) to retrieve rocket launch data. A response code of 200 indicated the response was successful. The response content was decoded using Json and turned into a Pandas dataframe.
- Click to view the completed SpaceX API calls notebook on GitHub.



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

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

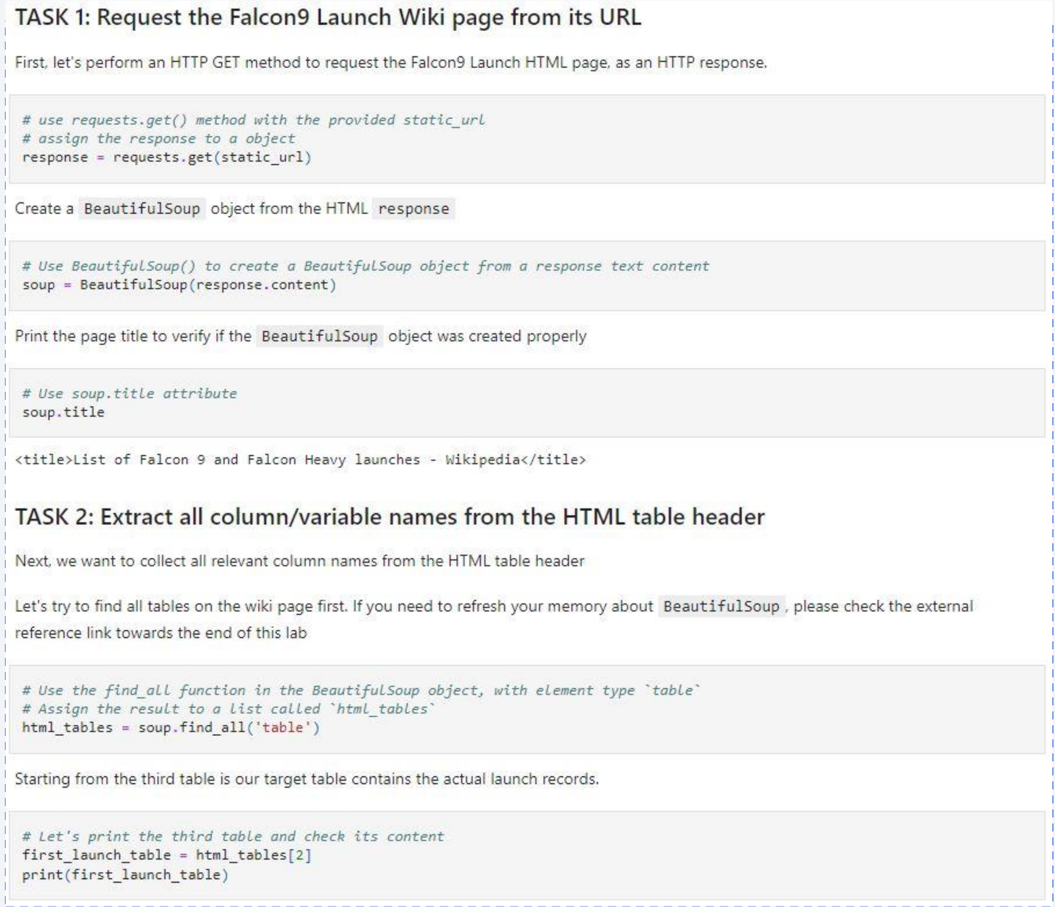
Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
# Use json_normalize meethod to convert the json result into a dataframe
df_sld = pd.json_normalize(response.json())
```

Data Collection - Scraping

- HTTP GET method to request the Falcon 9 launch HTML page as an HTTP response was performed. Falcon 9 launch records were extracted from HTML table from Wikipedia. A beautifulSoup objected was created from the HTML response. The table was parsed and converted into a Pandas data frame for manipulation.

- Click to view the completed web scraping notebook on GitHub.



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
response = requests.get(static_url)
```

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

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

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

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

```
<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

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

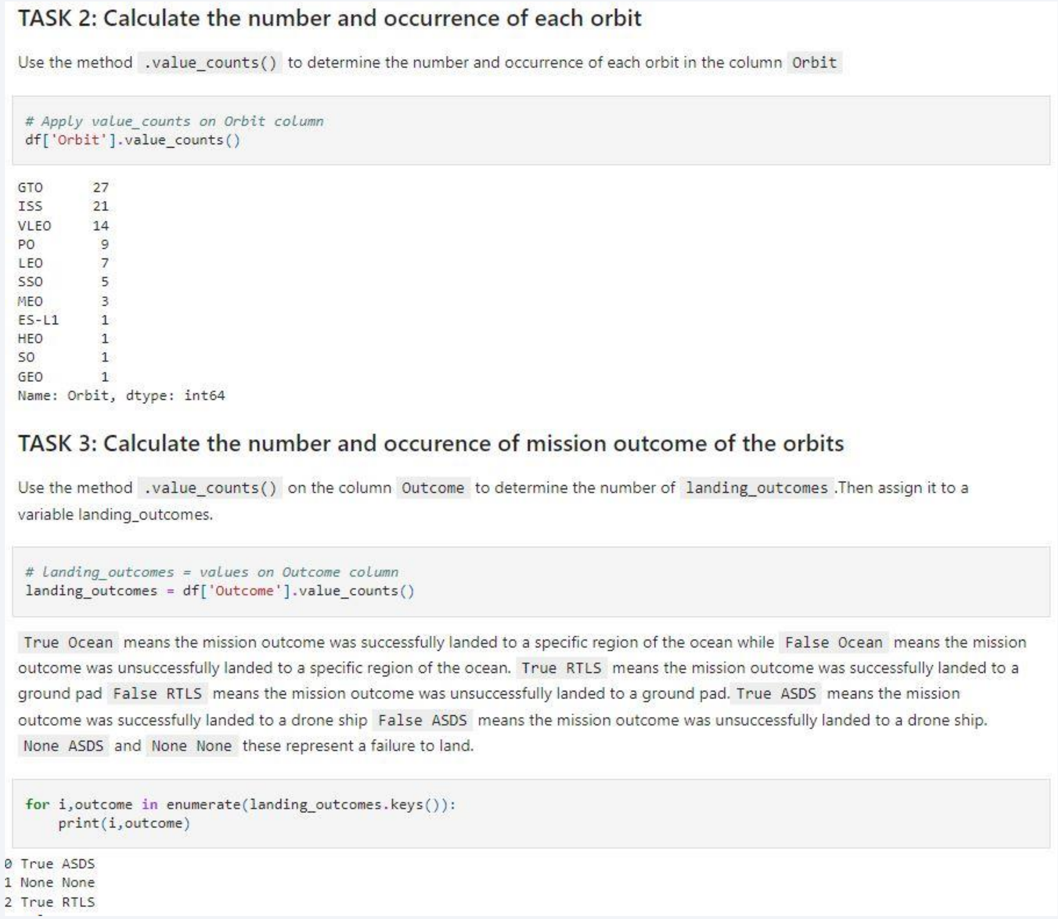
Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

Data Wrangling

- The data in the pandas data frame was filtered using the *BoosterVersion* column to show only Falcon 9 launches. There were cases of missing data in the *LandingPad* and *PayloadMass* columns which needed to be rectified. The mean of the *PayloadMass* was calculated and used to replace missing data in that column. Landing outcomes was converted to either 0 or 1 to depict failure or success respectively.

- Click to view the completed data wrangling related notebook on GitHub.



TASK 2: Calculate the number and occurrence of each orbit

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

TASK 3: Calculate the number and occurence of mission outcome of the orbits

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then assign it to a variable landing_outcomes.

```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
```

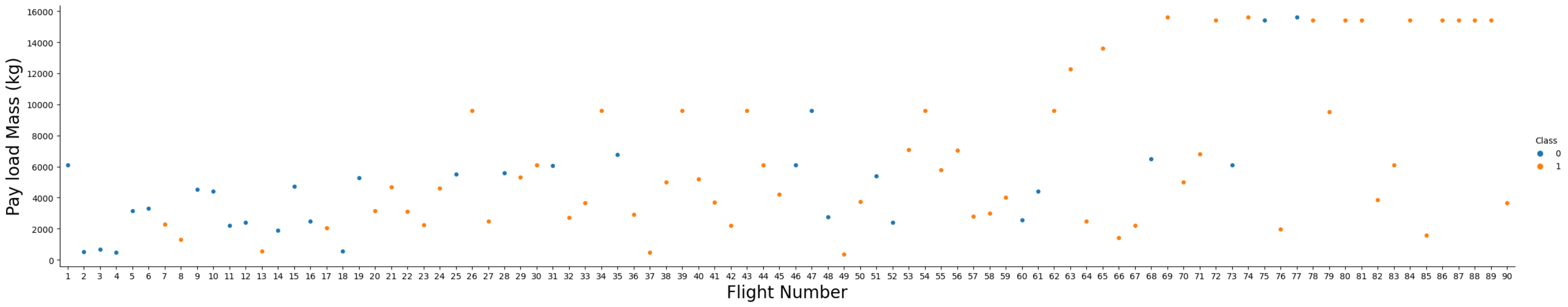
`True Ocean` means the mission outcome was successfully landed to a specific region of the ocean while `False Ocean` means the mission outcome was unsuccessfully landed to a specific region of the ocean. `True RTLS` means the mission outcome was successfully landed to a ground pad `False RTLS` means the mission outcome was unsuccessfully landed to a ground pad. `True ASDS` means the mission outcome was successfully landed to a drone ship `False ASDS` means the mission outcome was unsuccessfully landed to a drone ship. `None ASDS` and `None None` these represent a failure to land.

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
```

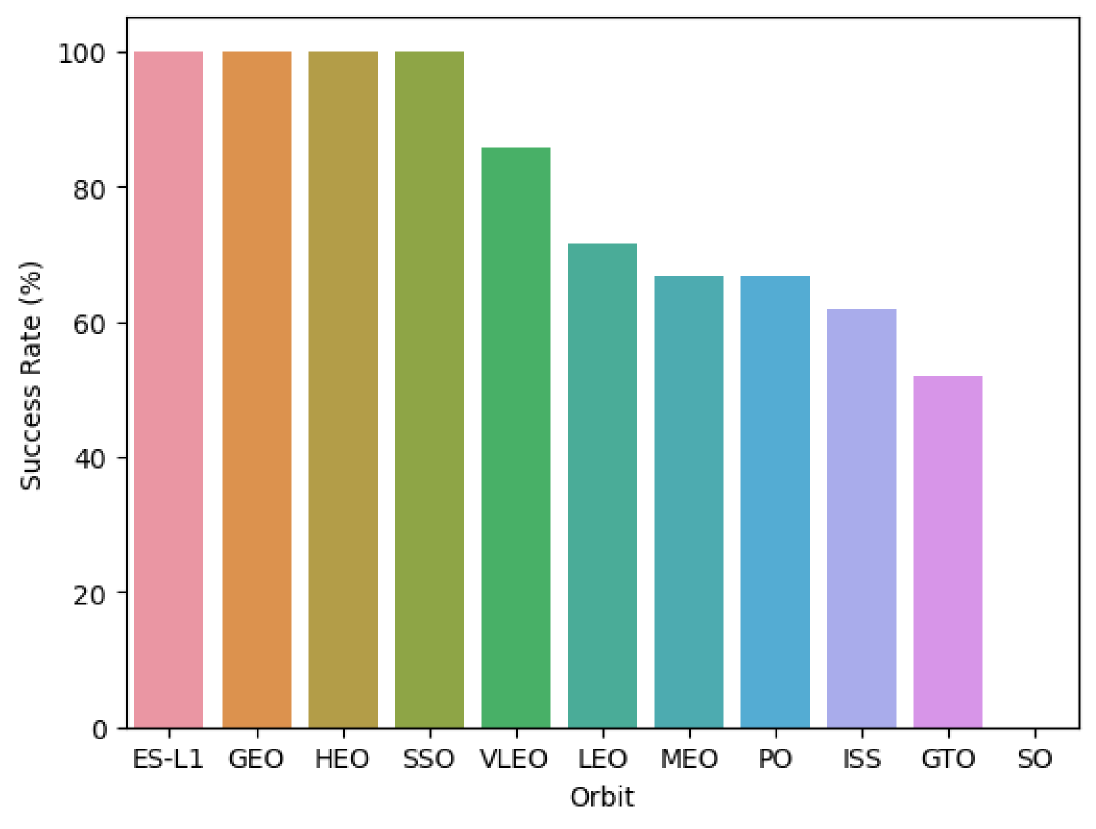
EDA with Data Visualization

- This stage involved Data Analysis and Feature Engineering using Pandas and Matplotlib.
- Scatter plots were used to visualize the relationship between Flight number and Payload mass, Payload and Launch site among others.



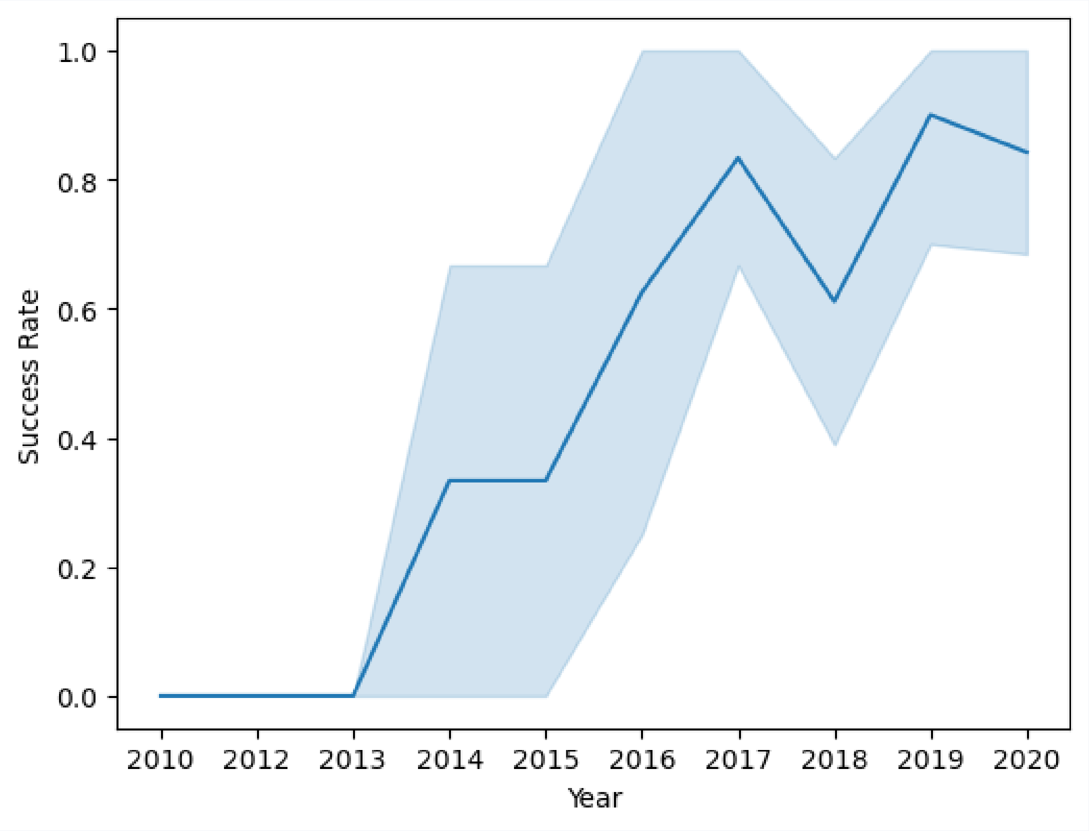
EDA with Data Visualization Cont'd

- A bar chart was used to visualize the relationship between success rate of each orbit type.
- It is observed that, orbits such as ES-L1, GEO, HEO and SSO have high success rates as compared to the others.
- The Orbit SO, has a very low success rate. Its success rate is the lowest among the orbits.



EDA with Data Visualization Cont'd

- A line chart was used to visualize the extracted year and the success rate.
- It is observed that, there was a rising success rate trend from the year 2013.
- In 2018, there was a fall in success rate but it however increased in 2019.

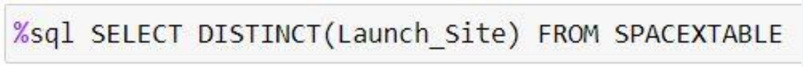


- Click to view the Exploratory Data Analysis with Data Visualization notebook on GitHub.

EDA with SQL

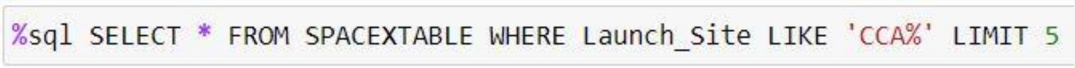
- Below are SQL queries performed for EDA:

1. Display the names of the unique launch sites in the space station



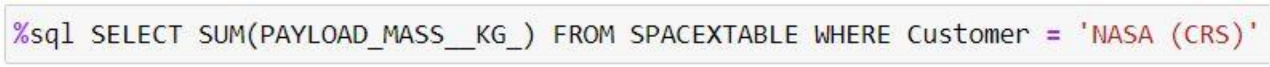
```
%sql SELECT DISTINCT(Launch_Site) FROM SPACEXTABLE
```

2. Display 5 records where launch sites begin with the string ‘CCA’



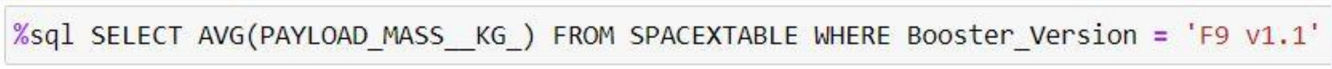
```
%sql SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5
```

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



```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)'
```

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

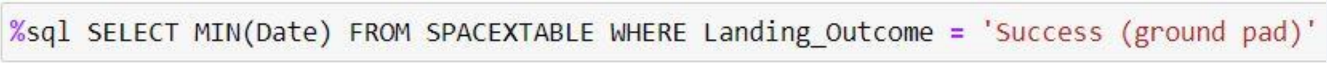


```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'
```

EDA with SQL Cont'd

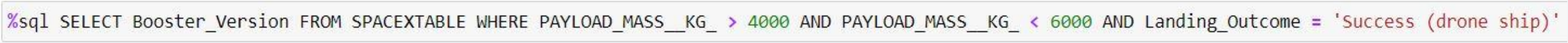
- Below are SQL queries performed for EDA:

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



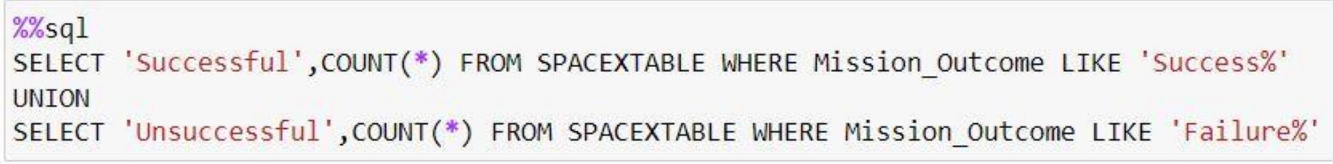
```
%sql SELECT MIN(Date) FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)'
```

6. 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 Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000 AND Landing_Outcome = 'Success (drone ship)'
```

7. List the total number of successful and failure mission outcomes

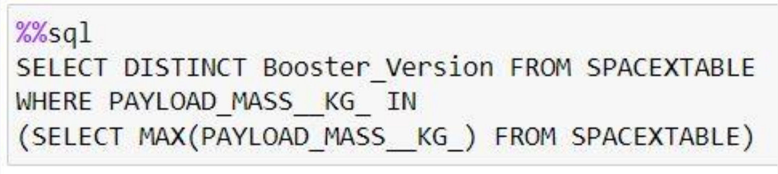


```
%%sql
SELECT 'Successful',COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE 'Success%'
UNION
SELECT 'Unsuccessful',COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE 'Failure%'
```

EDA with SQL Cont'd

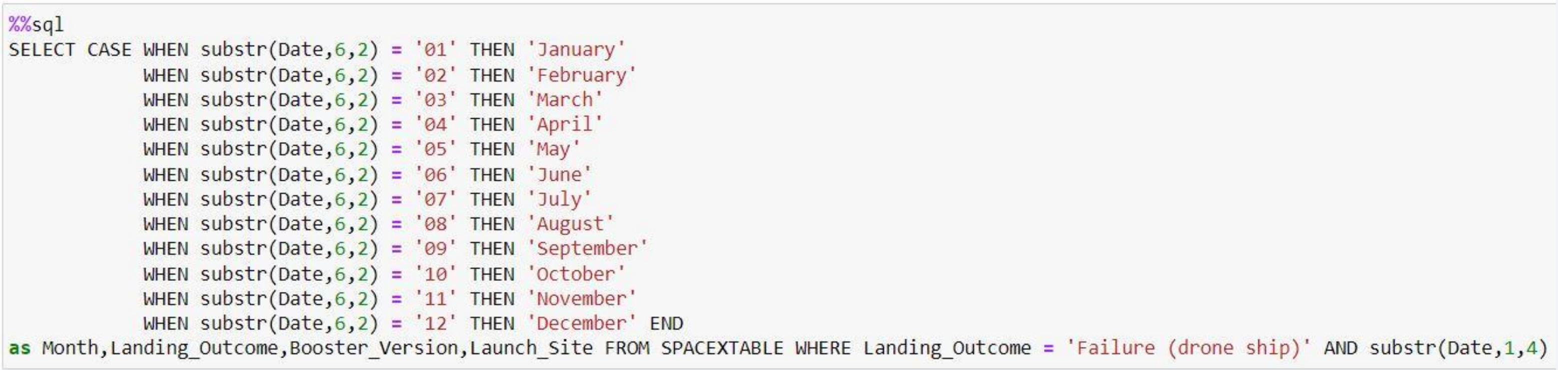
- Below are SQL queries performed for EDA:

8. List the names of the booster_versions which have carried the maximum payload mass. Use a subquery



```
%%sql
SELECT DISTINCT Booster_Version FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ IN
(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
```

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.

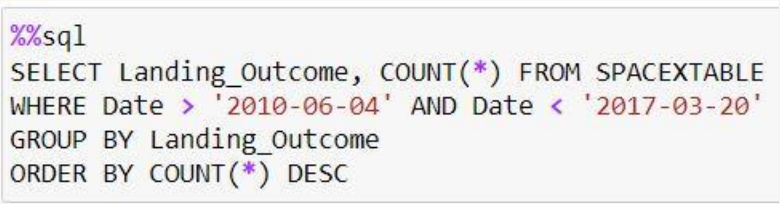


```
%%sql
SELECT CASE WHEN substr(Date,6,2) = '01' THEN 'January'
            WHEN substr(Date,6,2) = '02' THEN 'February'
            WHEN substr(Date,6,2) = '03' THEN 'March'
            WHEN substr(Date,6,2) = '04' THEN 'April'
            WHEN substr(Date,6,2) = '05' THEN 'May'
            WHEN substr(Date,6,2) = '06' THEN 'June'
            WHEN substr(Date,6,2) = '07' THEN 'July'
            WHEN substr(Date,6,2) = '08' THEN 'August'
            WHEN substr(Date,6,2) = '09' THEN 'September'
            WHEN substr(Date,6,2) = '10' THEN 'October'
            WHEN substr(Date,6,2) = '11' THEN 'November'
            WHEN substr(Date,6,2) = '12' THEN 'December' END
as Month,Landing_Outcome,Booster_Version,Launch_Site FROM SPACEXTABLE WHERE Landing_Outcome = 'Failure (drone ship)' AND substr(Date,1,4)
```

EDA with SQL Cont'd

- Below are SQL queries performed for EDA:

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.



```
%%sql
SELECT Landing_Outcome, COUNT(*) FROM SPACEXTABLE
WHERE Date > '2010-06-04' AND Date < '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY COUNT(*) DESC
```

Click to view the Exploratory Data Analysis with SQL notebook on GitHub.

Build an Interactive Map with Folium

- A folium map was created with all launch sites, map objects such as markers, circles and Polylines were used to mark the launch sites on the map, mark the success and failed launches for each site on the map and calculate the distances between a launch site to its proximities such as Railways, Highways, Coastlines and Cities.
- Launch outcomes were also indicated
- Click to view the Interactive Map with Folium notebook on GitHub.

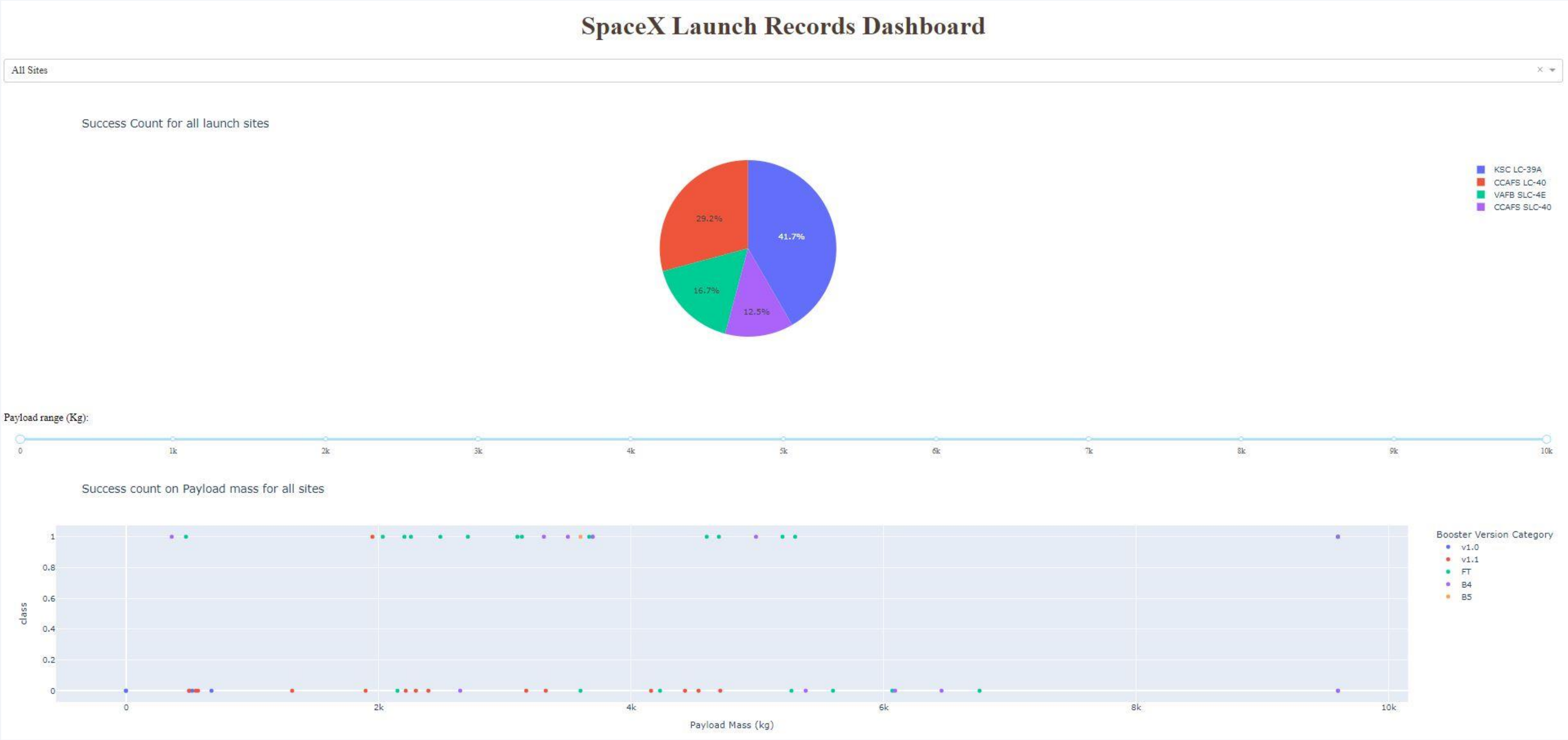
Build a Dashboard with Plotly Dash

- Below are the plots/graphs and interactions added to a dashboard:

  1. Added dropdown control for Launch site
  2. Added a callback function to render a success-pie-chart based on the selected launch site from the dropdown control
  3. Added a range slider for Payload selection
  4. Added a callback function to render the success-payload-scatter-chart plot

Click to view the Dashboard with Plotly Dash notebook on GitHub.

Build a Dashboard with Plotly Dash Cont’d



SpaceX Launch Records Dashboard

All Sites

Success Count for all launch sites

Payload range (Kg):

Success count on Payload mass for all sites

Predictive Analysis (Classification)

- Summary of how the best performing classification model was build, evaluated and improved.

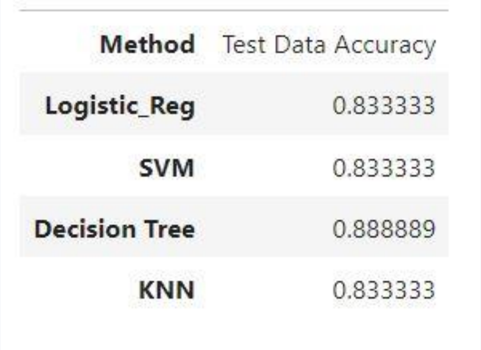
  a. The goal is to predict whether the first stage of Falcon 9 will land successfully
  b. Preprocessing allowed the standardization of the data. Standardization of the feature dataset (x) was done by a transformation using preprocessing.StandardScaler() function from Sklearn
  c. The data was then split into training and testing sets using the Train_test_split.
  d. The model was trained and we performed Grid Search allowing us to find the hyerparameters that allow a given algorithm to perform best. Created 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.
  e. Using the best hyperparameter values, we determined the model with the best accuracy using the training data
  f. We tested Logistic Regression, Support Vector Machines, Decision Tree Classifier and K-nearest neighbors.
  g. Finally, we output the confusion matrix.

- Finding the best Machine Learning model:

  a. Created an object for each algorithm. An object, GridSearchCV, was used and an assignment of parameters for each model done. GridSearchCV objtect had a cv – 10.
  b. After fitting the training set, output for the object, GridSearchCV was done for every model, then a display for the best parameters using the attribute best_score_.
  c. Testing was carried out using Logistic Regression, Support Vector Machine, Decision Tree Classifier and K-nearest neighbors. The Test data accuracy of all four models were determined.
  d. Accuracy of the test data was done for the models and a confusion matrix plotted using the test and predicted outcomes.
  e. All models show a relatively similar test data accuracy.

Predictive Analysis (Classification) Cont’d

- Test data accuracy score for each model



Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.888889
KNN	0.833333

- Click to view the Predictive Analysis (Classification) notebook on Github.

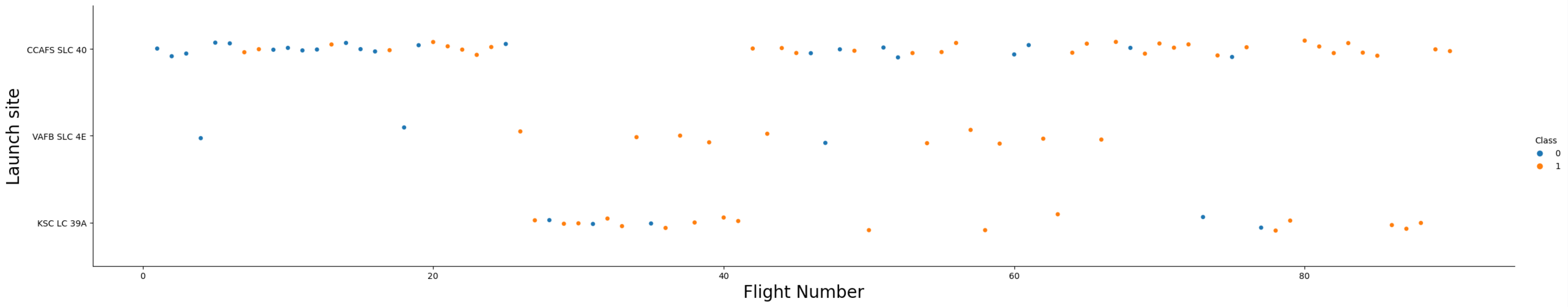
Results

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

Section 2

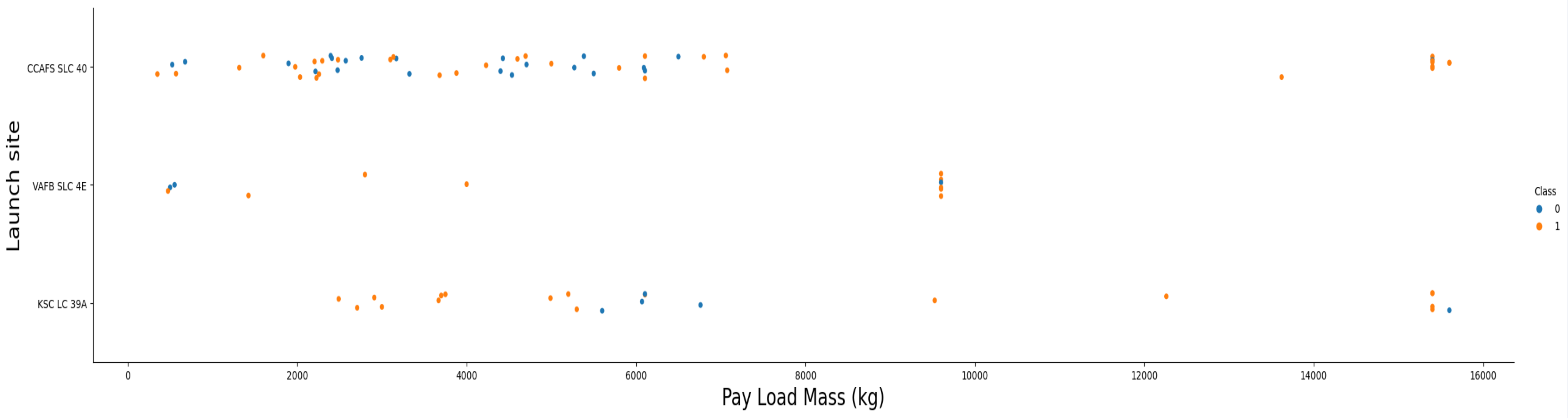
Insights drawn from EDA

Flight Number vs. Launch Site



It can be deduced that, as flight number increases, the success rate also increased. Launch sites KSC LC 39A and CCAFS SLC 40 both have a continuous success rate after the flight number 80. Launch site VAFB SLC 4E also has continuous success rate after flight 50.

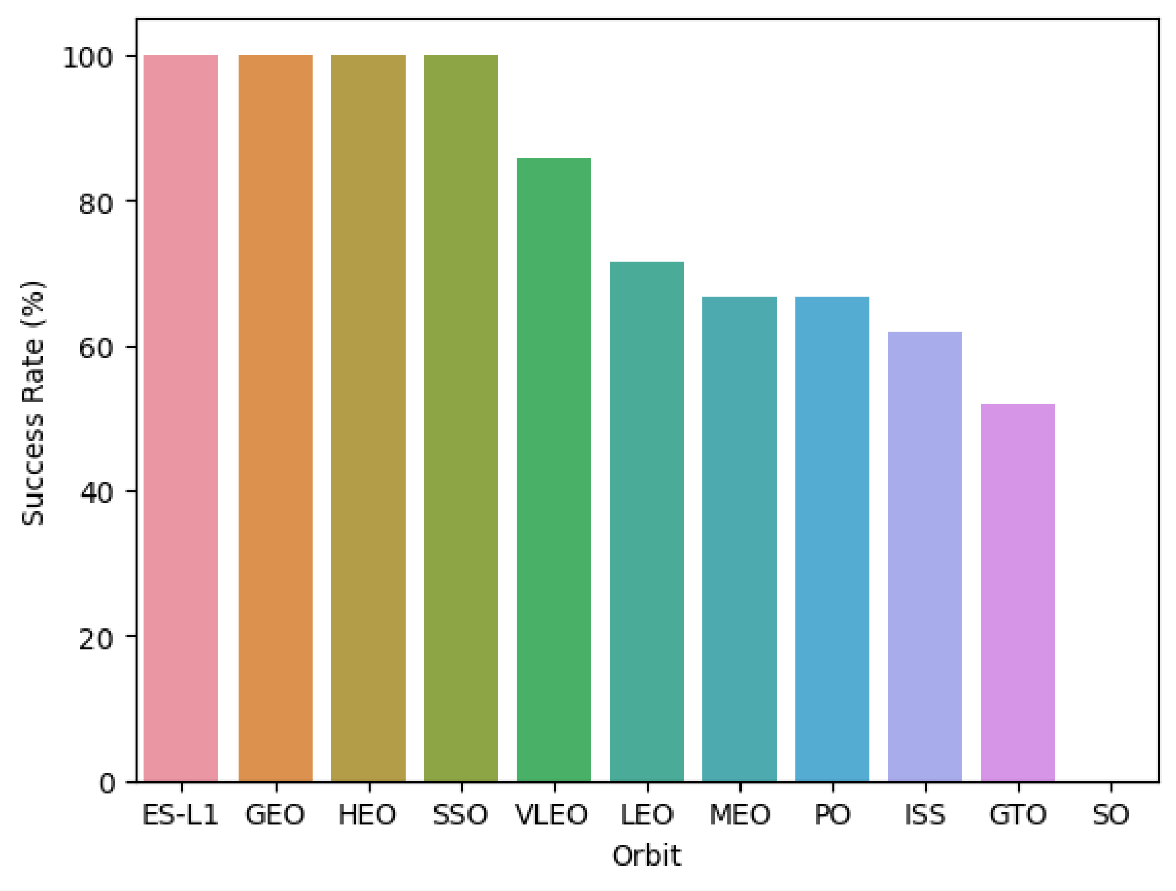
Payload vs. Launch Site



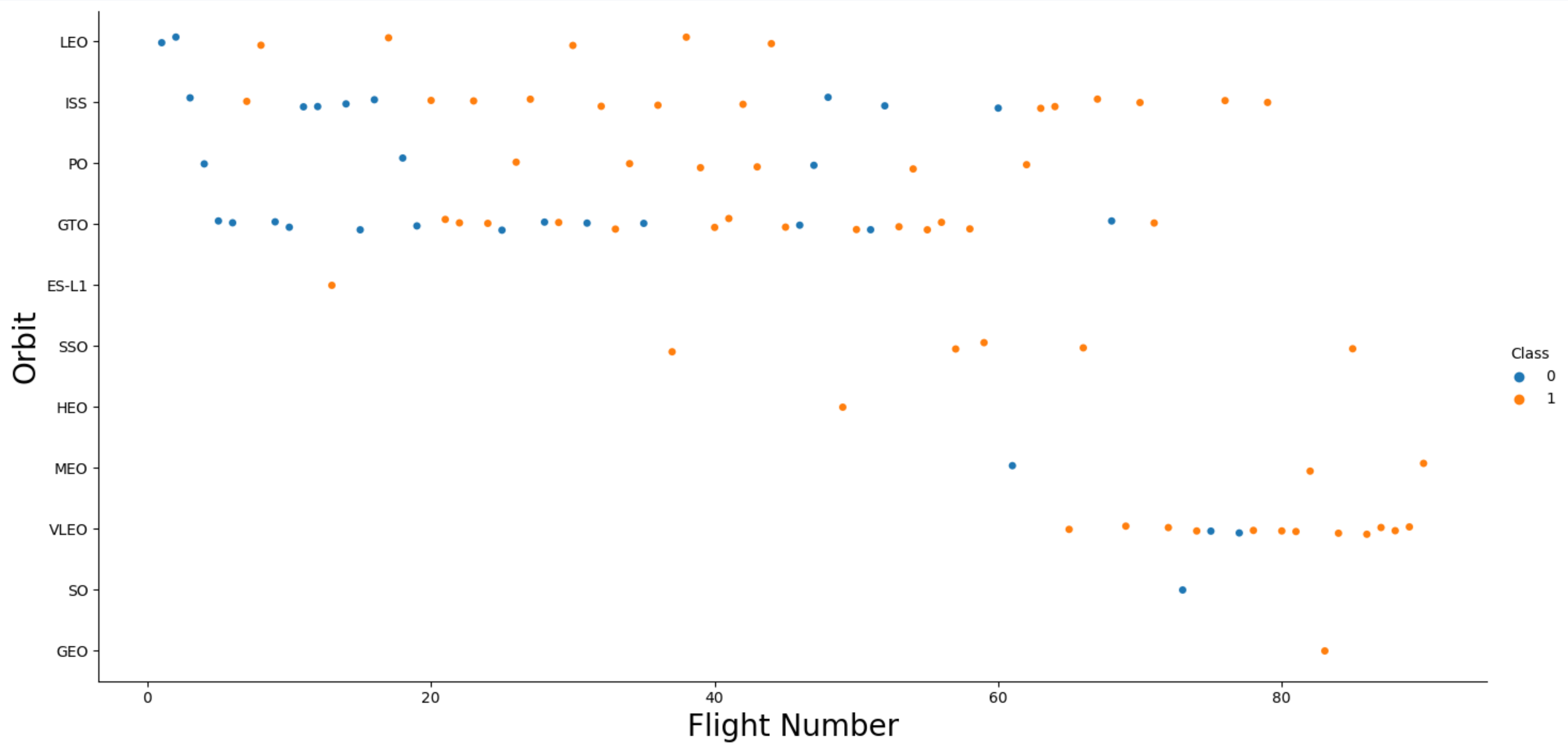
If you observe the Payload Vs. Launch Site scatter point chart, you will find for the VAFB-SLC launch site there are not rockets launched for heavypayload mass(greater than 10000)

Success Rate vs. Orbit Type

- Orbits ES-L1, GEO, HEO and SSO have success rates of 100%. Orbit SO has the lowest success rate.

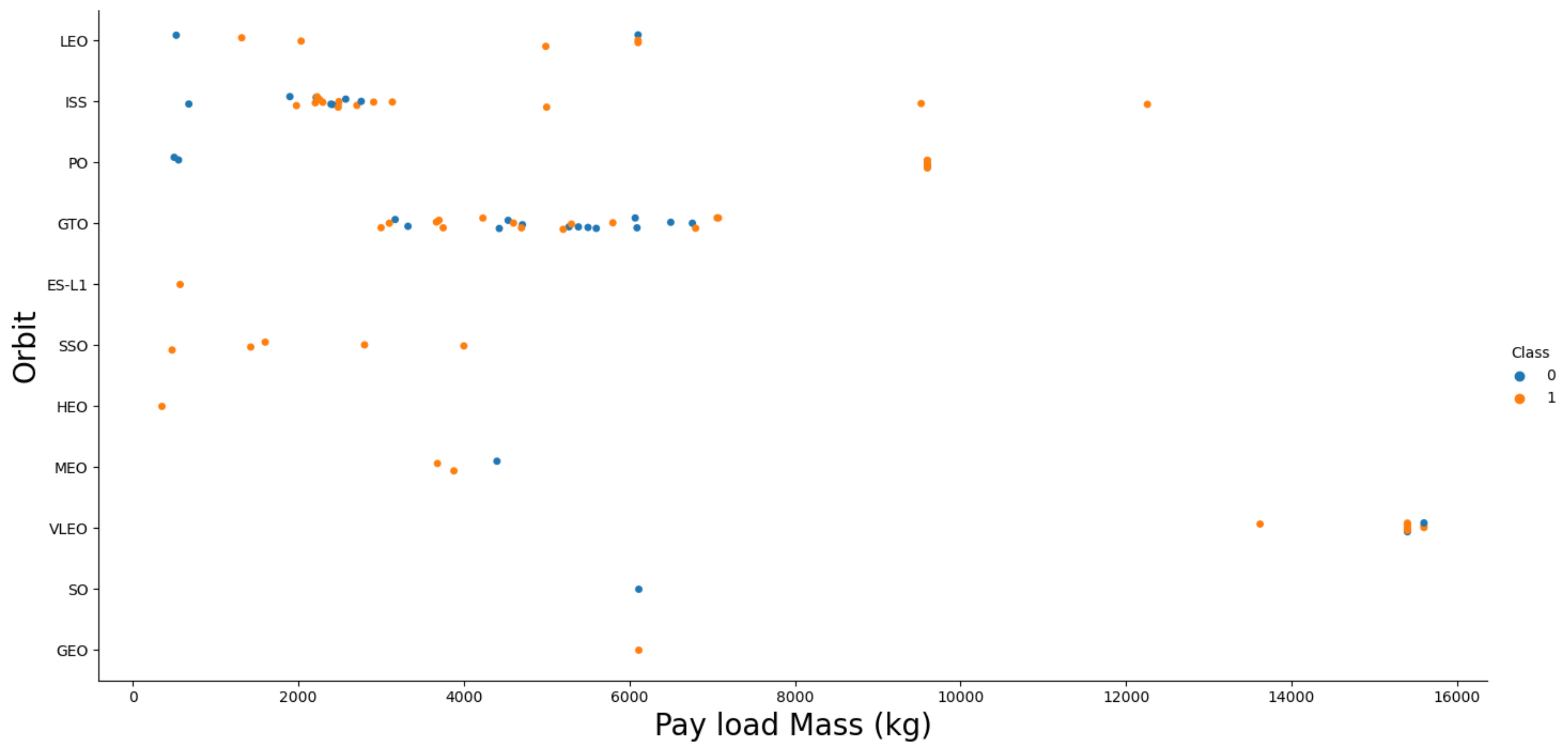


Flight Number vs. Orbit Type



In the LEO orbit the success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit

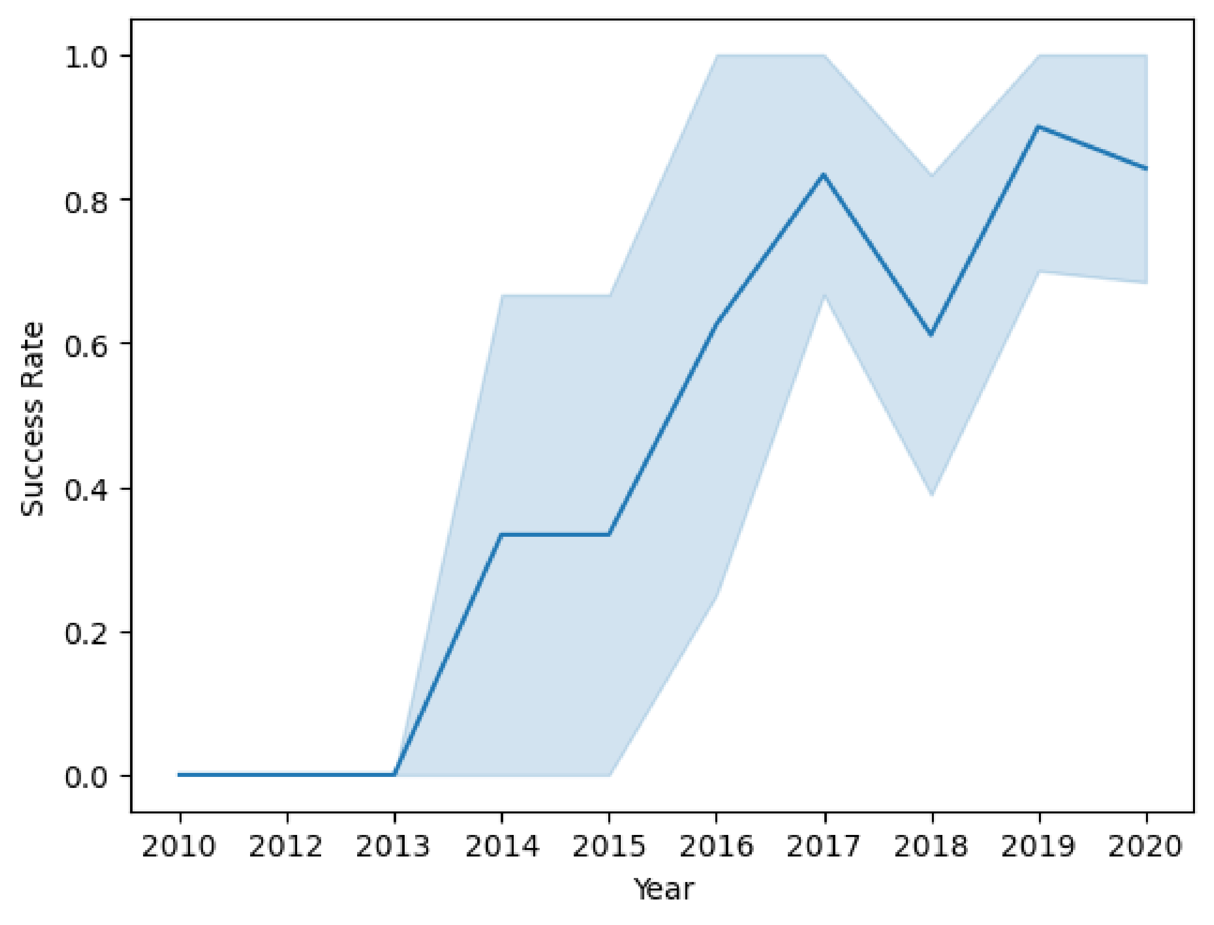
Payload vs. Orbit Type



- 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) are both there here.

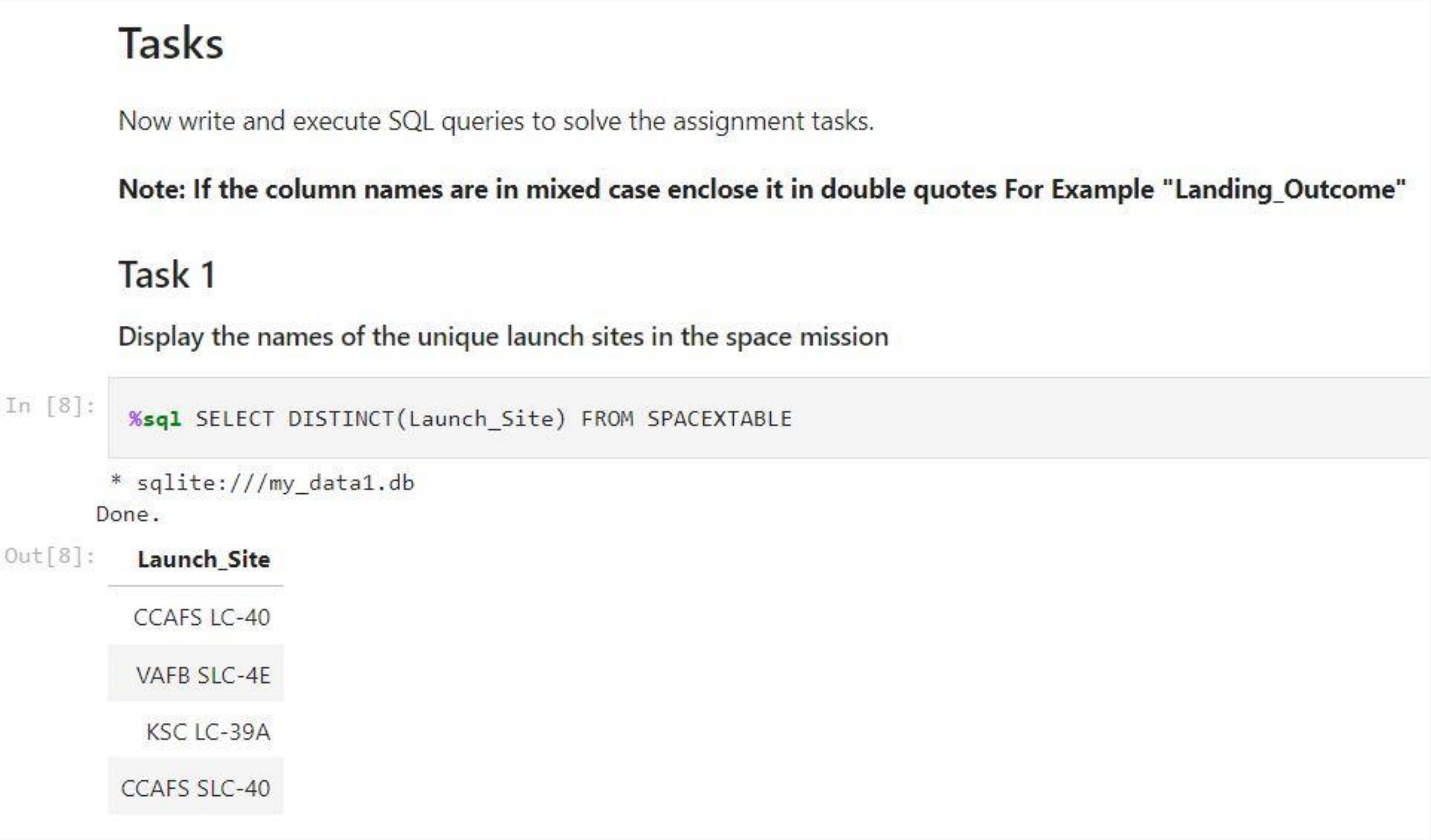
Launch Success Yearly Trend

- Success rate has been rising since 2013



All Launch Site Names

- The DISTINCT keyword was used to select only unique launch sites



Tasks

Now write and execute SQL queries to solve the assignment tasks.

Note: If the column names are in mixed case enclose it in double quotes For Example "Landing_Outcome"

Task 1

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

```
In [8]: %sql SELECT DISTINCT(Launch_Site) FROM SPACEXTABLE
```

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

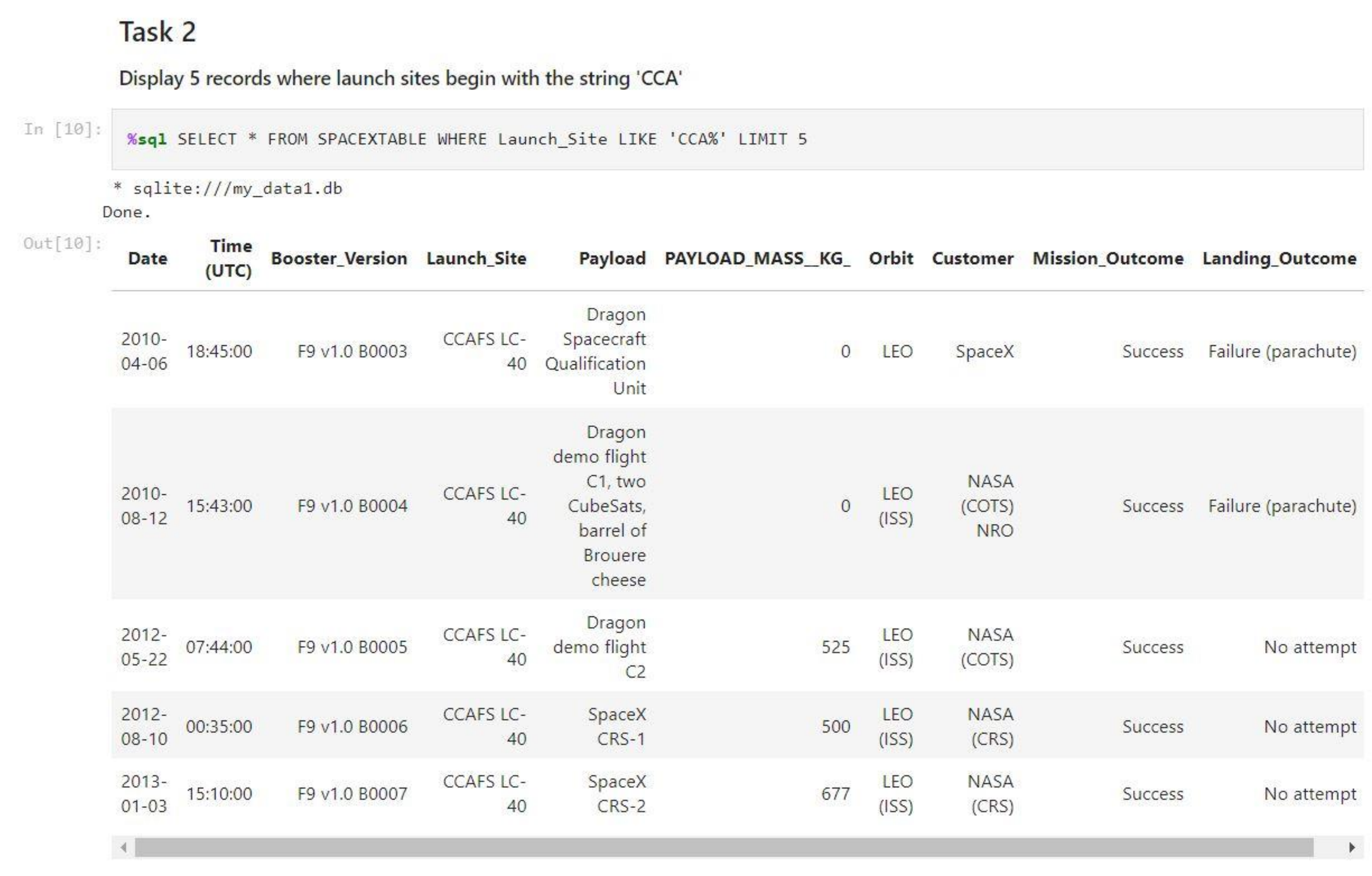
Out[8]:

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`:

- The LIMIT keyword is used to retrieve only 5 records



Task 2

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

```
In [10]: %sql SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5
```

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

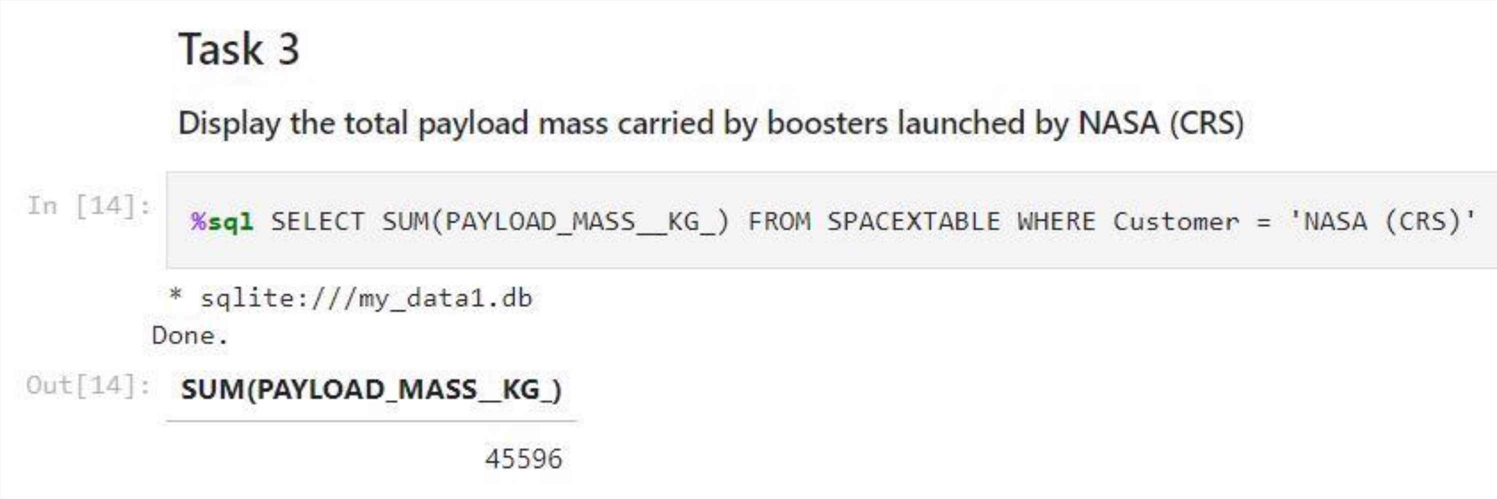
Out[10]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	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:

- Use the SUM function to aggregate the payload mass and filtered the records using the Customer column



Task 3

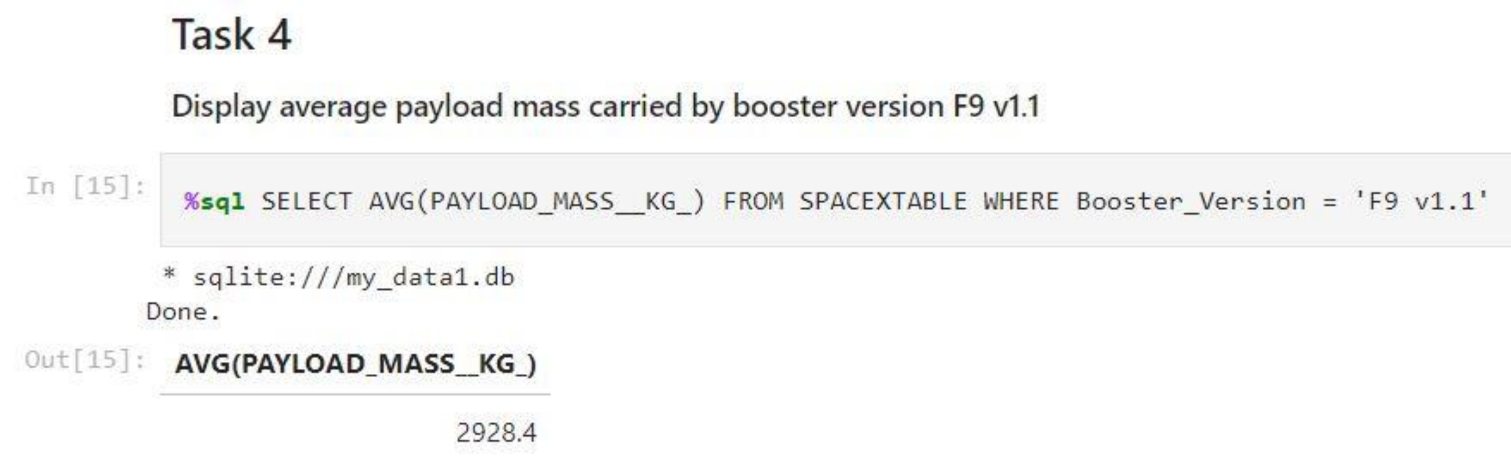
Display the total payload mass carried by boosters launched by NASA (CRS)

```
In [14]: %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)'

 * sqlite:///my_data1.db
Done.

Out[14]: SUM(PAYLOAD_MASS__KG_)
                          45596
```

Average Payload Mass by F9 v1.1

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

- The AVG function was used to get the average of the Payload mass and the records filtered to for Booster_version “F9 v1.1”



Task 4

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

```
In [15]: %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'
```

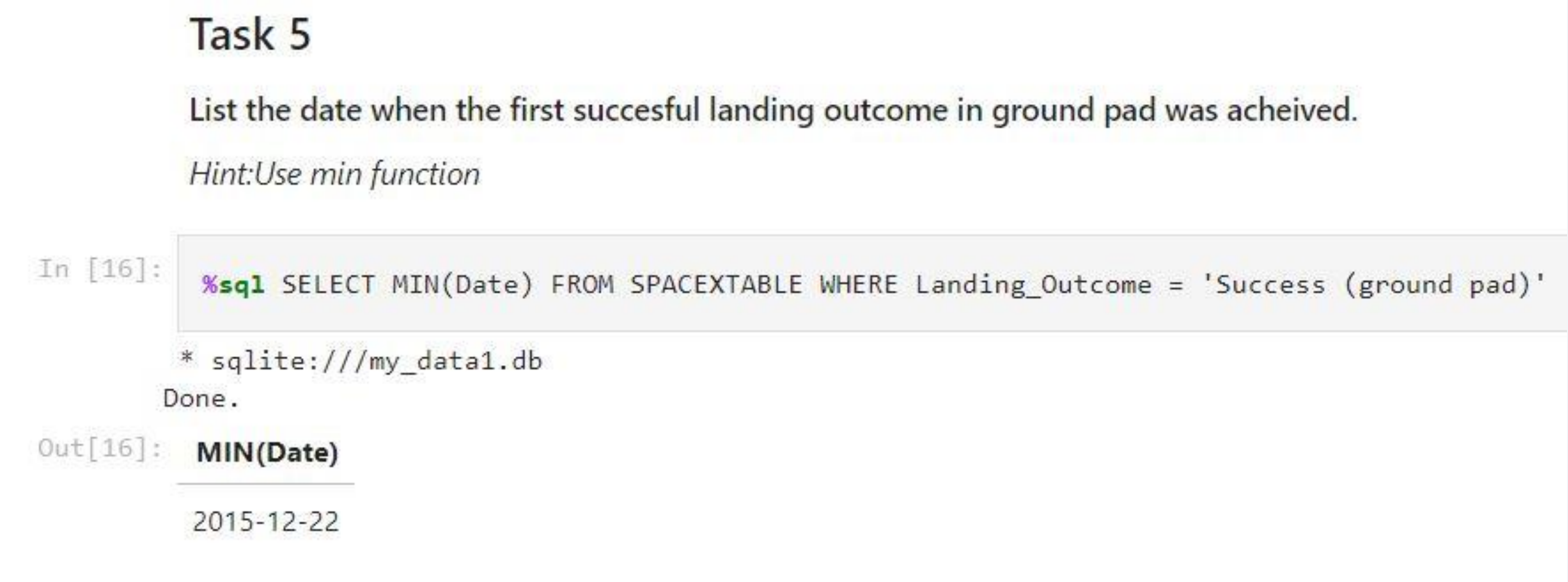
```
 * sqlite:///my_data1.db
Done.
```

Out[15]:

AVG(PAYLOAD_MASS__KG_)
2928.4

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.

Hint:Use min function

```
In [16]:  %sql SELECT MIN(Date) FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)'
```

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

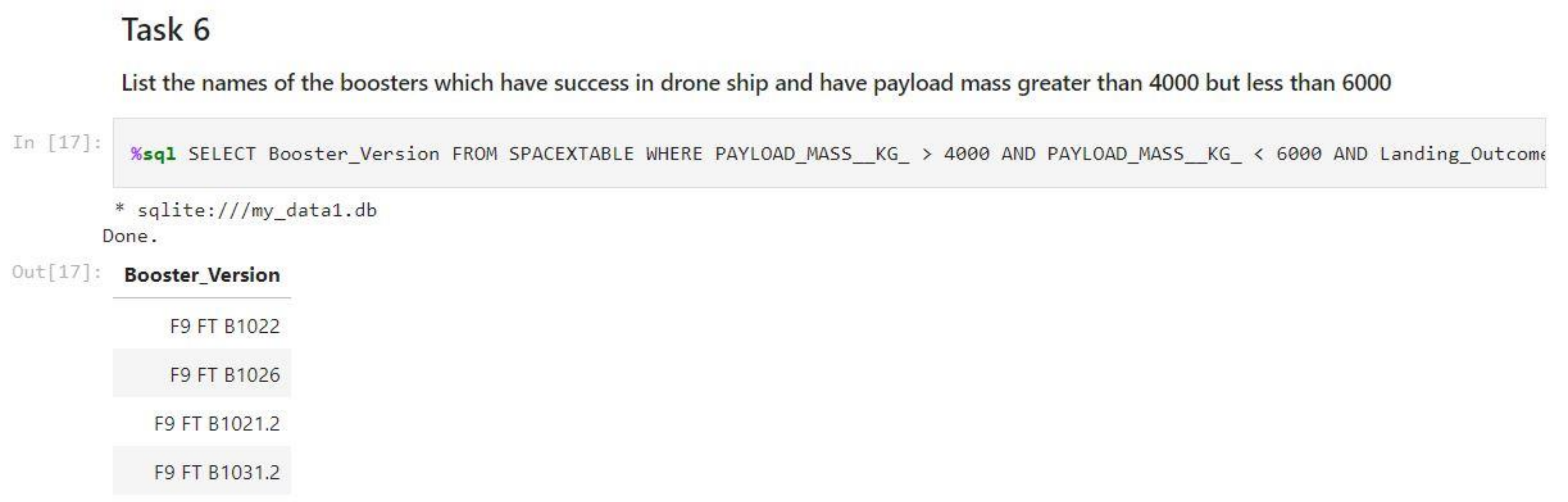
Out[16]:

MIN(Date)
2015-12-22

- The MIN function was used to return the date when the first successful landing outcome occurred together with filtering the Landing_outcome column

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:



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 [17]:
```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000 AND Landing_Outcome
```

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

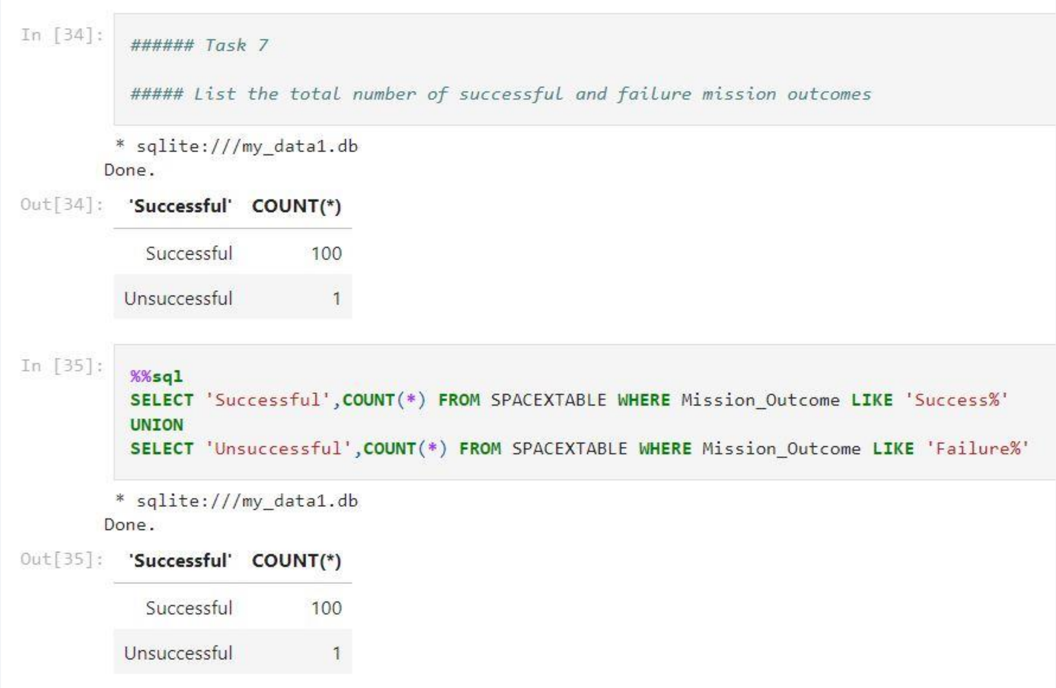
Out[17]:

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

- Booster_versions with Payload mass with the range of 4000 and 6000 were selected with a filtering of the Landing_outcome

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes



```
In [34]: ###### Task 7

         ##### List the total number of successful and failure mission outcomes
```

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

Out[34]:

'Successful'	COUNT(*)
Successful	100
Unsuccessful	1

```
In [35]: %%sql
         SELECT 'Successful',COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE 'Success%'
         UNION
         SELECT 'Unsuccessful',COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE 'Failure%'
```

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

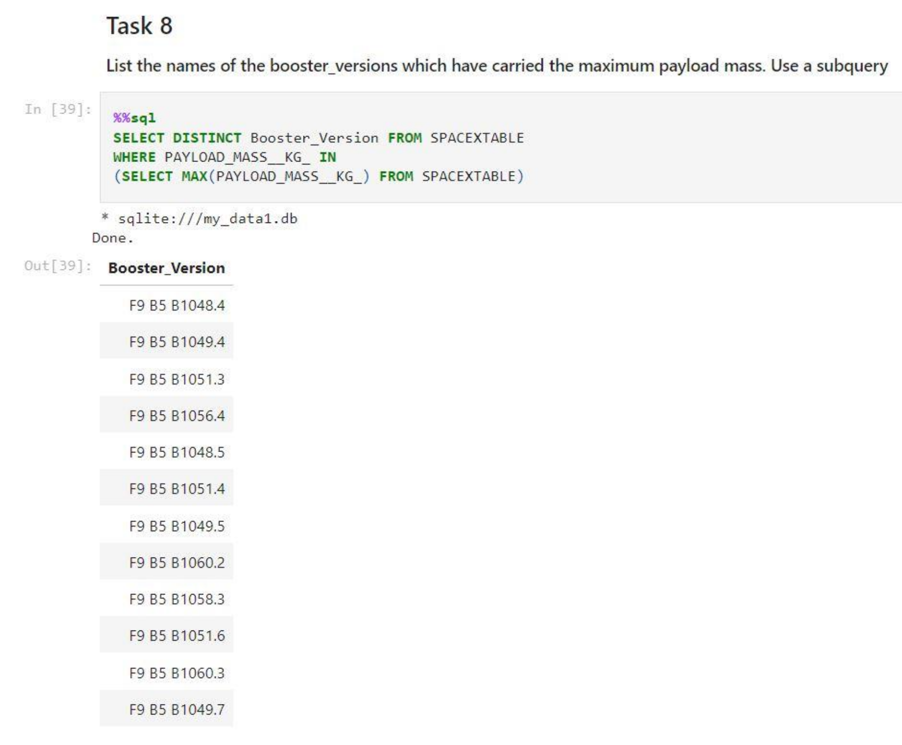
Out[35]:

'Successful'	COUNT(*)
Successful	100
Unsuccessful	1

- The COUNT function was used to filter successful mission outcomes and a union to unsuccessful outcomes

Boosters Carried Maximum Payload

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



Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [39]:

```
%%sql
SELECT DISTINCT Booster_Version FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ IN
(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
```

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

Out[39]:

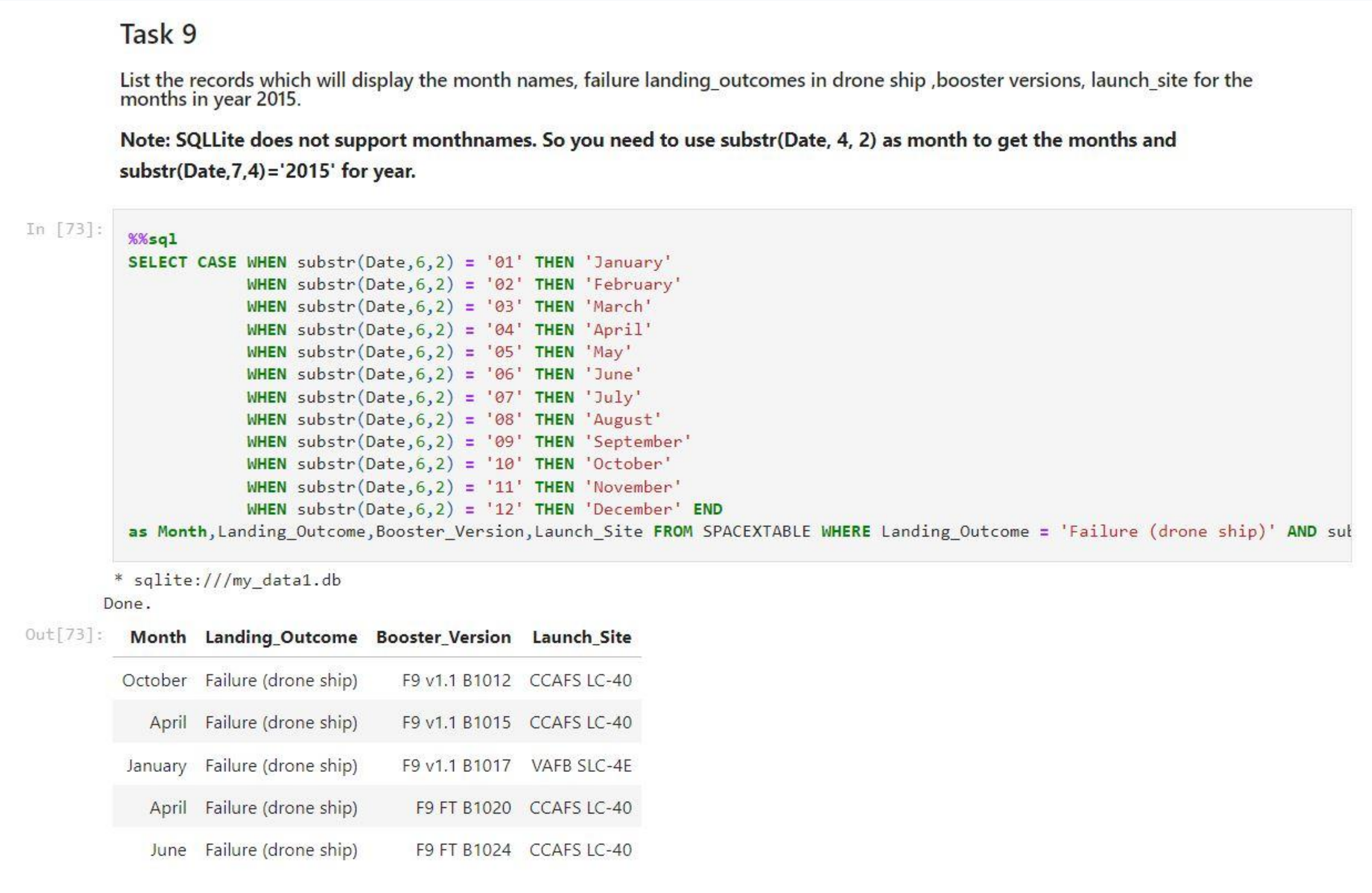
Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

- Distinct boosters where selected with the condition that their payload is equal to the maximum payload mass from the subquery

2015 Launch Records

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

- Used the CASE keyword to determine month name, filtered the records for failed outcome within the year 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, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [73]:

```
%%sql
SELECT CASE WHEN substr(Date,6,2) = '01' THEN 'January'
            WHEN substr(Date,6,2) = '02' THEN 'February'
            WHEN substr(Date,6,2) = '03' THEN 'March'
            WHEN substr(Date,6,2) = '04' THEN 'April'
            WHEN substr(Date,6,2) = '05' THEN 'May'
            WHEN substr(Date,6,2) = '06' THEN 'June'
            WHEN substr(Date,6,2) = '07' THEN 'July'
            WHEN substr(Date,6,2) = '08' THEN 'August'
            WHEN substr(Date,6,2) = '09' THEN 'September'
            WHEN substr(Date,6,2) = '10' THEN 'October'
            WHEN substr(Date,6,2) = '11' THEN 'November'
            WHEN substr(Date,6,2) = '12' THEN 'December' END
as Month,Landing_Outcome,Booster_Version,Launch_Site FROM SPACEXTABLE WHERE Landing_Outcome = 'Failure (drone ship)' AND sub
```

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

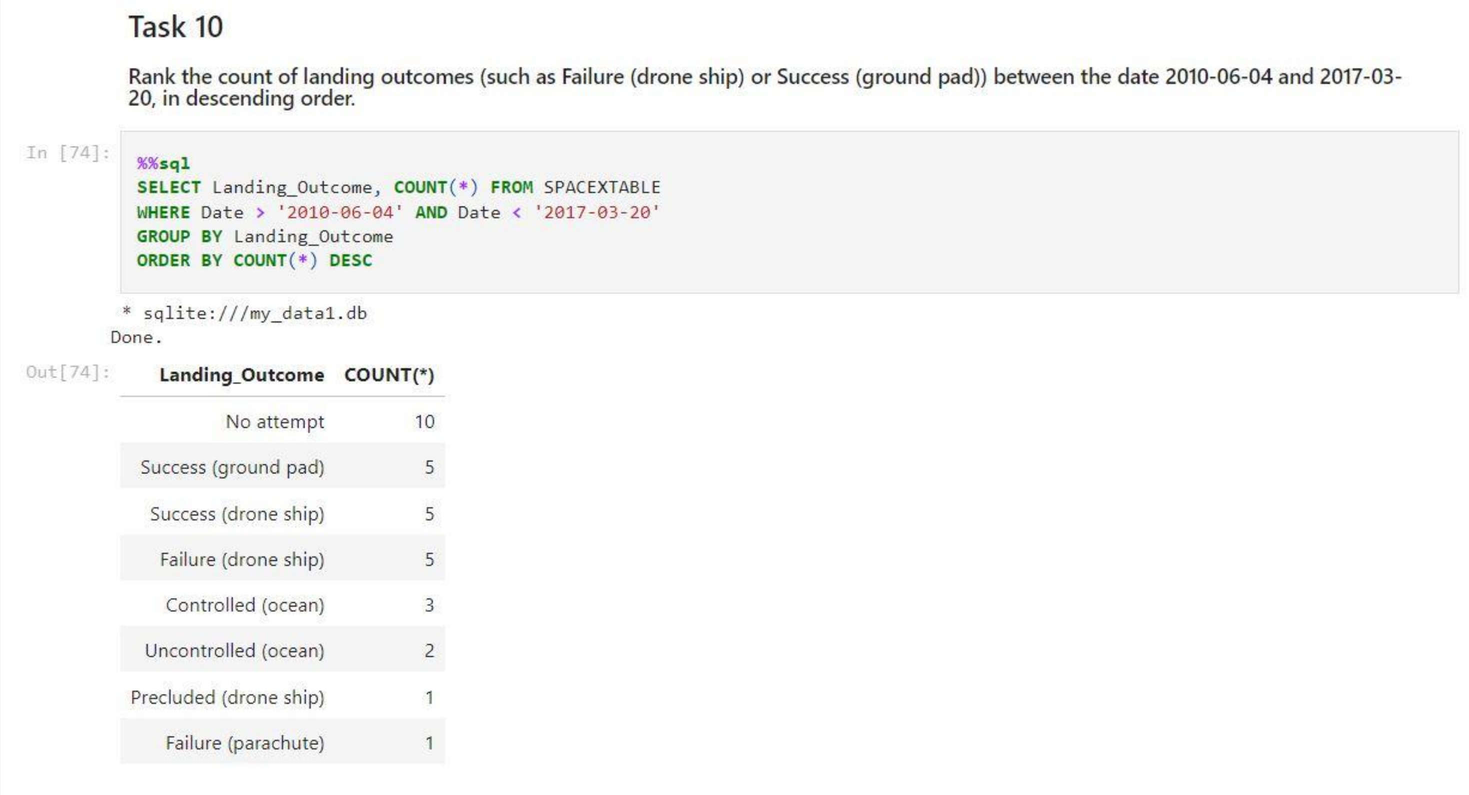
Out[73]:

Month	Landing_Outcome	Booster_Version	Launch_Site
October	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40
January	Failure (drone ship)	F9 v1.1 B1017	VAFB SLC-4E
April	Failure (drone ship)	F9 FT B1020	CCAFS LC-40
June	Failure (drone ship)	F9 FT B1024	CCAFS LC-40

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:

- Used the COUNT function to get the total number of occurrences and used the GROUP BY to group the results by the Landing_outcome



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 [74]:

```
%%sql
SELECT Landing_Outcome, COUNT(*) FROM SPACEXTABLE
WHERE Date > '2010-06-04' AND Date < '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY COUNT(*) DESC
```

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

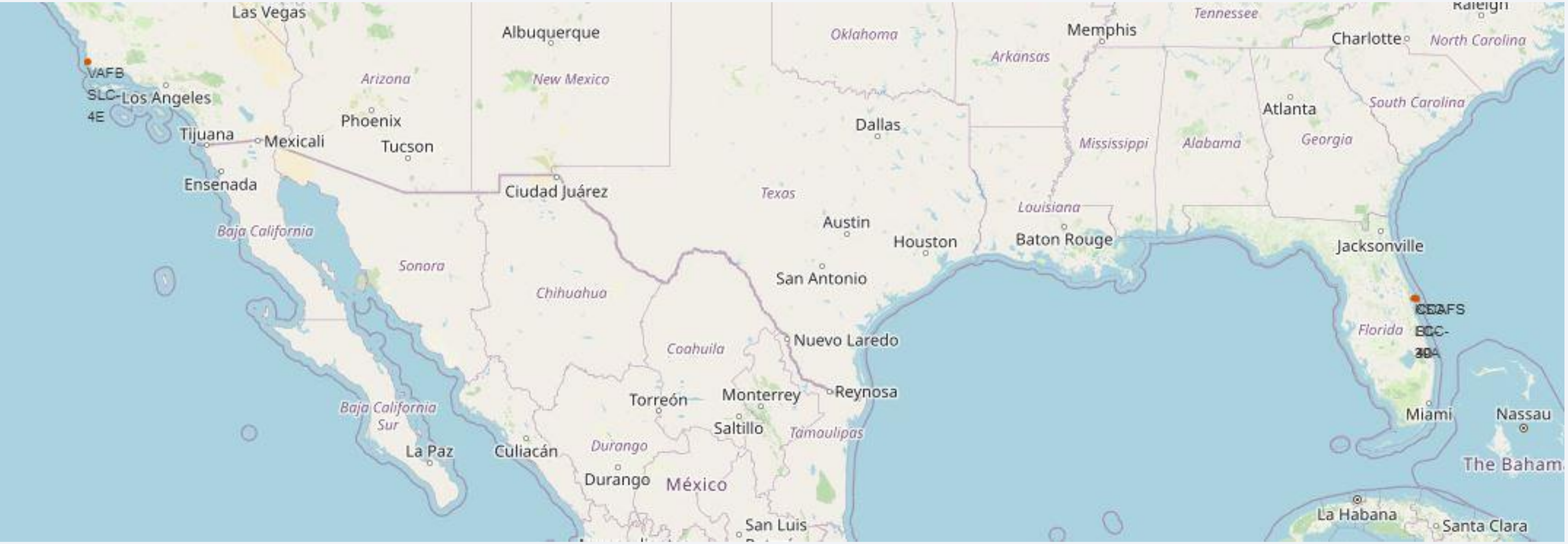
Out[74]:

Landing_Outcome	COUNT(*)
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Section 3

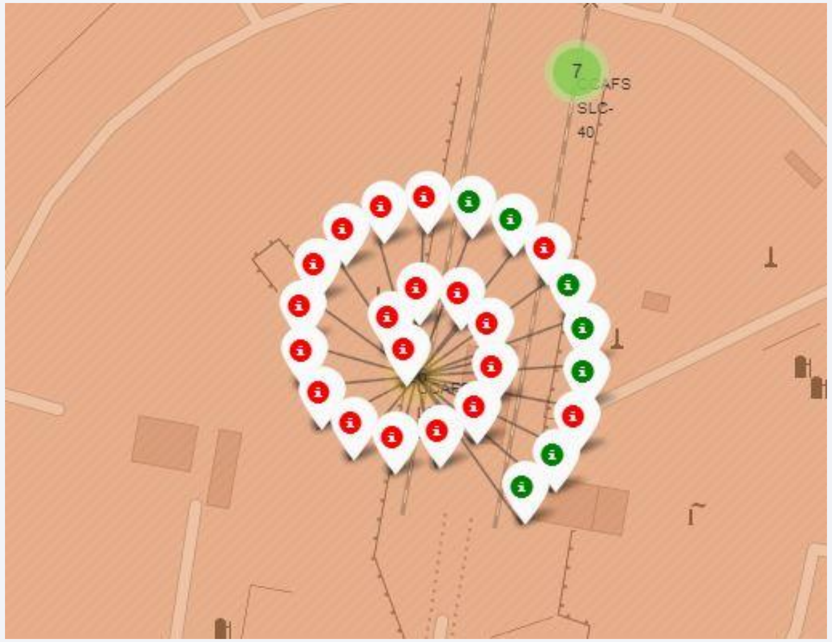
Launch Sites Proximities Analysis

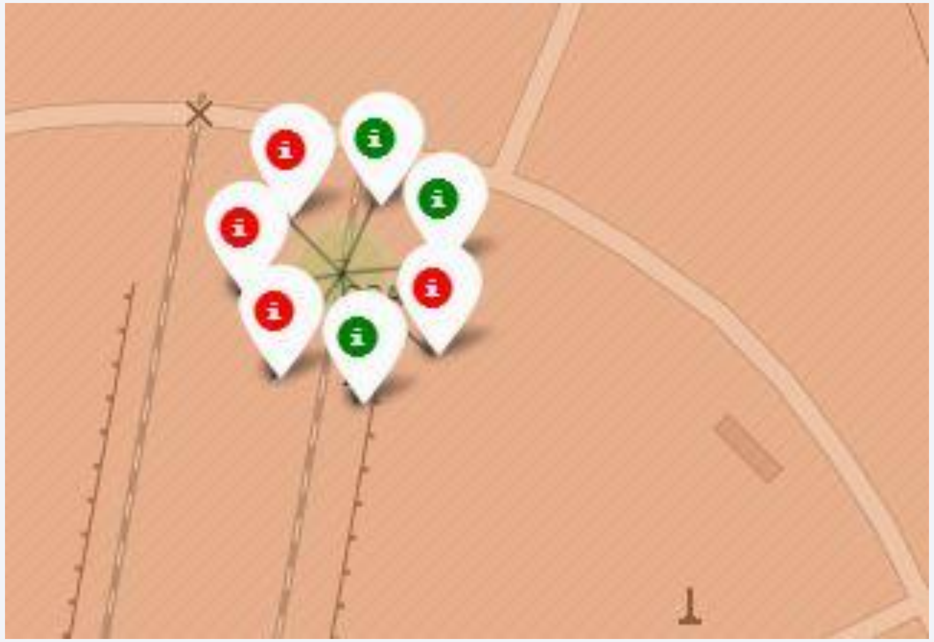
SpaceX Launch Sites



- The launch sites are close to the coastal lines.

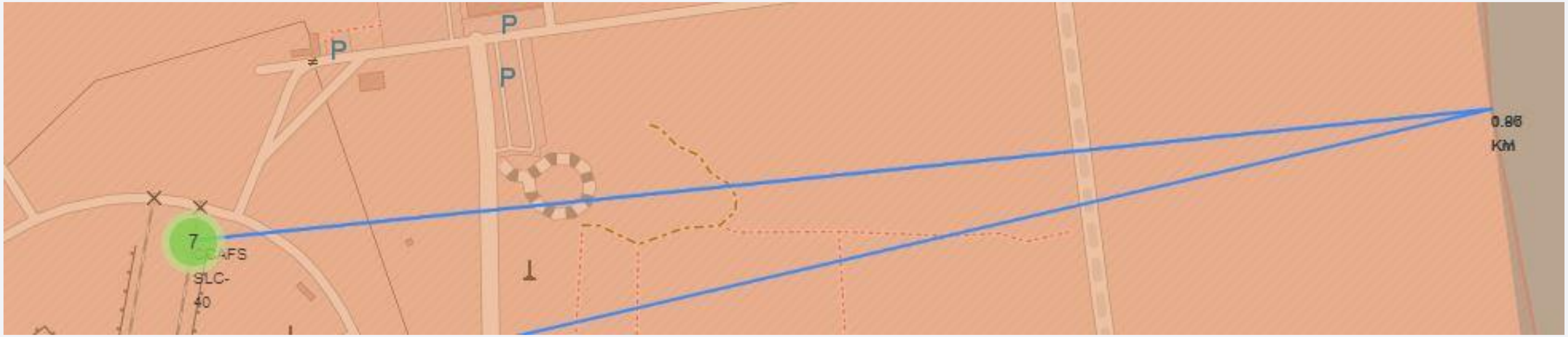
Launch outcomes for each site



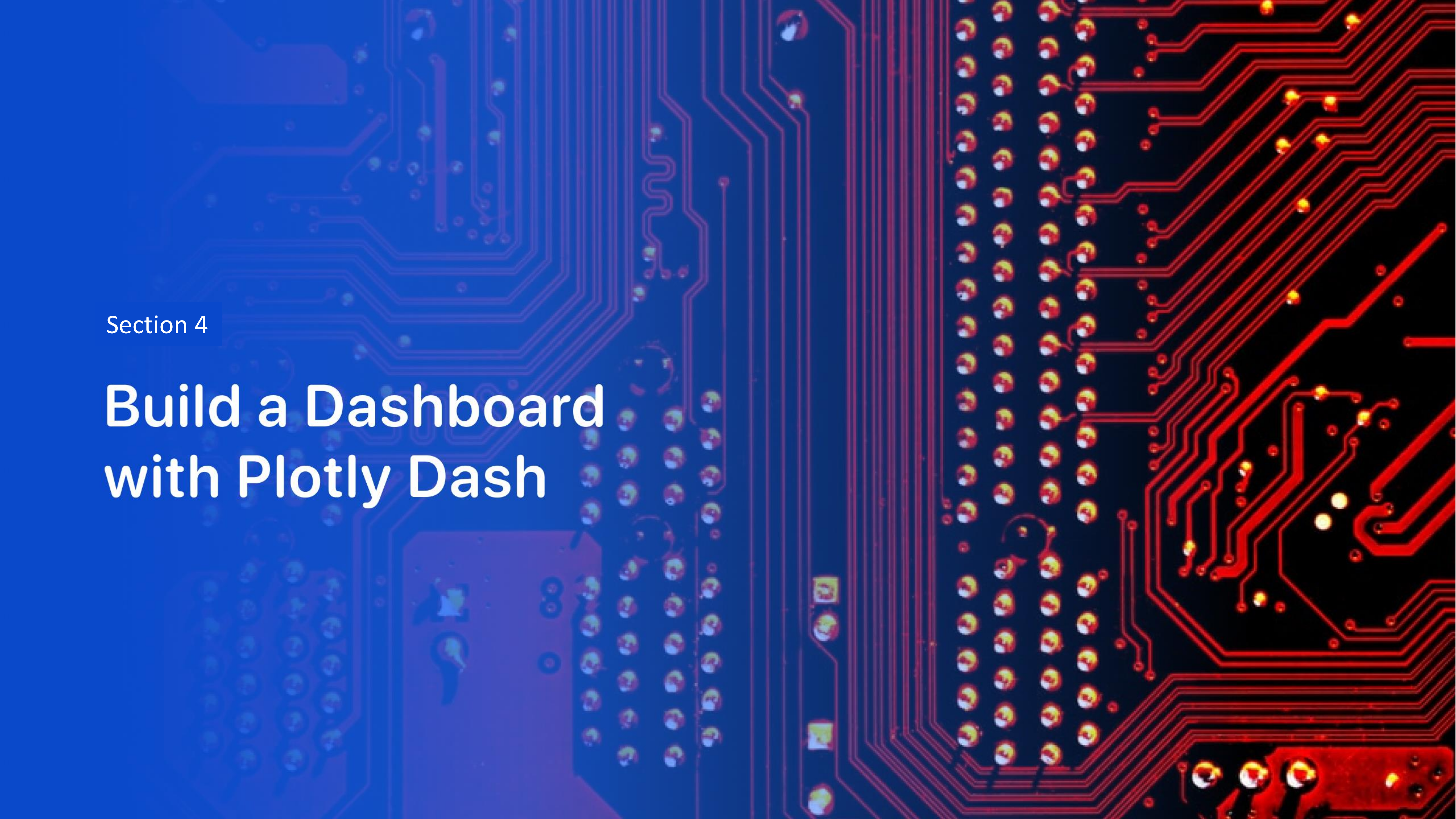


- Showing launch site in Florida.

Distance between launch site to its proximities



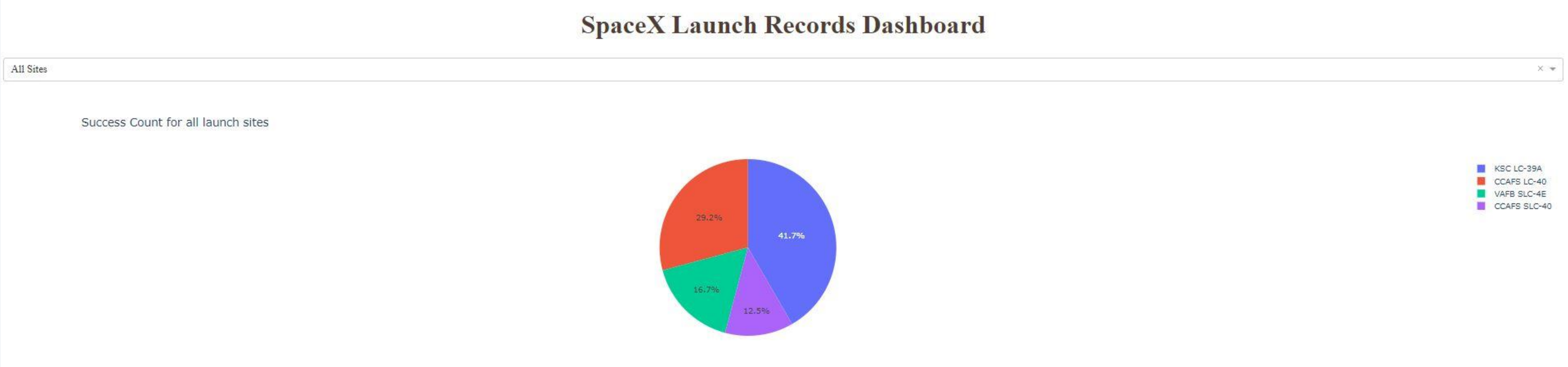
- Launch site SLC-40's proximity to coastline is 0.98km



Section 4

Build a Dashboard with Plotly Dash

Launch success for all sites



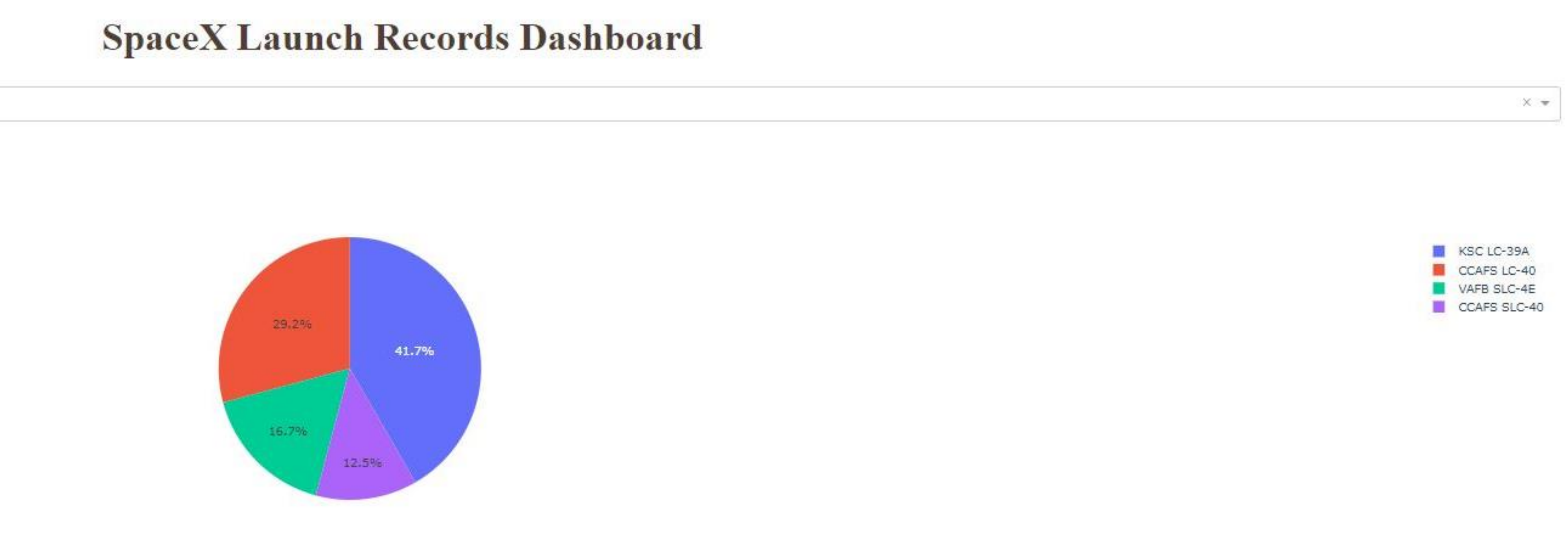
SpaceX Launch Records Dashboard

All Sites

Success Count for all launch sites

- Launch site KSC LC-39A has the highest success rate of 41.7% while CCAFS SLC-40 has the lowest

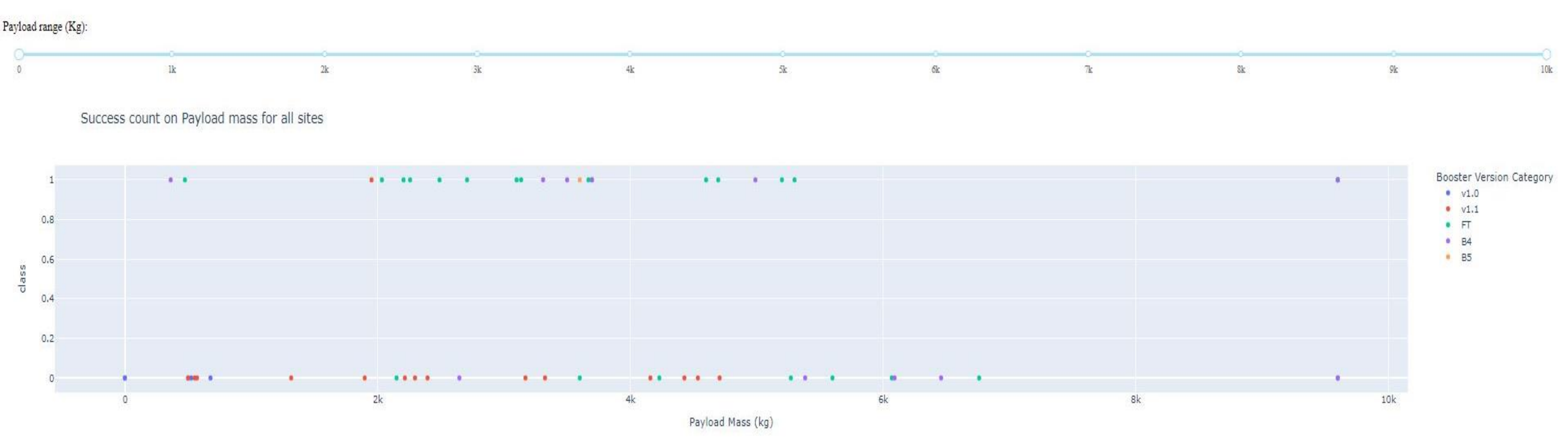
Launch site with highest success ratio



SpaceX Launch Records Dashboard

- KSC LC-39A with the highest success ratio of 41.7%

Payload vs. Launch Outcome for all sites

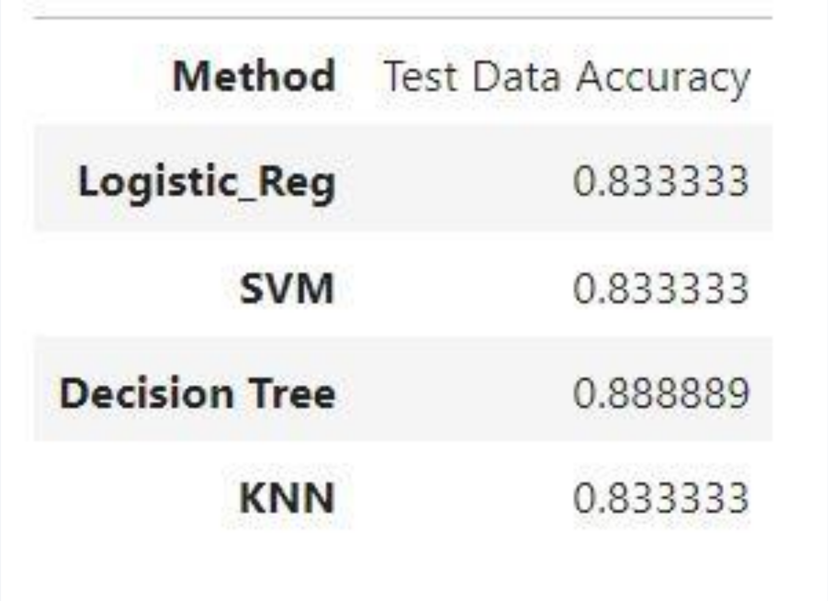


- Showing Payload mass in kg against the launch outcomes

Section 5

Predictive Analysis (Classification)

Classification Accuracy

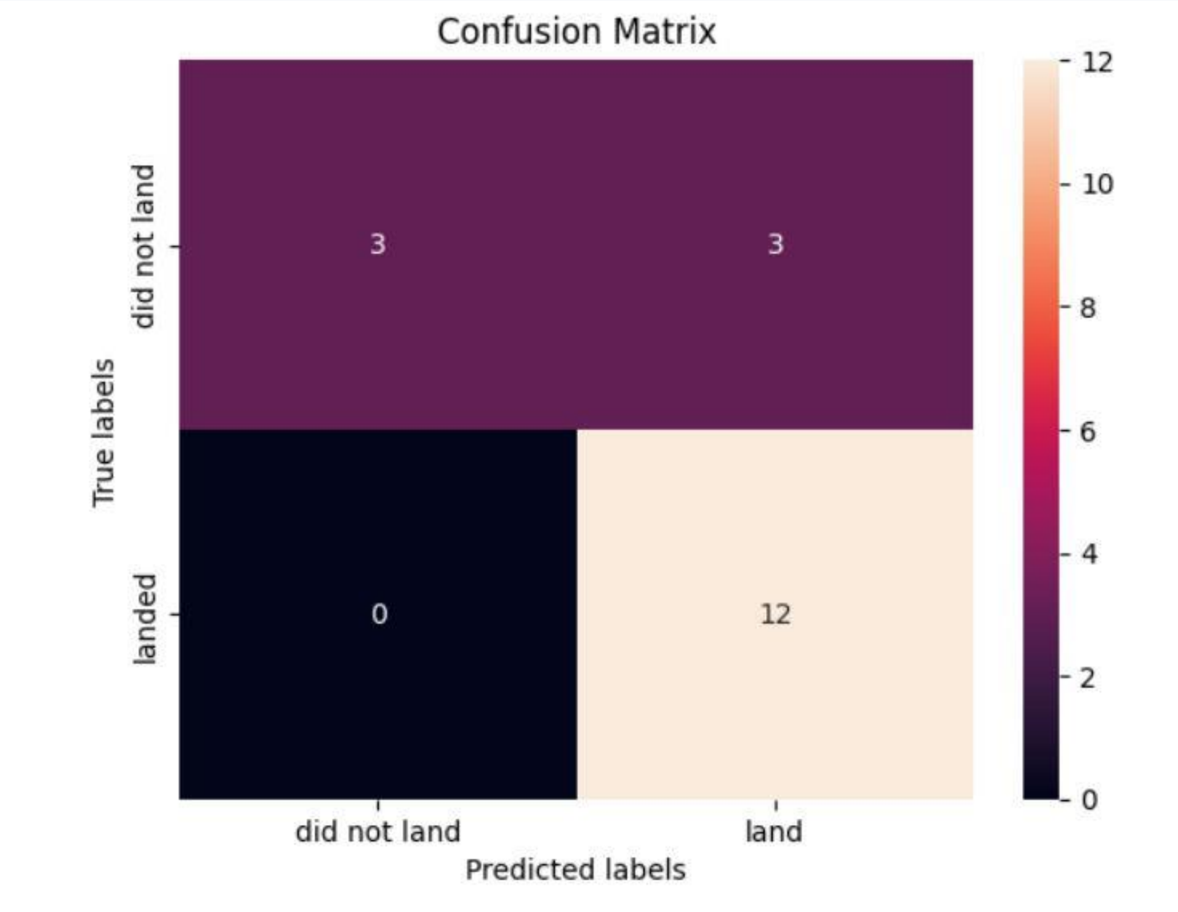


Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.888889
KNN	0.833333

Decision Tree has a slightly higher Test Data Accuracy

Confusion Matrix

We observe that logistic regression can distinguish between the various classes however, we notice the major problem is false positives



Conclusions

- Launch sites have varied success rates.
- An increase in flight number saw a corresponding increase in success rate.
- Four orbits, SSO, HEO, GEO and ES-L1 have the highest success rates of 100%
- There is no relationship between the number of flights and the success rate of the GTO orbit
- There is a relation between the success rate and the number of flights for the LEO orbit

Thank you!