IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Chima-Usim Evelyn Nnenna
24th July, 2024

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:
  - 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

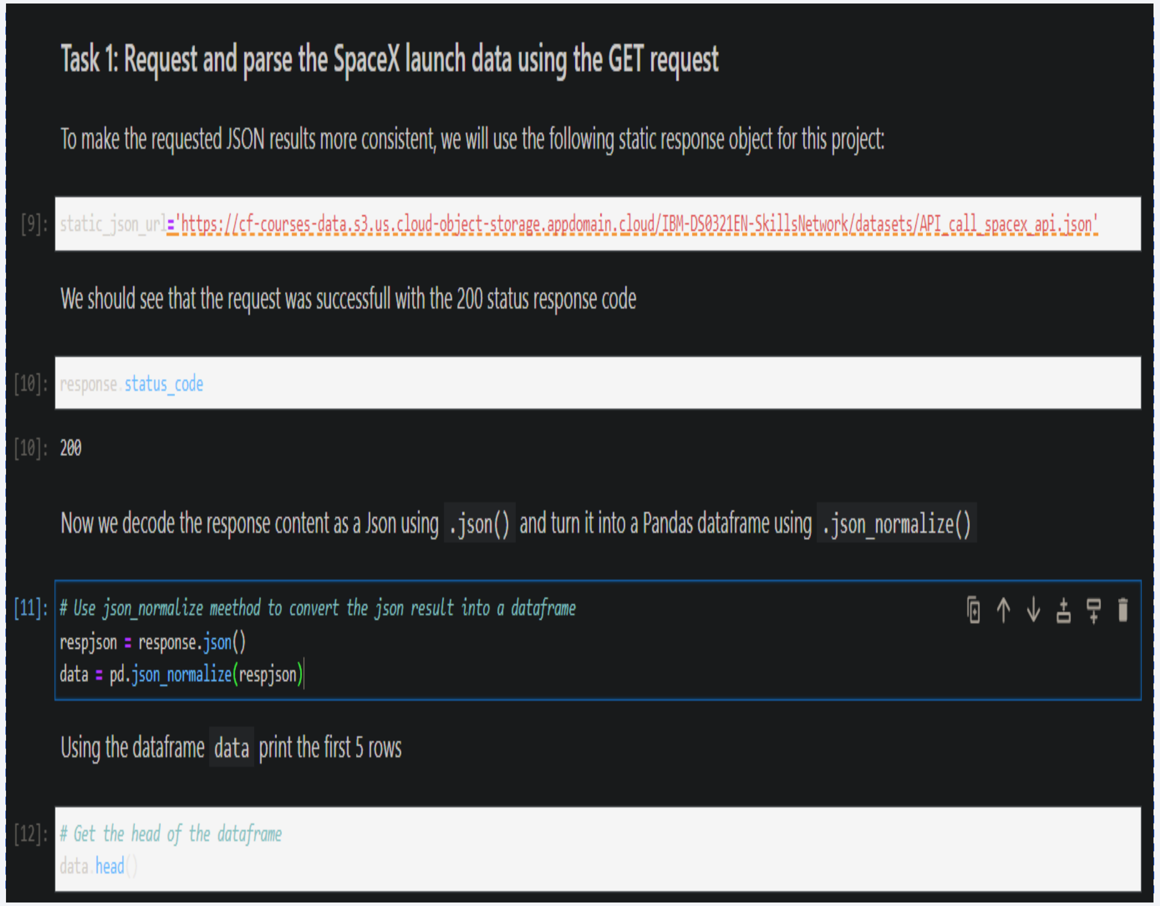
Data Collection

- Describe how data sets were 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 of 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 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 the launch records are stored in HTML. Using BeautifulSoup and requested Libraries, I extracted the Falcon 9 launch HTML table records from the Wikipedia page, Parsed the table, and converted it into a Panda data frame.
- You need to present your data collection process use key phrases and flowcharts

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts) by making a get request to the SpaceX API then requesting and parsing the SpaceX launch data using the GET request and decoding the response content as a JSON result which was then converted into a Pandas data frame.

- Add the GitHub URL of the completed SpaceX API calls notebook (https://github.com/chimausimevelyn/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/master/1.%20Spacex-data-collection-api.ipynb)



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:

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

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

```
[10]: response.status_code
[10]: 200
```

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

```
[11]: # Use json_normalize meethod to convert the json result into a dataframe
respjson = response.json()
data = pd.json_normalize(respjson)
```

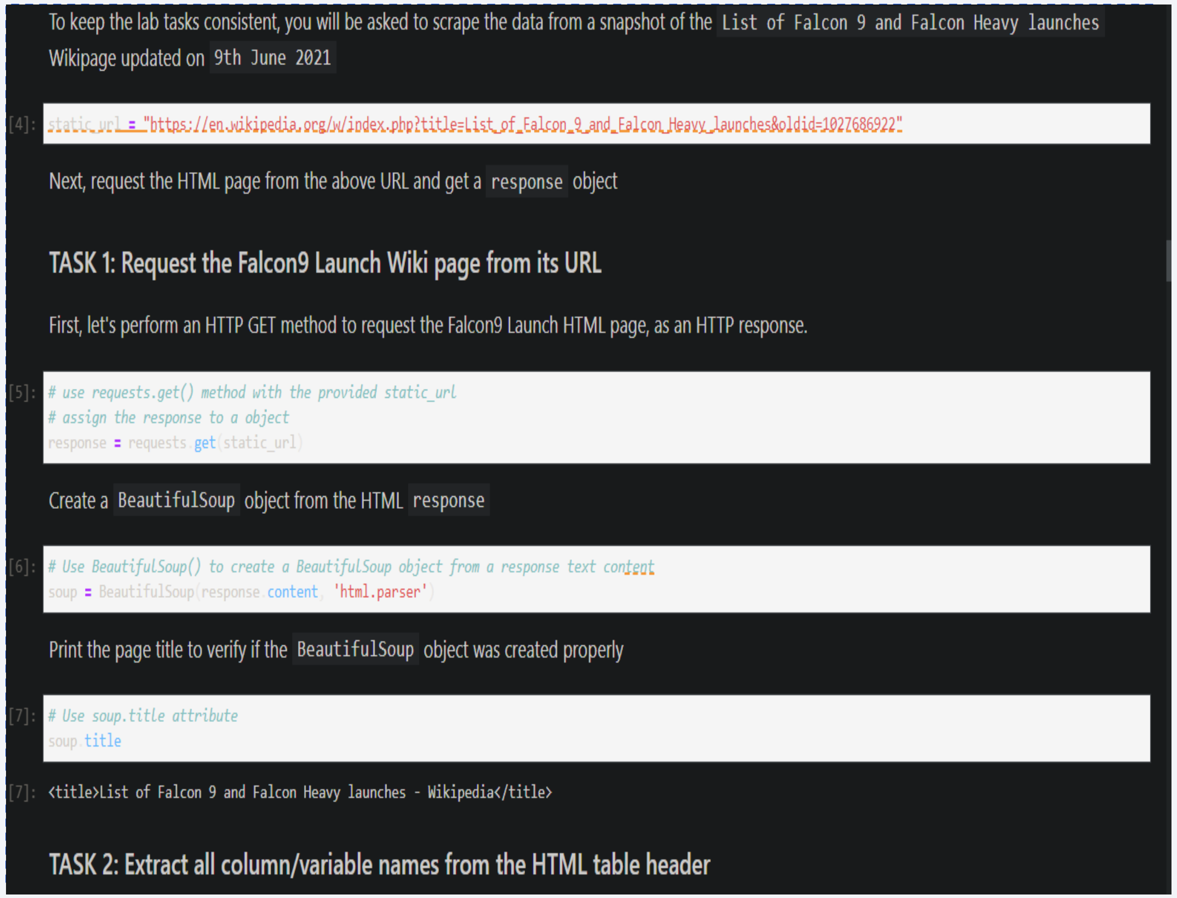
Using the dataframe `data` print the first 5 rows

```
[12]: # Get the head of the dataframe
data.head()
```

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts) Performed web scraping to collect Falcon 9 historical lunch records from Wikipedia using BeautifulSoup and requested, 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.

- Here is the GitHub URL of the completed web scraping notebook, (https://github.com/chimausimevelyn/SpaceX-Falcon-9-first-stage-Landing-



To keep the lab tasks consistent, you will be asked to scrape the data from a snapshot of the `List of Falcon 9 and Falcon Heavy launches` Wikipage updated on `9th June 2021`

```
[4]: static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

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.

```
[5]: # 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`

```
[6]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
     soup = BeautifulSoup(response.content, 'html.parser')
```

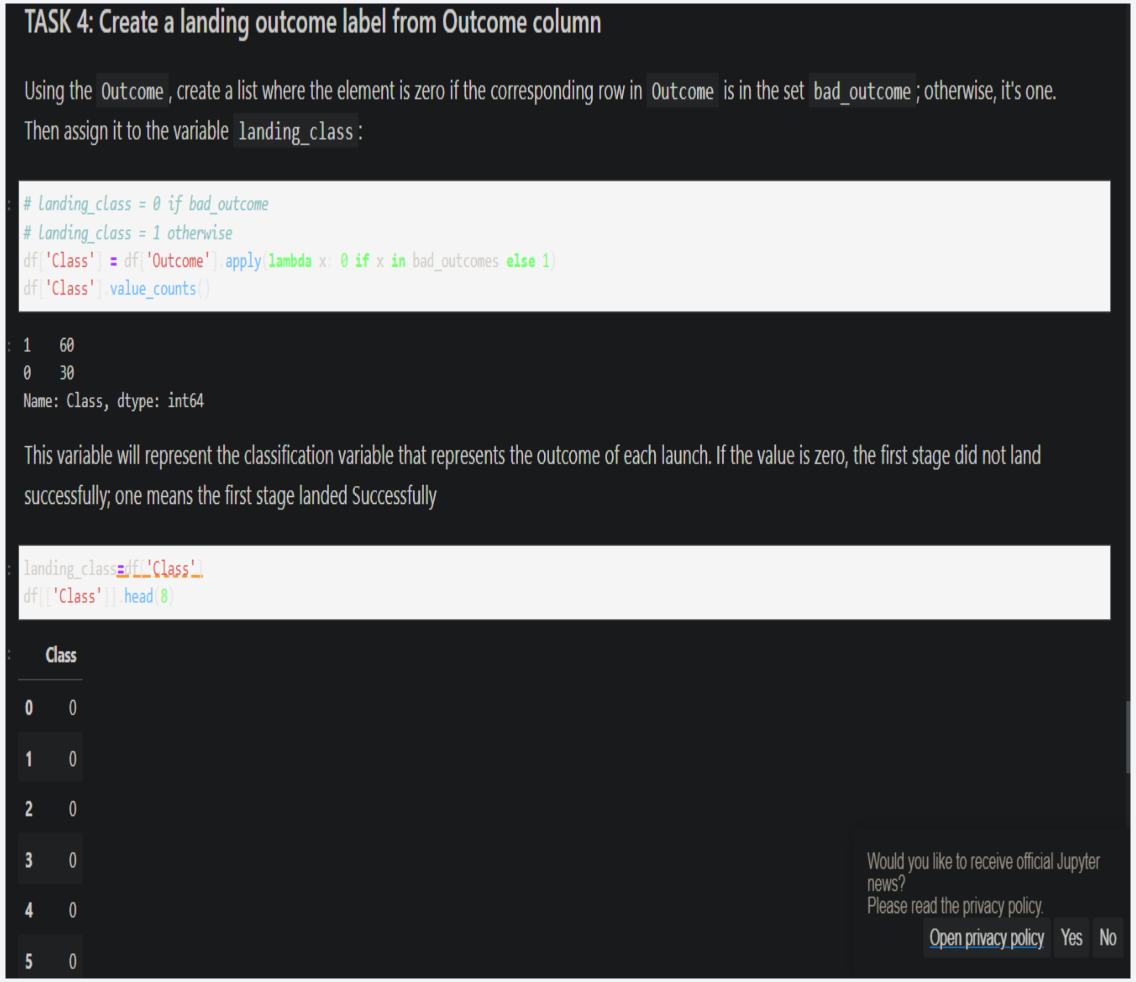
Print the page title to verify if the `BeautifulSoup` object was created properly

```
[7]: # Use soup.title attribute
     soup.title

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

TASK 2: Extract all column/variable 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 the mean value of the column.
- I also performed some exploratory data analysis (EDA) to find some patterns in the data and determine what the label would be for training supervised models.
- Here is the GitHub URL of the completed data wrangling related notebooks, (https://github.com/chimausimevelyn/SpaceX-Falcon-9-first-stage-Landing-



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`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
df['Class'] = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
df['Class'].value_counts()
```

```
1    60
0    30
Name: Class, dtype: int64
```

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

```
landing_class=df['Class']
df[['Class']].head(8)
```

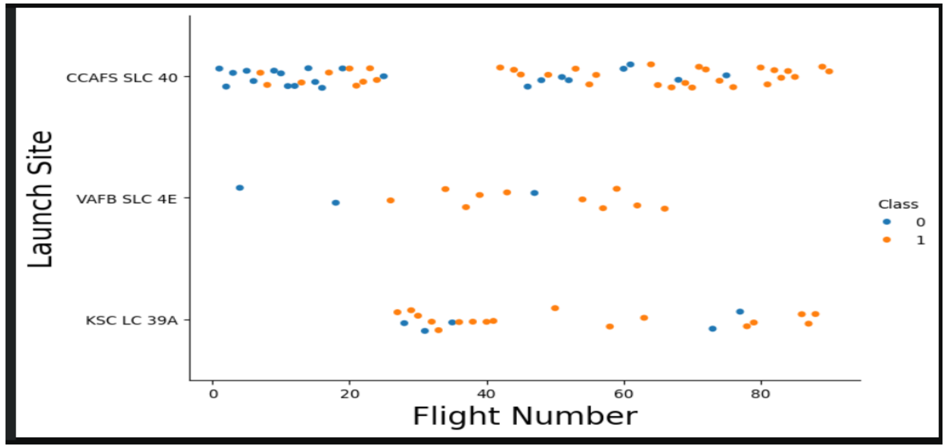
	Class
0	0
1	0
2	0
3	0
4	0
5	0

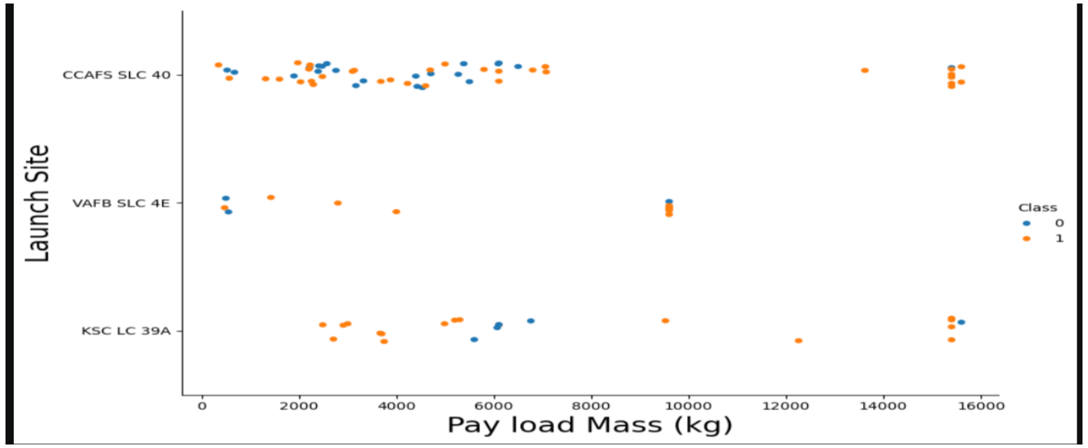
Would you like to receive official Jupyter news?
Please read the privacy policy.
Open privacy policy
Yes
No

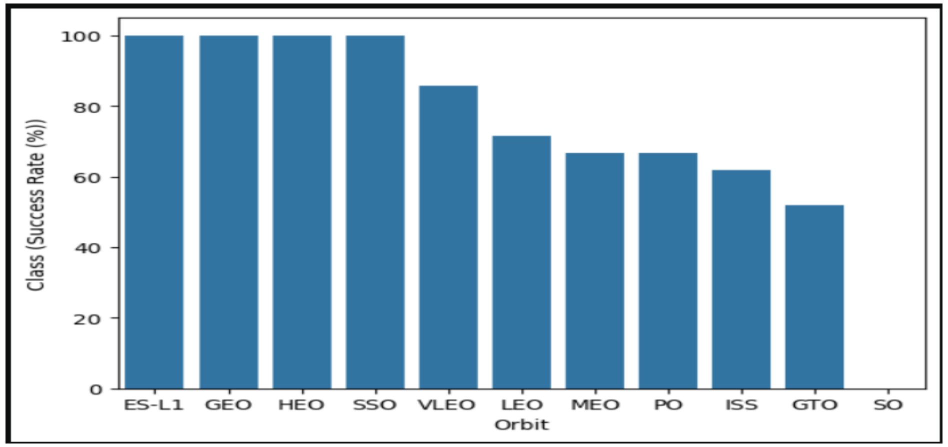
EDA with Data Visualization

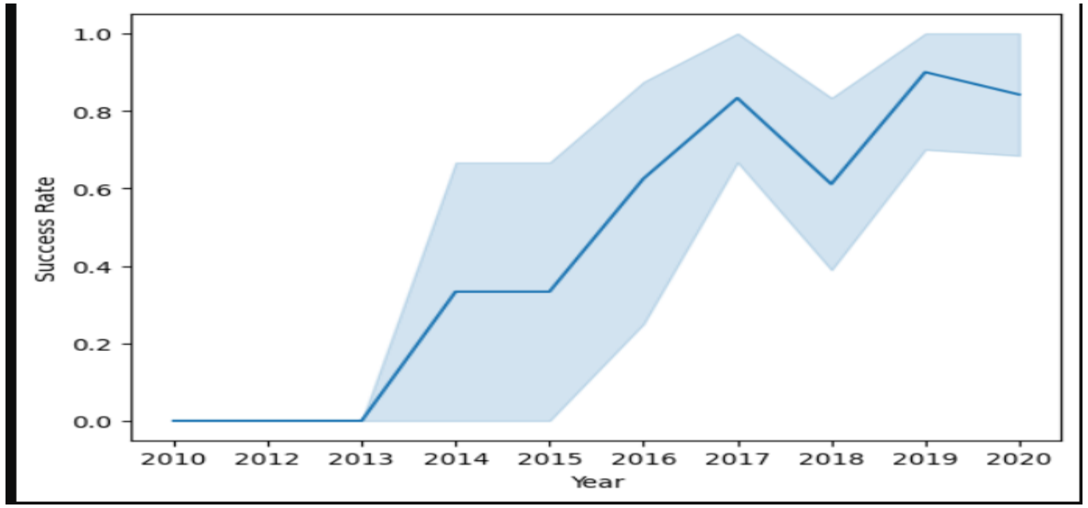
- Performed data Analysis and Feature Engineering using Pandas and Matplotlib .i.e.
  - Exploratory Data Analysis
  - Preparing Data Feature Engineering
- Used scatter plots to Visualize the relationship between Flight Number and Launch Site, Payload and Launch Site, FlightNumber and Orbit type, Payload and Orbit type.
- I 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 the completed EDA with data visualization notebook (https://github.com/chimausimevelyn/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/master/5.%20Edadataviz.ipynb)

EDA with Data Visualization (Plot Contd.)









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 (Contd.)

- 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 my completed EDA with SQL notebook (https://github.com/chimausimevelyn/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/master/4.%20Eda-sql-coursera_sqllite.ipynb)

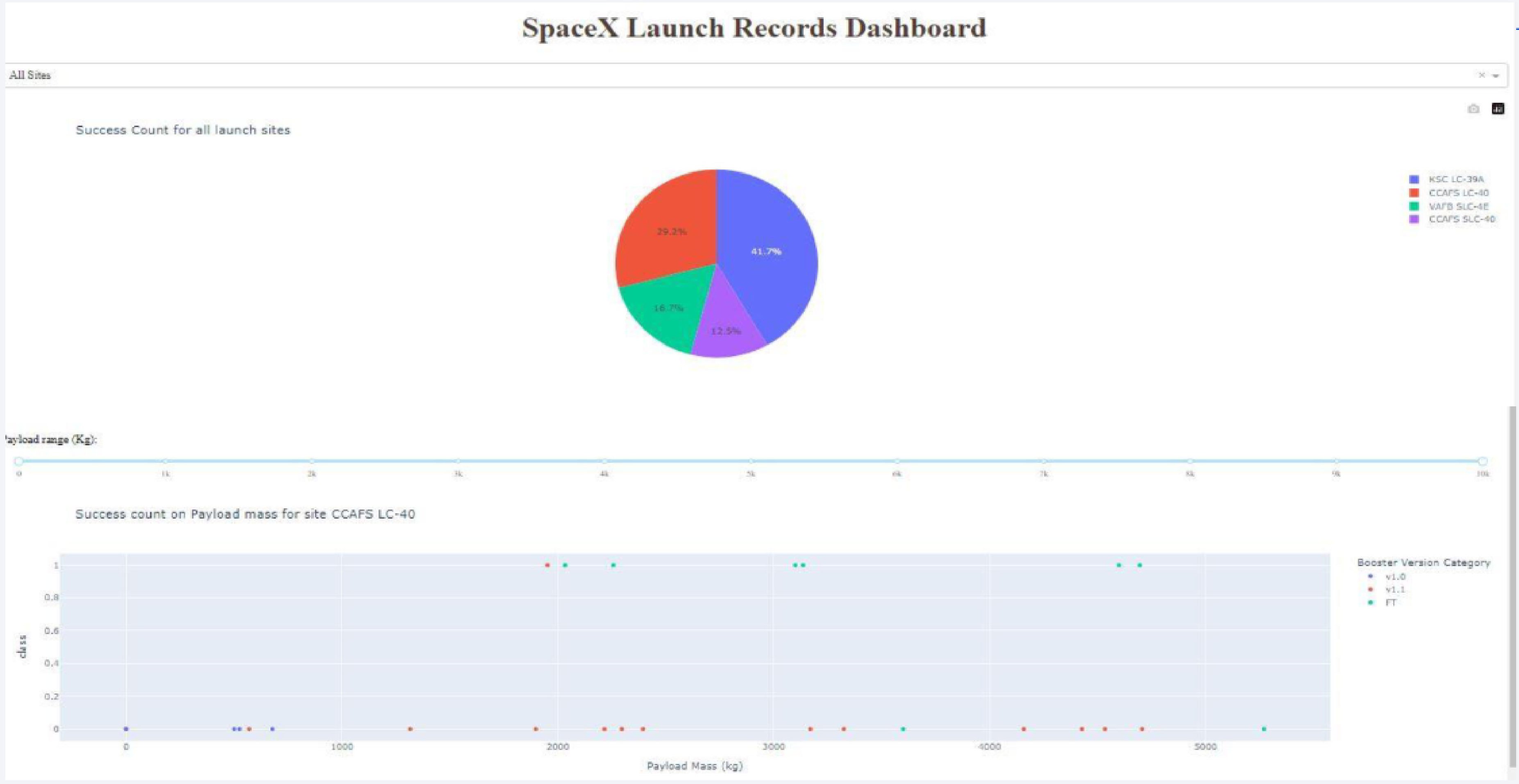
Build an Interactive Map with Folium

- I created a folium map to mark all the launch sites and created map objects such as markers, circles, and lines to mark the success or failure of launches for each launch site.
- Also created a launch set outcomes (failure=0 or success=1).
- Here is the GitHub URL of my completed interactive map with Folium map, as an external reference and peer-review purpose (https://github.com/chimausimevelyn/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/master/6.%20launch_site_location.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 the 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, as an external reference and peer-review purpose (https://github.com/chimausimevelyn/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/master/7.%20spacex_dash_app.py)

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)

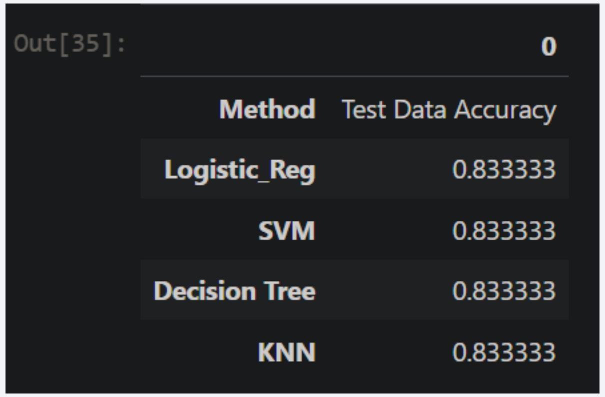
- Summary of how I built, evaluated, improved, and found the best performing classification model
- After loading the data as a Pandas dataframe, I set out to perform exploratory Data Analysis and determine Training Labels by:
  - Creating a Numpy array from the column class in data, by applying the method to Numpy() then assigned it to the variable Y as the outcome variable.
  - Then the feature dataset (X) was standardized by transforming it using the 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 perform best using the test data between SVM, Classification Trees, K nearest neighbors, and Logistic Regression;
  - First create an object for each of the algorithms, then create a GridSearchCV object and assign them a set of paraments for each model.
  - For each model under evaluation, the GridSearchCV object was created with cv=10, and then fit the training data into the GridSearch object for each to find the best Hyperparameter.
  - After fitting the training set, we output the GridSearchCV object for each of the models, then displayed the best parameters using the data attributes 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 Tree, K nearest neighbors, and logistic Regression;



```
Out[35]:
```

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

- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose (https://github.com/chimausimevelyn/SpaceX-Falcon-9-first-stage-Landing-Prediction/blob/master/8.%20SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb)

Results

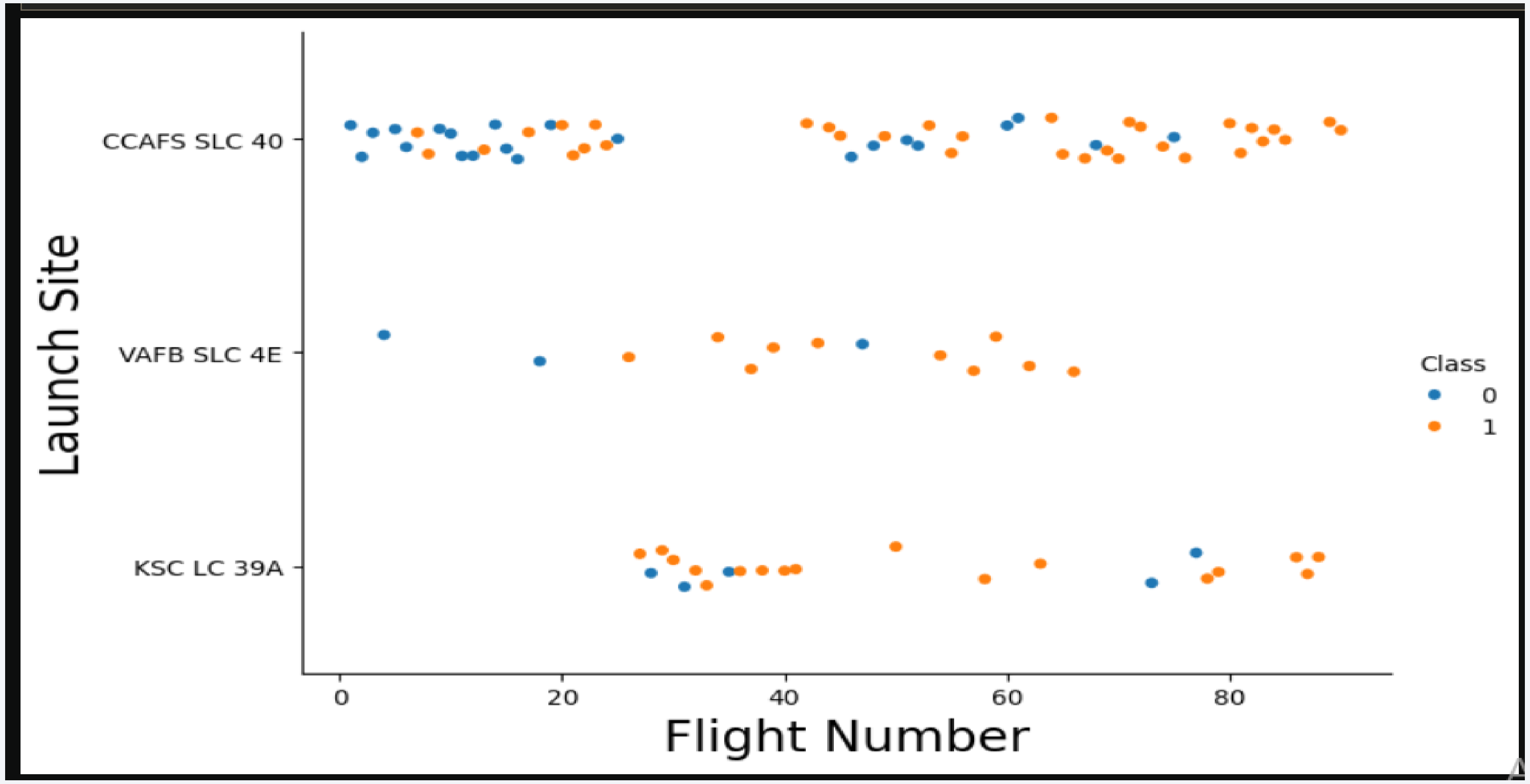
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

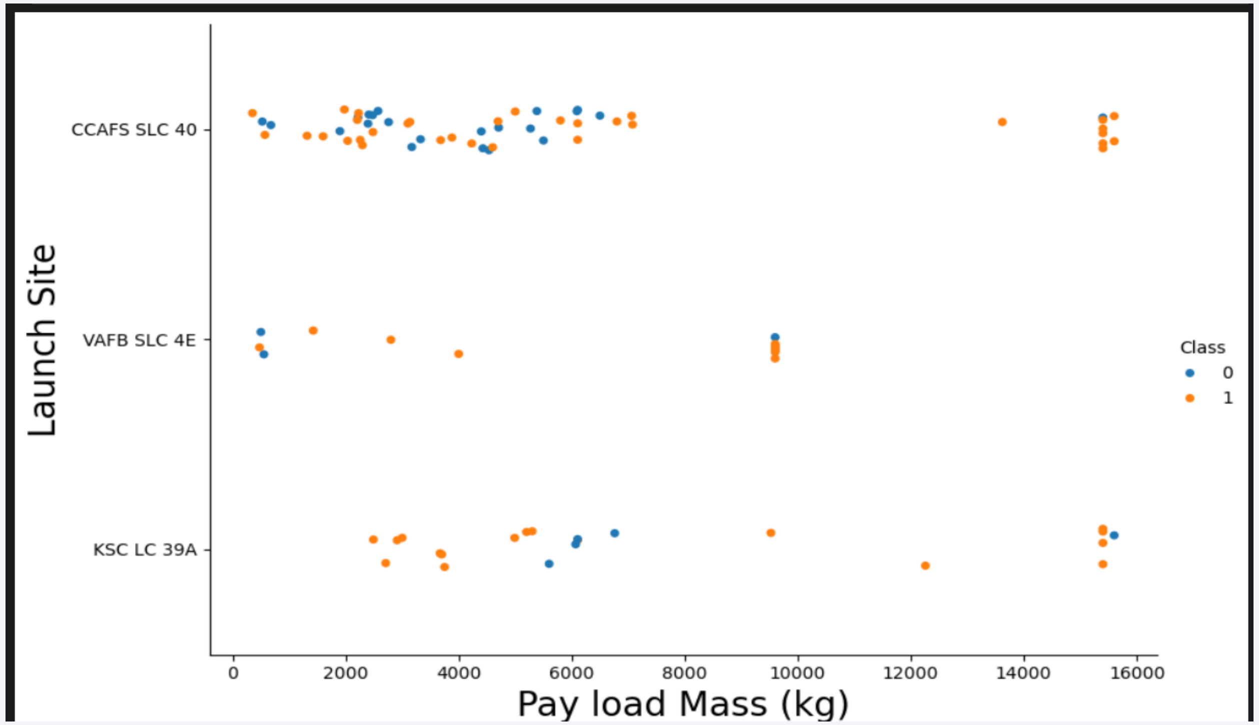
Flight Number vs. Launch Site

- A scatter plot of Flight Number vs. Launch Site



Payload vs. Launch Site

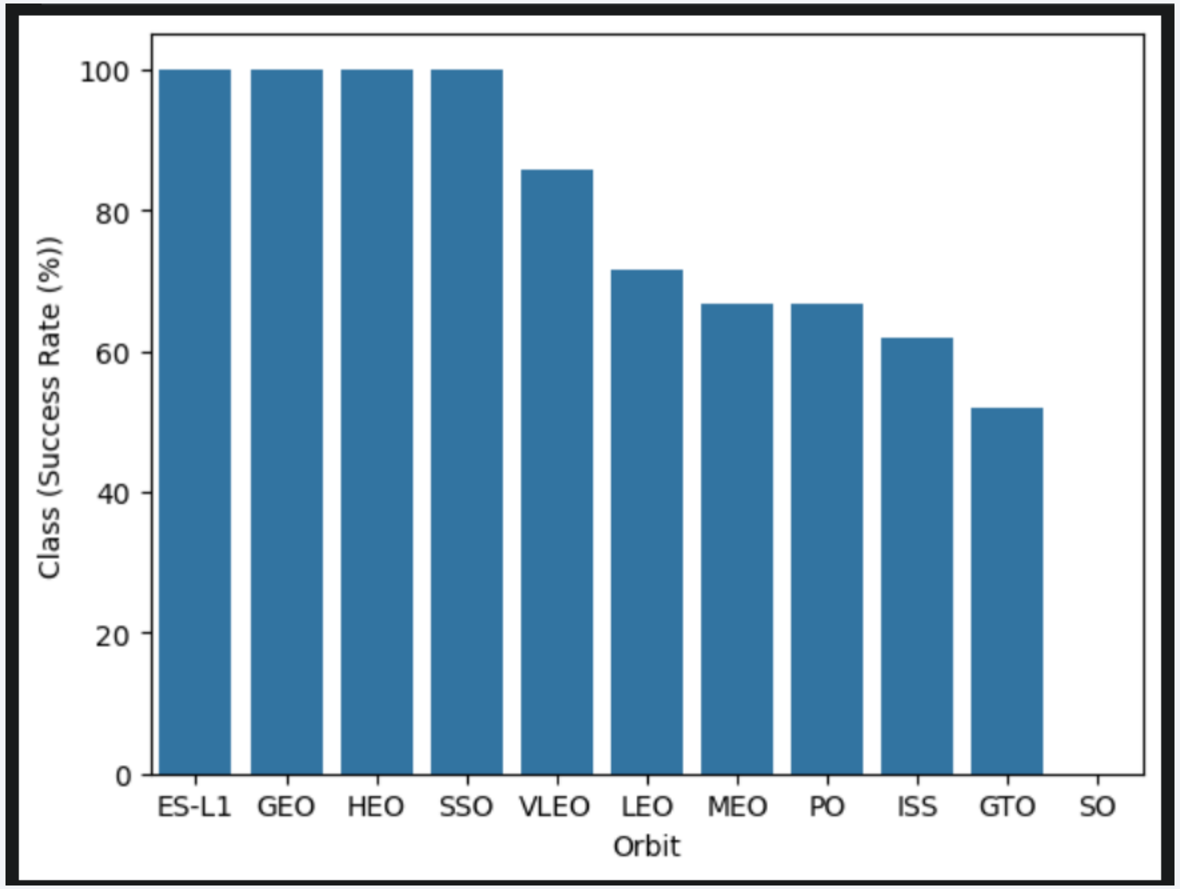
- Show a scatter plot of Payload vs. Launch Site



- 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).

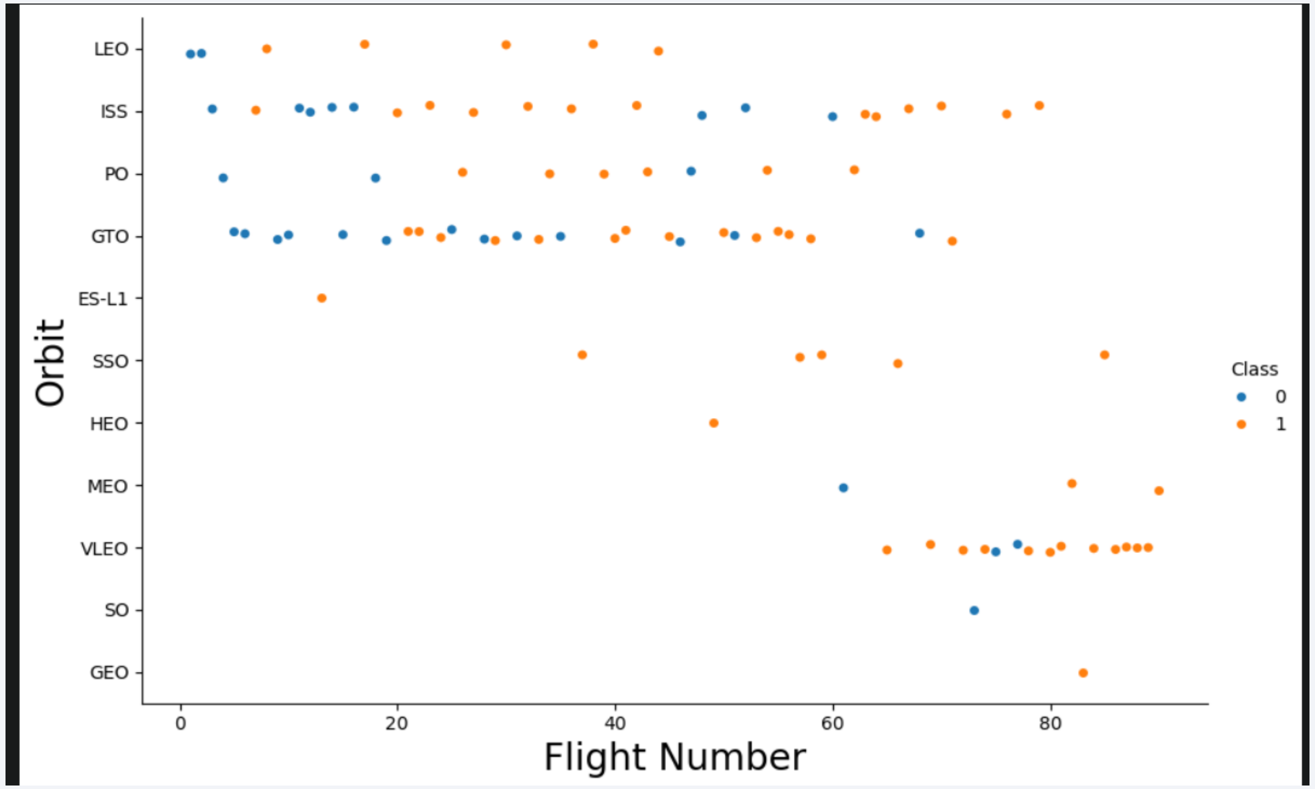
Success Rate vs. Orbit Type

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



Success Rate vs. Orbit Type

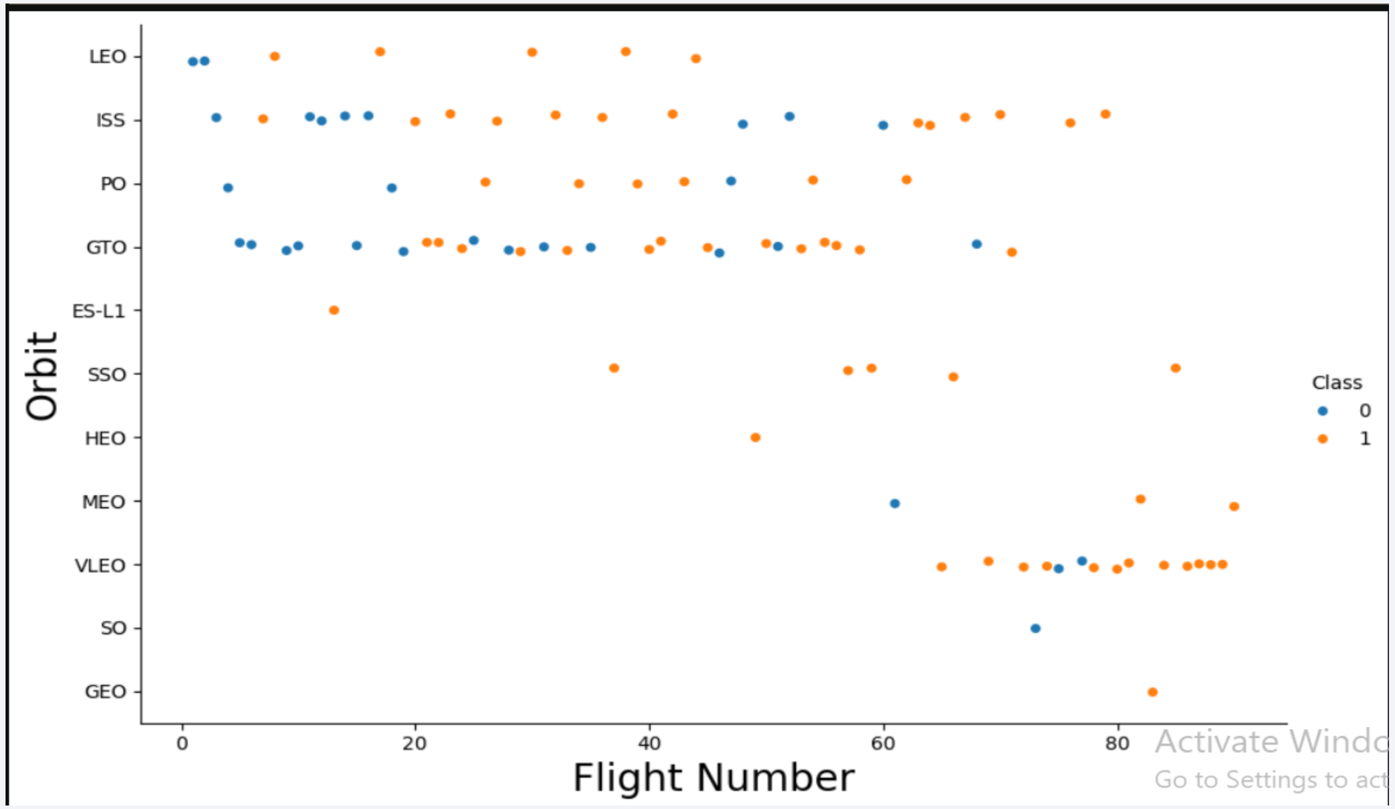
- Show the screenshot of the scatter plot with explanations



- 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.

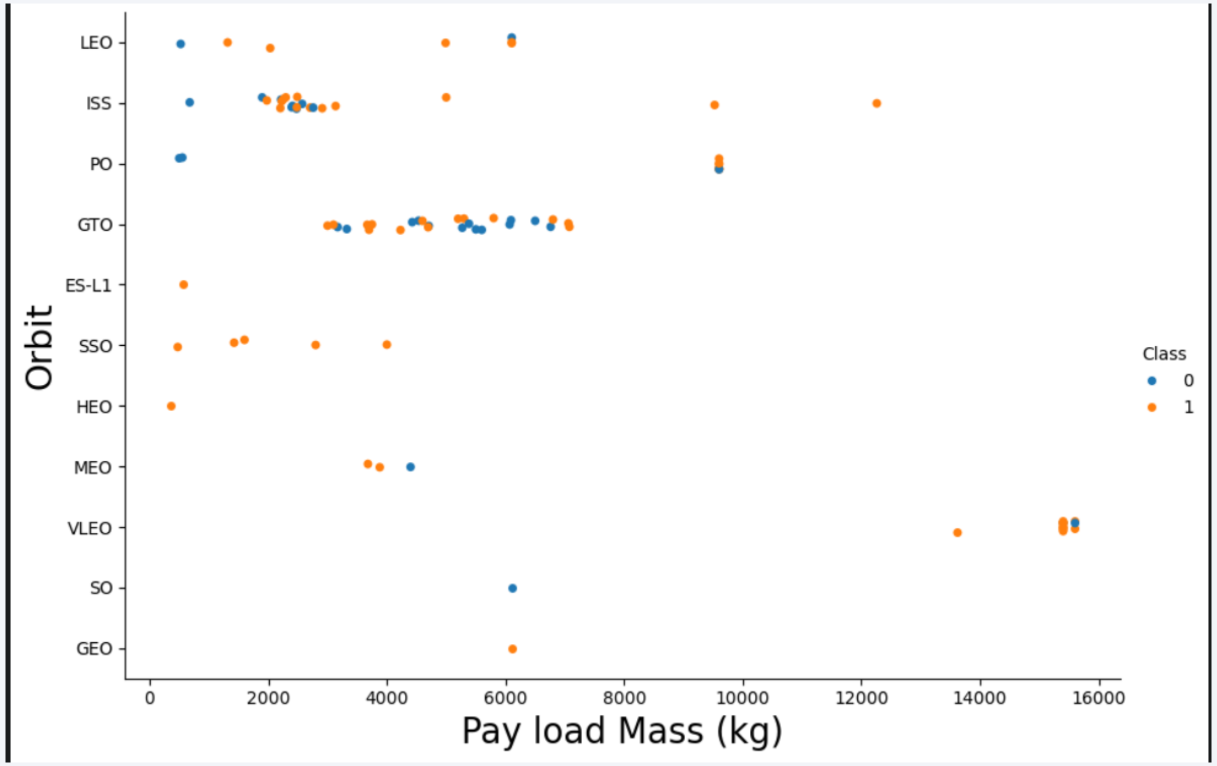
Flight Number vs. Orbit Type

- A scatter plot of Flight Number vs. Orbit Type



Payload vs. Orbit Type

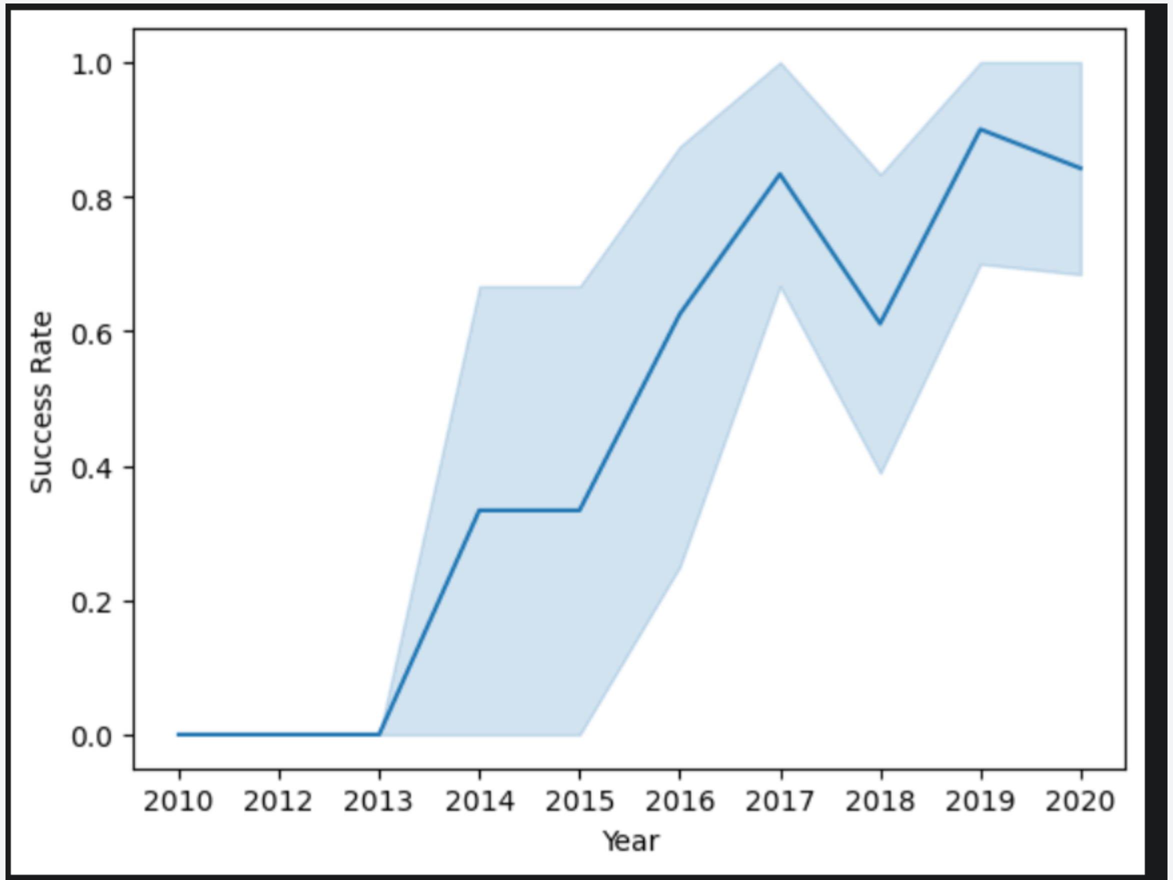
- Show a scatter point of payload vs. orbit type



- 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

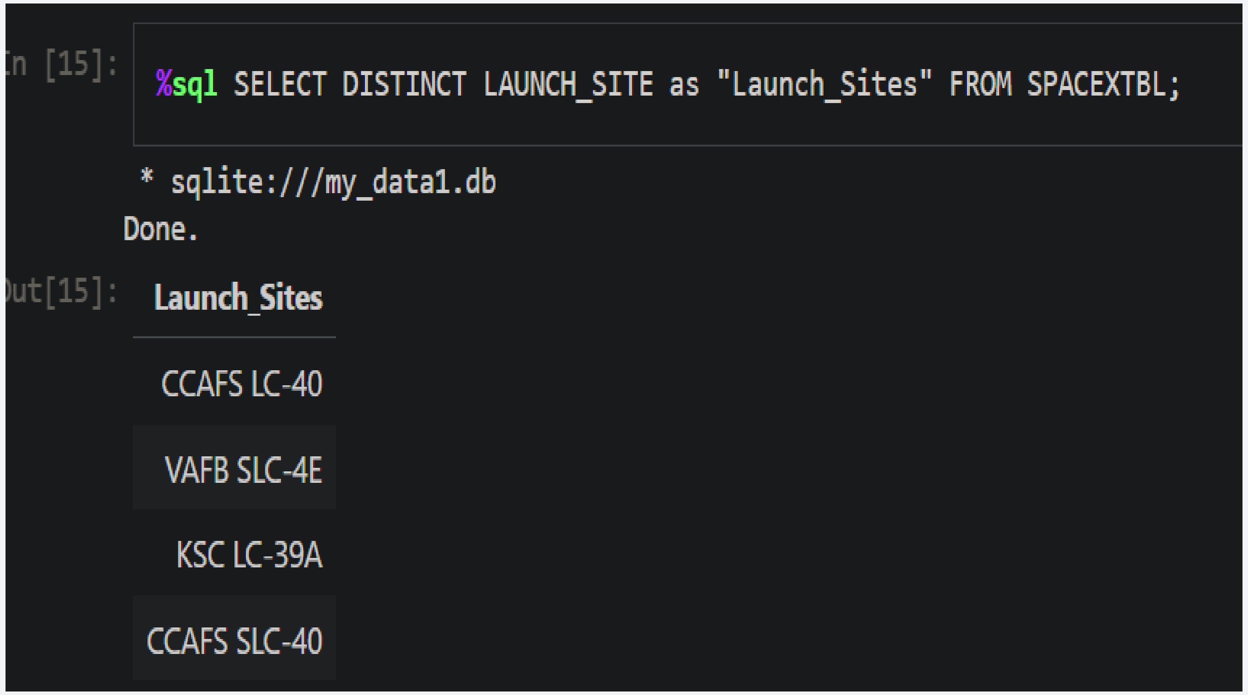
- Show a line chart of yearly average success rate



you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

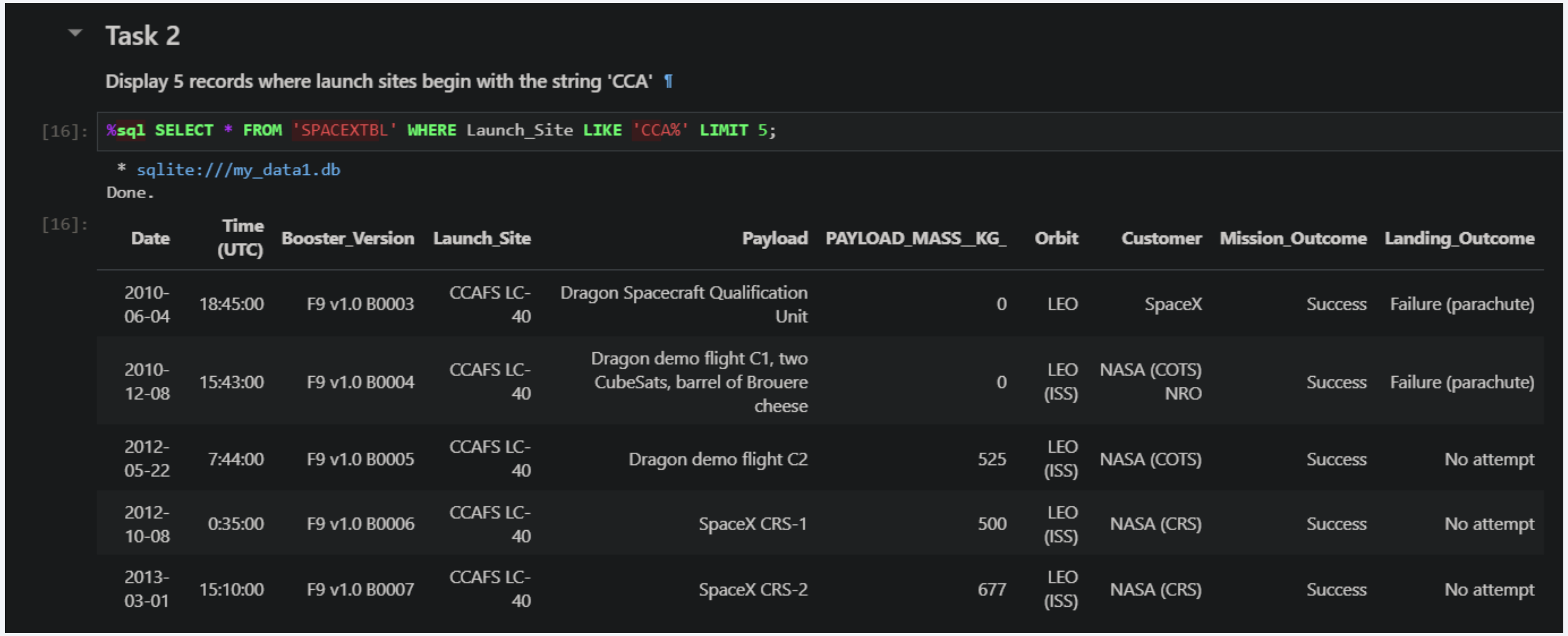
- Find the names of the unique launch sites



```
In [15]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;

 * sqlite:///my_data1.db
Done.

Out[15]: Launch_Sites
         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' ¶

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

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

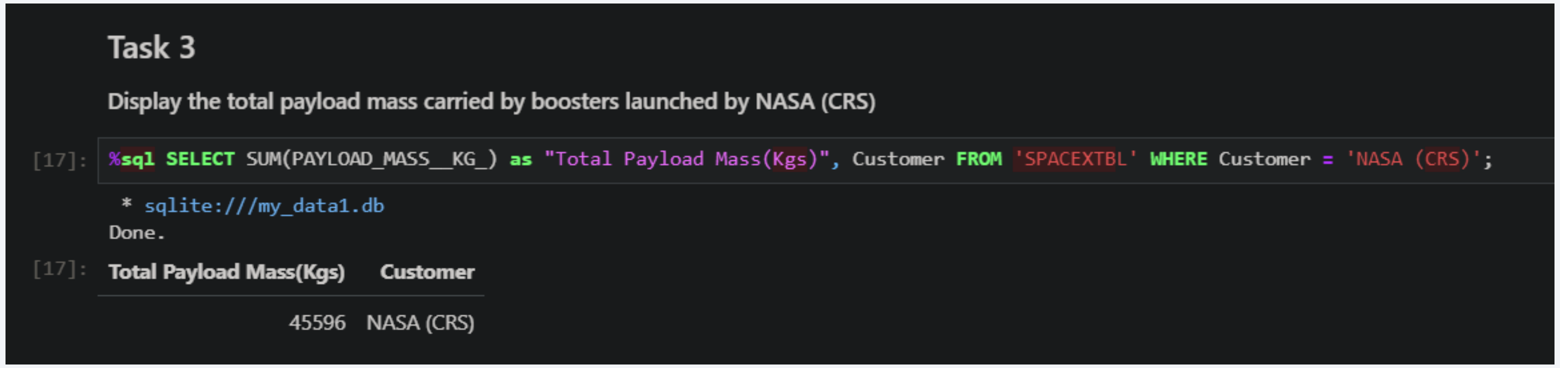
[16]:

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 the 'Like' command with '%' wildcard in the 'Where' clause to select and display a table of all records launch sites beginning with the string 'CCA'

Total Payload Mass

- Calculate the total payload carried by boosters from NASA



Task 3

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

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

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

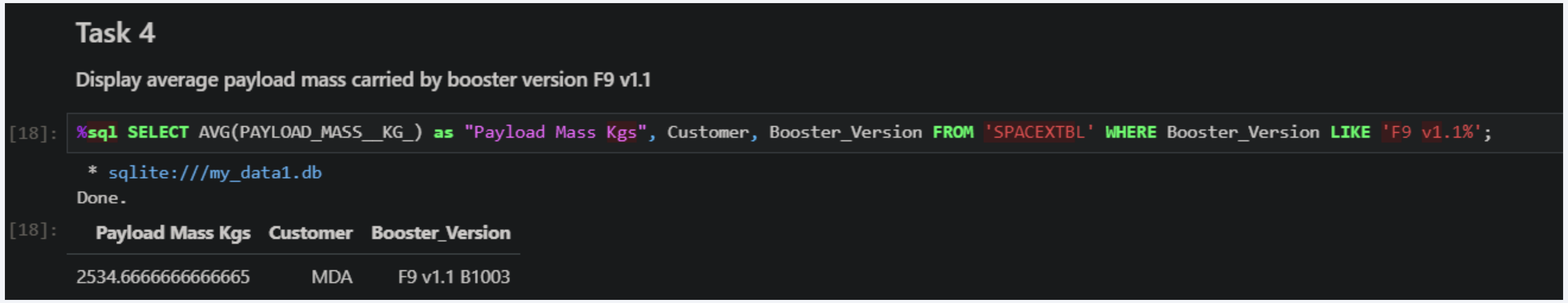
[17]:

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

- Used the ‘Sum()’ function to return and display 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

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

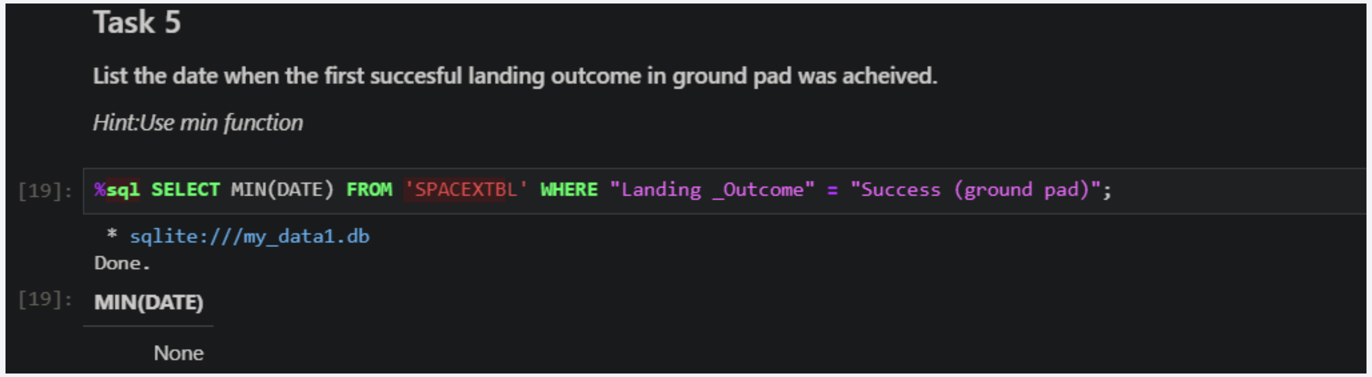
```
 * sqlite:///my_data1.db
Done.
```

```
[18]:   Payload Mass Kgs   Customer   Booster_Version
      2534.6666666666665     MDA        F9 v1.1 B1003
```

- Used the ‘Avg()’ function to return and display 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.

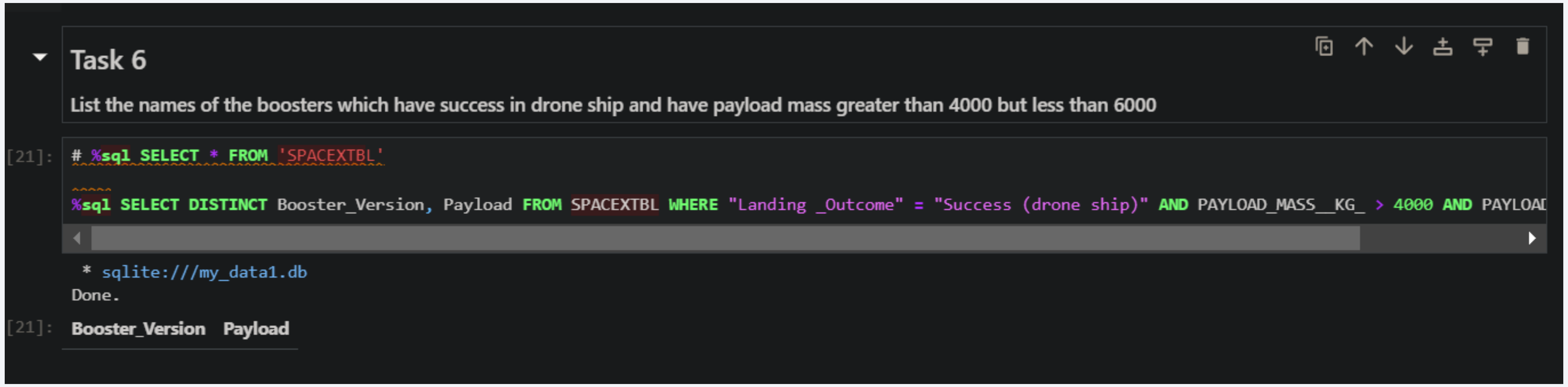
Hint:Use min function

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

 * sqlite:///my_data1.db
Done.
[19]: MIN(DATE)
          None
```

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

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

```
[21]: # %sql SELECT * FROM 'SPACEXTBL'

%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD
```

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

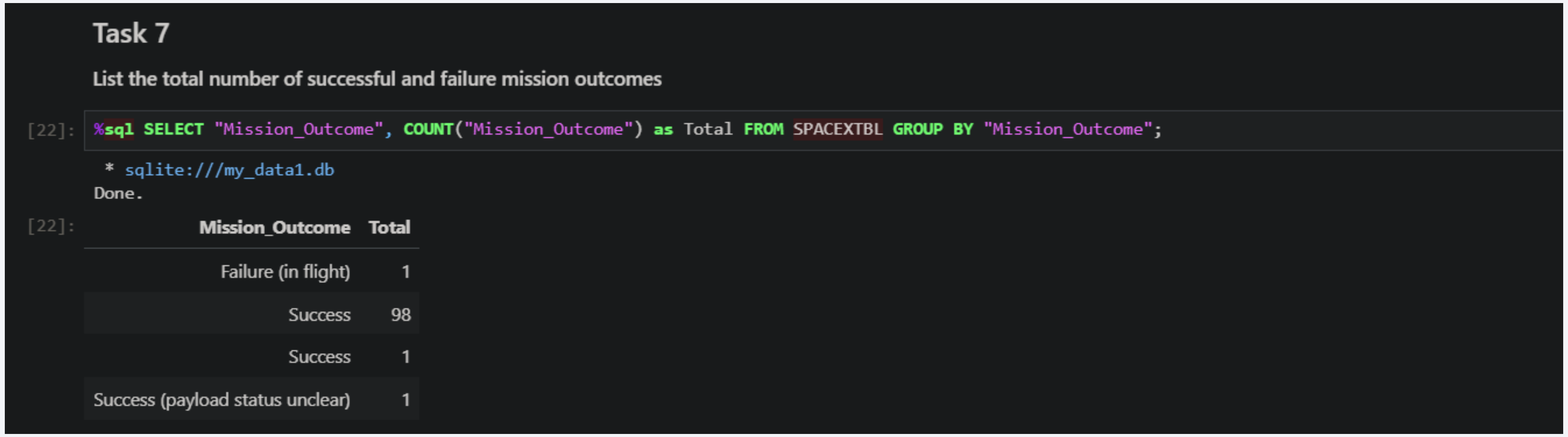
[21]:

Booster_Version	Payload

- Used the ‘Select Distinct’ statement to return and list the ‘Unique’ names of boosters with operators >4000 and <6000 to only list boosters with payloads between 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

```
[22]: %sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
 * sqlite:///my_data1.db
Done.
```

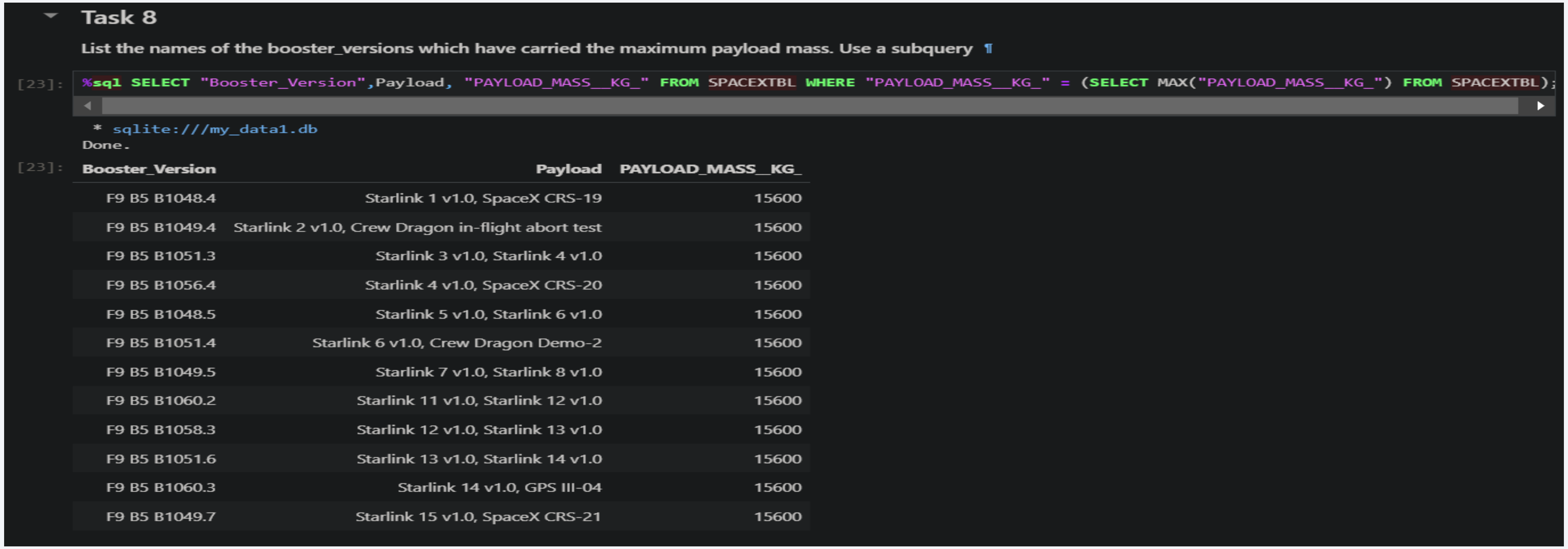
[22]:

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

- Used the ‘Count ()’ together with the ‘Group By’ statement to return the total number of missions 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 ¶

```
[23]: %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.
```

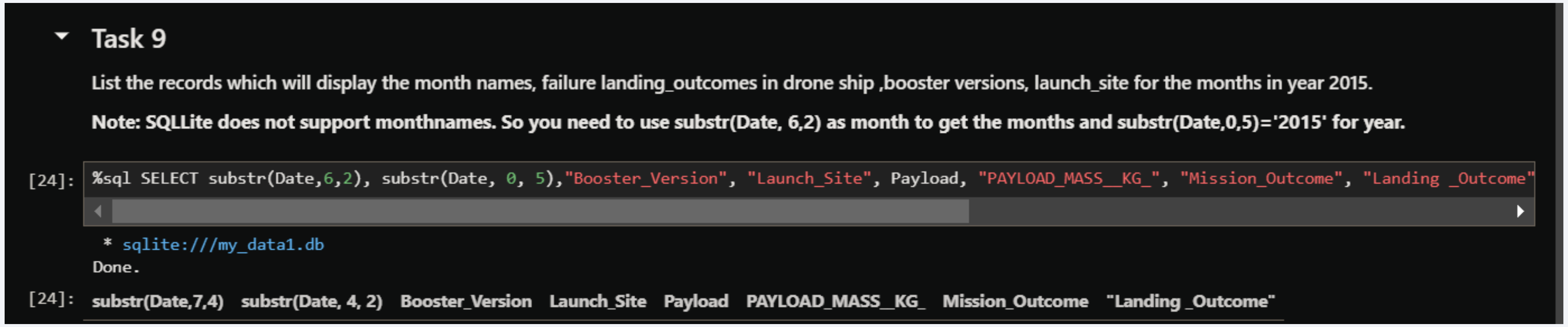
[23]:

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

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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in 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, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

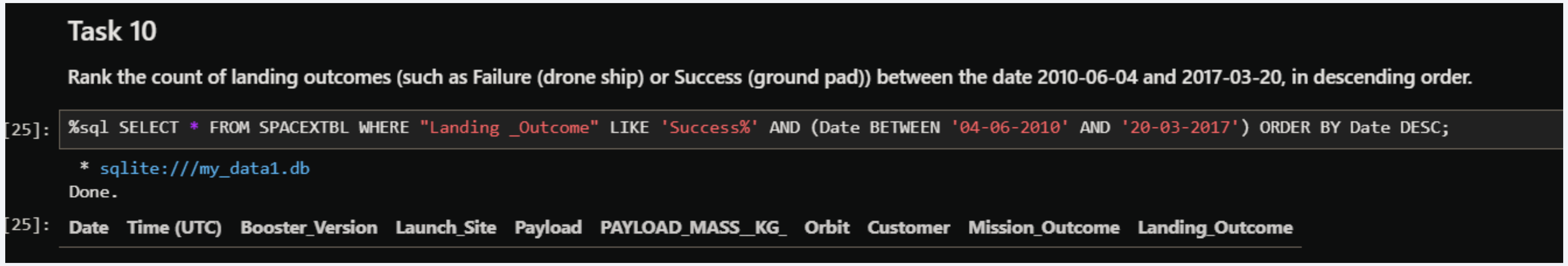
```
[24]: %sql SELECT substr(Date,6,2), substr(Date, 0, 5),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome"
```

```
 * sqlite:///my_data1.db
Done.
[24]: substr(Date,7,4)  substr(Date, 4, 2)  Booster_Version  Launch_Site  Payload  PAYLOAD_MASS__KG_  Mission_Outcome  "Landing _Outcome"
```

- Used the ‘substr ()’ in the select statement to get the month and year from the date column where substr(date, 6, 2)=‘2015’ for the year and landing_outcomes was ‘Failure (drone ship)’ and return the records matching 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

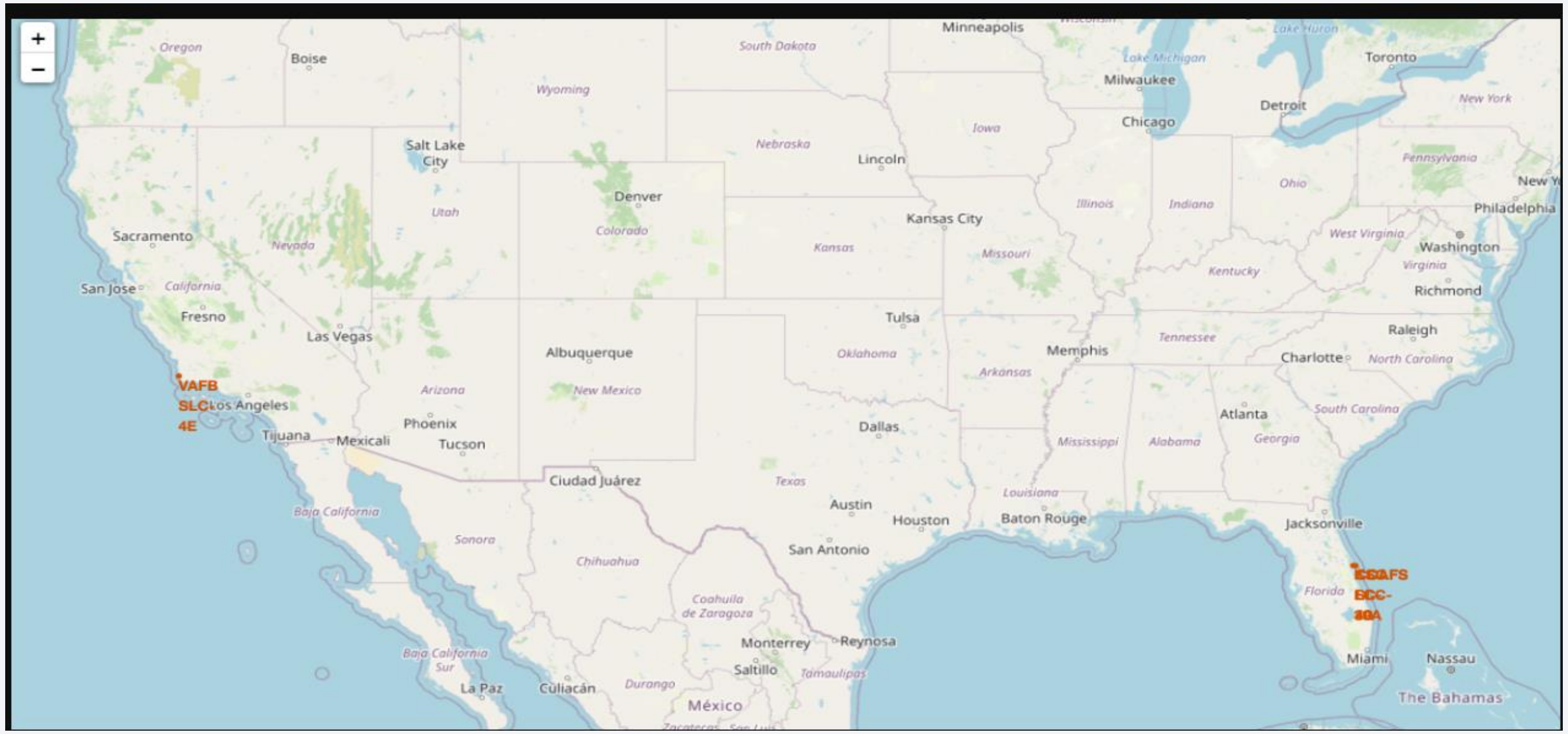
Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

```
[25]: %sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;

 * sqlite:///my_data1.db
Done.

[25]: Date  Time (UTC)  Booster_Version  Launch_Site  Payload  PAYLOAD_MASS__KG_  Orbit  Customer  Mission_Outcome  Landing_Outcome
```

Section 3

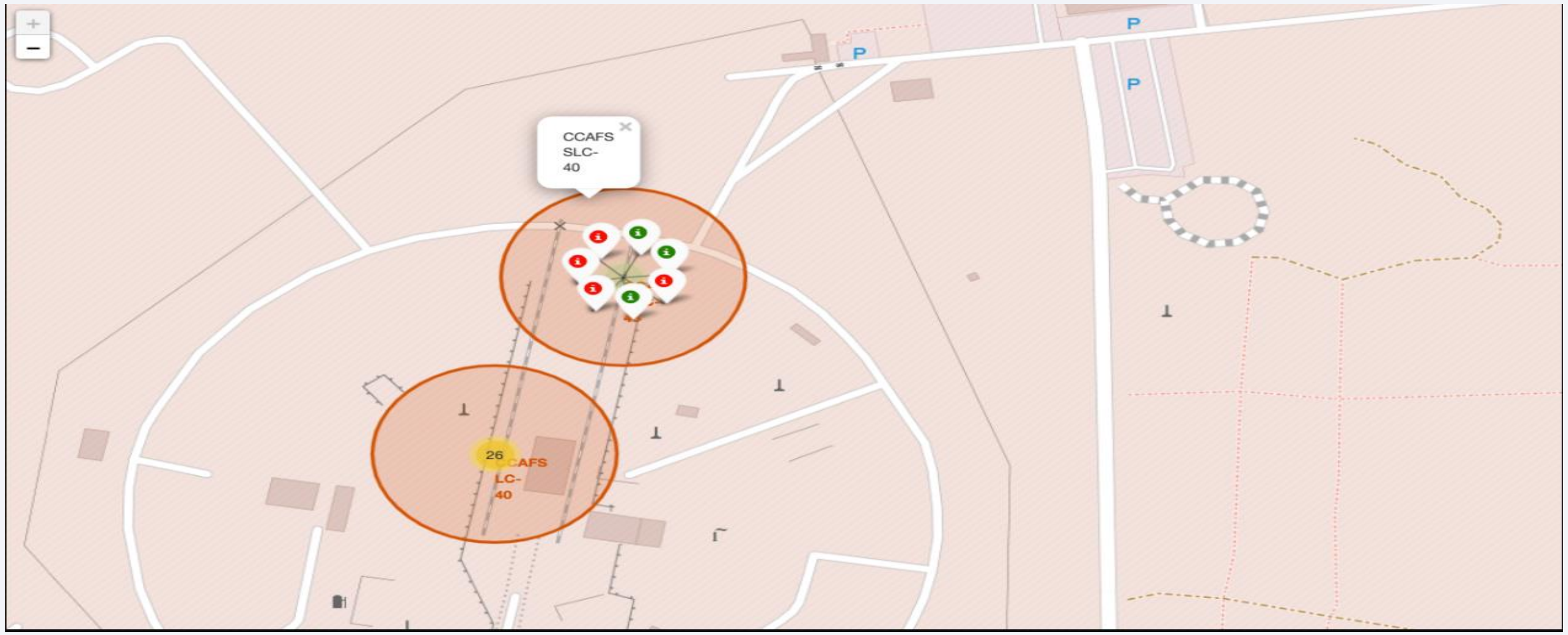
Launch Sites Proximities Analysis

Generated Map with Marked Launch Sites on a Global Map



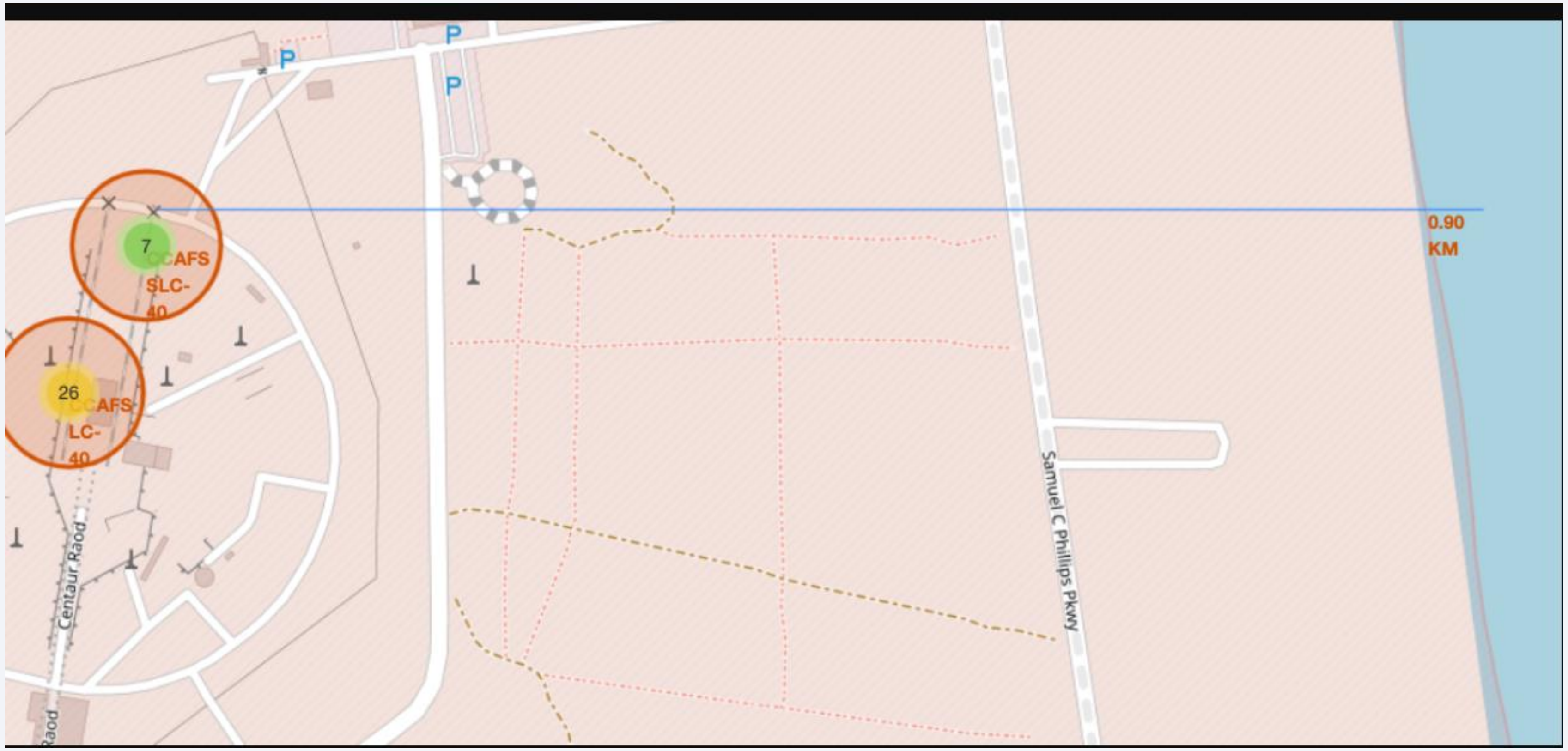
- All launch sites are in proximity to the Equator, (located southwards of the US map). Also, all the launch sites are in very close proximity to the coast.

Launch Outcome For Site on the Map with Color-Markers

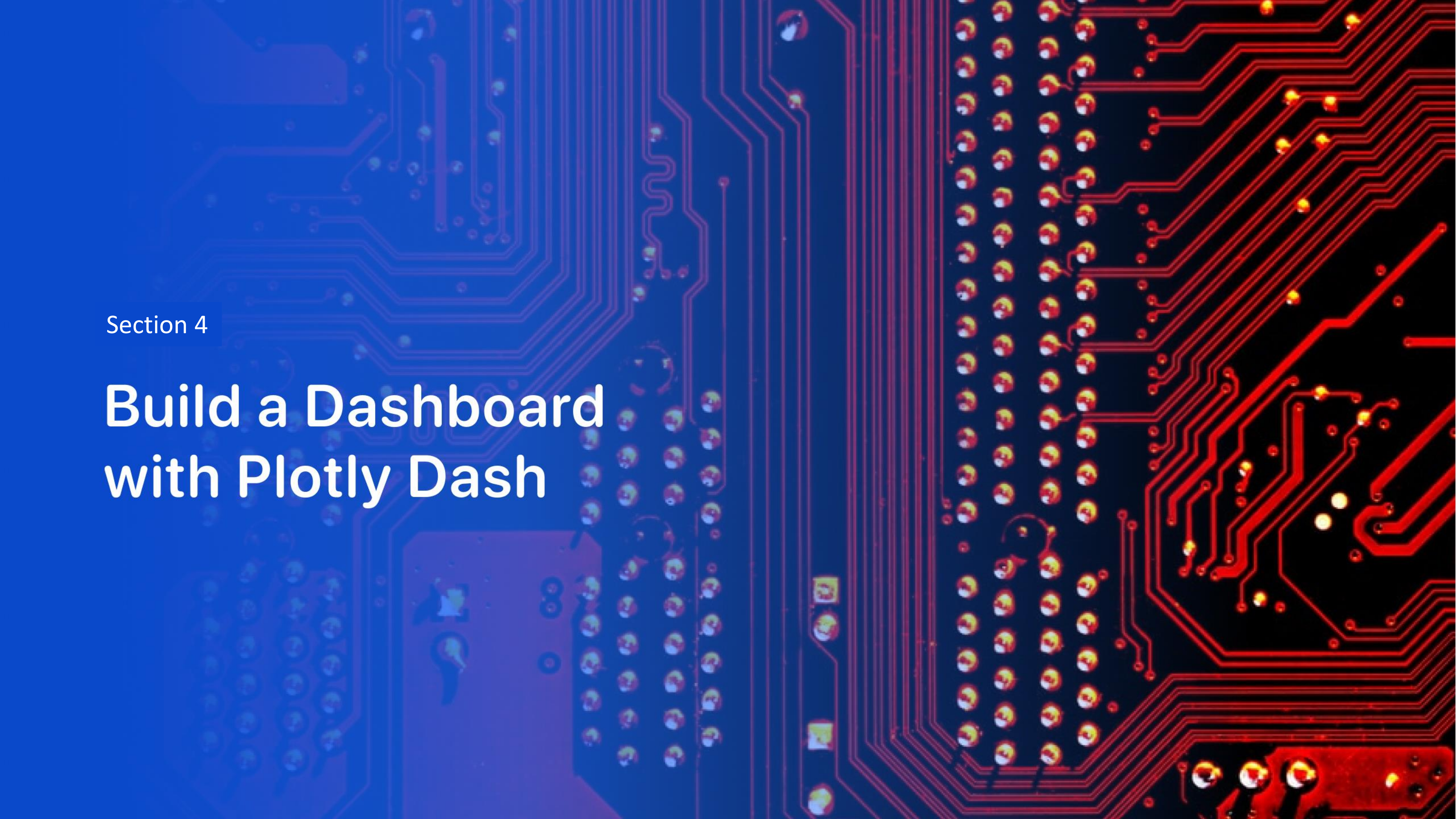


- In the Eastern coast (Florida) launch sites CCAFS SLC-40 & CCAFS LC-40 are relatively high.

Distance Between Launch Site to its Proximities



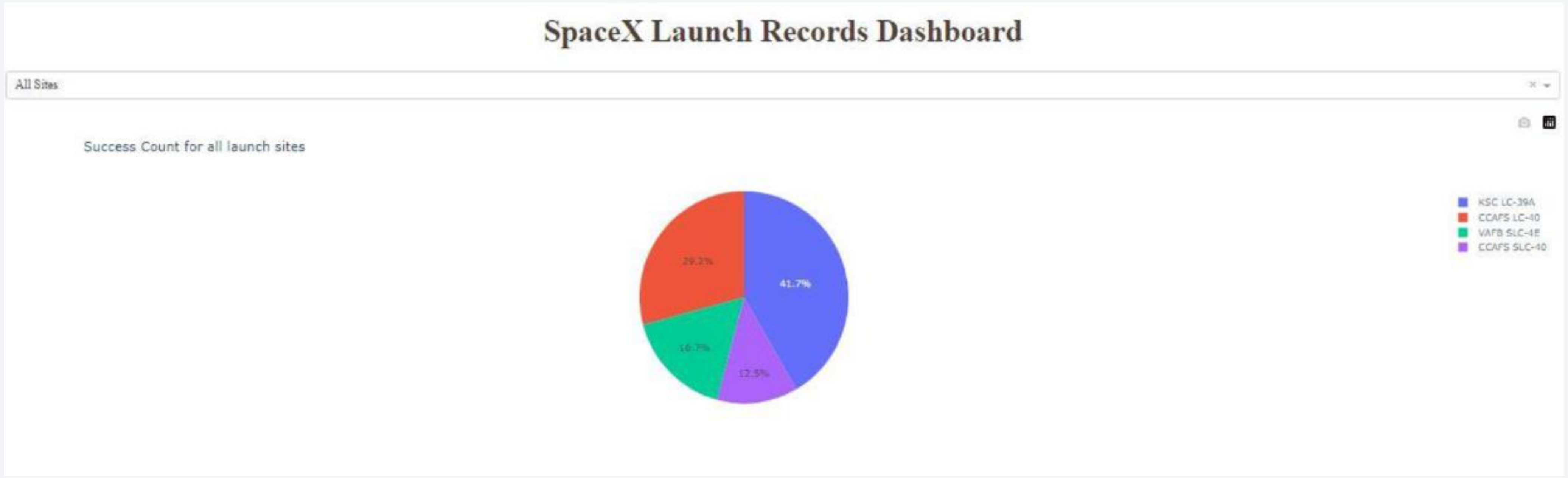
- Launch site CCAFS SLC-40 proximity coastline is 0.90km



Section 4

Build a Dashboard with Plotly Dash

Pie-Chart for Launch Success for all Site

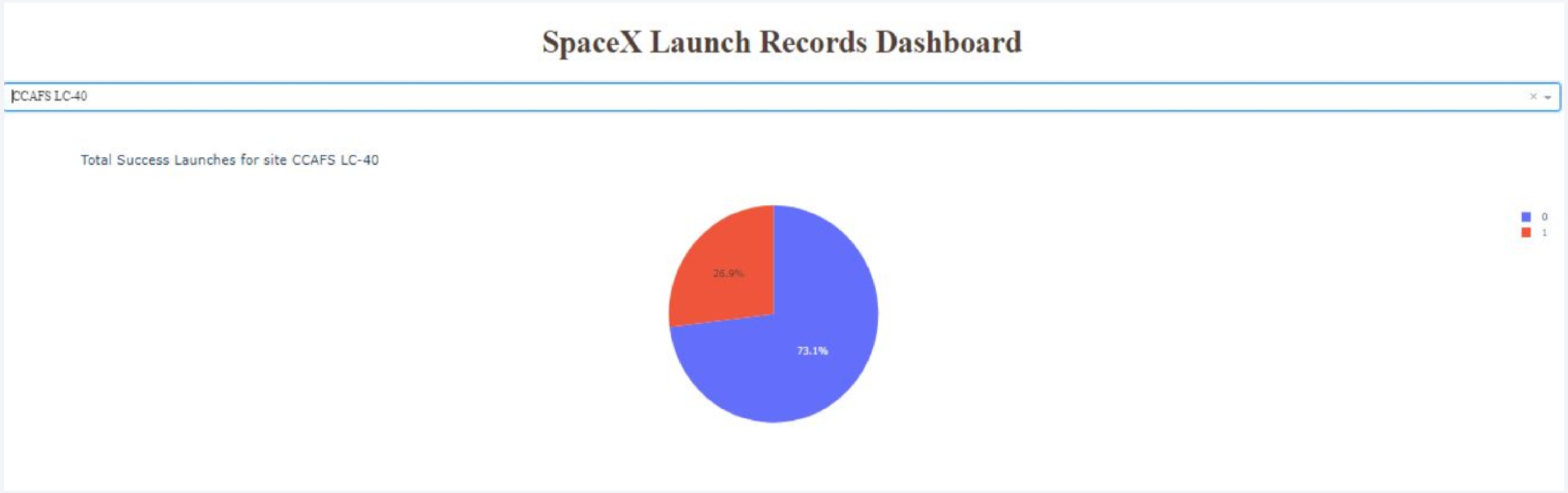

SpaceX Launch Records Dashboard

All Sites

Success Count for all launch sites

Launch site CCAFS SLC-40 has the highest launch success rate at 13%

Pie-Chart for the Launch Site with Highest Launch Success Ratio

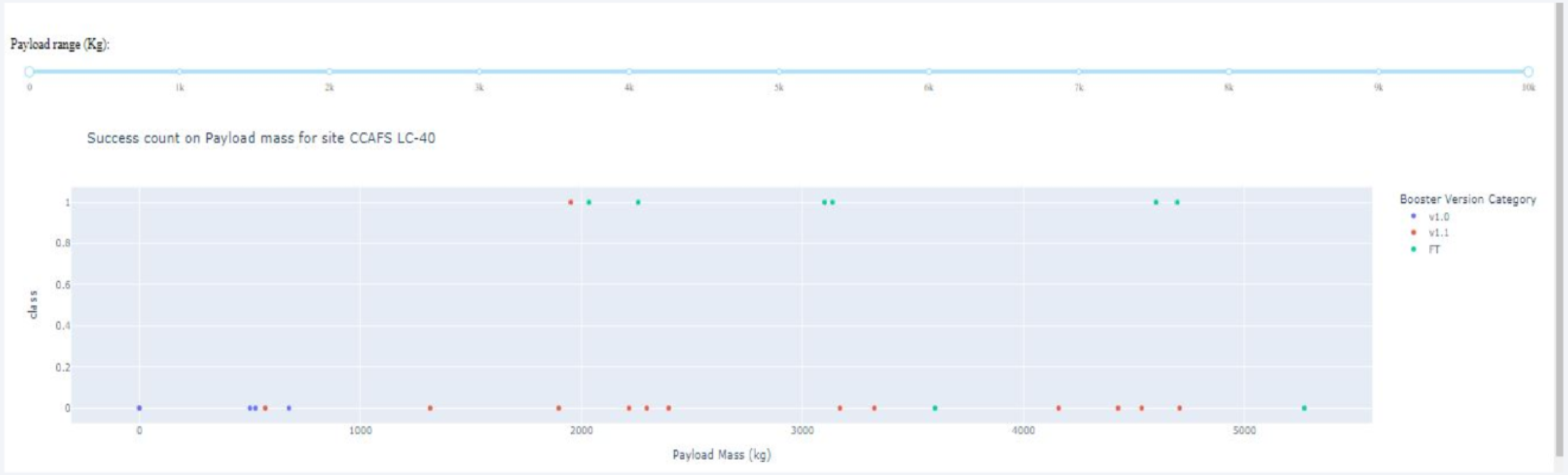

SpaceX Launch Records Dashboard

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 for all Site

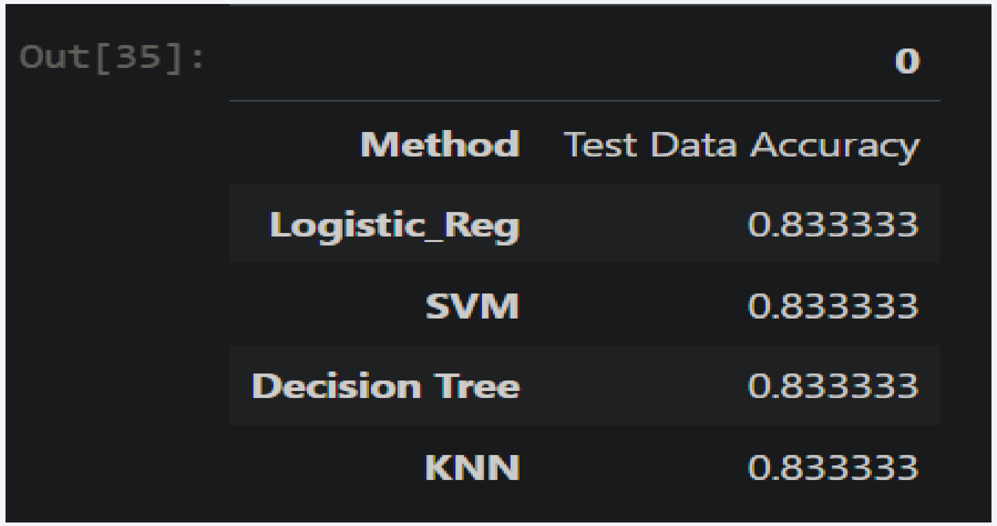


- For Launch site CCAFS LC-40 the booster version FT has the largest success rate from a payload mass of >2000kg

Section 5

Predictive Analysis (Classification)

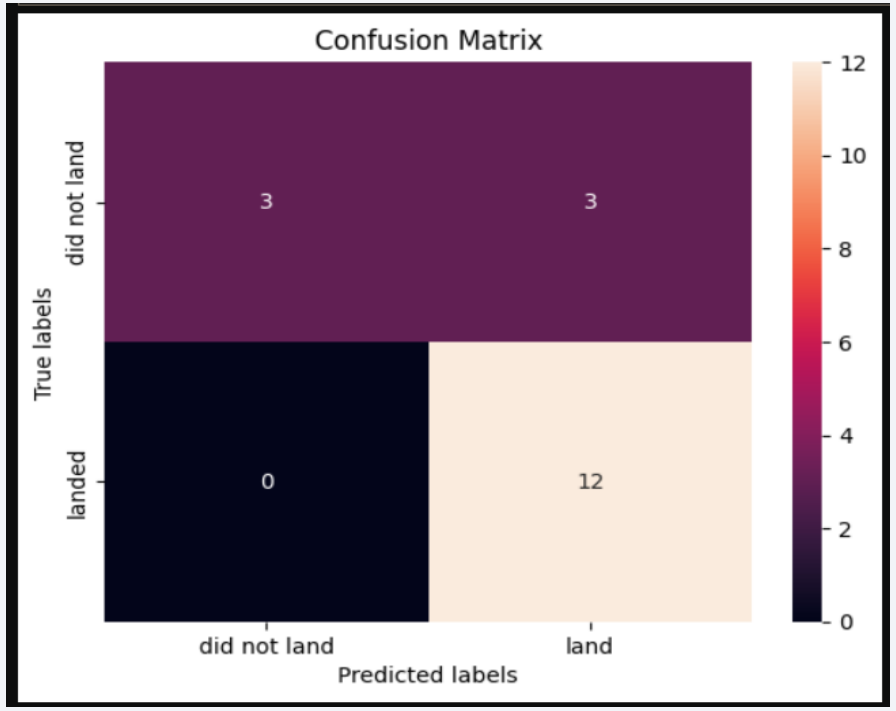
Classification Accuracy



```
Out[35]:
                                 0
        Method  Test Data Accuracy
  Logistic_Reg            0.833333
           SVM            0.833333
 Decision Tree            0.833333
           KNN            0.833333
```

- All the methods are performed equally, they all have the same accuracy of 0.833333 on the test data.

Confusion Matrix



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

Conclusions

- Different launch sites have different success rates. CCAFS LC-40, has a success rate of 73%, while CCAFS SLC-40 has a success rate of 13%.
- We can deduce that, as the flight number increases in each of the launch sites, so does the success rate. For example, the success rate for the CCAFS LC-40 launch site is 100% after Flight number 50. CCAFS SLC-40 has a 100% success rate after the 80th flight.
- If you observe the Payload vs. Launch Site Scatter point chart you will find for the VAFB-SLC launch site, there are no rockets launched for heavy payload mass (greater than 10000).
- LEO orbit Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight numbers when in GTO orbit.
- Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, with SO orbit having the lowest success rate at ~50%. Orbit SO has a 0% success rate.

Thank you!