Winning Space Race with Data Science

<Prasad Ratnaparkhi>

<03-Feb-2024>

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context

  Space X 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 Space X 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers

  -What factors determine if the rocket will land successfully?

  -The interaction amongst various features that determine the success rate of a successful landing.

  -What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX launch data that is gathered from an API, specifically the SpaceX REST API.
- 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

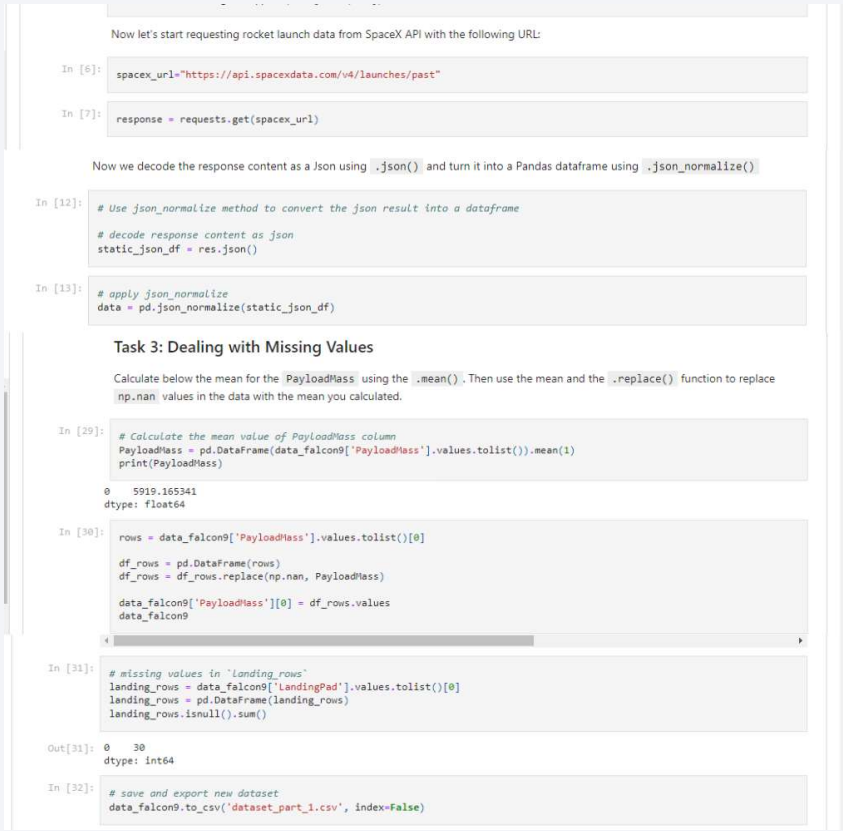
Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
  - We then cleaned the data, checked for missing values and fill in missing values where necessary.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.
- The link o the notebook is

https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Data%20Collecction%20API.ipynb



Now let's start requesting rocket launch data from SpaceX API with the following URL:

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"
In [7]: response = requests.get(spacex_url)
```

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

```
In [12]: # Use json_normalize method to convert the json result into a dataframe

         # decode response content as json
         static_json_df = res.json()
In [13]: # apply json_normalize
         data = pd.json_normalize(static_json_df)
```

Task 3: Dealing with Missing Values

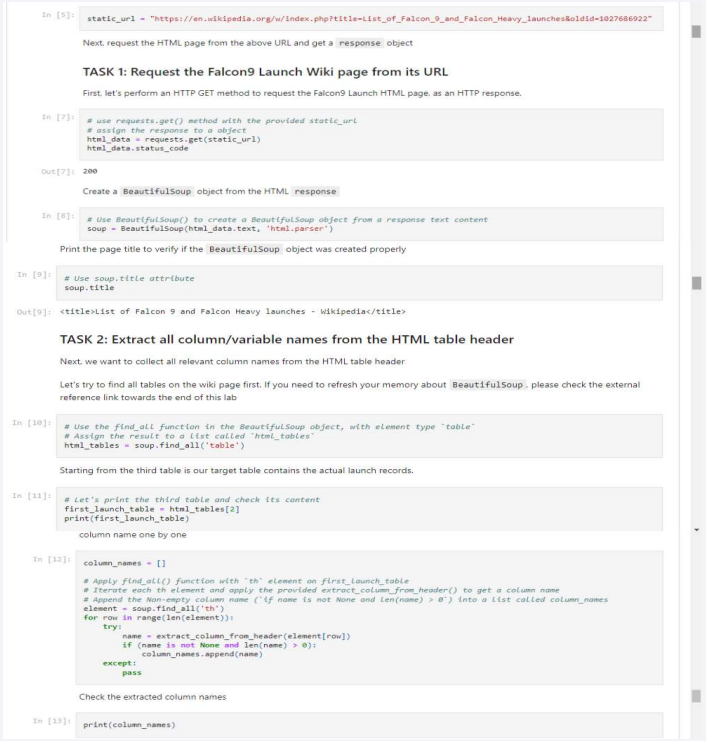
Calculate below the mean for the `PayloadMass` using the `.mean()`. Then use the mean and the `.replace()` function to replace `np.nan` values in the data with the mean you calculated.

```
In [29]: # Calculate the mean value of PayloadMass column
         PayloadMass = pd.DataFrame(data_falcon9['PayloadMass'].values.tolist()).mean(1)
         print(PayloadMass)

0    5919.165341
dtype: float64

In [30]: rows = data_falcon9['PayloadMass'].values.tolist()[0]

         df_rows = pd.DataFrame(rows)
         df_rows = df_rows.replace(np.nan, PayloadMass)

         data_falcon9['PayloadMass'][0] = df_rows.values
         data_falcon9

In [31]: # missing values in `Landing_rows`
         landing_rows = data_falcon9['LandingPad'].values.tolist()[0]
         landing_rows = pd.DataFrame(landing_rows)
         landing_rows.isnull().sum()

Out[31]: 0    30
         dtype: int64

In [32]: # save and export new dataset
         data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup.
- We parsed the table and converted it into a pandas dataframe.
- The link to the notebook is

https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Data%20Collection%20with%20Web%20Scraping.ipynb



In [5]:
```
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.

In [7]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url)
html_data.status_code
```

Out[7]: 200

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

In [8]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text, 'html.parser')
```

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

In [9]:
```
# Use soup.title attribute
soup.title
```

Out[9]: `<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

In [10]:
```
# 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.

In [11]:
```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

column name one by one

In [12]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

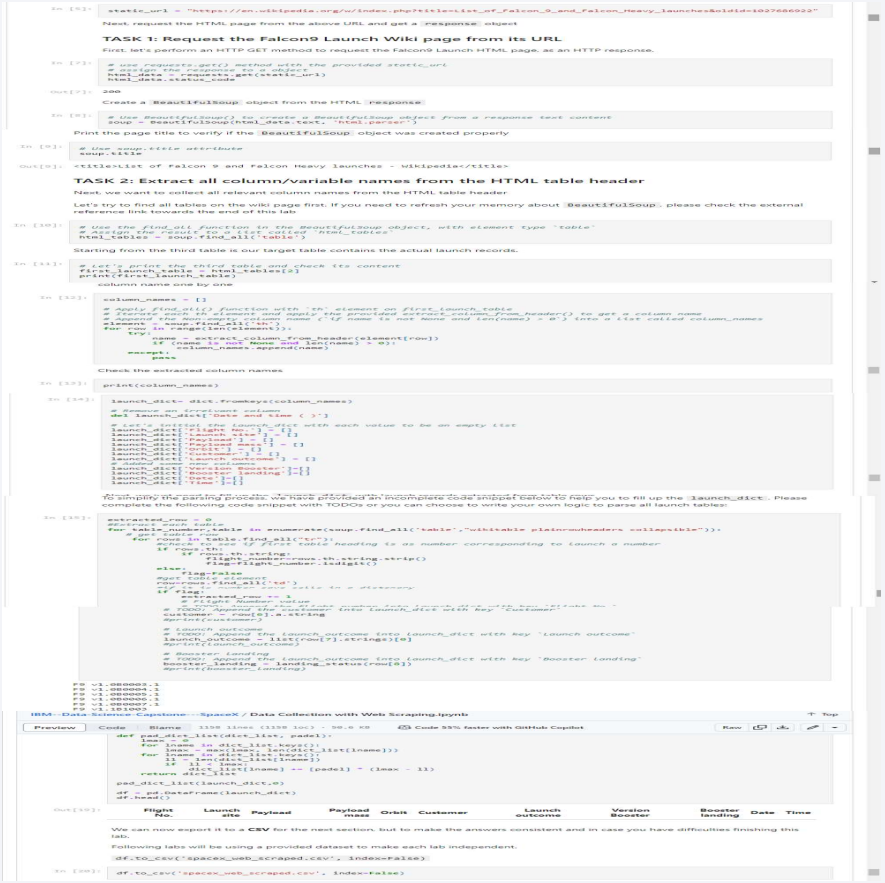
Check the extracted column names

In [13]:
```
print(column_names)
```

Data Wrangling

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is

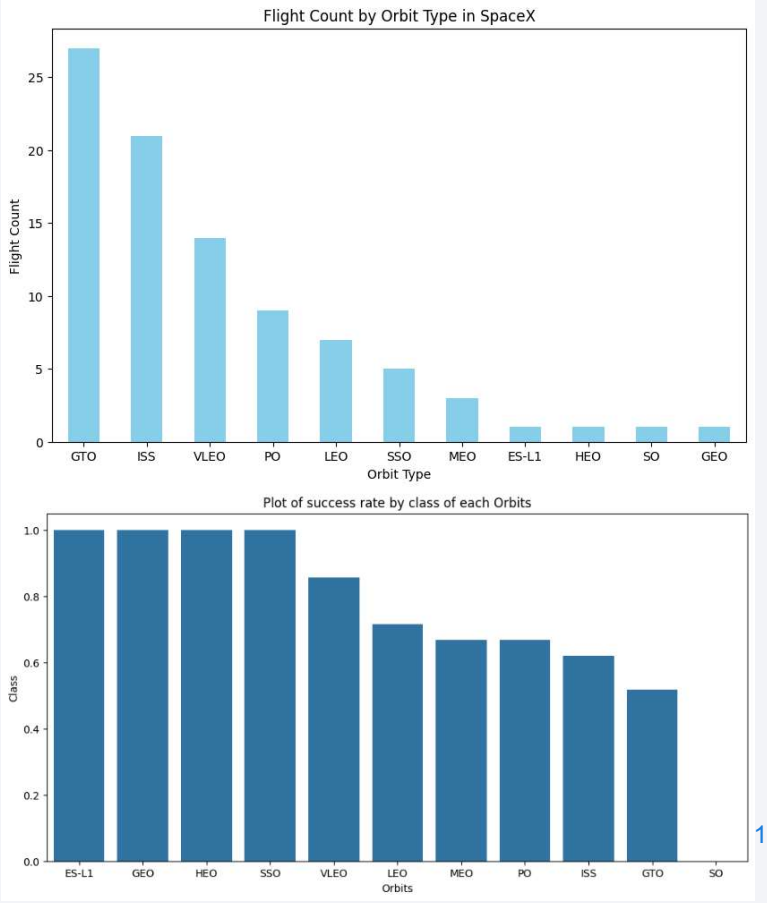
https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Data%20Wrangling.ipynb



EDA with Data Visualization

- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number, the launch success yearly trend and the orbit type.
- The link to the notebook is

https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/EDA%20with%20Data%20Visualization.ipynb



EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
  - List the date when the first successful landing outcome in ground pad was achieved.
- The link to the notebook is https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/EDA_with_SQL.ipynb

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
    - Are launch sites near railways, highways and coastlines.
    - Do launch sites keep certain distance away from cities.
- Added the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- Added the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/AppDash1.ipynb

Predictive Analysis (Classification)

- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- We loaded the data using Numpy and Pandas, transformed the data, split our data into training and testing set.
- We built different machine learning models and tune different hyper parameters using GridSearchCV.
- The link is given as https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Machine%20Learning%20Prediction.ipynb

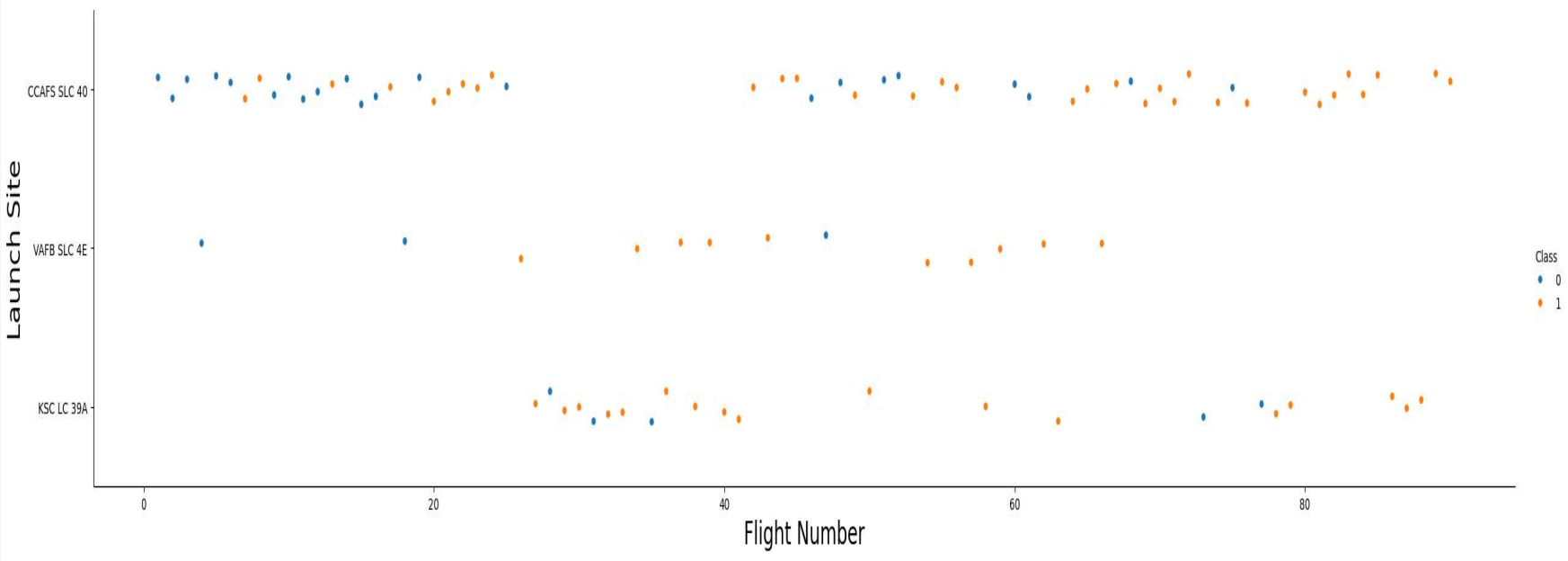
Results

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

Section 2

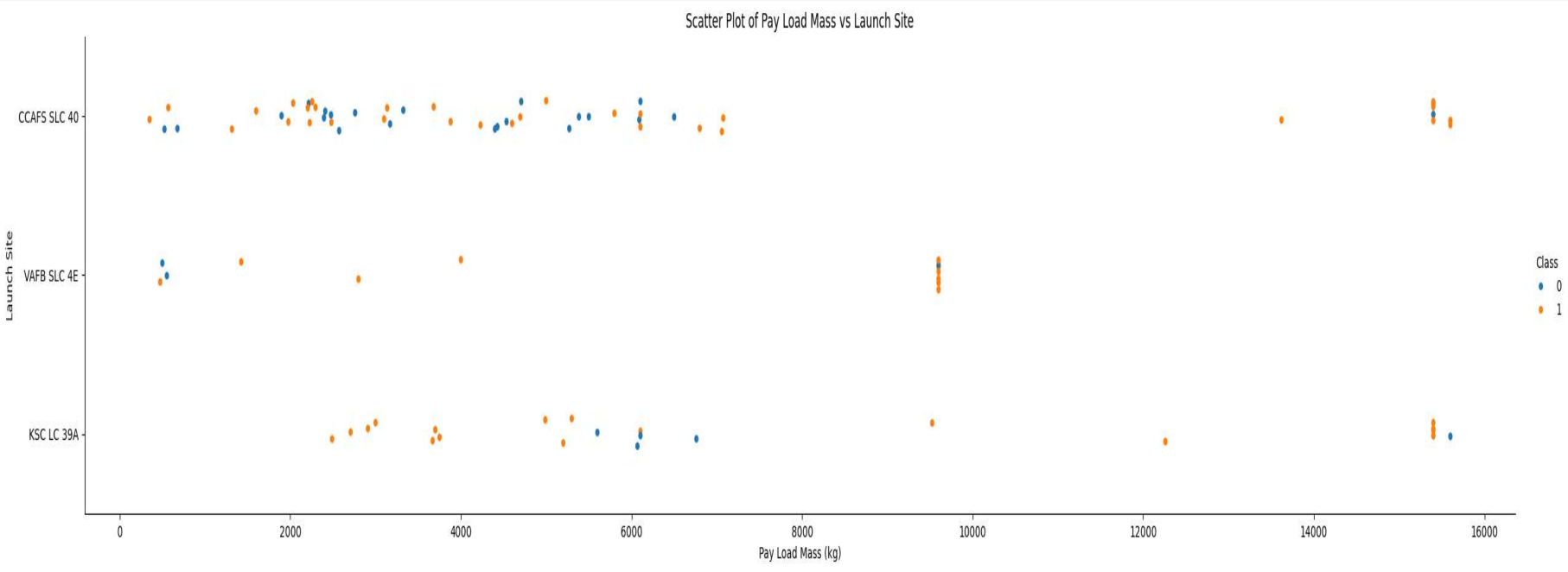
Flight Number vs. Launch Site

- A scatter plot of Flight Number vs. Launch Site
- As per observation from the plot, it is found that the larger the flight amount at a launch site, the greater the success rate at a launch site.



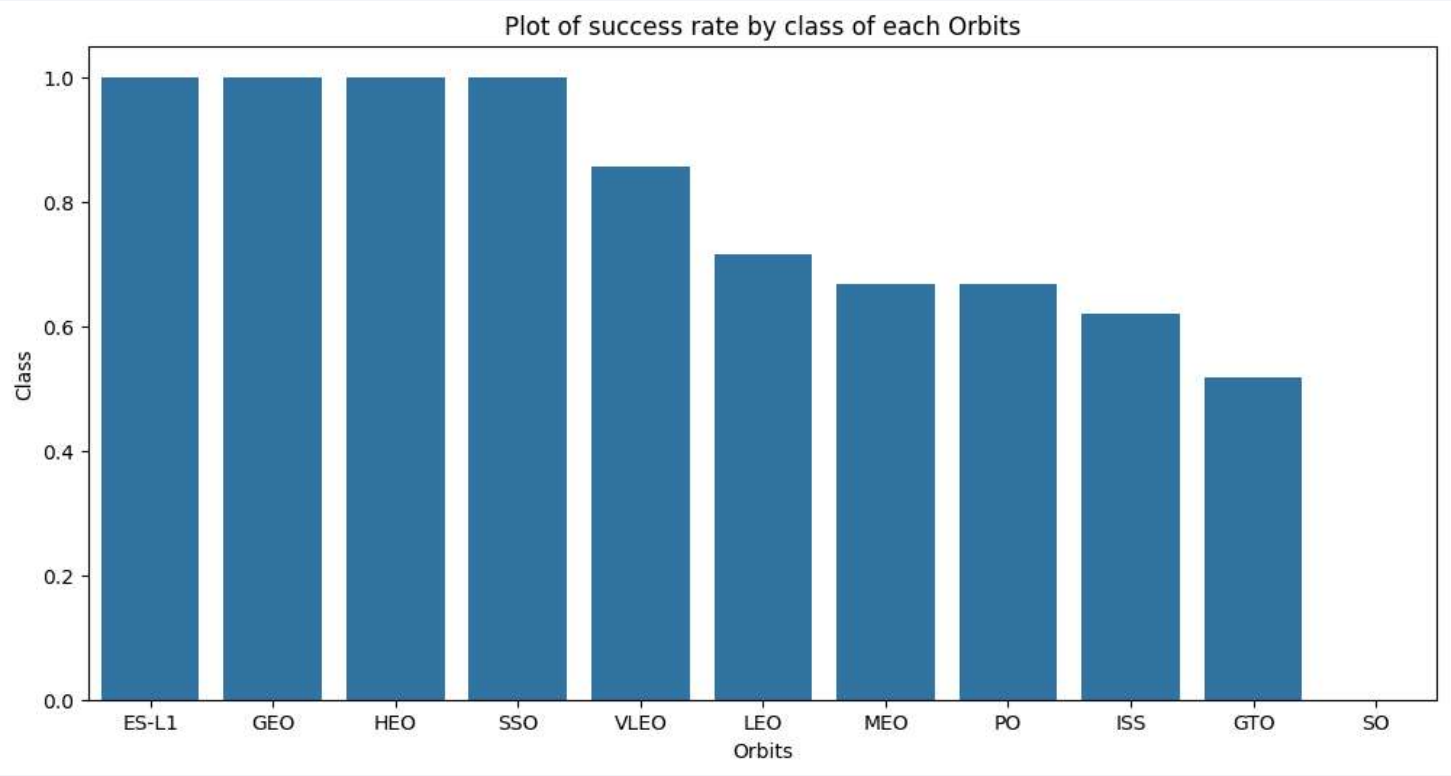
Payload vs. Launch Site

- As per observation, it is found that the greater the Pay Load Mass for Launch Site CCAFS SLC-40 the higher the success rate for the mission.



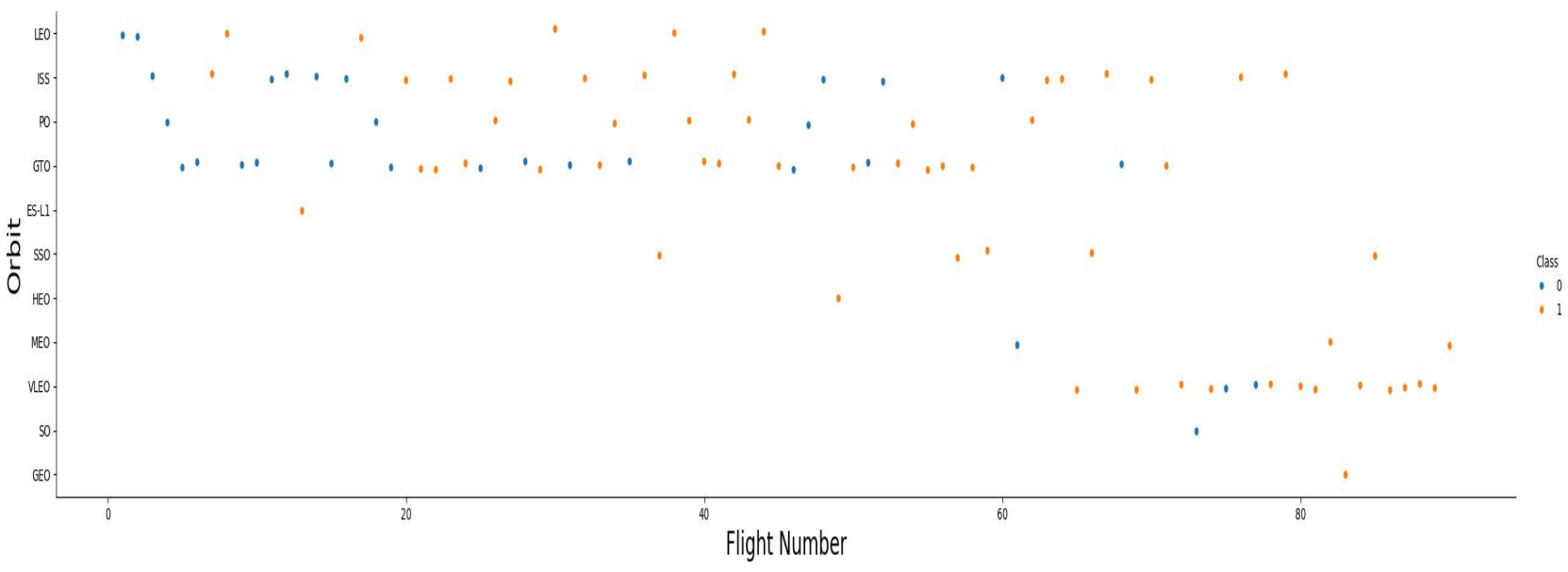
Success Rate vs. Orbit Type

- As per observation from the plot, it can been seen that ES-L1, GEO, HEO, SSO, VLEO has the most success rate.



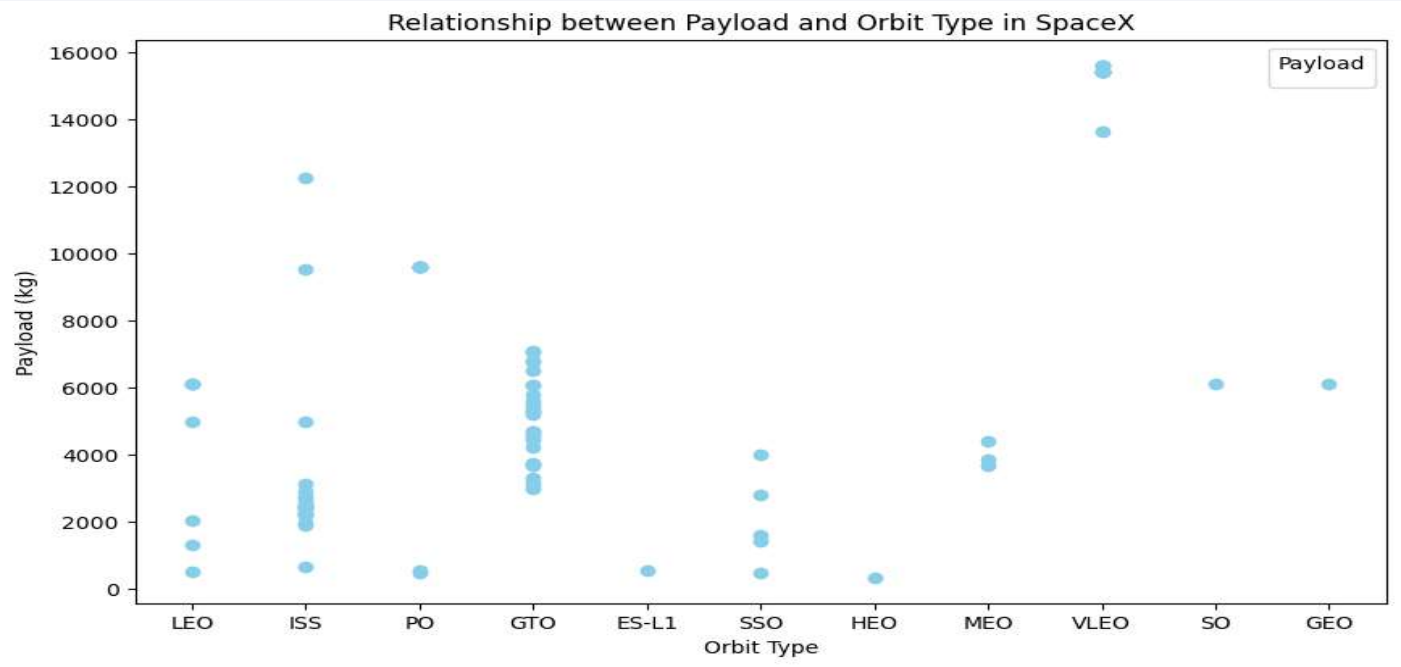
Flight Number vs. Orbit Type

- As per observation from the plot, it is seen that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



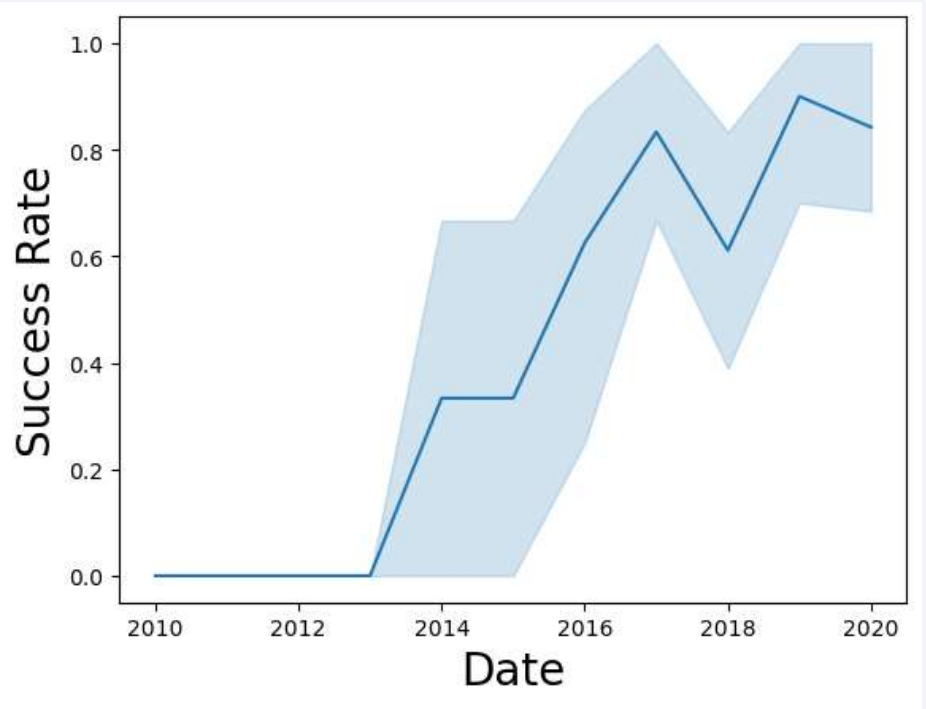
Payload vs. Orbit Type

- As per observation from the graph, it can be seen that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



Launch Success Yearly Trend

- As per observation from the graph, it can be seen that the success rate since 2013 kept increasing till 2020.



All Launch Site Names

- To find the names of the unique launch sites, we use SELECT DISTINCT launch_site from SPACEXTABLE.

Task 1 Display the names of the unique launch sites in the space mission

In [20]:
```
%sql select distinct launch_site from SPACEXTABLE;
```

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

Out[20]:

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

Launch Site Names Begin with 'CCA'

- To find the 5 records where launch sites begin with `CCA`, we use SELECT * from SPACEXTABE where launch_site like ‘CCA’ limit 5.

Task 2 Display 5 records where launch sites begin with the string 'CCA'

```
21]: %sql select * from SPACEXTABLE where launch_site like 'CCA%' limit 5;
 * sqlite:///my_data1.db
Done.
```

21]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Or
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	L
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	L (I
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	L (I
2012-	0:35:00	F9 v1.0 B0006	CCAFS LC-	SpaceX	500	L

Total Payload Mass

- After calculation the total payload carried by boosters from NASA is 45596 as per given query.

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

In [23]:
```
%sql select sum(payload_mass__kg_) as total_payload_mass from SPACEXTABLE where customer = 'NASA (CRS)';
```

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

Out[23]:

total_payload_mass
45596

Average Payload Mass by F9 v1.1

- After the calculation, the average payload mass carried by booster version F9 v1.1 is 2534.66

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

```
n [24]: %sql select avg(payload_mass__kg_) as average_payload_mass from SPACEXTABLE where booster_version like '%F9 v1.1%';
```

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

ut[24]:

average_payload_mass
2534.6666666666665

First Successful Ground Landing Date

- The dates of the first successful landing outcome on ground pad is seen as 22-12-2015.

Task 5 List the date when the first succesful landing outcome in ground pad was acheived.

In [33]:
```
%sql select min(Date) as first_successful_landing from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)';
```

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

Out[33]:

first_successful_landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Following are the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 using WHERE clause & AND condition.

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 [45]:
```
%sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000 AND LANDING_OUTCOME =
```

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

Out[45]:

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

Total Number of Successful and Failure Mission Outcomes

- After the calculation, the total number of successful and failure mission outcomes is given below.

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

```
n [47]: %sql select mission_outcome, count(*) as total_number from SPACEXTABLE group by mission_outcome;
```

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

ut[47]:

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

Boosters Carried Maximum Payload

- Following are the list of names of the booster which have carried the maximum payload mass.

```
In [54]: %sql select booster_version from SPACEXTABLE where payload_mass__kg_ = (select max(payload_mass__kg_) FROM SPACEXTABLE
  sqlite://
 * sqlite:///my_data1.db
Done.
```

Out[54]:

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

2015 Launch Records

- Following are 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.

In [63]:
```
%%sql SELECT strftime('%Y-%m', Date) AS month_year, Landing_Outcome,
       booster_version,
       launch_site
FROM SPACEXTABLE
WHERE strftime('%Y', Date) = '2015'
AND landing_outcome LIKE '%Failure (drone ship)%';
```

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

Out[63]:

month_year	Landing_Outcome	Booster_Version	Launch_Site
2015-01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- Following are the 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.

[66]:
```
%%sql select landing_outcome, count(*) as count_outcomes from SPACEXTABLE
      where date between '2010-06-04' and '2017-03-20'
      group by landing_outcome
      order by count_outcomes desc;
```

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

[66]:

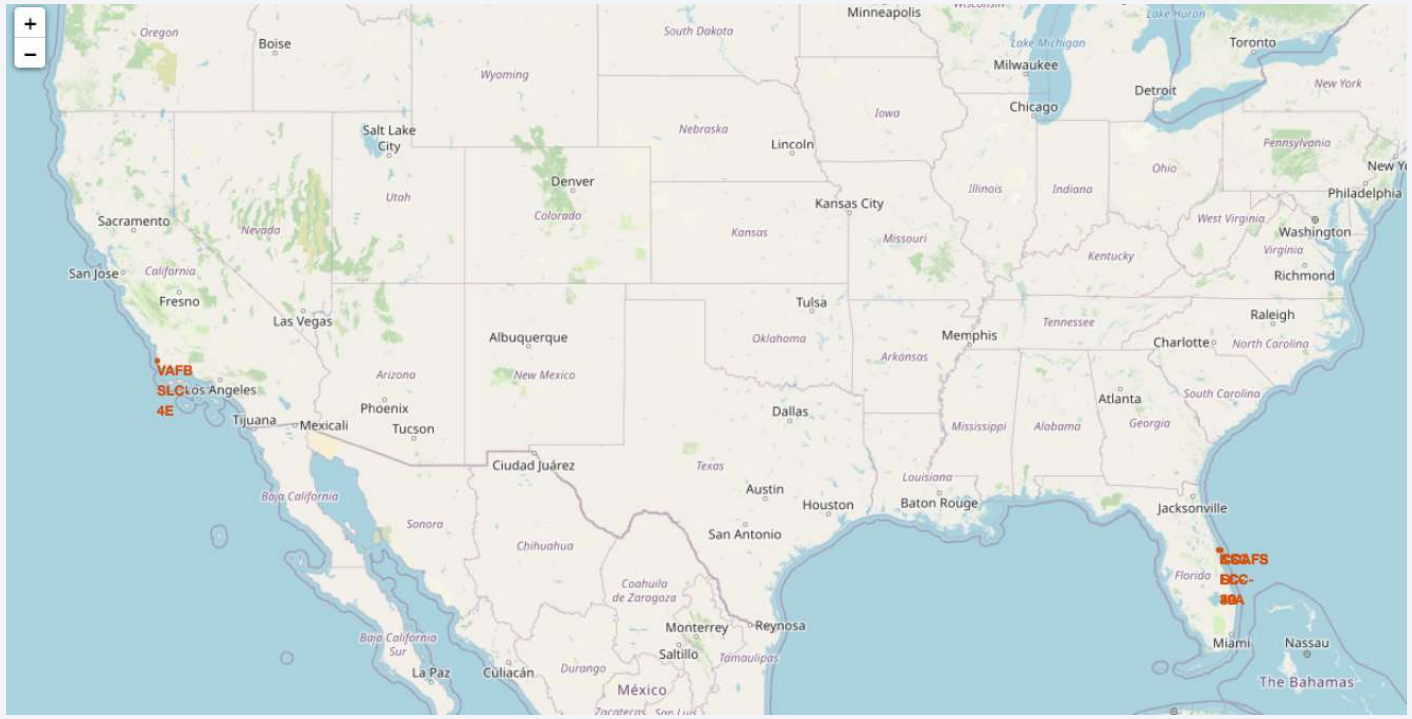
Landing_Outcome	count_outcomes
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

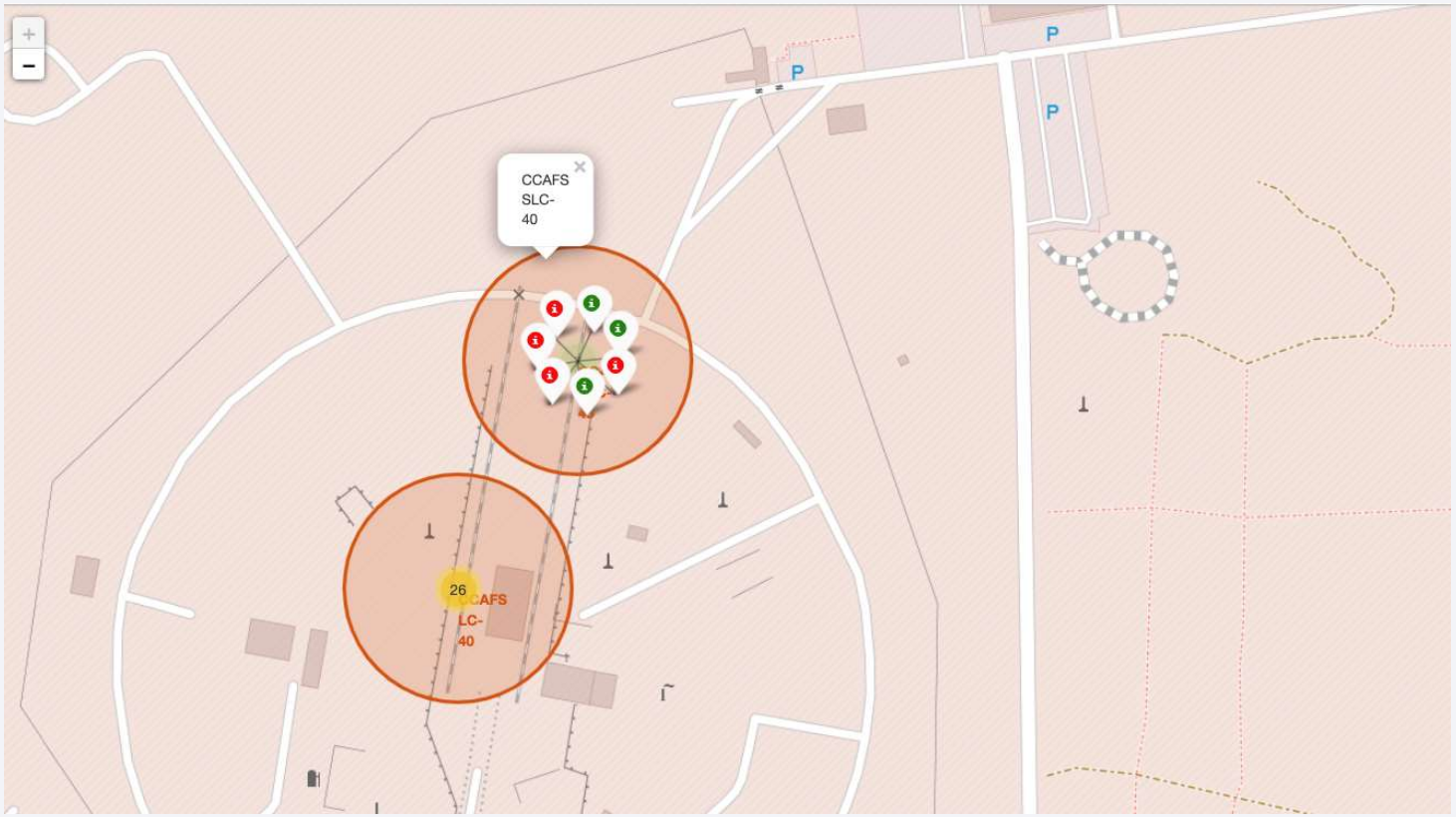
<Folium Map Screenshot 1>

- SpaceX launch sites are in USA at Florida and California Coastal Area.



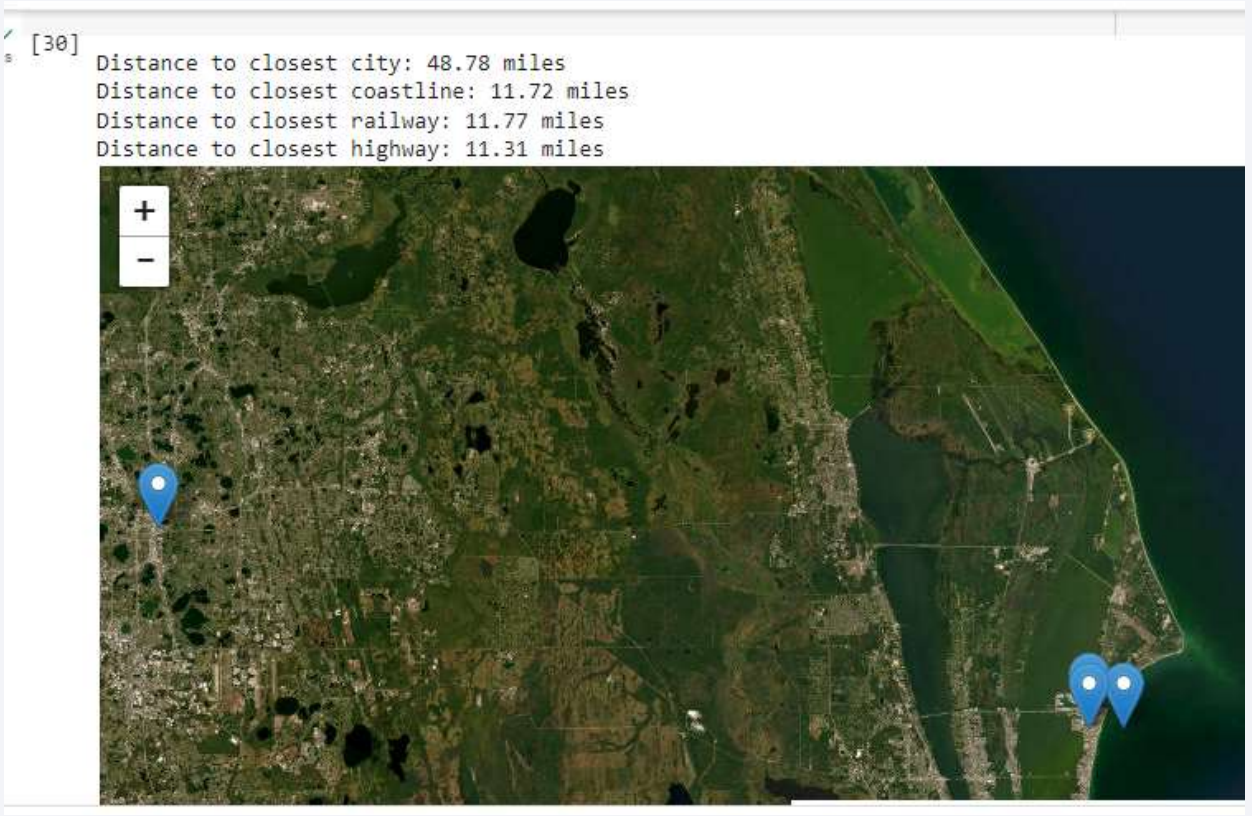
<Folium Map Screenshot 2>

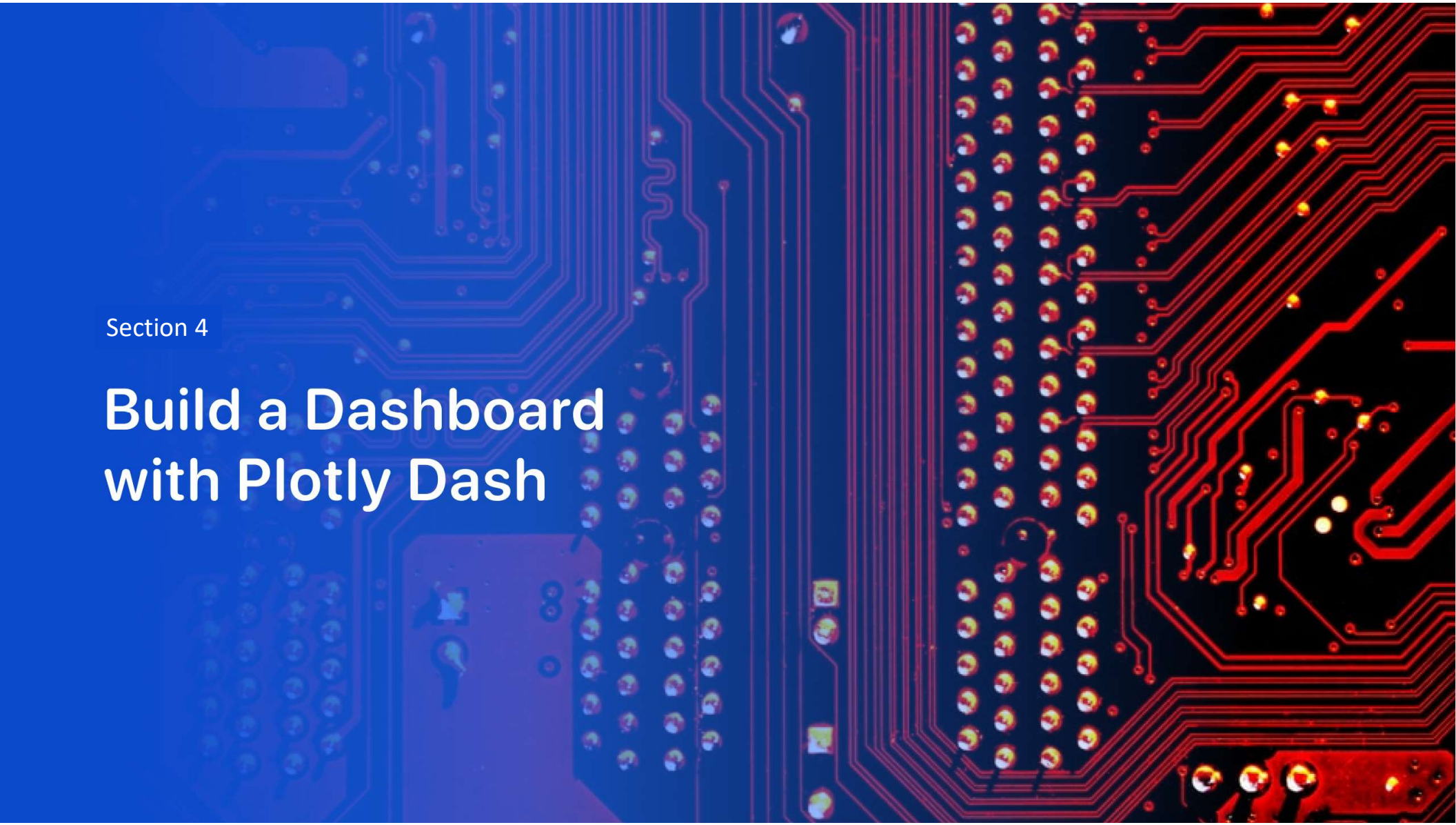
- Markers showing launch site in Florida : CCAFS SLC-40.



<Folium Map Screenshot 3>

- Shown on the map the markers for distance between the launch site and city, coastline, railway and the highway.
- The link is https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Folium_Interactive.ipynb



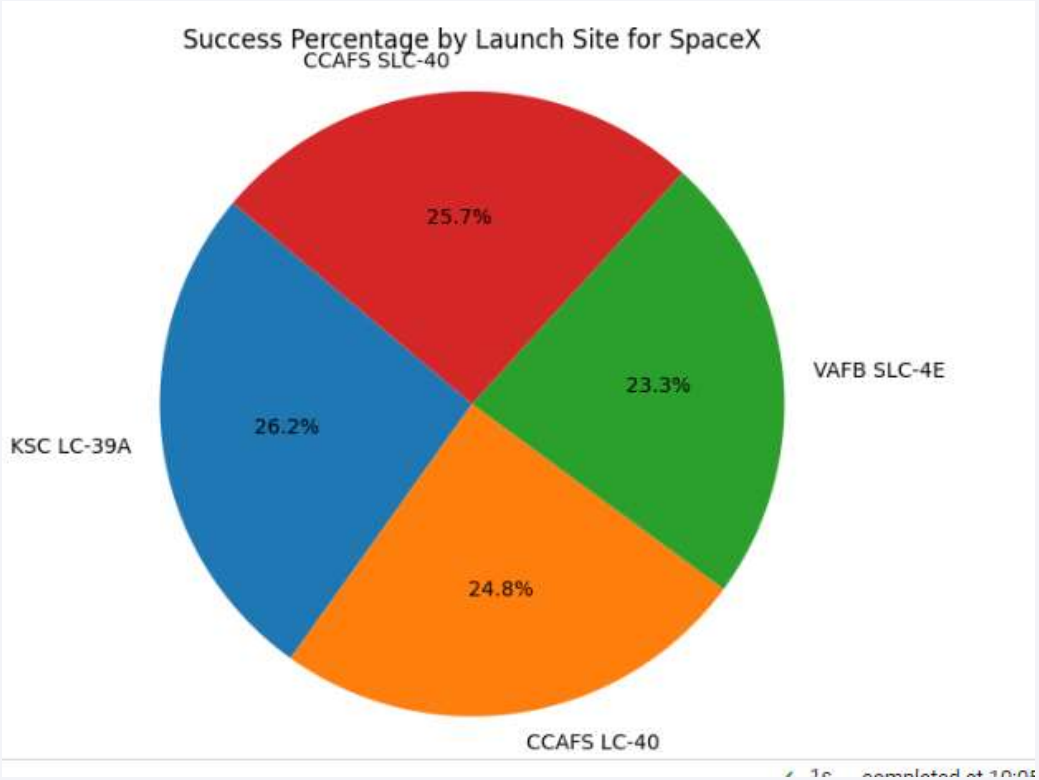


Section 4

Build a Dashboard with Plotly Dash

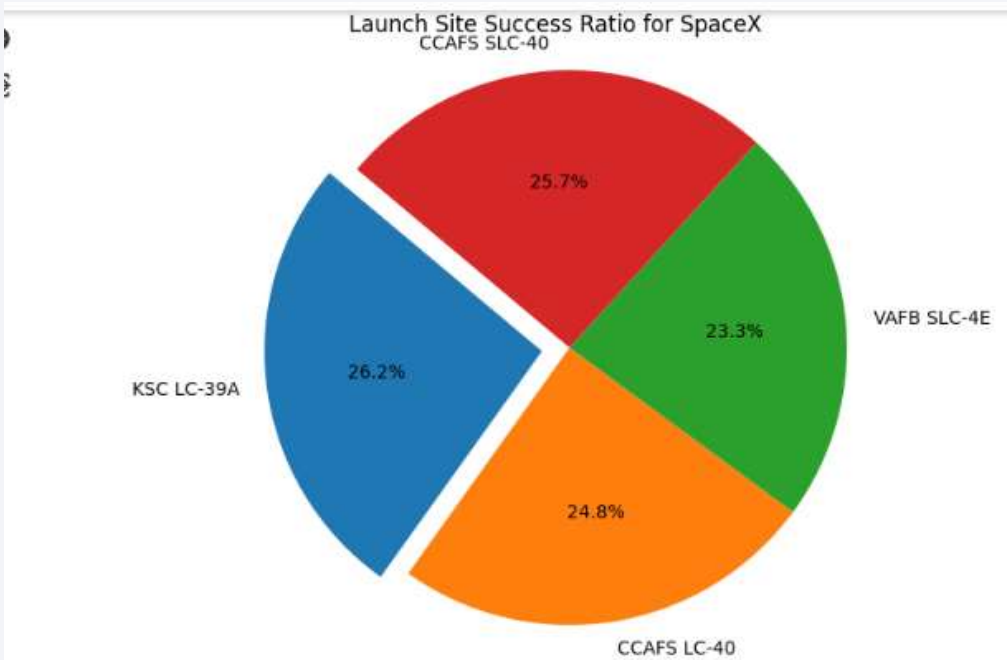
Pie chart showing the success percentage achieved by each launch site

- As per Pie chart KSC LC-39A has most success percentage achieved.
- The link is https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Misc.ipynb



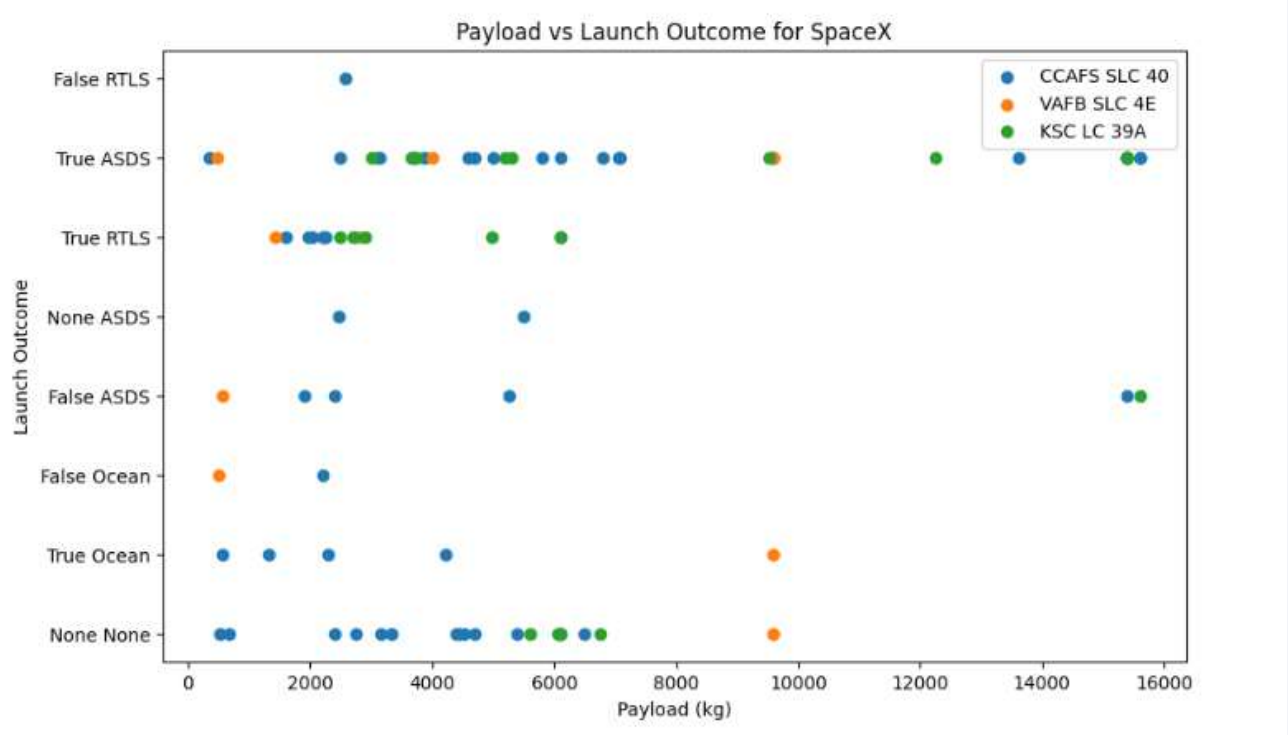
Pie chart showing the Launch site with the highest launch success ratio

- Launch site success ratio is highest for KSC LC-39A with success ratio 26.2% and failure rate with minimum value 10.0%.
- The link is https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Misc.ipynb



Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

- The Scatter plot showing the results for Payload and Launch Outcome.
- The link is https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Misc.ipynb



Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The classifier is the model with the highest classification accuracy

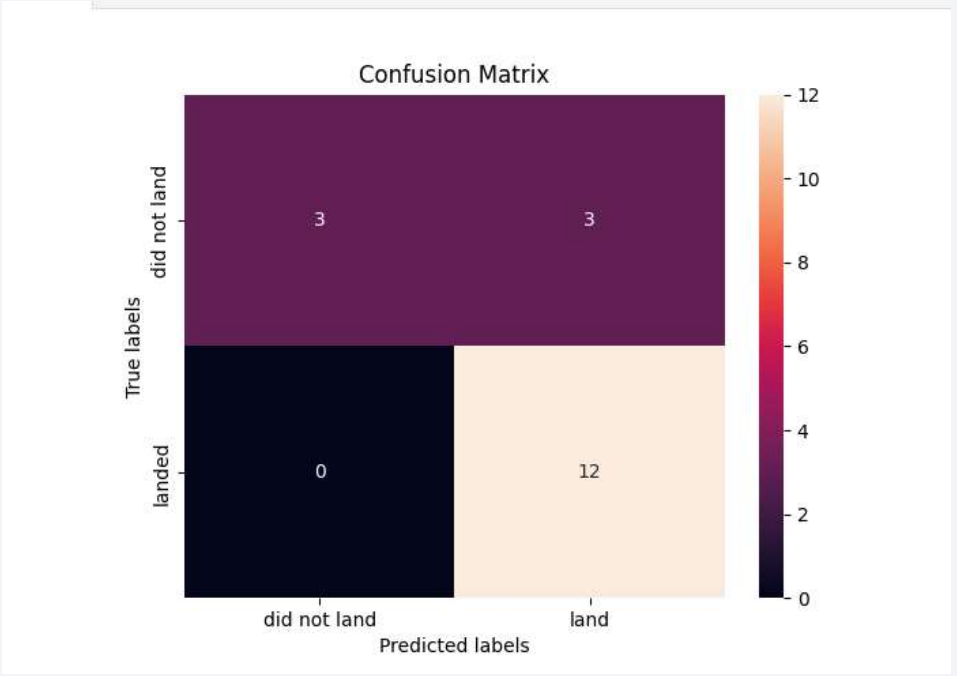
In [66]:

```
print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
print('Accuracy for K nearsdt neighbors method:', knn_cv.score(X_test, Y_test))
```

```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.9444444444444444
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.
- The link is https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Machine%20Learning%20Prediction.ipynb



Conclusions

We can conclude that:

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Appendix

- For miscellaneous program : https://github.com/prasadratnaparkhi/IBM--Data-Science-Capstone---SpaceX/blob/main/Misc.ipynb

Thank you!