IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Matteo Andrisani
22nd November 2023

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
  - With SpaceX advertising the the Falcon 9 rocket launches on it’s website, we want to compare its cost of 62 million dollars with the reuse of the first stage vs other providers which include costs of 165 million dollars each who cannot reuse the first stage. We use this info to see how other companies can bid against Space X for a rocket launch. We mainly want to see if we can predict if the first stage will land successfully using a machine learning pipeline.

- 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 need to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - One-hot encoding was applied to categorical variables.
- 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

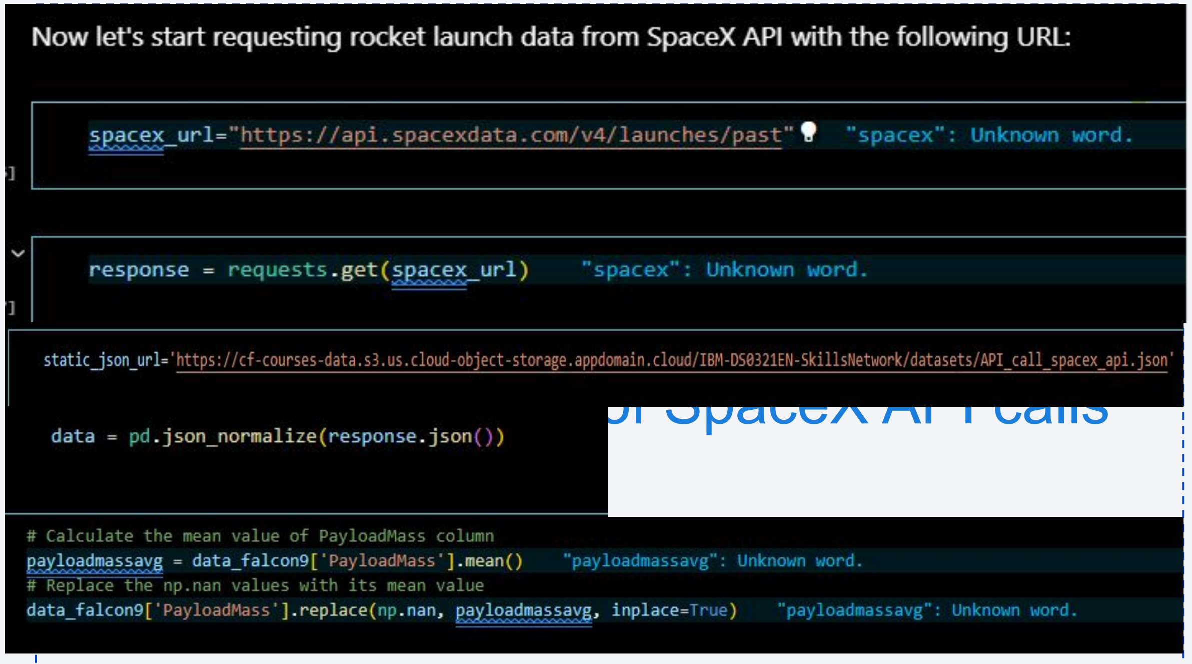
Data Collection

- Describe how data sets were collected.
    - Data was collected through the SpaceX API.
    - We then decoded the content as a Json file using the .json() function call and turn it into a pandas dataframe using .json_normalize().
    - Then we cleaned the data, ensure the data is complete, and fill the missing values where necessary.
    - Using Beautiful Soup, we performed web scraping from Wikipedia for Falcon 9 launch record.
    - The main takeaway of this 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

- Using the get request to the SpaceX API to collect datam cleaned the requested data, and performed some basic data wrangling and formatting.

- Link to the notebook: https://github.com/mandrisani/project/blob/main/SpaceX%20Data%20Collection.ipynb



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

```
spacex_url="https://api.spacexdata.com/v4/launches/past"    "spacex": Unknown word.
```

```
response = requests.get(spacex_url)    "spacex": Unknown word.
```

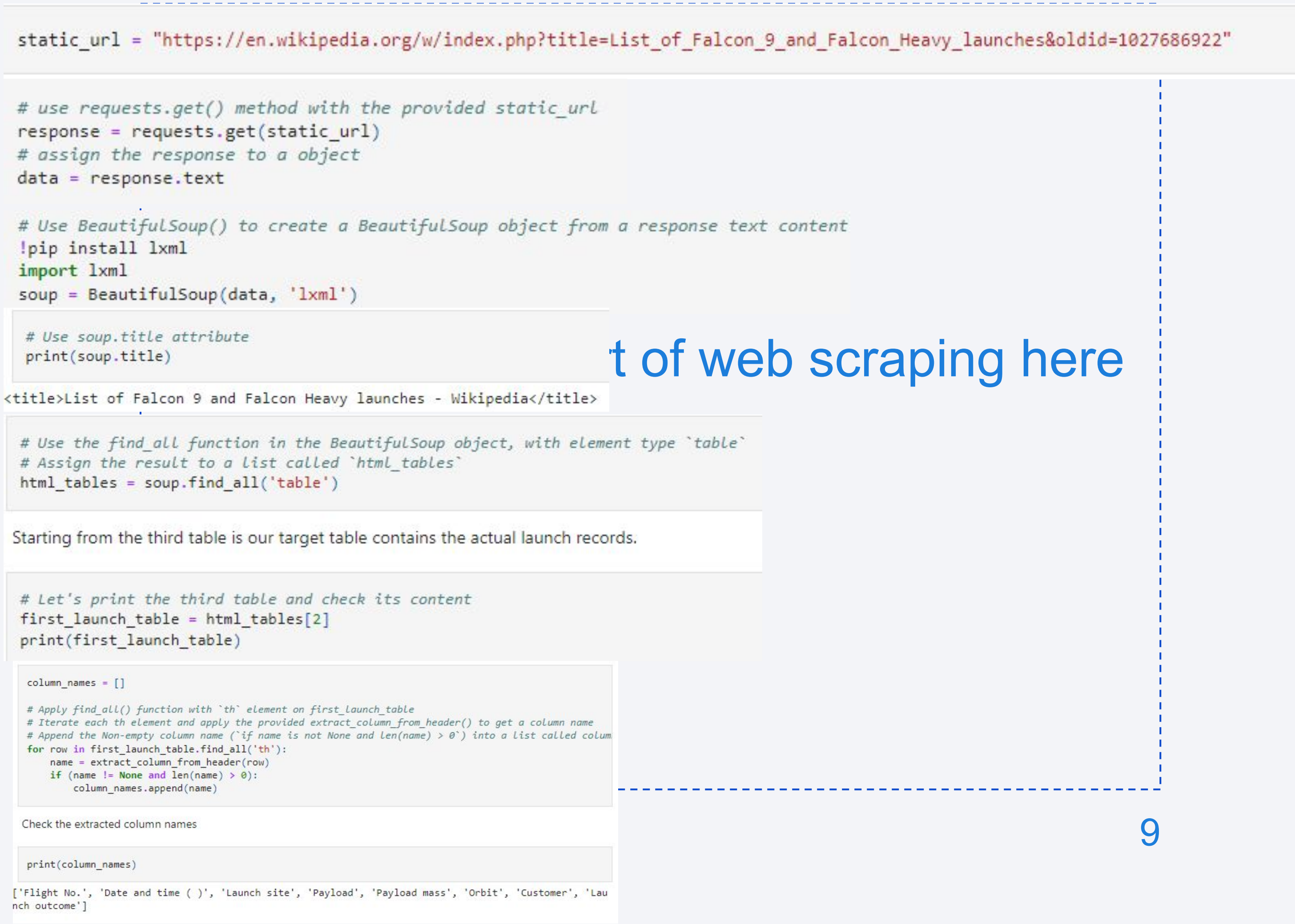
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

```
data = pd.json_normalize(response.json())
```

```
# Calculate the mean value of PayloadMass column
payloadmassavg = data_falcon9['PayloadMass'].mean()    "payloadmassavg": Unknown word.
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan, payloadmassavg, inplace=True)    "payloadmassavg": Unknown word.
```

Data Collection - Scraping

- Using BeautifulSoup, we applied web scraping to webscrape Falcon 9 launch records.
- We parsed the table and converted it into a pandas frame.
- The link to the notebook is https://github.com/mandrisani/project/blob/main/SpaceX%20Webscrapping.ipynb



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

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
!pip install lxml
import lxml
soup = BeautifulSoup(data, 'lxml')
```

```
# Use soup.title attribute
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

```
# 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)
```

```
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 colum
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if (name != None and len(name) > 0):
        column_names.append(name)
```

Check the extracted column names

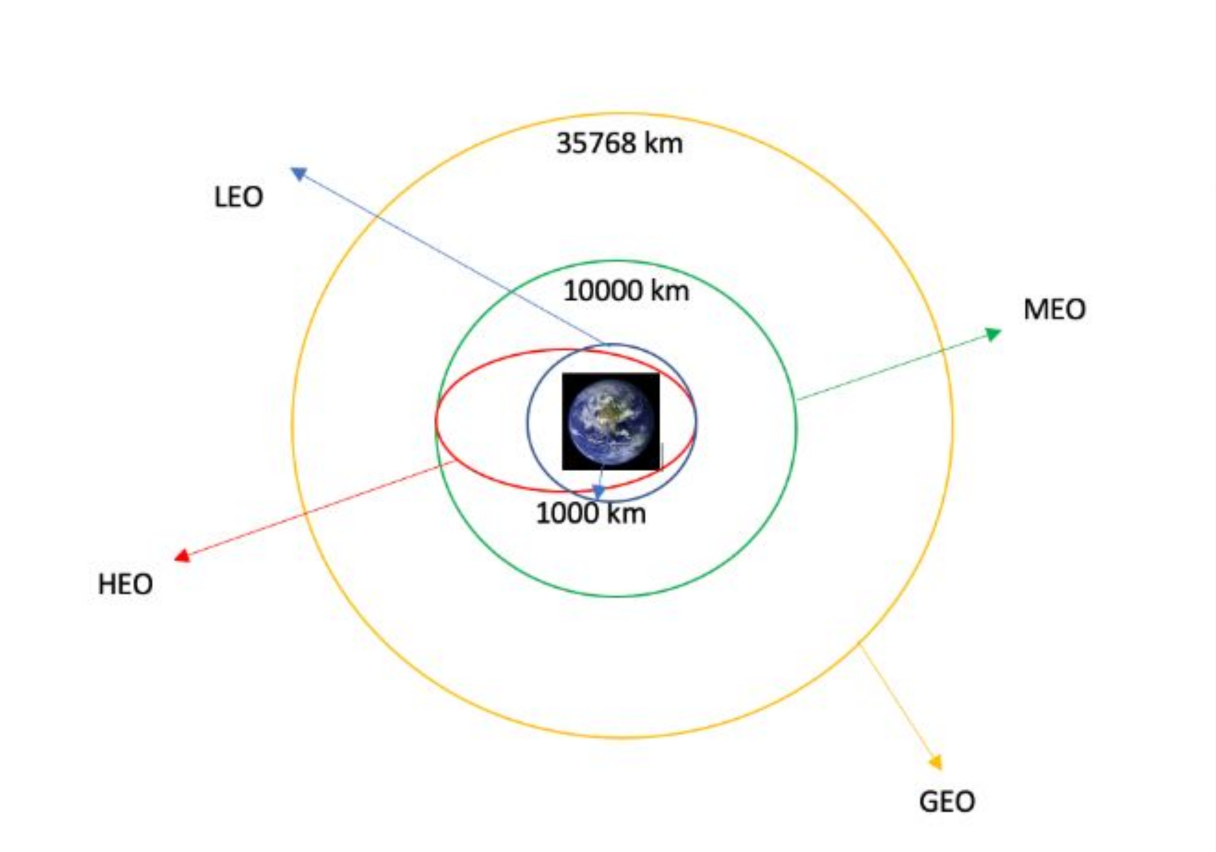
```
print(column_names)
```

```
['Flight No.', 'Date and time ( )', 'Launch site', 'Payload', 'Payload mass', 'Orbit', 'Customer', 'Lau
nch outcome']
```

t of web scraping here

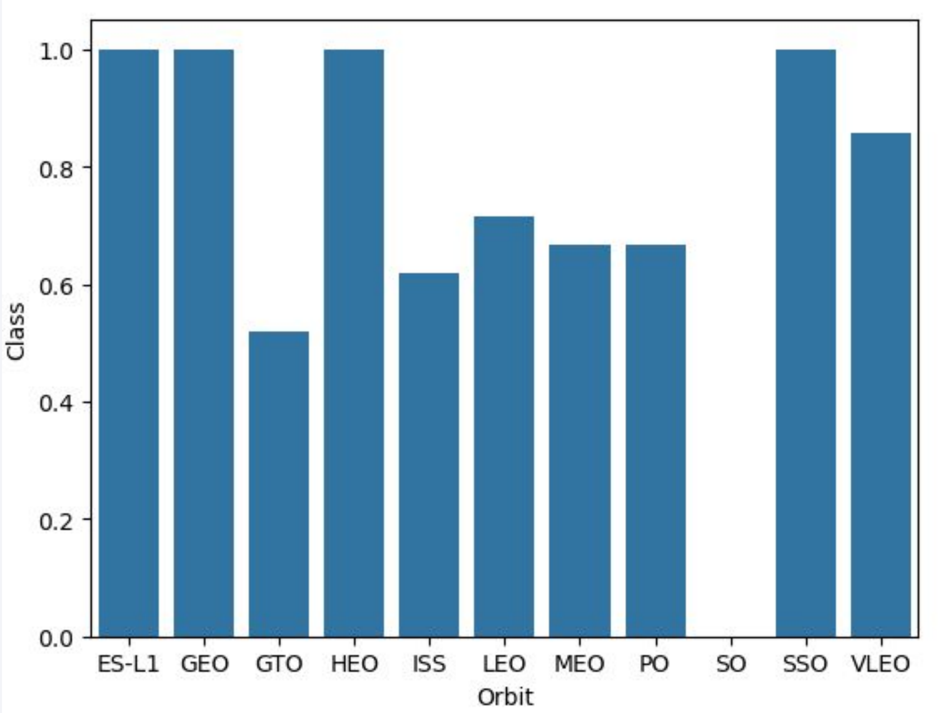
Data Wrangling

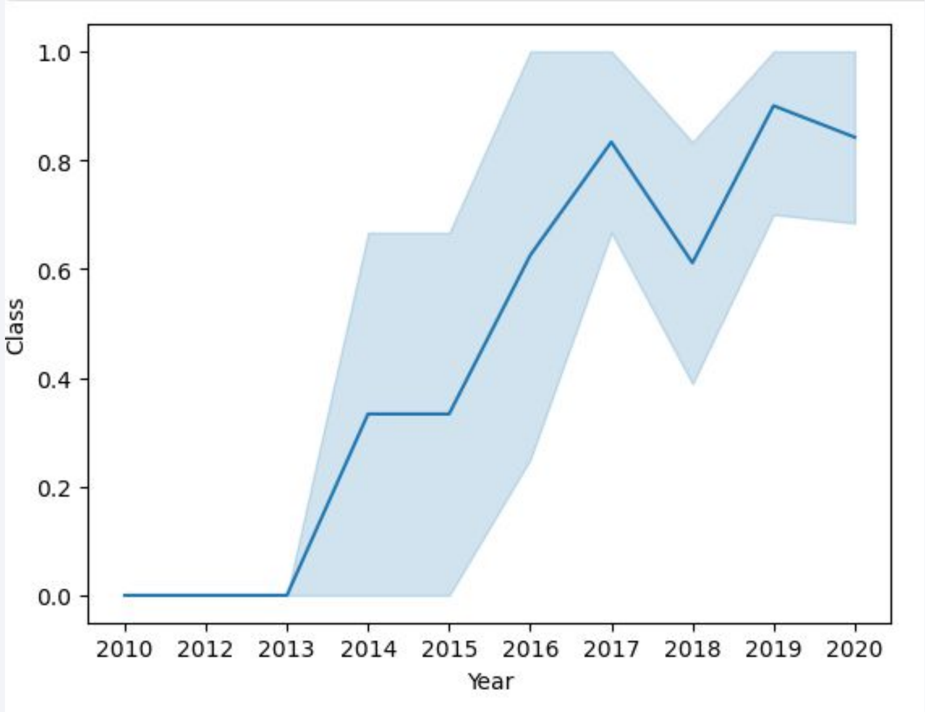
- Here we performed exploratory data analysis and determined the training labels.
- We also calculated the number of launches at each site and the number/occurance of each orbit.
- Then we created the landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is https://github.com/mandrisani/project/blob/main/SpaceX%20Data%20Wrangling.ipynb



EDA with Data Visualization

- Through visualization, we presented the relationship between flight number and launch site, payload and launch site success rate of each orbit type, flight number and orbit type, and the yearly trend of the launch success.





The link to the notebook is https://github.com/mandrisani/project/blob/main/SpaceX%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 and 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 ships, their booster version and launch site names.
- The link to the notebook is https://github.com/mandrisani/project/blob/main/SpaceX%20EDA%20with%20SQL.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 in the folium map.
- We assigned the feature launch outcomes (success or failure) to class 1 and 0 respectively.
- Using the colour-labeled marker clusters, we identified which launch sites have a relatively high success rate.
- We calculated the distances between a launch site to its proximities and answered some questions:
    - Are launch sites near railways, highways, and coastlines?
    - Do launch sites keep certain distance away from cities.

Build a Dashboard with Plotly Dash

- Built an interactive dashboard with Plotly Dash.
- Plotted pie charts showing the total launches by a certain site.
- Also graphed a scatter plot showing the relationship with Outcome and Payload Mass (Kg) for the different booster versions.
- The link to the notebook is https://github.com/mandrisani/project/blob/main/SpaceX%20Dashboard%20with%20Plotly%20Dash.py

Predictive Analysis (Classification)

- Loaded the data using numpy and pandas, transformed the data, and split the data into training and testing.
- Built different machine learning models and tune different hyperparameters using GridSearchCV.
- Used accuracy as the metric for the model, improved it by using feature engineering and algorithm tuning.
- Found the best performing classification model.
- The link to the notebook is https://github.com/mandrisani/project/blob/main/SpaceX%20Predictive%20Analysis.ipynb

Results

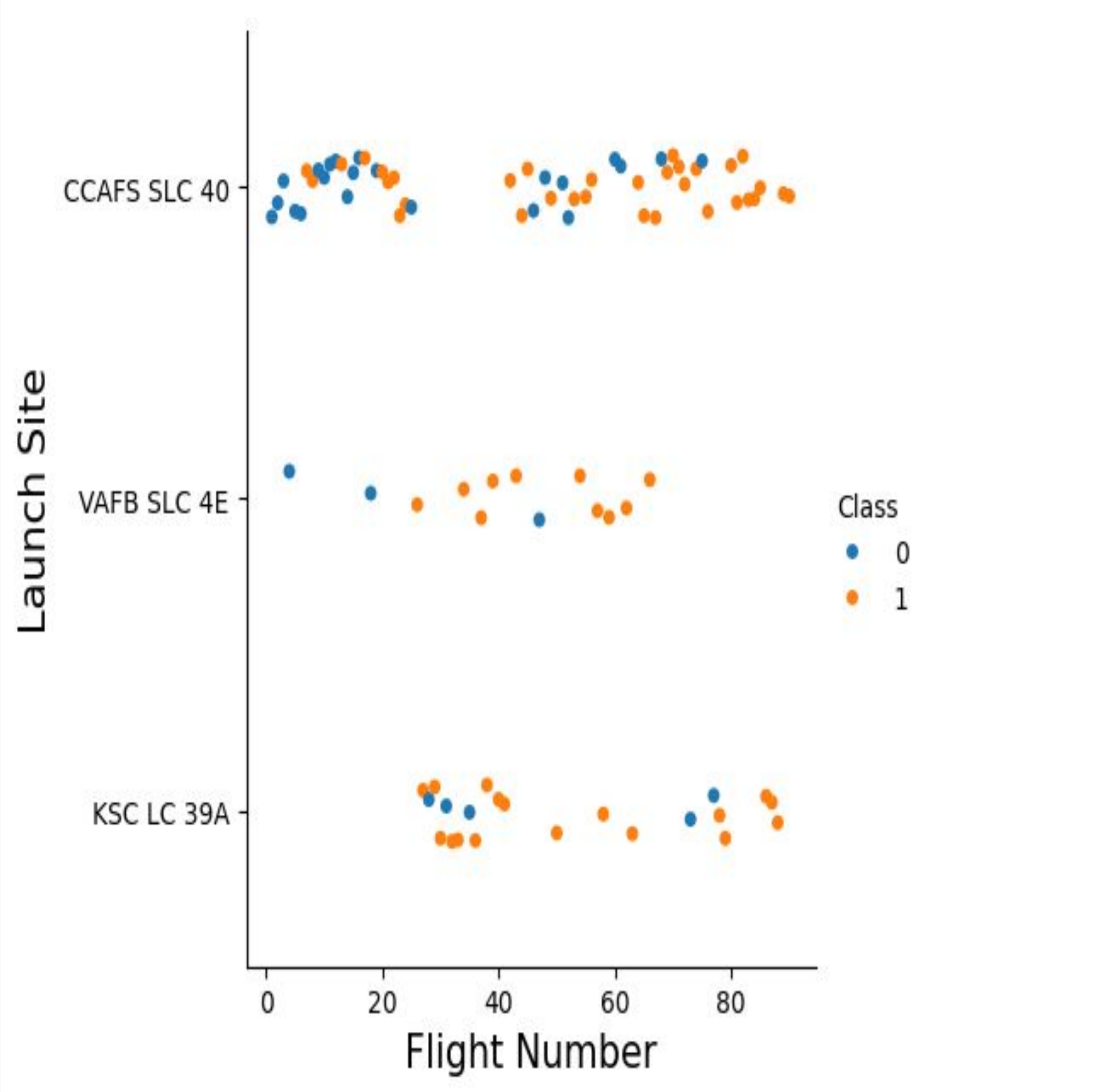
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

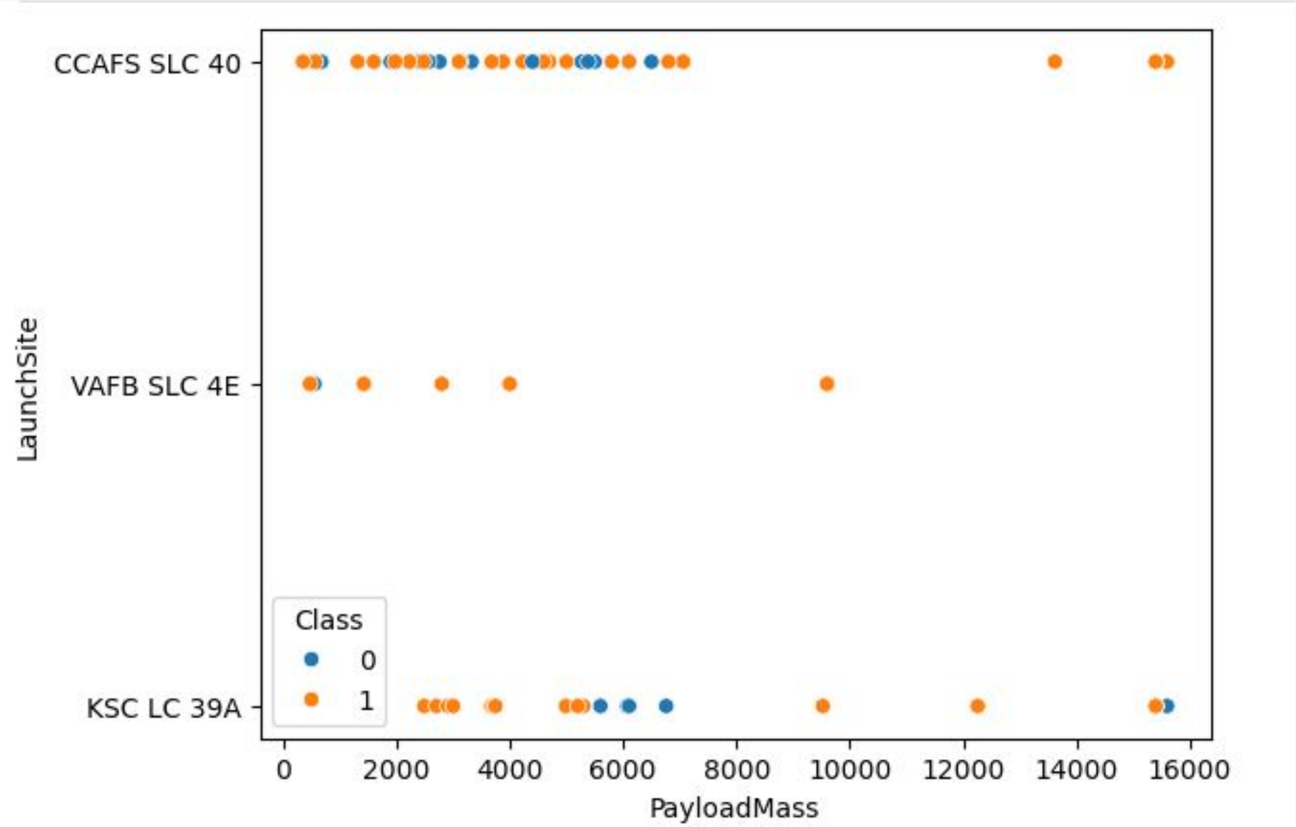
Flight Number vs. Launch Site

- We observed that the larger the flight amount at a launch site, the greater the success rate at a launch site.



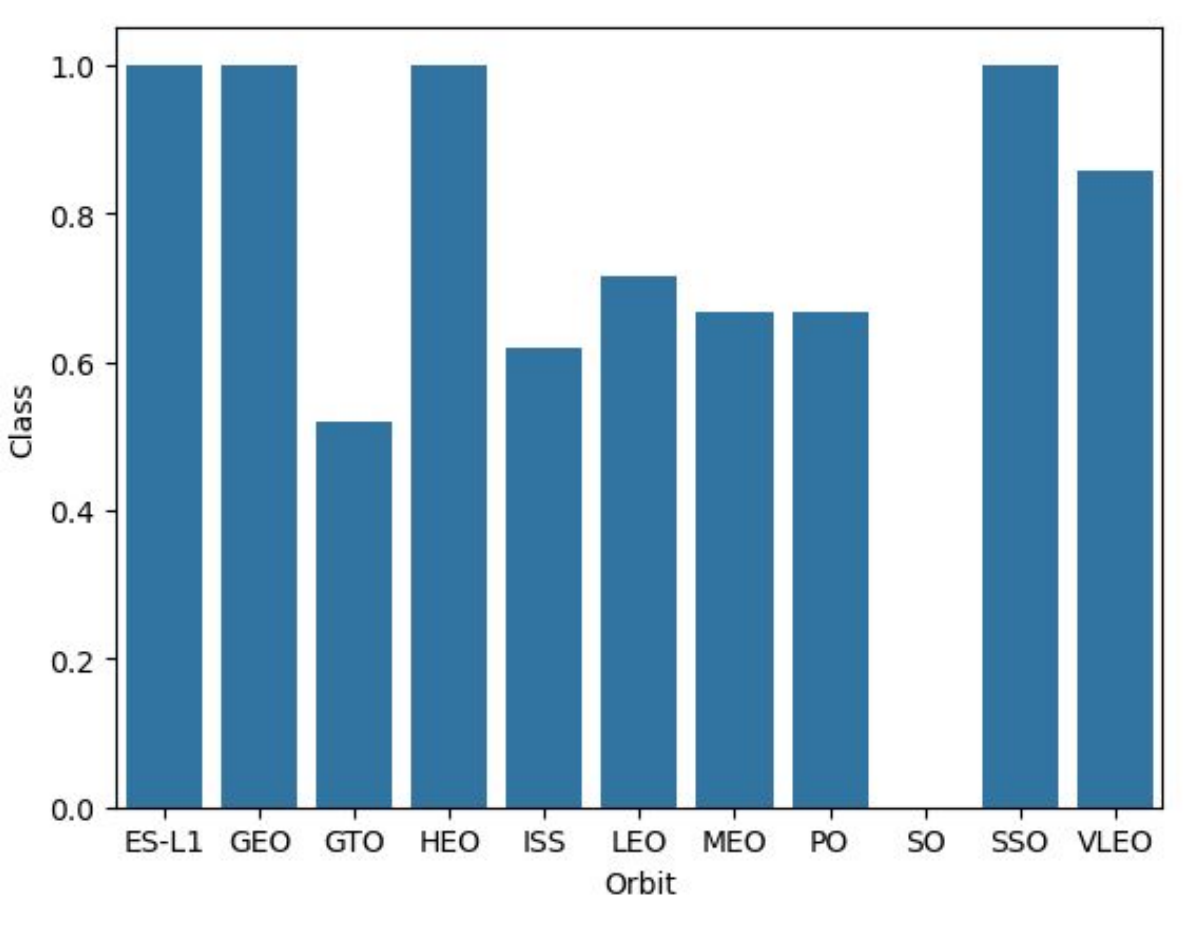
Payload vs. Launch Site

- The greater the payload mass for launch site CCAFS SLC 40, the higher the success rate for the rocket.



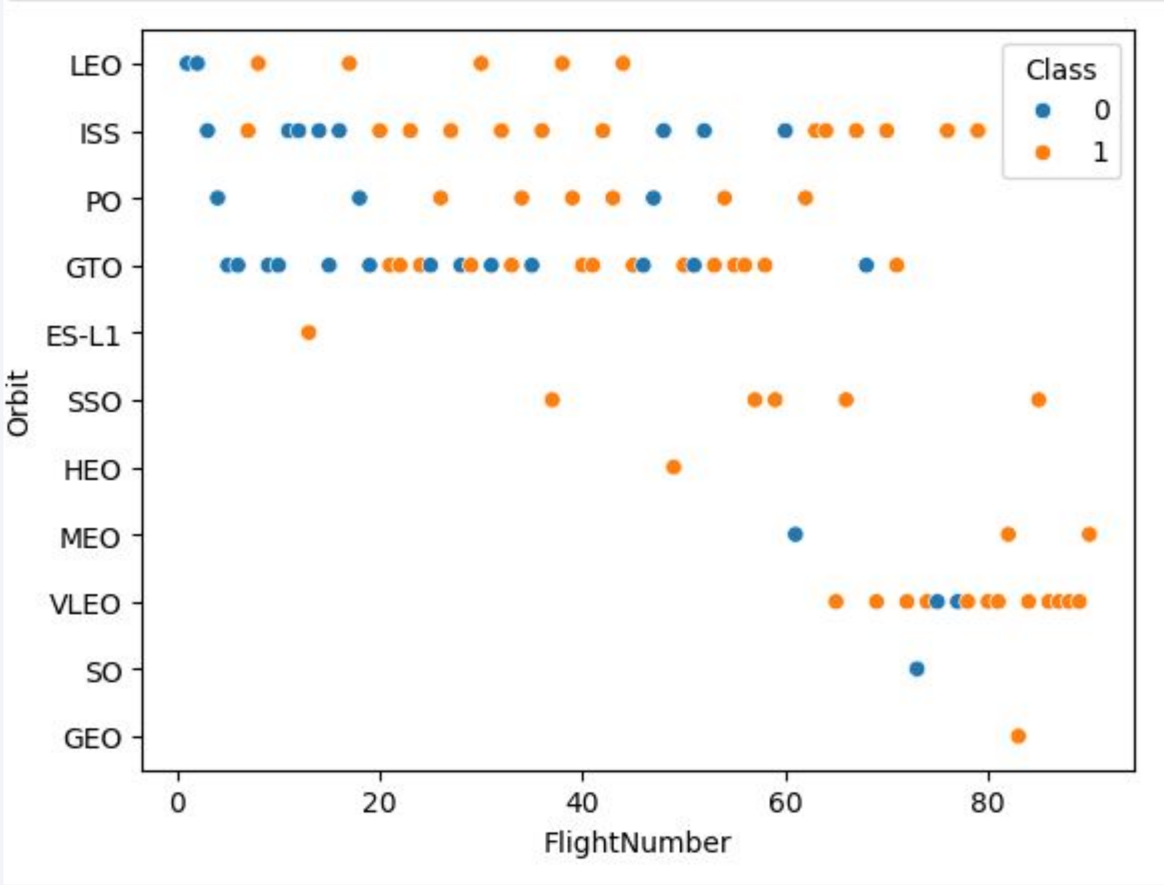
Success Rate vs. Orbit Type

- We can see that ES-L1, GEO, HEO, SSO, and VLEO had the most success.



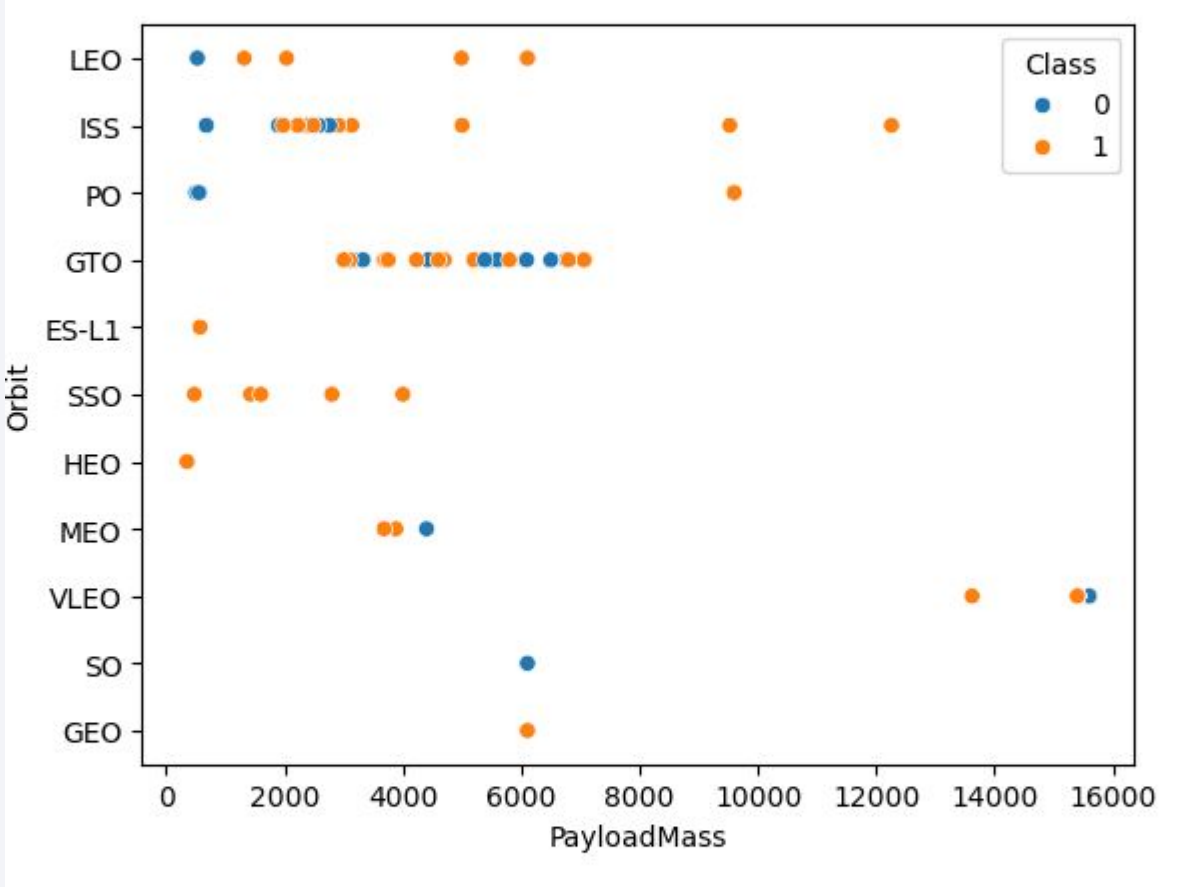
Flight Number vs. Orbit Type

- We found that in the LEO orbit, success is related to the number of flights whereas the GTO orbit has no relationship between flight number and the orbit.



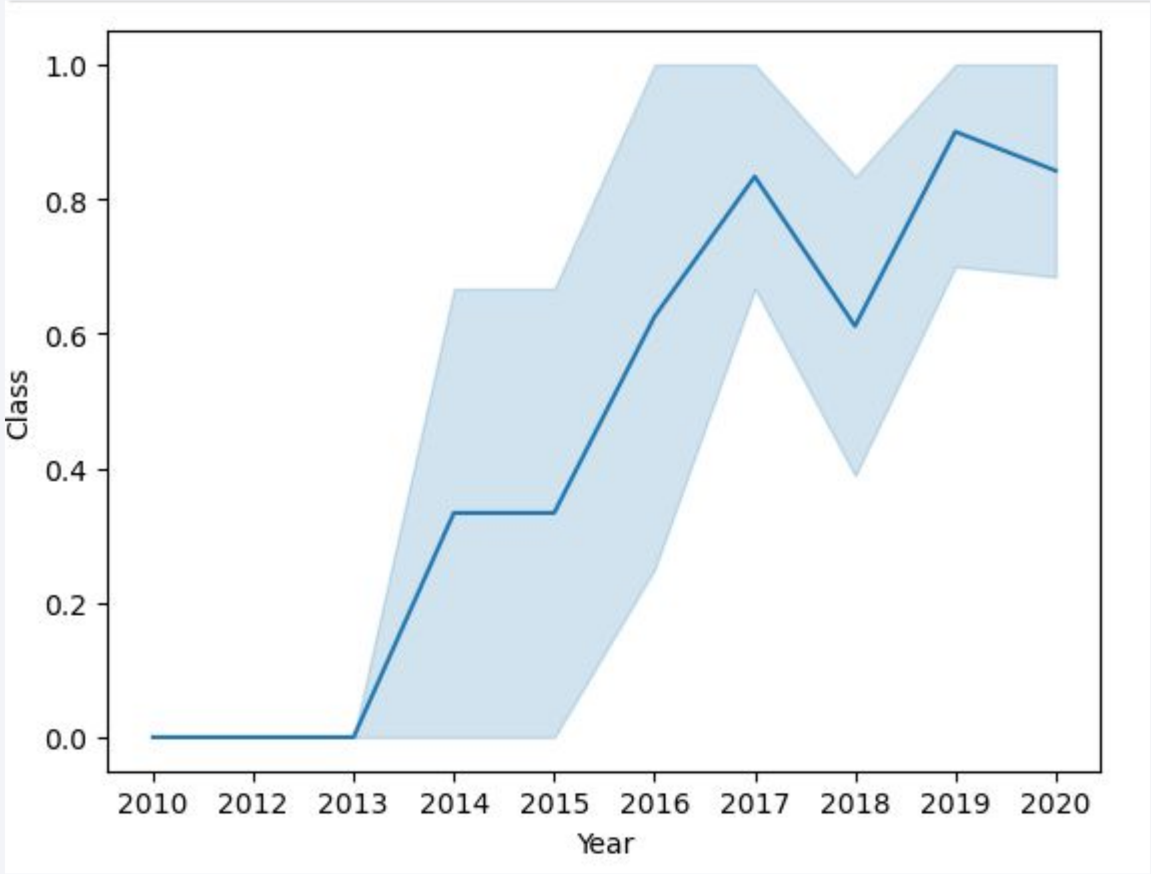
Payload vs. Orbit Type

- We have observed that with heavy payloads the successful landing are more for PO, LEO, and ISS orbits.



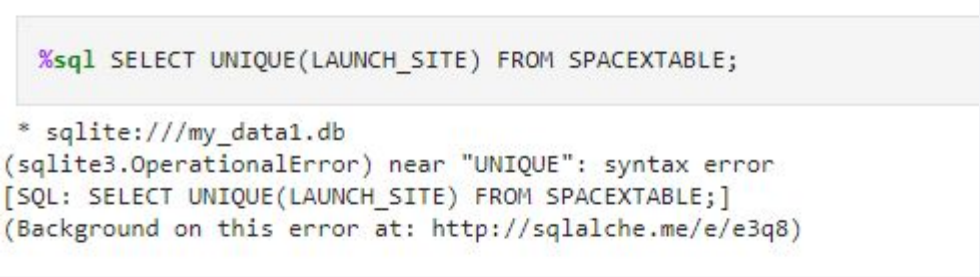
Launch Success Yearly Trend

- We have observed that the success rate since 2013 kept on increasing till 2020.



All Launch Site Names

- Used the keyword DISTINCT to show only unique launch sites from the SpaceX data.
- This did not work as expected.

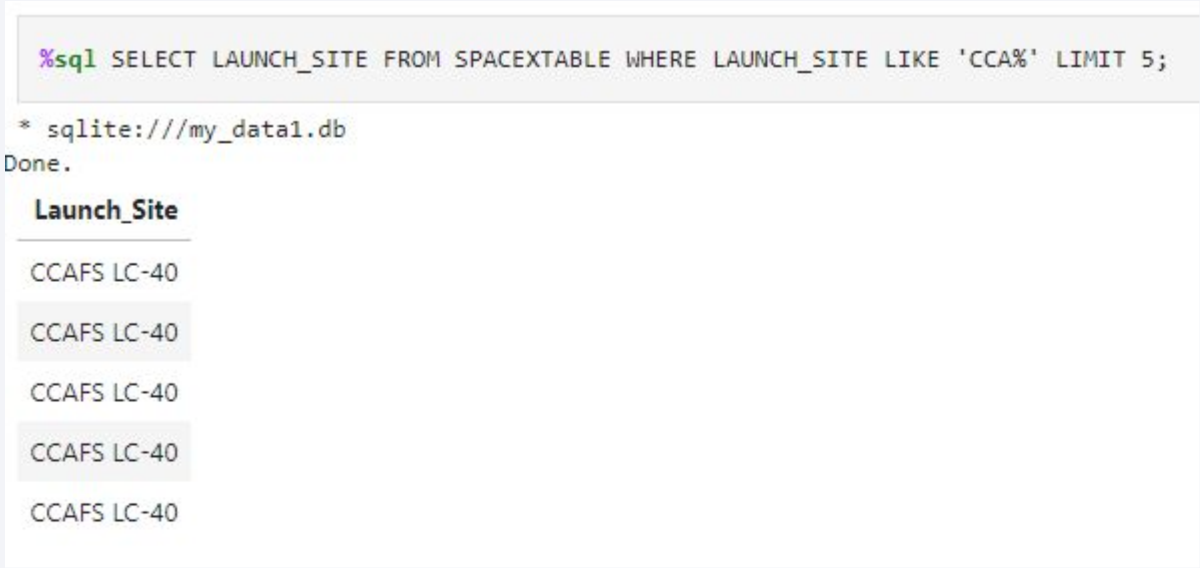


```
%sql SELECT UNIQUE(LAUNCH_SITE) FROM SPACEXTABLE;
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) near "UNIQUE": syntax error
[SQL: SELECT UNIQUE(LAUNCH_SITE) FROM SPACEXTABLE;]
(Background on this error at: http://sqlalche.me/e/e3q8)
```

Launch Site Names Begin with 'CCA'

- We used the following to display 5 records where launch sites begin with ‘CCA’.
- The results were not what we expected.



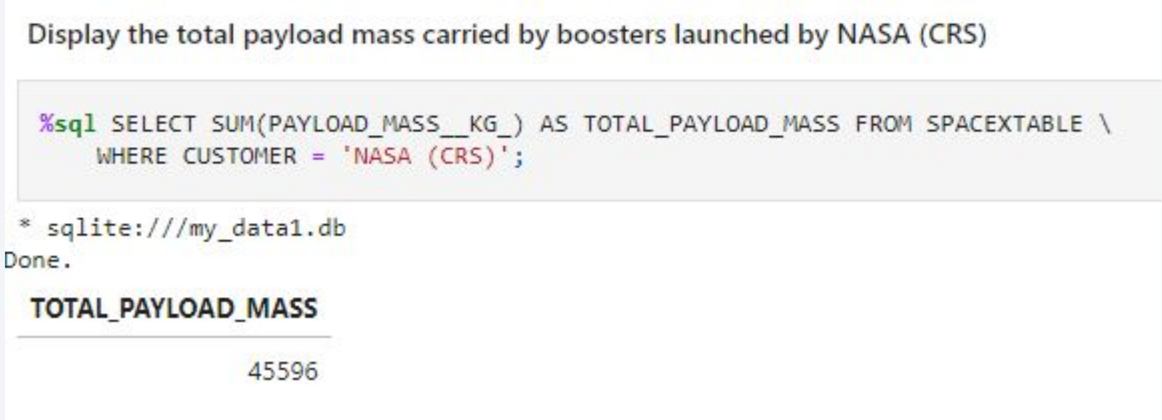
```
%sql SELECT LAUNCH_SITE FROM SPACEXTABLE WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- We found that 45596 was the calculated total payload carried by boosters from NASA.



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

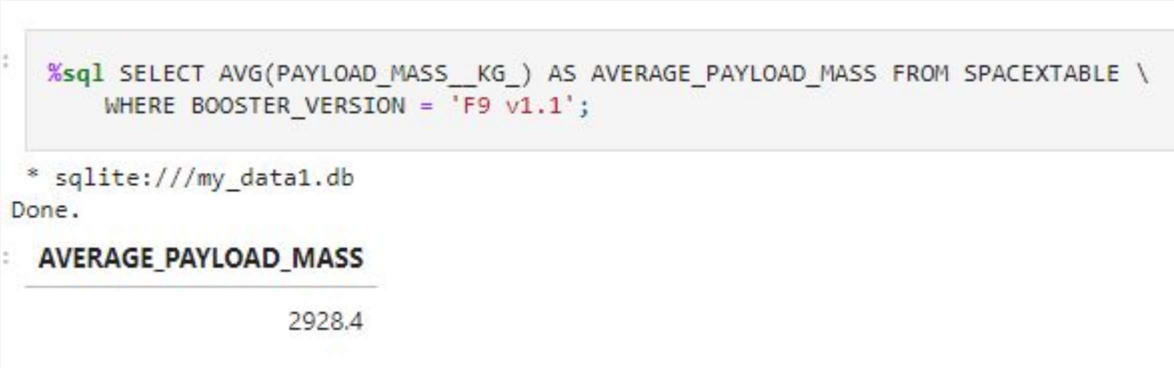
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS FROM SPACEXTABLE \
    WHERE CUSTOMER = 'NASA (CRS)';
```

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

TOTAL_PAYLOAD_MASS
45596

Average Payload Mass by F9 v1.1

- We found that 2928.4 was the calculated average payload mass carried by booster version F9 v1.1.



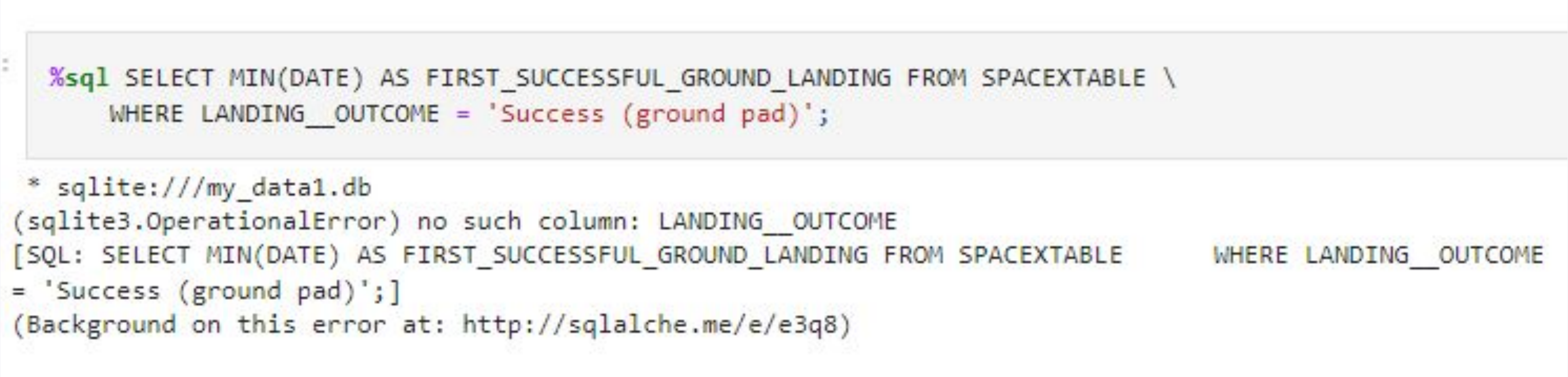
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVERAGE_PAYLOAD_MASS FROM SPACEXTABLE \
    WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

AVERAGE_PAYLOAD_MASS
2928.4

First Successful Ground Landing Date

- We unfortunately did not get the results we were looking for and find the date of the first successful landing.

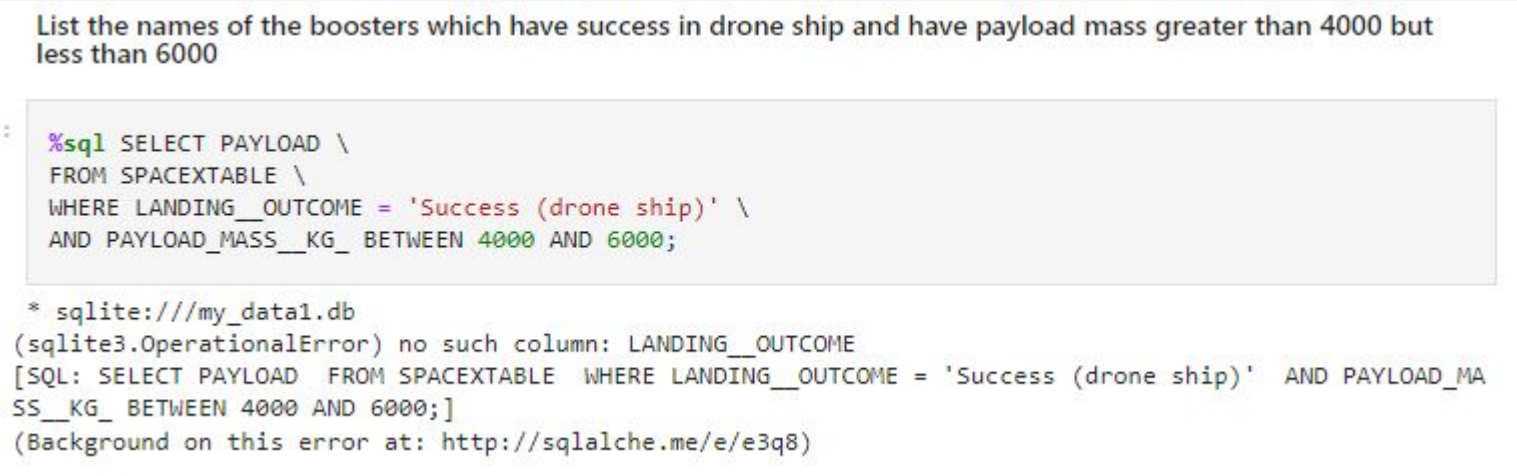


```
%sql SELECT MIN(DATE) AS FIRST_SUCCESSFUL_GROUND_LANDING FROM SPACEXTABLE \
    WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: LANDING__OUTCOME
[SQL: SELECT MIN(DATE) AS FIRST_SUCCESSFUL_GROUND_LANDING FROM SPACEXTABLE      WHERE LANDING__OUTCOME
= 'Success (ground pad)';]
(Background on this error at: http://sqlalche.me/e/e3q8)
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- We tried to use the WHERE clause to filter for boosters which have successfully landed on drone ship and apply the AND condition to determine successful landing with payload mass greater than 4000 but less than 6000.
- Unfortunately we did not get the results we were looking for.



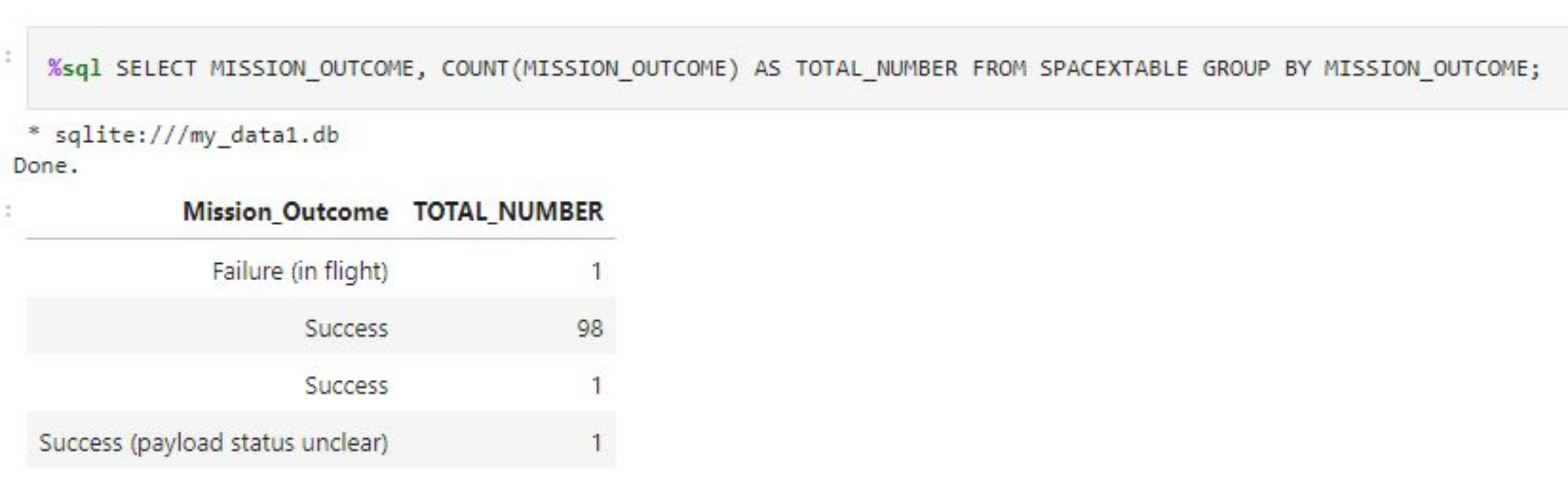
List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql SELECT PAYLOAD \
FROM SPACEXTABLE \
WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: LANDING__OUTCOME
[SQL: SELECT PAYLOAD  FROM SPACEXTABLE  WHERE LANDING__OUTCOME = 'Success (drone ship)'  AND PAYLOAD_MA
SS__KG_ BETWEEN 4000 AND 6000;]
(Background on this error at: http://sqlalche.me/e/e3q8)
```

Total Number of Successful and Failure Mission Outcomes

- We selected mission outcome to sort for total number of success and failures.



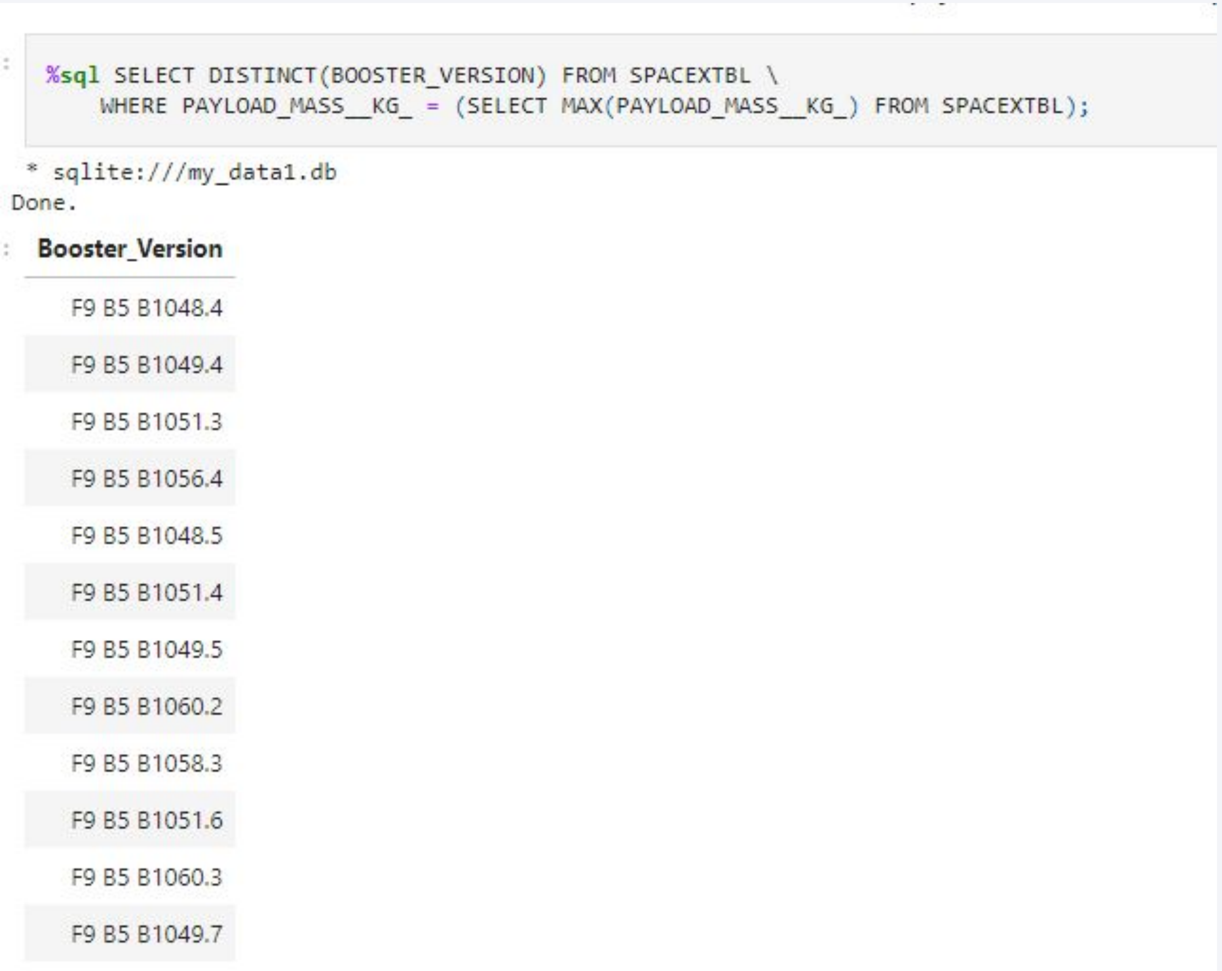
```
%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL_NUMBER FROM SPACEXTABLE GROUP BY MISSION_OUTCOME;
```

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

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

Boosters Carried Maximum Payload

- We determined the booster that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.



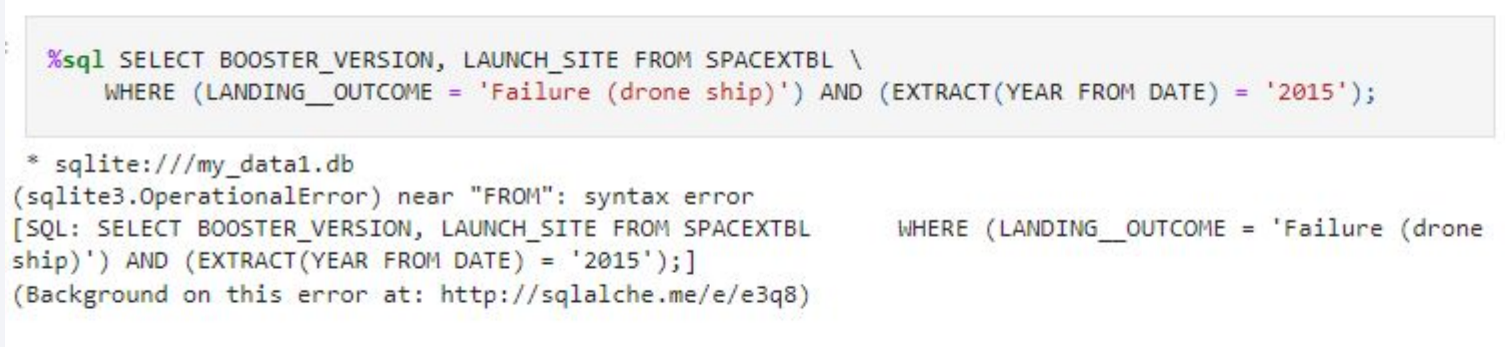
```
%sql SELECT DISTINCT(BOOSTER_VERSION) FROM SPACEXTBL \
    WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

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

- We tried to use a combination of WHERE and AND conditions to filter for failed landing outcomes in drone ships, their booster versions, and launch site names for year 2015.
- Unfortunately, this did not work as expected.

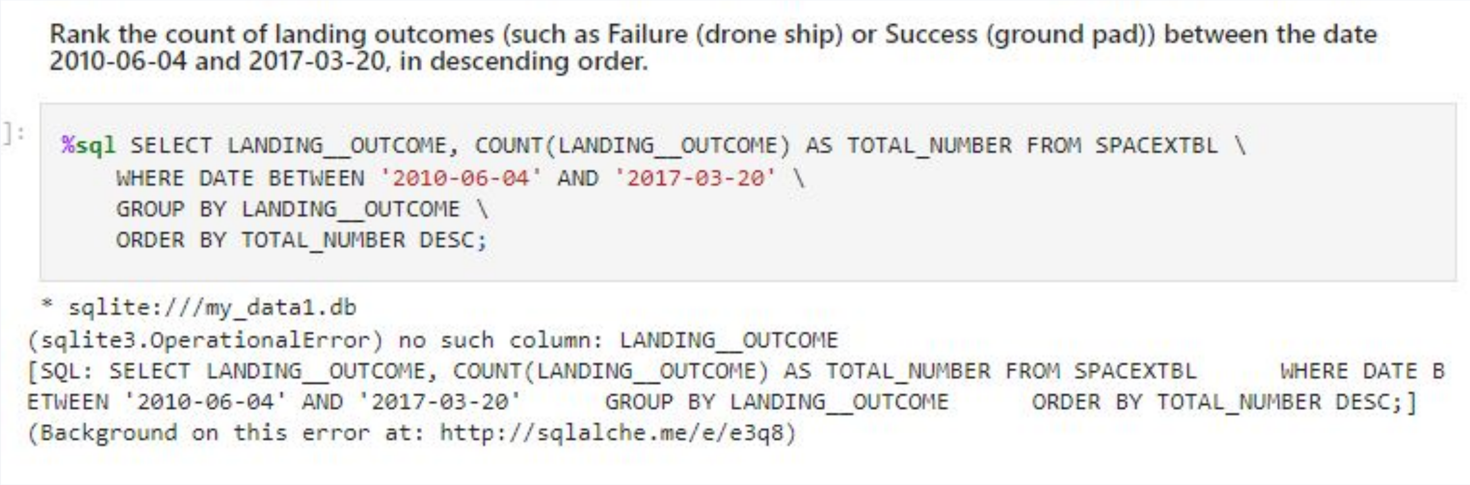


```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL \
    WHERE (LANDING__OUTCOME = 'Failure (drone ship)') AND (EXTRACT(YEAR FROM DATE) = '2015');
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) near "FROM": syntax error
[SQL: SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL      WHERE (LANDING__OUTCOME = 'Failure (drone
ship)') AND (EXTRACT(YEAR FROM DATE) = '2015');]
(Background on this error at: http://sqlalche.me/e/e3q8)
```

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

- We welected landing outcomes and the count of landing outcomes from the data and used WHERE clause to filter for landing outcomes between 2010-06-04 to 2017-03-20.
- We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcomes in descending order.
- Unfortunately this did not work as expected.



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(LANDING__OUTCOME) AS TOTAL_NUMBER FROM SPACEXTBL \
    WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
    GROUP BY LANDING__OUTCOME \
    ORDER BY TOTAL_NUMBER DESC;
```

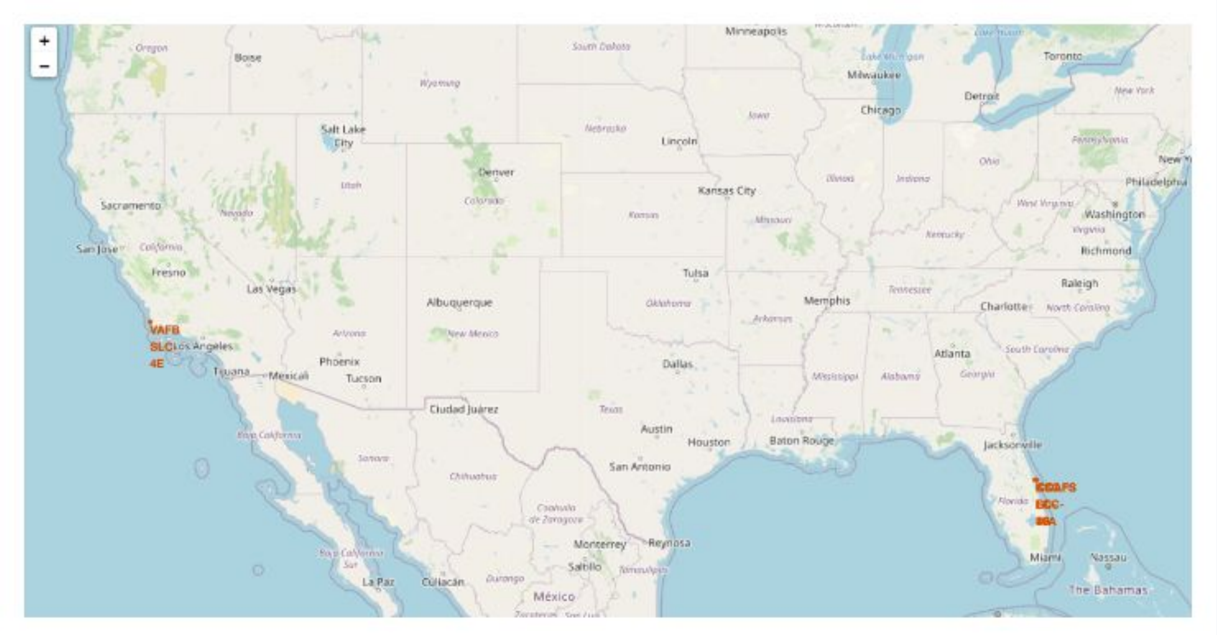
```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: LANDING__OUTCOME
[SQL: SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) AS TOTAL_NUMBER FROM SPACEXTBL      WHERE DATE B
ETWEEN '2010-06-04' AND '2017-03-20'      GROUP BY LANDING__OUTCOME      ORDER BY TOTAL_NUMBER DESC;]
(Background on this error at: http://sqlalche.me/e/e3q8)
```

Section 3

Launch Sites Proximities Analysis

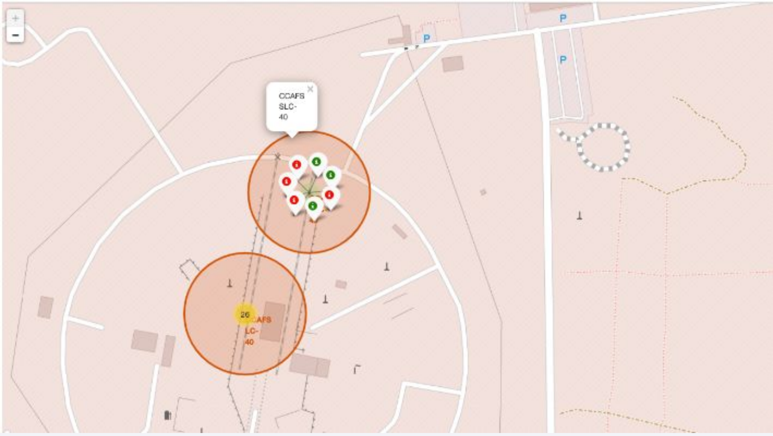
All launch sites global map markers

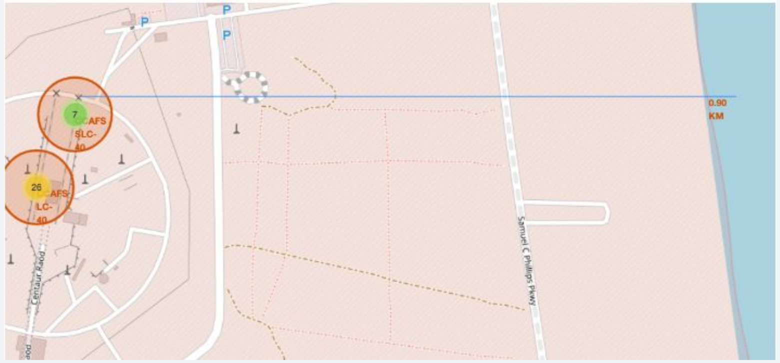
- We can see that the SpaceX launch sites are in the USA coasts, Florida and California.



Markers showing launch sites with colour labels

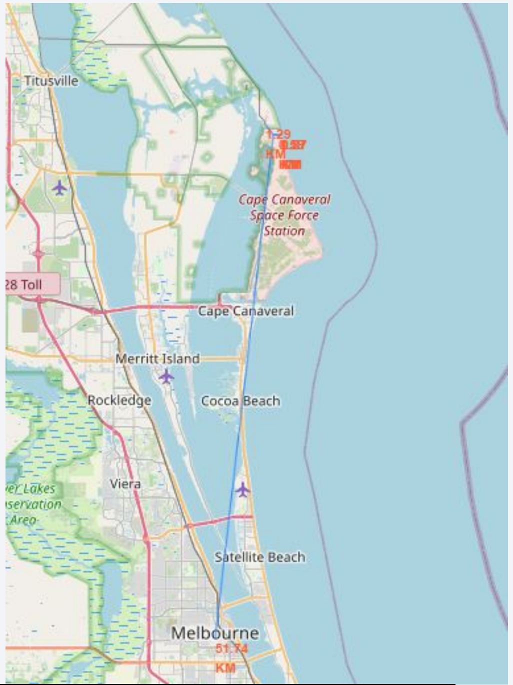
- Green Marker shows successful launches and Red Markers show Failures.

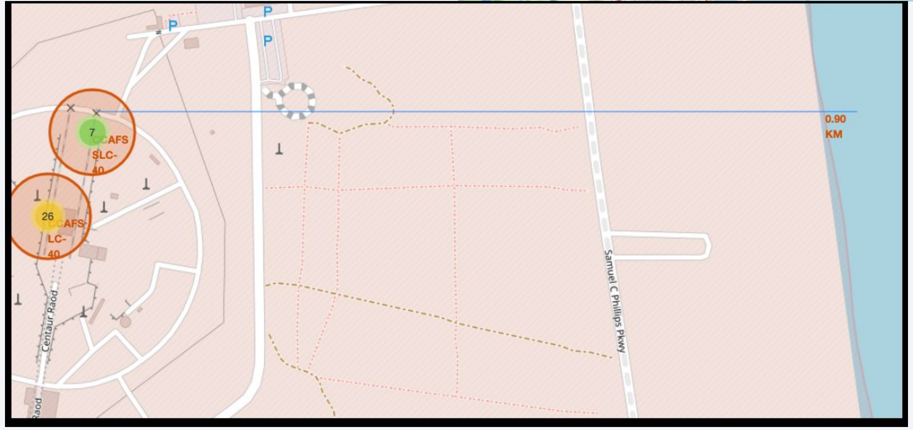




Launch Site distance to landmarks

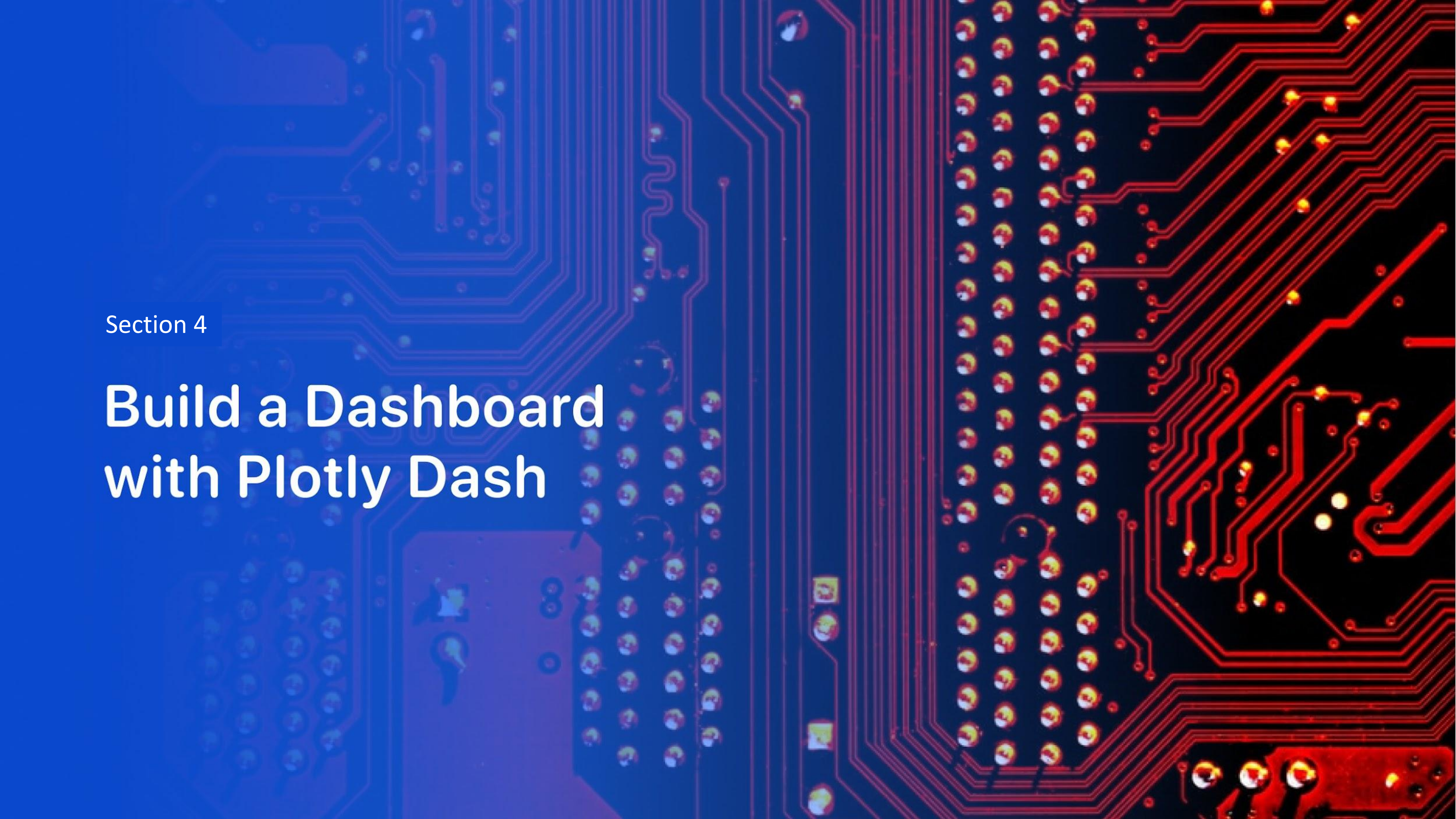
- No launch sites close to railways.
- No launch sites close to highways.
- Launch sites are close to coastlines.
- The launch sites do keep a certain distance away from cities.





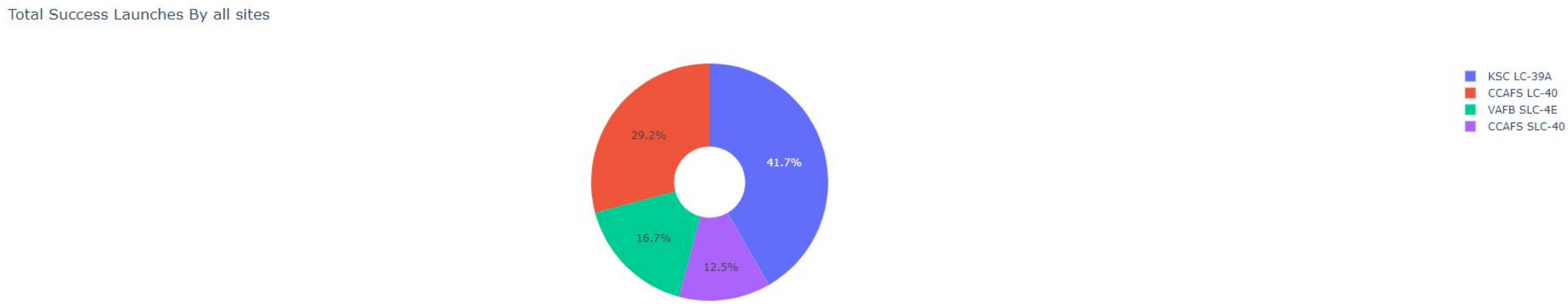
Section 4

Build a Dashboard with Plotly Dash



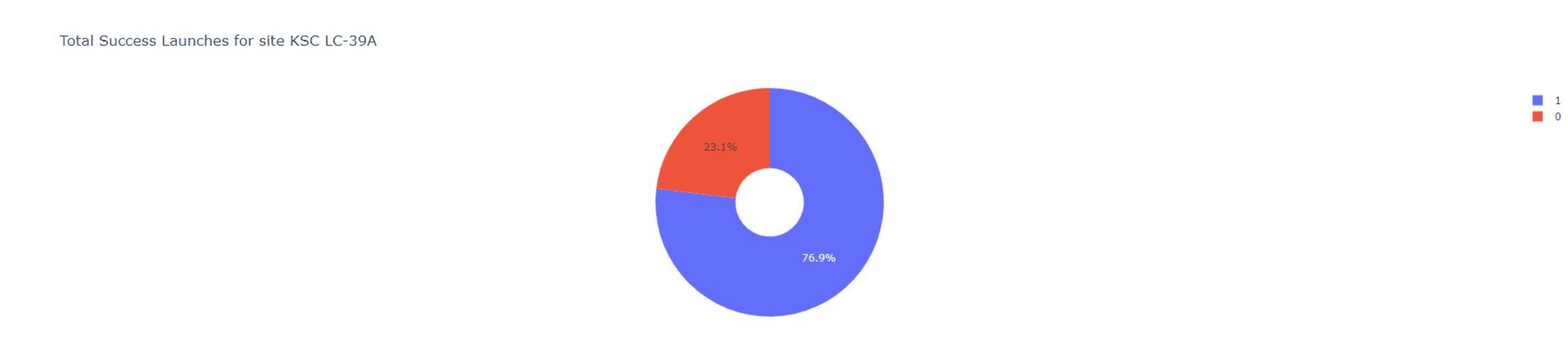
Pie chart showing the success percentage achieved by each launch site

- We can see that KSC LC-39A had the most successful launches from all the sites.



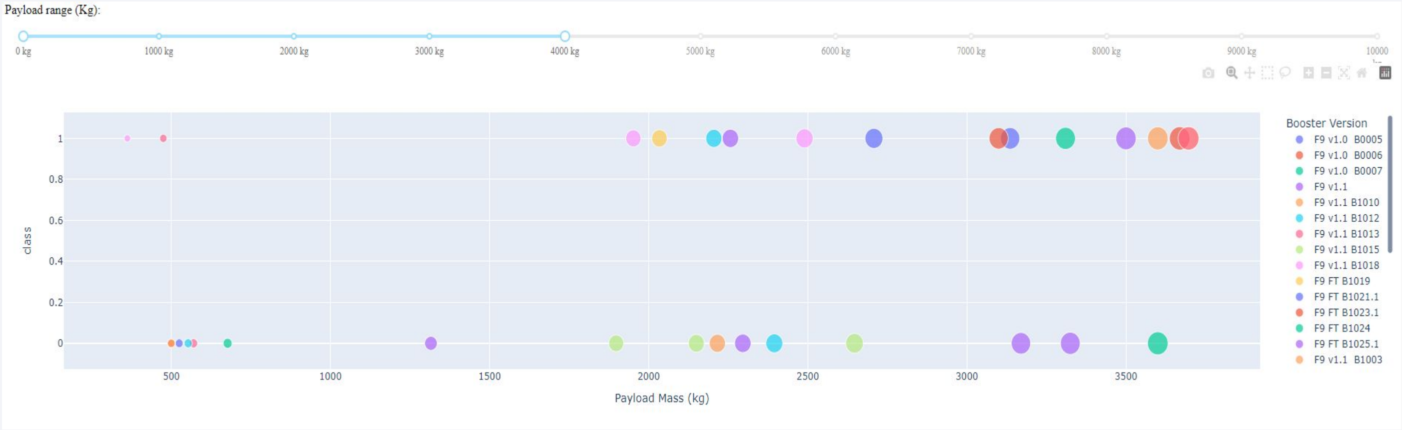
Pie chart showing the Launch site with the highest launch success ratio

- KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate.

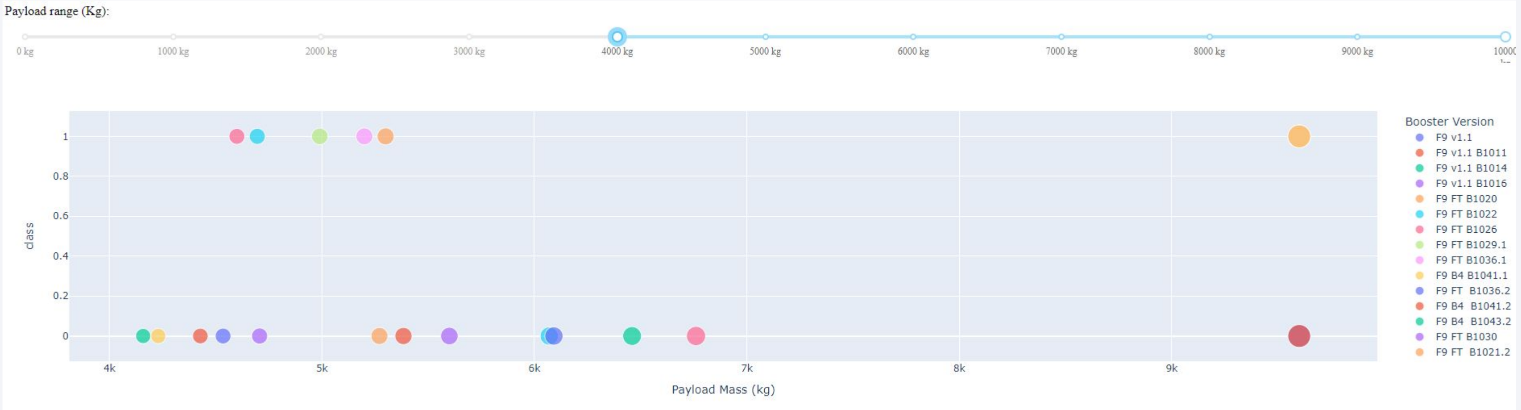


Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

- We can see the success rates for low weighted payloads is higher than the heavy weighted payloads.



Low Weighted Payload 0kg - 4000kg



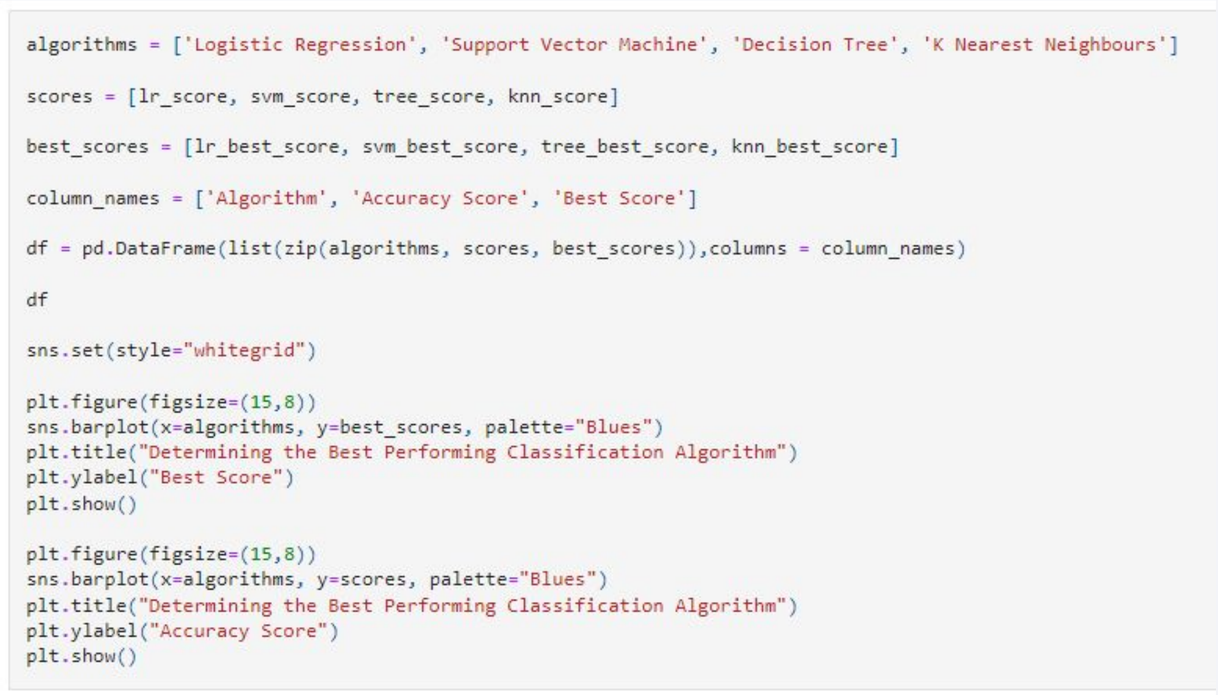
Heavy Weighted Payload 4000kg - 10000kg

Section 5

Predictive Analysis (Classification)

Classification Accuracy

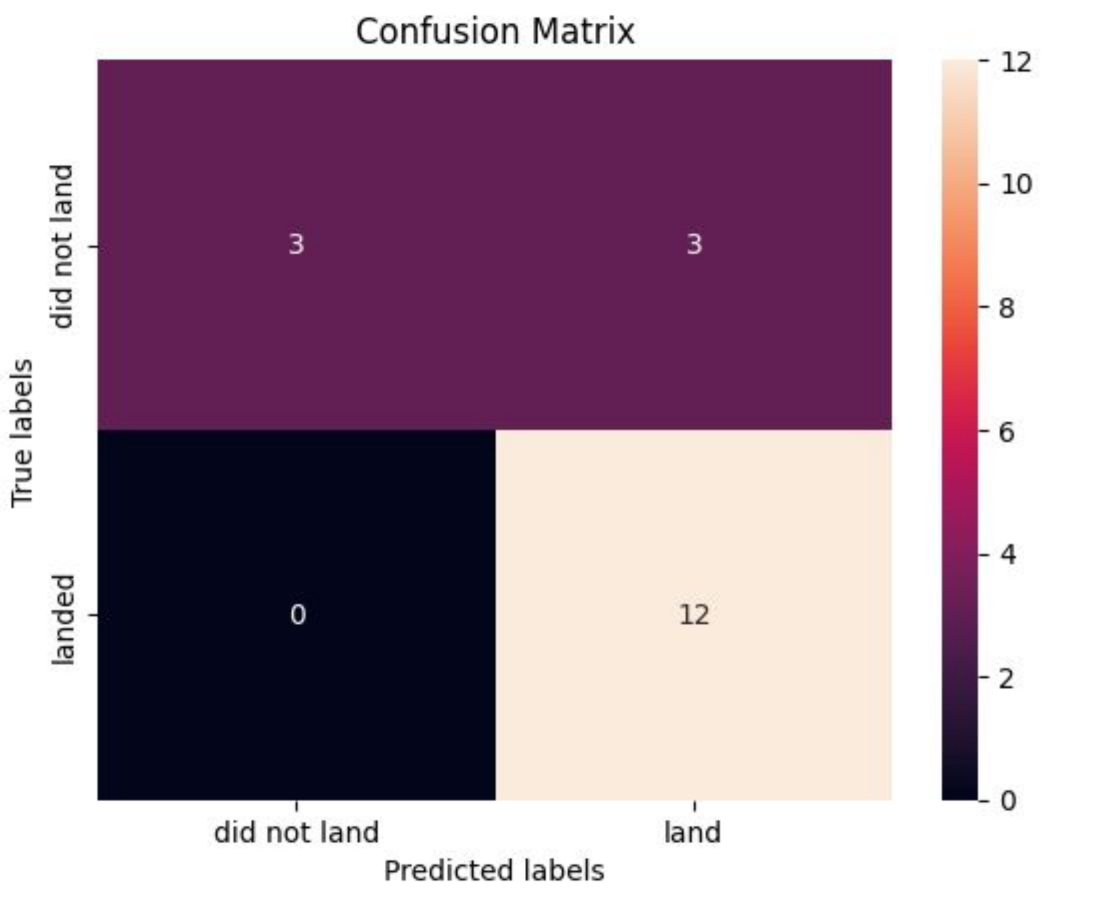
- The decision tree classifier is the model with the highest classification accuracy.



```
algorithms = ['Logistic Regression', 'Support Vector Machine', 'Decision Tree', 'K Nearest Neighbours']

scores = [lr_score, svm_score, tree_score, knn_score]

best_scores = [lr_best_score, svm_best_score, tree_best_score, knn_best_score]

column_names = ['Algorithm', 'Accuracy Score', 'Best Score']

df = pd.DataFrame(list(zip(algorithms, scores, best_scores)),columns = column_names)

df

sns.set(style="whitegrid")

plt.figure(figsize=(15,8))
sns.barplot(x=algorithms, y=best_scores, palette="Blues")
plt.title("Determining the Best Performing Classification Algorithm")
plt.ylabel("Best Score")
plt.show()

plt.figure(figsize=(15,8))
sns.barplot(x=algorithms, y=scores, palette="Blues")
plt.title("Determining the Best Performing Classification Algorithm")
plt.ylabel("Accuracy Score")
plt.show()
```

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes, with false positives being the biggest issue here. ie. unsuccessful landing marked as successful landing by the classifier.



Conclusions

- In conclusion …
  - 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.

Thank you!