IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Łukasz Bik>
<11.06.2024>

Outline

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)
- Summary of all results
  - EDA results
  - Interactive analytics
  - Predictive analysis

Introduction

- Project background and context

SpaceX offers Falcon 9 rocket launches on its website for a price of 62 million dollars, while other companies charge over 165 million dollars per launch. A significant portion of these savings is due to SpaceX's ability to reuse the rocket's first stage. Therefore, if we can predict whether the first stage will land successfully, we can estimate the cost of a launch. This information can be useful for other companies looking to compete with SpaceX in rocket launch bids. The aim of this project is to create a machine learning pipeline that can predict the successful landing of the first stage of the rocket.

- Problems you want to find answers

- What factors influence the successful landing of the rocket?
- What interactions between different features affect the success rate of the landing?
- What operational conditions need to be met to ensure a successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was gathered using the SpaceX API and through web scraping from Wikipedia.
- Perform data wrangling
  - One-hot encoding was used for the categorical features.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logistic Regression, k-nearest neighbors, Support Vector Machines, and decision tree models were developed to identify the most effective classification method.

Data Collection

Data was gathered through multiple approaches:

• Initial data acquisition involved making a GET request to the SpaceX API.

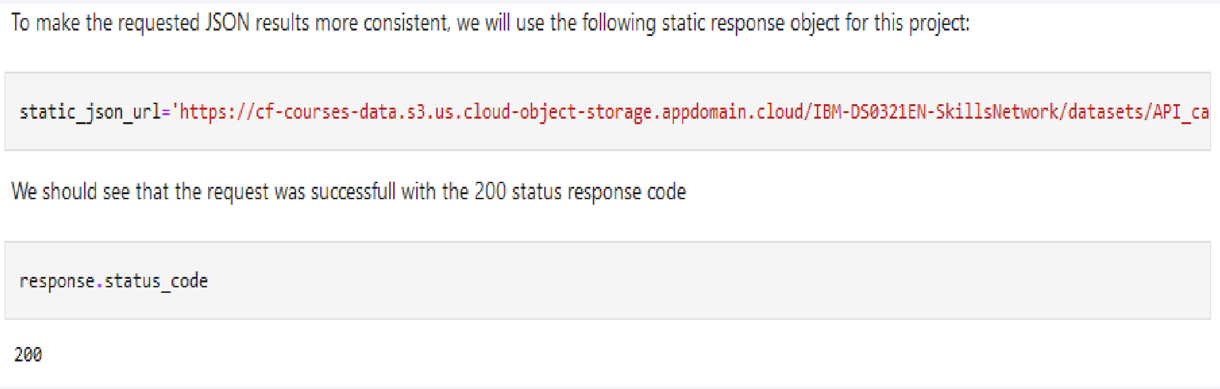
• Subsequently, we parsed the JSON response using the .json() function and transformed it into a pandas dataframe using .json_normalize().

• Following this, data cleaning was conducted, including checking for and filling in missing values as needed.

• Additionally, Falcon 9 launch records were extracted from Wikipedia using BeautifulSoup through web scraping.

• The goal was to retrieve launch data formatted as an HTML table, parse the table, and convert it into a pandas dataframe for future analysis.

Data Collection – SpaceX API



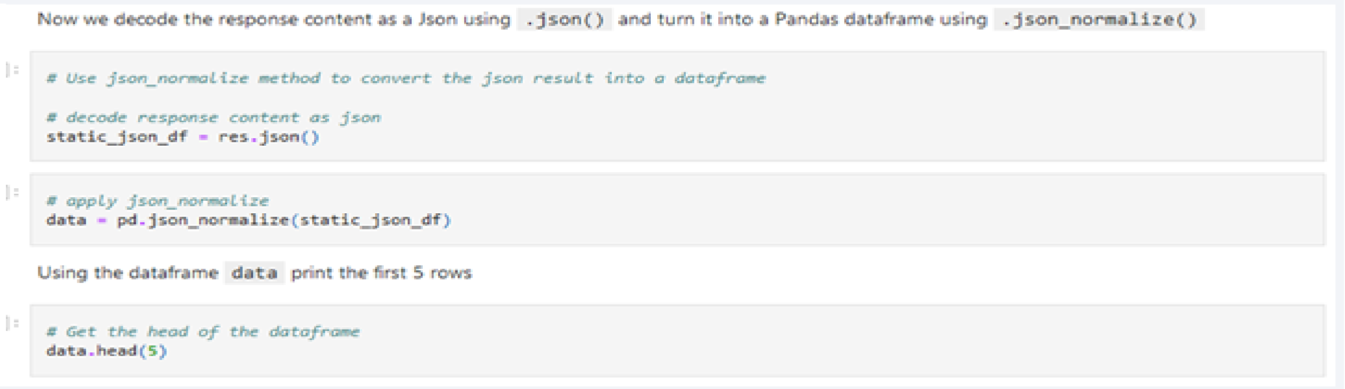
To make the requested JSON results more consistent, we will use the following static response object for this project:

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

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

```
response.status_code
```

```
200
```



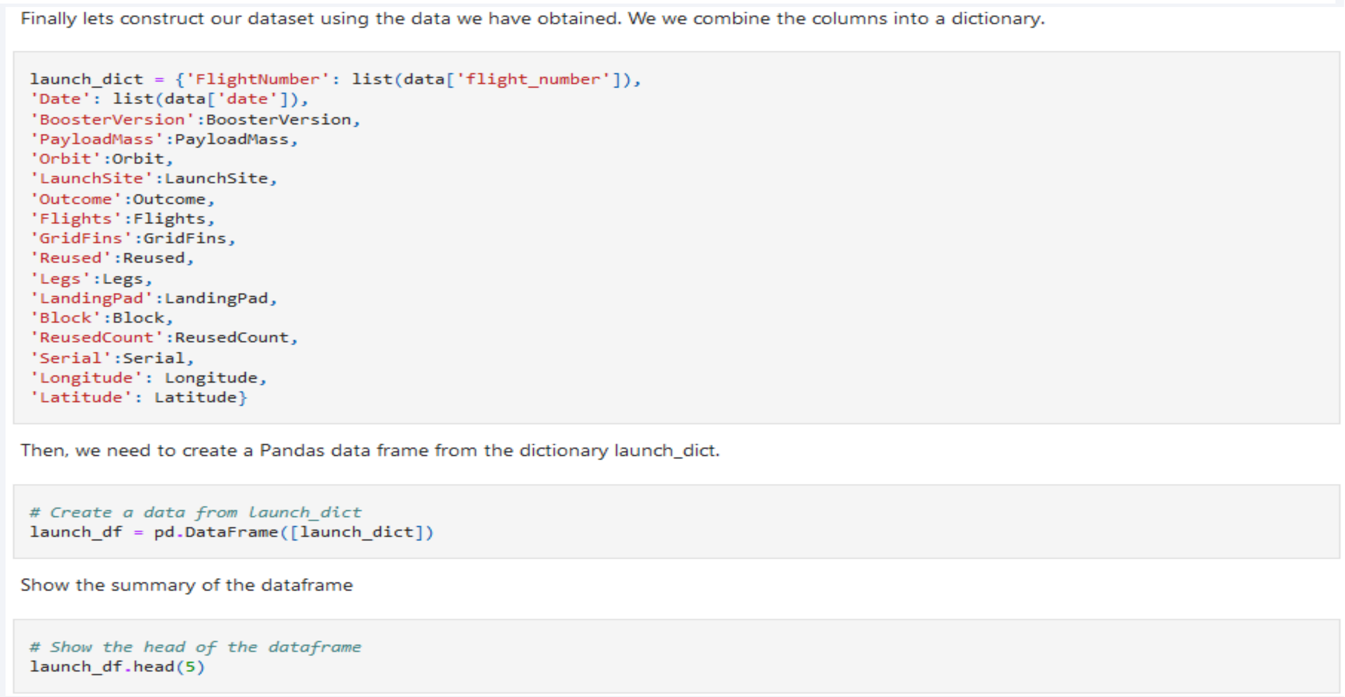
Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head(5)
```



Finally lets construct our dataset using the data we have obtained. We we combine the columns into a dictionary.

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
# Create a data from launch_dict
launch_df = pd.DataFrame([launch_dict])
```

Show the summary of the dataframe

```
# Show the head of the dataframe
launch_df.head(5)
```

1. Requested and parsed the data from the SpaceX API using a GET request.

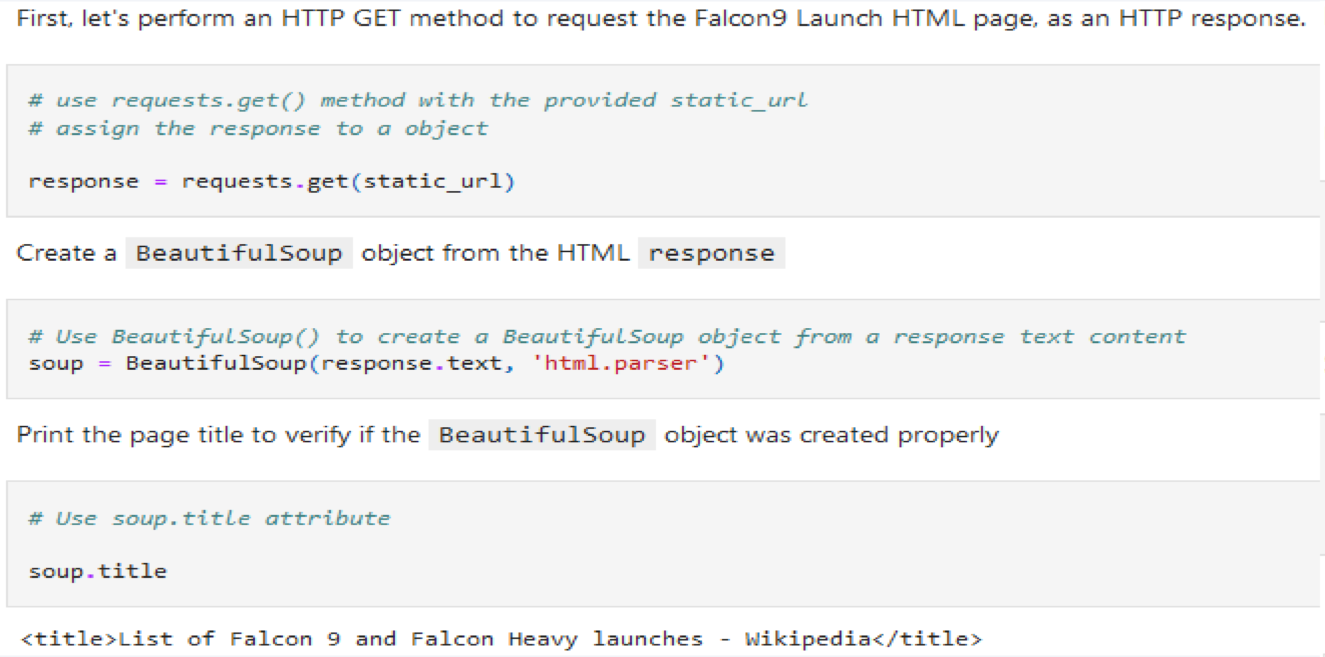
2. Stored the data and constructed the dataset into a new dictionary with relevant columns.

3. Filtered the dataframe to include only Falcon 9 launches required for project analysis.

4. Cleaned the data and removed missing values of PayloadMass.

GitHub link reference: Data Collection.ipynb

Data Collection - Scraping



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)
```

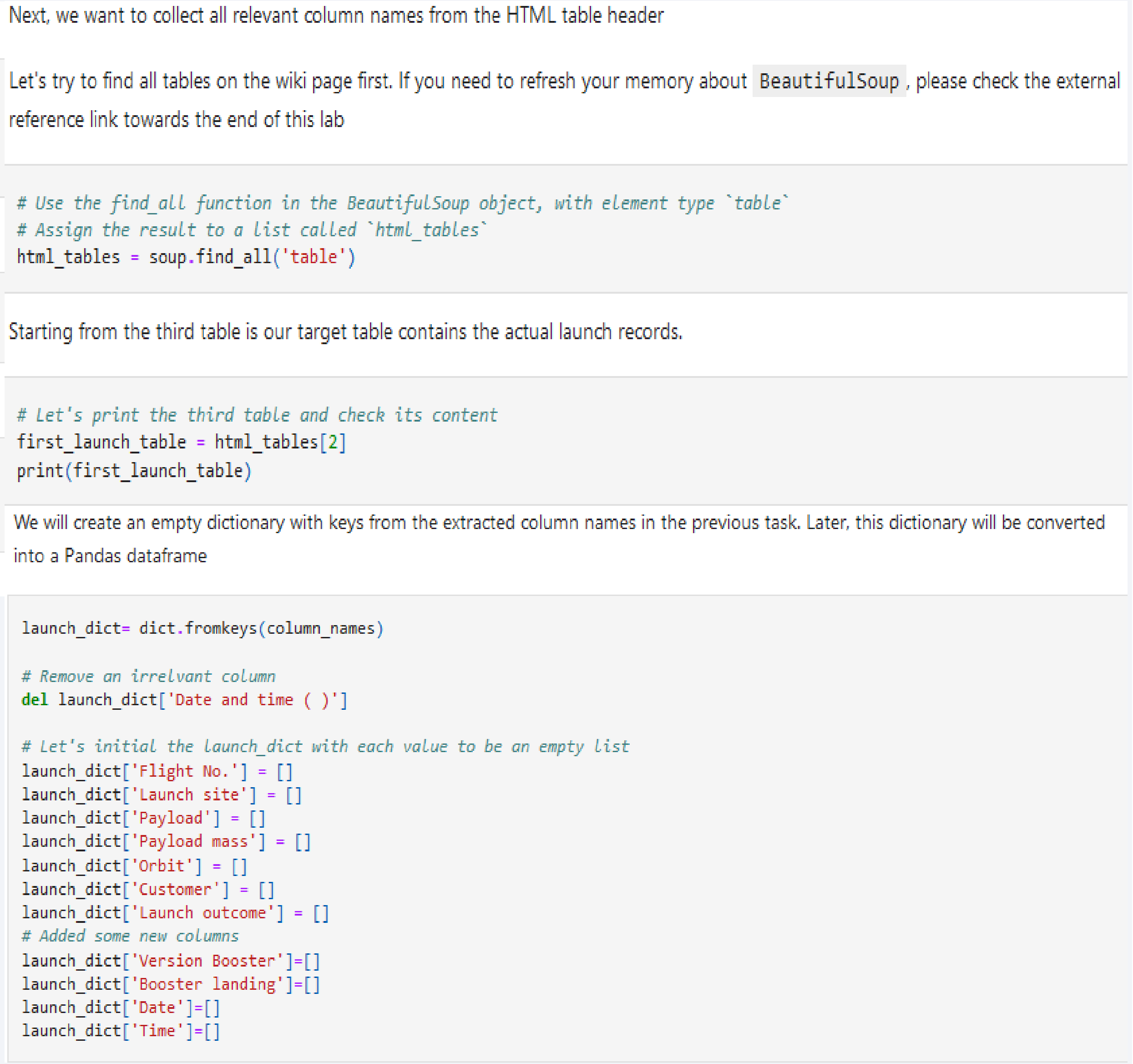
Create a `BeautifulSoup` object from the HTML `response`

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

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>
```



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)
```

We will create an empty dictionary with keys from the extracted column names in the previous task. Later, this dictionary will be converted into a Pandas dataframe

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

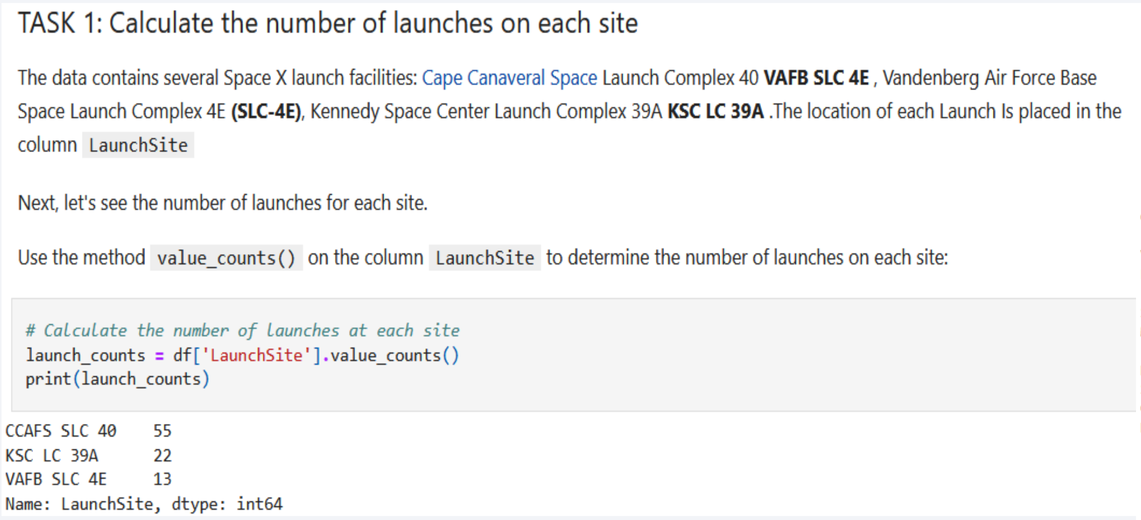
1. Requested data from Wikipedia using an HTTP GET request and BeautifulSoup.

2. Extracted all column names from the HTML table header.

3. Cleaned the column data, created an empty dictionary with the extracted columns, and appended the column names. Created a dataframe by parsing the launch HTML tables.

GitHub link reference: [Data collection using Web Scraping.ipynb]()

Data Wrangling



TASK 1: Calculate the number of launches on each site

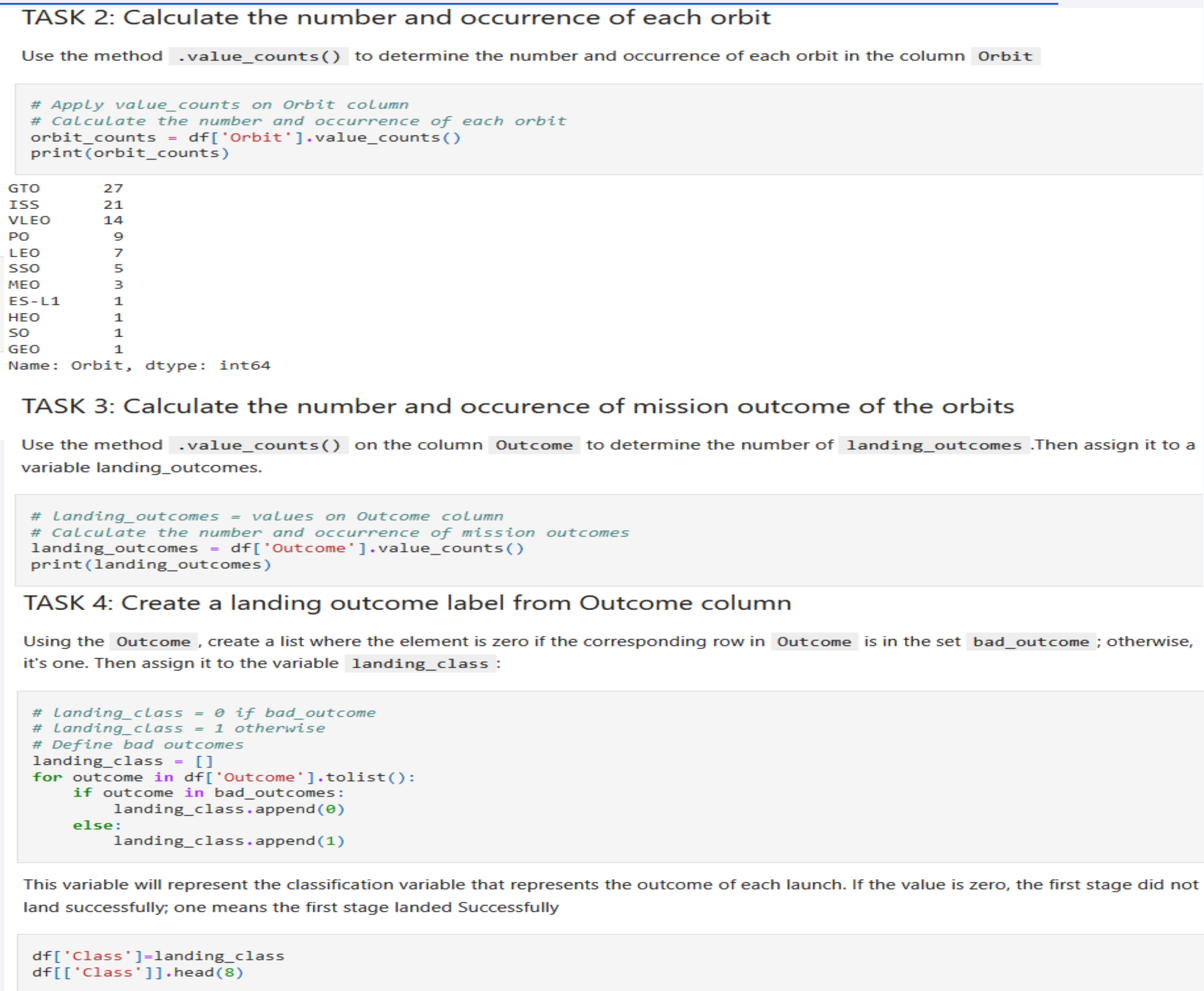
The data contains several Space X launch facilities: Cape Canaveral Space Launch Complex 40 **VAFB SLC 4E** , Vandenberg Air Force Base Space Launch Complex 4E **(SLC-4E)**, Kennedy Space Center Launch Complex 39A **KSC LC 39A** .The location of each Launch Is placed in the column `LaunchSite`

Next, let's see the number of launches for each site.

Use the method `value_counts()` on the column `LaunchSite` to determine the number of launches on each site:

```
# Calculate the number of launches at each site
launch_counts = df['LaunchSite'].value_counts()
print(launch_counts)
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```



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
# Calculate the number and occurrence of each orbit
orbit_counts = df['Orbit'].value_counts()
print(orbit_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
# Calculate the number and occurrence of mission outcomes
landing_outcomes = df['Outcome'].value_counts()
print(landing_outcomes)
```

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
# Define bad outcomes
landing_class = []
for outcome in df['Outcome'].tolist():
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

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

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

1. Determined the number of launches from each site.
2. Determined the count and occurrence of each orbit and its mission outcome.
3. Created a landing outcome label based on the outcome column.
4. Computed the average success rate.

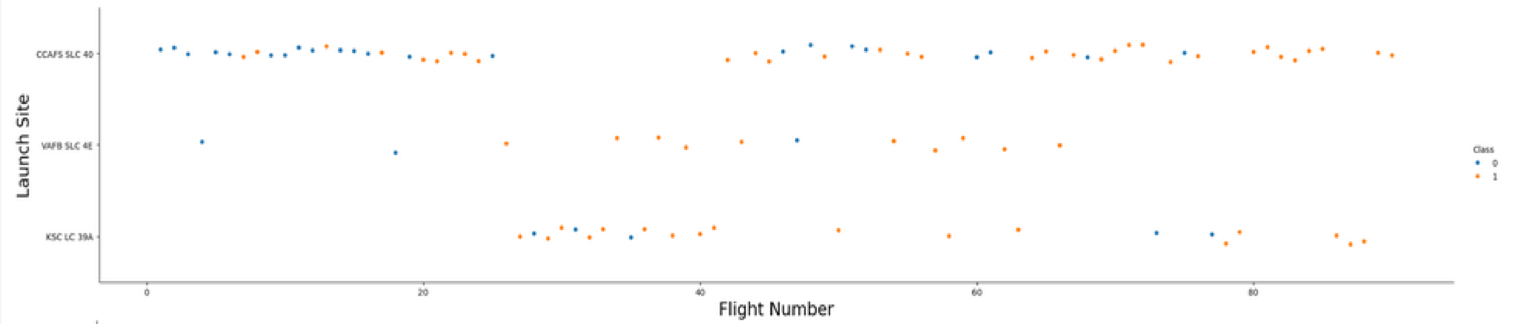
GitHub link reference: Data wrangling.ipynb

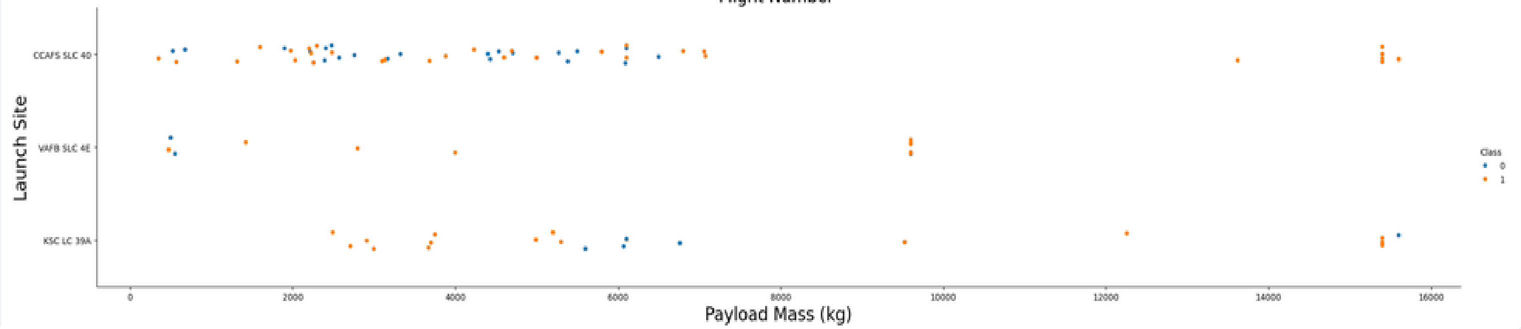
GitHub link reference: [Eda data with Visualization.ipynb]

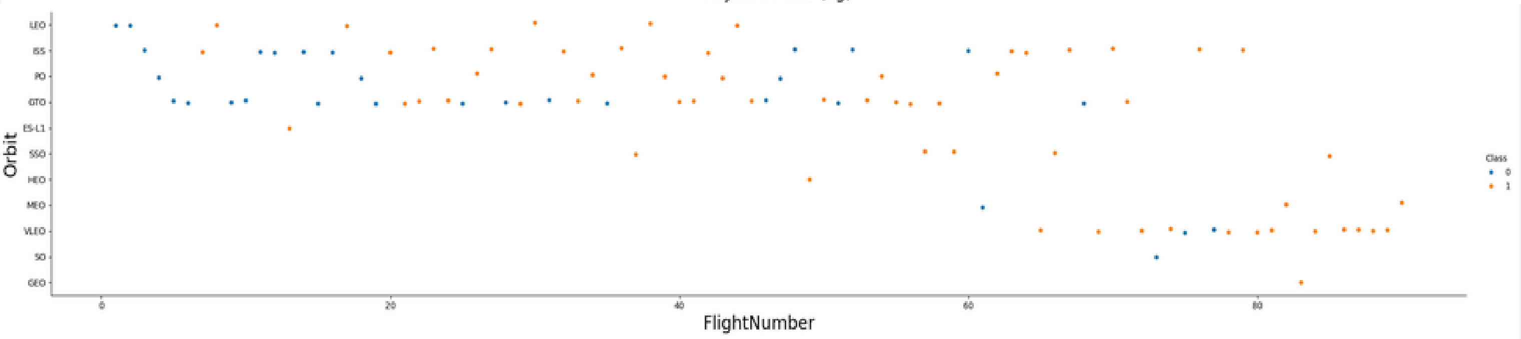
EDA with Data Visualization

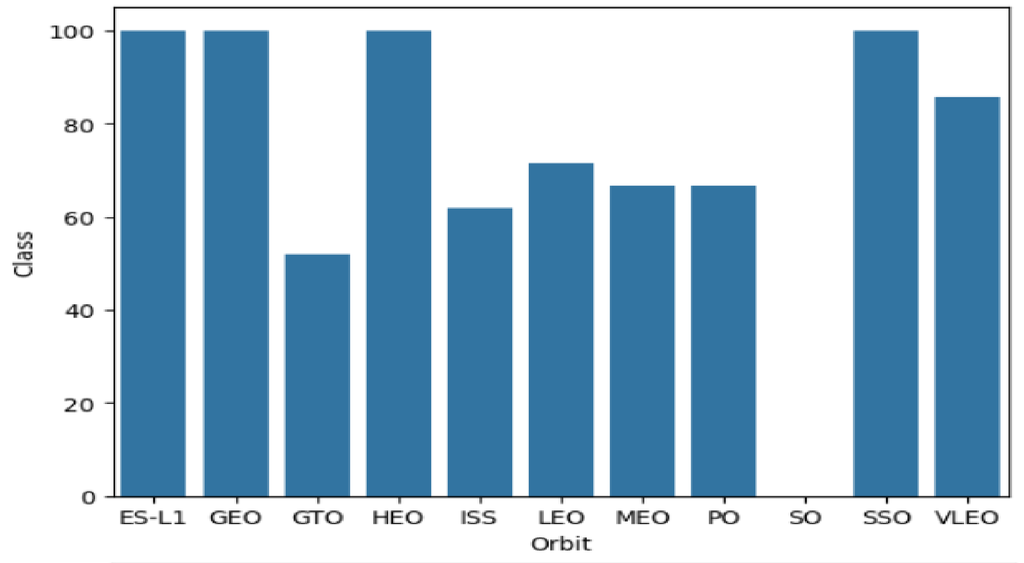
Scatter point charts are used to visualize the relationship between
(1) Flight number and lunch site, (2) Payload and lunch site, (3) Flight number and orbit type.

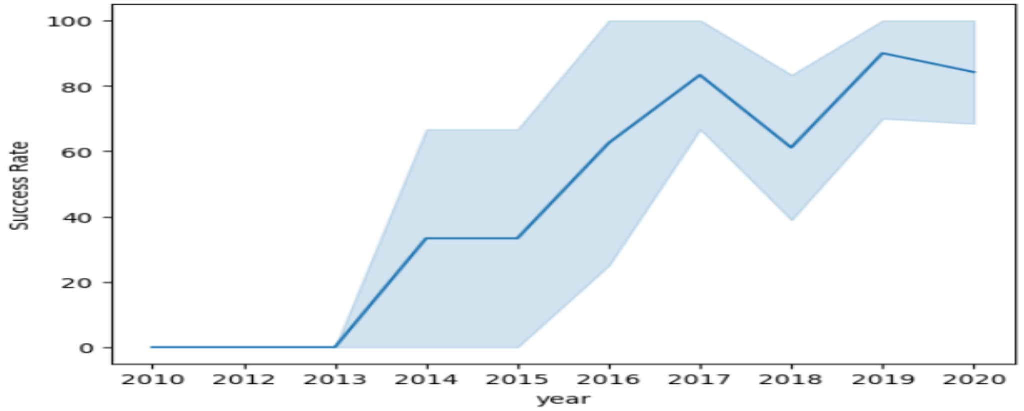
Bar chart was used to visualize the relationship between success rates of each orbit type.
Line chart was used to visualize the yearly trend of launch success.











EDA with SQL

SQL queries are performed to:

1.Retrieve the names of unique launch sites.

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

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

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

5.List the date when the first successful landing on a ground pad was achieved.

6.List the names of boosters that have successful landings on drone ships and have a payload mass greater than 4000 but less than 6000.

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

8.List the names of the booster versions that have carried the maximum payload mass using a subquery.

9.Display the failed landing outcomes on drone ships, along with their booster version and launch site names.

GitHub link reference: Eda - SQL.ipynb

Build an Interactive Map with Folium

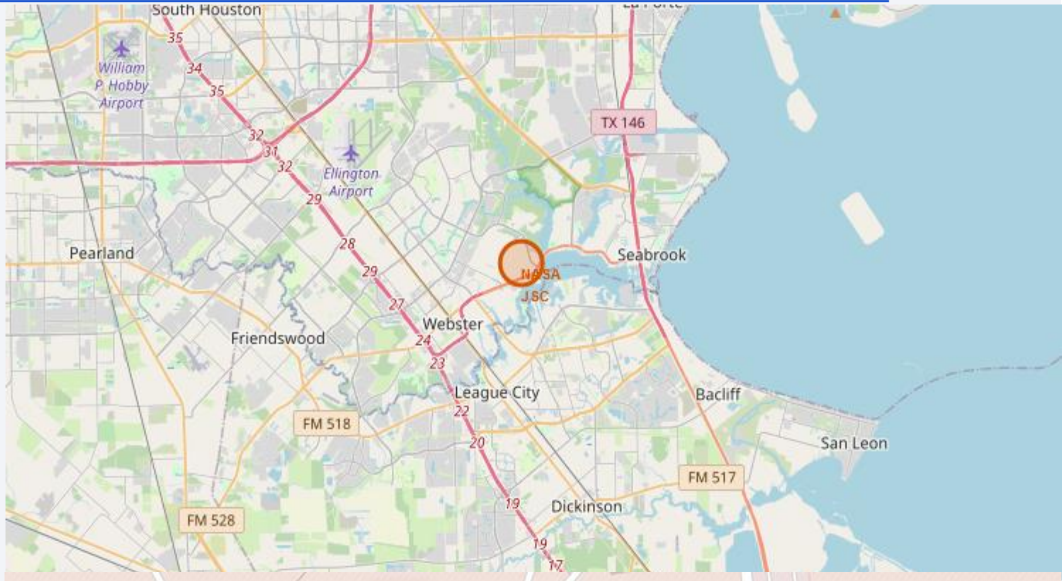
1. We marked all launch sites and added map objects such as markers, circles, and lines to indicate the success or failure of launches for each site on the folium map.

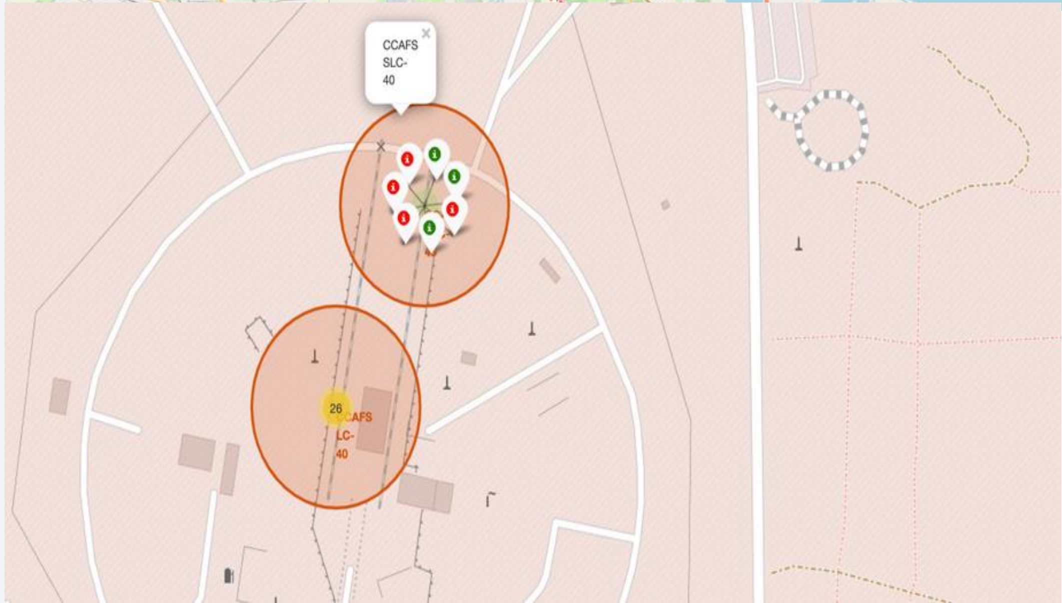
2. We assigned launch outcomes (failure or success) to classes 0 and 1, i.e., 0 for failure and 1 for success.

3. Using color-labeled marker clusters, we identified which launch sites have a relatively high success rate.

4. We calculated the distances between each launch site and its surroundings. We answered some questions, for instance:

- Are launch sites near railways, highways, and coastlines?
- Do launch sites maintain a certain distance from cities?





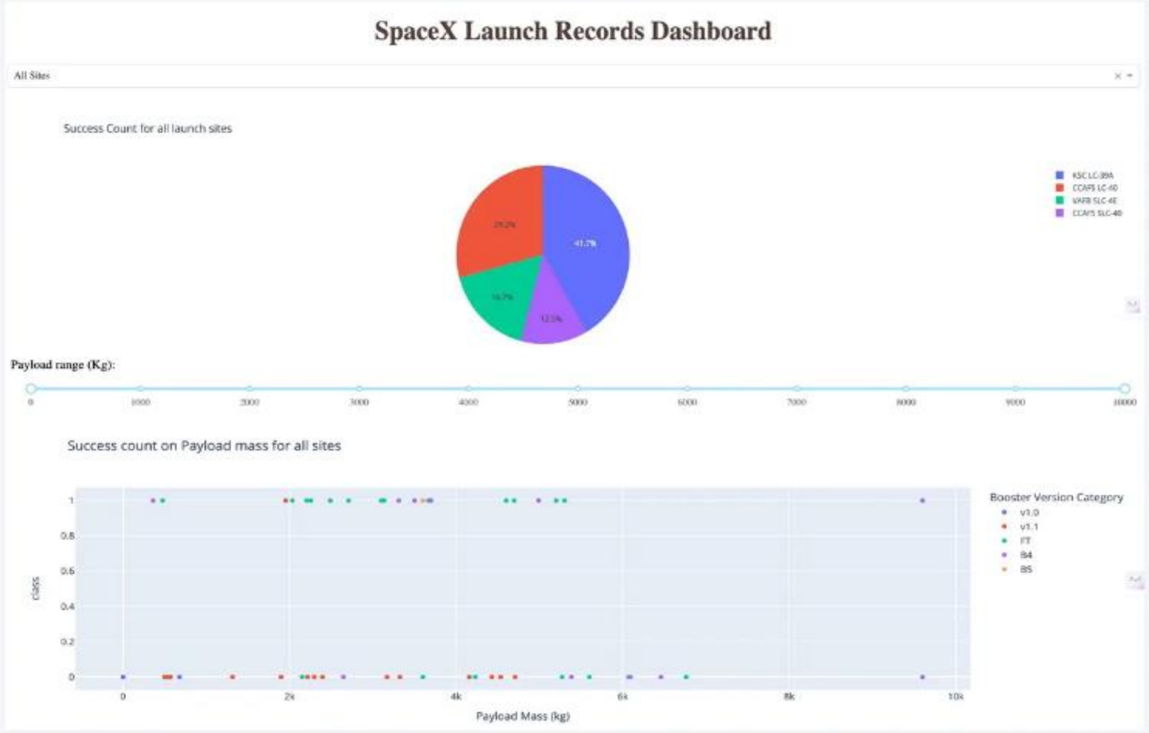
GitHub link reference: [Folium - Visual Analytics.ipynb]

Build a Dashboard with Plotly Dash

1. An interactive dashboard was constructed using Plotly Dash.

2. Pie charts were generated displaying the total launches from specific sites.

3. Scatter plots were created to illustrate the relationship between Outcome and Payload Mass (Kg) for various booster versions.



GitHub link reference: Build a Dashboard with Plotly Dash.py

Predictive Analysis (Classification)

1.Various machine learning models were constructed and hyperparameters were optimized using GridSearchCV.

2. Accuracy served as the primary metric for model evaluation, with enhancements made through feature engineering and algorithm tuning.

3. The most effective classification model was identified through experimentation.

4. Data was imported using numpy and pandas, processed, and divided into training and testing sets.

GitHub link reference: Machine Learning Prediction.ipynb

Results

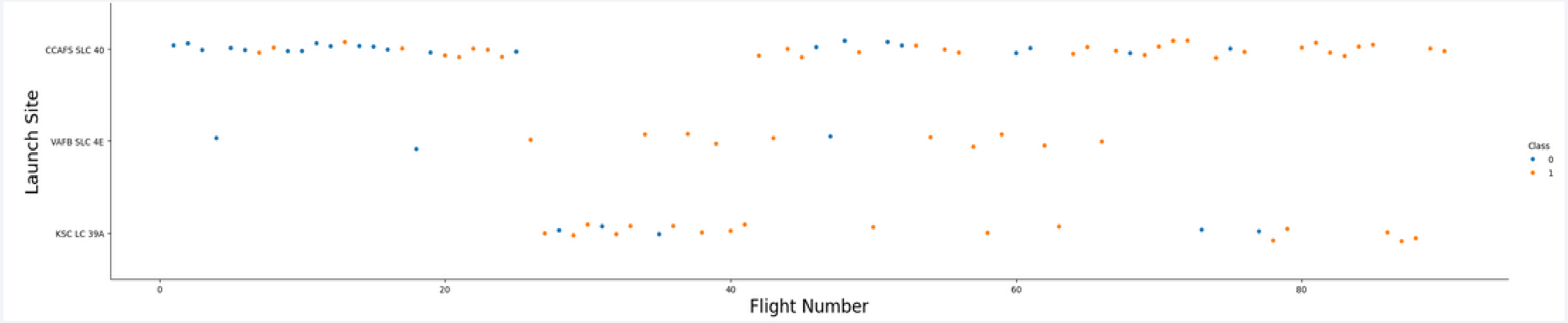
1. Logistic Regression (LR), Support Vector Machines (SVM), and K-Nearest Neighbors (KNN) are the most effective models for predicting outcomes in this dataset.
2. Payloads with lower weight demonstrate better performance compared to heavier ones.
3. The probability of a SpaceX launch succeeding increases with the accumulated years of experience, indicating a trend towards increasingly successful launches over time.
4. Launch Complex 39A at Kennedy Space Center boasts the highest number of successful launches among all launch sites.

Section 2

Insights drawn from EDA

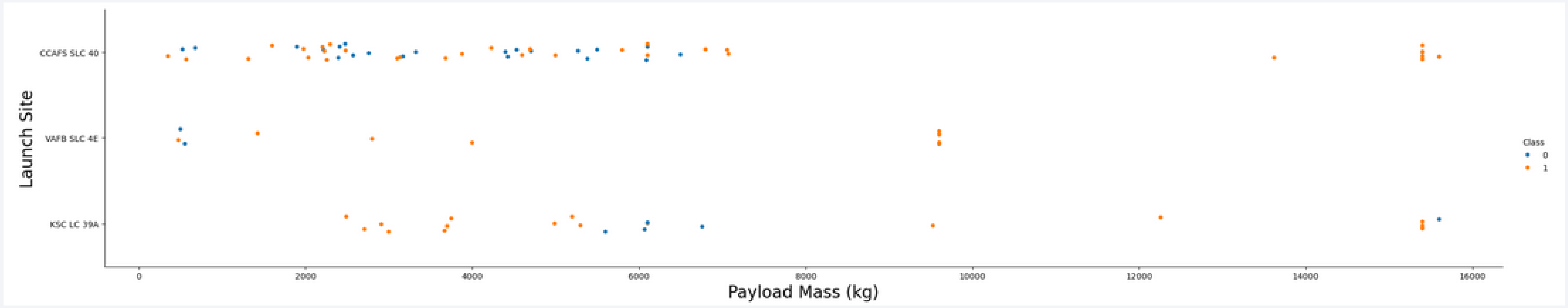
Flight Number vs. Launch Site

The total number of launches from Launch Complex 40 at Cape Canaveral Space Launch Complex (CCAFS SLC-40) is notably higher than from other launch sites.



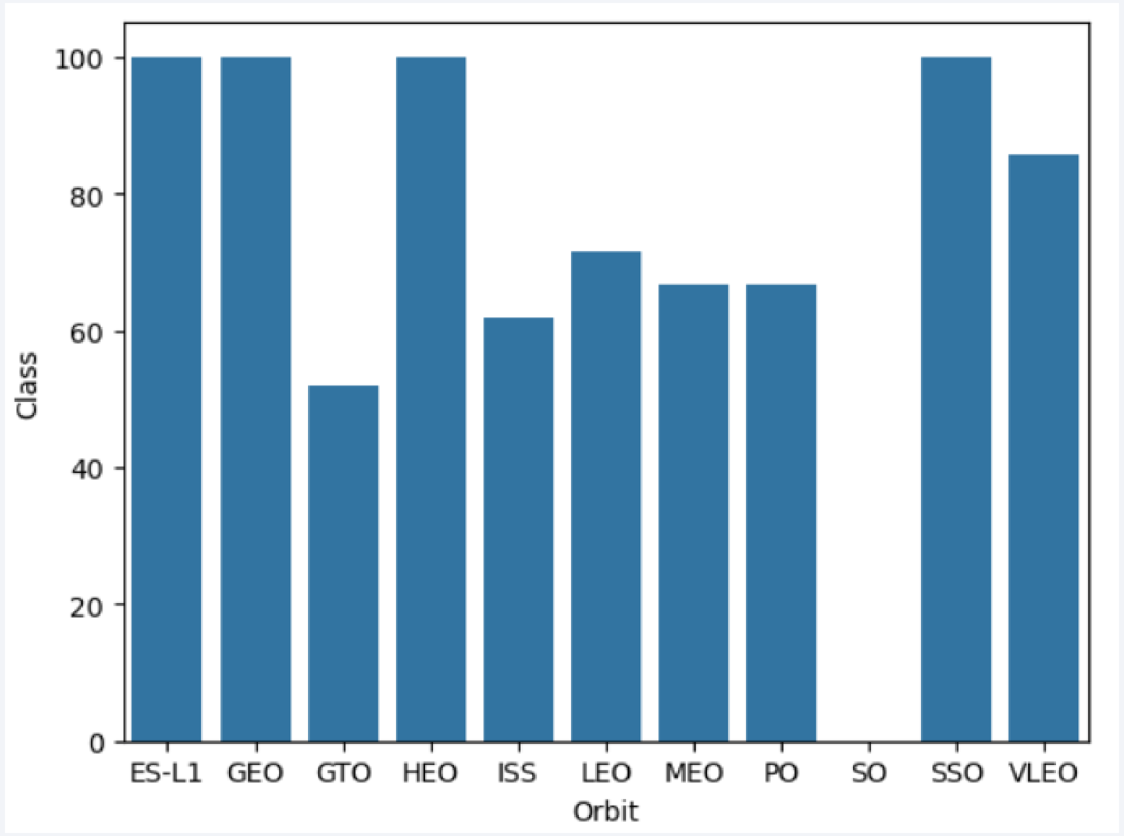
Payload vs. Launch Site

Payloads with lower mass have more launches compared to those with higher mass across all three launch sites.



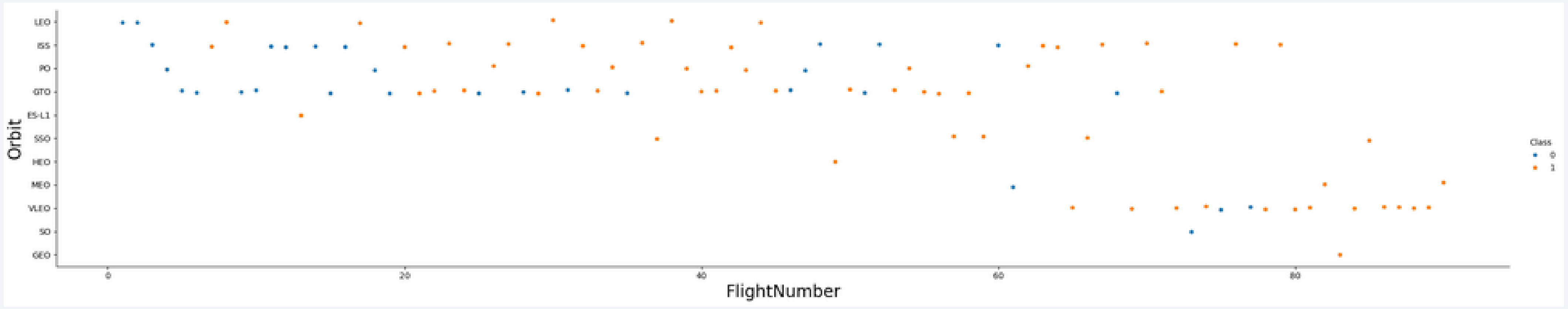
Success Rate vs. Orbit Type

The orbit types ES-L1, GEO, HEO, and SSO exhibit the highest success rates among all.



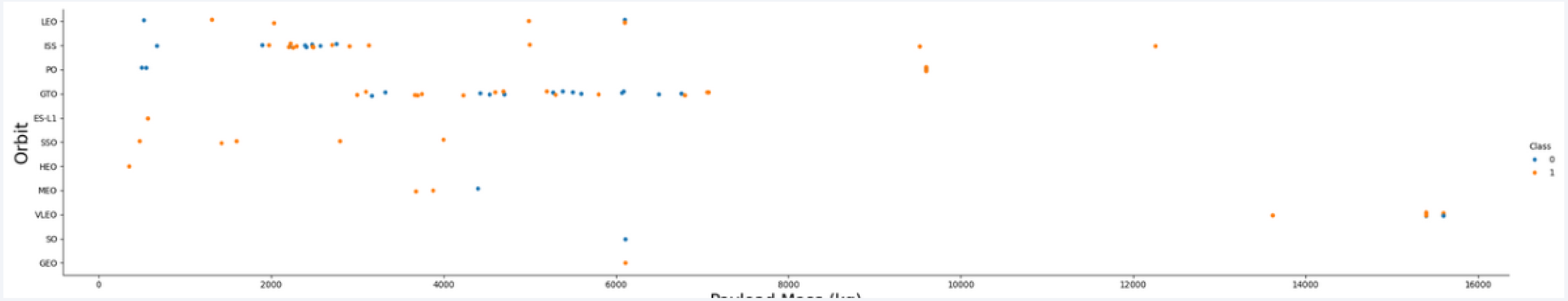
Flight Number vs. Orbit Type

In the earlier years, LEO, ISS, PO, and GTO orbits had the most launches, but over time, the focus gradually shifted to VLEO orbit.



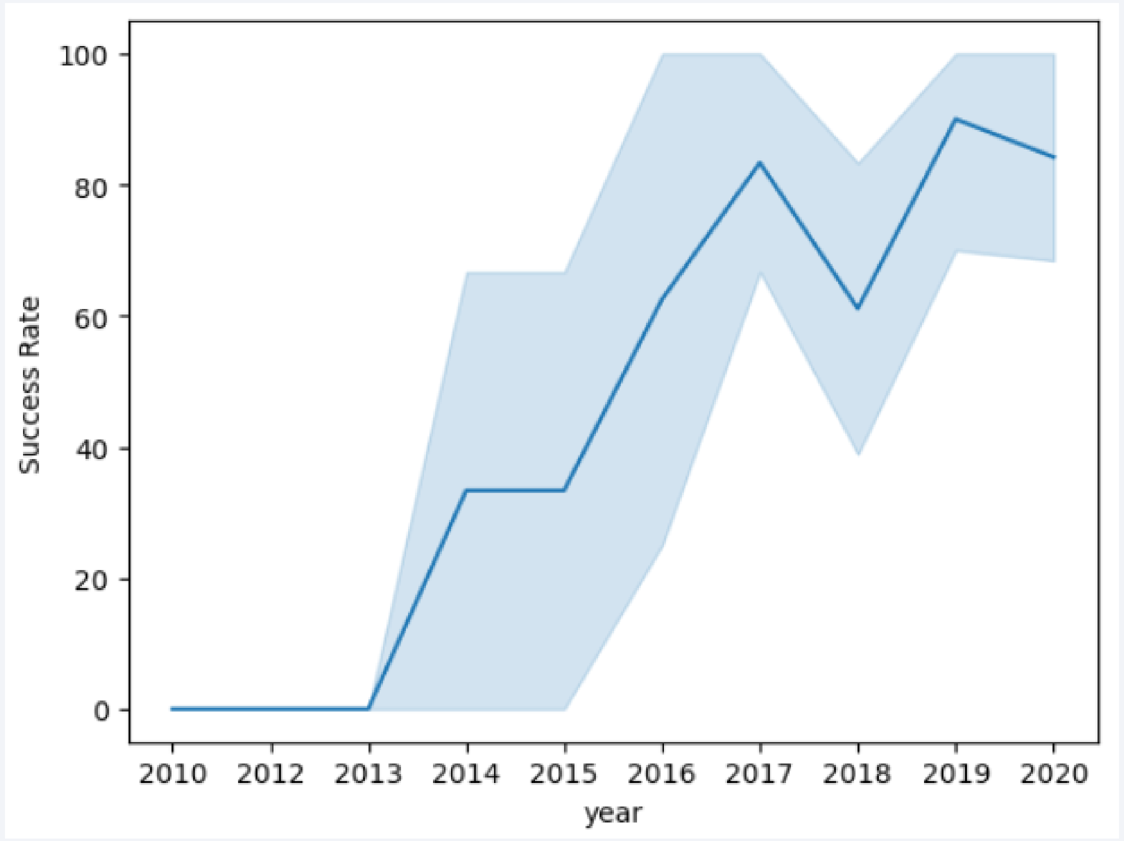
Payload vs. Orbit Type

Heavy payloads tend to have higher successful landing rates for PO, LEO, and ISS orbits, but for GTO orbit, success is less predictable with nearly an equal distribution of successes and failures.

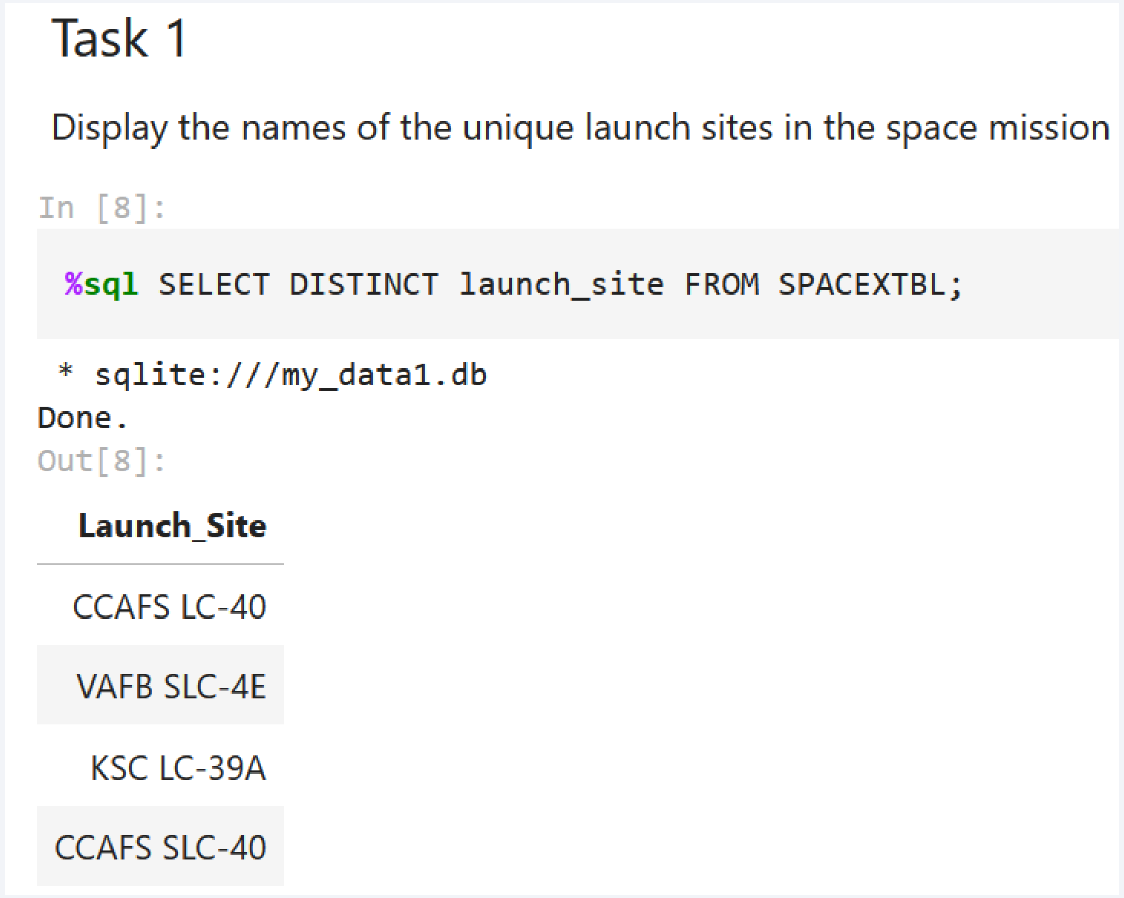


Launch Success Yearly Trend

The success rate of launches has been increasing from 2013 to 2020, likely due to advancements in technology and accumulated experience.



All Launch Site Names



Task 1

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

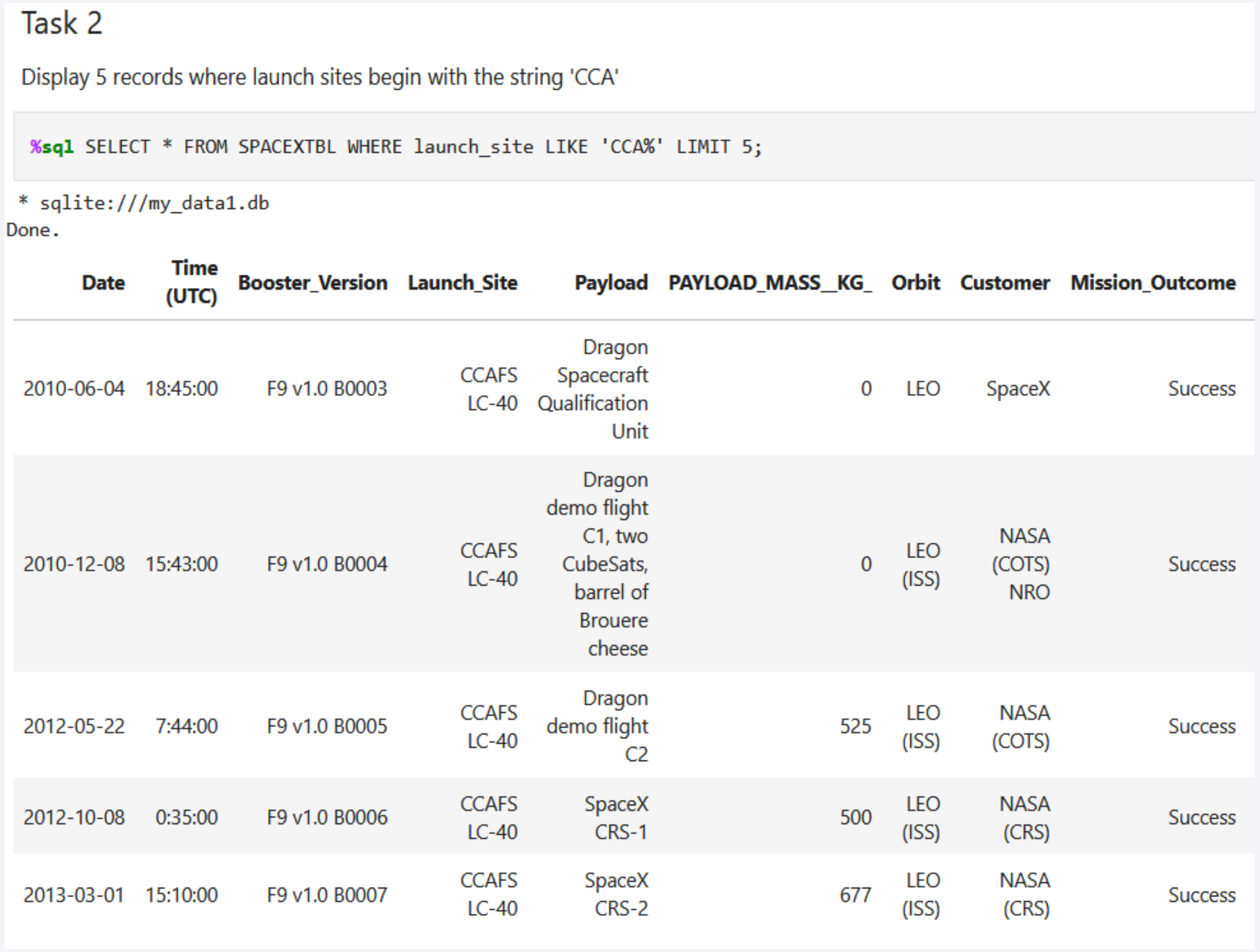
```
In [8]:
%sql SELECT DISTINCT launch_site FROM SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
Out[8]:
```

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

Executed an SQL query to retrieve all launch site names.

Launch Site Names Begin with 'CCA'



Task 2

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

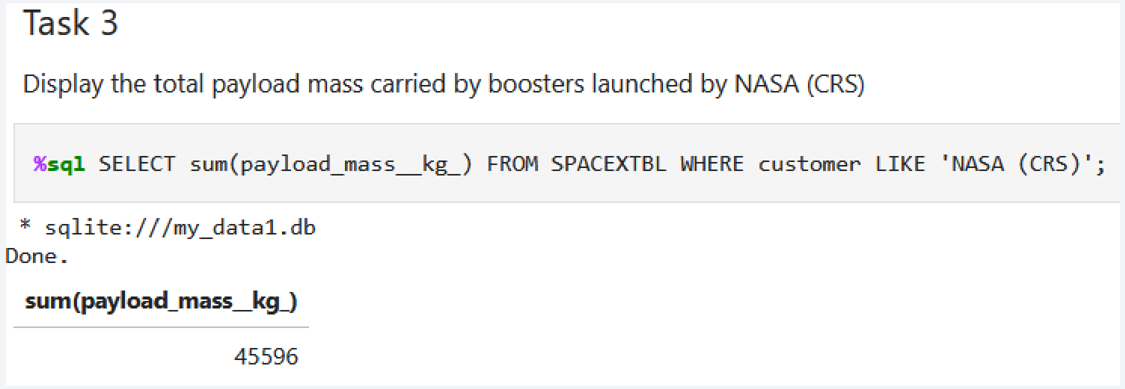
```
%sql SELECT * FROM SPACEXTBL WHERE launch_site LIKE 'CCA%' LIMIT 5;
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success
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
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success

Executed an SQL query to retrieve 5 launch site names that start with 'CCA'.

Total Payload Mass



Task 3

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

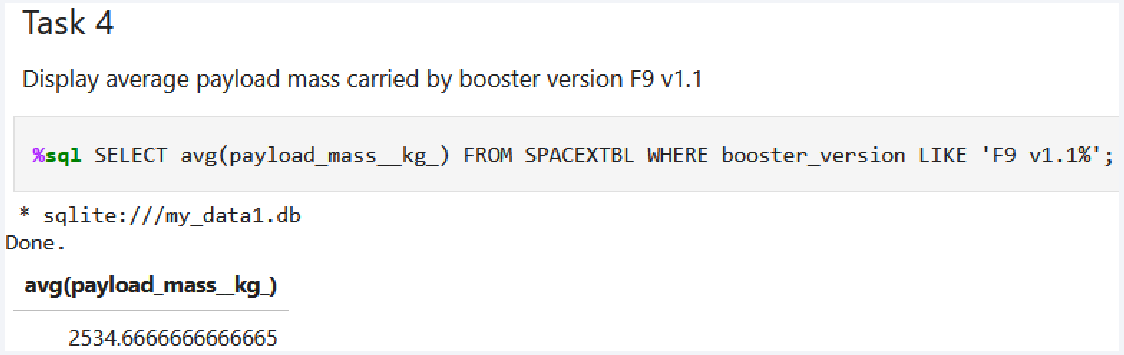
```
%sql SELECT sum(payload_mass__kg_) FROM SPACEXTBL WHERE customer LIKE 'NASA (CRS)';
```

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

sum(payload_mass__kg_)
45596

Executed an SQL query to retrieve the total payload mass carried by boosters launched by NASA (CRS).

Average Payload Mass by F9 v1.1


Task 4

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

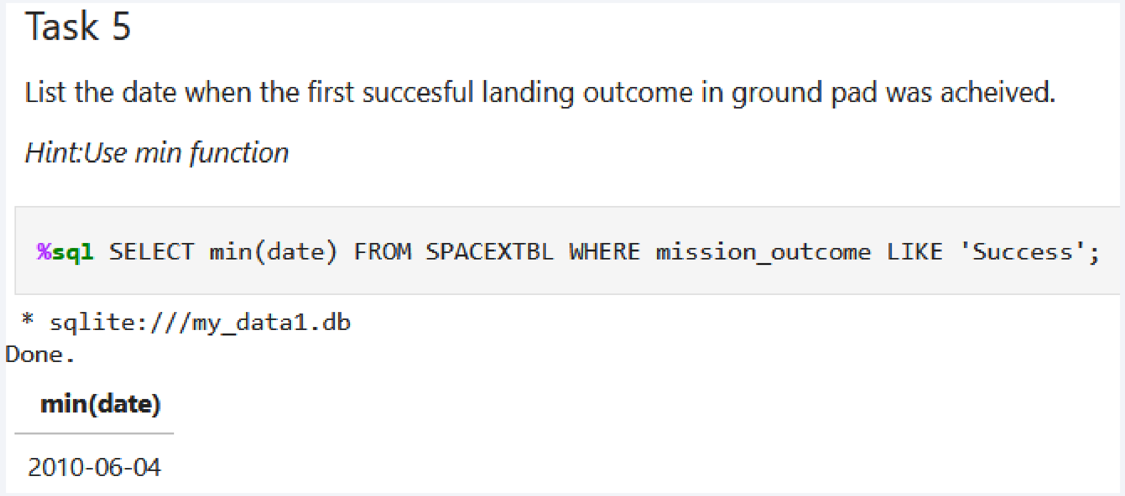
```
%sql SELECT avg(payload_mass__kg_) FROM SPACEXTBL WHERE booster_version LIKE 'F9 v1.1%';
```

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

avg(payload_mass__kg_)
2534.6666666666665

Executed an SQL query to calculate the average payload mass carried by booster version F9 v1.1.

First Successful Ground Landing Date



Task 5

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

Hint:Use min function

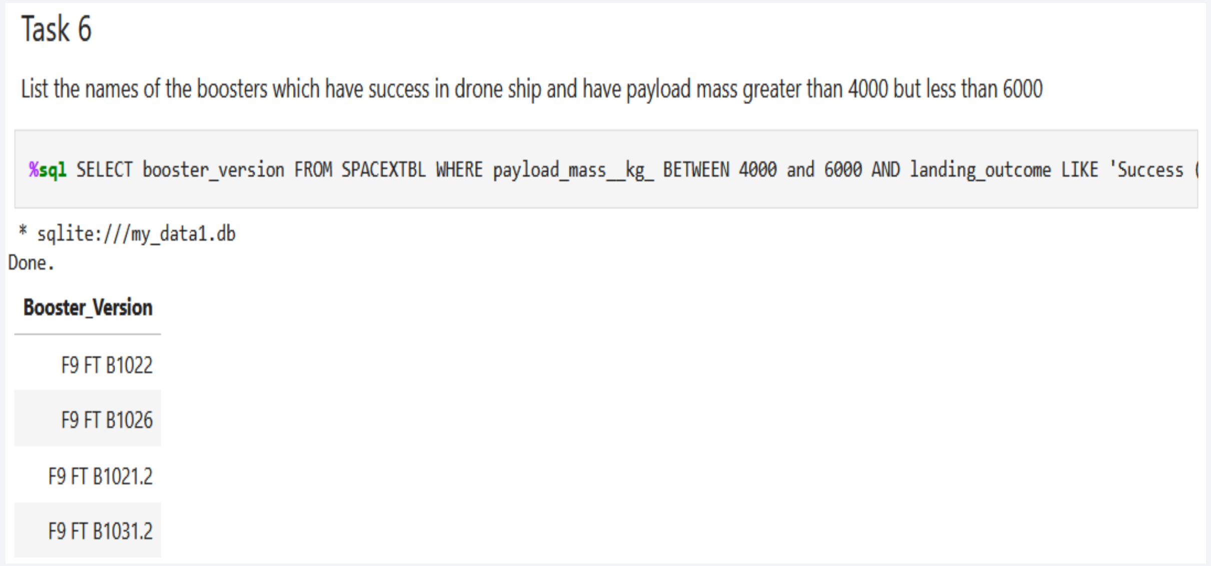
```
%sql SELECT min(date) FROM SPACEXTBL WHERE mission_outcome LIKE 'Success';
```

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

min(date)
2010-06-04

Executed an SQL query to find the dates of the first successful landing on a ground pad.

Successful Drone Ship Landing with Payload between 4000 and 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

```
%sql SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_ BETWEEN 4000 and 6000 AND landing_outcome LIKE 'Success (
```

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

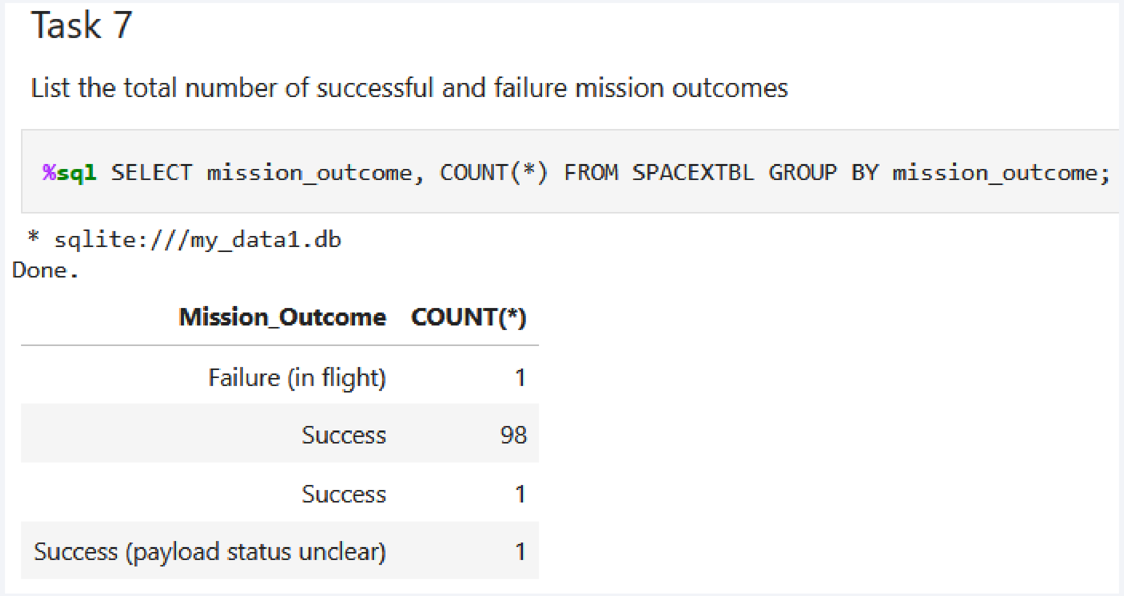
Booster_Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Executed an SQL query to list the names of boosters that successfully landed on a drone ship with a payload mass between 4000 and 6000.

entire code:

```
%sql SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_
BETWEEN 4000 and 6000 AND landing_outcome LIKE 'Success (drone ship)';
```

Total Number of Successful and Failure Mission Outcomes



Task 7

List the total number of successful and failure mission outcomes

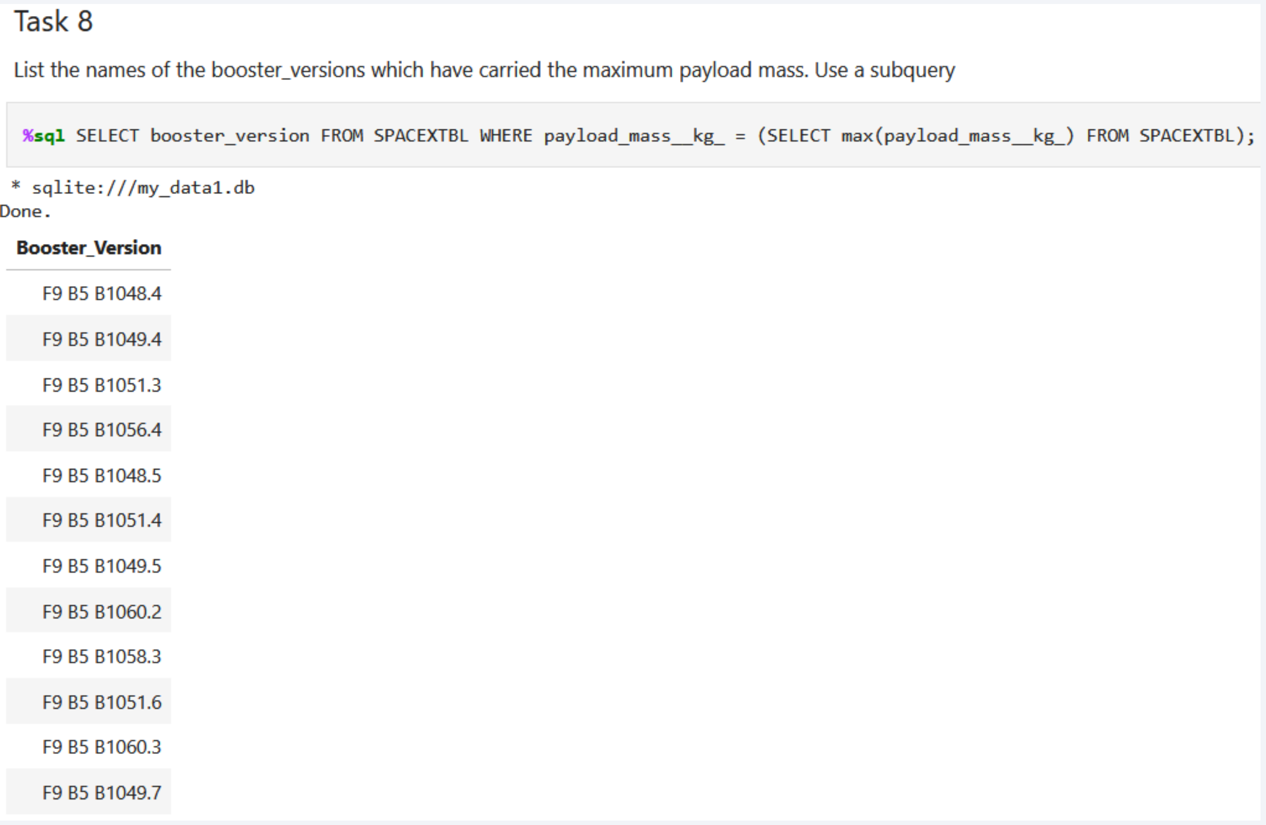
```
%sql SELECT mission_outcome, COUNT(*) FROM SPACEXTBL GROUP BY mission_outcome;
```

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

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

Executed an SQL query to calculate the total number of successful and failed mission outcomes.

Boosters Carried Maximum Payload



Task 8

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

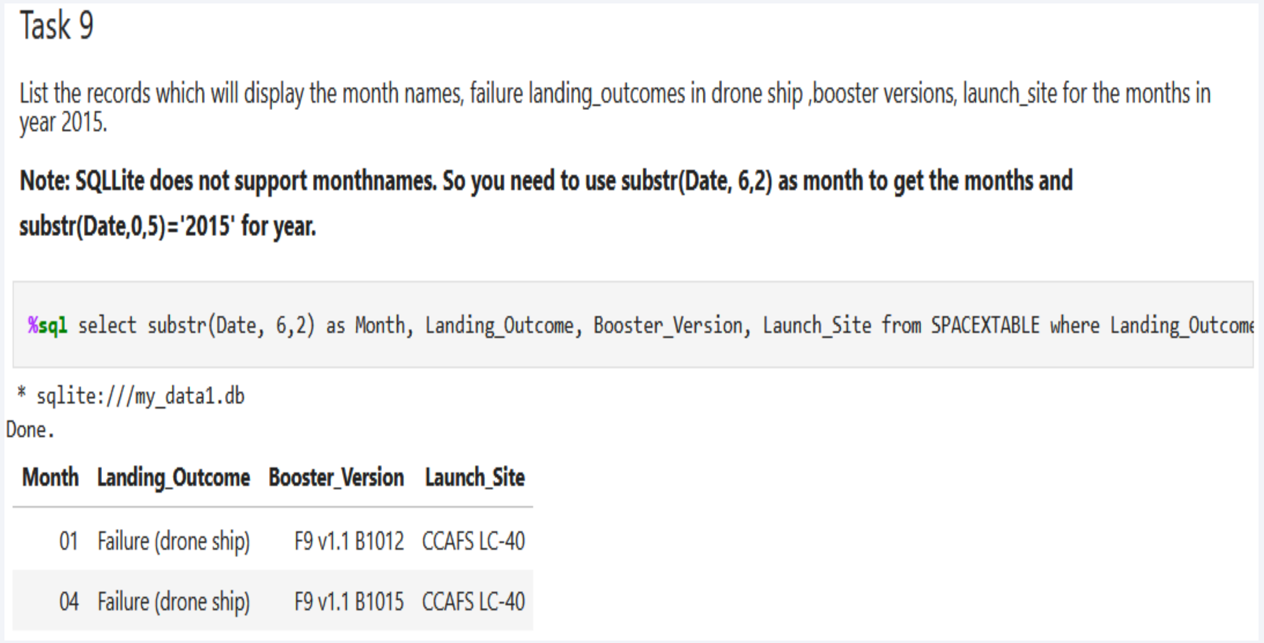
```
%sql SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_ = (SELECT max(payload_mass__kg_) FROM SPACEXTBL);
```

```
 * 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

Executed an SQL query to list the names of the booster that have carried the maximum payload mass.

2015 Launch Records



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.

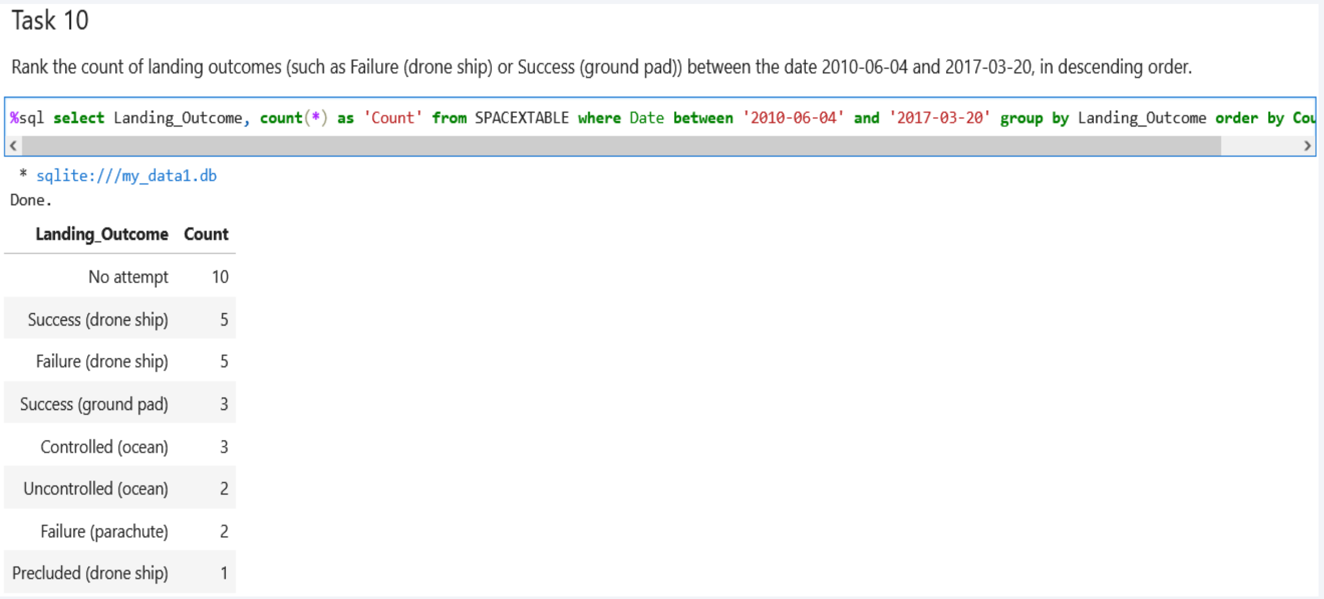
```
%sql select substr(Date, 6,2) as Month, Landing_Outcome, Booster_Version, Launch_Site from SPACEXTABLE where Landing_Outcome
```

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

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

Executed an SQL query to list the failed landing outcomes on drone ships, along with their booster versions and launch site names for the year 2015.

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



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.

```
%sql select Landing_Outcome, count(*) as 'Count' from SPACEXTABLE where Date between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by Cou
```

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

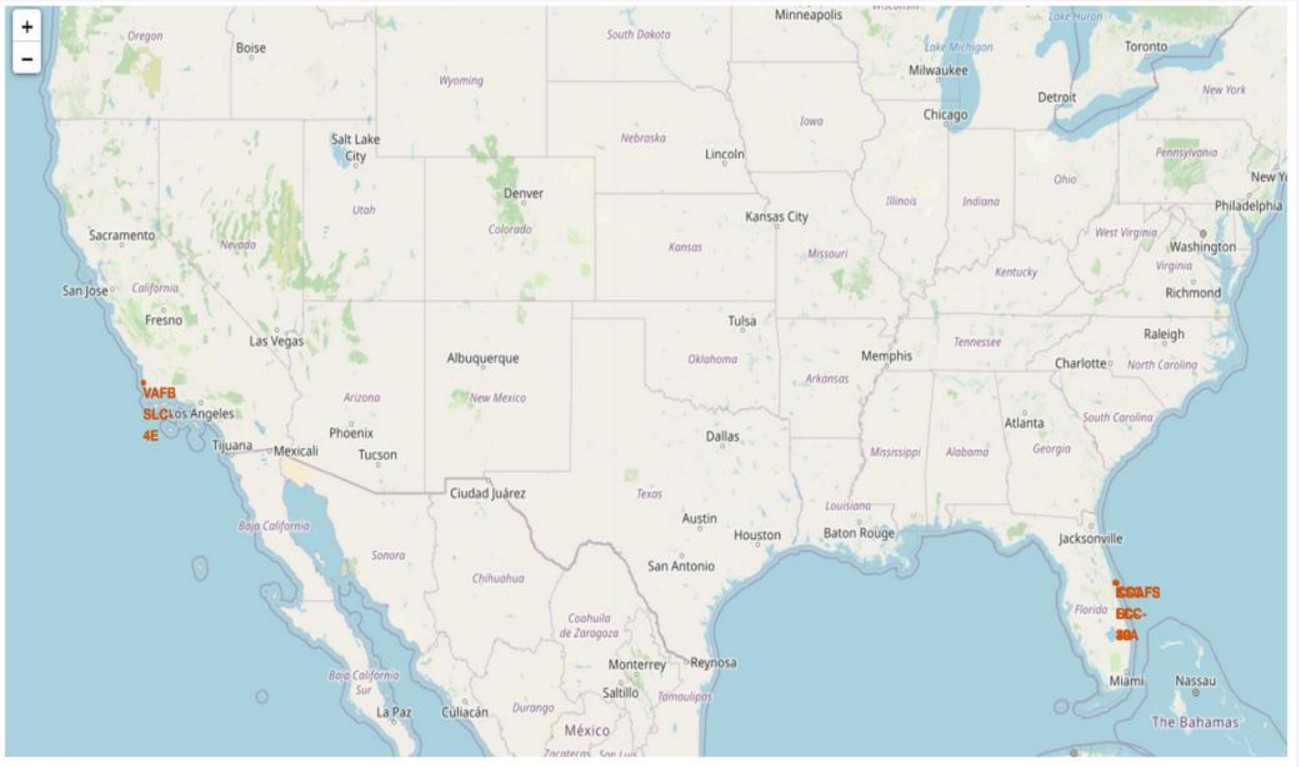
Landing_Outcome	Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Executed an SQL query to rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the dates 2010-06-04 and 2017-03-20 in descending order.

Section 3

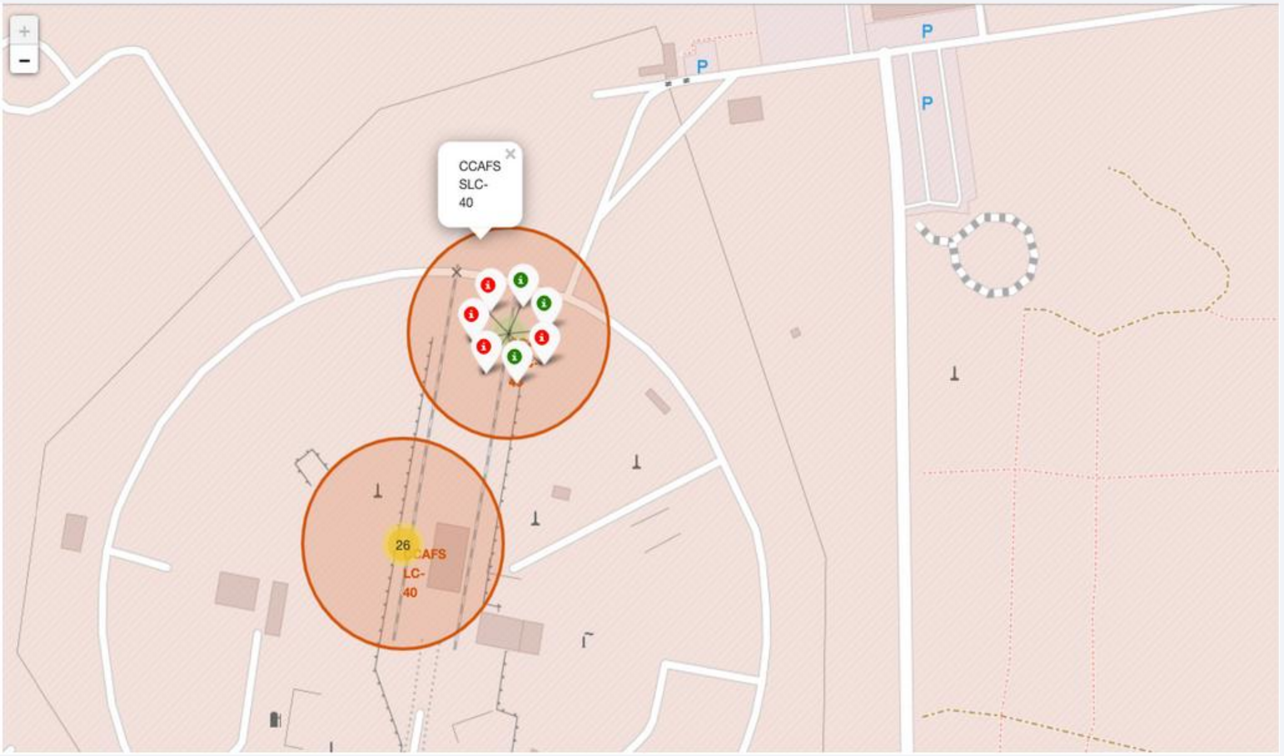
Launch Sites Proximities Analysis

All launch sites on a map



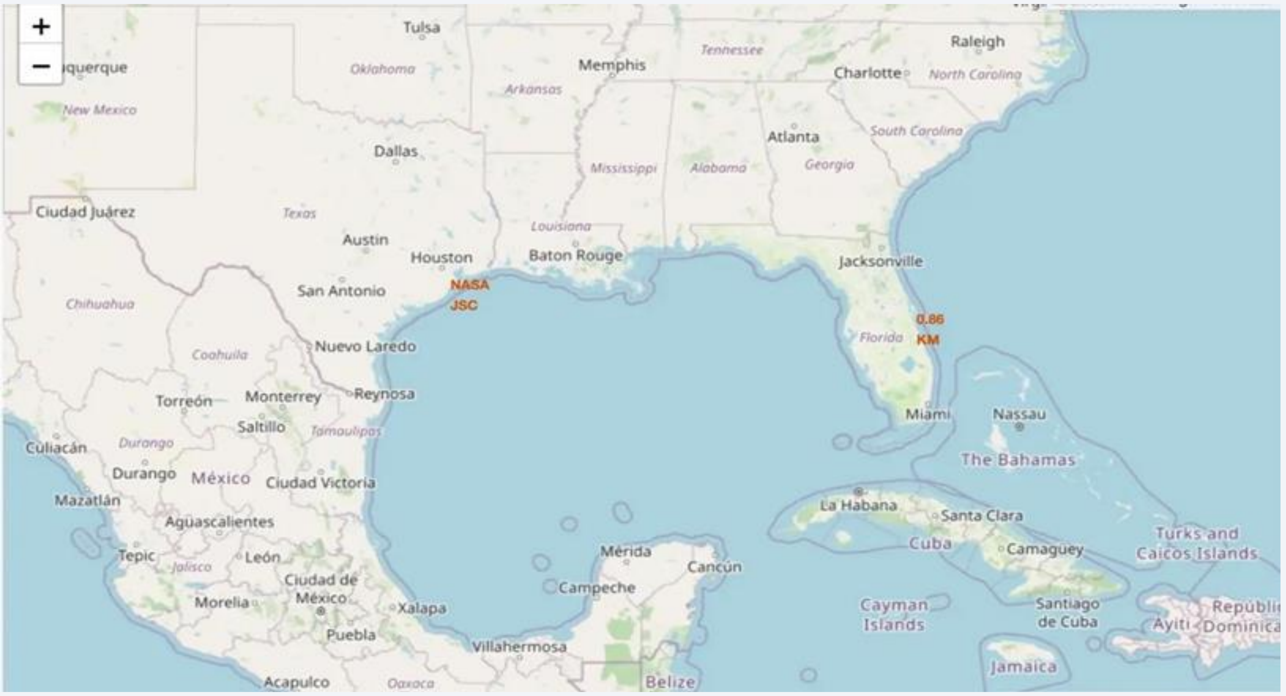
The launch sites are marked with a marker indicating their names on the map.

All success/failed launches for each site on the map

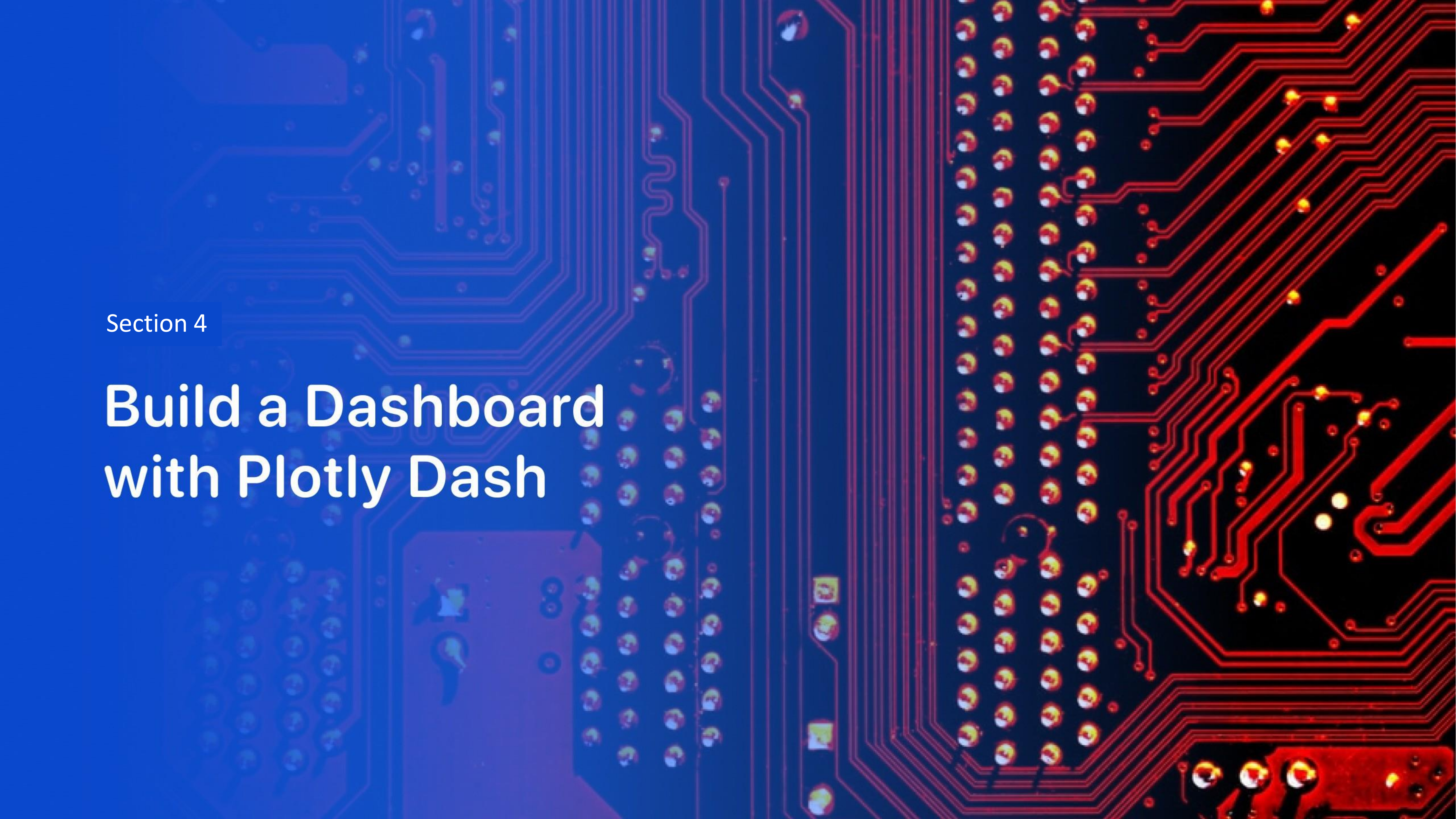


The launch records are clustered on the map and labeled with green markers for successful launches and red markers for unsuccessful ones.

Distances between a launch site to its proximities



The nearest coastline from NASA JSC is marked as a point using MousePosition, and the distance between the coastline point and the launch site is approximately 0.86 km

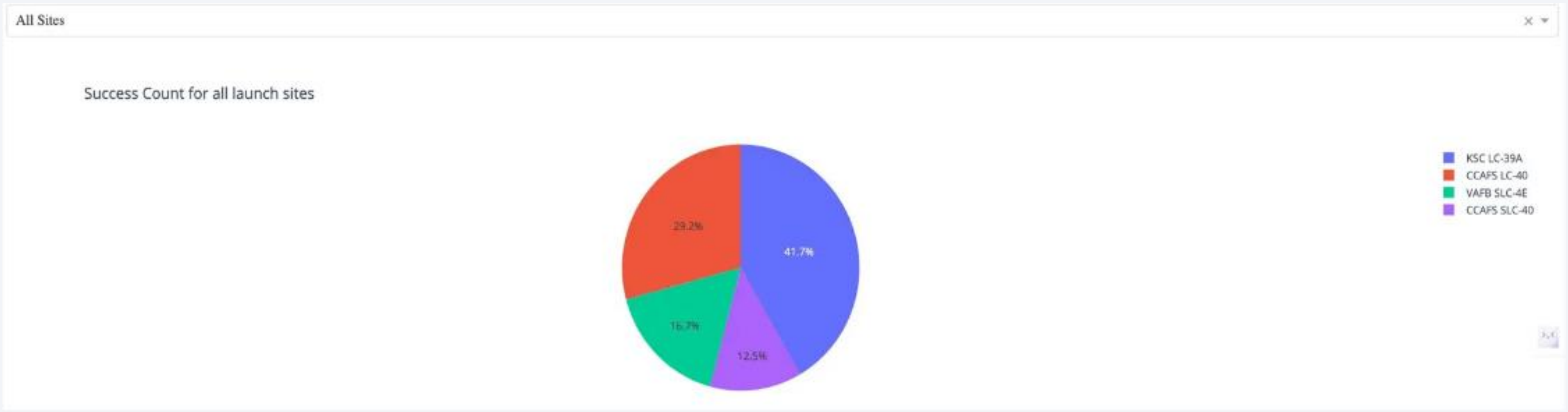


Section 4

Build a Dashboard with Plotly Dash

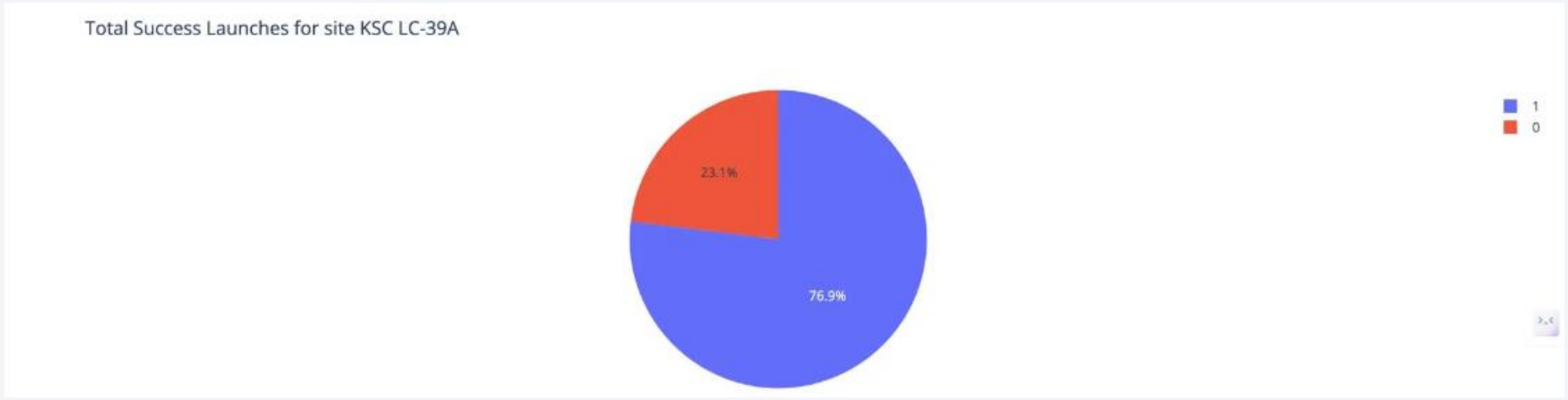
Total success launches for all sites

KSC LC-39A has the highest percentage of successfullaunches at 41.7% from the entire record, while CCAFS SLC-40 has the lowest percentage of successful launches at only 12.5%.



Success ratio of the launch site with the highest success launches

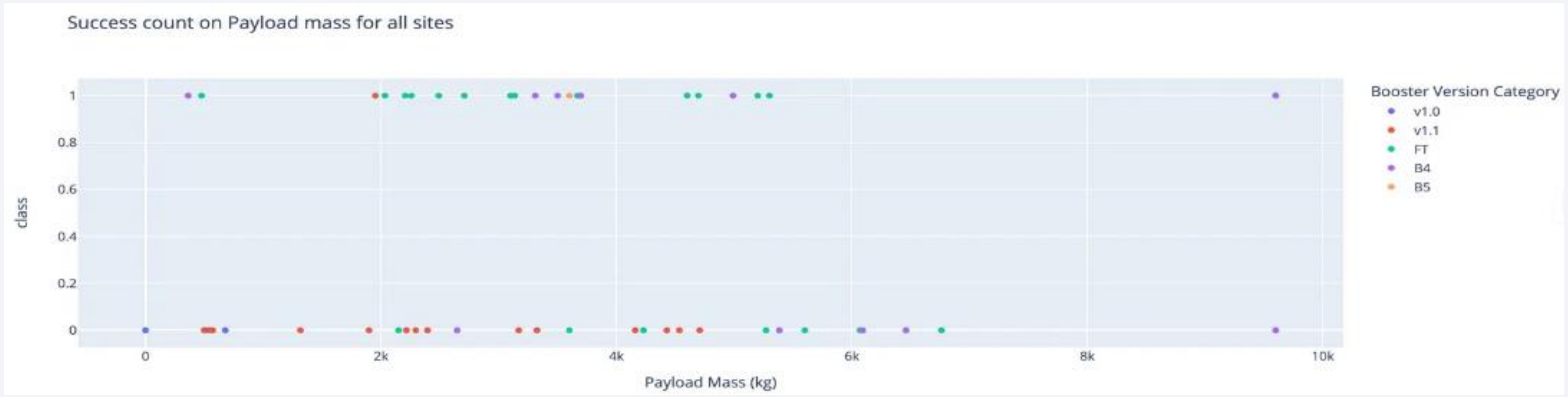
KSC LC-39A, the launch site with the highest number of successful launches, boasts a 76.9% success rate and a 23.1% failure rate for launches from its site.



Payload vs launch outcome

1.The payload range with the highest number of successful launches is between 2000 to 4000 kg, as evidenced by the largest number of occurrences in that range, followed by the payload range of 4000 to 6000 kg, which has the second highest number of occurrences.

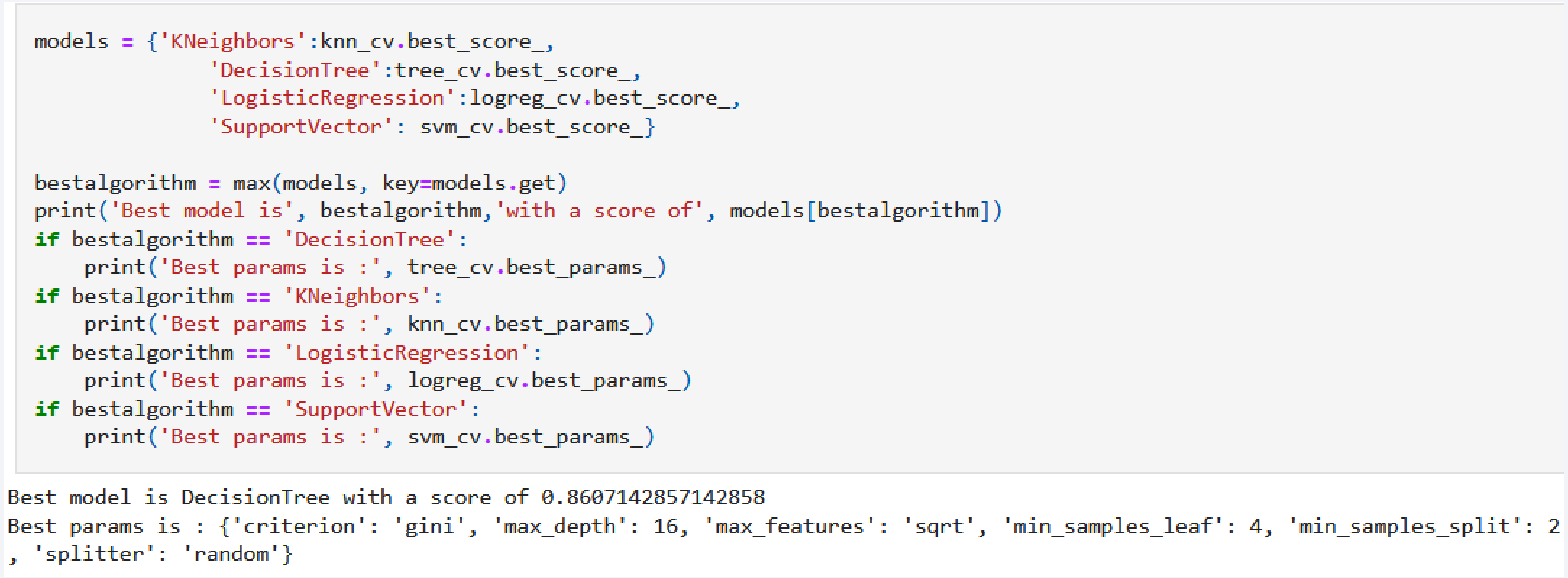
2.Booster version FT (green markers) has the highest number of successful launches, followed by B4 (purple markers) with the second highest number of successful launches among all booster versions.



Section 5

Predictive Analysis (Classification)

Classification Accuracy



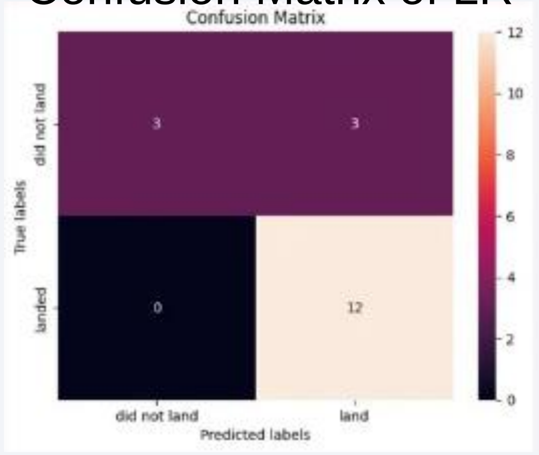
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8607142857142858
Best params is : {'criterion': 'gini', 'max_depth': 16, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 2
, 'splitter': 'random'}
```

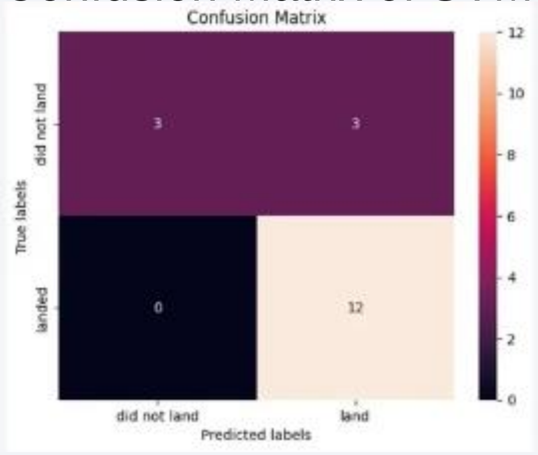
The most effective method is Decision Tree with a score of 0.8607. This method operates by recursively splitting the dataset based on features, enabling efficient prediction of target values for new data points.

Confusion Matrix

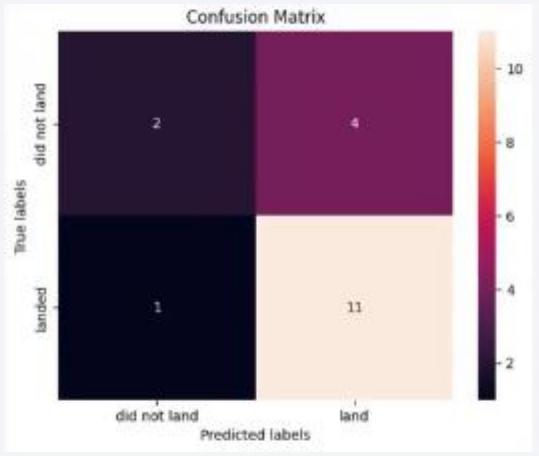
Confusion Matrix of LR



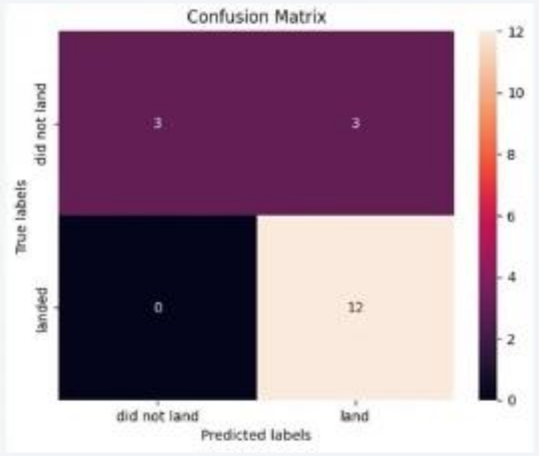
Confusion Matrix of SVM



Confusion Matrix of decision tree



Confusion Matrix of KNN



1. LR, SVM, KNN models perform well, as their confusion matrices show they correctly predicted all 12 successful landings without any errors.

2. However, the Decision Tree model correctly predicted 11 successful landings but incorrectly predicted one as a failure.

3. LR, SVM, and KNN models have the same accuracy of 83.33% as shown earlier, hence they share the same confusion matrix.

Conclusions

1. Logistic Regression (LR), Support Vector Machines (SVM), and K-Nearest Neighbors (KNN) are the leading models for predicting outcomes in this dataset.

2. Payloads with lower weight demonstrate better performance compared to heavier ones.

3. The probability of a SpaceX launch succeeding increases with accumulated years of experience, indicating a trend towards flawless launches over time.

4. Launch Complex 39A at Kennedy Space Center records the highest number of successful launches compared to other launch sites.

5. Orbit types GEO, HEO, SSO, and ES L1 exhibit the highest rates of successful launches.

Thank you!