IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Battogtokh
August 26, 2025

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context
- Problems you want to find answers

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected through the SpaceX API
- Perform data wrangling
  - Python Pandas and Numpy libraries to clean up the missing values.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Scikit Learn to predictive analysis

Data Collection

- Data collected from SpaceX API, and alternatively we scrapped the data from web page.
- Retrieved the data using the requests python library. The data further changed to include selected columns of data with desired cores and payload counts to limit to single core and payload rockets. The booster names, mass of payloads, launchpad and cores information are retrieved by calling to their respective endpoints.
- We also experimented the web scraping with beautiful soap python module.
- https://github.com/itoogii/ibm_ds/blob/main/1_jupyter-labs-spacex-data-collection-api.ipynb

Data Collection – SpaceX API

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

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

```
data.head()
```

	static_fire_date_utc	#
0	2006-03-17T00:00:00.000Z	
1	*Missing value*	
2	*Missing value*	
3	2008-09-20T00:00:00.000Z	

```
# Call getBoosterVersion
getBoosterVersion(data)
```

the list has now been update

```
BoosterVersion[0:5]
```

```
['Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 1', 'Falcon 9']
```

```
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}
```

Data Collection - Scraping

response = requests.get(static_url)
bs = BeautifulSoup(response.text, 'html.parser’)

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(bs.find_all('table',"wikitable plainrowheader
    # get table row
     for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
```

Data Wrangling

- In Data wrangling we did analyze the null data columns.
- The data wrangling was straightforward, and we filtered the data to exclude the ‘GTO’ from the ‘Orbit’ feature. We created the ‘Class’ feature from the successful and unsuccessful landing outcomes.
- https://github.com/itoogii/ibm_ds.git github repository.

EDA with Data Visualization

- We used the seaborn library to plot the scatter points, bar plot, and line graph. We created the scatter points to the detailed launch records, bar plot to success rates in every orbit, and line graph to show the success rates per year.
- https://github.com/itoogii/ibm_ds/blob/main/2_edadataviz.ipynb

EDA with SQL

- It used the Pandas library to read the data from the source and then wrote that to the sql database. After that we used the SQL alchemy to work with the sql directly in the jupyter notebook environment.
- https://github.com/itoogii/ibm_ds/blob/main/2_jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We added the markers of the Launch site locations and calculated the distance from the nearest shore, and then visualized that distance.
- We added those objects to mark the locations for visually represent the data on the map.
- https://github.com/itoogii/ibm_ds/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- We used the plotly library to add HTML elements and the corresponding decorator functions to trigger them on the changes of slider for the range of payload selection and dropdown element to choose the launch site for individual site data representation.
- On the web based interactive visualization, the plotly simply integrates the data to the web visualization. The web link can be shared with other users and stakeholders and let them access from anywhere on any device.
- https://github.com/itoogii/ibm_ds/blob/main/Dash/spacex-dash-app.py

Predictive Analysis (Classification)

- We used the grid search to find the best parameters with selected classifiers and parameters to search from.
- We used the training dataset to fit the model with logreg, svm, tree classifiers, and tested the performance with the test dataset.
- https://github.com/itoogii/ibm_ds/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

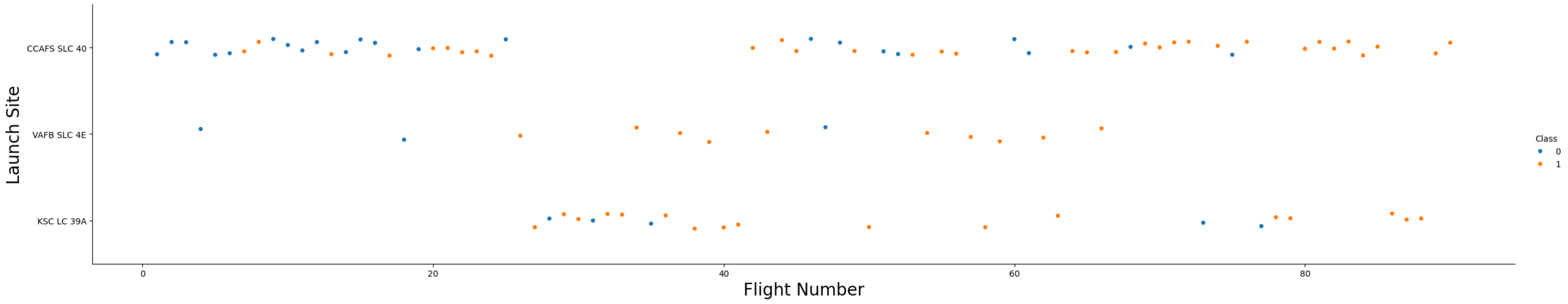
Results

- Exploratory Data analysis provides the insight into the dataset that we can gain understanding of successful or unsuccessful launches of SpaceX Falcon 9 spaceships.
- Interactive analysis with the Dash library facilitates simple web access to the data visualization.
- Our dataset was small, and it does not predict well.

Section 2

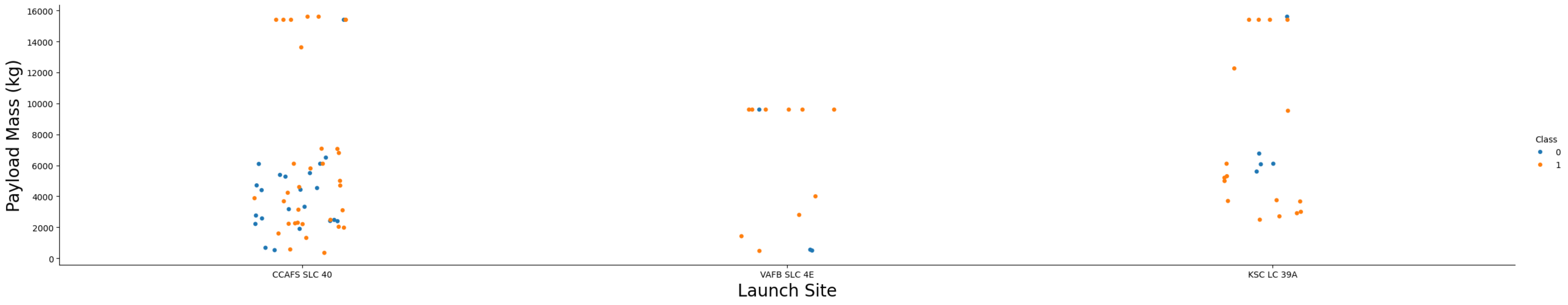
Insights drawn from EDA

Flight Number vs. Launch Site



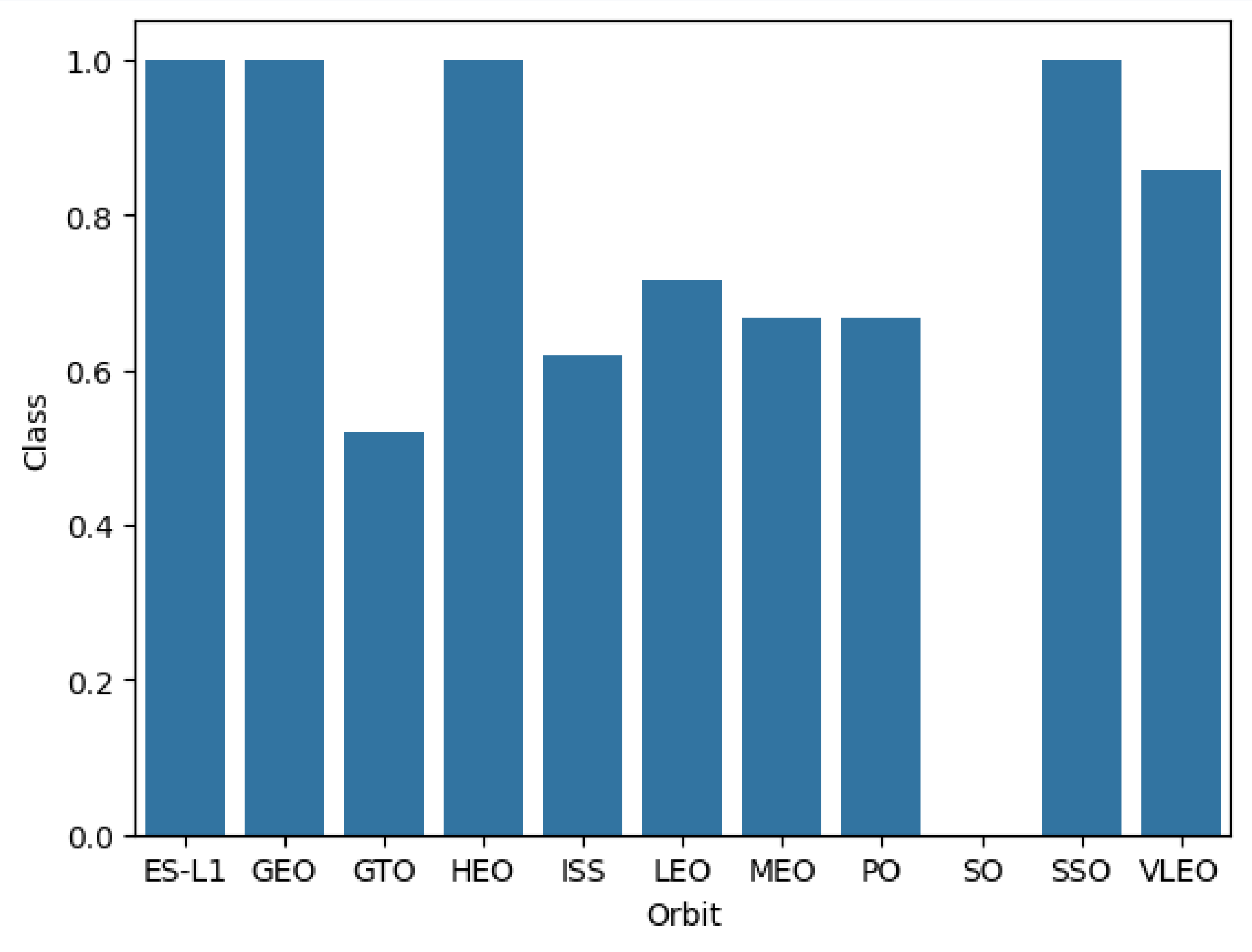
- Scatter plot of Flight Number vs. Launch Site
- 0 (unsuccessful) and 1 (successful)

Payload vs. Launch Site

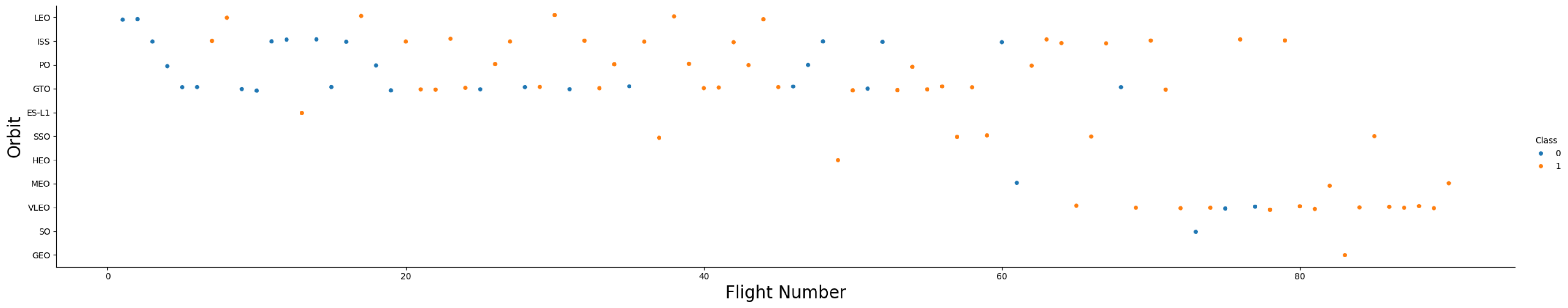


- scatter plot of Payload vs. Launch Site

Success Rate vs. Orbit Type

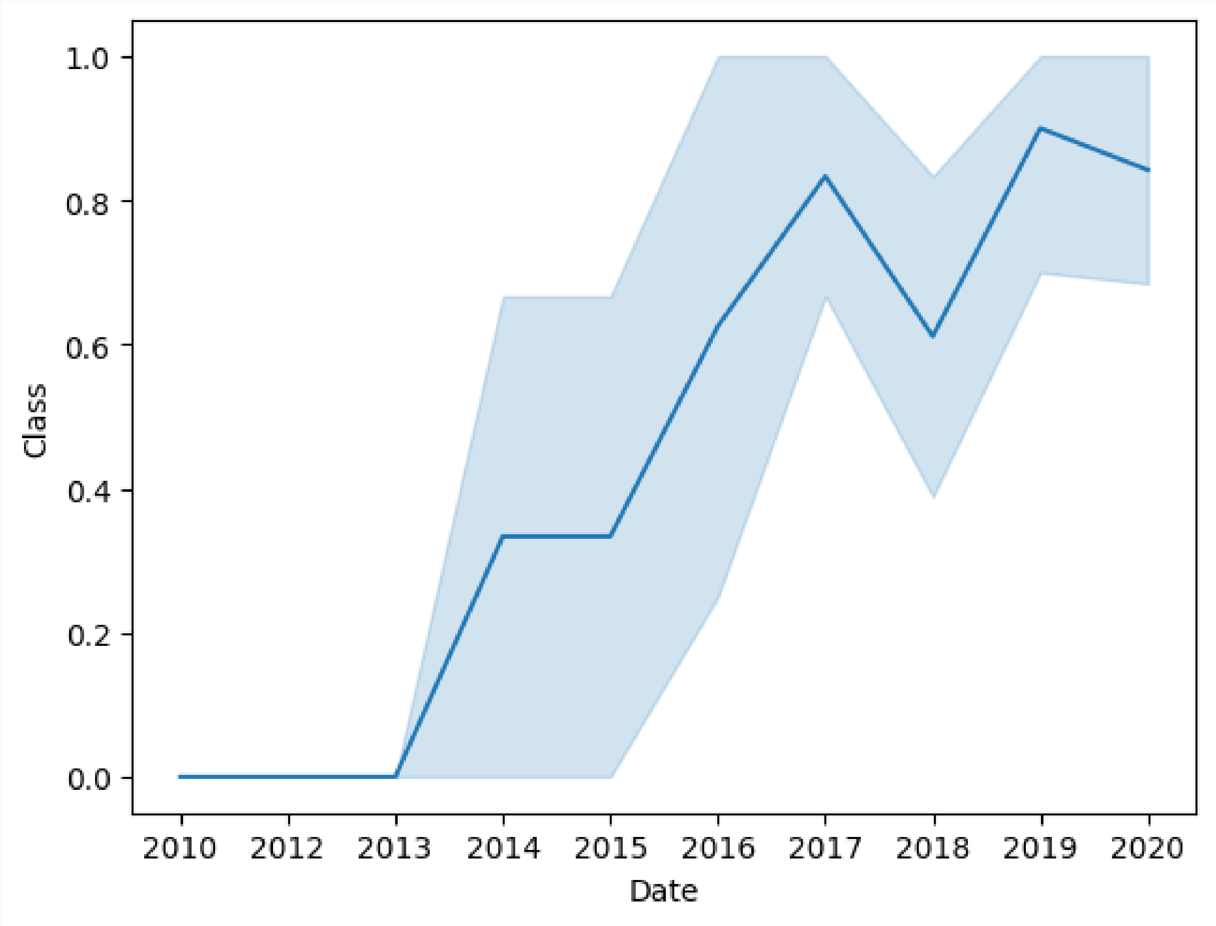


Flight Number vs. Orbit Type



Payload vs. Orbit Type

Launch Success Yearly Trend



All Launch Site Names

Using Pandas

```
spacex_df = pd.read_csv("spacex_launch_dash.csv")

spacex_df["Launch Site"].unique()
```

```
# Display the names of the unique launch sites in the space mission
%sql SELECT DISTINCT "Launch_Site" FROM SPACEXTABLE;
```

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

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

Launch Site Names Begin with 'CCA'

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

```
%sql SELECT * FROM SPACEXTABLE WHERE "Launch_Site" like 'CCA%' LIMIT 5;
```

Python

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

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

Total Payload Mass

```
%sql SELECT "Customer", SUM("PAYLOAD_MASS__KG_") as "TOTAL" FROM SPACEXTABLE
WHERE "Customer" = 'NASA (CRS)' GROUP BY "Customer" ;
```

Customer	TOTAL
NASA (CRS)	45596

Average Payload Mass by F9 v1.1

```
%sql SELECT "Booster_Version", AVG("PAYLOAD_MASS__KG_") AS Average_F9 FROM SPACEXTABLE
WHERE "Booster_Version" like 'F9 v1.1%' GROUP BY "Booster_Version";
```

Booster_Version	Average_F9
F9 v1.1	2928.4
F9 v1.1 B1003	500.0
F9 v1.1 B1010	2216.0
F9 v1.1 B1011	4428.0
F9 v1.1 B1012	2395.0
F9 v1.1 B1013	570.0
F9 v1.1 B1014	4159.0
F9 v1.1 B1015	1898.0
F9 v1.1 B1016	4707.0
F9 v1.1 B1017	553.0
F9 v1.1 B1018	1952.0

First Successful Ground Landing Date

```
%sql SELECT "Date" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)' Limit 1;
```

Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT "Booster_Version", "PAYLOAD_MASS__KG_" FROM SPACEXTABLE WHERE "Landing_Outcome" =
'Success (drone ship)' AND "PAYLOAD_MASS__KG_" BETWEEN 4000 AND 6000;
```

Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT COUNT(*) AS count, "Mission_Outcome" FROM SPACEXTABLE GROUP BY "Mission_Outcome";
```

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

Boosters Carried Maximum Payload

```
%sql SELECT "Booster_Version", "PAYLOAD_MASS__KG_" FROM SPACEXTABLE WHERE
"PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE);
```

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

```
%sql SELECT CASE substr("Date", 6, 2) \
   WHEN '01' THEN 'January' \
   WHEN '02' THEN 'February' \
   WHEN '03' THEN 'March' \
   WHEN '04' THEN 'April' \
   WHEN '05' THEN 'May' \
   WHEN '06' THEN 'June' \
   WHEN '07' THEN 'July' \
   WHEN '08' THEN 'August' \
   WHEN '09' THEN 'September' \
   WHEN '10' THEN 'October' \
   WHEN '11' THEN 'November' \
   WHEN '12' THEN 'December' \
   ELSE substr("Date", 6, 2) \
   END AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site" \
FROM SPACEXTABLE \
WHERE substr("Date", 0, 5) = '2015'\
   AND "Landing_Outcome" = "Failure (drone ship)";
```

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

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

```
%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 count DESC;
```

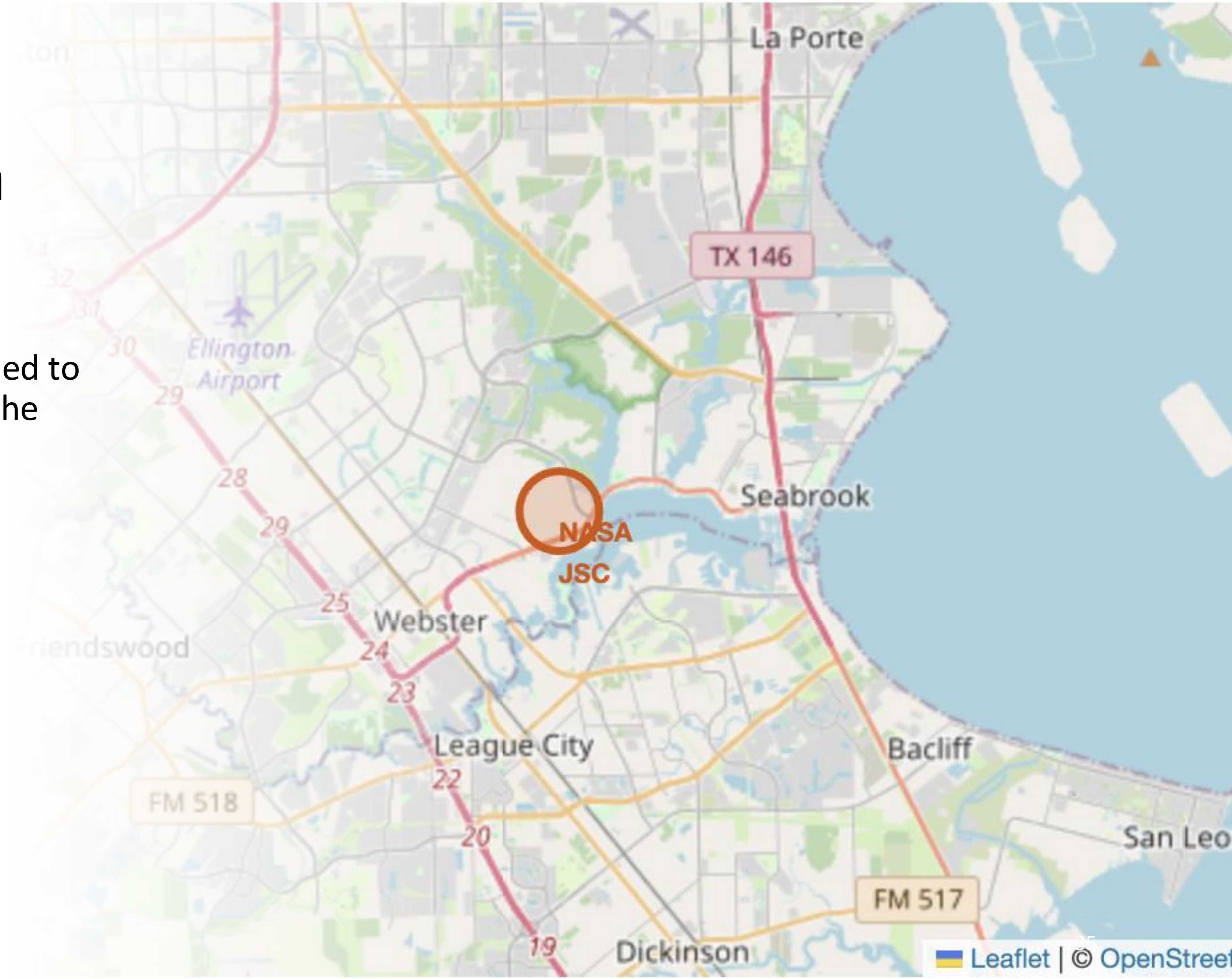
Landing_Outcome	count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

NASA location

- Folium circle marker is used to highlight the location of the NASA

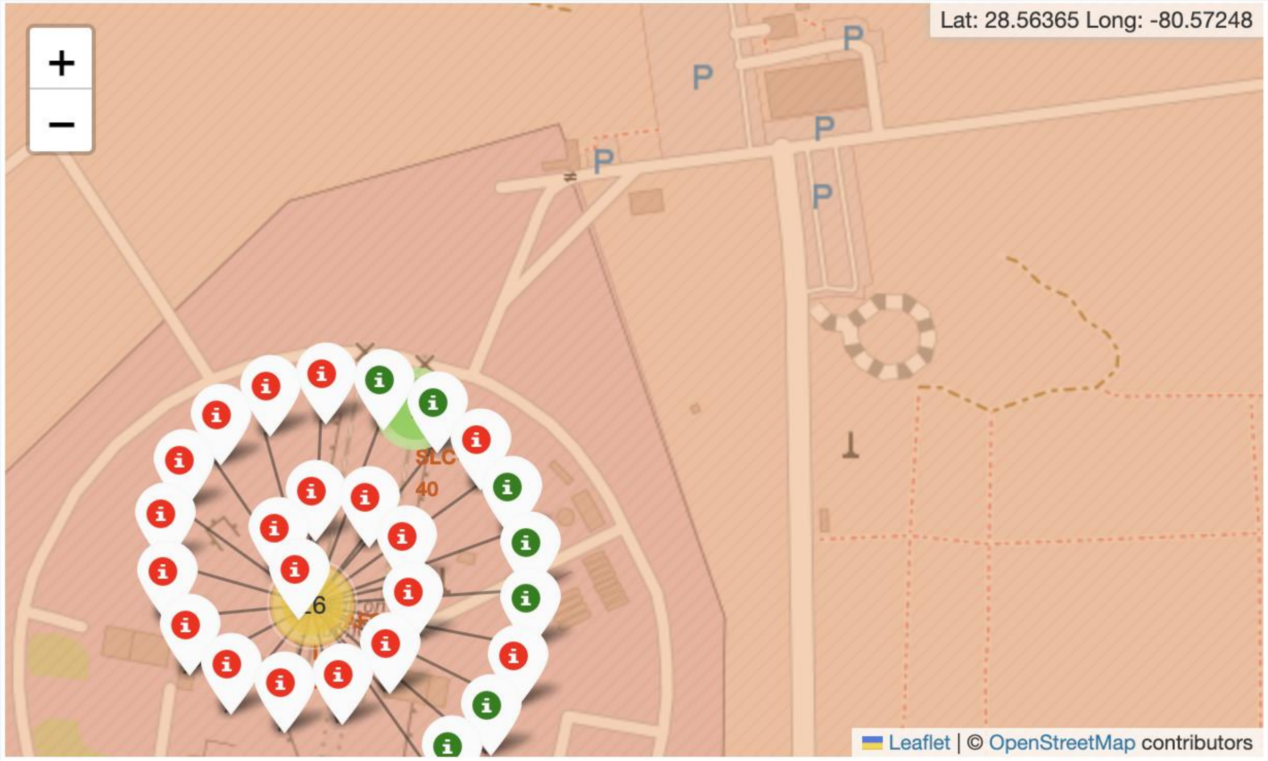


Launch sites

- Launch sites are marked on the map

Color labeled Markers

Explain the important elements and findings on the screenshot

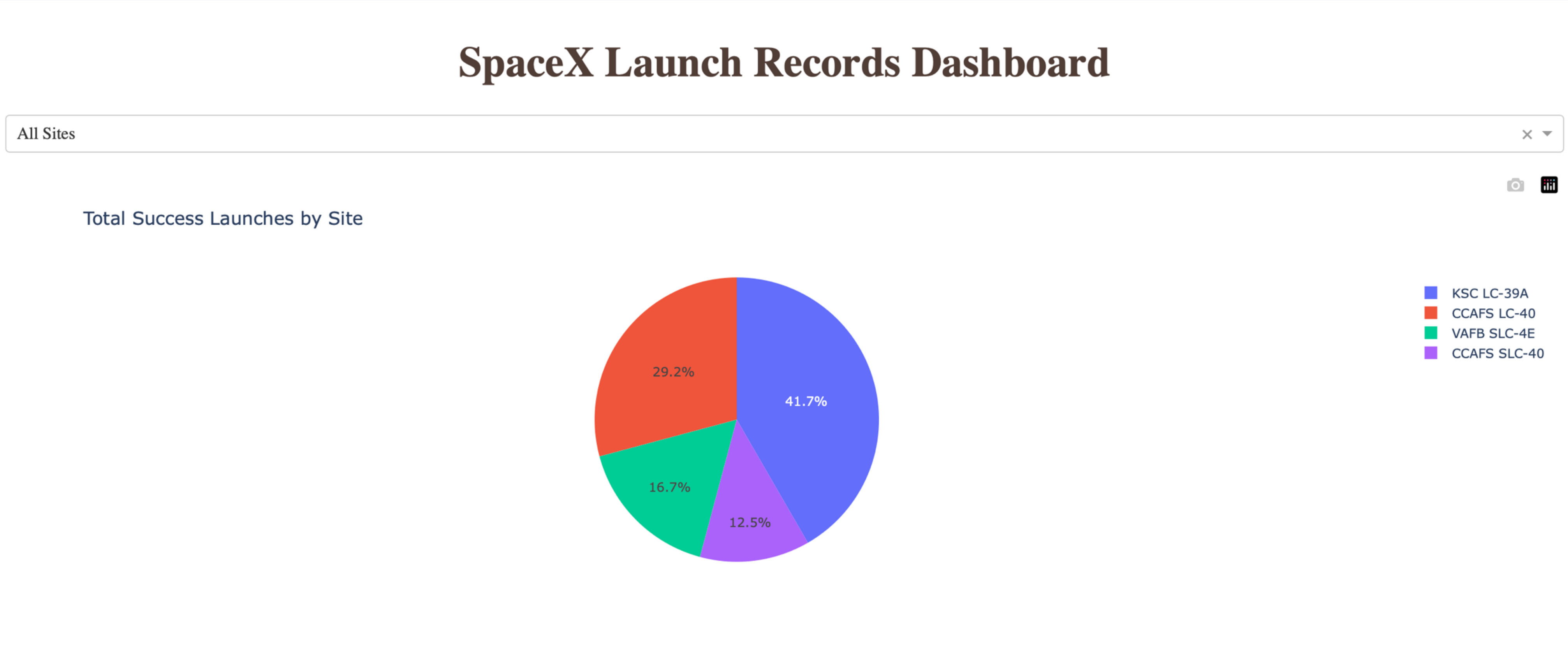


Section 4

Build a Dashboard with Plotly Dash

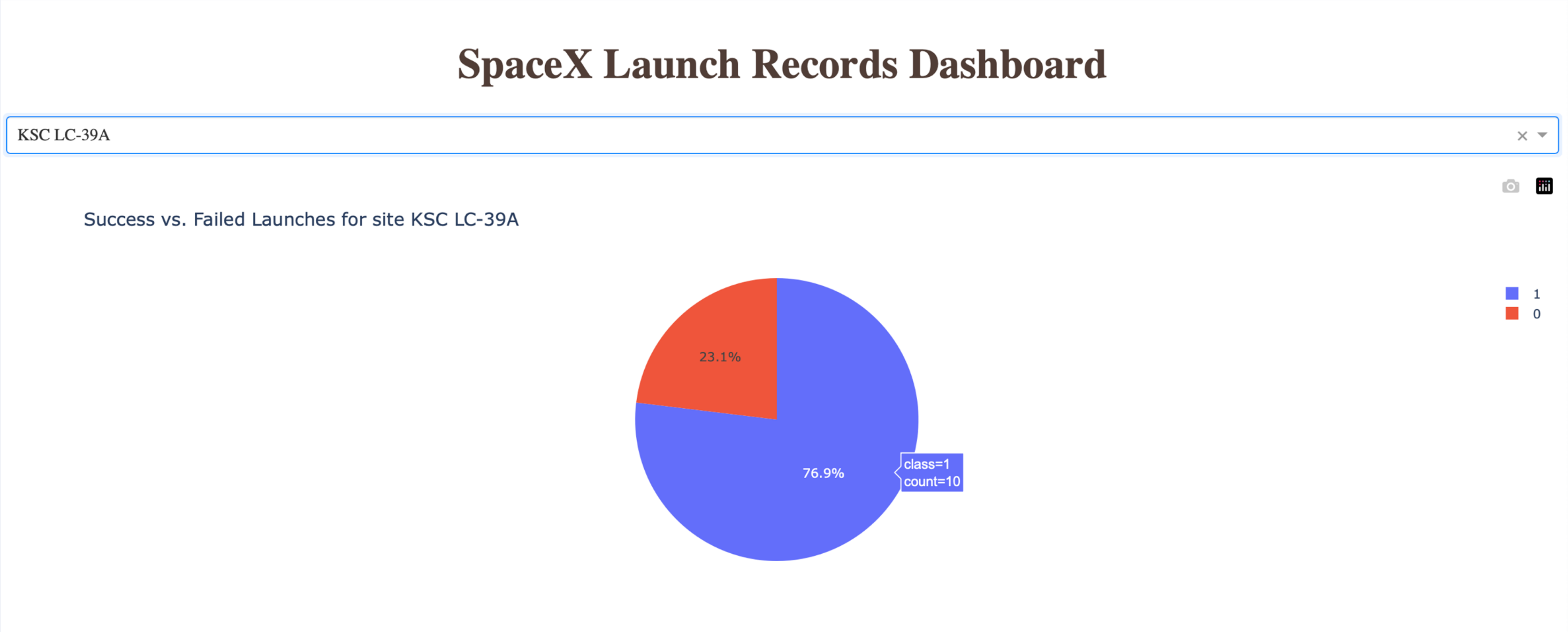
Launch Record Dashboard

Success ratio of total launches



Dashboard Pie chart

Highest launch success rate site info

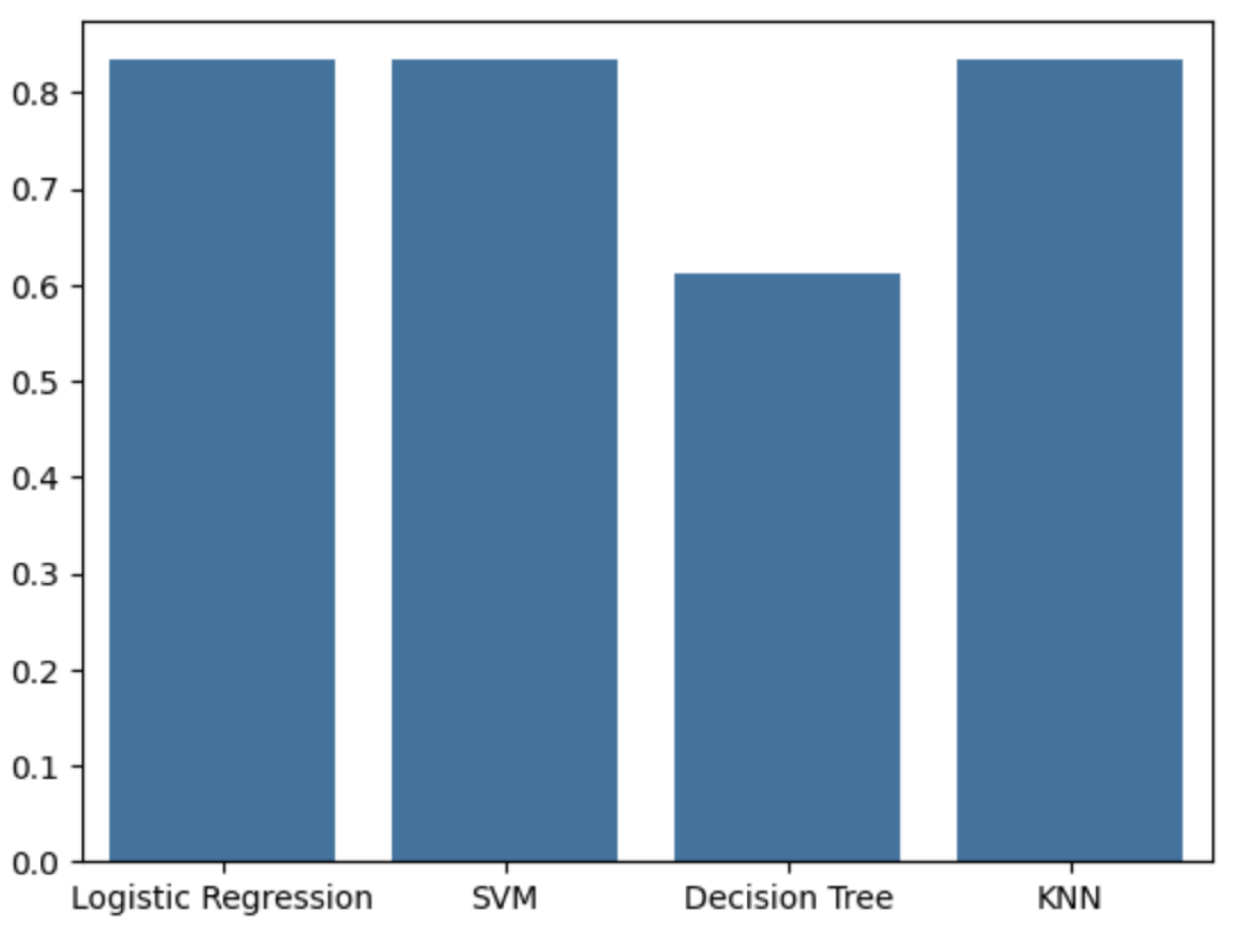


Payload and Success rate

Section 5

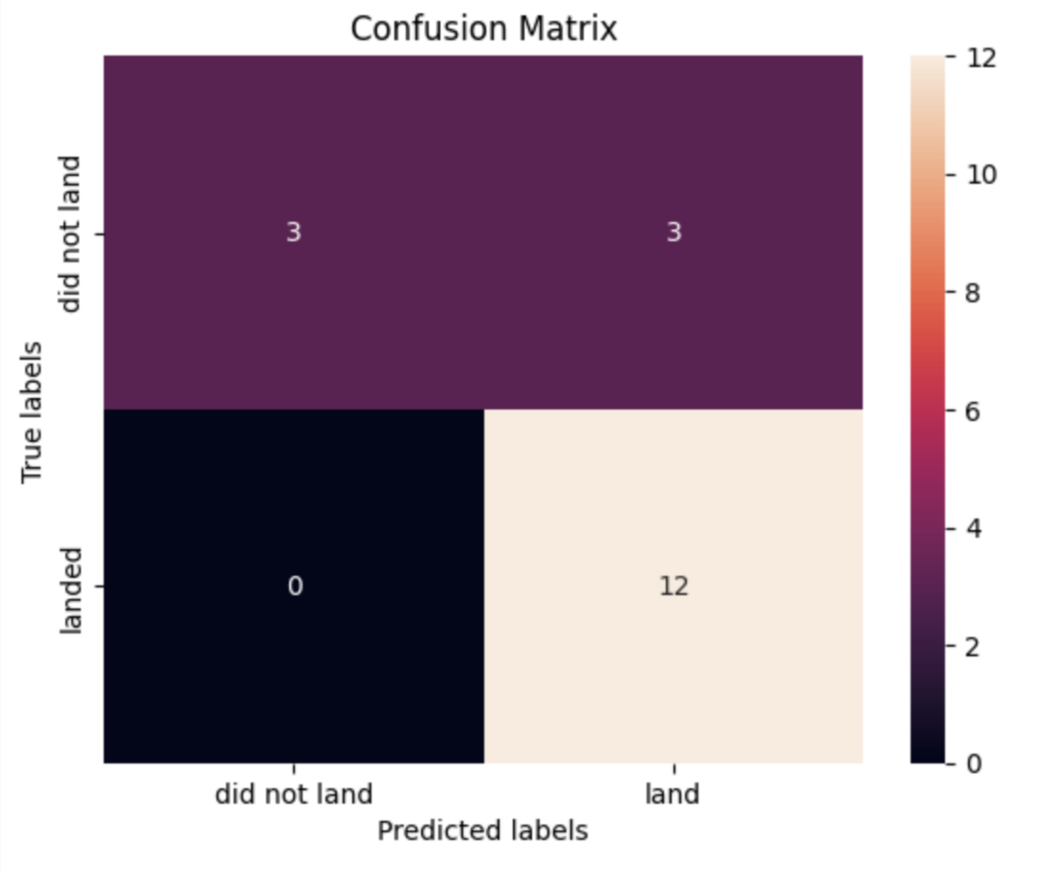
Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix

- Logistic regression and SVM classifications had the same result



Conclusions

- Data collection using web scrapping and API requests
- Data wrangling using Pandas and numpy libraries
- Data Visualization using seaborn and matplotlib
- Data analysis using SQL queries
- Web Visualization using Dash
- Map visualization using Folium
- Machine learning with scikit-learn

Appendix

- IBM Data Science Capstone project
- The work files are stored in https://github.com/itoogii/ibm_ds/tree/main

Thank you!