Winning Space Race with Data Science

Gang Wu
2022-12-11

Outline

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

Executive Summary

- In this report, we used a series of data science steps to predict if the SpaceX Falcon 9 first stage will land successfully. The data science steps includes: data collection, data wrangling, exploratory data analysis, interactive visual analytics, and finally prediction of the success rate using several machine learning models.
- We find that the machine-learning models can be built to determine the factors affecting the successful landing chance of the first stage, and even predict the success chance if the launching conditions are given.
- We hope that with the developed machine learning model, we can determine the cost of a launch, which can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Introduction

- SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.
- If we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.
- In this project, we will create a machine learning pipeline to predict if the first stage will land successfully.
  - ✓ Which factors determine if the first stage 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:
  - Fetching the historical data of the launches using the SpaceX API.
  - Web scraping records from Wikipedia.
- Perform data wrangling
  - In the data set, there are several different cases where the booster did not land successfully. We convert those outcomes into Training Labels with one-hot encoding.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - After standardizing the data, we split the data into training and test datasets.
  - Then, we fit the model with SVM, Classification Trees and Logistic Regression, and tuning the best Hyperparameter for the models.

Data Collection

- Two methods were used to collect the data
  - Data collection was done using get request to the SpaceX API.
    - 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.
  - Web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup was also performed.
    - The objective was extract to launch records as HTML table, then we parsed the table and converted 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 to the notebook is https://github.com/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project/blob/main/L1.1-Data%20Collection%20API%20Lab.ipynb

1. Get request for rocket launch data using API

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]: response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

```
In [16]: # Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

3. Perform data cleaning and filling in the missing values
   a. Filter the dataframe to only include Falcon 9 launches

```
In [32]: # Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data2[data2['BoosterVersion']!='Falcon 1']
data_falcon9.head()
```

   b. Dealing with Missing Values

```
In [40]: # Calculate the mean value of PayloadMass column
Mean_PM = data_falcon9.PayloadMass.mean()
# Replace the np.nan values with its mean value
data_falcon9.PayloadMass.replace(np.nan, Mean_PM, inplace=True)
```

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/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project/blob/main/L1.2-Data%20Collection%20with%20Web%20Scraping.ipynb

1. Apply HTTP Get method to request the Falcon 9 rocket launch page

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

```
In [5]: # use requests.get() method with the provided static_url
        # assign the response to a object
        html = requests.get(static_url).content
```

2. Create a BeautifulSoup object from the HTML response

```
In [6]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content

        soup = BeautifulSoup(html, 'html.parser', from_encoding='utf-8')
```

3. Extract all column names from the HTML table header

```
In [10]: 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
         tags = first_launch_table.find_all('th')

         for tag in tags:
             name = extract_column_from_header(tag)
             if name is not None and len(name) > 0:
                 column_names.append(name)
```

4. Create a dataframe by passing the launch HTML tables

```
In [13]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
            # get table row
             for rows in table.find_all("tr"):
                 #check to see if first table heading is as number corresponding to launch a number
                 if rows.th:
                     if rows.th.string:
                         flight_number=rows.th.string.strip()
                         flag=flight_number.isdigit()
                 else:
                     flag=False
```

5. Export data to CSV

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/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project/blob/main/L1.3-Data%20Wrangling.ipynb

1. Exploratory data analysis and determined the training labels.

```
In [4]: df.dtypes
```

2. Calculated the number of launches at each site, and the number and occurrence of each orbit.

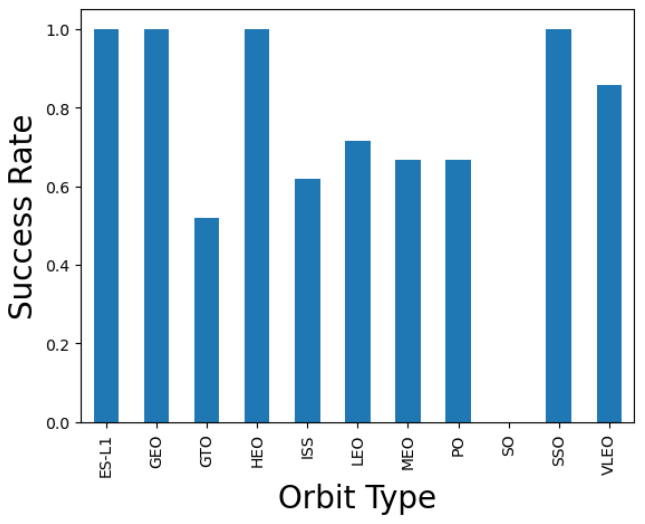
```
In [5]: # Apply value_counts() on column LaunchSite
        df['LaunchSite'].value_counts()

In [6]: # Apply value_counts on Orbit column
        df['Orbit'].value_counts()
```

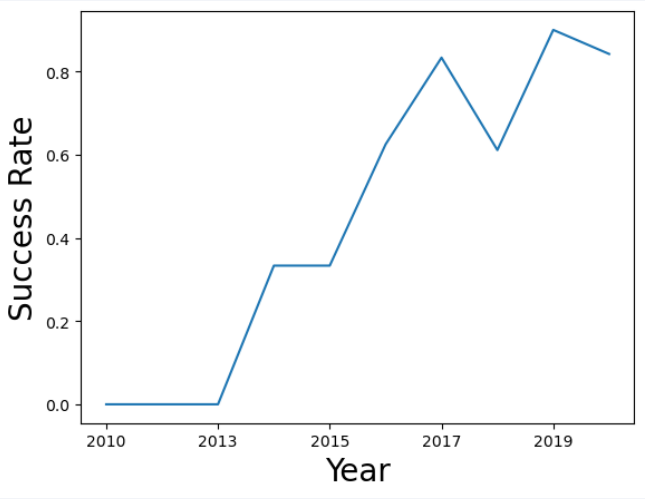
3. Created landing outcome label from outcome column and exported the results to csv.

```
In [7]: # landing_outcomes = values on Outcome column
        landing_outcomes = df['Outcome'].value_counts()
        print(landing_outcomes)
```

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 and orbit type, the launch success yearly trend.
- Bar chart compares the measure of categorical dimension, while line chart indicates trends and developments of numeric data over time. Scatter plot is commonly applied to identify regression type of relationships such as linear regression, logistic regression etc. It also provides a robust analysis of the correlation significance.
- The link to the notebook is https://github.com/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project/blob/main/L2.2-EDA%20with%20Visualization.ipynb





EDA with SQL

- We loaded the SpaceX dataset into a Db2 database.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of the unique launch sites in the space mission.
  - The records where launch sites begin with the string 'CCA'
  - 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.
- The link to the notebook is https://github.com/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project/blob/main/L2.1-EDA%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 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.
- The link to the notebook is https://github.com/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project/blob/main/L3.1-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.
- The link to the code is https://github.com/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project/blob/main/L3.2-spacex_dash_app.py

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- 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.
- The link to the notebook is https://github.com/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project/blob/main/L4-Machine%20Learning%20Prediction.ipynb

Results

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

Section 2

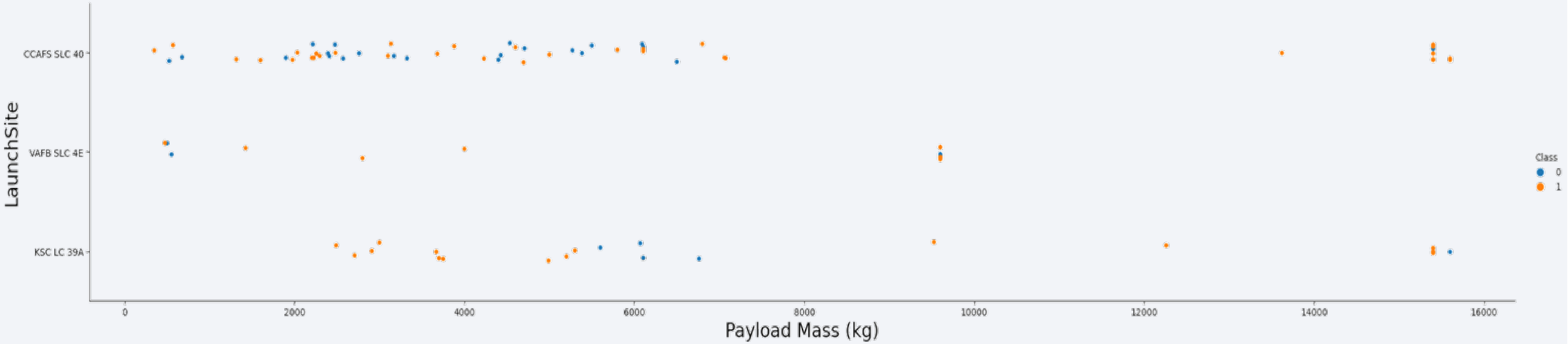
Insights drawn from EDA

Flight Number vs. Launch Site

- The larger the flight amount at a launch site, the greater the success rate at a launch site.

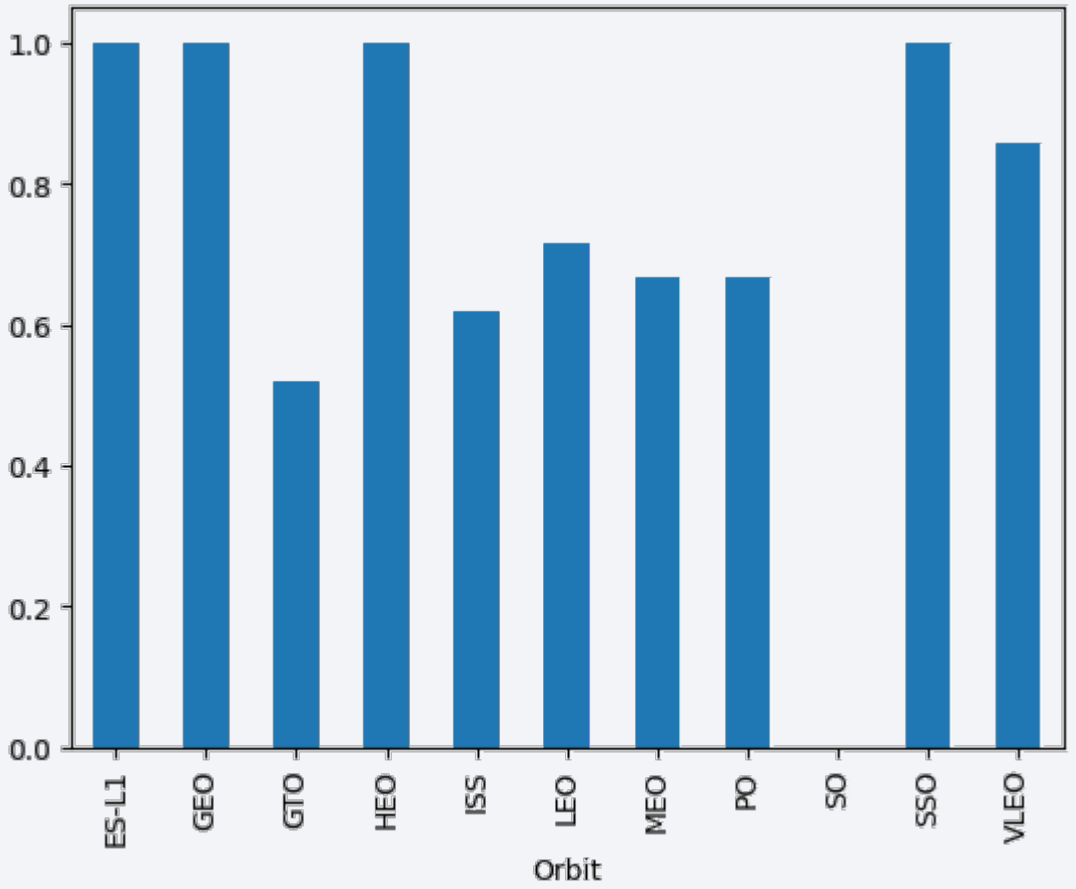
Payload vs. Launch Site

- The VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).
- For launch site CCAFS SLC 40, the greater the payload mass, the higher the success rate.



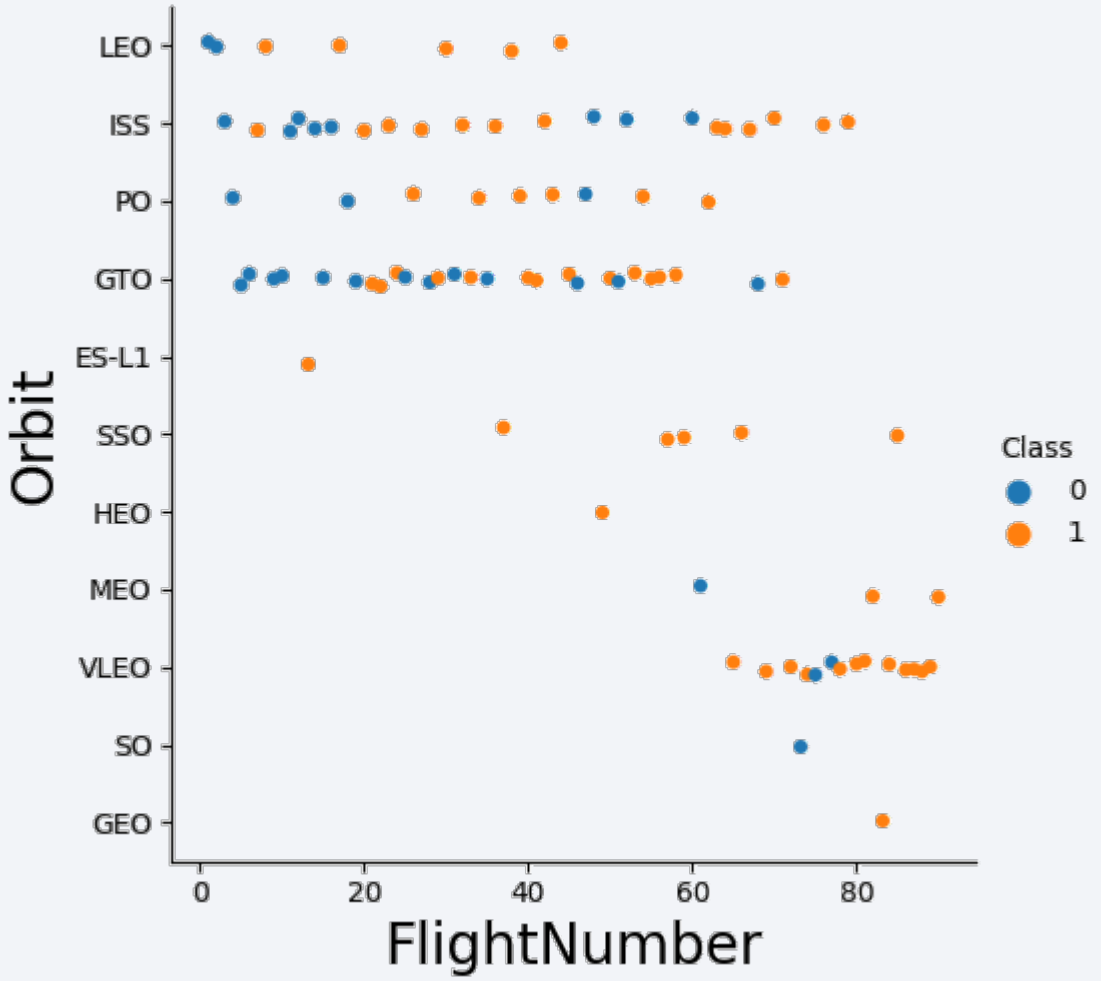
Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO, VLEO orbits have the highest success rate.



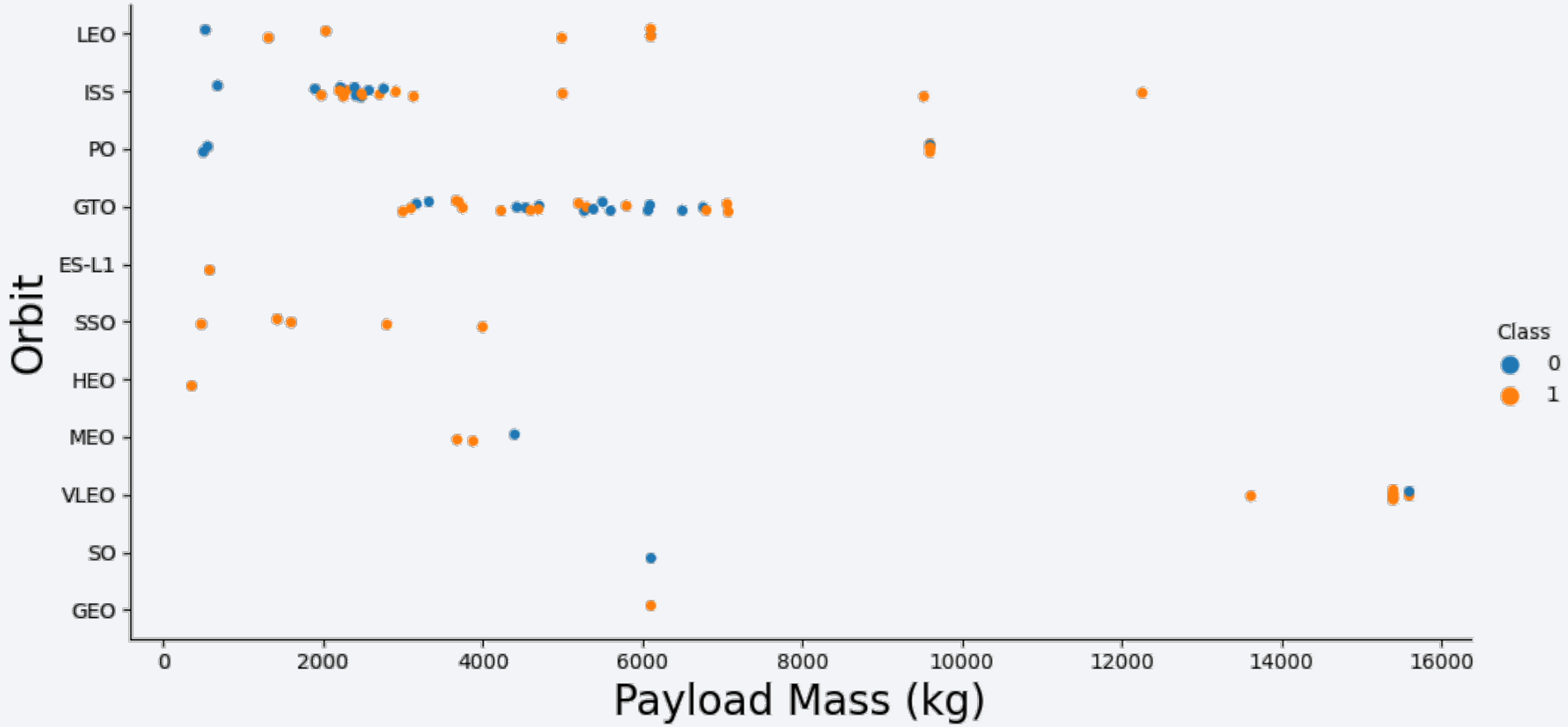
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type.
- For the LEO orbit, success chance is related to the number of flights.
- For the GTO orbit, there is no relationship between flight number and the orbit.



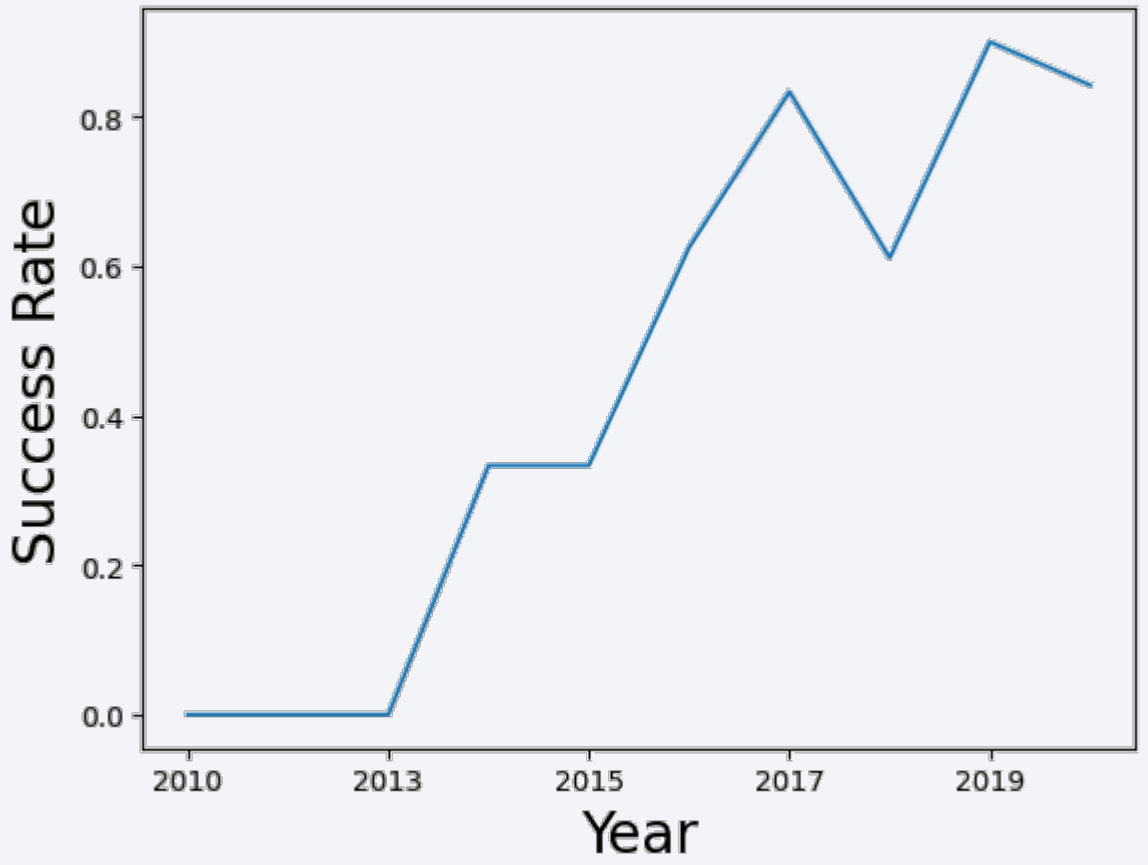
Payload vs. Orbit Type

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



Launch Success Yearly Trend

- Since 2013, success rate kept increasing till 2020.



All Launch Site Names

- We used the keyword **DISTINCT** to show only unique launch sites from the SpaceX data.

In [5]:

```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACEXTBL;
```

```
 * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a6c3e6
Done.
```

Out[5]:

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

Launch Site Names Begin with ‘CCA'

- We used the query below to display 5 records where launch sites begin with `CCA`

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

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- We calculated the total payload carried by boosters from NASA as 45596 using the query below

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

In [10]:

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE Customer = 'NASA (CRS)';
```

```
 * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g
Done.
```

Out[10]:

1
45596

Average Payload Mass by F9 v1.1

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

```
In  [11]: %%sql
          SELECT AVG(PAYLOAD_MASS__KG_)
          FROM SPACEXTBL
          WHERE Booster_Version LIKE 'F9 v1.1%';

           * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a
          Done.

Out[11]:
          1
          2928.4
```

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

In [30]:

```
%%sql
SELECT MIN(Date)
FROM SPACEXTBL
WHERE Landing__Outcome = 'Success (ground pad)';
```

```
 * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd
Done.
```

Out[30]:

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

In [33]:

```
%%sql
SELECT BOOSTER_VERSION
FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (drone ship)'
    AND 4000 < PAYLOAD_MASS__KG_ < 6000;
```

```
 * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0nqnrk39u98g.databases.appd
Done.
```

Out[33]:

booster_version
F9 FT B1021.1
F9 FT B1023.1
F9 FT B1029.2
F9 FT B1038.1
F9 B4 B1042.1
F9 B4 B1045.1
F9 B5 B1046.1

Total Number of Successful and Failure Mission Outcomes

- We used wildcard like '%' to filter for **WHERE** MissionOutcome was a success or a failure.

In [44]:

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

```
 * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd
Done.
```

Out[44]:

mission_outcome	total_number
Failure (in flight)	1
Success	99
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.

In [57]:

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

```
 * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cm
Done.
```

Out[57]:

booster_version
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

2015 Launch Records

- We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

In [62]:

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

```
 * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3
Done.
```

Out[62]:

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **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 outcome in descending order.

In [70]:

```
%%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
```

```
 * ibm_db_sa://ycp17474:***@2f3279a5-73d1-4859-88f0-a6c3e6b4b907.c3n41cmd0n
Done.
```

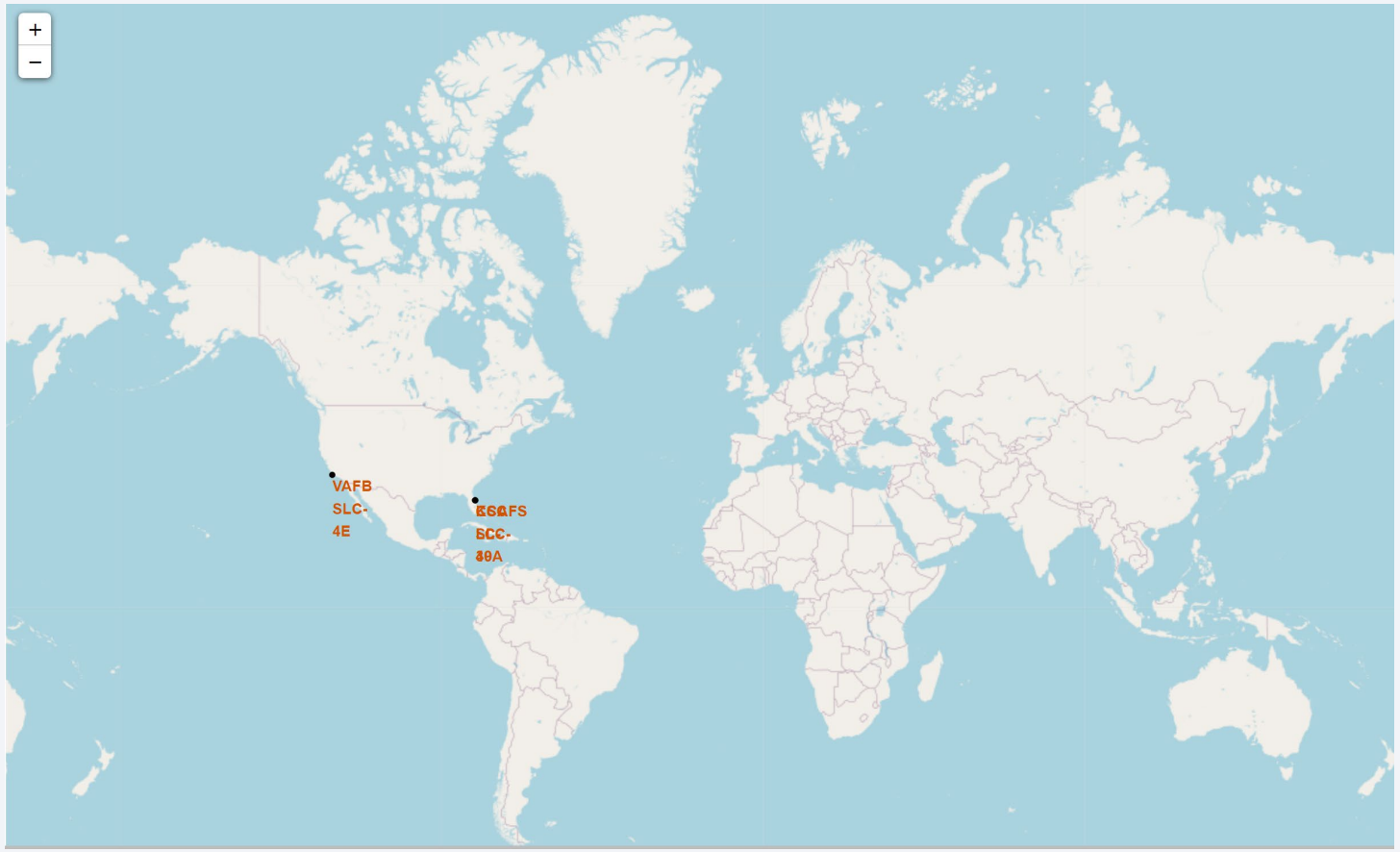
Out[70]:

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


Section 3
Launch Sites
Proximities Analysis

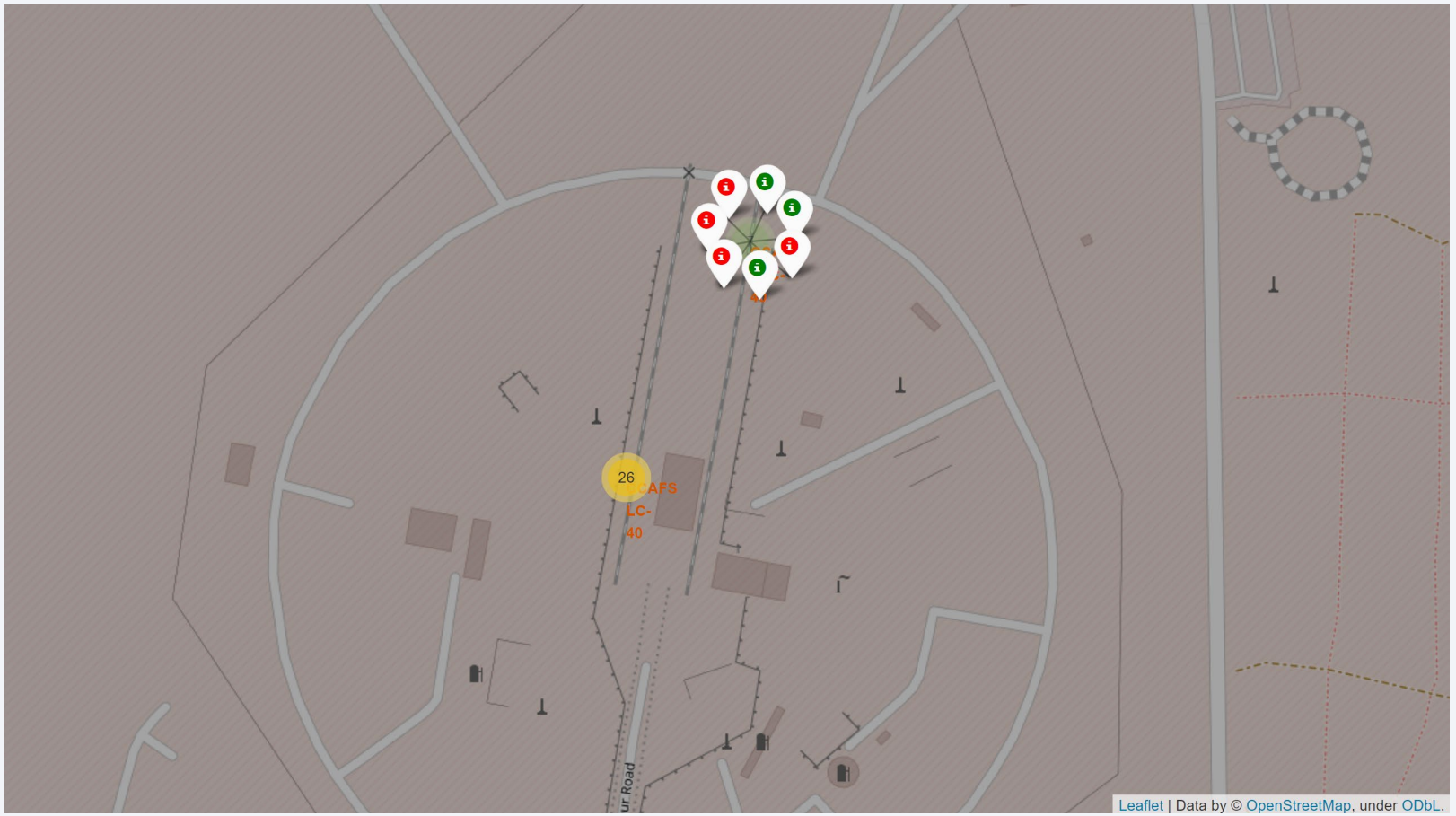
All launch sites on a global map

- All the SpaceX launch sites are in proximity to the Equator line to gain an additional boost to the rocket, due to the rotational speed of Earth.
- They are also in very close proximity to the coast, so that debris will not cause damage to the city if anything goes wrong. Also some rockets can be too big to transport via rail or road, then they can be transported using ships.



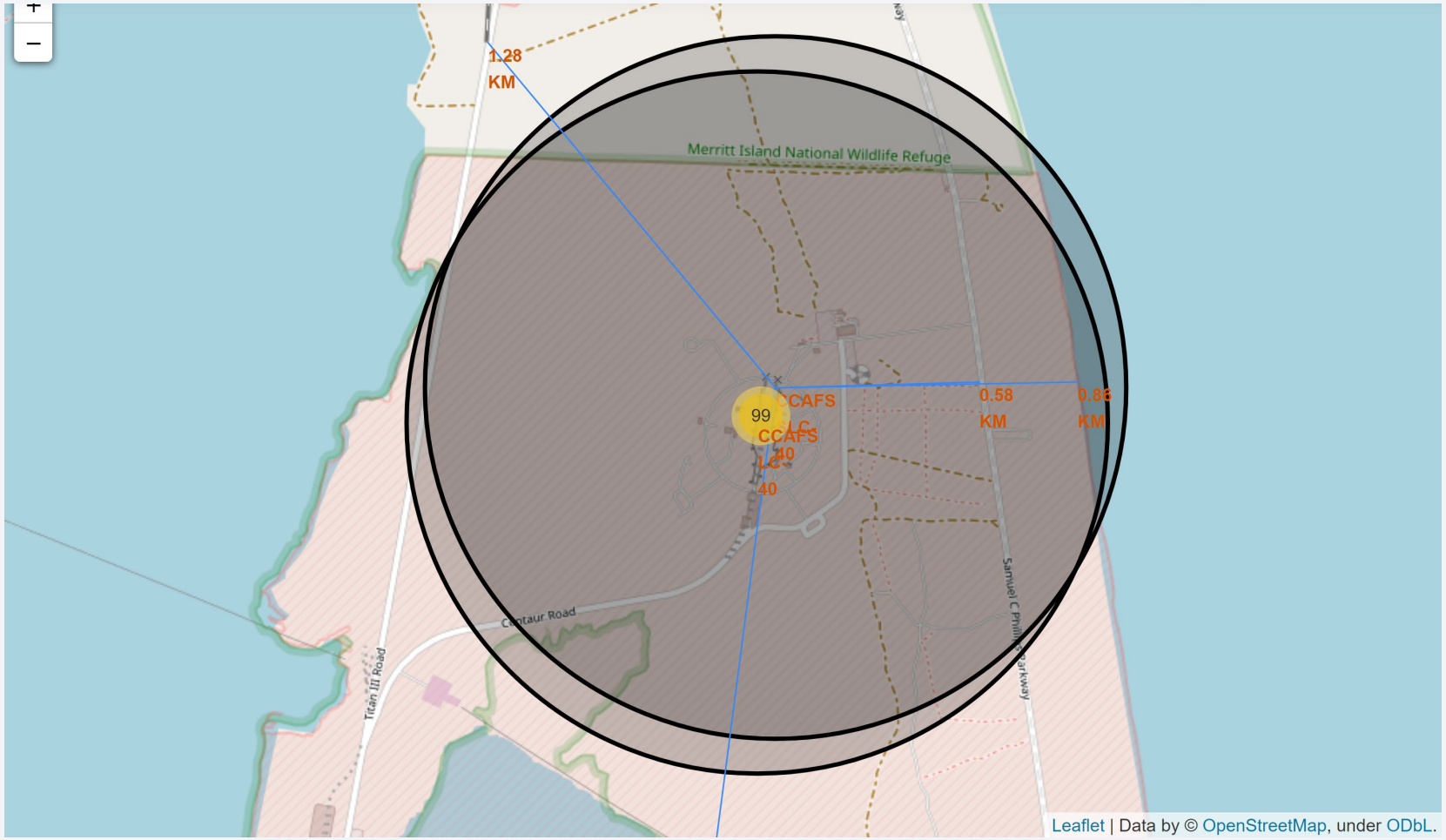
Markers showing launch sites with color labels

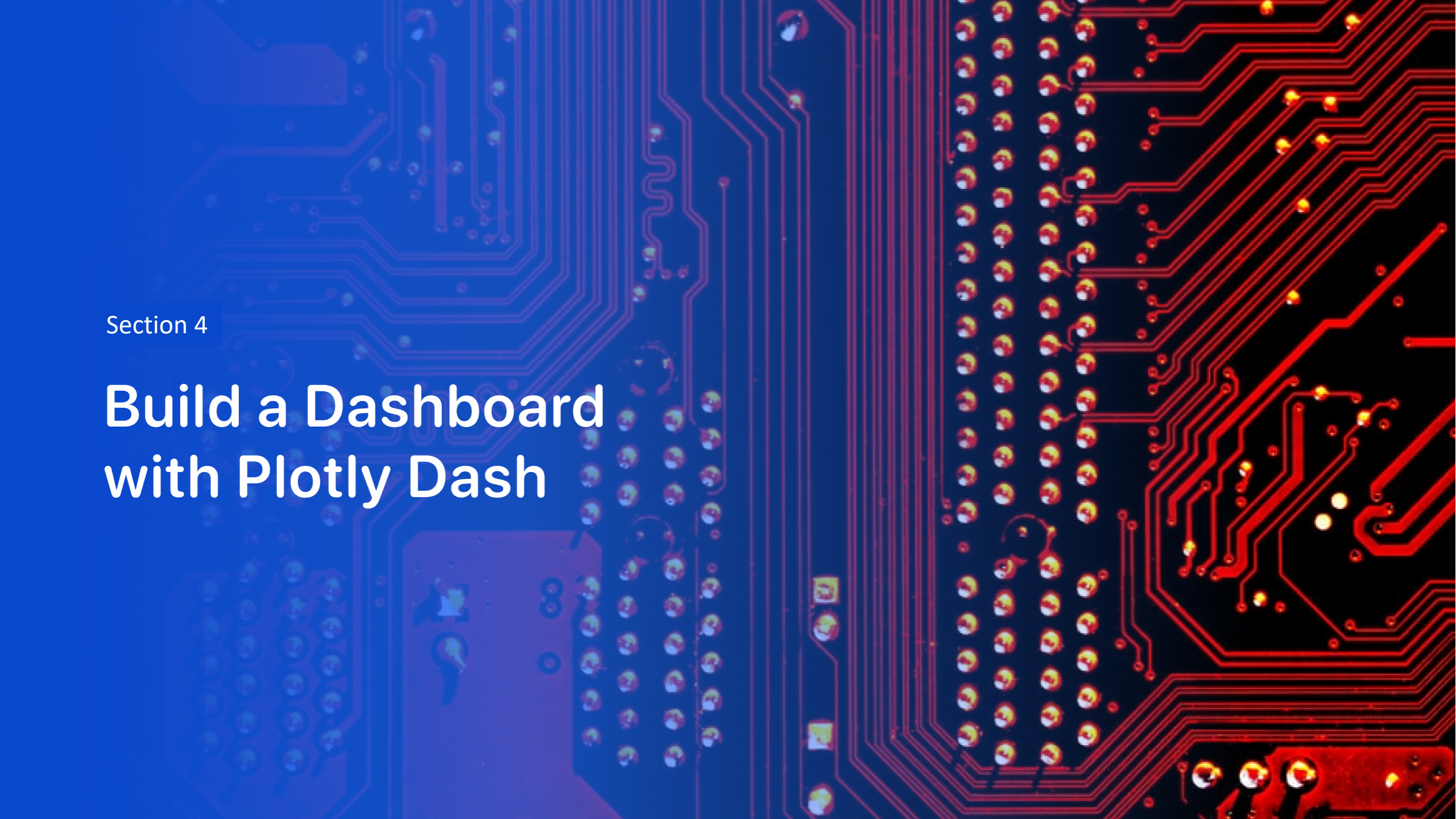
- Green Marker shows successful launches and Red Marker shows failures.



Launch Site distance to landmarks

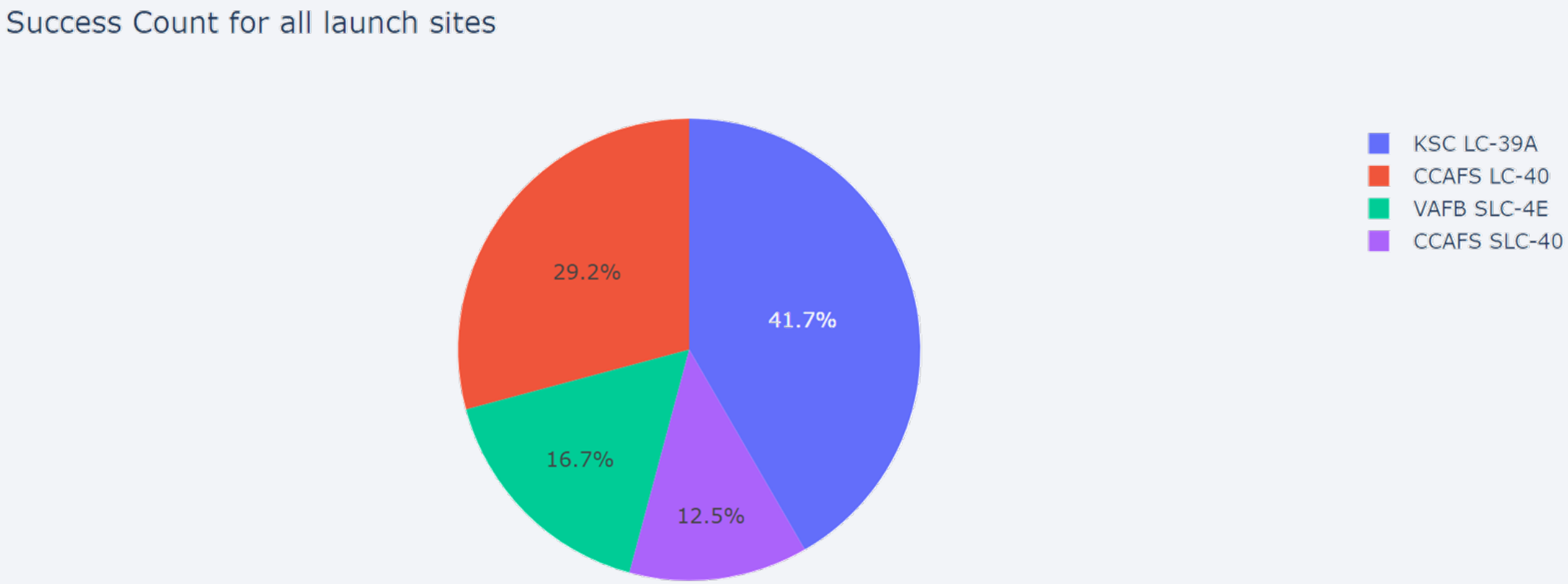
- If possible, launch sites are in close proximity to railways, highways, or coastline, so that the rockets or spacecraft can be transported to the launch site easily.
- Meanwhile, launch sites are usually far away from cities, so that debris will not cause damage to the city if anything goes wrong




Section 4
Build a Dashboard with Plotly Dash

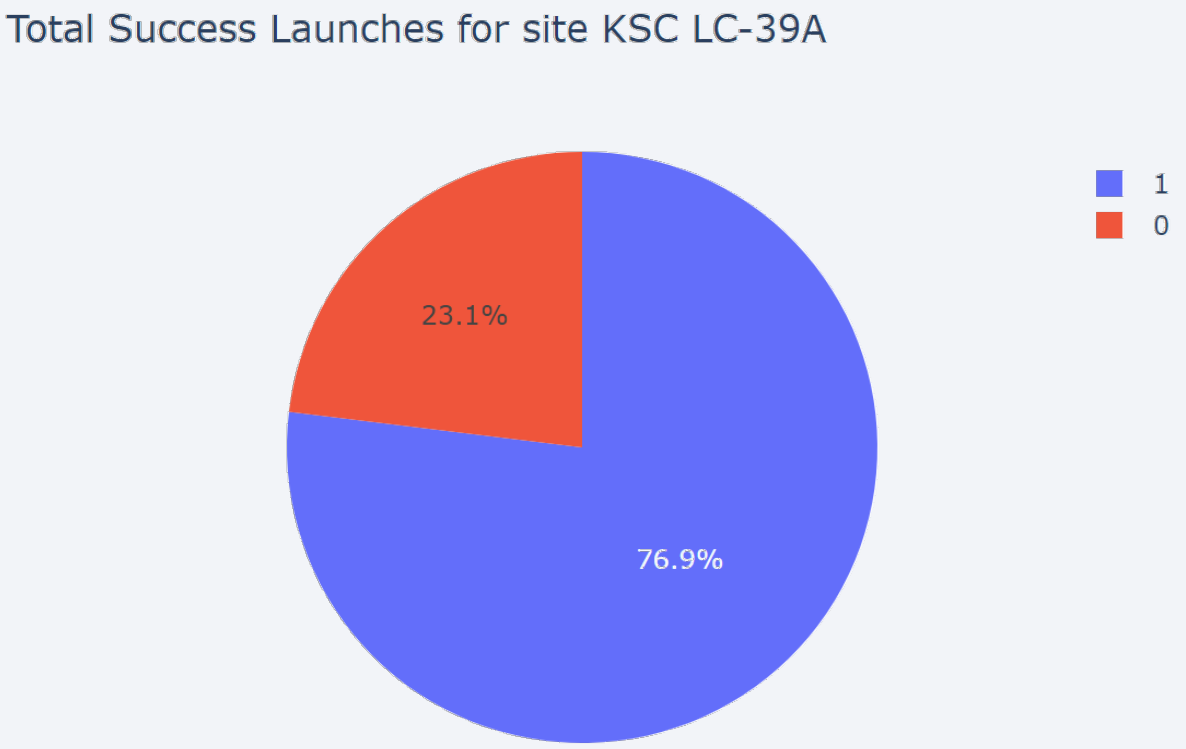
Success ratio achieved by each launch site

- KSC LC-39A has the highest percentage of successful launches among all the launch sites.

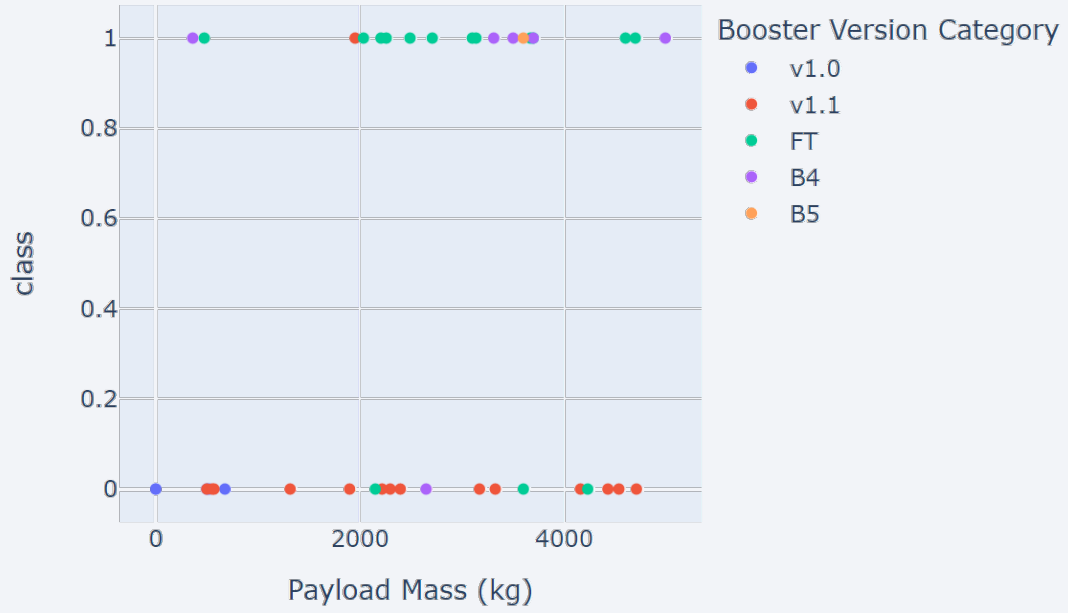


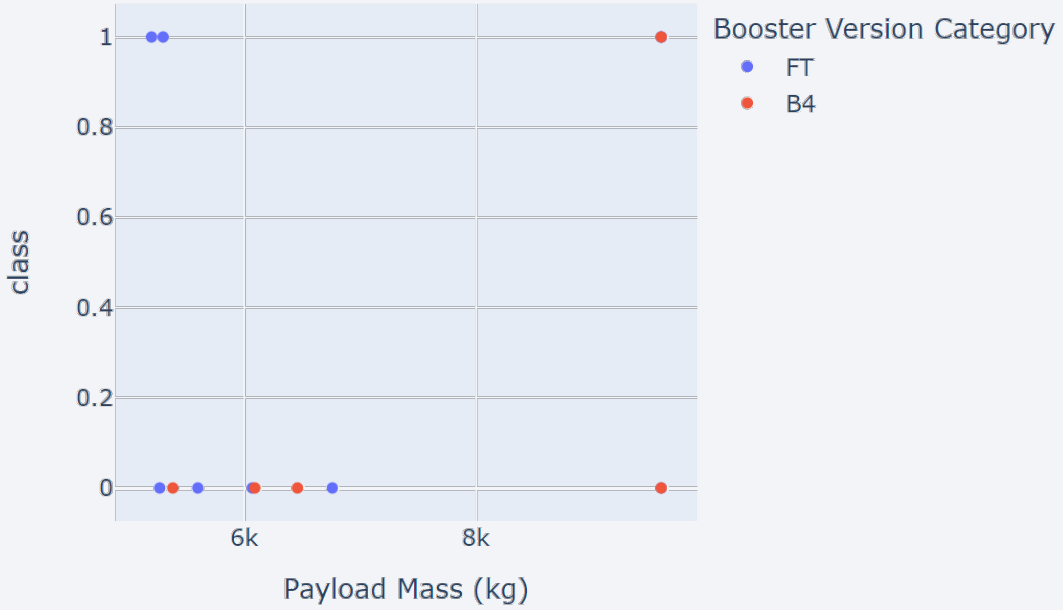
Launch site with the highest launch success ratio

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



Payload vs. Launch Outcome scatter plot for all sites





- By comparing the launch outcomes for different ranges of payload (light: 0~5000 kg and heavy: 5000~10000 kg), we can find that the lighter payload correlates to higher success rate.
- The Full Thrust (FT) booster shows very high success rate.

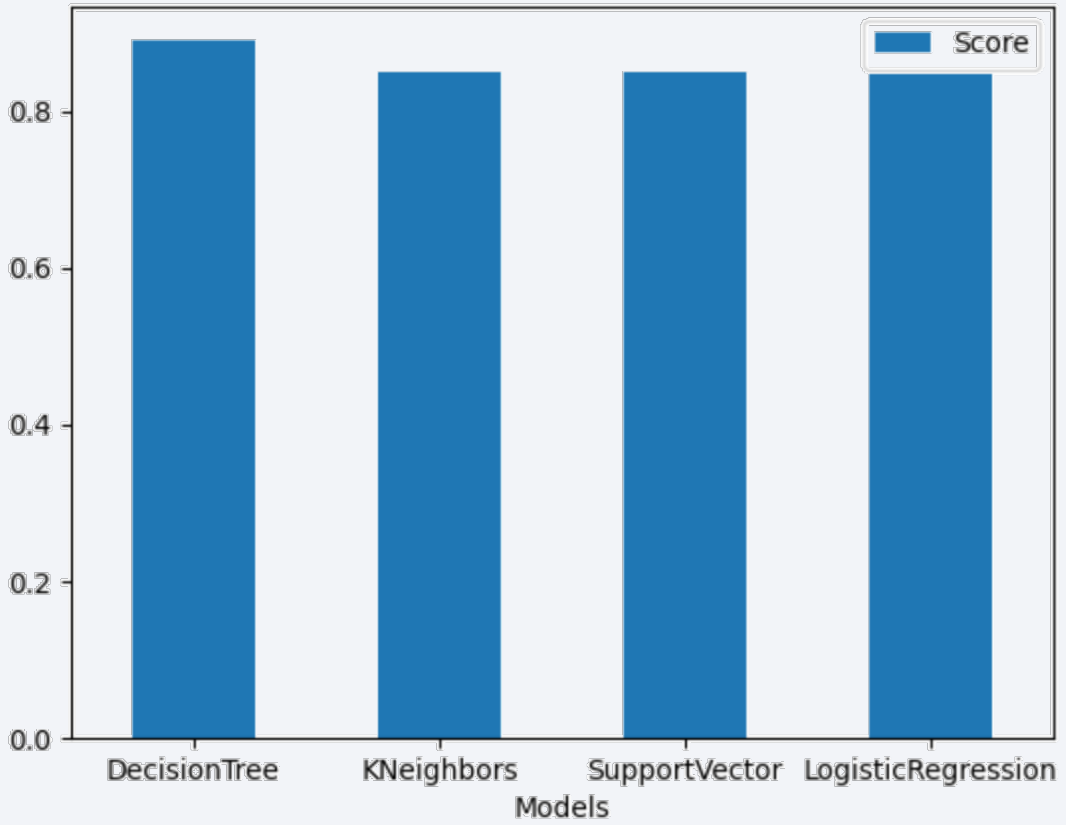
Section 5

Predictive Analysis (Classification)

Classification Accuracy

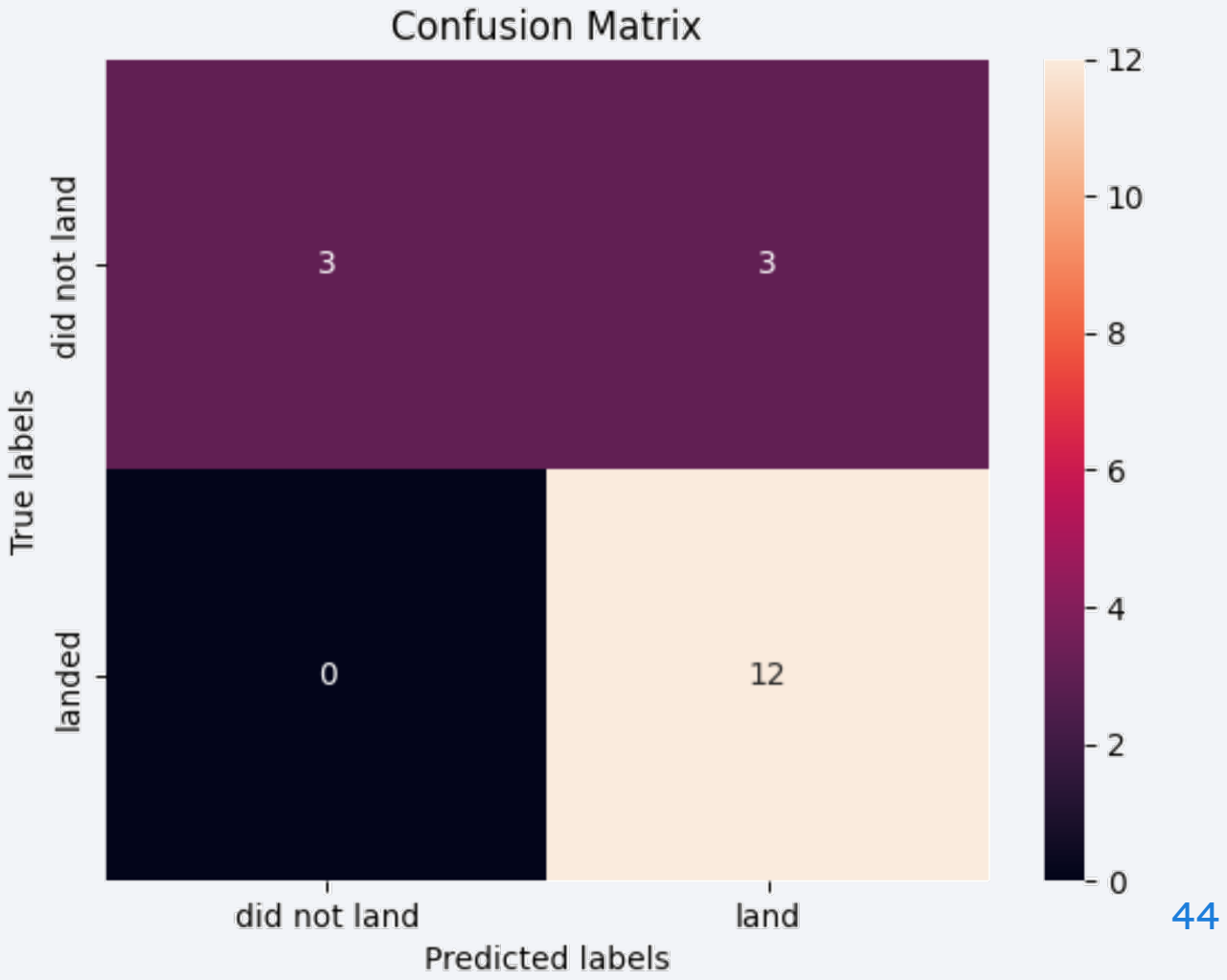
- The decision tree classifier is the model with the highest classification accuracy

```
In [66]: models = {'KNeighbors': knn_cv.best_score_,
                  'DecisionTree': tree_cv.best_score_,
                  'LogisticRegression': logreg_cv.best_score_,
                  'SupportVector': svm_cv.best_score_}

         pdscore = pd.Series(models).to_frame()
         pdscore.reset_index(inplace = True)
         pdscore.columns = ['Models','Score']
         pdscore.head()
         pdscore = pdscore.sort_values('Score', ascending = False)
         pdscore.plot.bar(x='Models', y='Score', rot=0)
         plt.show()
```



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.



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

- All the original python code can be found at https://github.com/wugaxp/IBM-DS0720EN-Data-Science-and-Machine-Learning-Capstone-Project
- The slides are prepared for the IBM DS0720EN course - **Data Science and Machine Learning Capstone Project** at https://learning.edx.org/course/course-v1:IBM+DS0720EN+2T2021/home

Thank you!