IBM Developer
SKILLS NETWORK

Falcon 9 Landing Prediction Project

Jairo Contreras

8/2/2023

https://github.com/jairocontrerasg

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data Collection through API and 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**
  - Data Analysis along with Interactive Visualization
  - Predictive Analytics Results

Introduction

- **Project background and context:**

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- **Problems we 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 landing.

  - What operating conditions need Space X to achieve the best results.

Methodology

- **Data collection methodology:**
  - Data was collected using SpaceX API and web scraping from Wikipedia
- **Perform data wrangling:**
  - Applied one hot encoding to categorical values
- **Perform exploratory data analysis (EDA) using visualization and SQL:**
  - Created scatter and bar plots to distinguish patterns in the data
- **Perform interactive visual analytics:**
  - Built an interactive dashboard using Folium and Plotly Dash
- **Perform predictive analysis using classification models:**
  - Built and evaluated classification models

Data Collection

- **The data was collected using several methods:**
  - First, we performed a get request to the SpaceX rest API
  - Next, we encoded the response as a Json file and turned it into a Pandas dataframe
  - We then filtered the data and filled missing values where necessary
  - Furthermore, we carried out Web Scraping from Wikipedia utilizing the BeautifulSoup library

Data Collection – SpaceX Rest API

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

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

Using the dataframe `data` print the first 5 rows

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

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	cap
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine	Engine failure at 33 seconds and loss of vehicle	[]	[]	

- We performed a get request from SpaceX API
- Converted the response to a Json file
- Applied custom functions to clean data
- Assigned list to dictionary and created a Pandas dataframe
- Filtered dataframe and exported to flat file

Data Collection – Web Scraping

- We used BeautifulSoup to get a response from HTML
- Created BeautifulSoup object
- Located target tables
- Got column names
- Created a dictionary and appended data to keys
- Converted dictionary to Pandas dataframe
- Turned dataframe to a .csv file

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

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

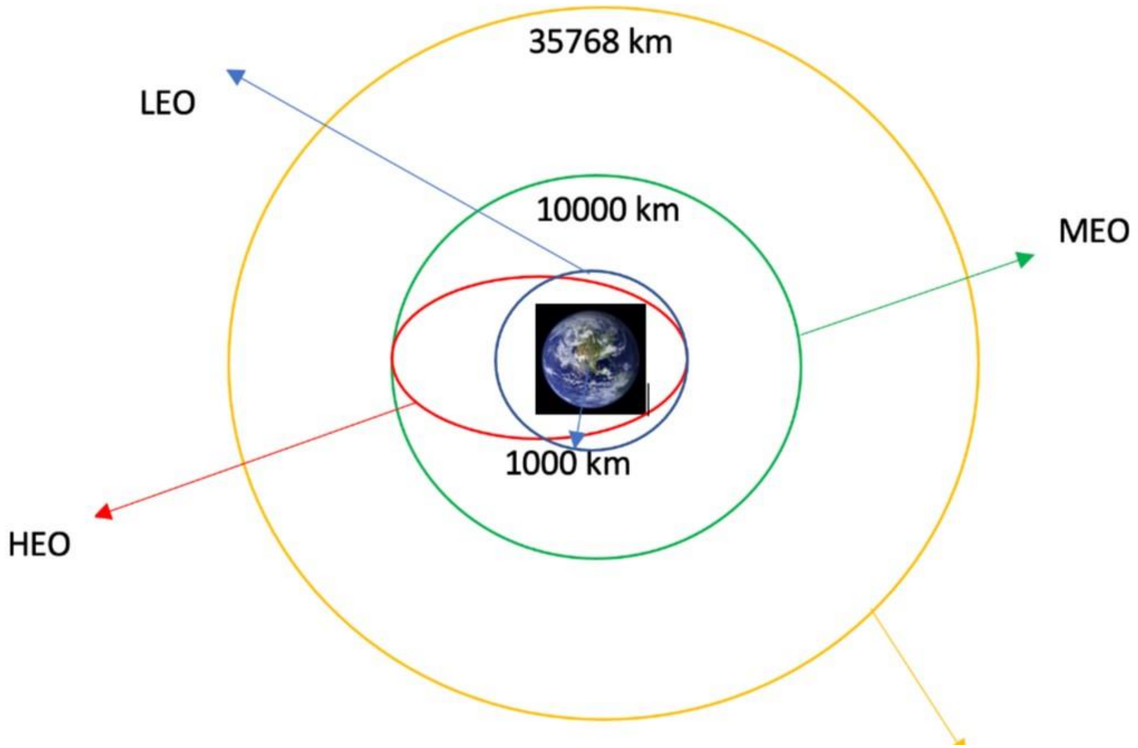
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response tex
html_file = BeautifulSoup(response, "html.parser")
```

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

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

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

Data Wrangling



We calculated the number of launches at each site, as well as the number and occurrence of each orbit

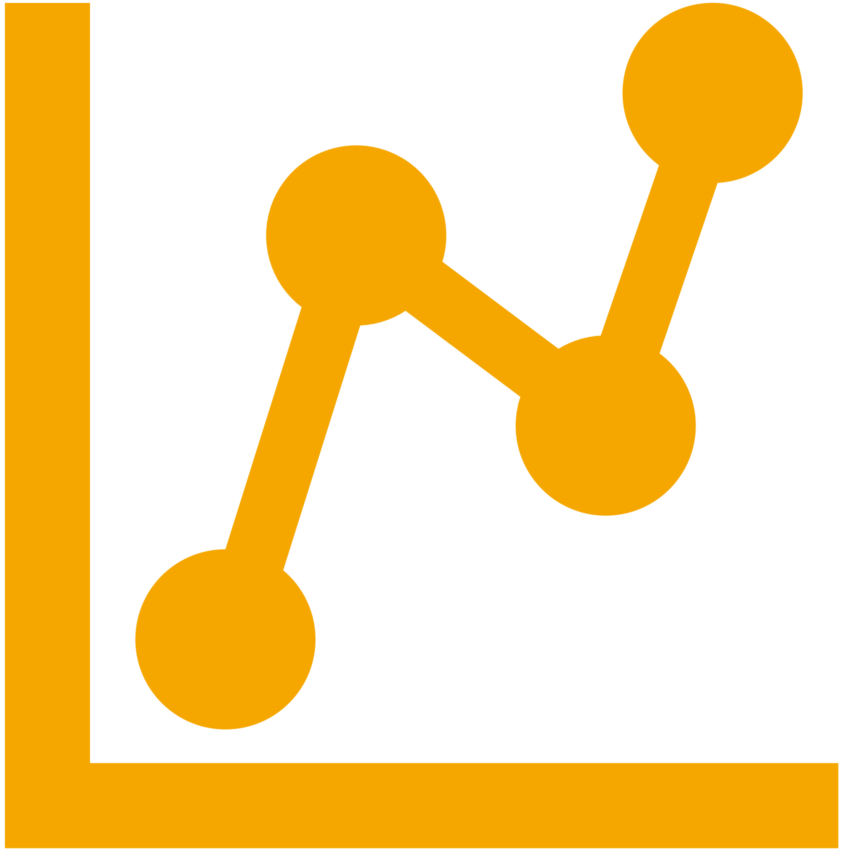
Estimated the number and occurrence of mission outcome per orbit type

Created outcome label from Outcome column

Exported dataset as .csv file

EDA with Data Visualization

- Created a diverse set of scatter plots to visualize the dependency between different variables in the data, such as payload, flight number, launch site and orbit type
- Plotted a bar graph to determine the success rate as a function of orbit type
- Displayed a line graph to visualize the launch success in a yearly trend



```
%%sql
select LANDING_OUTCOME, count(LANDING_OUTCOME) as TOTAL_
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
Done.
```

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

EDA with SQL

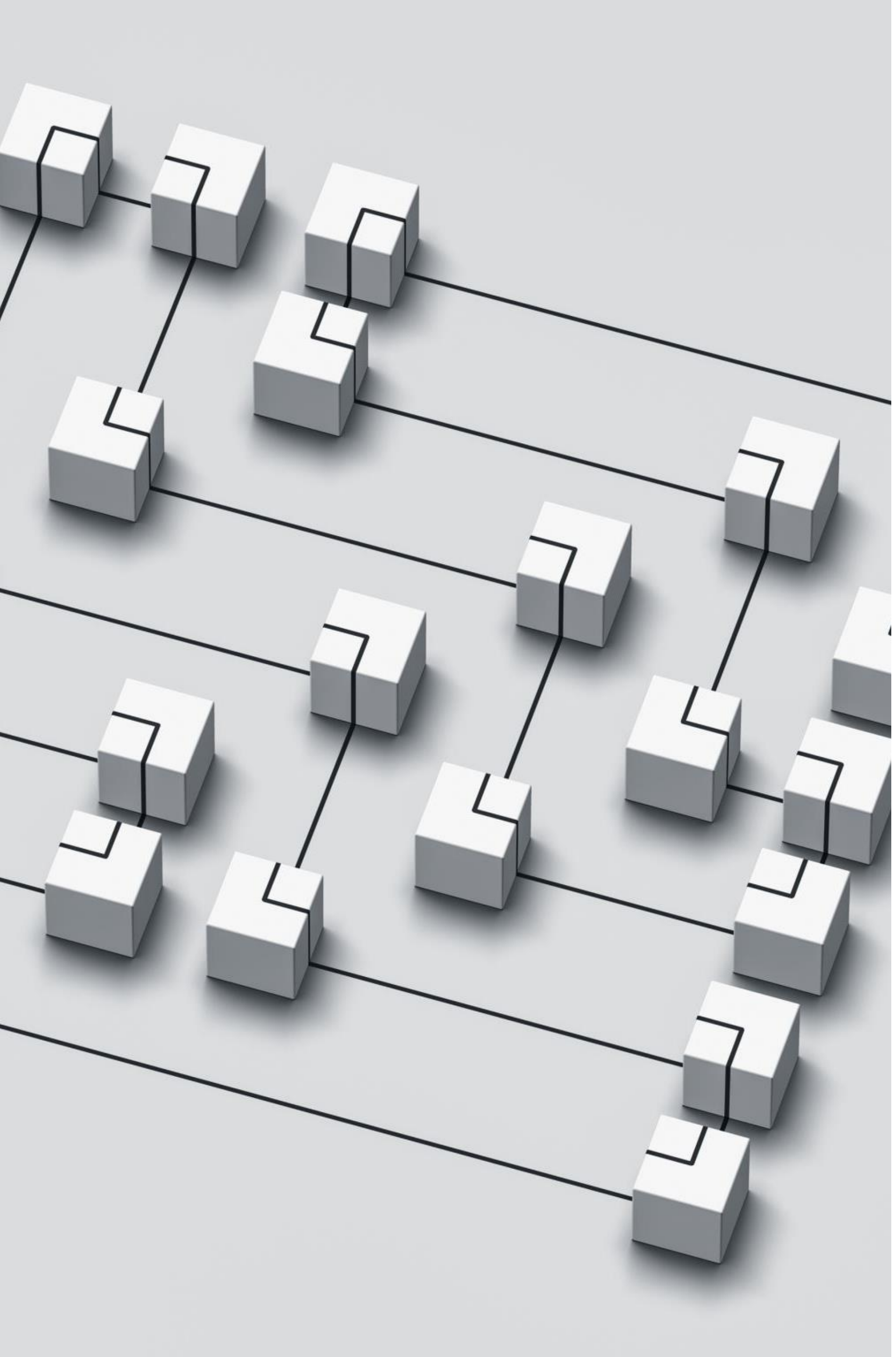
- Performed SQL queries to gather valuable insights from the data set, including:
    - Unique launch site names
    - Total payload mass carried by boosters launched by NASA
    - Average payload mass carried by booster version F9 v1.1
    - Total number of successful and failure mission outcomes
    - Failed landing outcomes in drone ship, their booster version and launch site names

Build an Interactive Map with Folium

- We marked all launch sites and added map objects such as markers, circles and lines to indicate the successes and failures of launches in each site
- Used the color-labeled marker to identify which launch sites have a relatively high success
- Calculated the distances between launch sites and its proximities, which helped us understand whether launch site are near railways or highways, and if launch sites are kept certain distance away from cities

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly Dash
- Plotted a pie chart showing the total success for all sites and by specific launch sites
- Displayed scatter graph showing the correlation between payload and success for all sites and by specific launch sites



Predictive Analysis (Classification)

- We loaded our feature engineered data into a dataframe
- Transformed and standardized it into NumPy arrays
- Split data into training and testing data sets
- Listed down machine learning algorithms we wanted to implement
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and trained our model

Results

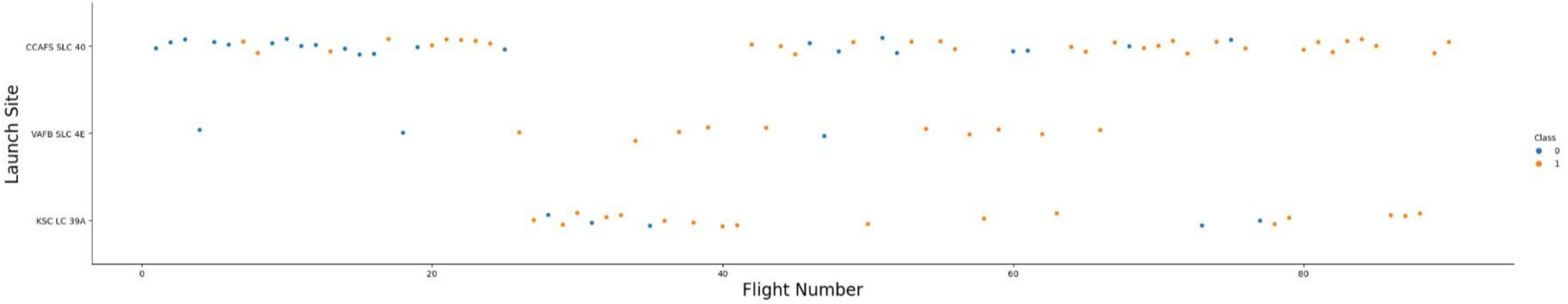
EXPLORATORY DATA ANALYSIS RESULTS

INTERACTIVE ANALYTICS DEMO IN SCREENSHOTS

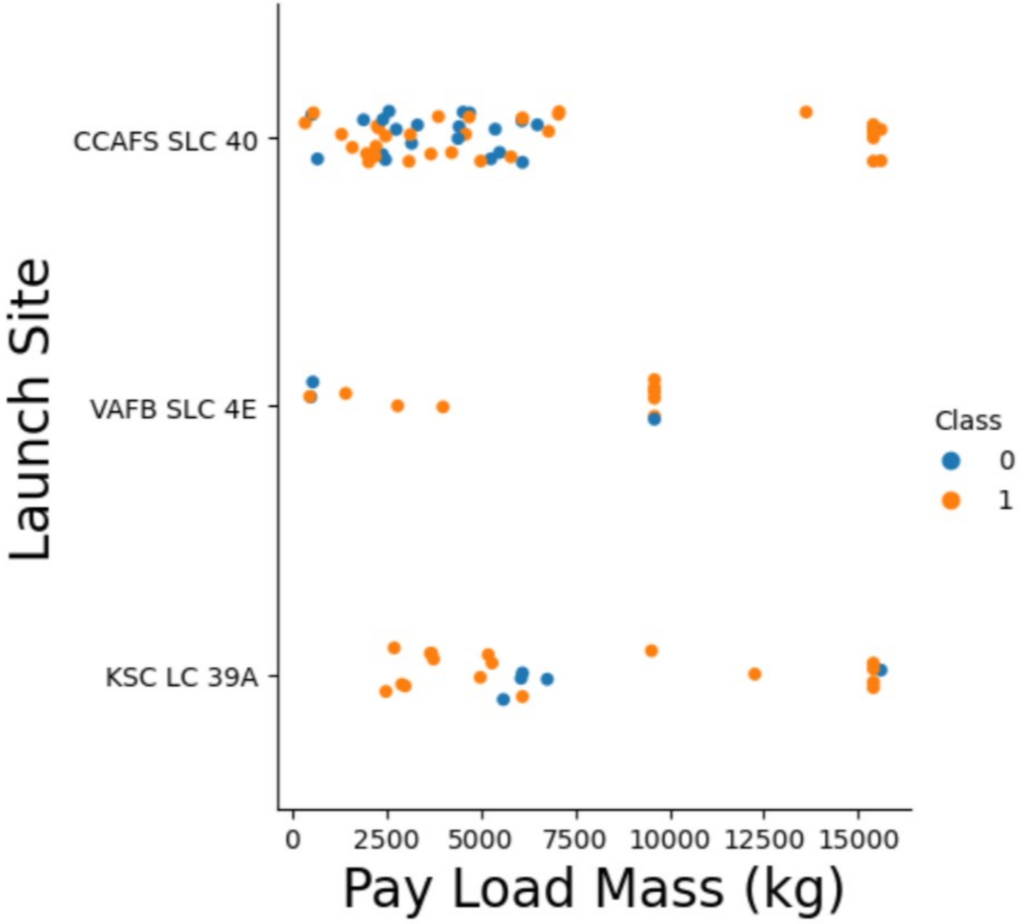
PREDICTIVE ANALYSIS RESULTS

EDA with Visualization

- Flight Number vs. Launch Site
  - With higher flight numbers (greater than 30) the success rate for the Rocket is increasing



Payload vs. Launch Site



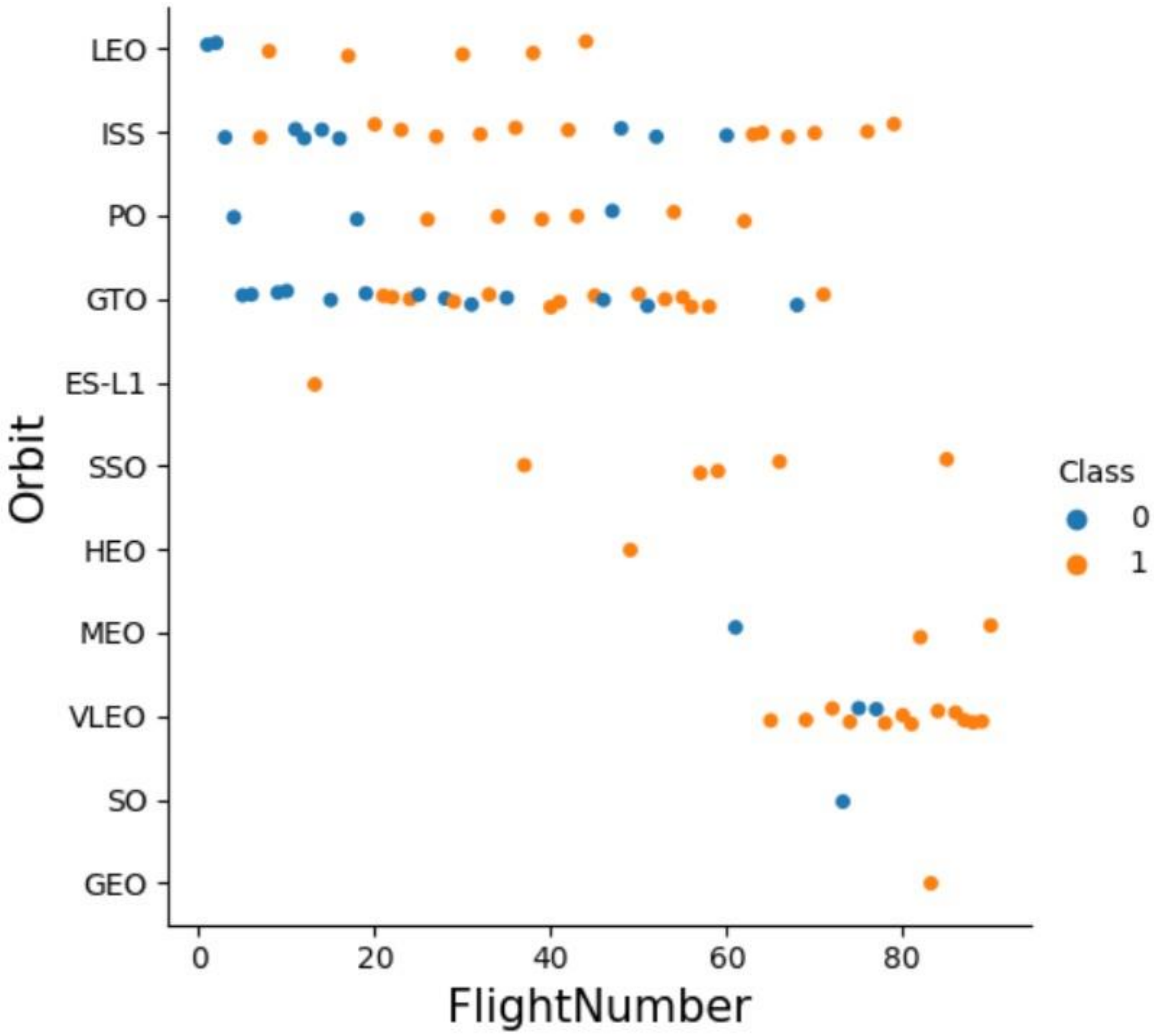
- Seems that the greater the payload mass (above 7000 kg), higher the success rate for the Rocket.

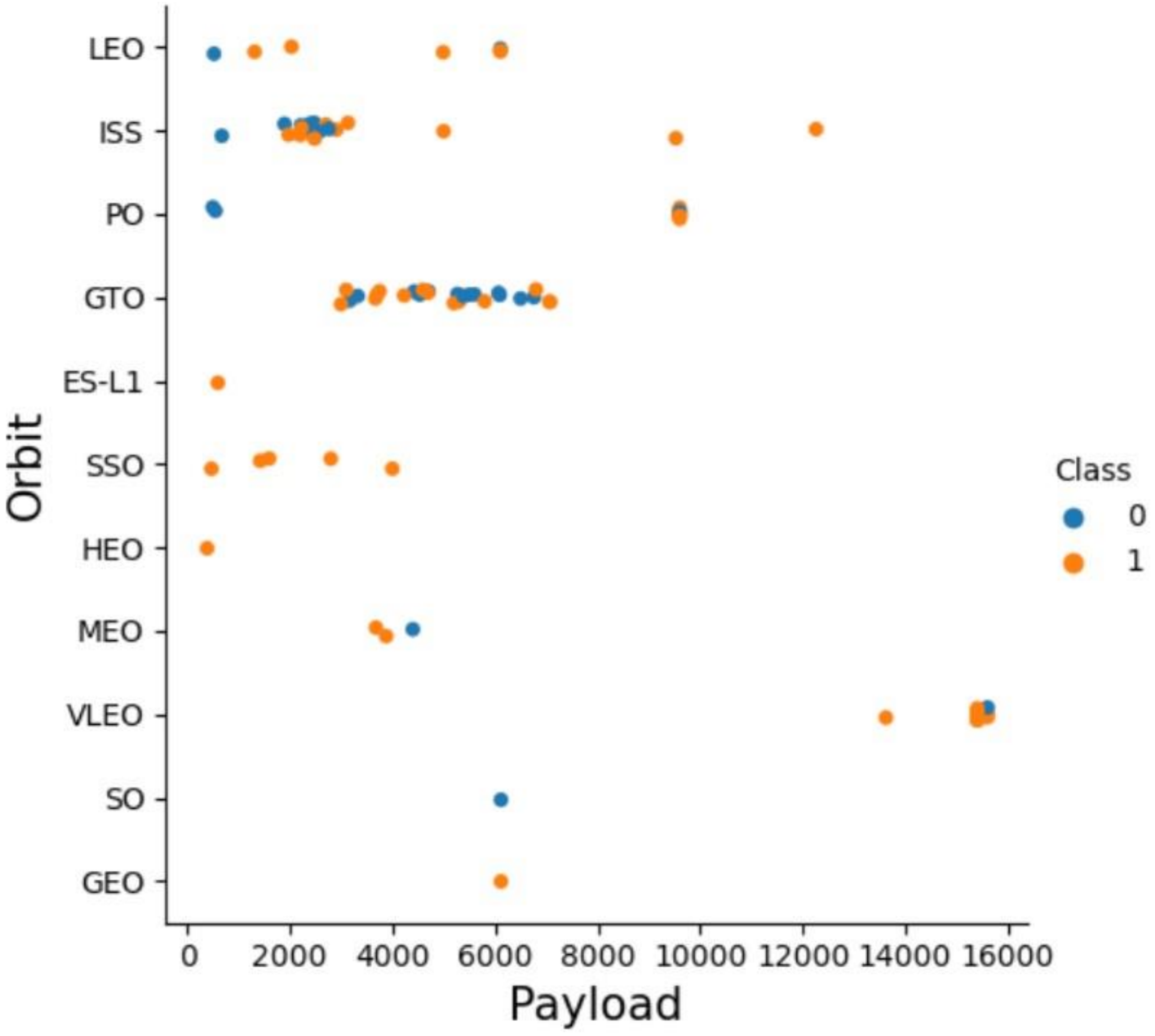
Success Rate vs. Orbit Type

ES-L1, GEO, HEO and SSO orbits had the highest success rates

Flight Number vs. Orbit Type

- We observed that for LEO orbit, the success rate increases with the number of flights. In contrast, there seems to be no correlation between flight number and the GTO orbit



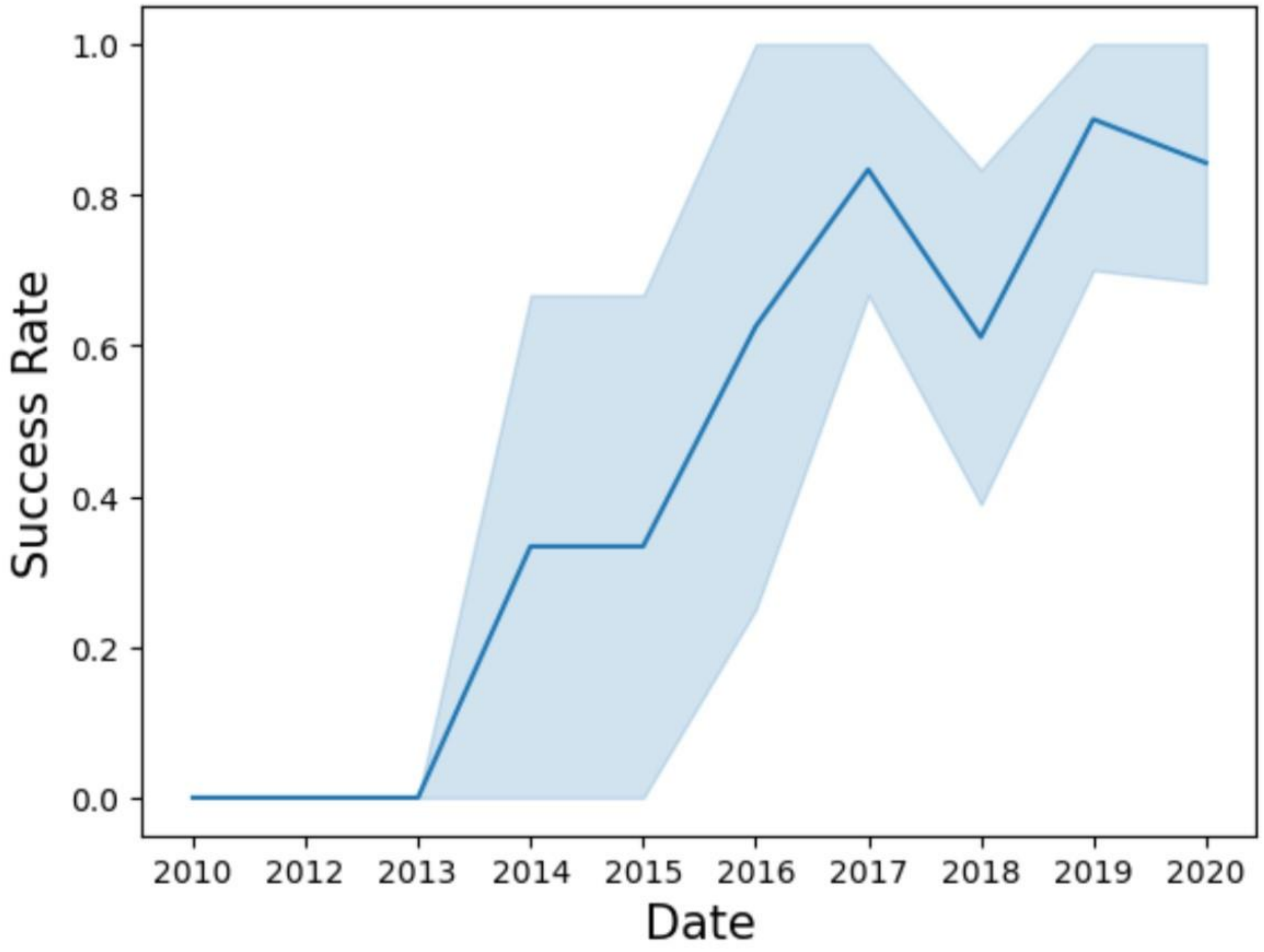


Payload vs. Orbit Type

- We observed that heavy payloads are related to successful landings for PO, LEO, and ISS orbits
- The opposite is true for MEO, GTO, and VLEO orbits

Launch Success Yearly Trend

- We observed that the success rate kept increasing since 2013 until 2020, even though there was a slight dip in 2018 and after 2019



EDA with SQL

- All Launch Site Names
  - Used key word DISTINCT in the query to pull unique values for Launch_Site column from the SpaceX table

```
%sql select Distinct(LAUNCH_SITE) from SPACEXTBL;
```

```
 * 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'

Used query to display 5 records where launch sites which begin with '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	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
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

- Calculated the total payload mass carried by boosters from NASA using the function SUM and the WHERE clause to filter the data

```
%sql select SUM(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER =
```

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where Booster_Version = 'F9 v1.1';
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

- Calculated the average payload mass carried by a specific booster using the function AVG and the WHERE clause

First Successful Ground Landing Date

- Used the function MIN and the WHERE clause filter to find out the first successful ground landing date

```
%sql select min(DATE) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)';
```

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

min(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Used the SELECT function and the WHERE clause to filter the dataset

```
%sql select BOOSTER_VERSION from SPACEXTBL WHERE PAYLOAD_MASS__KG_ between 4000 and 6000 and Landing_Outcome = 'Succes
```

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

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

Total Number of Successful and Failure Mission Outcomes

Used GROUP BY clause to filter by mission outcome

```
%%sql
select MISSION_OUTCOME, count(MISSION_OUTCOME) as TOTAL_NUMBER
from SPACEXTBL
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

Used a subquery in the WHERE clause and the MAX() function

```
%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

2015 Launch Records

- Used the WHERE clause in conjunction with the LIKE, AND conditions to filter for failed landing outcomes in drone ship, their booster version, and launch site names for the year 2015

```
%%sql
SELECT date, landing_outcome, booster_version, launch_site
FROM SPACEXTBL
WHERE landing_outcome = 'Failure (drone ship)' and Date like '2015%';
```

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

Date	Landing_Outcome	Booster_Version	Launch_Site
2015-10-01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

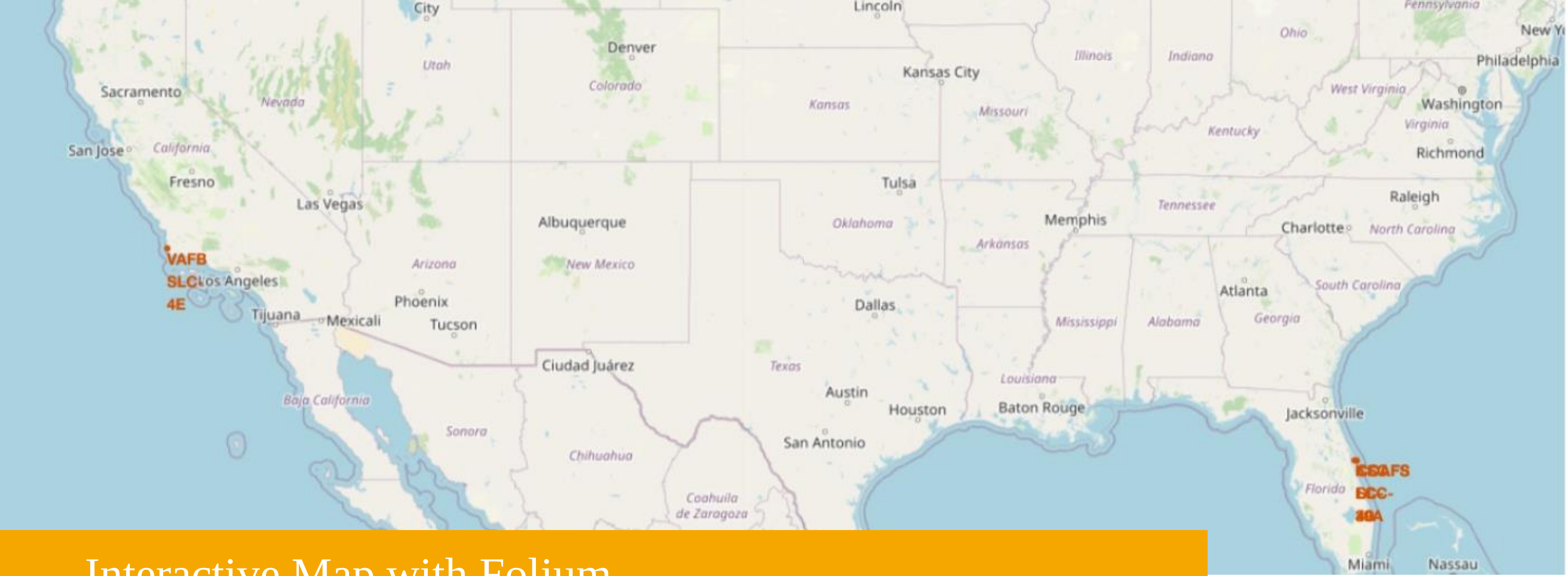
```
%%sql
select LANDING_OUTCOME, count(LANDING_OUTCOME) as TOTAL_NUMBER
from SPACEXTBL
where DATE between '2010-06-04' and '2017-03-20'
group by LANDING_OUTCOME
order by TOTAL_NUMBER desc
```

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

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

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

- Selected landing outcomes and their COUNT using the WHERE clause
- We also used the GROUP BY clause and the ORDER BY clause to order the grouped landing outcome in descending order

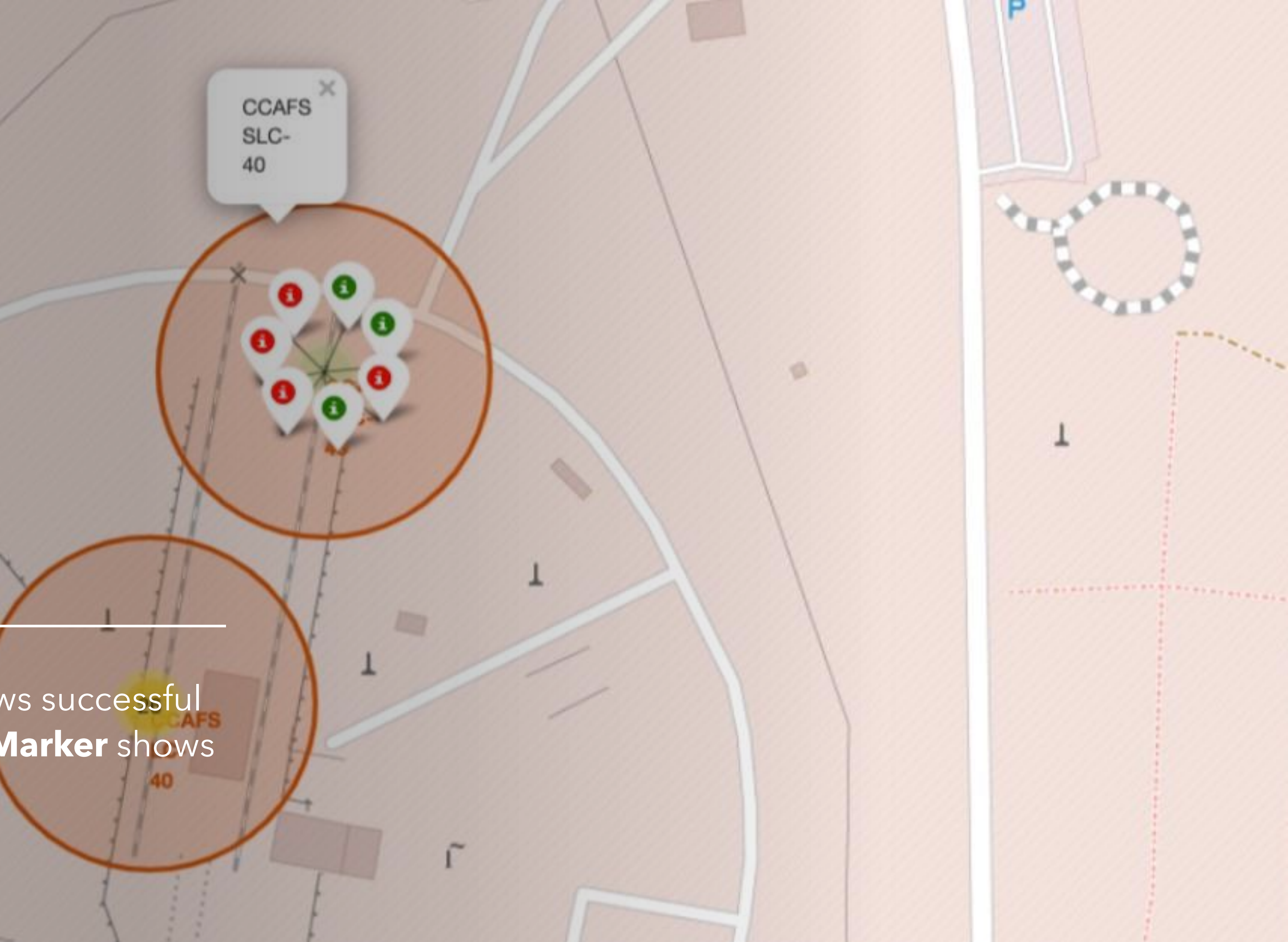


Interactive Map with Folium

- All Launch Sites on Folium Map
  - We can see that the SpaceX launch sites are in the United States of America coasts i.e., Florida and California Regions

Colored Labeled Launch Records

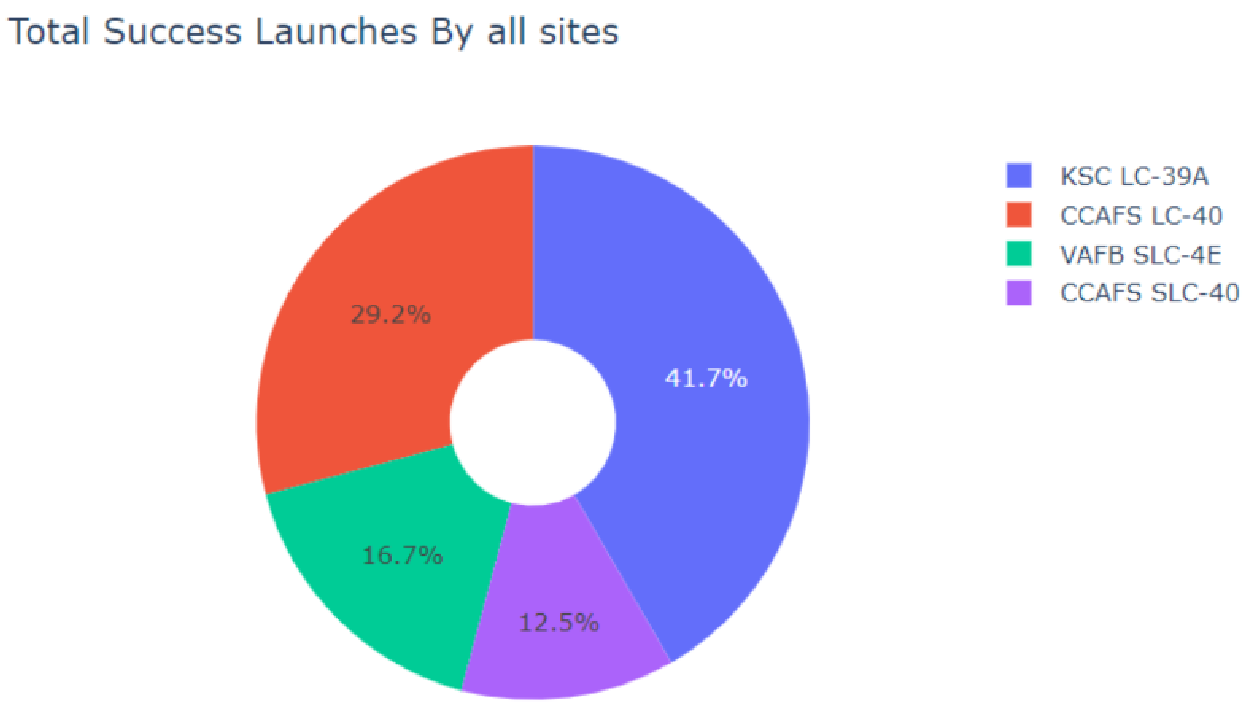
Green Marker shows successful launches and **Red Marker** shows failures



Launch Site Distances to Landmarks

- Distance from Equator is greater than 3000 km for all sites
- Distance for all launch sites from railway tracks are greater than 0.7 km for all sites
- Distance for all launch sites from cities is greater than 14 km for all sites
- Distance for all launch sites from cities is less than 4 km

Build a Dashboard with Plotly Dash

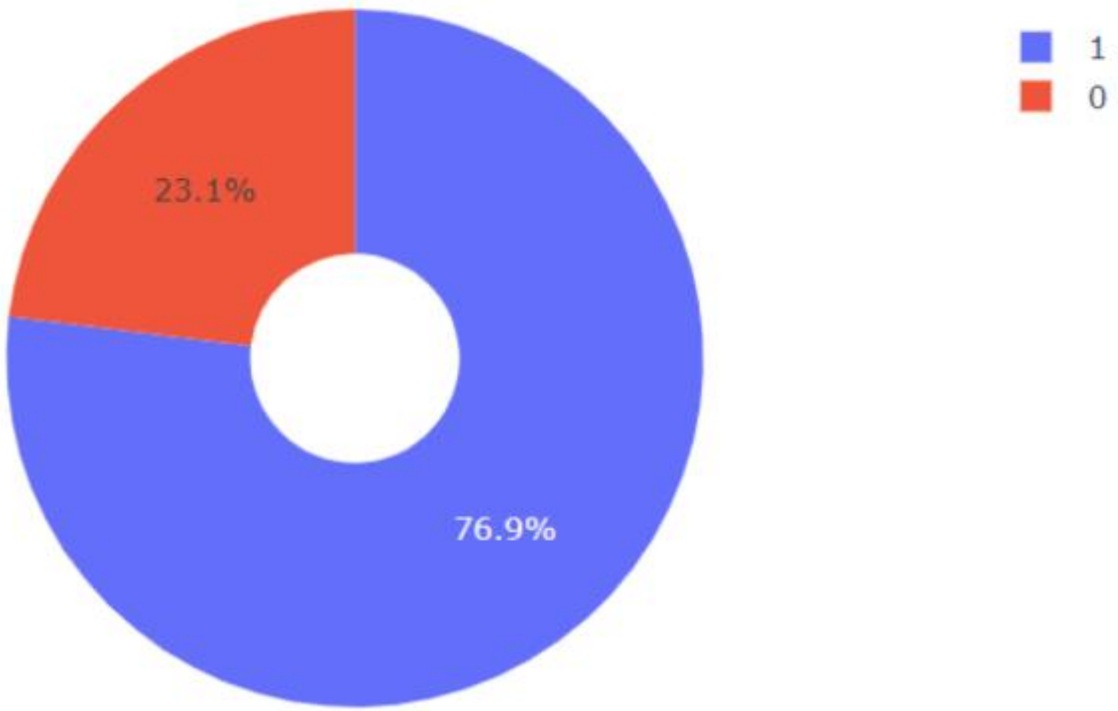


Pie Chart showing the success percentage achieved by each launch site

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

Launch Site with Highest Launch Success Ratio

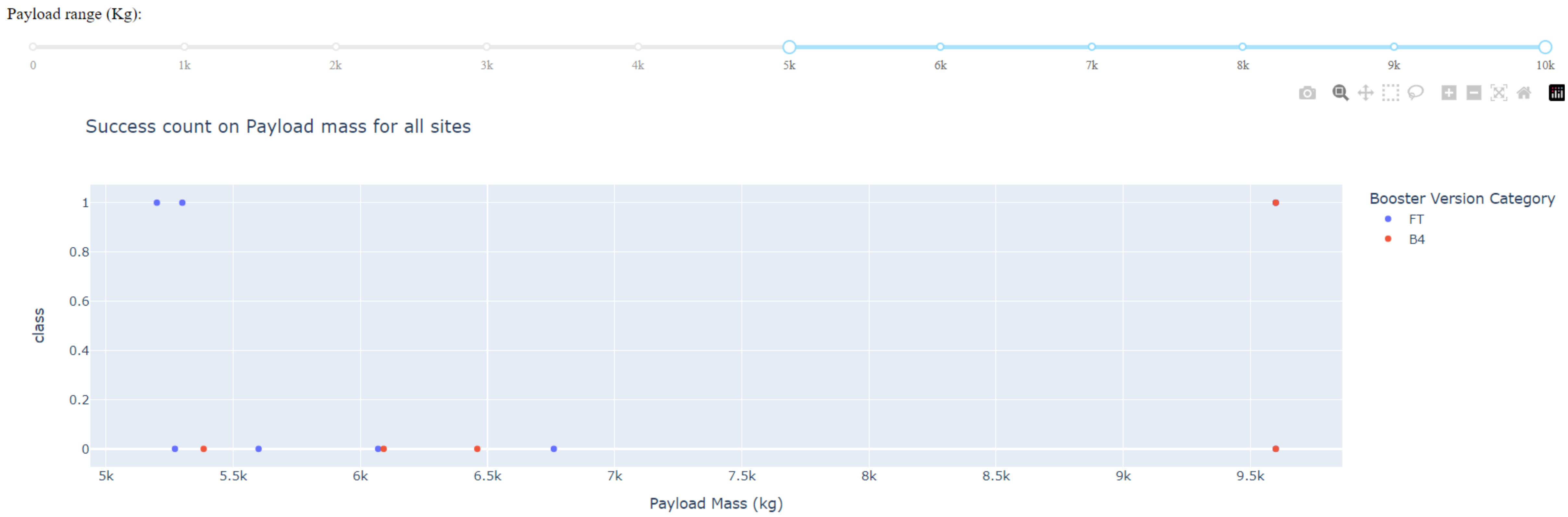
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate



Total Success Launches for KSC LC -39A

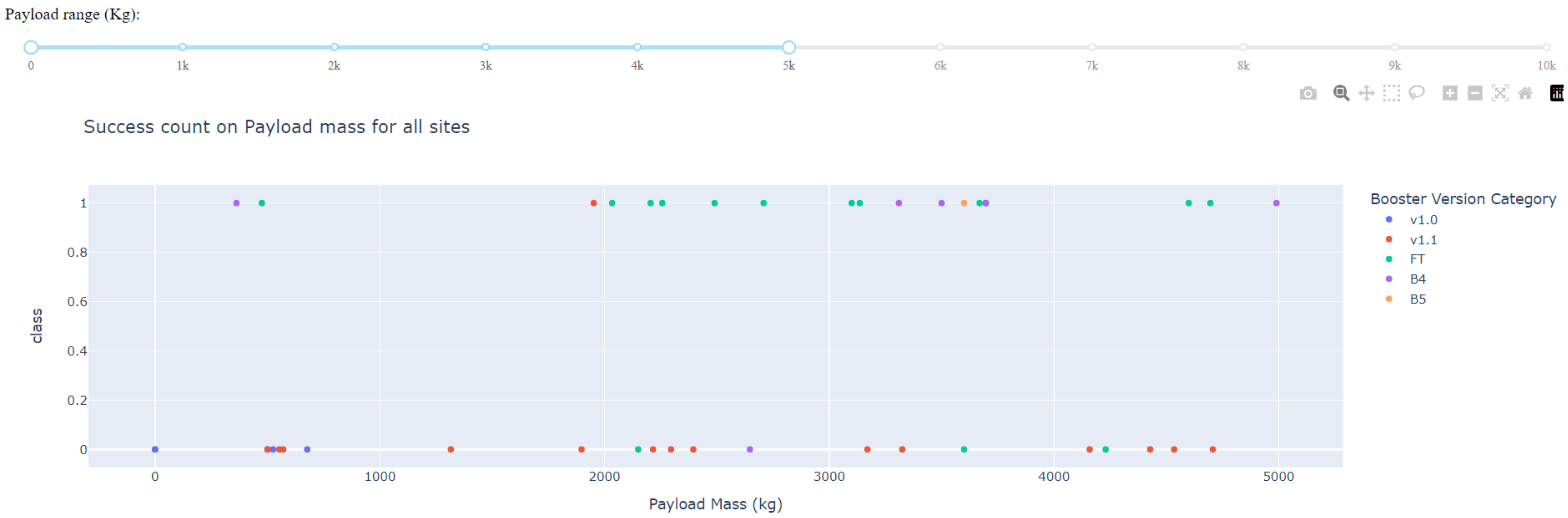
Payload vs. Launch Outcome Scatter Plot for All Sites: Heavy Payload

We can see the success rates are low for heavy payloads



Payload vs. Launch Outcome Scatter Plot for All Sites: Light Weight

- We can see the success rates are high for light payloads, compared to heavier payloads



Predictive Analysis (Classification)

```
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier(max_features='sqrt')
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
grid_search = GridSearchCV(tree, parameters, cv=10)
tree_cv = grid_search.fit(X_train, Y_train)
```

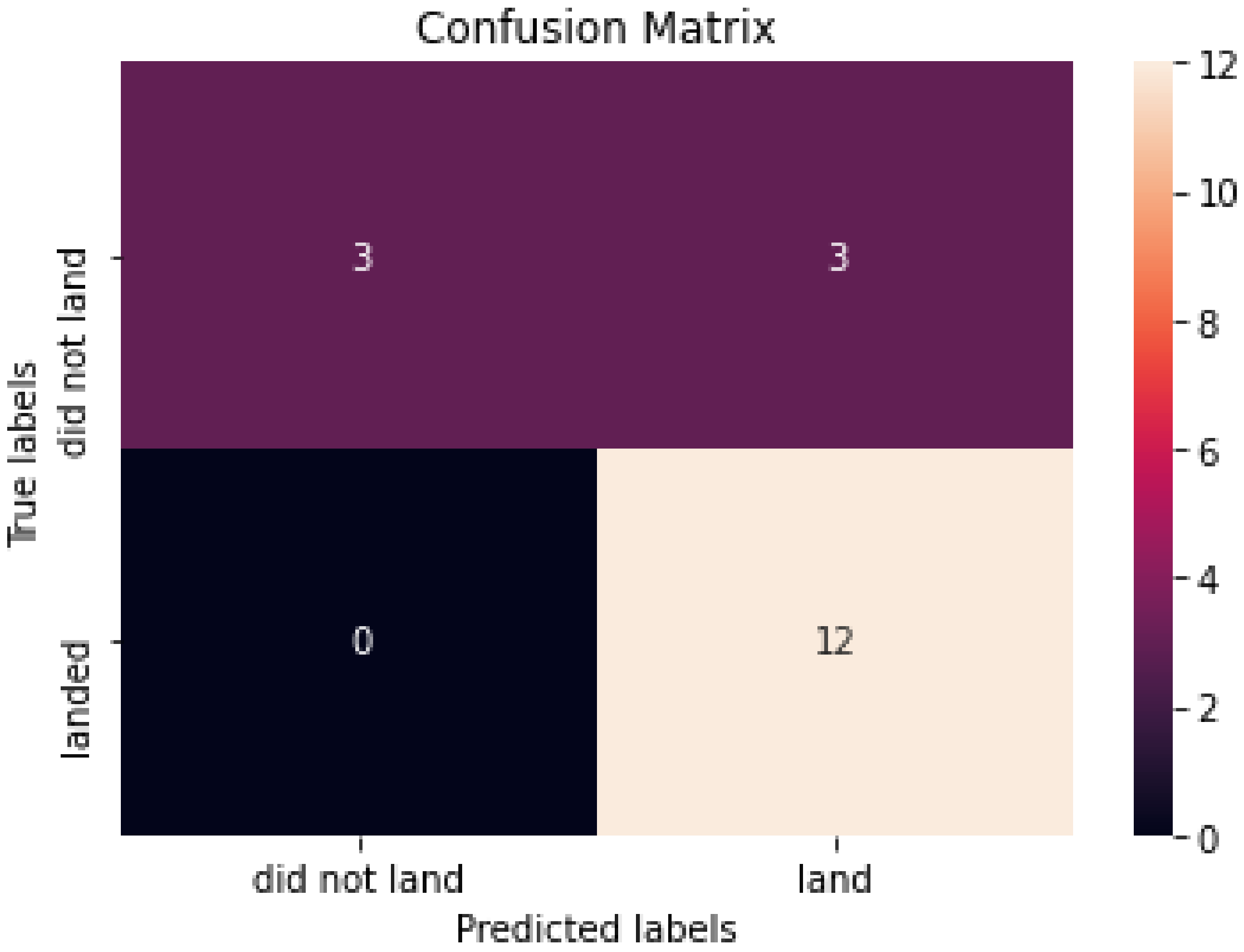
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

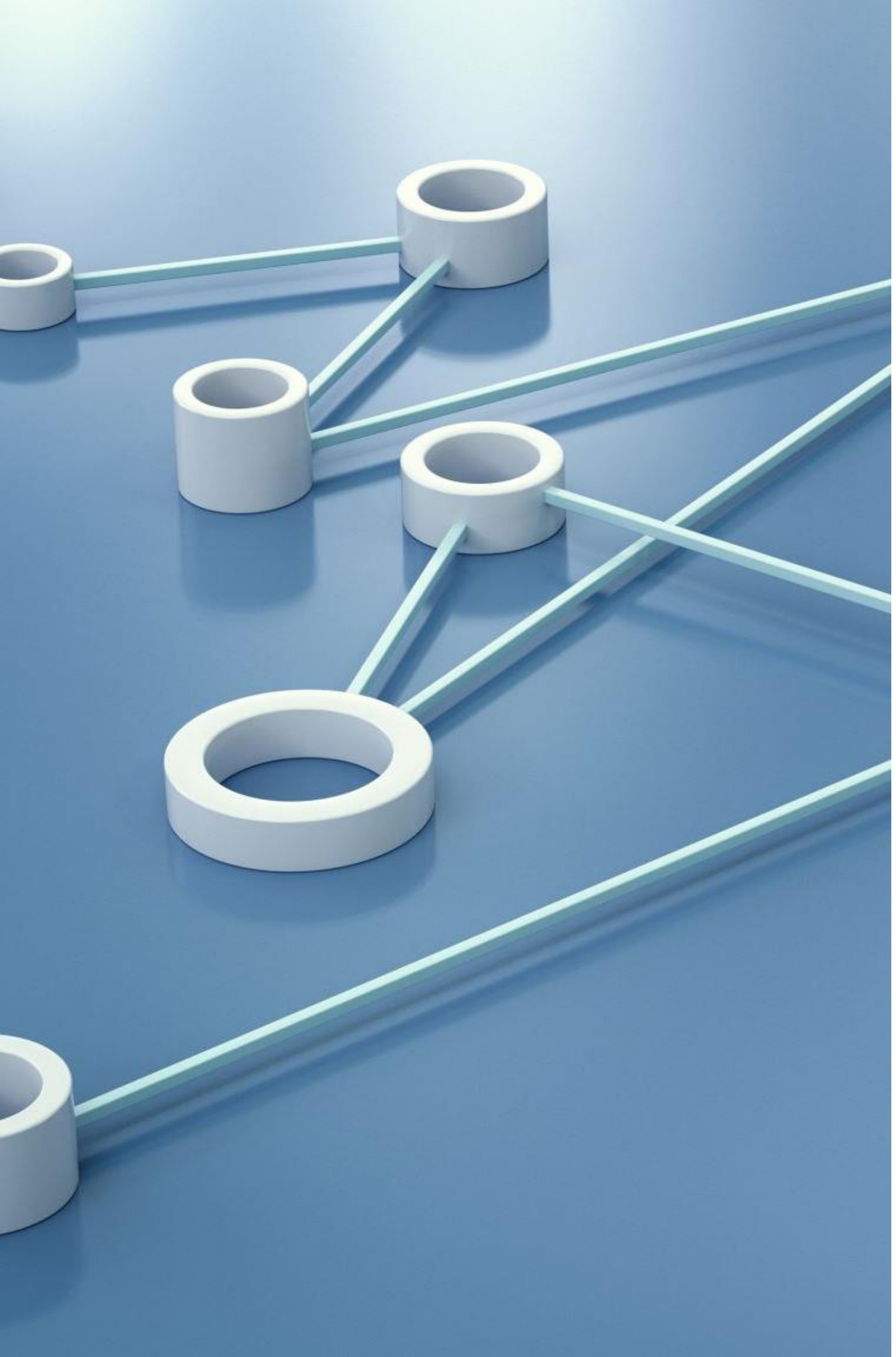
```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2,
'min_samples_split': 10, 'splitter': 'random'}
accuracy : 0.8875
```

- After creating a machine learning model employing four different algorithms: Logistic Regression, SVM, KNN, and Decision Tree, we observed that the Decision Tree classifier is the model with the highest accuracy on validation data – 88.75%
- Each model, however, had an accuracy of 83% on the test data

Confusion Matrix

- We found the same confusion matrix for all models. The major problem was false positives i.e., unsuccessful landing marked as successful by the classifier





Conclusion

- Orbits ES-L1, GEO, HEO, SSO have highest success rates
- Success rates for SpaceX launches have been increasing with time
- KSC LC-39A had the most successful launches in any site, but high payload mass seems to have a negative impact on success
- The Decision Tree Classifier Algorithm is the best for Machine Learning Model for provided dataset

Appendix

- The complete project with source code and visualizations can be found in my GitHub account. The following link will take you directly to the project: https://github.com/jairocontrerasg/Space-X-Falcon-9-Landing-Prediction/tree/main

Thank you!