Winning Space Race with Data Science

Vaibhav Sonkar

15-10-2022

Outline

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

Executive Summary

- Summary of methodologies
  1. Data Collection
  2. Data Wrangling
  3. EDA with SQL
  4. EDA with Matplotlib, and Seaborn
  5. Building an interactive map with Folium
  6. Building a dashboard with Plotly & Dash.
  7. Predictive Analysis using Classification techniques
- Summary of all results
  - EDA Results
  - Interactive Visual Analytics
  - Predictive Analysis

Introduction

- Project background and context

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million USD; other providers cost upward of 165 million USD 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 you want to find answers

1. What factors determine if the rocket will land successfully?
2. The interaction amongst various features that determine the success rate of a successful landing.
3. What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

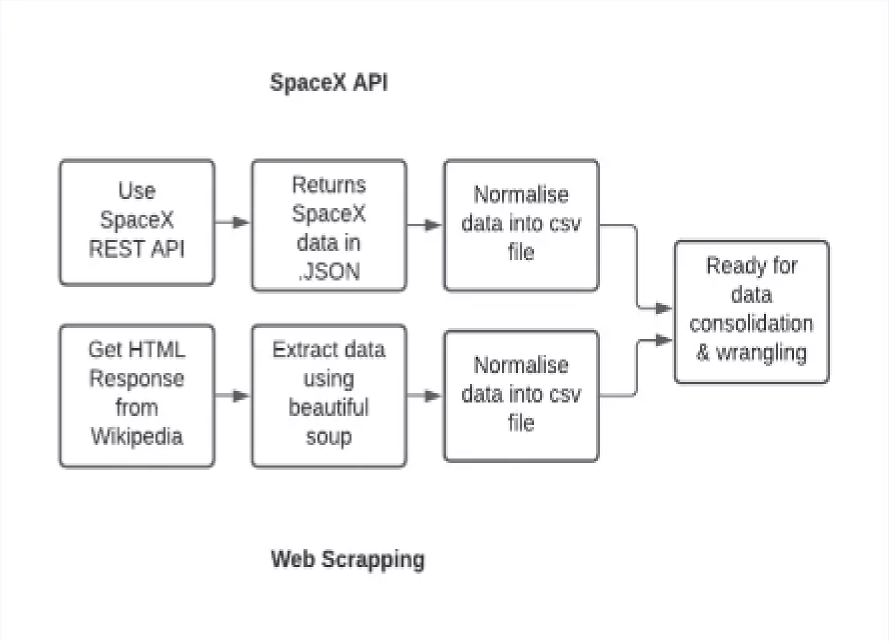
Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX REST API and web scraping from Wikipedia
- Perform data wrangling
  - Null values were replaced, One hot encoding was applied to 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
  - LR, KNN, SVM, DT models have been built and evaluated for the best classifier.

Data Collection

- Describe how data sets were collected.
  - SpaceX launch data is gathered from the SpaceX REST API.
  - It included information about the rockets used, payloads delivered, launch specifications, landing specifications and outcome.
  - Web scraping was also performed on the Wikipedia page for Falcon 9 launch data using BeautifulSoup.
- You need to present your data collection process use key phrases and flowcharts



Data Collection – SpaceX API

- Data collection with SpaceX REST calls:
- GitHub URL of the completed SpaceX API calls notebook:
- https://github.com/vaibhavov/DSproject/blob/main/Lab1%20Data%20Collection%20API.ipynb

1. Get request for rocket launch data using API

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

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

2. Use json_normalize method to convert json result to dataframe

In [12]:
```python
# Use json_normalize method to convert the json result into a dataframe

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

In [13]:
```python
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```python
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- Web Scraping from Wikipedia. The results were converted into a pandas dataframe.
- GitHub URL of the completed web scraping notebook:

https://github.com/vaibhavov/DSproject/blob/main/Lab2%20Webscraping.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_data = requests.get(static_url)
html_data.status_code
```

Out[5]:
```
200
```

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_data.text, 'html.parser')
```

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

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

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

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

4. Create a dataframe by parsing the launch HTML tables
5. Export data to csv

Data Wrangling

- Exploratory data analysis was performed 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.
- GitHub URL of your completed data wrangling notebook:
- https://github.com/vaibhavov/DSproject/blob/main/Lab3%20Data%20Wrangling.ipynb

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

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

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

```
# landing_outcomes = values on Outcome column
landing_outcomes = df.Outcome.value_counts()
```

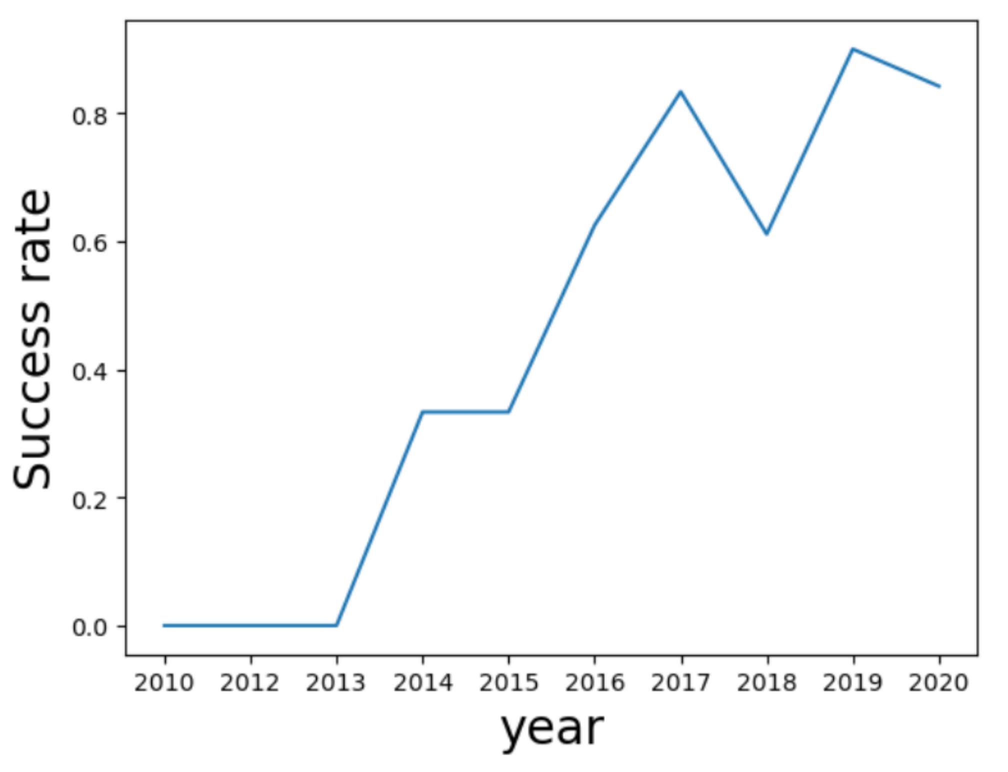
```
df.head(5)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit
0	1	2010-06-04	Falcon 9	6104.959412	LEO

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.
- GitHub URL of your completed EDA with data visualization notebook:
- https://github.com/vaibhavov/DSproject/blob/main/Lab5%20Visualization.ipynb

```
# Plot a line chart with x axis to be the extracted year and y axis to be the suc
plt.plot(average_by_year["Year"],average_by_year["Class"])
plt.xlabel("year",fontsize=20)
plt.ylabel("Success rate",fontsize=20)
plt.show()
```



EDA with SQL

- We worked with sqllite in the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch, 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.
- Add the GitHub URL of your completed EDA with SQL notebook:
  - https://github.com/vaibhavov/DSproject/blob/main/Lab4%20SQL%20EDA.ipynb

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 [LANDING _OUTCOME] = 'Success (drone ship)' AND (PAYLOAD_MASS__KG_ BETWEE
```

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

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

Task 7

List the total number of successful and failure mission outcomes

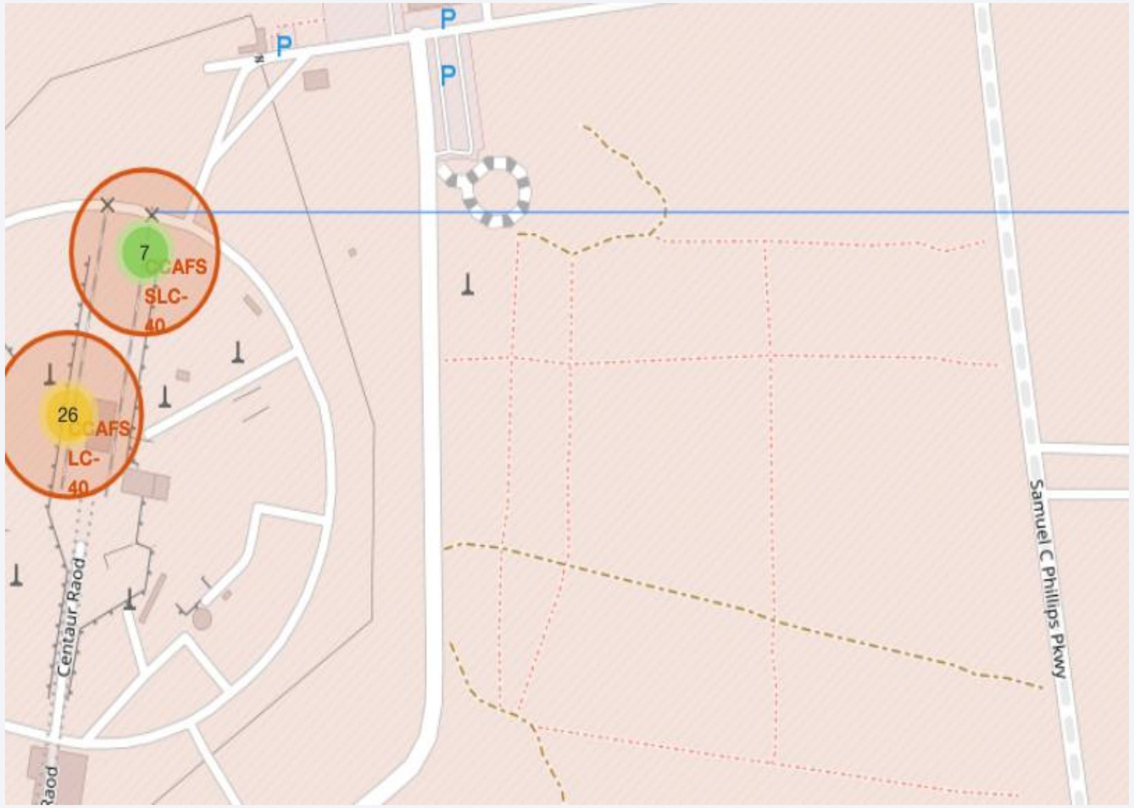
```
%sql SELECT [Mission_Outcome], COUNT([Mission_Outcome]) FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

Mission_Outcome	COUNT([Mission_Outcome])
Failure (in flight)	1
Success	98

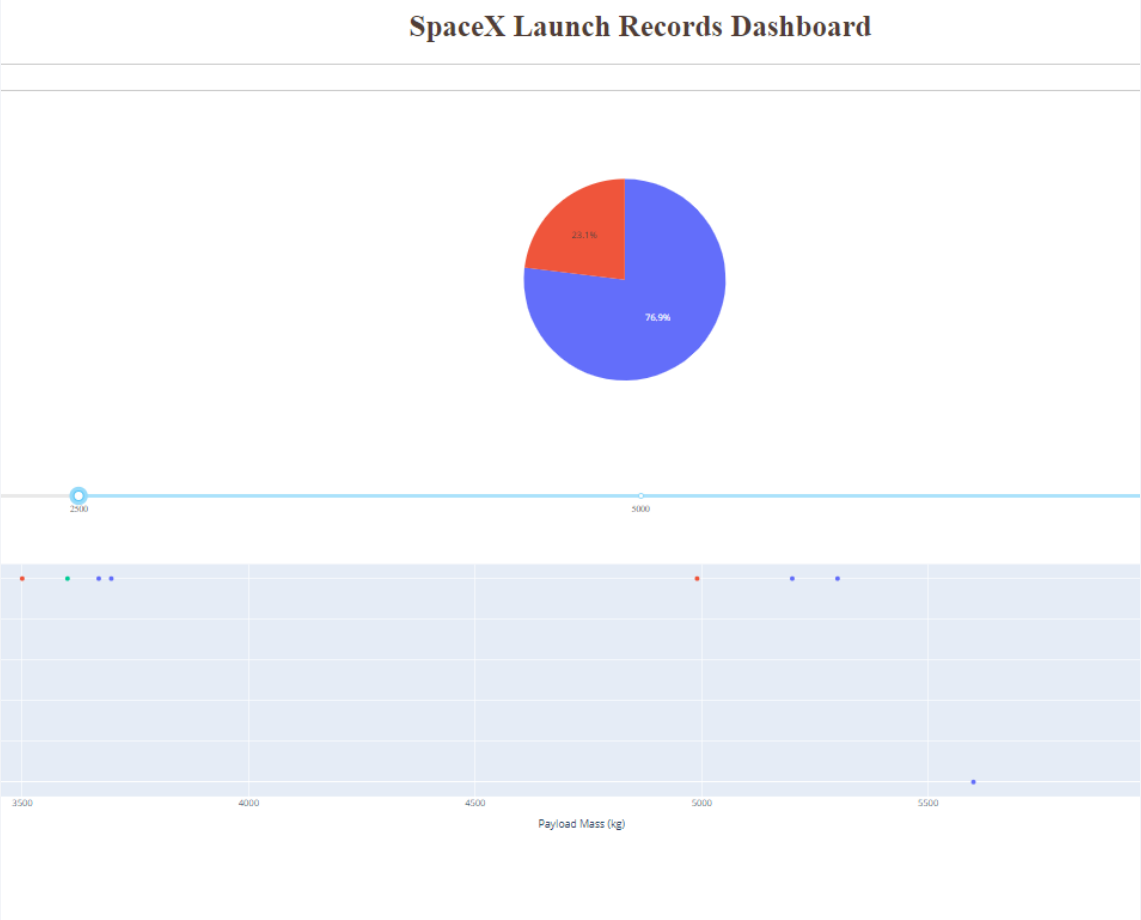
Build an Interactive Map with Folium

- 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.
- Add the GitHub URL: https://github.com/vaibhavov/DSproject/blob/main/Lab6%20Geospatial%20Visualization.ipynb



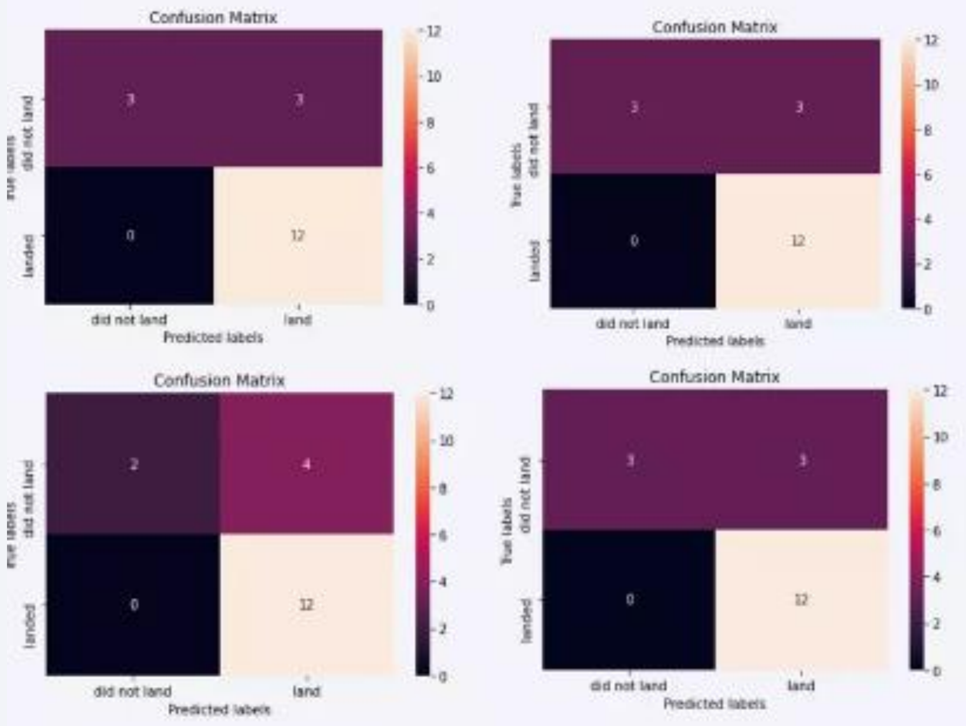
Build a Dashboard with Plotly Dash

- 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
- GitHub URL: https://github.com/vaibhavov/DSproject/blob/main/C10_dash.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.
- GitHub URL: https://github.com/vaibhavov/DSproject/blob/main/Lab7%20SpaceX-ML-Prediction.ipynb



Results

- The SVM, KNN, LR models are best in predicting accuracy of the dataset.
- Low weighted payloads performed better than heavier payloads.
- The success rate for SpaceX launches is directly proportional to the years.
- KSC LL 39A has the most successful launches from all sites.
- Orbit, GEO,, HEO, SSO, EES L1 has the best success rate.

```
print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
print('Accuracy for K nearsdt neighbors method:', knn_cv.score(X_test, Y_test))
```

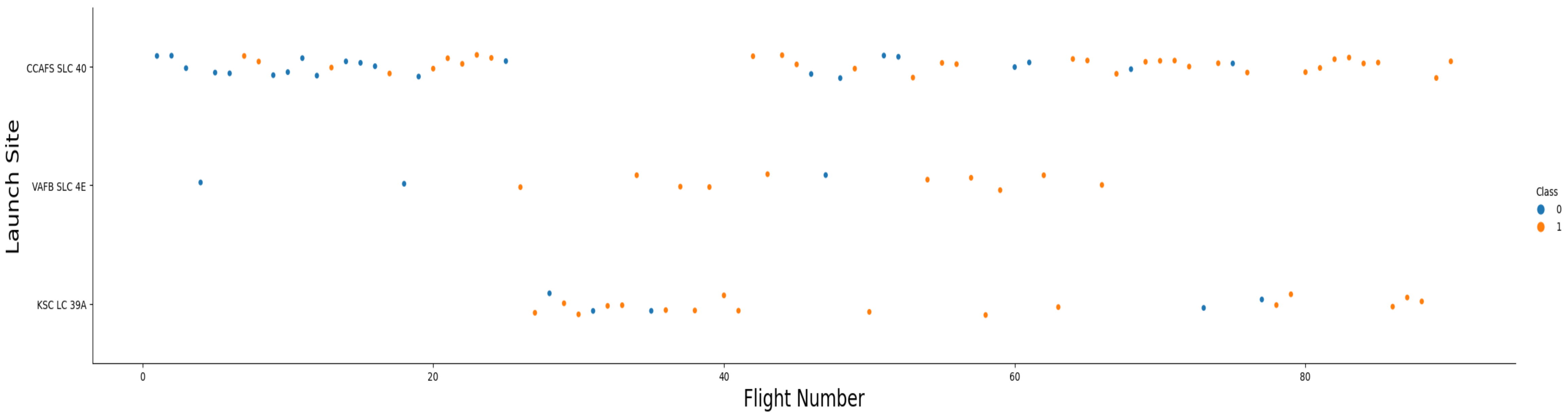
```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.6111111111111112
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

Section 2

Insights drawn from EDA

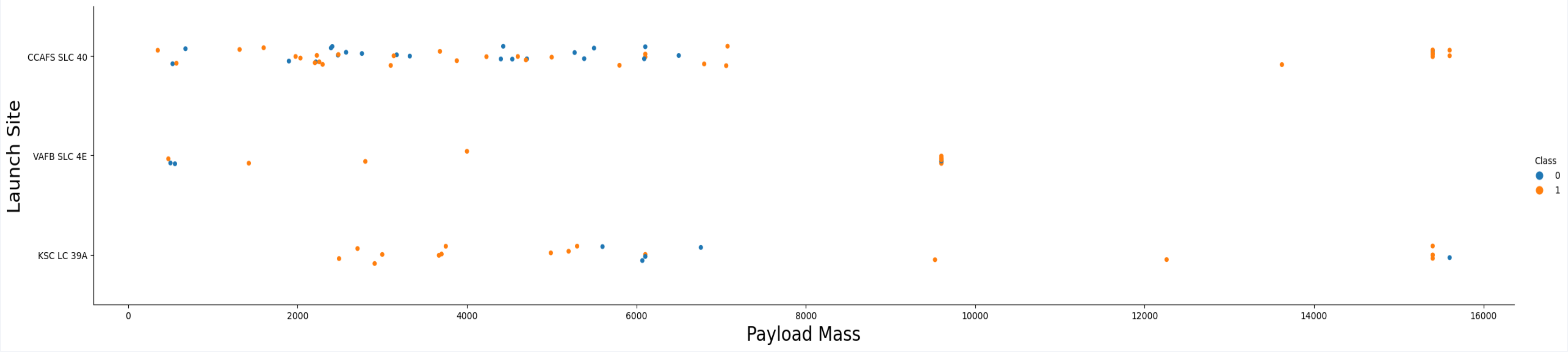
Flight Number vs. Launch Site

- Its clearly visible that launches from CCAFS SLC 40 are higher than from any other site.



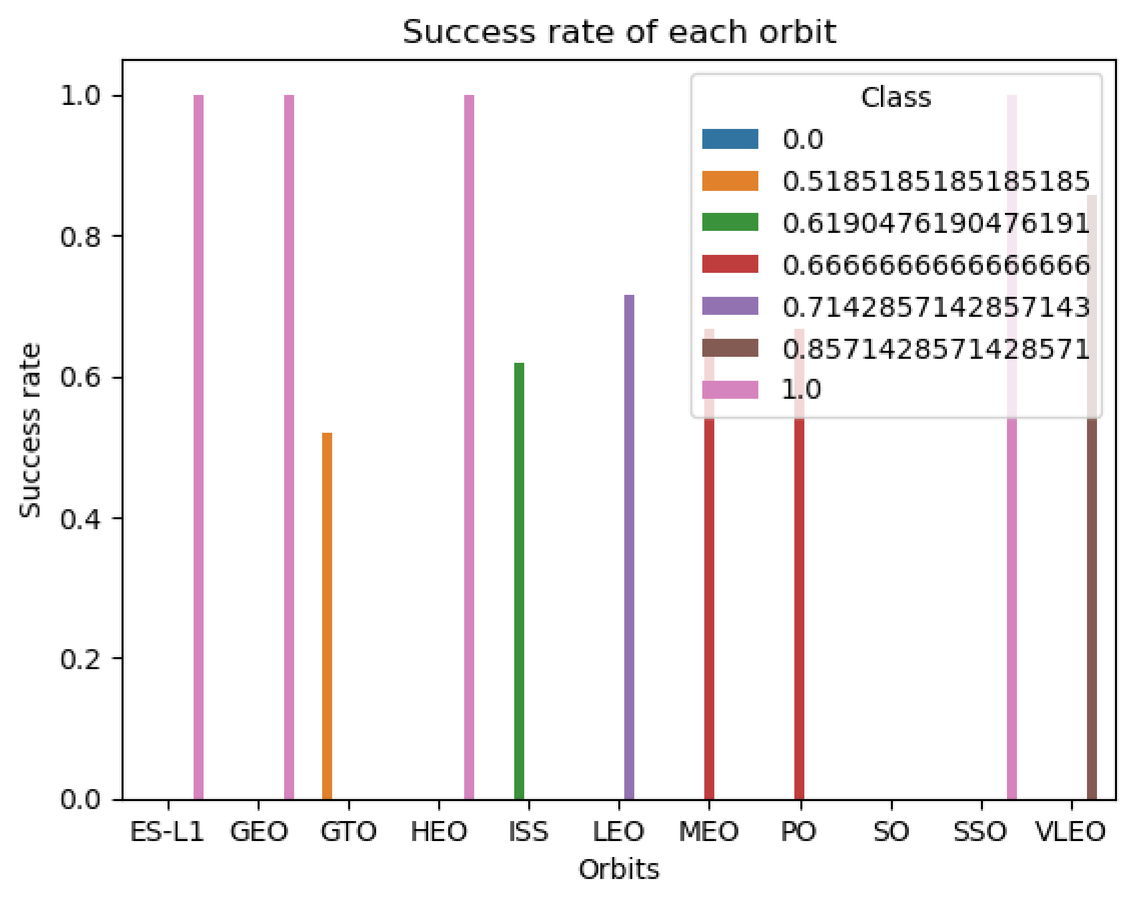
Payload vs. Launch Site

- More lower weight payloads launched are from CCAFS SLCC 40 site.



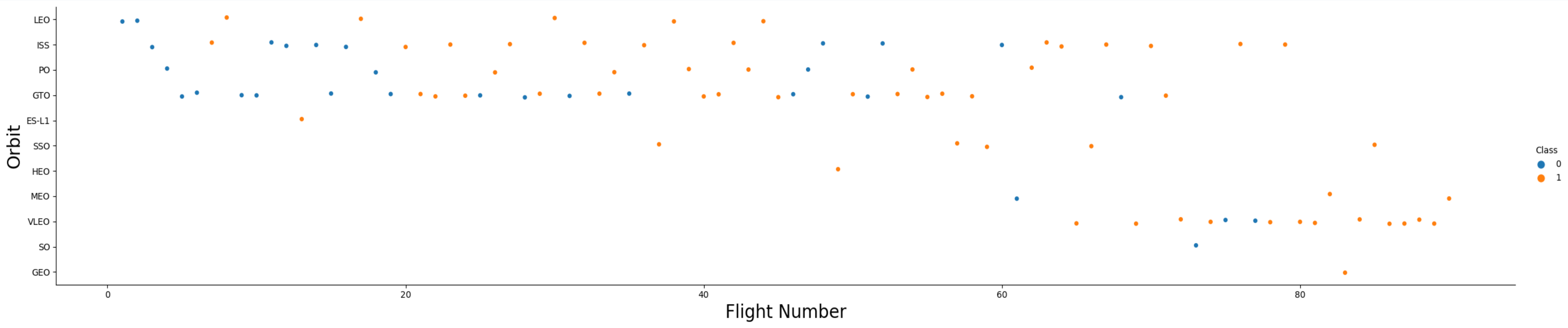
Success Rate vs. Orbit Type

- Orbit types ES L1, GEO, HEO, SSO have highest success rate while GTO has the lowest success rate.



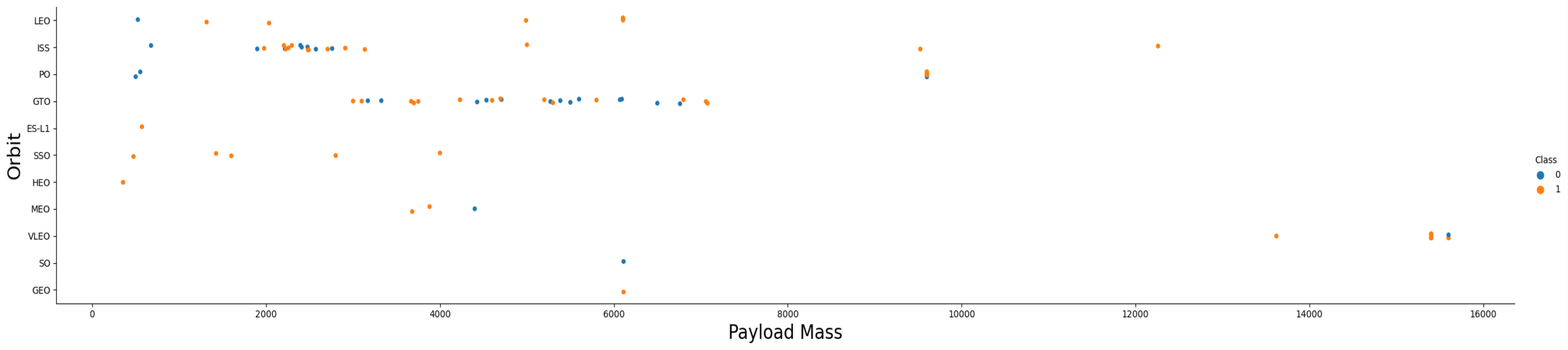
Flight Number vs. Orbit Type

- With time more flights are targeted towards VLEO orbit.



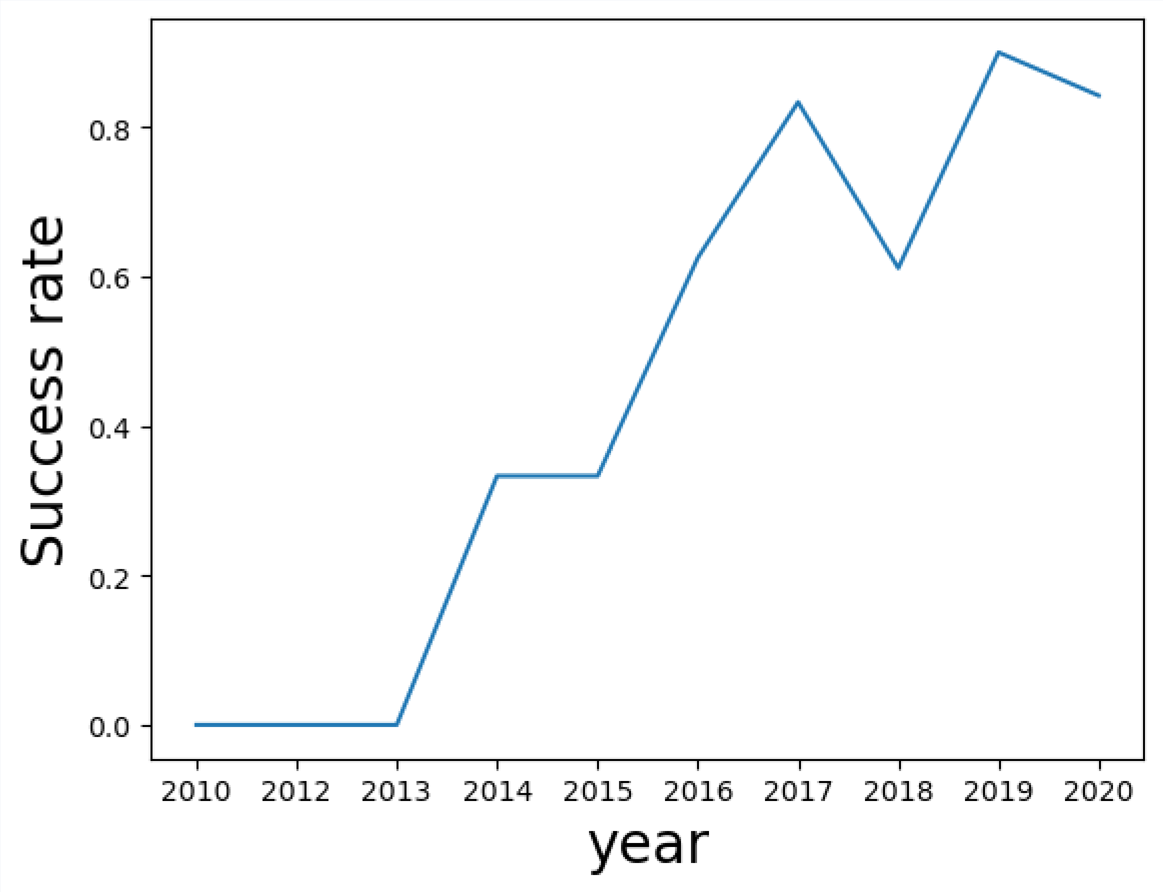
Payload vs. Orbit Type

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



Launch Success Yearly Trend

- Launch success rate has increased significantly since 2013, potentially due to advancement in technology.



All Launch Site Names

- **Query used:** %sql SELECT DISTINCT(Launch_Site) FROM SPACEXTBL limit 5;

```
In [8]: %sql SELECT DISTINCT(Launch_Site) FROM SPACEXTBL limit 5;

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

Out[8]:

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

Launch Site Names Begin with 'CCA'

- %sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%'LIMIT 5;

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	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Query used: %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Customer = 'NASA (CRS)';

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

```
 * 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';

```
%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

First Successful Ground Landing Date

- Query Used: %sql SELECT MIN(Date) FROM SPACEXTBL WHERE [LANDING _OUTCOME] = 'Success (ground pad)';

```
%sql SELECT MIN(Date) FROM SPACEXTBL WHERE [LANDING _OUTCOME] = 'Success (ground pad)';
```

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

MIN(Date)
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

- Query used: %sql SELECT [Booster_Version] FROM SPACEXTBL WHERE [LANDING _OUTCOME] = 'Success (drone ship)' AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000);

```
%sql SELECT [Booster_Version] FROM SPACEXTBL WHERE [LANDING _OUTCOME] = 'Success (drone ship)' AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000);
```

```
 * 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

- %sql SELECT [Mission_Outcome], COUNT([Mission_Outcome]) FROM SPACEXTBL GROUP BY Mission_Outcome;

```
%sql SELECT [Mission_Outcome], COUNT([Mission_Outcome]) FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

Mission_Outcome	COUNT([Mission_Outcome])
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 Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);

```
%sql SELECT Booster_Versio
```

```
 * 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

- %sql SELECT (substr(Date, 4, 2)) as MonthName, [LANDING _OUTCOME], Booster_Version, Launch_Site FROM SPACEXTBL WHERE ([LANDING _OUTCOME] = 'Failure (drone ship)') AND (substr(Date,7,4)='2015');
- 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

```
%sql SELECT (substr(Date, 4, 2)) as MonthName, [LANDING _OUTCOME], Booster_Version, Launch_Site F
```

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

MonthName	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

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

- %sql SELECT Date, COUNT([Landing _Outcome]) AS Successful_Landings FROM SPACEXTBL WHERE [Landing _Outcome] LIKE 'Success%' AND Date BETWEEN '04-06-2010' AND '20-03-2017' GROUP BY "DATE" \ ORDER BY COUNT([Landing _Outcome]) DESC;

```
%sql SELECT Date, COUNT([Landing _Outcome]) AS Successful_Landings FROM SPACEXTBL WHERE [Landing _Outcome] LIKE 'Success%' AND Date BETWEEN '04-06-201
    ORDER BY COUNT([Landing _Outcome]) DESC;
```

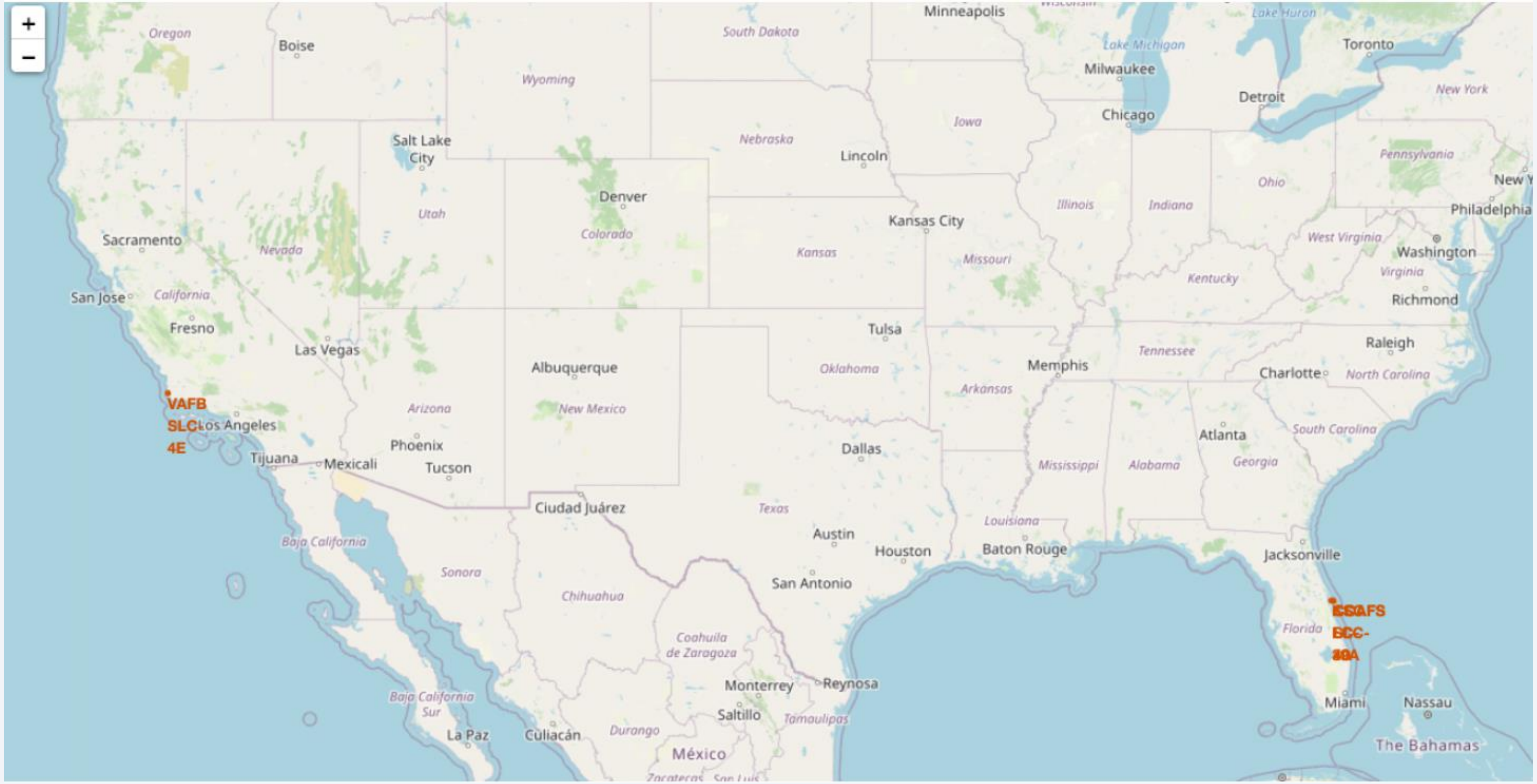
```
 * sqlite:///my_data1.db
Done.
```

Date	Successful_Landings
19-02-2017	1
18-10-2020	1
18-08-2020	1
18-07-2016	1
18-04-2018	1
17-12-2019	1
16-11-2020	1
15-12-2017	1
15-11-2018	1
14-08-2017	1
14-08-2016	1
14-01-2017	1
13-06-2020	1
12-06-2019	1
11-11-2019	1

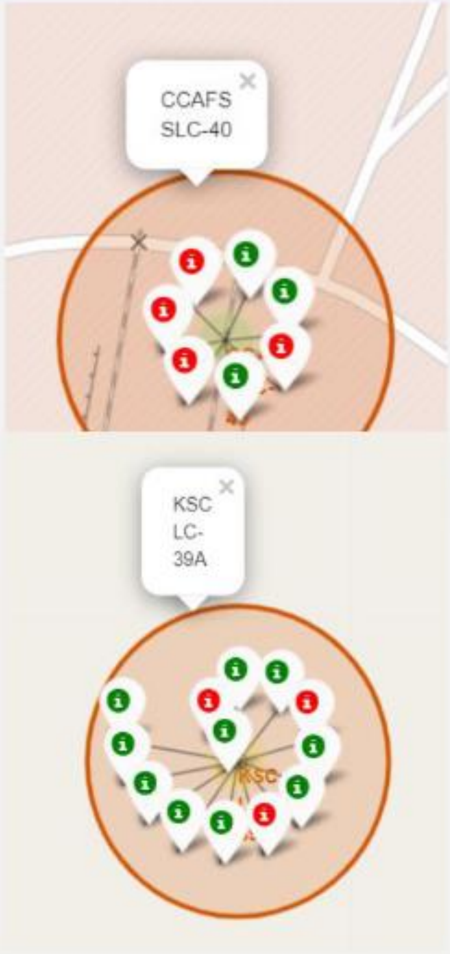
Section 3

Launch Sites Proximities Analysis

All launch sites global map markers



Markers showing launch sites with color labels



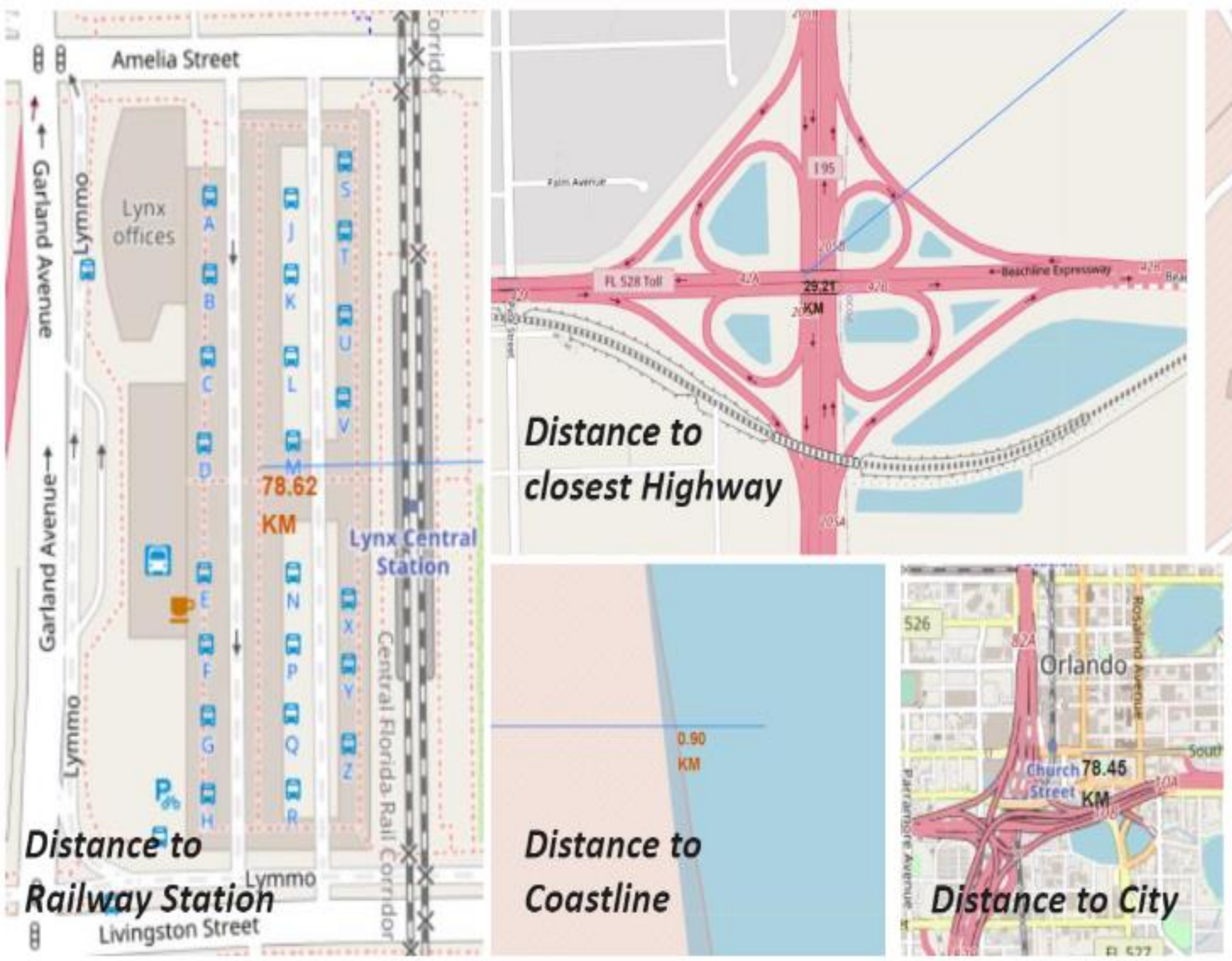
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

37

Launch Site distance to landmarks

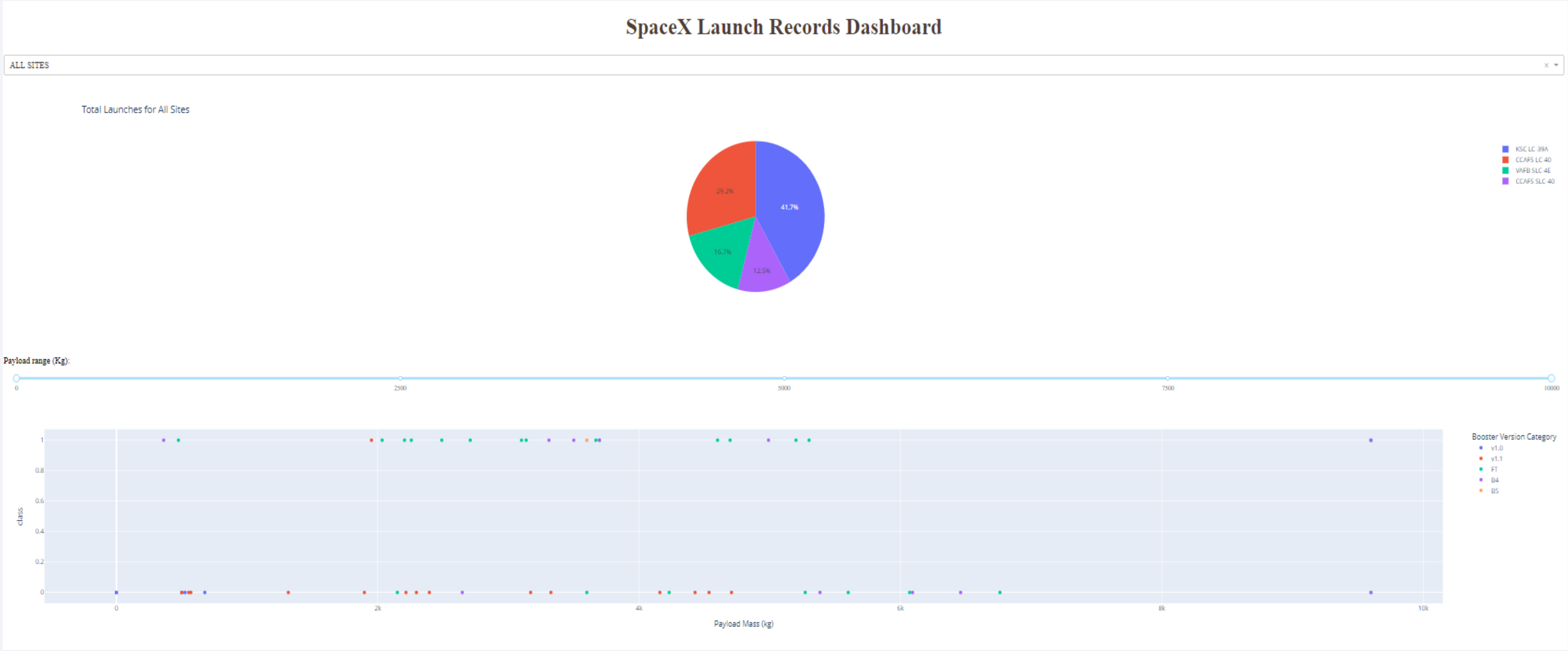


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

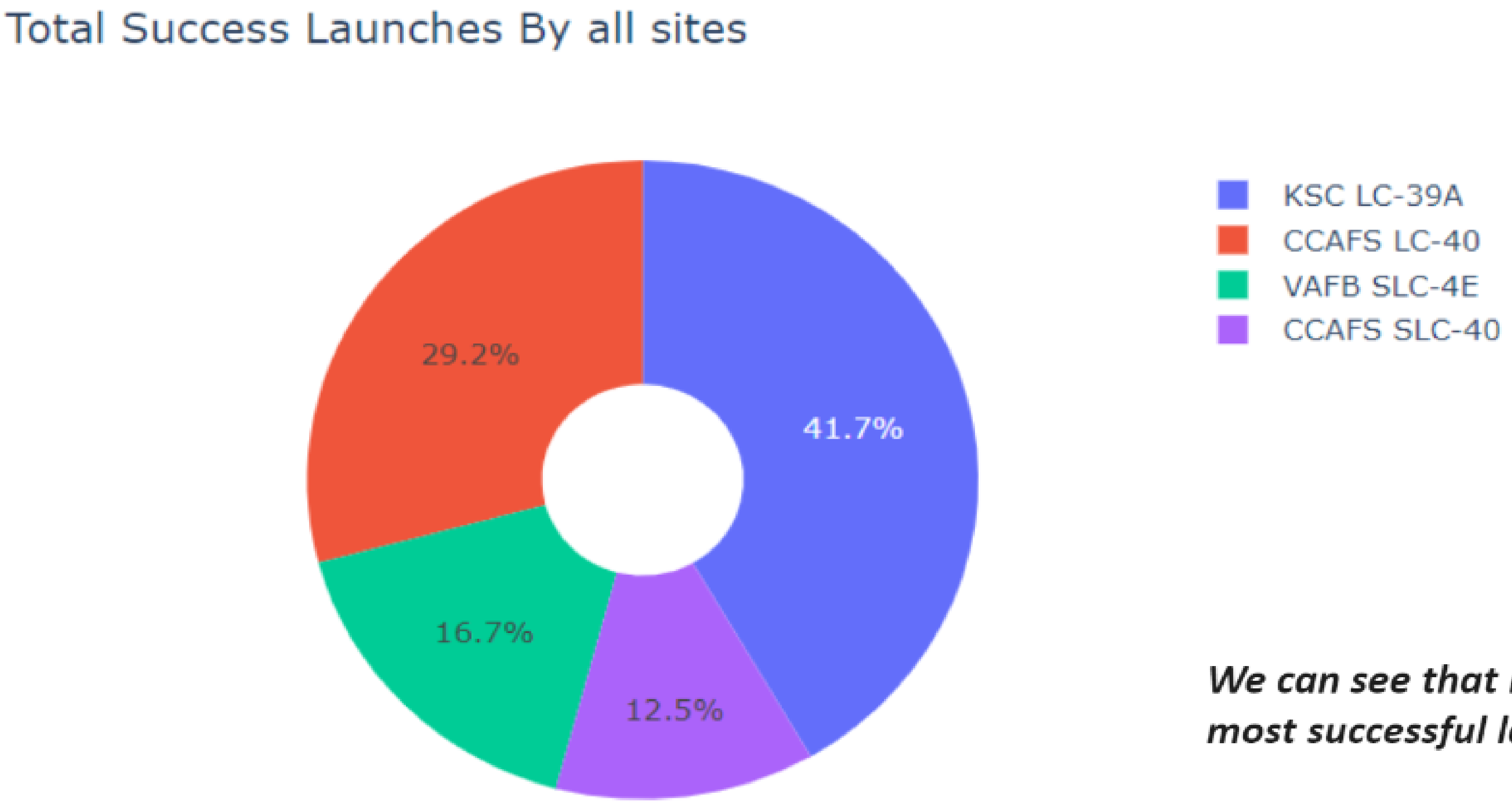
Section 4

Build a Dashboard with Plotly Dash

Plotly dash dashboard

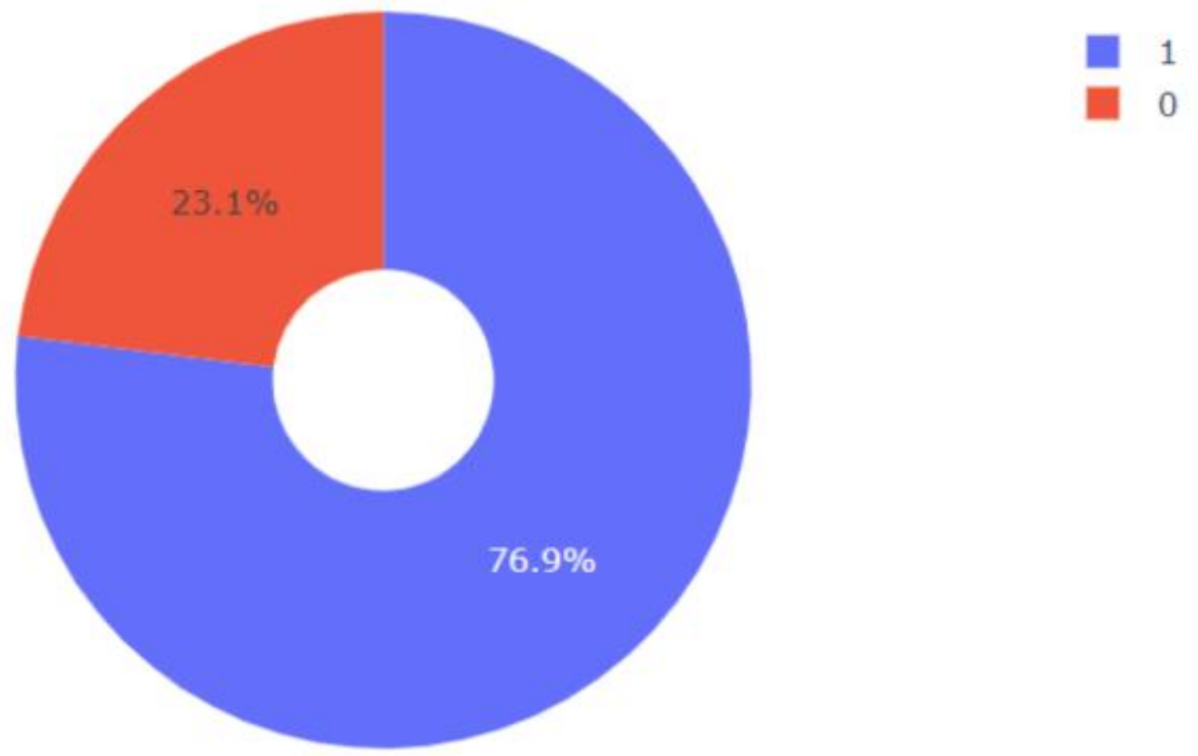


Pie chart showing the success percentage achieved by each launch site



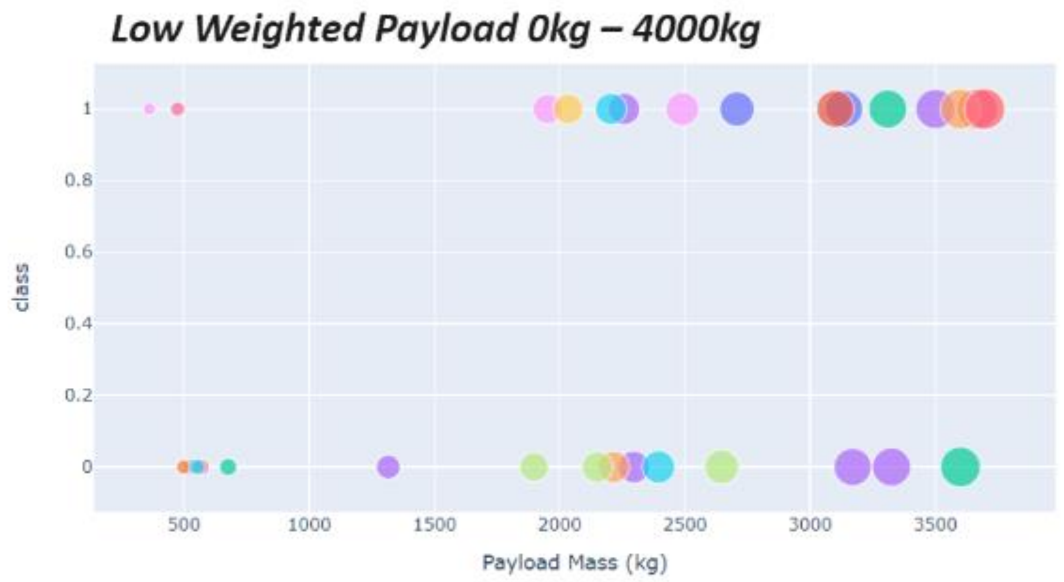
We can see that KSC LC-39A had the most successful launches from all the sites

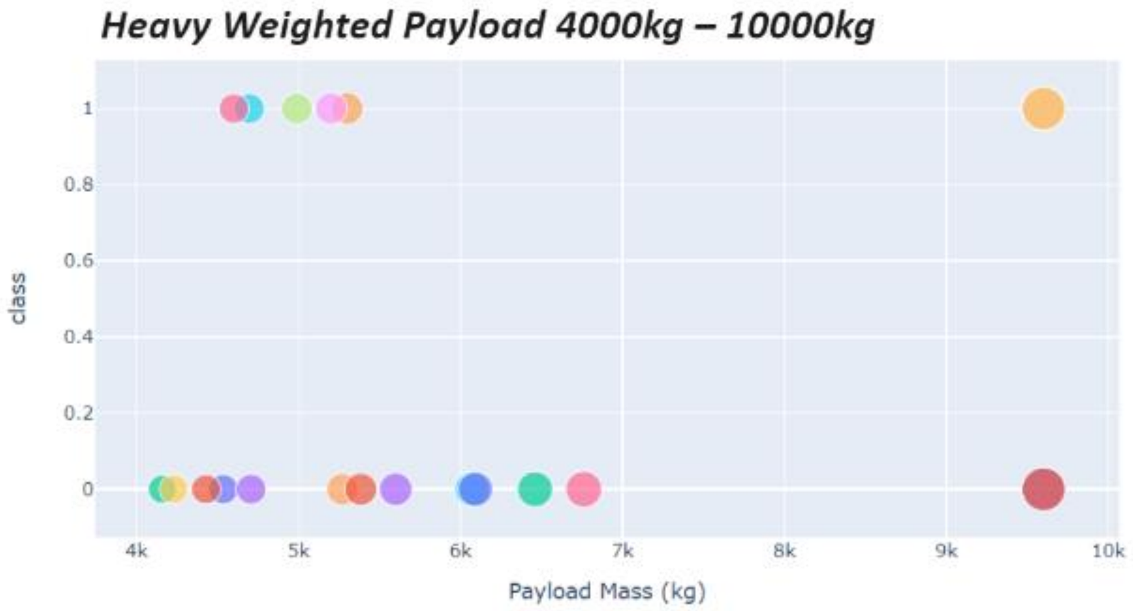
Pie chart showing the Launch site with the highest launch success ratio



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





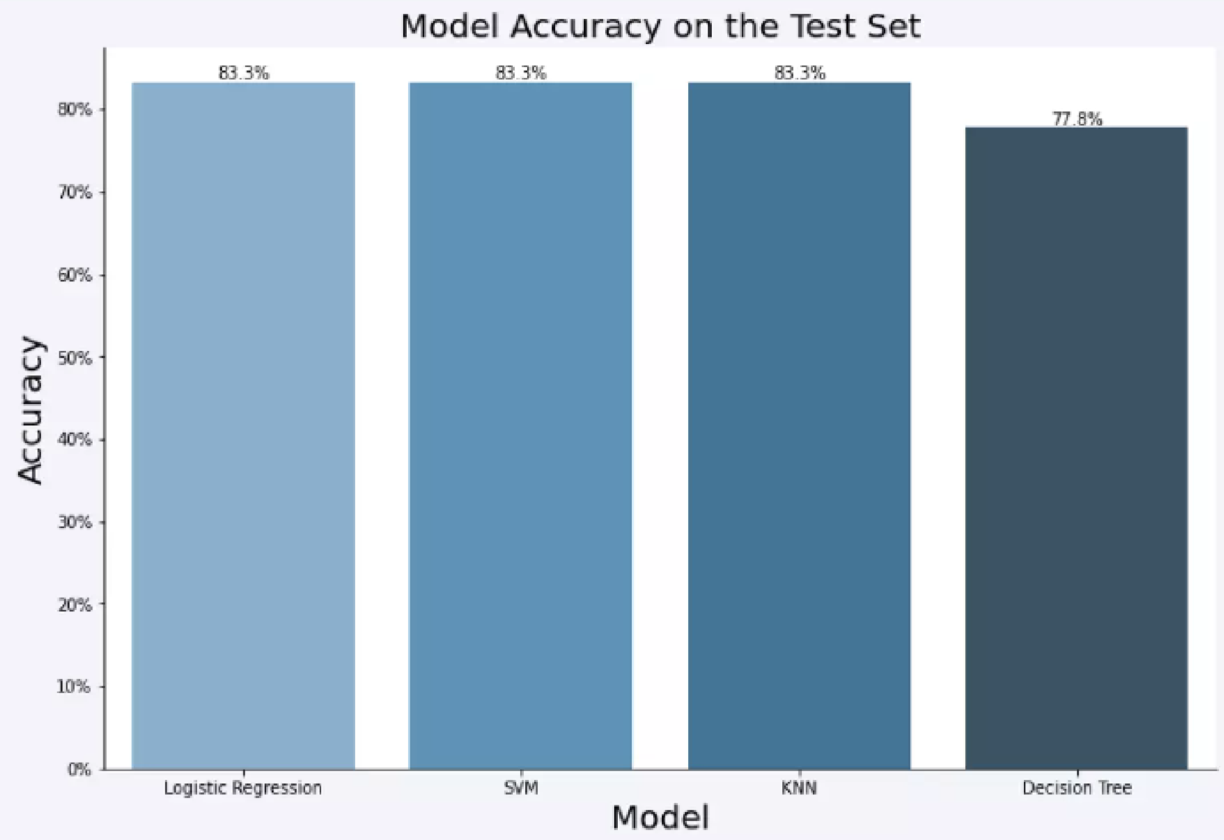
We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

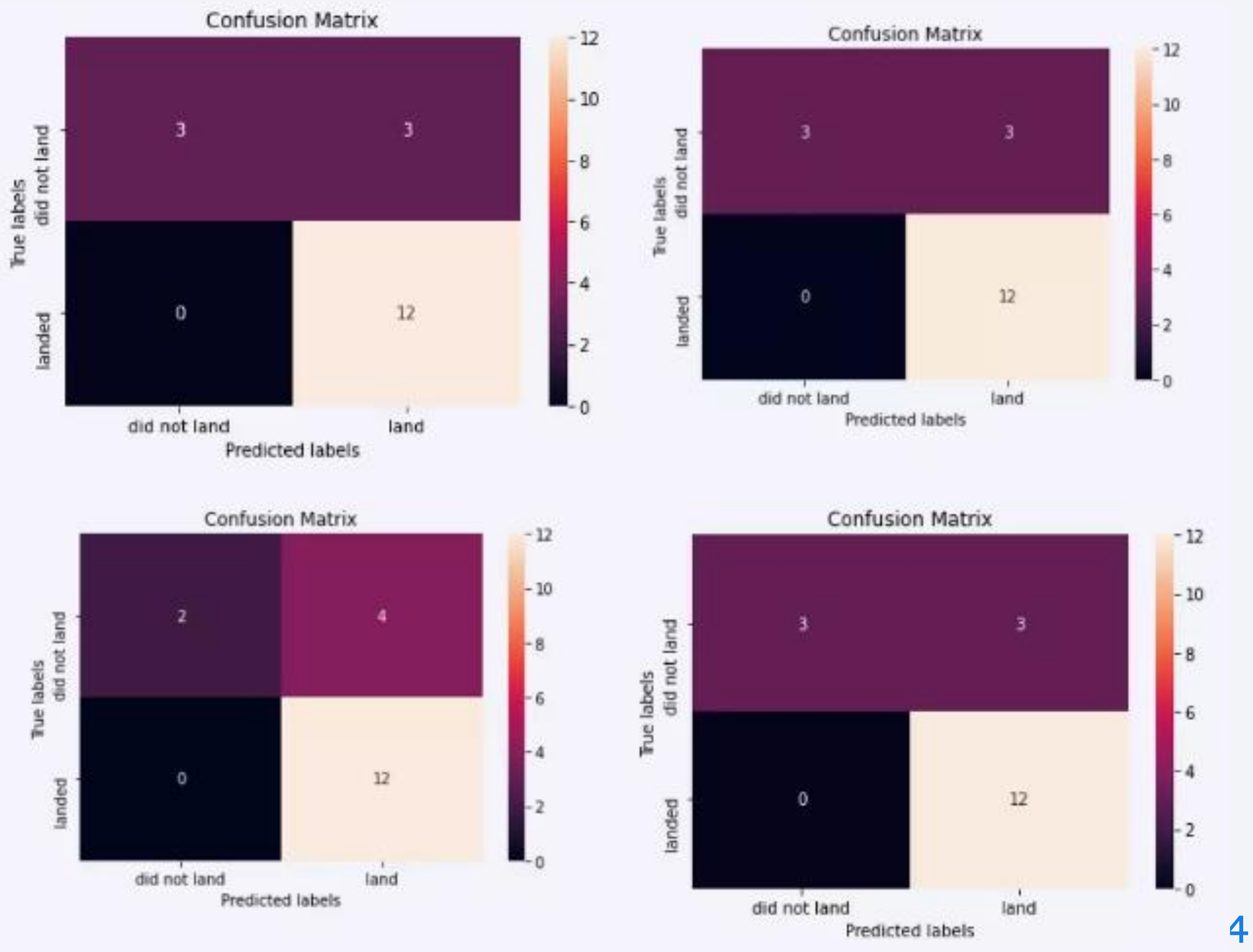
Classification Accuracy

- KNN, LR< SVM has the highest accuracy.



Confusion Matrix

- The confusion matrix shows that classifiers 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 SVM, KNN, LR classifications have best accuracy for this task.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!