Winning Space Race with Data Science

Brian Harris
March 7th 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis Result
  - Interactive analytics in screenshots
  - Predictive Analytics Result

Introduction

- Project background and context

  Space X advertises Falcon 9 Rocket launches for a cost of 62 million dollars. Comparable other providers of this service cost upwards of 165 million dollars each. The savings come from the reuse of the first stage of the launch. Therefore, if we can determine if the first stage is successful, we can determine the cost of the launch. This information can be used if an alternate company wants to place a bid against Space X for a rocket launch. The 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

  - What factors determine if the rocket will land successfully?

  - The interaction amongst various features that determine the success rate of a successful landing.

  - What operating conditions need to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- The data was collected using the following methods:

  - Data Collection was completed using get requests to the Space X API

  - The response was decoded as JSON using .json() function in the python notebook to turn it into a pandas dataframe using .json_normalize()

  - The data was then cleaned, checking for missing values and filling in where necessary

  - Web scraping was performed from Wikipedia for Falcon 9 launch records with BeautifulSoup

  - The objective was to extract the launch records as an HTML table, parse the table and convert it to a pandas dataframe for future analysis

Data Collection – SpaceX API

- The get request was used with the Space X API to collect data, clean the requested data and do some basic data wrangling and formatting
- The link to the notebook is here

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork
```

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

n [10]:
```
response.status_code
```

ut[10]:
```
200
```

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

n [11]:
```
# Use json_normalize meethod to convert the json result into a dataframe
static_json_df = response.json()
data = pd.json_normalize(static_json_df)
```

Using the dataframe `data` print the first 5 rows

n [12]:
```
# Get the head of the dataframe
data.head(5)
```

Data Collection - Scraping

- Web scraping was applied to Falcon 9 launch records with BeautifulSoup
- The table was parsed and converted into a pandas dataframe
- The link to the notebook is here

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
html_data = requests.get(static_url)
html_data.status_code
```

```
200
```

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

```
# 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

```
# Use soup.title attribute
soup.title
```

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

TASK 2: Extract all column/variable names from the HTML table header

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

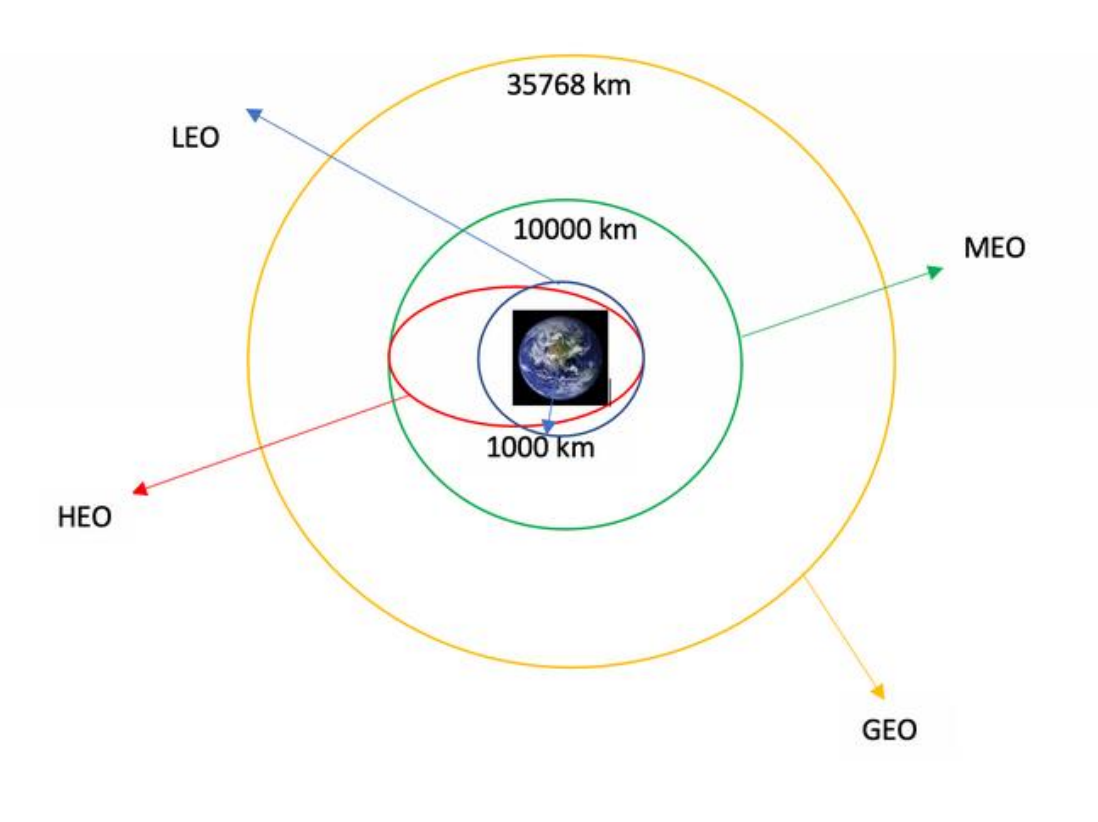
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

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)
```

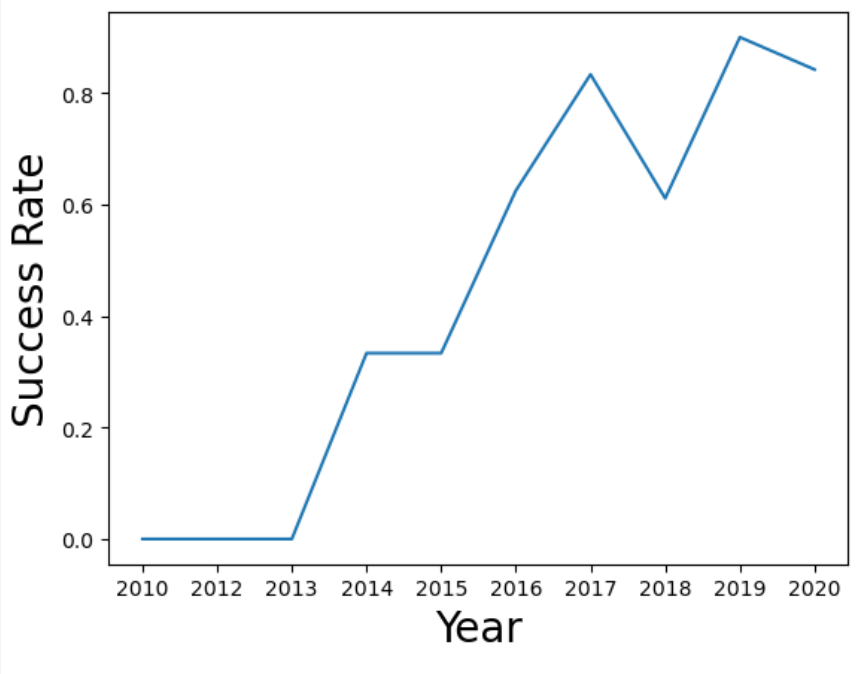
Data Wrangling

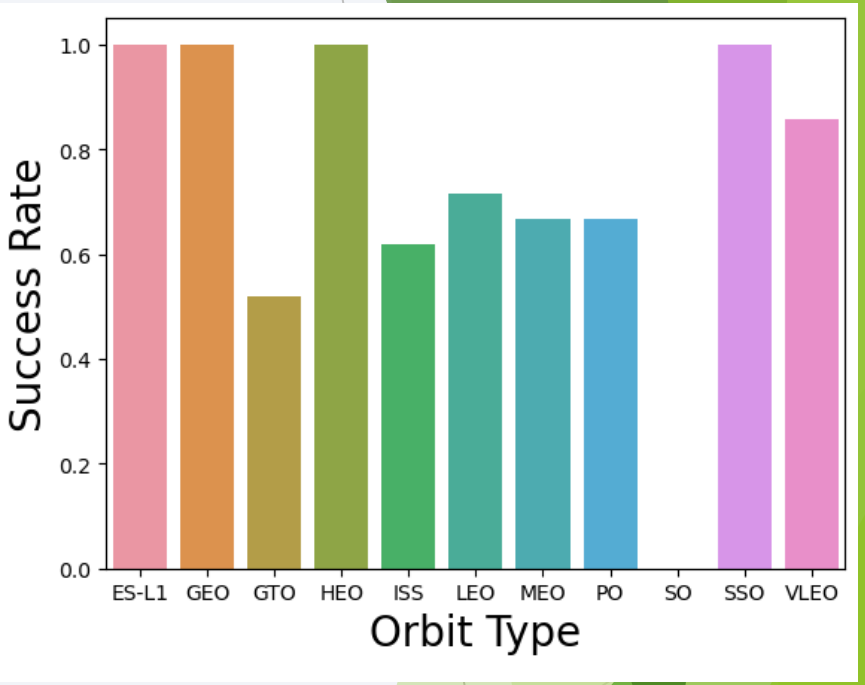
- Exploratory data analysis was performed and training labels were determined
- The number of launches were calculated at each site and the number of occurrence of each orbit
- A landing outcome label was created from the outcome column and results were exported to csv
- The link to the notebook is here



EDA with Data Visualization

- Data was explored and visualized to discover 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
- The link to the notebook is here





EDA with SQL

- Space X dataset was loaded into a SQL database while using jupyter notebook
- Applied EDA with SQL to get insight from the data. Queries were written to find the following:
  - The names of unique launch sites in the space mission
  - 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 link to the notebook is here

Build an Interactive Map with Folium

- Launch sites were marked and map objects were added to the map to mark the success or failure of launches for each site on the folium map.
- The feature launch outcomes(failure or success) were assigned to class 0 and 1
- Using the color-labeled marker clusters, launch sites were identified having high success rates
- The distances were calculated between launch sites and proximities

- The link to the notebook is here

Build a Dashboard with Plotly Dash

- An interactive dashboard was created with Plotly Dash
- Pie charts were created showing the total launches by certain sites
- Scatter graph were plotted showing the relationship with Outcome and Payload Mass (Kg) for the different booster versions.
- The link to the notebook is here

Predictive Analysis (Classification)

- Data was loaded using numpy and pandas. Data was transformed and split for training and testing purposes
- Different machine learning models were built for different parameters using GridSearchCV
- Accuracy was the metric used for our model and the model was improved using feature engineering and algorithm testing
- The best performing classification model was formed
- The link to the notebook is here

Results

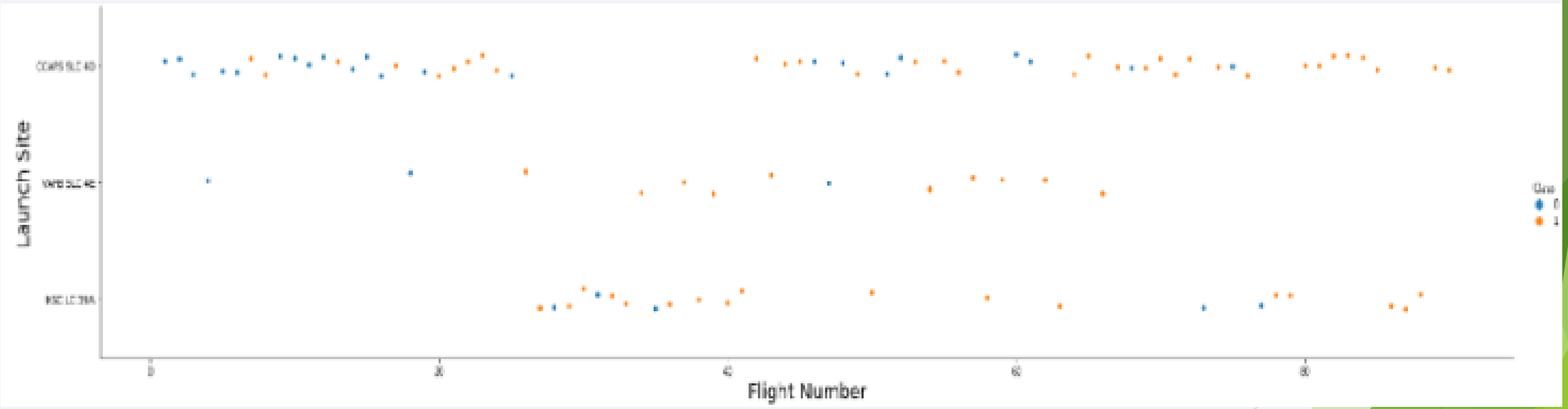
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

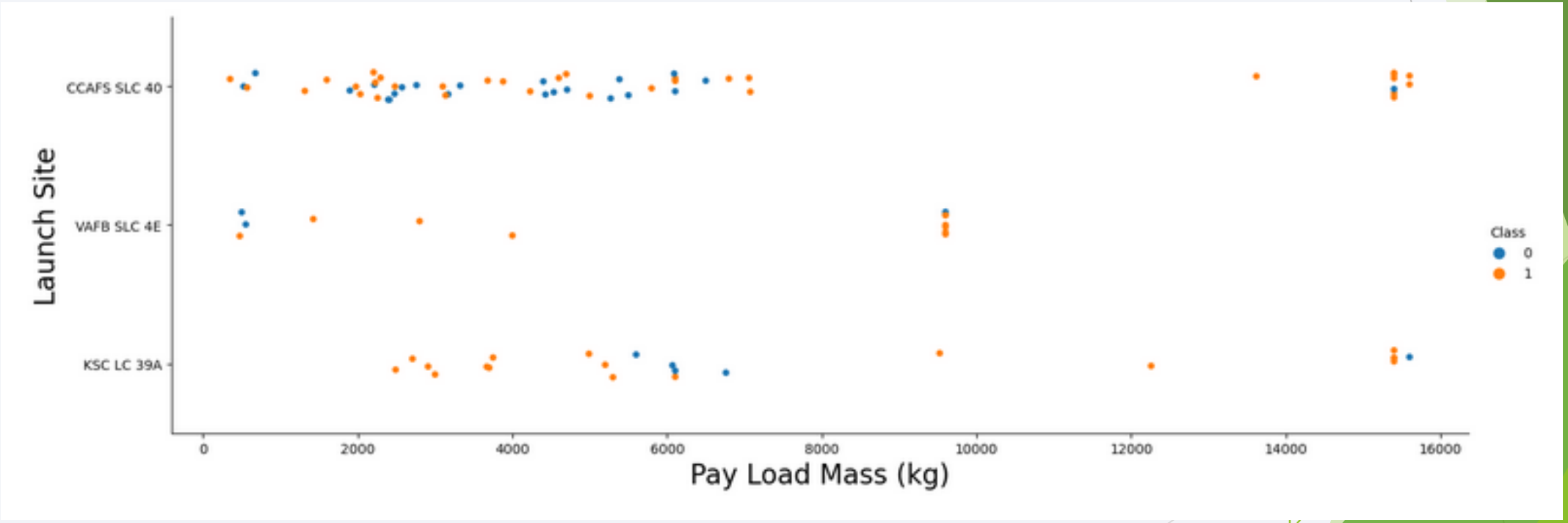
Flight Number vs. Launch Site

- This Plot shows that the larger the flight amount at a launch site, the greater the success rate at a launch site



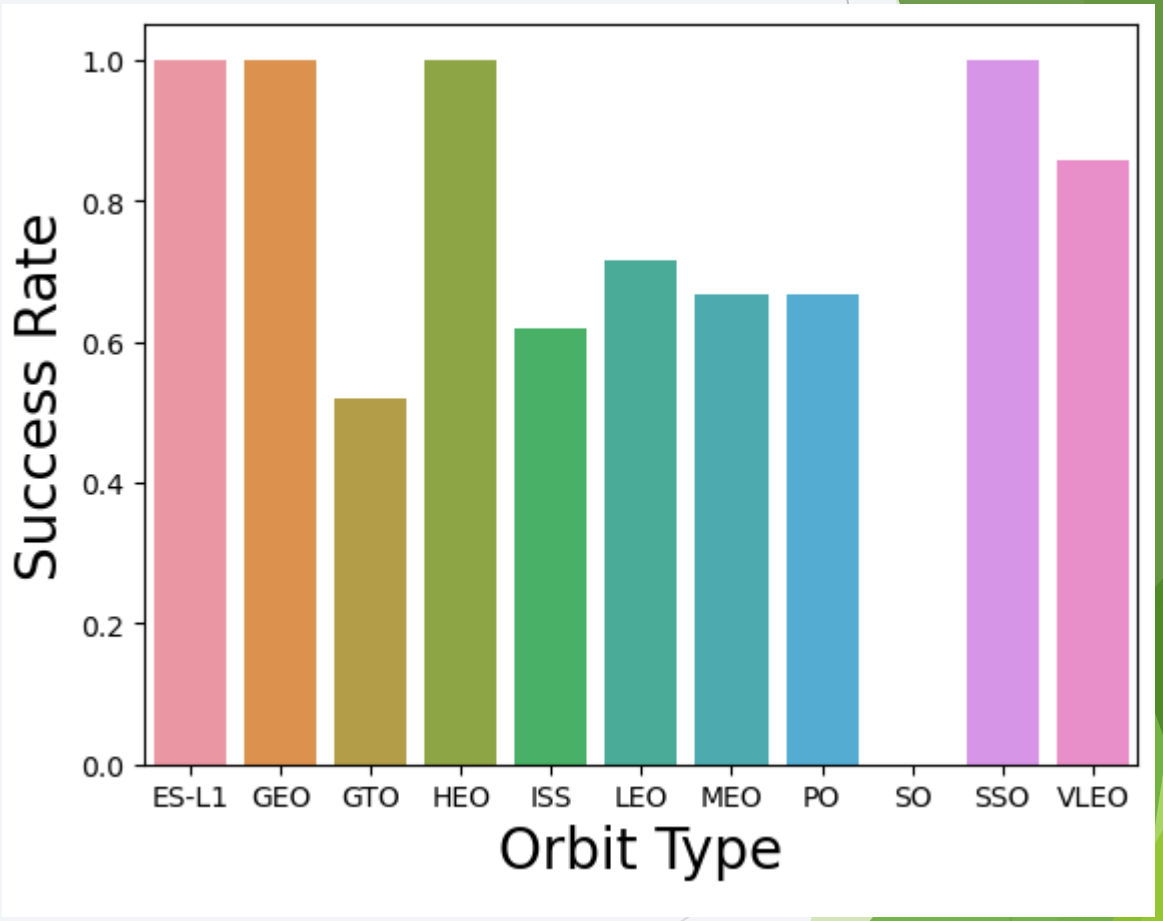
Payload vs. Launch Site

- The Greater the payload mass for Launch Site CCAFS SLC 40, the higher the success rate for the rocket



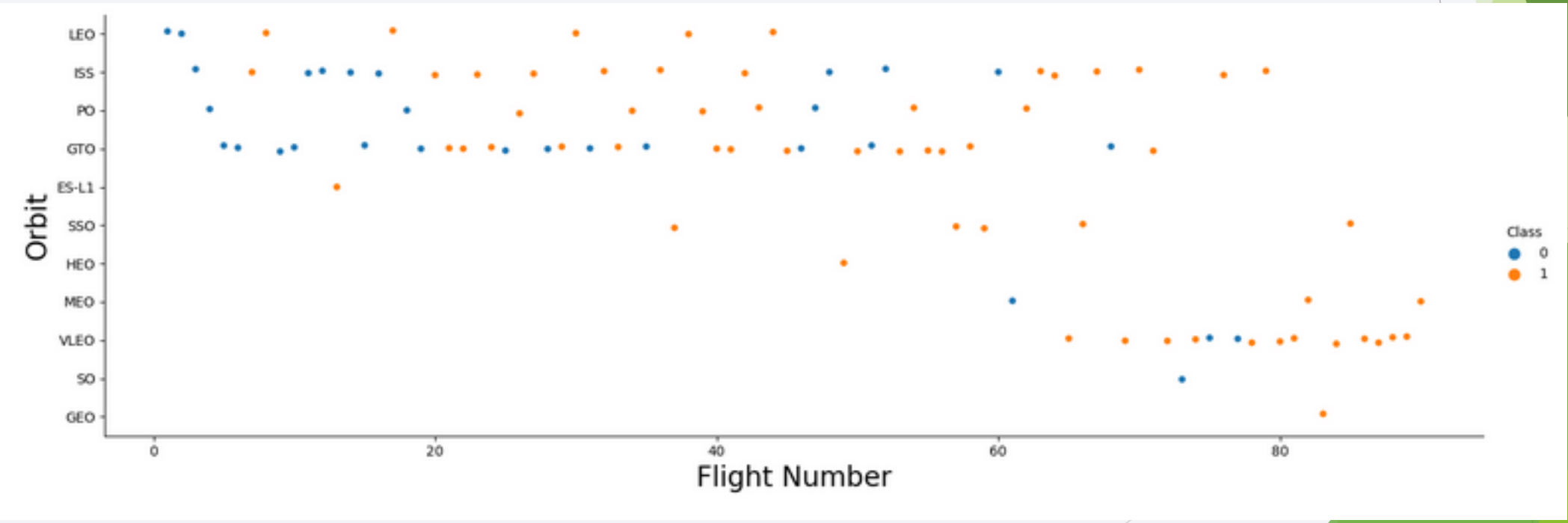
Success Rate vs. Orbit Type

- The most success rates are captured within ES-L1, GEO, HEO, SSO orbit types. This is illustrated in the bar chart here.



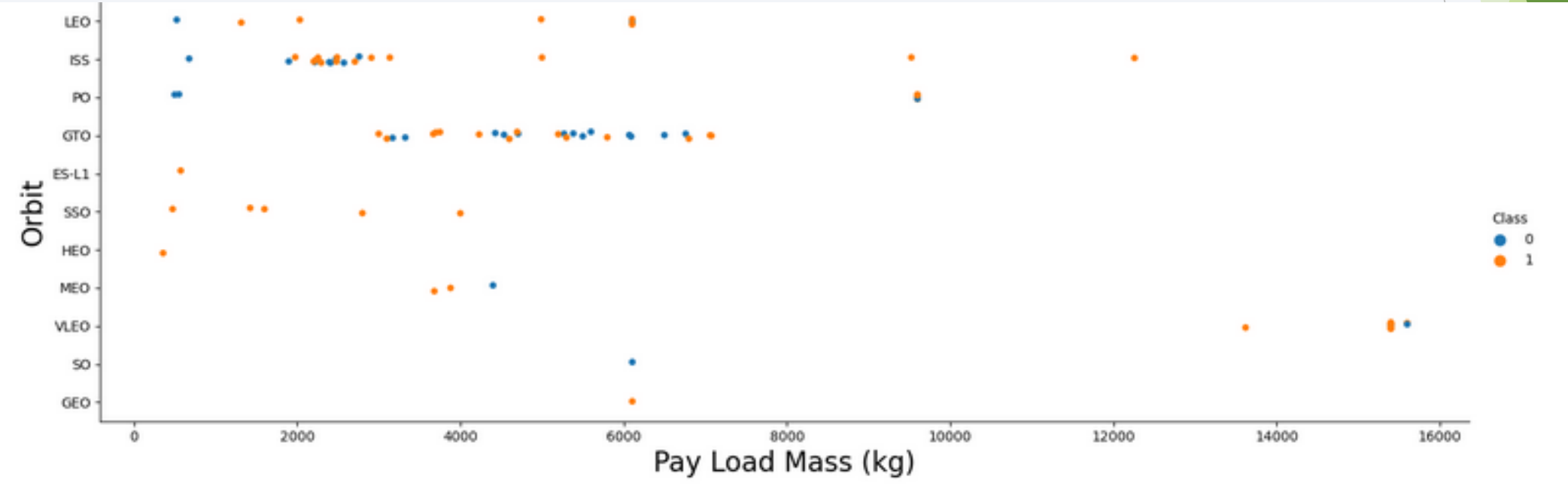
Flight Number vs. Orbit Type

- The below plot displays Flight Number vs. Orbit type. In LEO orbit, success is related to the number of flights where in GTO orbit there is no observable relationship.



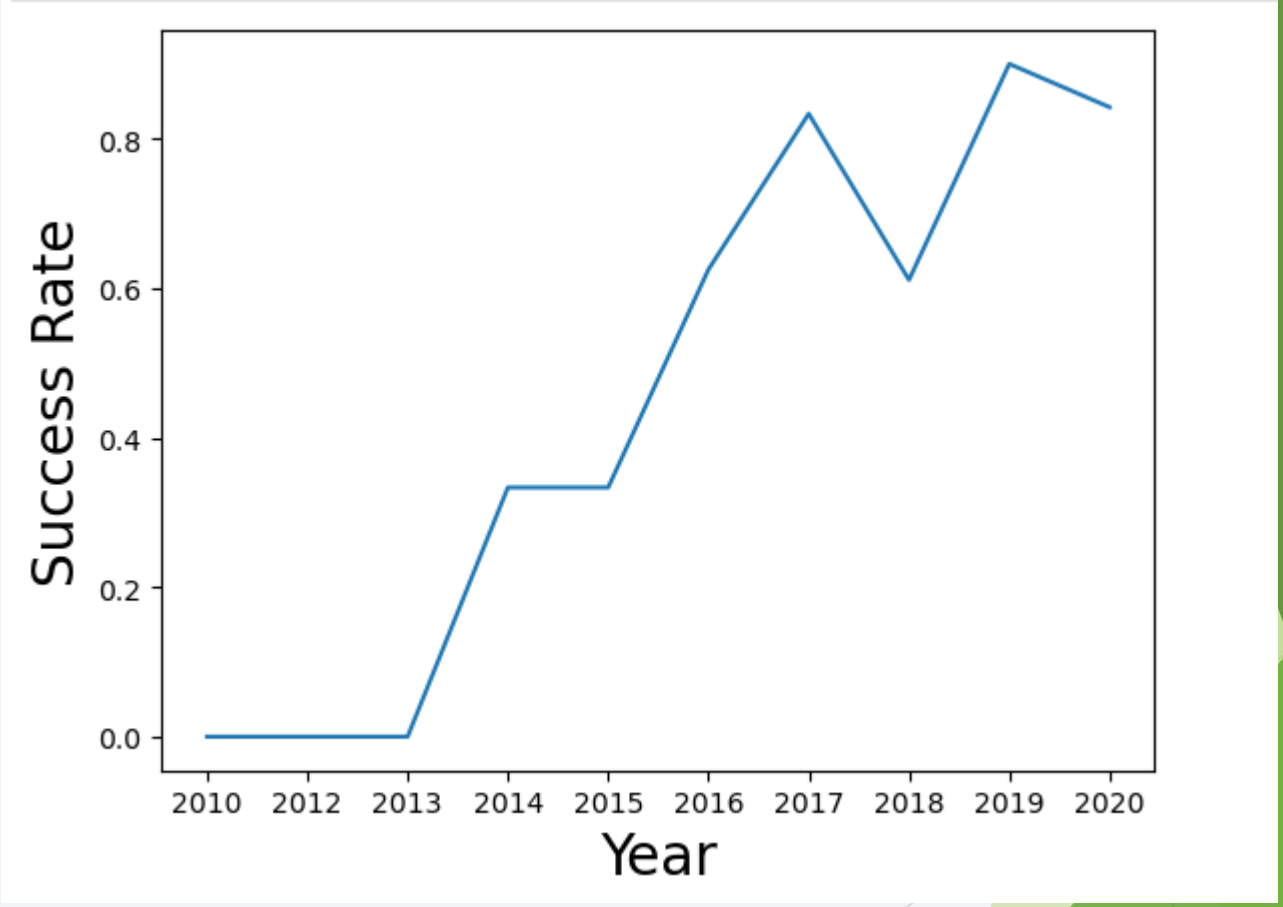
Payload vs. Orbit Type

- Below is the plot showing the less to middle payloads are most successful landings for PO, LEO, and ISS orbits.



Launch Success Yearly Trend

- In the plot we can observe the increasing success rate starting at 2013 to 2020.



All Launch Site Names

- Using the query for DISTINCT launch sites, we returned the correct listed results from the data.

```
%sql SELECT Distinct LAUNCH_SITE FROM SPACEXTBL
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

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

Launch Site Names Begin with 'CCA'

- Using the SQL query Like ‘CCA%’ with a limit, brings the launch sites containing those three consecutive letters in their names and limiting the output to the first 5.

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

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

DATE	time_utc_	booster_version	launch_site	payload	payload_mass_kg_	orbit	customer	mission_outcome	landing_outcc
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	None	0	LEO	SpaceX	Success	Failure (parach
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	None	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parach
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	None	525	LEO (ISS)	NASA (COTS)	Success	No atte
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	None	500	LEO (ISS)	NASA (CRS)	Success	No atte
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	None	677	LEO (ISS)	NASA (CRS)	Success	No atte

Total Payload Mass

- Using the SUM function the total payload mass for the customer ‘NASA’ was calculated from the table.

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

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

1
45596

Average Payload Mass by F9 v1.1

- Using the AVG function, the average payload mass is found from the F9 v1.1 booster version.

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION='F9 v1.1'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

1
2928.400000

First Successful Ground Landing Date

- Using the min function, the first date where the landing outcome = ‘Success’ is found from the table.

```
%sql SELECT min(DATE) FROM SPACEXTBL WHERE LANDING__OUTCOME='Success (ground pad)'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Using WHERE we can find the payload mass between 4000 and 6000 and having a successful landing

```
%sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ between 4000 and 6000 AND LANDING__OUTCOME=
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
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

- Using the COUNT function and LIKE we can find the types of success and failure by number

```
%sql SELECT COUNT(*) FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE '%Success%' OR MISSION_OUTCOME LIKE '%Failure%'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

1
101

Boosters Carried Maximum Payload

- We determined the booster that 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 SPACE
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
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

- Using the WHERE, LIKE, AND the query filters for failed outcomes in drone ship, their booster versions and launch sites for the year 2015.

```
%sql SELECT TO_CHAR(TO_DATE(MONTH("DATE"), 'MM'), 'MONTH') AS MONTH_NAME, \
    LANDING__OUTCOME AS LANDING__OUTCOME, \
    BOOSTER_VERSION AS BOOSTER_VERSION, \
    LAUNCH_SITE AS LAUNCH_SITE \
    FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND "DATE" LIKE '%2015%'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

month_name	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

- 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 and 2017-03-20.
- Group BY was also applied to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order.

```
%sql SELECT "DATE", COUNT(LANDING__OUTCOME) as COUNT FROM SPACEXTBL \
    WHERE "DATE" BETWEEN '2010-06-04' and '2017-03-20' AND LANDING__OUTCOME LIKE '%Success%' \
    GROUP BY "DATE" \
    ORDER BY COUNT(LANDING__OUTCOME) DESC
```

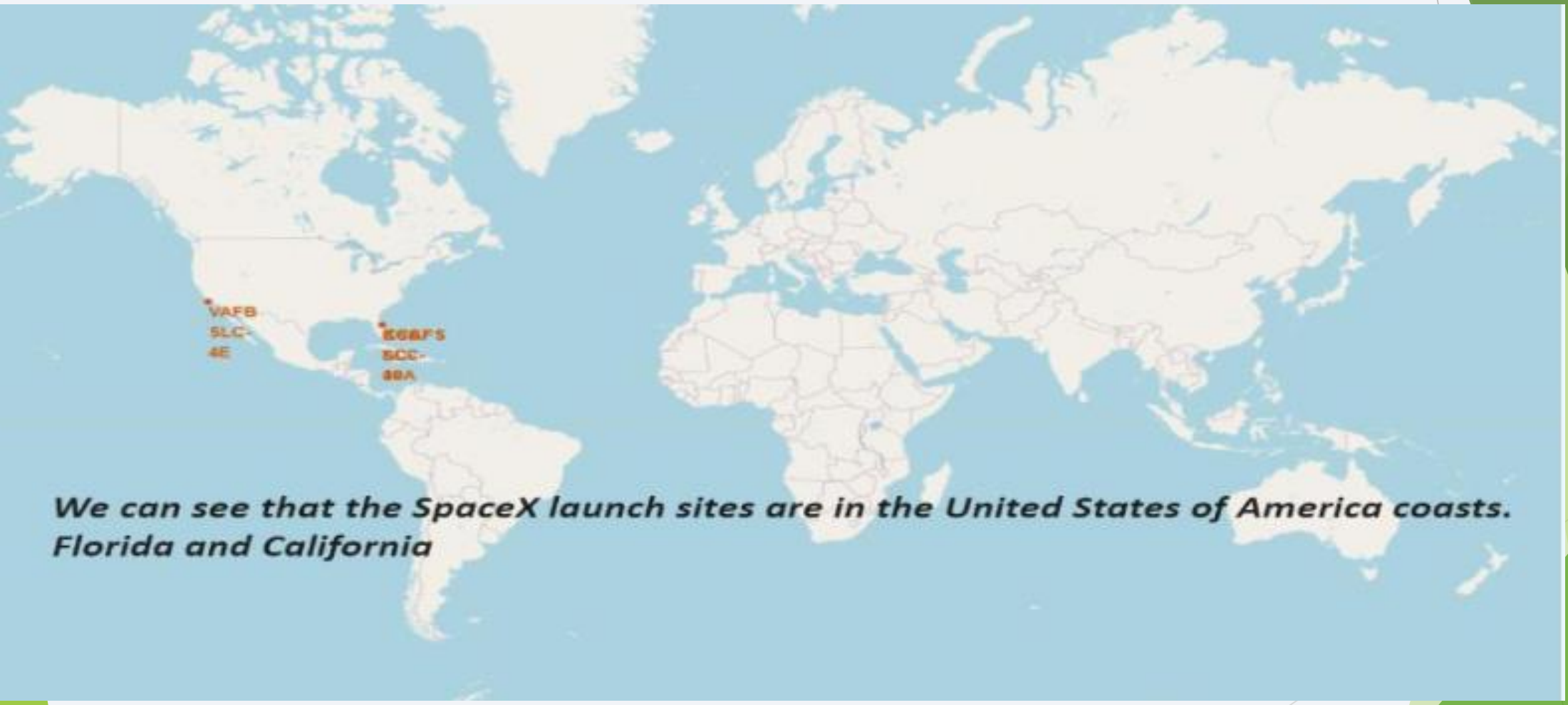
```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

DATE	COUNT
2015-12-22	1
2016-04-08	1
2016-05-06	1
2016-05-27	1
2016-07-18	1
2016-08-14	1
2017-01-14	1
2017-02-19	1

Section 3

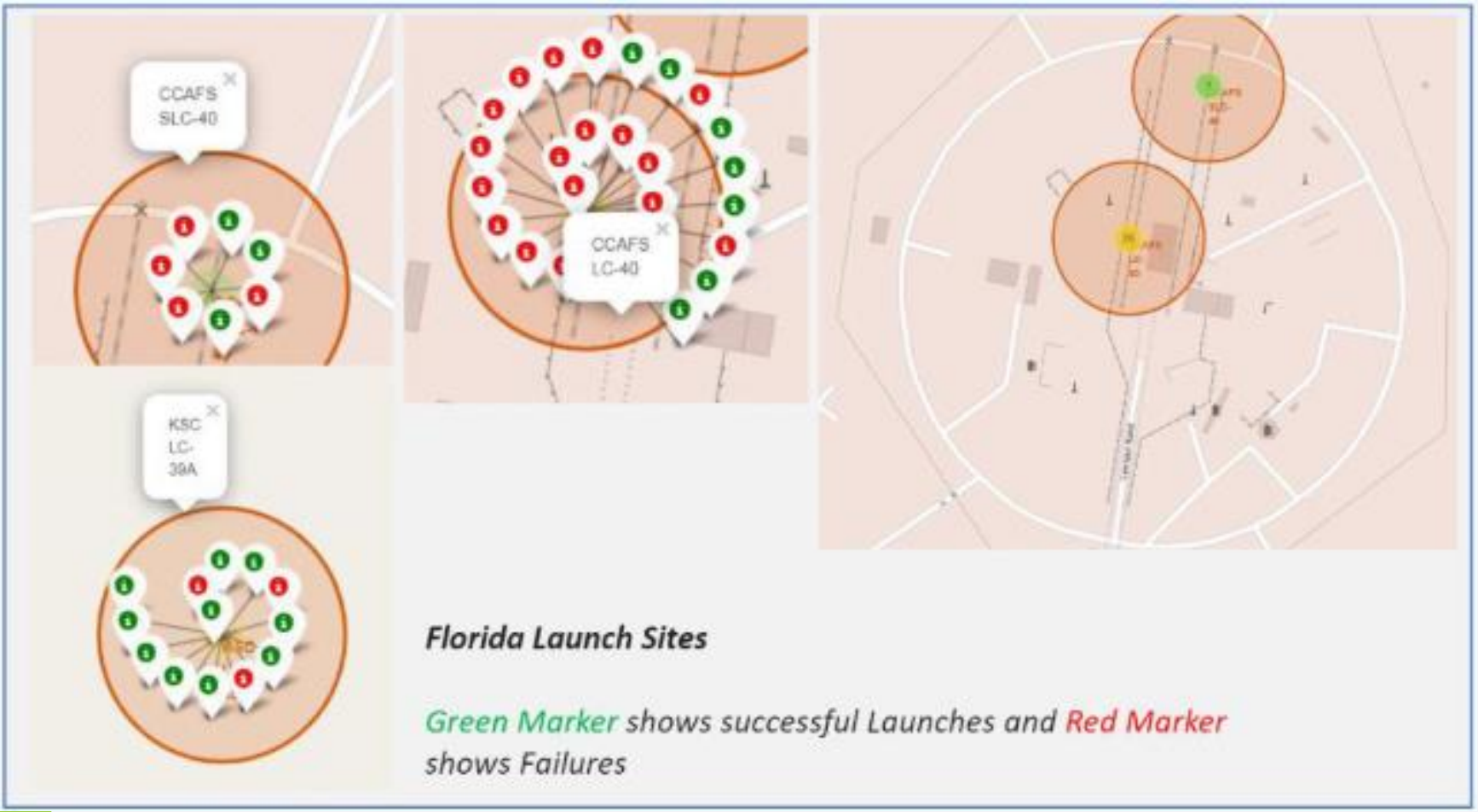
Launch Sites Proximities Analysis

All Launch Sites Global Map Markers



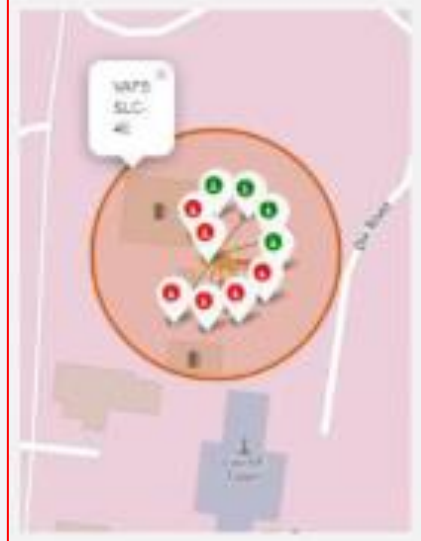
We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

Markers Showing Launch Sites with Color Labels



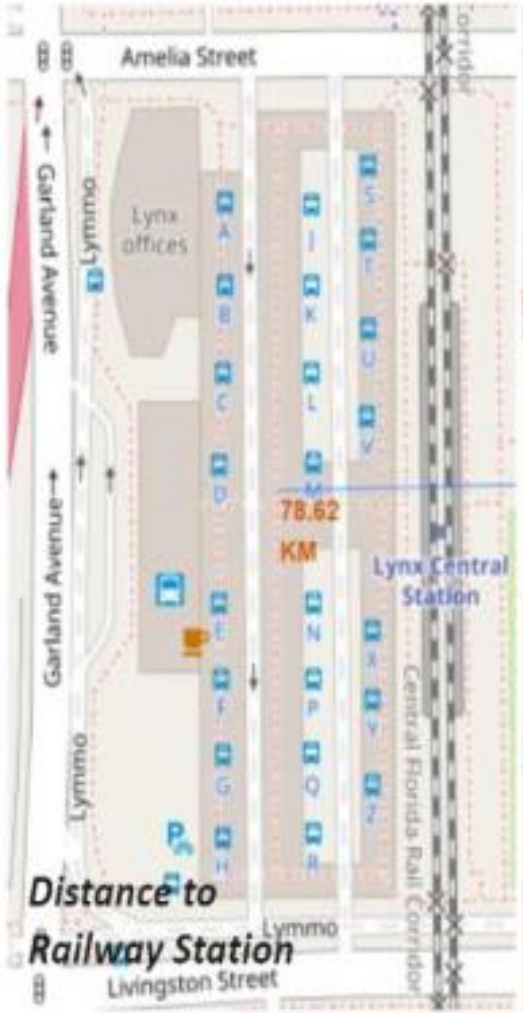
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

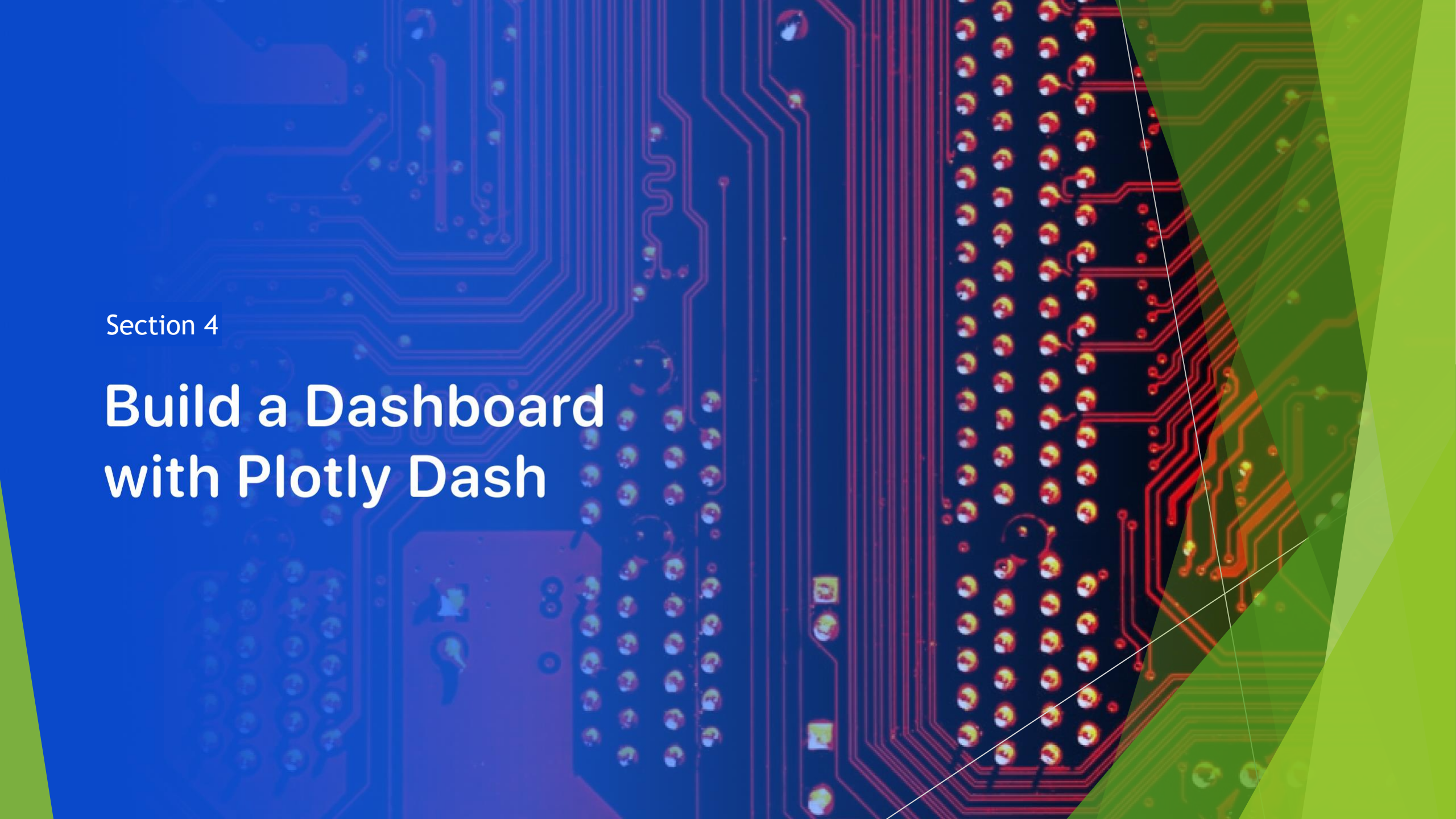


California Launch Site

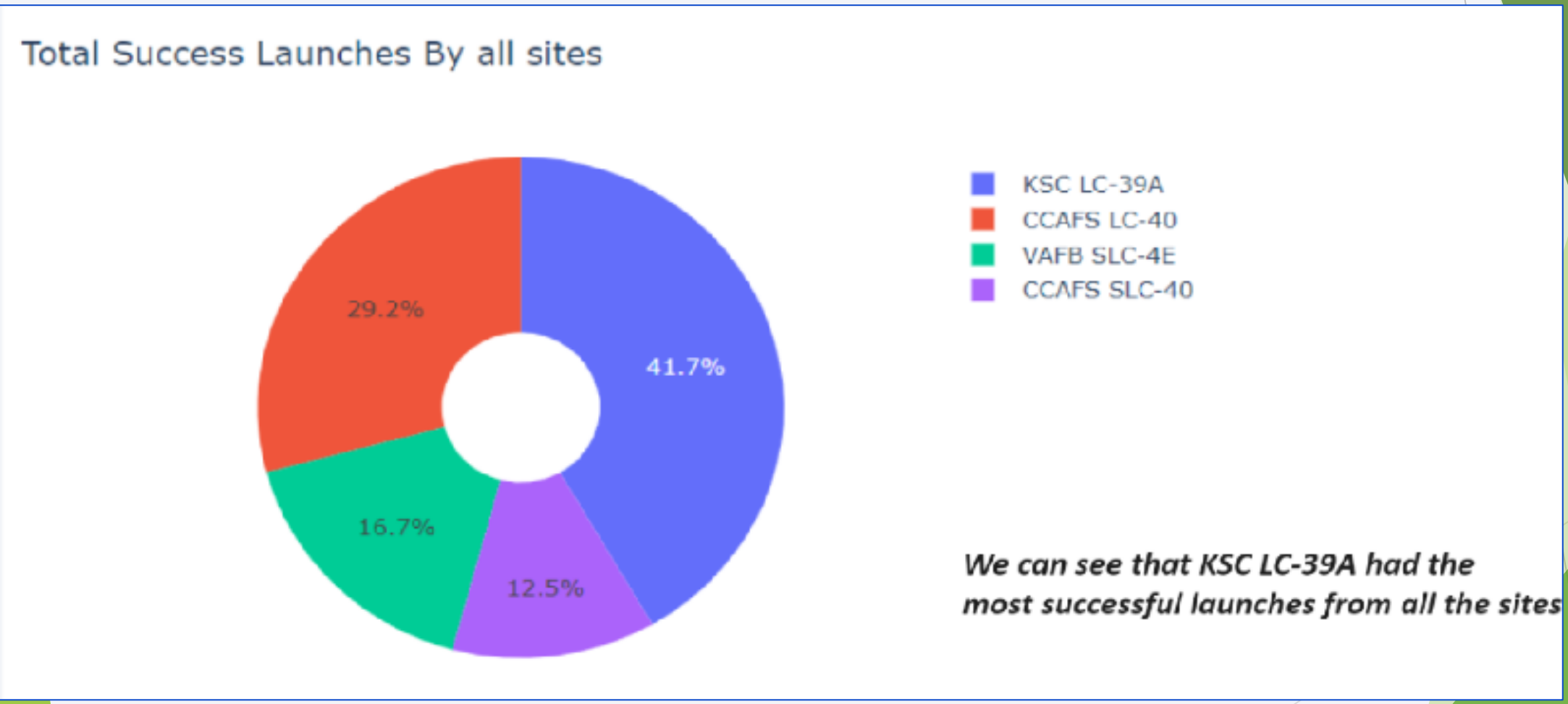
Launch Site Distance to Landmarks



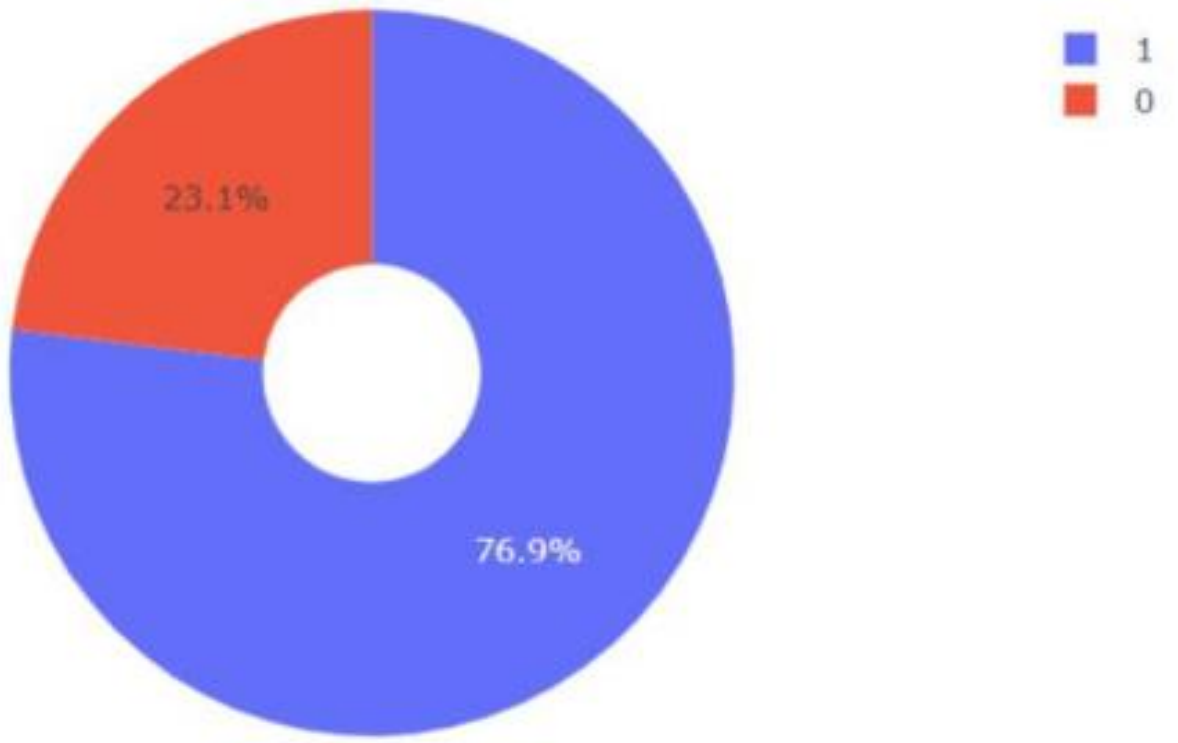
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes


Section 4
Build a Dashboard with Plotly Dash

Pie Chart Showing the Success Percentage Achieved by Each Launch Site

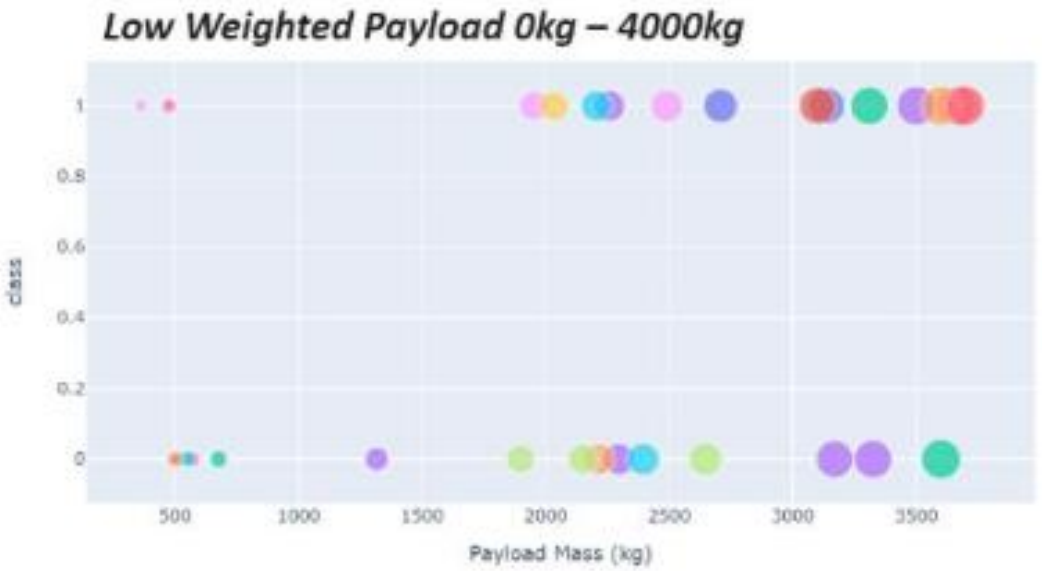


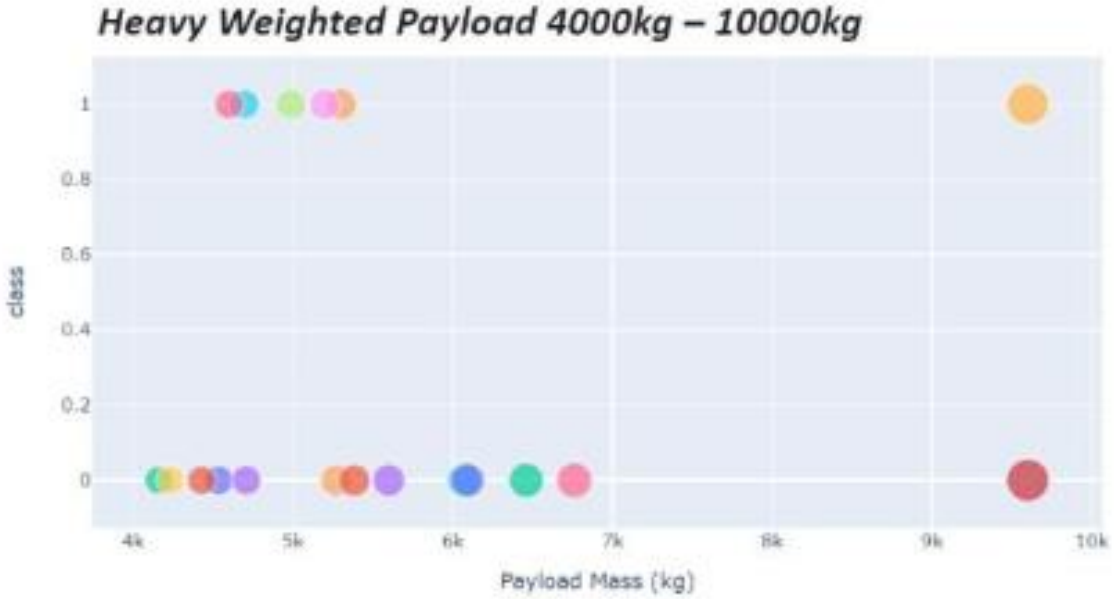
Pie Chart Showing the Launch Site with the Highest Launch Success Ratio



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

- The Decision Tree classifier is the model with the highest classification accuracy.

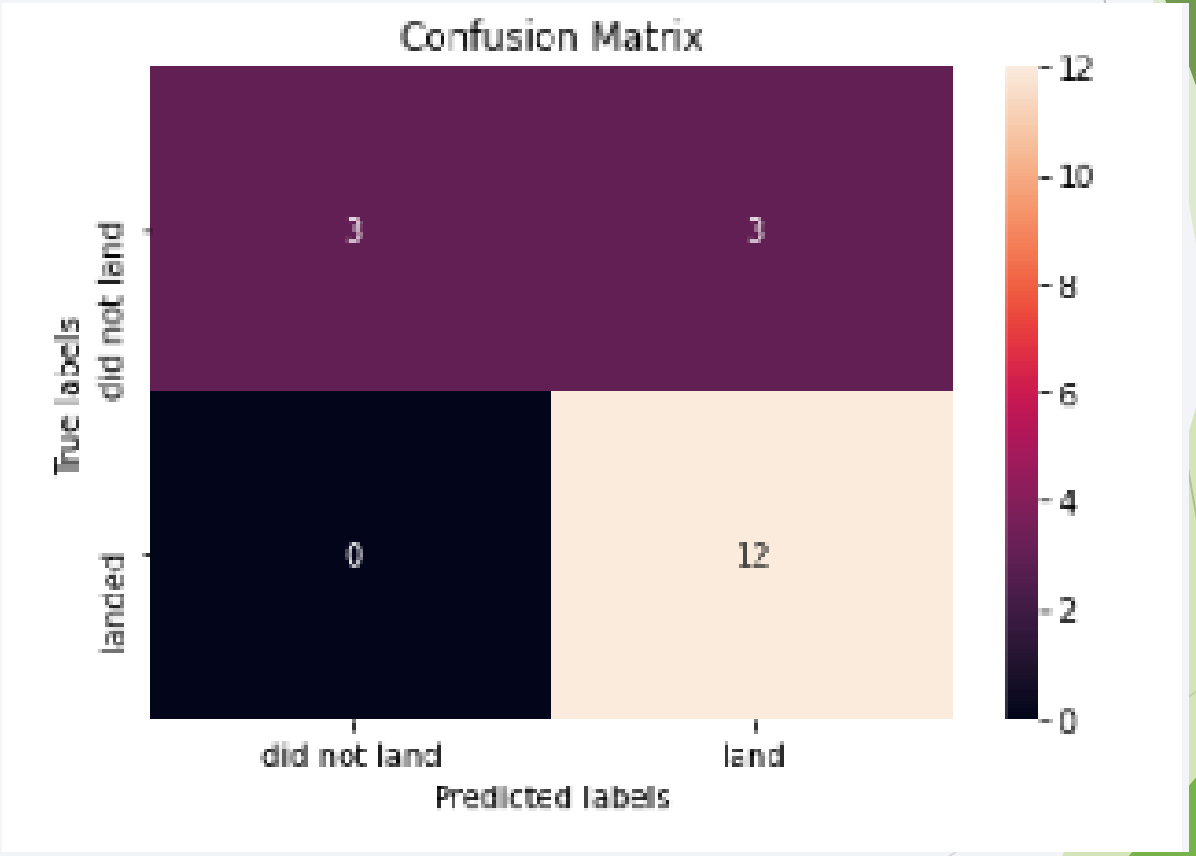
```
models = {'KNeighbors':knn_cv.best_score_,
          'DecisionTree':tree_cv.best_score_,
          'LogisticRegression':logreg_cv.best_score_,
          'SupportVector': svm_cv.best_score_}

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 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes.
- The major problem with the model is the false positives.



Conclusions

In conclusion, using the tools provided on this data:

- The larger the flight amount at the launch site, the greater the success rate at a launch site.
- Launch Success rate started to increase 2013 up until 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most successful launches.
- KSC LC-39A had the most successful launches of any sites.
- The decision tree classifier is the best machine learning algorithm for this task

Thank you!