IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Kamal Kharazi bin Khairuzin
<Date>

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
- Exploratory Data Analysis with Data Visualization
- Interactive Visual Analytics with Folium
- Machine Learning Prediction

Summary of all results

- Exploratory Data Analysis result
- Interactive analytics in screenshots
- Predictive Analytics result

Introduction

Background

- 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 needing 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 needs 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

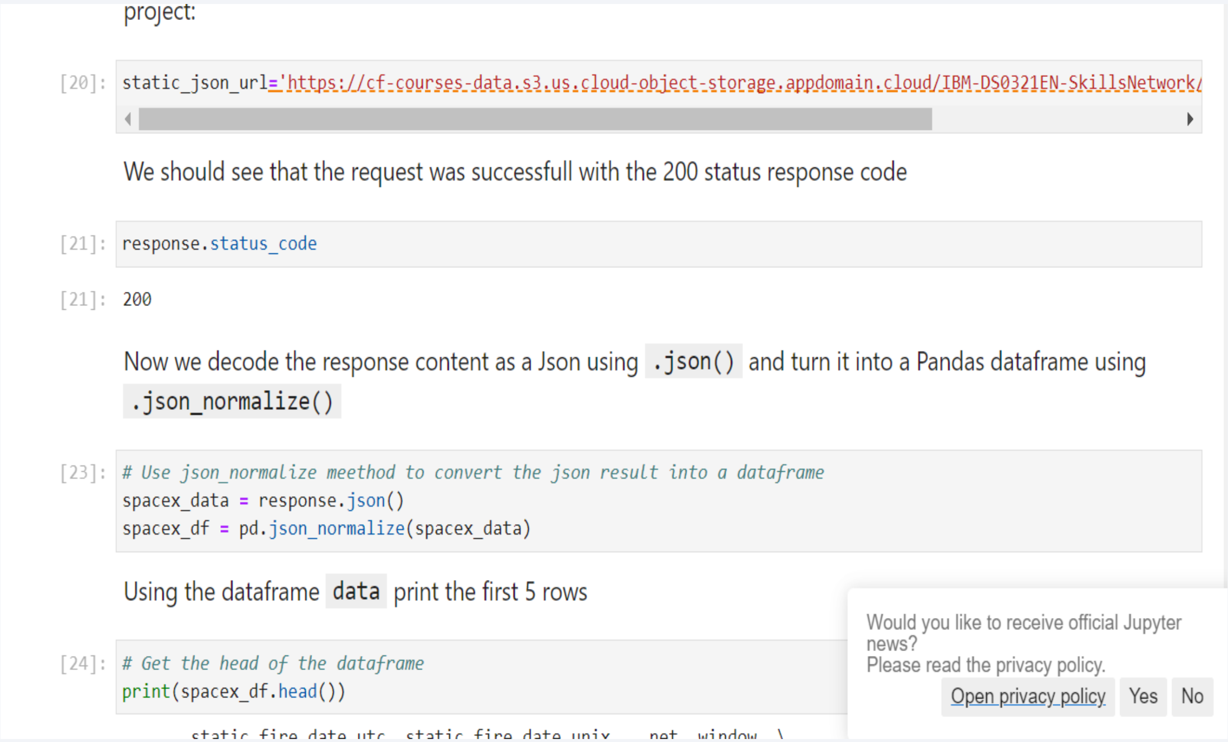
Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, 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.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Github Link: https://github.com/kmlrazi/dscapstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts
- Github Link: https://github.com/kmlrazi/dscapstone/blob/main/jupyter-labs-webscraping.ipynb

[4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027
```

Next, request the HTML page from the above URL and get a `response` object

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

[5]:
```
response = requests.get(static_url)
```

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

[6]:
```
soup = BeautifulSoup(response.text, 'html.parser')
```

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

[7]:
```
soup.title
```

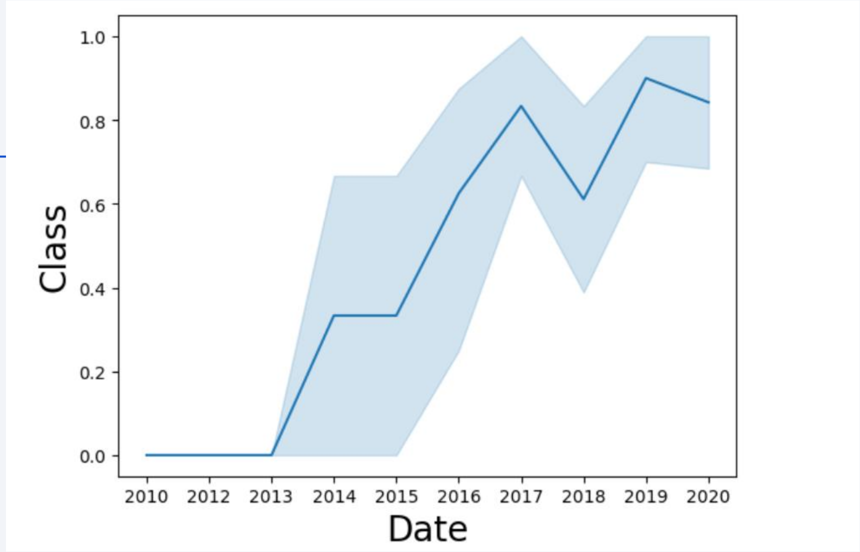
Would you like to receive official Jupyter news?
Please read the privacy policy.

Data Wrangling

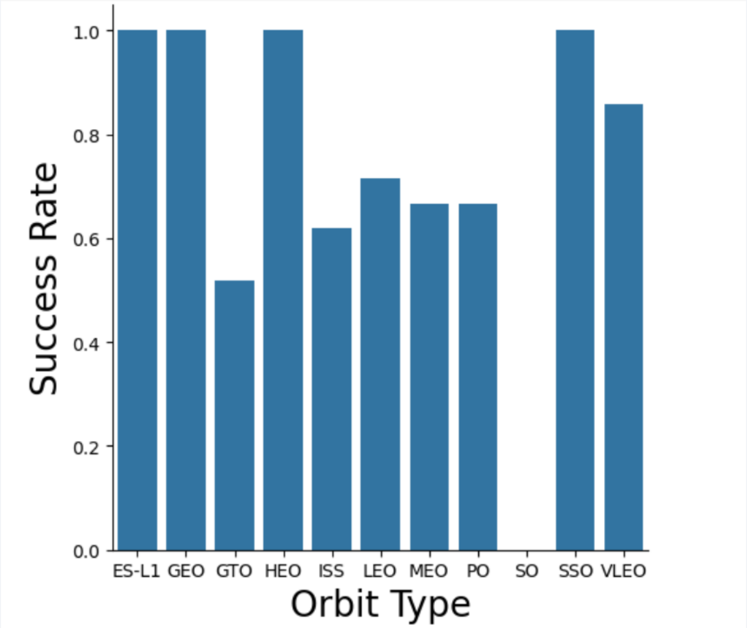
1. We performed exploratory Data Analysis and determined Training Labels
2. We then calculated the number of launches at each site, as well as the number and occurrence of each orbits
3. We then created landing outcome label from the outcome column

- GitHub Link: https://github.com/kmlrazi/dscapstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization



- Line chart was used for graphs explaining things over time for example the yearly success trend
- Meanwhile bar graph was used to explain the relationship between success rate and orbit type
- Github Link: https://github.com/kmlrazi/dscapstone/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb



EDA with SQL

- First load the SQL extension and establish a connection with the database
- We then removed blank rows from the table by using an sql statement
- We then used sql statements complete the tasks
- Github Link: https://github.com/kmlrazi/dscapstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We marked all the launch sites and made map elements such as markers, circles, and lines to indicate the outcomes (success or failure) of launches at each site on the folium map.
- The launch outcomes have been categorized into classes: 0 for failure and 1 for success.
- By utilizing color-coded marker clusters, we have discerned which launch sites exhibit a comparatively high success rate.
- Github Link: https://github.com/kmlrazi/dscapstone/blob/main/folium.ipynb

Build a Dashboard with Plotly Dash

- Built an interactive dashboard using Plotly dash
- Pie charts shows the total launches by a certain sites
- Scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- GitHub Link: https://github.com/kmlrazi/dscapstone/blob/main/visual_analytics_with_plotly.py

Predictive Analysis (Classification)

1. We loaded the data using numpy and pandas
2. We tuned different hyperparameters using GridSearchCV.
3. We used jaccard score, f1 score and accuracy as the metric to find the best performing classification algorithm

- GitHub Link: https://github.com/kmlrazi/dscapstone/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

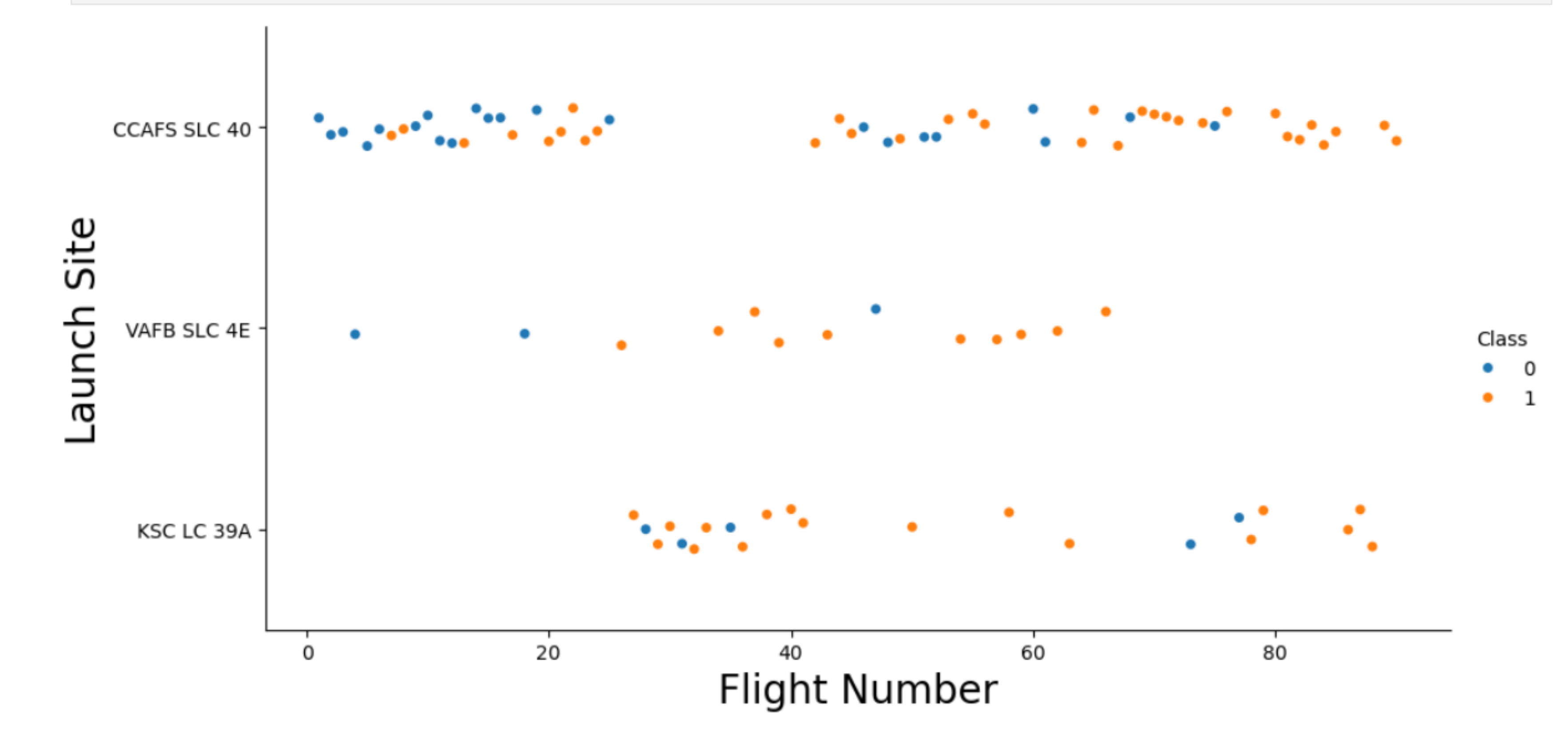
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

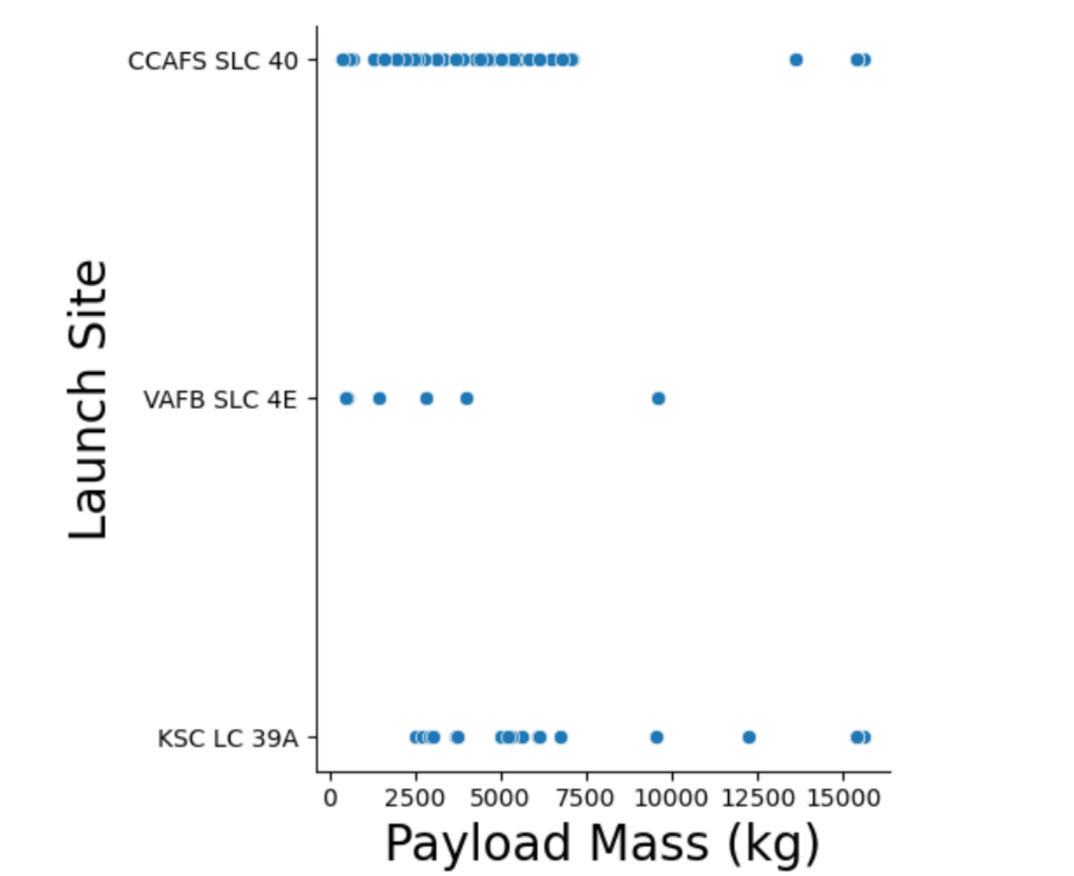
Flight Number vs. Launch Site

- We found that the larger the flight amount at a launch site, the greater the success rate at a launch site.



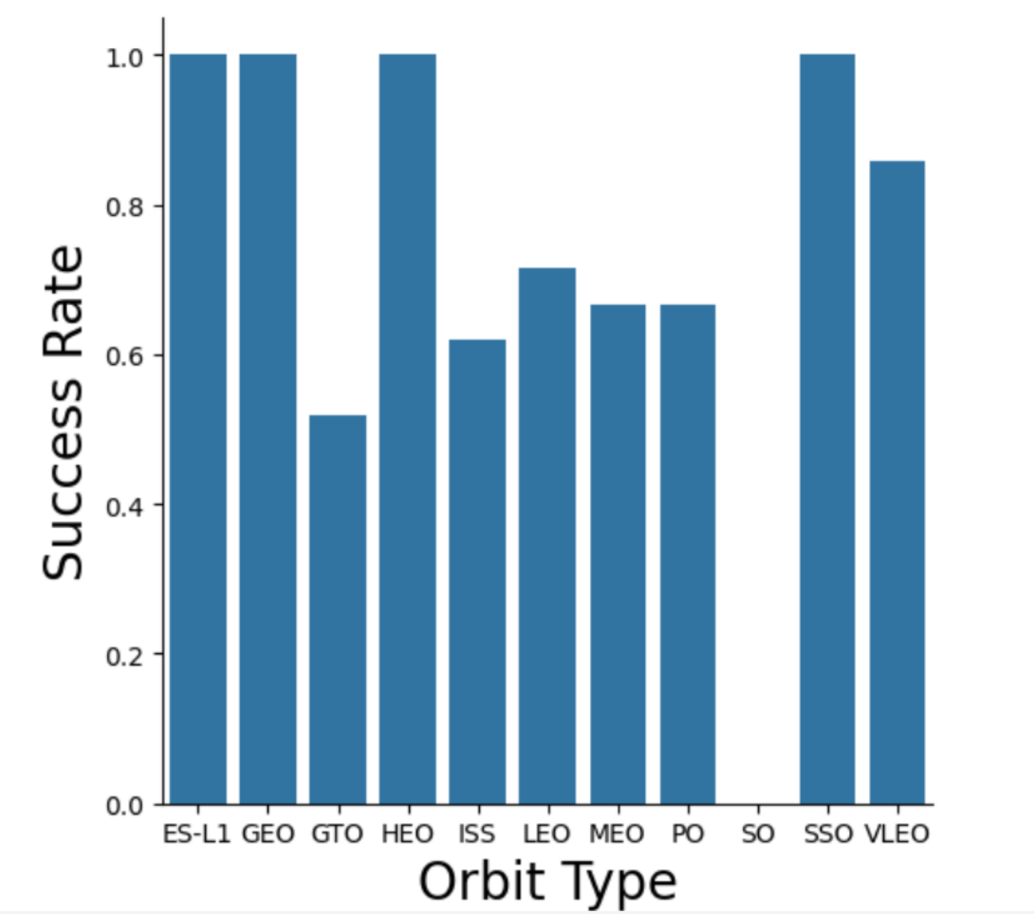
Payload vs. Launch Site

- We found that the greater the payload mass, the higher the success rate

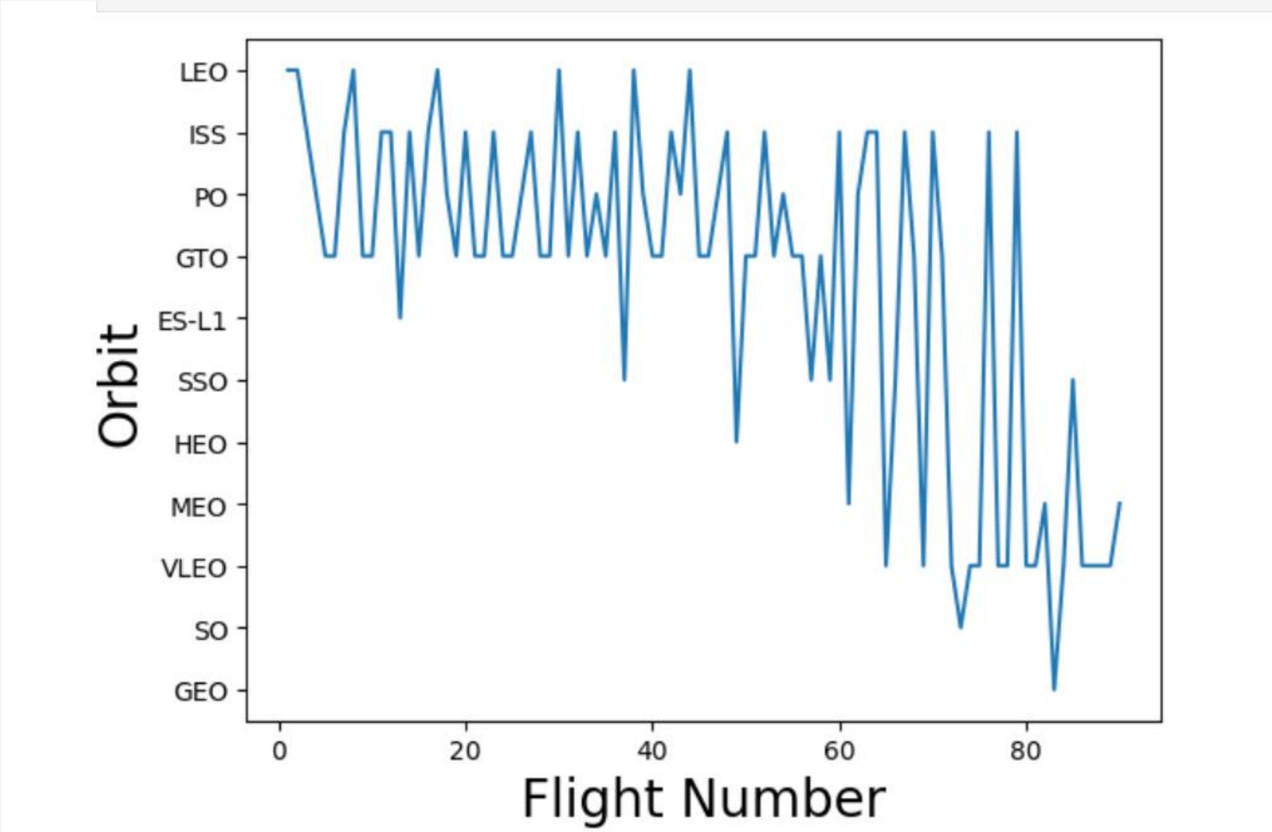


Success Rate vs. Orbit Type

- The bar chart shows the success rate of each orbit type with the highest being WS-L1, GEO, HEO and SSO

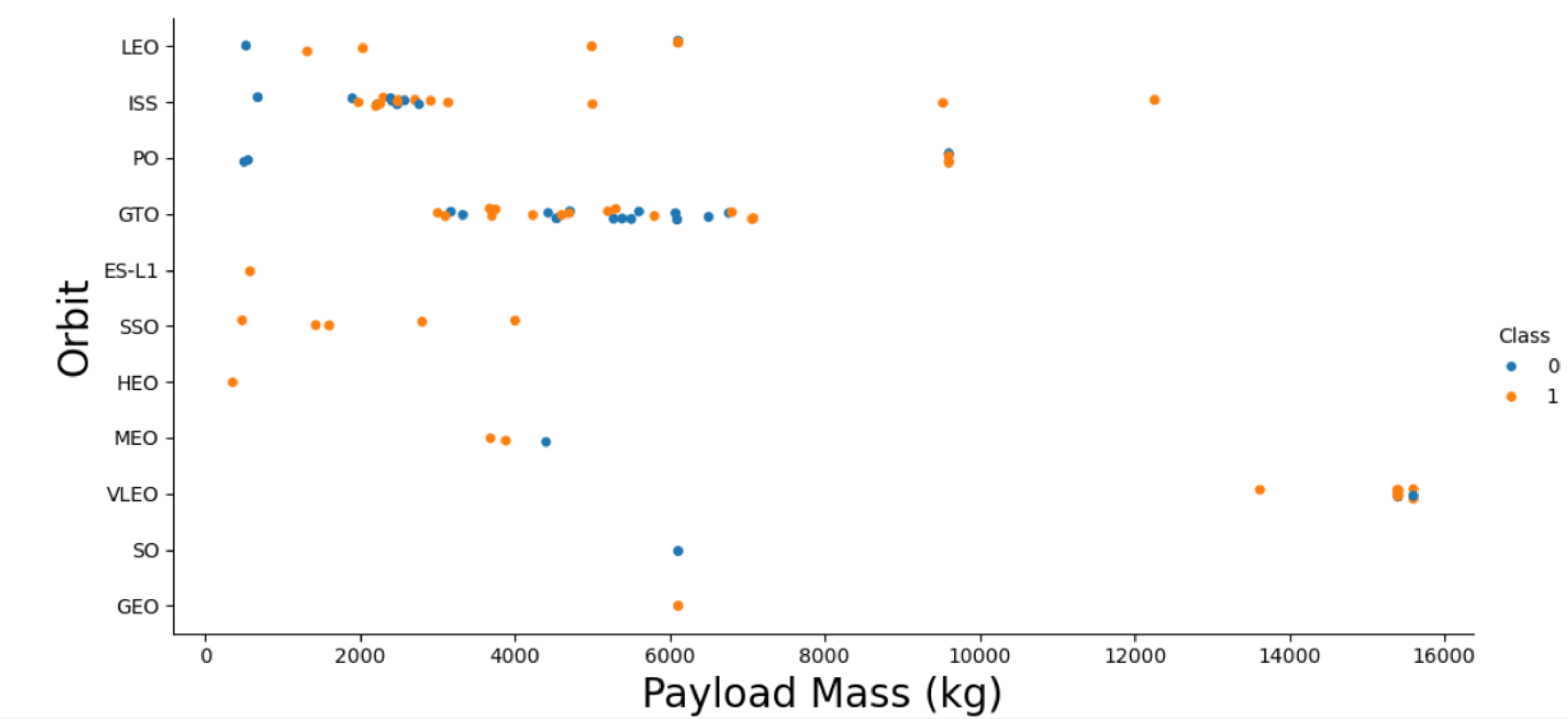


Flight Number vs. Orbit Type



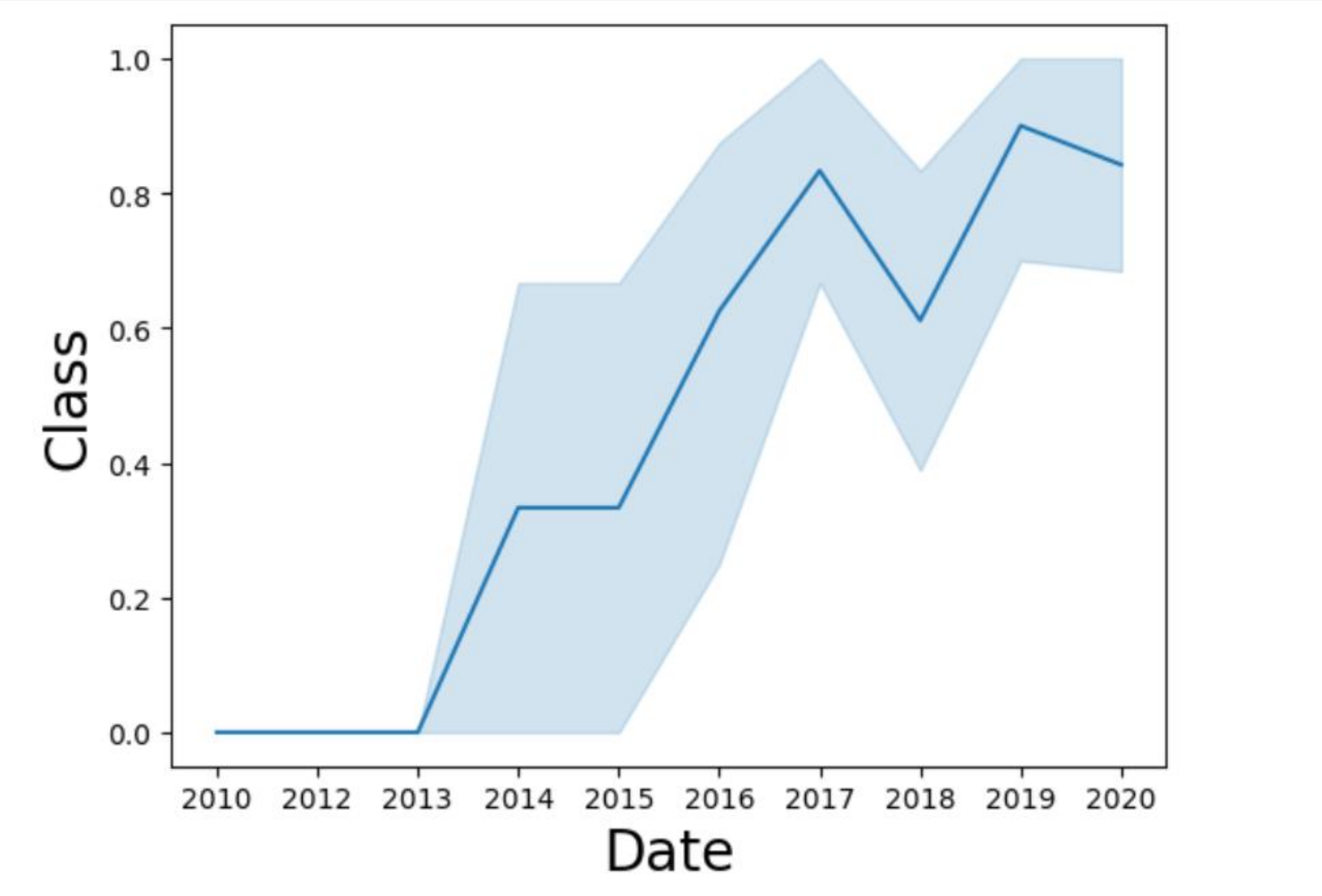
Payload vs. Orbit Type

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



Launch Success Yearly Trend

- From the graph, we can see that recently the yearly launch success rate has increased



All Launch Site Names

- We used the sql SELECT DISTINCT statement to get all the launch site names

```
In [21]: %sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL

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

Out[21]:

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

Launch Site Names Begin with 'CCA'

- We used the LIKE sql statement to get the site names starting with ‘CCA’ and then used the LIMIT statement to only get 5 instances

In [23]:
```
%sql SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE '%CCA%' LIMIT 5
```

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

Out[23]:

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

- We used the SELECT SUM function to get the total payload mass

In [24]:
```
%sql SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

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

Out[24]:

SUM("PAYLOAD_MASS__KG_")
45596

Average Payload Mass by F9 v1.1

- We used SELECT AVG function to get the average payload mass

Display average payload mass carried by booster version F9 v1.1

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

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

Out[11]:

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- We used the SELECT MIN(date) to get the first successful ground landing date

```
]: %sql select min(date) as first_successful_landing from SPACEXDATASET where landing__outcome = 'Success (ground pad)

 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/
Done.
```

first_successful_landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the WHERE function to get list the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less th

```
]: %sql select booster_version from SPACEXDATASET where landing__outcome = 'Success (drone ship)' and payl
```

```
 * ibm_db_sa://wzf08322:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.
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

- We used the SELECT COUNT function to get the total number of successful and failure mission outcomes

List the total number of successful and failure mission outcomes

```
[29]: %sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSIC
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" L
```

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

[29]:

SUCCESS	FAILURE
100	1

Boosters Carried Maximum Payload

- We used the SELECT DISTINCT function to only return on booster version and then WHERE function to get the boosters with the maximu payload

```
[31]: %sql SELECT DISTINCT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT max("PAYLOAD_M

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

[31]:

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

Would you like to receive official Jup
news?
Please read the privacy policy

2015 Launch Records

- We used the WHERE function to get the failed launch records

```
In [38]: %sql SELECT substr(Date,4,2) as month, DATE,BOOSTER_VERSION, LAUNCH_SITE, [Landing _Outcome] \
FROM SPACEXTBL \
where [Landing _Outcome] = 'Failure (drone ship)' and substr(Date,7,4)='2015';

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

Out[38]:

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

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 2010-03-20

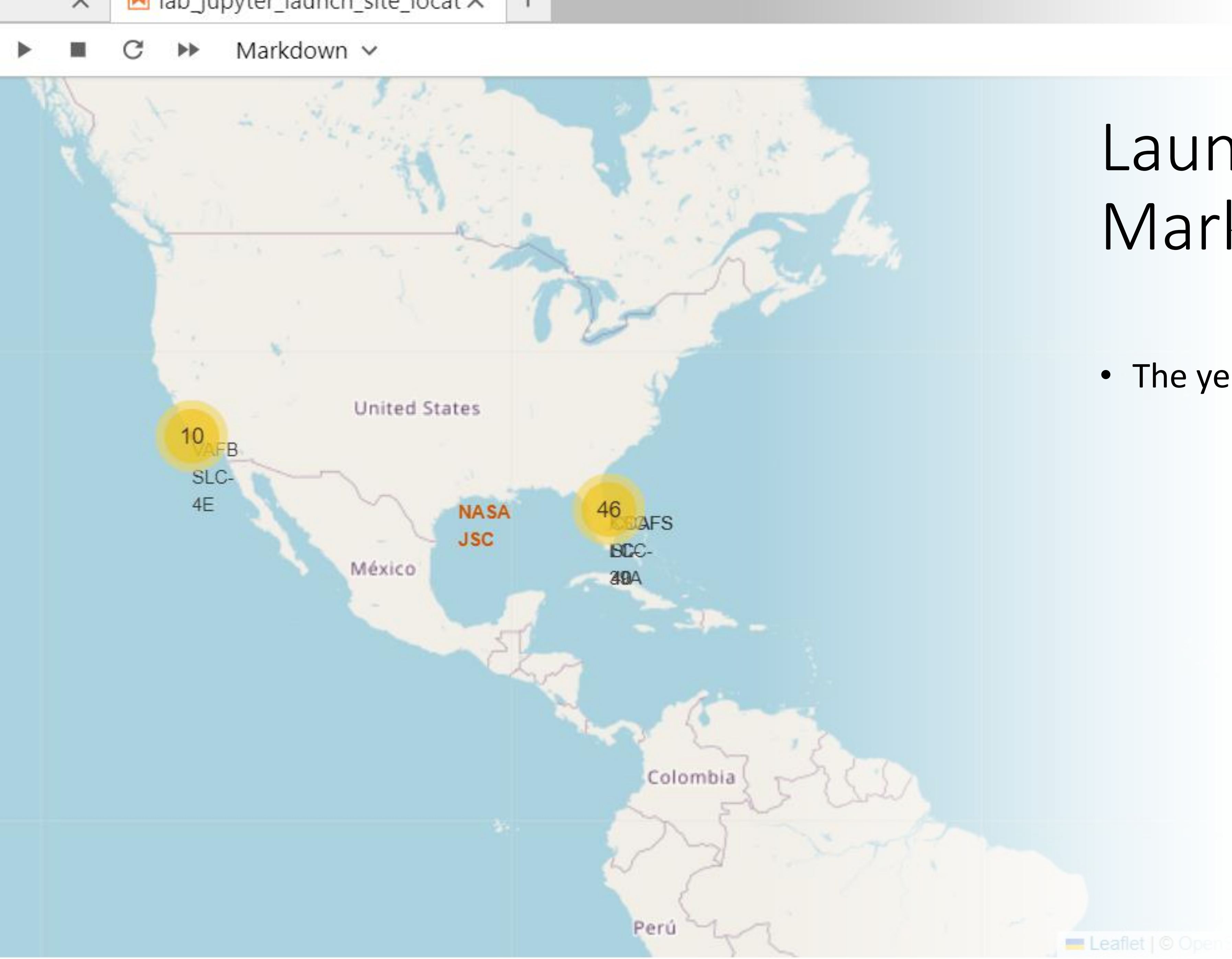
```
In [37]: %sql SELECT [Landing _Outcome], count(*) as count_outcomes \
         FROM SPACEXTBL \
         WHERE DATE between '04-06-2010' and '20-03-2017' group by [Landing _Outcome] order

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

Out[37]:

Landing _Outcome	count_outcomes
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

Section 3

Launch Sites Proximities Analysis

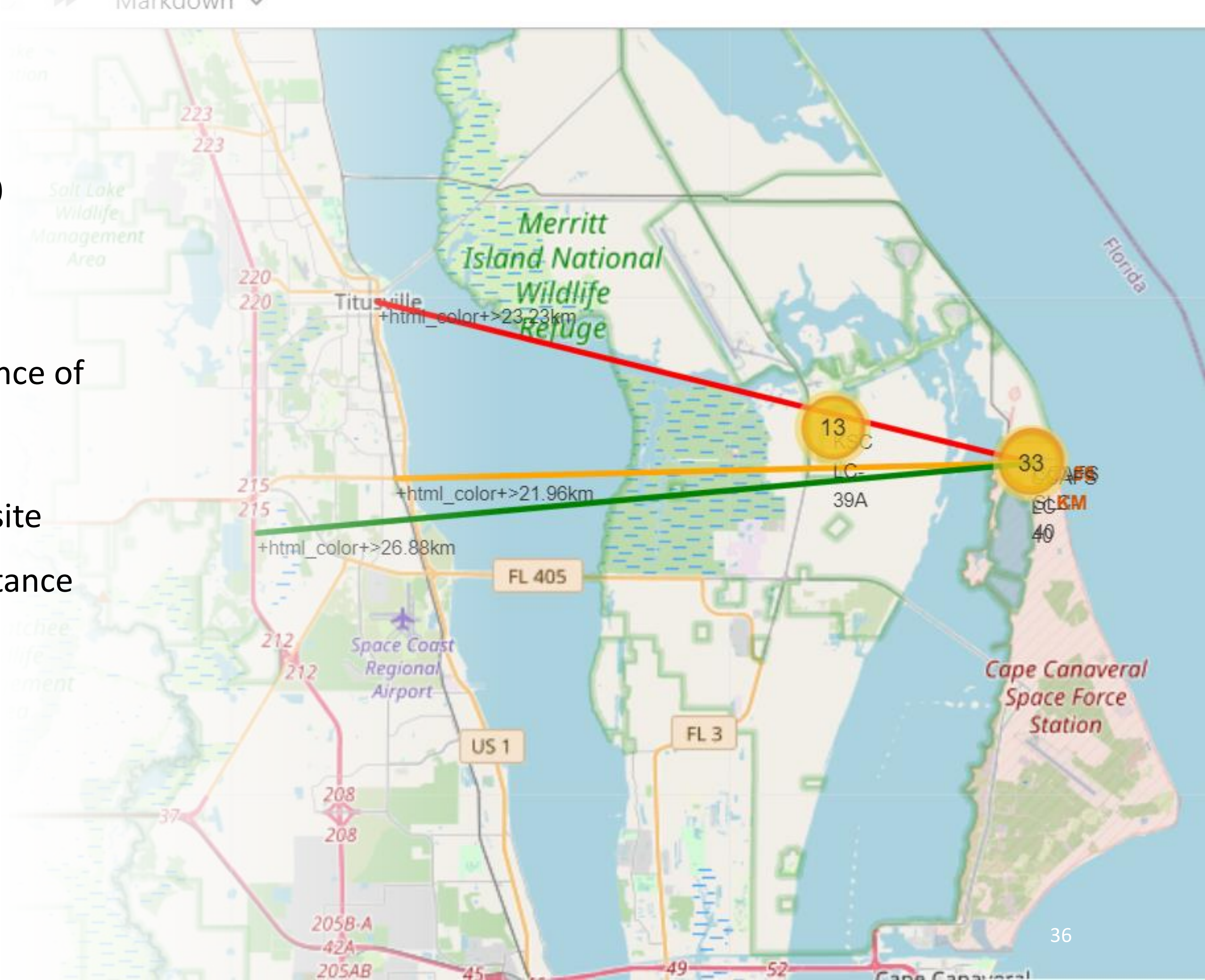


Launch Site Map Markers

- The yellow are the launch sites

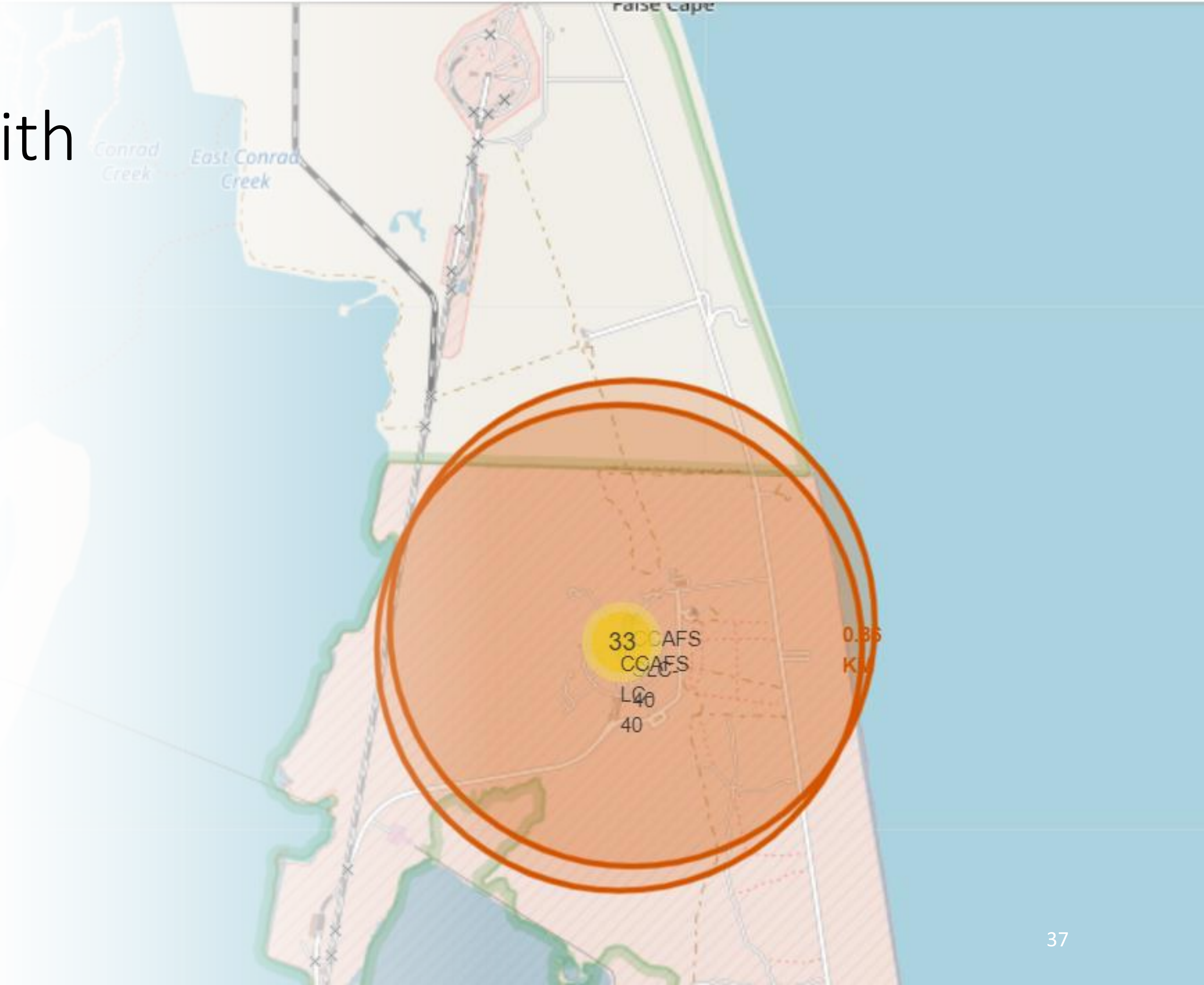
Distance of Launch Site to Landmarks

- The map shows the distance of the launch site to certain landmarks.
- The yellow is the launch site
- The line indicates the distance between launch site and landmarks



Launch sites with labels

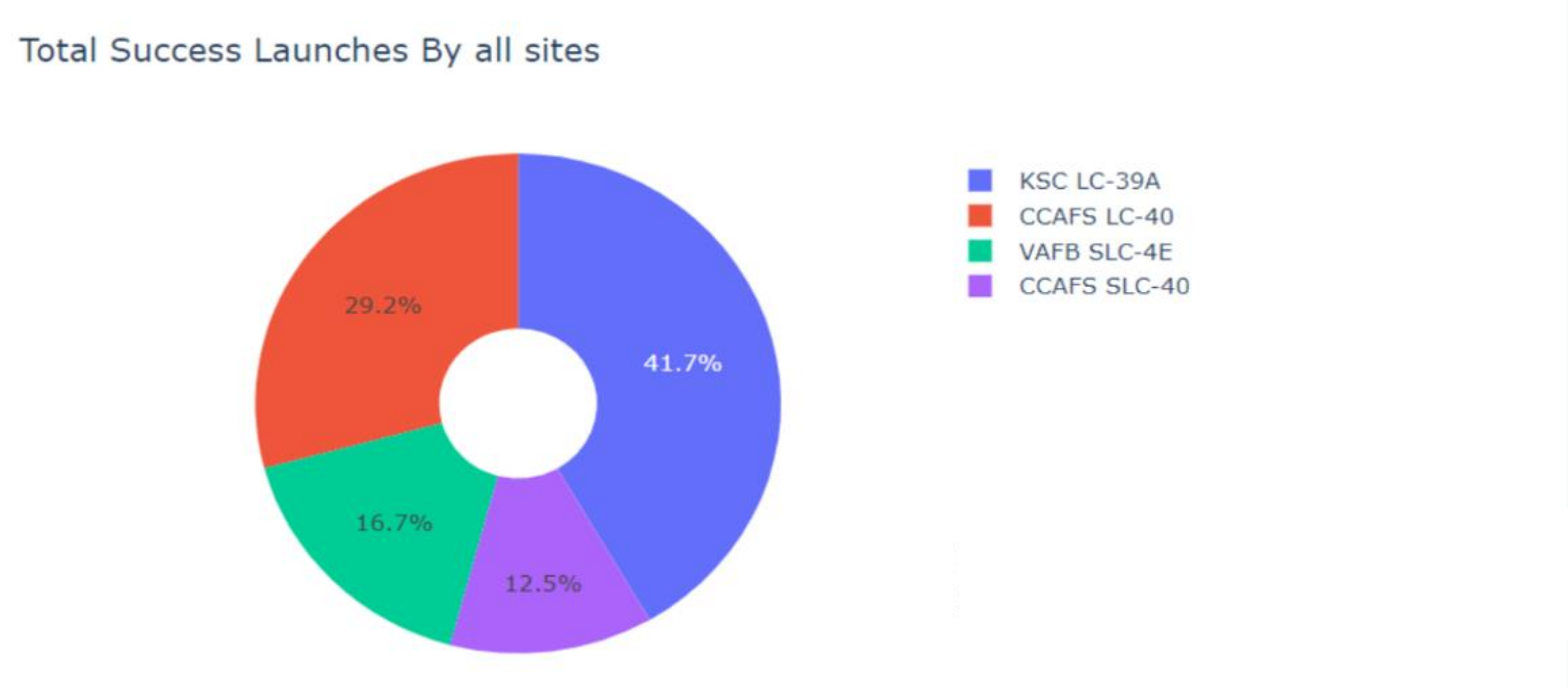
- The folium map shows the labelled launch sites



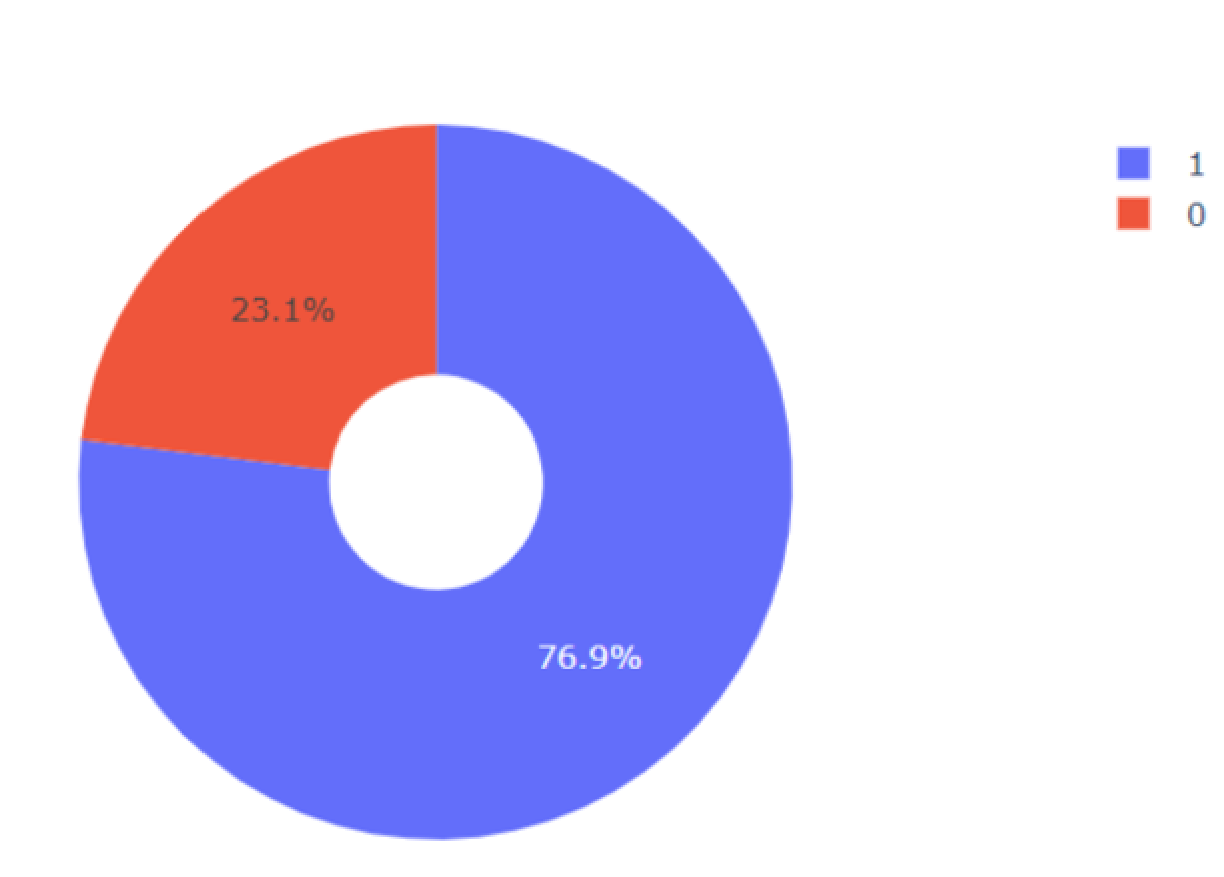
Section 4

Build a Dashboard with Plotly Dash

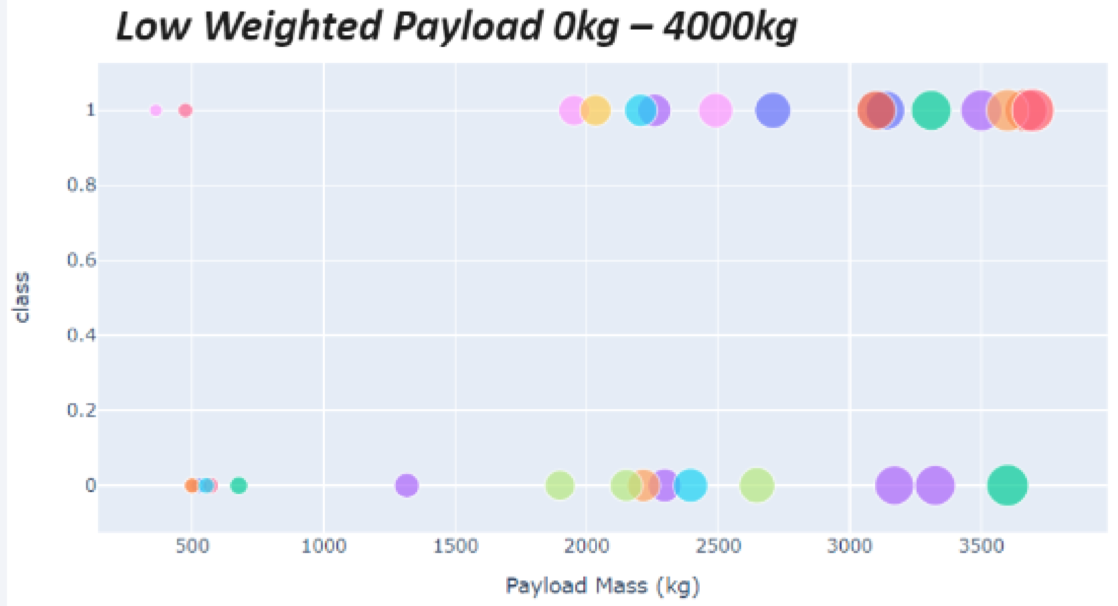
Pie chart for the success percentage achieved by each launch site

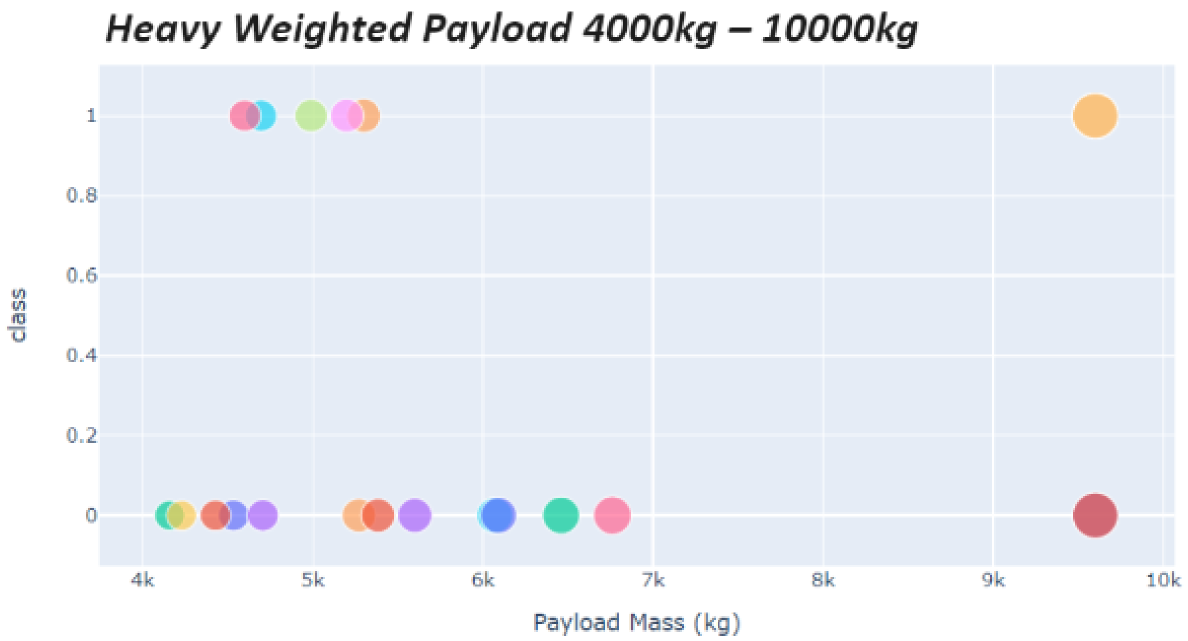


Pie chart for the Launch site with the highest launch success ratio



Payload vs Launch Outcome for all sites, with different payload selected in the range slider





Section 5

Predictive Analysis (Classification)

Classification Accuracy

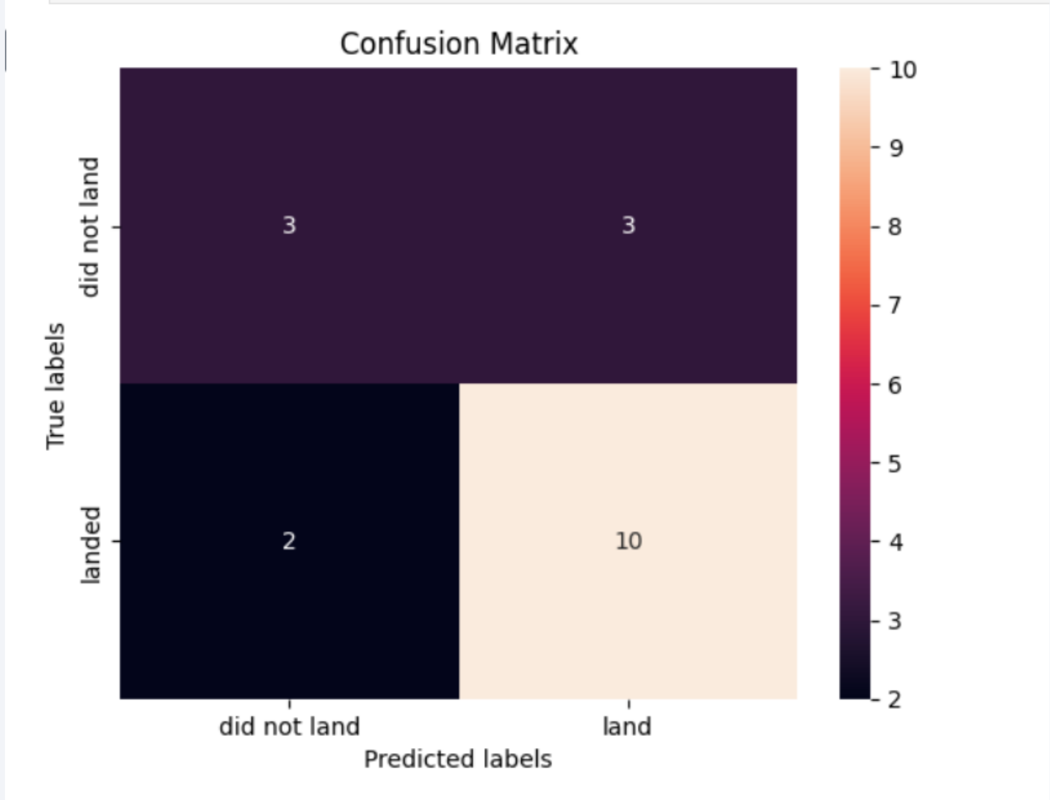
- The best model is the decision tree because it has the best accuracy as well as the best scores overall

Out[31]:

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.833333	0.845070	0.882353	0.819444
F1_Score	0.909091	0.916031	0.937500	0.900763
Accuracy	0.866667	0.877778	0.911111	0.855556

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



Conclusions

- The assessment of Test Set scores does not definitively establish the best performance of any particular method.
- The similarity in scores within the Test Set may stem from its limited sample size (18 samples).
- Consequently, we extended our evaluation to encompass the entire Dataset.
- The overall Dataset scores validate the Decision Tree Model as the top-performing model, showcasing not only superior scores but also the highest accuracy among the tested methods.

Thank you!