IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Molebatsi Potso
November 2023

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

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

The age of space travel is finally here. Companies such as Virgin Galactic, Rocket lab, Blue Origin, and SpaceX have popularized the commercial of space travel. The cost of space travel is upwards of $165 millions; however, SpaceX advertises that their Falcon 9 rocket only cost $65 millions. SpaceX can reuse the first stage hence such a huge cost saving.

This project aims to create predict if the first stage of Falcon 9 rocket will successfully. The project will try to 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:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - 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
  - 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

- The data was collected using GET Request on SPACEX API. We clean it and wrangling it.

- Here is the notebook link:

https://github.com/molebatsipotso/IBM-SPACEX-DATA-SCIENCE-CAPSTONE/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

```
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 meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Data Collection - Scraping

- Using a BeautifulSoup, we did web scraping to scrap records from Falcon 9. Then we converted the data to be in panda dataframe format.
- Here is the link:

https://github.com/molebatsipotso/IBM-SPACEX-DATA-SCIENCE-CAPSTONE/blob/main/jupyter-labs-webscraping.ipynb

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

```
200
```

Create a `BeautifulSoup` object from the HTML `response` soup = BeautifulSoup(response)

```python
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text)
```

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

```python
# Use soup.title attribute
print(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

```python
# 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.

```python
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

Data Wrangling

- For our wrangling we performed exploratory data analysis and determined the training labels. Calculated the number of launches at each site, and how often they occur for each orbits. At the end we created a column to capture the landing outcome.
- Here is the link:
- https://github.com/molebatsipotso/IBM-SPACEX-DATA-SCIENCE-CAPSTONE/blob/main/labs-jupyter-spacex-Data%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
```

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for i in df['Outcome']:
    if i in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

This variable will represent the classification variable that represents the
did not land successfully; one means the first stage landed Successfully

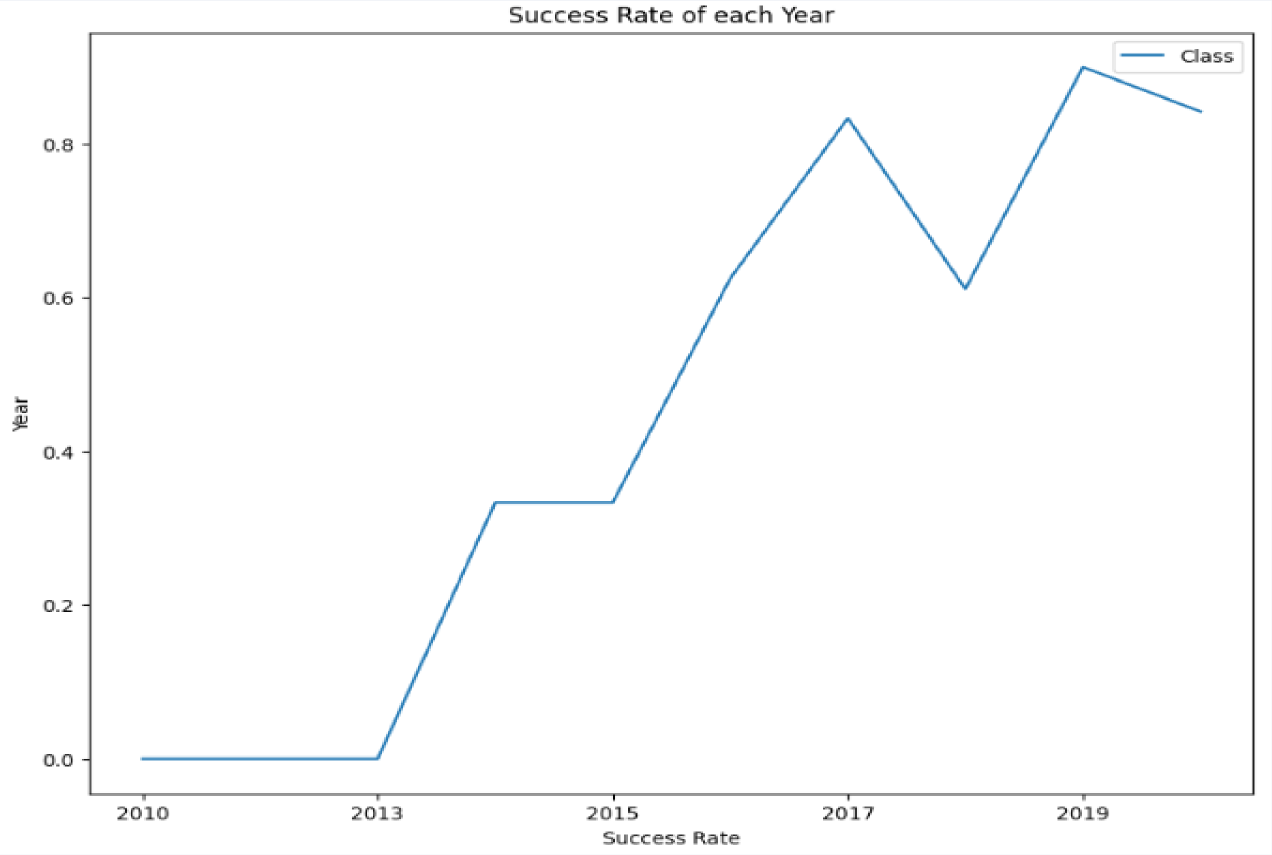
```
df['Class']=landing_class
df[['Class']].head(8)
```

EDA with Data Visualization

- We visualize 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.

- Here is the link:

https://github.com/molebatsipotso/IBM-SPACEX-DATA-SCIENCE-CAPSTONE/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb



EDA with SQL

- We use SQL queries to obtain insight on the data. Queries such as:
  - %sql SELECT * FROM SPACEXTBL limit 20;
  - %sql SELECT DISTINCT(Launch_Site) FROM SPACEXTBL;
  - %sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE "%CCA%" LIMIT 5
  - %sql SELECT SUM(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;
  - %sql SELECT Landing_Outcome FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY Date DESC;
- Here is the link:

https://github.com/molebatsipotso/IBM-SPACEX-DATA-SCIENCE-CAPSTONE/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- To mark the success or failure on each launch site, we marked launch sites on folium map with circles and lines. The successful launch to classed to 1 and 0 was for failed launches.
- We calculated the distances between a launch site to its proximities such as: railways, highways and coastlines
- Here is the link:

https://github.com/molebatsipotso/IBM-SPACEX-DATA-SCIENCE-CAPSTONE/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- Using Plotly, we built an interactive dashboard that shows insight such as:
  - Total launches for each sites
  - Relationship between outcome and payload mass for each booster version.

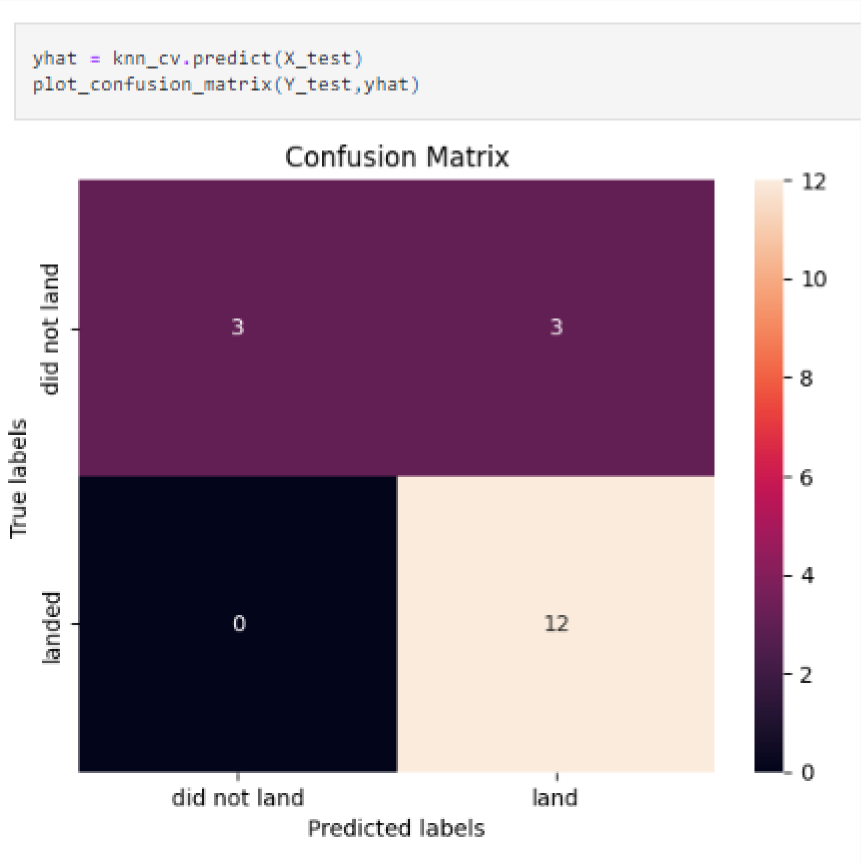
- Here is the link:

https://github.com/molebatsipotso/IBM-SPACEX-DATA-SCIENCE-CAPSTONE/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- To preform the predictive analysis on the data, we split the data into training and testing data. Built different machine learning models using GridSearchCV and improve the accuracy of each models
- Here is the link:

https://github.com/molebatsipotso/IBM-SPACEX-DATA-SCIENCE-CAPSTONE/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb



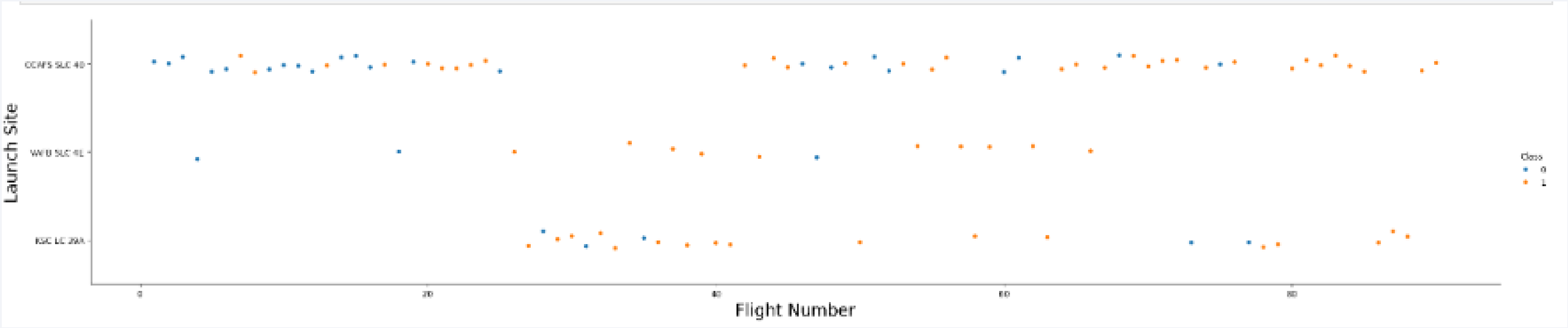
Results

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

Section 2

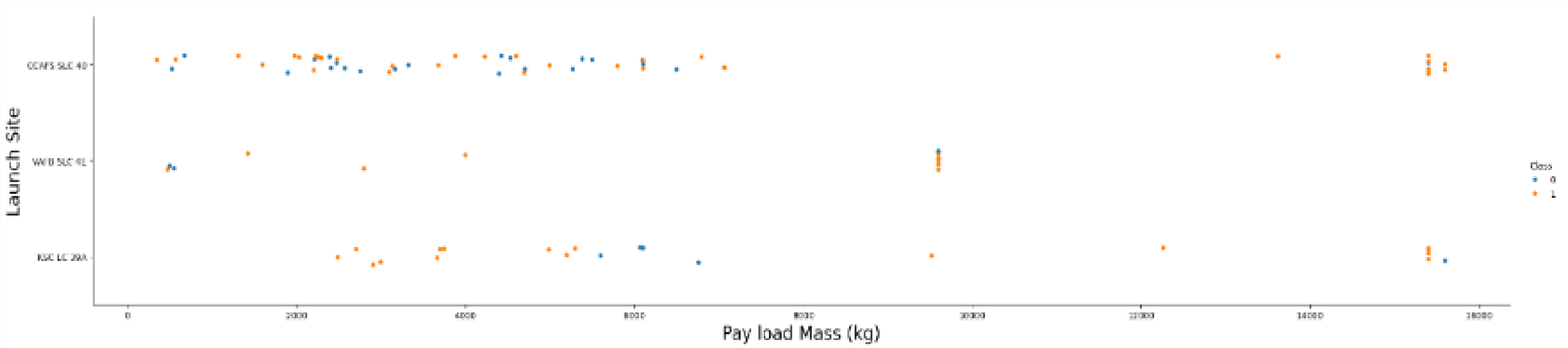
Insights drawn from EDA

Flight Number vs. Launch Site



- We can see that the larger the flight amount the greater the success rate at each launch site.

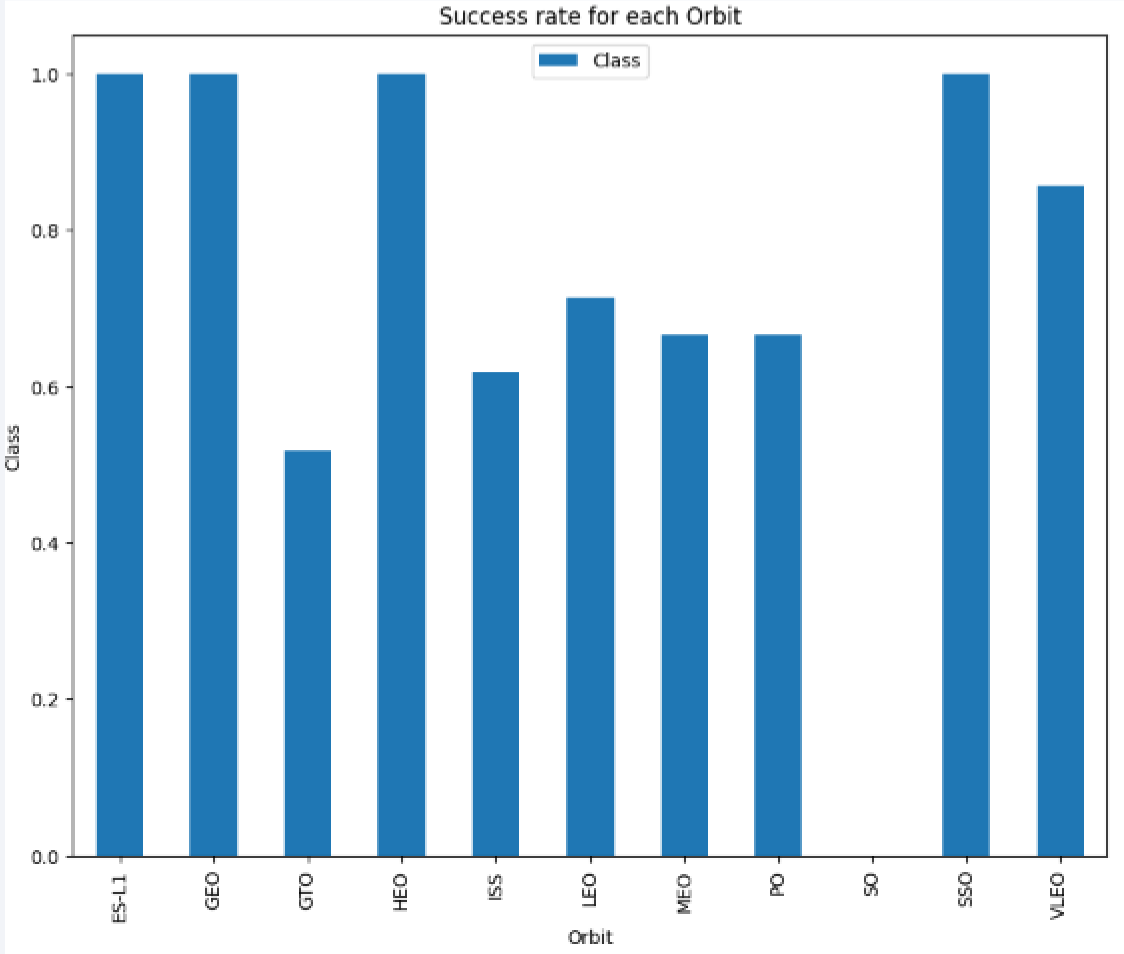
Payload vs. Launch Site



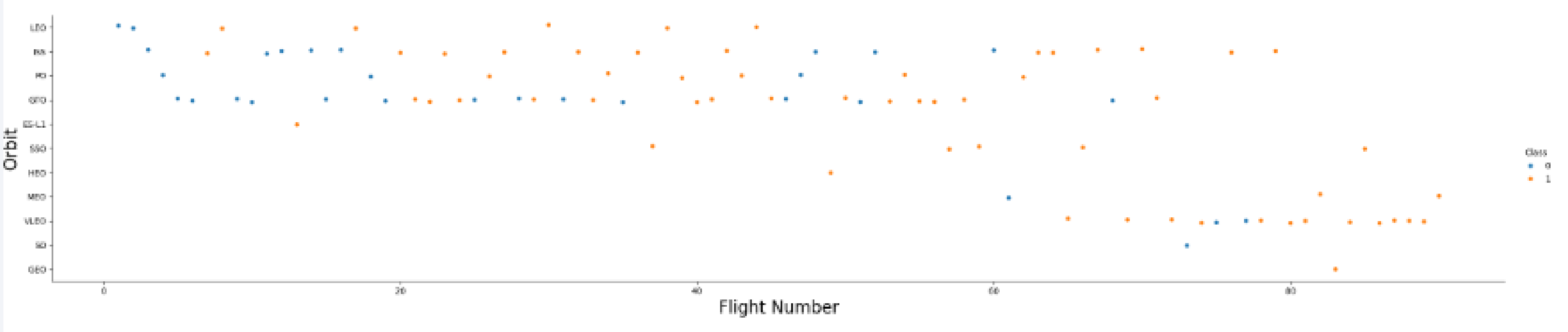
- For the launch site CCAFS SLC40 we can see that the greater the payload mass, the higher the success rate.

Success Rate vs. Orbit Type

- The most success rate observed are from the orbit: ES-L1, GEO, HEO, SSO, and VLEO.

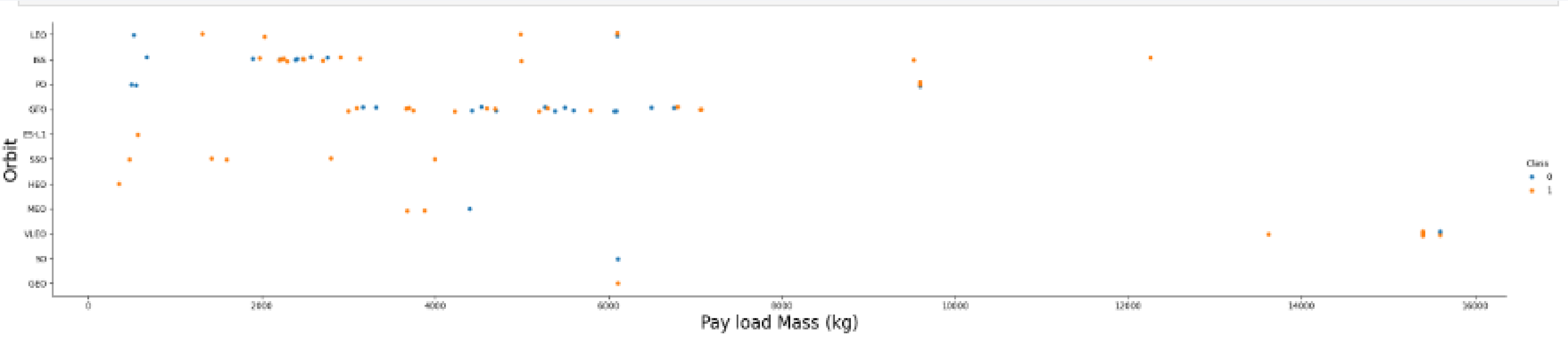


Flight Number vs. Orbit Type



- We can see here that the launch success depends on the number of flights for LEO orbit.
- However, for GTO orbit there is no clear relationship.

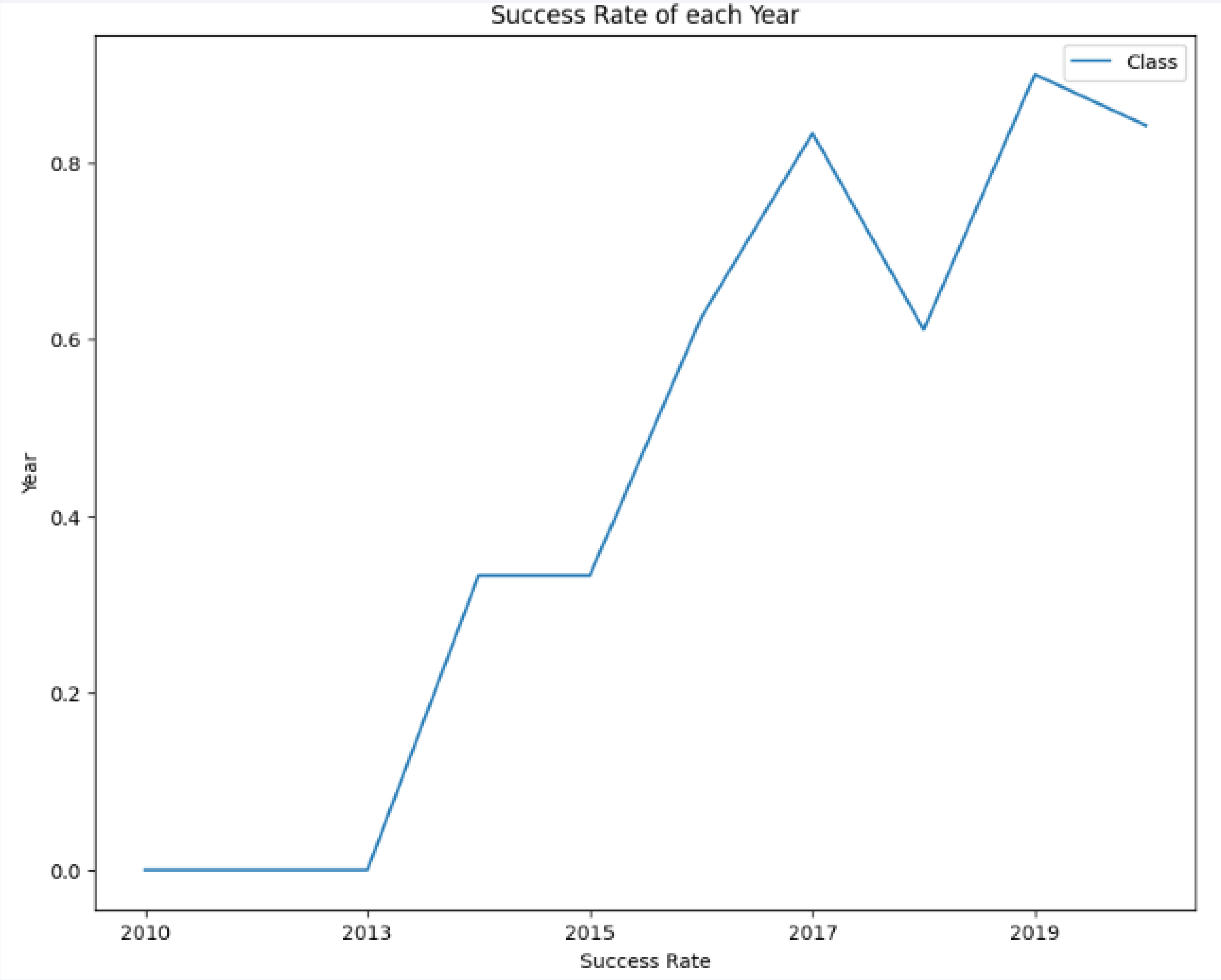
Payload vs. Orbit Type



- The heavier the payloads, the more successful landing for PO, LEO and ISS orbits.

Launch Success Yearly Trend

- The success rate has been increasing since 2013.



All Launch Site Names

- We used distinct to show unique launch sites.

```
%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'

- Using the query “LIKE” we display 5 records where launch sites begin with `CCA`

```
%sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' limit 5;
```

✓ 0.0s Python

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

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

- The total payload carried by boosters from NASA is calculates as 619967

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;
```

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

payloadmass
619967

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 was calculated as 6138.28

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as payloadmass FROM SPACEXTBL;
```

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

payloadmass
6138.287128712871

First Successful Ground Landing Date

- The first successful landing outcome on ground pad was on 6th April 2010.

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
%sql SELECT MIN(DATE) from SPACEXTBL;
```

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

MIN(DATE)
2010-04-06

Successful Drone Ship Landing with Payload between 4000 and 6000

- There were four successful drone ship landing with payload mass between 4000 and 6000.

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_ BETWEEN
```

```
 * 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

- We calculated the total number of success and failure of mission outcomes

List the total number of successful and failure mission outcomes

```
%sql SELECT COUNT(Mission_Outcome) as missionoutcomes FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

missionoutcomes
1
98
1
1

Boosters Carried Maximum Payload

- List of the booster carried the maximum payload mass

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
%sql SELECT Booster_Version AS boosterversion FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) F
```

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

boosterversion
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

- List of the failed landing_outcomes in drone ship for the year 2015

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
%sql SELECT SUBSTR(Date,6,2) as Month, Mission_Outcome, Booster_Version, Launch_Site FROM SPACEXTBL WHERE SUBSTR(Date,0
```

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

Month	Mission_Outcome	Booster_Version	Launch_Site
10	Success	F9 v1.1 B1012	CCAFS LC-40
11	Success	F9 v1.1 B1013	CCAFS LC-40
02	Success	F9 v1.1 B1014	CCAFS LC-40
04	Success	F9 v1.1 B1015	CCAFS LC-40
04	Success	F9 v1.1 B1016	CCAFS LC-40
06	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

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

- Ranking of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

```
%sql SELECT Landing_Outcome FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY Date DESC;
```

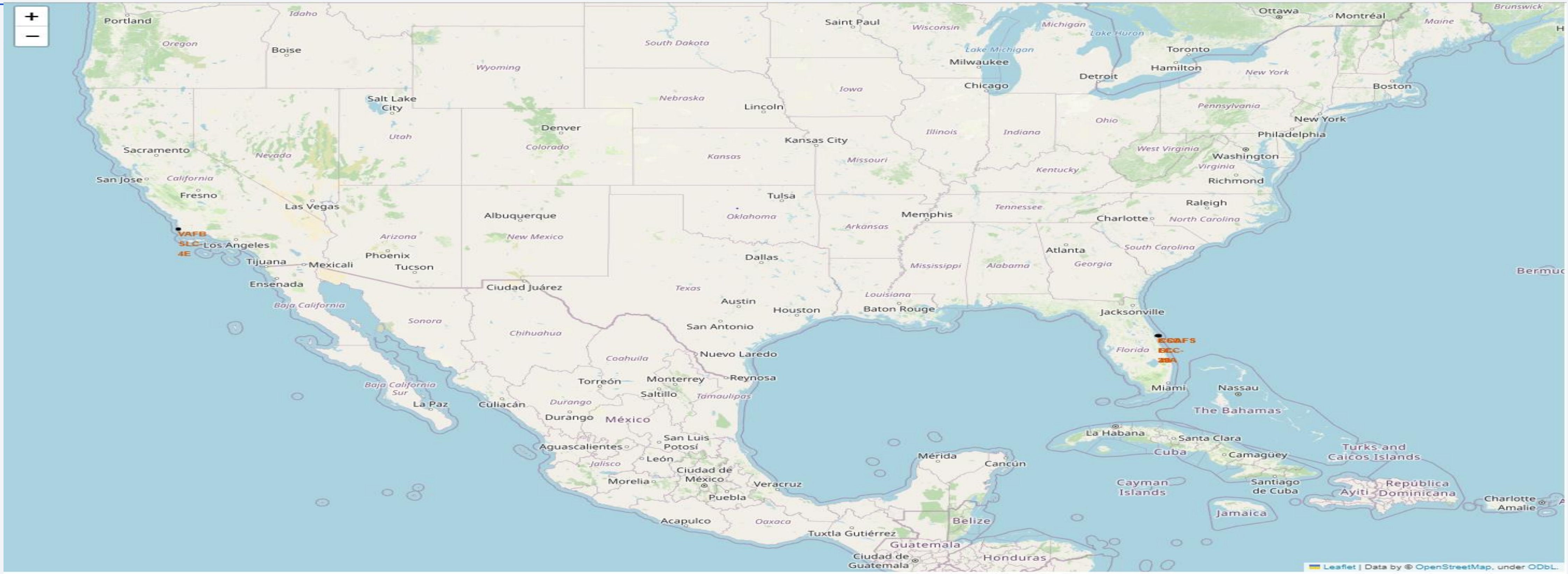
```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome
No attempt
Success (ground pad)
Success (ground pad)
Success (drone ship)
Success (ground pad)
Success (drone ship)
Success (drone ship)
Success (ground pad)
Failure (drone ship)
Success (drone ship)
Success (drone ship)
Failure (drone ship)

Section 3

Launch Sites Proximities Analysis

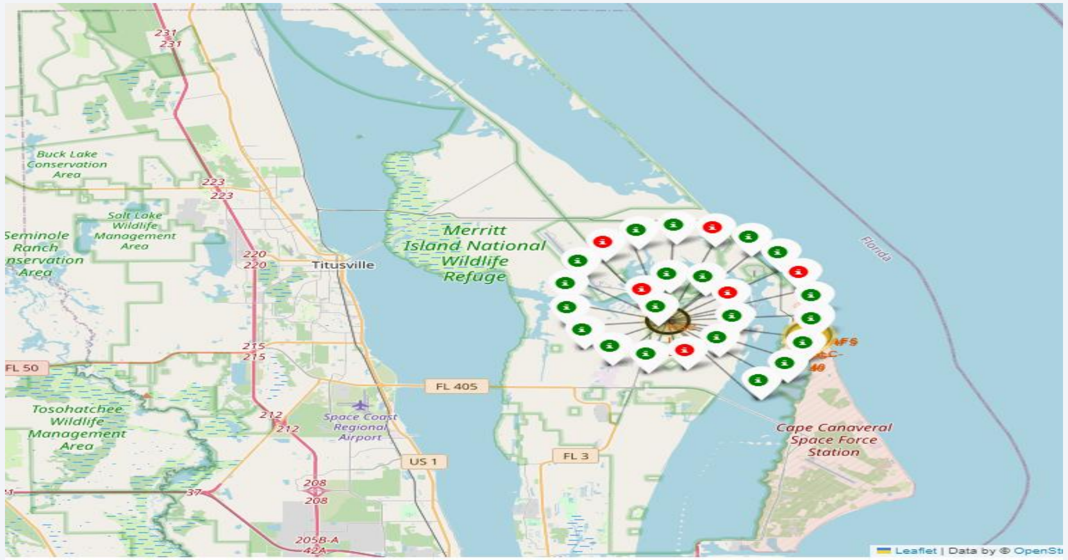
Global map with marked launch sites

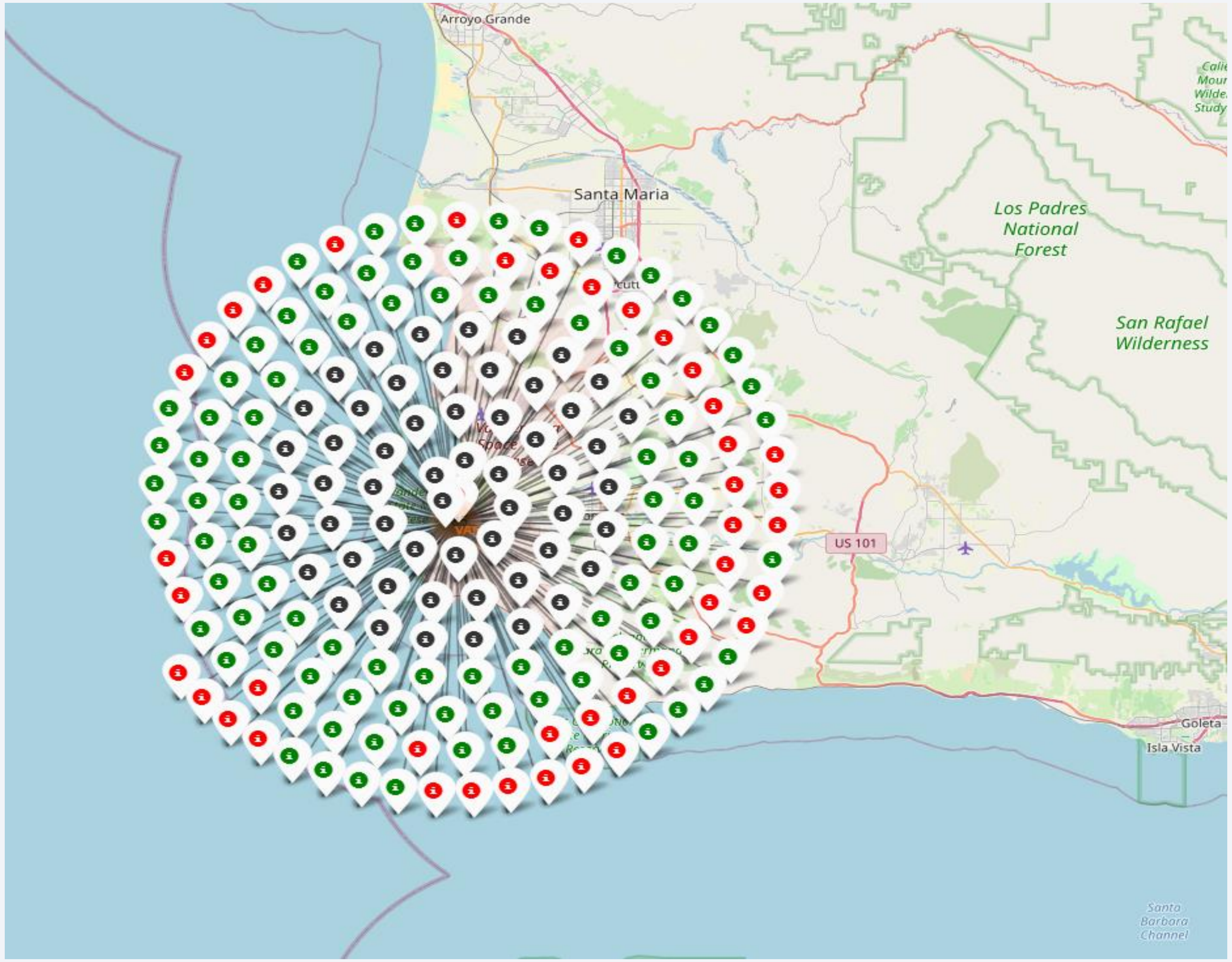


Launch sites are all near the coastlines

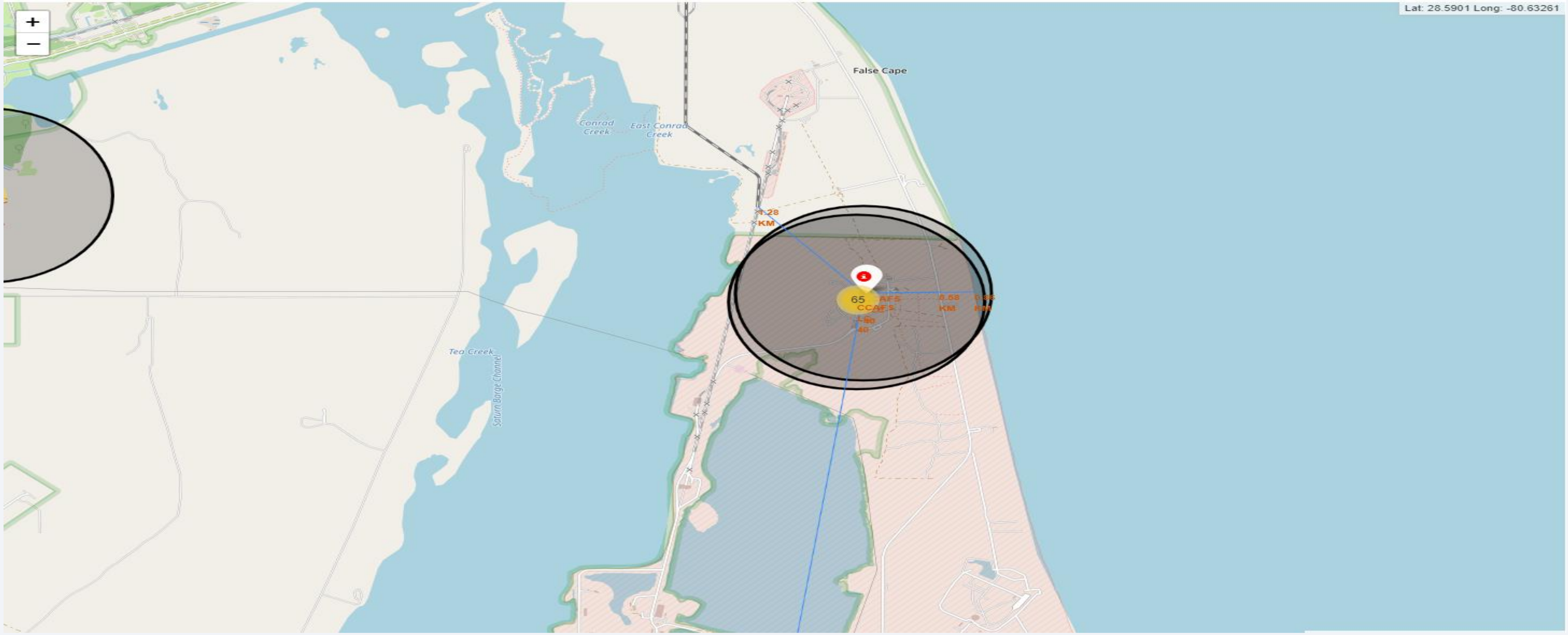
Launch site successful and failed launching

- The green marker show the successful landing;
- The red marker shows the failed landing.





Launch site proximities

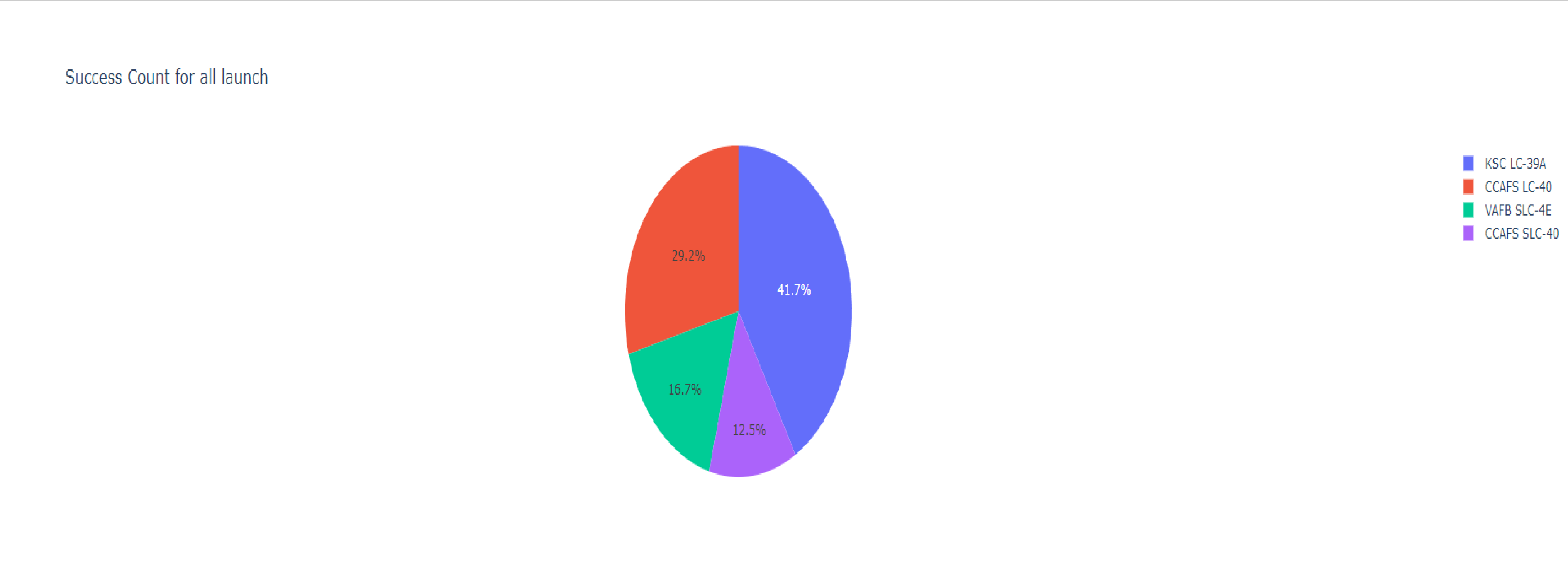


- The blue line shows the distance to the nearest landmark site.

Section 4

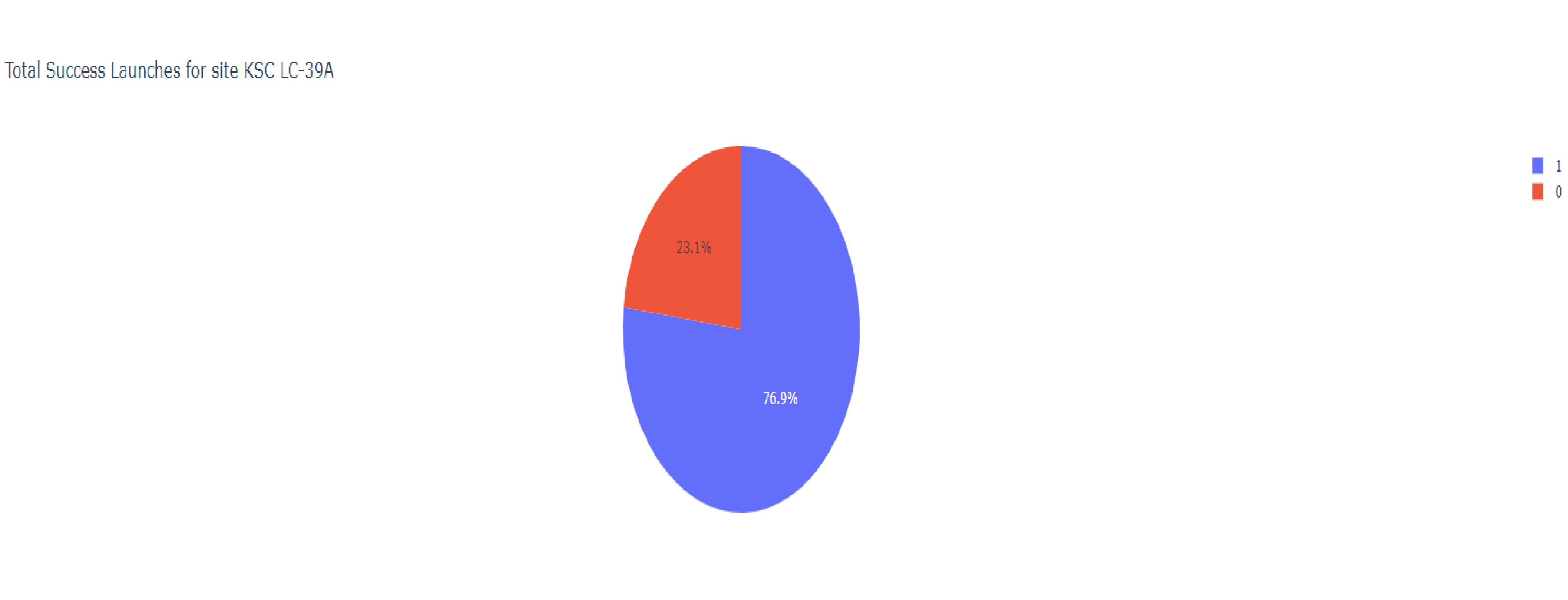
Build a Dashboard with Plotly Dash

Percentage of success from each launch site



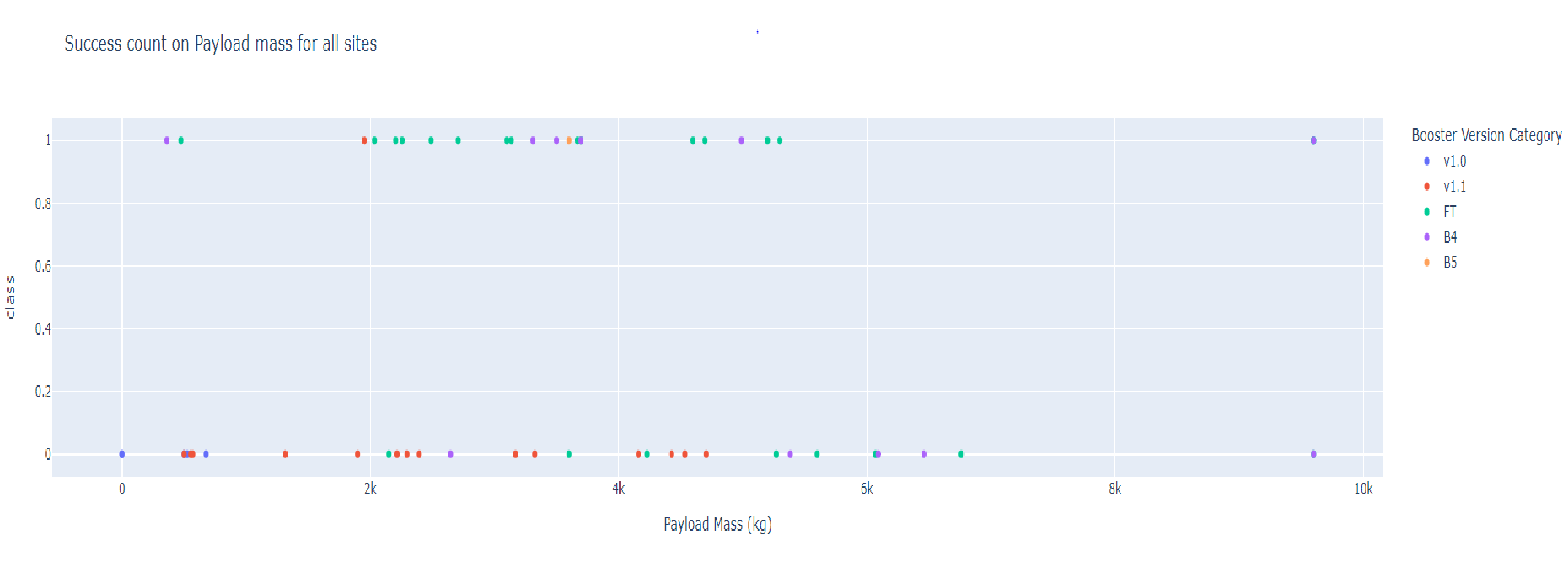
- The launch site with the highest success rate is KSC LC-39A with success rate of 41.7%

Success ratio for the Launch site with highest success rate



- KSC LC-39A has the highest success rate with successful launches of 76.9% and failure of 23.1%.

Launch success per payload mass

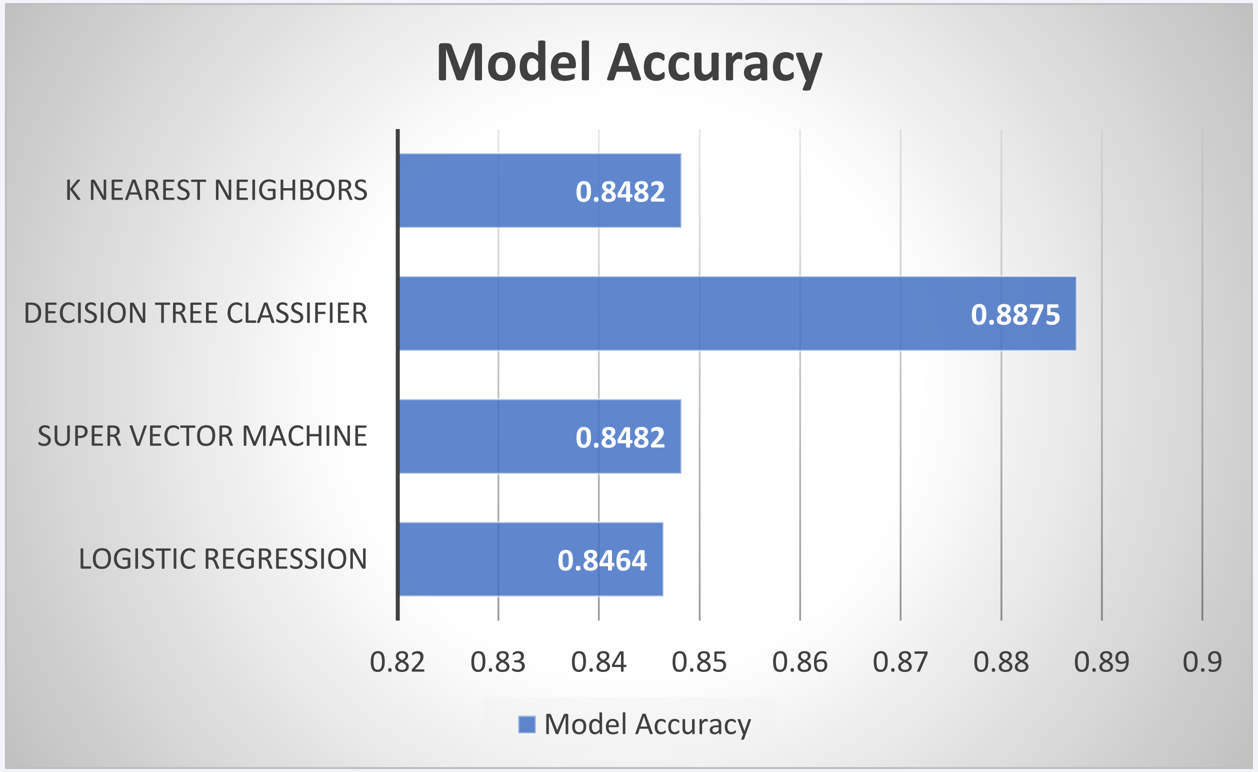


- From the plot we can see that the success rate for payload of more than 6000kg is low.

Section 5

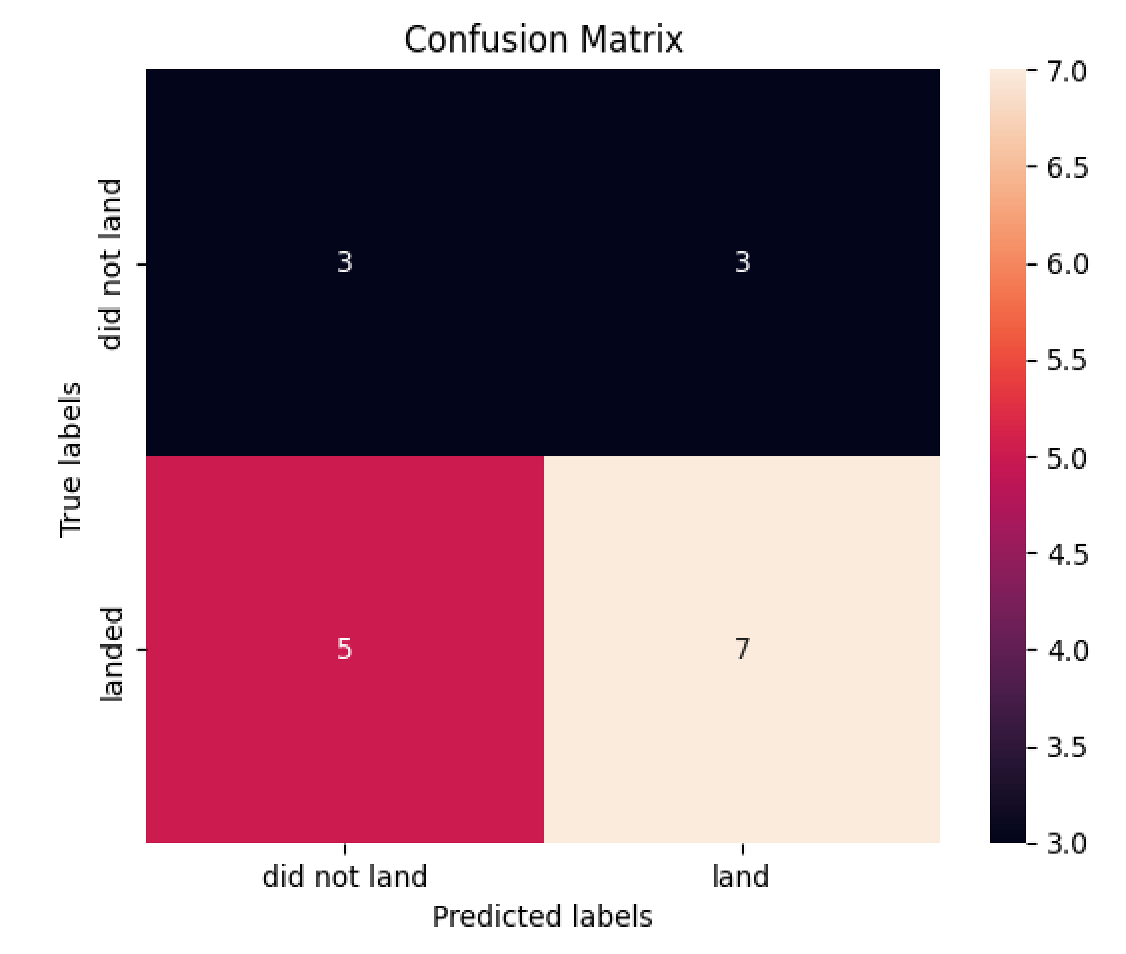
Predictive Analysis (Classification)

Classification Accuracy



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

Confusion Matrix



- The confusion matrix is a performance measure for a classification problem. It can have outputs of two or more.
- In this study, we can see that the confusion matrix for the decision tree classifier shows that out of 18 missions, 12 were labeled as landed and 7 were correctly predicted and 5 were incorrectly predicted.

Conclusions

We can conclude from this study that:

- Launch site with higher numbers of flight amounts has a greater success rate.
- And we have observed that the launch success rate begin to improve from 2013.
- We also observed that orbits with higher success rate are ES-L1, GEO, HEO, SSO, VLEO.
- The sites with the most successful launches is KSC LC-39A.
- The best model to use for predicting the success or failure of rocket landing is the decision tree classifier.

Thank you!