IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

PRATIKSHA G RAO
12TH APRIL 2022

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

- 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 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 needs to be in place to ensure a successful landing

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
    - Missing Values were taken care of by proper replacements
- 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 such as Logistic regression, KNN, Decision Trees and SVM.

Data Collection

- The data was collected as explained below:
  - 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

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

- The GitHub link to the notebook is https://github.com/PratikshaGRao/IBM-Data-Science-Capstone-SpaceX/blob/main/Data%20Collection%20API.ipynb

atson Studio Project / Data Collection API

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

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

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

Out[10]: 200

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

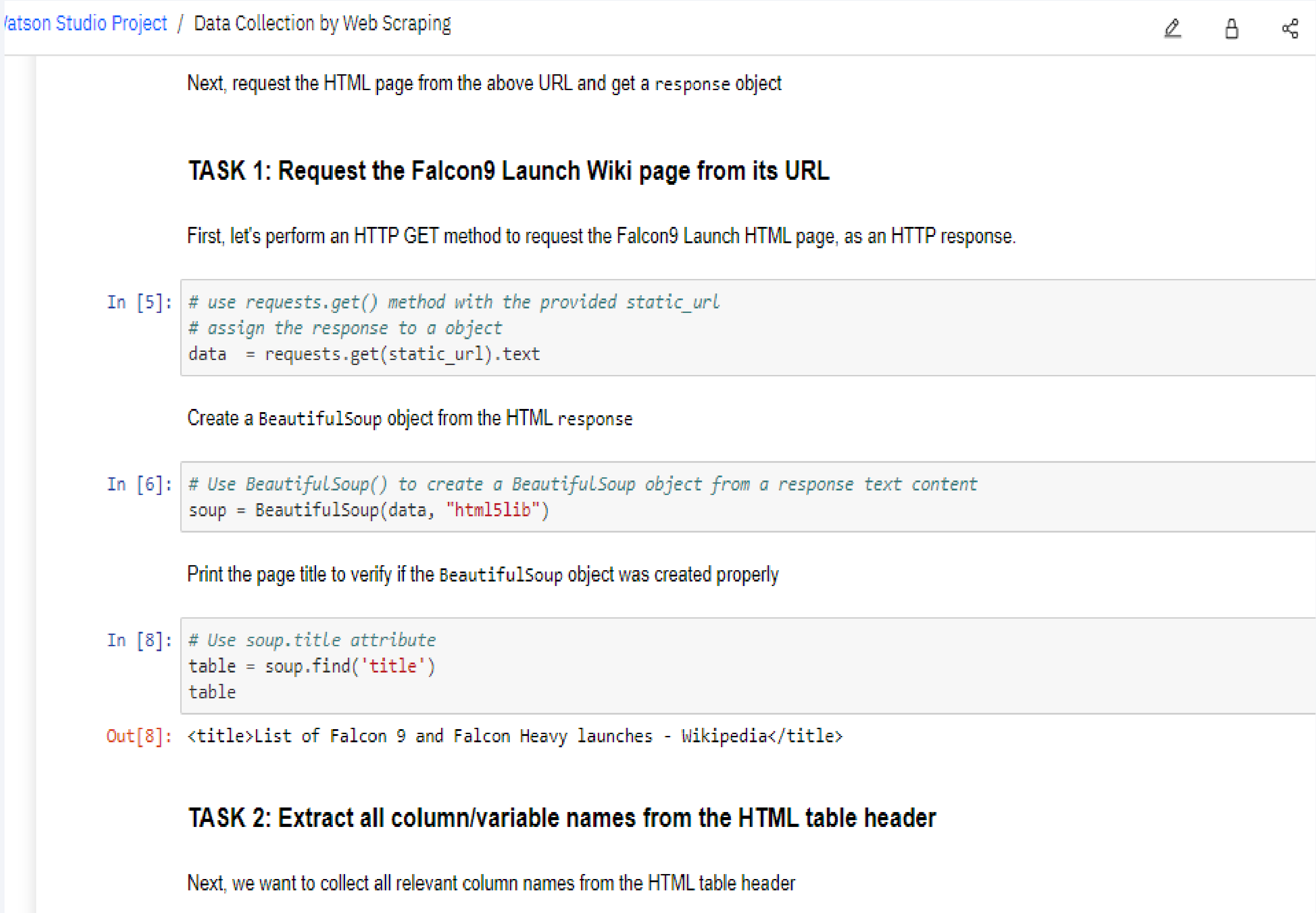
In [24]:
```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url)
data = response.json()
data = pd.json_normalize(data)
```

Using the dataframe `data` print the first 5 rows

In [25]:
```
# Get the head of the dataframe
data.head(5)
```

Data Collection - Scraping

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The link to the notebook is https://github.com/PratikshaGRao/IBM-Data-Science-Capstone-SpaceX/blob/main/Data%20Collection%20by%20Web%20Scraping.ipynb



/atson Studio Project / Data Collection by Web Scraping

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.

In [5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
data  = requests.get(static_url).text
```

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

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data, "html5lib")
```

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

In [8]:
```
# Use soup.title attribute
table = soup.find('title')
table
```

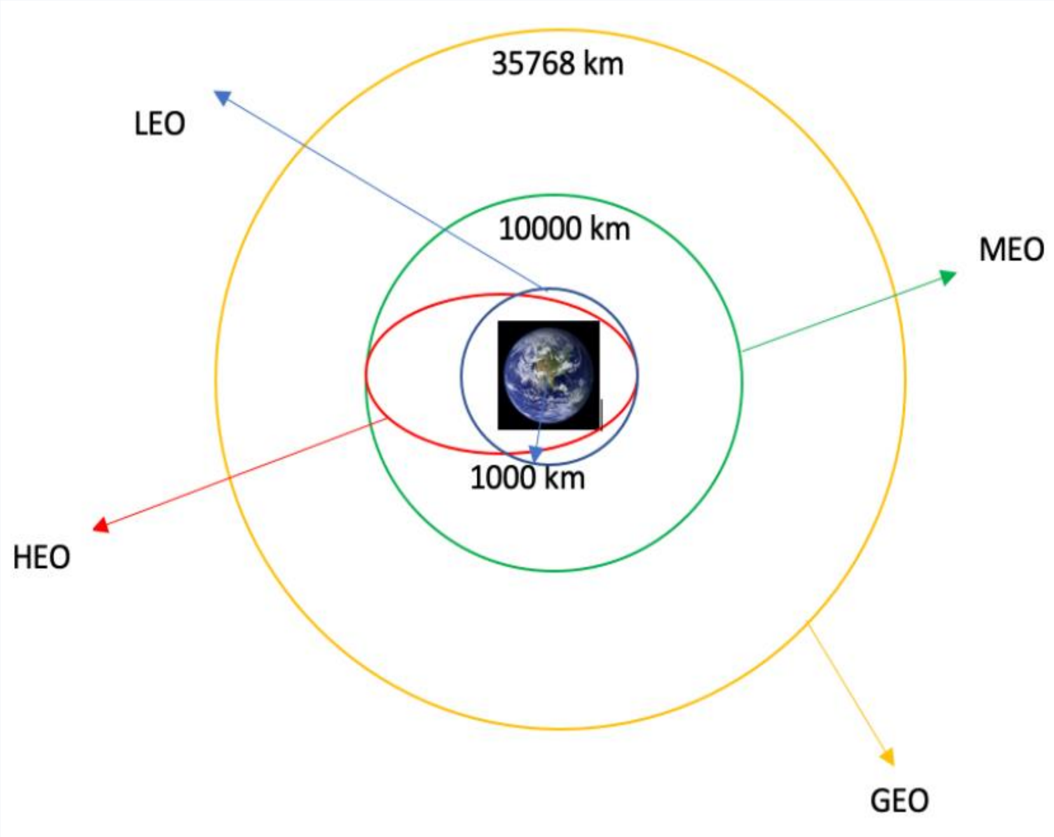
Out[8]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

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

Data Wrangling

- We performed exploratory data analysis (EDA) to find some patterns in the data and determined the training labels for supervised models.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We also calculated the mission outcome for each orbit type
- Finally, we created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is https://github.com/PratikshaGRao/IBM-Data-Science-Capstone-SpaceX/blob/main/EDA%20Lab.ipynb

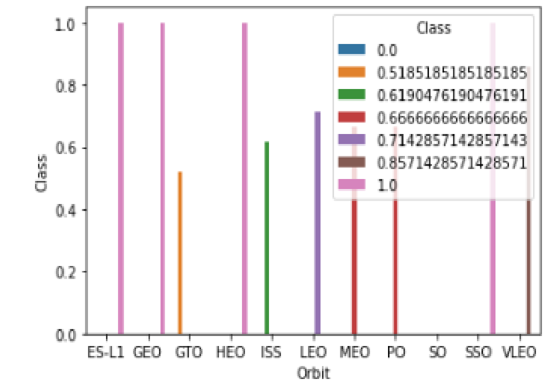


EDA with SQL

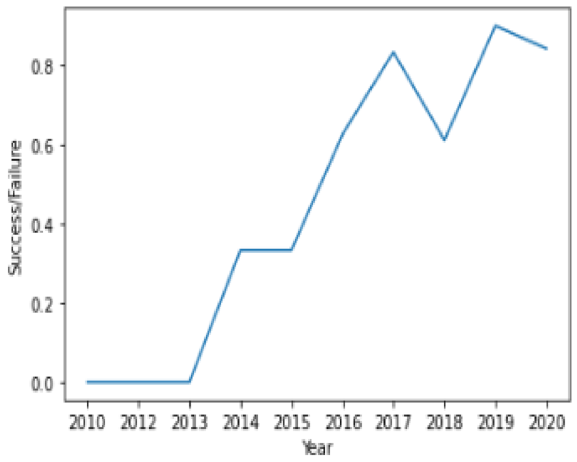
- We loaded the SpaceX dataset into a PostgreSQL database without leaving the Jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters 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 failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook is https://github.com/PratikshaGRao/IBM-Data-Science-Capstone-SpaceX/blob/main/EDA%20with%20SQL.ipynb

EDA with Data Visualisation

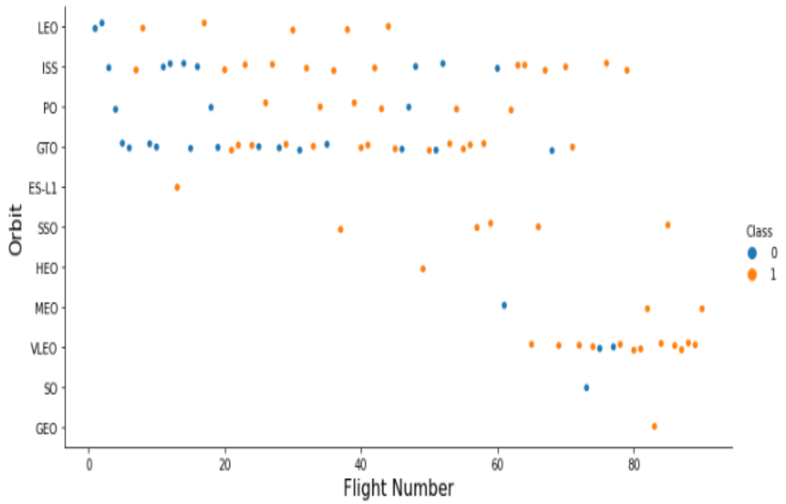
- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.
- The link to the notebook is https://github.com/PratikshaGRao/IBM-Data-Science-Capstone-SpaceX/blob/main/EDA%20with%20Data%20visualization.ipynb



Analyze the ploted bar chart try to find which orbits have high sucess rate.

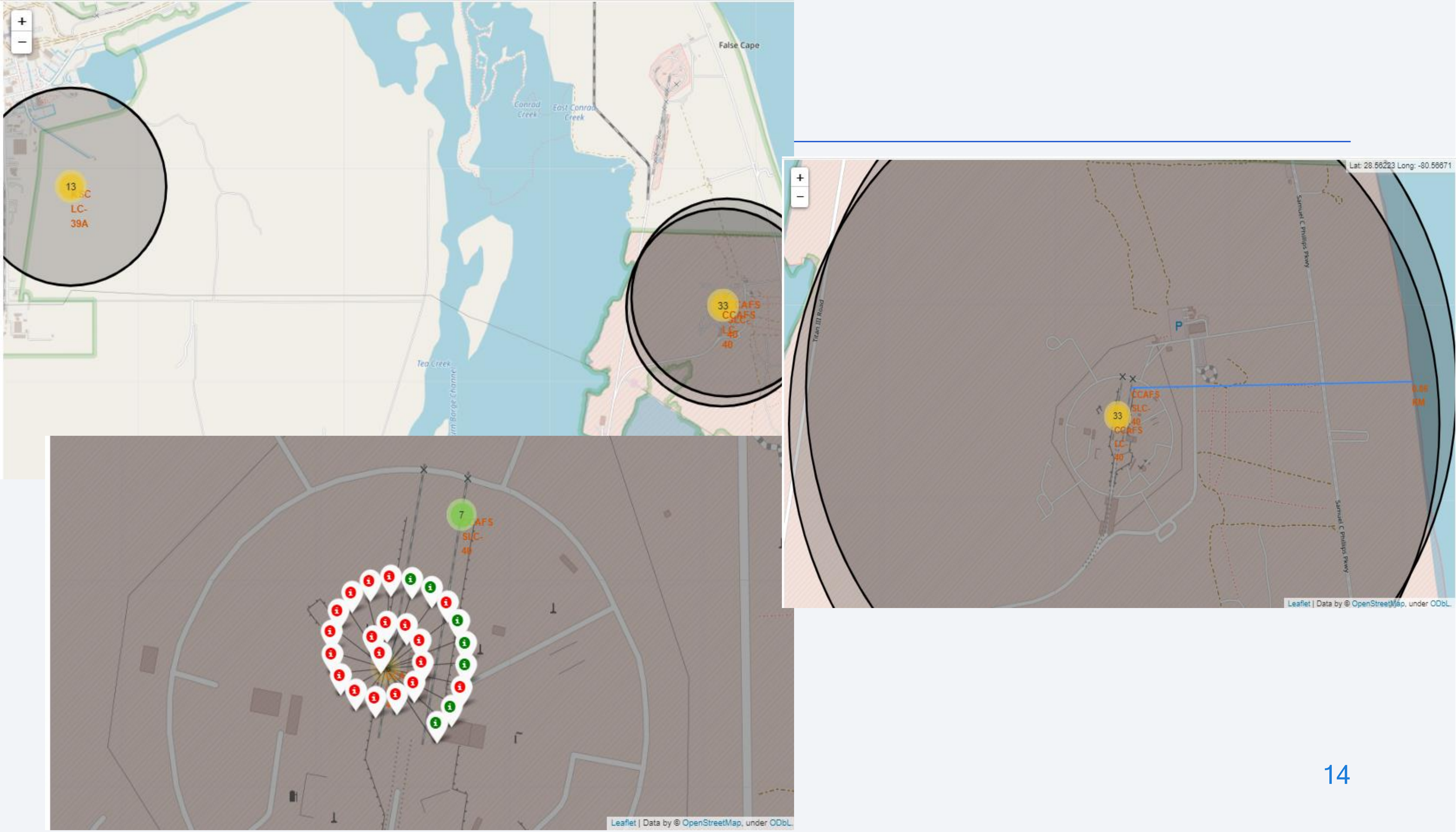


you can observe that the sucess rate since 2013 kept increasir



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.
- The link to the notebook is https://github.com/PratikshaGRao/IBM-Data-Science-Capstone-SpaceX/blob/main/Folium%20Lab.ipynb


False Cape
Conrad Creek
East Conrad Creek
Tea Creek
13
LC-
39A
33
7
CCAFS
SLC-
40
Titan III Road
Samuel C Phillips Pkwy
P
Lat: 28.56223 Long: -80.56671
Leaflet | Data by © OpenStreetMap, under ODbL.

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.
- The link to the notebook is https://github.com/PratikshaGRao/IBM-Data-Science-Capstone-SpaceX/blob/main/Machine%20Learning%20Prediction.ipynb

Results

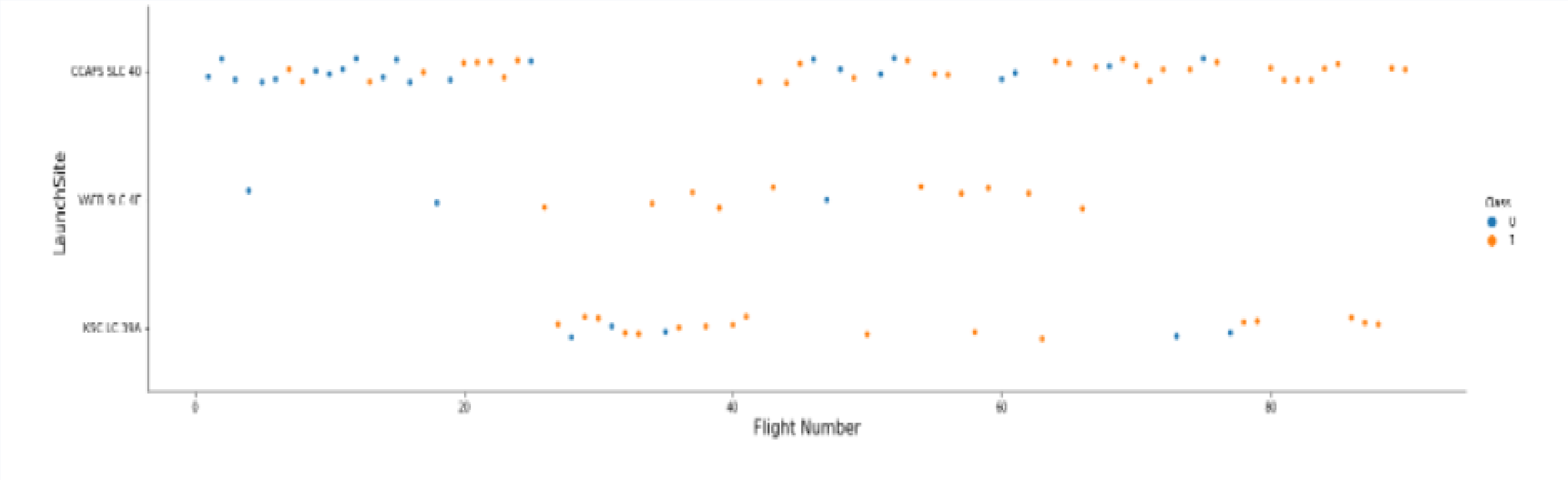
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

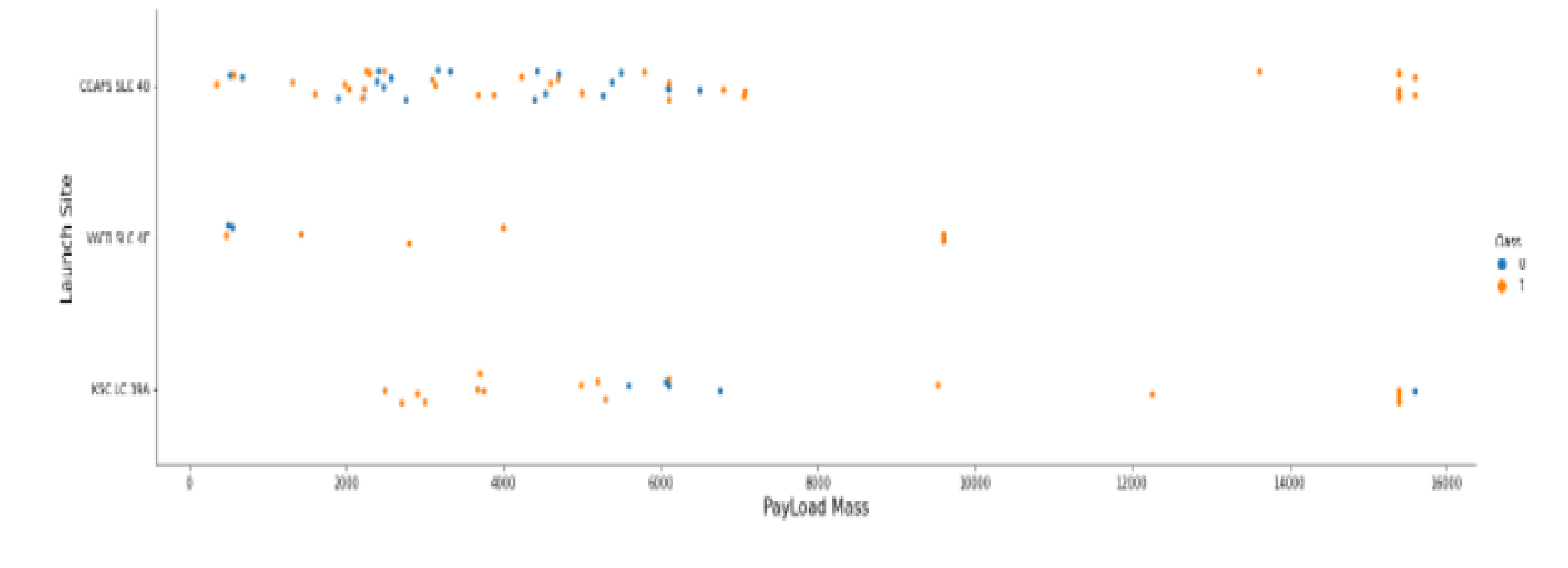
Flight Number vs. Launch Site

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



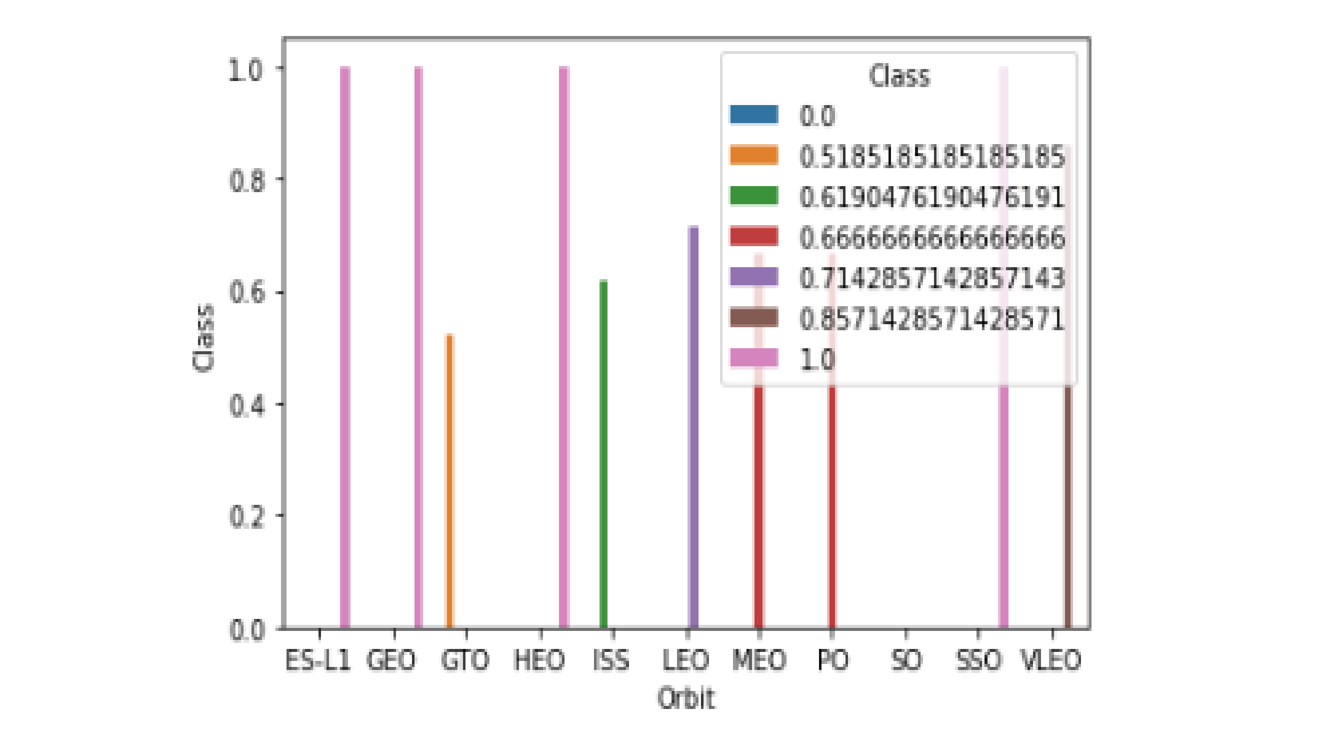
Payload vs. Launch Site

- The greater the payload mass for launch site CCAFS SLC 40, the higher the success rate for the rocket.



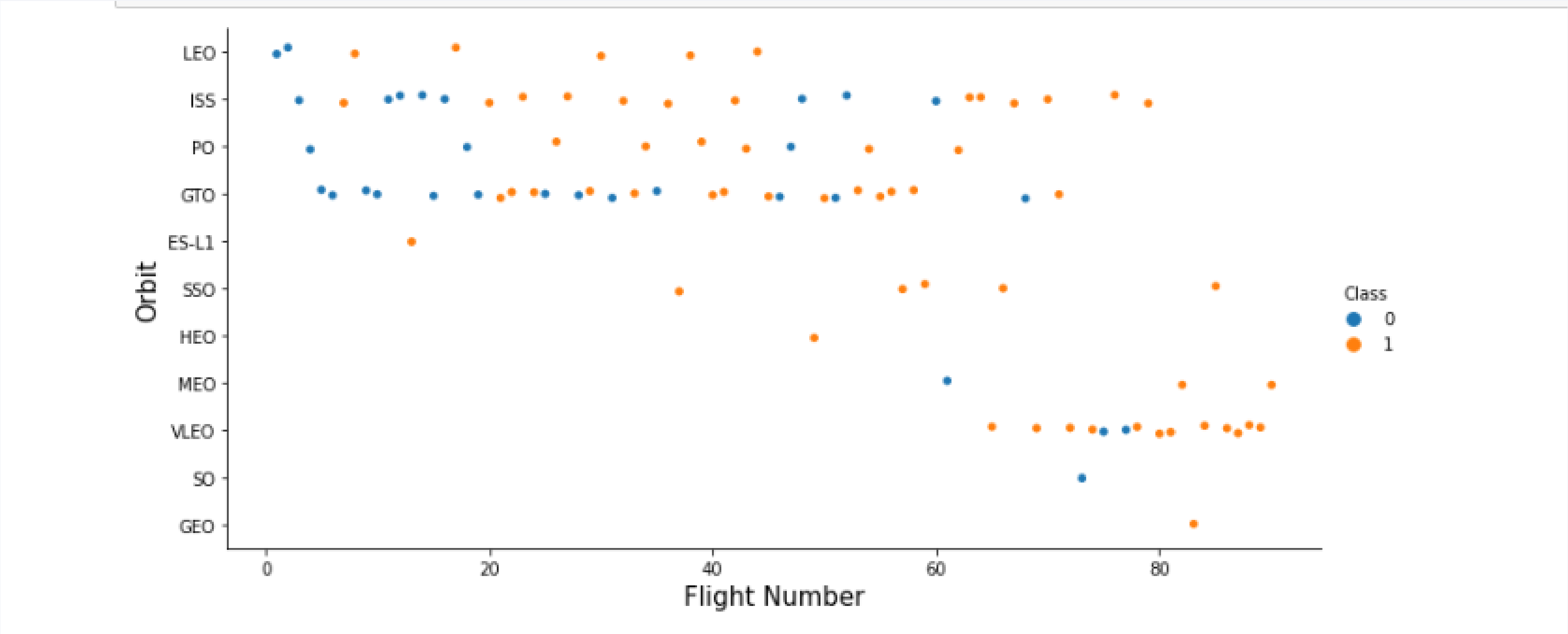
Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



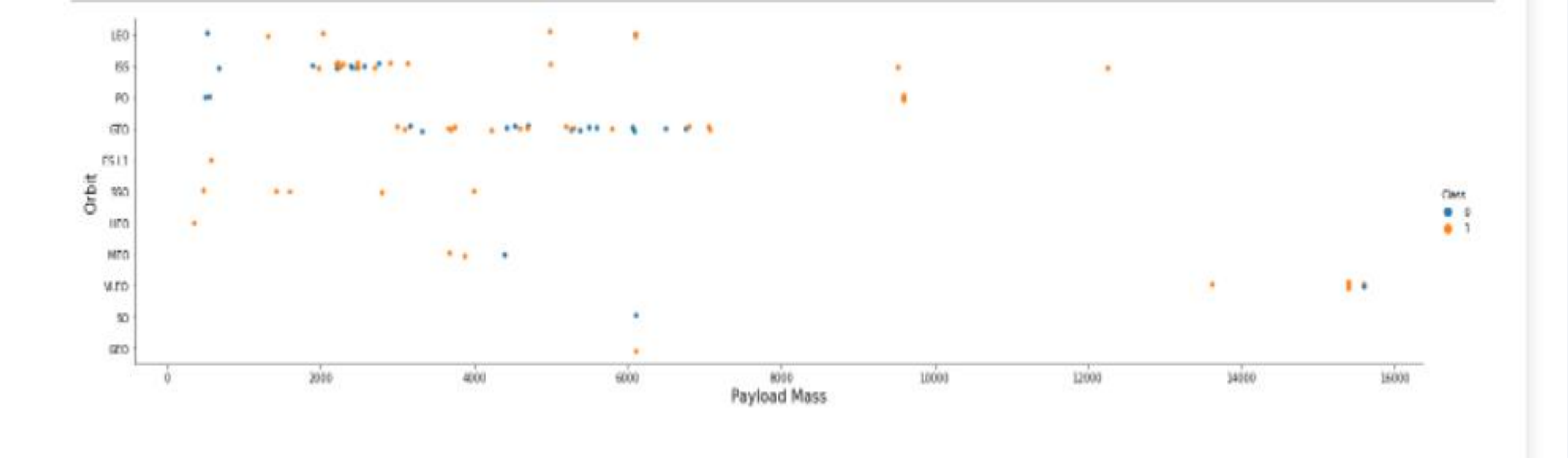
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



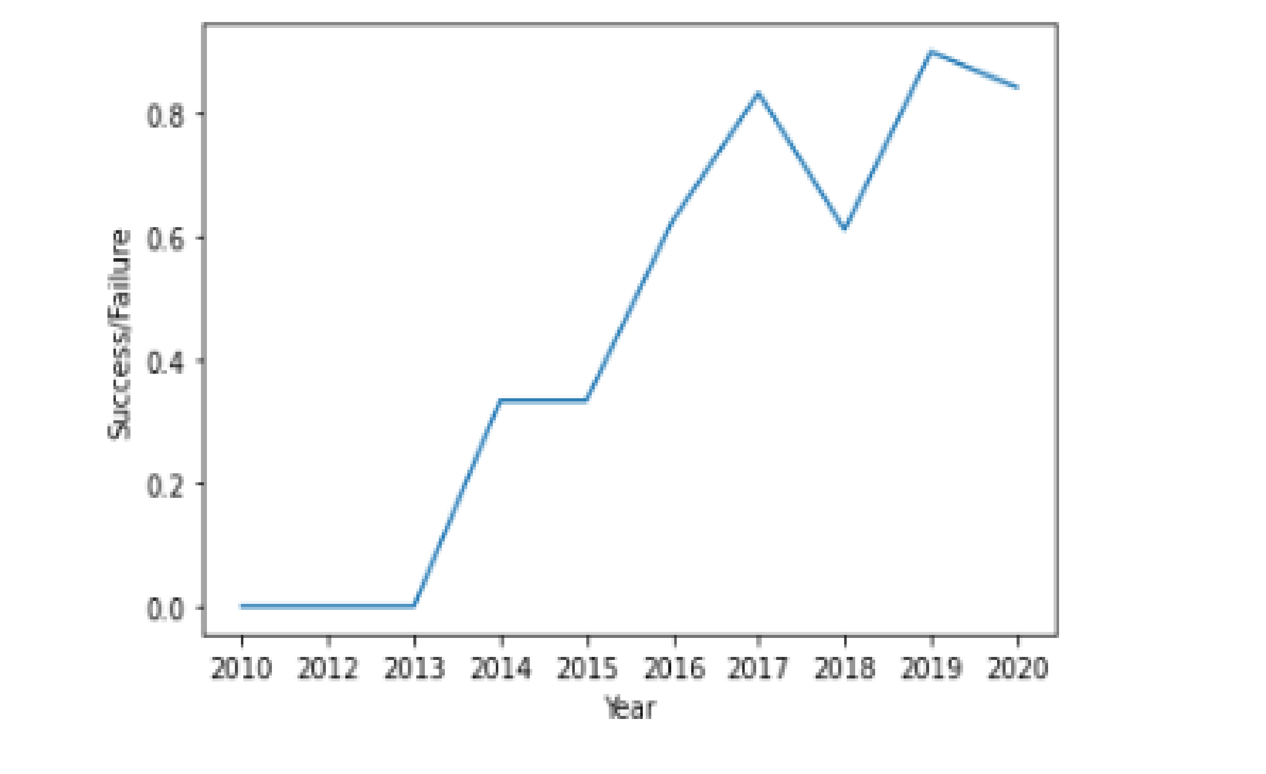
Payload vs. Orbit Type

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



Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

- We used the key word UNIQUE to show only unique launch sites from the SpaceX data.

Task 1

Display the names of the unique launch sites in the space mission

In [13]:
```
%%sql

select UNIQUE LAUNCH_SITE from SPACEXTBL
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[13]:

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

Launch Site Names Begin with 'CCA'

- We used the query below to display 5 records where launch sites begin with `CCA`

Task 2

Display 5 records where launch sites begin with the string 'CCA'

In [17]:
```
%%sql
select * from SPACEXTBL where LAUNCH_SITE LIKE 'CCA%' limit 5
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[17]:

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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 calculated the total payload carried by boosters from NASA as 45596 using the query below:

Task 3

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

In [39]:
```
%%sql
select SUM(payload_mass__kg_) as Total_Payload_Mass from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[39]:

total_payload_mass
45596

Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928

Task 4

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

In [38]:
```
%%sql
select AVG(payload_mass__kg_) as Average_Payload_Mass from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[38]:

average_payload_mass
2928

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

Task 5

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

Hint:Use min function

In [40]:
```
%%sql
select MIN(DATE) as First_succcessful_landing_date from SPACEXTBL where landing__outcome ='Success (ground pad)'
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[40]:

first_succcessful_landing_date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [34]:
```
%%sql
select BOOSTER_VERSION FROM SPACEXTBL where (landing__outcome = 'Success (drone ship)') and (payload_mass__kg_ between 4000 and 6000)
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[34]:

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 counted the mission outcome and used a group by on Mission_Outcome to find number of success or failure.

Task 7

List the total number of successful and failure mission outcomes

In [41]:
```
%%sql
Select MISSION_OUTCOME, count(MISSION_OUTCOME) as count from SPACEXTBL GROUP BY MISSION_OUTCOME
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqblod8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[41]:

mission_outcome	COUNT
Failure (in flight)	1
Success	99
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.

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

In [45]:
```
%%sql
select BOOSTER_VERSION FROM SPACEXTBL where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL) group by BOOSTER_
VERSION
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[45]:

booster_version
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

2015 Launch Records

- We used a combinations of the **WHERE** clause and **AND** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

Task 9

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [50]:
```
%%sql
select landing__outcome, booster_version, launch_site from SPACEXTBL where (landing__outcome = 'Failure (drone ship)') and (year
(date) = '2015')
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[50]:

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
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 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

Task 10

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

In [60]:

```
%%sql
select landing__outcome, count(landing__outcome) as Rank from SPACEXTBL where date between '2010-06-04' and '2017-03-20' group b
y landing__outcome order by count(landing__outcome) desc
```

```
 * ibm_db_sa://zvx44101:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

Out[60]:

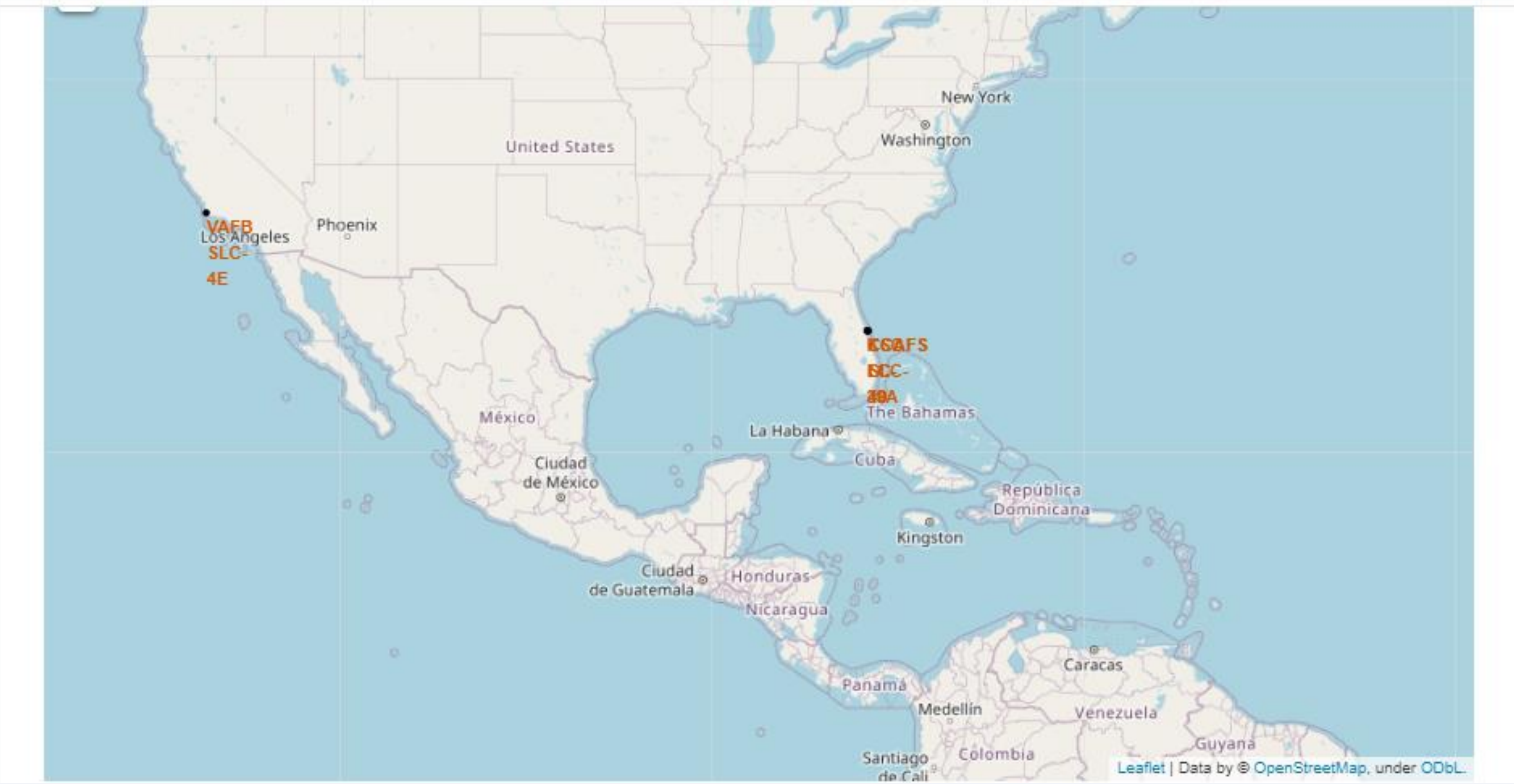
landing__outcome	RANK
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

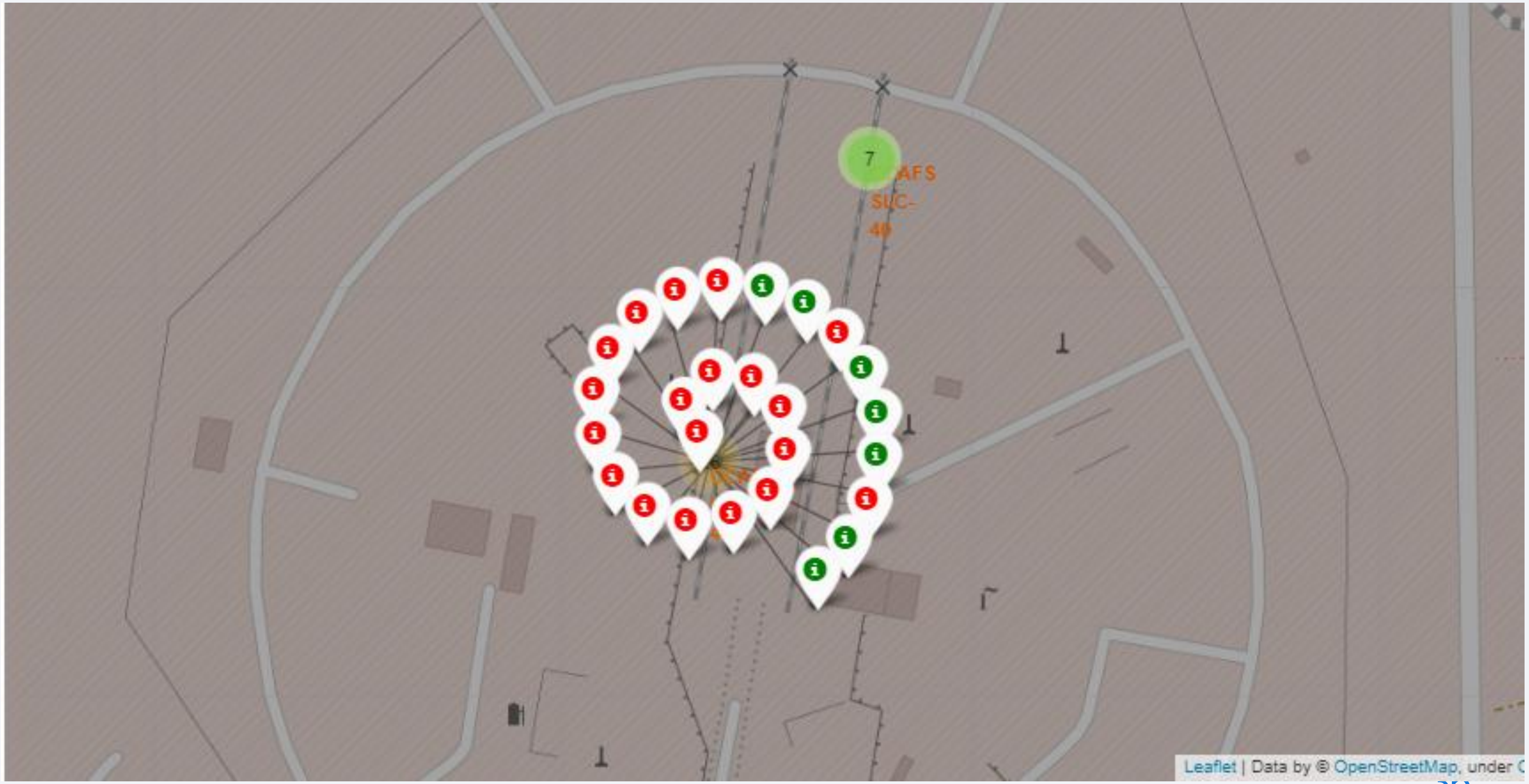
All launch sites global map markers

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



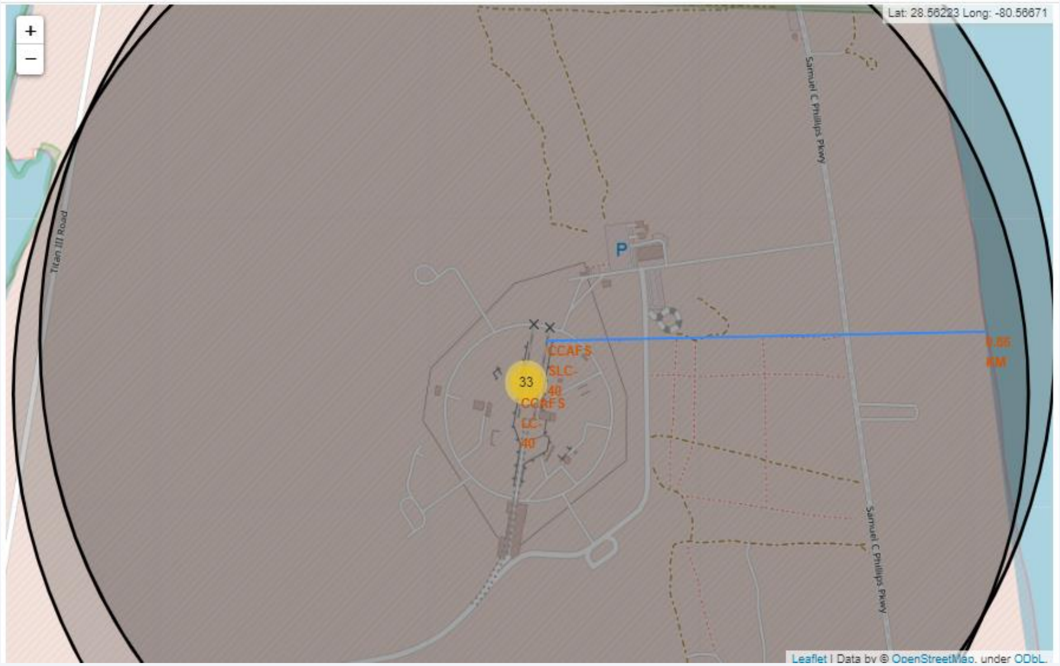
Markers showing launch sites with color labels

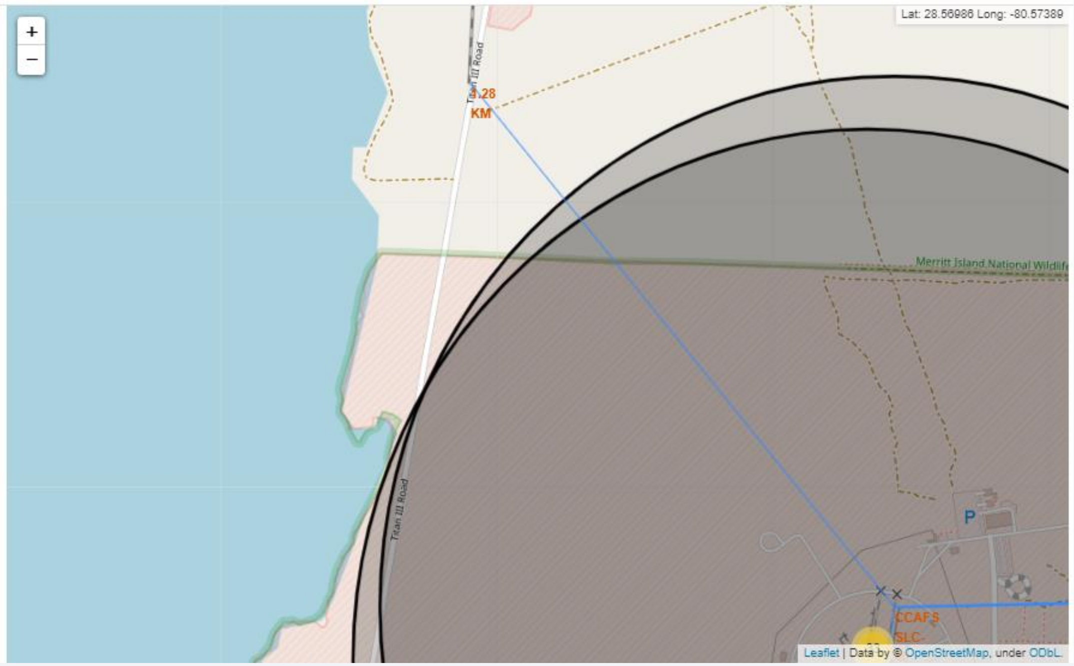
- Green marker shows successful launches where as Red marker shows Failures



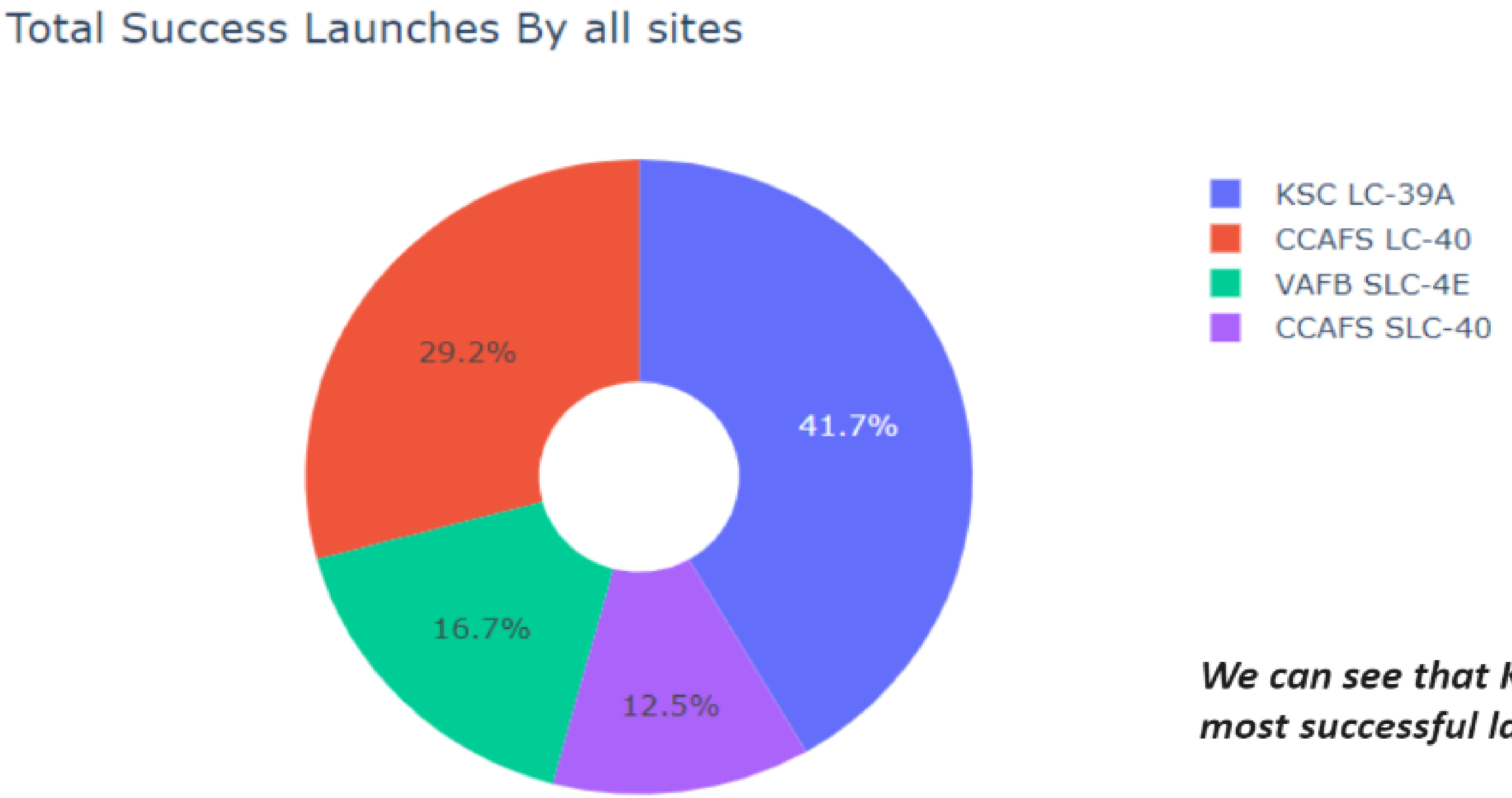
Launch Site distance to landmarks

- Distance to closest highway and closest railway line, respectively.



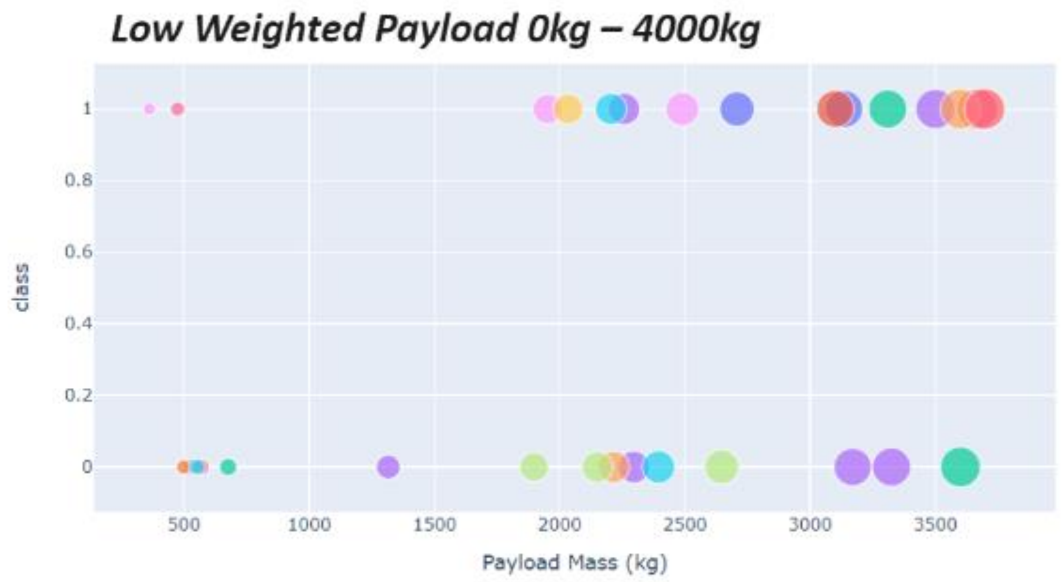


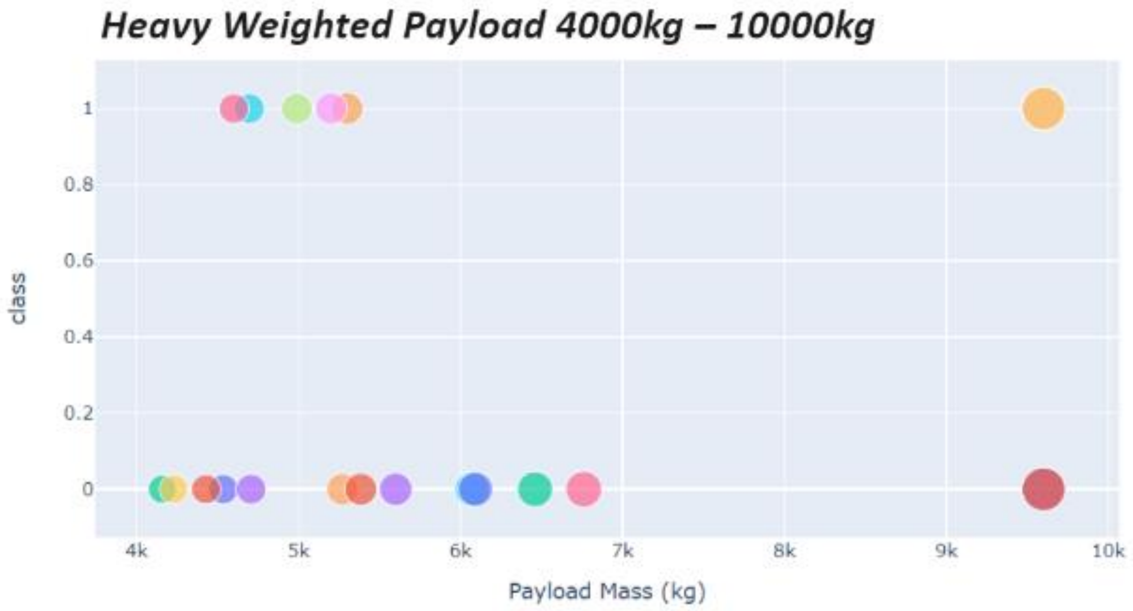
Pie chart showing the success percentage achieved by each launch site



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

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 logistic regression classifier is the model with the highest classification accuracy.

In [71]:
```
accuracy = [svm_score, logreg_score, knn_score, tree_score]
accuracy = [i * 100 for i in accuracy]

method = ['Support Vector Machine', 'Logistic Regression', 'K Nearest Neighbour', 'Decision Tree']
models = {'ML Method':method, 'Accuracy Score (%)':accuracy}

ML_df = pd.DataFrame(models)
ML_df
```

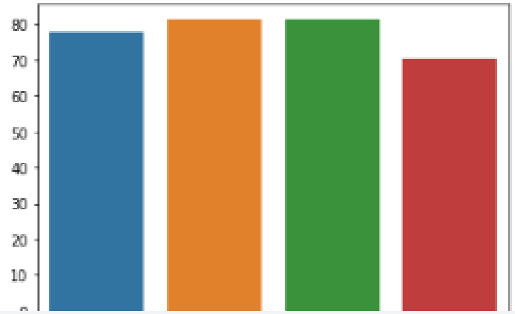
Out[71]:

	ML Method	Accuracy Score (%)
0	Support Vector Machine	77.777778
1	Logistic Regression	81.481481
2	K Nearest Neighbour	81.481481
3	Decision Tree	70.370370

In [76]:
```
sns.barplot(method, accuracy, data=ML_df)
```

```
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variables a
s keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without
an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[76]: <AxesSubplot:>



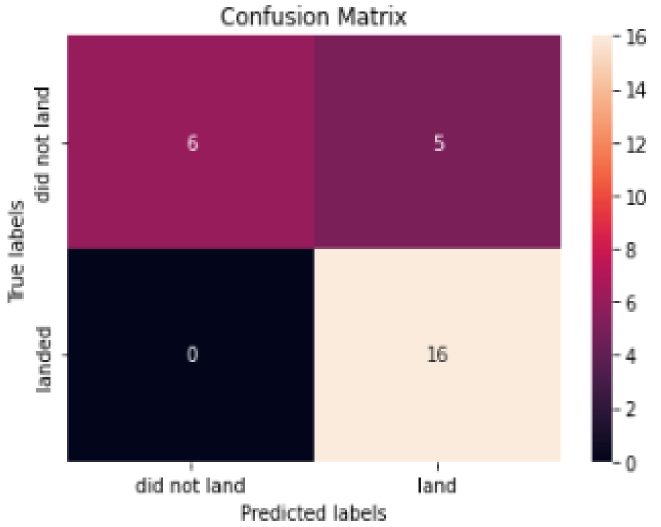
Confusion Matrix

In [58]:
```
logreg_score = logreg_cv.score(X_test, Y_test)
print("Logistic Score :", logreg_score)
```

```
Logistic Score : 0.8148148148148148
```

Lets look at the confusion matrix:

In [51]:
```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



- The confusion matrix for the logistic regression classifier shows that the classifier 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 Logistic Regression classifier is the best machine learning algorithm for this task.

Thank you!