IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

James Nainggolan
Jun 10, 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
- Interactive Analytics
- Predictive Analytics

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 program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected using SpaceX API and Web Scraping
- Perform data wrangling
  - Using One-hot encoder for 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.

And decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize(). Then cleaned the data, checked for missing values and fill in missing values where necessary.

And performing 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

Data collection with SpaceX REST calls using key phrases and flowcharts

Notebook : https://github.com/jmsxngl/IBM-Data-Science/blob/0405d6b628998a17d54f4ee9a2e66788be6f00a2/jupyter-labs-spacex-data-collection-api.ipynb

Now let's start requesting rocket launch data from SpaceX API with the following URL:

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

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

```
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

```
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())
```

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head()
```

Data Collection - Scraping

Web scraping process using key phrases and flowcharts

Notebook : https://github.com/jmsxngl/IBM-Data-Science/blob/0405d6b628998a17d54f4ee9a2e66788be6f00a2/jupyter-labs-webscraping.ipynb

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

on 9th June 2021

To keep the lab tasks consistent, you will be asked to scrape the data from a snapshot of the `List of Falcon 9 and Falcon Heavy launches` Wikipage updated

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
response = requests.get(static_url).text
```

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, 'html.parser')
```

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

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

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

Next, we just need to iterate through the `<th>` elements and apply the provided `extract_column_from_header()` to extract column name one by one

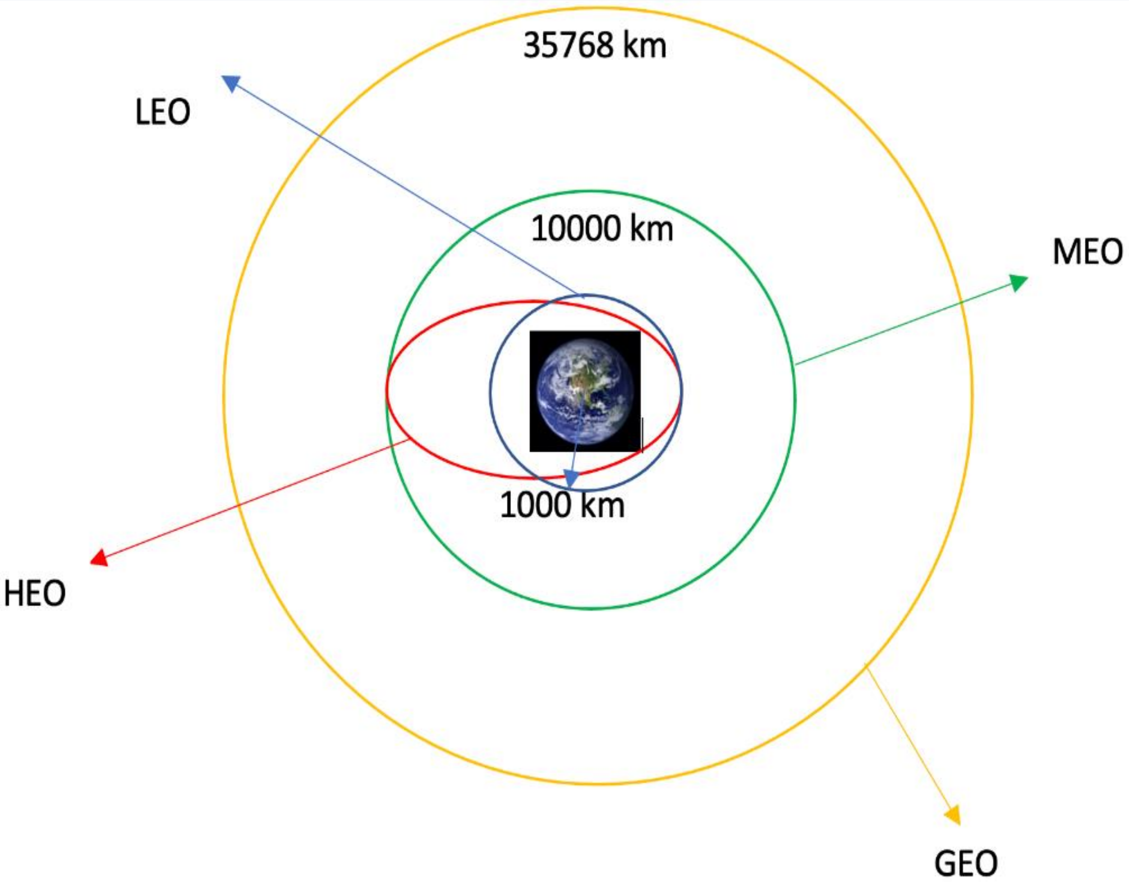
```
# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

column_names = []
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if (name != None and len(name) > 0):
        column_names.append(name)
```

Data Wrangling

With exploratory data analysis and determined the training labels.
And calculated the number of launches at each site, and the number and occurrence of each orbits then created landing outcome label from outcome column and exported the results to csv.

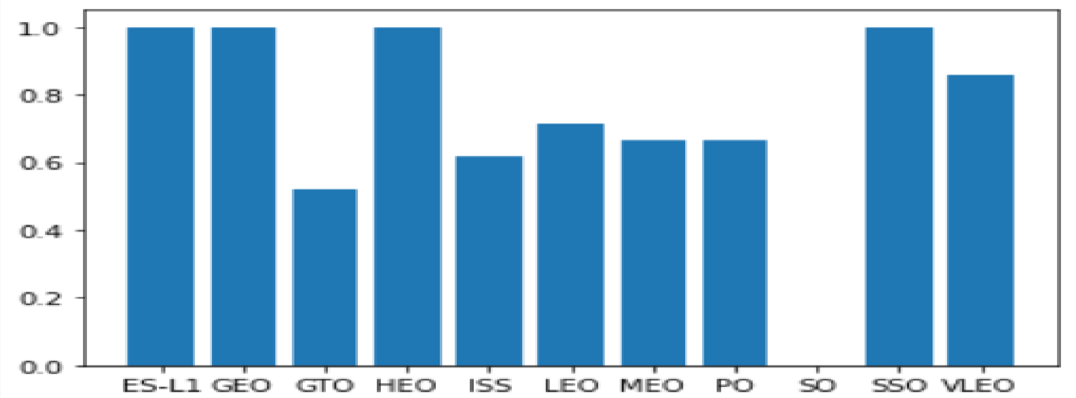
Notebook : https://github.com/jmsxngl/IBM-Data-Science/blob/0405d6b628998a17d54f4ee9a2e66788be6f00a2/labs-jupyter-spacex-Data%20wrangling.ipynb

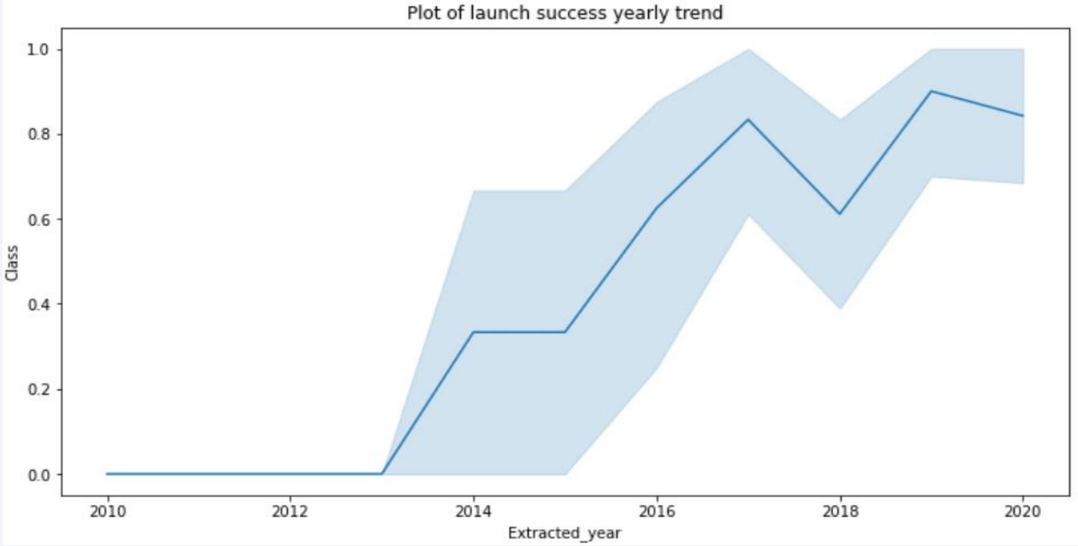


EDA with Data Visualization

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

Notebook : https://github.com/jmsxngl/IBM-Data-Science/blob/56a3604f348738876a80b4b3c7afa1ef46dfa649/jupyter-labs-eda-dataviz.ipynb





EDA with SQL

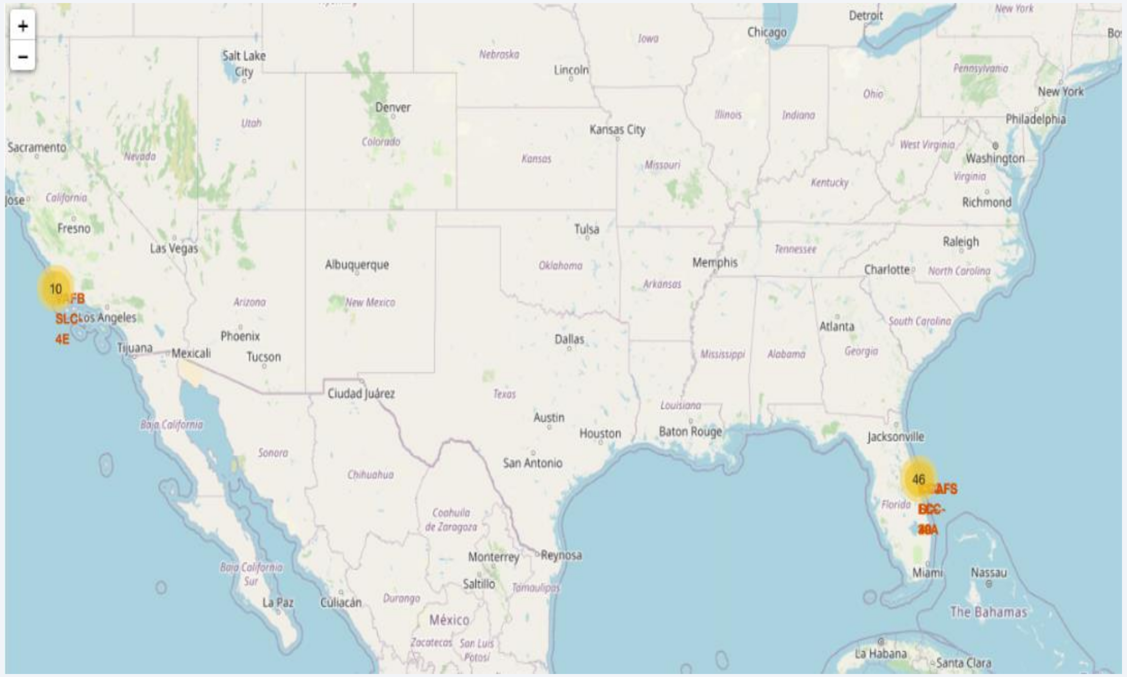
Load the SpaceX dataset into a PostgreSQL database. And apply EDA with SQL to get insight from the data. Wrote the queries to find out for instance:

- 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.

Notebook : https://github.com/jmsxngl/IBM-Data-Science/blob/56a3604f348738876a80b4b3c7afa1ef46dfa649/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.



Notebook : https://github.com/jmsxngl/IBM-Data-Science/blob/56a3604f348738876a80b4b3c7afa1ef46dfa649/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- An interactive dashboard with Plotly dash
- Pie charts showing the total launches by a certain sites
- Scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

Notebook :

https://github.com/jmsxngl/IBM-Data-Science/blob/56a3604f348738876a80b4b3c7afa1ef46dfa649/spacex_dash_app.py

Predictive Analysis (Classification)

The data using numpy and pandas, transformed the data, split our data into training and testing.
Different machine learning models and tune different hyperparameters using GridSearchCV.
Use accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
Found the best performing classification model.

Notebook : https://github.com/jmsxngl/IBM-Data-Science/blob/56a3604f348738876a80b4b3c7afa1ef46dfa649/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

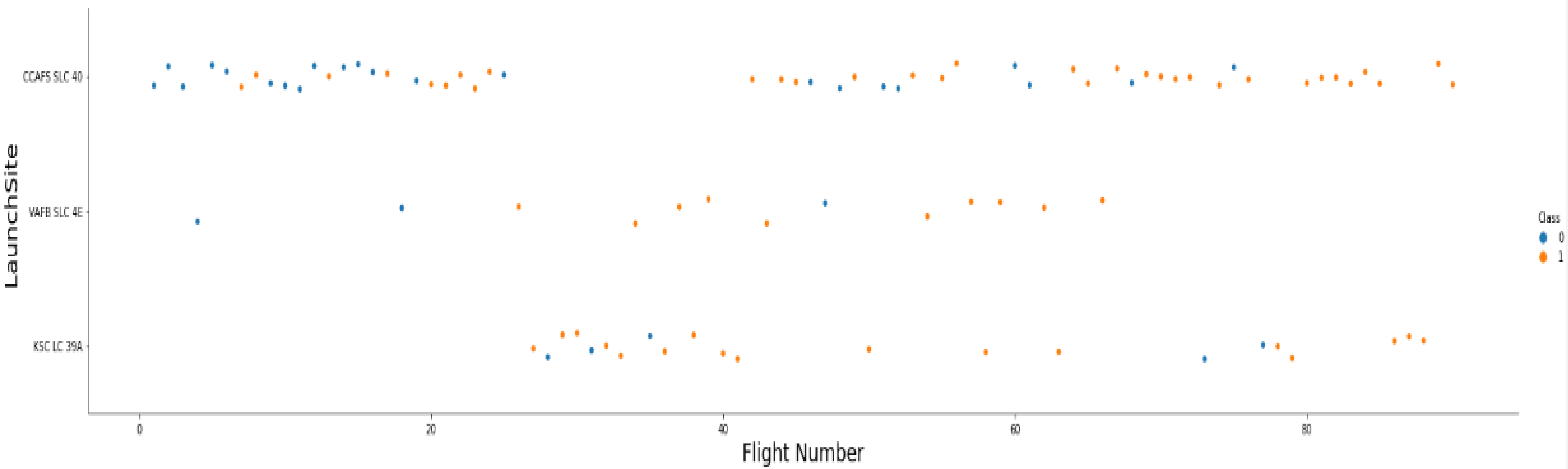
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

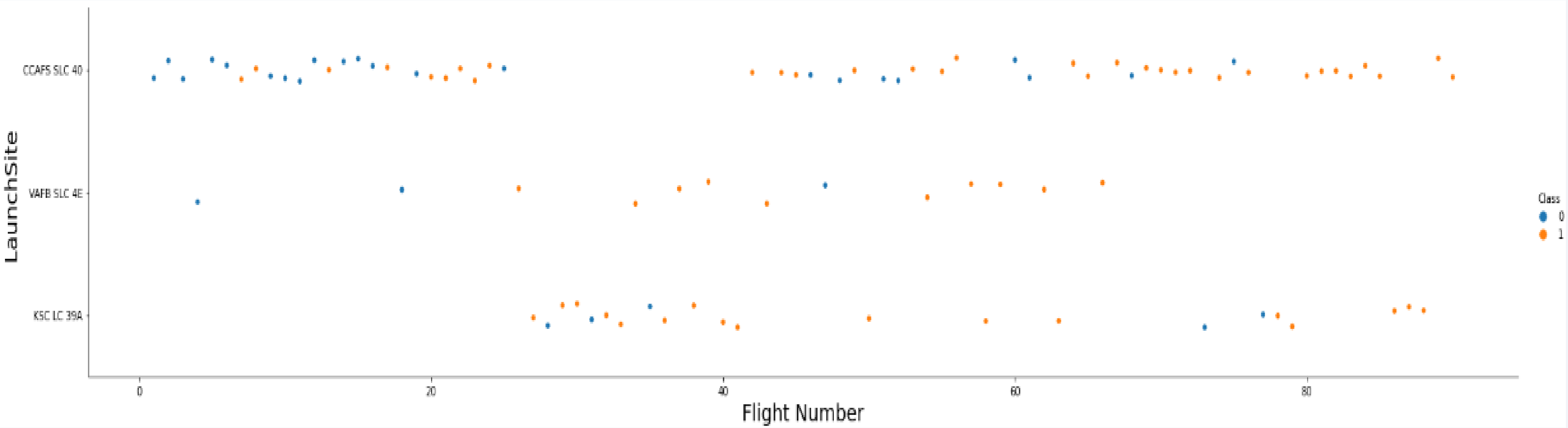
Flight Number vs. Launch Site

Found that the larger the flight amount at a launch site, the greater the success rate at a launch site.

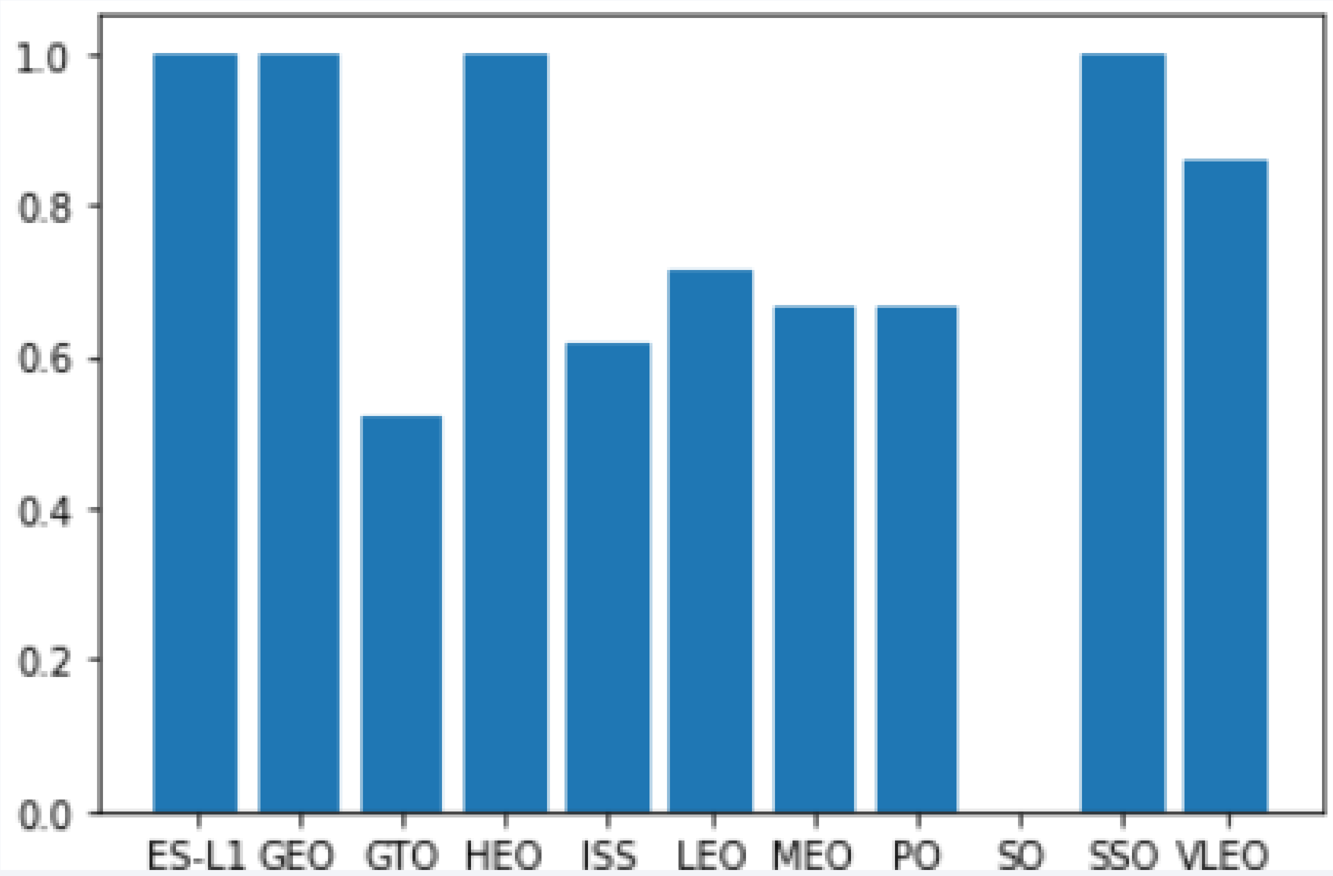


Payload vs. Launch Site

The higher success rate for the rocket depend on the the greater payload mass on launch site CCAFS SLC 40



Success Rate vs. Orbit Type



ES-L1

GEO

HEO

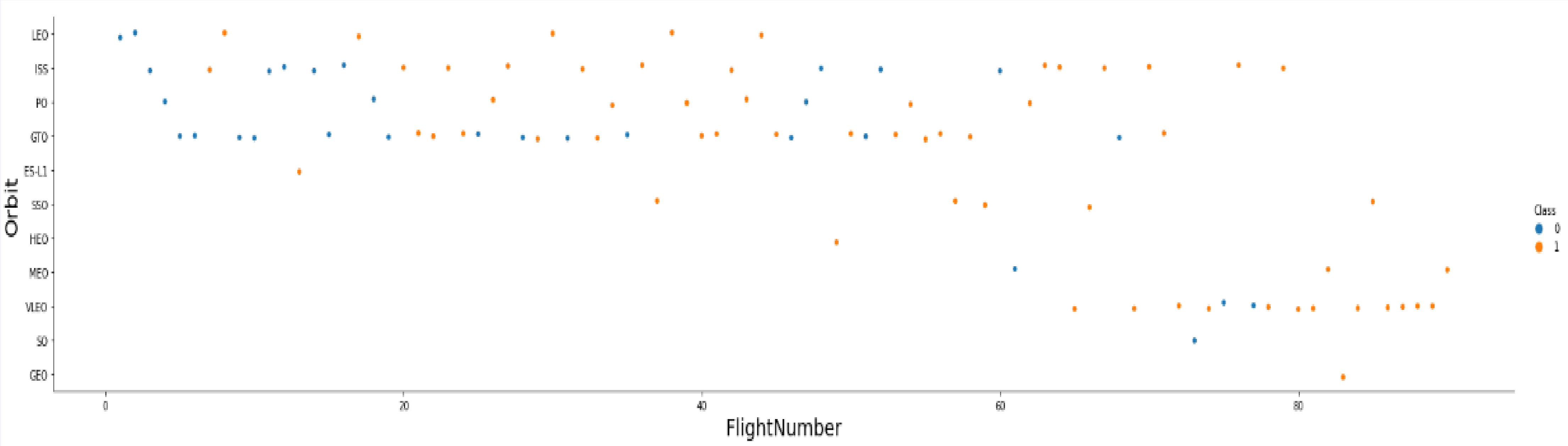
SSO

VLEO

The most success rate.

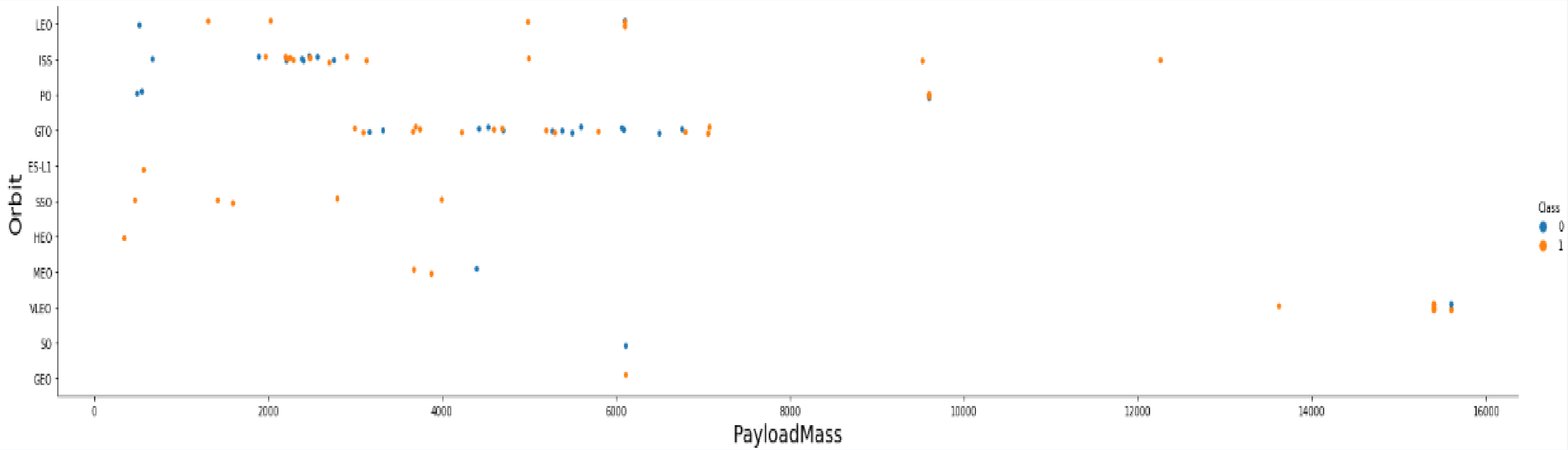
Flight Number vs. Orbit Type

The Flight Number vs. Orbit type.
Found 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.

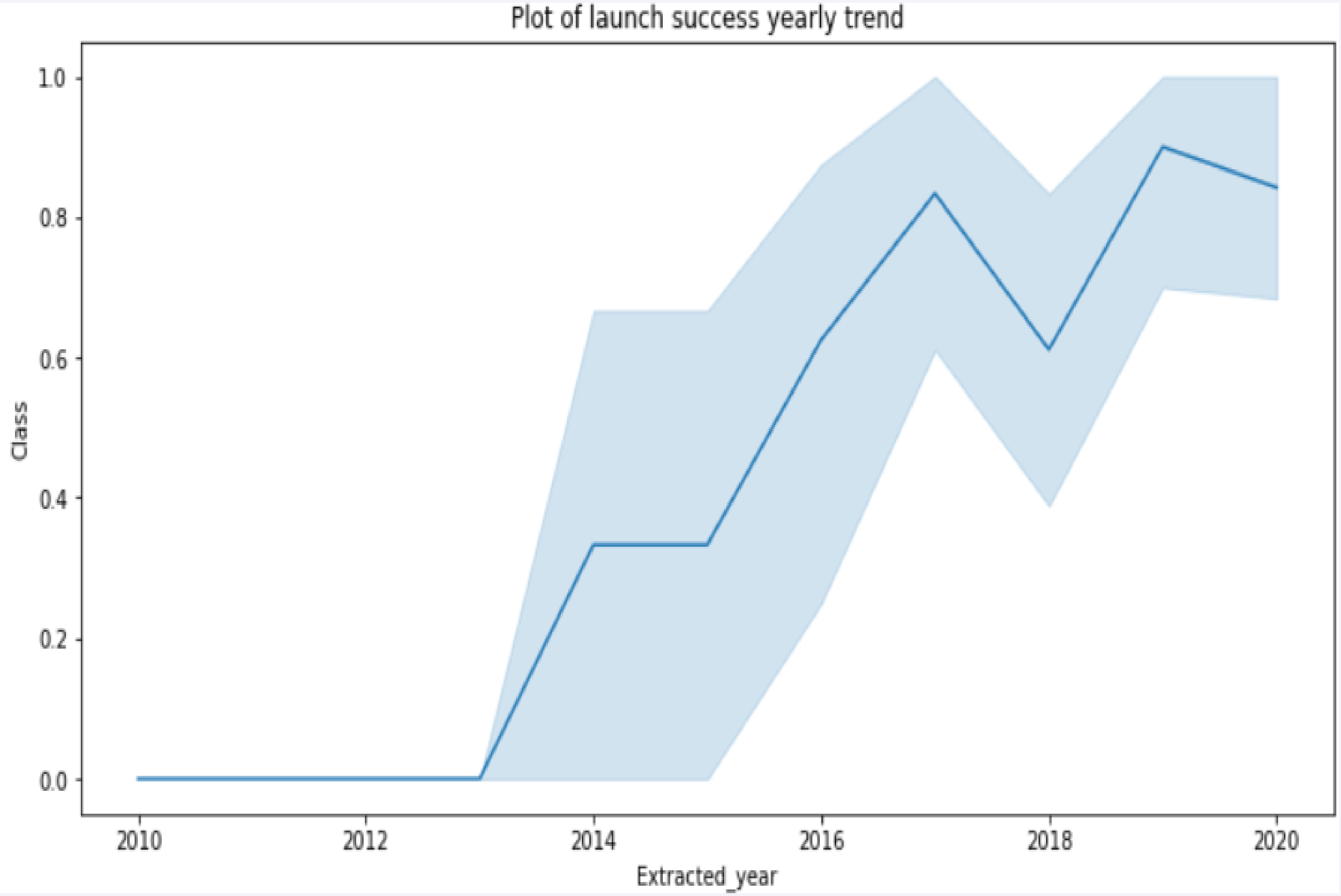


Payload vs. Orbit Type

The plot said, that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



Launch Success Yearly Trend



The success rate since 2013 kept on increasing till 2020.

All Launch Site Names

The unique launch sites from
SpaceX

```
%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'

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

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

Calculated the total payload carried by boosters from NASA

Display the total payload mass carried by boosters launched by 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

Calculated the average payload mass carried by booster version F9 v1.1

Display average payload mass carried by 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

The first successful landing outcome on ground pad

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

Hint:Use min function

```
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE "Landing_Outcome" LIKE 'Success (ground pad)';
```

	MIN(DATE)
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

WHERE clause to filter for boosters and applied the **AND** condition to determine successful landing with payload mass in range 4000 - 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_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

	Booster_Version
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

The total number of successful and failure mission outcomes

List the total number of successful and failure mission outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) from SPACEXTBL GROUP BY Mission_Outcome;
```

Mission_Outcome	COUNT(Mission_Outcome)
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

The names of the booster which have 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, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) from SPACEXTBL);
```

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

The failed landing_outcomes in drone ship, their booster versions, and launch site names for in 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.

```
%sql SELECT Booster_Version, Launch_Site,  LandingOutcome FROM SPACEXTBL WHERE "Landing_Outcome" LIKE 'Failure (drone ship)' AND DATE BETWEEN '2015-01-01' AND '2015-12-31';
```

	Booster_Version	Launch_Site	Landing_Outcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

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

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT "Landing_Outcome", COUNT("Landing_Outcome") FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY "Landing_Outcome" ORDER BY COUNT("Landing_Outcome") DESC;
```

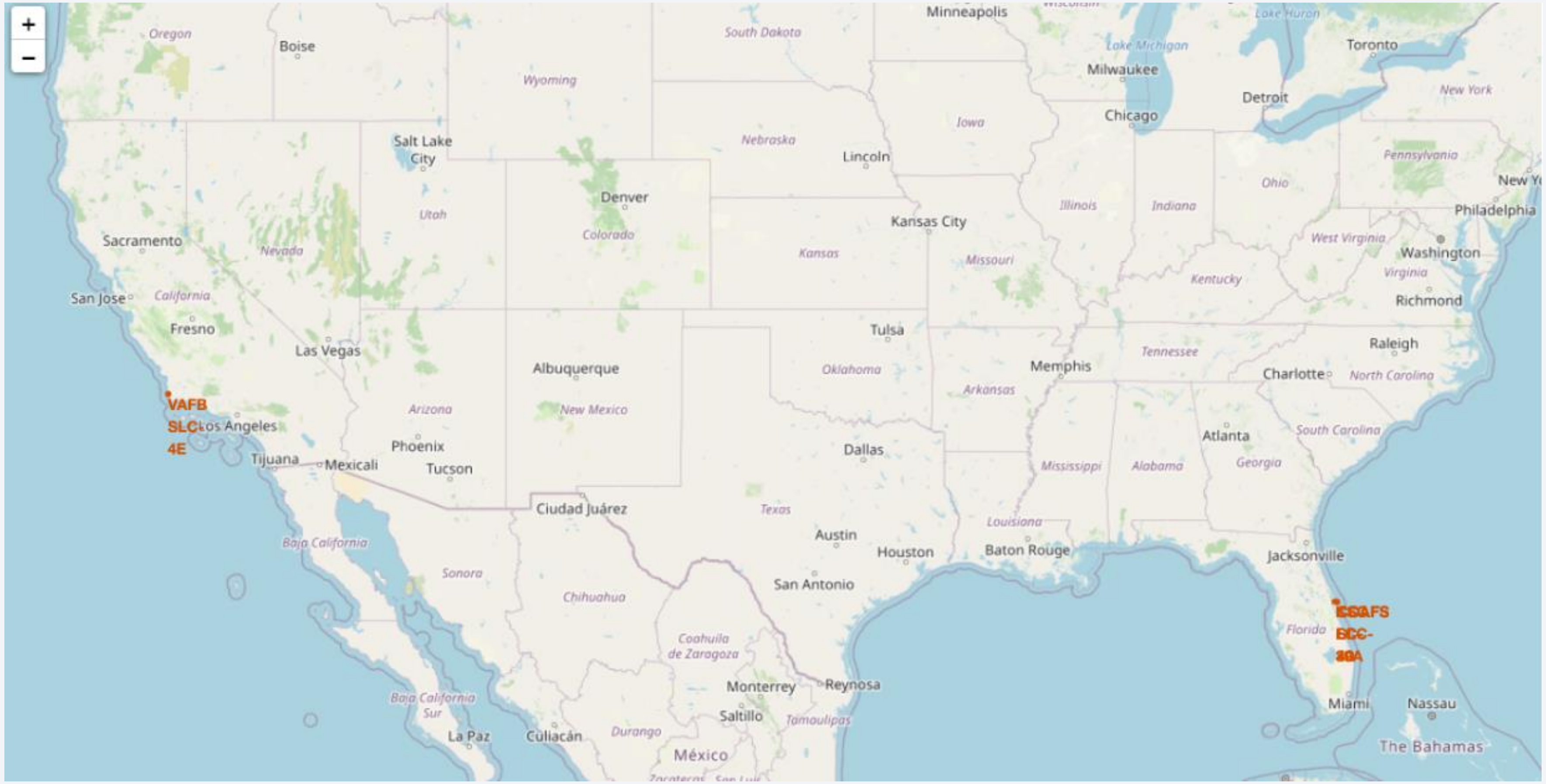
	"Landing_Outcome"	COUNT("Landing_Outcome")
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Section 3

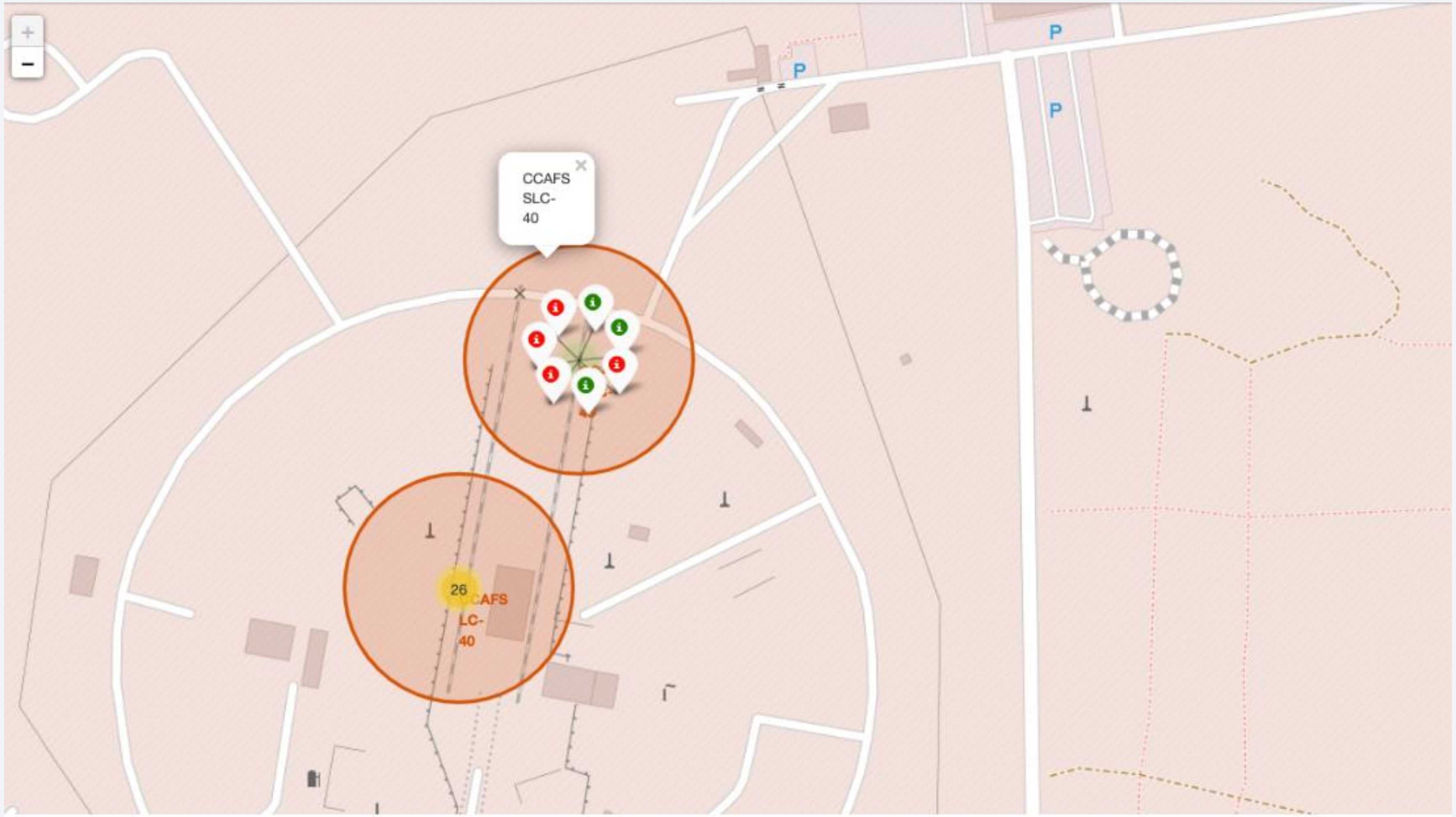
Launch Sites Proximities Analysis

Map with marked launch sites

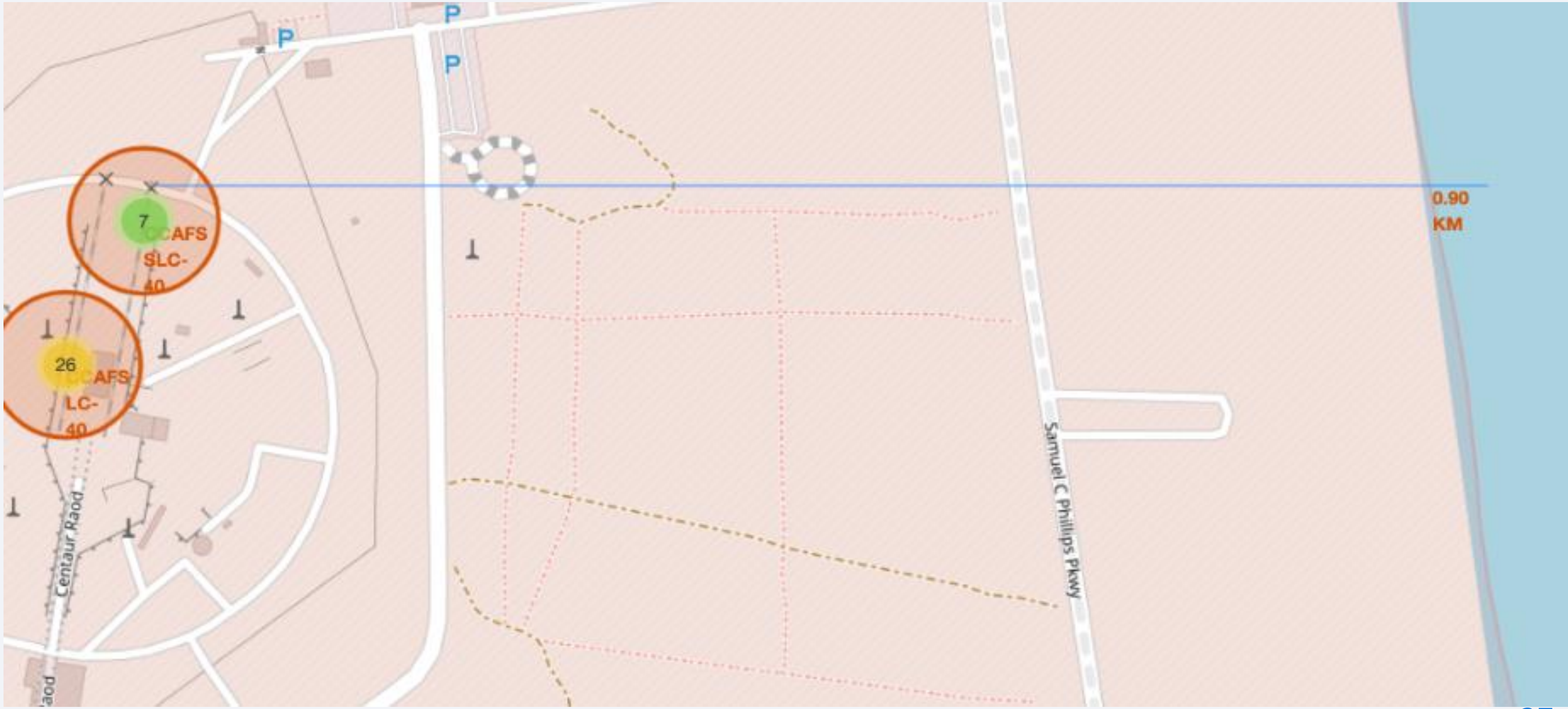
SpaceX launch sites are located in Florida and California of United States



Launch sites markers



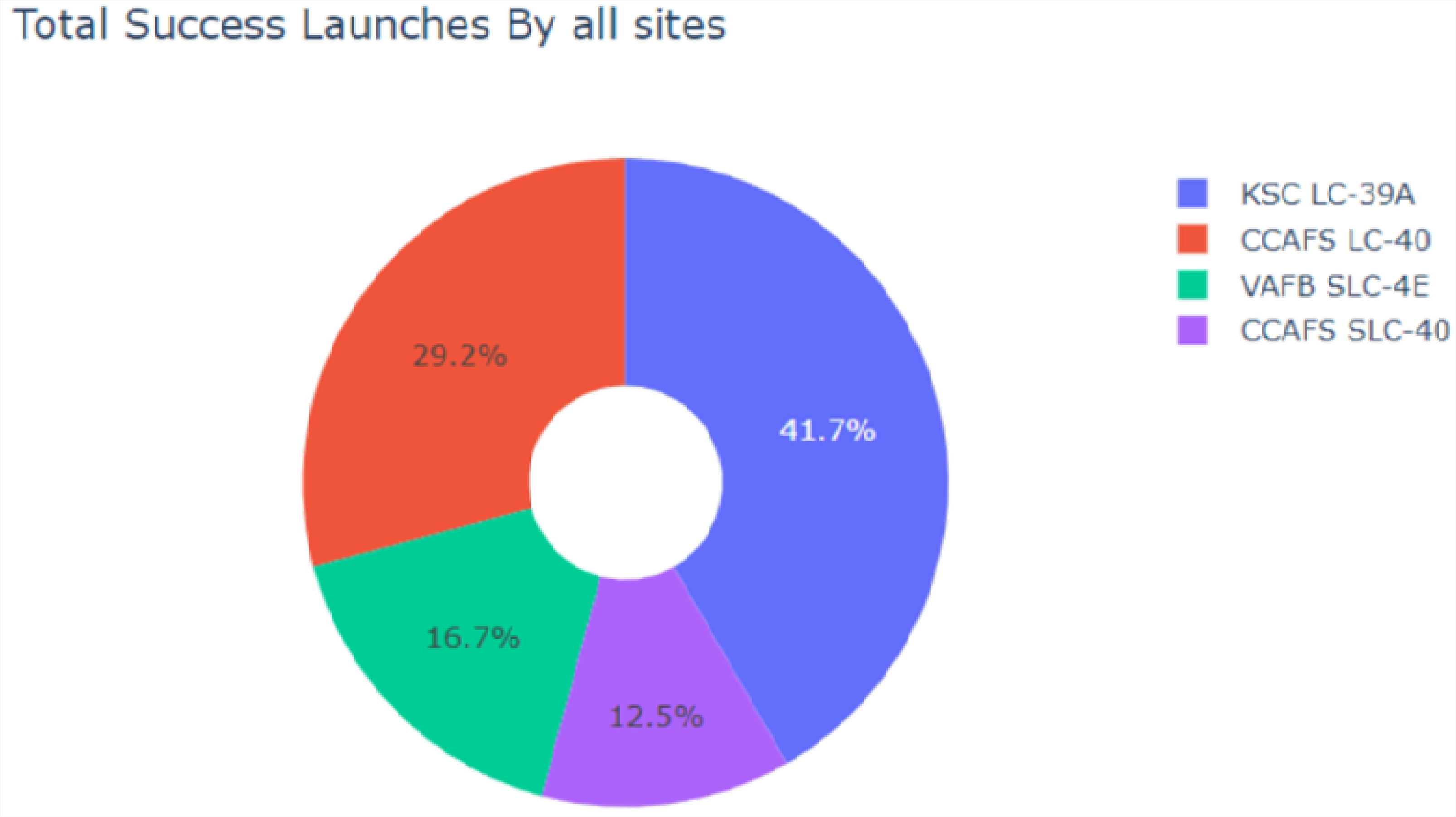
Launch Site Distance to Coastline Point



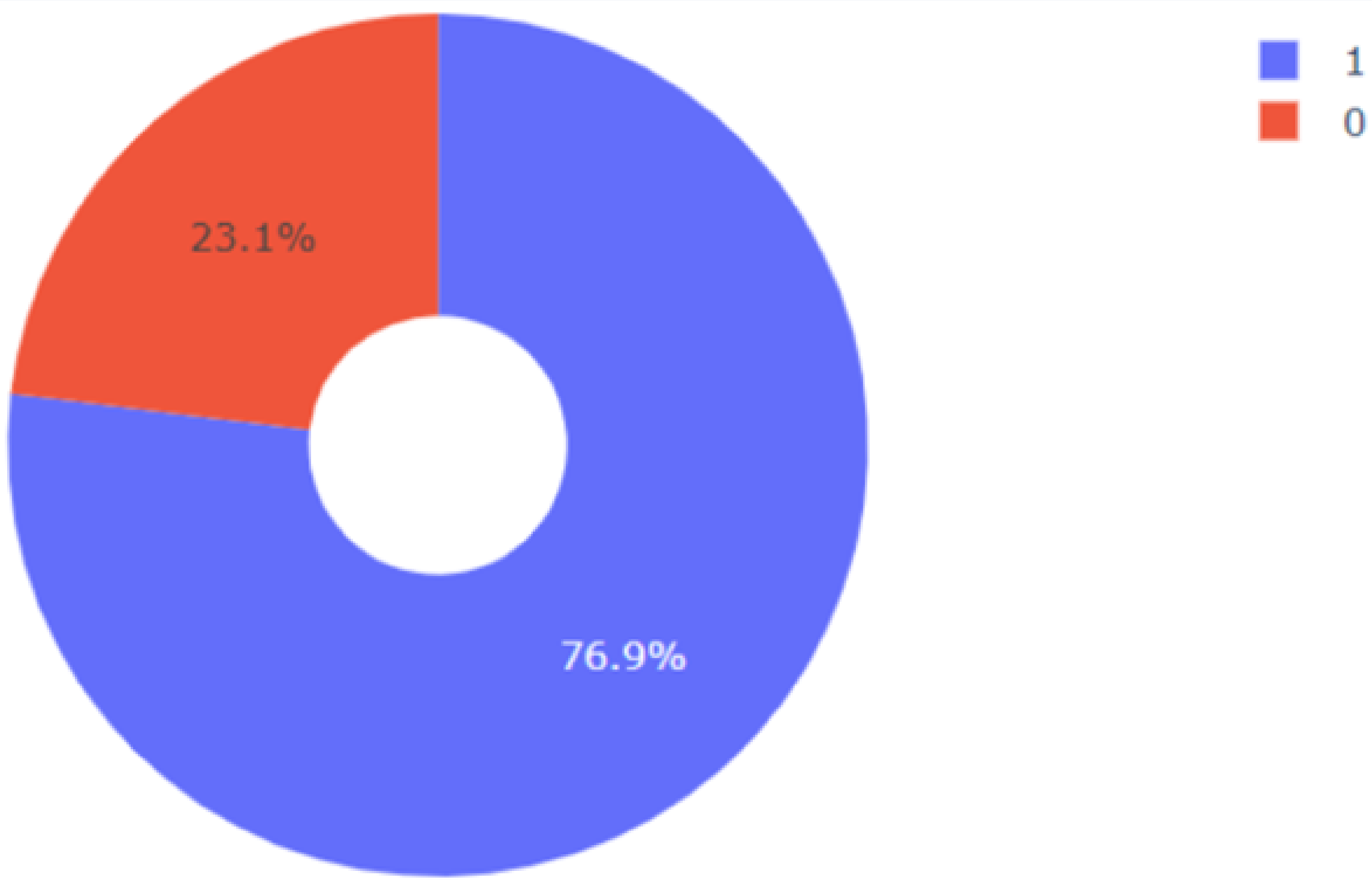
Section 4

Build a Dashboard with Plotly Dash

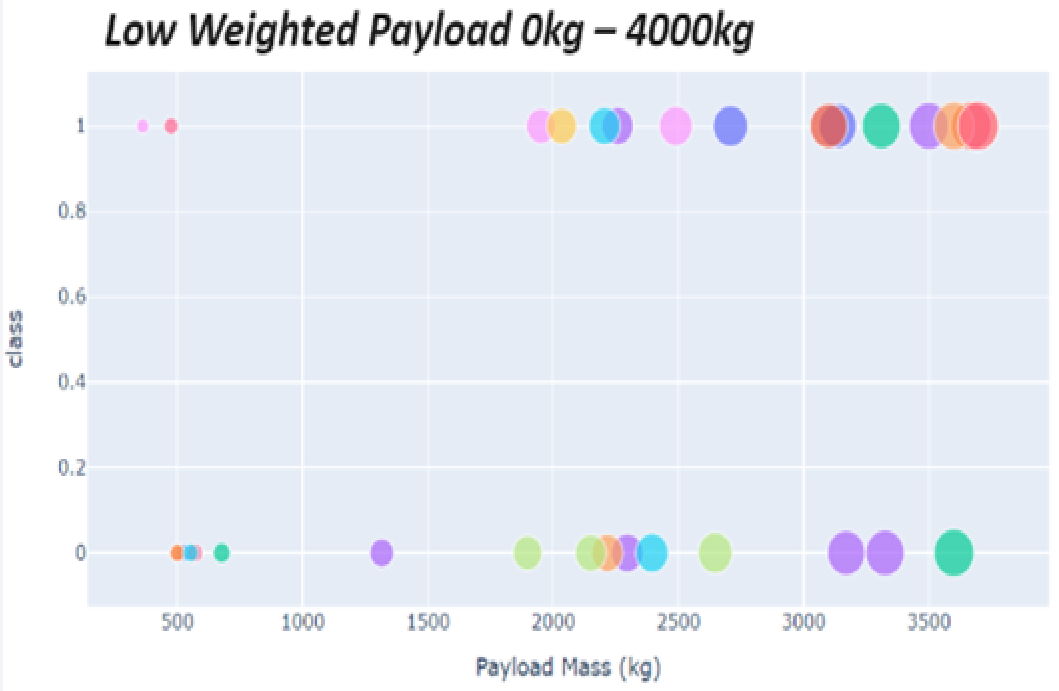
Launch Site Success Percentage

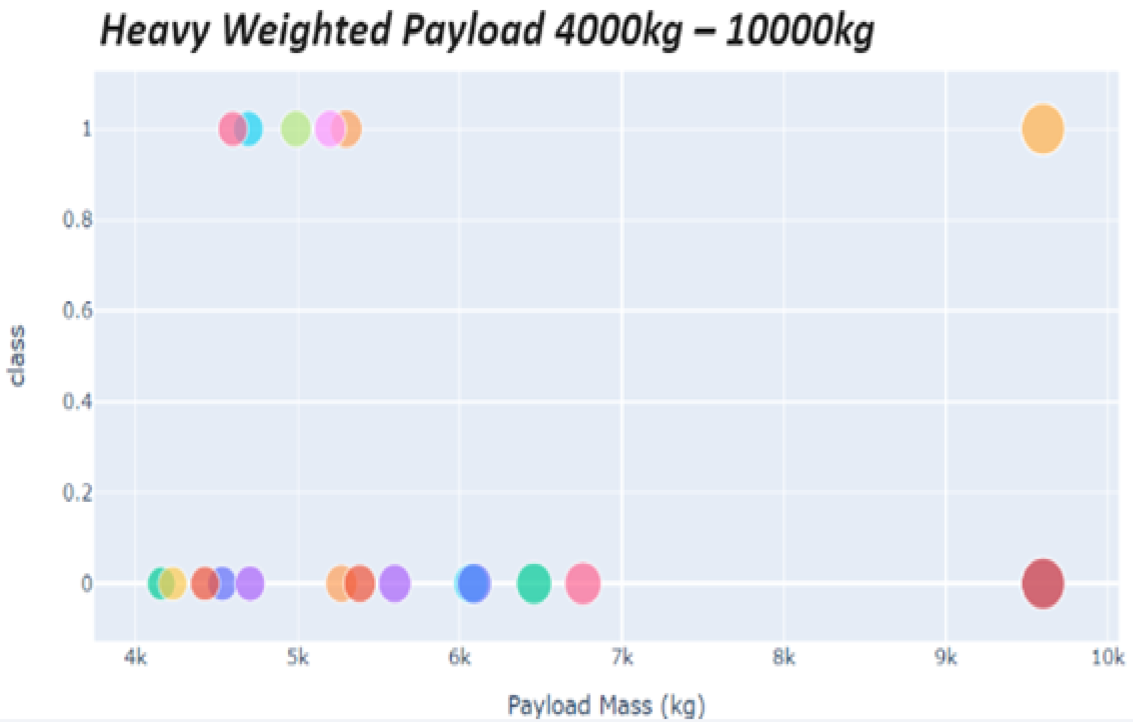


Launch Success Ratio



Payload vs Launch Outcome





Section 5

Predictive Analysis (Classification)

Classification Accuracy

Decision Tree Classifier is the best model
with highest accuracy score tha the other models

```
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier()
```

```
tree_cv = GridSearchCV(tree, parameters, cv=10)
tree_cv.fit(X_train, Y_train)
```

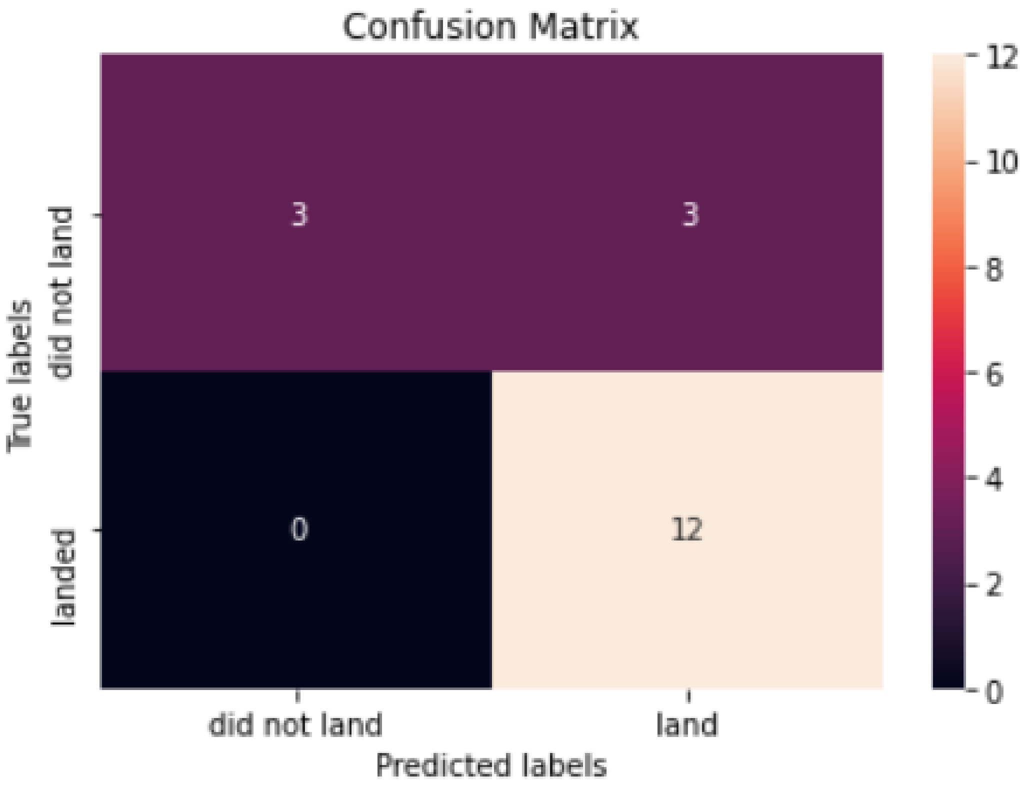
```
GridSearchCV(cv=10, error_score='raise-deprecating',
       estimator=DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best'),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'criterion': ['gini', 'entropy'], 'splitter': ['best', 'random'], 'max_depth': [2, 4, 6, 8, 10, 12, 14, 16, 18], 'max_features': ['auto', 'sqrt'], 'min_samples_leaf': [1, 2, 4], 'min_samples_split': [2, 5, 10]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
```

```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
accuracy : 0.8888888888888888
```

Confusion Matrix

```
yhat = svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Decision Tree Confusion Matrix of the best performing model.

Conclusions

The larger the flight amount at a launch site, the greater the success rate

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 launch site.

The Decision tree classifier is the best machine learning algorithm.

Appendix

All Scripts are stored in the GitHub Repo:

https://github.com/jmsxngl/IBM-Data-Science.git

Thank you!