Winning Space Race with Data Science

Sandro Hörler
1st February 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection with Web Crawling and over API
  - Data Wrangling / Cleaning
  - Exploratory Data Analysis (EDA)
    - With SQL
    - With Data Visualisation
  - Spacial Analytics with Folium
  - Exploratory Data Anala
  - Predictive Modelling

- Summary of all results
  - Exploratory Data Analysis (EDA) results
  - Predicitive Analytics results

Introduction

- Project background and context

  Space X promotes Falcon 9 rocket launches on its website with a price tag of $62 million. Other providers charge a minimum of $165 million per launch. The significant cost savings by Space X is due to its ability to reuse the first stage. Therefore, the success of the first stage landing directly impacts the cost of a launch. This information can be utilized by competing companies to bid against Space X for rocket launches. The aim of the project is to develop a machine learning pipeline that can predict the successful landing of the first stage.

- Problems you want to find answers

  - What launch Data can bring information gain to evaluate if a ticket will land successfully?

  - The inter-correlation between features between different features that define the success rate of a successful landing.

  - What operating conditions need to be in place to ensure a achieved 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
- 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

- Data has been collected by calling the open SpaceX API and Scraping the Falcon9 launch records on Wikipedia.

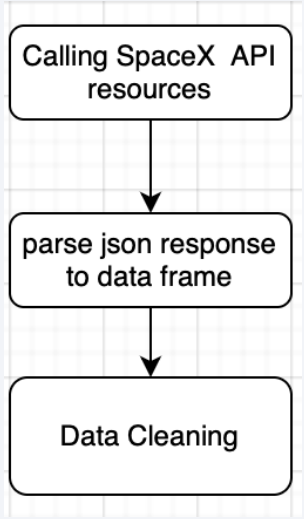
  - Extracted HTML table data with BeautifulSoup library, data cleaning has been applied and then converted to a pandas dataframe.

  - Parsed the called SpaceX API data from json to a pandas dataframe including cleaning of the raw data given by the API.

Data Collection – SpaceX API

- Called the SpaceX API over a simple GET request with requests library, normalised and parsed the response json to a dataframe.

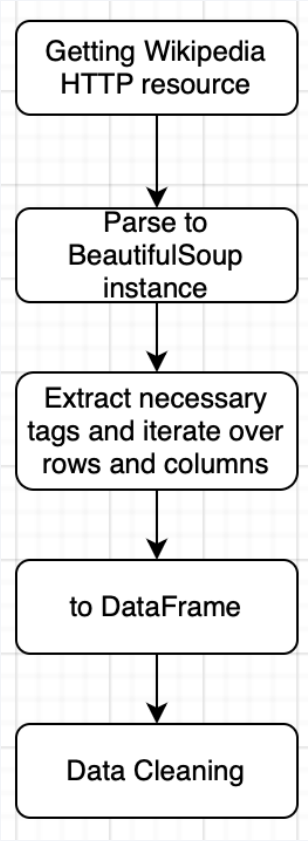
- Gihub Notebook: https://github.com/sandrohoerler/coursera_ibm_datascience/blob/master/jupyter-labs-spacex-data-collection-api.ipynb.ipynb



Data Collection - Scraping

- Called the Wikipedia Falcon9 launch Page with a GET request and requests library.
  - Parsed DOM tree to BeautifulSoup Object
  - Extracted necessary tags from tree and iterated over rows
  - Conversion to DataFrame

- Github Notebook: https://github.com/sandrohoerler/coursera_ibm_datascience/blob/master/jupyter-labs-spacex-data-collection-web-scraping.ipynb



Data Wrangling

- Exploratory Data Analysis (EDA) has been done
  - Data preprocessing, cleaning and interpolation
  - Correlation Analysis of potential features with target variable
  - Feature Engineering

- https://github.com/sandrohoerler/coursera_ibm_datascience/blob/master/jupyter-labs-spacex-data-wrangeling.ipynb

EDA with Data Visualization

- Exploration of relationship between several potential features like flight number, launch Site, Payload and as well as success rate of all orbit types
- Checked for relations only visible in one if plotted, like the yearly launch success trend

- https://github.com/sandrohoerler/coursera_ibm_datascience/blob/master/jupyter-labs-spacex-data-viz.ipynb

EDA with SQL

- Loading of the SpaceX dataset into a PostgreSQL database
- Application of EDA with SQL to find insights contained in the data
  - unique launch site names in the space mission
  - total payload mass carried by boosters launched by NASA (CRS)
  - average payload mass carried by booster version F9 v1.1
  - failed landing outcomes in drone ship, their booster version and launch site names
  - total number of successful and failure mission outcomes
- https://github.com/sandrohoerler/coursera_ibm_datascience/blob/master/jupyter-labs-spacex-eda-sql.ipynb

Build an Interactive Map with Folium

- Marking of all launch sites and adding of map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- Assignment of the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Added color-labeled marker clusters, been identified which launch sites have relatively high success rate.
- Calculation of distances between a launch site and its proximities.
    - Launch Sites near Railways, highways or coastlines
    - Certain distance away from cities

- https://github.com/sandrohoerler/coursera_ibm_datascience/blob/master/jupyter-labs-spacex-viz-folium.ipynb

Build a Dashboard with Plotly Dash

- Building of an interactive dashboard with Plotly Dash
- Plotting of pie charts with total launches for each city
- Plotting of scatter plots explaining the relationship with Outcome and Payload Mass for unique booster versions

- https://github.com/sandrohoerler/coursera_ibm_datascience/blob/master/app.py

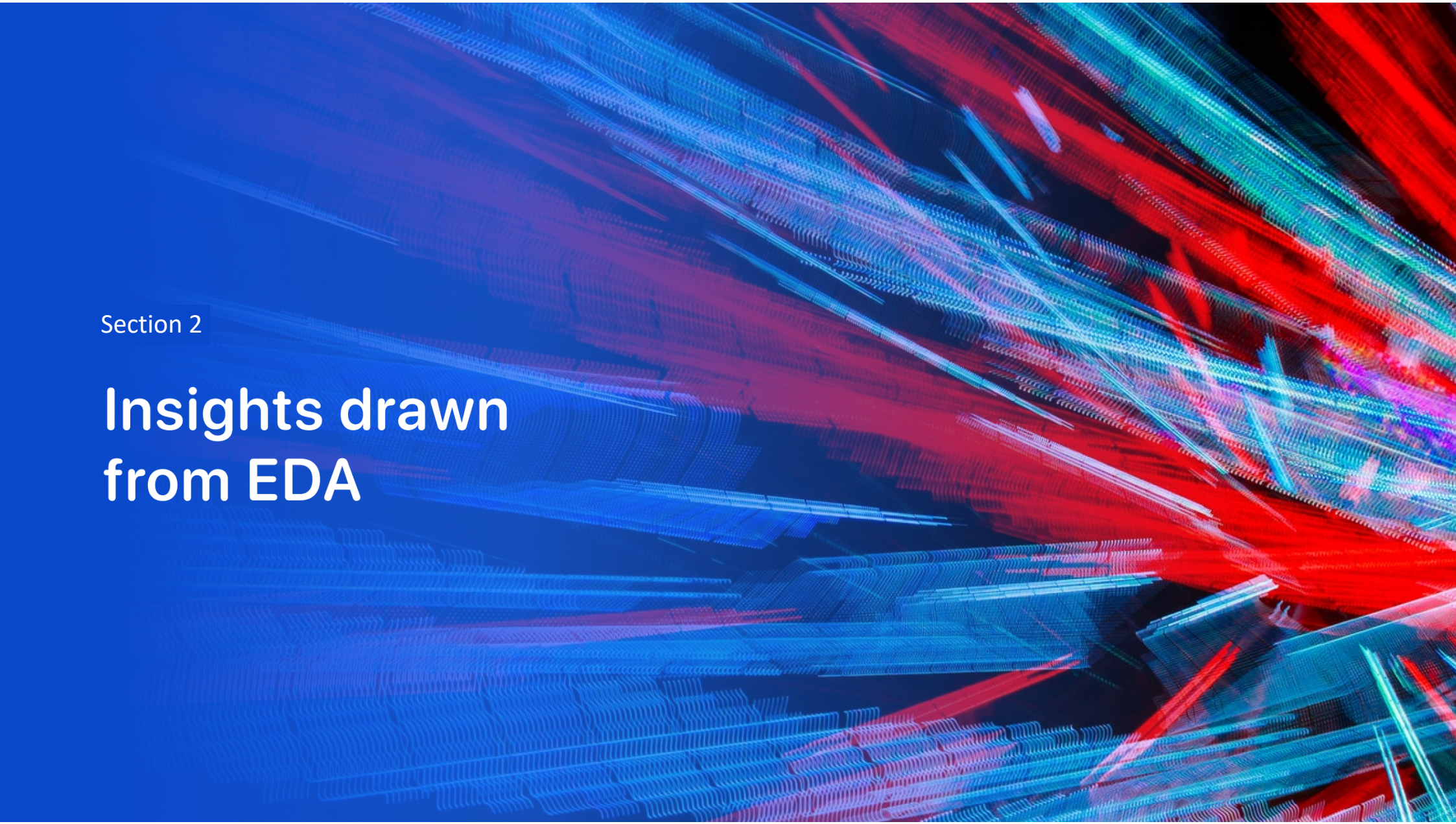
Predictive Analysis (Classification)

- Losing of the data using numpy and pandas, transformed the data, split our data into training and testing.
- Building of different machine learning models and tune different hyperparameters using GridSearchCV.
- Usage of accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- Finding the best performing classification model.

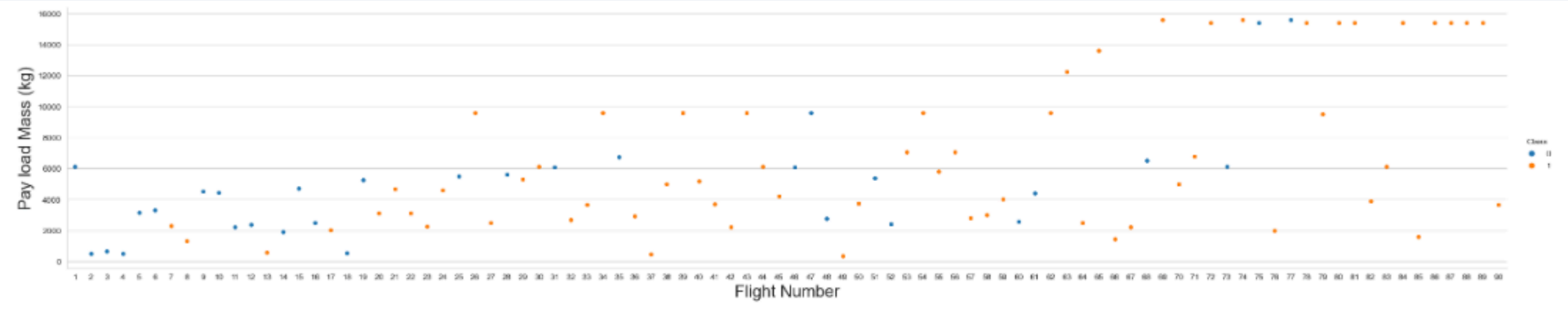
- https://github.com/sandrohoerler/coursera_ibm_datascience/blob/master/jupyter-labs-spacex-machine-learning-pred.ipynb

Results

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


Section 2
Insights drawn from EDA

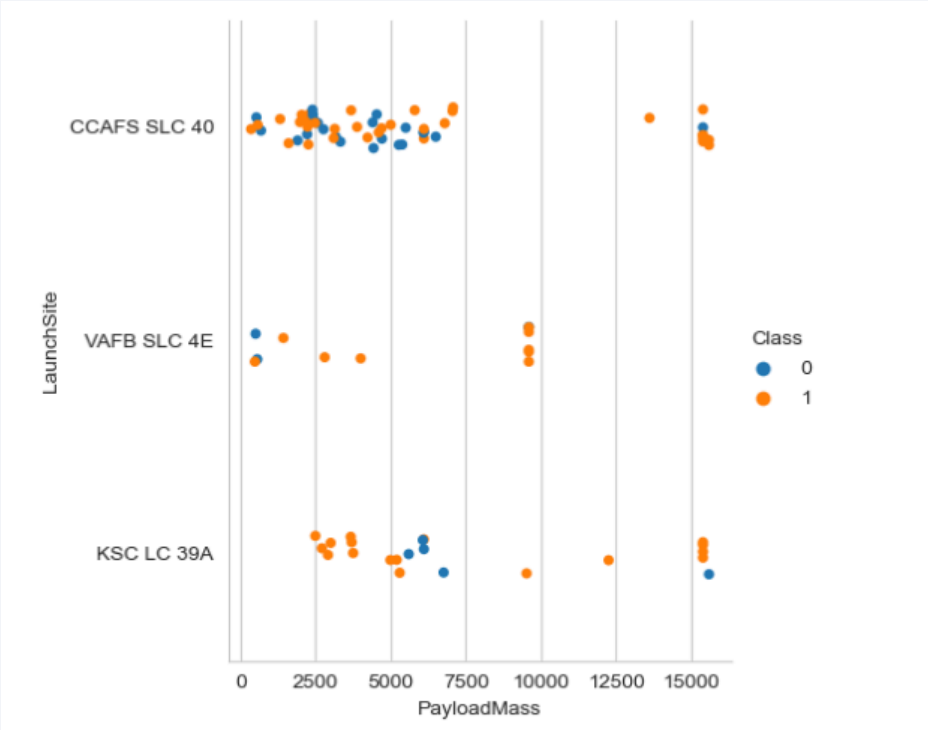
Flight Number vs. Payload Mass



- The plot shows that the the rockets launched with a high flight number carry more Load than in the lower flight numbers

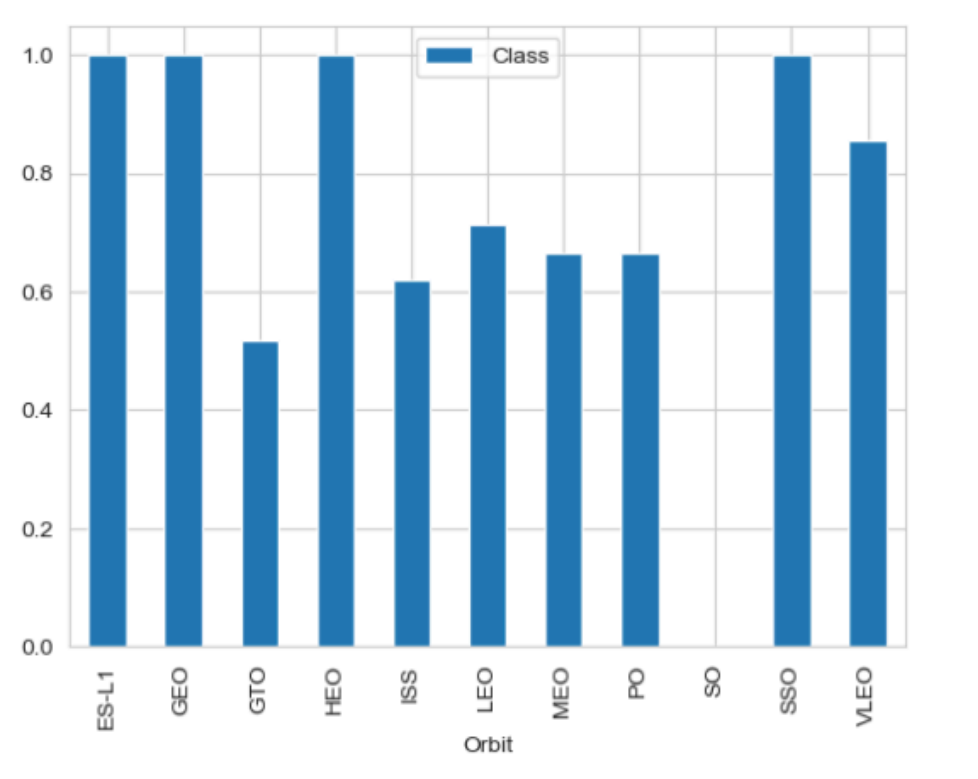
Payload vs. Launch Site

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

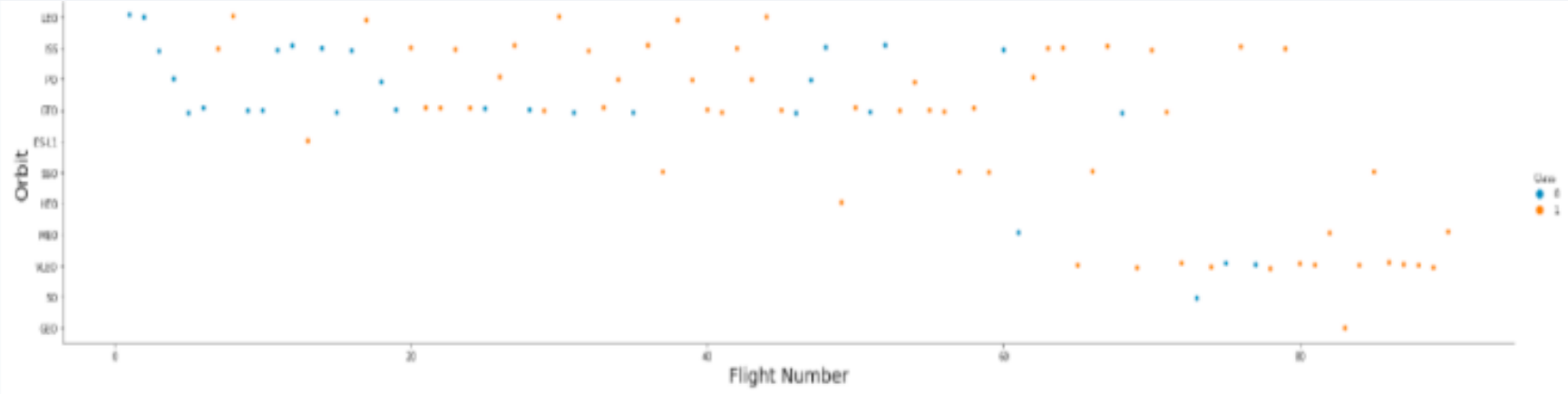


Success Rate vs. Orbit Type

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



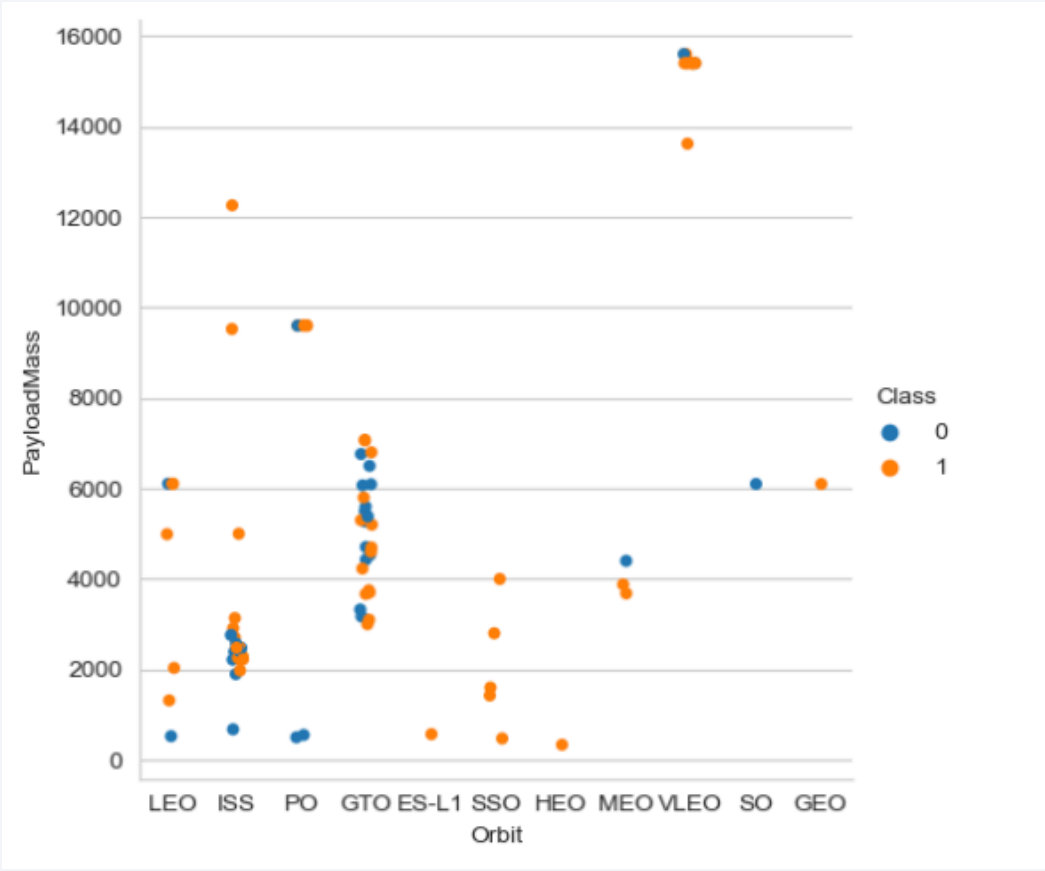
Flight Number vs. Orbit Type



- The plot below shows the Flight Number vs. Orbit type. It has been observed 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.

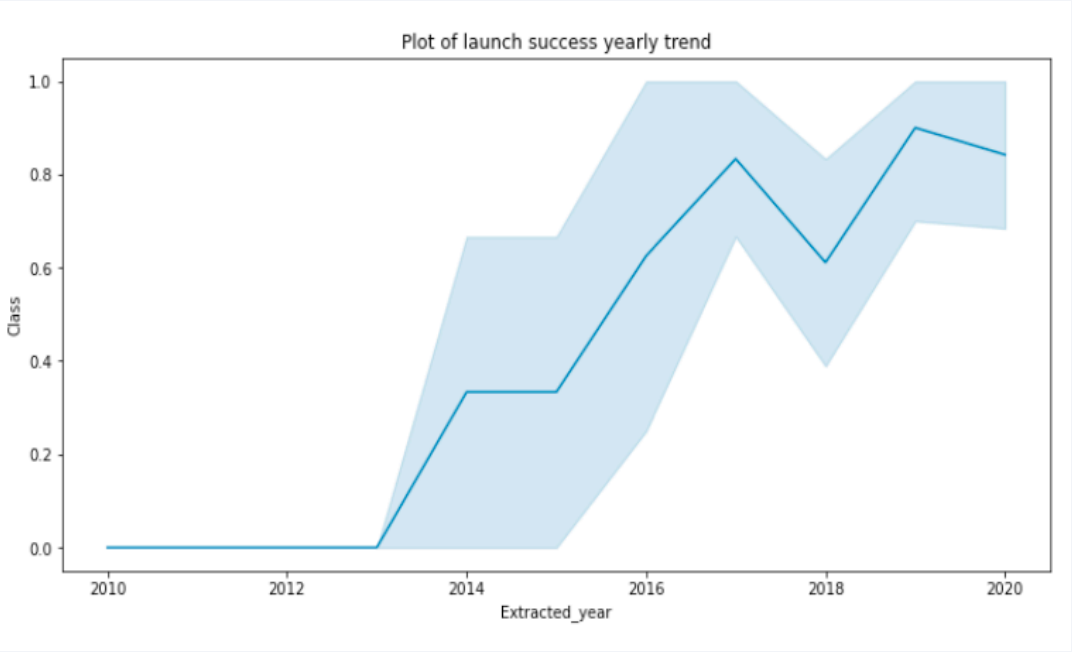
Payload vs. Orbit Type

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



Launch Success Yearly Trend

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



All Launch Site Names

- Distinct is the way to go =)
- In pandas that would be unique()

```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

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

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

* Writing of the query above to display 5 records where launch sites begin with `CCA`

Total Payload Mass

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

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

- Calculation of the total payload carried by boosters from NASA as 45596 using the query below

Average Payload Mass by F9 v1.1

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

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

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015Present your query result with a short explanation here

In [14]:
```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Usage of 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

In [15]:
```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
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

- Usage of wildcard like '%' to filter for WHERE MissionOutcome was a success or a failure.

List the total number of successful and failure mission outcomes

In [16]:
```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

Out[16]:

The total number of successful mission outcome is:

	successoutcome
0	100

The total number of failed mission outcome is:

	failureoutcome
0	1

Boosters Carried Maximum Payload

- 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 [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

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

In [18]:

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- Usage of combinations of the WHERE clause, LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

Rank Landing Outcomes

- 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

- After aggregation by LandingOutcome one can calculate the count in this group and order the results by this count in descending order

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

In [19]:
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

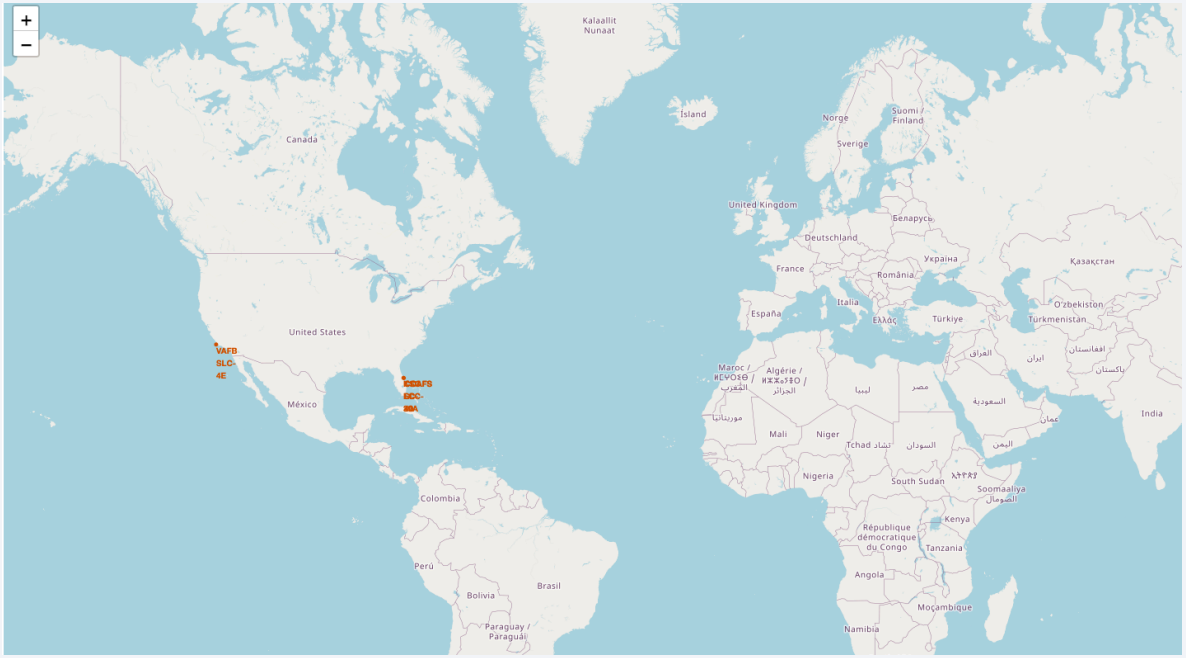
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Section 3

Launch Sites Proximities Analysis

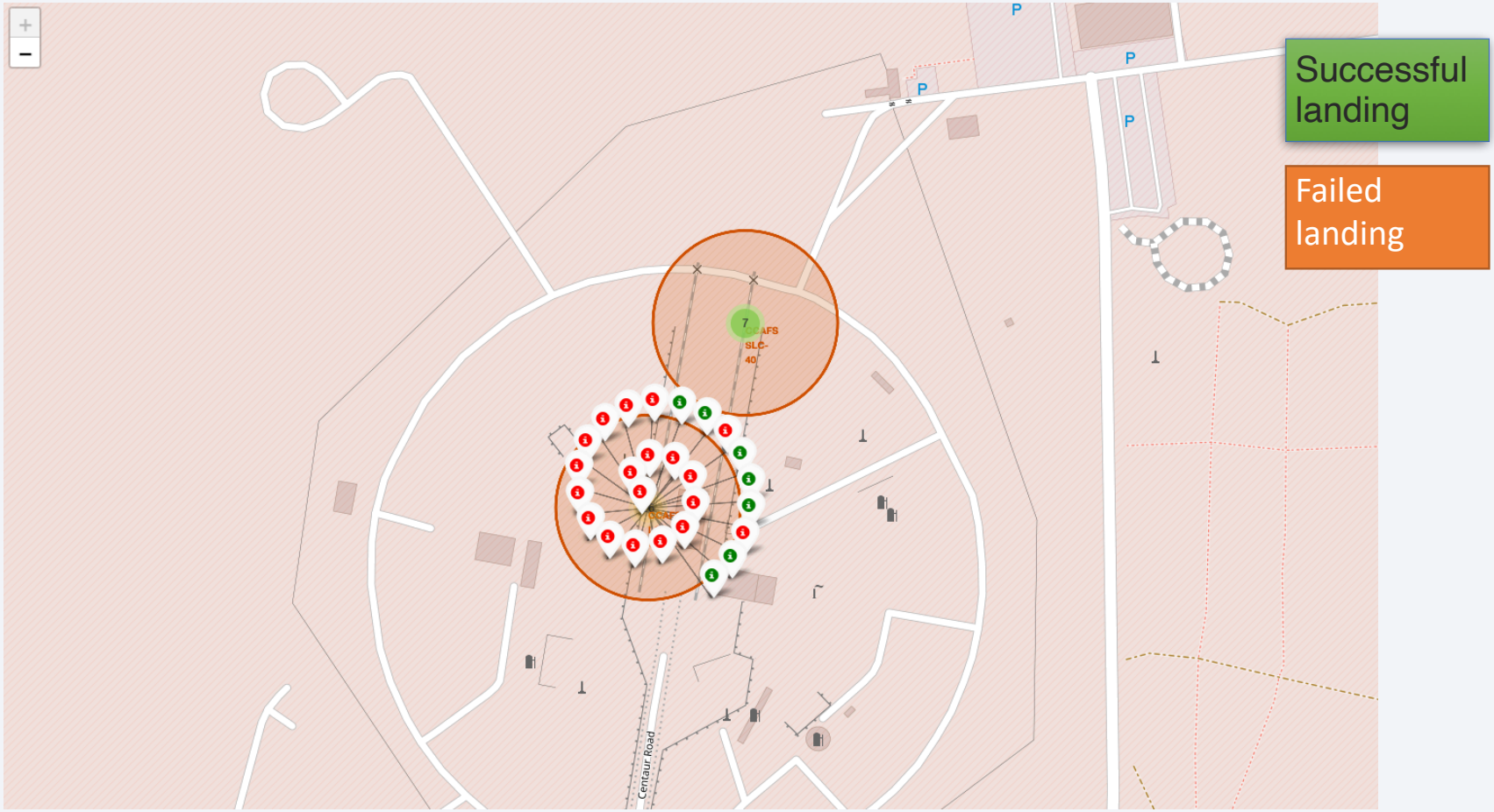
Global launch Sites

One can see that the launch Sites reside in USA at the coasts of Florida and California.

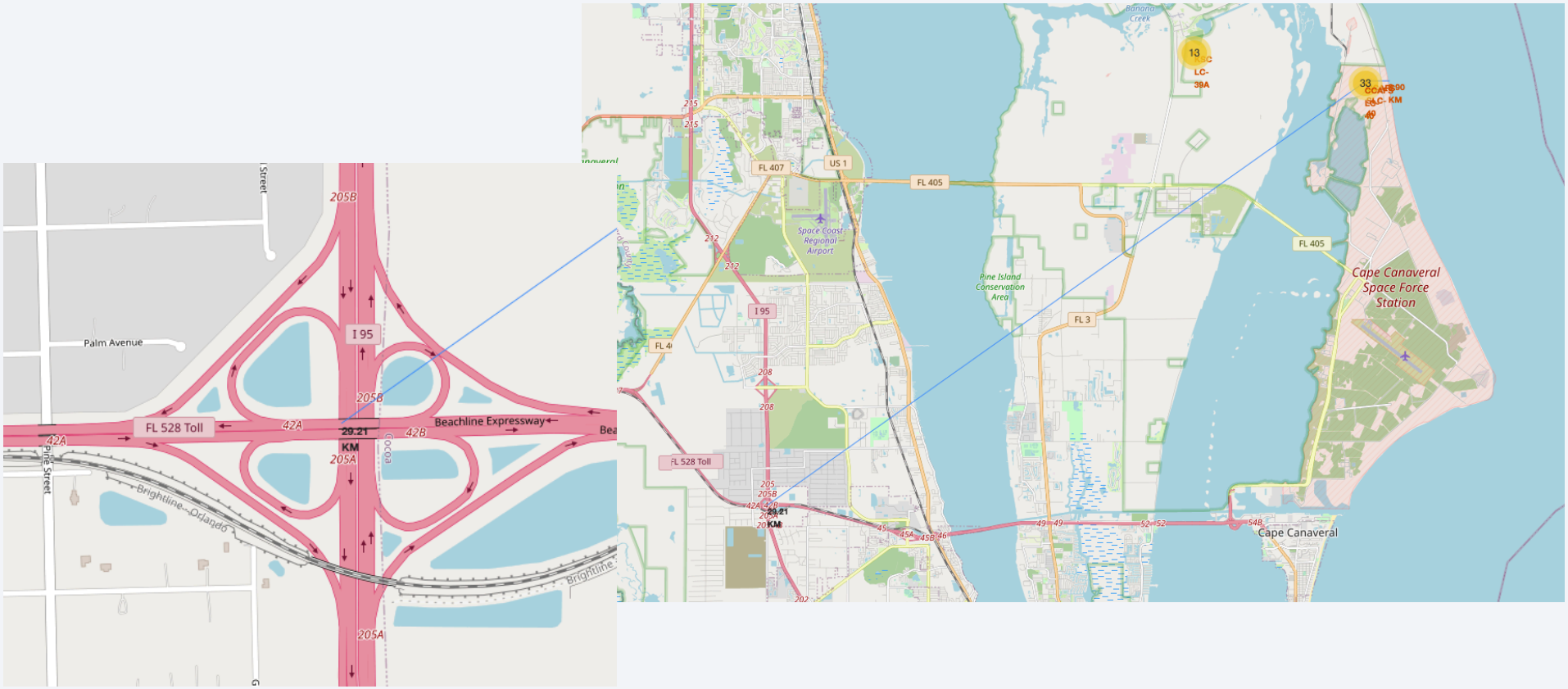


- Explore the generated folium map and make a proper screenshot to include all launch sites' location markers on a global map

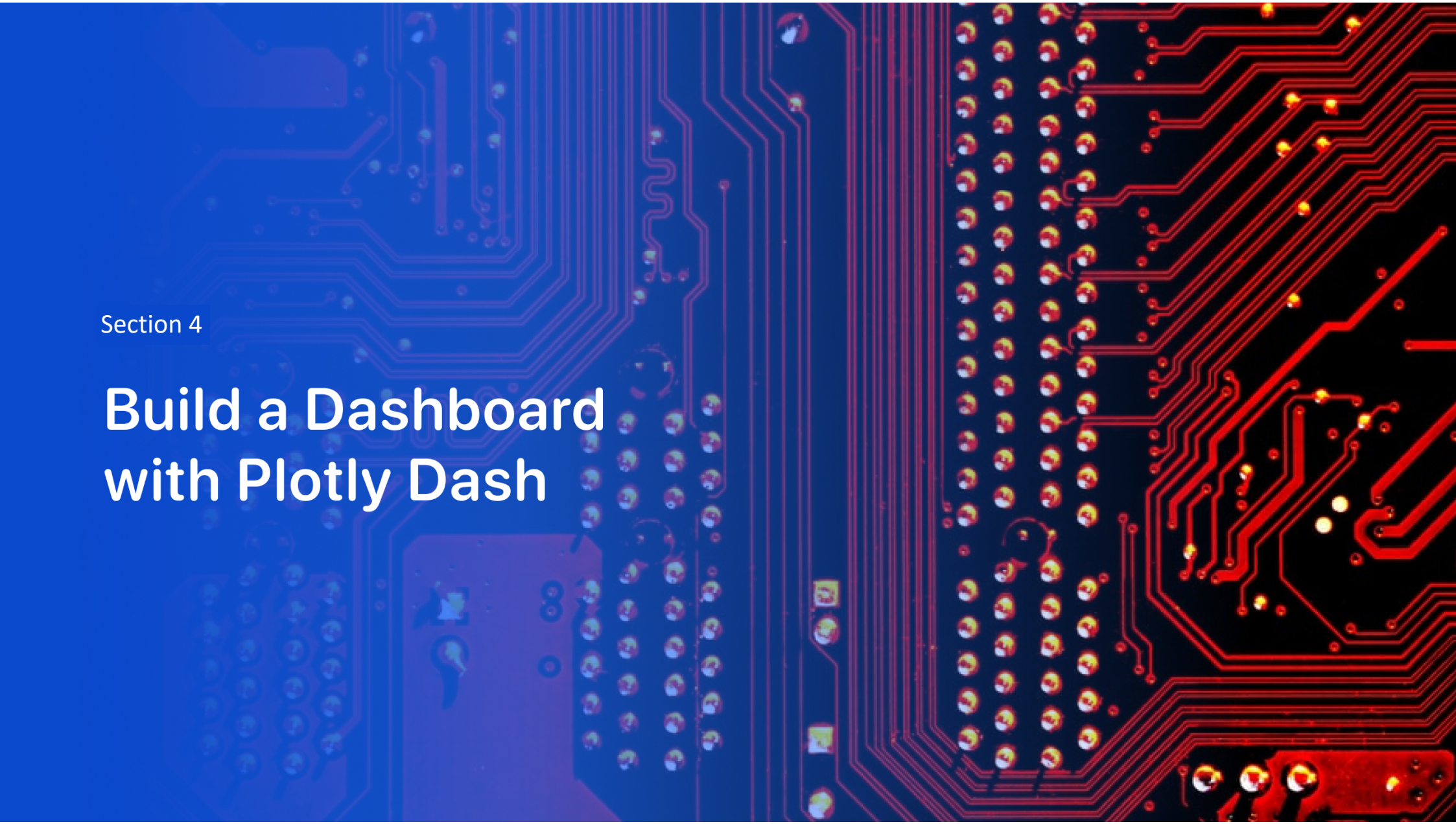
Markers indicating success or failure of launches



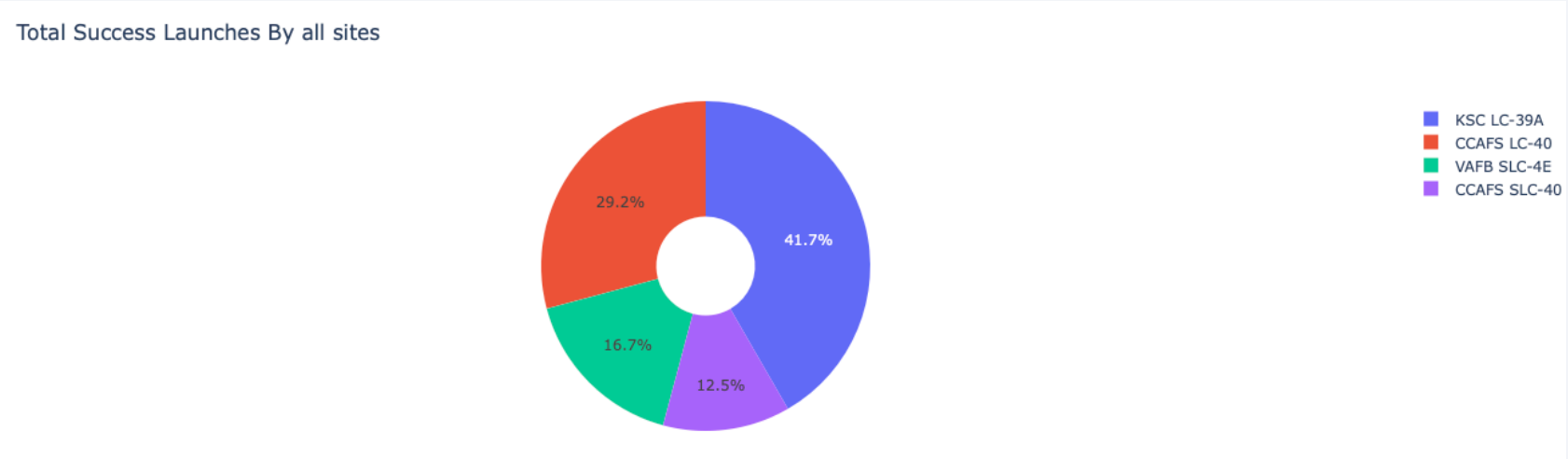
Site distance to potential inferences



- Distance to proximities indicate potential inference sources

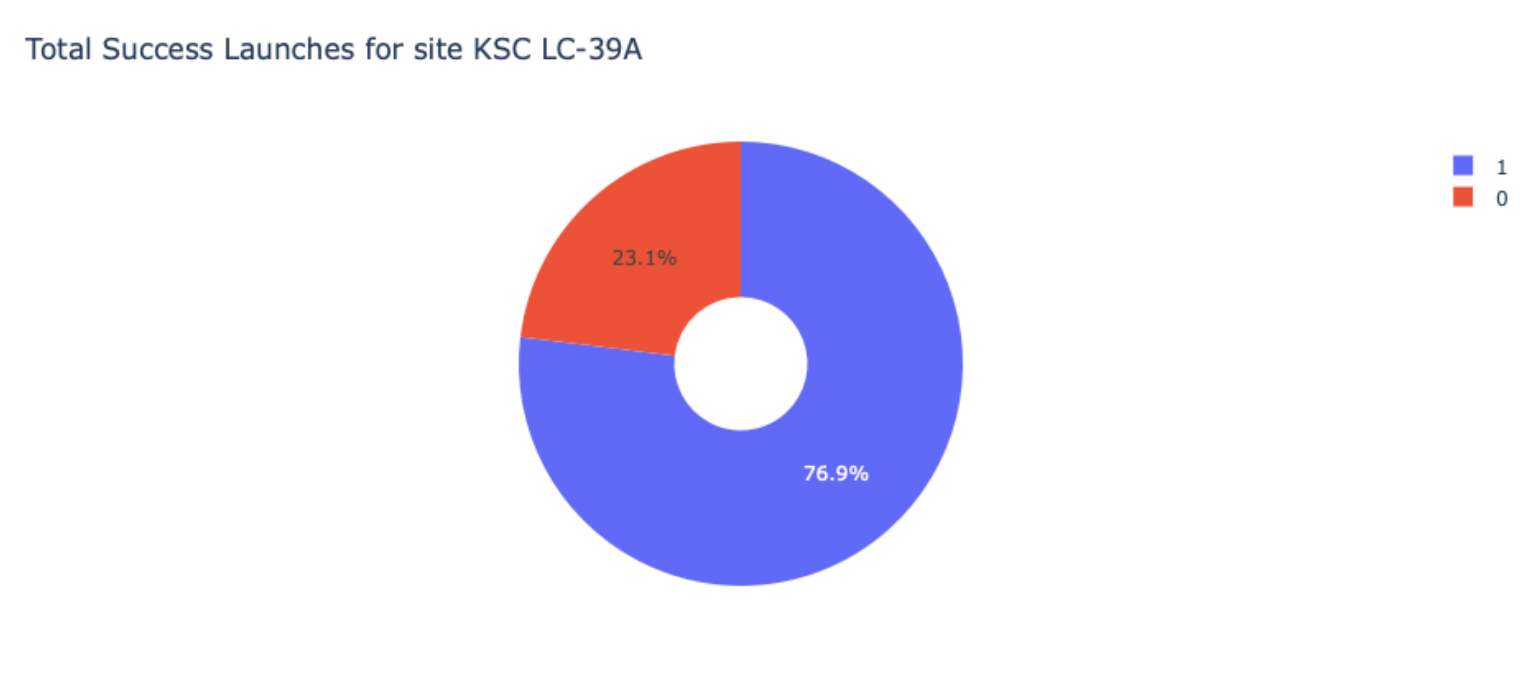

Section 4
Build a Dashboard with Plotly Dash

Success rate by launch Site



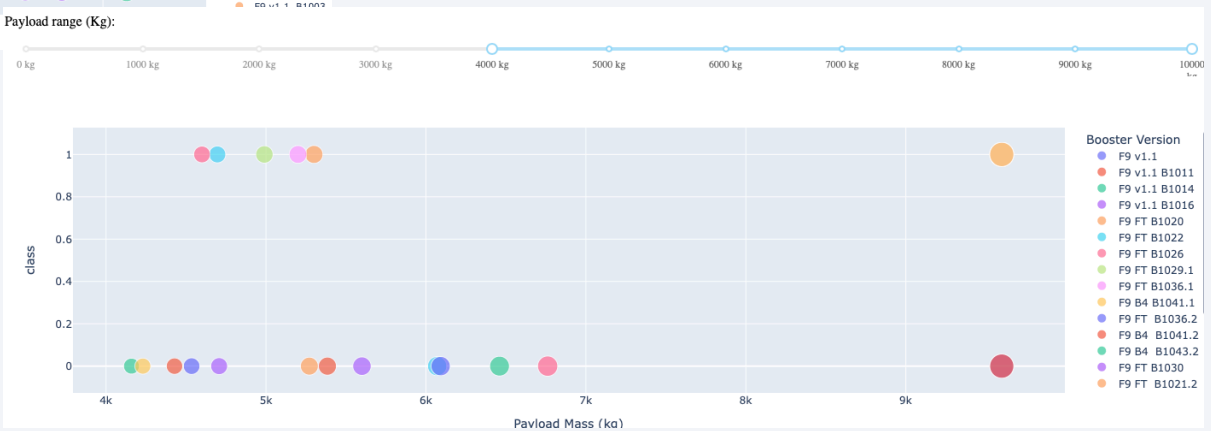
- KSC LC-39A has the most successful launches

Dive into launches of most successful site



- KSC LC-39A has a success ratio of 76.9% to 23.1%

Payload vs Launch Outcome



- The success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

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_)
```

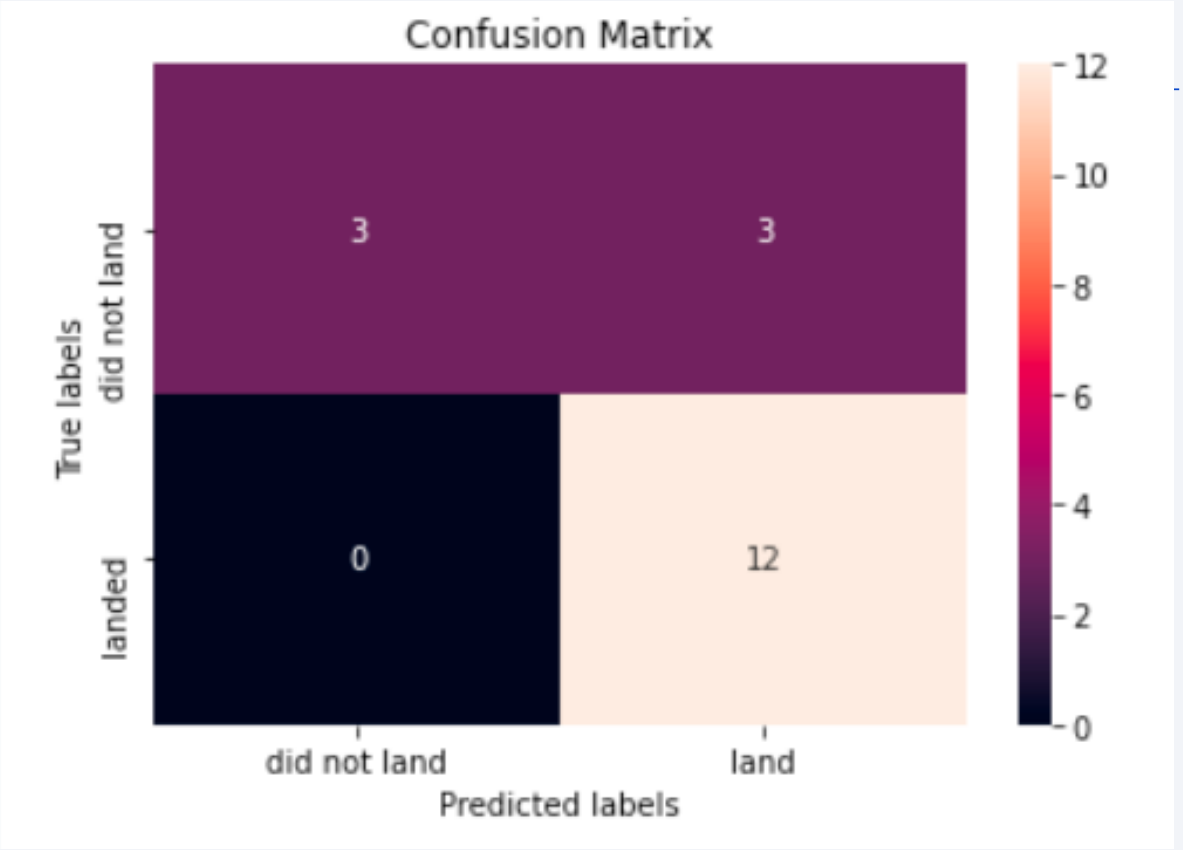
Best model is DecisionTree with a score of 0.873

Chosen Params

- criterion: gini
- max_depth: 6
- max_features: auto
- min_samples_leaf: 2
- min_samples_split: 5
- splitter: random

Confusion Matrix

- In this case, false positives are the biggest problem. As they indicate that a predicted launch outcome was success, but it was originally a failure.
- Shifting the class boarder towards success can be used to address this issue.



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 Decision tree classifier is the best machine learning algorithm for this task.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!