IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Fiyinfoluwa Oloyede
March, 2023

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

  The commercial space age is here, companies like Virgin Galactic , Rocket Lab and Blue origin among others are making space travel affordable for everyone. Perhaps the most successful is SpaceX. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards of 165 million dollars each, much of the savings is because SpaceX 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 by other rocket providers that are in competition with space X for a rocket launch. The aim of this project is to train 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 was collected using Space X API and web scrapping 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 data frame for subsequent 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.

- GitHub URL: https://github.com/FiyinOloyede/IBM_Data_Science_SpaceX_Landing_Prediction/blob/main/Data%20collection%20API.ipynb

```
[12]: 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

```
[13]: response.status_code
```

```
[13]: 200
```

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

```
[14]: # Use json_normalize meethod to convert the json result into a dataframe
data = response.json()
data = pd.json_normalize(data)
```

Using the dataframe `data` print the first 5 rows

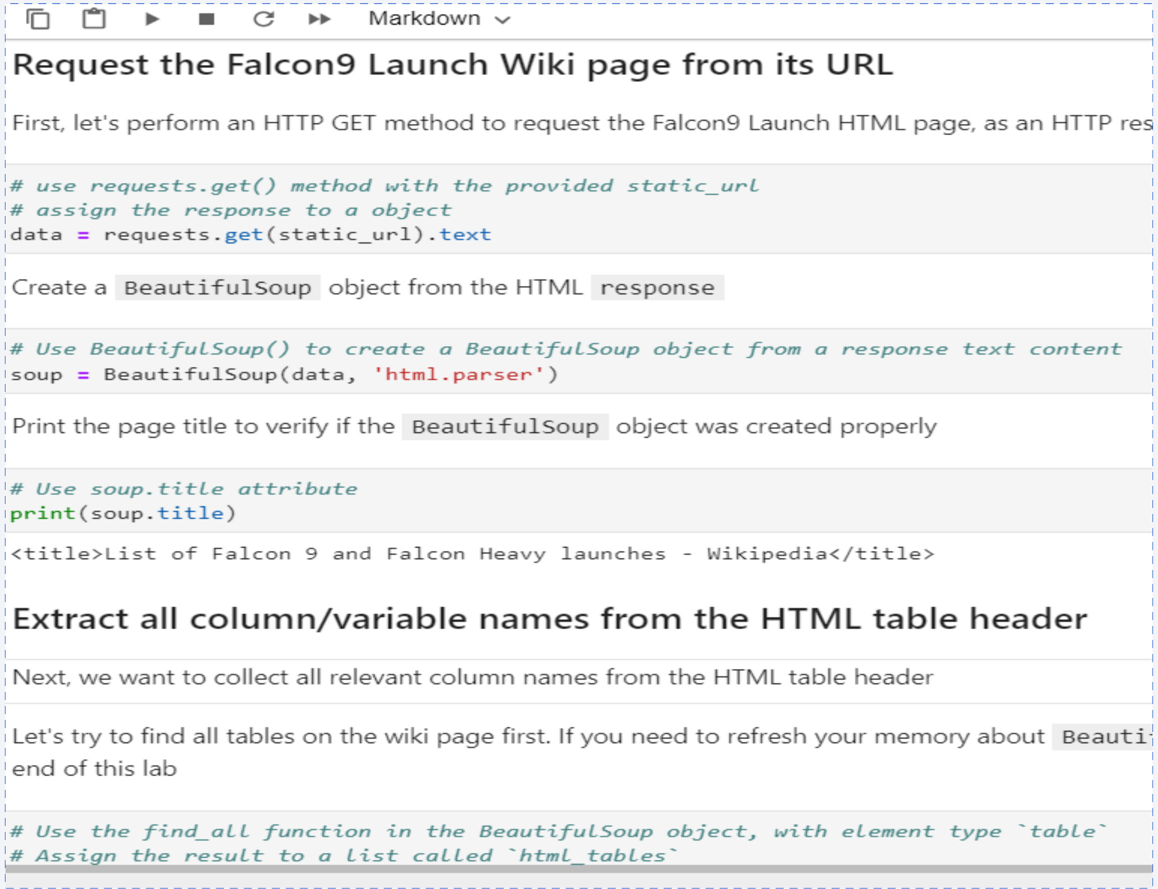
```
[15]: # Get the head of the dataframe
data.head()
```

[15]:

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules	
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]	[5eb0e4b5b6c3bb

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts
- GitHub URL: https://github.com/FiyinOloyede/IBM_Data_Science_SpaceX_Landing_Prediction/blob/main/Data%20collection%20with%20web%20scraping.ipynb



Data Wrangling

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is: https://github.com/FiyinOloyede/IBM_Data_Science_SpaceX_Landing_Prediction/blob/main/Data%20wrangling.ipynb

Markdown

Calculate the number and occurrence of each orbit

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

Calculate the number and occurence of mission outcome per orbit t

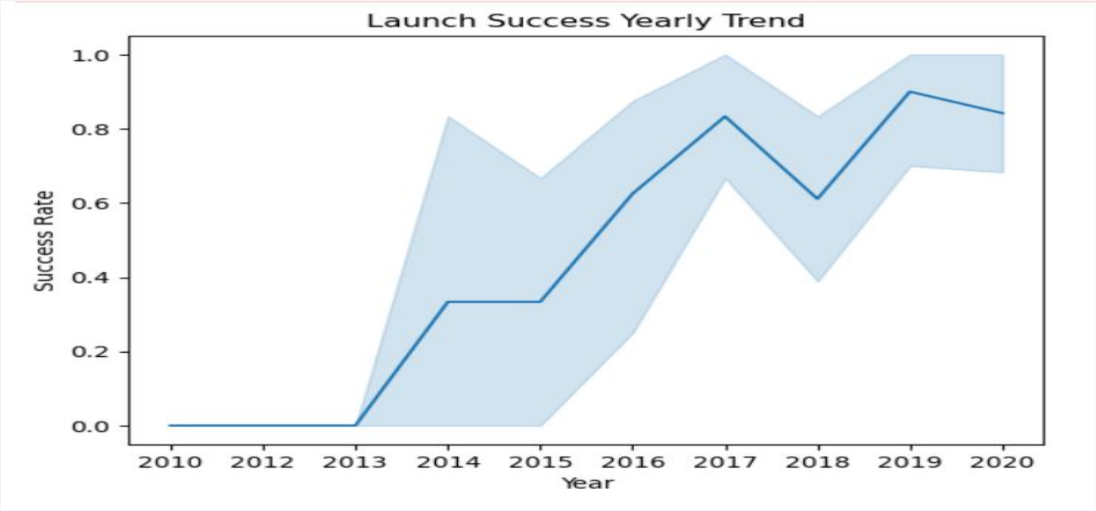
Use the method `.value_counts()` on the column `Outcome` to determine the number of `landi`

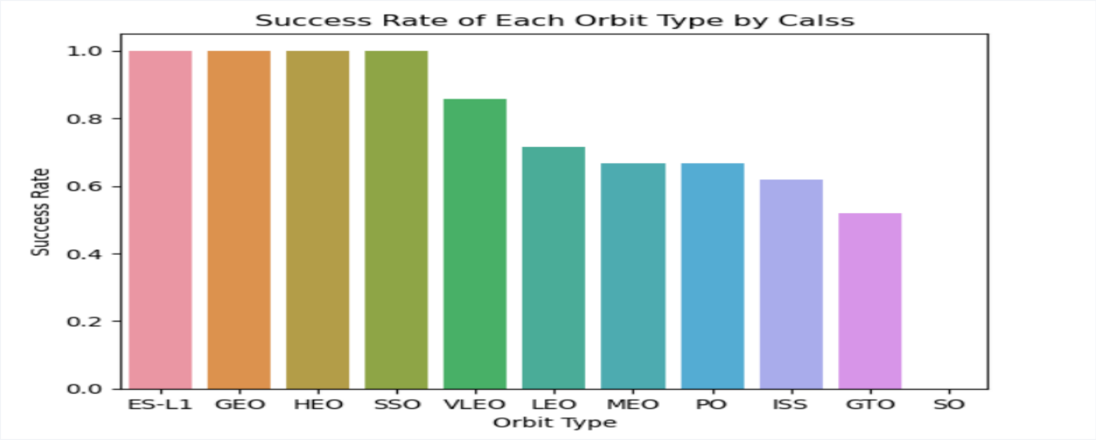
```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

```
True ASDS      41
None None      19
```

EDA with Data Visualization

- 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.
- GitHub URL: https://github.com/FiyinOloyede/IBM_Data_Science_SpaceX_Landing_Prediction/blob/main/EDA%20with%20Data%20viz.ipynb





EDA with SQL

- Below is the summary of the SQL queries performed
  - 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.
- GitHub URL: https://github.com/FiyinOloyede/IBM_Data_Science_SpaceX_Landing_Prediction/blob/main/EDA%20with%20SQL.ipynb

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
- GitHub URL: https://github.com/FiyinOloyede/IBM_Data_Science_SpaceX_Landing_Prediction/blob/main/Visual%20analytics%20with%20folium.ipynb

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

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.
- GitHub URL: https://github.com/FiyinOloyede/IBM_Data_Science_SpaceX_Landing_Prediction/blob/main/Machine%20Learning%20Prediction.ipynb

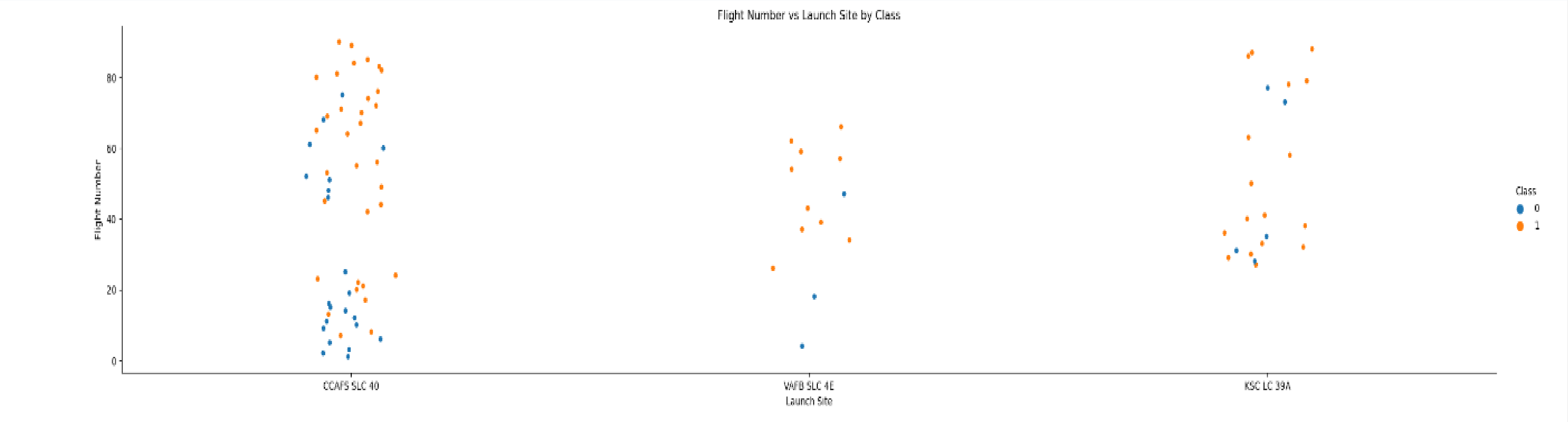
Results

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

Section 2

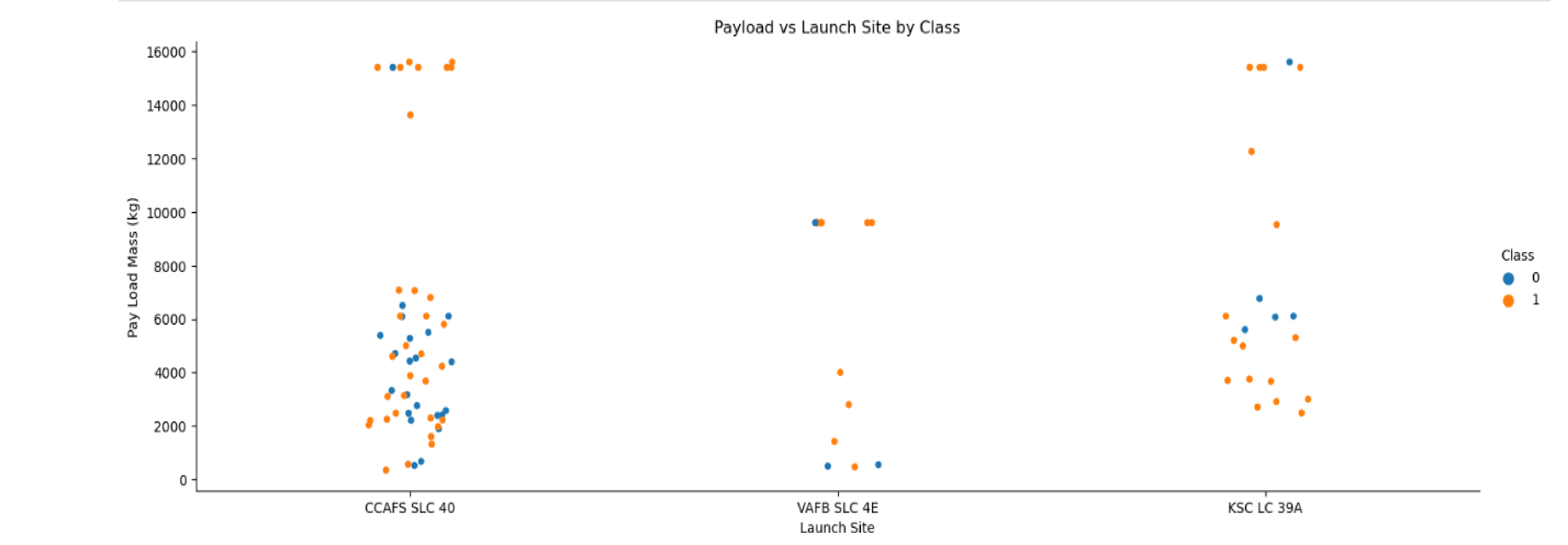
Insights drawn from EDA

Flight Number vs. Launch Site



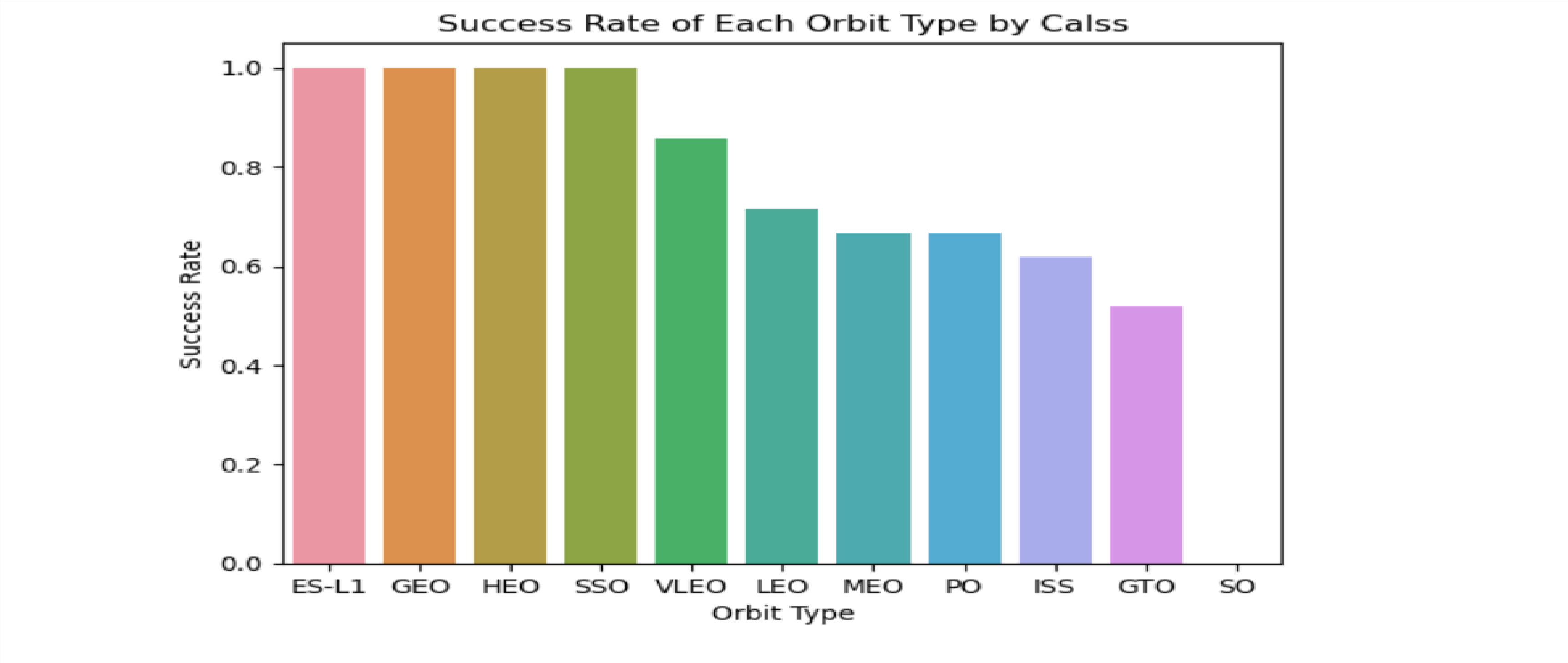
- We see that as the flight number increases, the first stage is more likely to land successfully.

Payload vs. Launch Site



- For the VAFB-SLC launch site, there are no rockets launched for heavy payload mass(greater than 10000)

Success Rate vs. Orbit Type

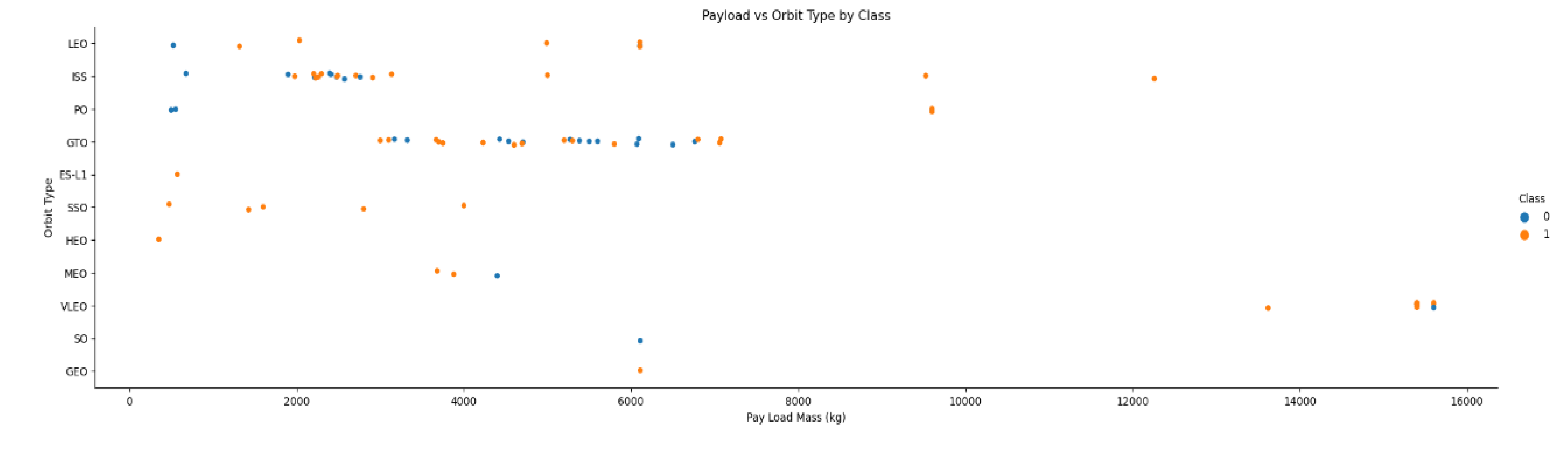


- We can see that ES-L1, GEO, HEO, SSO, VLEO have the most success rate while GTO has the least success rate

Flight Number vs. Orbit Type

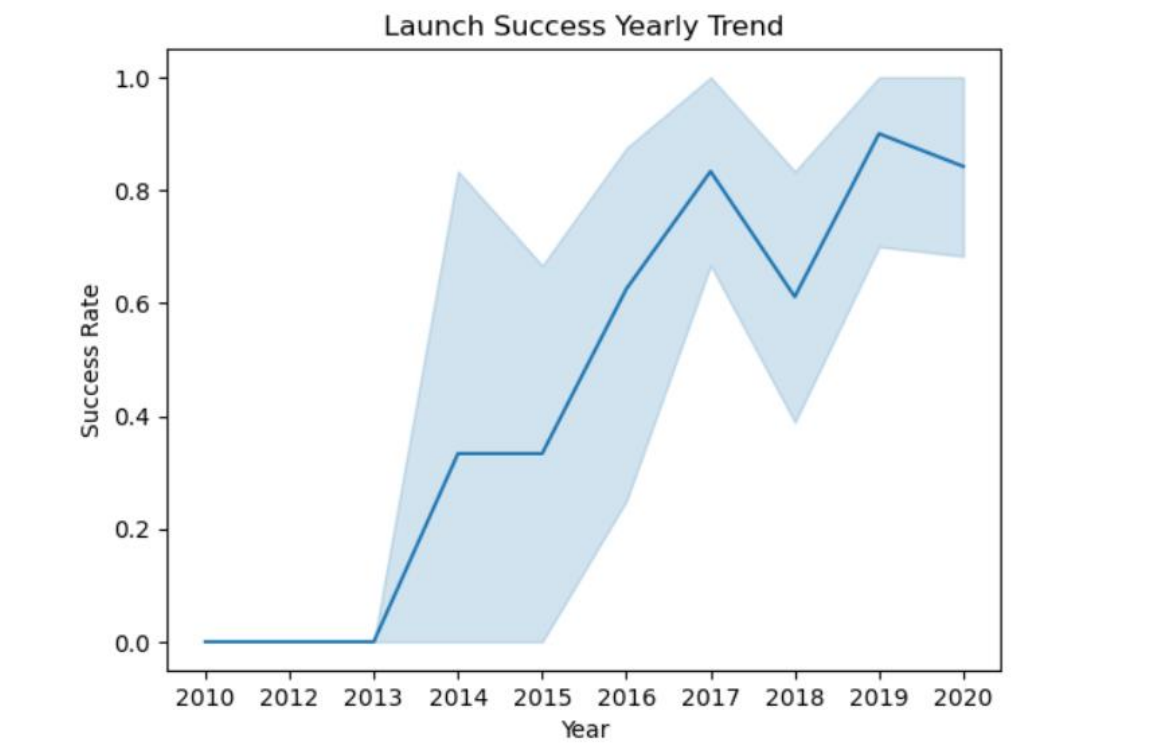
- The plot shows that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However, for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here

Launch Success Yearly Trend



- We can observe that the success rate kept on increasing since 2013 till 2020

All Launch Site Names

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

```
%sql SELECT DISTINCT (LAUNCH_SITE) F
 * mssql+pyodbc://LAPTOP-TLS8KKFO/Sp
Done.
```

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

Launch Site Names Begin with 'CCA'

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

[23]:
```
%%sql
SELECT TOP 5 *
FROM Spacex
WHERE Launch_Site LIKE 'CCA%'
```

```
 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[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	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

- We used 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)

[27]:
```
%%sql

SELECT SUM(PAYLOAD_MASS_KG) AS Total_Payload_NASACRS
FROM Spacex
WHERE Customer = 'NASA (CRS)'
```

```
 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[27]:

Total_Payload_NASACRS
45596

- We calculated the total payload carried by boosters from NASA as 45596 using the query below

Average Payload Mass by F9 v1.1

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

```
[29]: %%sql

SELECT AVG(PAYLOAD_MASS_KG) AS AVG_Payload_F9v11
FROM Spacex
WHERE Booster_Version = 'F9 v1.1'
```

```
 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[29]:

AVG_Payload_F9v11
2928

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4

First Successful Ground Landing Date

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

[47]:
```
%%sql

SELECT MIN(Date) AS Date_first_Sucfl_Lndng
FROM Spacex
WHERE Landing_Outcome = 'Success (Ground pad)'
```

```
 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[47]:

Date_first_Sucfl_Lndng
2015-12-22

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

Successful Drone Ship Landing with Payload 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

[54]:
```
%%sql

SELECT Booster_Version, PAYLOAD_MASS_KG, Landing_Outcome
FROM Spacex
WHERE Landing_Outcome = 'Success (Drone ship)' AND (PAYLOAD_MASS_KG BETWEEN 4000 AND 6000)
```

```
 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[54]:

Booster_Version	PAYLOAD_MASS_KG	Landing_Outcome
F9 FT B1022	4696	Success (drone ship)
F9 FT B1026	4600	Success (drone ship)
F9 FT B1021.2	5300	Success (drone ship)
F9 FT B1031.2	5200	Success (drone ship)

- 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

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

[81]:
```
%%sql

SELECT Mission_Outcome, COUNT(Mission_Outcome) AS COUNT
FROM Spacex
WHERE Mission_Outcome = 'Success' OR Mission_Outcome = 'Failure (in flight)'
GROUP BY Mission_Outcome
```

```
 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[81]:

Mission_Outcome	COUNT
Failure (in flight)	1
Success	99

- We used COUNT and GROUP BY function to count and the WHERE clause to filter Mission Outcome (Success or Failure)

Boosters Carried Maximum Payload

```
[72]: %%sql

SELECT Booster_Version, PAYLOAD_MASS_KG AS Max_Payload_Mass
FROM Spacex
WHERE PAYLOAD_MASS_KG = (SELECT MAX(PAYLOAD_MASS_KG) FROM Spacex)

 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[72]:

Booster_Version	Max_Payload_Mass
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

- We determined the boosters that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

2015 Launch Records

```
[114]: %%sql

SELECT MONTH(Date) AS Month, Landing_Outcome AS Failed_Landing_Outcomes, Booster_Version, Launch_Site
FROM Spacex
WHERE Landing_Outcome = 'Failure (Drone Ship)' AND (DATE LIKE '2015%')

 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[114]:

Month	Failed_Landing_Outcomes	Booster_Version	Launch_Site
1	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
4	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

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.

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

[110]:
```
%%sql

SELECT Landing_Outcome, COUNT(Landing_Outcome) AS Count
FROM Spacex
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY Count DESC
```

```
 * mssql+pyodbc://LAPTOP-TLS8KKFO/Spacex?driver=ODBC Driver 17 for SQL Server
Done.
```

[110]:

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

Section 3

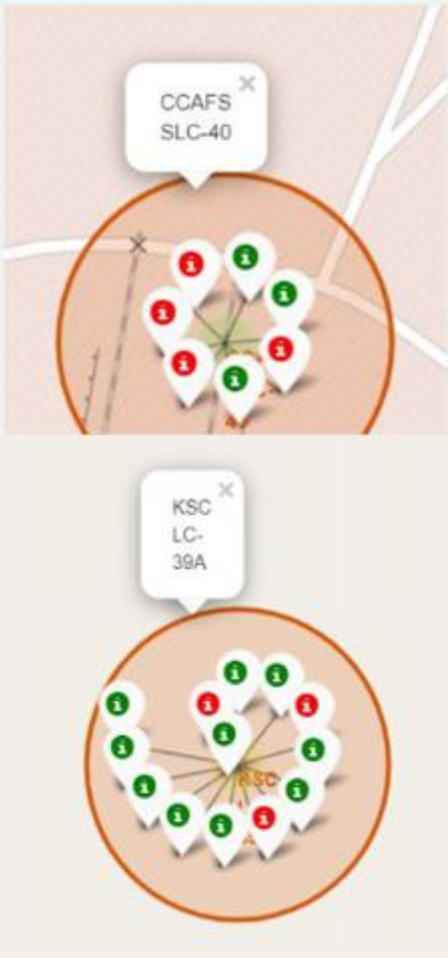
Launch Sites Proximities Analysis

All launch sites global map markers

[64]:

We can see that space x launch sites are in the United State of America coast lines; Florida and california

Markers showing launch sites with color labels



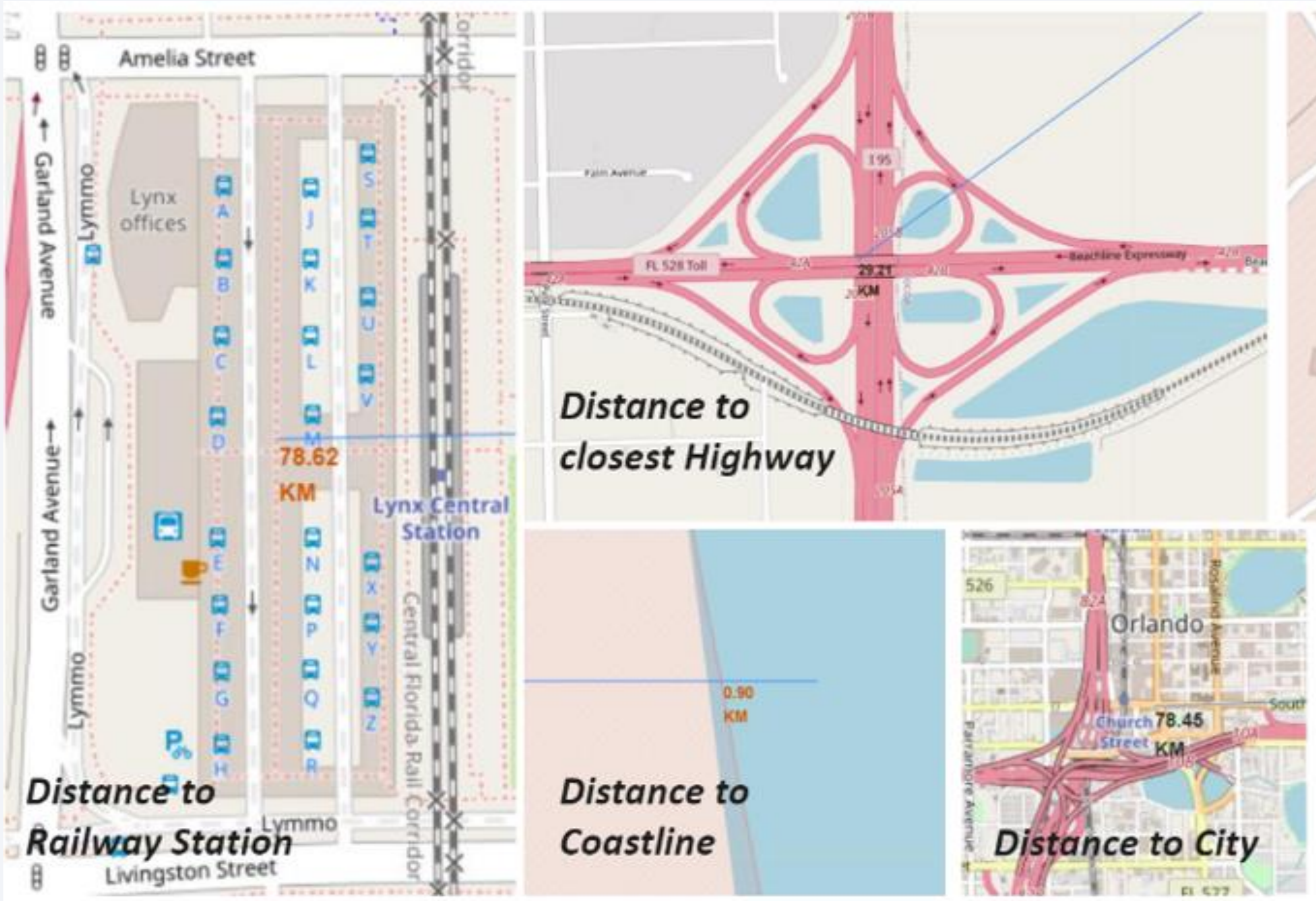
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

37

<Folium Map Screenshot 3>



- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

Build a Dashboard with Plotly Dash

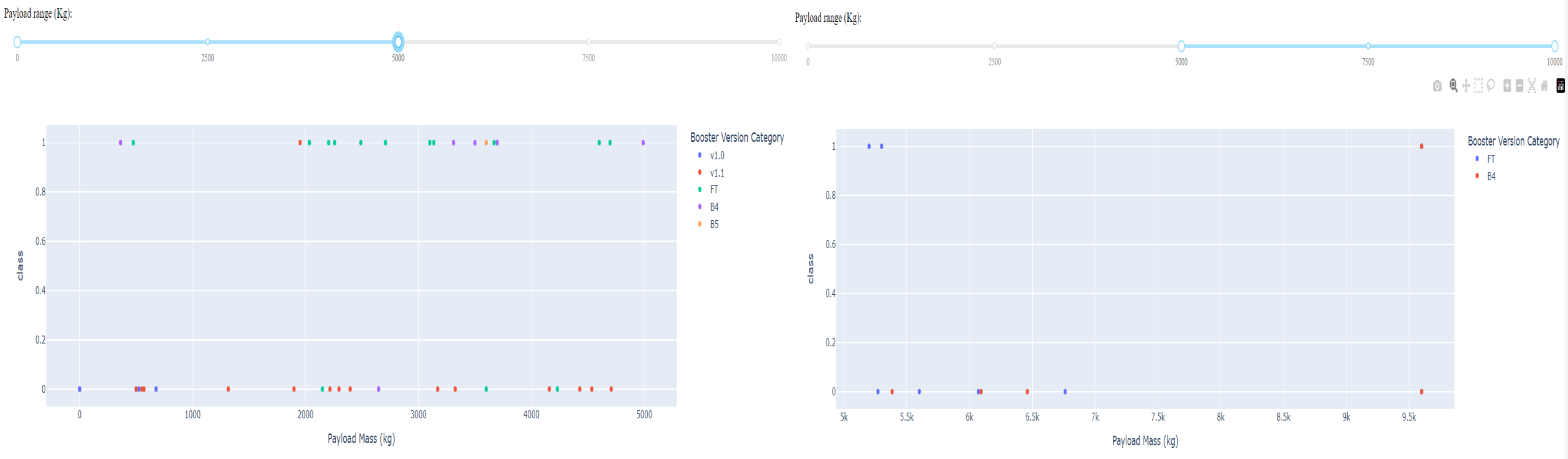
Success Percentage Achieved by Each Launch Site

We can see that KSC LC – 39A has the most successful launches

Launch Site with the Highest Launch Success Ratio

KSC LC – 39A achieved a 76.9% success rate with just 23.1% failure rate

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



We can see we have more booster versions for low weighted payloads than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Find the method performs best:

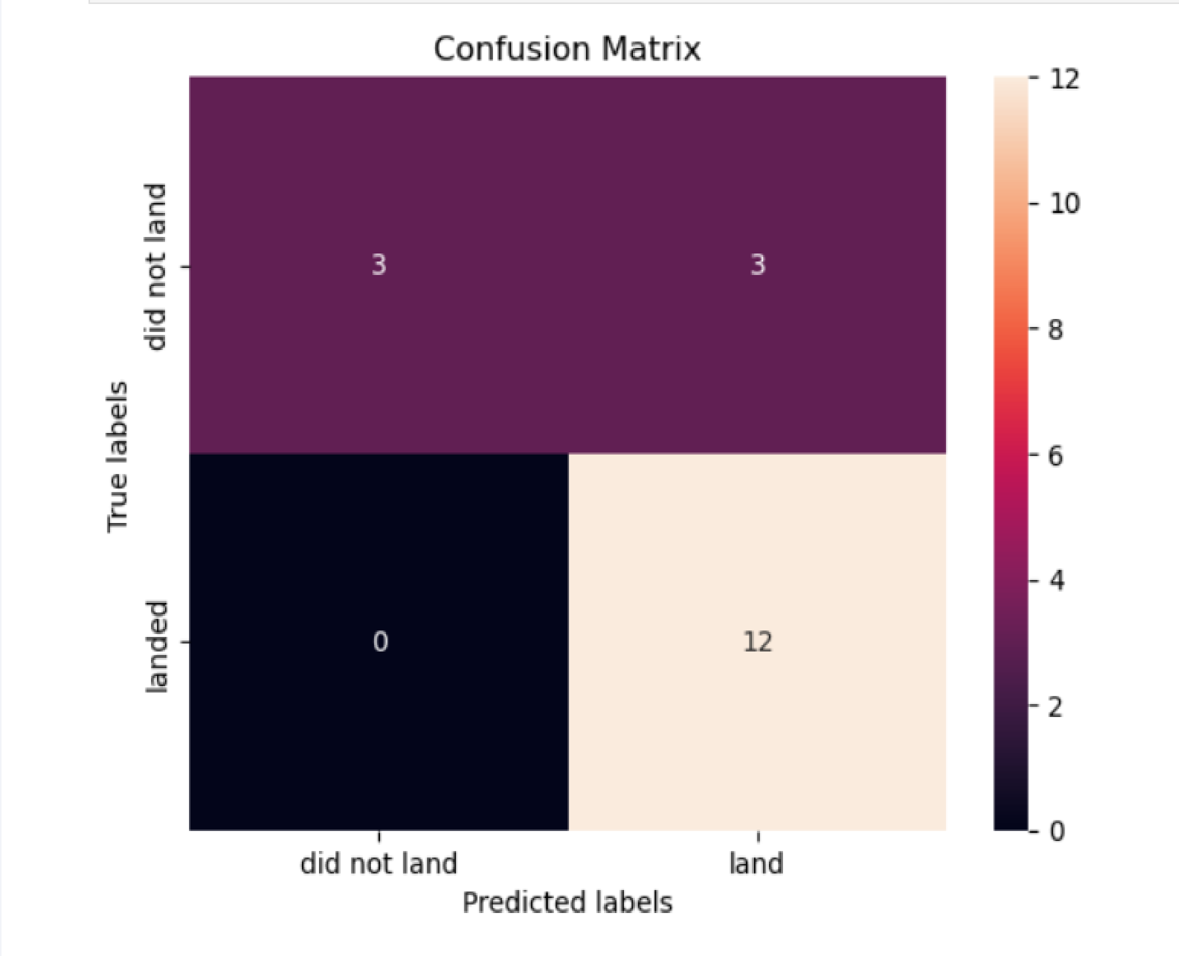
[127]:
```
models = {'LogisticRegression':logreg_cv.best_score_,
          'SupportVector': svm_cv.best_score_,
          'DecisionTree':tree_cv.best_score_,
          'KNeighbors':knn_cv.best_score_,
         }

best_model = max(models, key=models.get)
print('Best model is', best_model,'with a score of', models[best_model])
if best_model == 'LogisticRegression':
    print('Best parameters are :', logreg_cv.best_params_)
elif best_model == 'SupportVector':
    print('Best parameters are :', svm_cv.best_params_)
elif best_model == 'DecisionTree':
    print('Best parameters are :', tree_cv.best_params_)
else:
    print('Best parameters are :', knn_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8625
Best parameters are : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter':
'random'}
```

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

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 .i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

We can conclude that:

- as the flight number increases, the first stage is more likely to land successfully.
- the success rate kept on increasing since 2013 till 2020.
- ES-L1, GEO, HEO, SSO, VLEO have the most success rate while GTO has the least success rate.
- KSC LC-39A site had the most successful launches.

The Decision tree classifier is the best machine learning algorithm to predict if the first stage will land successfully

Thank you!