IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Linus Kubis
25 October 2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through a REST API
  - Data Collection through Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis (EDA):
    - Using SQL
    - Data Visualization
    - Data Analysis with Folium
    - Data Analysis using Plotly Dash
  - Machine Learning:
    - Classification
- Summary of all results
  - Results of the Exploratory Data Analysis
  - Predictive Analytics from the Machine Learning Algorithmns

Introduction

- Project background and context
- SpaceX 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 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 if an alternate company wants to bid against SpaceX for a rocket launch. The goal of this project is to determine whether the first stage will land successfully.
- Problems you want to find answers
- Identify the factors that influence the landing outcome
- What are the best conditions to increase the probability of a succesfull landing

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - The Data was collected using the SpaceX REST API and web scraping from Wikipedia
- Perform data wrangling
  - Corrected missing values and using One-Hot-Encoding 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 collection was done using the requests.get method to the SpaceX REST API
- Decode the response in the json() format and turn it into a Pandas dataframe using the .json_normalize() function
- The data set contained a lot of IDs, so the API was used again to get new information to replace the IDs with actual data
- Replace missing values

- For web scraping BeautifulSoup was used to extract the data from the HTML tables
- Parse those tables and turn it into a Pandas Dataframe for further analysis

Data Collection – SpaceX API

- Get request method used on the API
- Turn the json() format into a Pandas Dataframe
- Replace missing values

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

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

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

- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/Data-Collection.ipynb

```
# Calculate the mean value of PayloadMass column
mean = data_falcon9["PayloadMass"].mean()

# Replace the np.nan values with its mean value
data_falcon9["PayloadMass"].replace(np.nan, mean, inplace=True)
data_falcon9.isnull().sum()
```

Data Collection - Scraping

- Request the Falcon9 Launch Wikipage from the URL
- Create BeautifulSoup Object from response
- Extract the colums/variable from the HTML table header

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

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

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

```
# 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")
```

```
column_names = []

# 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
rows = first_launch_table.find_all("th")
for row in rows:
    name = extract_column_from_header(row)
    if name !=  None and len(name) > 0:
        column_names.append(name)
```

- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/Webscraping.ipynb

Data Wrangling

- The process of cleaning and unifying messy data in preparation for the EDA
- First calculate the number of launches on each site
- Calculate the number of occurrence of each orbit and with this calculate the number and occurence of mission outcome of the orbits
- Then create landing outcome label from the outcome column for the purpose of the EDA, visualization and ML
- Lastly export the dataframe as a CSV file

- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/Data-Wrangling.ipynb

EDA with Data Visualization

- Used scatter plots to find the relationship between these attributes:
  - Payload Mass and Flight Number
  - Flight Number and Launch Site
  - Payload Mass and Launch Site
  - Flight Number and Orbit Type
  - Payload Mass and Orbit Type

- From this we learned what factors are the most important ones for the success of the landing outcome
- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/edadataviz.ipynb

EDA with Data Visualization

- Furthermore we used bar graphs to identify the relationship between categorical values:
  - Success Rate and Orbit Type
- To show trends or time dependent attributes a line graph was used to show the success rate of the launches in every Year
- Lastly dummy variables were created to turn into categorical columns for the Feature Engineering used for the prediction of the future module

- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/edadataviz.ipynb

EDA with SQL

- The queries performed using SQL:
  - Display the names of launch sites
  - Display 5 records where launch sites begin with the string "CCA"
  - Display the total payload mass carried by boosters launched by NASA
  - Display the average payload mass carried by booster version F9 v1.1
  - Display the date the first successful landing in ground pad was achieved
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - List the total number of successful and failure mission outcomes
  - List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
  - 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.
  - 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.

- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/SQL_EDA.ipynb

Build an Interactive Map with Folium

- The latitude and the longitude coordinates from each launch site were taken and visualized on an interactive folium map. To each launch site a circle marker was added with the name of the launch site
- The outcomes from the dataframe_outcomes column were visualized with Red (0, failure) and Green (1, success) marker on the map in a MarkerCluster()
- Then the distance from launch sites to various interesting landmarks was calculated to answer the questions:
  - Are launch sites in close proximity to railways?
  - Are launch sites in close proximity to highways?
  - Are launch sites in close proximity to coastline?
  - Do launch sites keep certain distance away from cities?

- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- An interactive dashboard using Plotly Dash was created
- The Dashboard contains pie charts which show the total launches per launch site
- A scatter plot was drawn to show the relationship with Outcome between the different booster versions and the Payload Mass

- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/spacex_dash_app.py

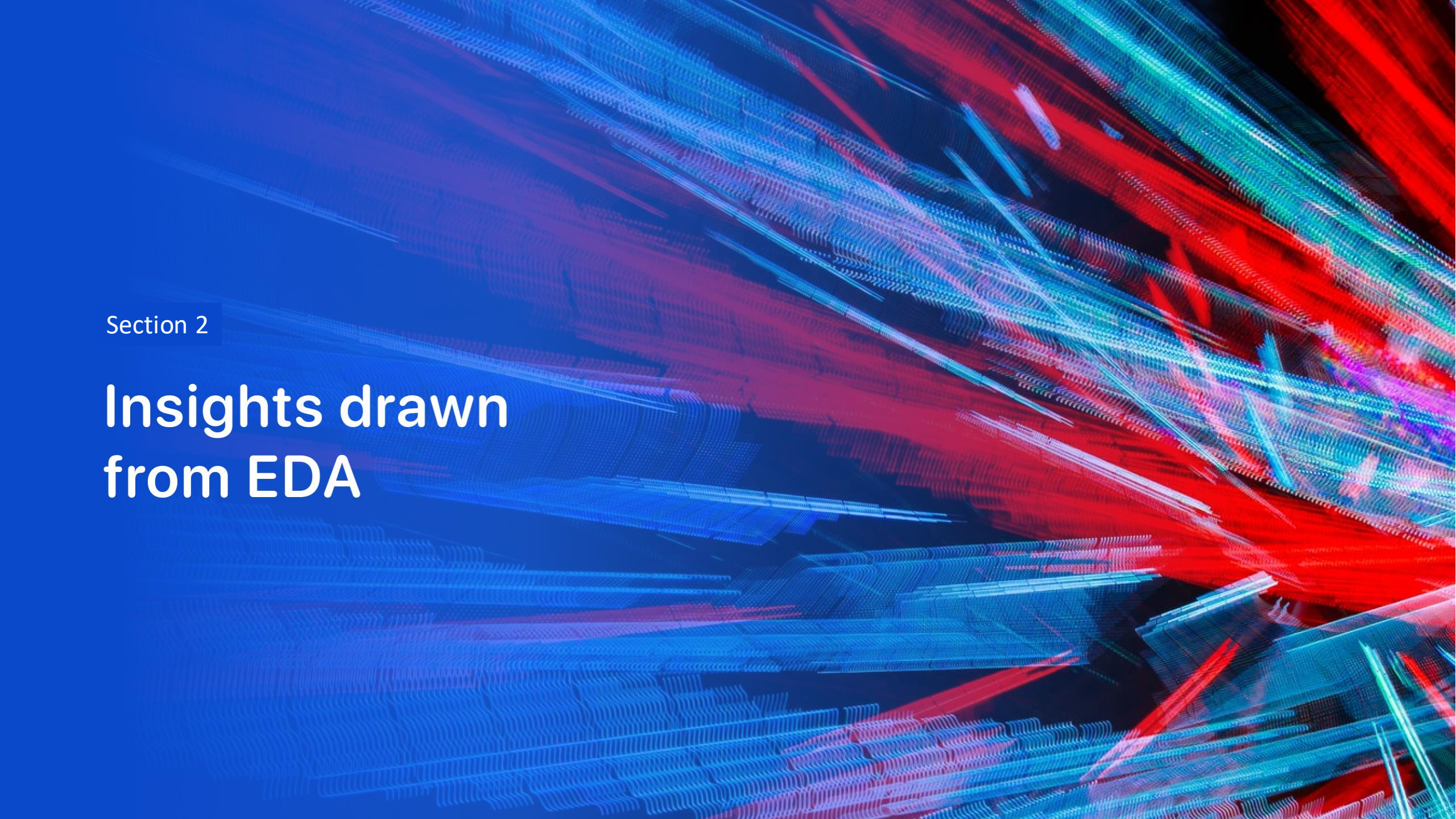
Predictive Analysis (Classification)

- Building the model:
  - Load the dataset into NumPy and Pandas
  - Transform and split the data into training and test sets
  - Decide the type of ML model
  - Search the optimal parameters using GridSearchCV and fit the dataset
- Evaluating the model:
  - Calculate the accuracy for each model
  - Plot the confusion matrix
- Improving the model:
  - Tune the algorithm or parameters
- Find the best Model:
  - Use the model with the best accuracy

- https://github.com/Ai-create-byte/IBM-Data-Science_Course-SpaceX/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

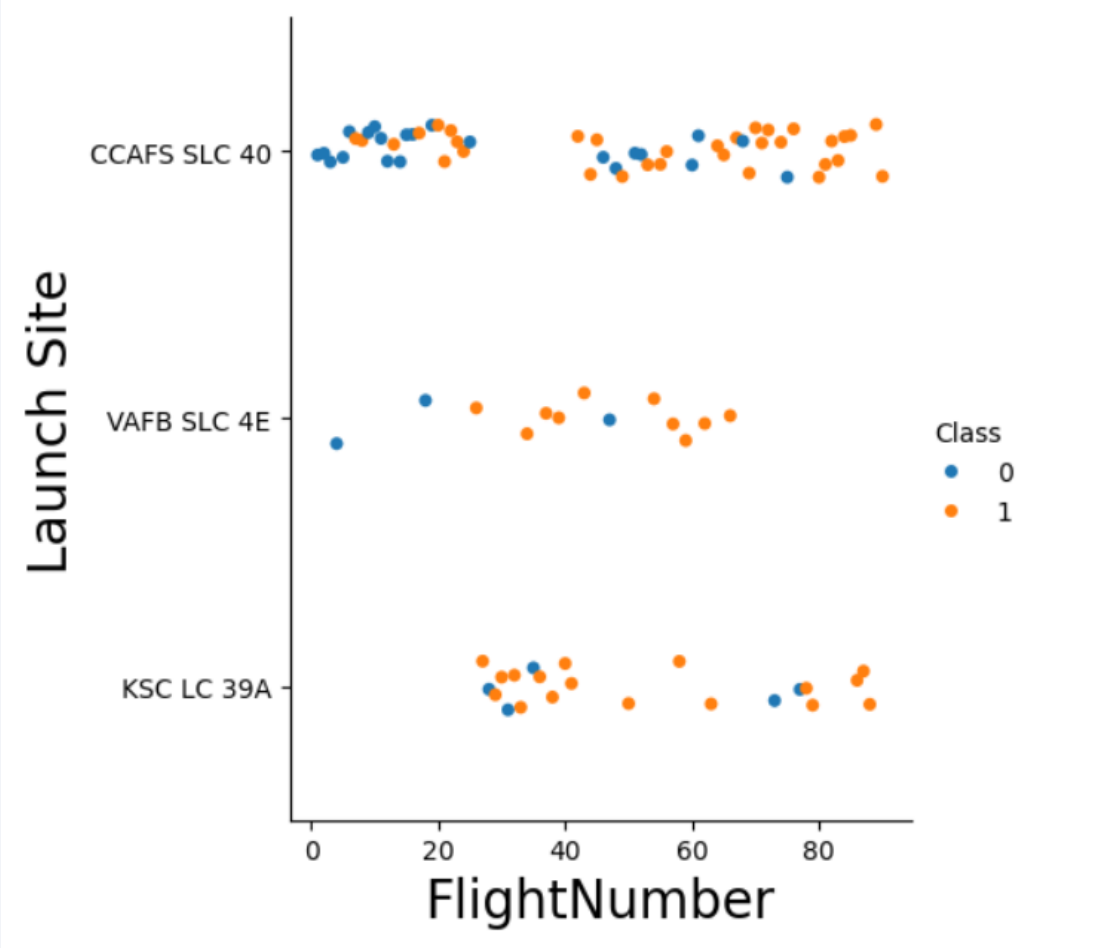
Results

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


Section 2
Insights drawn from EDA

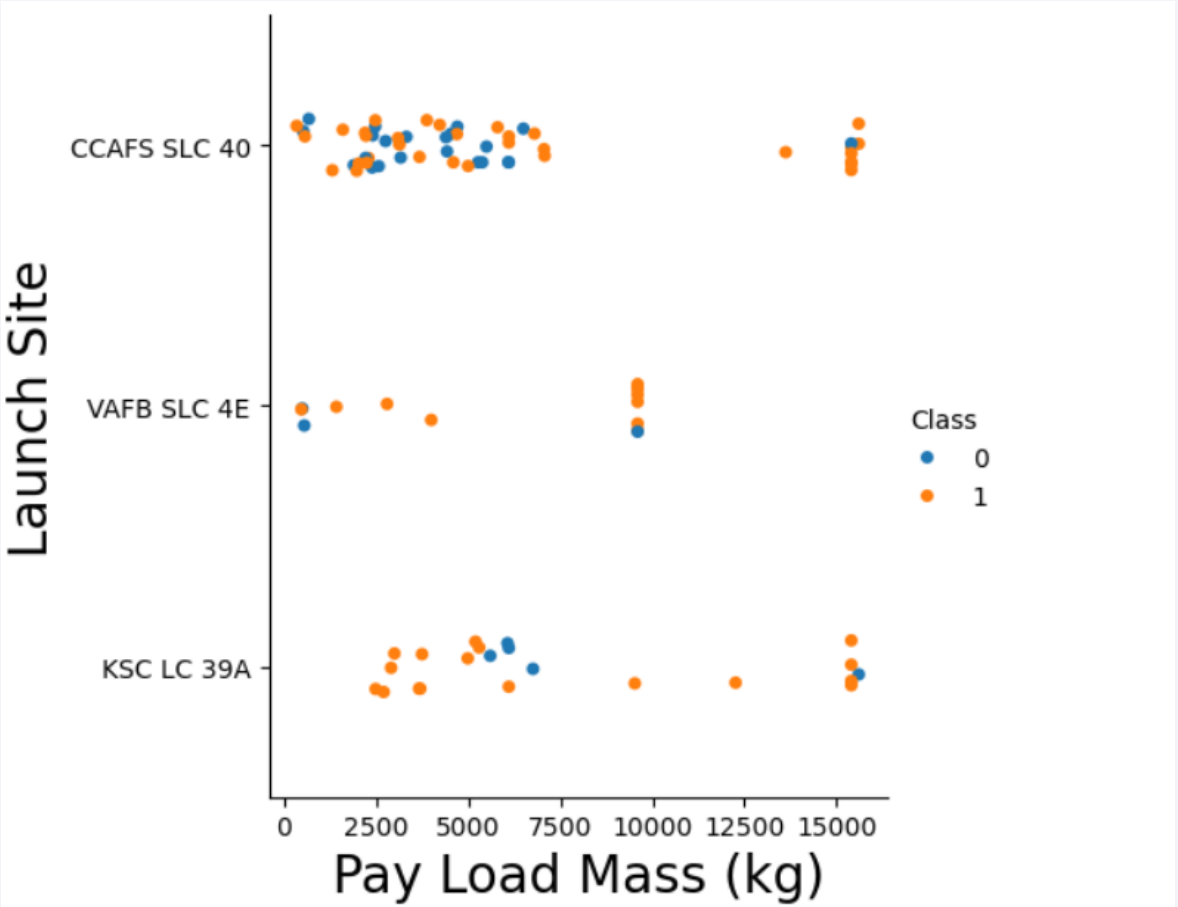
Flight Number vs. Launch Site

- The scatter plot shows that with more launches from a site the success rate will increase
- CCAFS SLC 40 doesn't really show this



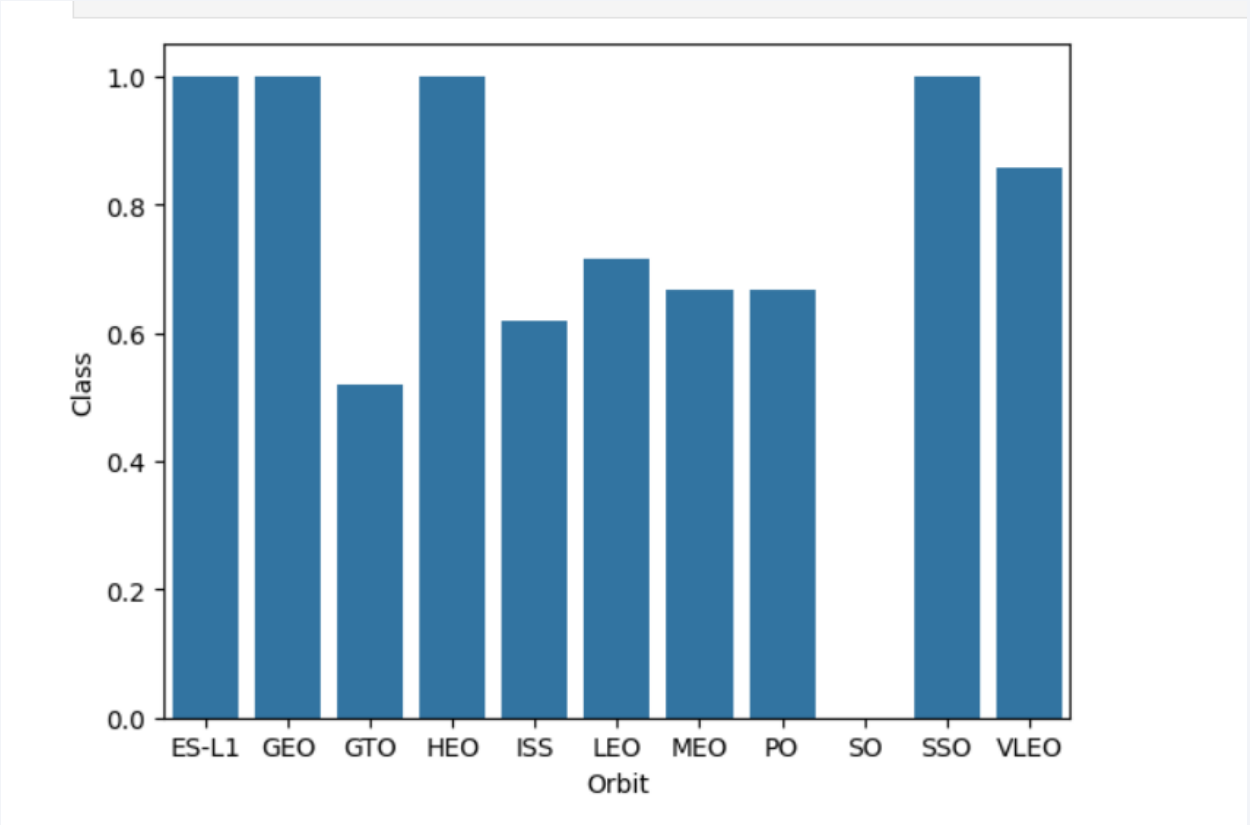
Payload vs. Launch Site

- If the Payload Mass is greater than 7000kg the success rate is highly increased.
- It seems that there isnt a clear pattern that the launch site is dependent to the Payload Mass for the success rate.



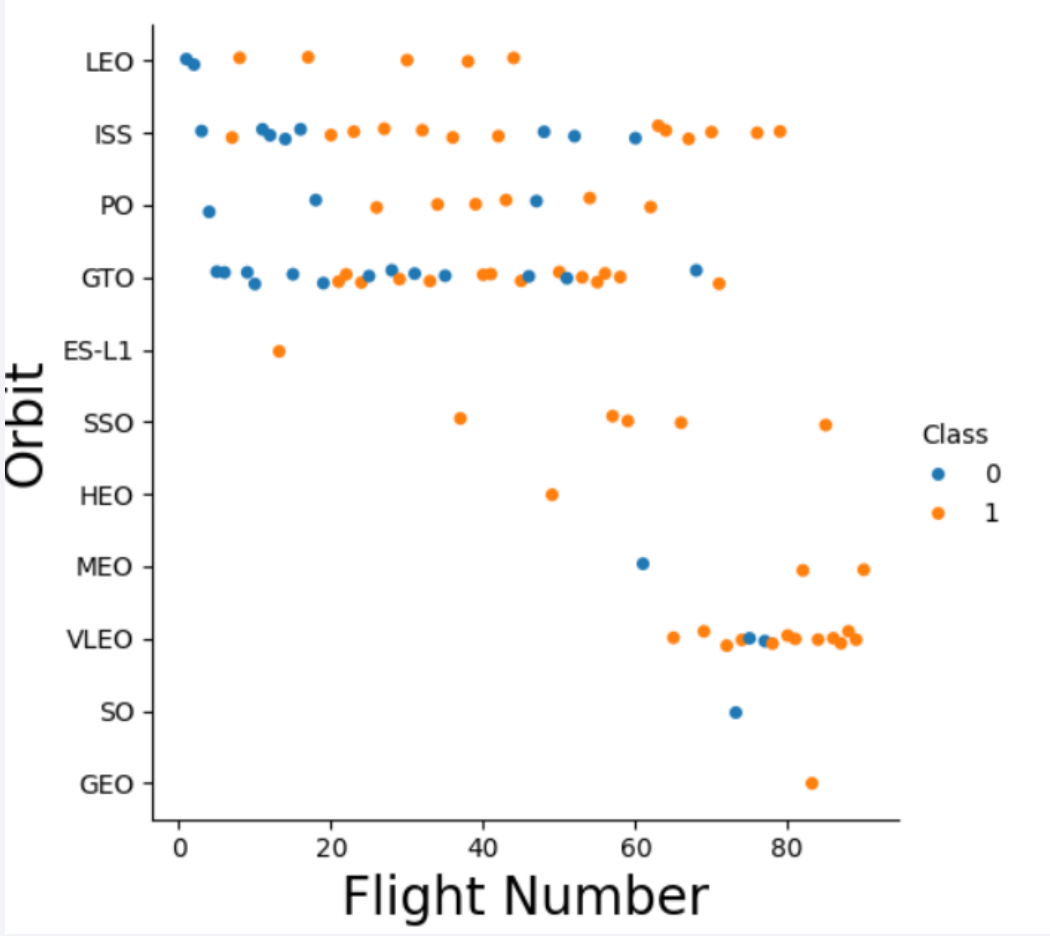
Success Rate vs. Orbit Type

- This figure shows the success rate for each orbit
- Some orbits have a success rate of 100% like ES-L1, GEO, HEO, SSO while SO has 0%
- There isnt really a clear by just looking at the orbit types alone because some orbits only have 1 datapoint



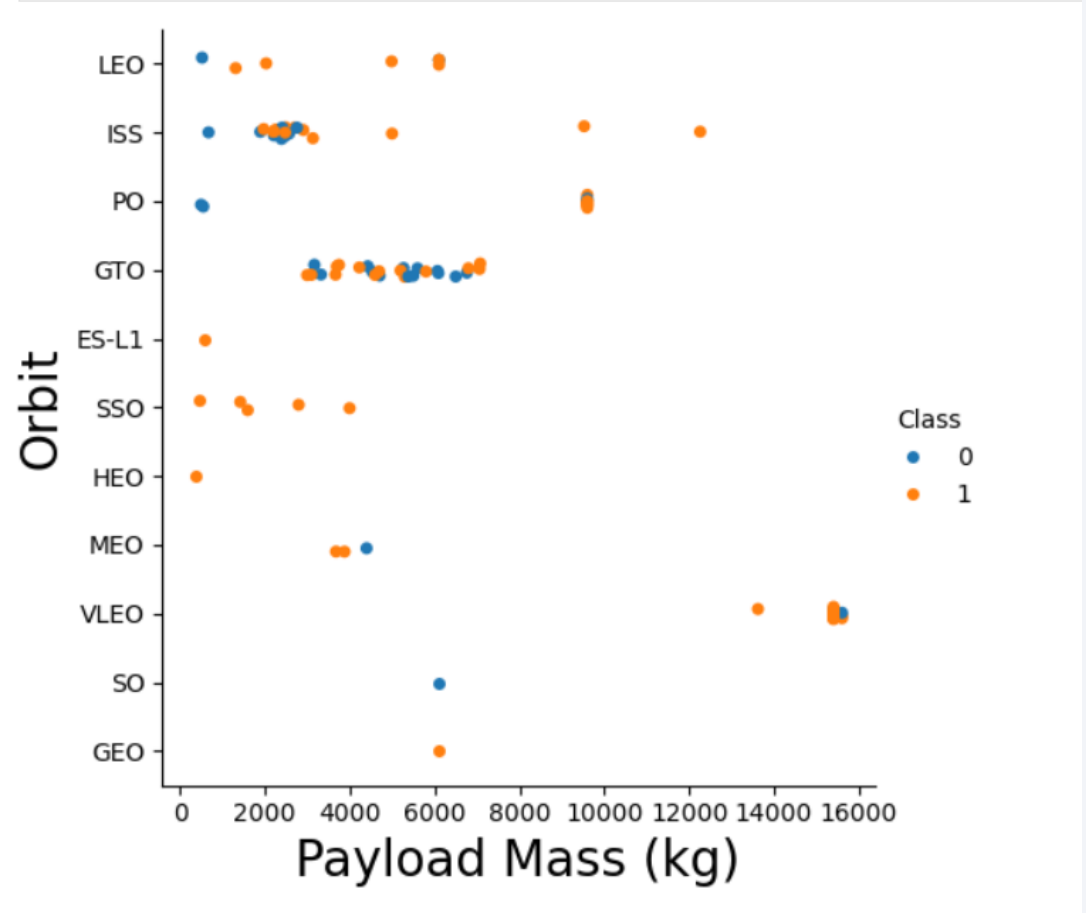
Flight Number vs. Orbit Type

- With more flights the success rate on each orbit increases except for GTO there isnt really a relationship between those two attributes
- We you can see that some orbits only have 1 datapoint so they should be excluded from the dataset



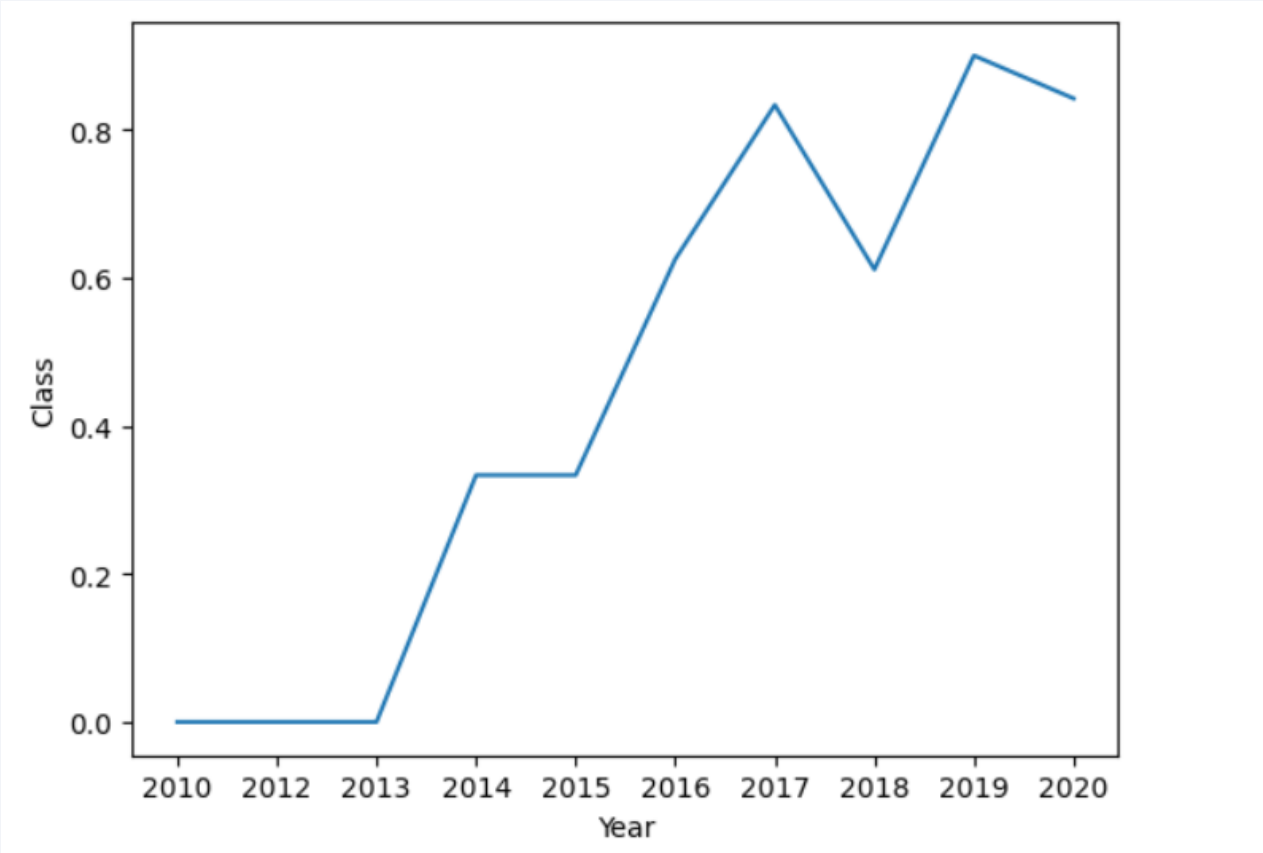
Payload vs. Orbit Type

- Heavier Payload has positive impact on ISS, LEO and PO orbit. It has a negative impact on MEO orbit
- GTO doesn't have a relation between these two attributes
- GEO, SO, HEO, ES-L1 only have one entry, so they need more data



Launch Success Yearly Trend

- Increasing trend from 2013 to 2020
- Maybe the learned from the mistakes and the new technology helps the success of the missions



All Launch Site Names

```
%sql SELECT DISTINCT(Launch_Site) from SPACEXTABLE;
```

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

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

Use DISTINCT to only show unique launch sites.

Launch Site Names Begin with 'CCA'

```
%sql SELECT * FROM SPACEXTABLE 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
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

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE
```

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

SUM(PAYLOAD_MASS__KG_)
619967

Average Payload Mass by F9 v1.1

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

```
%sql SELECT BOOSTER_VERSION, AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE BOOSTER_VERSION LIKE "F9 v1.1"
```

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

Booster_Version	AVG(PAYLOAD_MASS__KG_)
F9 v1.1	2928.4

First Successful Ground Landing Date

```
%sql SELECT MIN(DATE) FROM SPACEXTABLE WHERE LANDING_OUTCOME LIKE "Success (ground pad)"
```

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

MIN(DATE)
2015-12-22

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

```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

Total Number of Successful and Failure Mission Outcomes

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

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

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

Boosters Carried Maximum Payload

```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ == (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

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

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

```
%sql SELECT SUBSTR("Date", 6, 2) AS Month, "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Failure (drone ship)' AND SUBSTR("Date", 1, 4) = '2015';
```

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

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

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

```
%sql SELECT Date FROM SPACEXTBL WHERE "Landing _Outcome" like 'Succes%' AND Date BETWEEN '04-06-2010' AND '20-03
```

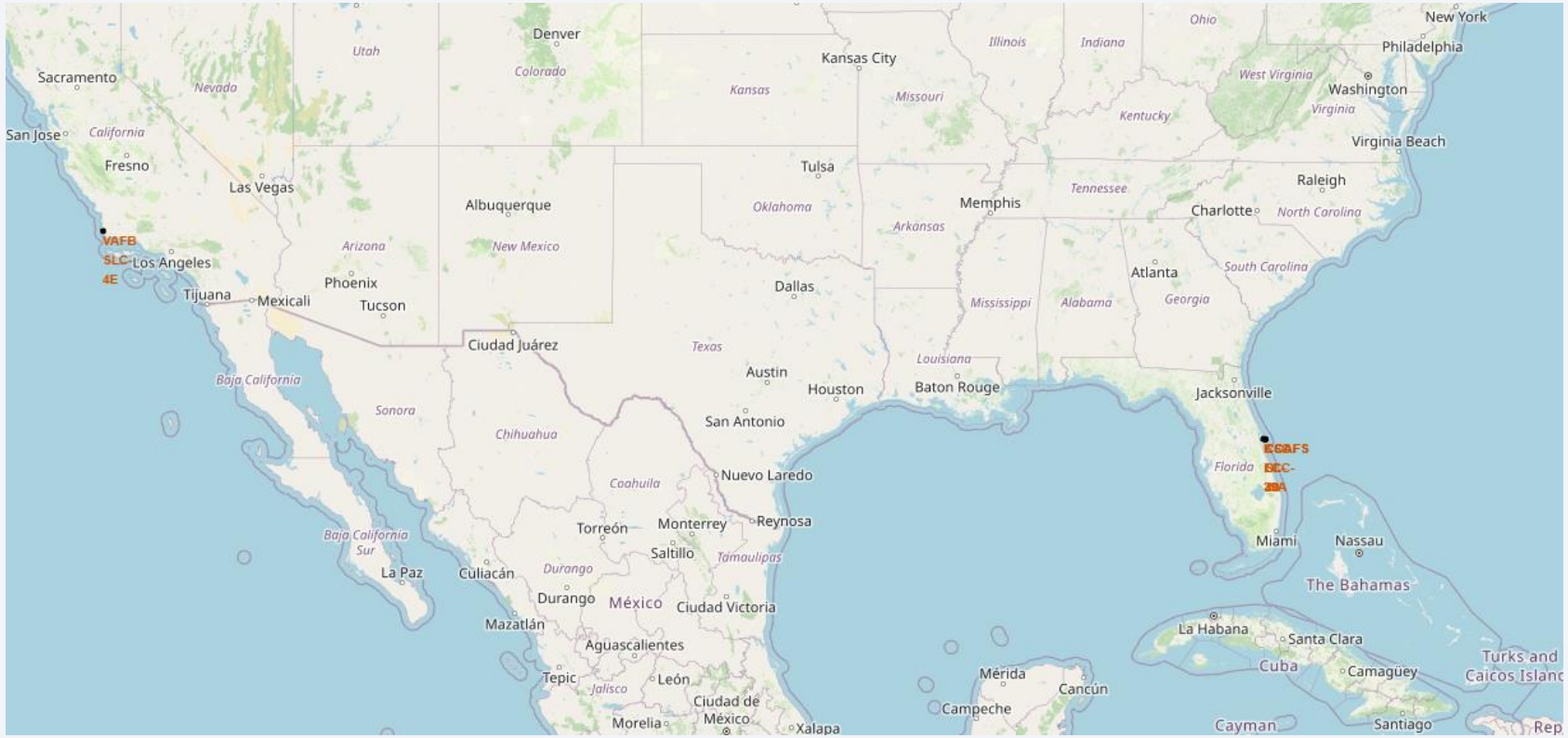
```
 * sqlite:///my_data1.db
Done.
```

Date
19-02-2017
18-10-2020
18-08-2020
18-07-2016
18-04-2018

Section 3

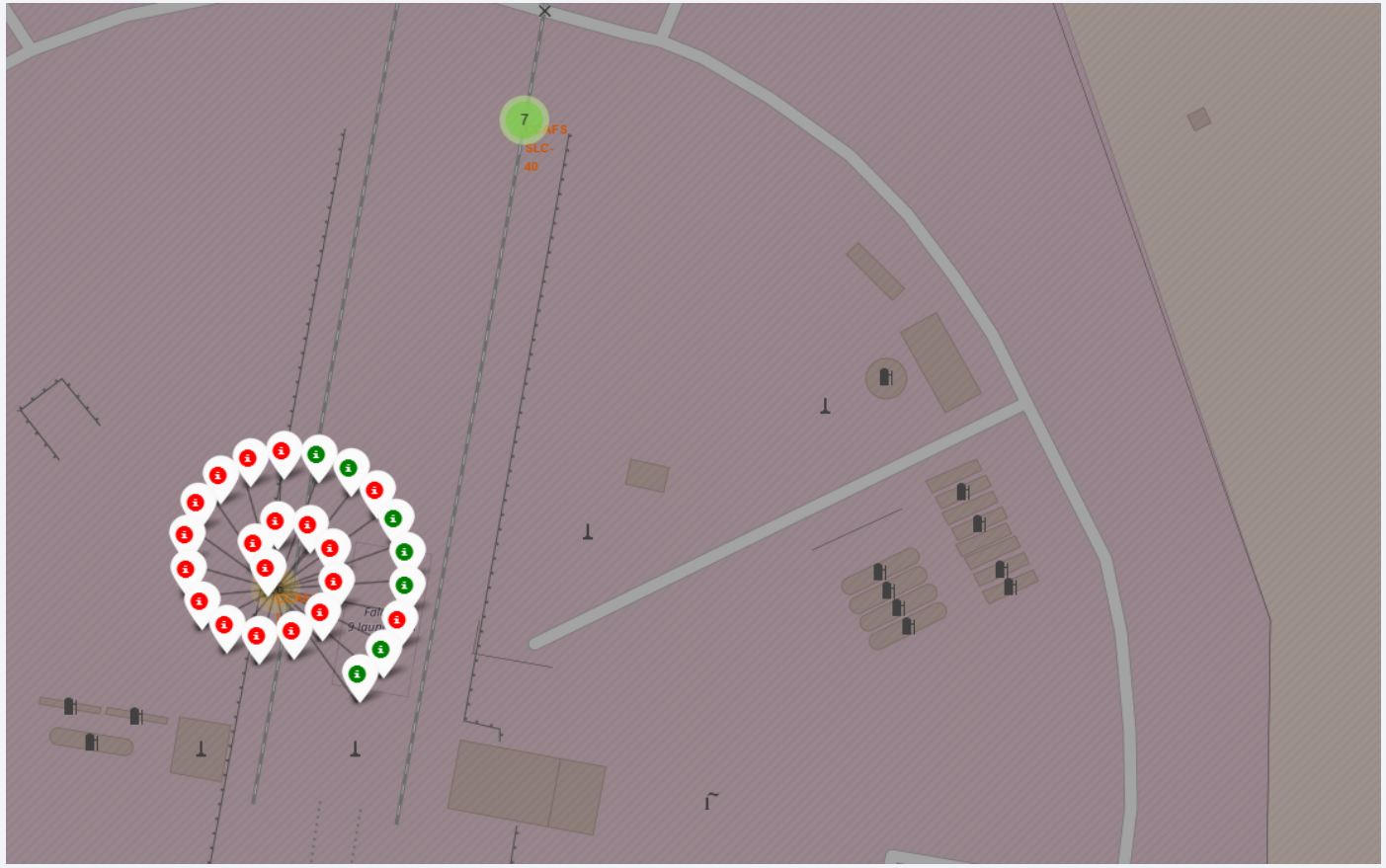
Launch Sites Proximities Analysis

All Launch Sites



Shows that all the lauch sites are in the U.S. and near the coast.

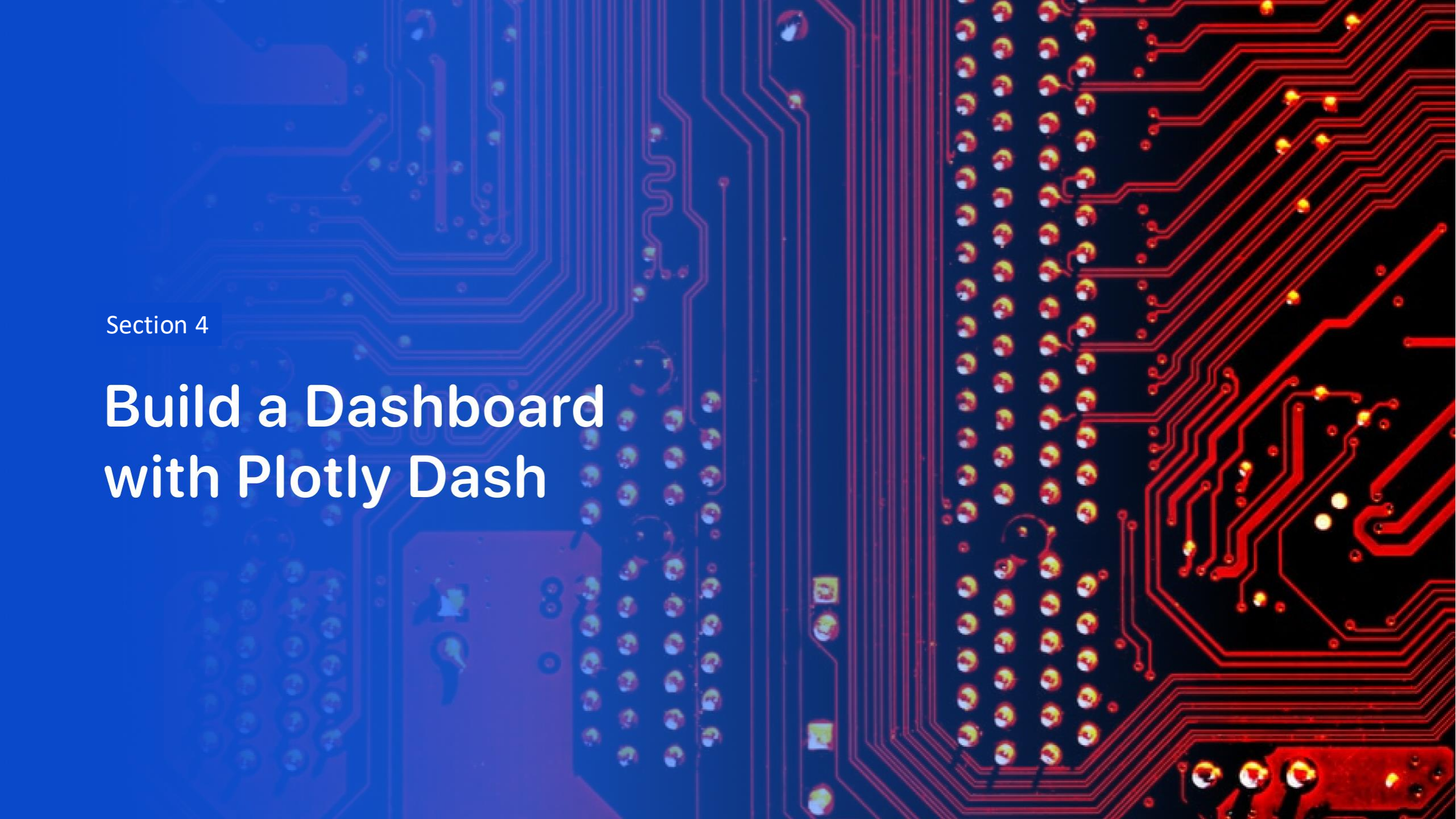
Launch Site with Markers



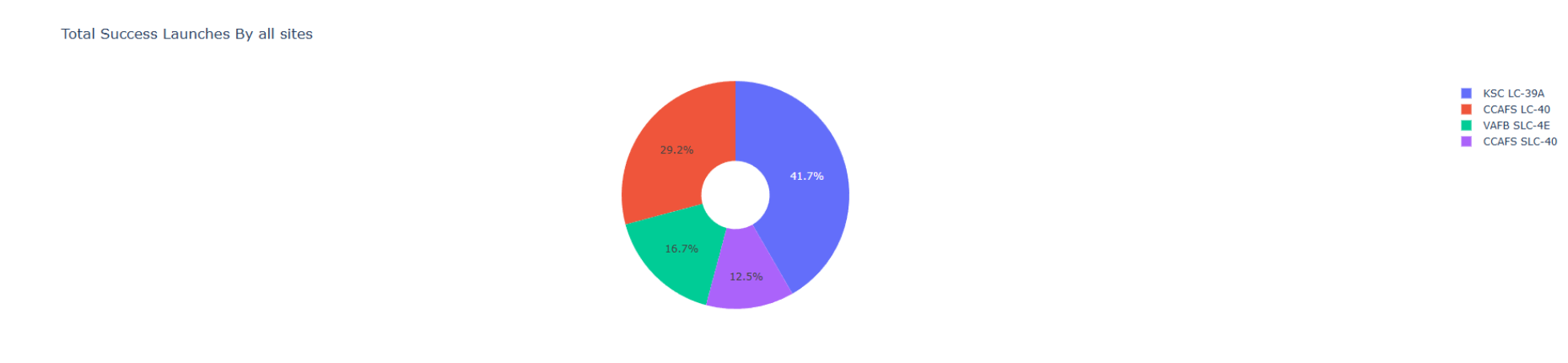
The markers show the number of launches and the outcome of those. (Red – failure, Green - success)

Lauch Site Proximities

Each launch site is the coast and far from city centers. There arent any public roads or railroads near the lauch sites.

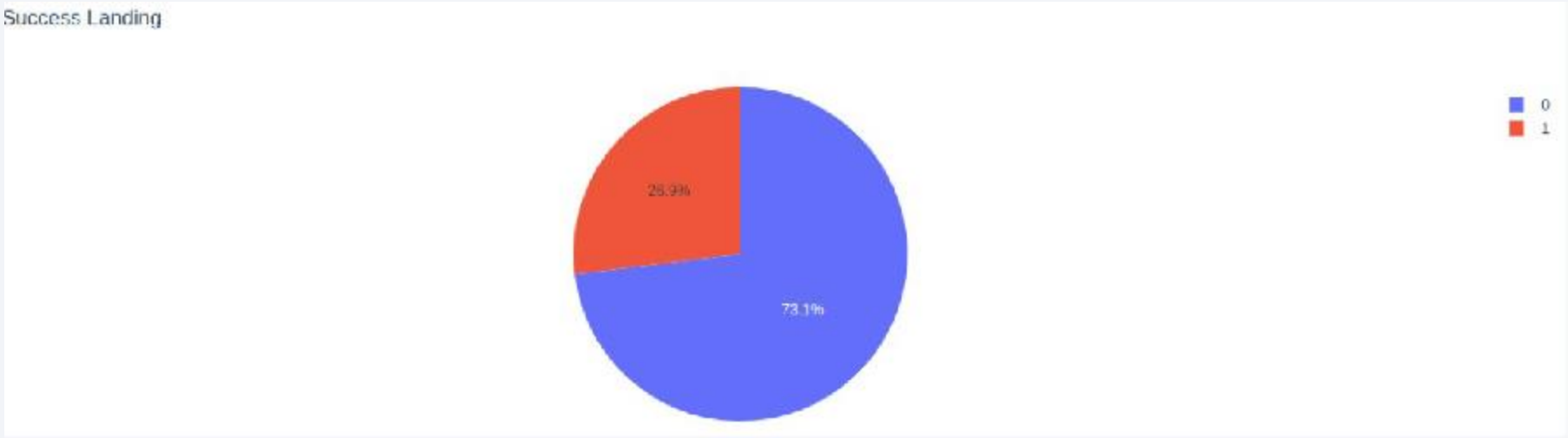

Section 4
Build a Dashboard with Plotly Dash

Pie Chart - Success Count per Launch Site



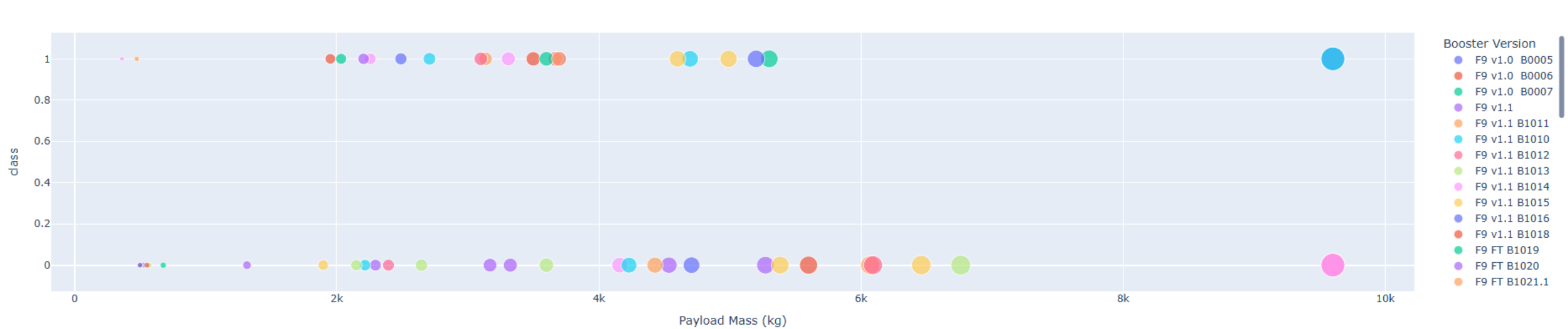
The total succesfull lauches at all sites in comparison.

CCAFS LC-40 Success Rate (Pie Chart)



The CCAFS LC-40 launch site has the highest success rate with 73.1%

Scatter Plot by Payload Mass



There isnt a clear relationship between the booster version / Payload Mass on the outcome of the mission.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

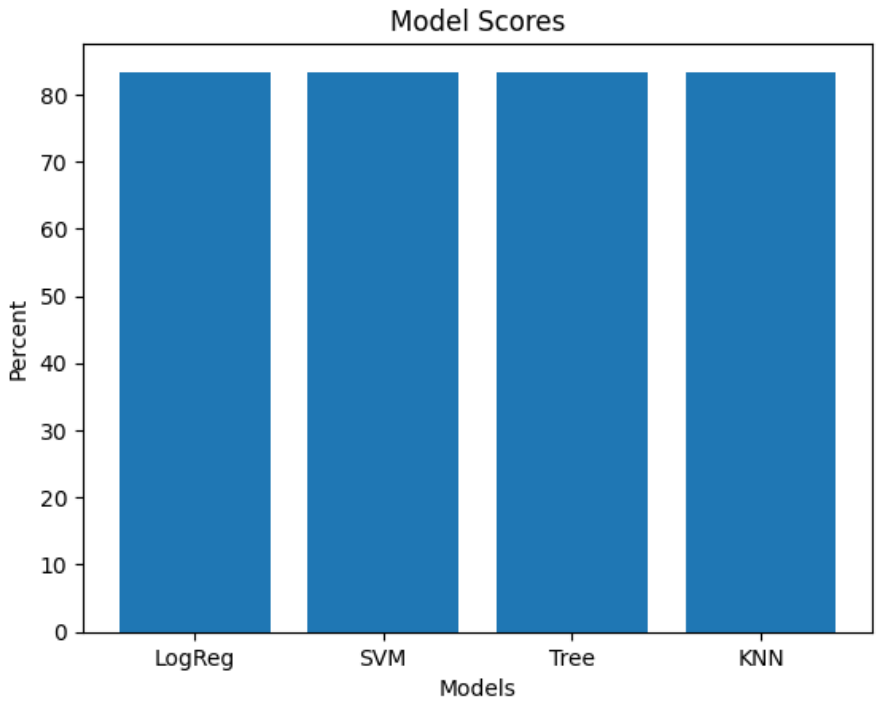
```
print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
print('Accuracy for K nearsdt neighbors method:', knn_cv.score(X_test, Y_test))
```

```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.8333333333333334
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

In this case the performance of the models is identical.

This model has the best accuracy with the best_score_ attribute.

```
[83.33333333333334, 83.33333333333334, 83.33333333333334, 83.33333333333334]
['LogReg', 'SVM', 'Tree', 'KNN']
```

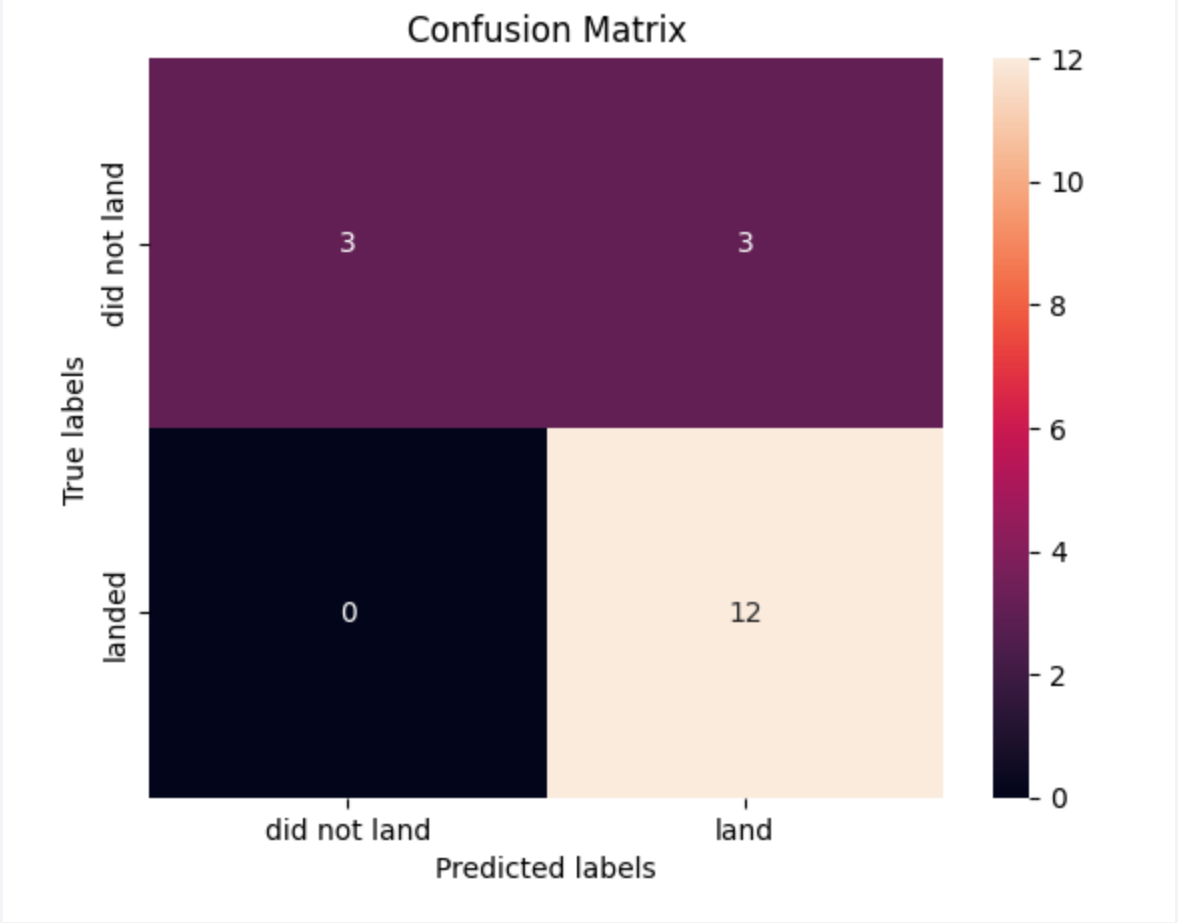



```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
accuracy : 0.8732142857142857
```

Confusion Matrix

- 12 were correctly predicted
- 3 were predicted to land but didn't land in real life
- 3 were correctly classified as did not land



Conclusions

- The Decision Tree Classifier is the best Algoritm for this problem
- The success rate increased from 2013 until today
- The CCAFS LC-40 launch site has the highest success rate
- SSO has a 100% succes rate and has more than

Thank you!