IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ryan Saul
September 1, 2022

OUTLINE

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

EXECUTIVE SUMMARY

- **Summary of Methodologies**
  - Data Collection with API, SQL and Webscraping
  - Data Wrangling
  - Exploratory Data Analysis (EDA) with SQL
  - Exploratory Data Analysis (EDA) with Visualization
  - Interactive Visualizations with Folium
  - Machine Learning Predictive Analysis
- **Summary of All Results**
  - Exploratory Data Analysis (EDA) Results
  - Predictive Analysis Results

INTRODUCTION

- **Project background and context**
  - SpaceX Falcon 9 cost is $62 million, others are around $165 million.
  - We are attempting to predict whether or not the SpaceX Falcon 9 rocket will have a successful land in its first stage.
  - This is significant because of money saved if the first stage can be recovered.
- **Problems that need to be addressed**
  - Significant factors influencing landing outcome
  - Machine Learning prediction models accuracy for recovery
  - Using EDA insight for trends

Section 1

Methodology

METHODOLOGY

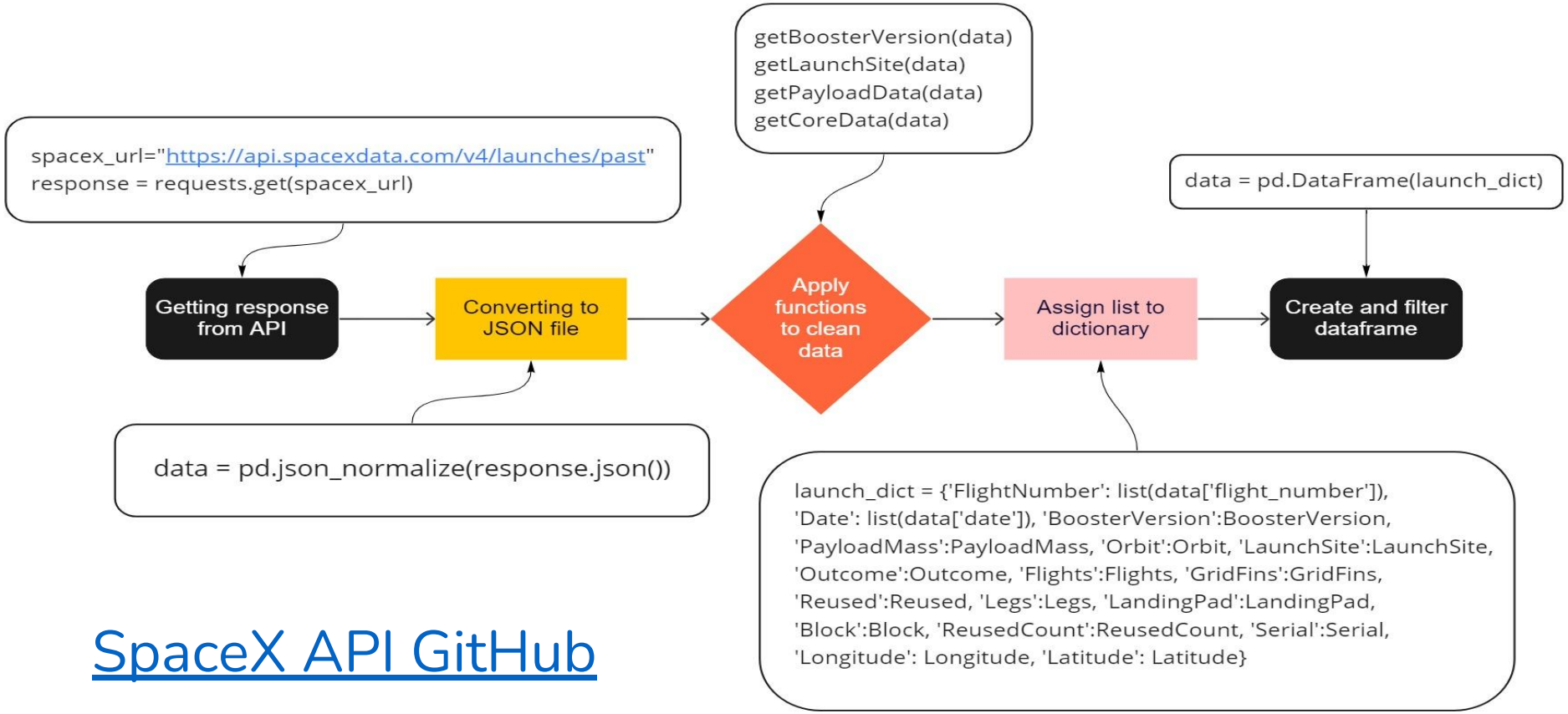
Executive Summary

- Data collection methodology
  - SpaceX API, webscraping with BeautifulSoup
- Perform data wrangling
  - Launch sites, orbit types and occurrences, create landing class
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Building, tuning, evaluating classification models

DATA COLLECTION

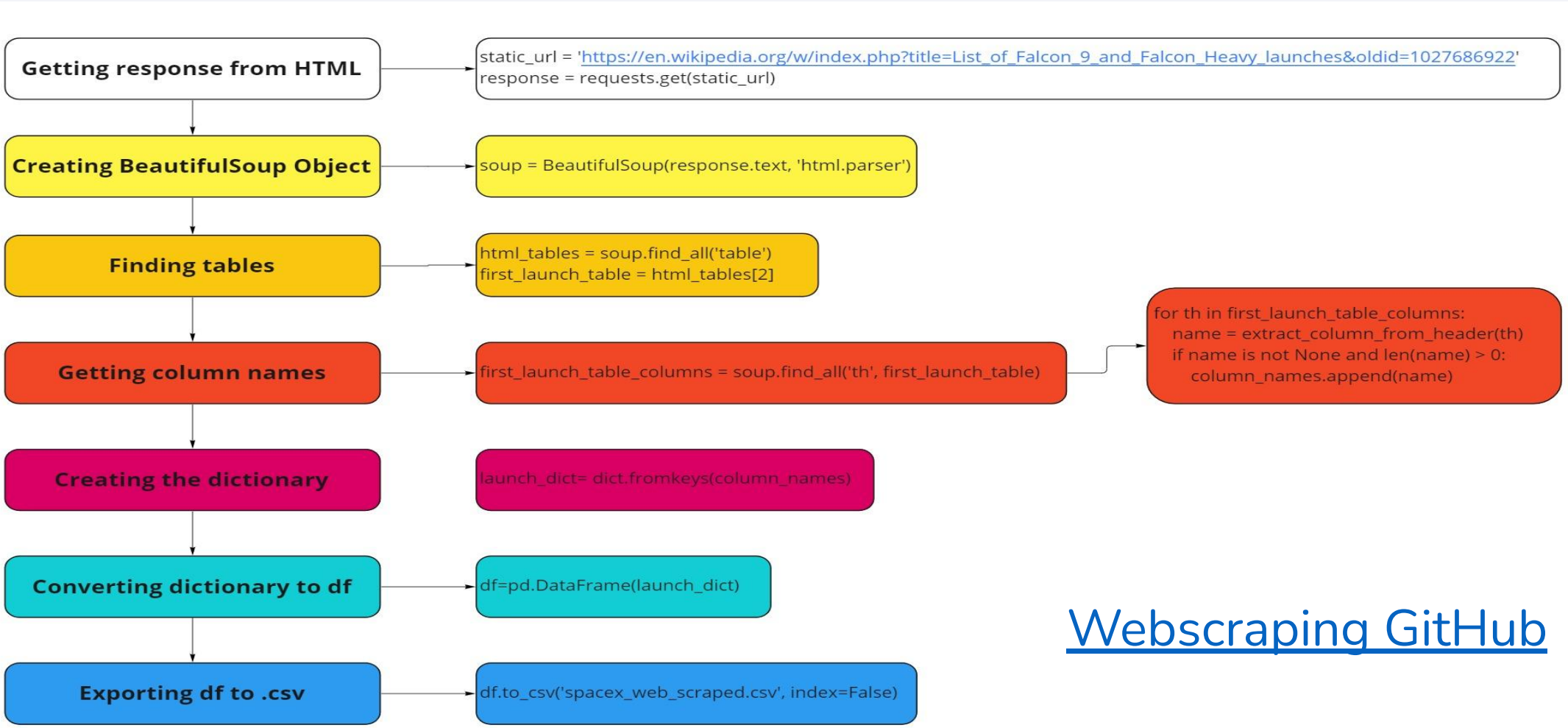
- **Data collection process**
  - SpaceX API get request
  - JSON normalization
  - Filter Falcon 9 launches
  - Adjust missing values
  - Webscraping with BeautifulSoup
  - Create SpaceX dataframe

DATA COLLECTION – SpaceX API



[SpaceX API GitHub]

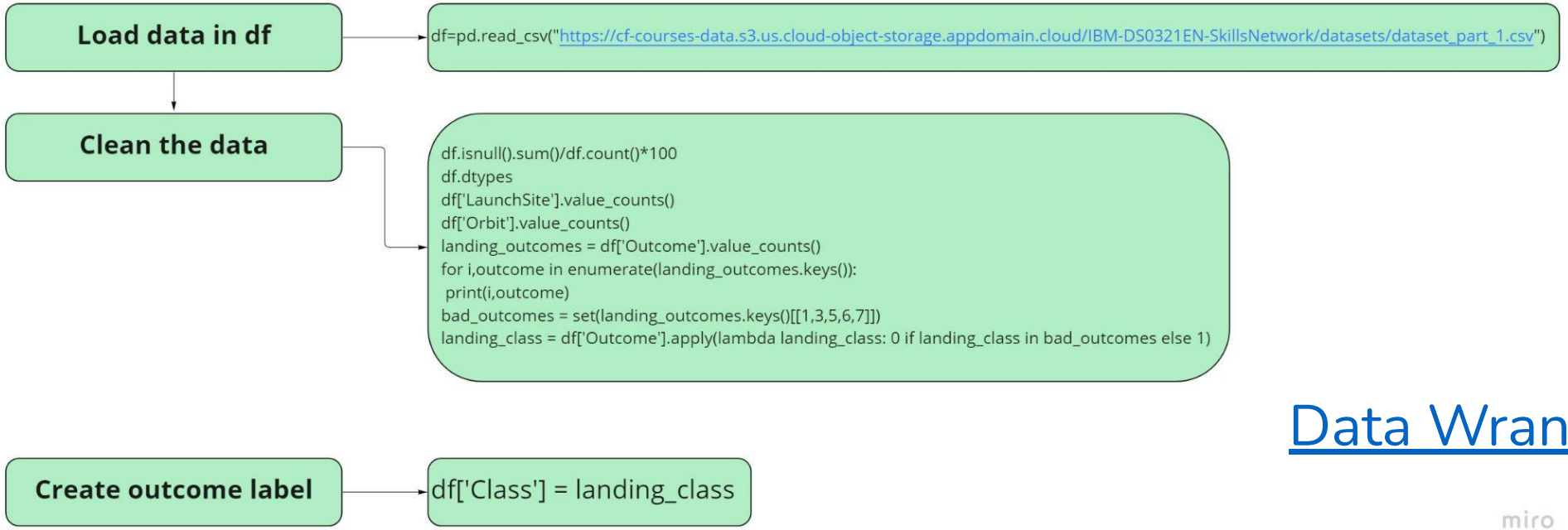
DATA COLLECTION - Scraping



[Webscraping GitHub]

DATA WRANGLING

- A variety of processes designed to transform raw data into more readily used formats
- The main goal here is to create a training label with the outcomes labeled as:
  - 0 as bad outcomes
  - 1 as good outcomes



[Data Wrangling GitHub]

WHAT IS EDA?

- The critical process of performing initial investigations on data so as to discover patterns,to spot anomalies,to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.
- Typical commands
    - df.shape
    - df.info()
    - df.describe()
    - df[‘variable’].unique()
    - df[‘variable’].value_counts()
- Som plot and chart types
    - box
    - scatter
    - line
    - bar
    - distribution

EDA WITH DATA VISUALIZATION

- Swarm plot
  - Flight Number vs Launch Site
  - Payload vs Launch Site
  - Flight Number vs Orbit Type
  - Payload vs Orbit Type
- Category plot
  - Success Rate vs Orbit Type
- Line plot
  - Yearly launch success trend
- Created dummy variables for:
  - Orbit
  - LaunchSite
  - LandingPad
  - Serial
- Converted all numeric variable to float

EDA with Visualization GitHub

Visuals in Appendix

EDA WITH SQL

SQL queries performed:

- Names of unique launch sites
- Launch sites beginning with ‘ksc’
- Total payload mass carried by boosters launched by NASA (CRS)
- Average payload mass carried by booster version F9 v1.1
- Date for successful landing outcome in drone ship
- Booster names with ground pad success and payload mass between 4000 and 6000
- Total number of successful and failed missions
- booster_versions names that carried a maximum payload mass
- List records in the format of month names, successful landing_outcomes in ground pad, booster versions, launch_site for the months in the year 2017
- Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order

EDA with SQL GitHub

BUILD AN INTERACTIVE MAP WITH FOLIUM

MAP OBJECT	CODE	RESULT
Map Marker	folium.Marker()	puts a marker on a specific point on a map
Icon Marker	folium.Icon()	puts an icon on a specific point on a map
Circle Marker	folium.Circle()	creates a circle where a specific marker is being put
Polyline	folium.PolyLine()	draws a line between points on a map
Marker Cluster	MarkerCluster()	simplifies a map by grouping points together until zoomed in close enough
Popup Marker	folium.Popup()	adds a popup with information to a specific marker on a map

Folium Interactive Map Jupyter

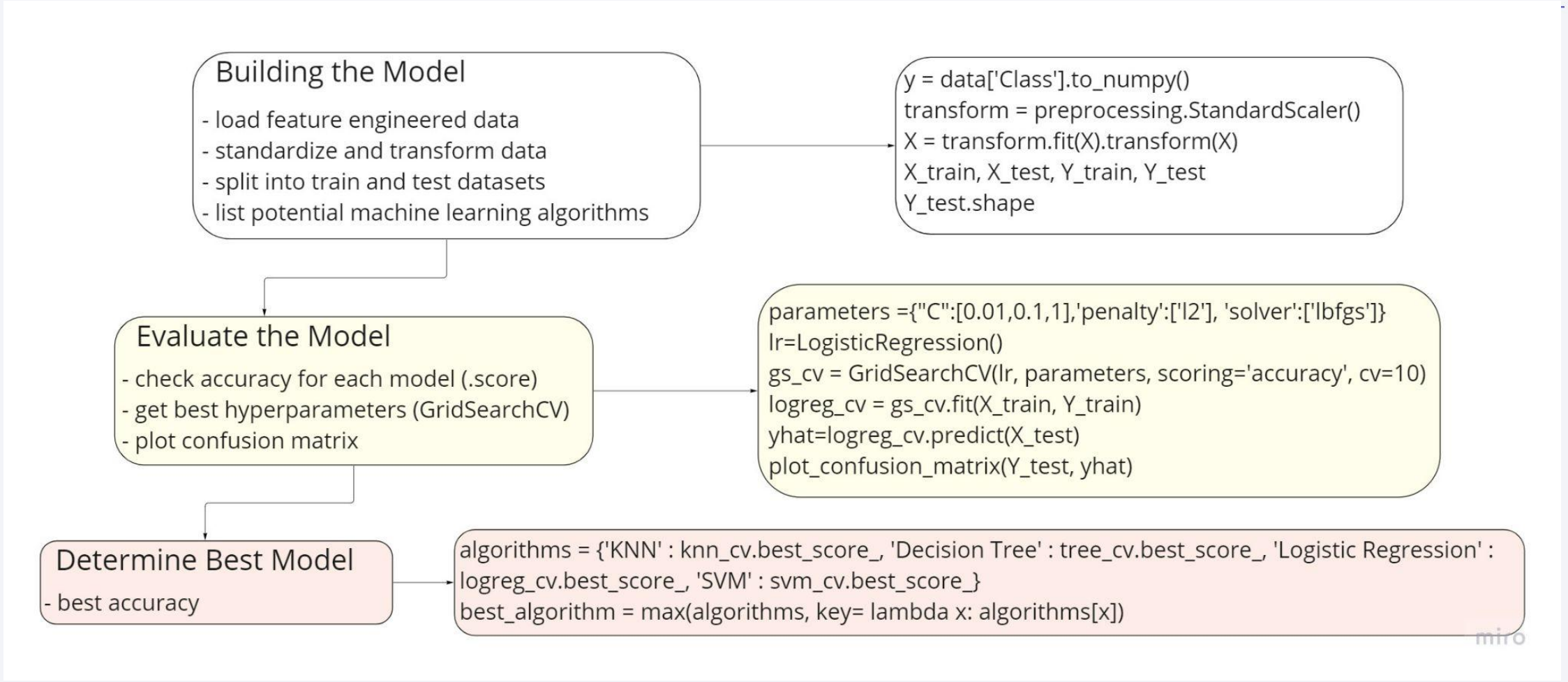
Folium Code GitHub

BUILD A DASHBOARD WITH PLOTLY DASH

Plotly Dash GitHub

DASHBOARD	CODE	RESULT
Pie Chart	px.pie()	success rate
Scatter Plot	px.scatter()	variable correlation
Dash & Components	import dash import dash_html_components as html import dash_core_components as dcc from dash.dependencies import Input, Output	python framework created by plotly in-depth data analysis and visualizations
Pandas	import pandas as pd	getting data and dataframe
Plotly	import plotly.express as px	plot interactive graphs
Dropdown	dcc.Dropdown	dropdown list for launch site
RangeSlider	dcc.RangeSlider()	rangeslider for payload mass

PREDICTIVE ANALYSIS (CLASSIFICATION)



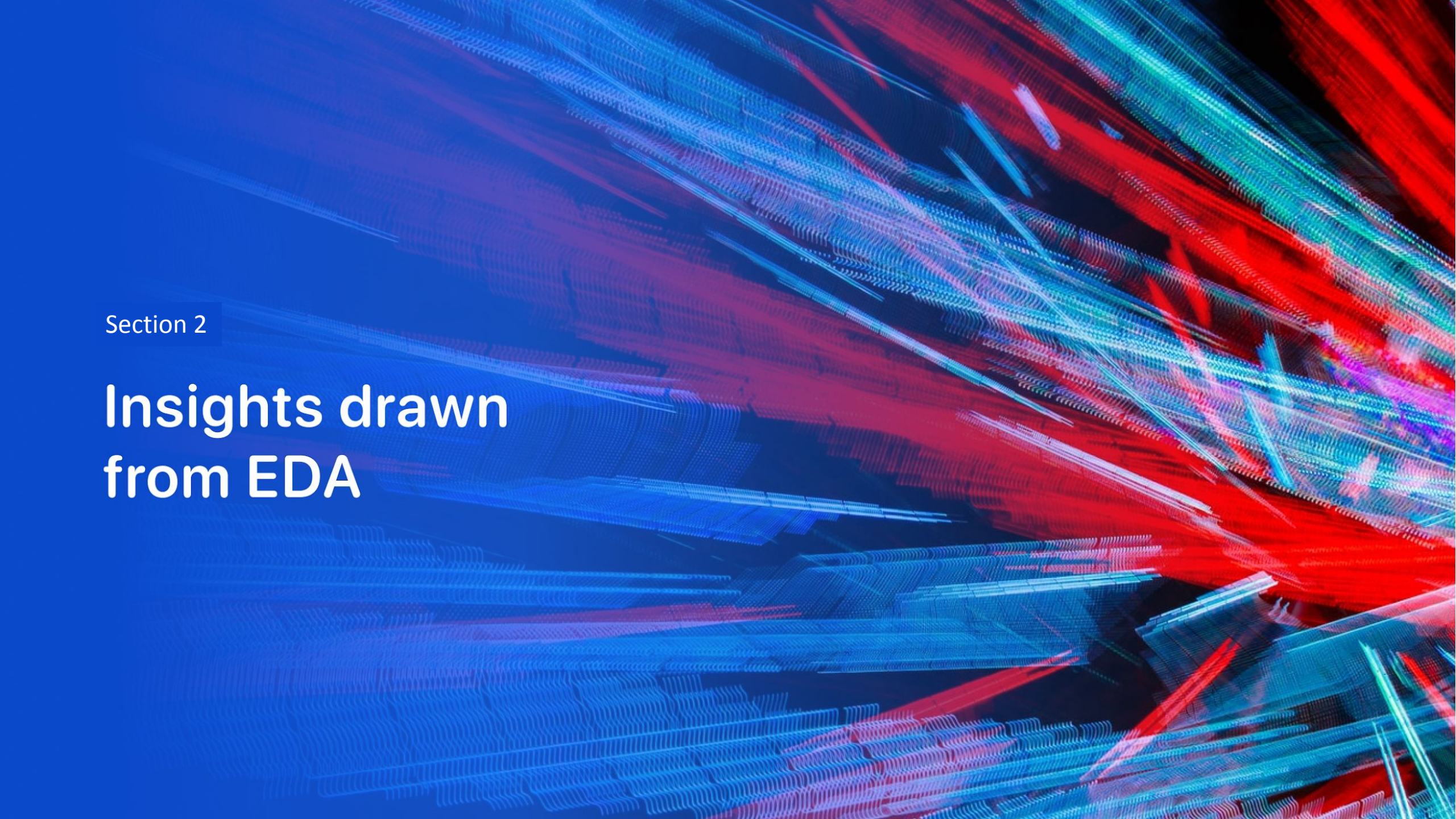
[Predictive Analysis GitHub]

RESULTS

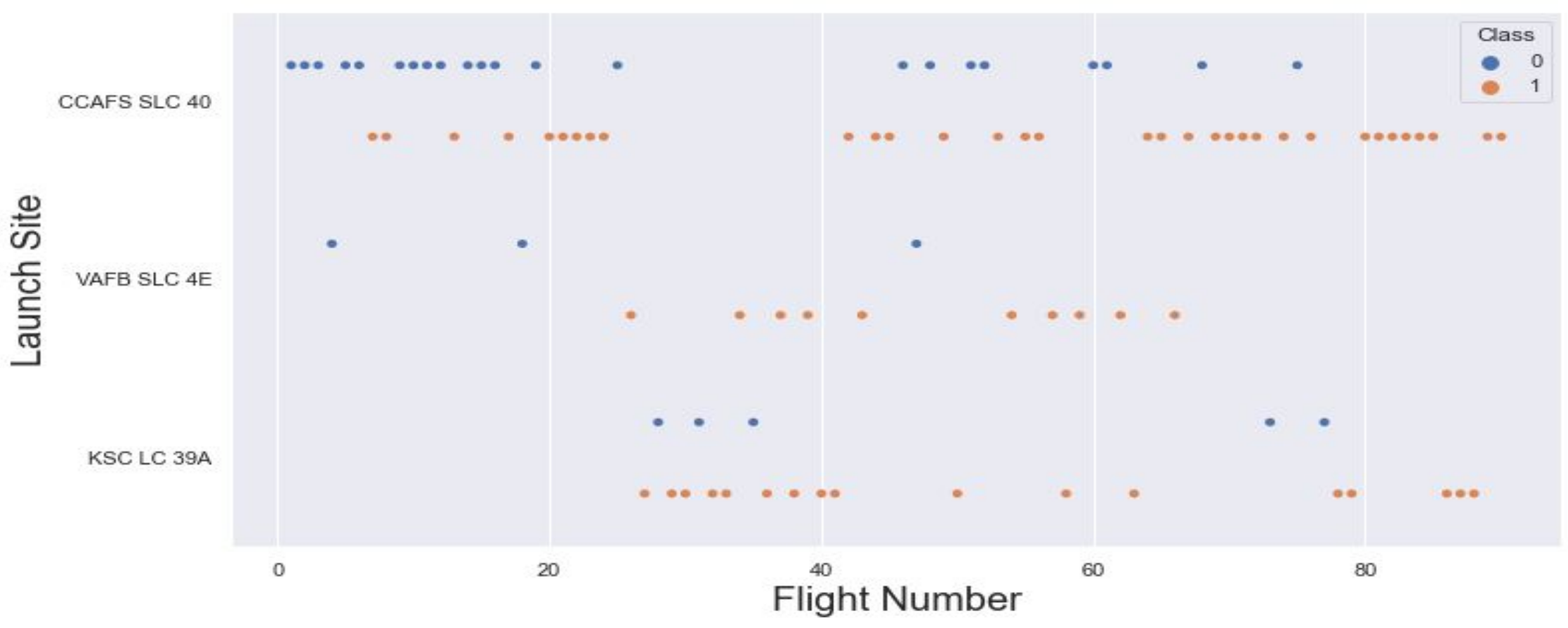
- **Insights Drawn From EDA**
  - Qualitative
  - Quantitative
- **Launch Sites Proximities Analysis**
  - with Folium
- **Build a Dashboard**
  - with Plotly Dash
- **Predictive Analysis (Classification)**
  - Accuracy
  - Confusion Matrix

Section 2

Insights drawn from EDA

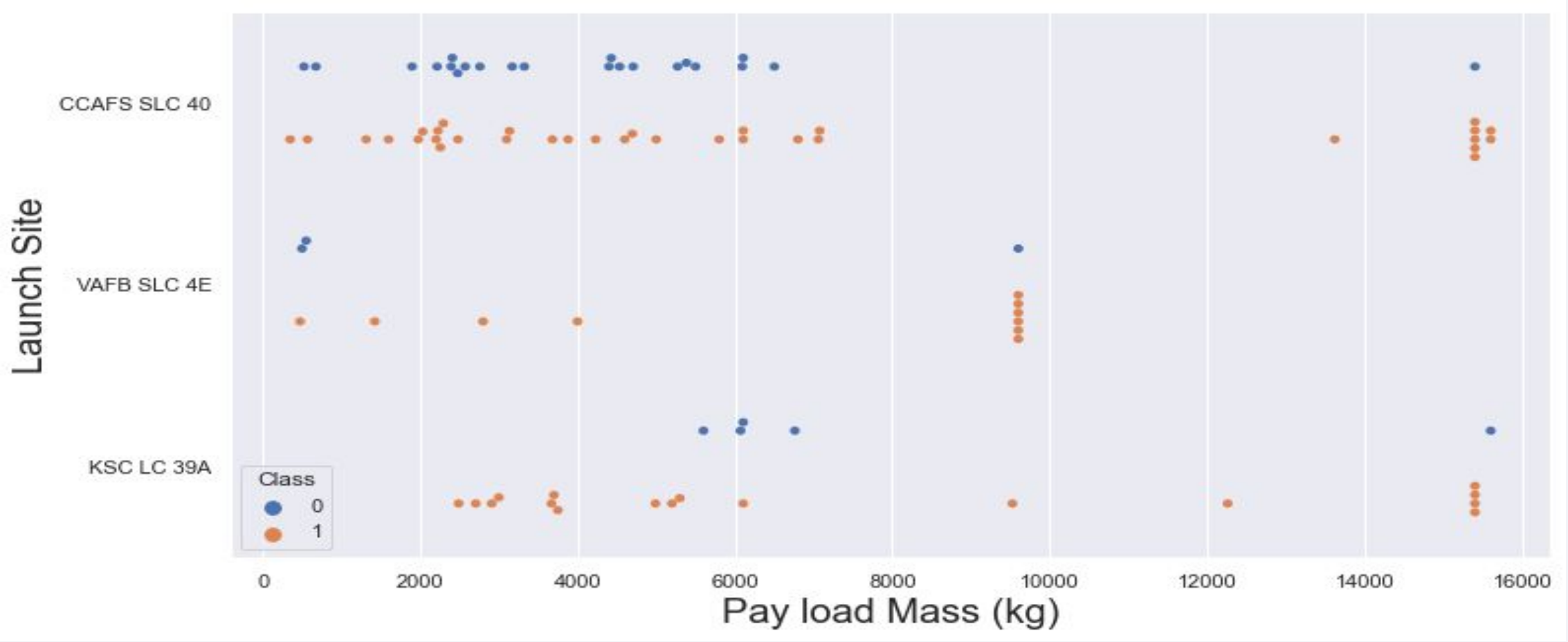


FLIGHT NUMBER VS. LAUNCH SITE



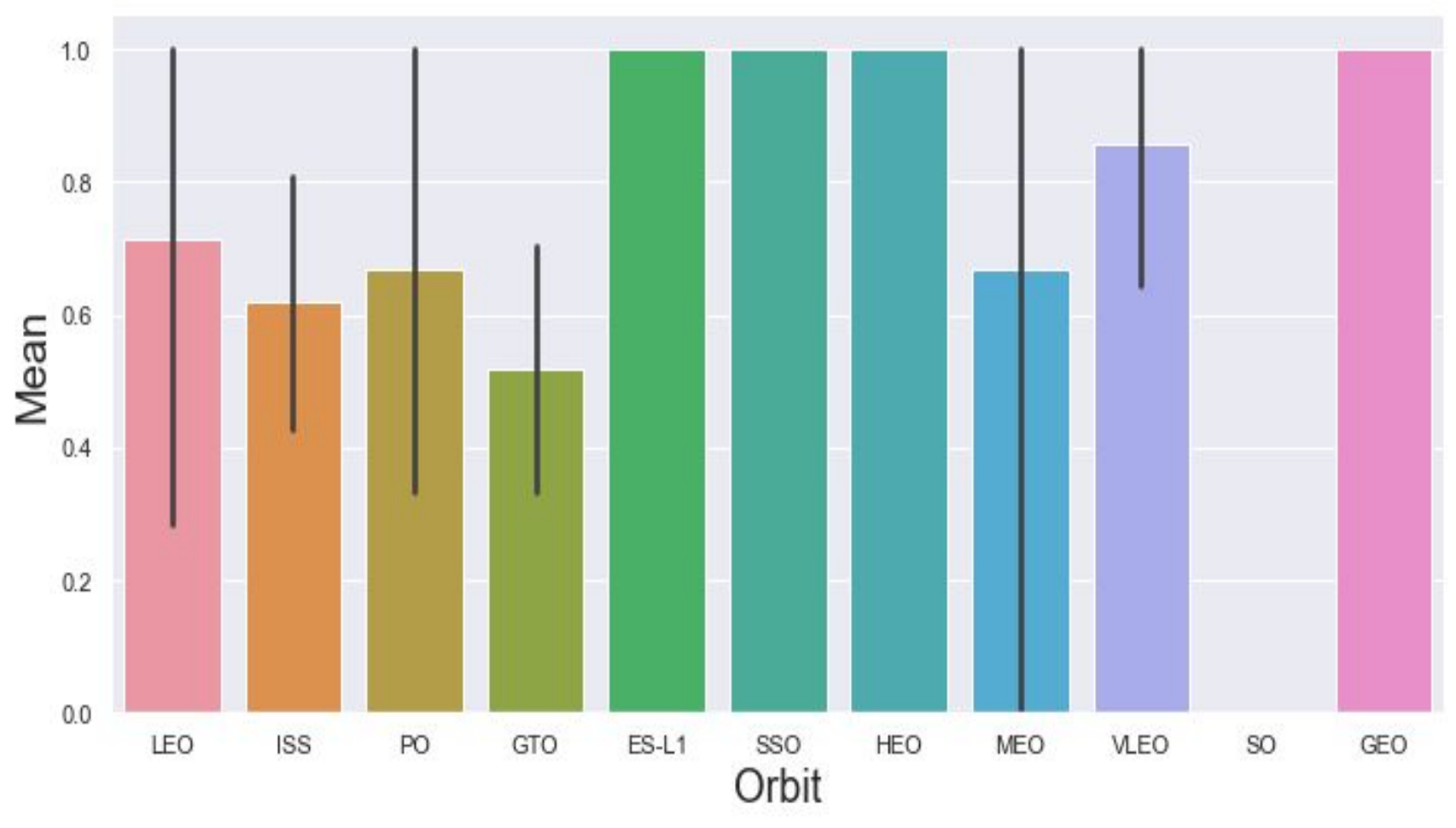
The recovery success rate increases steadily after flight 30 with a high recovery rate from flight 36 and on.

PAYLOAD VS. LAUNCH SITE



With a payload mass over 7000 kg, we have a much better success rate.

SUCCESS RATE VS. ORBIT TYPE



Orbit types with perfect success rate:

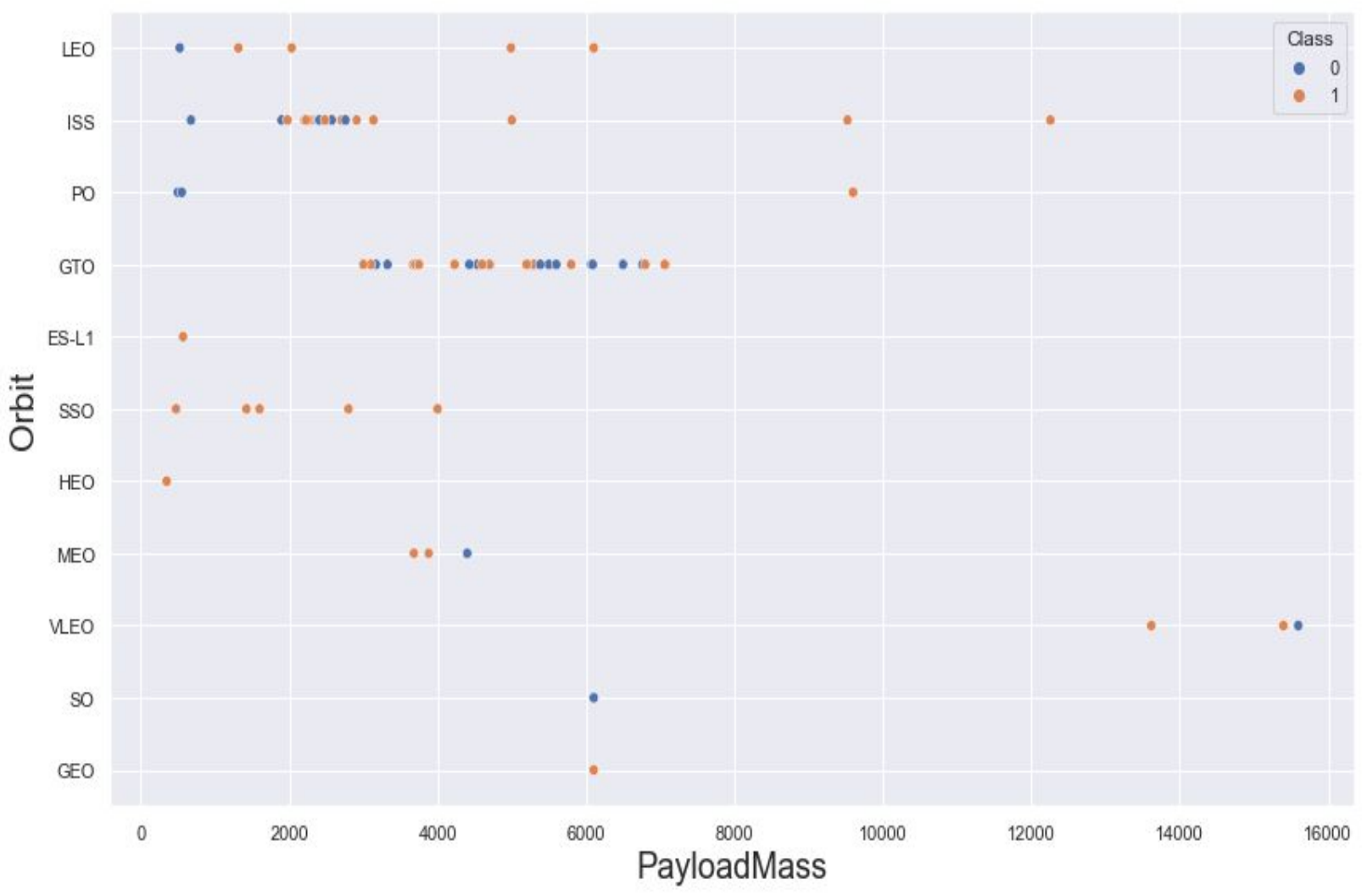
- ES-L1
- SSO
- HEO
- GEO

But only SSO has enough data for a significance

FLIGHT NUMBER VS. ORBIT TYPE

- LEO, GTO improved with flight number
- SSO perfect over flights
- VLEO occurrences increases with flight number

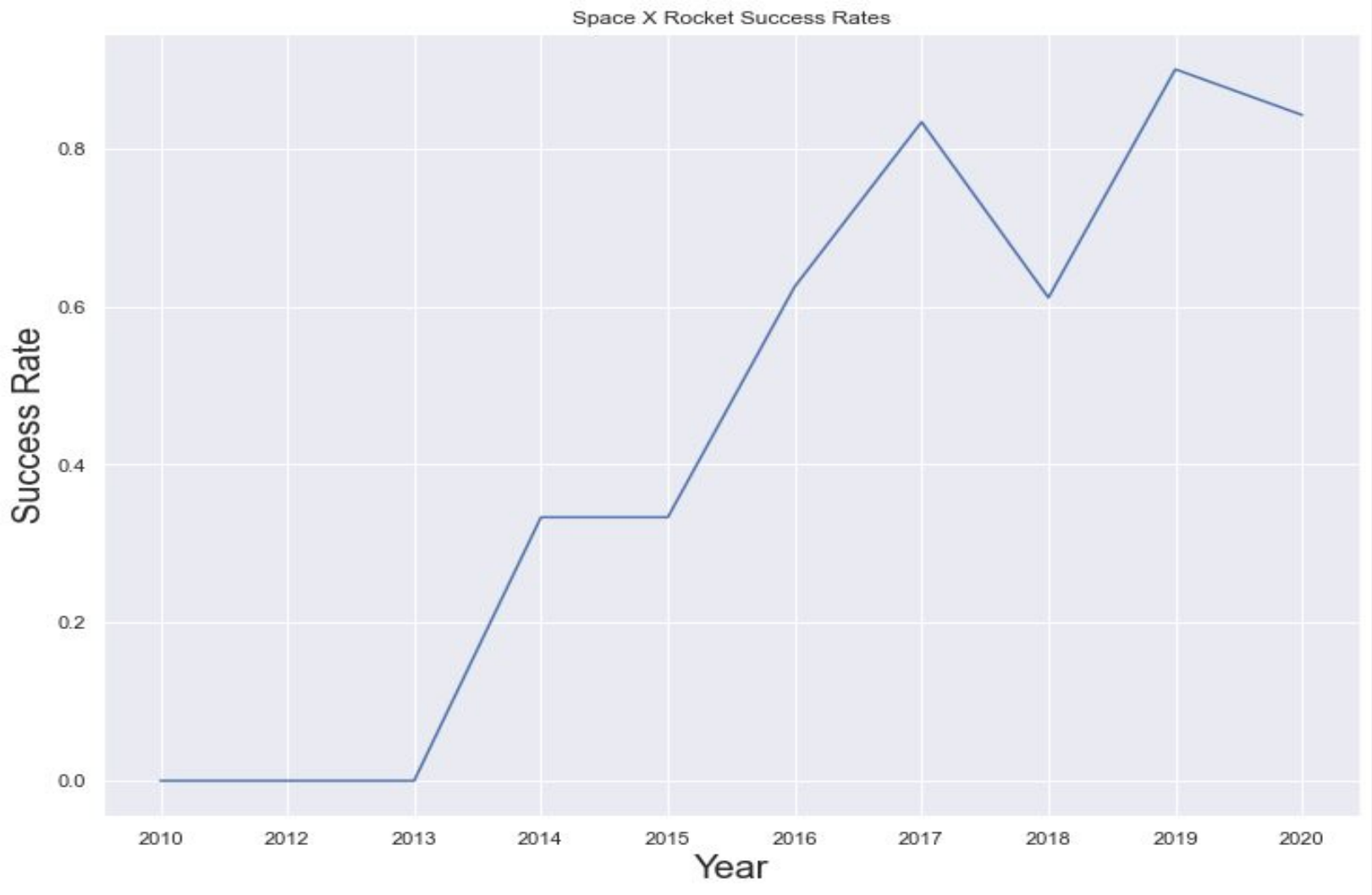
PAYLOAD VS. ORBIT TYPE



- SSO is only used for low payloads
- VLEO is only used for high payloads
- LEO, PO is better as payload increases
- GTO is random
- ISS works great with payload over 3000 kg

We need more data to validate some of these observations

LAUNCH SUCCESS YEARLY TREND



Overall, the success rate increased from 2013 on with 2018 being the biggest decrease in success rate.

ALL LAUNCH SITE NAMES

%sql SELECT DISTINCT launch_site FROM spacex

This will return each distinct launch site from the SpaceX data.

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

LAUNCH SITE NAMES BEGINNING WITH ‘KSC’

%sql SELECT launch_site FROM spacex
WHERE launch_site LIKE 'KSC%' limit 5;

This will return the first 5 values where the LaunchSite begins with KSC.

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	Reus
26	27	2017-02-19	Falcon 9	2490.000000	ISS	KSC LC 39A	True RTLS	1	True	False	True	5e9e3032383ecb267a34e7c7	3.0	
27	28	2017-03-16	Falcon 9	5600.000000	GTO	KSC LC 39A	None None	1	False	False	False	NaN	3.0	
28	29	2017-03-30	Falcon 9	5300.000000	GTO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3032383ecb6bb234e7ca	2.0	
29	30	2017-05-01	Falcon 9	6104.959412	LEO	KSC LC 39A	True RTLS	1	True	False	True	5e9e3032383ecb267a34e7c7	3.0	
30	31	2017-05-15	Falcon 9	6070.000000	GTO	KSC LC 39A	None None	1	False	False	False	NaN	3.0	

TOTAL PAYLOAD MASS

%sql SELECT sum(payload_mass) AS NASA payload_mass Total FROM spacex
WHERE customer = 'NASA (CRS)';

This will give us the sum of the the payload mass in kg from all NASA (CRS) entries

NASA payload_mass Total
22007

AVERAGE PAYLOAD MASS BY F9 v.1.1

%sql SELECT avg(payload_mass) AS avg_payload_mass F9 v1.1 FROM spacex
WHERE BoosterVersion = ‘F9 v1.1’;

This will give us the average payload mass where the booster version is F9 v1.1

avg_payload_mass F9 v1.1
2928

FIRST SUCCESSFUL GROUND LANDING DATE

%sql SELECT min(DATE) AS First Success Ground Pad FROM spacex
WHERE Outcome = 'Success (ground pad)';

This will give us the first date where there was a successful ground pad landing.

First Success Ground Pad
2015-12-22

SUCCESSFUL DRONE SHIP LANDING WITH PAYLOAD BETWEEN 4000 AND 6000

%sql SELECT booster_version FROM spacex WHERE Outcome = 'Success (drone ship)' AND payload_mass BETWEEN 4000 and 6000;

This will give us results for all booster versions the had a Success (drone ship) and having a payload mass between 4000 and 6000 kgs.

booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

TOTAL NUMBER OF SUCCESSFUL AND FAILURE MISSION OUTCOMES

%sql SELECT COUNT(Outcome) AS Mission Outcome FROM spacex
GROUP BY Outcome;

This will give us the number of each outcome.

Mission Outcome
106
1

BOOSTERS CARRIED MAXIMUM PAYLOAD

%sql SELECT booster_version AS booster_version Max
FROM spacex
WHERE payload_mass = (SELECT max(payload_mass)
FROM spacex;

This will give us the booster version with a maximum payload mass.

booster_version Max
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
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

2015 LAUNCH RECORDS

%sql SELECT booster_version, launch_site, year(DATE) FROM spacex
WHERE extract(YEAR FROM DATE) = '2015' AND Outcome = 'Failure (drone ship)';

This will give us the booster_version and launch_site from the year 2015 where there was a Failure (drone ship) outcome.

booster_version	launch_site	3
F9 v1.1 B1012	CCAFS LC-40	2015
F9 v1.1 B1015	CCAFS LC-40	2015

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

%sql SELECT Outcome, COUNT(Outcome) AS total from spacex
WHERE outcome LIKE 'Success%' AND date BETWEEN
'2010-06-04' AND '2017-03-20'
GROUP BY outcome
ORDER BY COUNT(outcome) desc;

THis will give is the outcome and the number of occurrences for dates between 2010-06-04 and 2017-03-20 and it sort the results the outcome and the number of occurrences and display them in descending order.

outcome	total
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

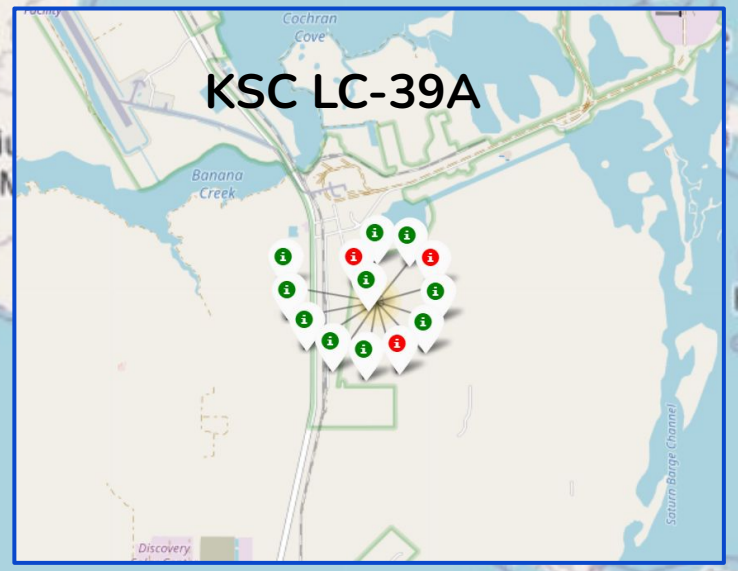
LAUNCH SITES

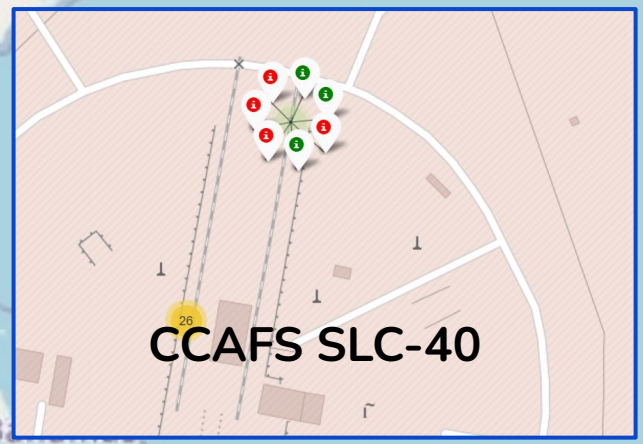
[Folium GitHub]

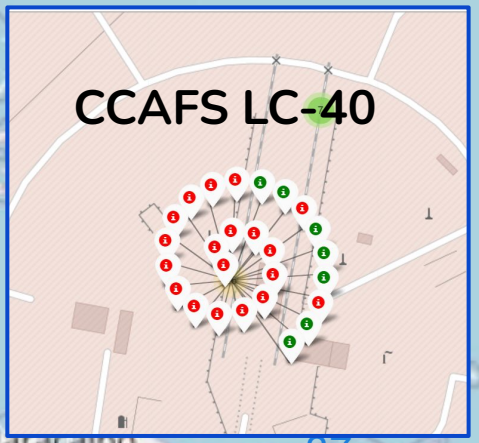
We can see that the launch sites are near the coast in California and Florida

LAUNCH OUTCOMES

[Folium GitHub]

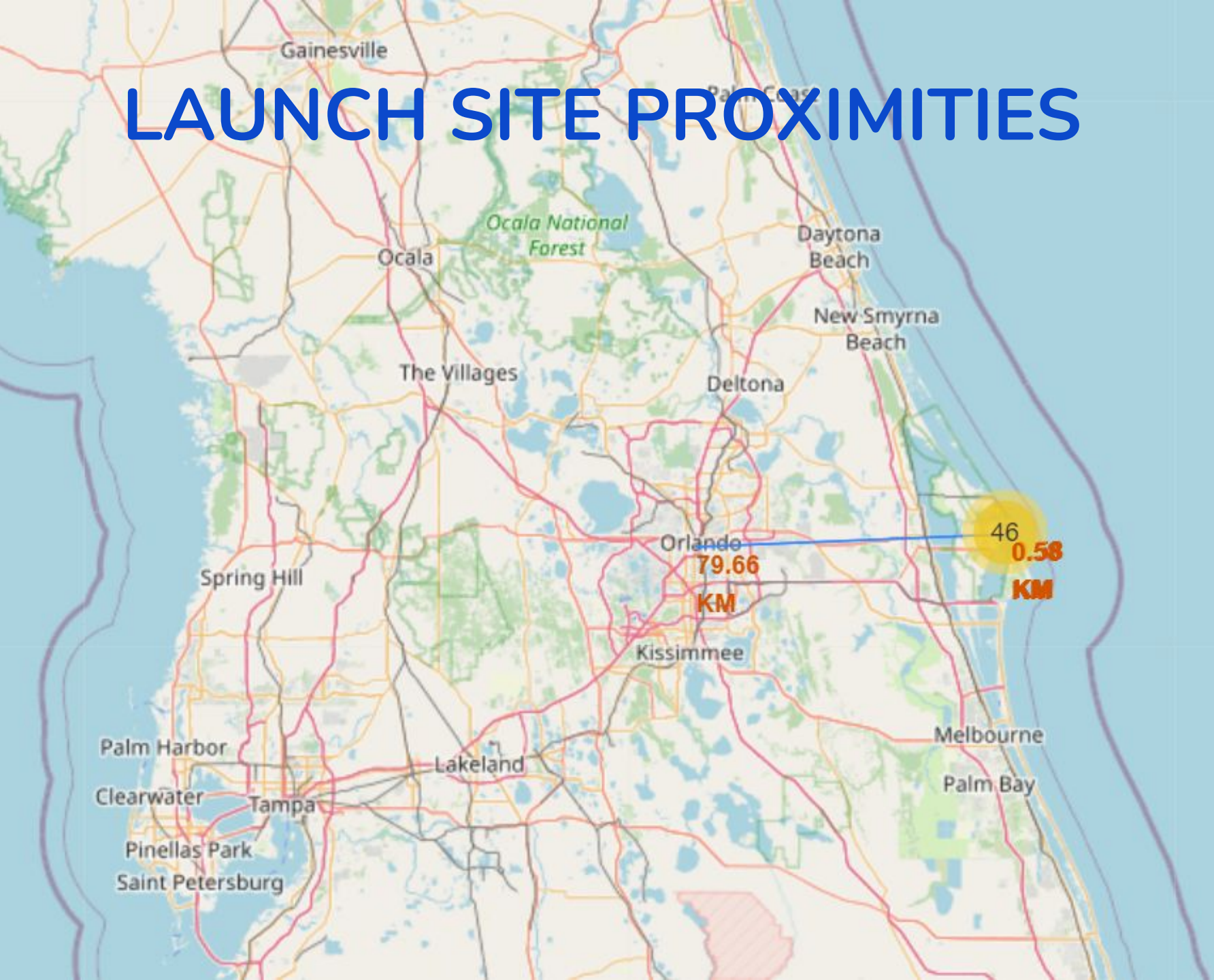






KSC LC-39A has the highest good outcome probability

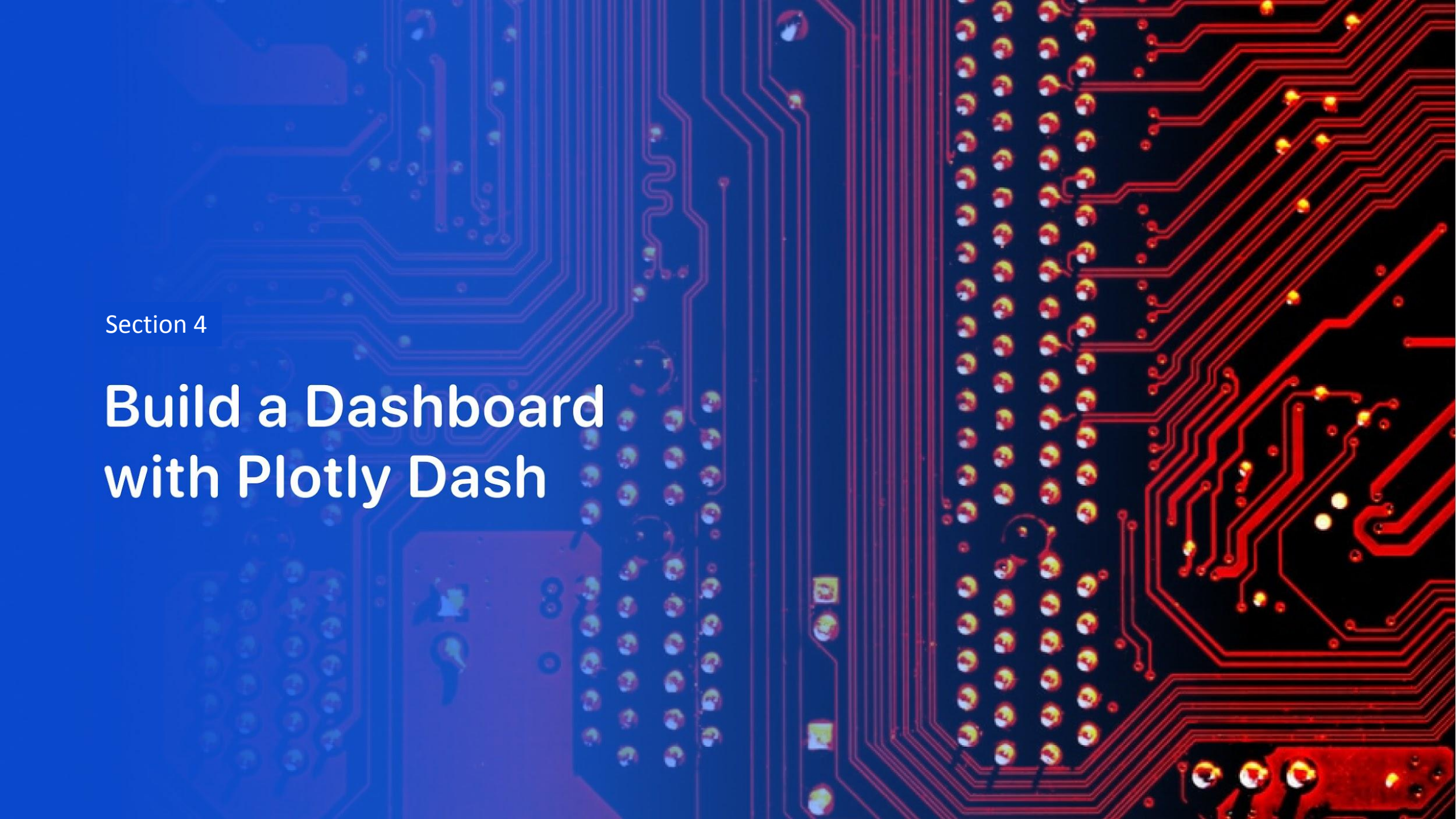
LAUNCH SITE PROXIMITIES



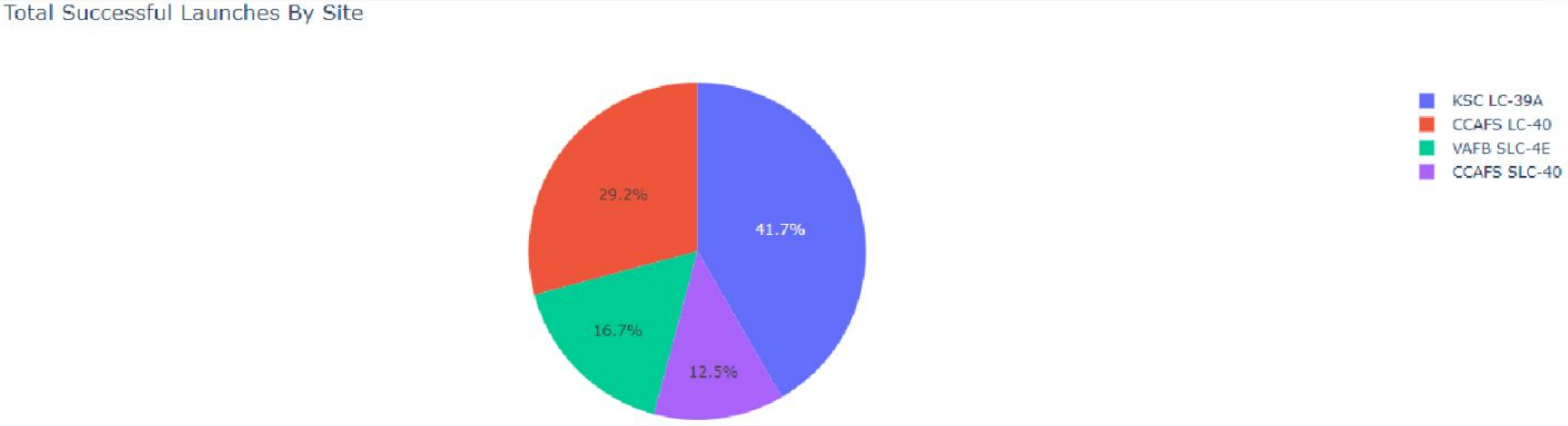
[Folium GitHub]

0.58 KM - distance to coast from launch site

79.66 KM - distance to Orlando from launch site


Section 4
Build a Dashboard with Plotly Dash

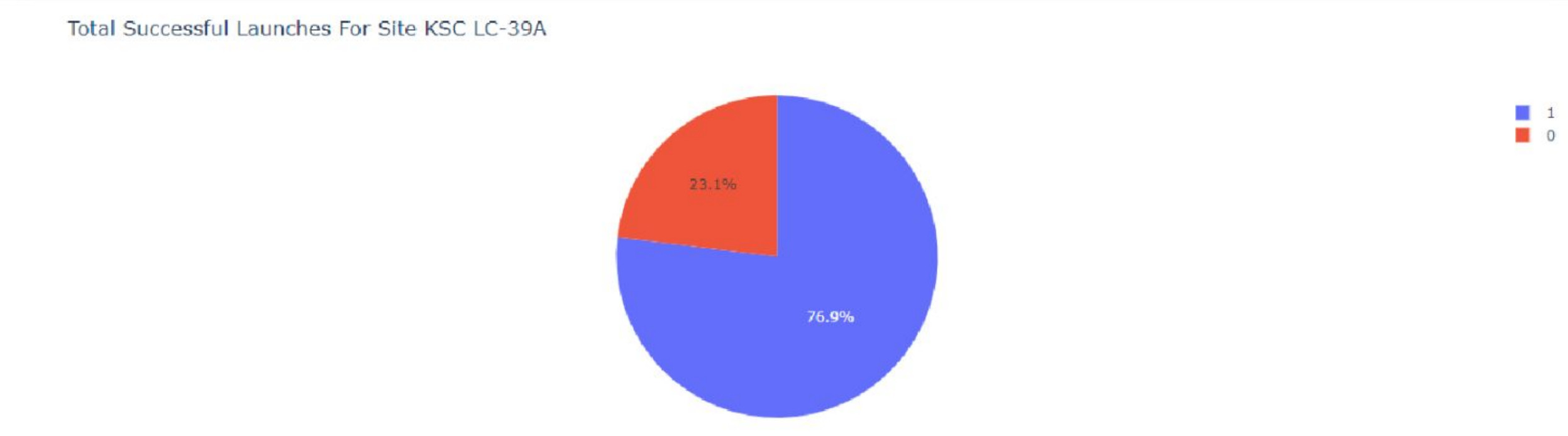
LAUNCH SUCCESS CHART



KSC LC-39A has the most successful launches at 41.7%

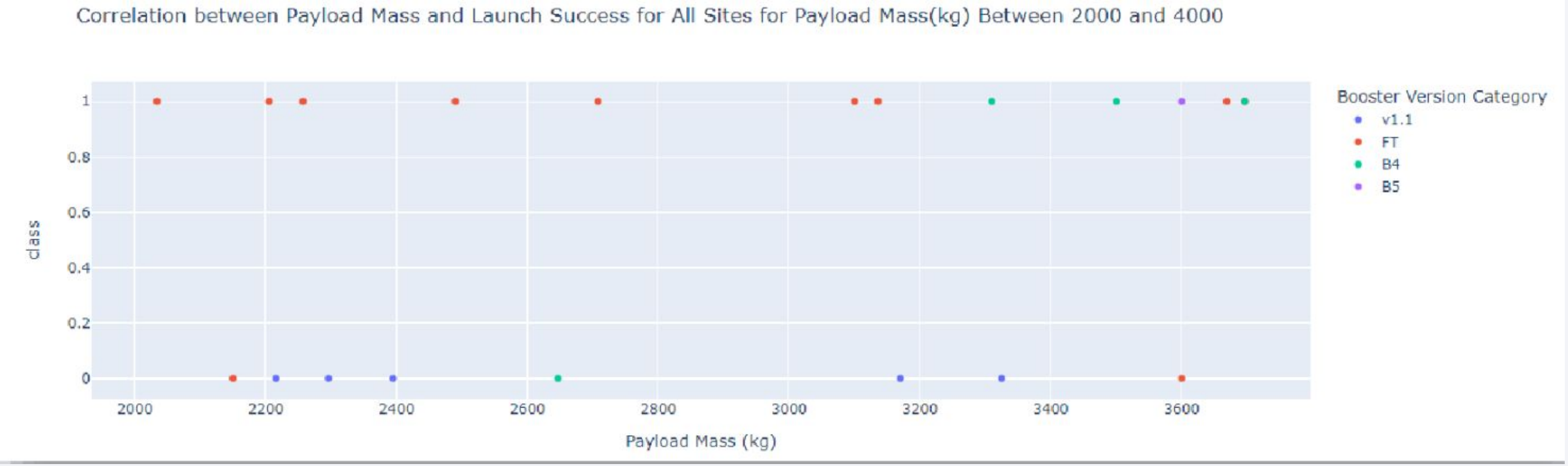
CCAFS SLC-40 has the least successful launches at 12.5%

KSC LC-39A SUCCESS CHART



KSC LC-39A had the most successful launches and achieved these with a success rate of 76.9%

PAYLOAD MASS vs SUCCESS RATE

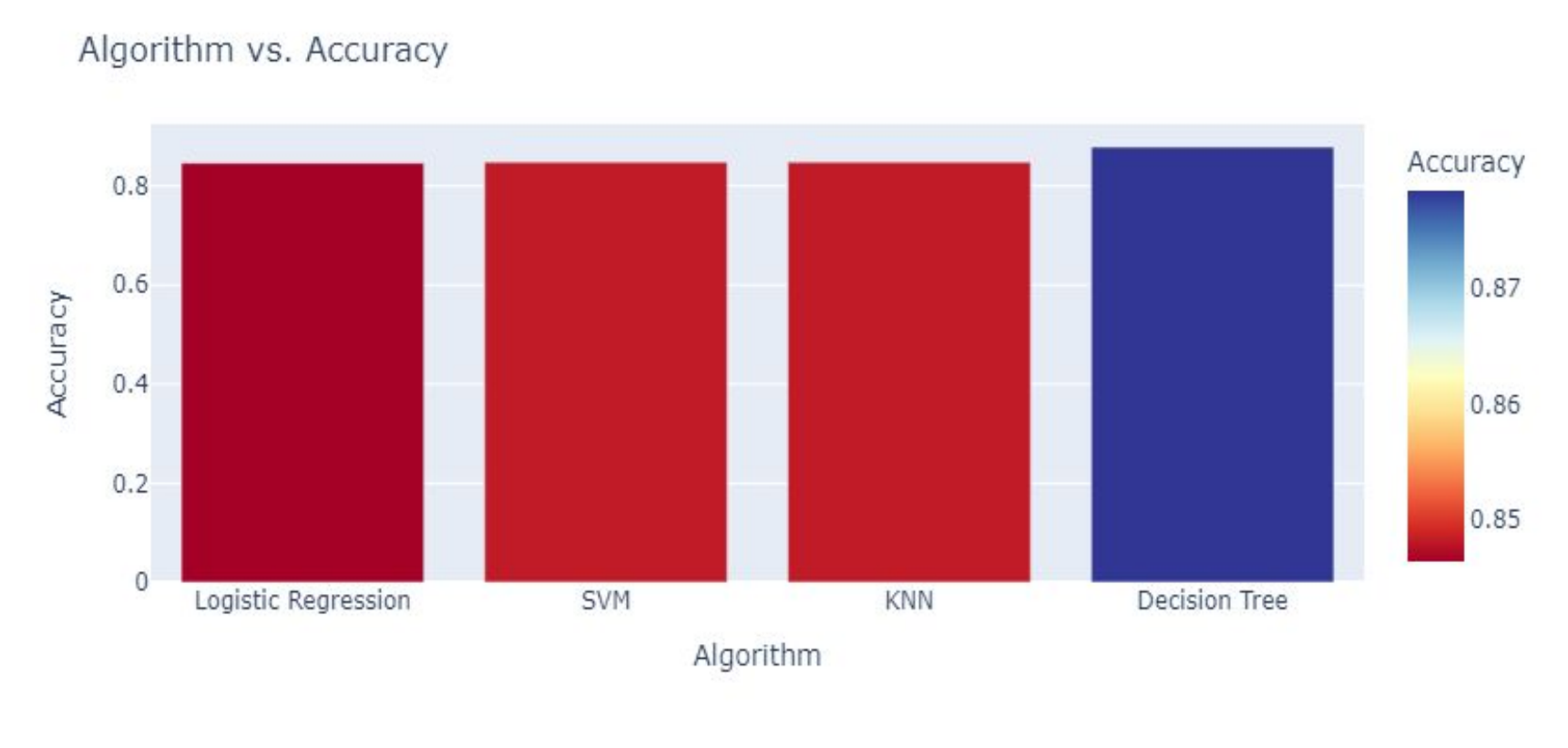


Between 2000 kg and 4000 kg had the highest success rates with FT booster version producing the most successful launches.

Section 5

Predictive Analysis (Classification)

CLASSIFICATION ACCURACY



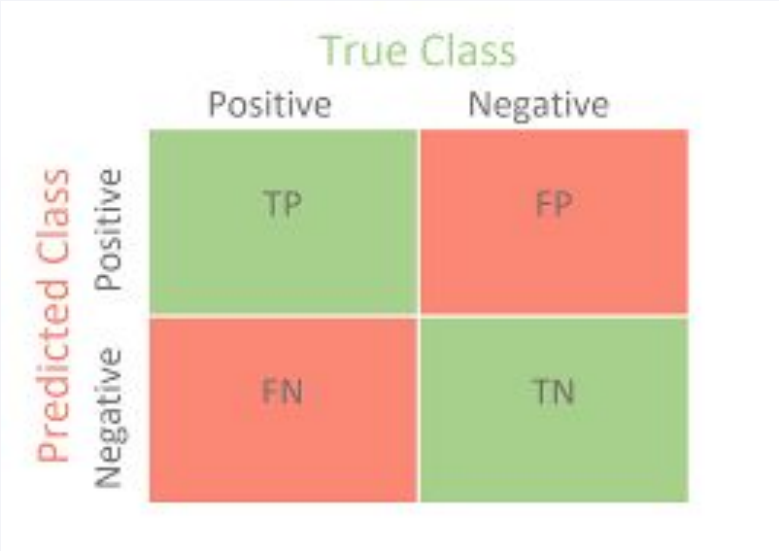
Algorithm	Accuracy
Logistic Regression	0.846429
SVM	0.848214
KNN	0.848214
Decision Tree	0.901786

The Decision Tree has the highest accuracy at 90.1786%

CONFUSION MATRIX

The Decision Tree model was the best

- 5 true positives
- 1 false positive
- 9 true negatives
- 3 false negatives



Conclusions

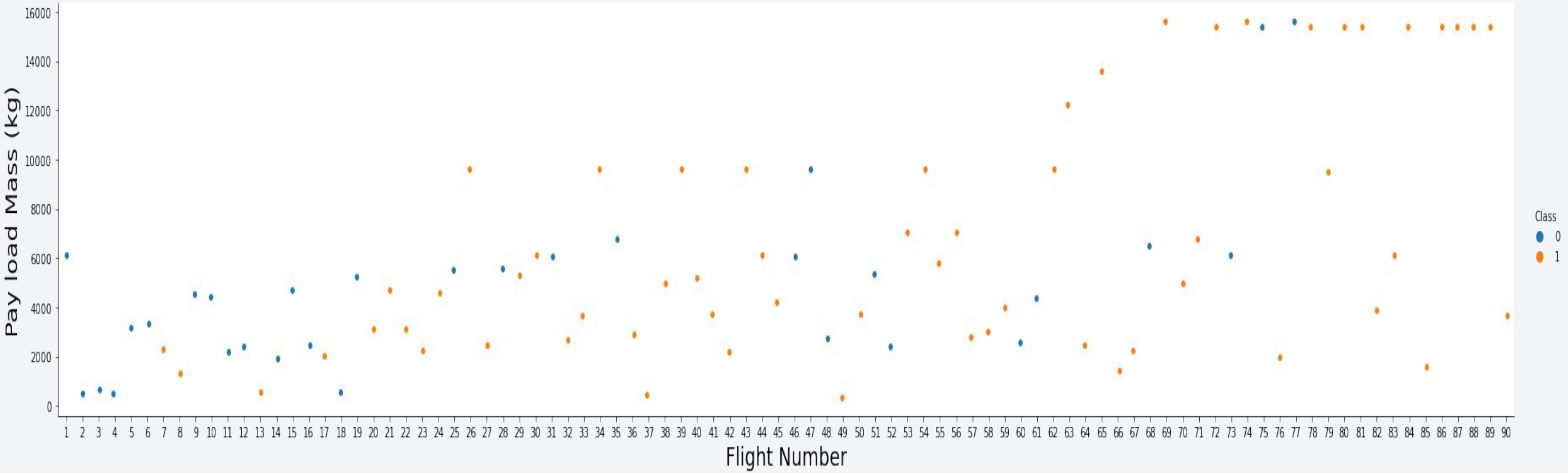
- Orbits ES-L1, SSO, HEO, GEO had the highest success rates.
- SpaceX launches have increased their overall success rates since 2013.
- KSC LC-39A had the most successful launches with a success rate of 76.9%.
- The Decision Tree Classifier had the highest accuracy rate of 0.878571.

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

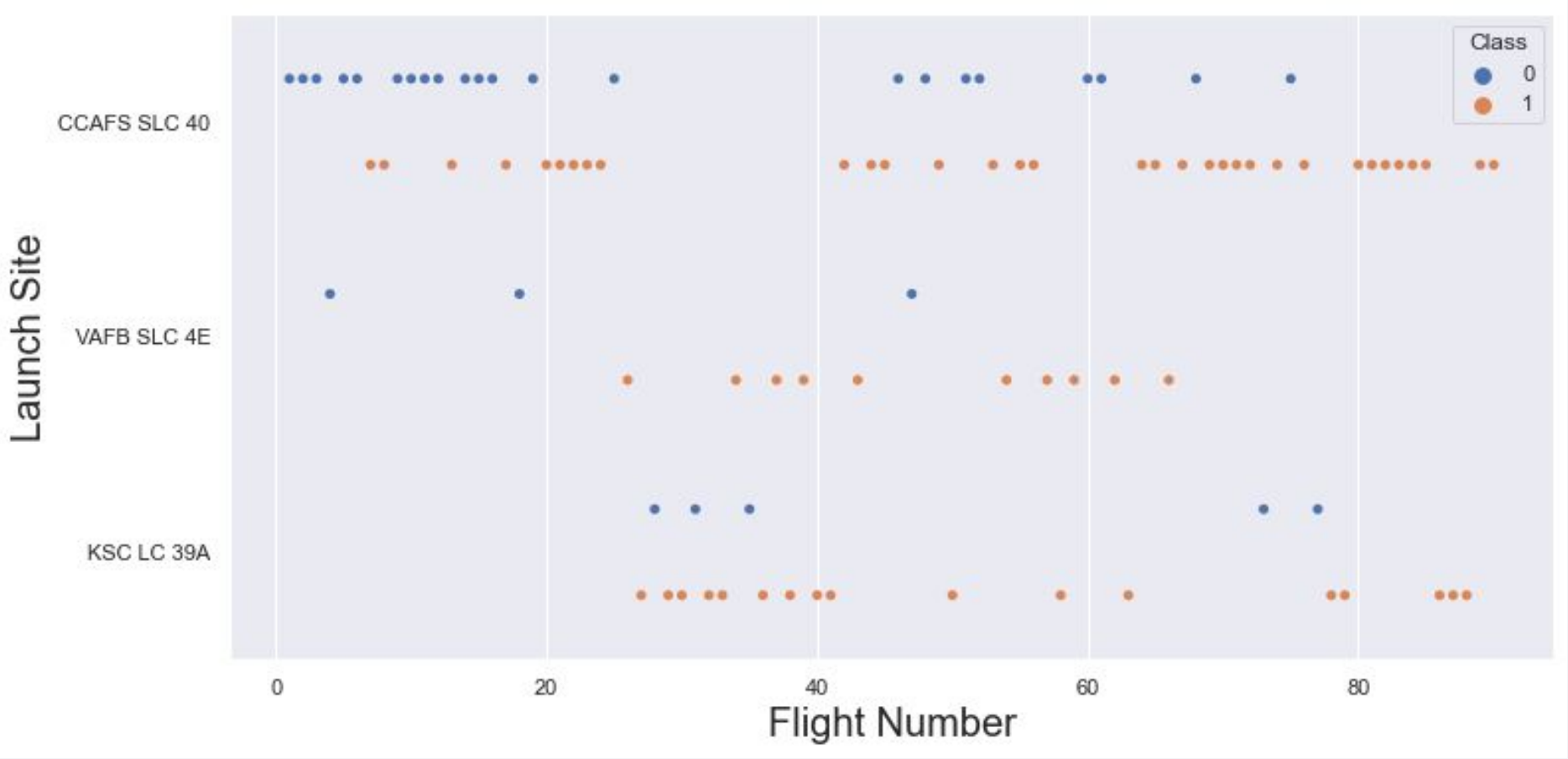
APPENDIX

Payload Mass vs Flight Number successful/unsuccessful launches



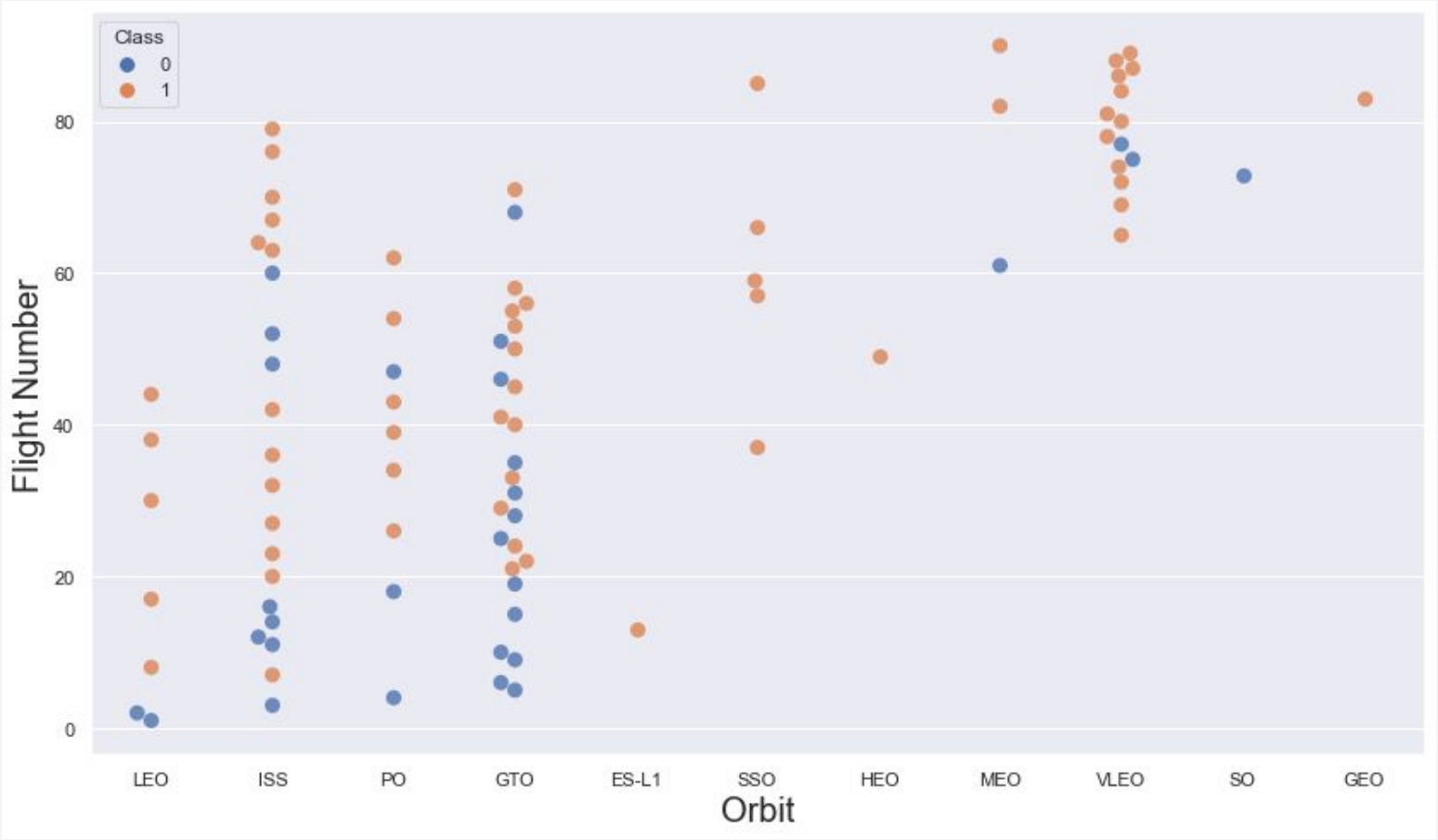
APPENDIX

Launch Site vs Flight Number successful/unsuccessful launches



APPENDIX

Flight Number vs Orbit successful/unsuccessful launches



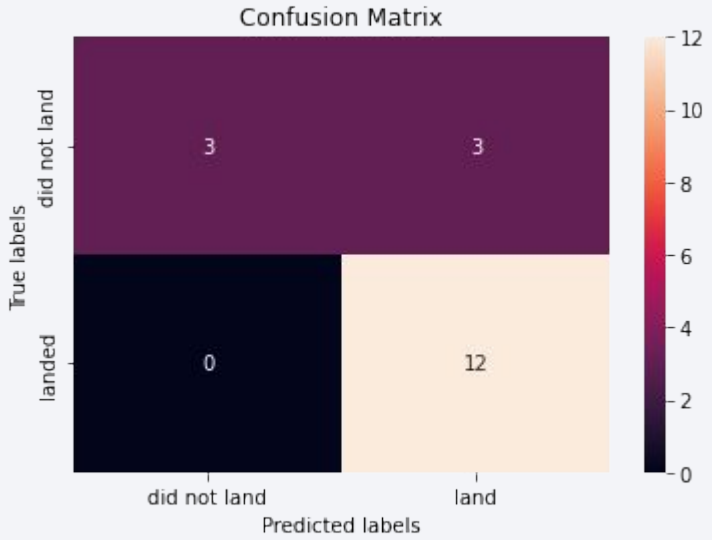
APPENDIX

Using `get_dummies` and `features` dataframe to apply OneHotEncoder to the column `Orbits`, `LaunchSite`, `LandingPad`, and `Serial`.

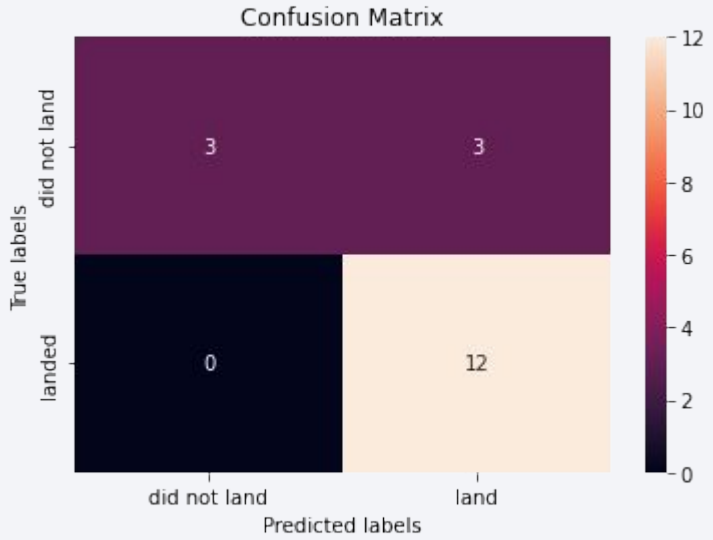
```
features_one_hot = features

features_one_hot = pd.concat([features_one_hot,pd.get_dummies(df['Orbit'])], axis=1)
features_one_hot.drop(['Orbit'], axis = 1,inplace=True)

features_one_hot = pd.concat([features_one_hot,pd.get_dummies(df['LaunchSite'])], axis=1)
features_one_hot.drop(['LaunchSite'], axis = 1,inplace=True)

features_one_hot = pd.concat([features_one_hot,pd.get_dummies(df['LandingPad'])], axis=1)
features_one_hot.drop(['LandingPad'], axis = 1,inplace=True)

features_one_hot = pd.concat([features_one_hot,pd.get_dummies(df['Serial'])], axis=1)
features_one_hot.drop(['Serial'], axis = 1,inplace=True)

features_one_hot.head()
```

APPENDIX

Logistic Regression



SVM



KNN

APPENDIX

```
algo_df = pd.DataFrame.from_dict(algorithms, orient='index', columns=['Accuracy'])
algo_df.sort_values(['Accuracy'], inplace=True)
algo_df
```

	Accuracy
Logistic Regression	0.846429
SVM	0.848214
KNN	0.848214
Decision Tree	0.901786

Thank you!