Winning Space Race with Data Science

Federico Guzzo
07-11-2022

Outline

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

Executive Summary

- **Summary of methodologies**
- Data collection via API, Web Scraping
- Exploratory Data Analysis (EDA) with Data Visualization
- EDA with SQL
- Interactive MAP with Folium
- Predictive Analysis
- **Summary of all results**
- Exploratory Data Analysis result
- Interactive maps and dashboard
- Predictive results

Introduction

- Project background and context

The aim of this project is to predict if the Falcon 9 first stage will successfully land. Space says on its website that the Falcon 9 rocket launch cost 62 million dollars. Other providers cost upward of 165 million dollars each. The price difference is explained by the fact that Space can reuse the first stage. By determining if the stage will land, we can determine the cost of a launch. This information is interesting for another company if it wants to compete with Space for a rocket launch.

- Problems you want to find answers

What are the main characteristics of a successful or failed landing?

What are the effects of each relationship of the rocket variables on the success or failure of a landing?

What are the conditions which will allow SpaceX to achieve the best landing success rate?

Section 1

Methodology

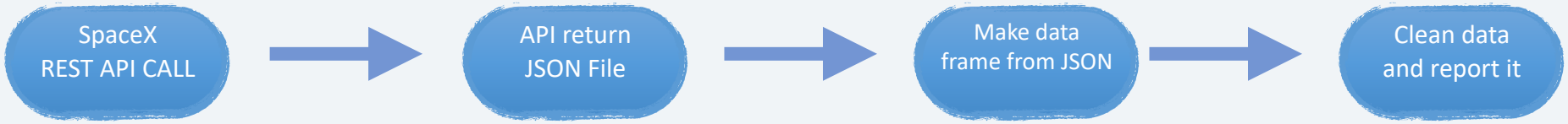
Methodology

Executive Summary

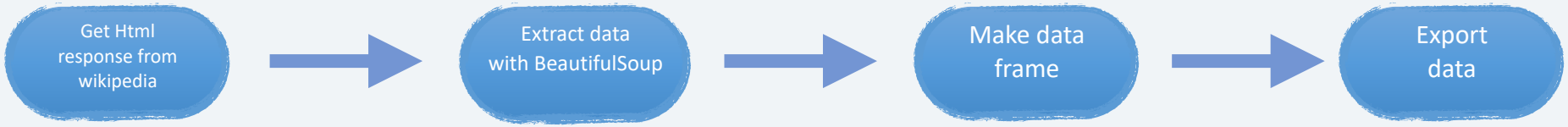
- Data collection methodology:
  - SpaceX REST API
  - Web Scraping from wikipedia
- Perform data wrangling
  - Dropping columns unnecessary
  - One hot encoding for classification model
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- **Dataset are collected from Rest SpaceX API and web scraping wikipedia.**
- The information obtained by the API are rocket, launches, payload information.
- The spaceX REST API URL —> https://api.spacexdata.com/v4



- The information obtained by the web scraping of the wikipedia are launches, landing, Payload information
- URL —> https://en.wikipedia.org/w/index.php?title=List of Falcon 9 and Falcon Heavy launches&oldid=1027686922



Data Collection – SpaceX API

1 - Getting response from API

```
In [34]: spacex_url="https://api.spacexdata.com/v4/launches/past"
In [35]: response = requests.get(spacex_url)
```

2 - convert response to JSON file

```
In [38]: data = pd.json_normalize(response.json())
```

3 - Transform Data

```
# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)

# Call getBoosterVersion
getBoosterVersion(data)
```

4 - Create dictionary with data

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

5 - Create dataframe

```
# Create a data from launch_dict
data = pd.DataFrame(launch_dict)
```

6 - Filter dataframe

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data[data.BoosterVersion == 'Falcon 9']
data_falcon9
```

7 - Export to file csv

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

1 - Getting Response from HTML

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

2 - Create and clean Object

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows tha
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the fea
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

3 - Create list

```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

4 - Create dataframe and show data

```
# Create a data from launch_dict
data = pd.DataFrame(launch_dict)
```

Show the summary of the dataframe

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

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Se
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merli
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merli
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merli
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merli
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0

5 - export file to csv

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Wrangling

- In the dataset, there are several cases where the booster did not land successfully.
- True Ocean, True RTLS, True ASDS means the mission has been successful.
- False Ocean, False RTLS, False ASDS means the mission was a failure.

We need to transform string variables into categorical variables where 1 means the mission has been successful and O means the mission was a failure.

1 - Calculate launches number for each site

```
# Apply value_counts() on column LaunchSite
df.LaunchSite.value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

2 - Calculate the number and occurence of each orbit

```
# 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
```

3 - Calculate number and of mission outcome per orbit type

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

```
True ASDS       41
None None       19
True RTLS       14
False ASDS       6
True Ocean       5
False Ocean      2
None ASDS        2
False RTLS       1
Name: Outcome, dtype: int64
```

4 - Create a landing outcome label from Outcome column

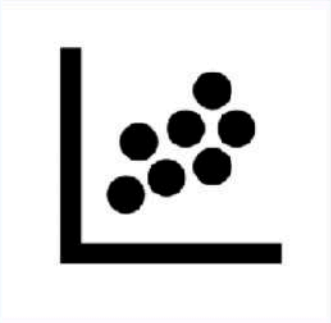
```
In [31]: landing_class = []

         for key, value in df['Outcome'].items():
             if value in bad_outcomes:
                 landing_class.append(0)
             else:
                 landing_class.append(1)
```

5 - Export to file

```
df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Data Visualization

- Scatter Graphs
  - Flight Number vs. Payload Mass
  - Flight Number vs. Launch Site
  - Payload vs. Launch Site
  - Orbit vs. Flight Number
  - Payload vs. Orbit Type
  - Orbit vs. Payload Mass



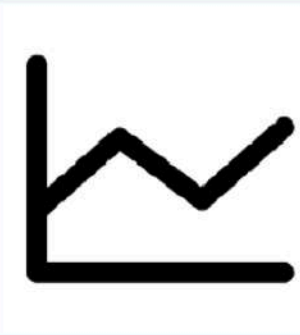
Scatter plots show relationship between variables. This relationship is called the correlation.

- Bar Graph
  - Success rate vs. Orbit

Bar graphs show the relationship between numeric and categoric variables.

- Line Graph
  - Success rate vs. Year

Line graphs show data variables and their trends. Line graphs can help to show global behavior and make prediction for unseen data.



EDA with SQL

- We performed SQL queries to gather and understand data from dataset:
  - Displaying the names of the unique lauunch sites in the space mission.
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS).
  - Display average payload mass carried by booster version F9 v1.1.
  - List the date when the first successful landing outcome 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.
  - List the records which will display the month names, faiilure landing_ouutcomes in drone ship, booster versions, launch_site for the months in year 2015.
  - Rank the count of successful landiing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

Build an Interactive Map with Folium

- Folium map object is a map centered on NASA Johnson Space Center at Houson, Texas
  - Red circle at NASA Johnson Space Center's coordinate with label showing its name *(folium.Circle, folium.map.Marker).*
  - Red circles at each launch site coordinates with label showing launch site name *(folium.Circle, folium.map.Marker, folium.features.DivIcon).*
  - The grouping of points in a cluster to display multiple and different information for the same coordinates *(folium.plugins.MarkerCluster).*
  - Markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing. *(folium.map.Marker, folium.Icon).*
  - Markers to show distance between launch site to key locations (railway, highway, coastway, city) and plot a line between them. *(folium.map.Marker, folium.PolyLine, folium.features.DivIcon)*
- These objects are created in order to understand better the problem and the data. We can show easily all launch sites, their surroundings and the number of successful and unsuccessful landings.

Build a Dashboard with Plotly Dash

- Dashboard has dropdown, pie chart, rangeslider and scatter plot components
  - Dropdown allows a user to choose the launch site or all launch sites *(dash_core_components.Dropdown)*.
  - Pie chart shows the total success and the total failure for the launch site chosen with the dropdown component *(plotly.express.pie)*.
  - Rangeslider allows a user to select a payload mass in a fixed range *(dash_core_components.RangeSlider)*.
  - Scatter chart shows the relationship between two variables, in particular Success vs Payload Mass *(plotly.express.scatter)*.

Predictive Analysis (Classification)

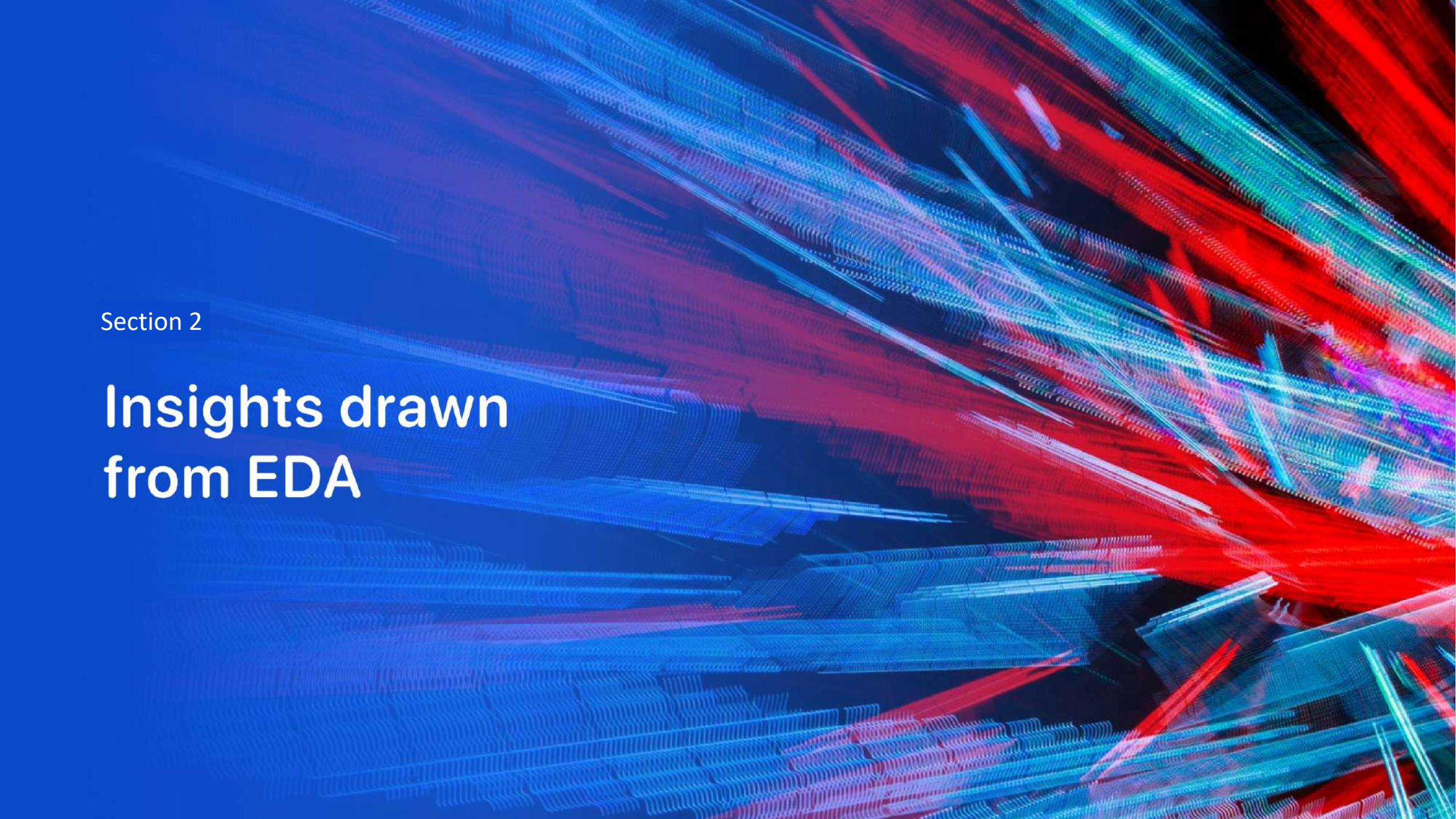
- Data preparation
  - Load dataset
  - Normalize data
  - Split data into training and test sets.
- Model preparation
  - Selection of machine learning algorithms
  - Set parameters for each algorithm to GridSearchCV
  - Training GridSearchModel models with training dataset
- Model evaluation
  - Get best hyperparameters for each type of model
  - Compute accuracy for each model with test dataset
  - Plot Confusion Matrix
- Model comparison
  - Comparison of models according to their accuracy
  - The model with the best accuracy will be chosen (see Notebook for result)

Results

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

Section 2

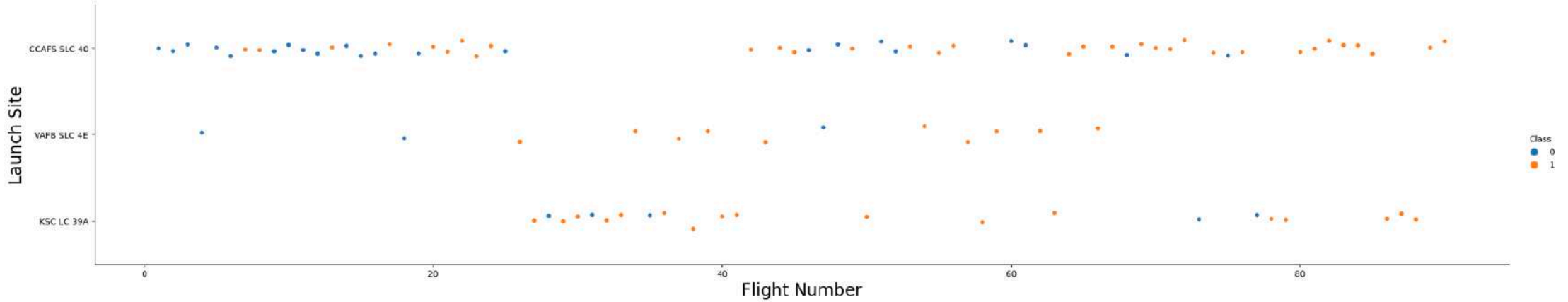
Insights drawn from EDA



Flight Number vs. Launch Site

TASK 1: Visualize the relationship between Flight Number and Launch Site

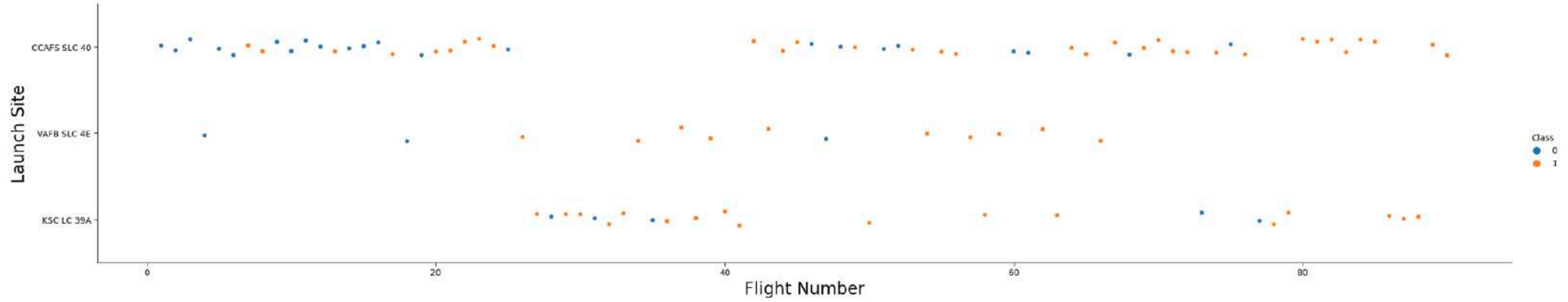
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the cla
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```



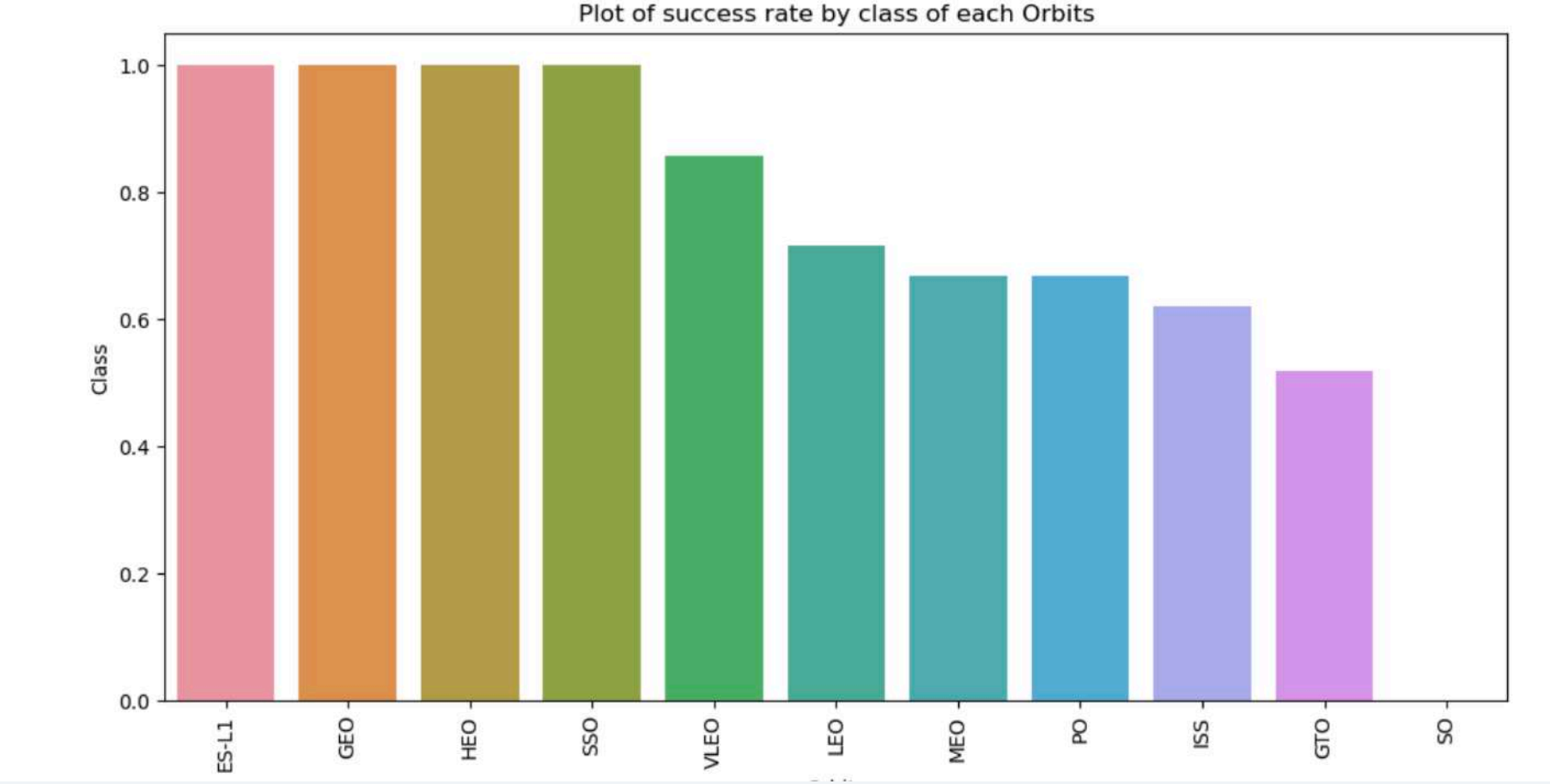
Payload vs. Launch Site

TASK 2: Visualize the relationship between Payload and Launch Site

```
[5]: # Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be th
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



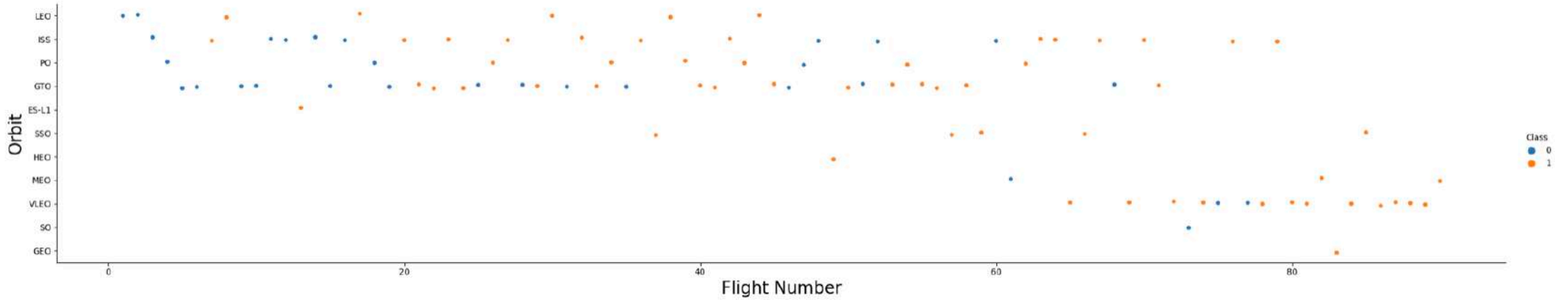
Success Rate vs. Orbit Type



Flight Number vs. Orbit Type

4: Visualize the relationship between FlightNumber and Orbit type

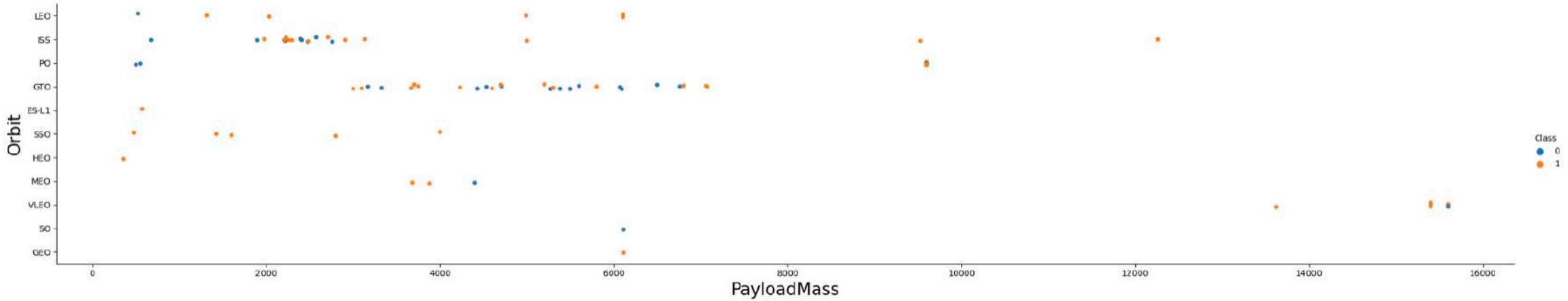
```
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Payload vs. Orbit Type

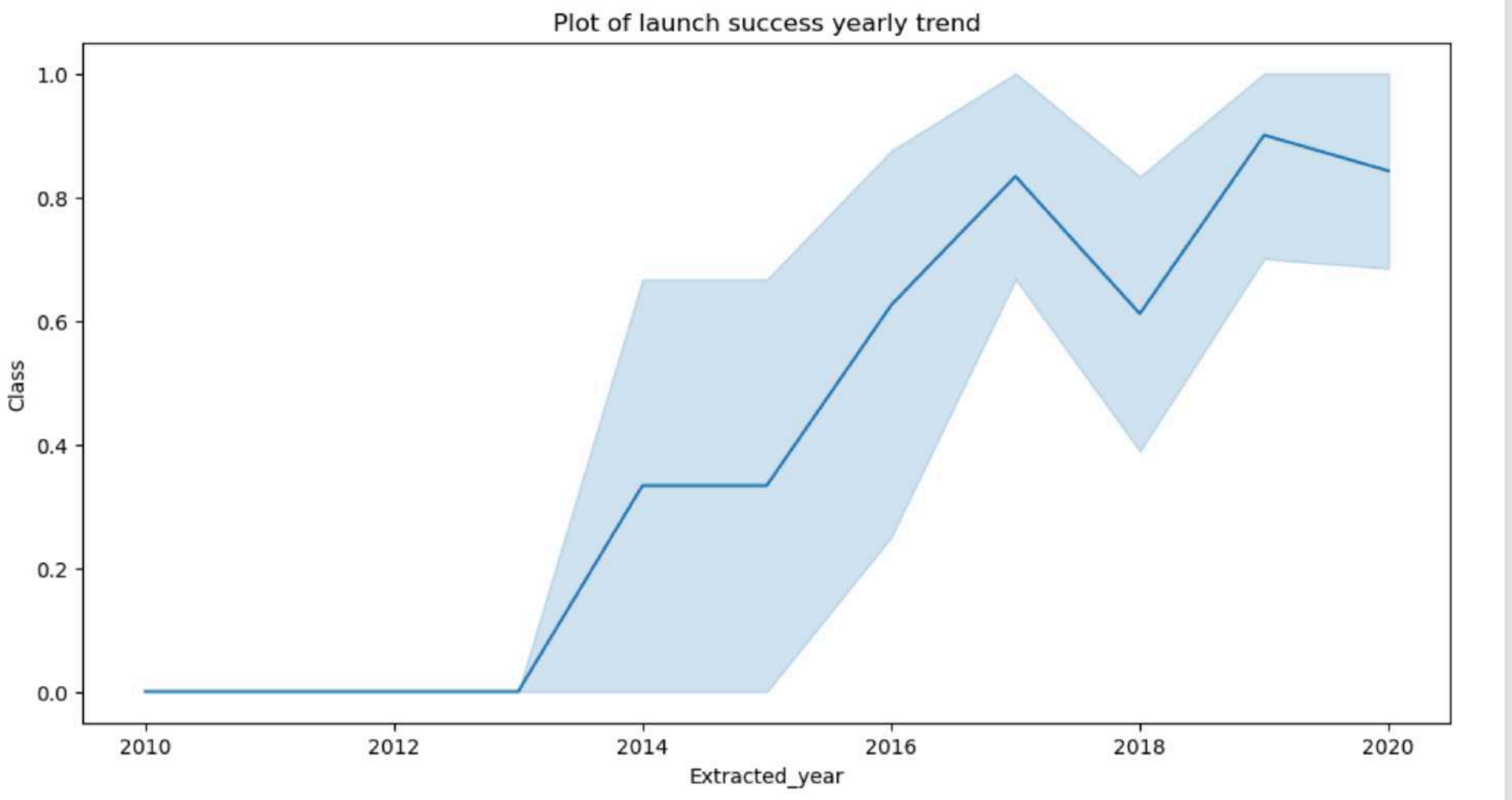
5: Visualize the relationship between Payload and Orbit type

```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

6: Visualize the launch success yearly trend



All Launch Site Names

SQL Query

```
SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

Results

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

Explanation

The use of DISTINCT in the query allows to remove duplicate LAUNCH_SITE.

Launch Site Names Begin with 'CCA'

SQL Query

```
SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE '%CCA%' LIMIT 5
```

Explanation

The WHERE clause followed by LIKE clause filters launch sites that contain the substring CCA.
LIMIT 5 shows 5 records from filtering.

Results

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
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
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)

Total Payload Mass

SQL Query

```
SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

Results

SUM("PAYLOAD_MASS__KG_")
45596

Explanation

This query returns the sum of all payload masses where the customer is NASA (CRS).

Average Payload Mass by F9 v1.1

SQL Query

```
SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "BOOSTER_VERSION" LIKE '%F9 v1.1%'
```

Results

AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

Explanation

This query returns the average of all payload masses where the booster version contains the substring F9 v1.1.

First Successful Ground Landing Date

SQL Query

```
SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing _Outcome" LIKE '%Success%'
```

Results

MIN("DATE")
01-05-2017

Explanation

With this query, we select the oldest successful landing.
The WHERE clause filters dataset in order to keep only records where landing was successful. With the MIN function, we select the record with the oldest date.

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL Query

```
%sql SELECT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "LANDING _OUTCOME" = 'Success (drone ship)' \
AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000;
```

Results

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

Explanation

This query returns the booster version where landing was successful and payload mass is between 4000 and 6000 kg. The WHERE and AND clauses filter the dataset.

Total Number of Successful and Failure Mission Outcomes

SQL Query

```
%sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Success%') AS SUCCESS, \
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Failure%') AS FAILURE
```

Results

SUCCESS	FAILURE
100	1

Explanation

With the first SELECT, we show the subqueries that return results. The first subquery counts the successful mission. The second subquery counts the unsuccessful mission. The WHERE clause followed by LIKE clause filters mission outcome. The COUNT function counts records filtered.

Boosters Carried Maximum Payload

SQL Query

```
%sql SELECT DISTINCT "BOOSTER_VERSION" FROM SPACEXTBL \
WHERE "PAYLOAD_MASS__KG_" = (SELECT max("PAYLOAD_MASS__KG_") FROM SPACEXTBL)
```

Results

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

Explanation

We used a subquery to filter data by returning only the heaviest payload mass with MAX function. The main query uses subquery results and returns unique booster version (SELECT DISTINCT) with the heaviest payload mass.

2015 Launch Records

SQL Query

```
%sql SELECT substr("DATE", 4, 2) AS MONTH, "BOOSTER_VERSION", "LAUNCH_SITE" FROM SPACEXTBL\
WHERE "LANDING _OUTCOME" = 'Failure (drone ship)' and substr("DATE",7,4) = '2015'
```

Results

MONTH	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

Explanation

This query returns month, booster version, launch site where landing was unsuccessful and landing date took place in 2015. Substr function process date in order to take month or year. Substr(DATE, 4, 2) shows month. Substr(DATE,7, 4) shows year.

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

SQL Query

```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "DATE" >= '04-06-2010' and "DATE" <= '20-03-2017' and "LANDING _OUTCOME" LIKE '%Success%'\
GROUP BY "LANDING _OUTCOME" \
ORDER BY COUNT("LANDING _OUTCOME") DESC ;
```

Results

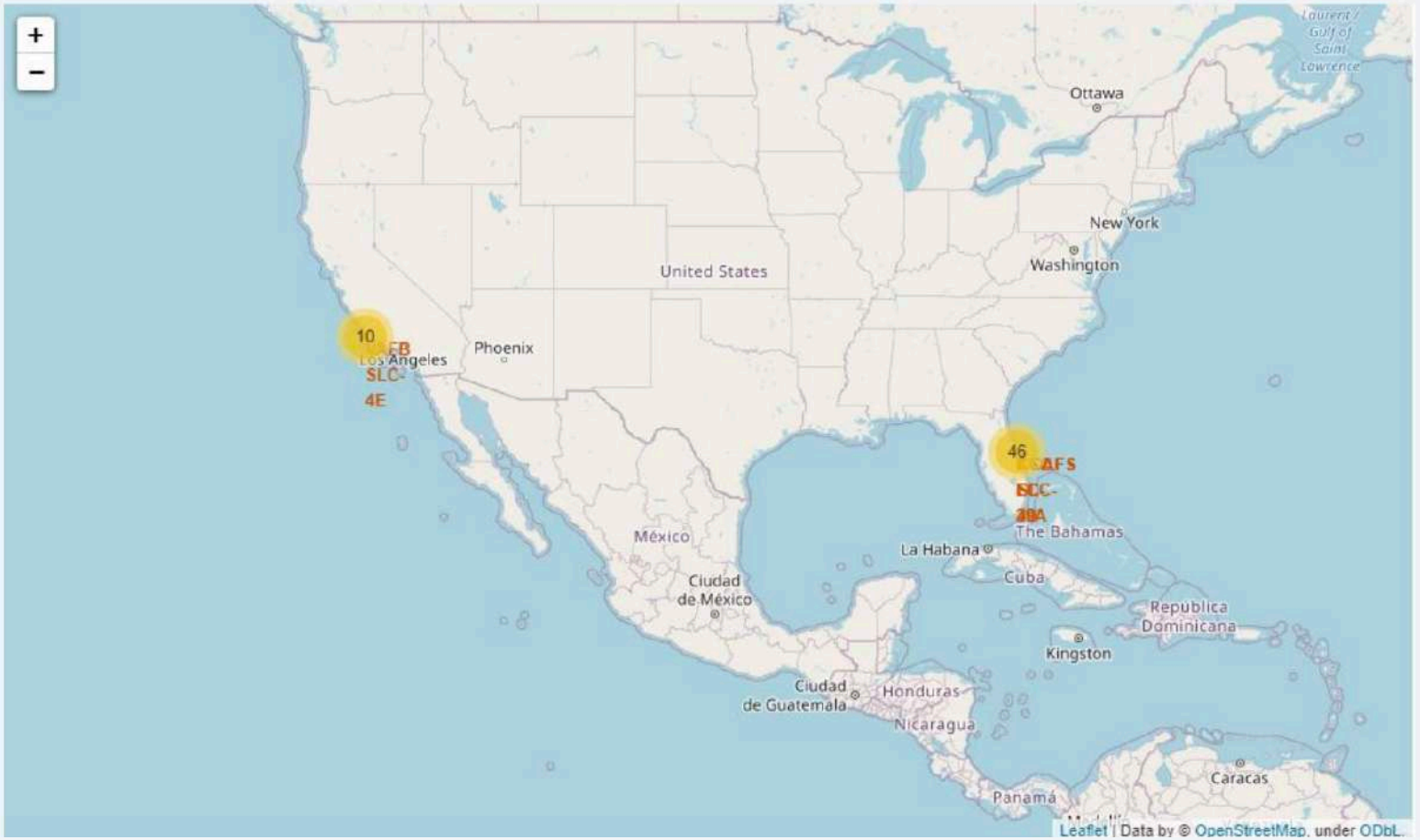
Landing _Outcome	COUNT("LANDING _OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	6

Explanation

This query returns landing outcomes and their count where mission was successful and date is between 04/06/2010 and 20/03/2017. The GROUP BY clause groups results by landing outcome and ORDER BY COUNT DESC shows results in decreasing order.


Section 3
Launch Sites
Proximities Analysis

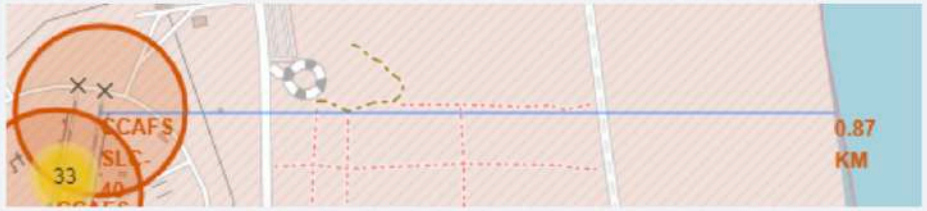
Folium Map Screenshot 1

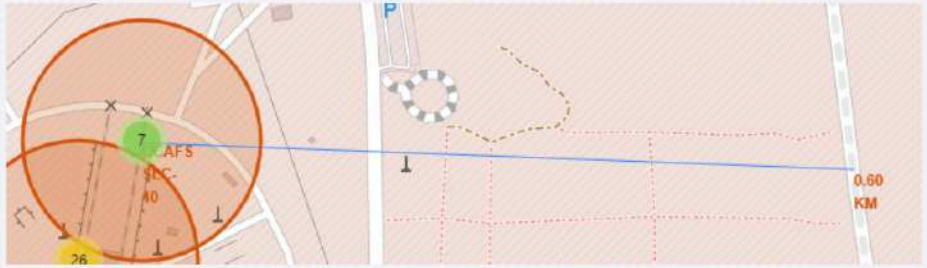


We see that Space X launch sites are located on the coast of the United States

Folium Map color labeled markers

Folium Map – Distances between CCAFS SLC-40 and its proximities

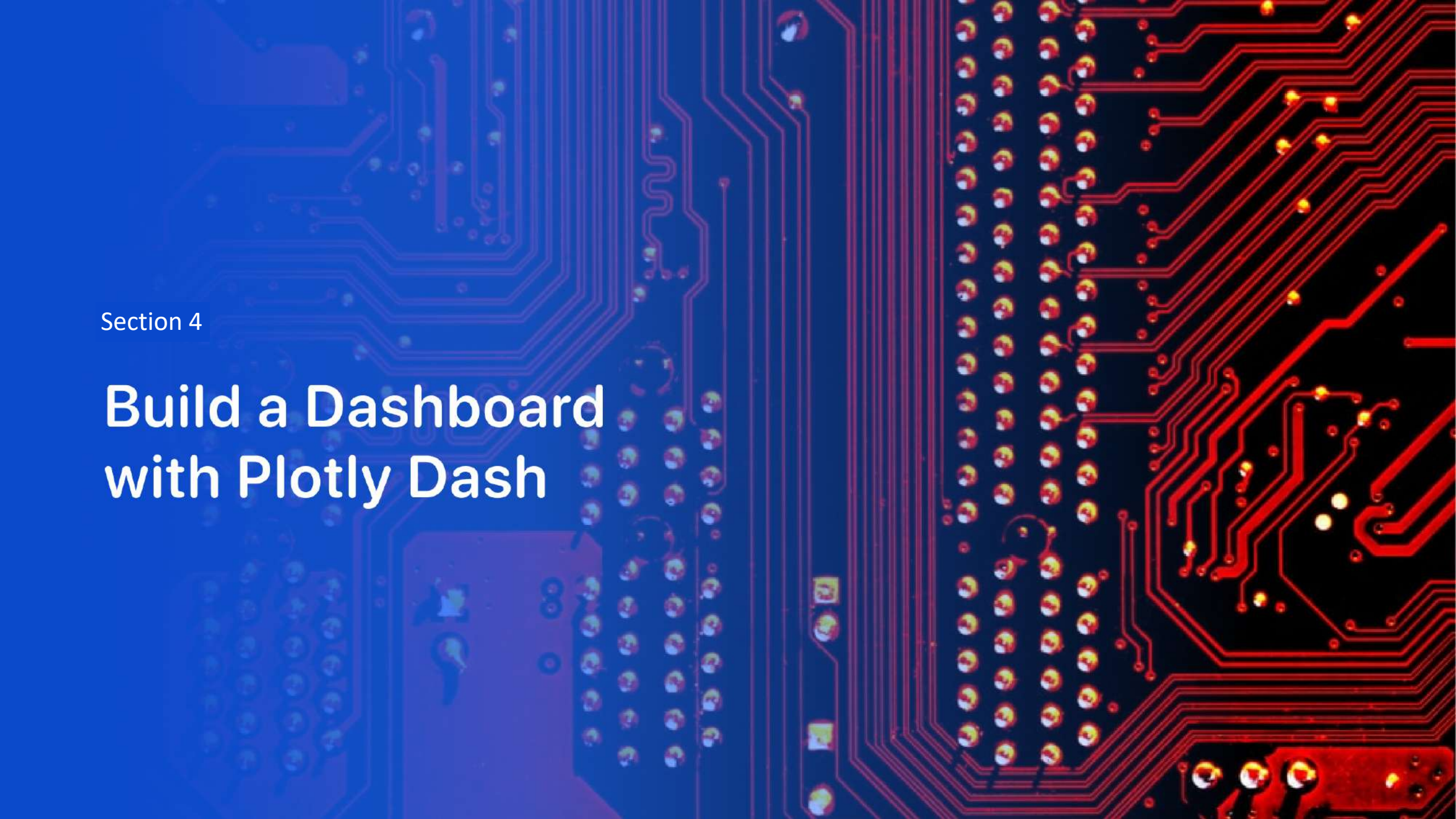




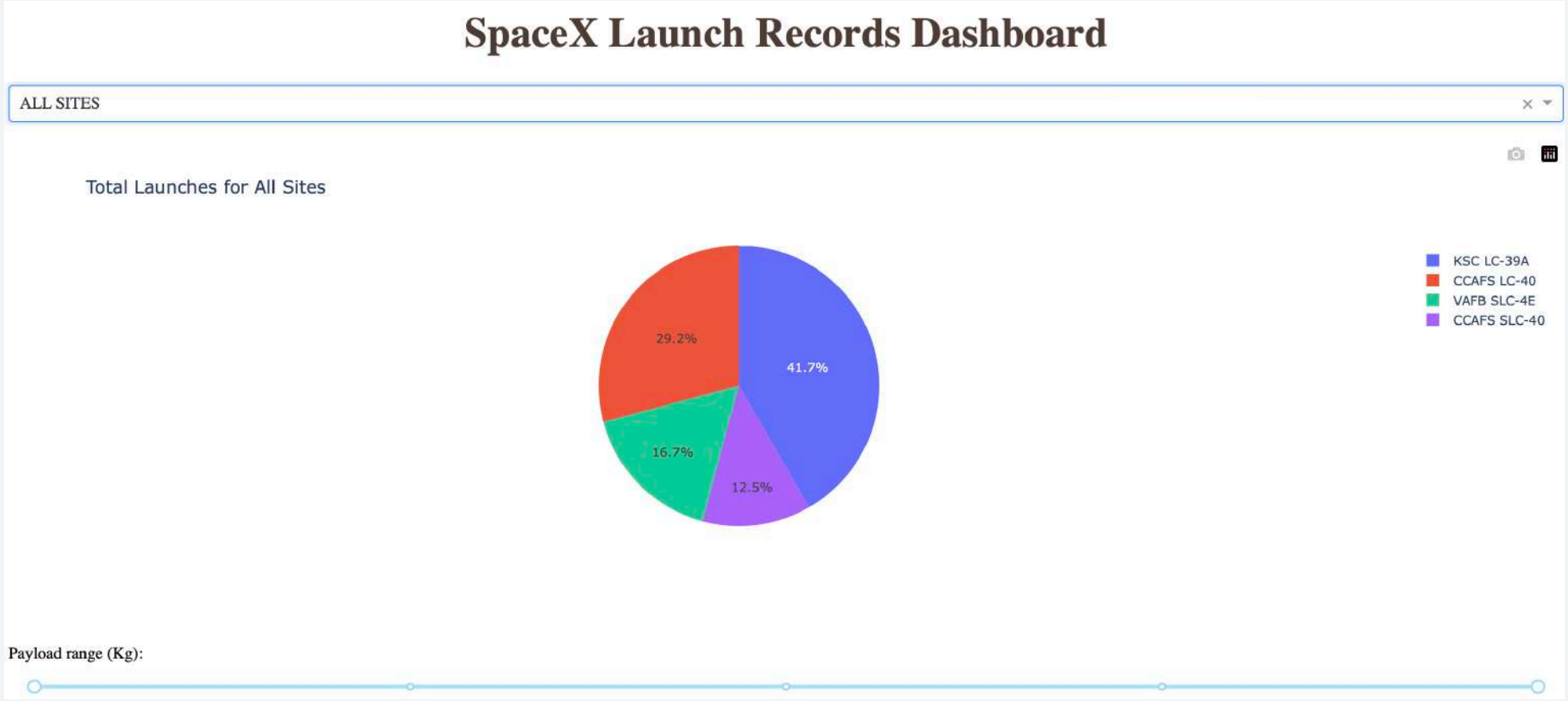
Is CCAFS SLC-40 in close proximity to railways ? Yes
Is CCAFS SLC-40 in close proximity to highways ? Yes
Is CCAFS SLC-40 in close proximity to coastline ? Yes
Do CCAFS SLC-40 keeps certain distance away from cities ? No

Section 4

Build a Dashboard with Plotly Dash

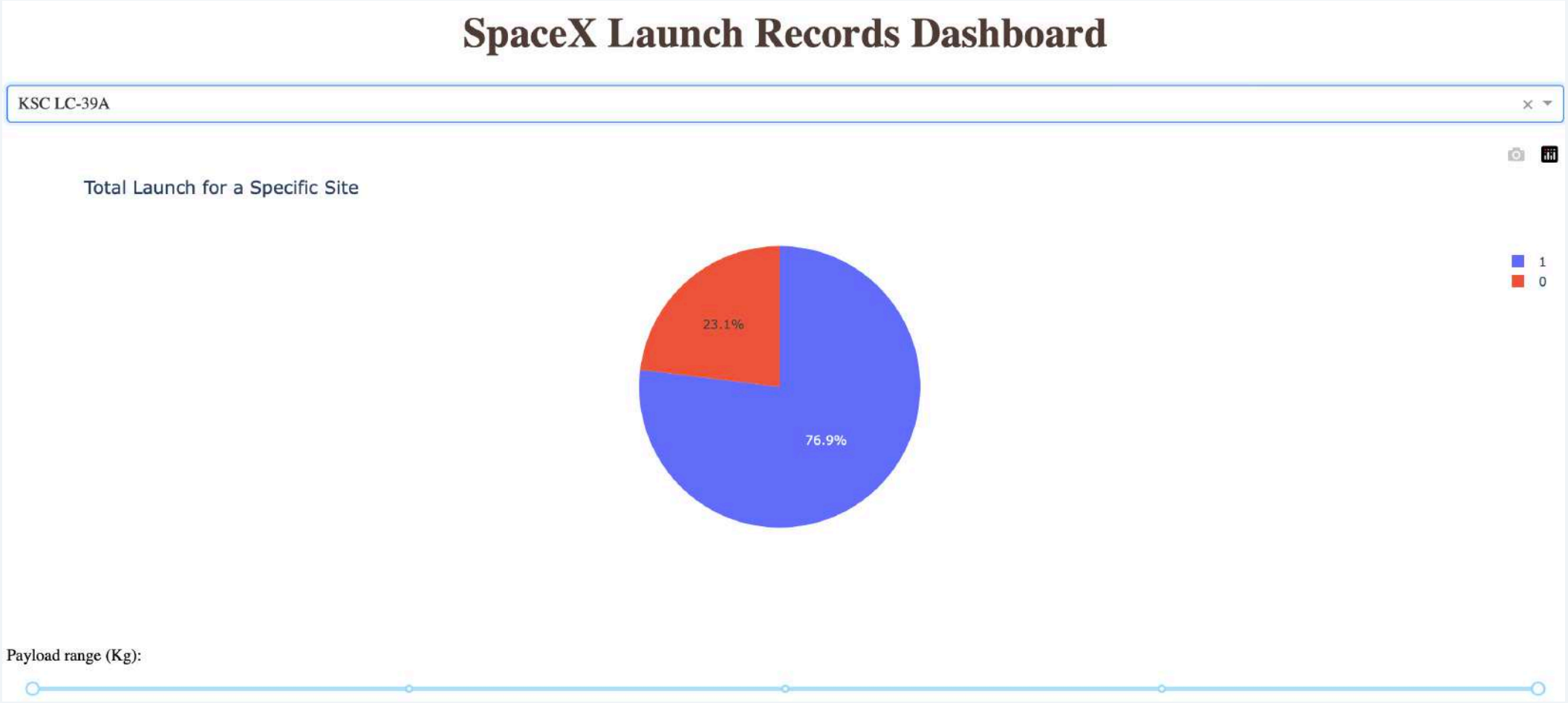


Dashboard – Total success by Site

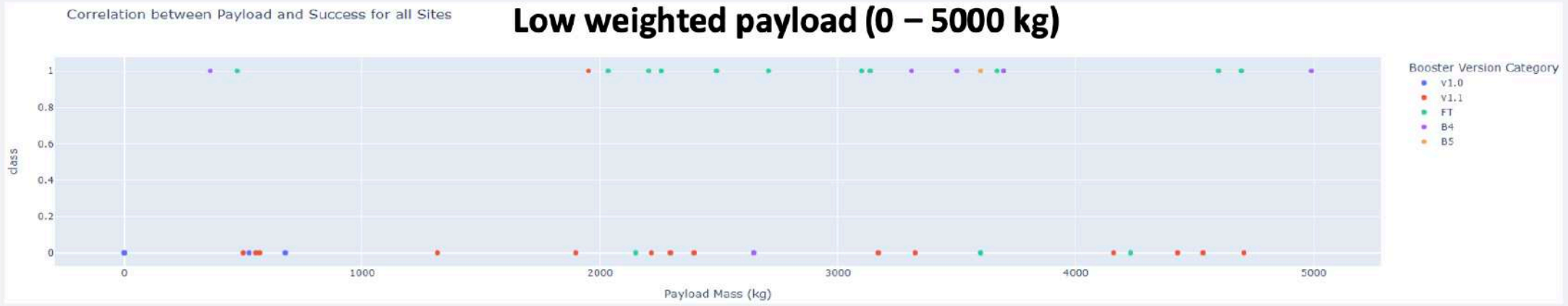


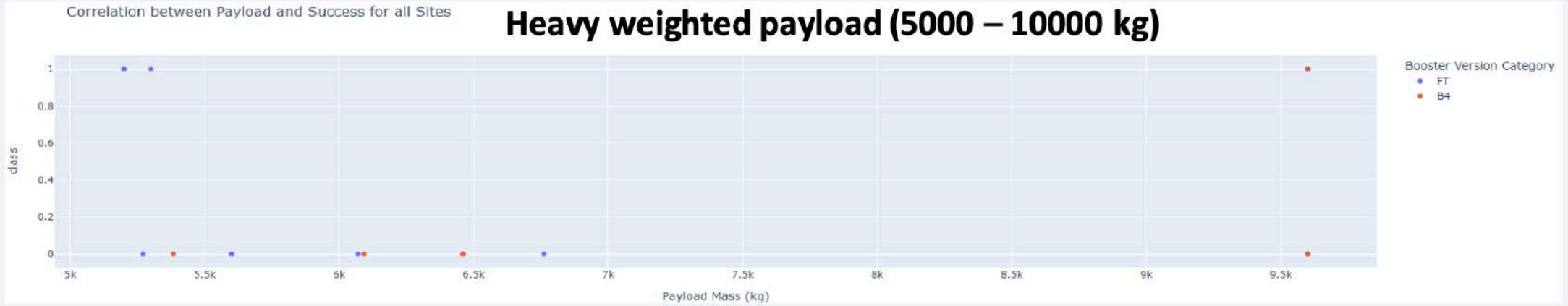
We see that KSC LC-39A has the best success rate of launches

Dashboard – Total success launches for Site KSC LC-39A



Dashboard – Payload mass vs Outcome for all sites with different payload mass selected





Section 5

Predictive Analysis (Classification)

Classification Accuracy

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

```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

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

```
tuned hpyerparameters :(best parameters)  {'algorithm': 'ball_tree', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```

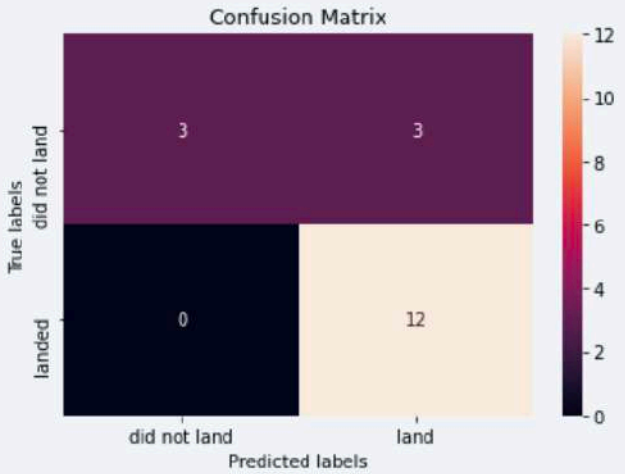
In [25]:
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_sample
s_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
accuracy : 0.8732142857142856
```

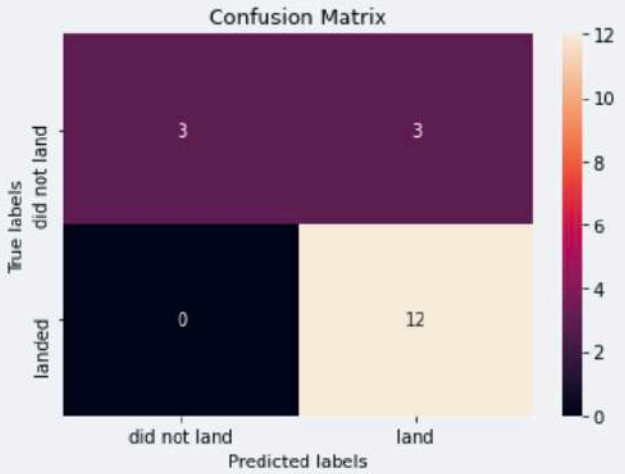
Decision tree best parameters

Confusion Matrix

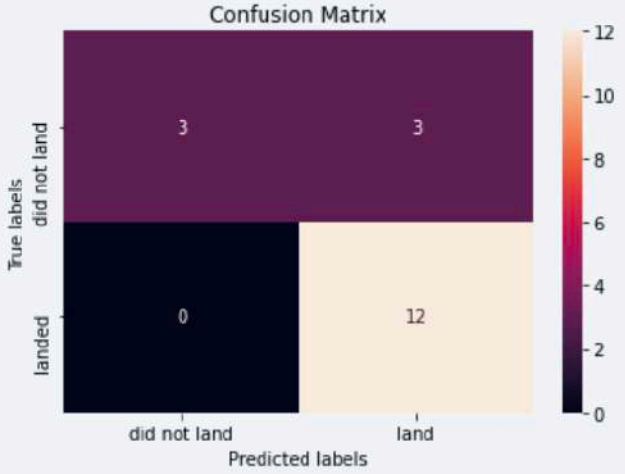
Logistic regression



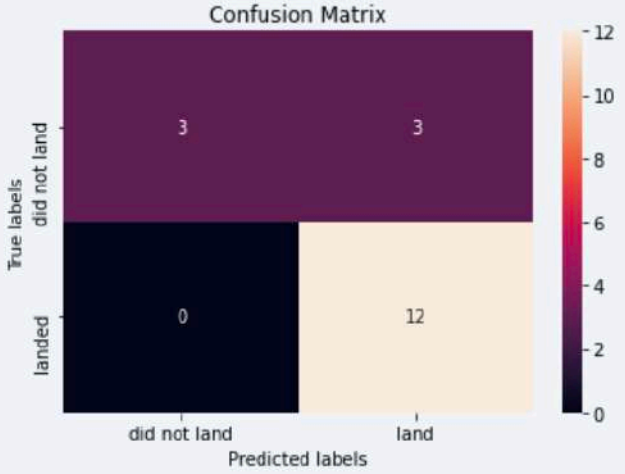
Decision Tree



kNN



SVM



Conclusion

• The success of a mission can be explained by several factors such as the launch site, the orbit and especially the number of previous launches. Indeed, we can assume that there has been a gain in knowledge between launches that allowed to go from a launch failure to a success.
• The orbits with the best success rates are GEO, HEO, SSO, ES-L1.
• Depending on the orbits, the payload mass can be a criterion to take into account for the success of a mission. Some orbits require a light or heavy payload mass. But generally low weighted payloads perform better than the heavy weighted payloads.
• With the current data, we cannot explain why some launch sites are better than others (KSC LC-39A is the best launch site). To get an answer to this problem, we could obtain atmospheric or other relevant data. • For this dataset, we choose the Decision Tree Algorithm as the best model even if the test accuracy between all the models used is identical. We choose Decision Tree Algorithm because it has a better train accuracy.

Thank you!