IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Yenchi Wang
October 13th 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection with web scraping
  - Data wrangling
  - Exploratory data analysis with SQL, Pandas and Matplotlib
  - SpaceX launch sites analysis with Folium visual analysis and Ploty dashboard
  - Machine Learning with Scikit-learn

- Summary of all results
  - Exploratory data analysis report
  - Interactive visual analytics and dashboards
  - Predictive analysis (Classification)

Introduction

- Project background and context

SpaceX lists the price for Falcon 9 rocket launches on its website as 62 million dollars, while other providers charge over 165 million dollars for similar services. A significant portion of SpaceX's cost savings comes from its ability to reuse the first stage of the rocket. Thus, by predicting whether the first stage will land successfully, we can estimate the launch cost. This data would be valuable for any competitor aiming to offer a competitive bid against SpaceX for a rocket launch.

- Problems you want to find answers

In this capstone project, we'll forecast the success of the Falcon 9's first stage landing using data sourced from Falcon 9 rocket launch details available on its website.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- 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

- Initially, data was gathered from the SpaceX API (a RESTful service) using a 'get' request. To facilitate this, a set of helper functions were created to assist in extracting information based on identification numbers within the launch data. This data was then pulled from the SpaceX API endpoint.
- To ensure consistency in the JSON results, we fetched the SpaceX launch data using a GET request. After receiving the response, we decoded the content into a JsonResult and subsequently transformed it into a Pandas dataframe.
- Additionally, we scraped Falcon 9 historical launch records from a Wikipedia page titled "List of Falcon 9 and Falcon Heavy launches". These records, present in an HTML format, were extracted using the BeautifulSoup and requests libraries. The HTML table details were then parsed and converted into another Pandas dataframe.

Data Collection – SpaceX API

- Data was gathered from the SpaceX API (a RESTful service) using a 'get' request. After fetching the SpaceX launch information through this request, the response content was decoded from JSON and subsequently transformed into a Pandas dataframe.

- The Github file URL (https://github.com/YenchiSomnambule/IBM_SpaceX_Prediction/blob/main/1.%20SpaceX%20%20Falcon%209%20first%20stage%20Landing%20Prediction.ipynb)

-

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_
```

We should see that the request was successfull with the 200 status response code

In [10]:
```
response.status_code
```

Out[10]: 200

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

In [16]:
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

Data Collection - Scraping

- I used web scraping techniques to retrieve Falcon 9 historical launch data from a Wikipedia page. Utilizing BeautifulSoup and the requests library, I extracted the Falcon 9 launch details from the Wikipedia page's HTML table. This extracted data was then parsed and converted into a dataframe.

- Github URL: https://github.com/YenchiSomnambule/IBM_SpaceX_Prediction/blob/main/2.%20Web%20scraping%20Falcon%209%20and%20Falcon%20Heavy%20Launches%20Records%20from%20Wikipedia.ipynb

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

[5]:
```
# 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`

[6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

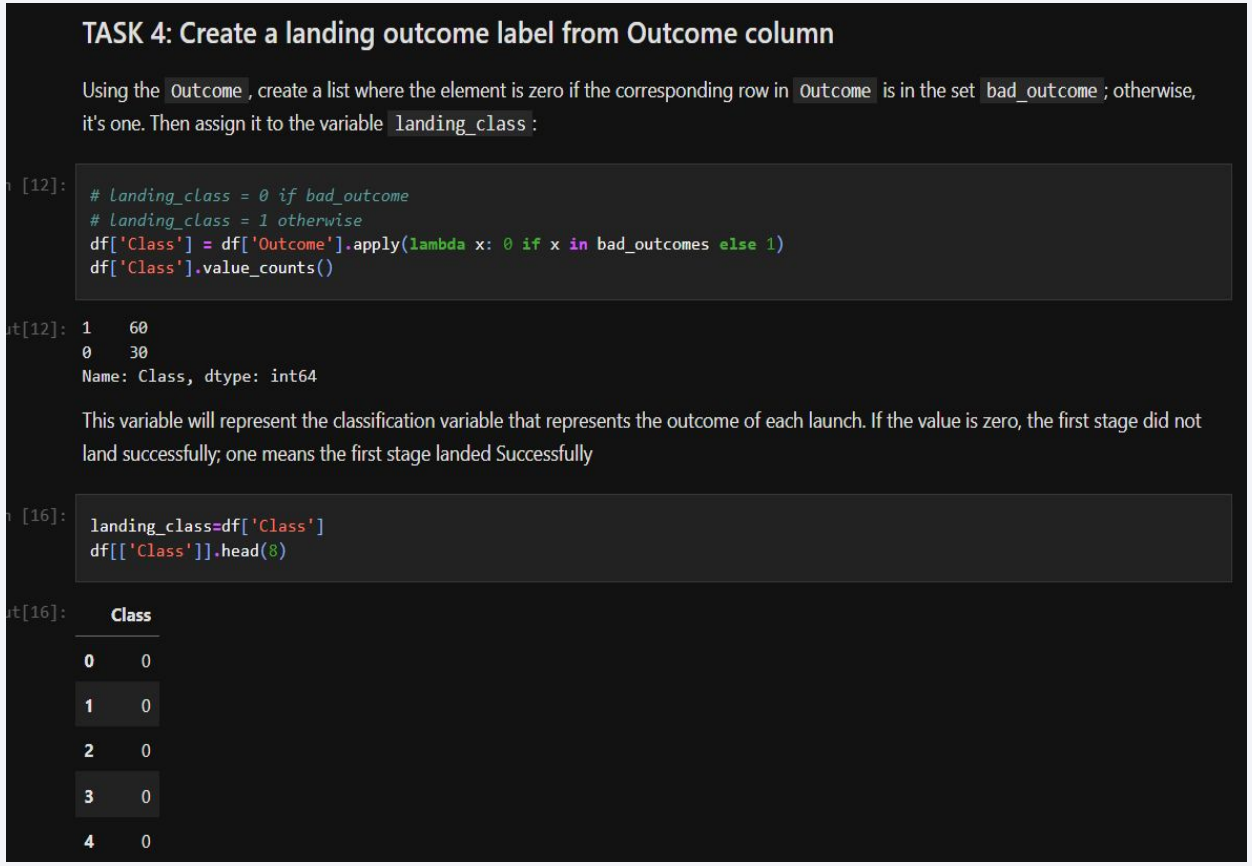
Print the page title to verify if the `BeautifulSoup` object was created properly

[7]:
```
# Use soup.title attribute
soup.title
```

[7]: `<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>`

Data Wrangling

- After transforming the gathered data into a Pandas dataframe, I filtered it using the 'BoosterVersion' column to retain only the Falcon 9 launches. I then addressed the missing values in the 'LandingPad' and 'PayloadMass' columns. For 'PayloadMass', I substituted missing entries with the column's average value.
- I also conducted exploratory data analysis (EDA) to identify patterns in the data and establish the label for training the supervised models.
- Github URL: https://github.com/YenchiSomnambule/IBM_SpaceX_Prediction/blob/main/3.%20Data%20wrangling.ipynb
-



TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

[12]:
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
df['Class'] = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
df['Class'].value_counts()
```

t[12]:
```
1    60
0    30
Name: Class, dtype: int64
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

[16]:
```
landing_class=df['Class']
df[['Class']].head(8)
```

t[16]:

	Class
0	0
1	0
2	0
3	0
4	0

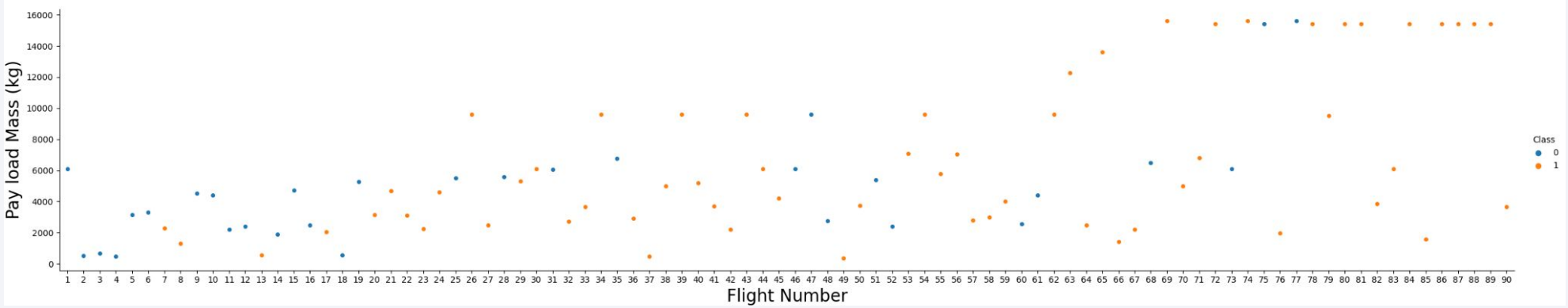
EDA with Data Visualization

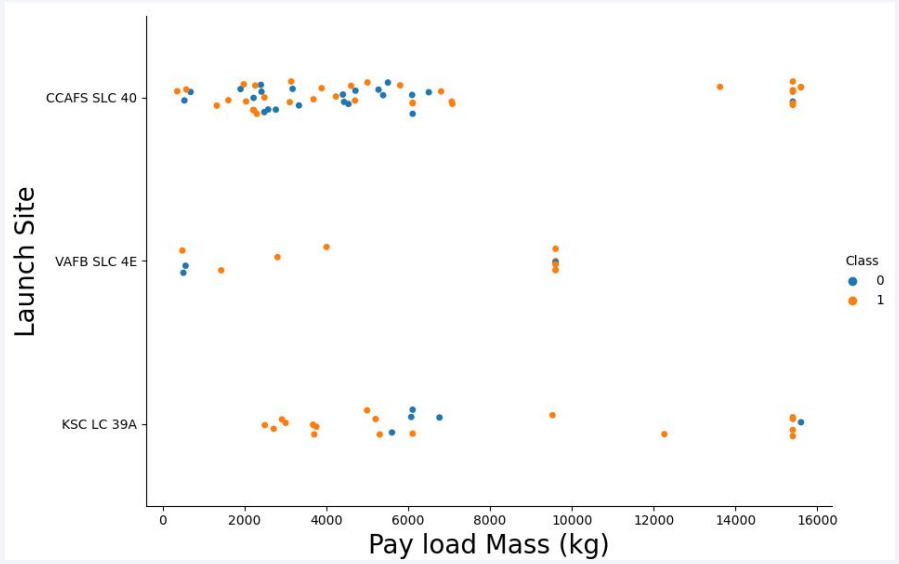
I carried out data analysis and feature engineering utilizing Pandas and Matplotlib, which included:

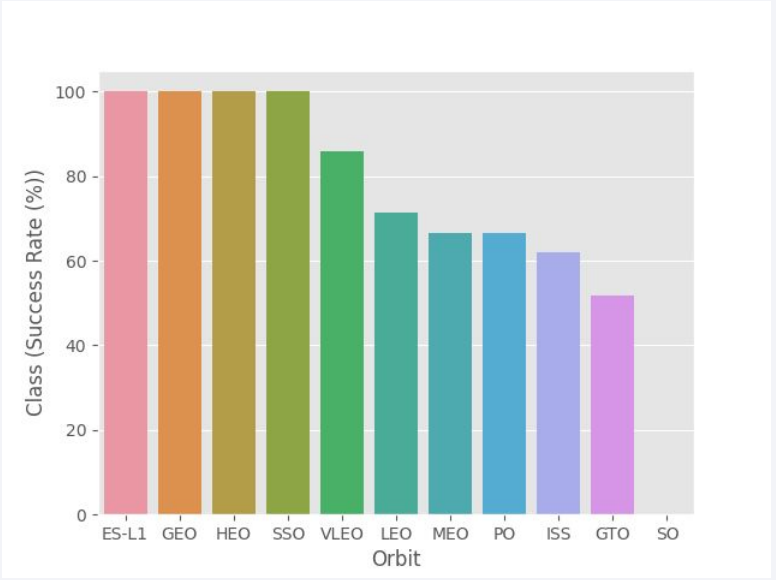
- Conducting exploratory data analysis (EDA)
- Engaging in data preparation and feature engineering
- Utilizing scatter plots to examine the relationships between Flight Number & Launch Site, Payload & Launch Site, Flight Number & Orbit Type, and Payload & Orbit Type.
- Employing bar charts to assess the success rate for each orbit type
- Using line plots to track the annual trend of launch successes

Github URL:
https://github.com/YenchiSomnambule/IBM_SpaceX_Prediction/blob/main/5.%20Space-X%20EDA%20DataViz%20Using%20Pandas%20and%20Matplotlib%20-%20SpaceX.ipynb

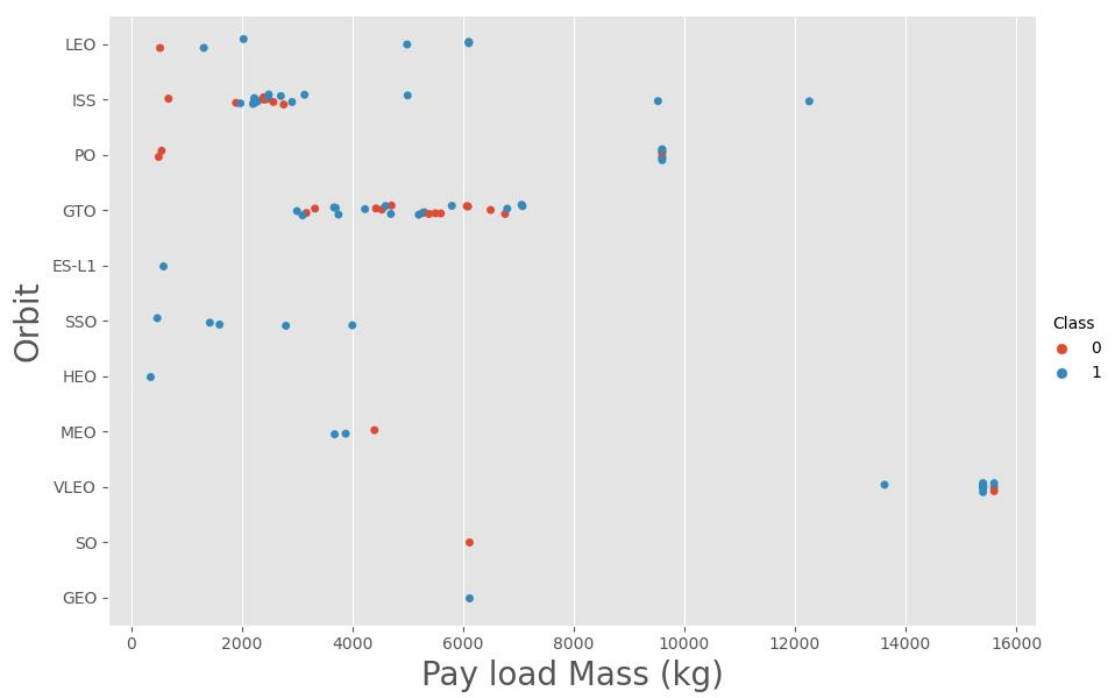
EDA with Data Visualization

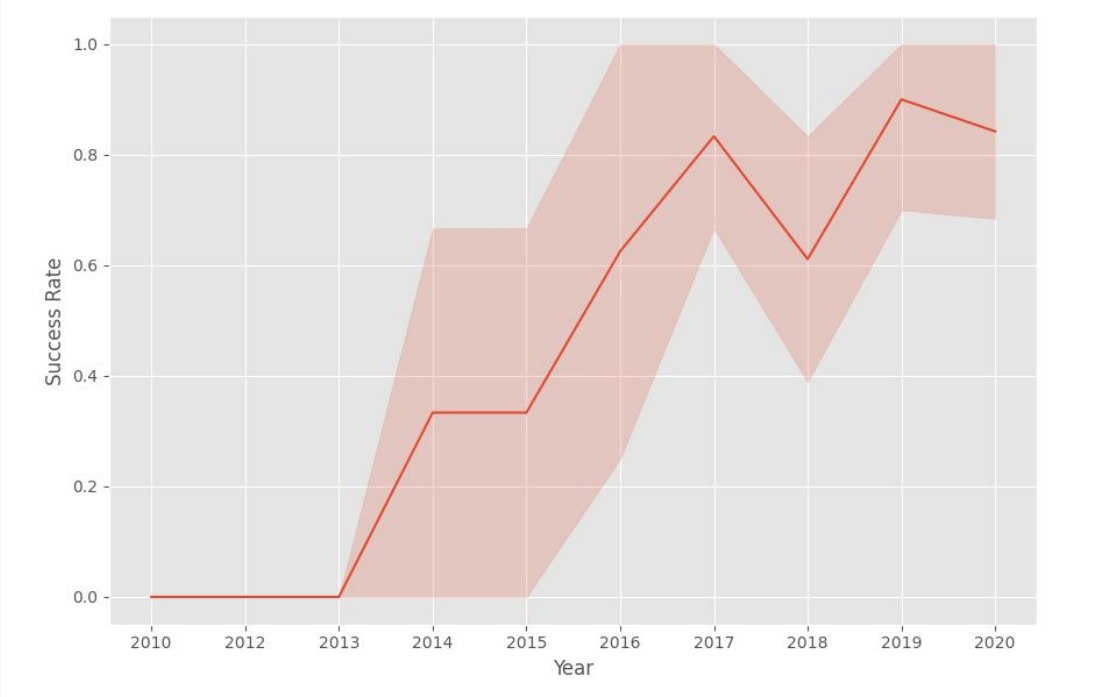






EDA with Data Visualization





EDA with SQL

The following SQL are used for EDA

- Display the names of the unique launch sites in the space mission
  %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
- Display 5 records where launch sites begin with the string 'CCA'
  %sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
- Display the total payload mass carried by boosters launched by NASA (CRS)
  %sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
- Display average payload mass carried by booster version F9 v1.1
  %sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
- List the date when the first succesful landing outcome in ground pad was acheived.
  %sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";

EDA with SQL

The following SQL are used for EDA

- 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 DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
- List the total number of successful and failure mission outcomes
  %sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
  %sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);

EDA with SQL

The following SQL are used for EDA

- 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.
  %sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome" FROM SPACEXTBL WHERE substr(Date,7,4)='2015' AND "Landing _Outcome" = 'Failure (drone ship)';
- Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.
  %sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;

Github URL:
https://github.com/YenchiSomnambule/IBM_SpaceX_Prediction/blob/main/4e.%20Space-X%20EDA%20Using%20SQL.ipynb

Build an Interactive Map with Folium

- I developed a folium map to pinpoint all the launch sites and incorporated map elements like markers, circles, and lines to denote the success or failure of launches at each site.
- I also established a set of launch outcomes, categorizing them as failure (0) or success (1).

Github URL:
https://github.com/YenchiSomnambule/IBM_SpaceX_Prediction/blob/main/6.Space-X%20Launch%20Sites%20Locations%20Analysis%20with%20Folium-Interactive%20Visual%20Analytics.ipynb

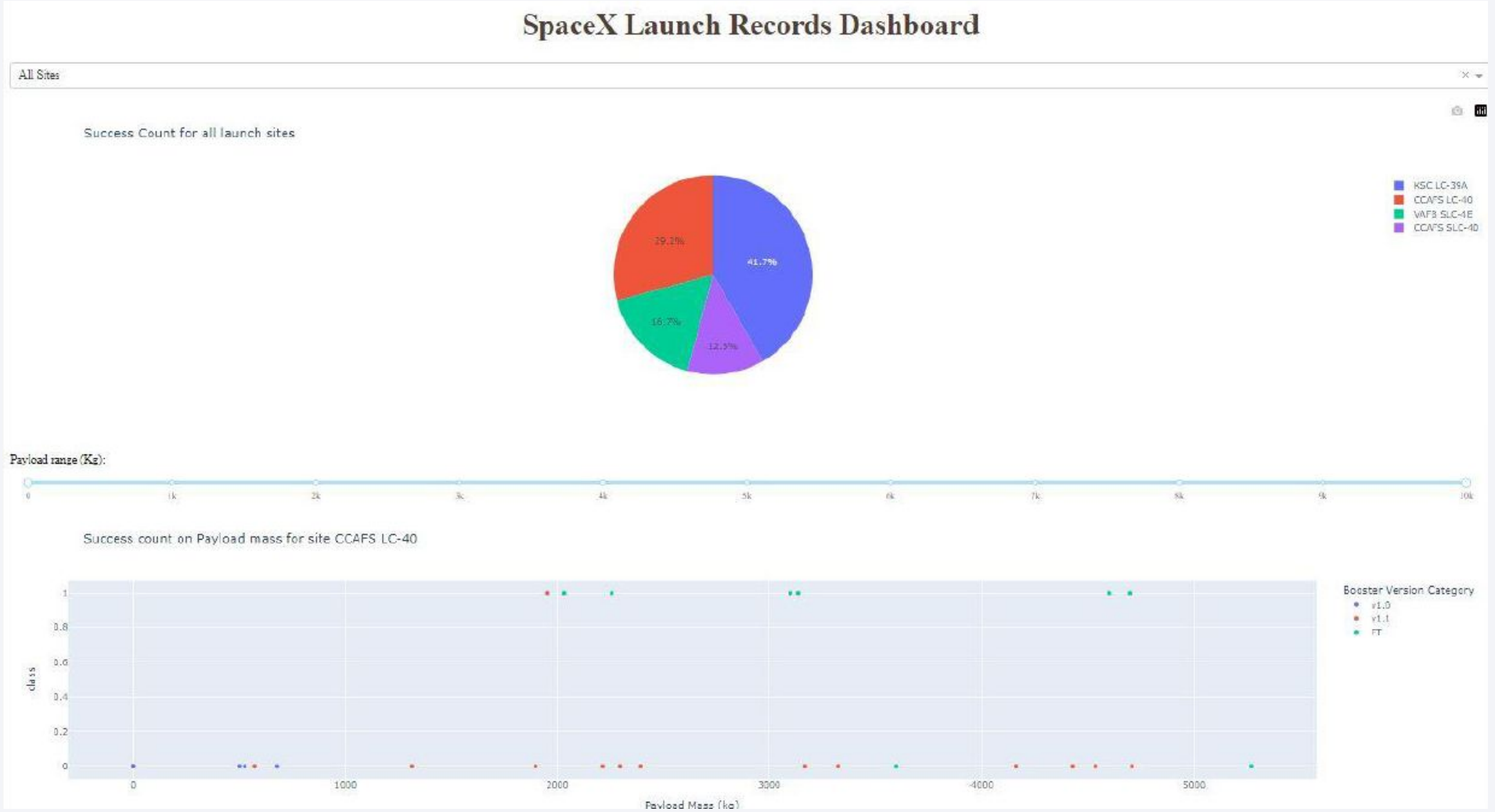
Build a Dashboard with Plotly Dash

- I constructed an interactive dashboard using Plotly Dash, which involved:
- Incorporating a dropdown input component for selecting a Launch Site
- Integrating a callback function to display a success pie chart corresponding to the chosen site from the dropdown
- Embedding a Range Slider for payload selection
- Instituting a callback function to showcase the success-payload-scatter-chart as a scatter plot.

Github URL:

https://github.com/YenchiSomnambule/IBM_SpaceX_Prediction/blob/main/7.%20Build%20an%20Interactive%20Dashboard%20with%20Ploty%20Dash%20-%20spacex_dash_app.py

Build a Dashboard with Plotly Dash



Predictive Analysis (Classification)

- Here's a brief overview of how I developed, assessed, refined, and identified the optimal classification model
- Upon loading the data into a Pandas Dataframe, I embarked on exploratory data analysis and defined the training labels. This involved:
    - Converting the 'Class' column from the data into a NumPy array using the to_numpy() method and designating it to the 'Y' variable as our target.
    - I then standardized the feature dataset (x) utilizing the preprocessing.StandardScaler() method from Sklearn.
    - Subsequently, I partitioned the data into training and testing subsets using train_test_split from sklearn.model_selection, setting test_size to 0.2 and random_state to 2.

Predictive Analysis (Classification)

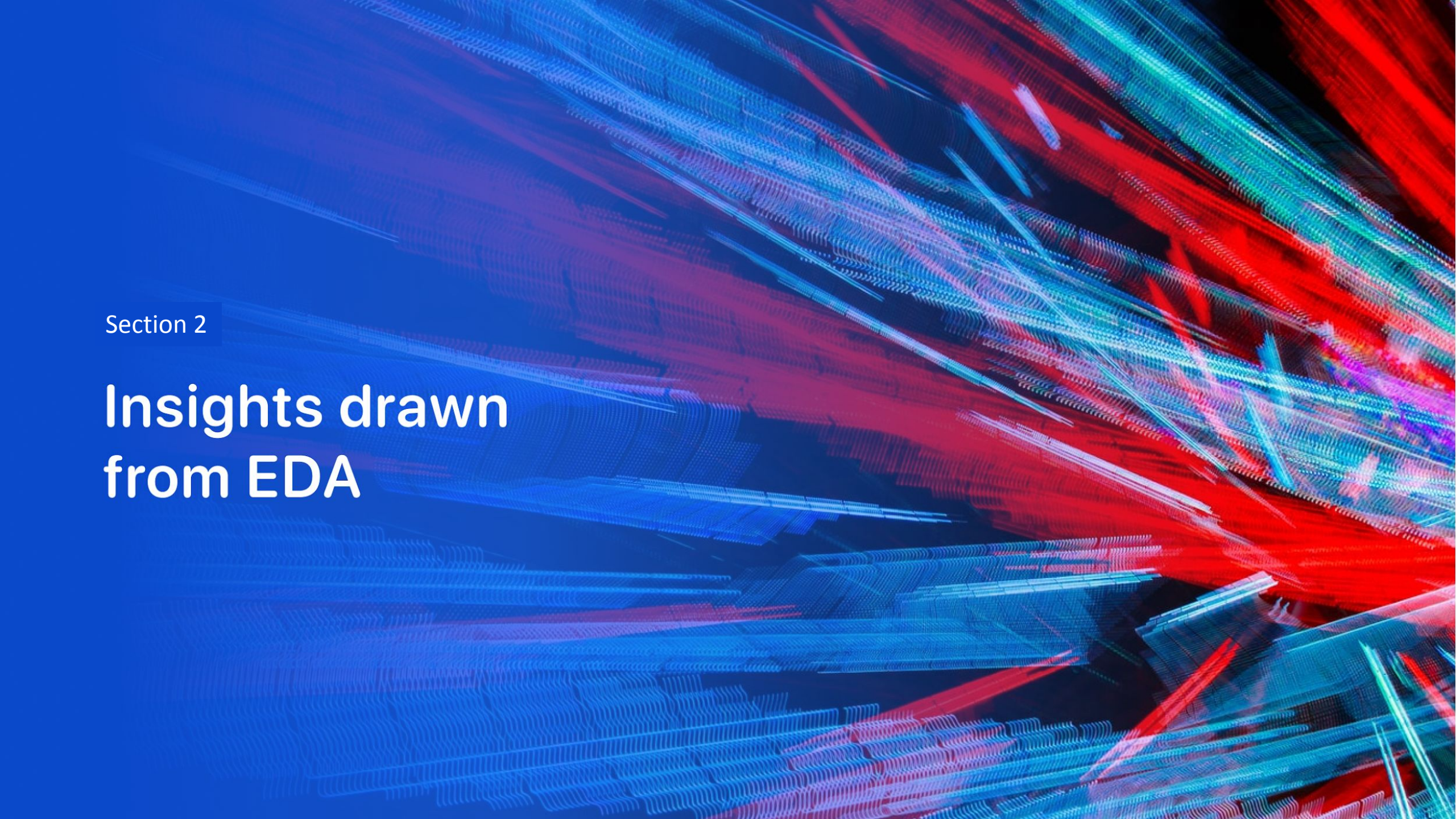
- To determine the most effective ML model/method for the test data among SVM, Classification Trees, k-nearest neighbors, and Logistic Regression:
    - I began by initializing an object for each algorithm and then set up a GridSearchCV object, providing a set of parameters for each model.
    - For every model being assessed, I instantiated the GridSearchCV object with cv=10, then integrated the training data into each GridSearch object to identify the optimal hyperparameters.
    - Post training, I retrieved the best parameters for each model using the best_params_ attribute and ascertained the validation data accuracy with the best_score_ attribute.
    - Concluding the process, I employed the score method to gauge the test data accuracy for each model. Additionally, I visualized the results by plotting a confusion matrix using the actual and predicted outcomes for each model.
- Eventually, the test accuracy of Logistic Regression, SVM, Decision tree and KNN are calculated with all of them are 0.833
- Github URL: https://github.com/YenchiSomnambule/IBM_SpaceX_Prediction/blob/main/8.%20SpaceX%20Machine%20Learning%20Prediction.ipynb
-

Results

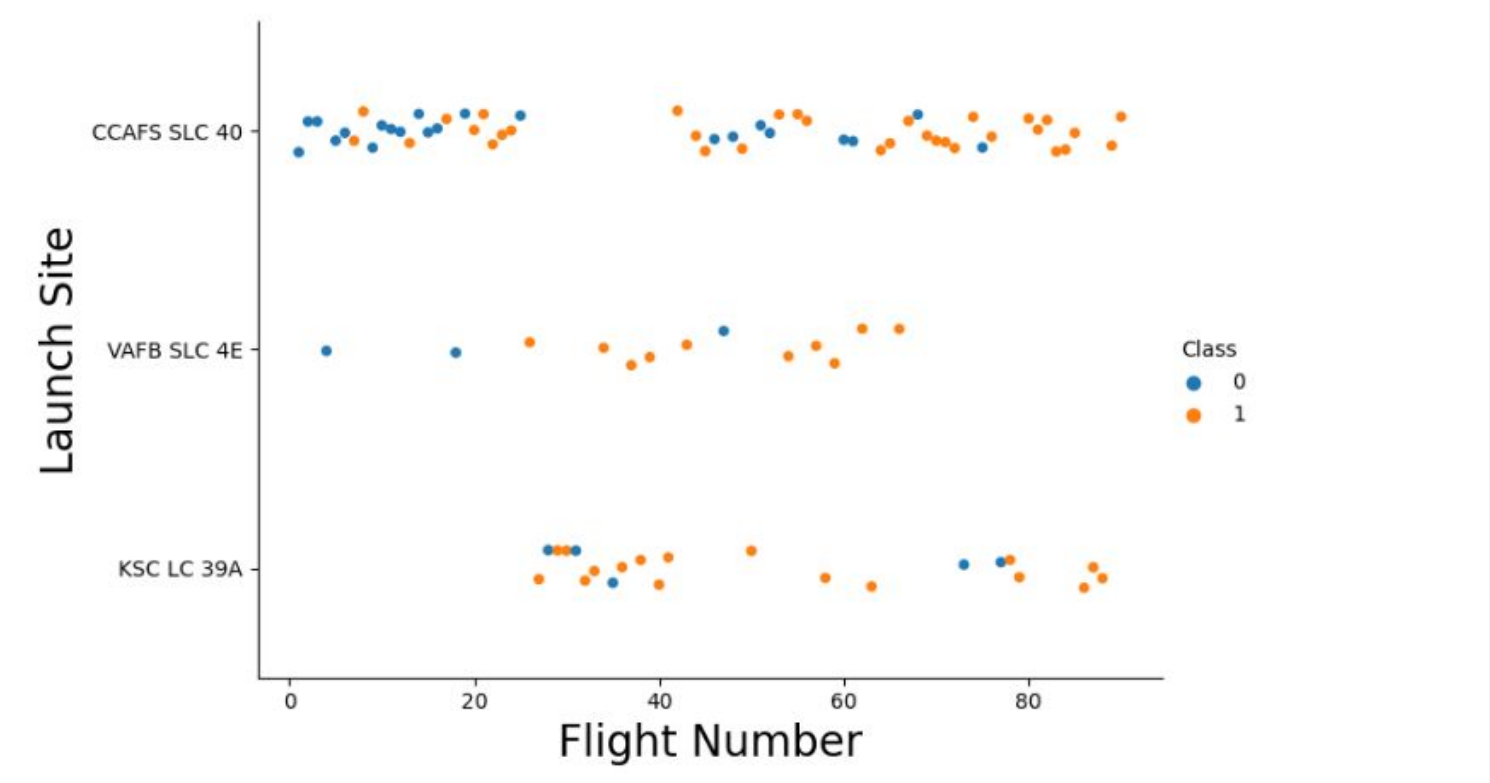
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

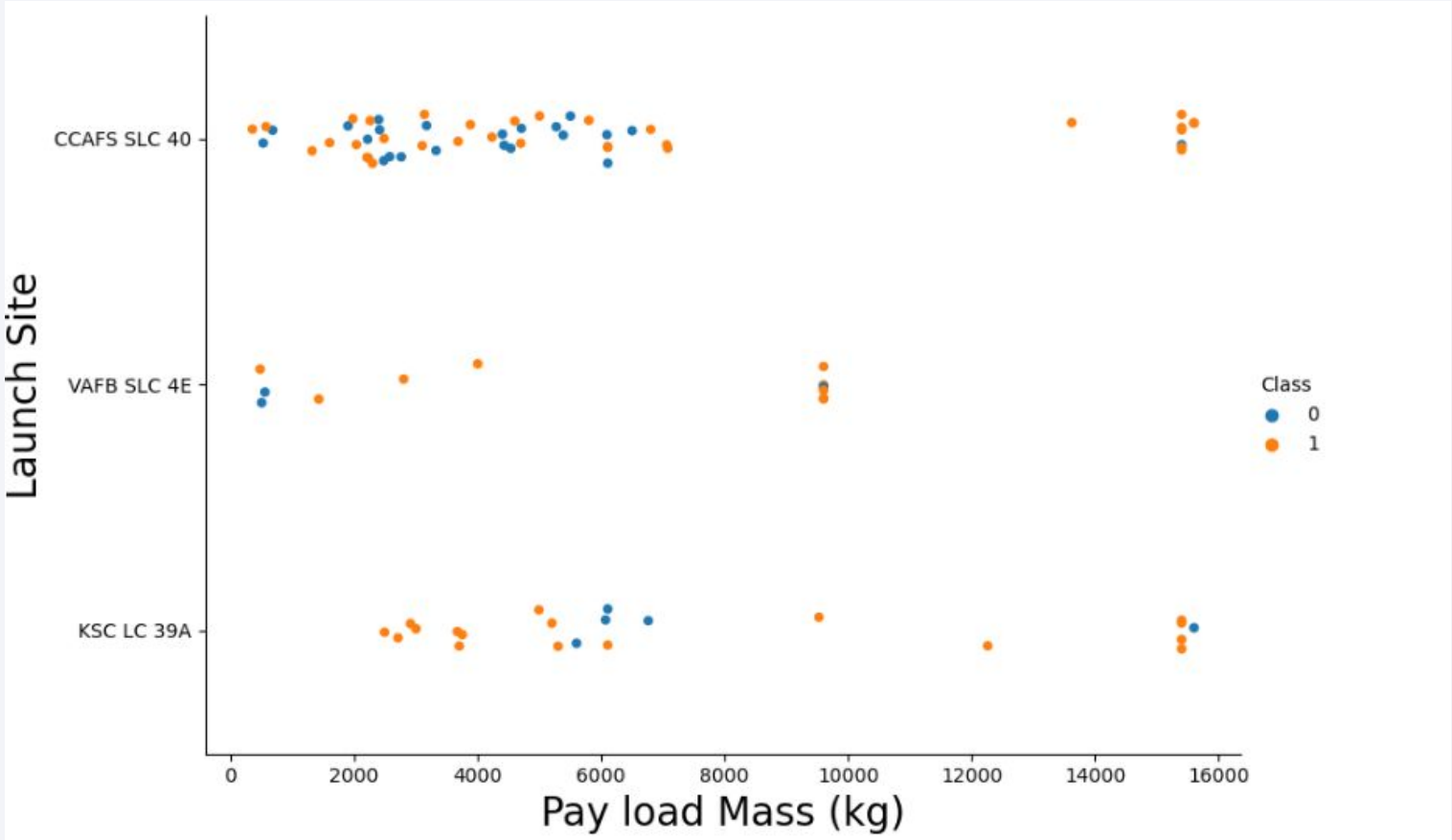
Insights drawn from EDA



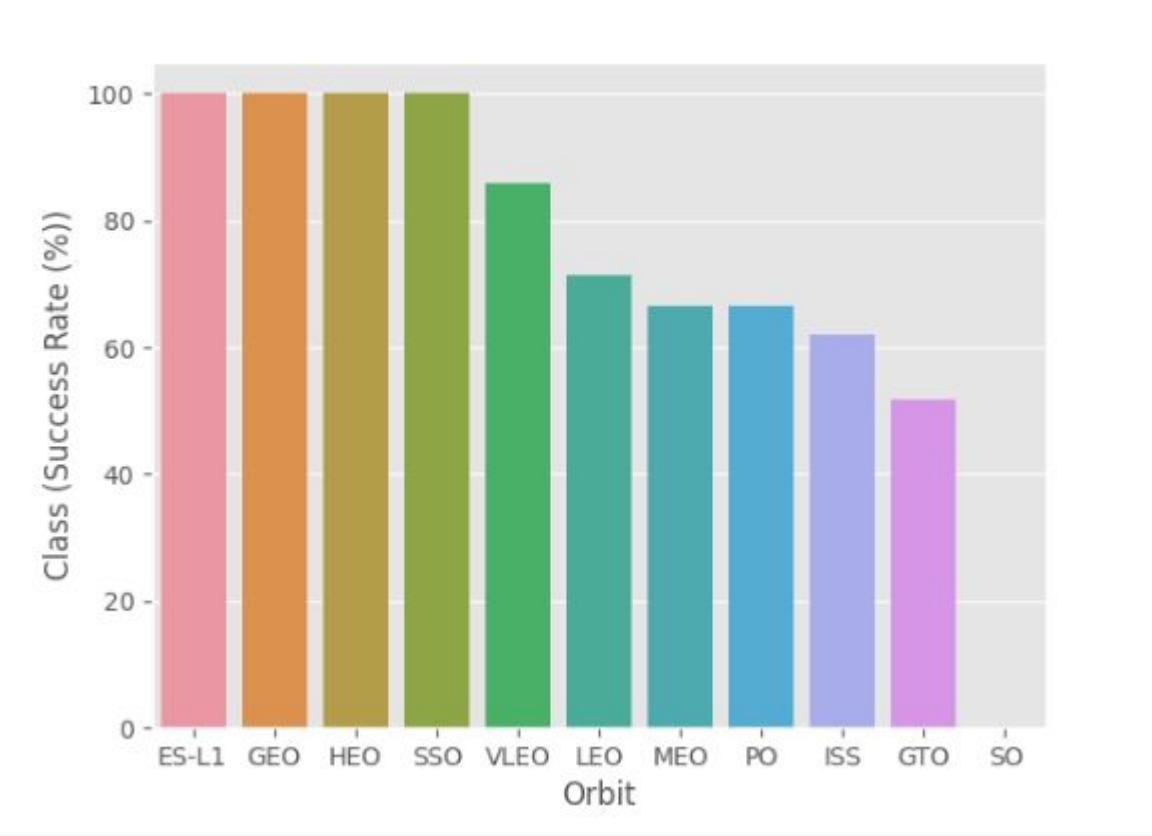
Flight Number vs. Launch Site



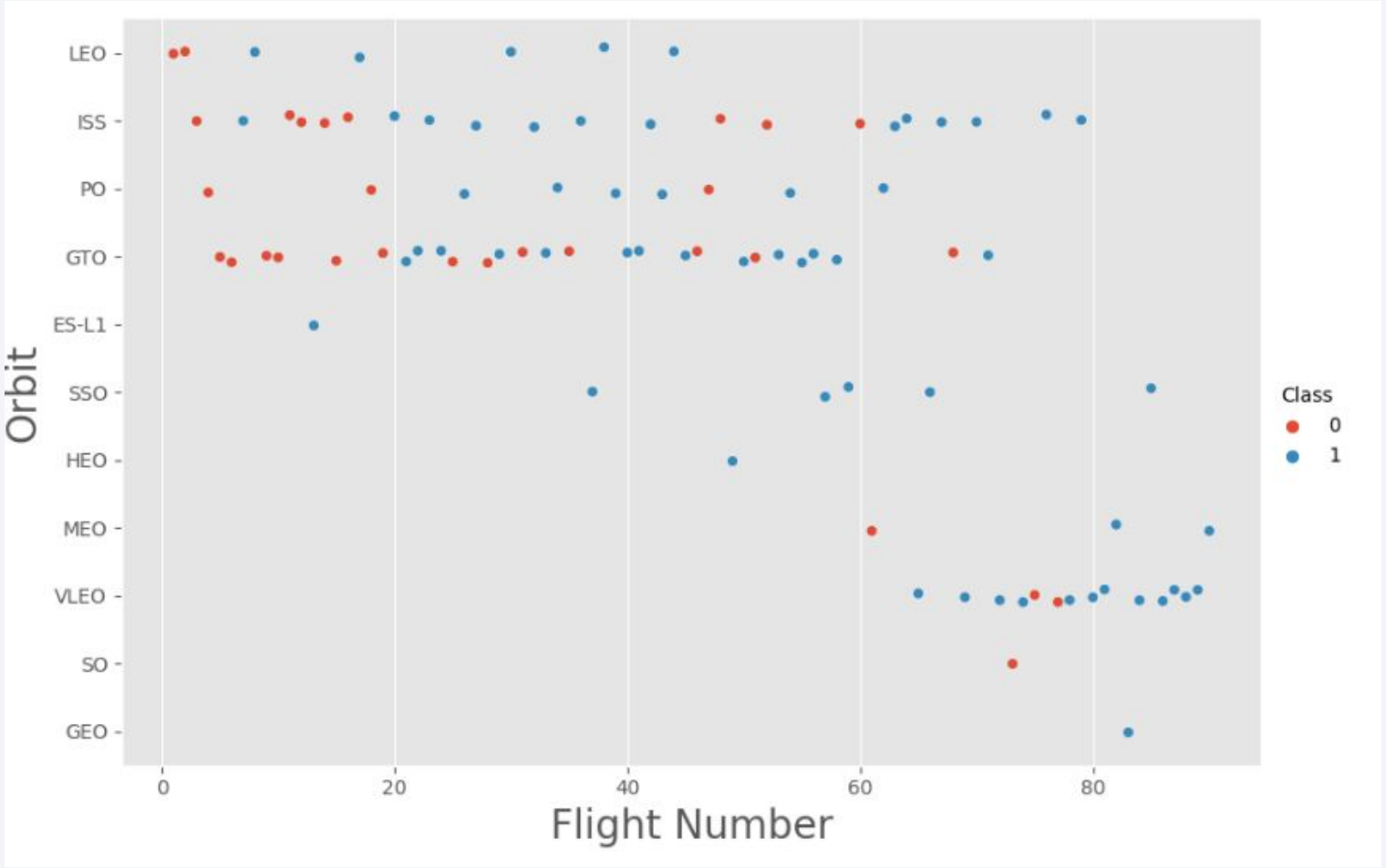
Payload vs. Launch Site



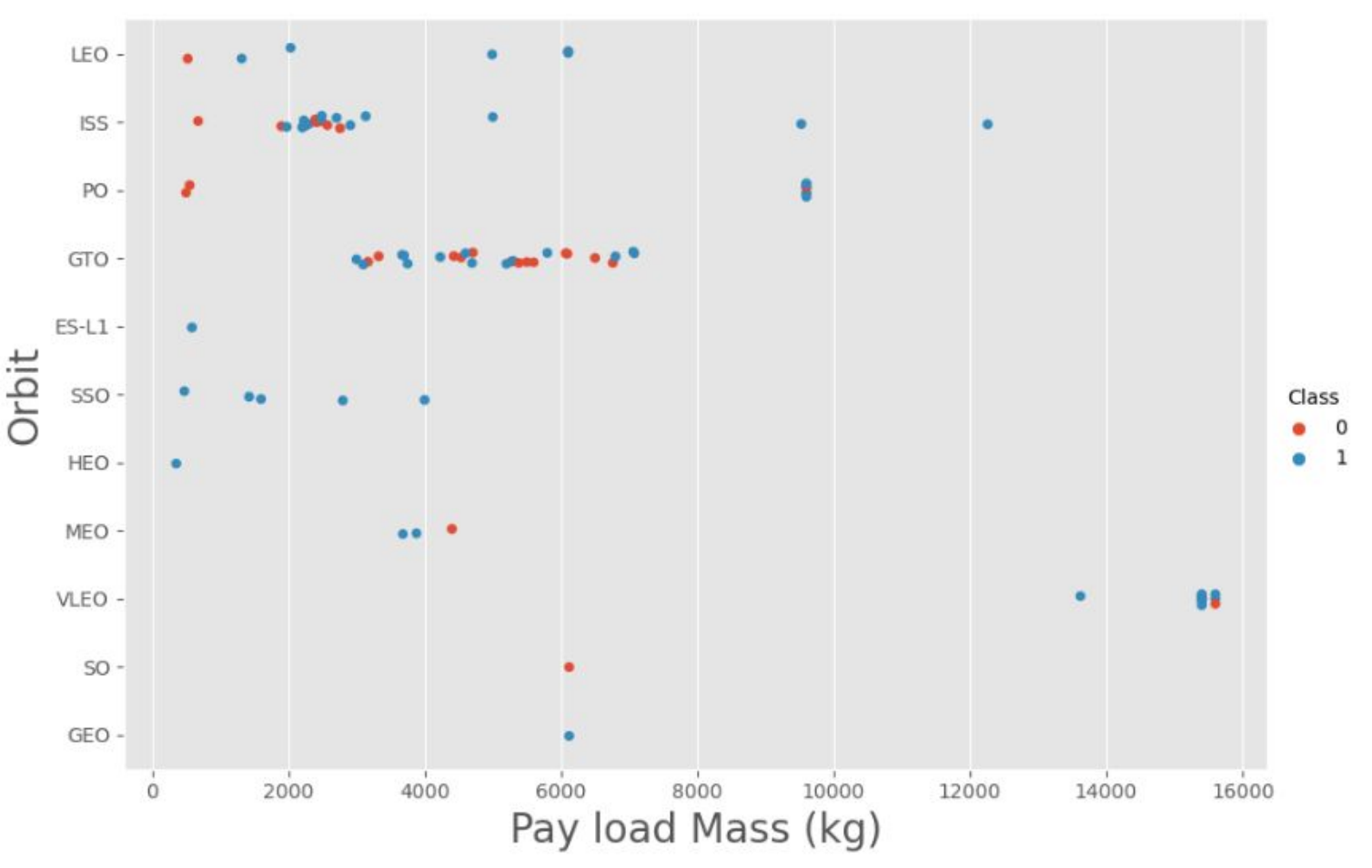
Success Rate vs. Orbit Type



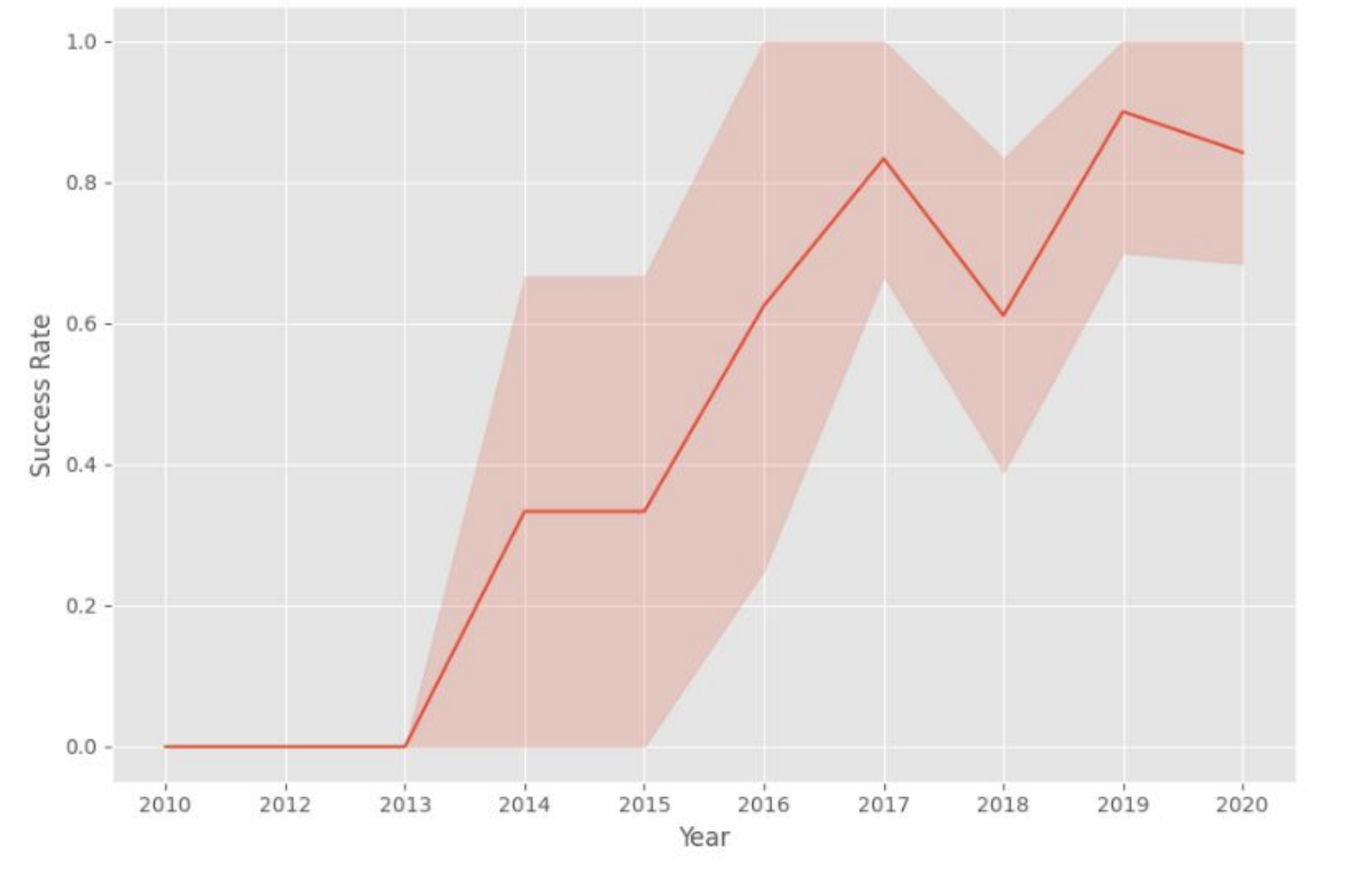
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

Task 1

Display the names of the unique launch sites in the space mission

In [31]:
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

Out[31]:

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

Launch Site Names Begin with 'CCA'

Task 2

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

In [72]:
```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

Out[72]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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	Success	Failure (parachute)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

Task 3

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

In [17]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

Out[17]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1

Task 4

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
```

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

MIN(DATE)
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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 * FROM 'SPACEXTBL'
```

```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOA
```

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

Booster_Version	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

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

Boosters Carried Maximum Payload

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

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

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

2015 Launch Records

Task 9

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.

```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome"
```

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

substr(Date,7,4)	substr(Date, 4, 2)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing _Outcome
2015	01	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
2015	04	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	Success	Failure (drone ship)

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

Task 10

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

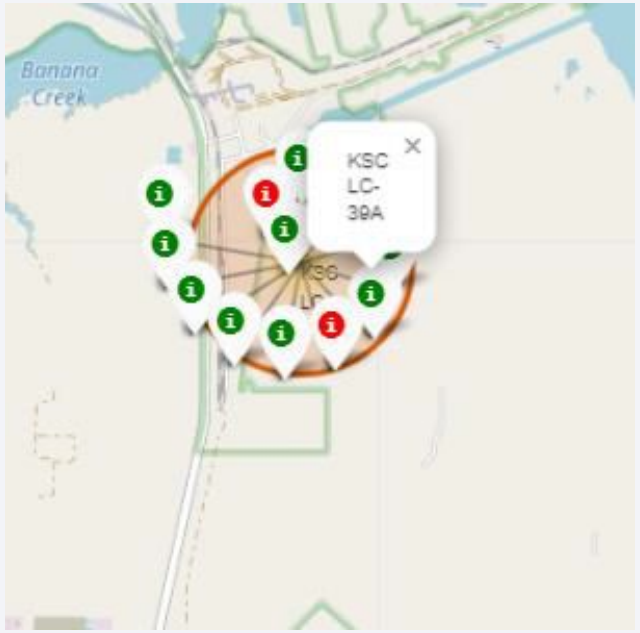
```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
19-02-2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-10-2020	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600	LEO	SpaceX	Success	Success
18-08-2020	14:31:00	F9 B5 B1049.6	CCAFS SLC-40	Starlink 10 v1.0, SkySat-19, -20, -21, SAOCOM 1B	15440	LEO	SpaceX, Planet Labs, PlanetIQ	Success	Success
18-07-2016	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-04-2018	22:51:00	F9 B4 B1045.1	CCAFS SLC-40	Transiting Exoplanet Survey Satellite (TESS)	362	HEO	NASA (LSP)	Success	Success (drone ship)

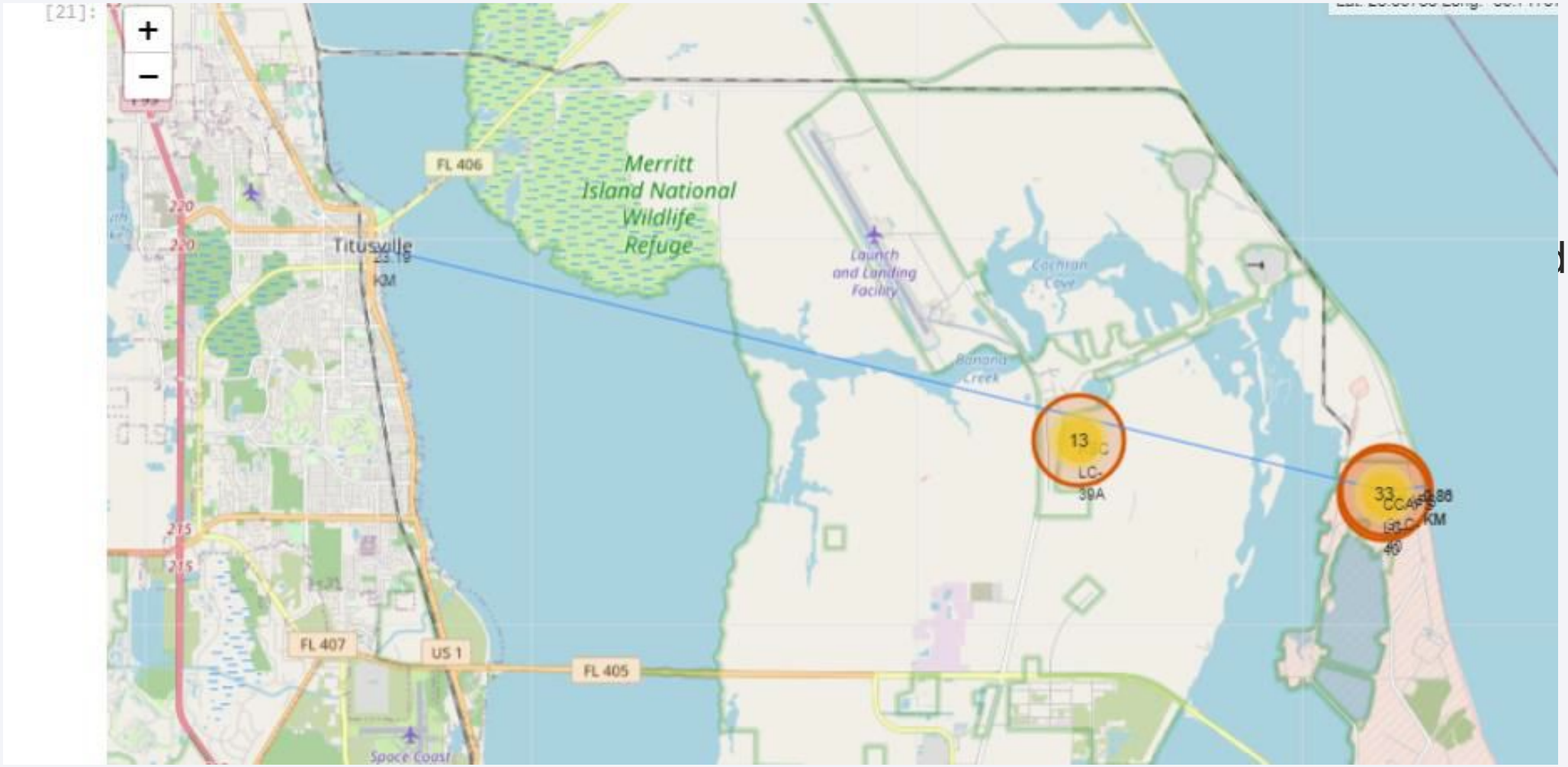

Section 3
Launch Sites
Proximities Analysis

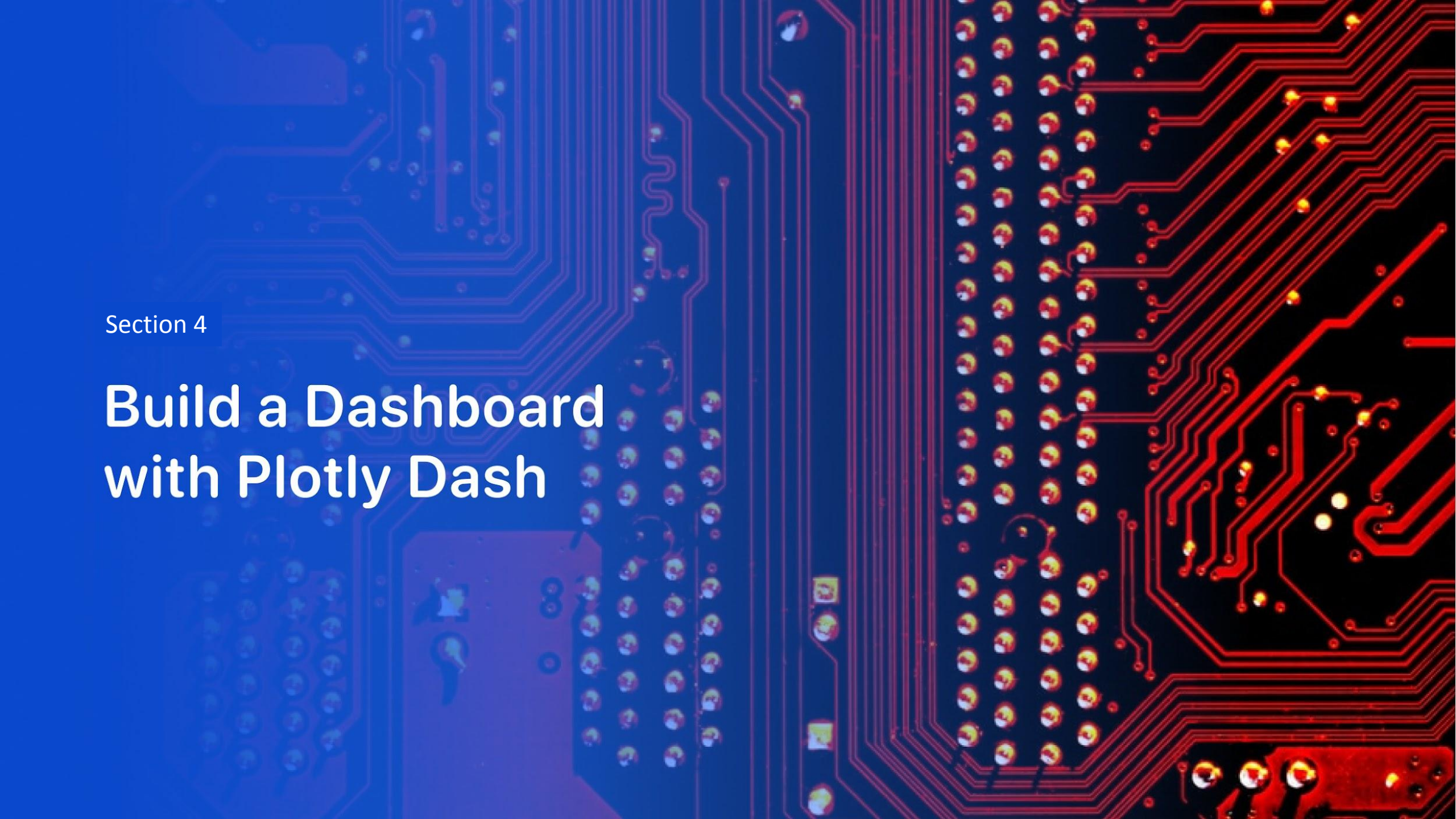
<Launch Sites Clusters plots>



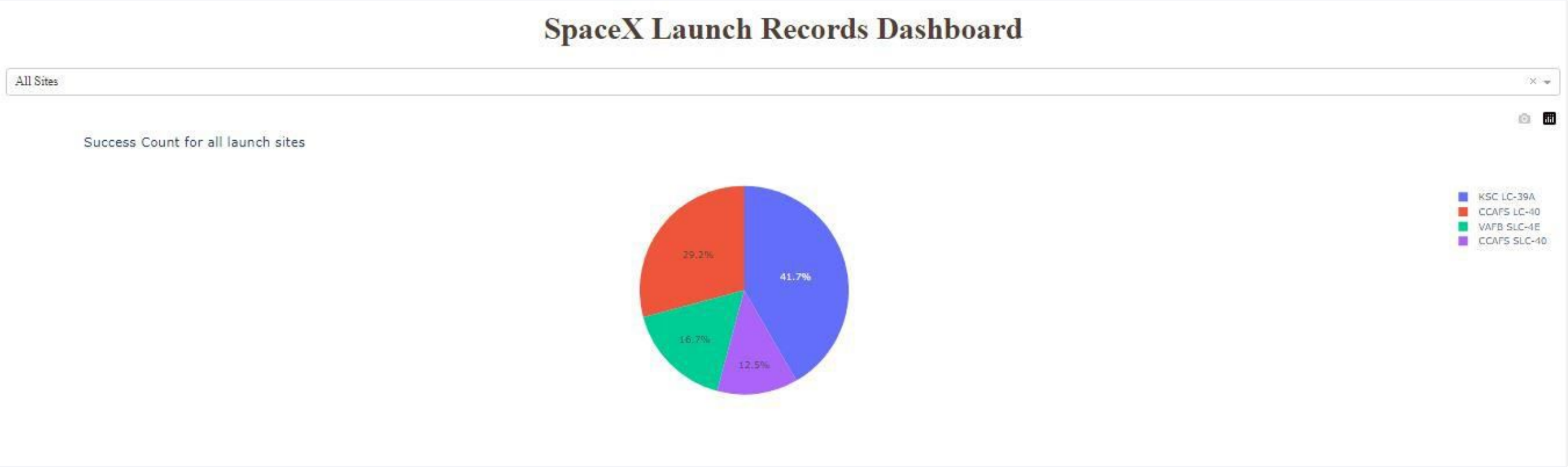
<Launch Sites Distance plot>

<Launch Sites distance plot>

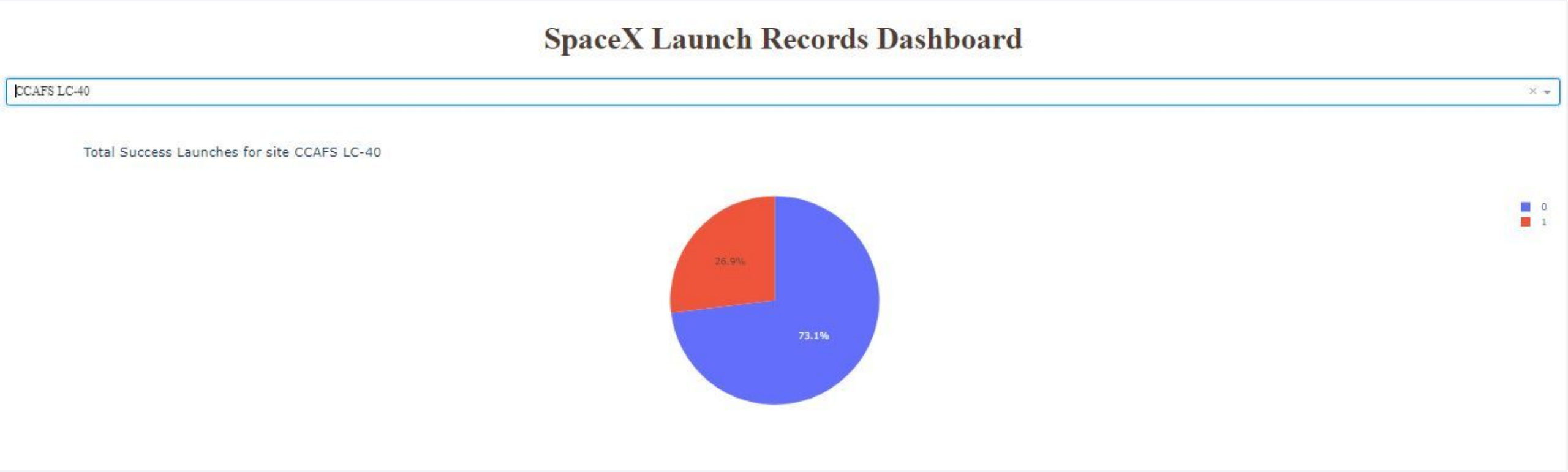



Section 4
Build a Dashboard with Plotly Dash

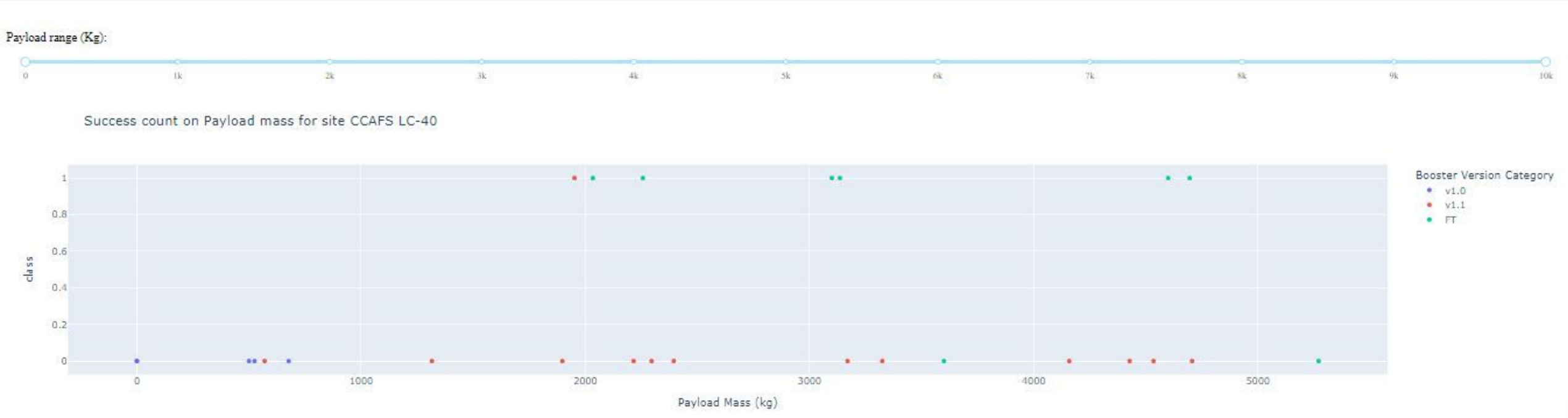
Success Count for all launch sites (Pie chart)



Total Success Launches for site CCAFS LC-40 (Pie Chart)



Success count on payload mass for site CCAFS LC-40 (scatter plot)



Section 5

Predictive Analysis (Classification)

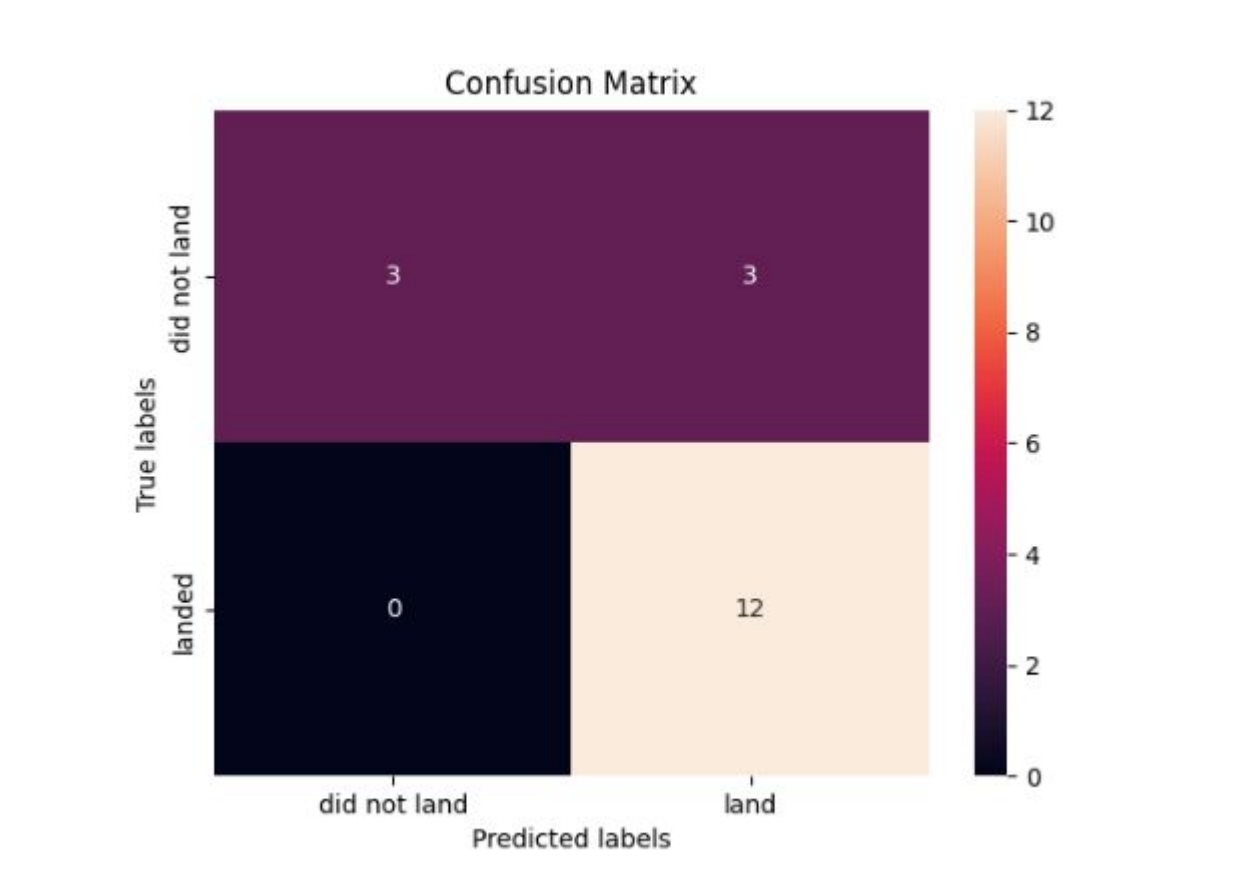
Classification Accuracy

Out[68]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

All the methods perform equally on the test data: i.e. They all have the same accuracy of 0.833333 on the test Data

Confusion Matrix



Conclusions

- A trend emerges when observing each of the three launch sites: as the flight number rises, so does the success rate. For instance, post the 50th flight, VAFB SLC 4E maintains a 100% success rate. Both KSC LC 39A and CCAFS SLC 40 reach a 100% success benchmark after their 80th flight.
- Examining the Payload vs. Launch Site scatter chart, it's evident that the VAFB-SLC launch site hasn't embarked on launches for heavy payloads (exceeding 10,000).
- Orbits ES-L1, GEO, HEO, and SSO consistently demonstrate top-tier success rates of 100%. In contrast, the SO orbit trails behind, only achieving roughly a 50% success rate, with certain instances of a 0% success rate.
- For the LEO orbit, success seems to correlate with the number of flights. Yet, this correlation doesn't hold true for the GTO orbit.
- For substantial payloads, successful landings or high landing success rates are predominantly observed for Polar, LEO, and ISS orbits. However, in the case of GTO, the distinction isn't as clear, with both successful and unsuccessful landings being present.
- Moreover, from 2013 onwards, there's been a consistent uptrend in the success rate, peaking in 2020.

Thank you!