IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

ABDERRAZZAK EL-HADRY
02 March 2023

Outline

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

Executive Summary

- Summary of methodologies :
  - Data Collection with an API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Data Analysis Using SQL
  - Data Analysis with Data Visualization
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics
  - Predictive Analytics result

Introduction

- **Project background and context**

  SpaceX is a space transportation and aerospace manufacturer, it advertises Falcon 9 rocket launches on its website, with a cost of $62 million while other providers cost over $165 million each, thanks to the possibility of reusing the first stage of the launch. In this project, we will predict whether Falcon 9's first stage will be successful using machine learning models as well as the cost of each launch.

- **Problems you want to find answers**
  - What factors influence the landing outcome?
  - What features control the success rate of landing?
  - What conditions lead to a successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected with API and Webscraping
- Perform data wrangling
  - One-hot encoding was applied for Categorical Data
- 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, and evaluate classification models (LR, KNN, SVM, DT)

Data Collection

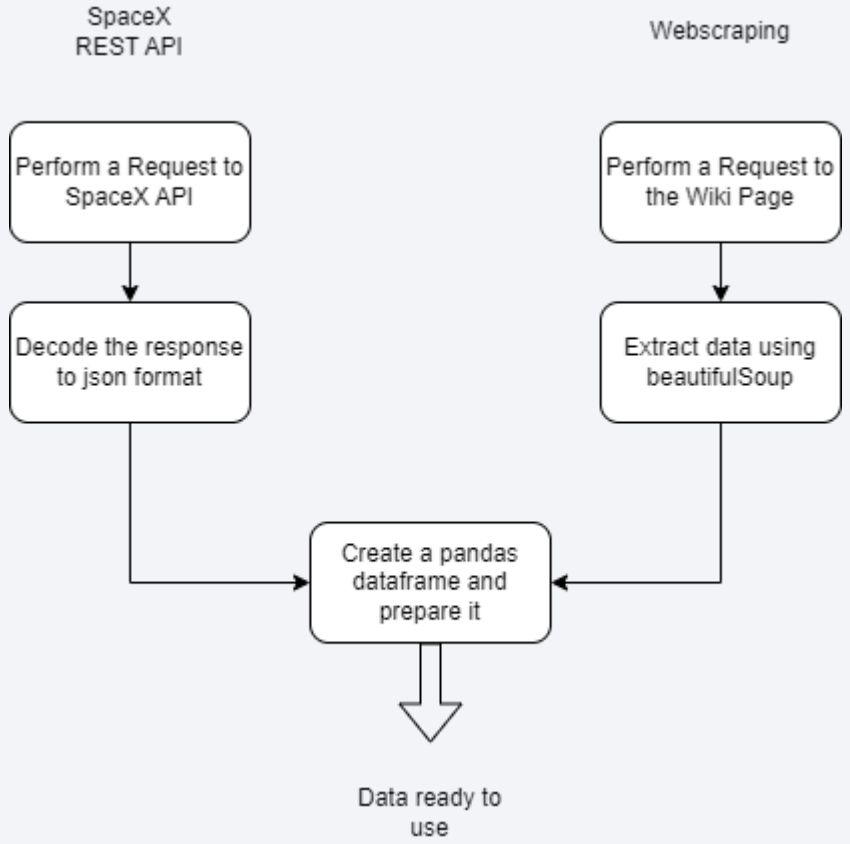
The process of **gathering** and analyzing **data** from various sources to find answers to problems, The dataset used in this project was collected with SpaceX API and Webscraping from a Wikipedia page.

- SpaceX API :

  First, we perform a request from SpaceX API, then we decode the response as Json format and convert it to a pandas data frame using **.json_normalize(),** finally, we applied a data cleaning process to prepare data.

- Webscraping :

  We request the falcon9 launch Wiki page from its URL, then we create a **BeautifulSoup** object from the HTML response to extract data and transform it into a data frame.



Data Collection – SpaceX API

- Data Collection with SpaceX REST API

  https://github.com/4bdex/SpaceX_flacon_9/blob/main/data_collection_API.ipynb

Now let's start requesting rocket launch data from SpaceX API with the following URL:

```
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())
```

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number,
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
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 lis
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 leav
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)]
```

Data Collection - Scraping

- Data Collection with Webscraping

  https://github.com/4bdex/SpaceX_falcon_9/blob/main/data_collection_webscraping.ipynb

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falc
```

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
data = BeautifulSoup(response.content,'html.parser')
```

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = data.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 ge
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a
th = data.find_all('th')
for row in range(len(th)):
    try:
        name = extract_column_from_header(th[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

Data Wrangling

- Data Wrangling is the process of removing errors and combining complex datasets to make them easier to access and analyze. Exploratory Data Analysis (EDA) is used to find patterns in the data and determine what would be the label for training supervised models.

- Data Wrangling notebooks:

  https://github.com/4bdex/SpaceX_flacon_9/blob/main/Data_wrangling.ipynb

  https://github.com/4bdex/SpaceX_flacon_9/blob/main/data_collection_API.ipynb

```
# Calculate the mean value of PayloadMass column
mean_payloadMass=data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].fillna(mean_payloadMass,inplace=True)
```

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

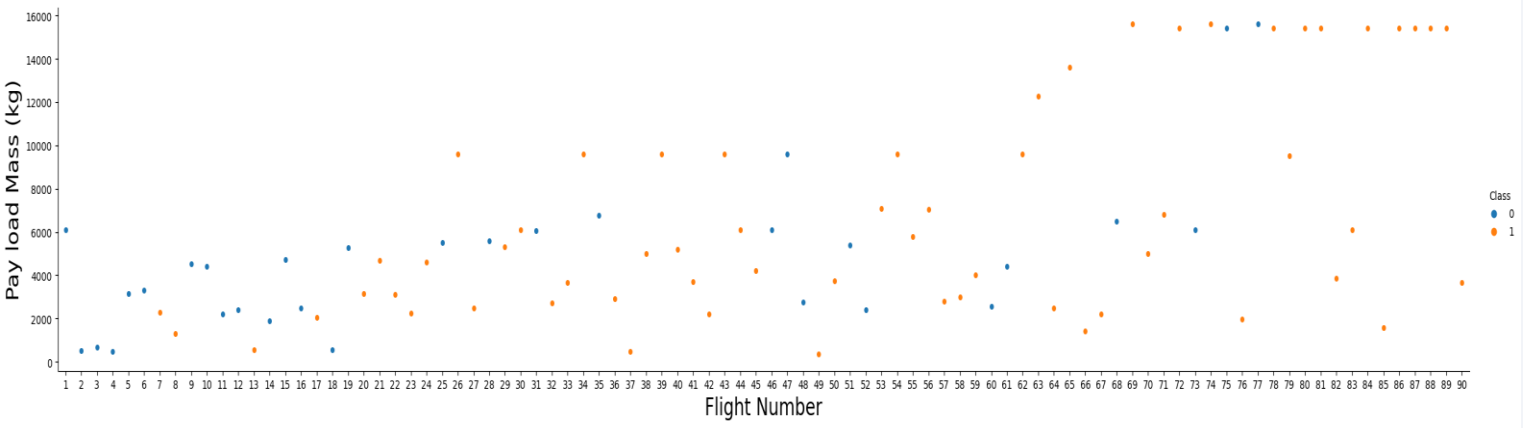
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for i,outcome in enumerate(df.Outcome):
    if outcome in list(bad_outcomes):
        landing_class.append(0)
    else: landing_class.append(1)
```

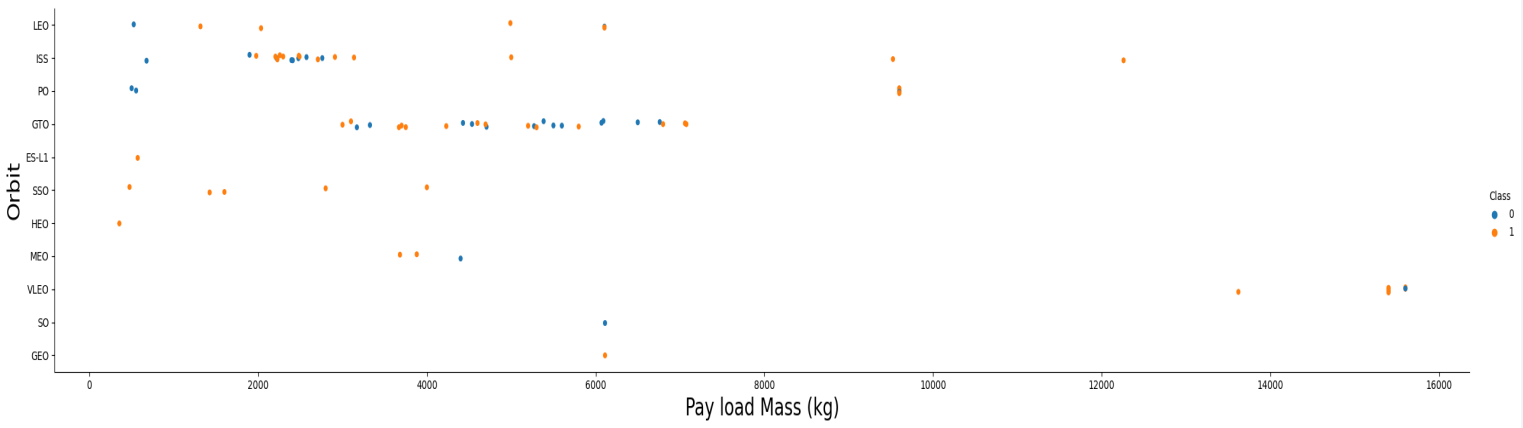
We can use the following line of code to determine the success rate:

```
df["Class"].mean()
```

```
0.6666666666666666
```

EDA with Data Visualization





A scatter plot is a chart type that is normally used to observe and visually display the relationship between variables or patterns when the data are taken as a whole.

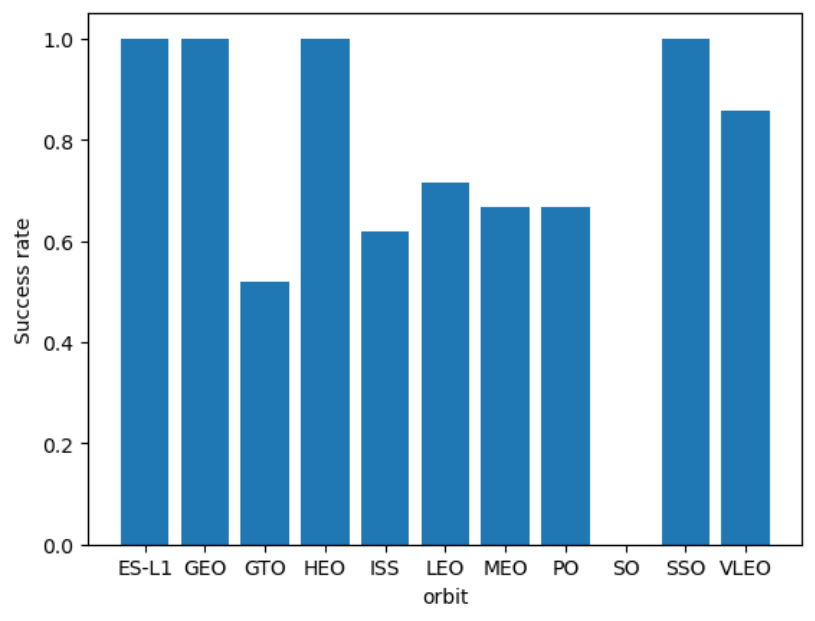
Relations visualized :

- Payload and Flight Number.
- Flight Number and Launch Site.
- Payload and Launch Site.
- Flight Number and Orbit Type.
- Payload and Orbit Type

Data Visualization with EDA:

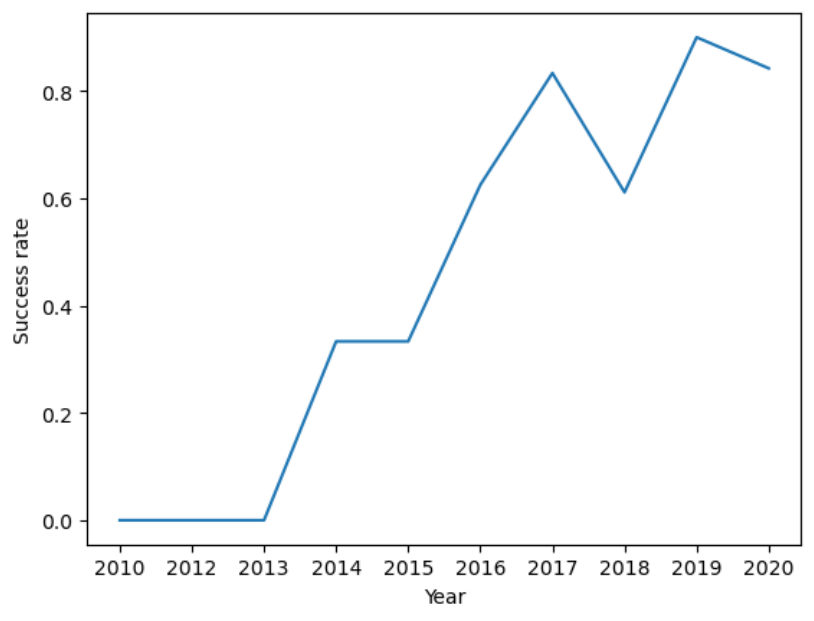
https://github.com/4bdex/SpaceX_flacon_9/blob/main/EDA_Data_Visualization%20.ipynb

EDA with Data Visualization



A bar graph is used to compare data across different categories.

- This bar graph compares the probability of success between orbits.



A line graph is a graph that uses lines to connect individual data points.

- This line graph describes how probabilities of success change over years.

Data Visualization with EDA:
https://github.com/4bdex/SpaceX_flacon_9/blob/main/EDA_Data_Visualization%20.ipynb

EDA with SQL

- The SQL queries performed in this project :
  - Display the names of the unique launch 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 the ground pad was achieved.
  - List the names of the boosters which have success in drone ships and have payload mass greater than 4000 but less than 6000.
  - List the total number of successful and failed 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, failure landing_outcomes in drone ship, booster versions, and launch_site for the months in the year 2015.
  - Rank the count of successful landing outcomes between the dates 04-06-2010 and 20-03-2017 in descending order.

EDA with SQL:

https://github.com/4bdex/SpaceX_flacon_9/blob/main/EDA_sql_sqllite.ipynb

Build an Interactive Map with Folium

Tasks performed to build an interactive Map using Folium:

1. Mark all launch sites on a map.
2. Mark the success/failed launches for each site on the map.
3. Calculate and draw the distances between a launch site to its proximities.

Questions to be answered :

- Are launch sites in close proximity to railways/highways/coastline?
- Do launch sites keep a certain distance away from cities?

Interactive Map with Folium :

https://github.com/4bdex/SpaceX_flacon_9/blob/main/Launch_Sites_Locations_Analysis_Folium.ipynb

Build a Dashboard with Plotly Dash

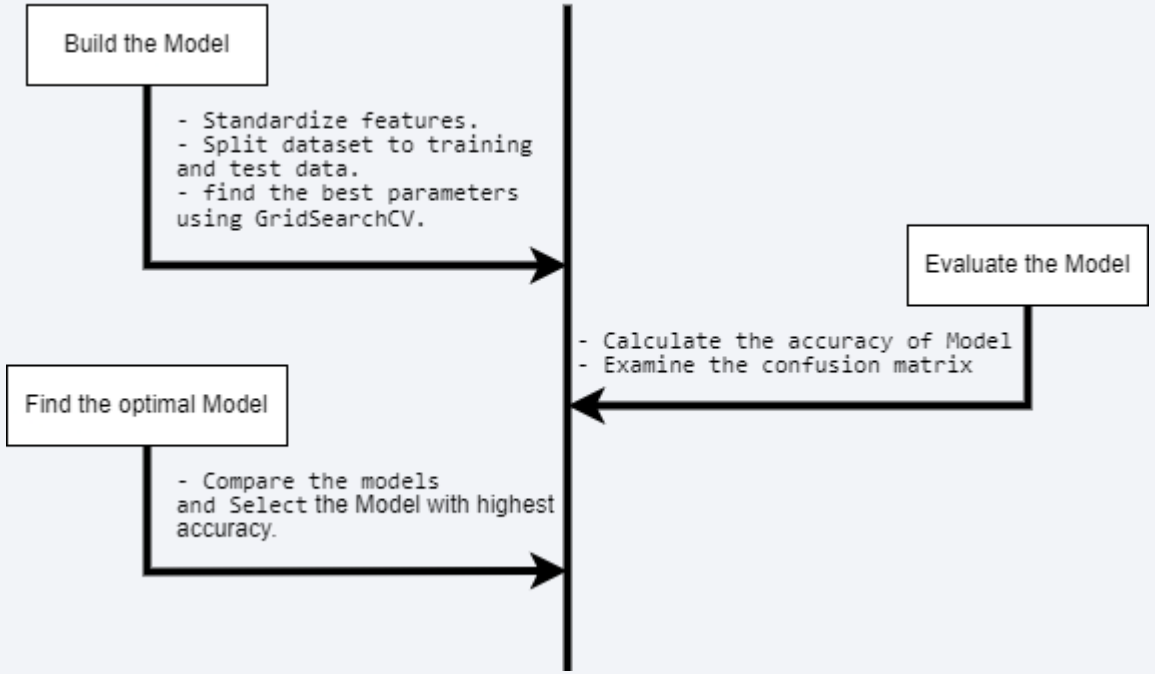
- We built an interactive dashboard with a plotly dash that allows us to visualize data.
- For plotting, we used a Pie chart to show the total successful launches count for sites and a scatter plot to show the correlation between payload and launch success.

Interactive Dashboard with Ploty Dash：

https://github.com/4bdex/SpaceX_flacon_9/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

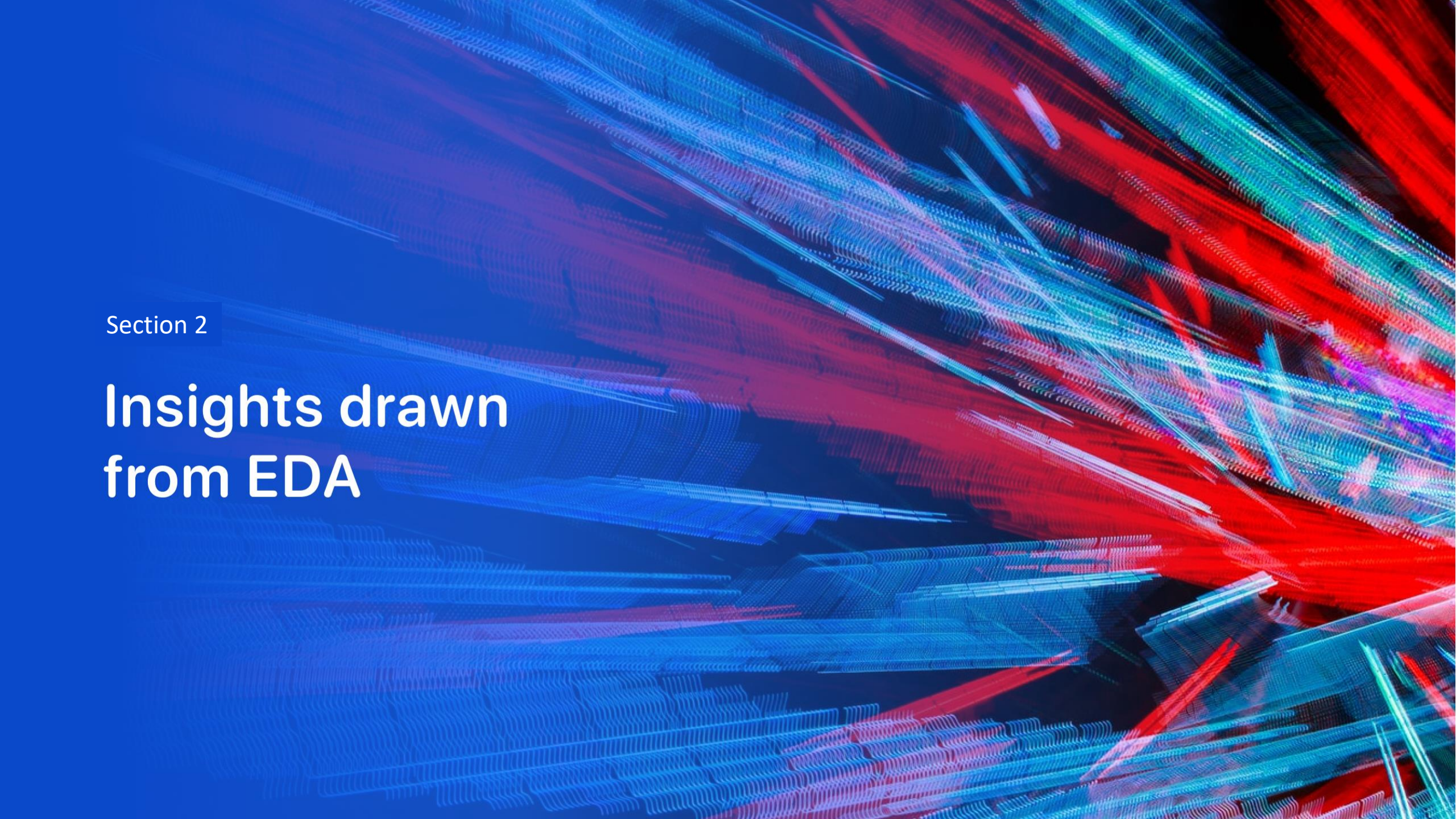
- We will find the best Hyperparameter for SVM, Classification Trees, and Logistic Regression and select the optimal Model.



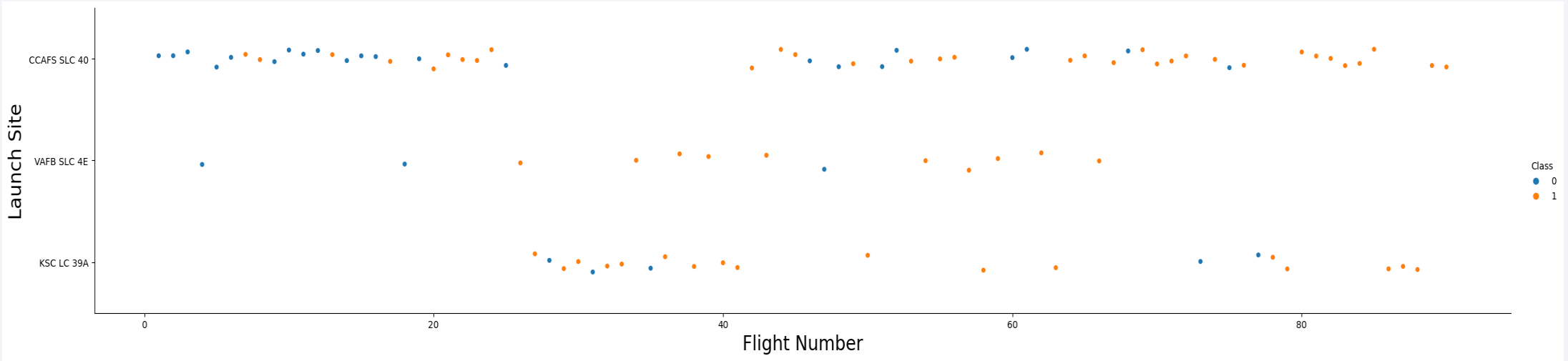
- SpaceX_Machine Learning Prediction :
  - https://github.com/4bdex/SpaceX_flacon_9/blob/main/Machine_Learning_Prediction.ipynb

Results

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

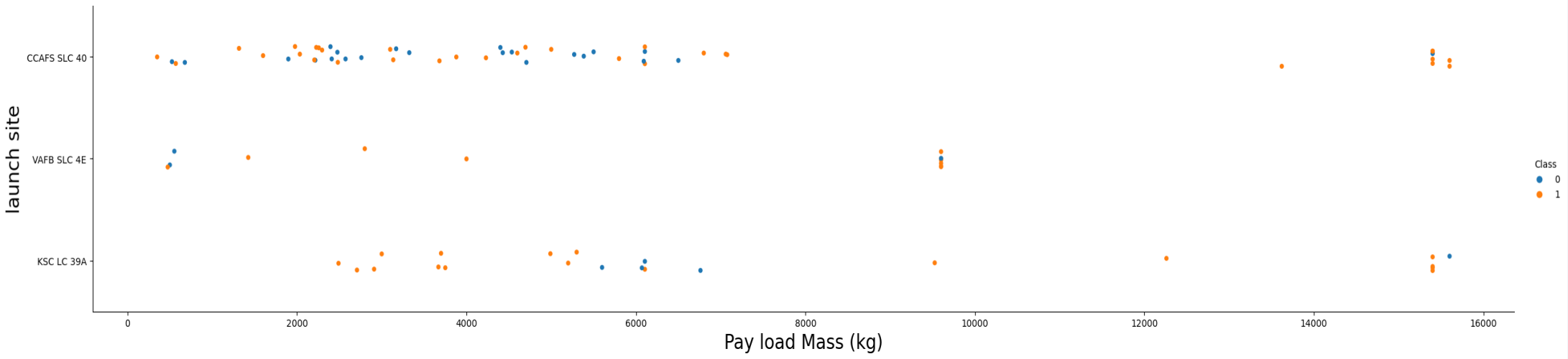

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- We see that different launch sites have different flight Numbers. The success rate of each launch is increasing over the number of flights.

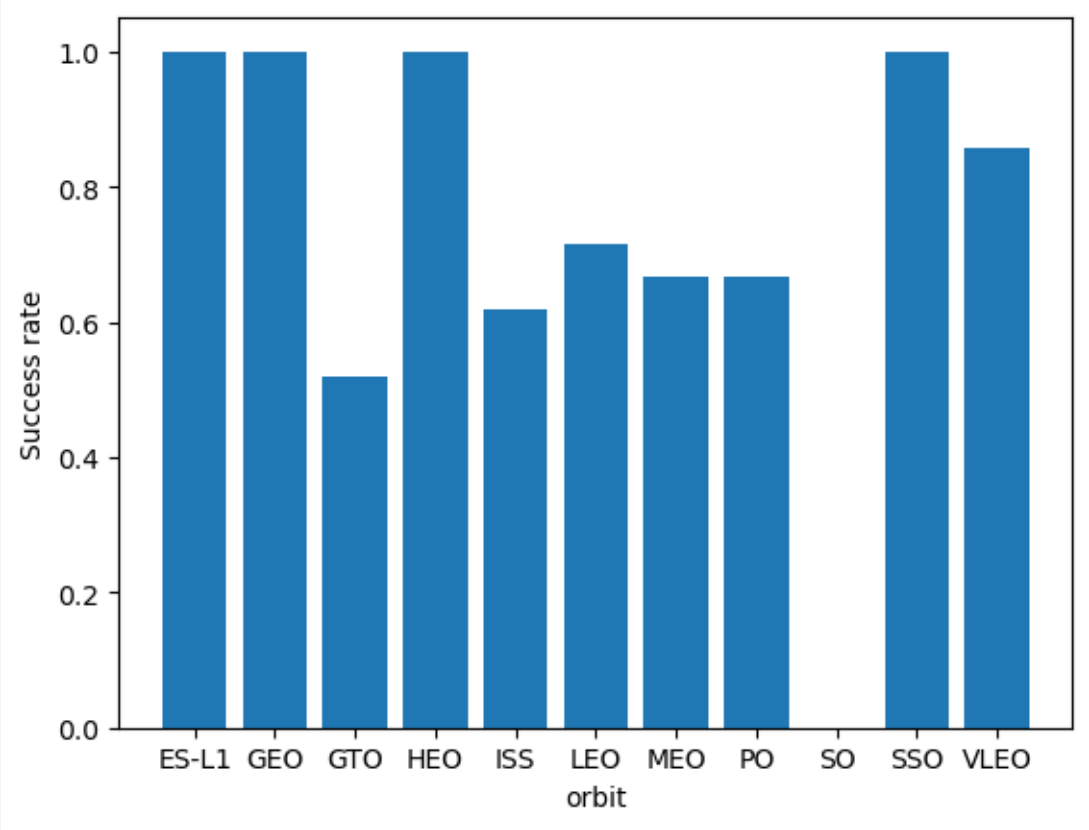
Payload vs. Launch Site



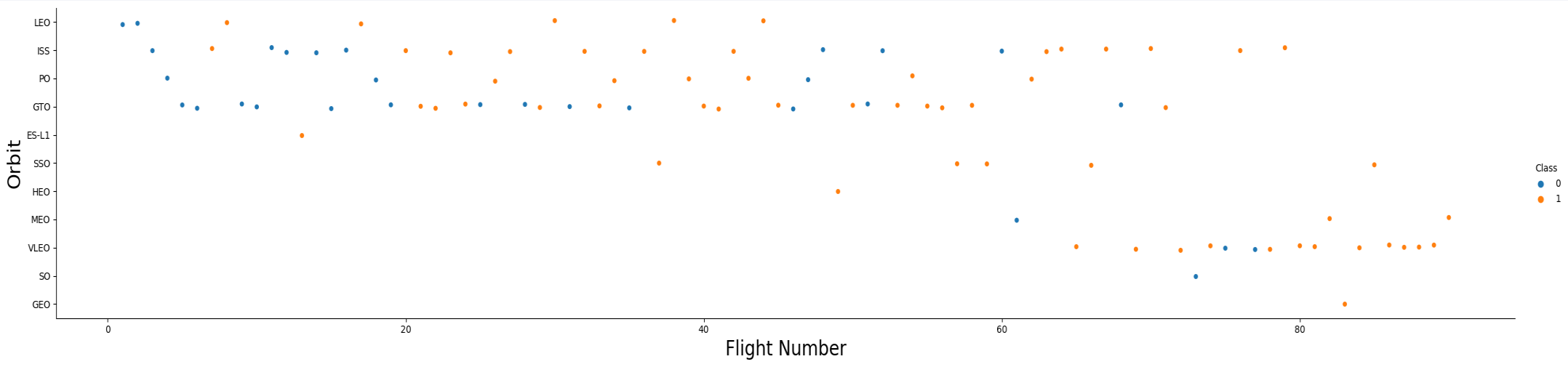
- We see that there are no rockets launched for heavy payload mass (greater than 10000).

Success Rate vs. Orbit Type

- We can see that the most successful orbits are :
- ES-L1
- GEO
- HEO
- SSO
- On the other side, all the flights of SO seem to be failed.

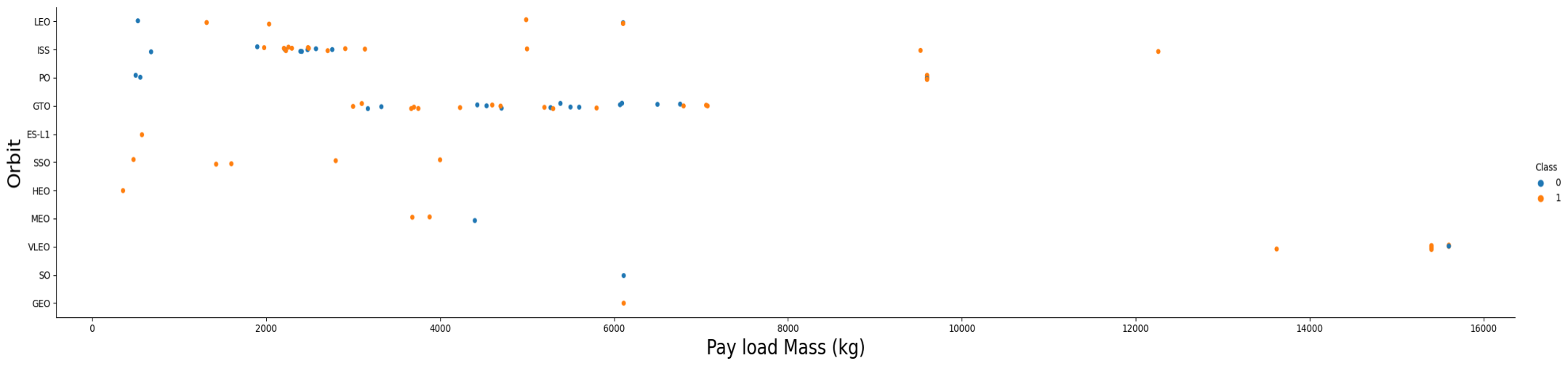


Flight Number vs. Orbit Type



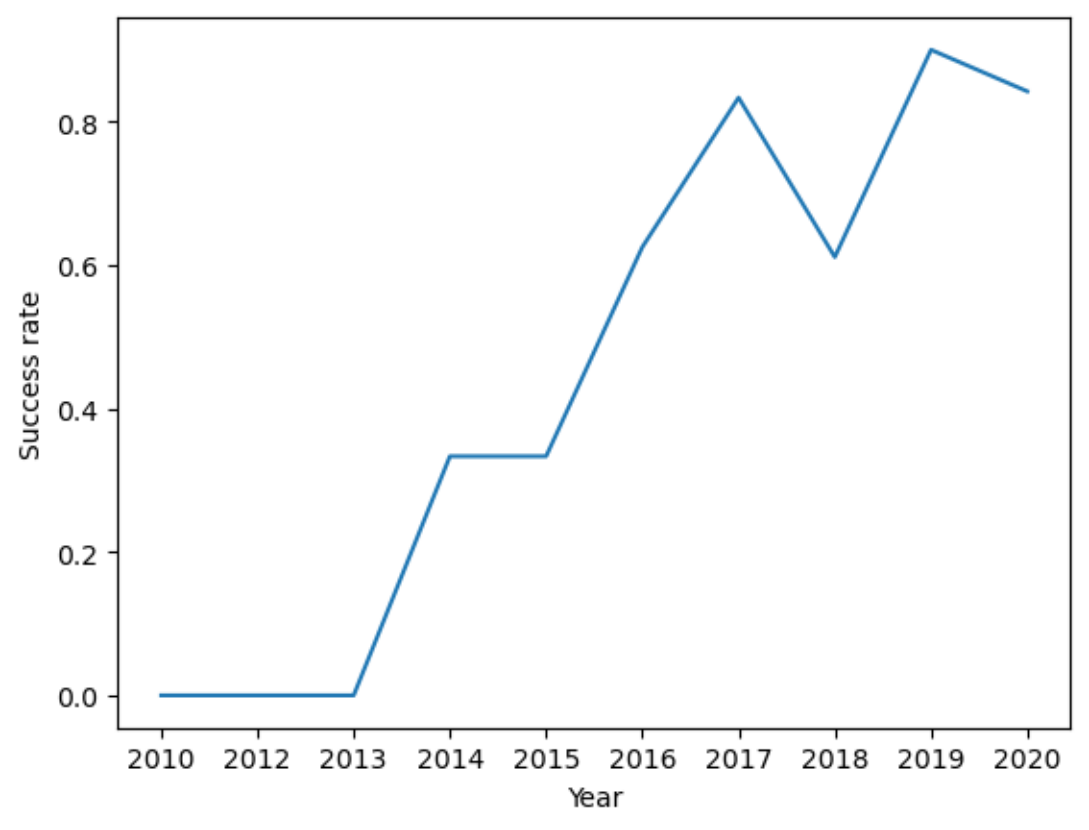
- Successful landing rate is higher for Polar, LEO, and ISS with heavy loads.
- However for GTO we cannot separate this well like two unsuccessful missionaries out there both here.

Payload vs. Orbit Type



- We can see that the LEO orbit correlates to the number of flights.
- On the other hand, there seems to be no relationship between flight number and GTO orbit.

Launch Success Yearly Trend



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

All Launch Site Names

```
%sql SELECT distinct Launch_Site FROM SPACEXTBL;
```

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

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

- We can use “distinct” to get the unique values in column Launch_site.

Launch Site Names Begin with 'CCA'

```
%sql SELECT * FROM SPACEXTBL 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
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

- We can see that the launch sites displayed have a successful mission outcome.

Total Payload Mass

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

```
%sql select sum(PAYLOAD_MASS__KG_) from spacextbl where customer = 'NASA (CRS)' ;
```

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

sum(PAYLOAD_MASS__KG_)
45596

- the total payload carried by boosters from NASA is 45596 kg.

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.1 is 2928.4 kg

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

```
%sql select avg(PAYLOAD_MASS__KG_) from spacextbl where Booster_Version = 'F9 v1.1' ;
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

```
%sql select min(date) from spacextbl where [Landing _Outcome] = 'Success (ground pad)';
```

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

min(date)
01-05-2017

- the first successful landing outcome in the ground pad was achieved on 01 May 2017.

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 FROM SPACEXTBL WHERE [LANDING _OUTCOME] = 'Success (drone ship)' AND  PAYLOAD_MASS__KG_ between 4000
```

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

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

- We used the WHERE clause to select only the boosters which have successfully landed on a drone ship.
- The between clause was used to filter the payload mass.

Total Number of Successful and Failure Mission Outcomes

- ***the total number of successful and failure mission outcomes***

```
%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) as  total_number   FROM SPACEXTBL GROUP BY MISSION_OUTCOME;
```

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

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

- The total number of successful missions is 100 while we have one failed mission outcome.

Boosters Carried Maximum Payload

- the names of the booster which have carried the maximum payload mass

```
%sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE  PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from spacextbl);
```

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

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

- We used a subquery to get the maximum payload mass.

2015 Launch Records

- The failed landing outcomes in drone ships in the year 2015

```
%sql select substr(Date, 4, 2) as month,[LANDING _OUTCOME], booster_version,launch_site from spacextbl \
where substr(Date,7,4)='2015' and [Landing _Outcome] = 'Failure (drone ship)';
```

```
 * 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

- We used substr() to get the Month/Year only and we used the where clause to filter data.

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

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the dates 2010-06-04 and 2017-03-20, in descending order

```
%sql SELECT [LANDING _OUTCOME], COUNT([LANDING _OUTCOME]) as success_landing FROM SPACEXTBL \
WHERE DATE BETWEEN '04-06-2010' AND '20-03-2017' GROUP BY [LANDING _OUTCOME] ORDER BY success_landing DESC
```

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

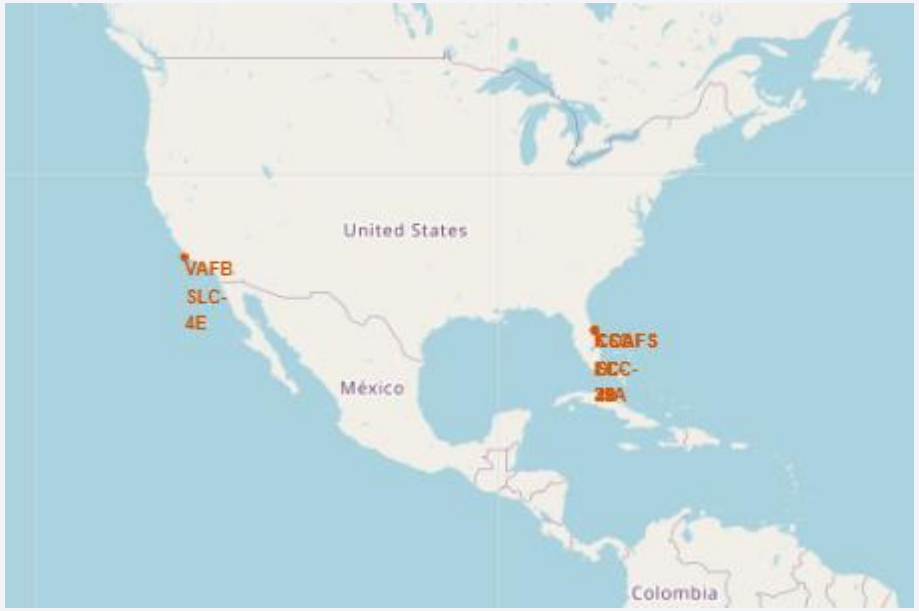
Landing _Outcome	success_landing
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

- We used the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the output in descending order.

Section 3

Launch Sites Proximities Analysis

Location of the launch sites

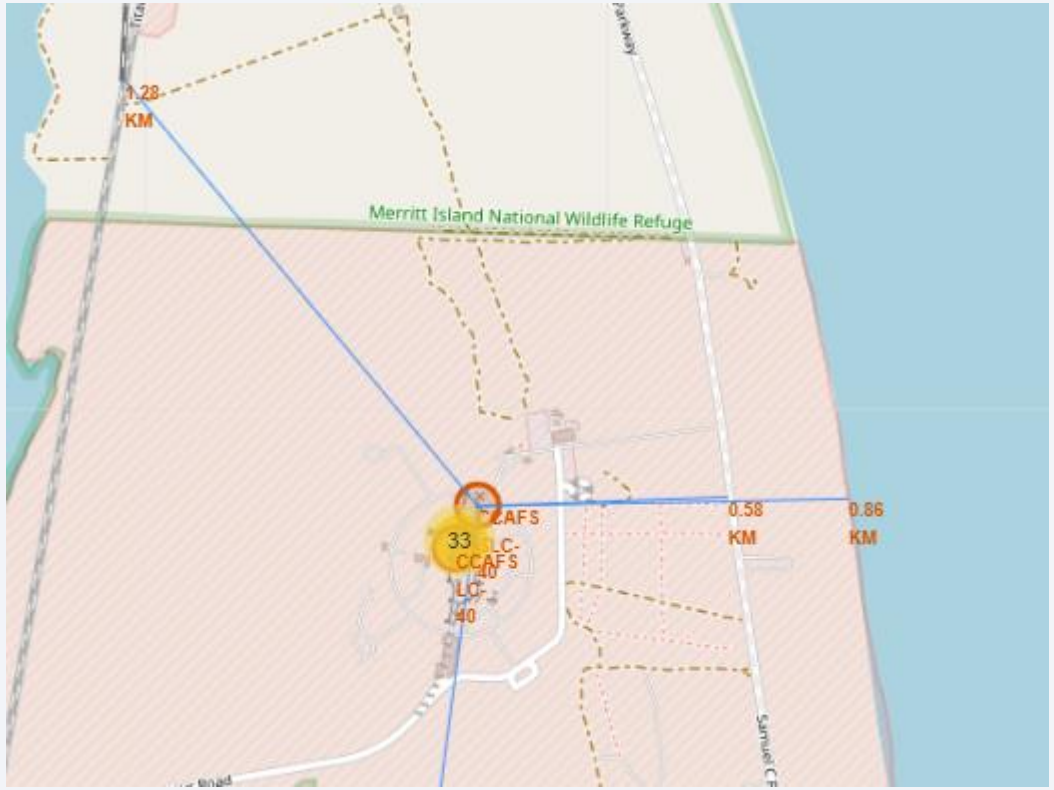


- The launch sites are in proximity to the equator and the coast.
- The launch sites are in close proximity to the coast.

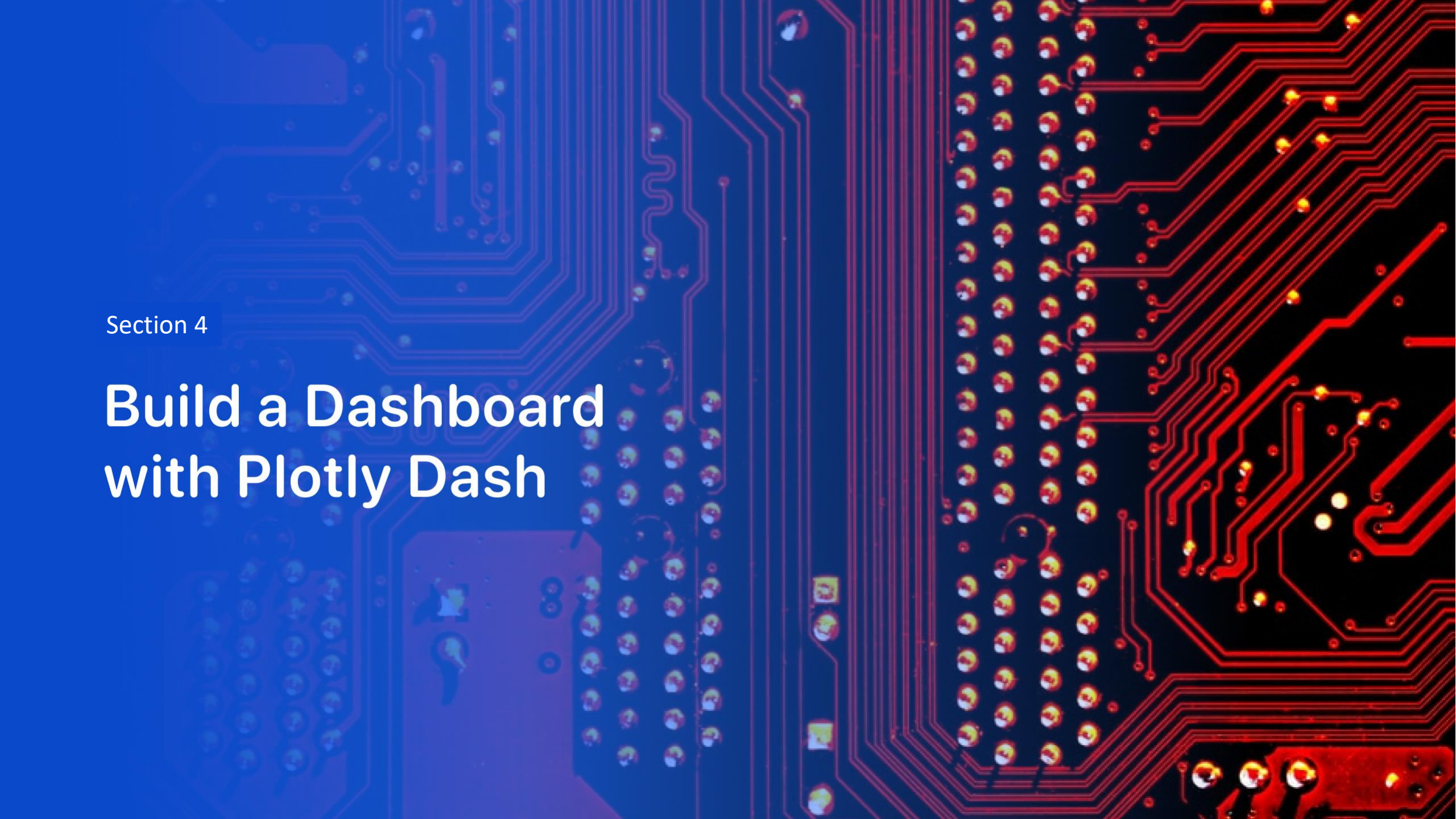
Mark the landing outcome

Green marker for the successful launches
Red marker for the failed launches

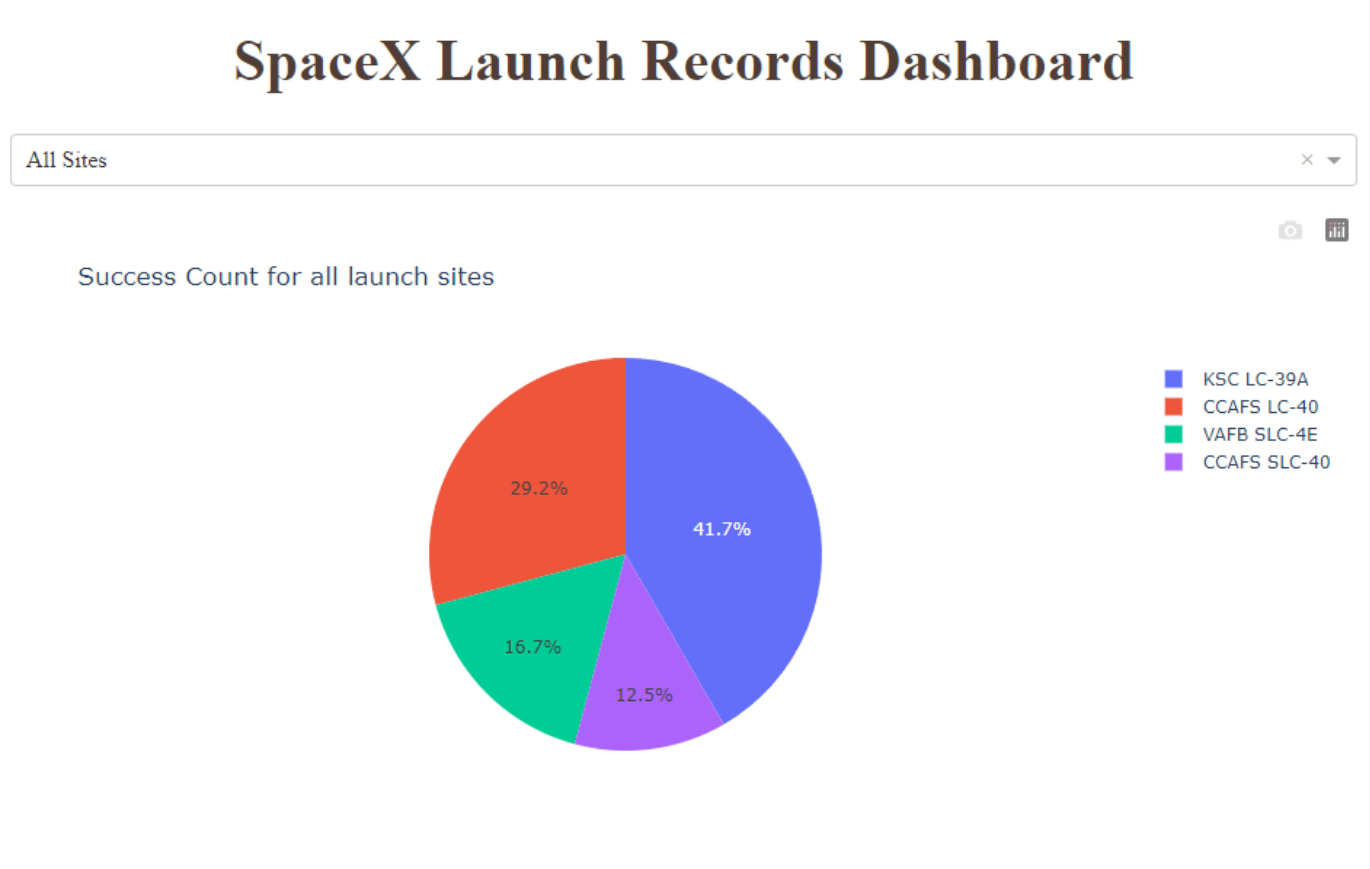
Proximities of the launch sites



- launch sites aren’t in close proximity to railways or highways. However, they are in close proximity to the coastline. This may be related to transport and safety.
- The launch sites keep a certain distance away from cities for safety reasons.

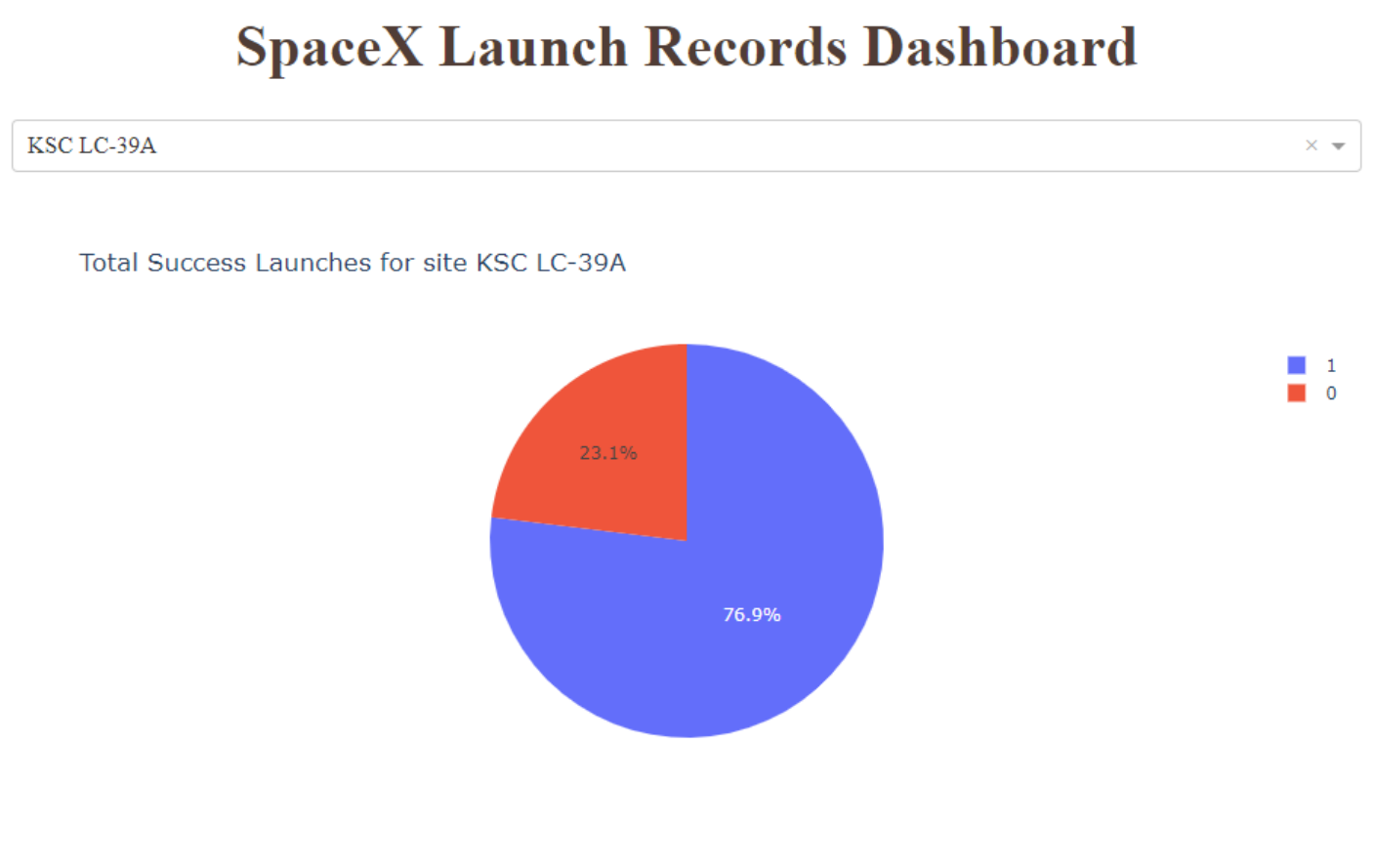

Section 4
Build a Dashboard with Plotly Dash

The success rate percentage by each sites



- We can see the percentage of the success rate of each launch site.

Most successful launch site



- We can observe the KSC LC-39A has the highest success rate with 76.9%.

Payload Mass vs Success rate

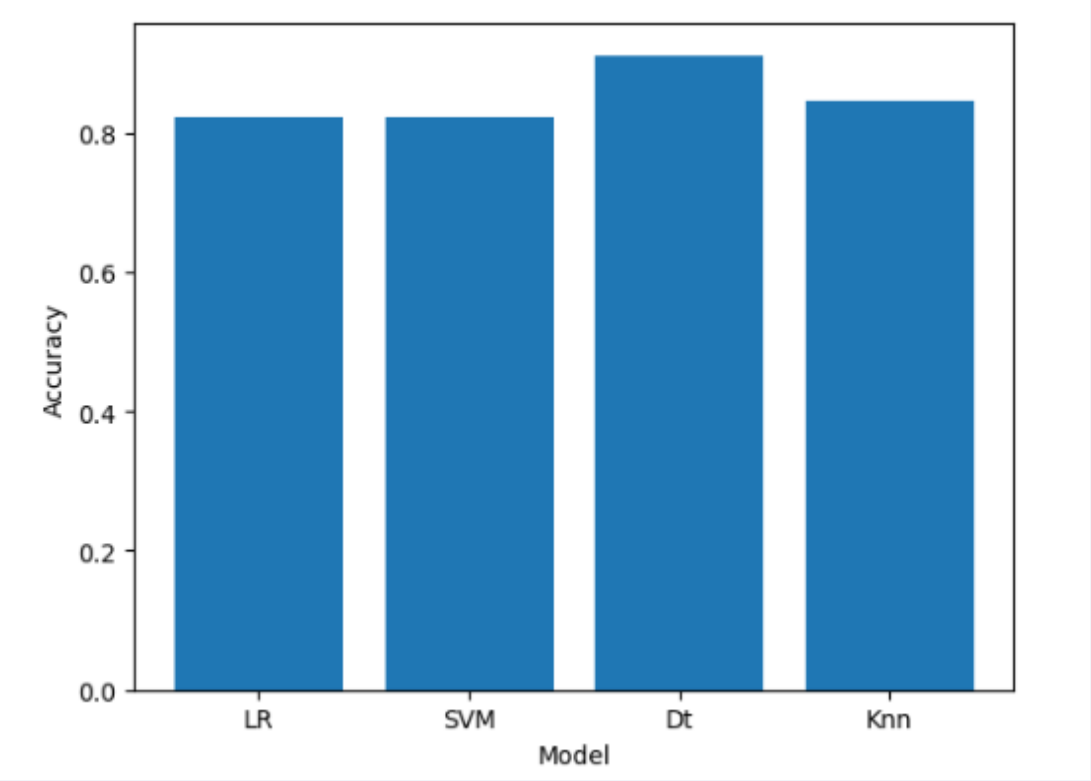
- We can see that lower payload mass sites are more successful than heavy payload mass.

Section 5

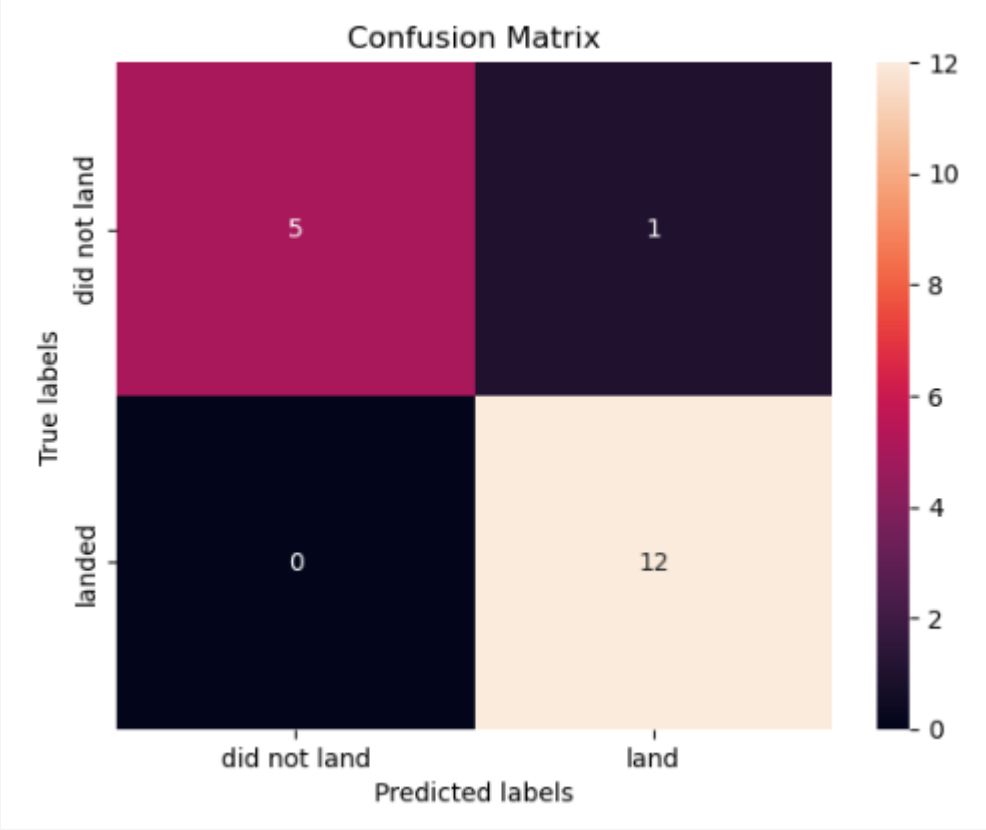
Predictive Analysis (Classification)

Classification Accuracy

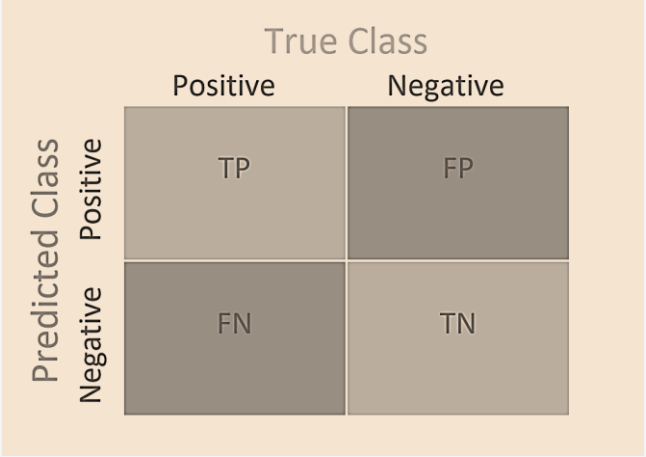
- The Decision Tree Algorithm has the highest classification accuracy among other algorithms (91.1%).



Confusion Matrix



A confusion matrix is a table that is used to define the performance of a classification algorithm.



Conclusions

- Low Payload Mass (< 4000kg) performs better than heavy loads.
- The success rates of launches is increasing over the years.
- KSC LC 39A is the most successful site.
- The most successful orbits are : ES-L1, GEO, HEO, SSO.
- The Decision tree classifier has the highest predicting accuracy for our Data.

Thank you!