IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Name>
<Date>

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis (EDA) with SQL
  - Exploratory Data Analysis (EDA) with Data Visualization
  - Interactive Visual Analytics with Folium
  - Interactive Visual Analytics with Ploty Dashboard
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis
  - Interactive Analytics
  - Predictive Analytics

Introduction

- Project background and context

  Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers

  - What factors determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Use of machine learning to build the models and use testing dataset to tune the models, and evaluate classification models with confusion matrix

Data Collection

- Data collection phrases

1. Data collection was done using get request to the SpaceX REST API.
2. Decoded the response content as a JSON function call and turn it into a pandas dataframe
3. Cleaned the data, checked for missing values and fill in missing values where necessary.
4. Performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
5. Extract the launch records as HTML table, parse the table
6. Convert it to a pandas dataframe for future analysis

Data Collection – SpaceX API

- Get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting
- The link to the notebook: https://github.com/Khong2612/testrepo/blob/3c31c982183dab7e2eb444b80c2e3488b7b2116e/1-jupyter-labs-spacex-data-collection-api.ipynb

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

In [10]:

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
```

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

In [ ]:

```
response.status_code
```

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

In [14]:

```
# 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

In [15]:

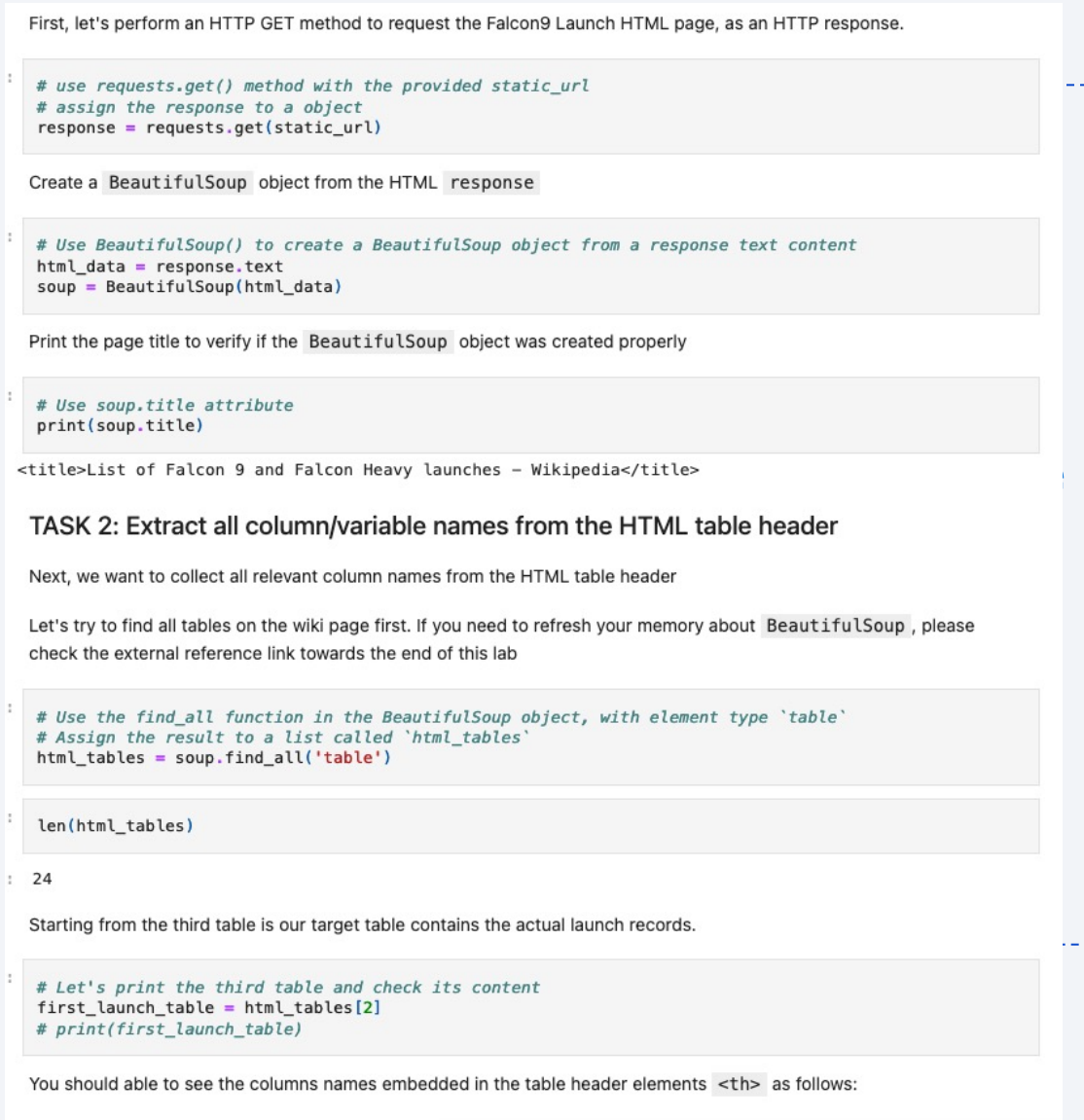
```
# Get the head of the dataframe
data.head()
```

Out[15]:

	static_fire_date_utc	static_fire_date_unix	net	window	rocket
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb

Data Collection - Scraping

- Applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- Parsed the table and converted it into a pandas dataframe.
- The link to the notebook: https://github.com/Khong2612/testrepo/blob/3c31c982183dab7e2eb444b80c2e3488b7b2116e/2-jupyter-labs-webscraping.ipynb



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

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

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

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

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

```
# Use soup.title attribute
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches – Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

```
len(html_tables)
```

```
24
```

Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
# print(first_launch_table)
```

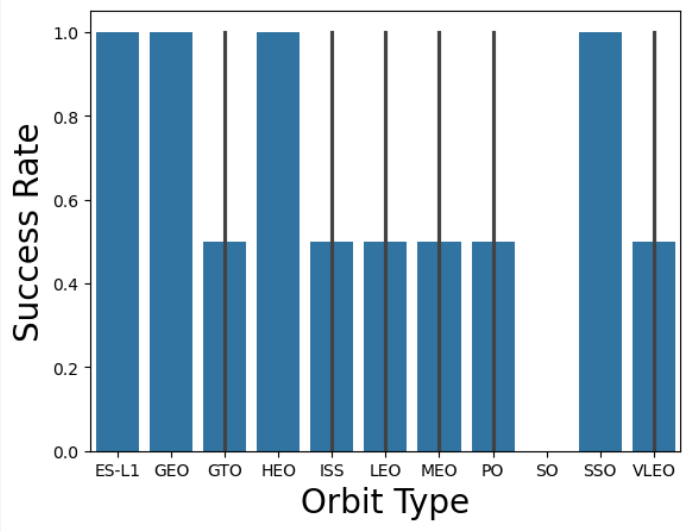
You should able to see the columns names embedded in the table header elements `<th>` as follows:

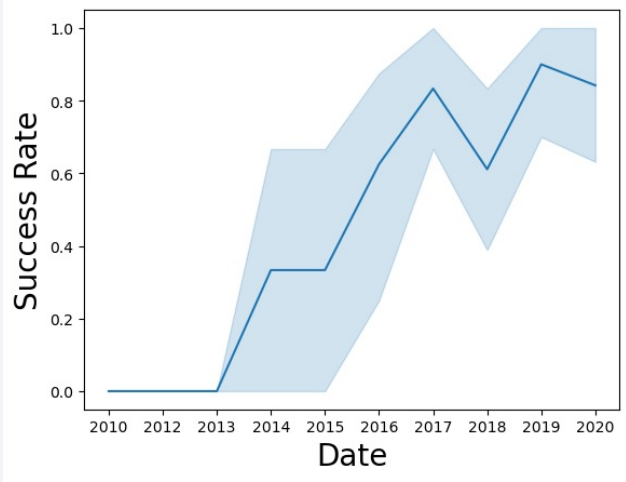
Data Wrangling

- Performed exploratory data analysis and determined the training labels.
- Calculated the number of launches at each site, and the number and occurrence of each orbits
- Created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook: https://github.com/Khong2612/testrepo/blob/3c31c982183dab7e2eb444b80c2e3488b7b2116e/3-jupyter-spacex-data-wrangling.ipynb

EDA with Data Visualization

- Explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.





- The link to the notebook: https://github.com/Khong2612/testrepo/blob/3c31c982183dab7e2eb444b80c2e3488b7b2116e/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- Loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- Applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failed mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook https://github.com/Khong2612/testrepo/blob/3c31c982183dab7e2eb444b80c2e3488b7b2116e/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Marked all launch sites, and added map objects such as markers, circles, and lines to mark the success or failure of launches for each site on the folium map.
- Assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rates.
- Calculated the distances between a launch site to its proximities. We answered some questions for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep a certain distance away from cities.
- The link to the notebook https://github.com/Khong2612/testrepo/blob/3c31c982183dab7e2eb444b80c2e3488b7b2116e/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

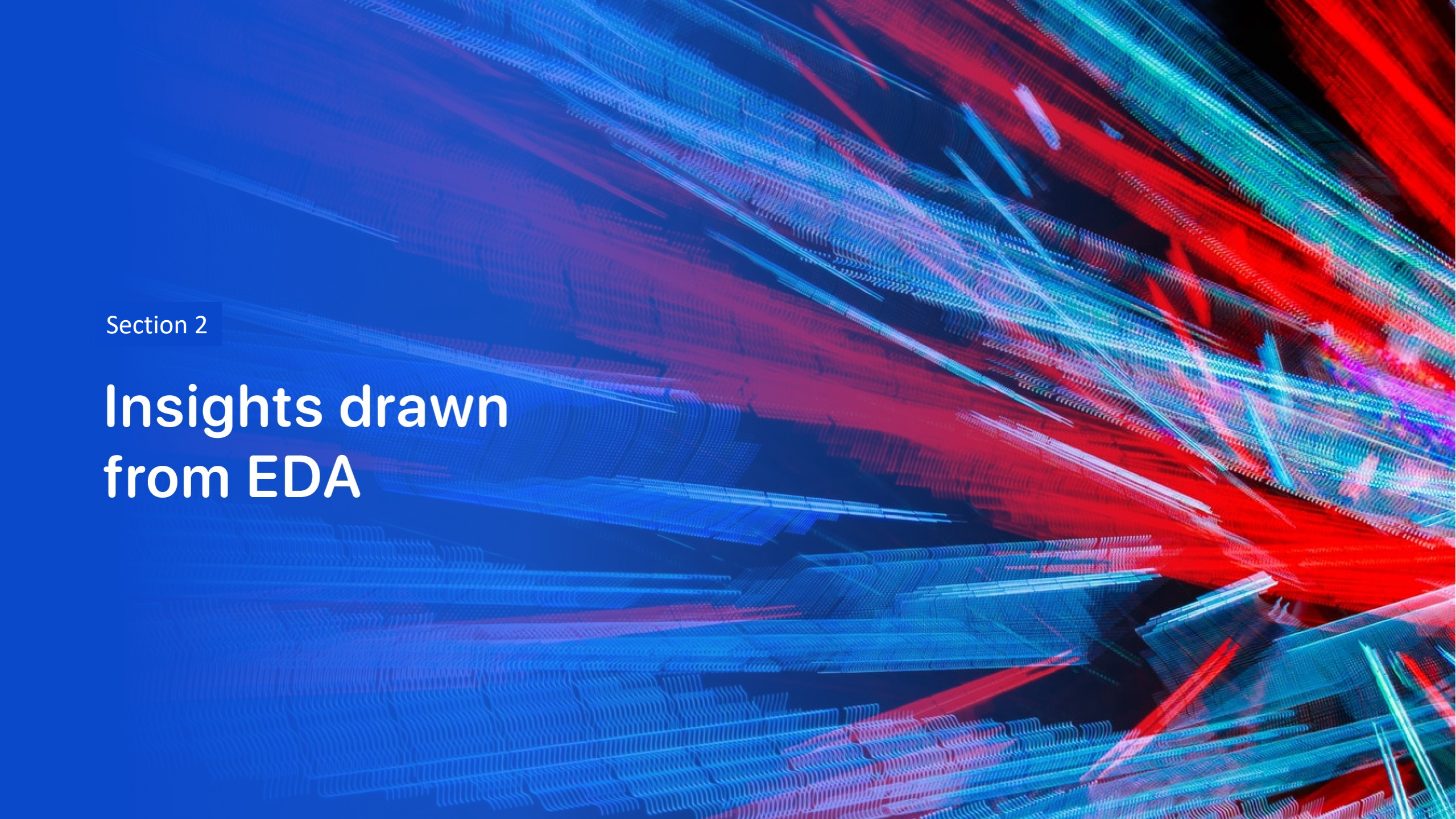
- Built an interactive dashboard with Plotly dash
- Plotted pie charts showing the total launches by a certain sites
- Plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The link to the notebook https://github.com/Khong2612/testrepo/blob/444f7f3775dc75438678adff35971bbe30911d3e/spacex_dash_app.py

Predictive Analysis (Classification)

- Loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- Built different machine-learning models and tune d different hyperparameters using GridSearchCV.
- Used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- Found the best-performing classification model.
- The link to the notebook https://github.com/Khong2612/testrepo/blob/444f7f3775dc75438678adff35971bbe30911d3e/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

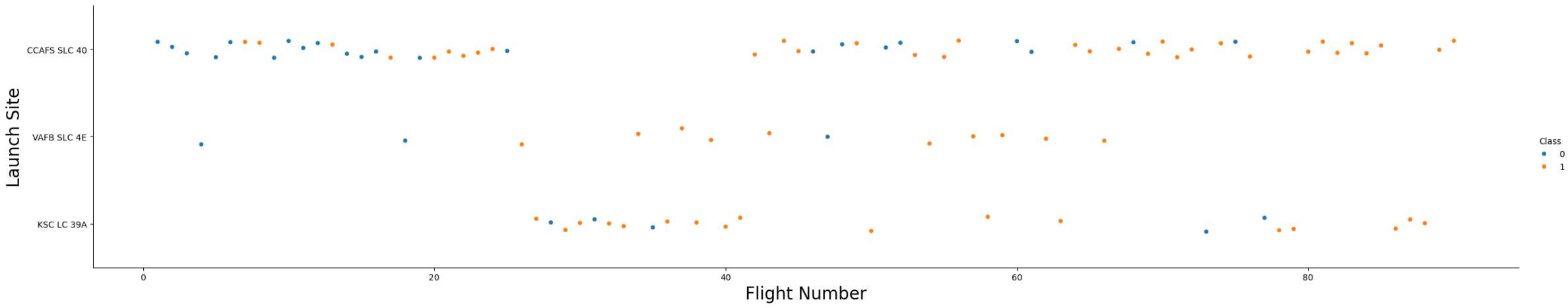
Results

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


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

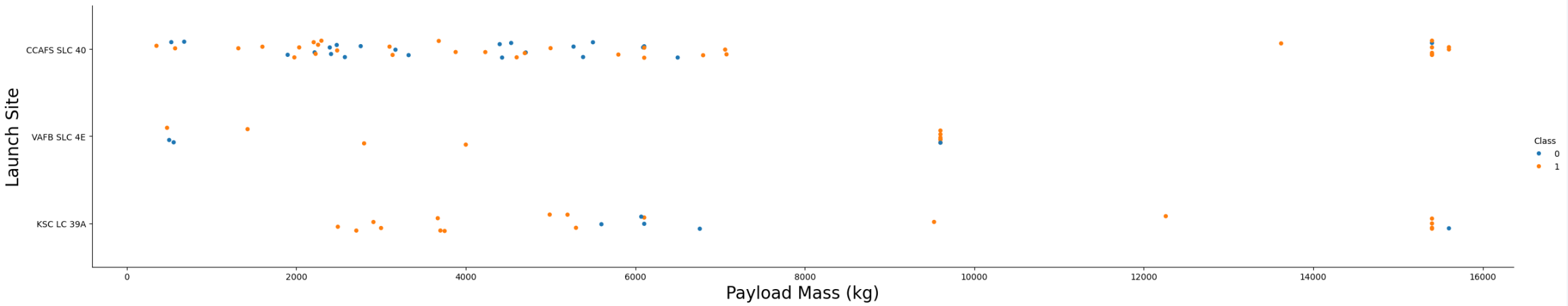
- A scatter plot of Flight Number vs. Launch Site



- The larger the flight amount at a launch site, the greater the success rate at a launch site.

Payload vs. Launch Site

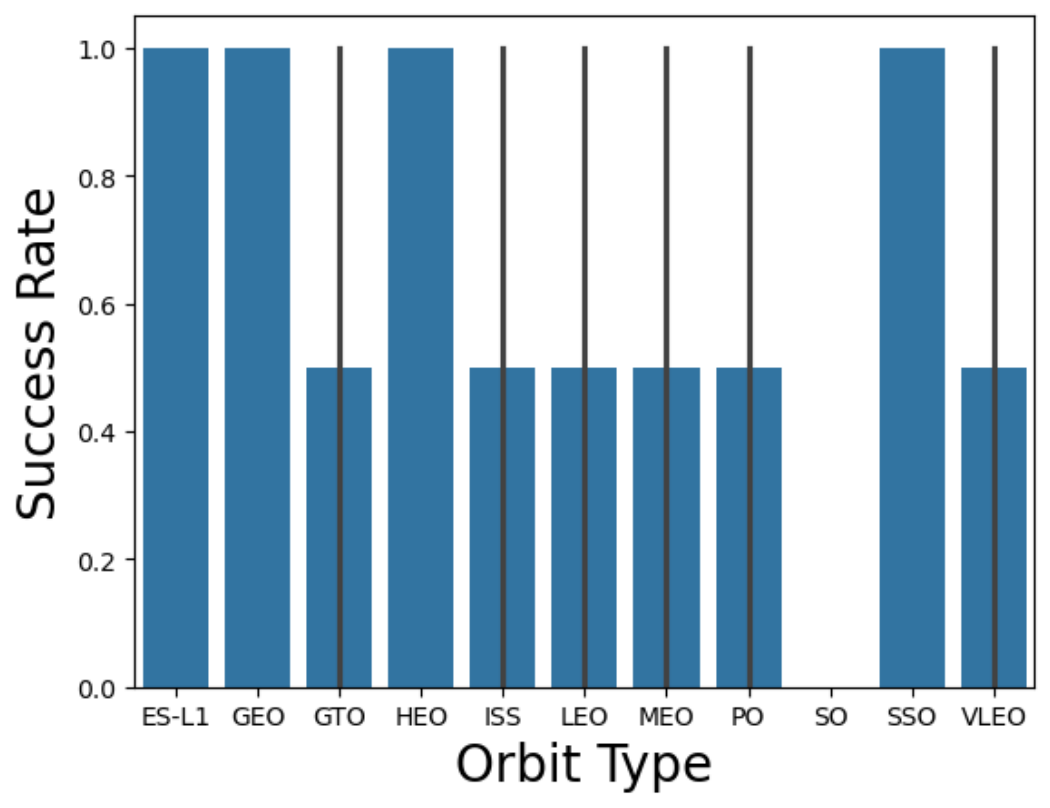
- A scatter plot of Payload vs. Launch Site



- The greater the payload mass for launch site CCAFS SLC 40 the higher the success rate for the rocket

Success Rate vs. Orbit Type

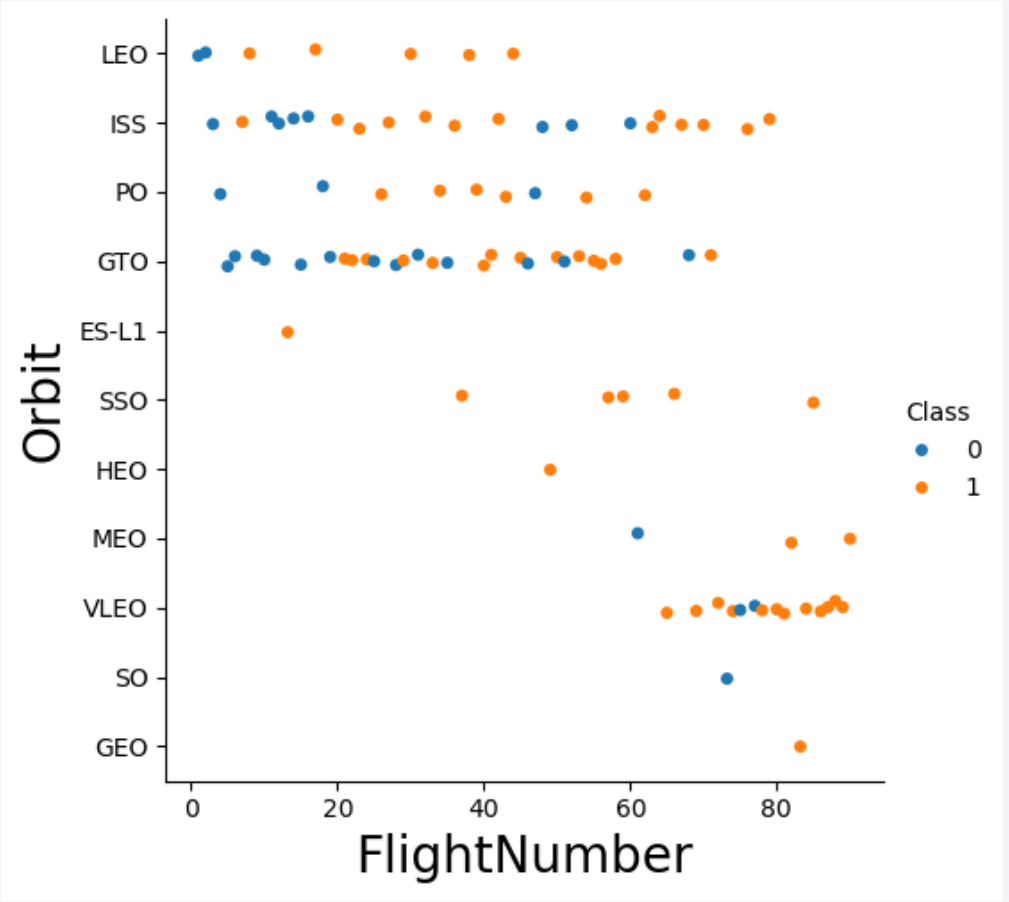
- Abar chart for the success rate of each orbit type



- ES-L1, GEO, HEO, SSO, VLEO had the most success rate.

Flight Number vs. Orbit Type

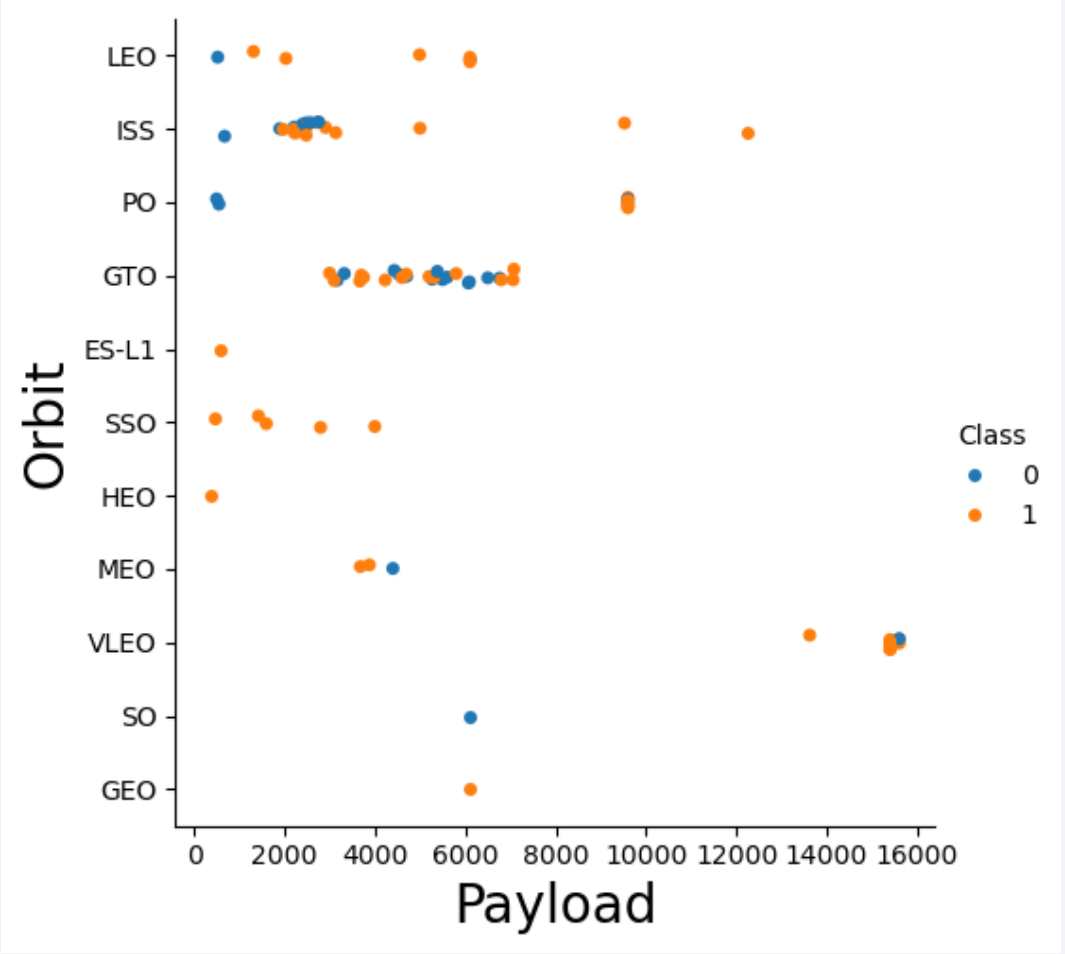
- A scatter point of Flight number vs. Orbit type
- In the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



Payload vs. Orbit Type

- A scatter point of payload vs. orbit type

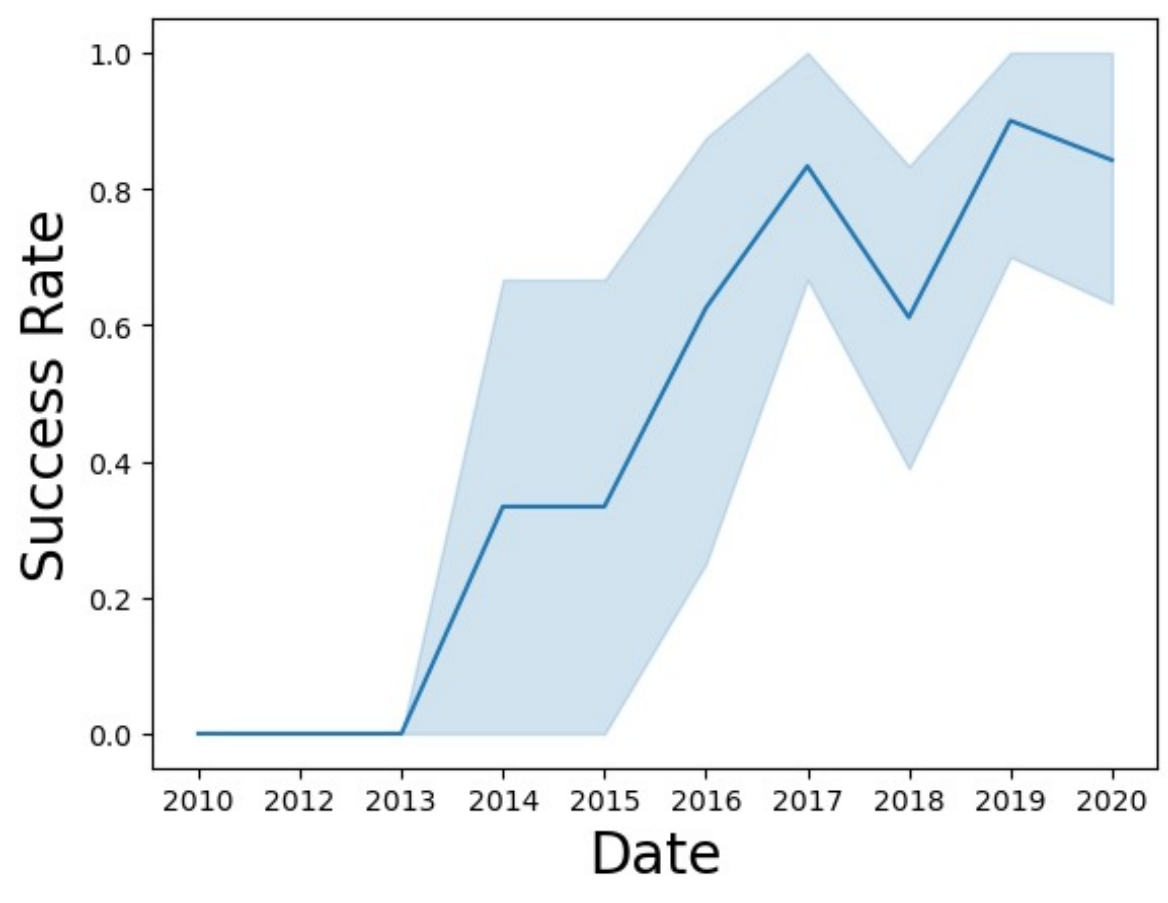
- Observed that with heavy payloads, the successful landing is more for PO, LEO and ISS orbits.



Launch Success Yearly Trend

- A line chart of yearly average success rate

- Observed that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

- The names of the unique launch sites
- The Key word DISTINCT is used to show only unique launch sites from the SpaceXdata.

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

In [10]:

```
%sql select distinct launch_site from SPACEXTBL
```

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

Out[10]:

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

Launch Site Names Begin with 'CCA'

- 5 records where launch sites begin with `CCA`
- Where clause and keyword like are used to subset the observations with the launch sites begin with ‘CCA’.
- The keyword limit is used to control the number of output.

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

In [11]:

```
%sql select * from SPACEXTBL where launch_site like 'CCA%' limit 5
```

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

Out[11]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Cus
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	S
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	(
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	(
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	

Total Payload Mass

- The total payload carried by boosters from NASA is 45596.
- SUM function is used to calculate the total payload carried.

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

In [12]:

```
%sql select sum(payload_mass__kg_) from SPACEXTBL where customer = 'NASA (CRS)'
```

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

Out[12]:

sum(payload_mass__kg_)
45596

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is 2928.4
- AVG function is used to calculate the average payload mass

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

In [13]:

```
%sql select avg(payload_mass__kg_) from SPACEXTBL where booster_version = 'F9 v1.1'
```

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

Out[13]:

avg(payload_mass__kg_)
2928.4

First Successful Ground Landing Date

- The dates of the first successful landing outcome on ground pad is 2015-12-22
- Min function is used to extract the dates of the first successful landing outcome from success (ground pad)

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

Hint:Use min function

In [24]:

```
%sql select min(DATE) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
```

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

Out[24]:

min(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

- Where clause and comparison operators are used.
- Four boosters was extracted

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_ > 4000 and payload_mass__kg_ < 6000
```

Pyth

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

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

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failure mission outcomes is 99
- Count function is used

List the total number of successful and failure mission outcomes

```
[26] sql select count(mission_outcome) from SPACEXTBL where mission_outcome = 'Success' or mission_outcome = 'Failure (in flight)'
                                                                                                                    Python
```

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

count(mission_outcome)
99

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Max function and subquery are used to generate the list

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

+ Code + Markdown

```
%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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions and launch site names for in year 2015
- Substr function are used to extract the month and from date.
- Two sites was extracted

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.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
%sql select substr(Date,6,2) as month , landing_outcome , booster_version, launch_site from spacextbl
where substr(date,0,5)='2015' and landing_outcome = 'Failure (drone ship)'
```

3] Python

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

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

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

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order
- Group by function is used to consolidate the landing_outcome data.
- 8 landing outcomes was ranked. The top one is NO Attempt

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

```
%sql select landing_outcome , count(*) as rank from SPACEXTBL where (DATE between '2010-06-04' and '2017-03-20')
group by landing_outcome order by rank desc
```

Python

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

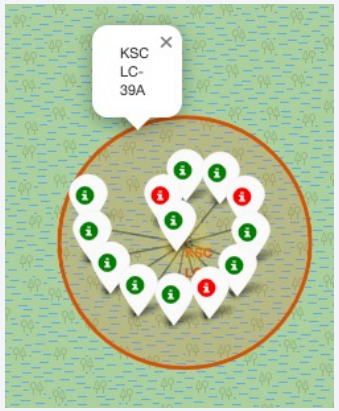
Landing_Outcome	rank
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

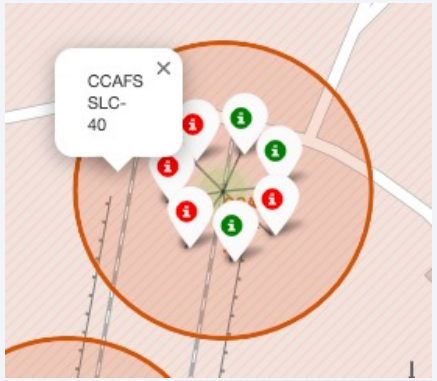

Section 3
Launch Sites
Proximities Analysis

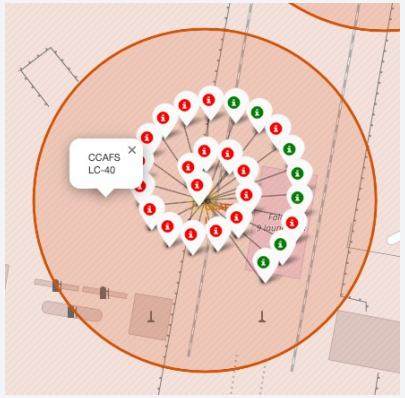
All launch sites global map markers

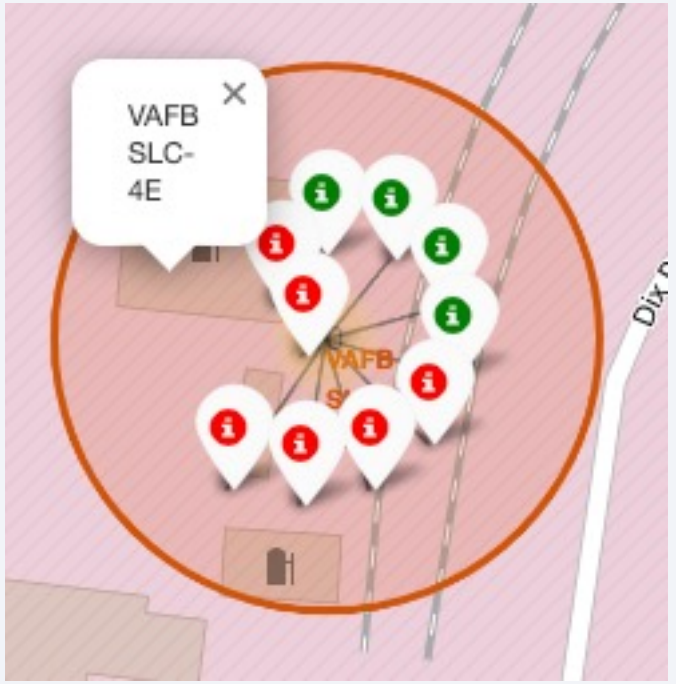
- The SpaceX launch sites are in the US (Florida and California)

Markers showing launch sites with color labels









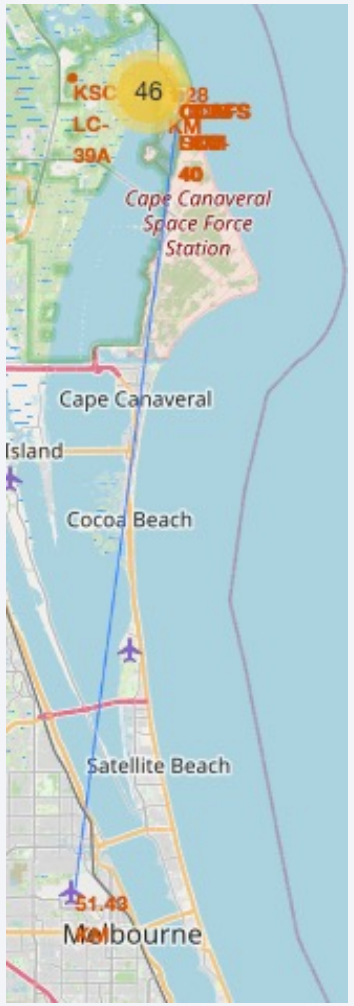
- Florida Launch Sites
- California Launch Sites

Green Marker shows successful launches

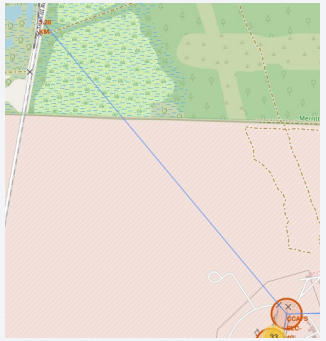
Red Marker shows the failure

Launch Site distance to landmarks

- Distance to City:

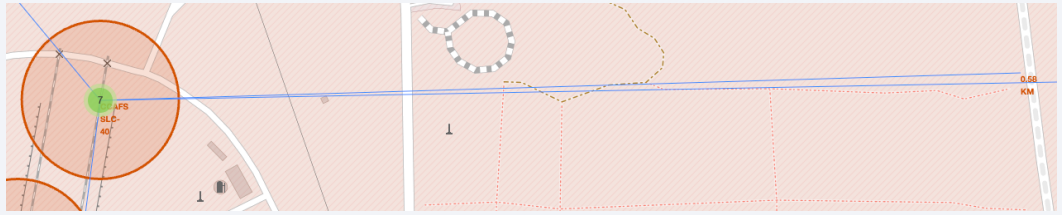


- Distance to railway:

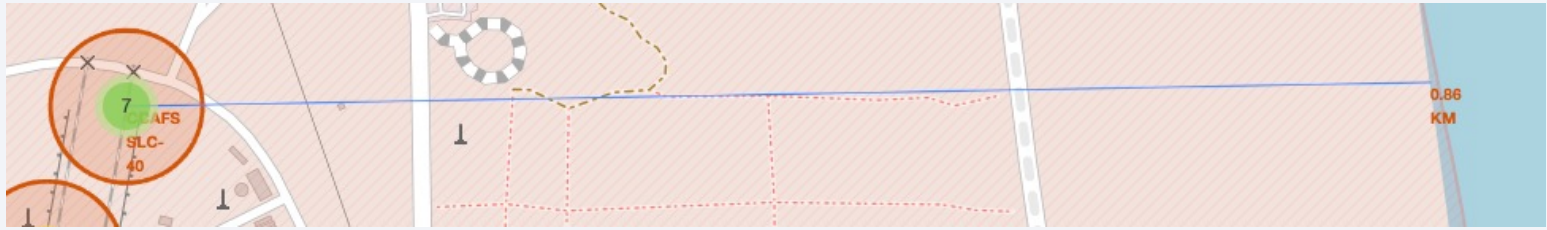


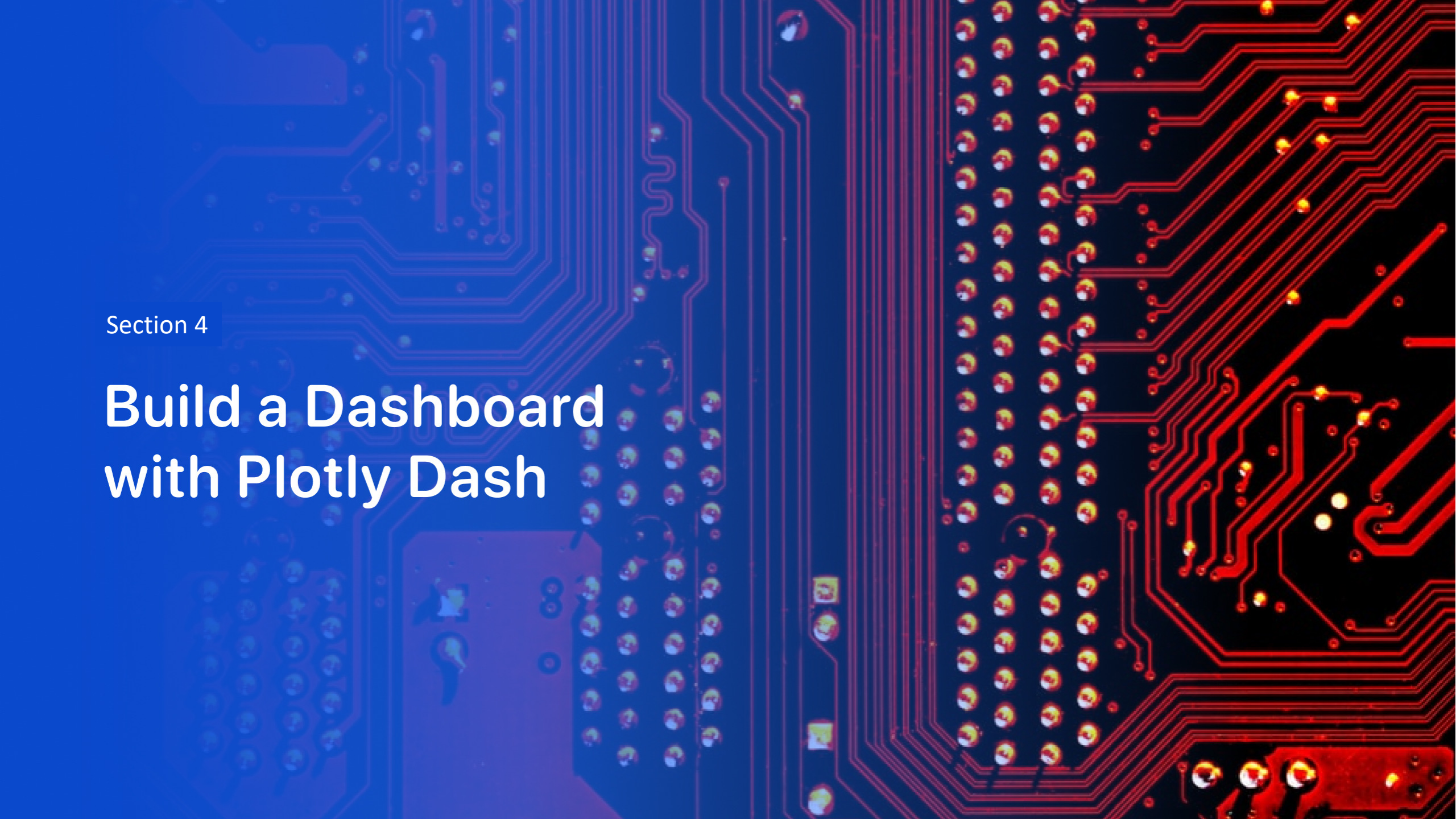
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? Yes
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

- Distance to Highway:



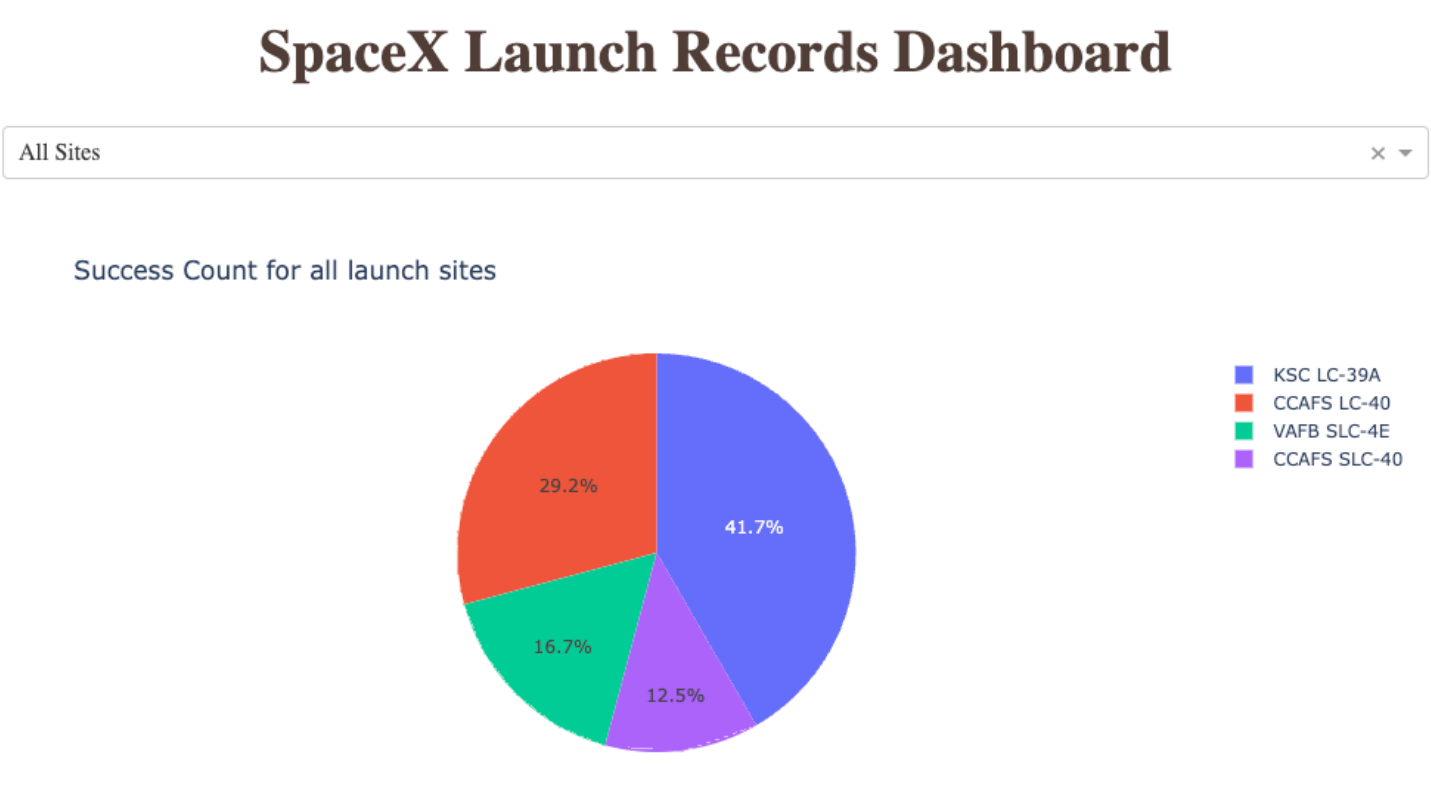
- Distance to coastline




Section 4
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site

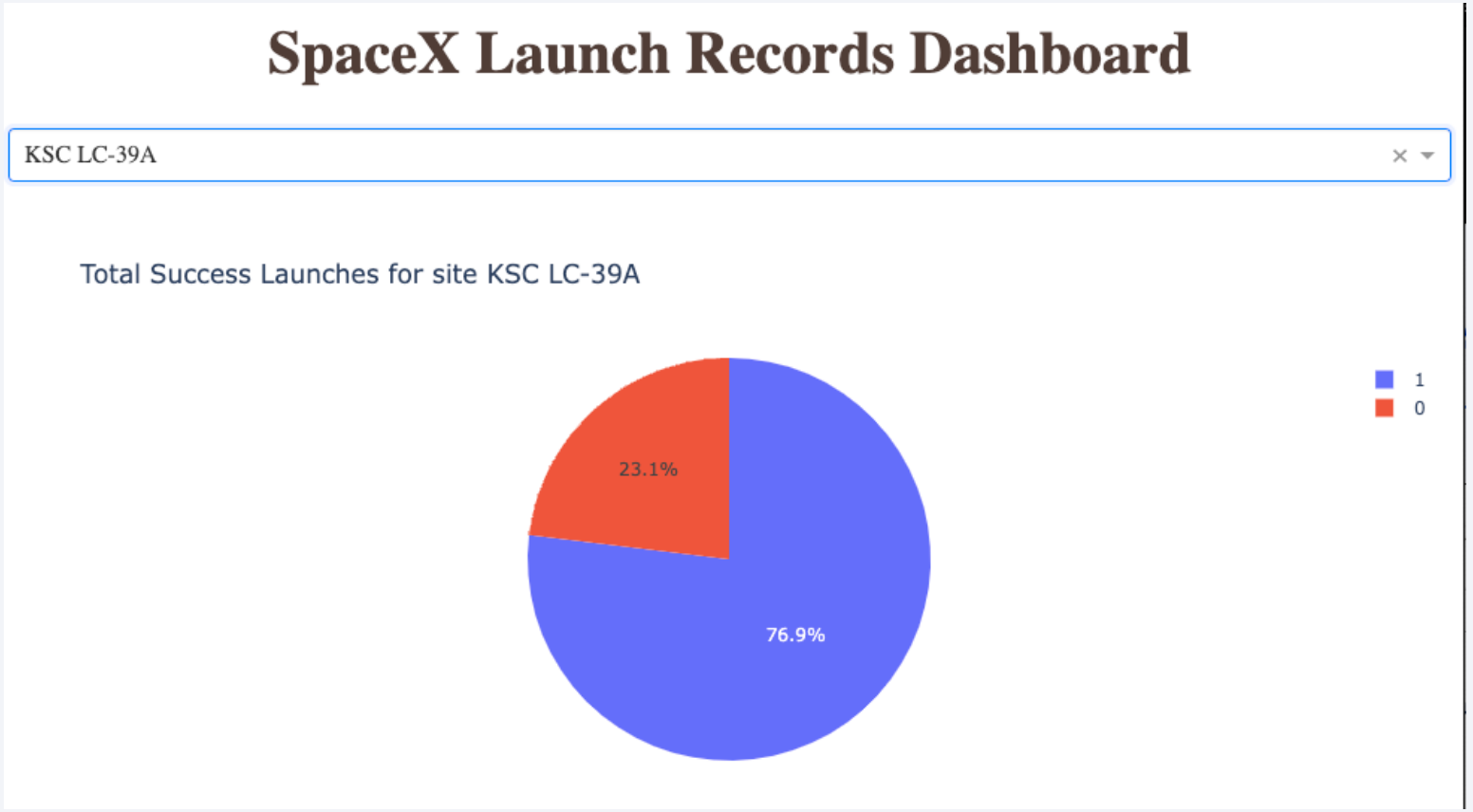
- Launch success count for all sites



- KSC LC-39A had the most successful launches from all the sites

Pie chart showing the Launch site with the highest launch success ratio

- The pie chart for the launch site with the highest launch success ratio



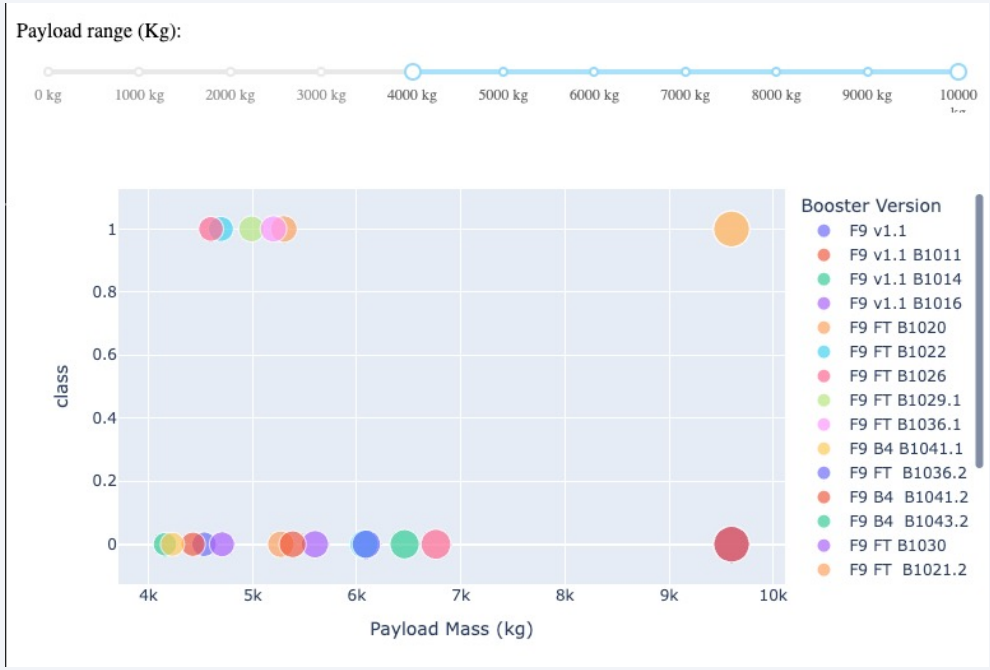
- KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Scatter plot of Payload vs Launch Outcome for all sites,

- Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

Low Weighted Payload 0 – 4000kg:

Heavy Weighted Payload 4000 – 10000kg:



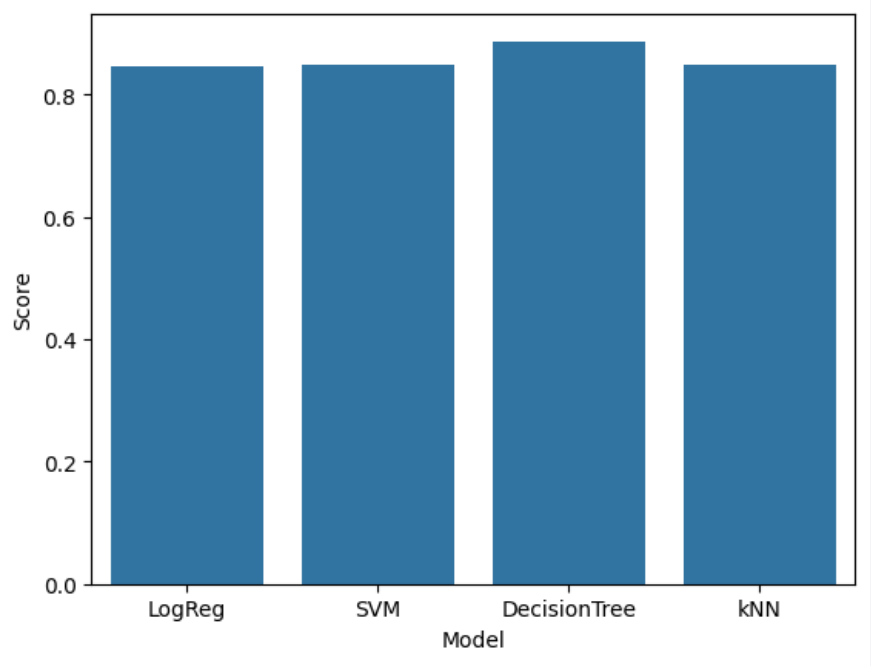
The Success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Decision Tree model has the highest classification accuracy with 0.8875




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

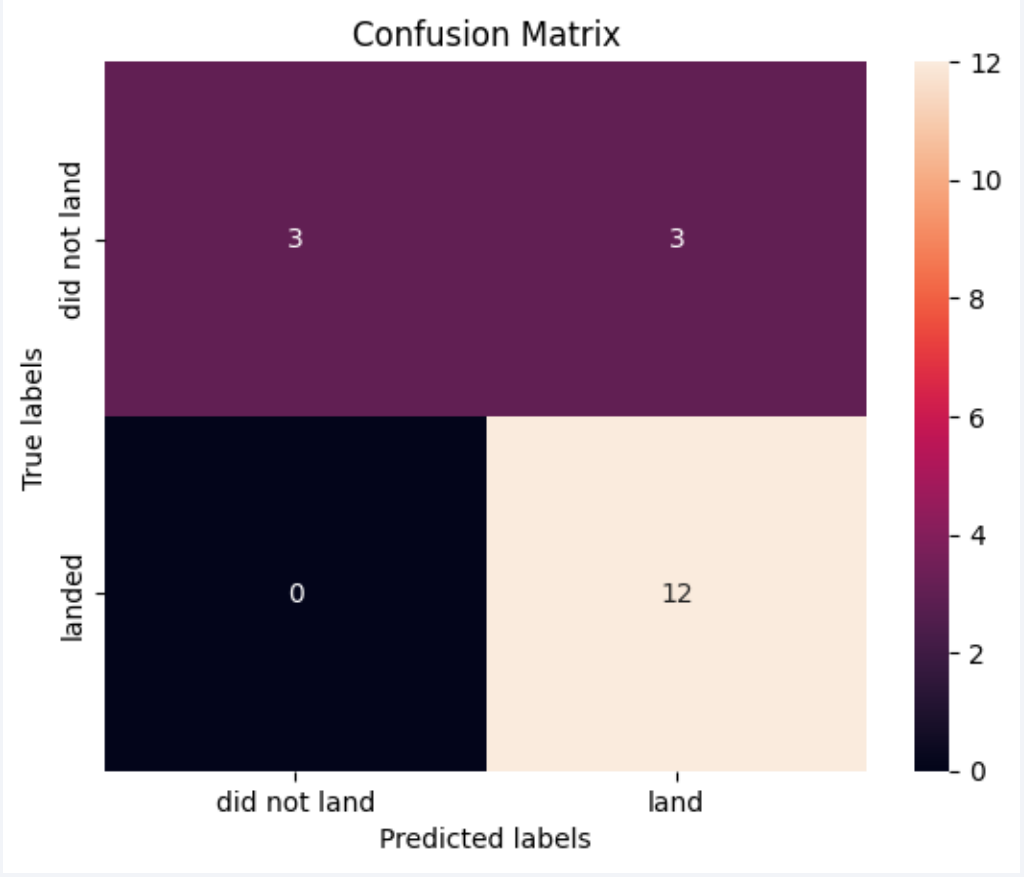
Python

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 14, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 5, 'splitter': 'random'}
accuracy : 0.8875
```

+ Code + Markdown

Confusion Matrix

- The classifier can distinguish between the different classes.
- The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launchsite.
- Launch success rate started to increase in 2013 till 2020.
- The most success rate was in Orbits ES-L1, GEO, HEO, SSO, VLEO.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine-learning model.

Thank you!