IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Qingwen Yi
11.09.2024

Outline

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

Executive Summary

- **Summary of methodologies**
  - Collecting data with API
  - Collecting data with web scraping from relevant Wiki pages
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- **Summary of all results**
  - CSV Dataset prepared and created by data collection

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.

- Problems you want to find answers

  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.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Collecting data with REST API and web scraping from relevant Wiki pages
- 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

- Describe how data sets were collected.

1. **Collecting data with API**
    - Request to the SpaceX API by conducting get request to the SpaceX API to collect data
    - Clean the requested data by first using json_normalize method to convert the json result into a datafra and filling missing data with means
    - Export data in CSV
2. **Collecting data with web scraping from relevant Wiki pages**
    - Data collection was performed by using BeautifulSoup function in Python
    - Parsing the table and converting it into a pandas dataframe
    - Export data in CSV

Data Collection – SpaceX API

- Data collection with SpaceX REST calls was performed by conducting get request to the SpaceX API to collect data, using json_normalize method to convert the json result into a datafra and clean the requested data by filling missing data with means
- GitHub URL of the completed SpaceX API calls notebook: https://github.com/yiqingwen01/IBM-Data-Science-Professional-Certificate/blob/main/Data%20Collection%20API.ipynb

Get request to SpaceX API

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]: response = requests.get(spacex_url)
```

Convert json result into dataframe

```
In [17]: # Use json_normalize meethod to convert the json result into a dataframe
         static_json_df = res.json()
         data = pd.json_normalize(static_json_df)
```

Clean data by filling missing data with means

```
In [33]: # Calculate the mean value of PayloadMass column
         PayloadMass = pd.DataFrame(data_falcon9['PayloadMass'].values.tolist()).mean(1)
         print(PayloadMass)

         # Replace the np.nan values with its mean value
         rows = data_falcon9['PayloadMass'].values.tolist()[0]

         df_rows = pd.DataFrame(rows)
         df_rows = df_rows.replace(np.nan, PayloadMass)

         data_falcon9['PayloadMass'][0] = df_rows.values
         data_falcon9
```

Data Collection - Scraping

- Data collection with web scraping was performed by using BeautifulSoup function in Python and parsing the table and converting it into a pandas dataframe
- GitHub URL of the completed web scraping notebook: https://github.com/yiqingwen01/IBM-Data-Science-Professional-Certificate/blob/main/Data%20Collection%20with%20Web%20Scraping.ipynb

Get request to Falcon 9 Wiki page

```
In [5]:
# use requests.get() method with the provided static_url
# assign the response to a object

html_data = requests.get(static_url)
html_data.status_code
```

```
Out[5]: 200
```

Use BeautifulSoup function to parse table

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

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

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

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

```
In [10]:
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 get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called colu

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

Data Wrangling

- How data were processed:
  - Exploratory Data Analysis
  - Determine Training Labels
- Data wrangling process:
  1) Import relevant libraries
  2) Calculate the number of launches on each site
  3) Calculate the number and occurrence of each orbit
  4) Calculate the number and occurence of mission outcome of the orbits
  5) Create a landing outcome label from Outcome column
  6) Export data in CSV
- GitHub URL of completed data wrangling related notebooks: https://github.com/yiqingwen01/IBM-Data-Science-Professional-Certificate/blob/main/Data%20Wrangling.ipynb

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
  - Visualize the relationship between Flight Number and Launch Site using plot chart to have an overview of the relationship
  - Visualize the relationship between Payload Mass and Launch Site using plot chart to have an overview of the relationship
  - Visualize the relationship between success rate of each orbit type using bar chart to understand the success rate of each class
  - Visualize the relationship between FlightNumber and Orbit type using plot chart to have an overview of the relationship
  - Visualize the relationship between Payload Mass and Orbit type using plot chart to have an overview of the relationship
  - Visualize the launch success yearly trend using line chart to better understand how the trend develops
- GitHub URL of the completed EDA with data visualization notebook: https://github.com/yiqingwen01/IBM-Data-Science-Professional-Certificate/blob/main/EDA%20with%20Data%20Visualization.ipynb

EDA with SQL

- Summary of the SQL queries performed:
  1) Display the names of the unique launch sites in the space mission
  2) Display 5 records where launch sites begin with the string 'CCA'
  3) Display the total payload mass carried by boosters launched by NASA (CRS)
  4) Display average payload mass carried by booster version F9 v1.1
  5) List the date when the first succesful landing outcome in ground pad was acheived
  6) List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  7) List the total number of successful and failure mission outcomes
  8) List the names of the booster_versions which have carried the maximum payload mass
  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
  10) 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

- GitHub URL of completed EDA with SQL notebook: https://github.com/yiqingwen01/IBM-Data-Science-Professional-Certificate/blob/main/EDA%20with%20SQL

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
  - All launch sites are marked, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map
- Explain why you added those objects
  - Color-labeled marker clusters are used to identify which launch sites have relatively high success rate
  - The distances between a launch site to its proximities was calculated so that below questions can be answered
    1) Are launch sites in close proximity to railways?
    2) Are launch sites in close proximity to highways?
    3) Are launch sites in close proximity to coastline?
    4) Do launch sites keep certain distance away from cities?
- GitHub URL of completed interactive map with Folium map: https://github.com/yiqingwen01/IBM-Data-Science-Professional-Certificate/blob/main/Interactive%20Map%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

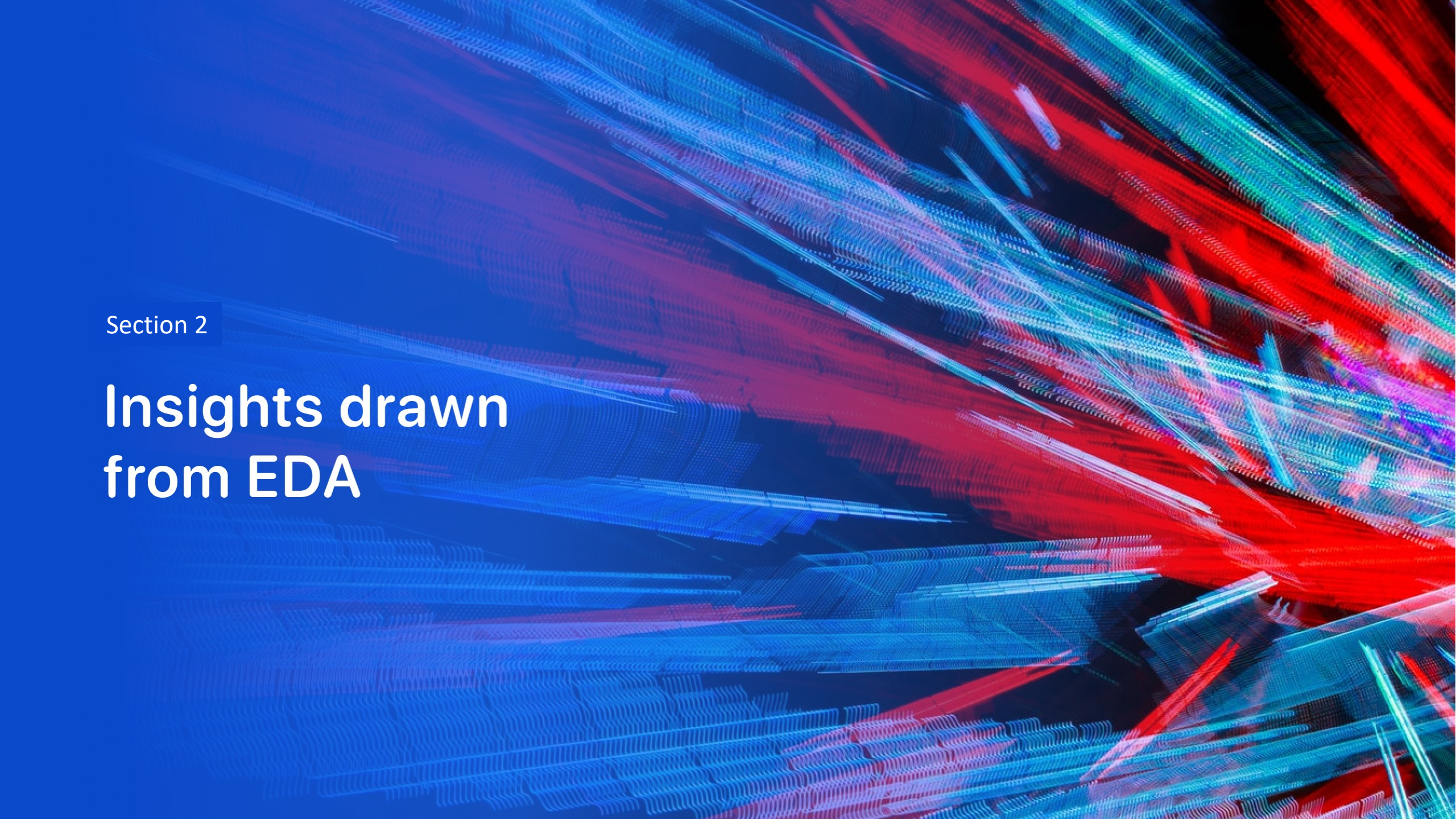
- Plots/graphs and interactions added to the dashboard and reasons:
  - A pie charts was added to show the total launches by a certain sites
  - A scatter graph was added to show the relationship with Outcome and Payload Mass (Kg) for the different booster version
- The GitHub URL of the completed Plotly Dash lab: https://github.com/yiqingwen01/IBM-Data-Science-Professional-Certificate/blob/main/Interactive%20Dashboard%20with%20Ploty%20Dash.ipynb

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model:
  - Data was loaded using numpy and pandas, transformed and split the data into training and testing sets
  - Different machine learning models were built and tuned different hyperparameters using GridSearchCV
  - Accuracy was selected as the metric for the model, the model was improved by using feature engineering and algorithm tuning
  - The best performing classification model was found
- GitHub URL of the completed predictive analysis lab: https://github.com/yiqingwen01/IBM-Data-Science-Professional-Certificate/blob/main/Machine%20Learning%20Prediction.ipynb

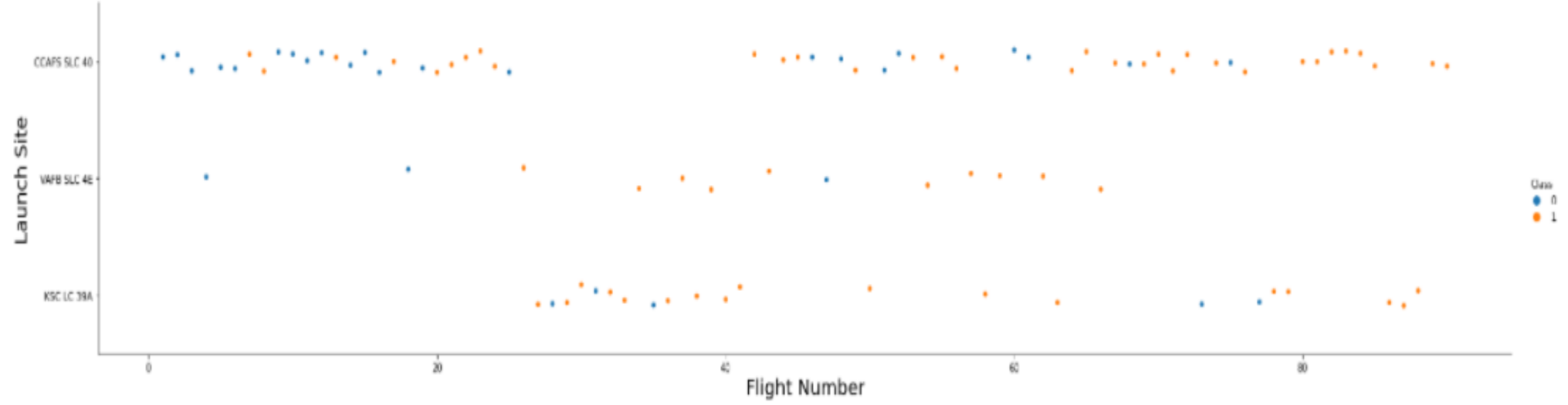
Results

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


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

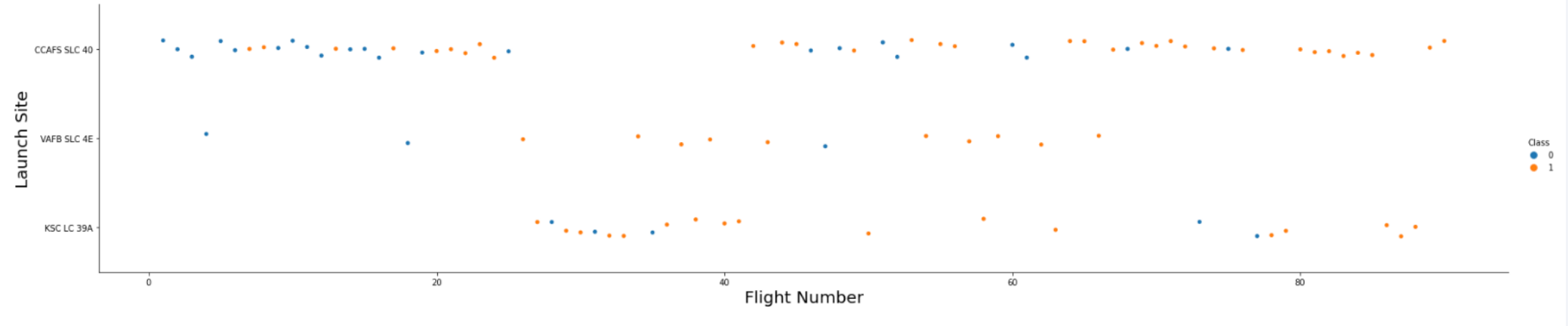
- Scatter plot of Flight Number vs. Launch Site



Remark: the larger the flight amount at a launch site, the greater the success rate at a launch site

Payload vs. Launch Site

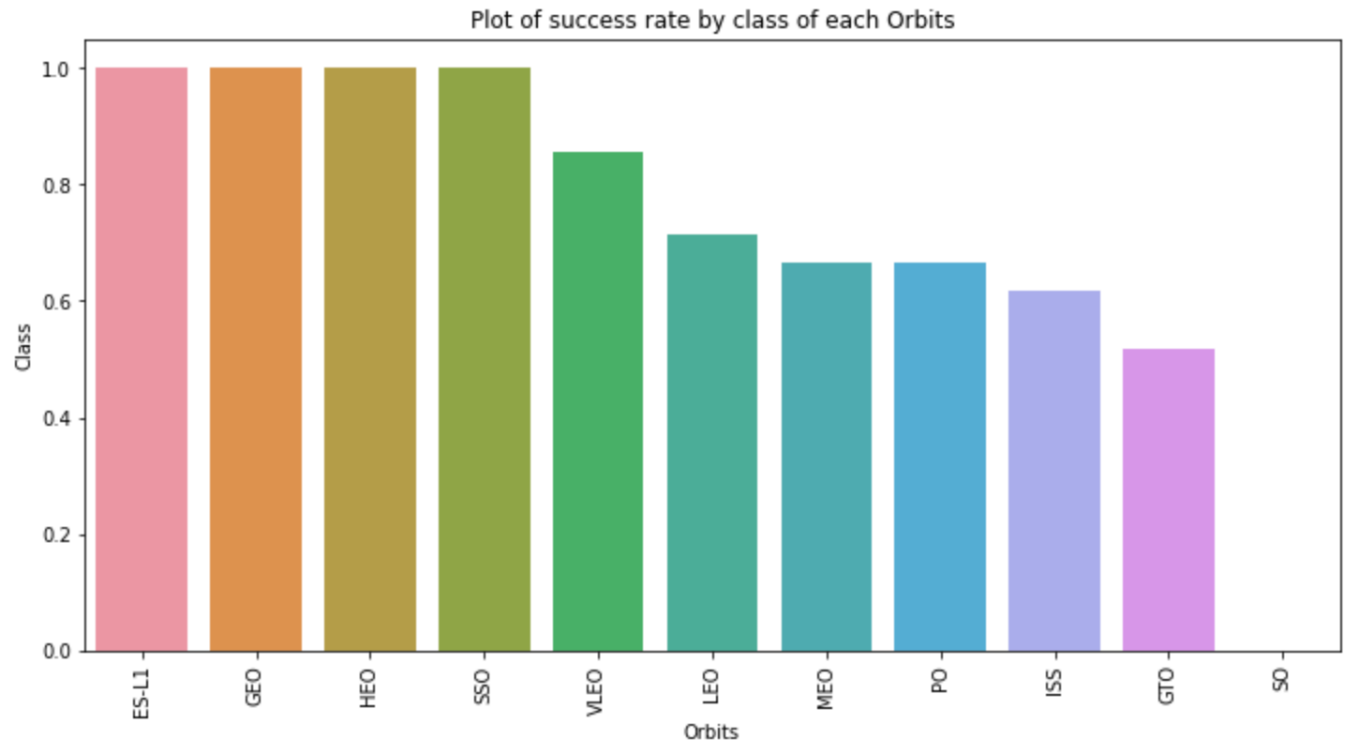
- Show a scatter plot of Payload vs. Launch Site



Remark: the greater the payload mass for launch site CCAFS SLC 40 is, the higher the success rate for the rocket

Success Rate vs. Orbit Type

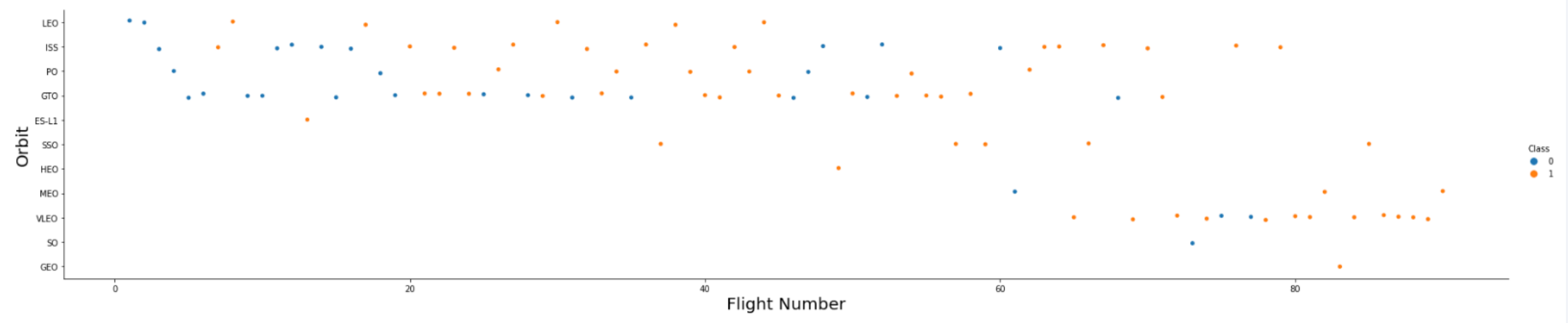
- Show a bar chart for the success rate of each orbit type



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

Flight Number vs. Orbit Type

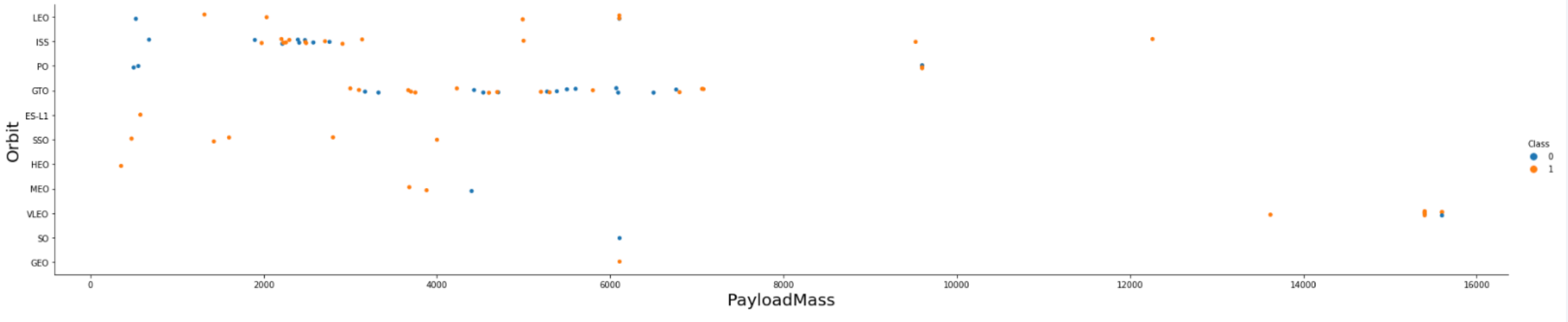
- Show a scatter point of Flight number vs. Orbit type



Remark: the relationship between flight number and orbit type is observed: 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

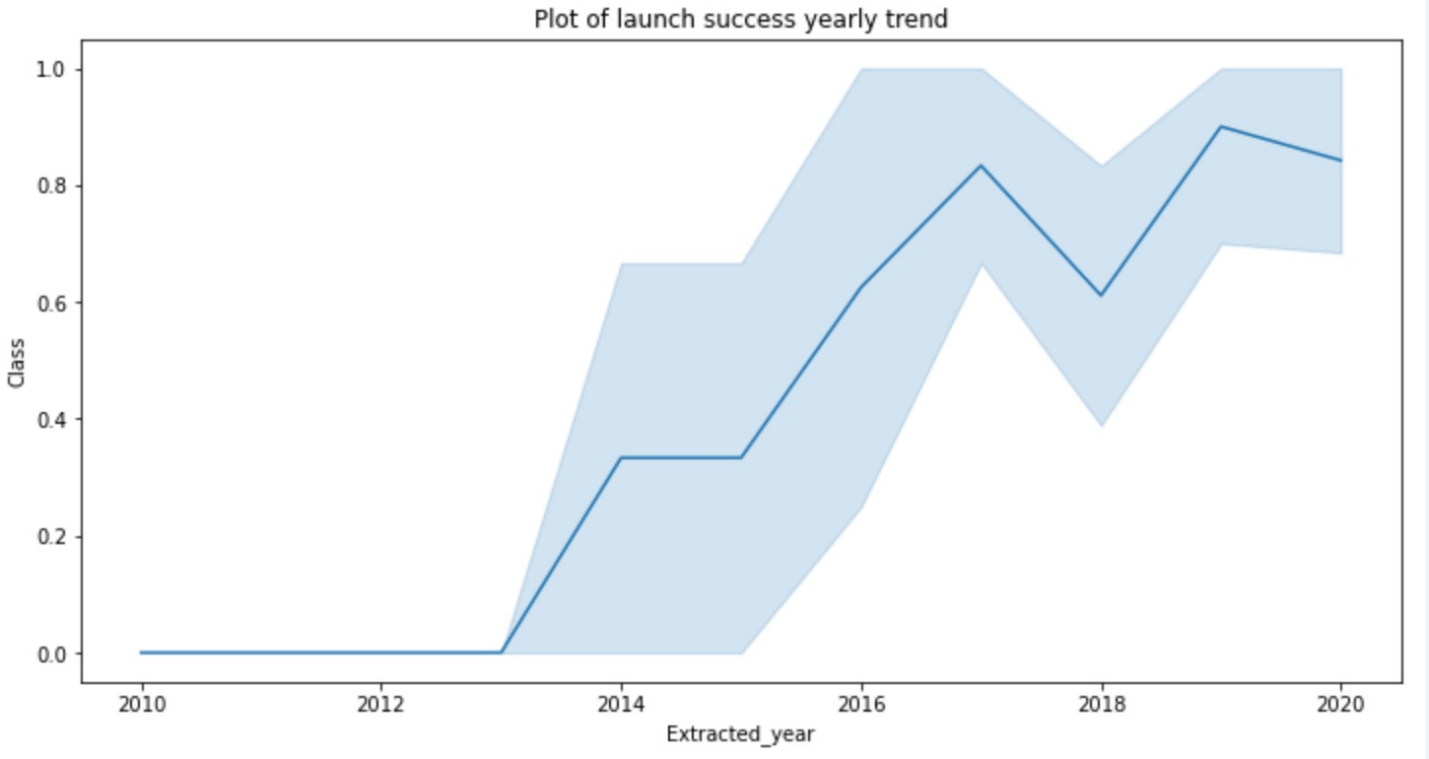
- Show a scatter point of payload vs. orbit type



Remark: the successful landings are more for PO, LEO and ISS orbits when payloads are heavier

Launch Success Yearly Trend

- Show a line chart of yearly average success rate



Remark: launch success rate increased from 2013 to 2020

All Launch Site Names

- Find the names of the unique launch sites

In [10]:

```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Remark: keyword DISTINCT is used here to select the unique names of launch sites

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	lan
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	

Remark: conditions where LaunchSite LIKE ‘CCA%’ and LIMIT 5 are used to display 5 records where launch sites begin with CCA

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

Remark: sum( ) function was used to calculate the total payload carried by boosters from NASA

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

Remark: sum( ) function was used to calculate the total payload carried by boosters from NASA

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

In [14]:

```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Remark: min( ) function was used to find out the date of the first successful landing outcome on ground pad

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

In [15]:

```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Remark: conditions PayloadMassKG > 4000 AND PayloadMassKG < 6000 were applied here in order to limit the the payload mass range

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

In [16]:
```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

```
The total number of successful mission outcome is:
```

	successoutcome
0	100

```
The total number of failed mission outcome is:
```

Out[16]:

	failureoutcome
0	1

Remark: count( ) function was used to count out the total number of successful and failure mission outcomes with where condition applied accordingly

Boosters Carried Maximum Payload

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

In [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

Remark: max( ) function was used in combination with where condition in order to find out the names of booster which have carried the maximum payload mass

2015 Launch Records

- List the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [18]:

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Remark: where conditions were applied here in order to find out the failed landing outcomes and date range in year 2015

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

In [19]:

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

Remark: count( ) function was used in combination with where conditions in order to list out the count of landing outcomes in descending order


Section 3
Launch Sites
Proximities Analysis

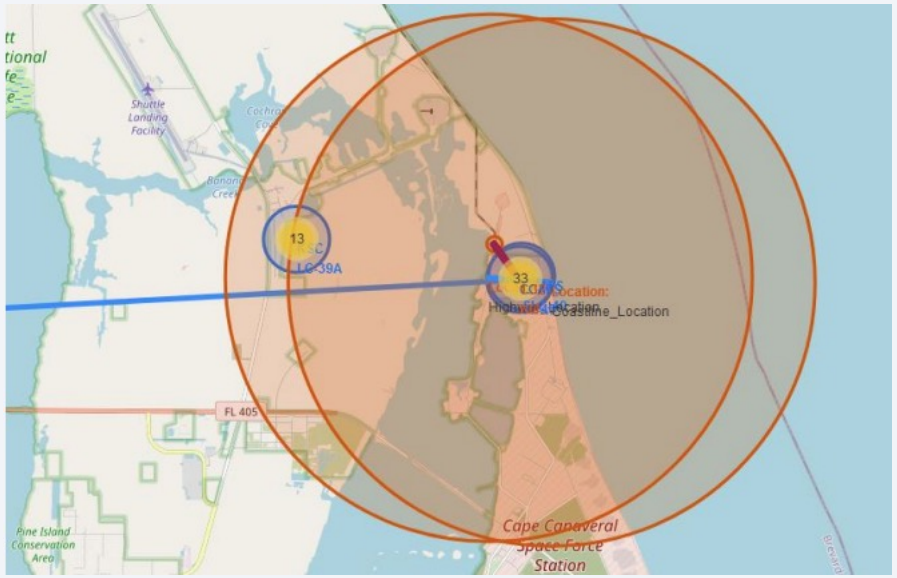
Folium Map: Launch Sites

Remark: SpaceX launch sites are in United States in Florida and California

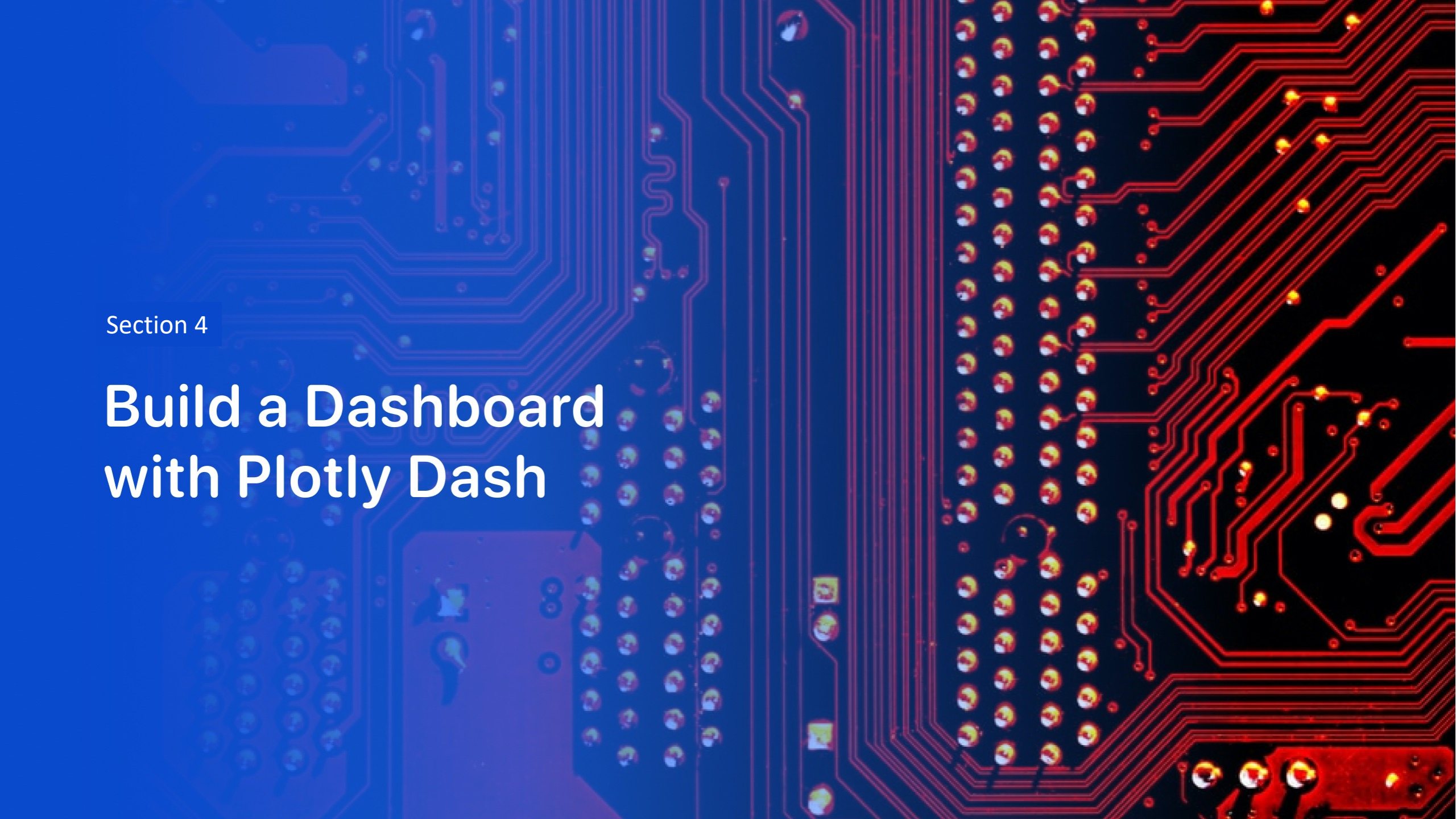
Folium Map: success rate for different launch locations

Remark: green markers showed the successful launches, red markers showed the failed launches

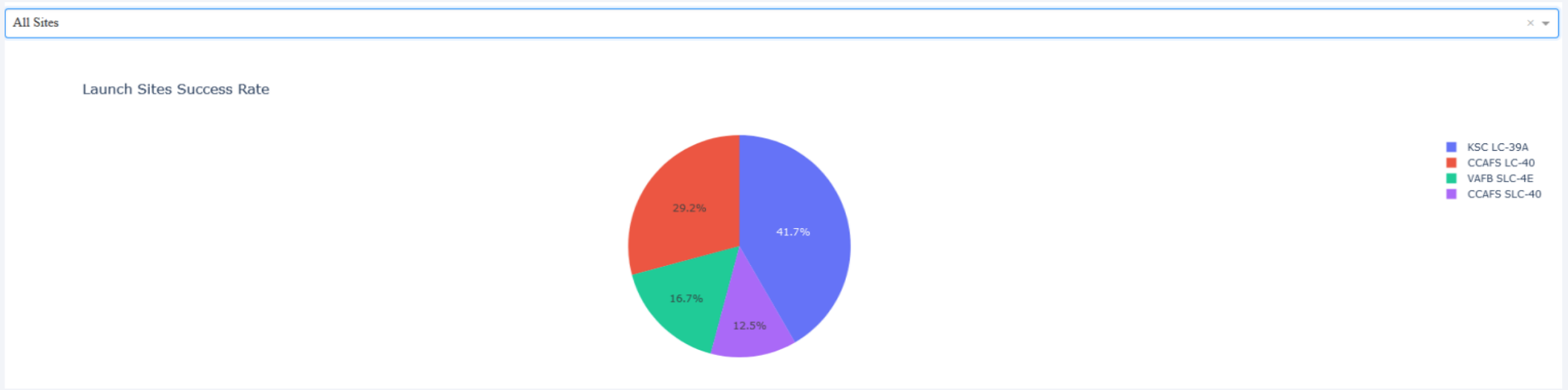
Folium Map: launch sites to its proximities



Remark: distance between launch sites and its proximities was calculated

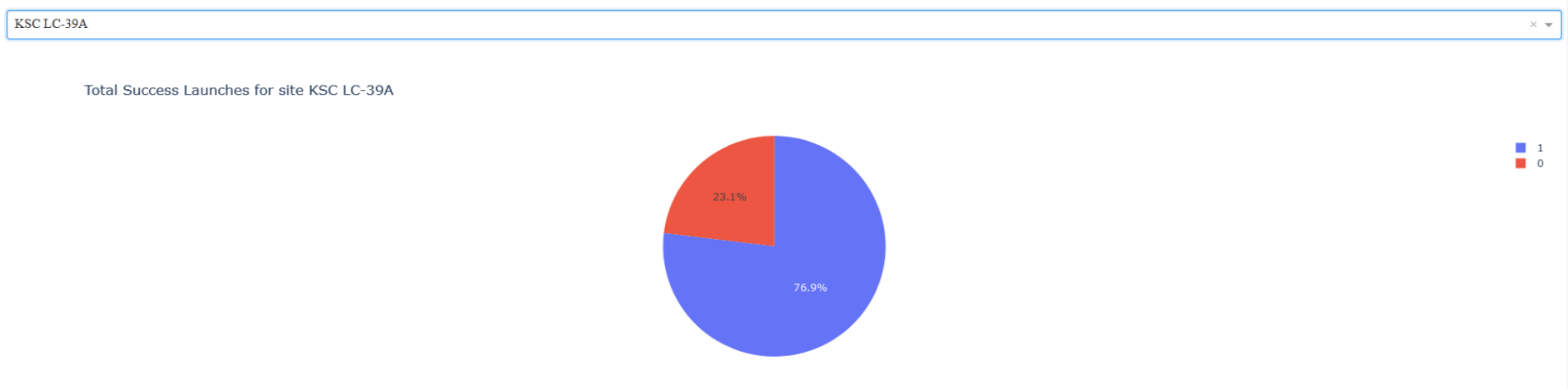

Section 4
Build a Dashboard with Plotly Dash

Dashboard: launch success count for all sites



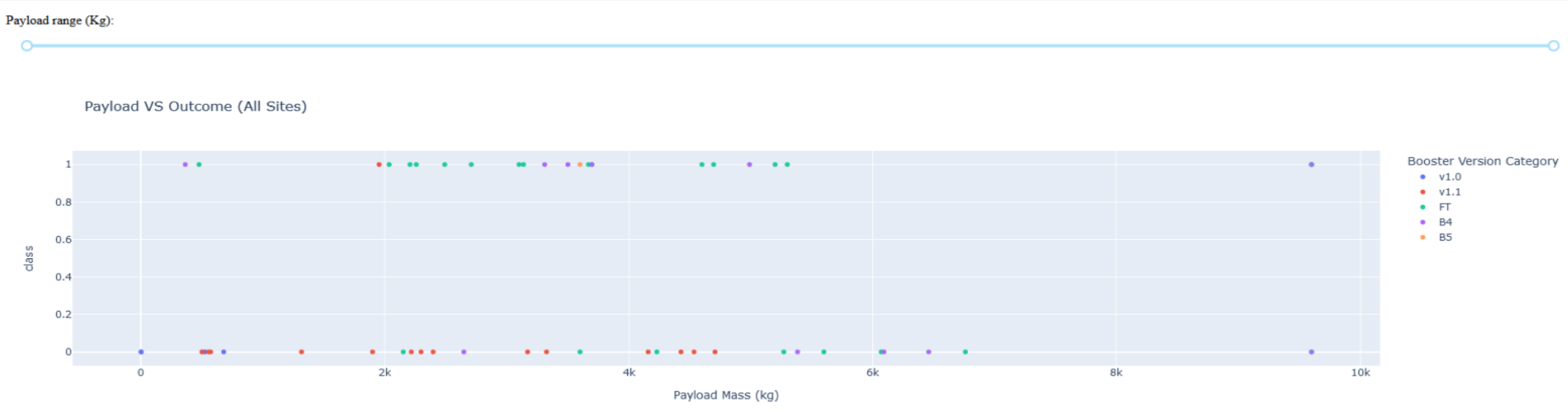
Remark: KSC LC-39A achieved the highest launch success rate of 41.7% among all sites

Dashboard: success launch ratio of KSC LC-39A



Remark: 76.9% of the launches at KSC LC-39A were successful, 23.1% of the launches failed

Dashboard: Payload vs. Launch Outcome



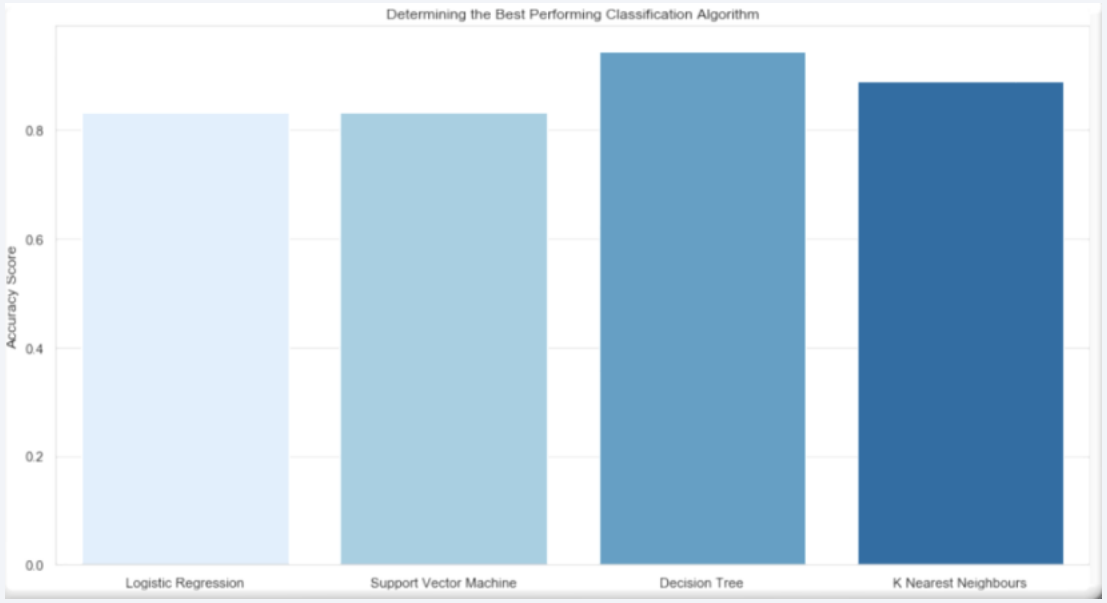
Remark: success rates of low weighted payloads were higher than those of heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

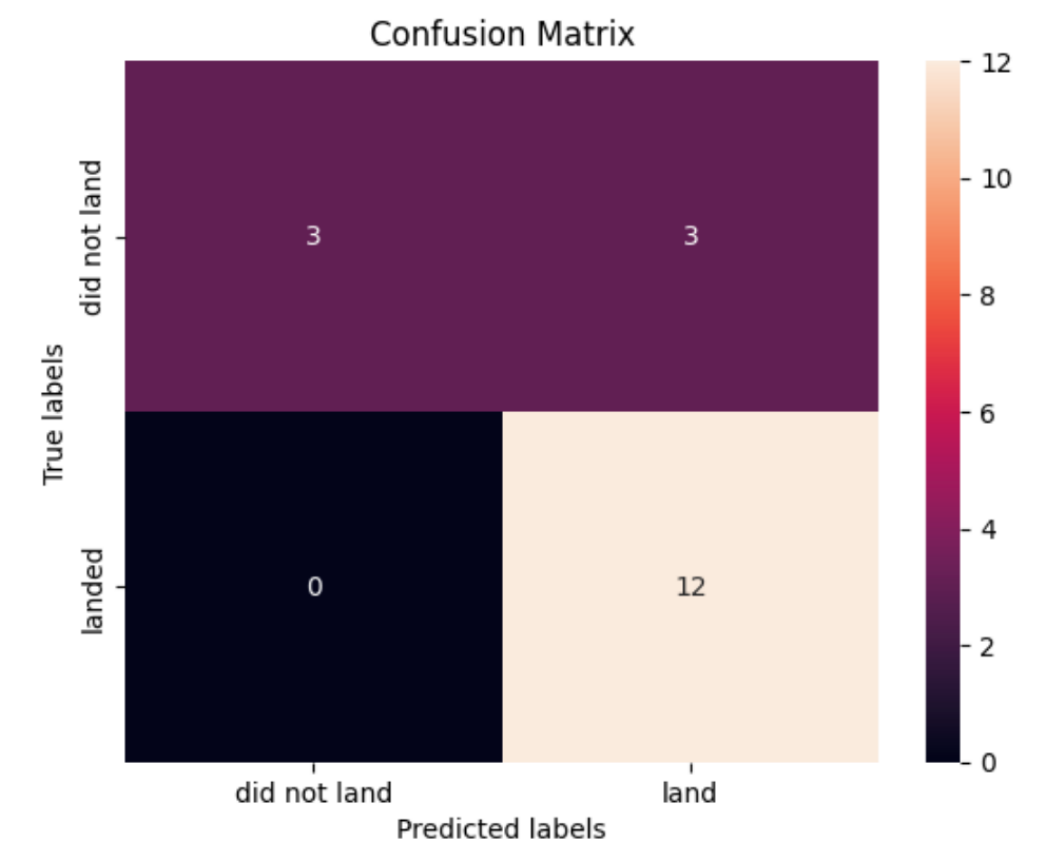
- Visualize the built model accuracy for all built classification models, in a bar chart



Remark: decision tree model has the highest model accuracy among all models compared

Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



Remark:

- decision tree model has the highest model accuracy among all models compared
- this can be explained by the confusion matrix, which showed 3 out of 18 results classified incorrectly (false positive in the top-right corner) , the other 15 results were correctly classified (3 did not land, 12 landed)

Conclusions

- As the number of flights increased, the success rate of a launch site increased
- Orbit types ES-L1, GEO, HEO, and SSO, generated the highest (100%) success rate
- The launch site KSC LC-39 A generated the highest launch success rate of 76.9%, contributed 41.7% of the total successful launches
- The success rate of massive payloads (over 4000kg) was lower than which of low payloads
- The best performed classification model was the Decision Tree model with model accuracy of 94.44%

Thank you!