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 was collected through API/Web Scraping
  - ❑ Data transformation
  - ❑ Data Analysis with SQL
  - ❑ Data visualization/Exploration
  - ❑ Analytics with Folium
  - ❑ Implementation of machine learning algorithms
- Summary of all results
  - ❑ Data analysis result in screenshots
  - ❑ Predictive Analytics result

Introduction

- Project background and context

  Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars, vs 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

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

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

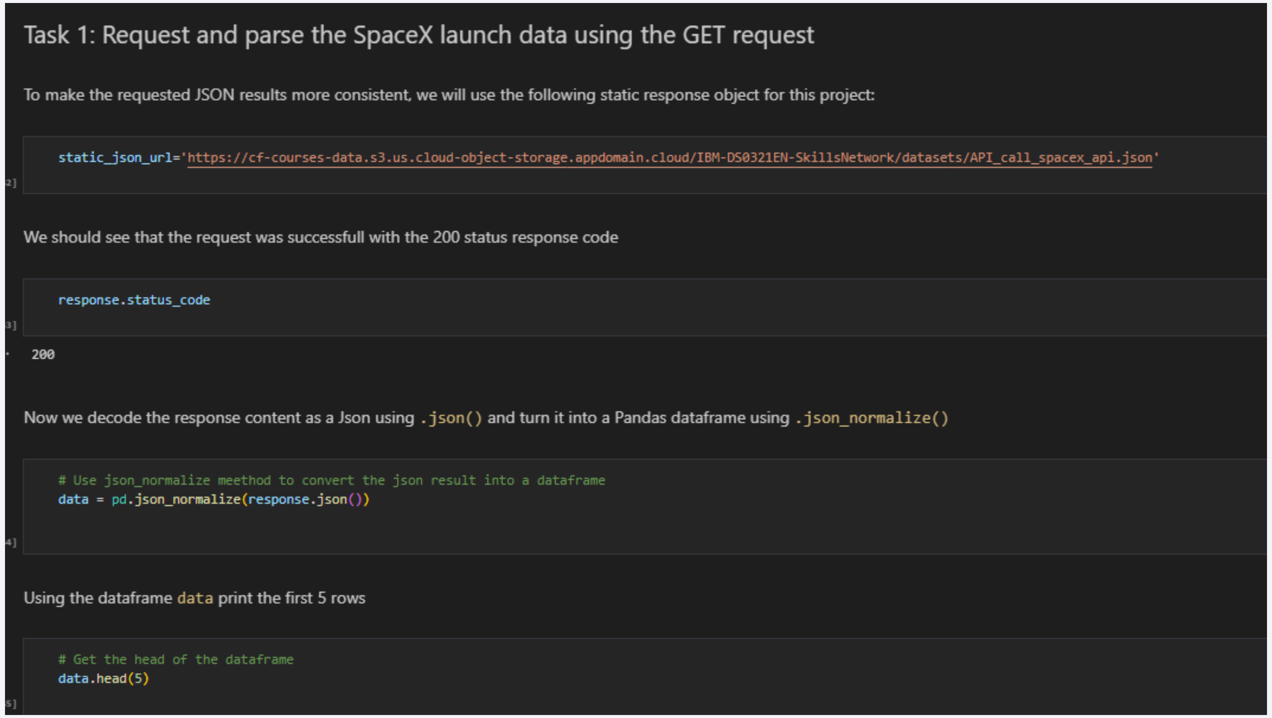
- Describe how data sets were collected.

  - ❑The data collection process involved utilizing different techniques. We made use of GET requests to access the SpaceX API and then decoded the response using the .json() function. This information was transformed into a Pandas dataframe using .json_normalize(). We proceeded to clean the data, fill in any missing values and check for other gaps. Furthermore, we implemented web scraping from Wikipedia for Falcon 9 launch records using BeautifulSoup. Our goal was to obtain the launch records as an HTML table, parse the table and convert it into a Pandas dataframe for further analysis.

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Add the GitHub URL of the completed SpaceX API calls notebook (https://github.com/ckampan/Applied-Data-Science-Capstone/blob/fab1b50c4fb31091dd47f3f668db2c91889109f6/jupyter-labs-spacex-data-collection-api.ipynb), as an external reference and peer-review purpose



Data Collection - Scraping

- Web scraping was used to obtain Falcon 9 launch records by means of BeautifulSoup. The HTML table was parsed and transformed into a Pandas dataframe for future analysis.

- GitHub URL is the following: https://github.com/ckampan/Applied-Data-Science-Capstone/blob/36c00c67f191d09b50c32bc8868de4c4e43c8bef/jupyter-labs-webscraping.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.text, 'html')
```

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

- Our exploratory data analysis led us to identify the training labels. We also calculated the launch count at each location and the frequency and number of each orbit. Additionally, we derived a landing outcome label from the outcome column and saved the results as a CSV file.
- Link to notebook: https://github.com/ckampan/Applied-Data-Science-Capstone/blob/ba21a2d20889a61c41290eb12ed8ea4364708d67/labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
- During the EDA with Data visualization, the following charts were utilized:
  - Scatterplot: to analyze the relationship between different features.
  - Bar chart: to review values for each one of the orbit types.
  - Line Chart: to look for trend increase/decrease on the success rate of rocket launches.
- Link to notebook: https://github.com/ckampan/Applied-Data-Science-Capstone/blob/af90f100663e6d30d48df97c6e1da1f3f5f544a1/IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

- SpaceX dataset was loaded into a SQL database.
- In order to extract insight from the data. We established the connection to the SQL database.
- We wrote queries to retrieve the following information:
  - The name of unique launch sites.
  - Total payload mass carried by boosters launched by NASA.
  - Average payload mass carried by booster.
  - Total number of successful failure mission outcomes.
  - Launch sites and booster version of failed landing outcomes.
- Link to notebook: https://github.com/ckampan/Applied-Data-Science-Capstone/blob/3ffa2c1322b002efa1c45ec6e8713750aa39c888/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- All launch sites have been designated on the folium map and marked with various objects such as markers, circles, and lines to indicate the success or failure of launches at each site.
- We categorized launch outcomes into two classes, class 0 for failures and class 1 for successes. Using color-coded markers and clusters, we were able to determine which launch sites had a higher success rate. Additionally, we calculated the proximity distances between each launch site.
- GitHub URL:https://github.com/ckampan/Applied-Data-Science-Capstone/blob/0ced14670da5888d2eb441a5cb9d94ddcc680138/lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- We created an interactive plotly dashboard, with pie charts showing the total launches per sites.
- We added a scatter graph to show the relationship between the outcome and payload mass for the different booster
- GitHub URL: https://github.com/ckampan/Applied-Data-Science-Capstone/blob/08679f8f9ef9d16ceaa210f85e2b422a39bc7f0e/spacex_dash_app.py

Predictive Analysis (Classification)

- The data was loaded using numpy and pandas, underwent transformation, and was divided into training and testing sets.
- Different machine learning models were constructed and their hyperparameters were optimized using GridSearchCV.
- Accuracy was utilized as the metric for evaluating the model, which was further improved through feature engineering and algorithm tuning.
- GitHub URL: https://github.com/ckampan/Applied-Data-Science-Capstone/blob/9874096d9aa92a20430d8ddeae7746b62d386423/Machine%20Learning%20Prediction.ipynb

Results

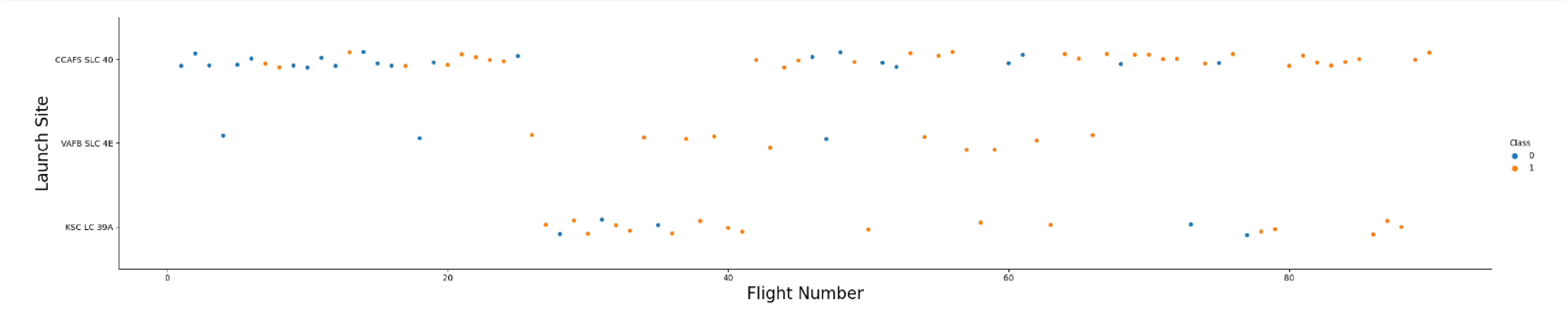
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

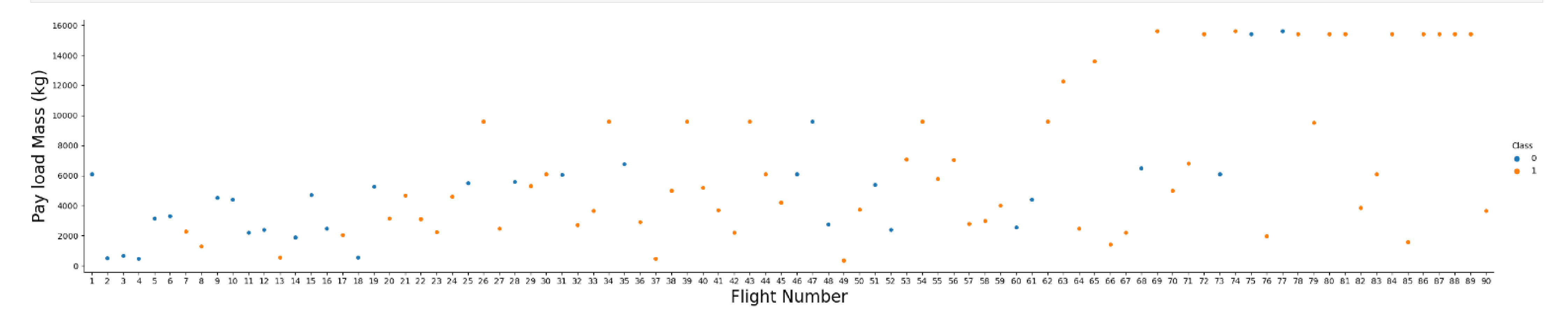
- Number vs. Launch Site



The success rate at a launch site increases as the flight amount becomes larger.

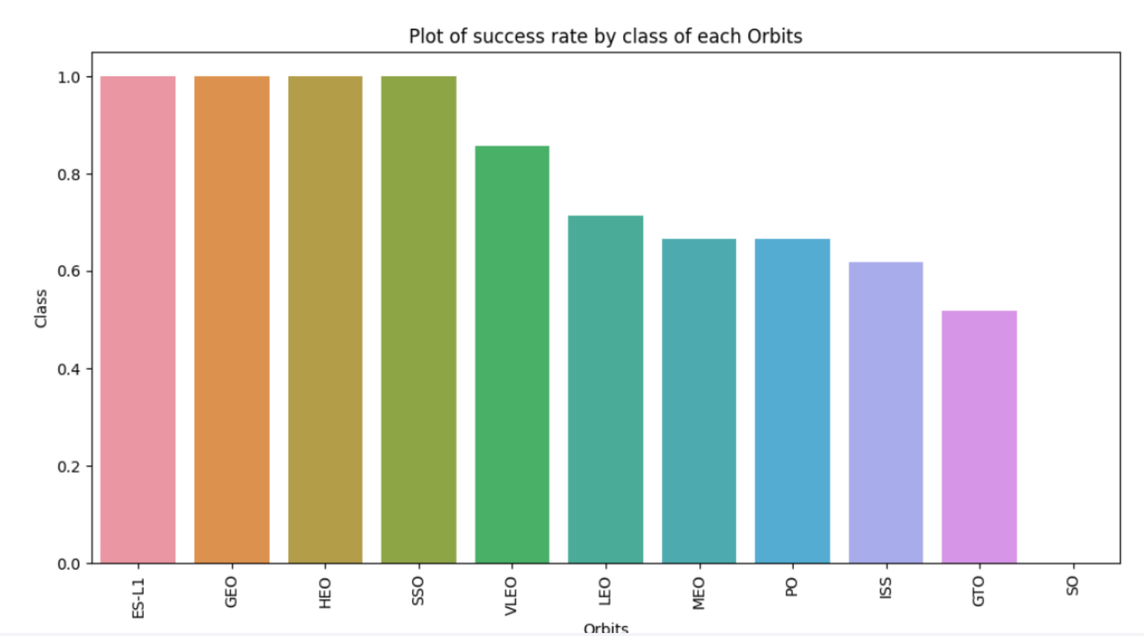
Payload vs. Launch Site

- Payload vs. Launch Site



The larger the payload mass the higher the success rate.

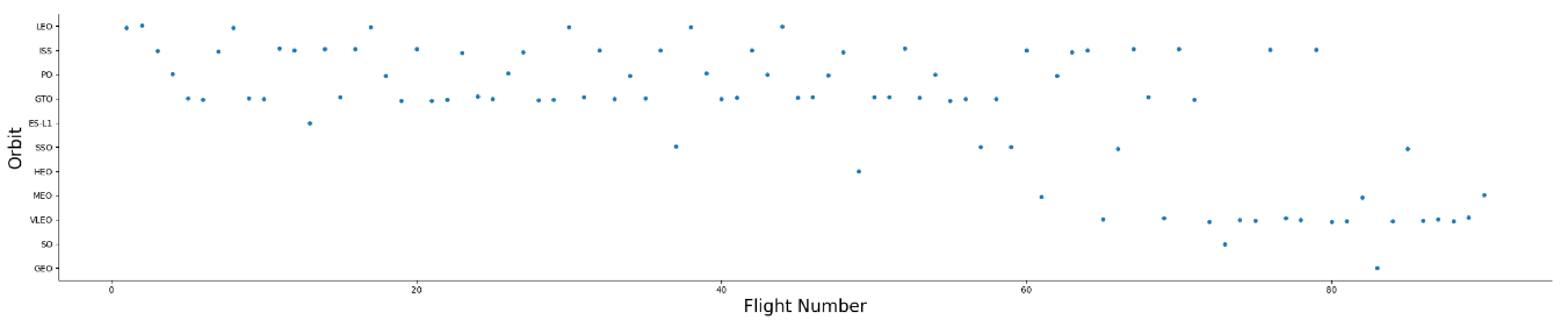
Success Rate vs. Orbit Type



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

Flight Number vs. Orbit Type

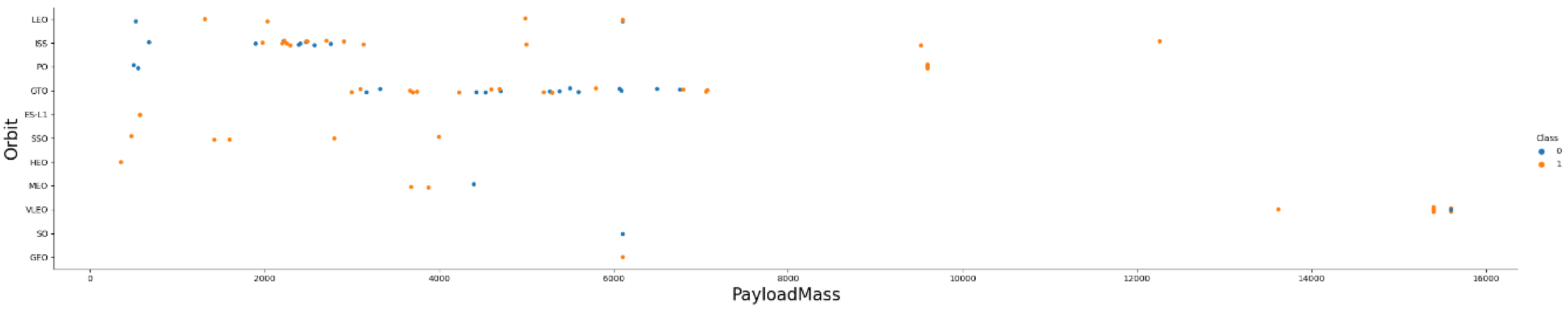
- Flight number vs. Orbit type



The plot depicts the relationship between flight number and orbit type. It is noted that in Low Earth Orbit (LEO), success is correlated with the number of flights, while no such correlation exists in Geostationary Transfer Orbit (GTO).

Payload vs. Orbit Type

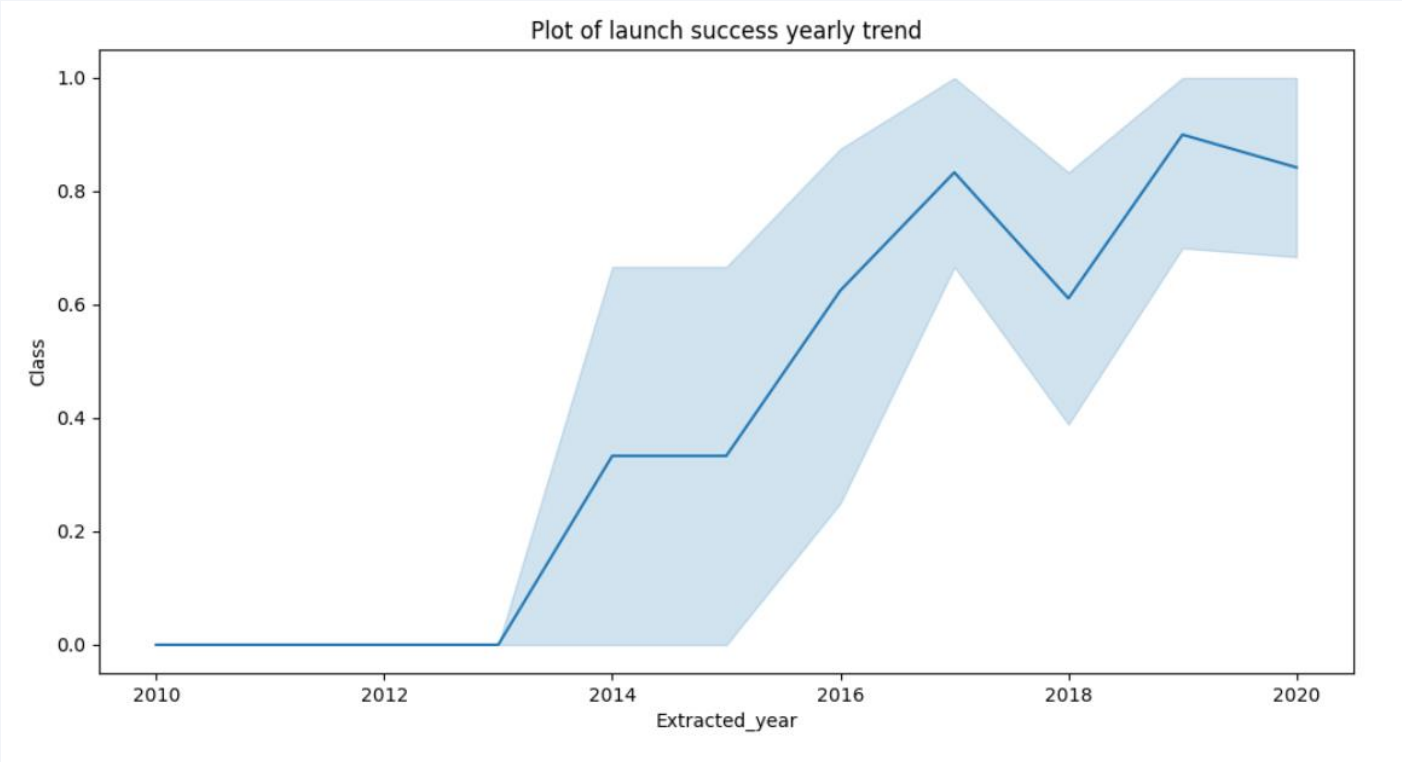
- Payload vs. orbit type



The successful landing rate is higher for PO, LEO, and ISS orbits when carrying heavy payloads.

Launch Success Yearly Trend

- The plot shows that the success rate has consistently increased from 2013 to 2020.



All Launch Site Names

- We use the distinct function to query the unique launch sites.

Display the names of the unique launch sites in the

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

Launch Site Names 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	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- We executed the query to display five records where the launch site names start with "CCA"

Total Payload Mass

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

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

- We used the query above and the function SUM to retrieve the total payload mass of 45596.

Average Payload Mass by F9 v1.1

Display 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

- Average payload mass was 2928.4 kg.

First Successful Ground Landing Date

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

- Dates of the first successful landing outcome on ground pad was 22nd December 2015

Successful Drone Ship Landing with Payload between 4000 and 6000

- The WHERE clause was utilized to filter for boosters that have successfully landed on a drone ship and the AND condition was applied to identify successful landings with payload mass between 4000 and 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

Total Number of Successful and Failure Mission Outcomes

- The wildcard character '%' was employed with the LIKE operator to filter WHERE Mission Outcome was either a success or failure.

List 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

Boosters Carried Maximum Payload

- The booster that carried the maximum payload was determined by using a subquery in the WHERE clause combined with the MAX() function.

List the names of the booster_versions which have carried the maxim

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

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)

- The WHERE clause, LIKE, AND, and BETWEEN conditions were combined to filter for failed landing outcomes on drone ships, their booster versions, and launch site names in the year 2015.

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

- We selected the landing outcomes and their count from the data and used the WHERE clause to filter for landing outcomes between June 4th, 2010 and March 20th, 2010. The GROUP BY clause was used to group the landing outcomes, and the ORDER BY clause was used to sort the grouped landing outcomes in descending order.

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

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

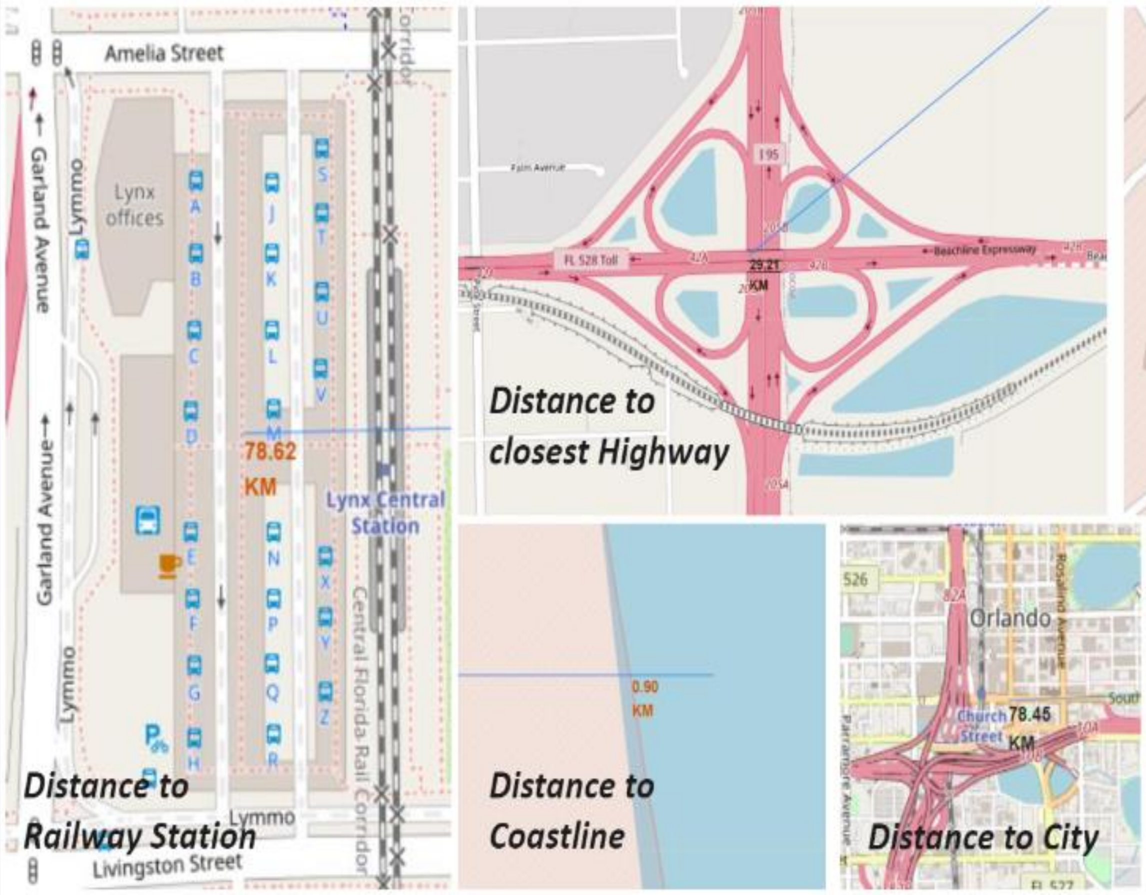
Section 3

Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

<Folium Map Screenshot 2>

<Folium Map Screenshot 3>

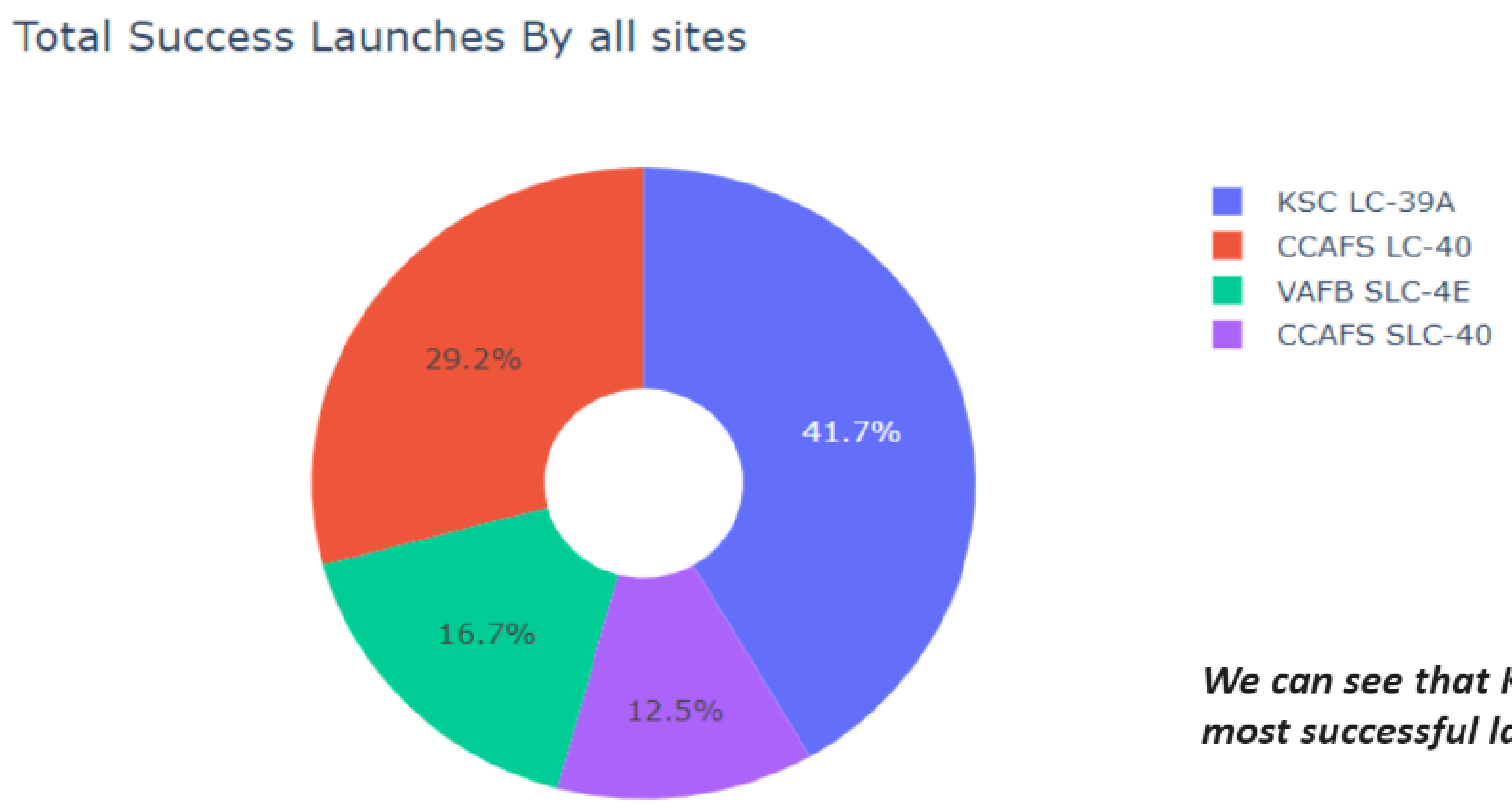


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

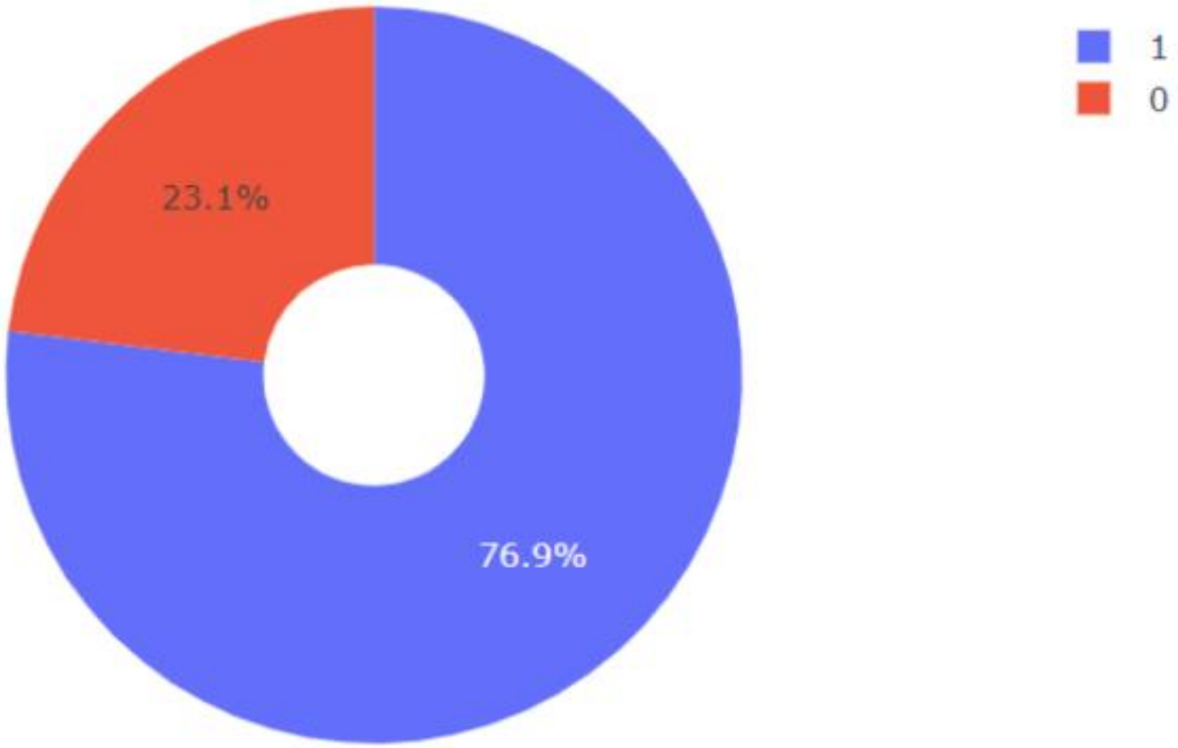
Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>



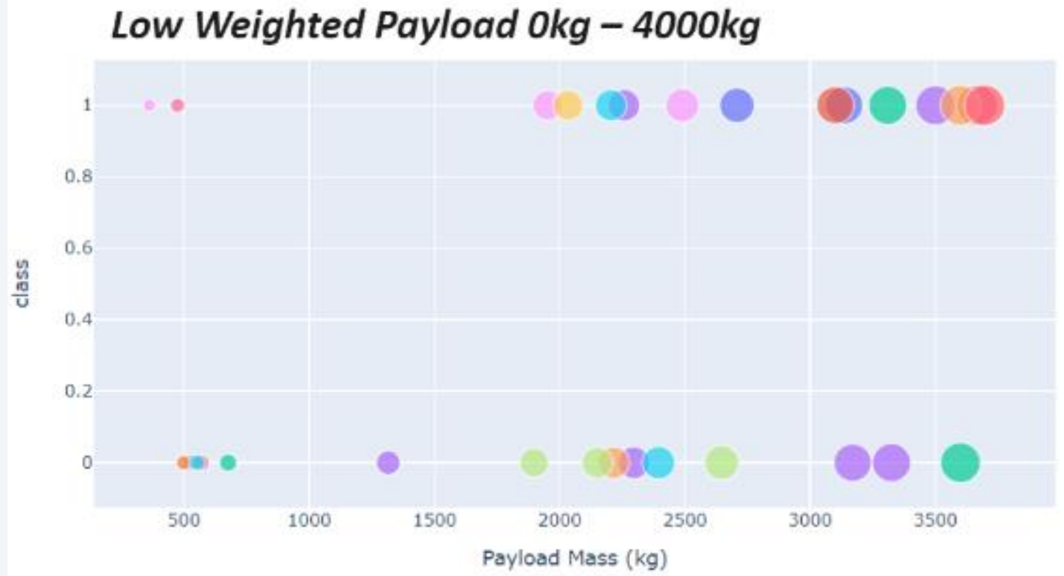
We can see that KSC LC-39A had the most successful launches from all the sites

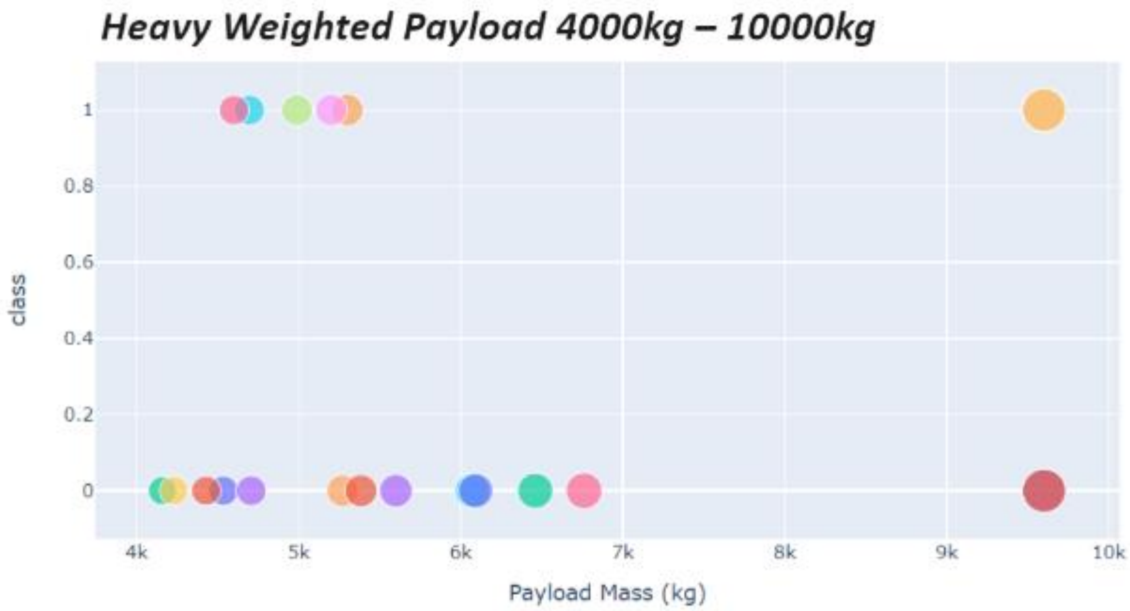
<Dashboard Screenshot 2>



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

<Dashboard Screenshot 3>





We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

Classification Accuracy

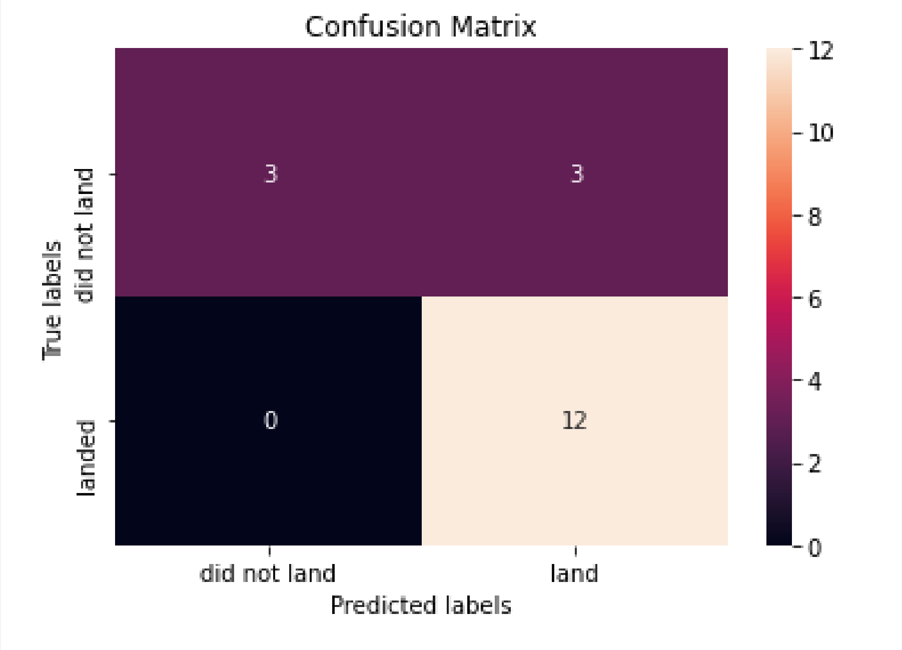
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

- The decision tree classifier model exhibits the highest classification accuracy

Confusion Matrix

- The confusion matrix for the decision tree classifier demonstrates the classifier's ability to differentiate between the various classes. The primary issue is the high number of false positives, where the classifier mistakenly labels an unsuccessful landing as a successful one.



Conclusions

- Based on our analysis, we can draw the following conclusions:

1. The success rate at a launch site increases as the flight amount grows larger.
2. The launch success rate saw an upward trend from 2013 to 2020.
3. The ES-L1, GEO, HEO, SSO, and VLEO orbits had the highest success rate.
4. Kennedy Space Center LC-39A was the most successful launch site.
5. The decision tree classifier is the most effective machine learning algorithm for this task.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!