



IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science



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 with SQL
 - Exploratory Data Analysis with Data Visualization
 - Interactive Visual Analytics with Folium
 - Machine Learning Prediction
- Summary of all results
 - Exploratory Data Analysis result
 - Interactive analytics 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; 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
 - How to build, tune, evaluate classification models

Data Collection

The data was collected using various methods

- Data collection was done using get request to the SpaceX API.
- Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
- We then cleaned the data, checked for missing values and fill in missing values where necessary.
- In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
- The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

- We request and parse the data using GET, filter the requested data and did some basic data wrangling and formatting.

The link to the notebook is

<https://github.com/inakilopezjuan/IBMDatascienceCapstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb>

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

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

```
response.status_code
```

```
200
```

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

```
# Hint data['BoosterVersion']!='Falcon 1'  
data_falcon9 = launch_df[launch_df['BoosterVersion']!='Falcon 1']  
  
# Display the first few rows of the filtered DataFrame to verify  
data_falcon9.head()
```

```
# Calculate the mean value of PayloadMass column
```

```
# Replace the np.nan values with its mean value  
mass_mean = data_falcon9["PayloadMass"].mean()  
data_falcon9["PayloadMass"] = data_falcon9["PayloadMass"].replace(np.nan, mass_mean,inp
```

Data Collection - Scraping

- We Requests the falcon9 Launch wiki page from its url
- Extract all names from HTML table header
- We parsed the table and converted it into a pandas dataframe.

The link to the notebook is

<https://github.com/inakilopezjuan/IBMDatascienceCapstone/blob/main/jupyter-labs-webscraping.ipynb>

```
# 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  
soup = BeautifulSoup(response.content, 'html.parser')
```

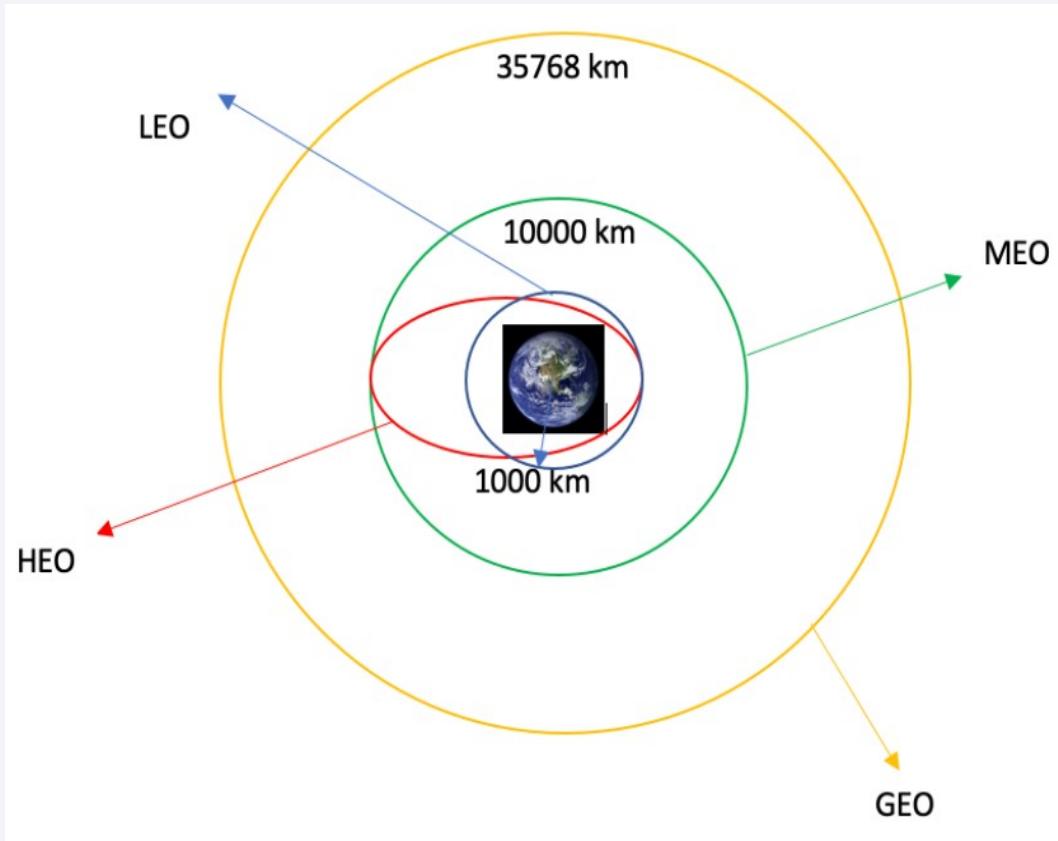
```
# Use the find_all function in the BeautifulSoup object, with element type `table`  
# Assign the result to a list called `html_tables`  
html_tables = soup.find_all("table")
```

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]
```

```
df = pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

Data Wrangling



We calculated number of launches on each site, number of occurrence of each orbit, mission outcome and created a landing outcome label.

The link to the notebook is

<https://github.com/inakilopezjuan/IBMDatascienceCapstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb>

EDA with Data Visualization

We plotted the following charts:

- 1) FlightNumber vs LaunchSite
- 2) LaunchSite vs PayloadMass
- 3) SuccessRate and Orbit
- 4) FlightNUmber and Orbit
- 5) PayloadMass and Orbit
- 6) LaunchSuccess yearly trend

Github: <https://github.com/inakilopezjuan/IBMDatascienceCapstone/blob/main/edadataviz.ipynb>

EDA with SQL

To better understand SpaceX data set we performed these queries:

- 1) Display the names of the unique launch site
- 2) Display 5 records where launch begins with CCA
- 3) Display load mass carried by booster launched by NASA
- 4) Display average payload mass carried by booster version F9 v1.1
- 5) List the date when the first successful landing outcome in ground pad was achieved.
- 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. Use a subquery
- 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.

Build an Interactive Map with Folium

We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.

We 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 rate.

We calculated the distances between a launch site to its proximities. We answered some question for instance:

- Are launch sites near railways, highways and coastlines.
- Do launch sites keep certain distance away from cities.

Build a Dashboard with Plotly Dash

We built an interactive dashboard with Plotly dash

We plotted pie charts showing the total launches by a certain sites

We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

Predictive Analysis (Classification)

We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.

We built different machine learning models and tune different hyperparameters using GridSearchCV.

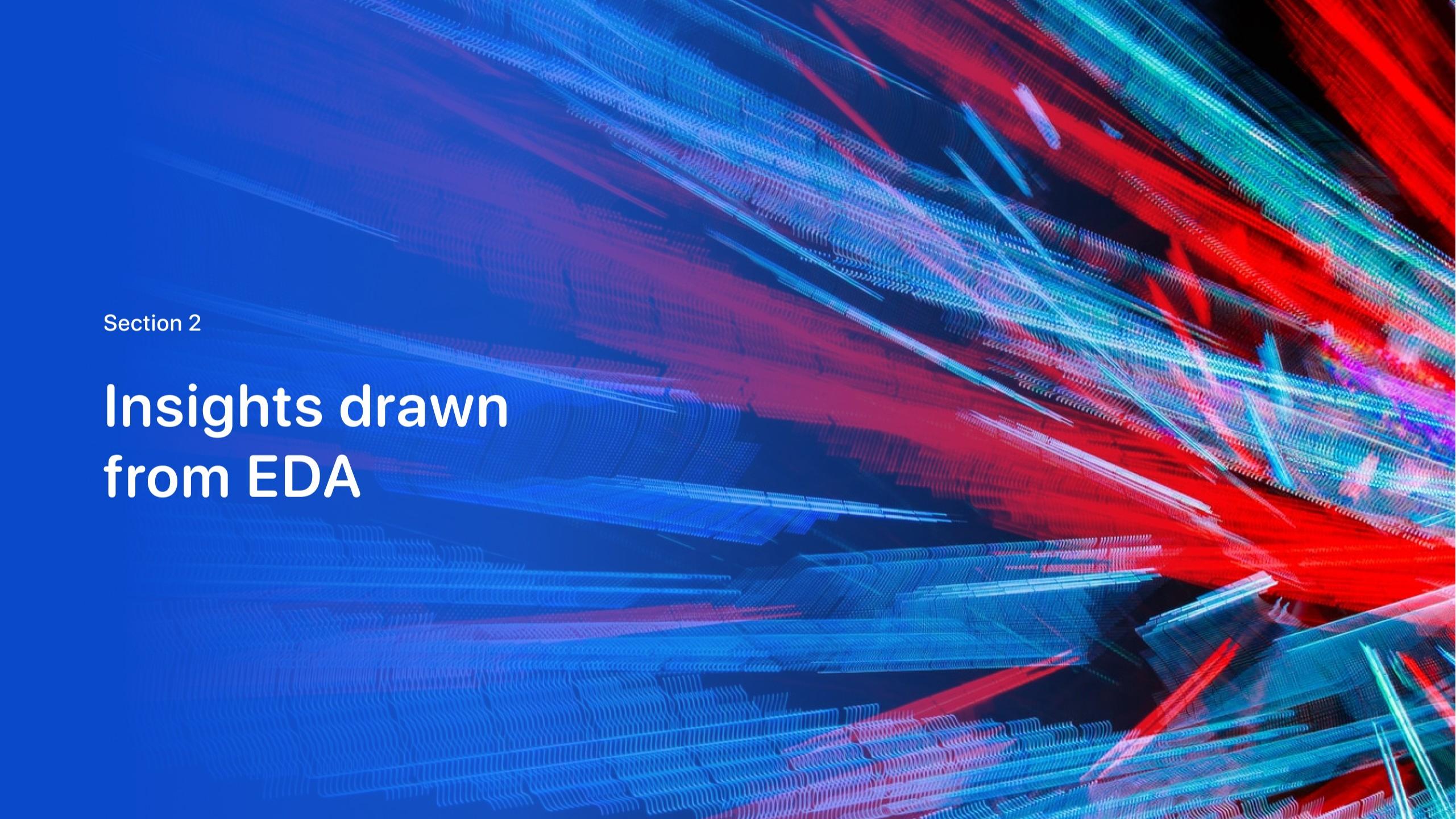
We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.

We found the best performing classification model.

[https://github.com/inakilopezjuan/IBMDatascienceCapstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5%20\(1\).ipynb](https://github.com/inakilopezjuan/IBMDatascienceCapstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5%20(1).ipynb)

Results

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

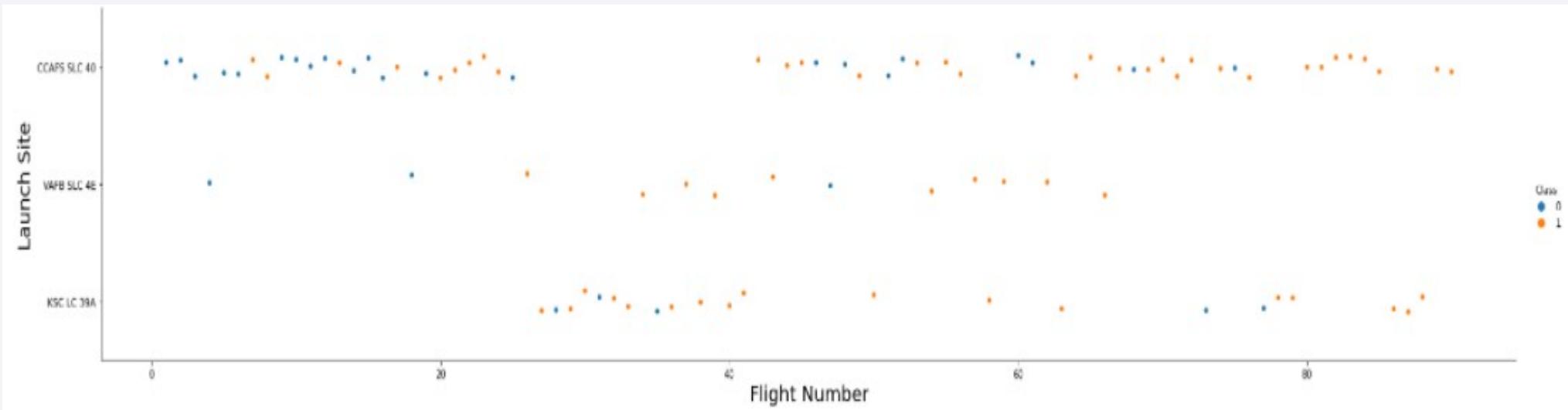
The background of the slide features a complex, abstract digital visualization. It consists of numerous thin, glowing lines that create a sense of depth and motion. The lines are primarily blue and red, with some green and purple highlights. They form a grid-like structure that curves and twists across the frame, resembling a three-dimensional space or a network of data points. The overall effect is futuristic and dynamic.

Section 2

Insights drawn from EDA

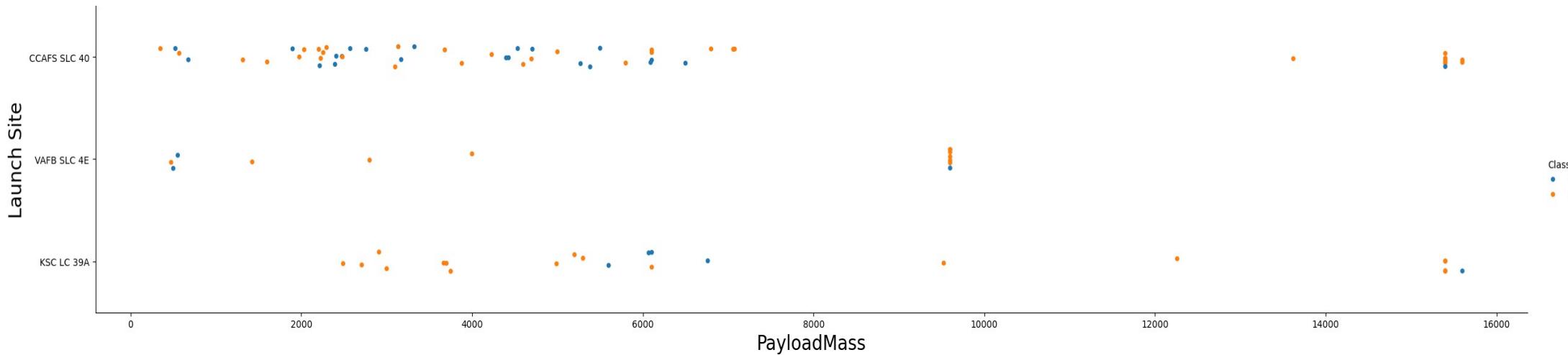
Flight Number vs. Launch Site

- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site.



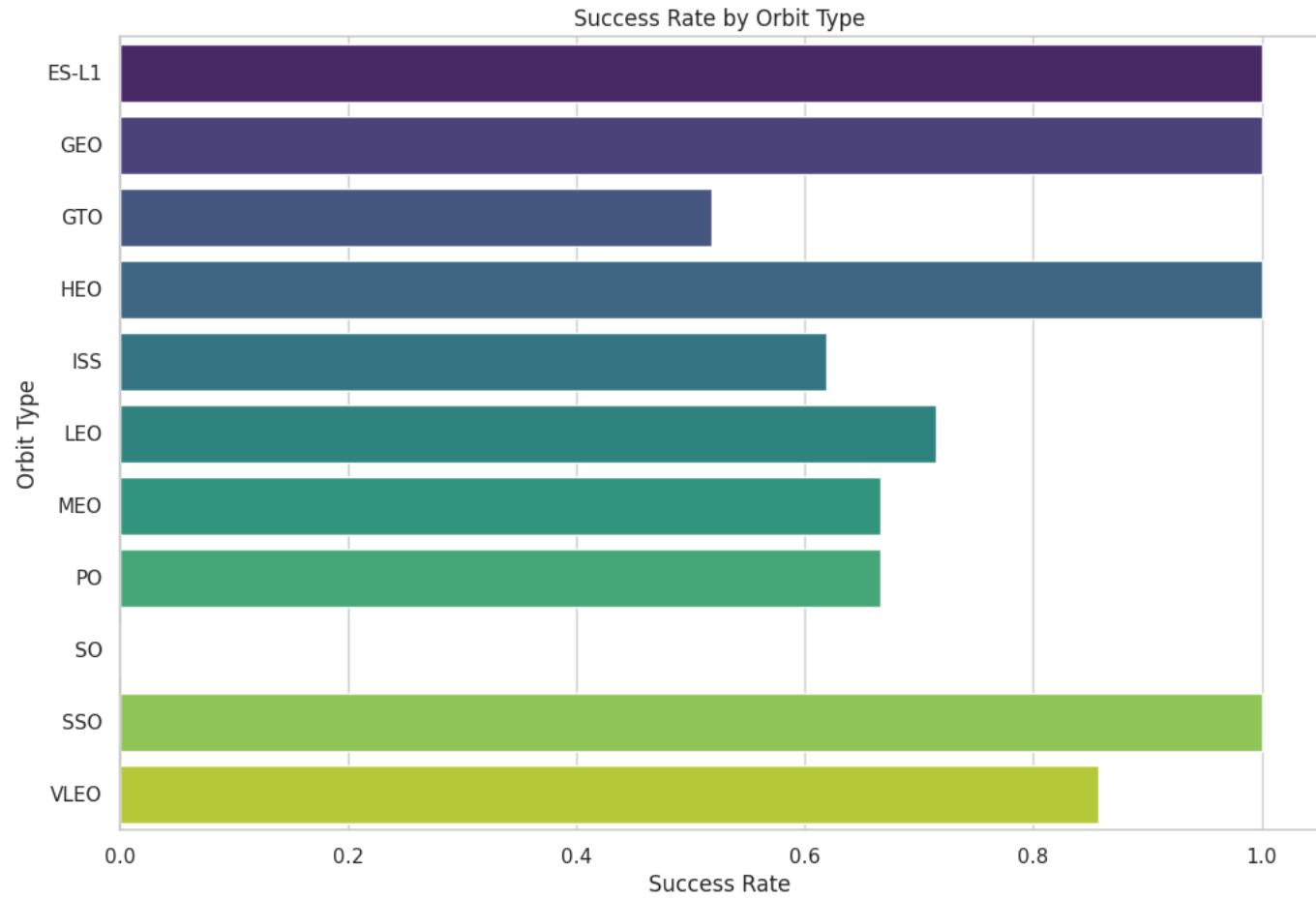
Payload vs. Launch Site

- For every launch site the higher the mass, the higher the success rate



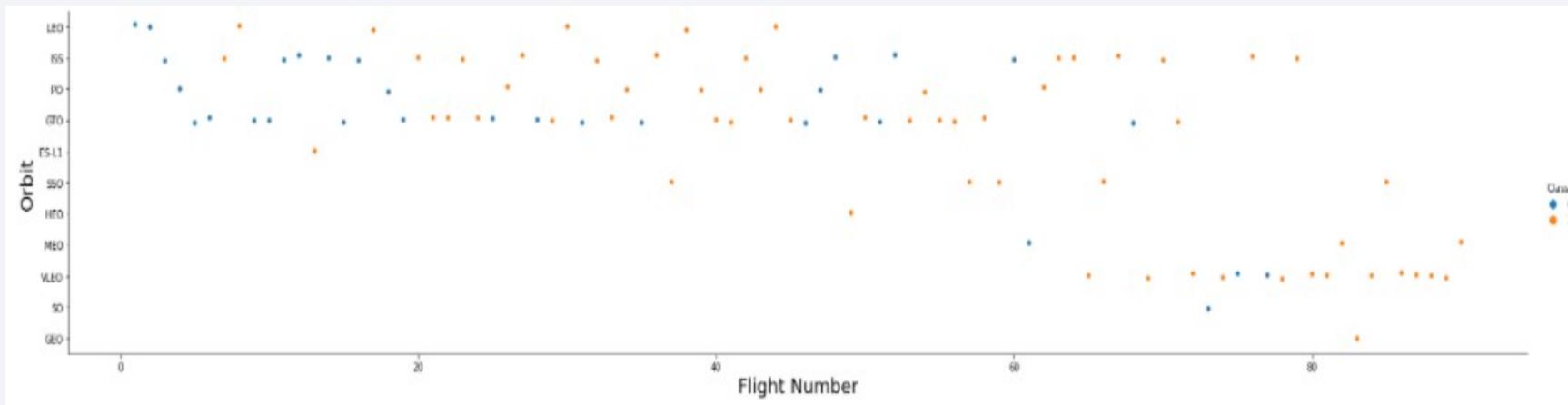
Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



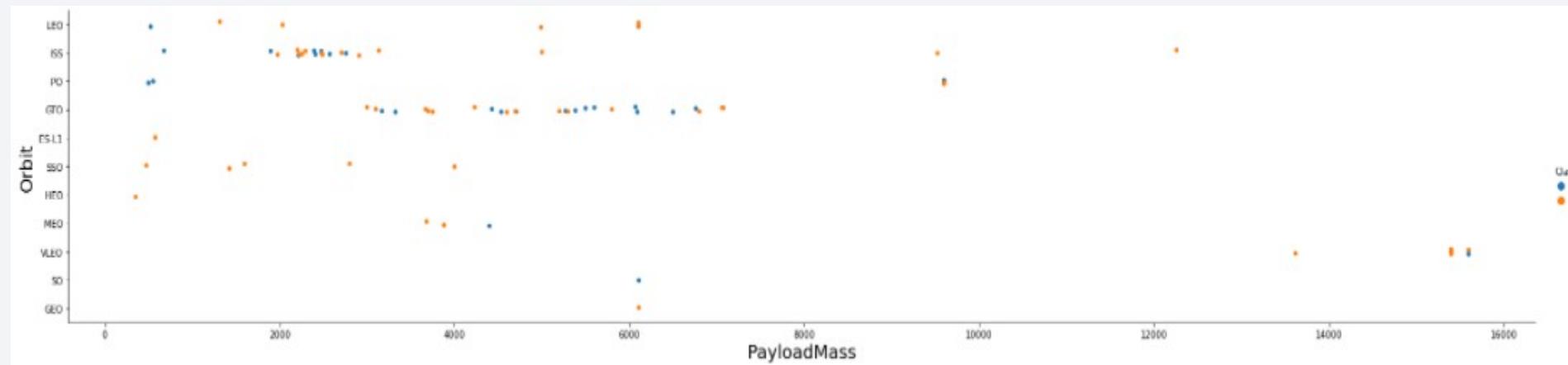
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that 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.



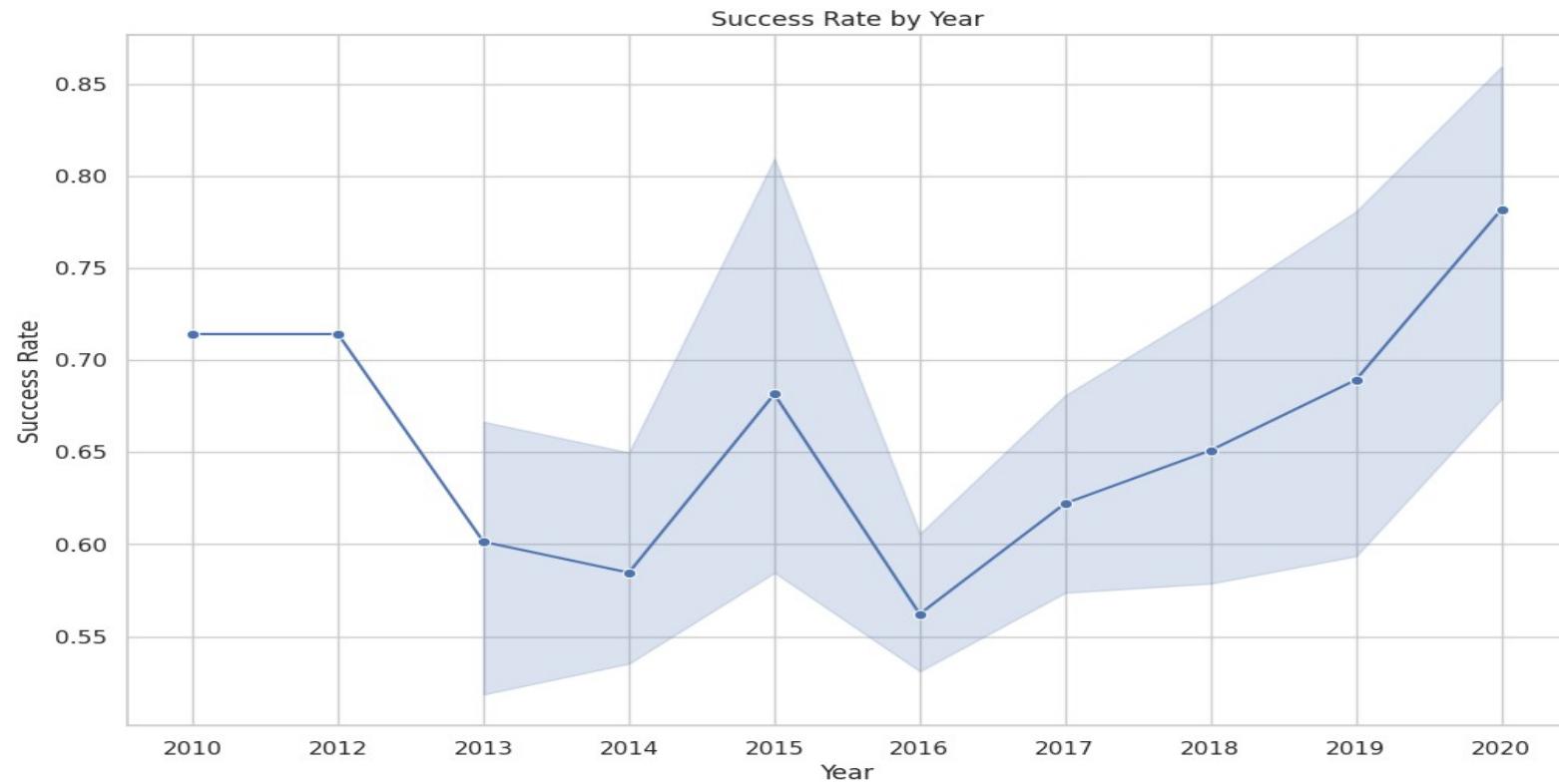
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2016 kept on increasing till 2020.



All Launch Site Names

- Query:query ='SELECT DISTINCT "Launch_Site" FROM SPACEXTBL;'
- `Launch_Site`

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

Launch Site Names Begin with 'CCA'

- Query: query_2 = 'SELECT * FROM SPACEXTBL WHERE "Launch_Site" LIKE "CCA%" LIMIT 5;'
launch_sites_CCA = pd.read_sql_query(query_2, con)
launch_sites_CCA

	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS_KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-12-08	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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- query ='SELECT SUM("PAYLOAD_ MASS __ KG_") AS Total_Payload_Mass FROM SPACEXTBL WHERE "Customer" ="NASA (CRS)";'
- Result:: 45596

Average Payload Mass by F9 v1.1

- Query: 'SELECT AVG("PAYLOAD_MASS__KG_") AS Average_Payload_Mass FROM SPACEXTBL WHERE "Booster_Version" = "F9 v1.1";'
- Result: 2928.4

First Successful Ground Landing Date

We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

```
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
```

Successful Drone Ship Landing with Payload 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

We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

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

failureoutcome
0 1

We used wildcard like '%' to filter for **WHERE** MissionOutcome was a success or a failure.

Boosters Carried Maximum Payload

We determined the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

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

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

We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

```
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)

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))

```
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
```

We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.

We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

The background of the slide is a photograph taken from space at night. It shows the curvature of the Earth against the dark void of space. City lights are visible as numerous small white and yellow dots, primarily concentrated in the lower right quadrant where the United States appears. In the upper left quadrant, the green and blue glow of the aurora borealis (Northern Lights) is visible, appearing as horizontal bands of light.

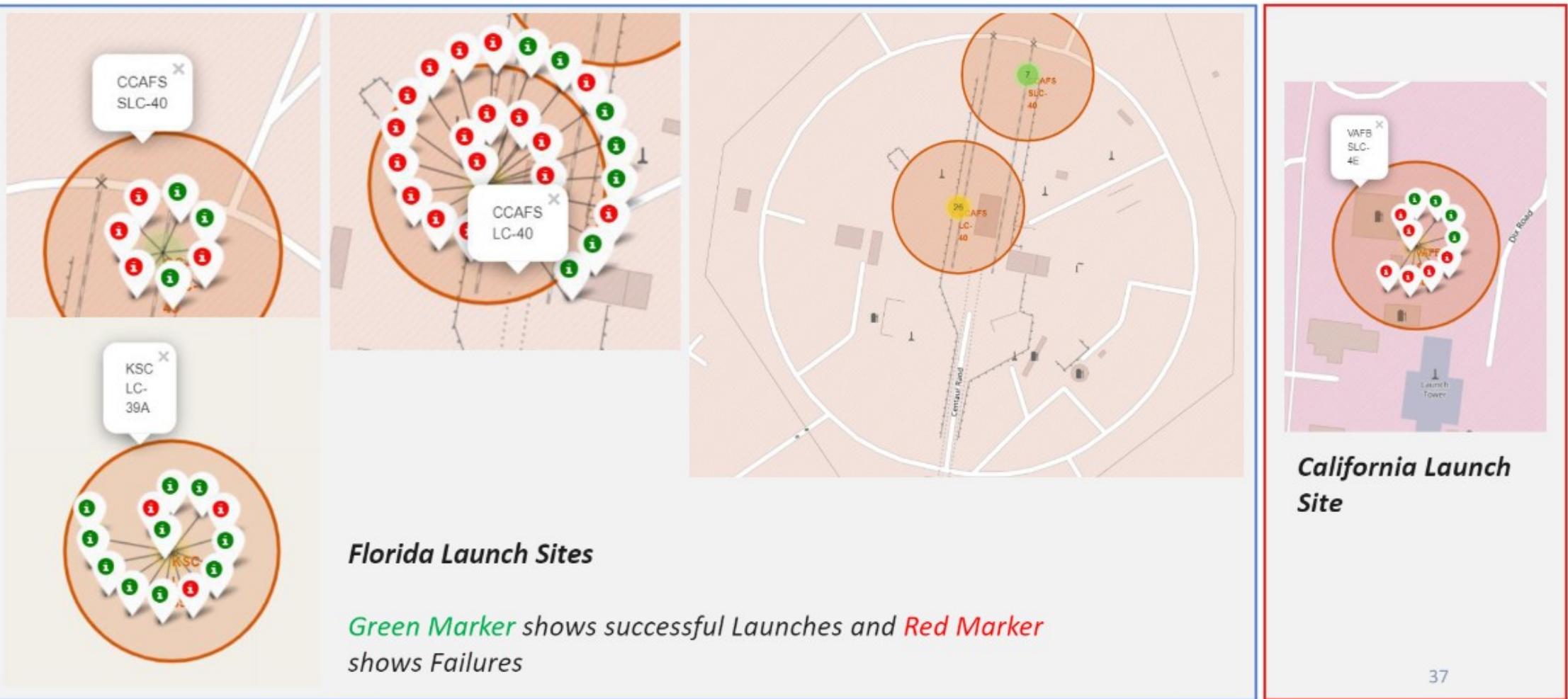
Section 4

Launch Sites Proximities Analysis

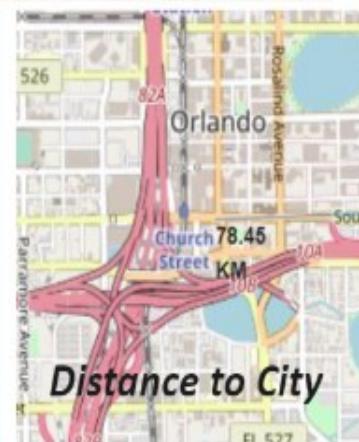
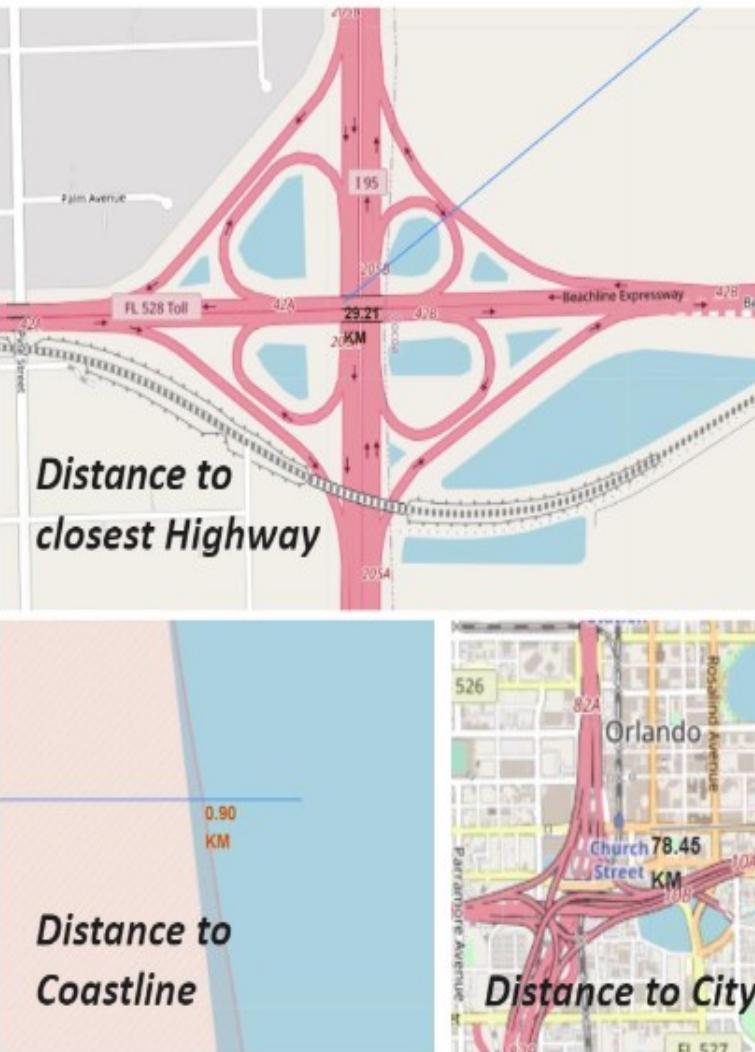
All launch sites global map markers



Markers showing launch sites with color labels



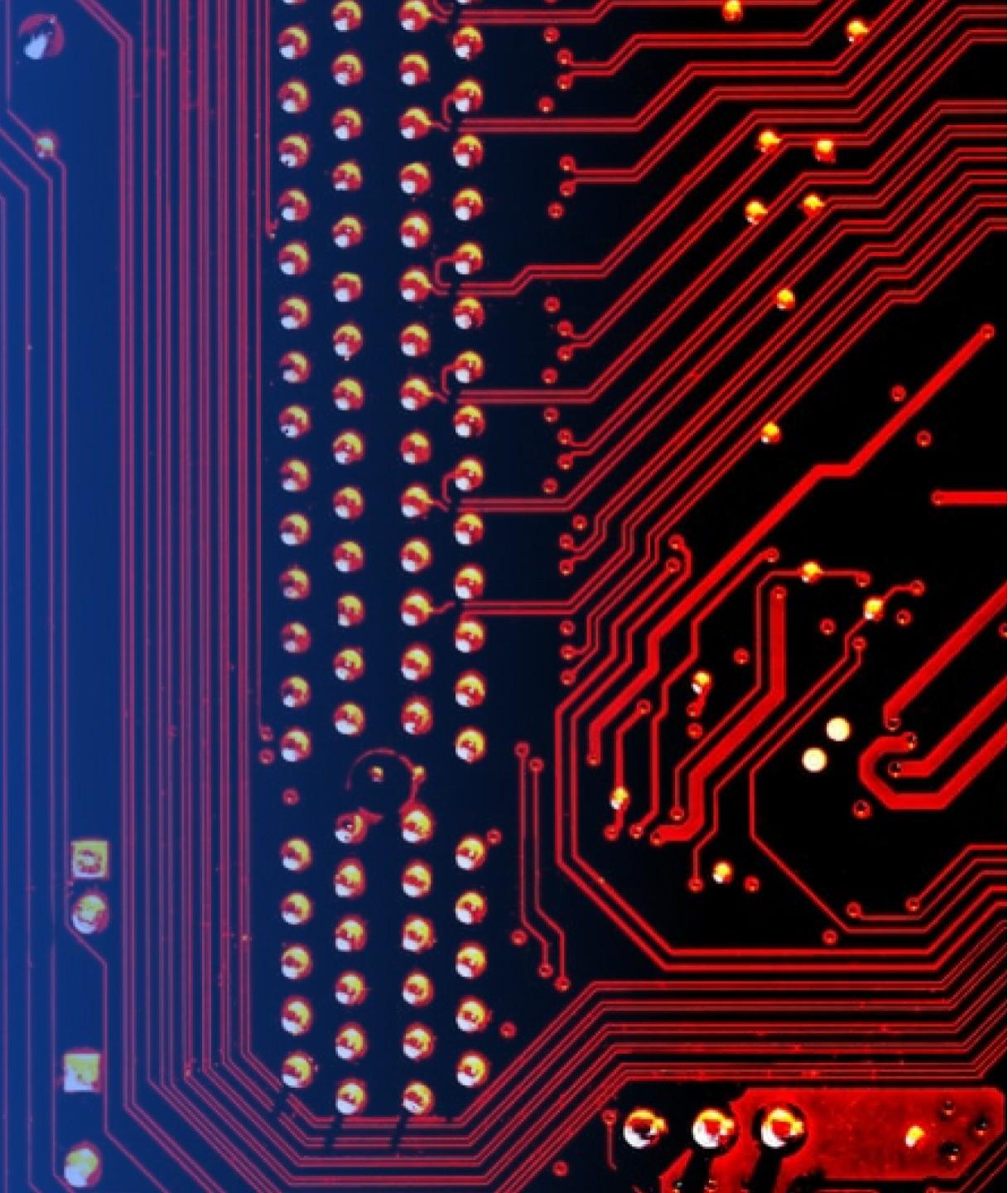
Launch Site distance to landmarks



- 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

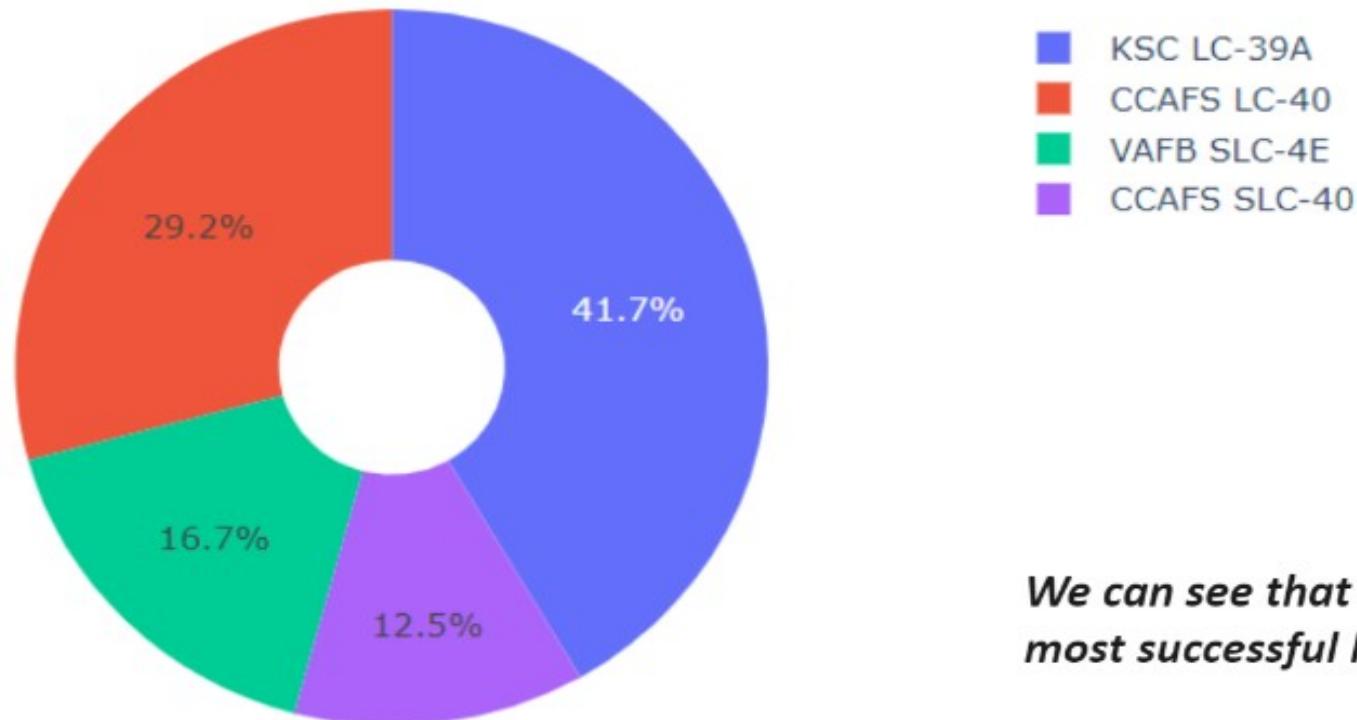
Section 5

Build a Dashboard with Plotly Dash



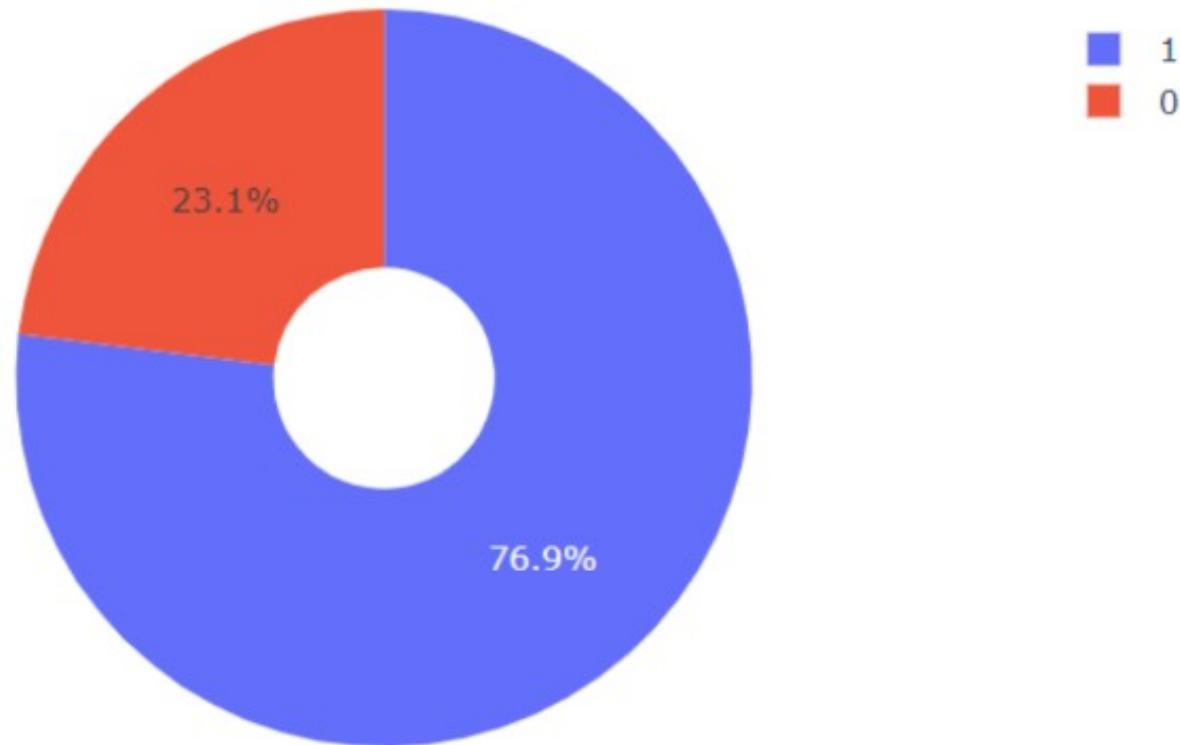
Pie chart showing the success percentage achieved by each launch site

Total Success Launches By all sites



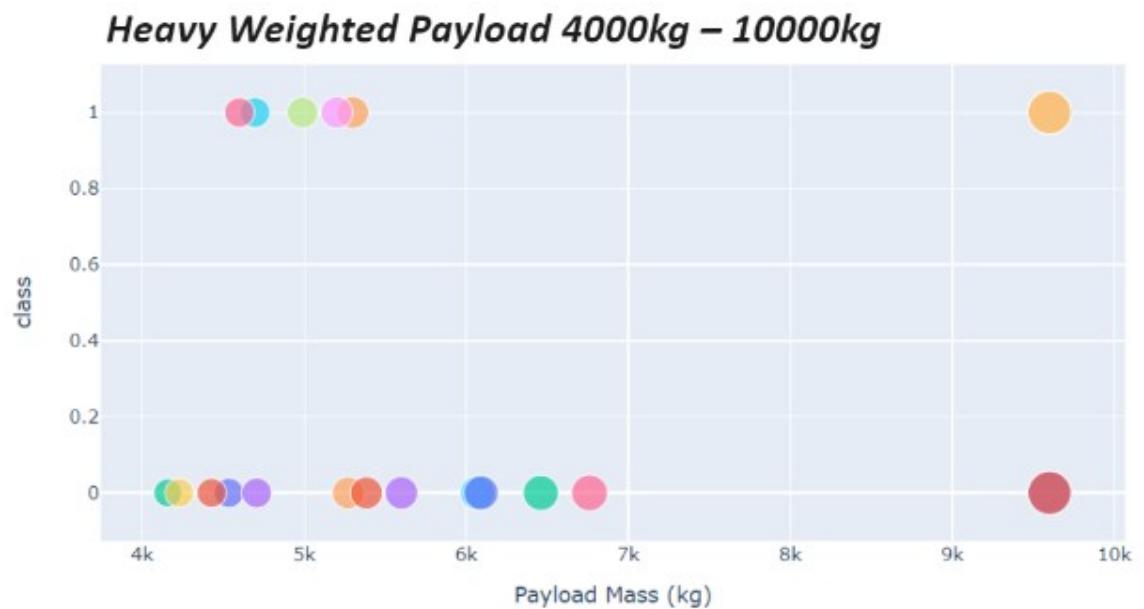
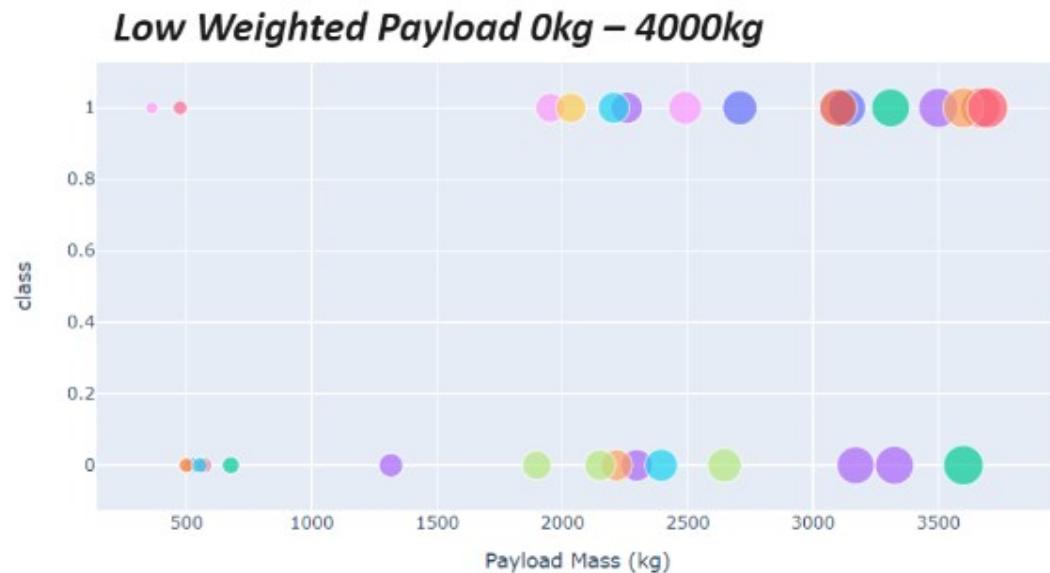
We can see that KSC LC-39A had the most successful launches from all the sites

Pie chart showing 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, with different payload selected in the range slider



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

Section 6

Predictive Analysis (Classification)

Classification Accuracy

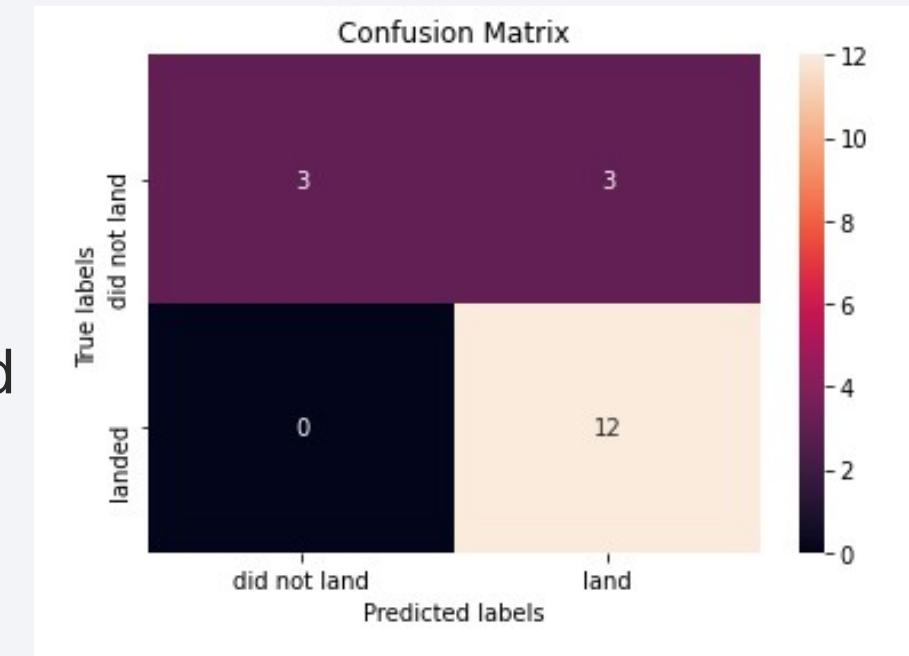
- The decision tree classifier is the model with the highest classification accuracy

```
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'}
```

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that 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

We can conclude that:

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

Launch success rate started to increase in 2013 till 2020.

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

KSC LC-39A had the most successful launches of any sites.

The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!

