IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Nouhaila El Morjani
2025-10-15

Outline

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

Executive Summary

Objective: To provide a new rocket company, SpaceY, with a data-driven strategy to compete against SpaceX on launch costs.

Methodology: The project employed a full data science pipeline:

1. Data Collection: Mission data was gathered from public sources, including the SpaceX API.

2. Data Analysis & Wrangling: Data was cleaned and explored using SQL and Python to identify patterns and key success factors.

3. Predictive Modeling: A machine learning model was developed to forecast mission outcomes.

Key Results: The analysis successfully identified the critical mission parameters that influence launch success. The resulting predictive model can reliably determine the likelihood of a SpaceX Falcon 9 first-stage booster landing successfully. As a reusable booster represents a significant cost saving (over $15 million), this prediction serves as a direct indicator of launch cost. This enables SpaceY to formulate competitive and informed bids against SpaceX.

Introduction

Project Background and Context:

SpaceX has revolutionized the space industry with its reusable Falcon 9 rocket, offering launches at $62 million—a fraction of the competitor's cost. This project analyzes SpaceX's launch data to help new company, Space Y, understand and compete in this market.

Problems to Find Answers For:

1. Can we predict if a Falcon 9's first stage will land successfully based on factors like payload, orbit, and launch site?
2. Where is the best location to conduct launches?
3. What is the most effective way to estimate launch costs by predicting first-stage landing success?

Section 1

Methodology

Methodology

Executive Summary

- **Data collection methodology:**
  - Describe Data from Space X was obtained from 2 sources:

    Space X API (https://api.spacexdata.com/v4/rockets/)

    WebScaping

    (https://en.wokopedia.org/wiki/List_of_Falcon/_9/_and_Falcon_Heavy_Launches)
- **Perform data wrangling**
- Identifying orbit frequencies , Successful vs failed landings , and labelling missions accordingly

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    - Data that was collected until this step were normalized, divided in training and test data sets and evaluated by four different classification models ,being the accuracy of each model evaluated using different combination of parameters

Github URL: https://github.com/nouhaila-elmorjani/IBM-Data-Science-Capstone-SpaceX

Data Collection

The data was collected using a multi-source approach:

- Primary data collection was done using GET requests to the SpaceX API
- We decoded the response content as JSON using json() function and converted it into a pandas dataframe using json_normalize()
- We performed data cleaning, handled missing values, and conducted basic data wrangling
- Additionally, we implemented web scraping from Wikipedia for Falcon 9 launch records using BeautifulSoup
- The objective was to create a comprehensive dataset by combining structured API data with historical launch records

Data Collection – SpaceX API

API Request & Data Acquisition

- GET request to SpaceX API endpoint for past launch data
- JSON response parsing and conversion to structured format

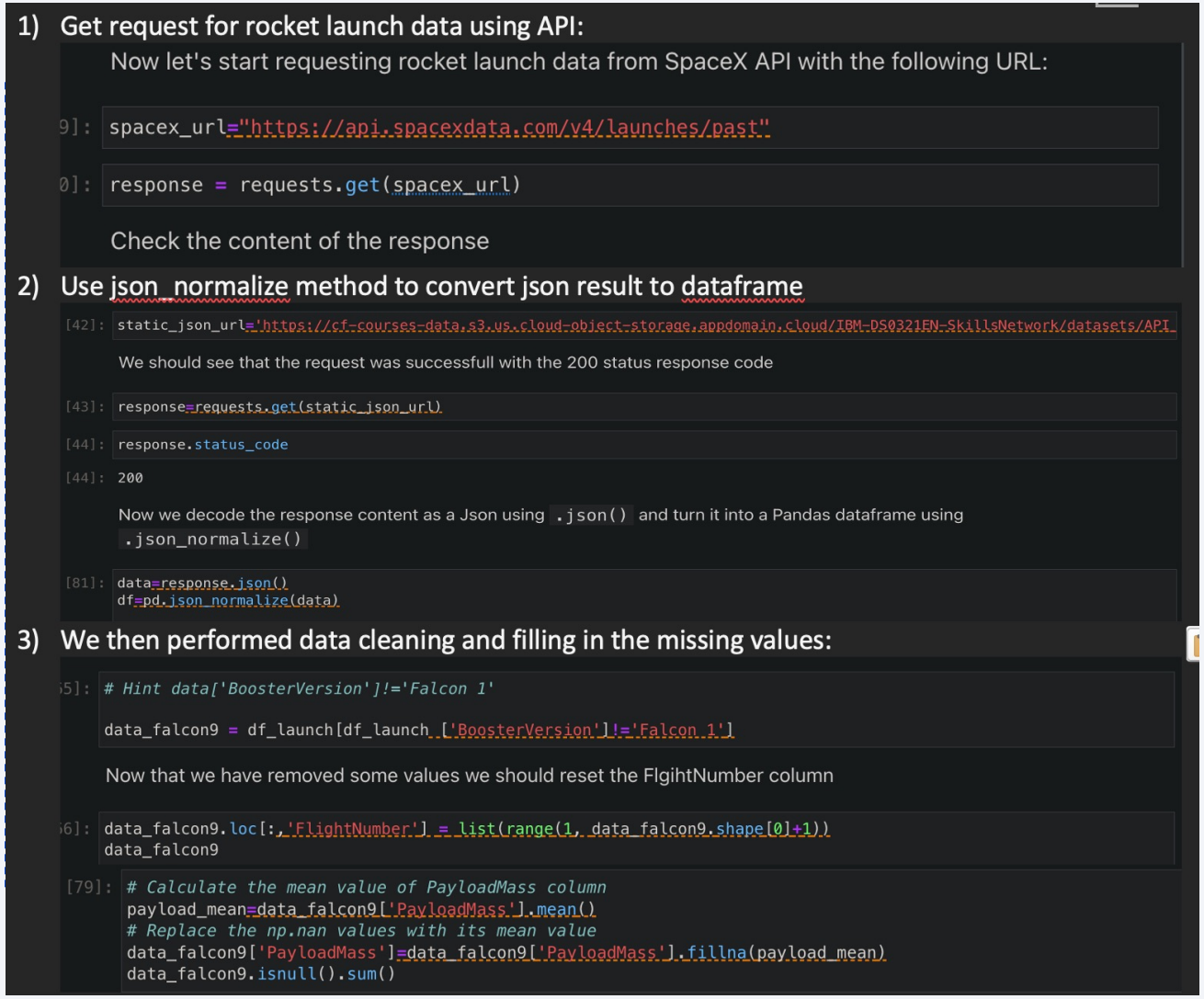
Data Processing & Normalization

- Applied json_normalize() to convert JSON response into dataframe
- Extracted and transformed complex nested JSON structures

Data Cleaning & Preparation

- Handled missing values in critical columns like PayloadMass
- Performed data wrangling and quality assurance
- Prepared clean dataset for analysis

GithubURL:https://github.com/nouhaila-elmorjani/IBM-Data-Science-Capstone-SpaceX/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- **Implementation**
  - Utilized BeautifulSoup for HTML parsing of Wikipedia pages
  - Extracted Falcon 9 launch records from structured tables
- **Data Extraction & Conversion**
  - Retrieved comprehensive launch history data
  - Converted HTML tables to pandas dataframe format
  - Merged with API data for enhanced dataset completeness
- **Data Validation & Enhancement**
  - Provided cross-reference validation for API data
  - Added historical context and missing launch records
  - Ensured comprehensive coverage of all Falcon 9 missions

Github URL: https://github.com/nouhaila-elmorjani/IBM-Data-Science-Capstone-SpaceX/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

1) Apply HTTP Get method to request the Falcon 9 rocket launch page

```
[7]: # use requests.get() method with the provided static_url and headers
     # assign the response to a object
     response=requests.get(static_url,headers=headers)
```

2) Create BeautifulSoup object from t he HTML response

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

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

```
]: # Use soup.title attribute
   print(f"Status: {response.status_code},Title : {soup.title.text}")
```

```
Status: 200,Title : List of Falcon 9 and Falcon Heavy launches - Wikipedia
```

3) Extract all column names from the HTML table header

```
l9]: column_names = []

     # Apply find_all() function with `th` element on first_launch_table
     th_elements=first_launch_table.find_all('th')
     # 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
     for th in th_elements:
         name=extract_column_from_header(th)
         if name is not None and len(name)>0:
             column_names.append(name)
```

4) Create a dataframe by parsing the launch HTML tables

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

5) Export data to csv

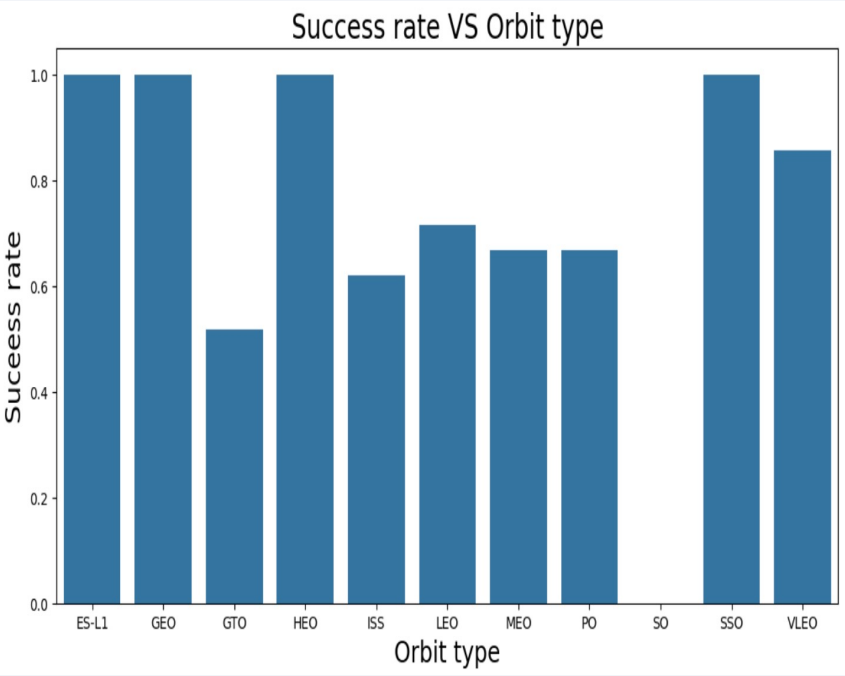
```
31]: df.to_csv('spacex_web_scraped.csv', index=False)
```

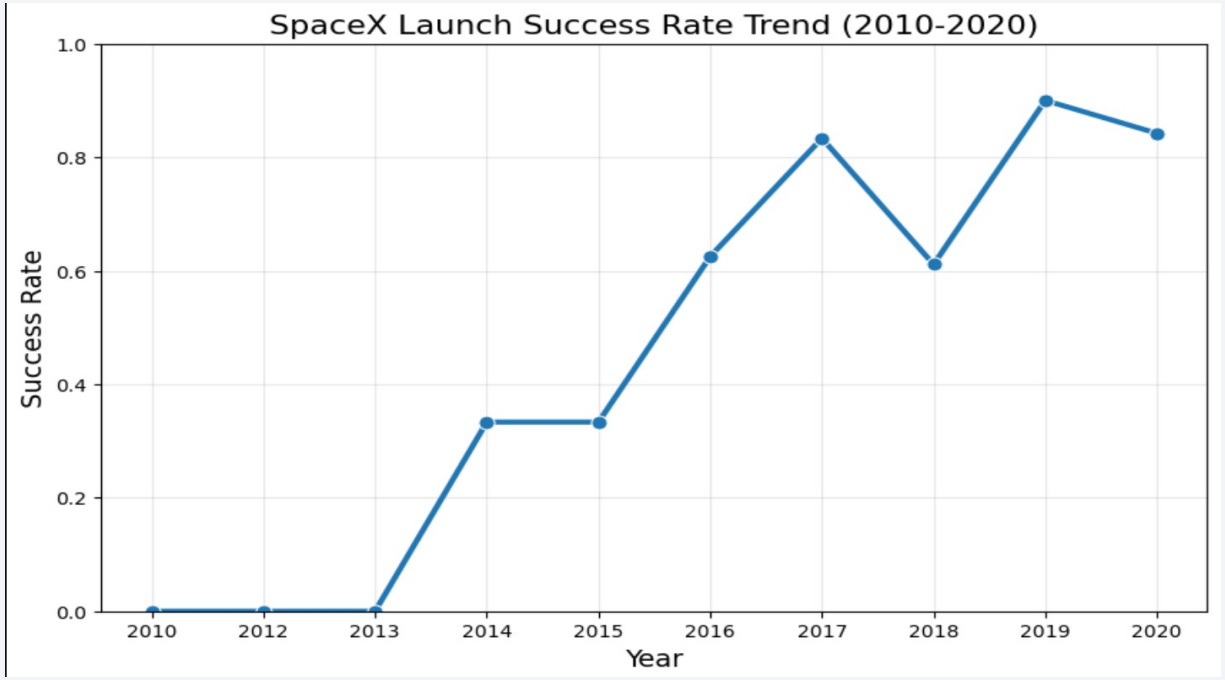
Data Wrangling

- **Data Integration & Validation**
  - Merged datasets from API and web scraping into unified structure
  - Validated data consistency and removed duplicate records
  - Ensured standardized column names and data formats
- **Feature Engineering & Transformation**
  - Converted data types to appropriate formats (datetime, numerical)
  - Created landing outcome labels from outcome column
  - Generated new features for analysis and modeling
  - Normalized numerical features for consistency
- **Exploratory Analysis & Label Creation**
  - Analyzed launch distribution across sites and orbits
  - Calculated occurrence rates for different orbital classes
  - Created training labels for machine learning pipeline
  - Exported processed dataset for subsequent analysis
- **GitHub URL:**https://github.com/nouhaila-elmorjani/IBM-Data-Science-Capstone-SpaceX/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- We 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.
- Github URL:https://github.com/nouhaila-elmorjani/IBM-Data-Science-Capstone-SpaceX/blob/main/edadataviz.ipynb





EDA with SQL

- We loaded the SpaceX dataset into a database directly from Jupyter notebook
- Performed SQL-based exploratory data analysis to extract key insights
- Executed queries to identify:
  - Unique launch sites in the space mission
  - Total payload mass carried by NASA (CRS) boosters
  - Average payload mass for F9 v1.1 booster version
  - Success and failure mission outcome counts
  - Failed drone ship landing outcomes with booster and site details

Github URL:https://github.com/nouhaila-elmorjani/IBM-Data-Science-Capstone-SpaceX/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Created an interactive map visualizing all SpaceX launch sites and their outcomes
- Implemented color-coded markers and clusters to identify success patterns across locations
- Added map features including circles, lines, and popups to display launch details
- Analyzed spatial relationships between launch sites and key infrastructure:
  - Proximity to railways, highways, and coastlines
  - Distance from populated urban areas
  - Geographic distribution of success rates

Github URL:https://github.com/nouhaila-elmorjani/IBM-Data-Science-Capstone-SpaceX/blob/main/lab_jupyter_launch_site_location%20(1).ipynb

Build a Dashboard with Plotly Dash

- Developed a comprehensive web dashboard using Plotly Dash for real-time data exploration
- Implemented interactive visualizations including:
  - Pie charts displaying launch distribution across sites
  - Scatter plots analyzing payload mass vs. launch outcomes
  - Booster version performance comparisons
- Created dynamic filtering capabilities for user-driven analysis
- Enabled real-time data exploration and pattern discovery
- Github URL:https://github.com/nouhaila-elmorjani/IBM-Data-Science-Capstone-SpaceX/blob/main/spacex-dash-app.py

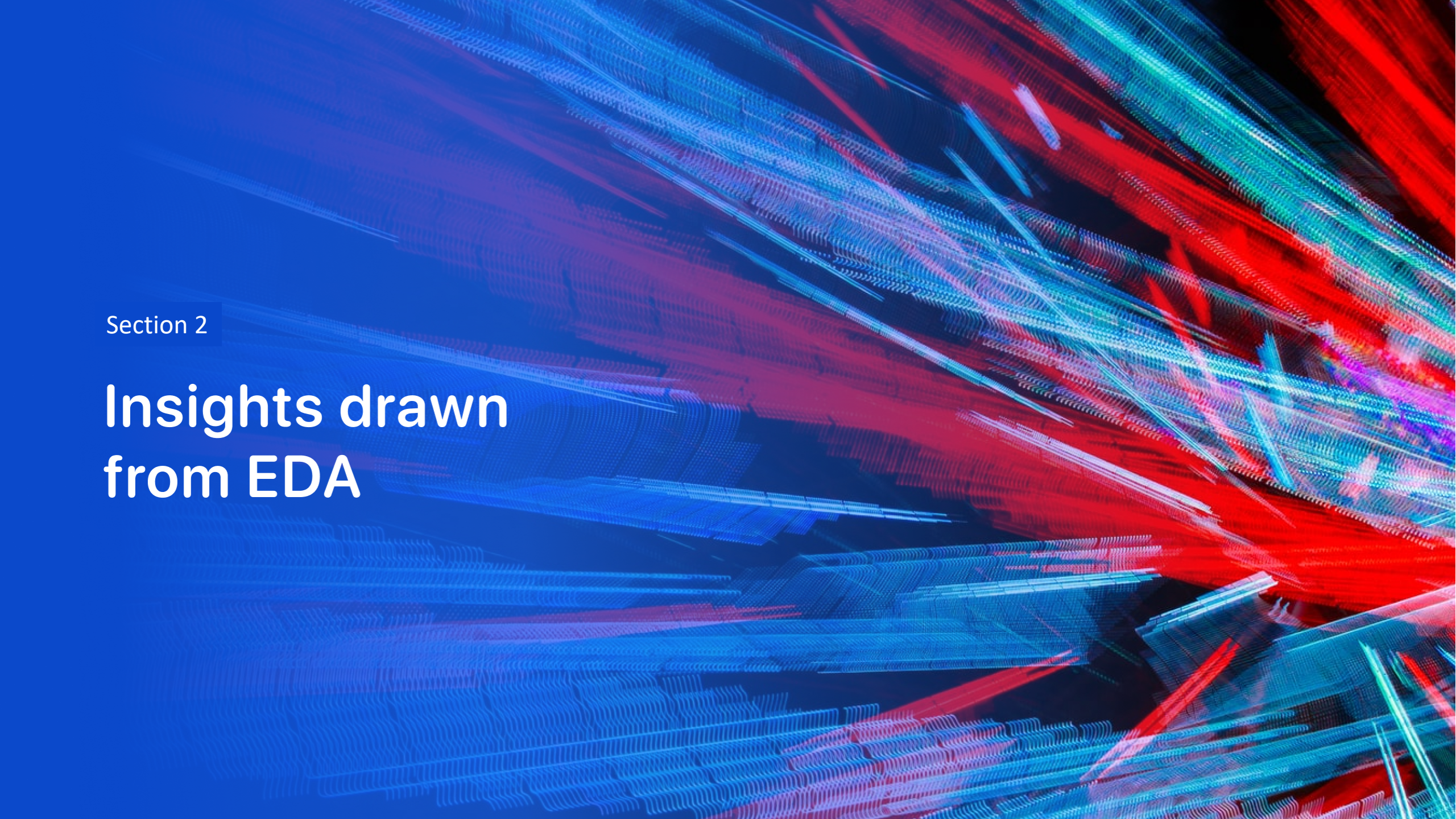
Predictive Analysis (Classification)

- Implemented classification models to predict Falcon 9 landing success
- Loaded and transformed data using numpy and pandas
- Split dataset into training and testing sets for model validation
- Built multiple machine learning models with hyperparameter tuning using GridSearchCV
- Used accuracy as primary evaluation metric
- Optimized performance through feature engineering and algorithm tuning
- Identified best-performing classification model for deployment

Github URL:https://github.com/nouhaila-elmorjani/IBM-Data-Science-CapstoneSpaceX/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

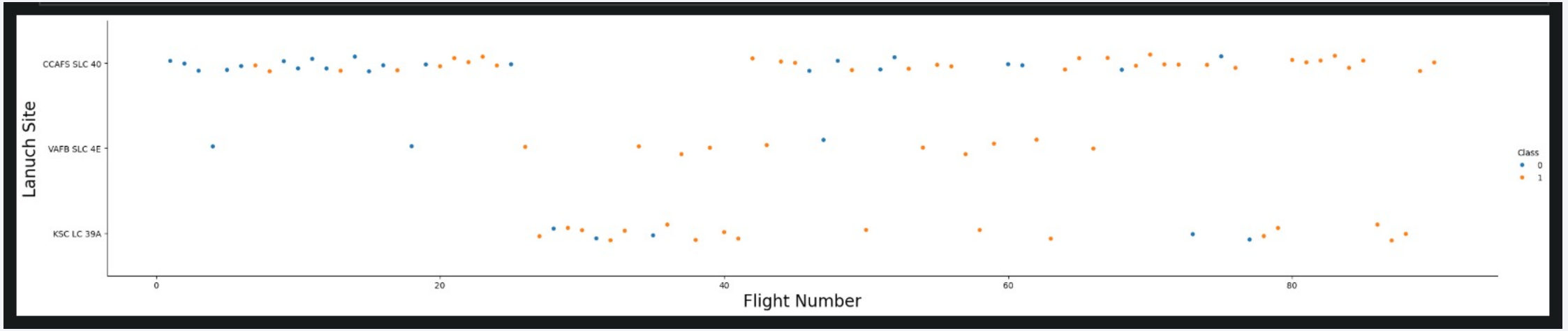
Results

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


Section 2
Insights drawn from EDA

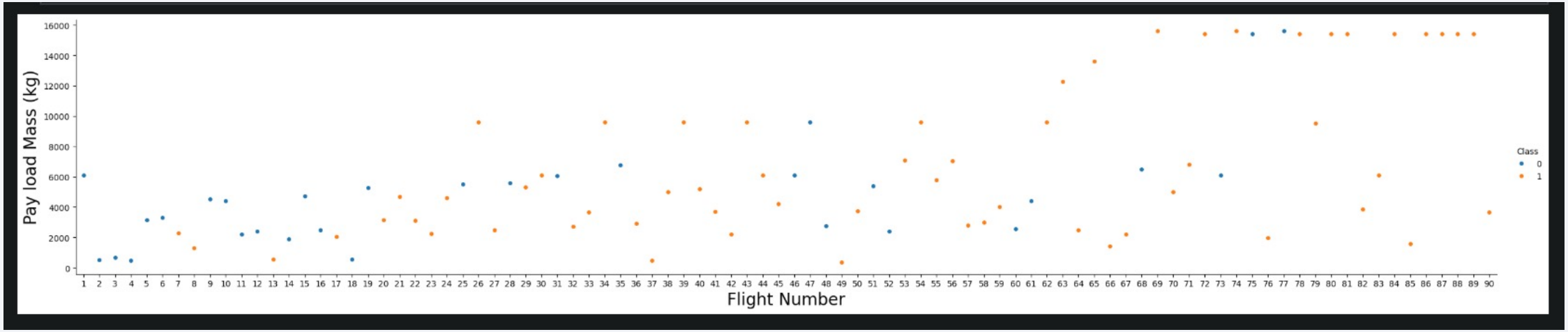
Flight Number vs. Launch Site

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



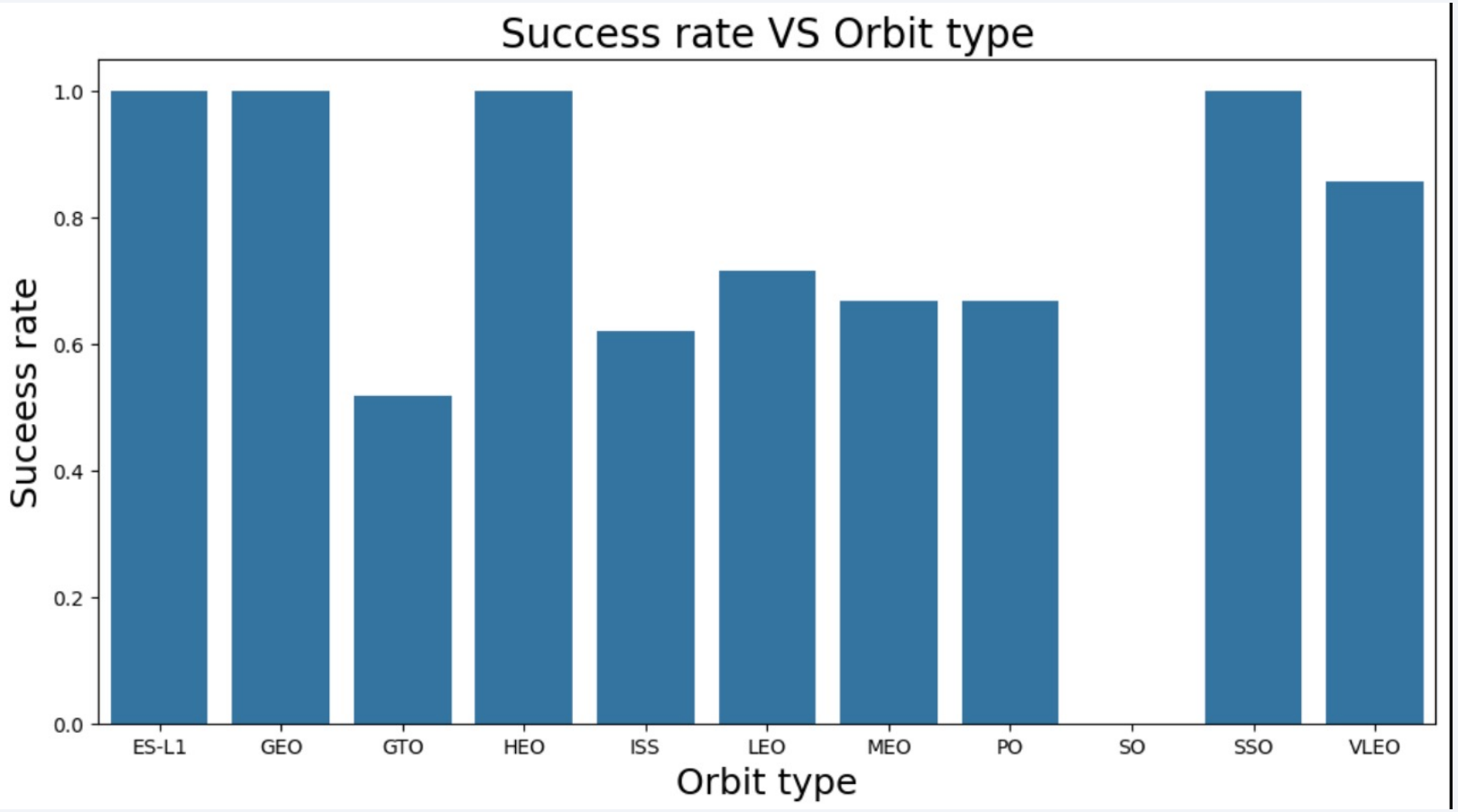
Payload vs. Launch Site

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



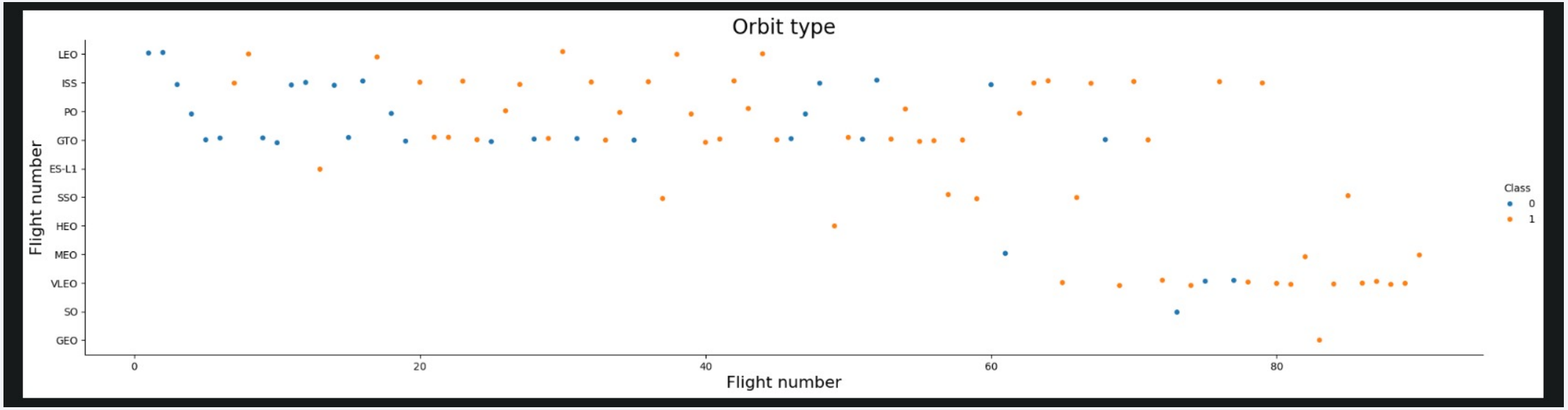
Success Rate vs. Orbit Type

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



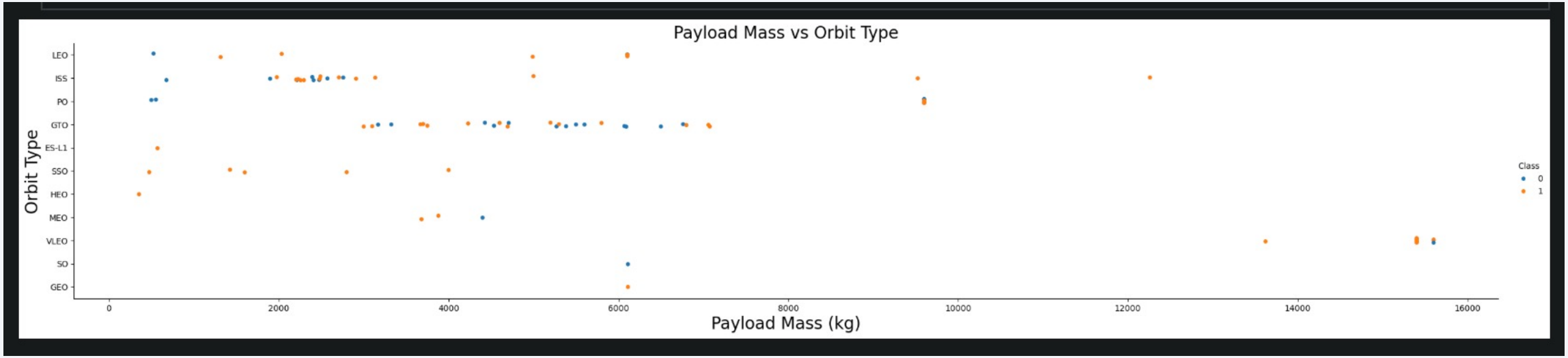
Flight Number vs. Orbit Type

We observe that in 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



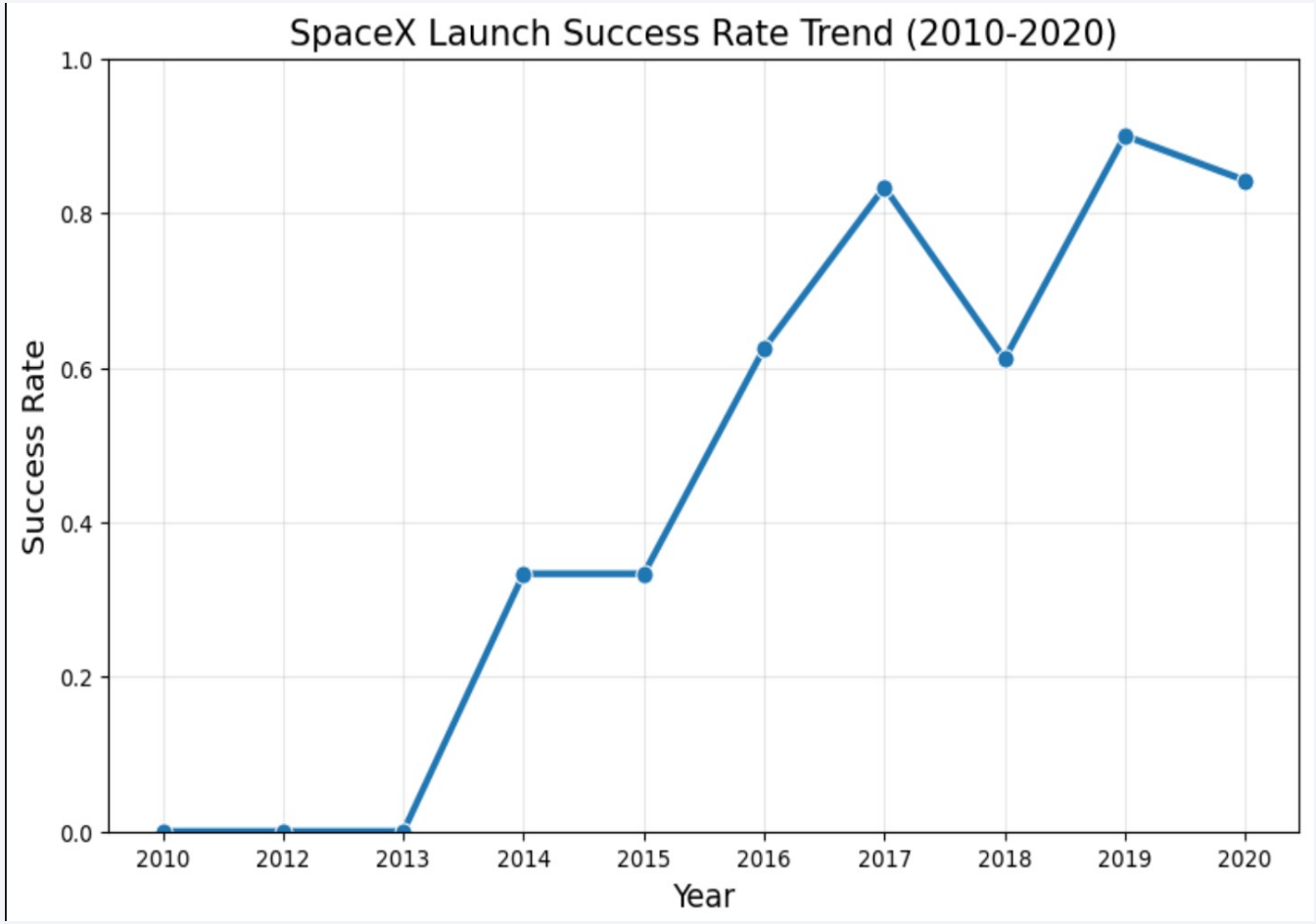
Payload vs. Orbit Type

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 2013 kept on increasing till 2020



All Launch Site Names

Task 1

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

In [54]:
```
%sql select distinct "launch_site" from SPACEXTABLE;
```

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

Out[54]:

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

Launch Site Names Begin with 'CCA'

Task 2

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

```
55]: %sql select * from SPACEXTABLE where "Launch_Site" like 'CCA%' limit 5;
 * sqlite:///my_data1.db
Done.
```

55]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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)	NASA (COTS) NRO	Success	Failure (parachute)
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
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
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

Task 3

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

```
[56]: %sql select SUM("PAYLOAD_MASS_KG__") as "TotalPayloadMass" from SPACEXTABLE where "Customer" like '%NASA (CRS)%';
 * sqlite:///my_data1.db
Done.
```

[56]:

TotalPayloadMass
0.0

Average Payload Mass by F9 v1.1

Task 4

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

```
[57]: %sql select avg("PAYLOAD_MASS__KG_") from SPACEXTABLE where "Booster_Version" like '%F9 v1.1%'

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

[57]:

avg("PAYLOAD_MASS__KG_")
2534.6666666666665

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

```
[58]: %sql select min("Date") from SPACEXTABLE where "Landing_Outcome"='Success (ground pad)'
```

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

[58]:

min("Date")
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
[59]: %%sql select booster_version from SPACEXTABLE where (mission_outcome like 'Success')
      and (payload_mass__kg_  between  4000 and 6000) and (landing_outcome like 'Success (drone ship)');
```

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

[59]:

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

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

[64]:
```
%%sql select "Mission_Outcome" , count(*)
from  SPACEXTABLE
where "Mission_Outcome" in ('Success','Failure (in flight)')
group by "Mission_Outcome";
```

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

[64]:

Mission_Outcome	count(*)
Failure (in flight)	1
Success	98

Boosters Carried Maximum Payload

Task 8

List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.

[65]:
```
%%sql select "Booster_Version" ,"PAYLOAD_MASS__KG_"
from SPACEXTABLE
where "PAYLOAD_MASS__KG_" =(select max(PAYLOAD_MASS__KG_) from SPACEXTABLE )
```

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

[65]:

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

Task 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.

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.

[67]:
```
%%sql SELECT
    CASE substr("Date", 6, 2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END as Month_Name,
    "Booster_Version",
    "Launch_Site",
    "Landing_Outcome"
FROM SPACEXTABLE
WHERE "Landing_Outcome" LIKE '%Failure%drone ship%'
AND substr("Date", 1, 4) = '2015';
```

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

[67]:

Month_Name	Booster_Version	Launch_Site	Landing_Outcome
January	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
April	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

Task 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.

[68]:
```
%%sql SELECT "Landing_Outcome", COUNT(*) as Outcome_Count
FROM SPACEXTABLE
WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20'
AND "Landing_Outcome" IS NOT NULL
AND "Landing_Outcome" != ''
GROUP BY "Landing_Outcome"
ORDER BY Outcome_Count DESC;
```

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

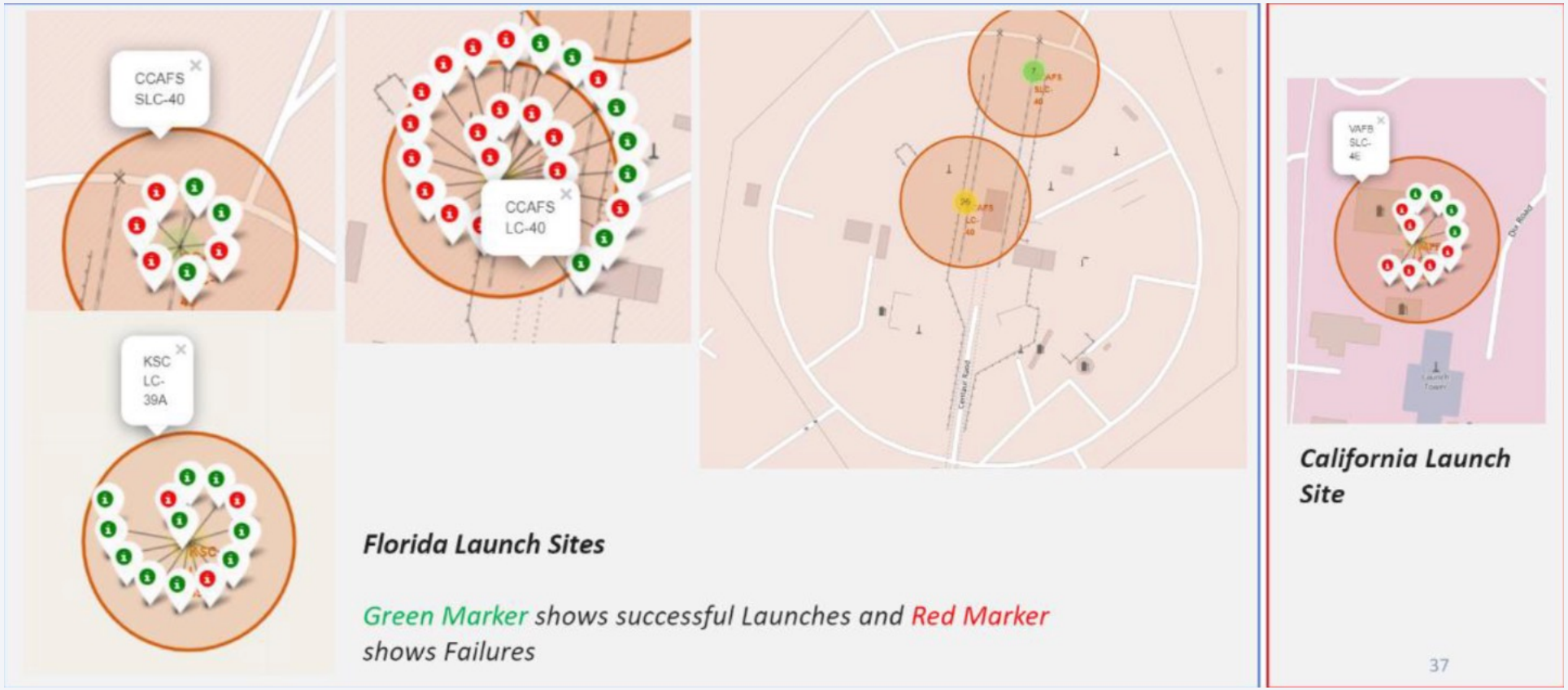
[68]:

Landing_Outcome	Outcome_Count
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

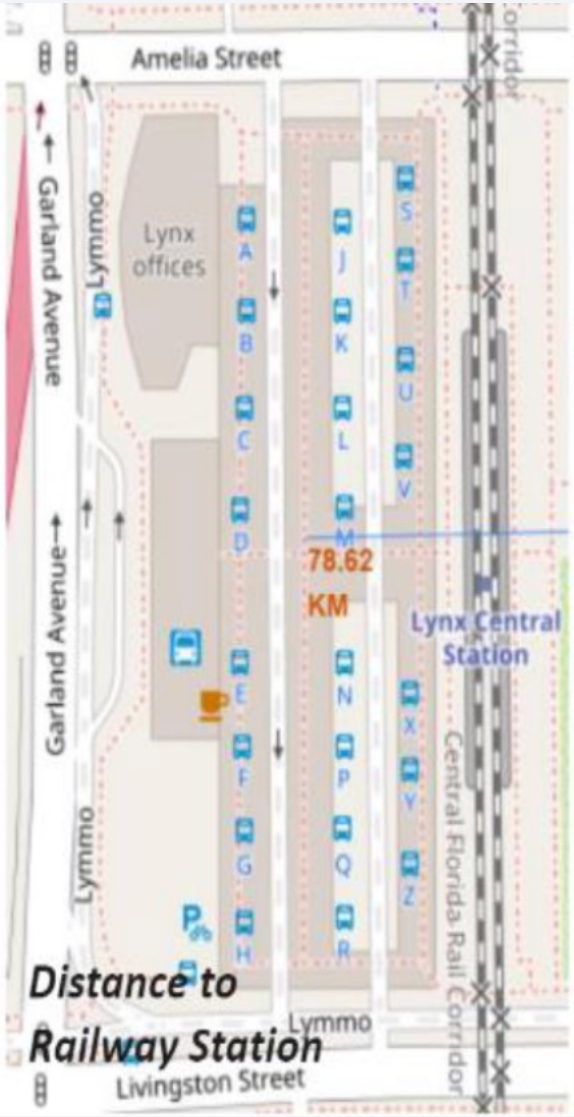

Section 3
Launch Sites
Proximities Analysis

Task 1:Mark all lunch sites on a map

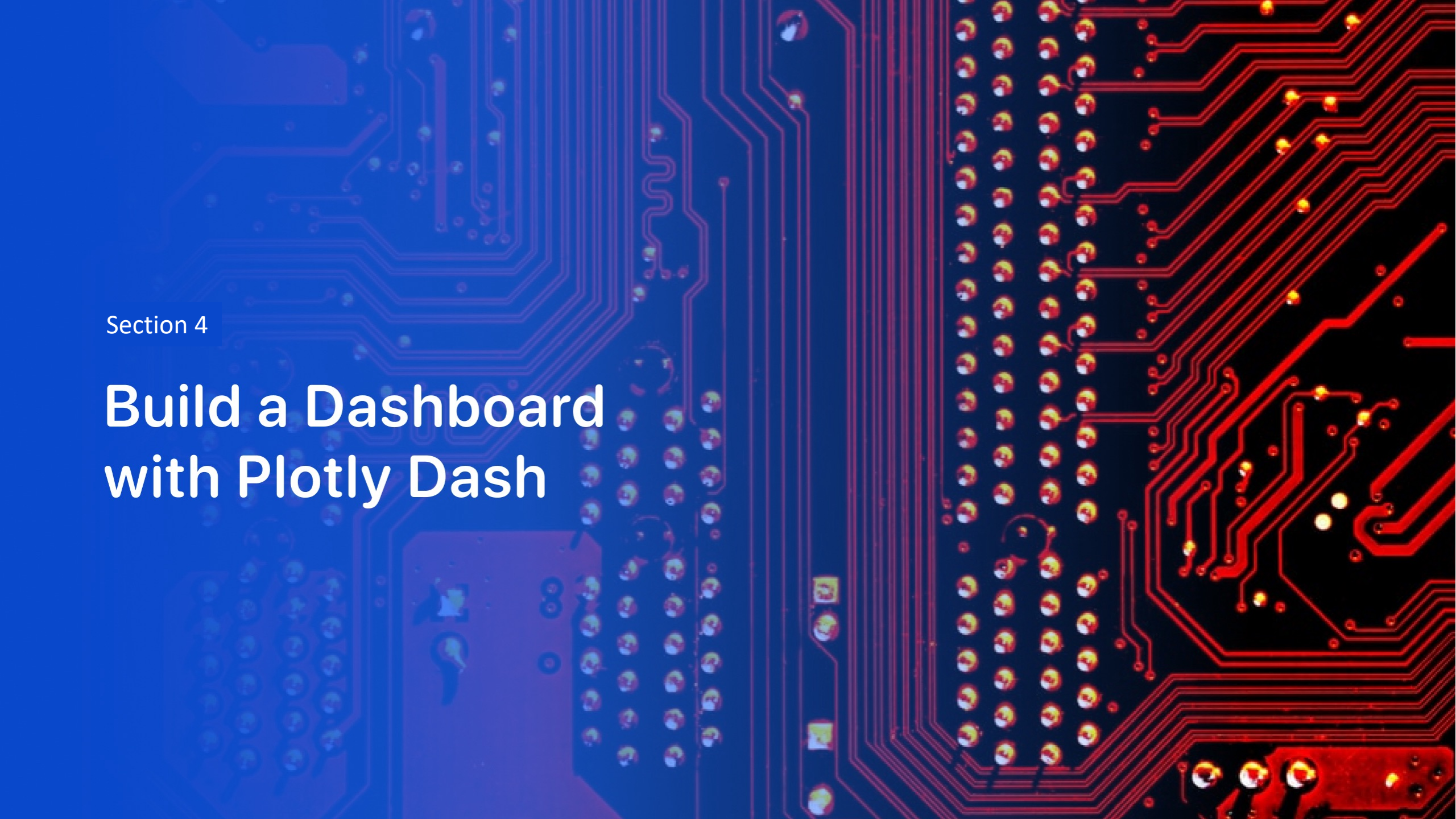
Task 2: Mark the success/failed launches for each site on the map



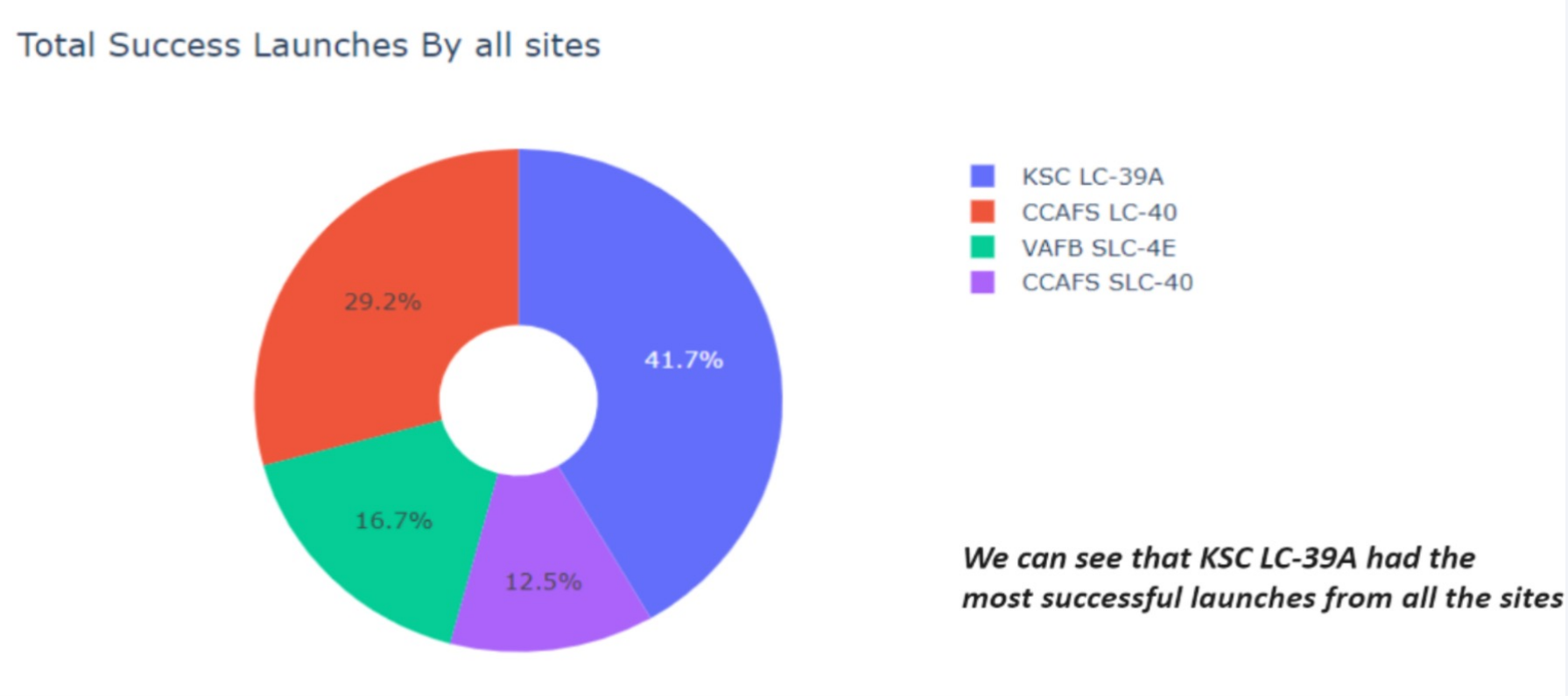
Task 3: Calculate the distances between a launch site to its proximites



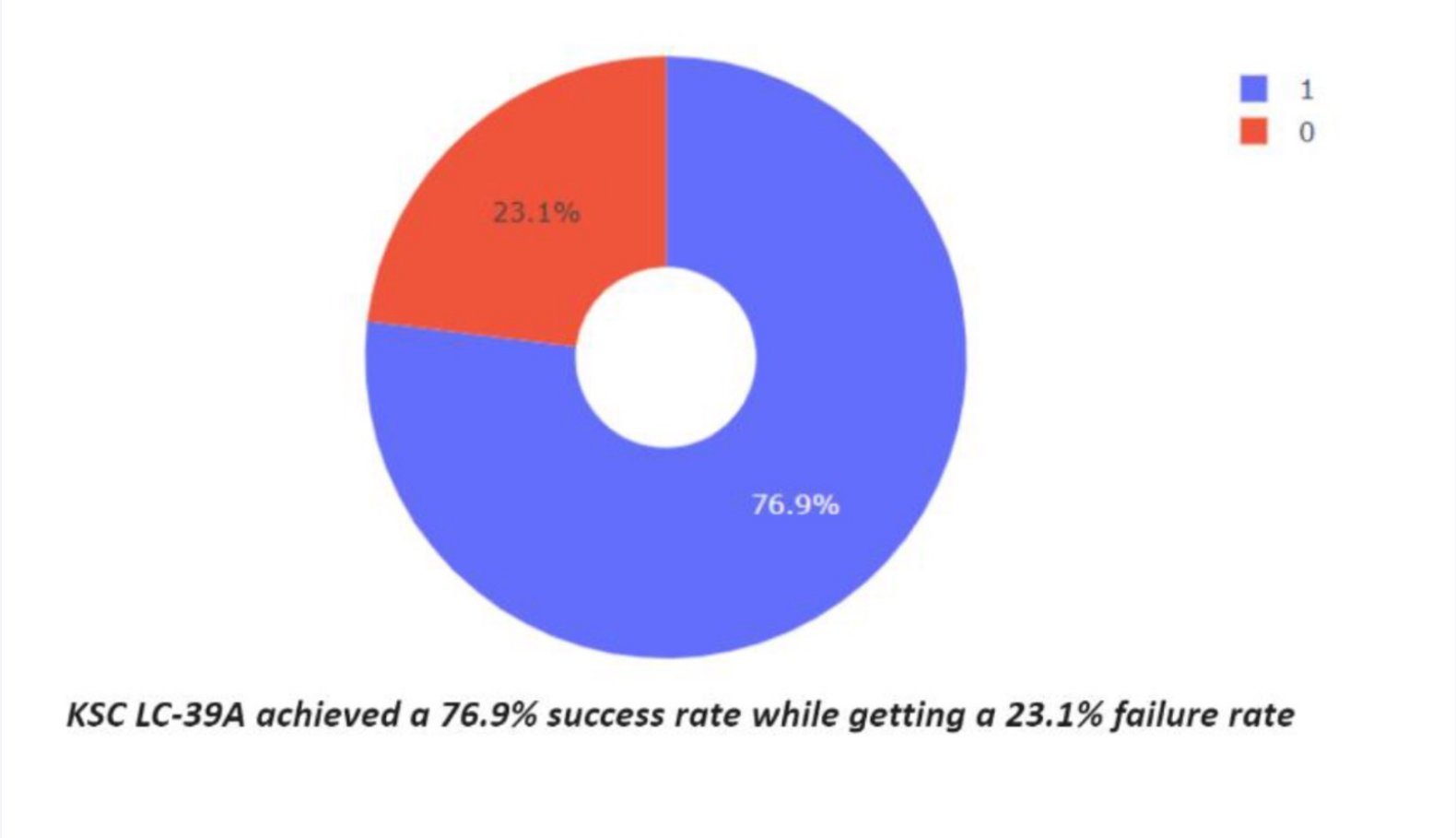
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes


Section 4
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



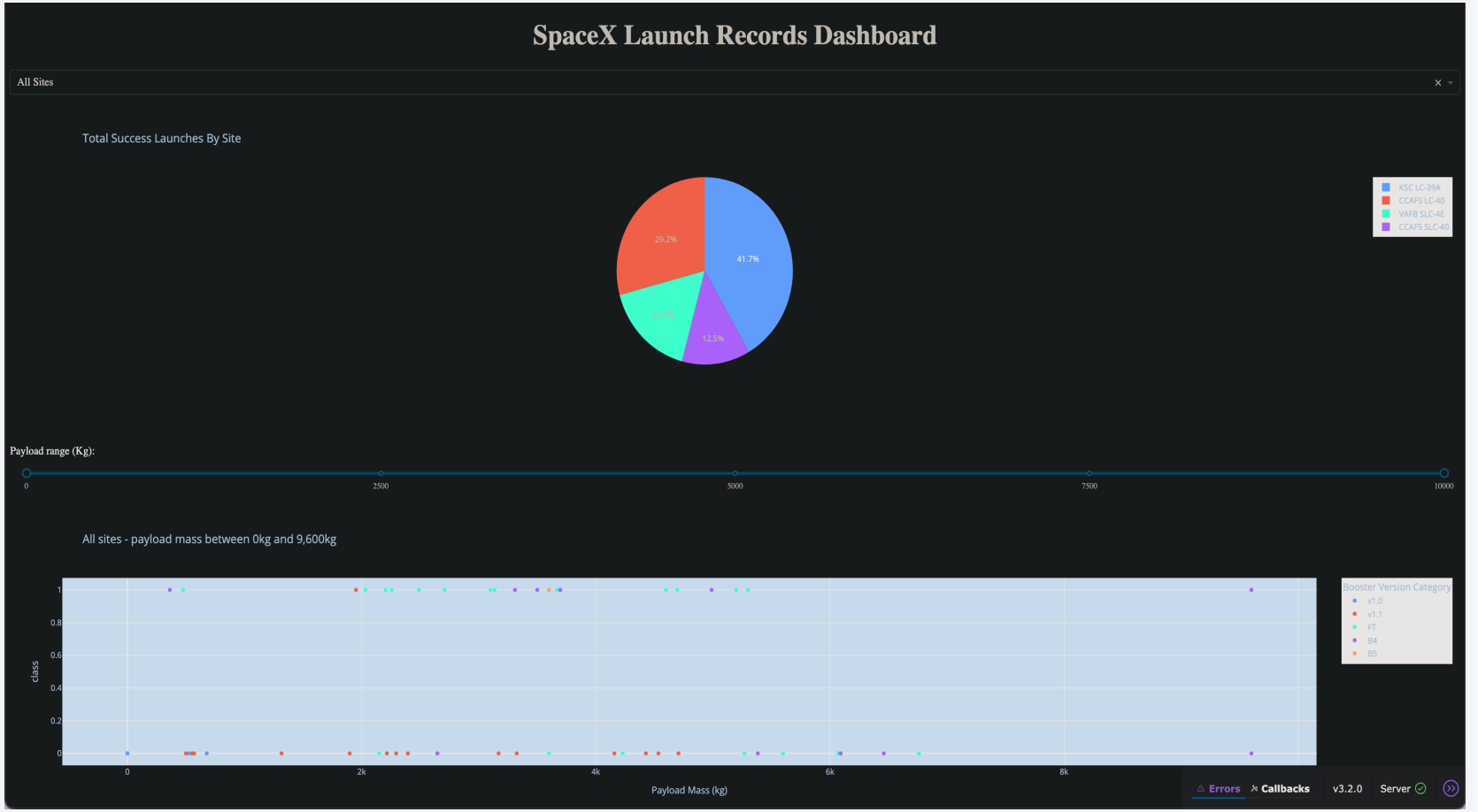
Pie chart showing the Launch site with the highest launch success ratio



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

Key Performance Insights and Sites Reliability analysis

- **Launch Site Performance:** CCAFS LC-40 leads with 41.7% success rate, establishing it as the most reliable launch location
- **Booster Version Analysis:** FT booster demonstrates consistent performance across various payload masses with high success frequency
- **Payload-Outcome Relationship:** No clear correlation between higher payload mass and lower success rates across booster versions
- **Comparative Performance:** Dashboard reveals significant variability in success rates across different launch facilities
- **Data-Driven Decisions:** Insights support strategic planning for future launch site selection and booster configuration



Section 5

Predictive Analysis (Classification)

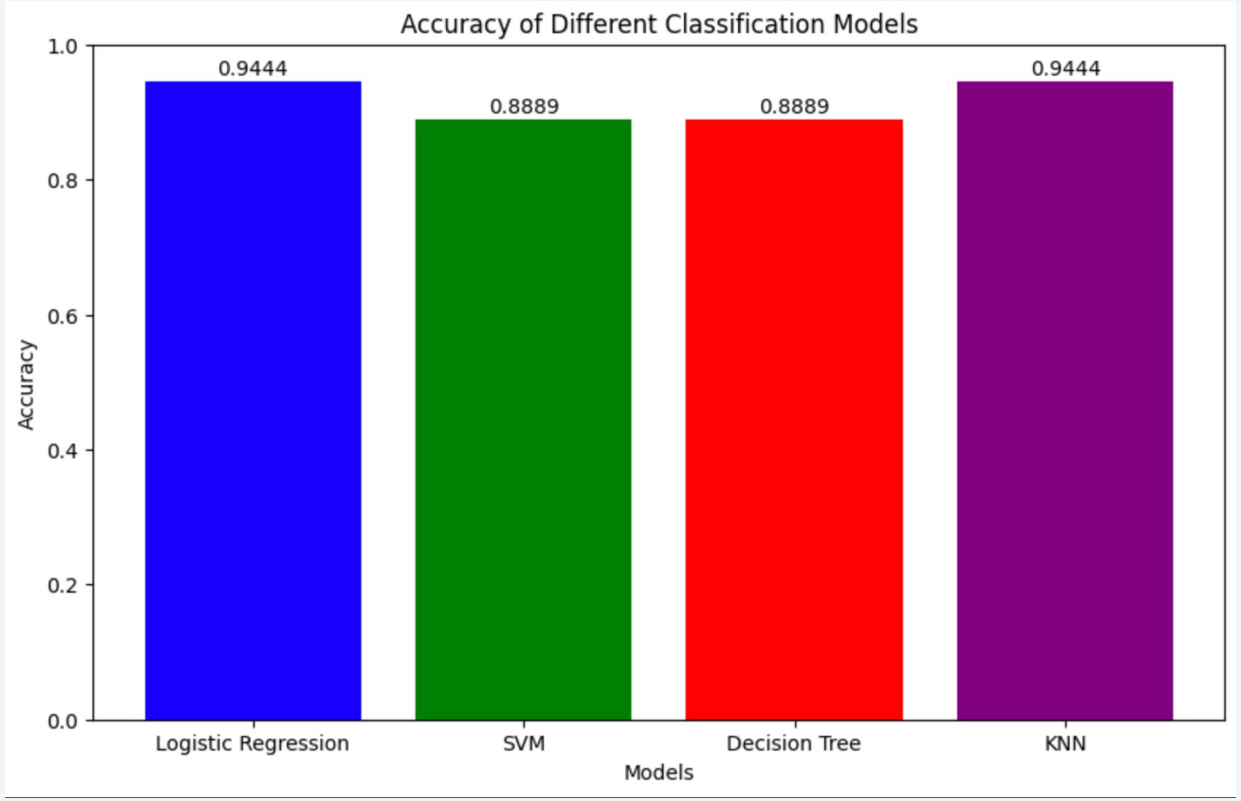
Classification Accuracy

Model Accuracy Comparison:

- **Decision Tree**: 0.9444 (94.44%)
- **KNN**: 0.9444 (94.44%)
- **Logistic Regression**: 0.8333 (83.33%)
- **SVM**: 0.8333 (83.33%)

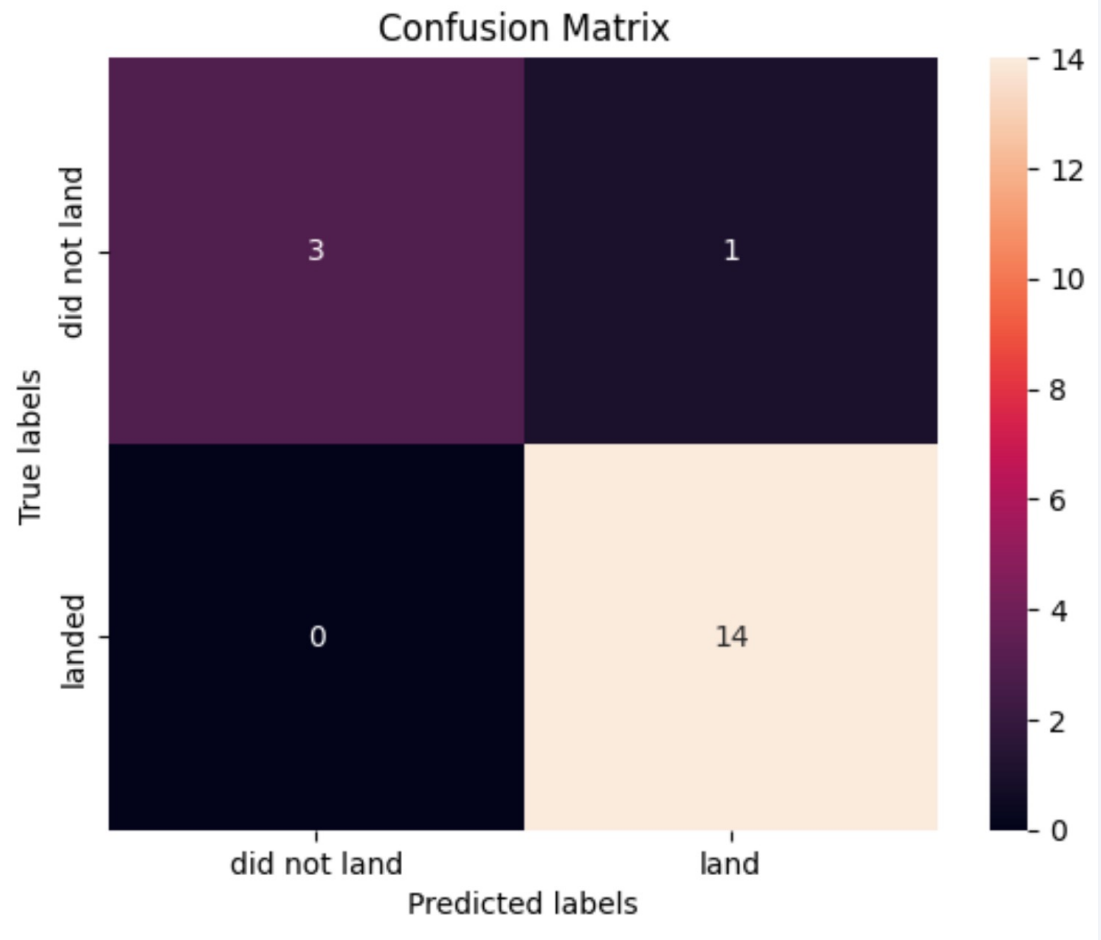
The Decision Tree and KNN models achieved the highest classification accuracy

- Both top-performing models reached 94.44% accuracy on test data
- Decision Tree selected as optimal model for deployment due to interpretability



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

- **Point 1:** Launch site CCAFS LC-40 has the highest success rate at 41.7%, indicating optimal launch conditions and operational efficiency at this location.
- **Point 2:** The FT booster version demonstrates consistent high performance across various payload masses, confirming its reliability for successful missions.
- **Point 3:** Both Decision Tree and KNN classifiers achieved the highest accuracy of 94.44% for predicting landing success, making them the optimal machine learning models for this task.
- **Point 4:** Launch success rates show significant improvement from 2013-2020, reflecting SpaceX's continuous technological advancements and process optimization.

Thank you!