IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

A. M. ABEER HASAN

19th October, 2024

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- Summary of Methodologies
  - ➢Data Collection
  - ➢Data Wrangling
  - ➢Exploratory Data Analysis
  - ➢Interactive Visual Analytics using Folium and Plotly Dash
  - ➢Predictive Analysis Using Classification Models

Executive Summary

- Summary of All Results

a. No correlation between success rate and Payload Mass or Flight Numbers.

b. Success rate increasing from 2013 onwards

c. The average payload mass carried by booster version F9 v1.1

d. Booster version FT has the highest success rate

e. Launch sites keep safe distance from cities but in very close proximities to coast line and railway.

f. The range between 2000kgs and 5000kgs of Payload Mass has the highest proportion of success rate.

g. Support Vector Machine(SVM) model has the highest accuracy(88.88%) among all other predictive models.

Introduction

- This project deals with datasets of SpaceX REST API. SpaceX is the pioneering spacecraft manufacturer and expeditors nowadays. I explored insights from SpaceX database to make it succeed in launching carriers better and determine which factors affect the most in launching and landing success rate.
- A machine learning model is built to explore further insights and generate outcomes efficiently by own algorithm.
- Further,a dashboard is created for interactive analytics to present the dynamic data to the policymakers of SpaceX.
- I ran predictive analysis for the machine learning models and adopted the model of best accuracy.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Calling SpaceX REST API
  - Web scraping
- Perform data wrangling
  - Using Pandas and NumPy libraries
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Built Logistic Regression, SVM, DecisionTree , kNN models
  - Split the data into training and test sets and fit them into the model
  - Evaluated the accuracy of the models

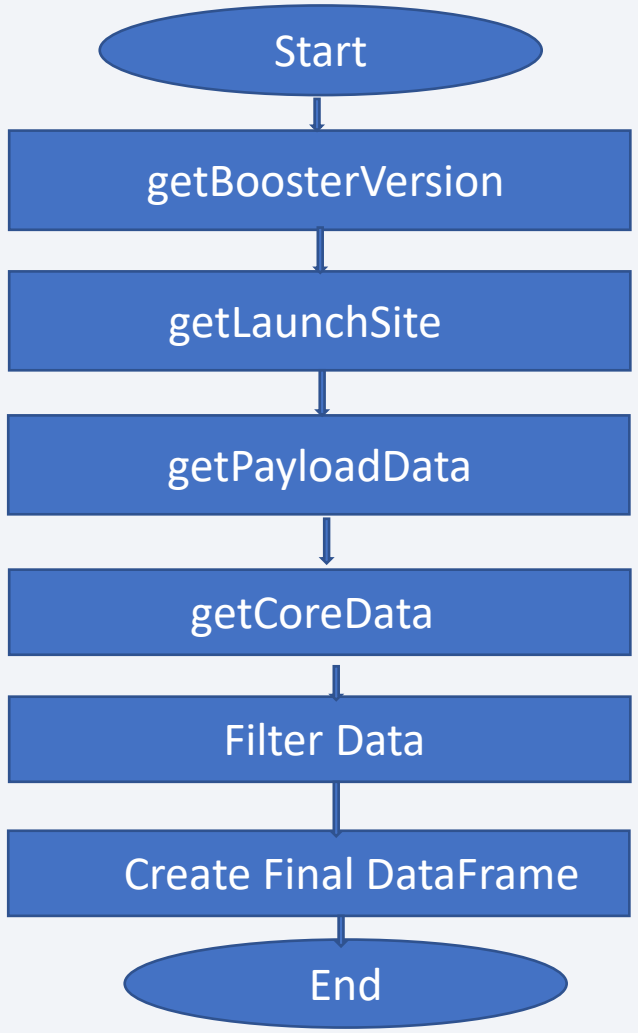
Data Collection

- Datasets were collected from

  - SpaceX REST API
  - Web Scraping

Data Collection – SpaceX API

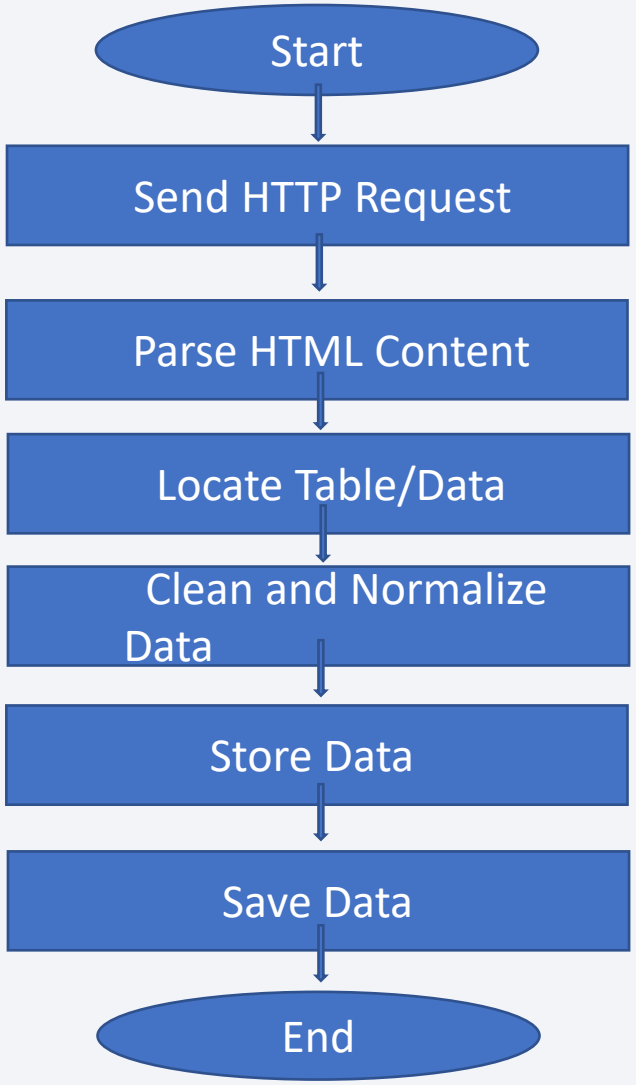
- REST API calls
  - getBoosterVersion
  - getLaunchSite
  - getPayloadData
  - getCoreData
- SpaceX API Notebook GitHub URL: DataScienceEcosystem/jupyter-labs-spacex-data-collection-api.ipynb at main · abeerhasan364/DataScienceEcosystem



Data Collection - Scraping

- Send HTTP Request: Use requests to send an HTTP GET request to the website.
- Parse HTML Content: Use BeautifulSoup to parse the received HTML content.
- Locate Table/Data: Identify the specific HTML elements (tables, rows) that contain the desired data.
- Extract Data: Extract relevant data (text, attributes) from HTML elements.

- SpaceX Web Scraping GitHub URL :[DataScienceEcosystem/jupyter-labs-webscraping.ipynb at main · abeerhasan364/DataScienceEcosystem]



Data Wrangling

- **Load Data**: Load the dataset from a source (e.g., CSV, database).
- **Inspect Data**: Check the data structure using .head(), .info(), .describe().
- **Check for Missing Values**: Use .isnull()
- **Handle Missing Data**: Fill, remove, or interpolate missing data as required.
- **Filter/Select Relevant Data**: Filter rows/columns based on criteria (e.g., remove outliers or irrelevant columns).
- **Transform/Format Data**: Convert data types, normalize, or reformat columns.
- **Classify Data**: Create new columns or modify existing columns (e.g., creating a classification column).
- **Group/Sort Data**: Organize the data using grouping or sorting operations.
- **Export Data**: Save the cleaned dataset to a CSV, database, or other format.

- Data Wrangling Notebook GitHub URL: DataScienceEcosystem/labs-jupyter-spacex-Data wrangling.ipynb at main · abeerhasan364/DataScienceEcosystem

EDA with Data Visualization

- Charts plotted

i. Flight Number vs. Launch Site : To show if there is any correlation between these two variables and with launching success rate
ii. Payload vs. Launch Site : To show if there is any correlation between these two variables and with launching success rate
iii. Success Rate vs. Orbit Type : To show if there is any correlation between these two variables
iv. Flight Number vs. Orbit Type : To show if there is any correlation between these two variables and with success rate
v. Launch Success Yearly Trend : To show the yearly trend of launching success rate

- EDA with data visualization notebook GitHub URL :
  DataScienceEcosystem/edadataviz.ipynb at main · abeerhasan364/DataScienceEcosystem

EDA with SQL

- Drop Table if Exists: Dropped the existing SPACEXTABL table if it already exists to avoid conflicts when creating a new table.
- Create Table with Non-Null Dates: Created a new table SPACEXTABLE by selecting records from SPACEXTBL where the Date field is not null.

- EDA with SQL notebook GitHub URL: DataScienceEcosystem/jupyter-labs-eda-sql-coursera_sqllite.ipynb at main · abeerhasan364/DataScienceEcosystem

Build an Interactive Map with Folium

- Markers:
  - Coastline Marker
  - Launch Site Markers
  - City, Railway, Highway Markers
- Circle Markers:
  - NASA Johnson Space Center
  - Launch Site Circles
- Polylines:
  - Launch Site to Coastline Line
  - Lines to City, Railway, and Highway
- Marker Cluster:

Build an Interactive Map with Folium(Continued)

- **Explanation of Why These Objects Were Added:**
- **Markers**: To clearly display the geographic locations of the SpaceX launch sites and surrounding landmarks such as coastlines, cities, highways, and railways.
- **Circles**: To emphasize certain locations, particularly NASA Johnson Space Center and the SpaceX launch sites, making them stand out on the map.
- **Lines**: To visually connect launch sites to important nearby locations (e.g., coastline, cities), providing insight into distances and geographic relationships.
- **Marker Cluster**: To ensure the map remains easy to read by grouping markers together when zoomed out, avoiding overlap or clutter.

- Folium Notebook GitHub URL: DataScienceEcosystem/DV0101EN-Exercise-Generating-Maps-in-Python (1).ipynb at main · abeerhasan364/DataScienceEcosystem

Build a Dashboard with Plotly Dash

- **Plots:**
  - **Pie Chart:** This chart shows the distribution of successful and failed launches, either for all launch sites combined or for a specific launch site selected from a dropdown menu.
  - **Scatter Chart:** This chart visualizes the relationship between payload mass (on the x-axis) and launch success (on the y-axis), colored by the booster version category. It allows users to explore if there's a correlation between payload weight and launch outcome.
- **Interactions:**
  - **Dropdown Menu:** Users can select a specific launch site ("All Sites" is the default option) to filter the data and see the pie chart and scatter chart focused on that particular launch location.
  - **Range Slider:** Users can adjust the range slider to filter the data by payload mass, allowing them to focus on launches within a specific weight range and see how it affects the scatter chart.

    - Dashboard Notebook GitHub URL: DataScienceEcosystem/spacex_dash_app.py at main · abeerhasan364/DataScienceEcosystem

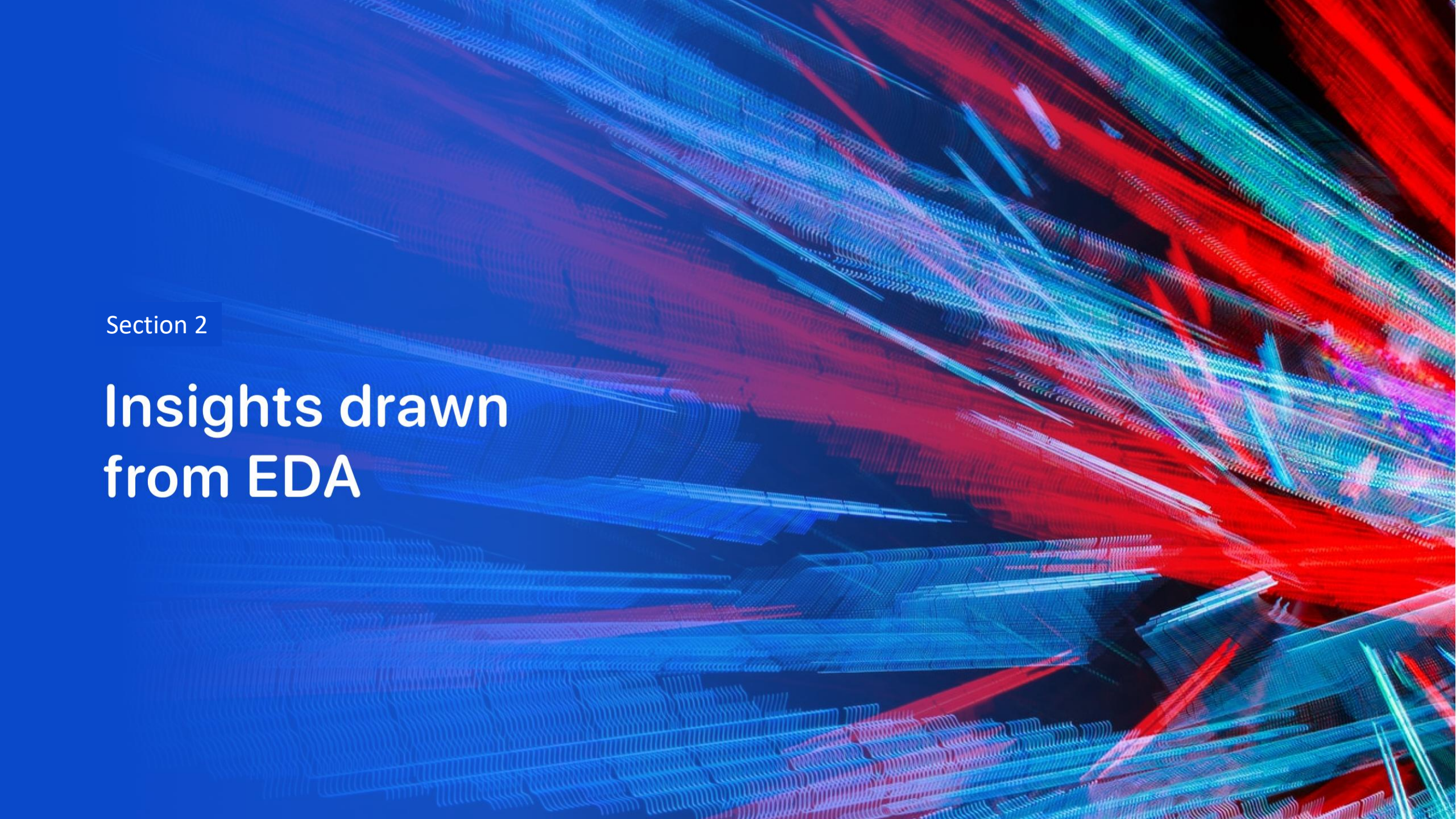
Predictive Analysis (Classification)

- Imports libraries like requests, pandas, preprocessing, train_test_split, GridSearchCV from scikit-learn, etc.
- Loads two datasets from URLs using requests and pandas.
- Splits the data into training and testing sets using train_test_split.
- Defines hyperparameter grids for three machine learning models: Logistic Regression, Support Vector Machine (SVM), and Decision Tree Classifier.
- Uses GridSearchCV to perform grid search on each model to find the best hyperparameters based on accuracy.
- Trains the models with the best hyperparameters.
- Evaluates the models on the testing set using the score method.
- Generates predictions for the testing set using the trained models.

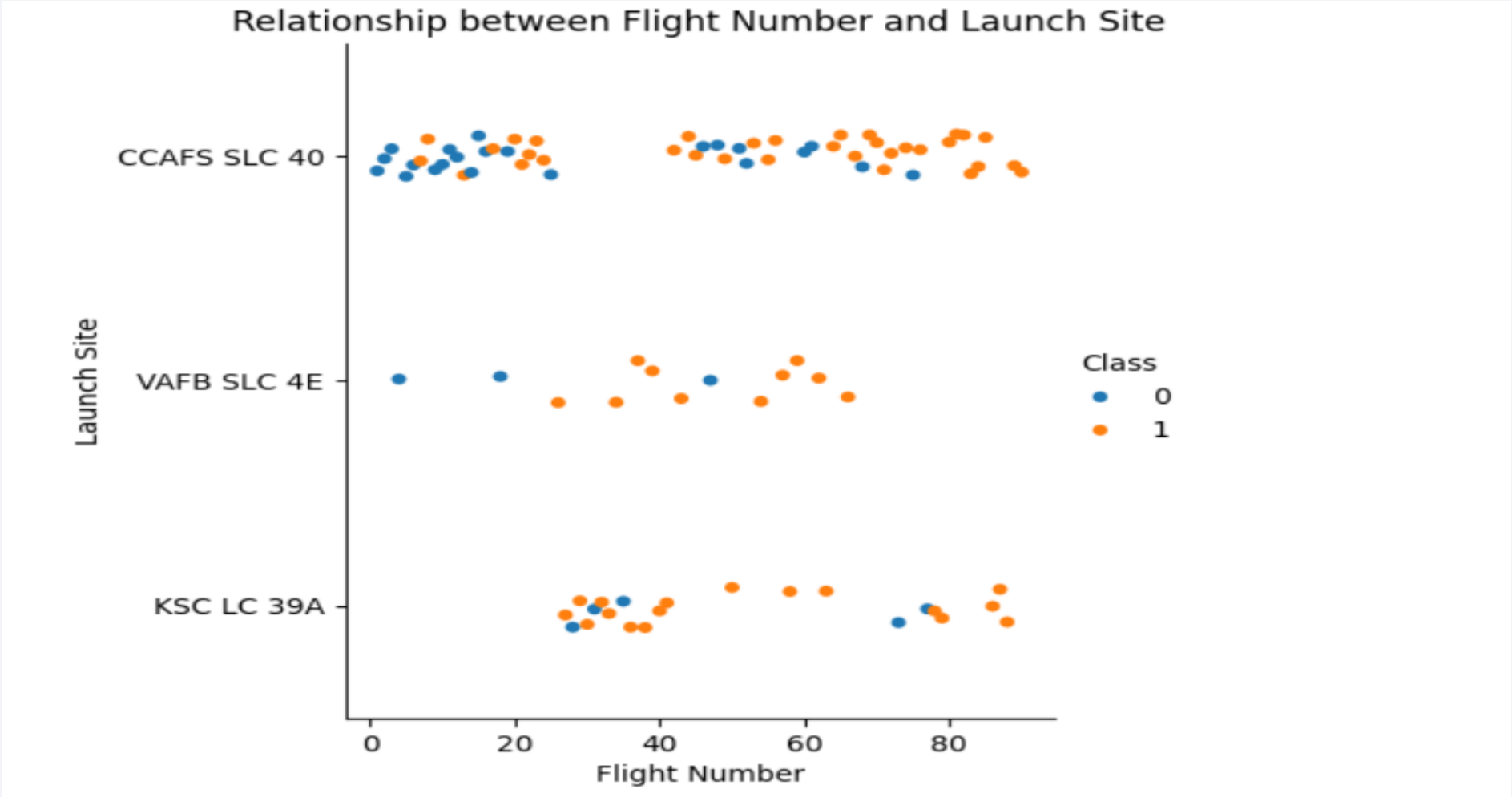
Predictive Analysis (Classification)- Continued

•Creates confusion matrices for each model's predictions using the plot_confusion_matrix function.
•Defines a new hyperparameter grid for the K-Nearest Neighbors (KNN) model.
•Performs grid search on KNN to find the best hyperparameters.
•the accuracy scores of all four models (Logistic Regression, SVM, Decision Tree, KNN) using a bar chart
•Trains the KNN model with the best hyperparameters.
•Evaluates the KNN model on the testing set.
•Generates predictions for the testing set using the KNN model.
•Creates a confusion matrix for KNN predictions using the plot_confusion_matrix function.
•Compares

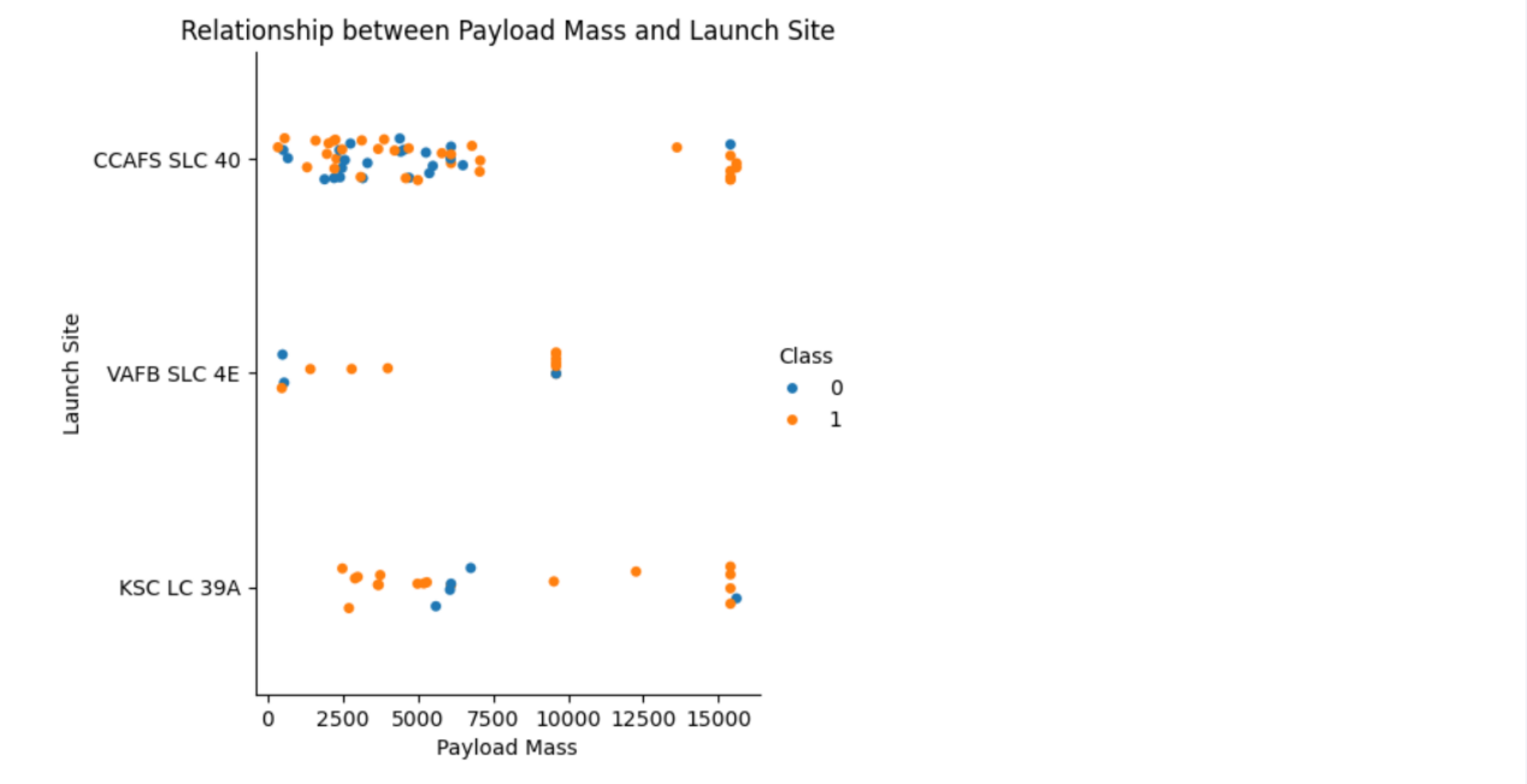
•Predictive Analysis Notebook GitHub URL: DataScienceEcosystem/SpaceX_Machine Learning Prediction_Part_5 (1).ipynb at main · abeerhasan364/DataScienceEcosystem


Section 2
Insights drawn from EDA

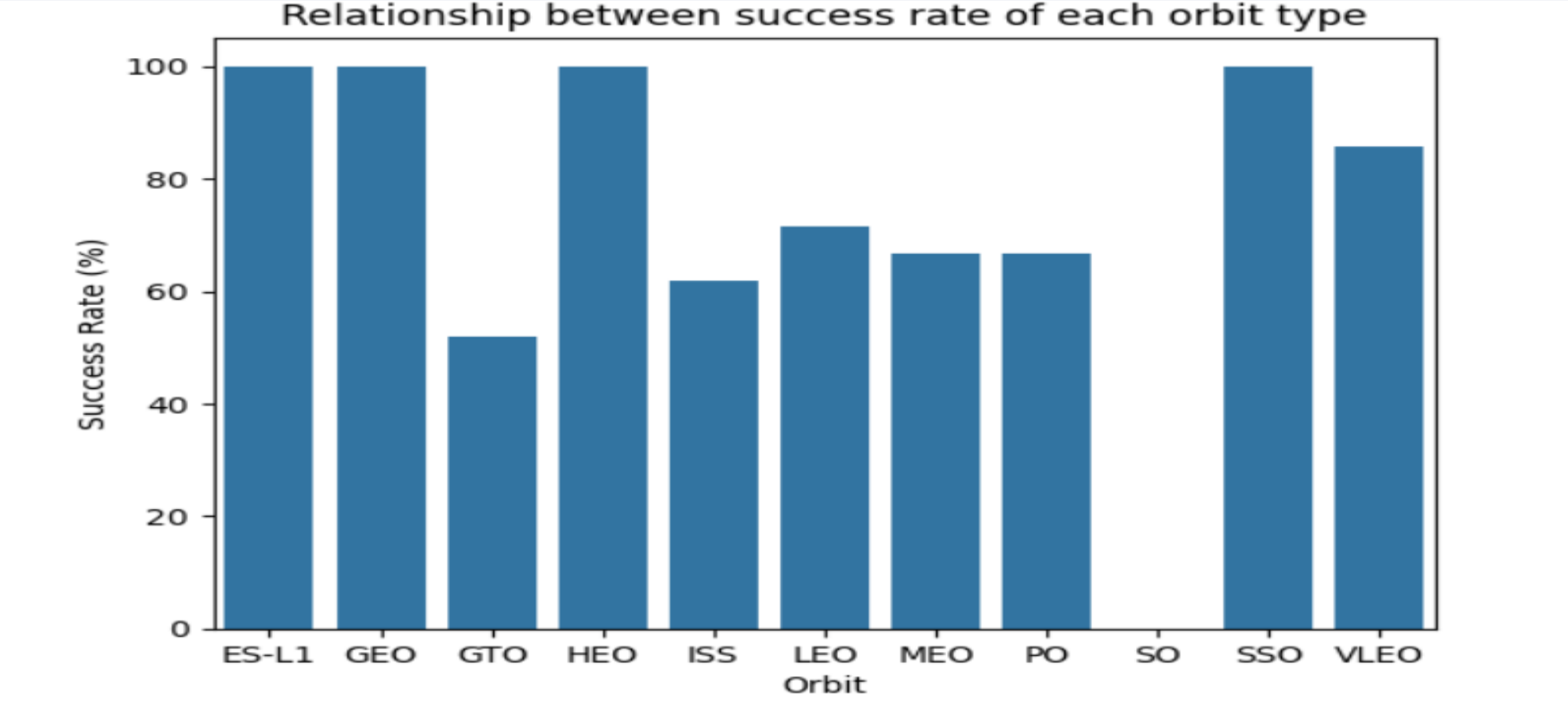
Flight Number vs. Launch Site



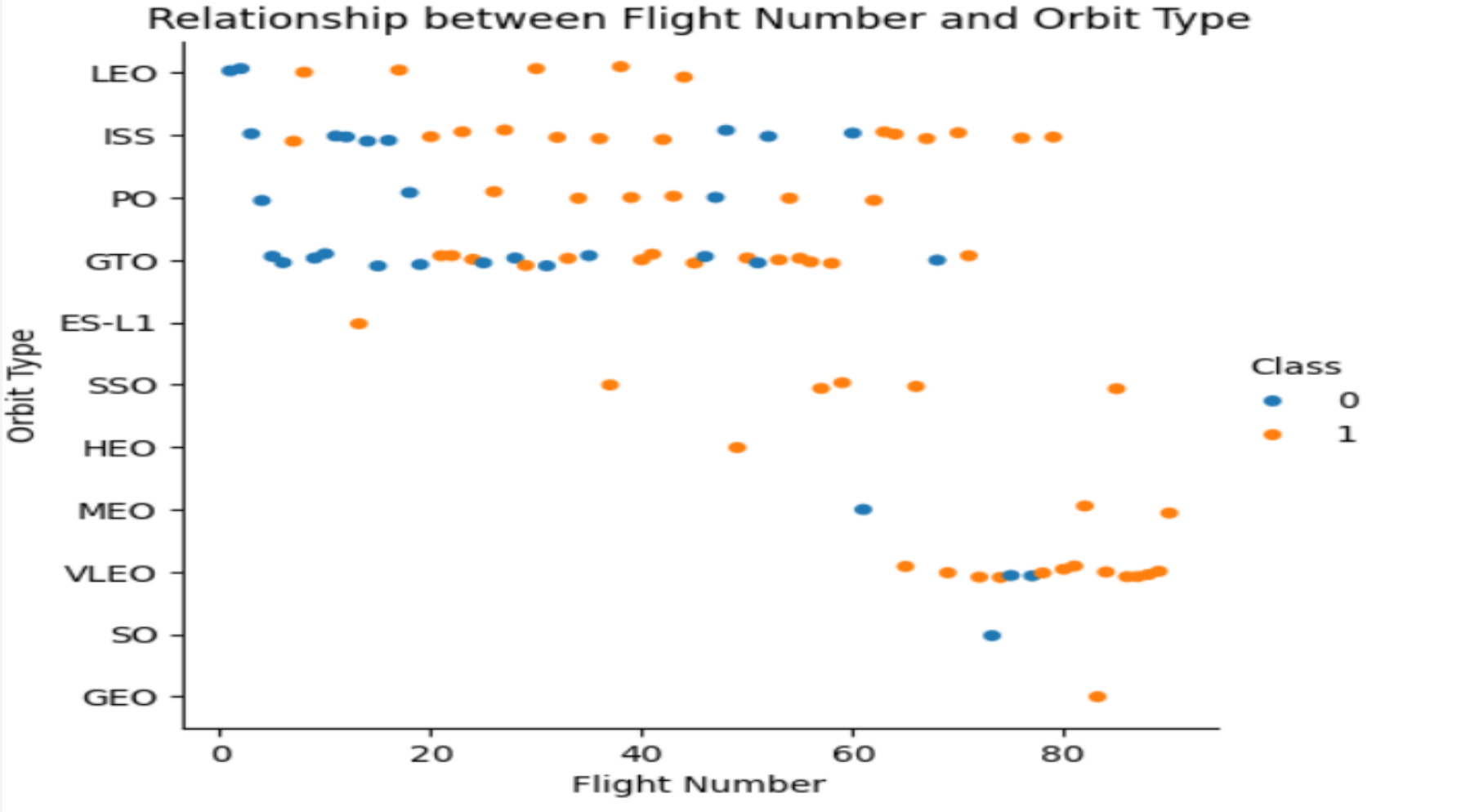
Payload vs. Launch Site



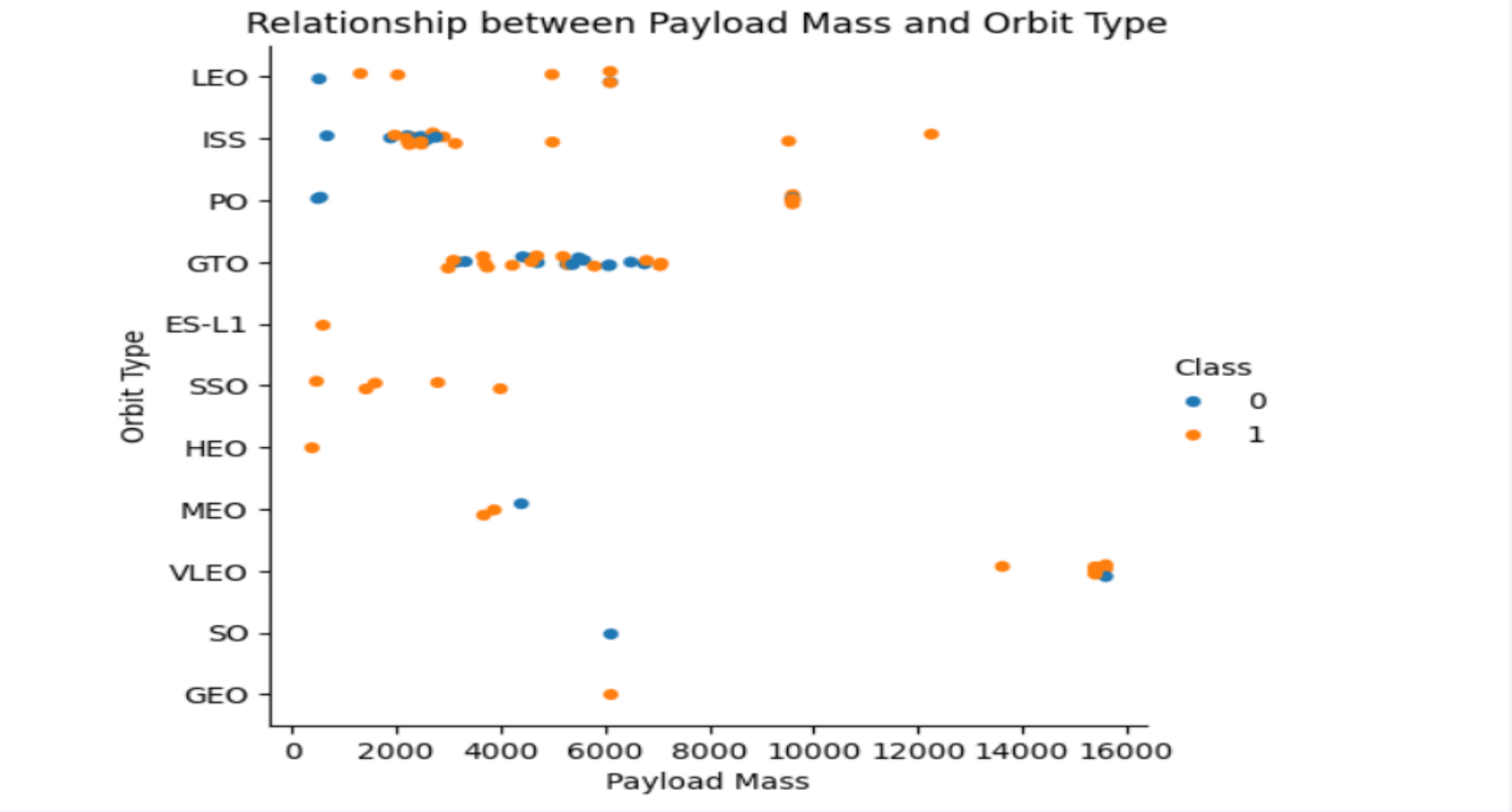
Success Rate vs. Orbit Type



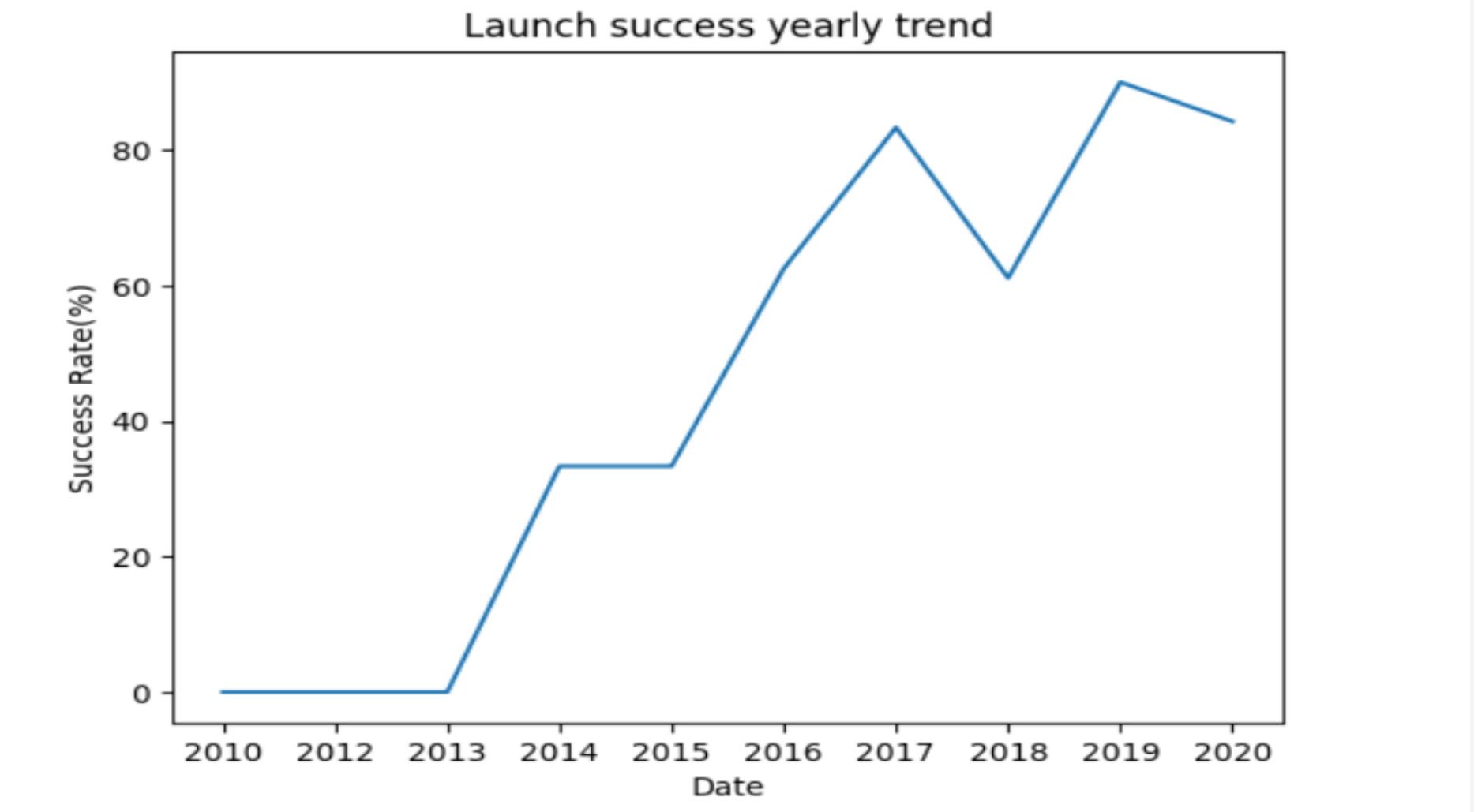
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

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

```
array(['CCAFS LC-40', 'VAFB SLC-4E', 'KSC LC-39A', 'CCAFS SLC-40'],
      dtype=object)
```

Launch Site Names Begin with 'CCA'

Some spacecrafts have their launch sites name beginning with CCA. They have different Payload Mass, Booster Version or Customer. Five of them are presented below.`

	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

- The total payload carried by boosters from NASA amounts to 45596 kgs

```
df2 = df[df["Customer"]=="NASA (CRS)"]
df2 = df2["PAYLOAD_MASS__KG_"].sum()
df2
```

```
np.int64(45596)
```

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is 2928.4 kgs

```
df3 = df[df["Booster_Version"] == "F9 v1.1"]["PAYLOAD_MASS__KG_"].mean()
df3
```

```
np.float64(2928.4)
```

First Successful Ground Landing Date

- The first successful landing outcome on ground pad was on 22nd December,2015

```
]: df4 = df[df["Landing_Outcome"] == "Success (ground pad)"]["Date"].min()
   df4

]: '2015-12-22'
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- Boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
  - F9 FT B1022
  - F9 FT B1026
  - F9 FT B1021.2
  - F9 FT B1031.2

```
# Filter the DataFrame for specific conditions
boosters = df[
    (df["Landing_Outcome"] == "Success (drone ship)") &
    (df["PAYLOAD_MASS__KG_"] > 4000) &
    (df["PAYLOAD_MASS__KG_"] < 6000)
]

# Display the filtered DataFrame
boosters = boosters["Booster_Version"].tolist()
boosters
```

```
['F9 FT B1022', 'F9 FT B1026', 'F9 FT  B1021.2', 'F9 FT  B1031.2']
```

Total Number of Successful and Failure Mission Outcomes

- 98 success outcomes
- 1 failure(in flight) outcome
- 1 success(payload status unclear) outcome
- 1 success outcome

Mission_Outcome	
Success	98
Failure (in flight)	1
Success (payload status unclear)	1
Success	1

Boosters Carried Maximum Payload

- 12 boosters have carried the maximum payload mass
  - i. F9 B5 B1048.4
  - ii. F9 B5 B1049.4
  - iii. F9 B5 B1051.3
  - iv. F9 B5 B1056.4
  - v. F9 B5 B1048.5
  - vi. F9 B5 B1051.4
  - vii. F9 B5 B1049.5
  - viii. F9 B5 B1060.2
  - ix. F9 B5 B1058.3
  - x. F9 B5 B1051.6
  - xi. F9 B5 B1060.3
  - xii. F9 B5 B1049.7

```
# Find the maximum payload mass
max_payload_mass = df["PAYLOAD_MASS__KG_"].max()

# Use a subquery to get booster versions with the maximum payload mass
booster_versions_max_payload = df[df["PAYLOAD_MASS__KG_"] == max_payload_mass]["Booster_Version"].unique()

# Display the result
print("Booster Versions that have carried the maximum payload mass:")
booster_versions_max_payload
```

```
Booster Versions that have carried the maximum payload mass:

array(['F9 B5 B1048.4', 'F9 B5 B1049.4', 'F9 B5 B1051.3', 'F9 B5 B1056.4',
       'F9 B5 B1048.5', 'F9 B5 B1051.4', 'F9 B5 B1049.5',
       'F9 B5 B1060.2 ', 'F9 B5 B1058.3 ', 'F9 B5 B1051.6',
       'F9 B5 B1060.3', 'F9 B5 B1049.7 '], dtype=object)
```

2015 Launch Records

- 2 Failed landing_outcomes in drone ship in year 2015

1. 1st one in January with booster version F9 v1.1 B1012 & launch site at CCAFS LC-40
2. 2nd one in April with booster version F9 v1.1 B1015 & launch site at CCAFS LC-40

```
import pandas as pd
df['Date'] = pd.to_datetime(df['Date'])
failure_drone_ship = df[
    (df['Landing_Outcome'] == 'Failure (drone ship)') &
    (df['Date'].dt.year == 2015)
]
failure_drone_ship['Month'] = failure_drone_ship['Date'].dt.month_name()
result = failure_drone_ship[['Month', 'Landing_Outcome', 'Booster_Version', 'Launch_Site']]
result
```

```
/tmp/ipykernel_85/620662705.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/inde
opy
  failure_drone_ship['Month'] = failure_drone_ship['Date'].dt.month_name()
```

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

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

- 10 had no attempt of landing outcomes
- 5 had failure drone ship outcome
- 5 succeeded in drone ship landing
- 3 had controlled landing on ocean
- 3 succeeded in landing on ground pad
- 2 failed in parachute landinh
- 2 faced uncontrolled ocean landing
- 1 had precluded drone ship landing outcome

```
Landing_Outcome
No attempt                10
Failure (drone ship)       5
Success (drone ship)       5
Controlled (ocean)         3
Success (ground pad)       3
Failure (parachute)        2
Uncontrolled (ocean)       2
Precluded (drone ship)     1
Name: count, dtype: int64
```

Section 3

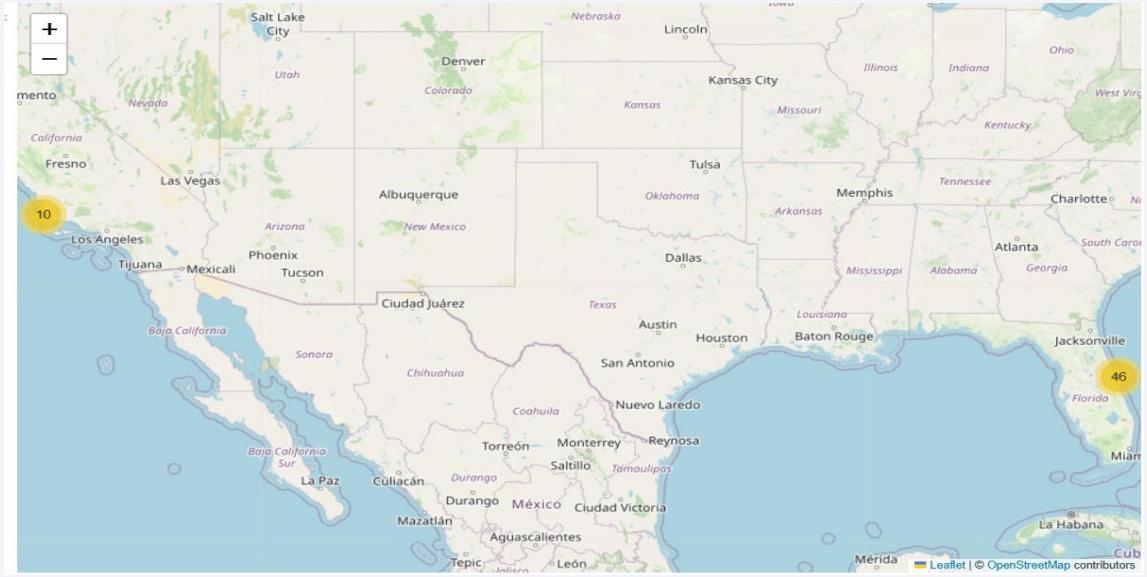
Launch Sites Proximities Analysis

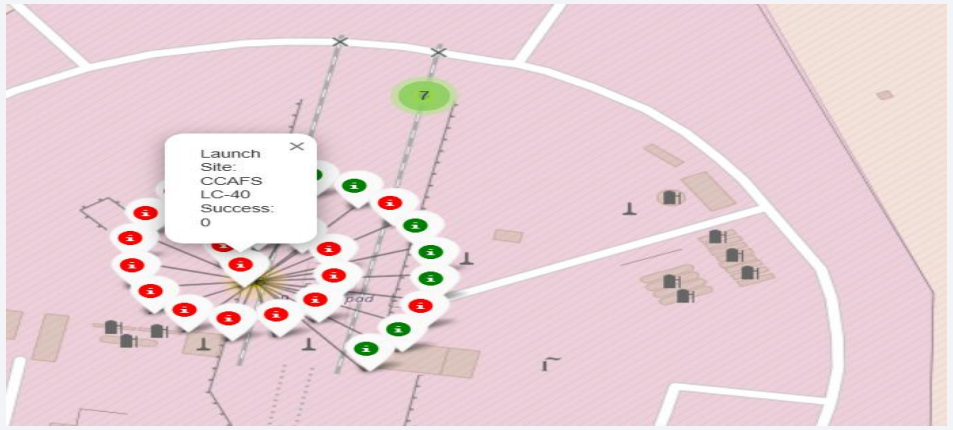
All launch sites on Map

- The launch sites are shown on map on two clusters marked by yellow marker
- All launch sites in proximity to the Equator line
- All launch sites in proximity to the coast

Outcomes For Each Launch Sites On Map

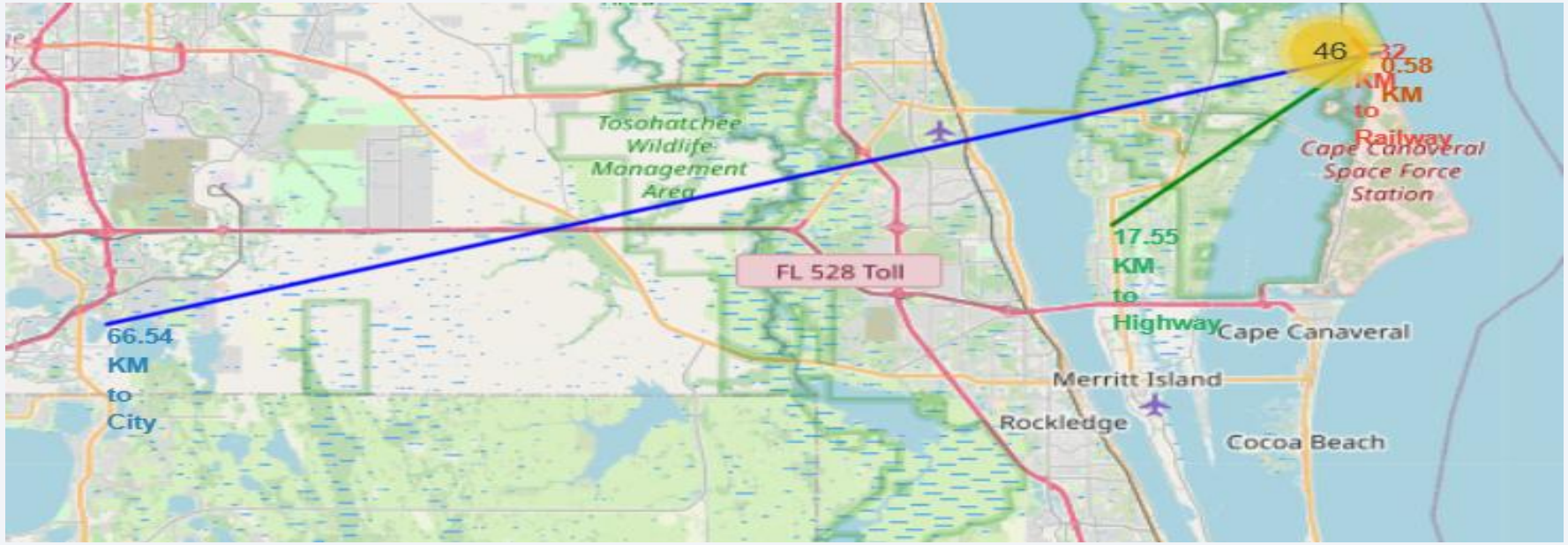
- The upper map shows the number of attempts on each launch sites
- Zooming in to each site, red markers mark the failed outcomes while green ones show the succeeded ones for each site
- The lower map shows outcomes for the launch site CCAFS LC-40

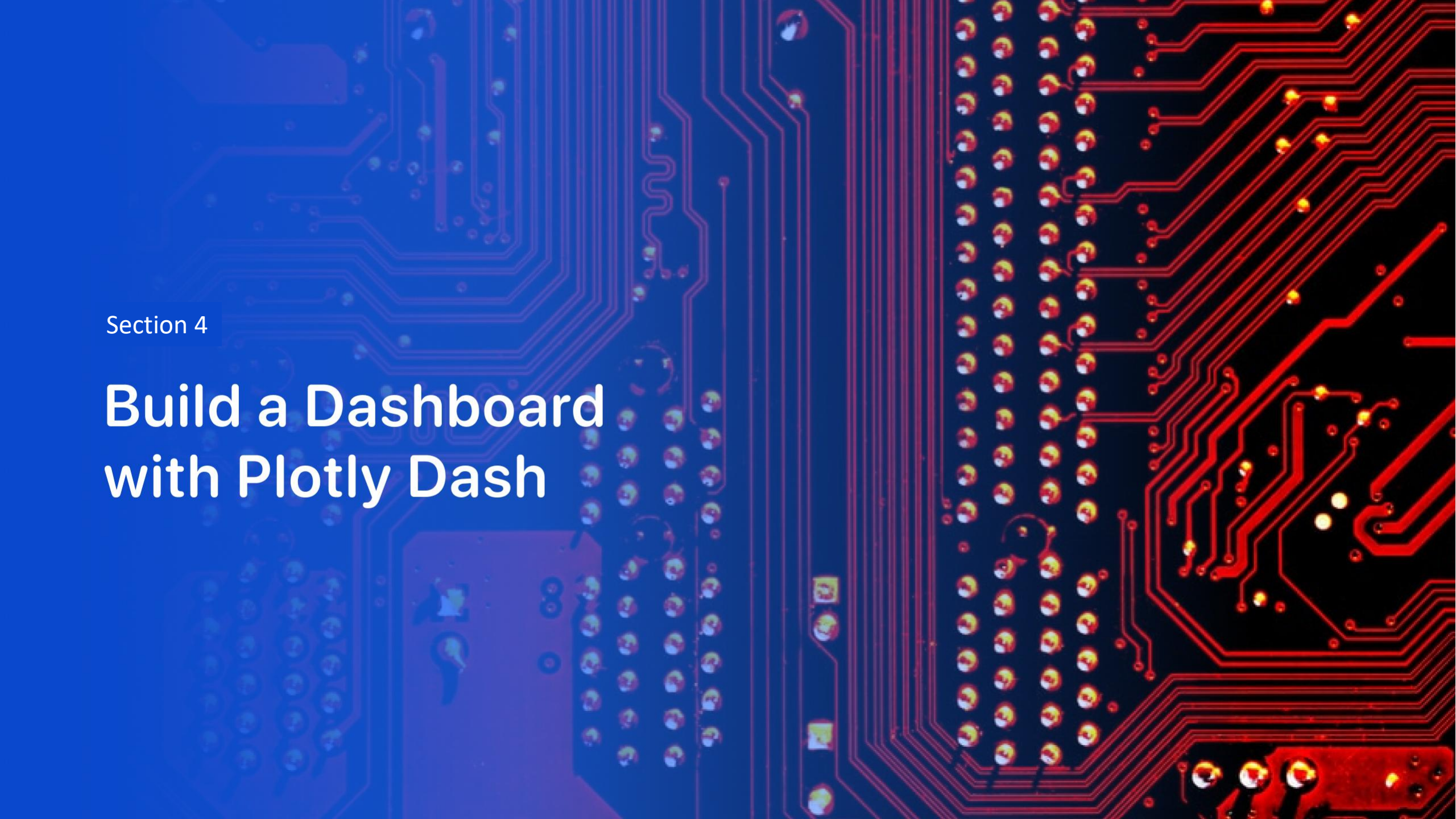




Proximities from Launch Sites on Map

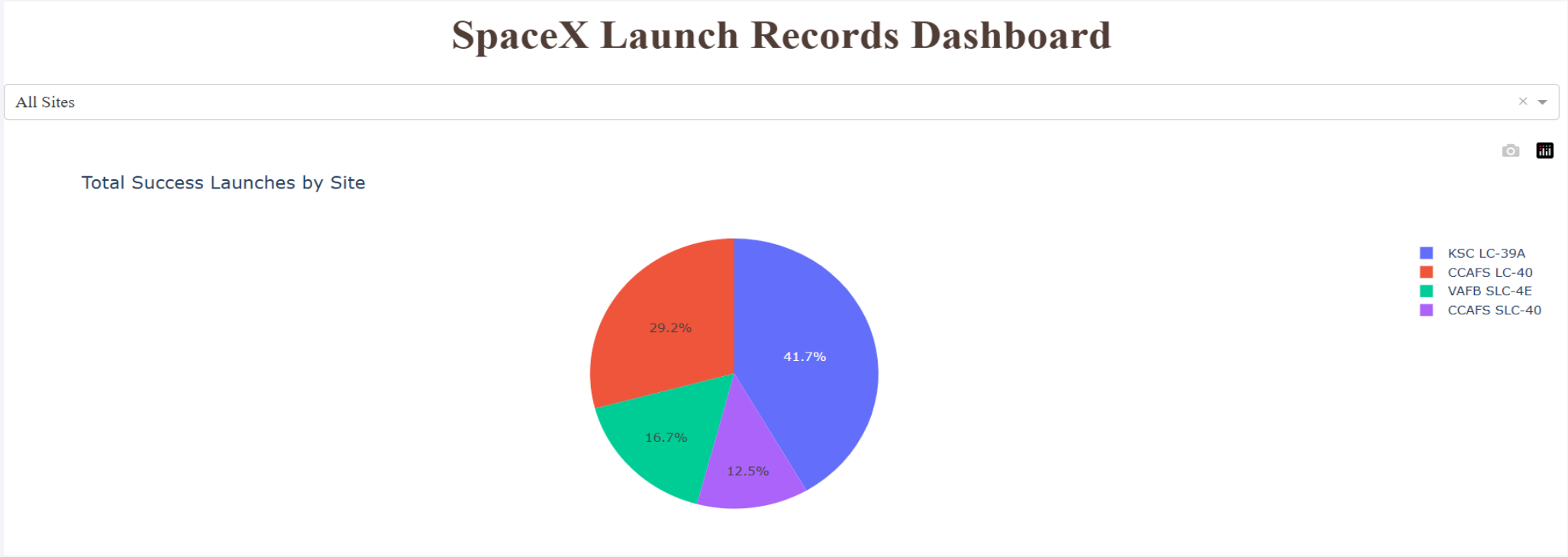
- Launch sites keep safe distance from cities but in very close proximities to coast line and railway.
- Highways are moderately close to the launch sites




Section 4
Build a Dashboard with Plotly Dash

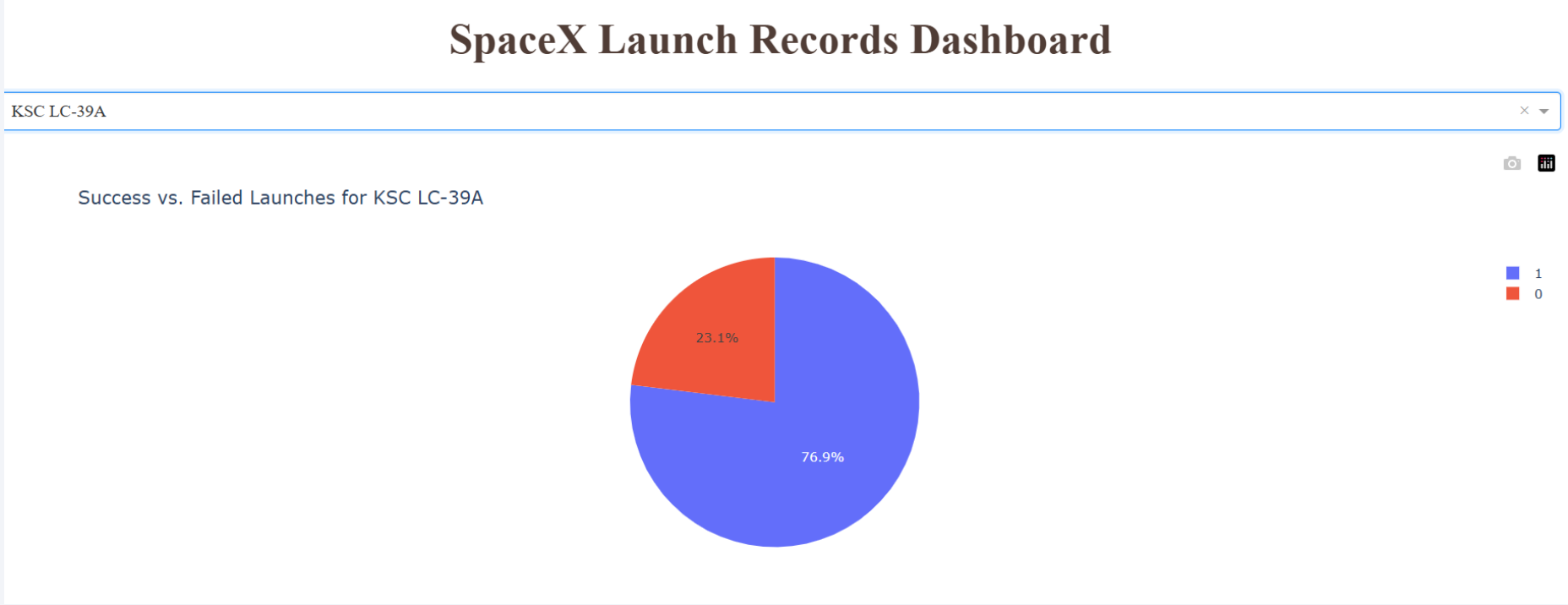
Total Success Launches by All Sites

- KSC LC-39A tops in proportion of successful launches. Among the success launches ,41.7% comprises of this launching site.
- CCAFS SLC-40 is the lowest in proportion with its contribution of 12.5%



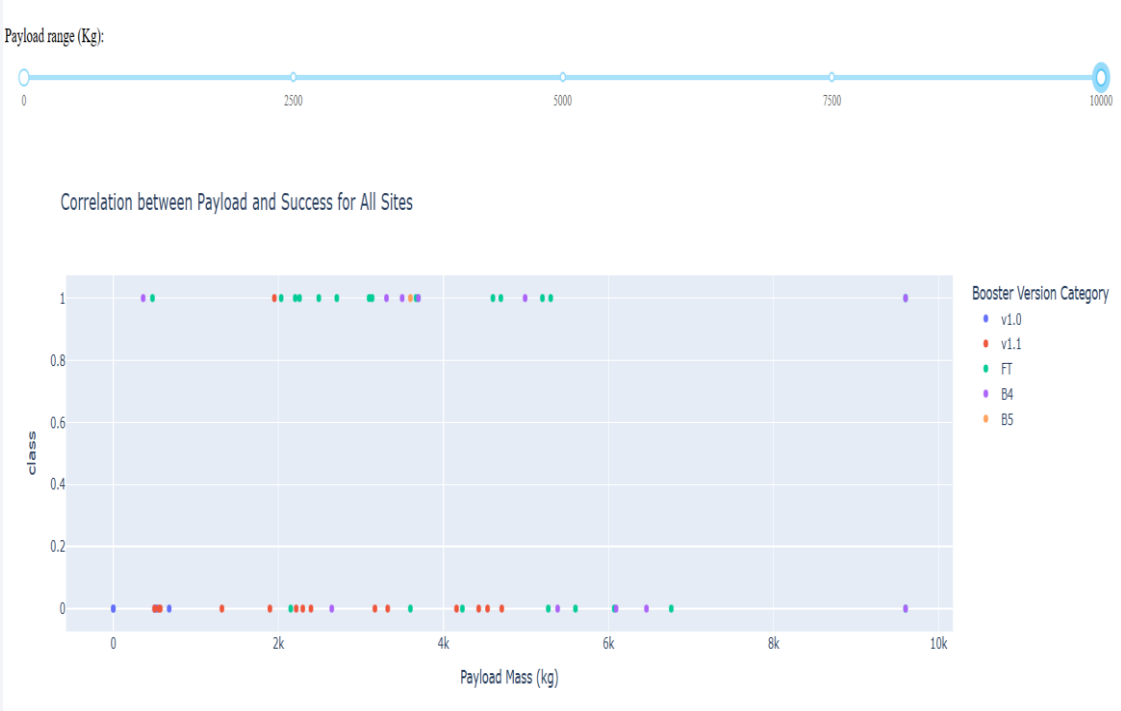
Success Vs Failed Launches for KSC LC-39A

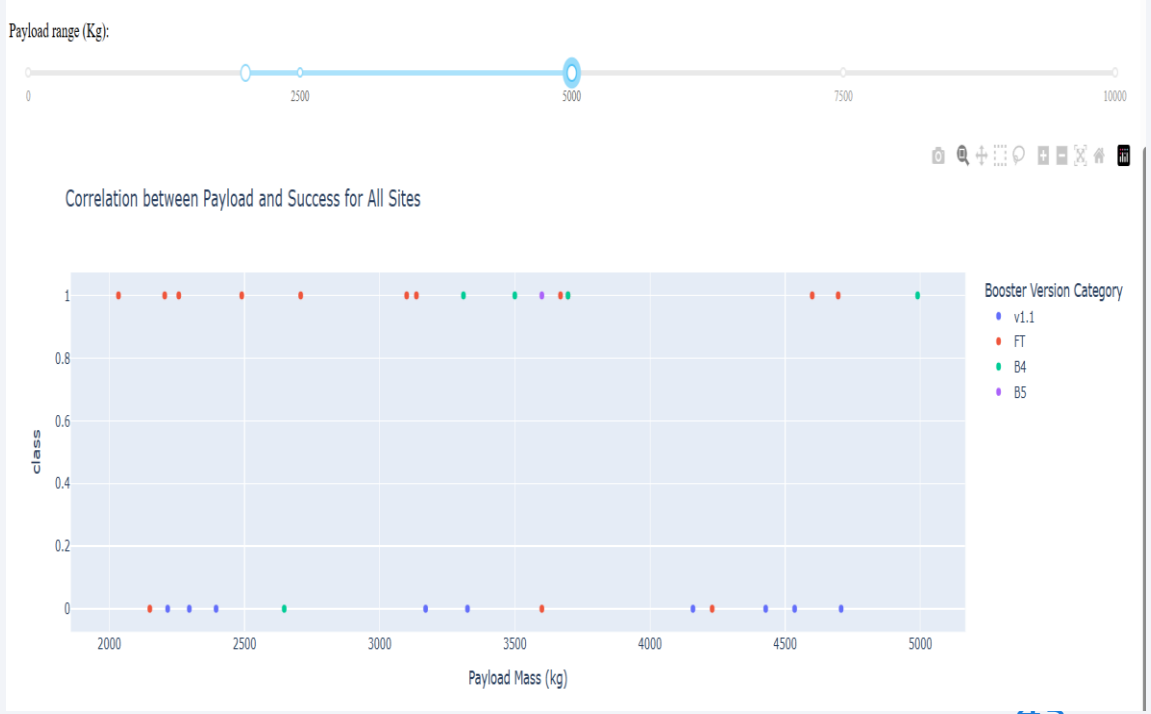
- KSC LC-39A has the highest success launches ratio.
- 76.9% of its attempts succeeded while 23.1% failed in launches.



Payload Vs. Launch Outcome Scatter Plot

- The range between 2000kgs and 5000kgs of Payload Mass has the highest proportion of success rate.
- Booster version FT has the highest success rate.



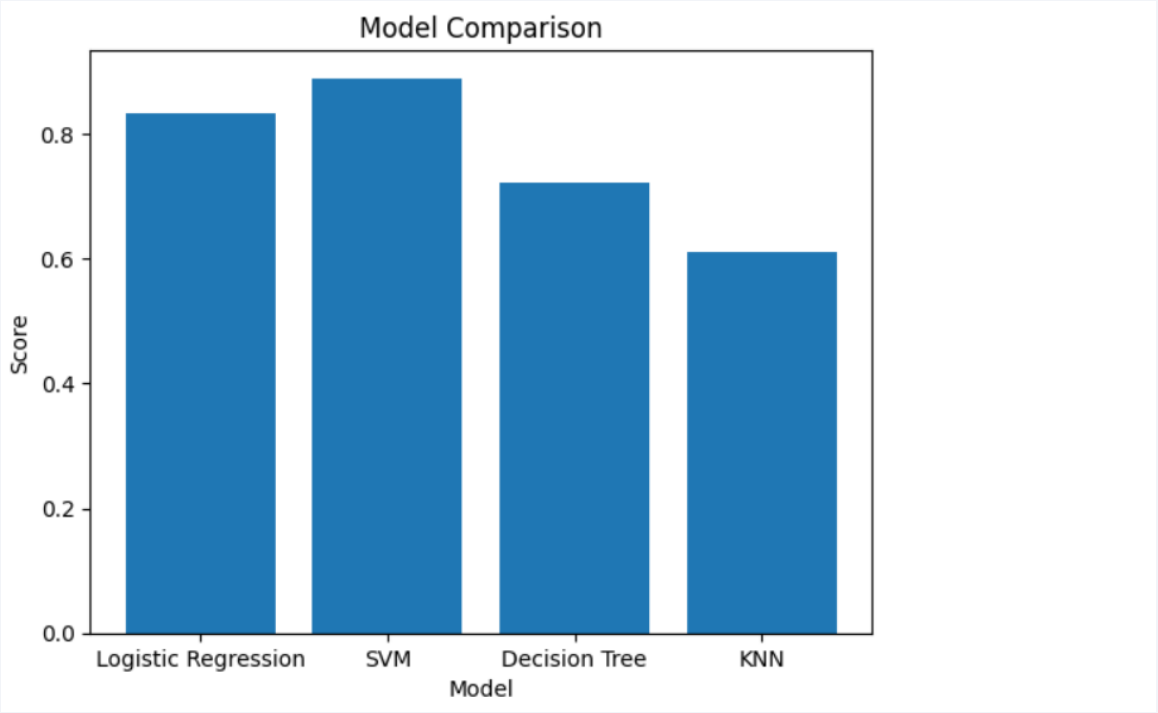


Section 5

Predictive Analysis (Classification)

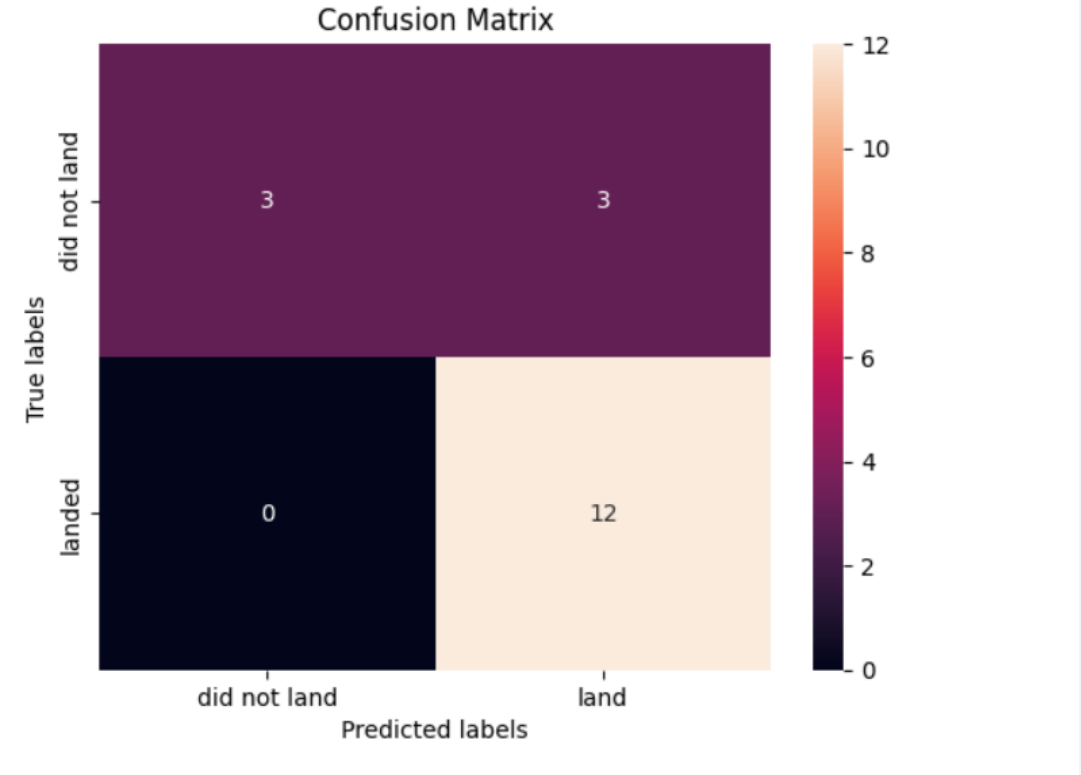
Classification Accuracy

- Support Vector Machine(SVM) model has the highest accuracy(88.88%) among all others.
- Accuracy is measured by the value of R-squared



Confusion Matrix

- It's the confusion matrix of the SVM model which was best performing in terms of accuracy
- There are 12 true positives and 3 false positives
- It means the model predicted 12 times landed when it actually landed, and predicted landed 3 times when it did not truly landed.



Conclusions

- There is no correlation between success rate and Payload Mass or Flight Numbers.
- Success Rate is increasing rapidly from 2013 onwards
- The average payload mass carried by booster version F9 v1.1 is 2928.4 kgs
- Launch sites keep safe distance from cities but in very close proximities to coast line and railway.

Conclusion(Continued)

- KSC LC-39A tops in proportion of successful launches. Among the success launches ,41.7% comprises of this launching site whereas CCAFS SLC-40 is the lowest in proportion with its contribution of 12.5%
- The range between 2000kgs and 5000kgs of Payload Mass has the highest proportion of success rate.
- Booster version FT has the highest success rate.
- Support Vector Machine(SVM) model has the highest accuracy(88.88%) among all other predictive models.

Thank you!