IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Tegar Ahmad Arsy
10th January 2025

Outline

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

Executive Summary

Summary of methodologies

- data collection and data wrangling
- EDA and interactive visual analytics
- predictive analysis

Summary of all results

- EDA with visualization
- EDA with SQL
- interactive map with Folium
- Plotly Dash dashboard
- predictive analysis (classification)

Introduction

Project background and context

SpaceX stands as the leading company in the commercial space industry, revolutionizing space travel by significantly reducing costs. The company promotes Falcon 9 rocket launches on its website at a price of $62 million, whereas other providers charge upwards of $165 million per launch. A substantial portion of these savings stems from SpaceX's ability to reuse the rocket's first stage. Consequently, predicting whether the first stage can successfully land allows us to estimate the launch cost. Leveraging publicly available data and machine learning models, this study aims to predict the likelihood of SpaceX reusing the first stage.

Problems you want to find answers

- What is the impact of variables on the success of the first-stage landing?
- Has the success rate of first-stage landings improved over time?
- Which machine learning algorithm is most effective for binary classification in this scenario?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data on rockets and launches was retrieved from the SpaceX API, accessible at https://api.spacexdata.com/v4/rockets/.
  - Additional data was extracted through web scraping from the Wikipedia page, "List of Falcon 9 and Falcon Heavy launches," available at https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches.
- Perform data wrangling
  - In the dataset, booster landing outcomes were categorized into successful (1) and unsuccessful (0) based on mission results, such as landings on ground pads, drone ships, or ocean regions

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
    - The process begins by creating a NumPy array from the "Class" column and standardizing the data using StandardScaler. The dataset is then split into training and testing sets using train_test_split. Next, a GridSearchCV object with cv = 10 is used to find the best hyperparameters, and it's applied to Logistic Regression (LogReg), Support Vector Machine (SVM), Decision Tree, and K-Nearest Neighbors (KNN) models. The accuracy of each model is evaluated on the test data using the .score() method, and confusion matrices are analyzed for all models. Finally, the Jaccard and F1 score metrics are used to determine the best-performing model.

Data Collection

- Data sets were collected from Space X API (https://api.spacexdata.com/v4/rockets/) and Wikipedia (https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches) using web scraping.

Data Collection – SpaceX API

1. Retrieve Data From API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)
```

2. Convert the response into a JSON file

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Process the data

```
# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)
```

4. Build a dictionary with the data

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

5. Create dataframe

```
df = pd.DataFrame.from_dict(launch_dict)
```

6. Filter dataframe

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

7. Export to CSV file

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

[Open Github]

Data Collection - Scraping

1. Send a request to access the Falcon9 Launch Wiki page

2. Extract all column/variable names from the header of the HTML table.

3. Create a DataFrame by parsing the launch data from the HTML tables.

Open Github

Data Wrangling

1. Load Dataset into Dataframe

```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_pa
df.head(10)
```

2. Find Patterns

```
df['LaunchSite'].value_counts()
```

3. Creating Label

```
landing_class = []
for outcome in df['Outcome']
```

```
df['Class']=landing_class
df[['Class']].head(8)
```

```
df.head(5)
```

[Open Github]

EDA with Data Visualization

As part of the Exploratory Data Analysis (EDA), the following charts were created to extract additional insights from the dataset:

1. **Scatter Plot:**
    1. Displays the relationship or correlation between two variables, making patterns easier to identify.
    2. The following relationships were visualized using scatter plots:
        1. Flight Number vs. Launch Site
        2. Payload vs. Launch Site
        3. Flight Number vs. Orbit Type
        4. Payload vs. Orbit Type
2. **Bar Chart:**
    1. Often used to compare the values of a variable at a specific point in time. Bar charts allow for easy comparison of groups, with the length of each bar proportional to the value it represents.
    2. A bar chart was used to visualize:
        1. The success rate for each orbit type
3. **Line Chart:**
    1. Commonly used to track changes over time, line charts help highlight trends.

Open Github

EDA with SQL

SQL queries were performed to extract and analyze data from the dataset as follows:

- **Displayed the names** of unique launch sites involved in the space missions.
- **Displayed 5 records** where launch sites begin with the string "CCA."
- **Calculated the total payload mass** carried by boosters launched by NASA (CRS).
- **Calculated the average payload mass** carried by booster version F9 v1.1.
- **Listed the date** when the first successful landing outcome in a ground pad was achieved.
- **Listed the names of boosters** that succeeded in drone ship landings and had a payload mass greater than 4000 but less than 6000.
- **Displayed the total number** of successful and failed mission outcomes.
- **Listed the names of booster versions** that carried the maximum payload mass.
- **Displayed records** showing the month names, failure landing outcomes in drone ships, booster versions, and launch sites for the months in the year 2015.
- **Ranked the count of successful landing outcomes** between the dates 04-06-2010 and 20-03-2017 in descending order.

Open Github

Build an Interactive Map with Folium

The **Folium interactive map** is used to analyze geospatial data, providing more interactive visual analytics to better understand the factors, such as the location and proximity of launch sites, that impact launch success rates.

- The following map objects were created and incorporated into the map:
- **Marked all launch sites** on the map, allowing a visual representation of their locations.
- Utilized **'folium.circle'** and **'folium.marker'** to highlight areas around each launch site with a text label.
- Implemented **'MarkerCluster()'** to display success (green) and failure (red) markers for each launch site.
- Calculated the distances from launch sites to nearby features such as coastlines, railroads, highways, and cities.
- Added **'MousePosition()'** to display the coordinates of the mouse position over any point on the map.
- Used **'folium.Marker()'** to show the distance (in kilometers) from the launch site to features like coastlines, railroads, highways, and cities.
- Applied **'folium.Polyline()'** to draw lines between the launch site and the relevant features (coastline, railroad, highway, city).
- Repeated the steps to add markers and draw lines connecting launch sites to nearby features, such as coastlines, railroads, highways, and cities.

Open Github

Build a Dashboard with Plotly Dash

The dashboard includes the following components:

- **Dropdown**: Allows users to select a specific launch site or all launch sites using the dash_core_components.Dropdown.
- **Pie Chart**: Displays the total success and failure for the selected launch site from the dropdown component, visualized with plotly.express.pie.
- **RangeSlider**: Enables users to select a payload mass within a fixed range using dash_core_components.RangeSlider.
- **Scatter Chart**: Illustrates the relationship between two variables, specifically Success vs. Payload Mass, using plotly.express.scatter.

Open Github

Predictive Analysis (Classification)

Data Preparation
1.Load the dataset
2.Normalize the data
3.Split the data into training and test sets
Model Preparation
1.Select machine learning algorithms
2.Set parameters for each algorithm to GridSearchCV
3.Train GridSearchCV models with the training dataset
Model Evaluation
1.Retrieve the best hyperparameters for each model
2.Compute the accuracy for each model using the test dataset
3.Plot the confusion matrix
Model Comparison
1.Compare the models based on their accuracy
2.Choose the model with the best accuracy

Open Github

Results

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

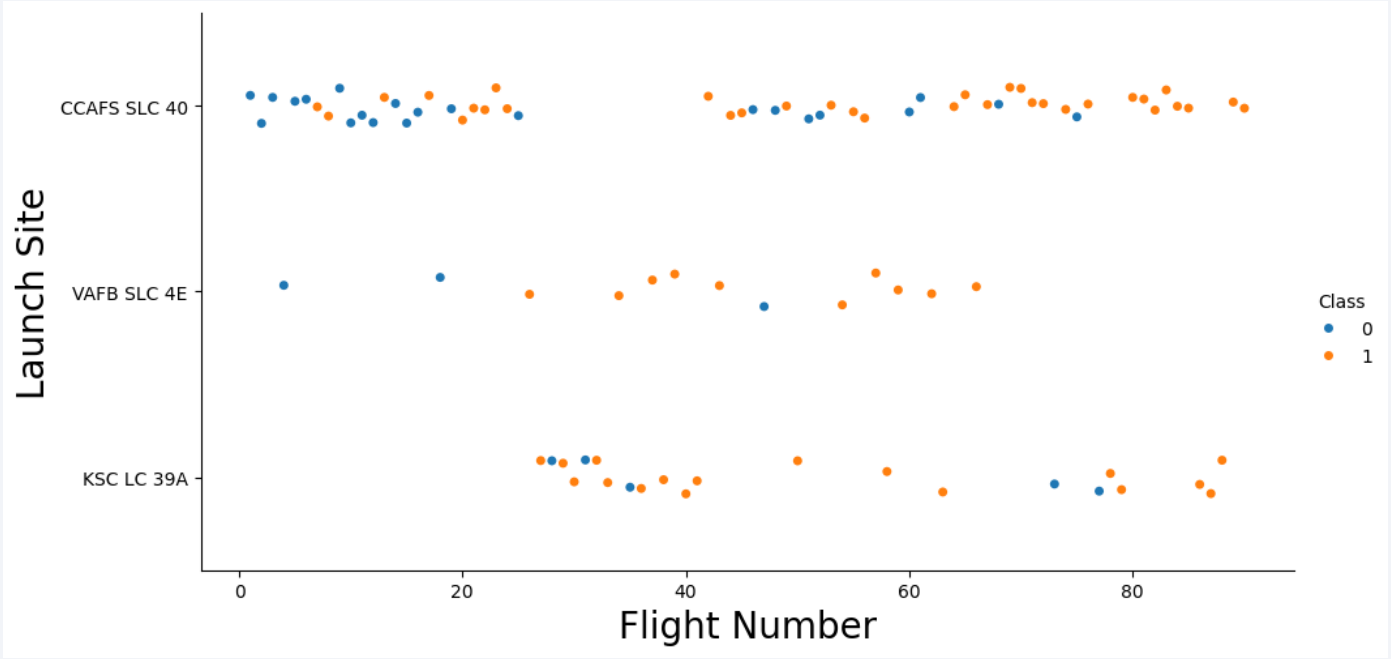
Open Github

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

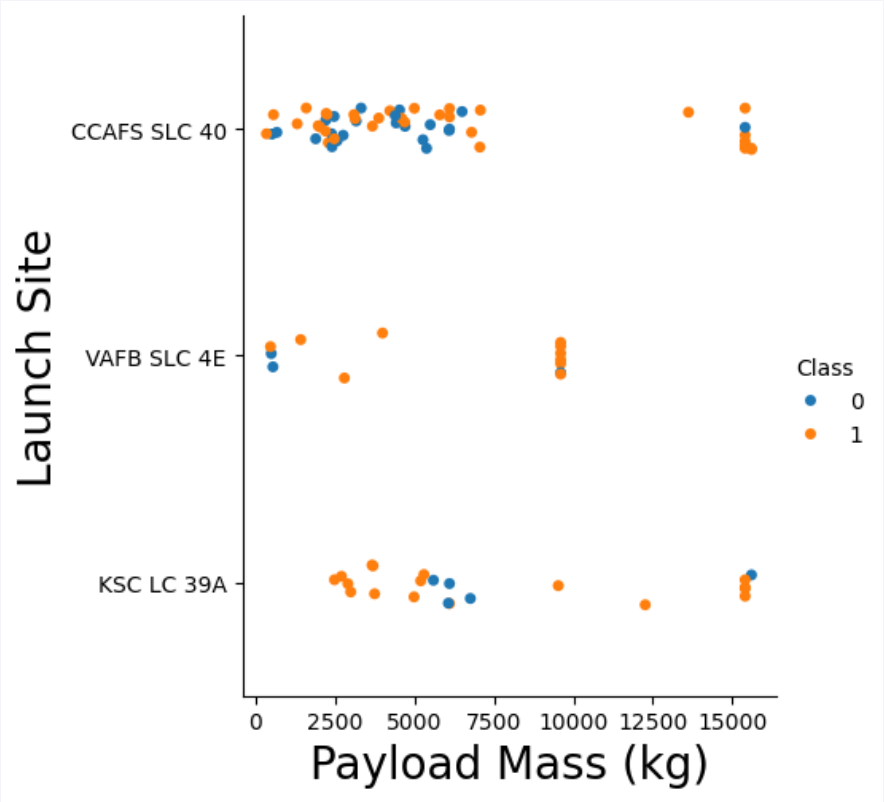
The scatterplot shows the relationship between flight number and launch location, with success and failure outcomes represented by color. Over time, a clear trend emerges where successful launches become more frequent, particularly at locations such as KSC LC 39A and VAFB SLC 4E. CCAFS SLC 40 shows a more balanced distribution of successes and failures, but an increase is seen as the number of flights increases. Overall, the data reflects a pattern of learning and technological advancement, leading to higher success rates on subsequent flights at all locations.



Open Github

Payload vs. Launch Site

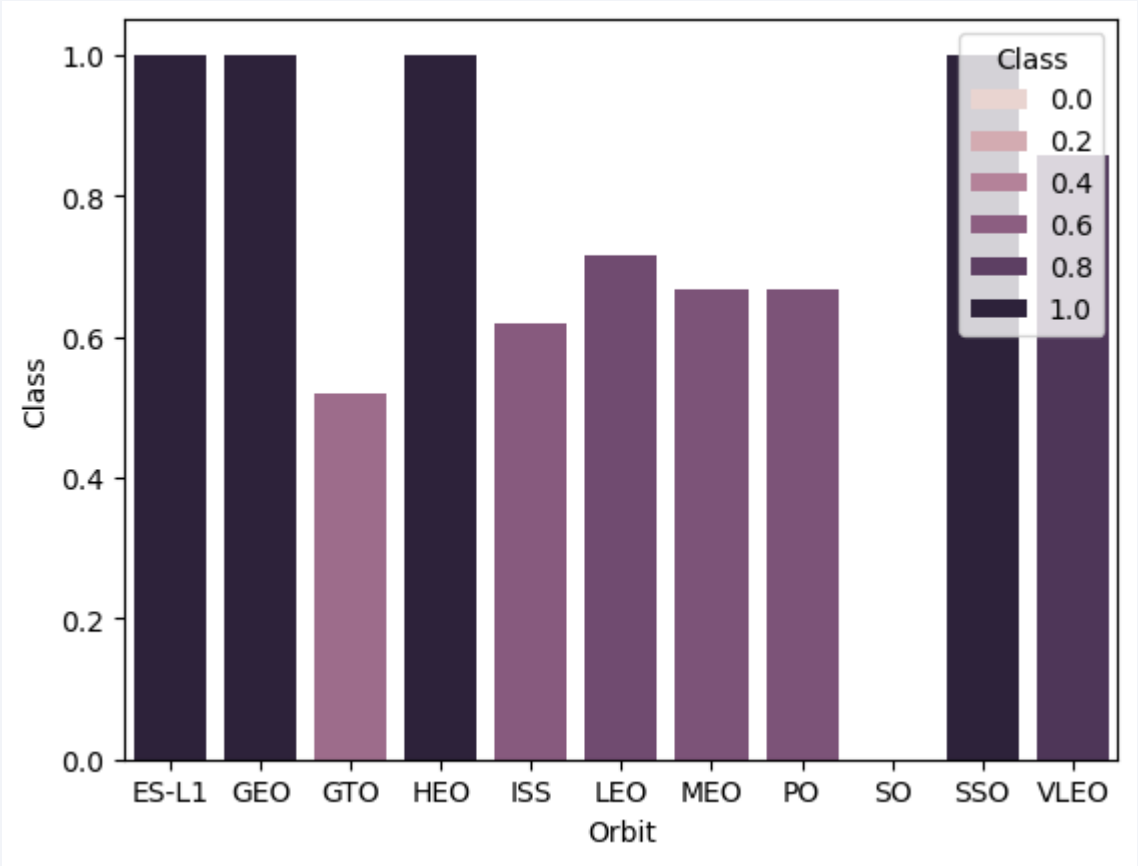
- For the launch site **‘VAFB SLC 4E’**, there were no rockets launched for payloads greater than 10,000 kg.
- The percentage of **successful launches** (Class = 1) at **‘VAFB SLC 4E’** increases as the payload mass increases.
- There is **no clear correlation or pattern** between the **launch site** and the **payload mass** across other sites.



Open Github

Success Rate vs. Orbit Type

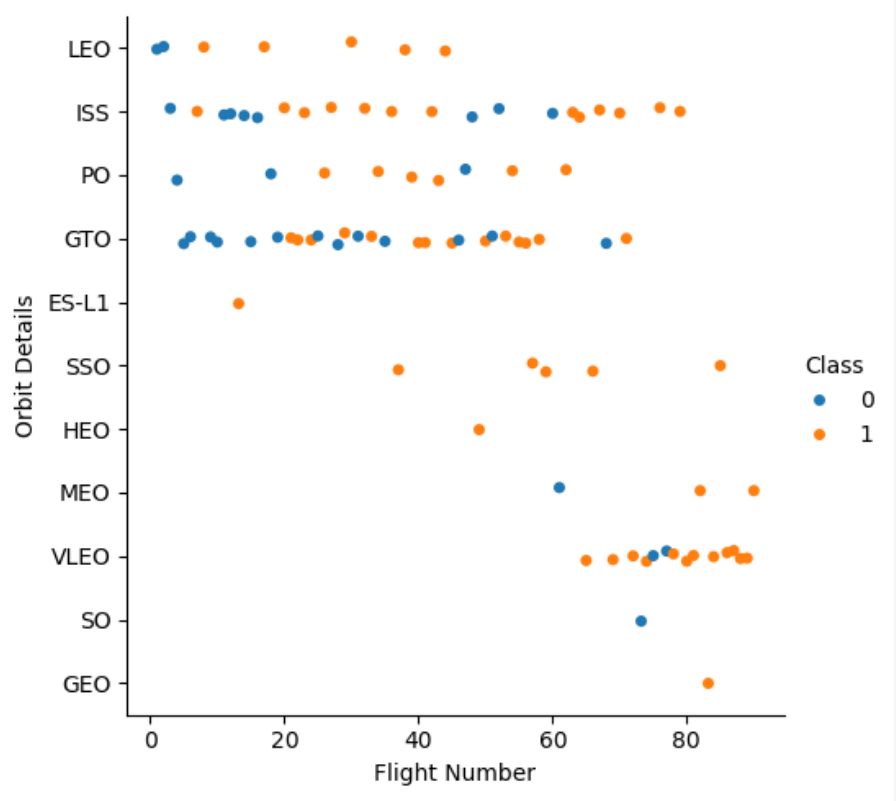
• **Orbits** such as **ES-LI**, **GEO**, **HEO**, and **SSO** exhibit the highest success rates.
•The **GTO** orbit has the lowest success rate.



Open Github

Flight Number vs. Orbit Type

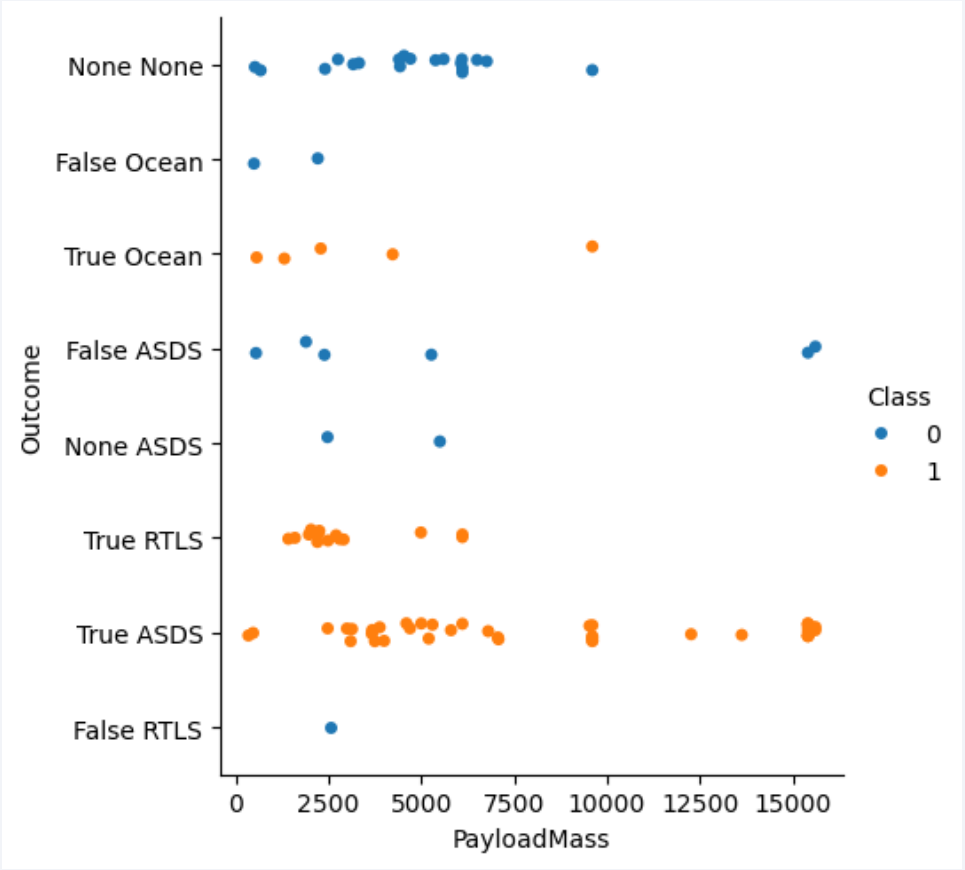
- For the **VLEO** orbit, the first successful landing (Class = 1) doesn't occur until after 60+ flights.
- For most orbits (**LEO**, **ISS**, **PO**, **SSO**, **MEO**, **VLEO**), the **successful landing rates** seem to increase with the number of flights.
- There is **no relationship** between **flight number** and **orbit** for **GTO**.



Open Github

Payload vs. Orbit Type

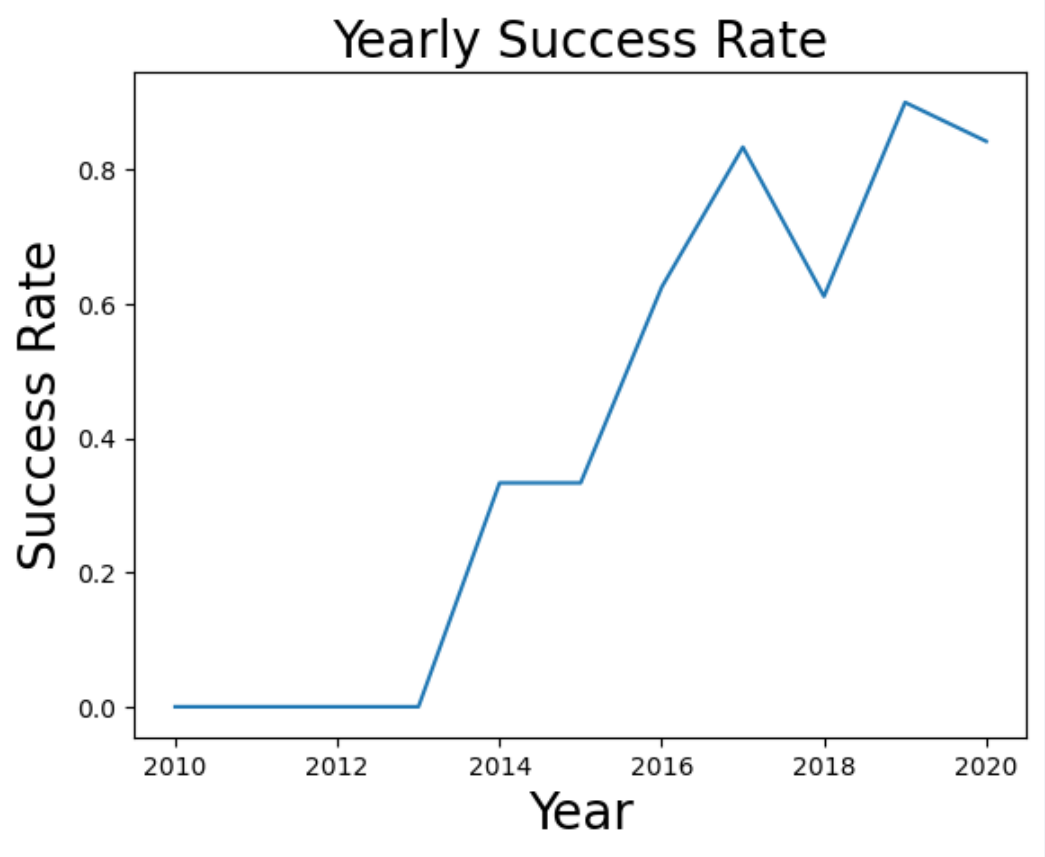
- **Successful landing rates** (Class = 1) appear to increase with payload for orbits such as **LEO**, **ISS**, **PO**, and **SSO**.
- For the **GEO** orbit, there is **no clear pattern** between **payload** and **landing success or failure**.



Open Github

Launch Success Yearly Trend

- The **success rate** (Class = 1) increased by approximately **80%** between **2013 and 2020**.
- **Success rates** remained constant between **2010 and 2013** and between **2014 and 2015**.
- **Success rates** decreased between **2017 and 2018** and between **2019 and 2020**.



Open Github

All Launch Site Names

- '**distinct**' returns only unique values from the queries column (Launch_Site)
- There are **4 unique** launch sites

```
%sql SELECT DISTINCT(Launch_site) FROM SPACEXTBL
```

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

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

Open Github

Launch Site Names Begin with 'CCA'

- Using the **keyword 'Like'** with the format **'CCA%'**, the query returns records where the **‘Launch_Site’** column starts with **"CCA"**.
- The **Limit 5** clause restricts the number of returned records to **5**.

```
%sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE '%CCA%' LIMIT 5
```

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

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

Open Github

Total Payload Mass

The SQL query SELECT sum(PAYLOAD_MASS__KG_) AS payloadmass FROM SPACEXTBL calculates the **total payload mass** (in kilograms) from the **PAYLOAD_MASS__KG_** column in the **SPACEXTBL** table. The result is labeled as **payloadmass**.

```
%sql SELECT sum(PAYLOAD_MASS__KG_) AS payloadmass FROM SPACEXTBL
 * sqlite:///my_data1.db
Done.
```

payloadmass
619967

Open Github

Average Payload Mass by F9 v1.1

The **'avg'** keyword calculates the **average payload mass** from the **PAYLOAD_MASS_KG** column for records where the **booster version** is **'F9 v1.1'**. This helps in determining the typical payload mass for that specific booster version.

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS payloadmass FROM SPACEXTBL
 * sqlite:///my_data1.db
Done.
```

payloadmass
6138.287128712871

[Open Github]

First Successful Ground Landing Date

- The **WHERE clause** filters the dataset to include only records where the landing was successful.
- The **MIN** function is used to retrieve the **earliest date** from the filtered records, effectively selecting the record with the **oldest successful landing**.

```
%sql SELECT MIN(DATE),* FROM SPACEXTBL
```

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

MIN(DATE)	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)

Open Github

Successful Drone Ship Landing with Payload between 4000 and 6000

This query returns the **booster version** where the landing was successful and the **payload mass** is between **4000** and **6000 kg**.

- The **WHERE** clause filters the dataset for records where the landing was successful (e.g., **Class = 1**).
- The **AND** clause further filters the dataset to include only records where the **payload mass** is within the specified range (4000 kg to 6000 kg).

```
%sql SELECT booster_version FROM SPACEXTBL WHERE landing_outcome = 'Success (drone ship)' AND payload_mass__kg_ BETWEEN 4000 AND
 * sqlite:///my_data1.db
Done.
```

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

[Open Github]

Total Number of Successful and Failure Mission Outcomes

- The **'GROUP BY'** keyword is used to arrange identical data in a column into groups.
- In this case, the **number of mission outcomes** are grouped by the **types of outcomes**, and the result is displayed in the **'counts'** column. This allows for summarizing the count of each outcome type.

```
%sql SELECT COUNT(mission_outcome), mission_outcome FROM SPACEXTBL GROUP BY mission_outcome
```

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

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

Open Github

Boosters Carried Maximum Payload

- The subquery uses the **'max'** keyword to retrieve the maximum payload mass from the payload mass column.
- The main query then returns **booster versions** along with their respective payload mass where the payload mass is **15600**, which is the maximum value identified by the subquery.

```
%sql SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL)
```

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

Booster_Version
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

Open Github

2015 Launch Records

This query returns the **month**, **booster version**, and **launch site** for records where the landing was **unsuccessful** and the landing date occurred in **2015**.

- The **SUBSTR** function is used to extract parts of the **DATE**:
  - **SUBSTR(DATE, 4, 2)** extracts the **month** from the **DATE** column.
  - **SUBSTR(DATE, 7, 4)** extracts the **year** from the **DATE** column.
- The query filters the data for **unsuccessful landings** and ensures that the landing year is **2015**.

```
%sql SELECT substr(Date,6,2) AS month, DATE, BOOSTER_VERSION, LAUNCH_SITE, landing_outcome FROM SPACEXTBL WHERE landing_outcome =
 * sqlite:///my_data1.db
Done.
```

month	Date	Booster_Version	Launch_Site	Landing_Outcome
01	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Open Github

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

This query returns the **landing outcomes** and their **count** where the mission was successful (e.g., **Class = 1**) and the **date** is between **04/06/2010** and **20/03/2017**.

- The **GROUP BY** clause groups the results by **landing outcome**.
- The **ORDER BY COUNT DESC** clause arranges the results in **decreasing order** based on the count of each landing outcome.

```
%sql SELECT LANDING_OUTCOME FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY DATE DESC
```

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

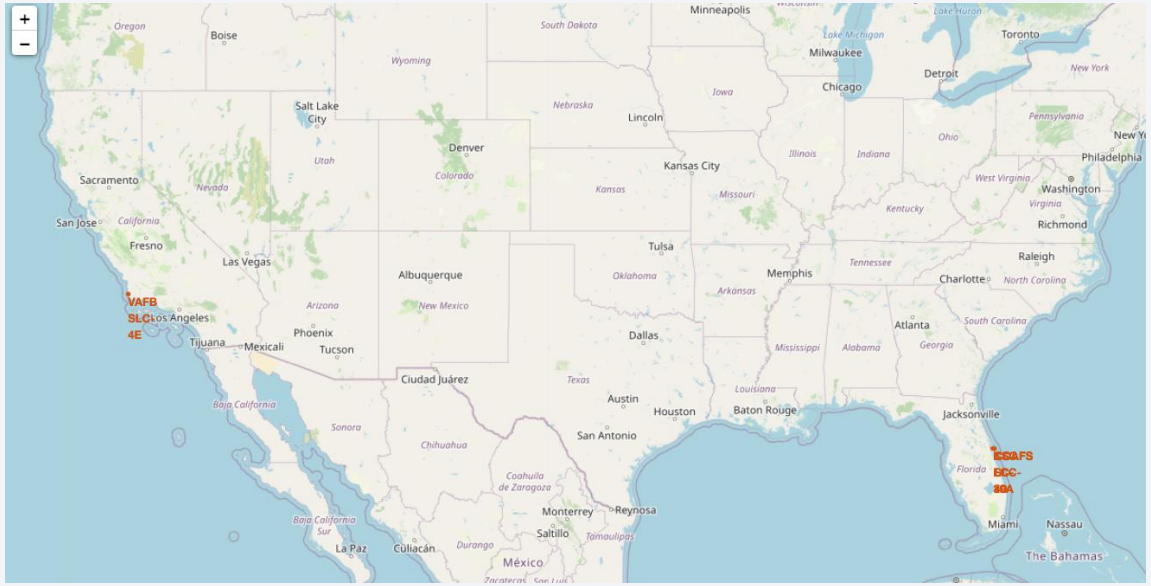
Landing_Outcome
No attempt
Success (ground pad)
Success (drone ship)
Success (drone ship)
Success (ground pad)
Failure (drone ship)
Success (drone ship)
Success (drone ship)
Success (drone ship)
Failure (drone ship)
Failure (drone ship)
Success (ground pad)

Open Github

Section 3

Launch Sites Proximities Analysis

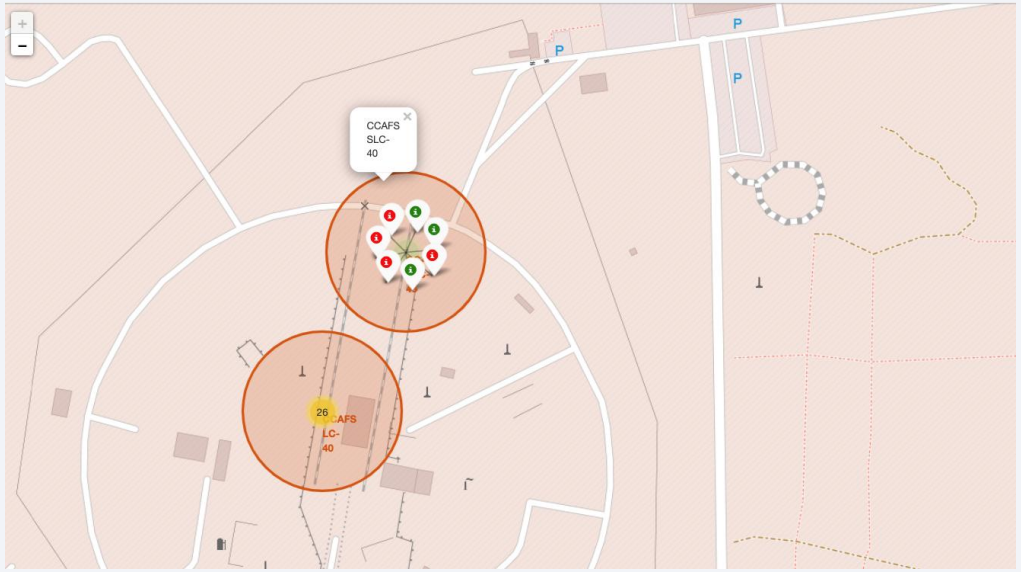
Launch Sites SpaceX



The **SpaceX launch sites** are situated along the **coastline of the United States**, which likely provides advantages related to safety, access to international airspace, and the ability to launch rockets over the ocean, reducing risks to populated areas.

[Open Github]

Succes/Failed Launch Map



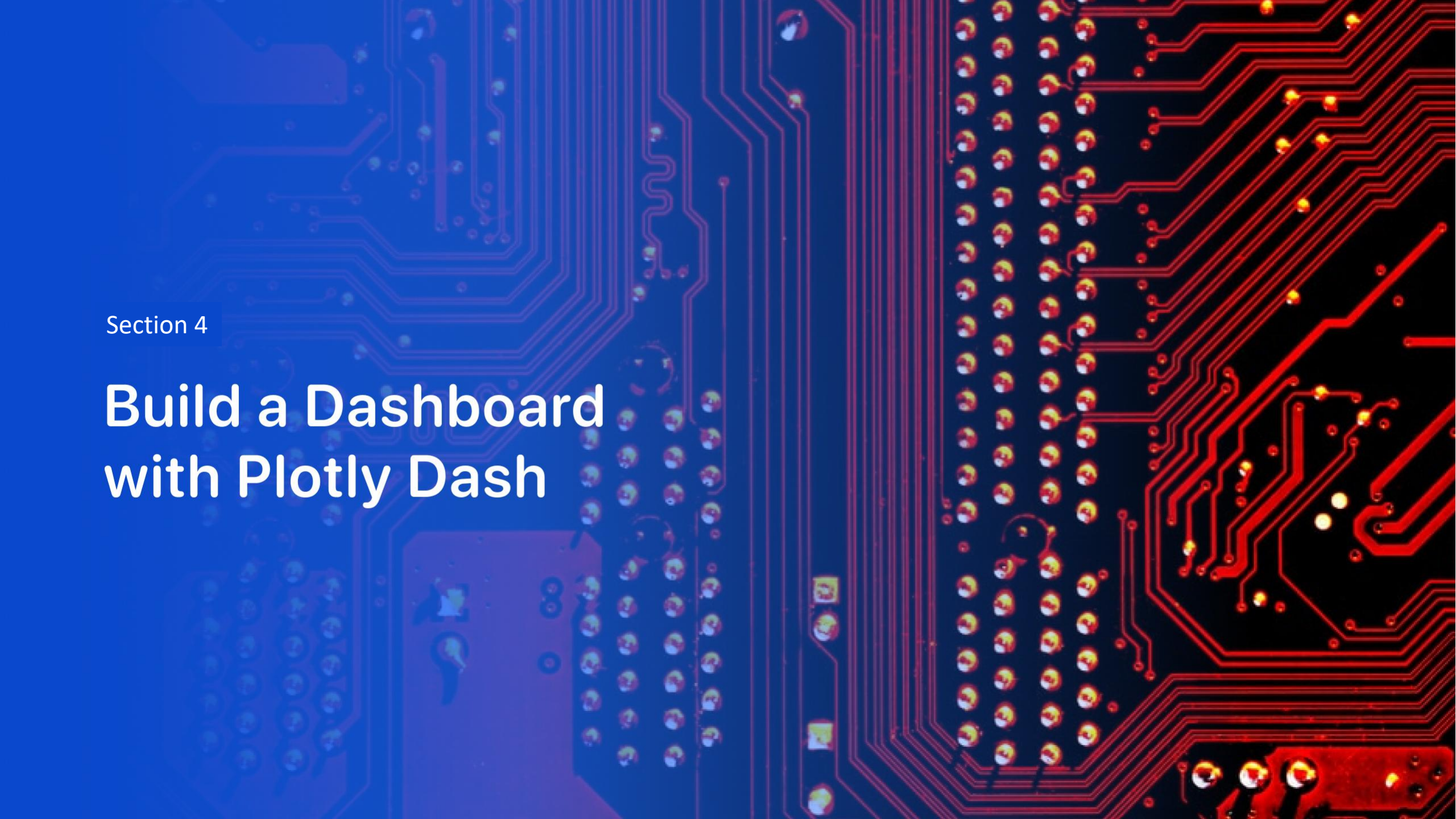
The **green marker** represents **successful launches**, while the **red marker** represents **unsuccessful launches**. It is observed that **KSC LC-39A** has a **higher launch success rate** compared to other sites.

Open Github

Launch Sites Distance Map

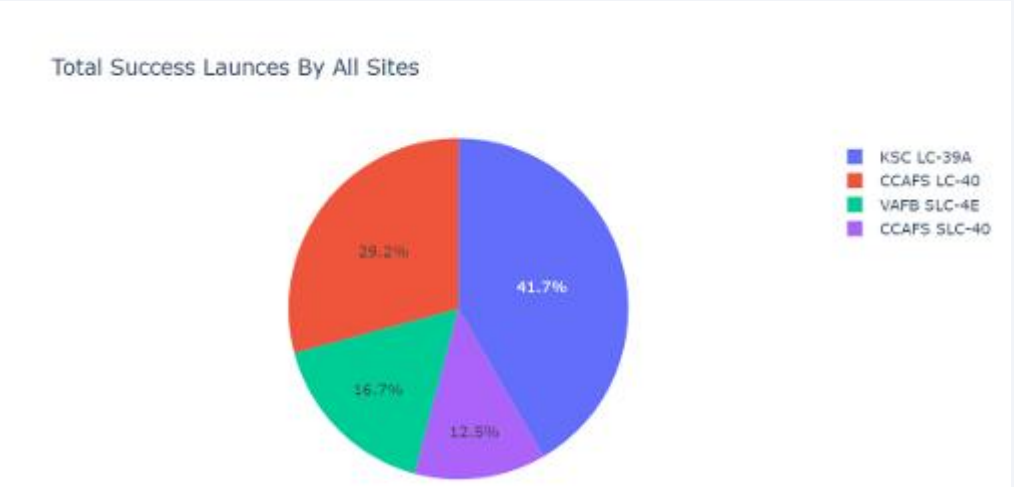
The **launch sites** are strategically located away from **cities** to minimize the impact of any potential accidents on the general public and infrastructure. These sites are positioned near the **coastline**, **railroads**, and **highways**, ensuring easy access to **resources** and efficient transportation logistics.

Open Github


Section 4
Build a Dashboard with Plotly Dash

Launch Success

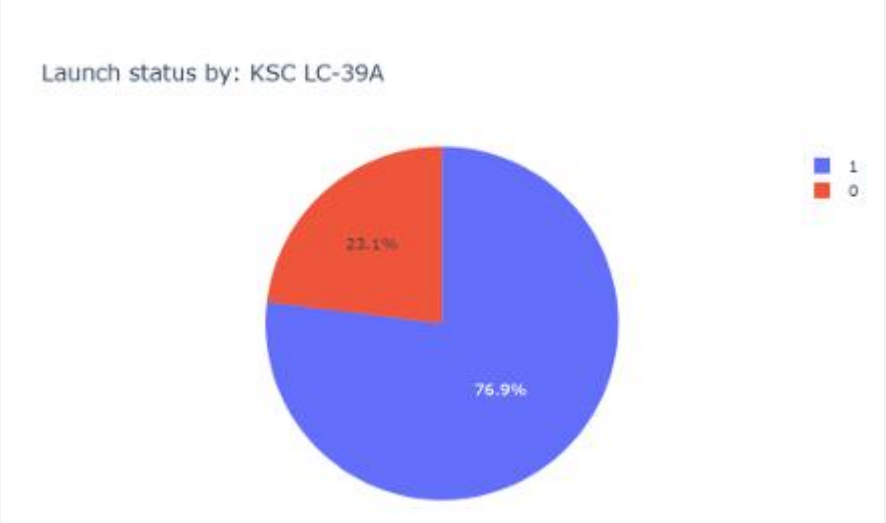
It is observed that **KSC LC-39A** has the **best success rate** of launches among the other launch sites, indicating that it has consistently achieved higher success in its missions.



Open Github

Total success launches for Site KSC LC-39A

It is observed that **KSC LC-39A** has achieved a **76.9% success rate** and a **23.1% failure rate**, demonstrating a relatively high success rate compared to its failure rate in rocket launches.



[Open Github]

Payload vs. Launch Outcome Scatter Plot

• The majority of **successful launches** occur within the **payload range of 2000 to 5500 kg**.
• The **booster version** category **'FT'** has the highest number of successful launches.

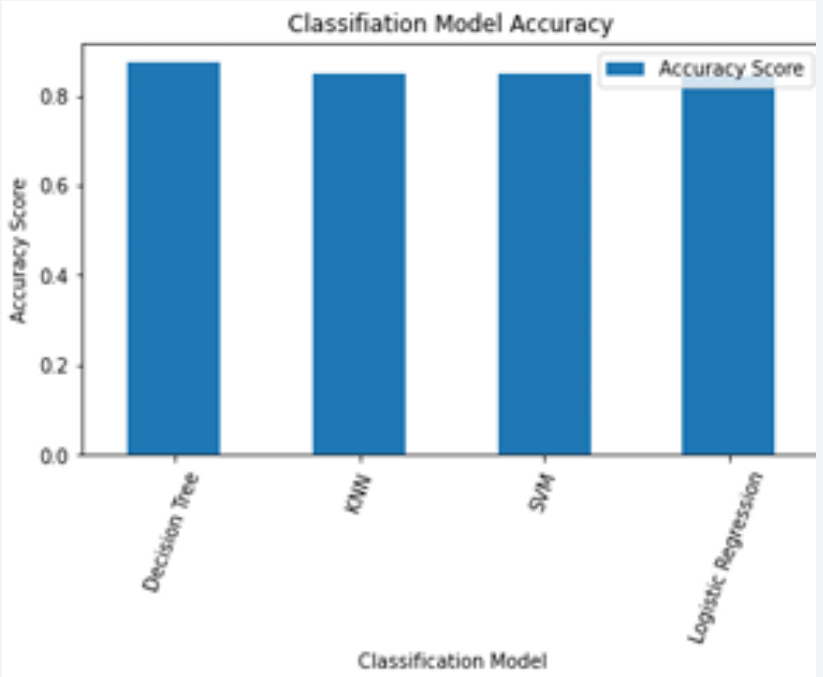
Open Github

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Based on the **accuracy scores** and as shown in the **bar chart**, the **Decision Tree** algorithm has the highest classification score, with a value of **0.8750**.
- The **accuracy score** on the test data is the same for all the classification algorithms, with a value of **0.8333**.
- Given that the accuracy scores for the classification algorithms are very close and the test scores are the same, it may be necessary to use a **broader dataset** to further **tune the models** and improve performance.



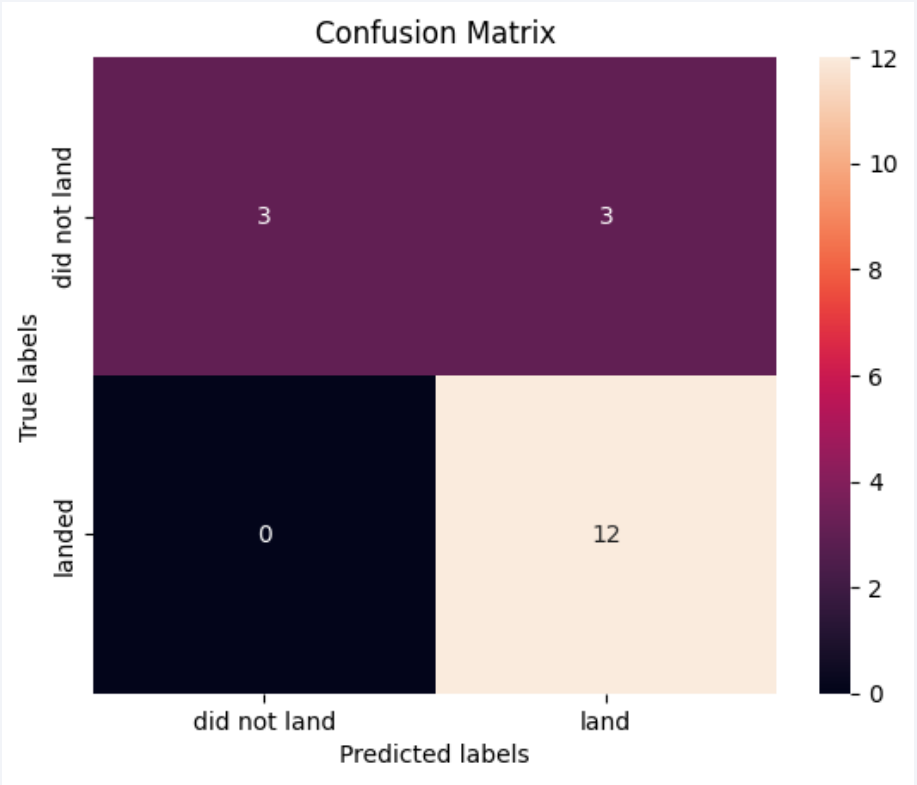
Open Github

Confusion Matrix

The **confusion matrix** is the same for all the models (**LR, SVM, Decision Tree, KNN**), and based on the matrix:

- The classifier made **18 predictions**.
- **12 scenarios** were predicted as **Yes** for landing, and they **did land successfully** (**True Positive**).
- **3 scenarios** (top left) were predicted as **No** for landing, and they **did not land** (**True Negative**).
- **3 scenarios** (top right) were predicted as **Yes** for landing, but they **did not land successfully** (**False Positive**).

Overall, the classifier is **correct about 83%** of the time (**(TP + TN) / Total**) with a **misclassification or error rate** of about **16.5%** (**(FP + FN) / Total**).



Open Github

Conclusions

• The **success of a mission** can be influenced by several factors, such as the **launch site**, **orbit**, and especially the **number of previous launches**. It's assumed that experience and knowledge gained between launches can contribute to a successful mission after overcoming initial failures.

• The **orbits with the best success rates** are **GEO, HEO, SSO**, and **ES-L1**.

•For different orbits, **payload mass** can be a crucial factor for mission success. Some orbits may require a lighter or heavier payload mass. However, in general, **low-weighted payloads tend to perform better** than heavier ones.

• Based on the current dataset, it's difficult to explain why some **launch sites perform better** than others, such as **KSC LC-39A**, which has the highest success rate. To answer this, additional data, like **atmospheric conditions**, could be collected.

• For this analysis, the **Decision Tree Algorithm** was chosen as the best model, despite the test accuracy being identical across all models. The Decision Tree was preferred because it had a **better training accuracy**, making it more effective for this particular dataset.

Open Github

Thank you!