IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Devin P.
18.06.2025

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 results
  - Interactive analytics in screenshots
  - Predictive Analytics results

Introduction

- SpaceX is the most successful company of the commercial space age, revolutionizing the industry by making space travel significantly more affordable. The company offers Falcon 9 rocket launches for around 62 million dollars, while traditional providers often charge 165 million dollars or more. A key reason for this cost advantage is SpaceX's ability to reuse the first stage of its rockets by successfully landing them after launch. Reusing rocket stages drastically reduces the cost per launch. Therefore, predicting whether the first stage will land successfully is essential for estimating launch costs. As a data scientist at a startup aiming to compete with SpaceX, the goal of this project is to develop a machine learning pipeline that can predict the landing outcome of the first stage. This insight will be critical for making informed bidding decisions when offering competitive launch services.
- Problems:
  - What factors influence the success of the first stage landing?
  - How can the success rate of the landing outcome be increased?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Collected using web scrapping from Wikipedia and SpaceX REST API
- Perform data wrangling
  - Data processed using one-hot encoding
- 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 API requests from SpaceX REST API and Web Scraping from the SpaceX Wikipedia page.
- For the REST API: 1. getting the data via get request 2. normalized the Json to a pandas dataframe 3. Data cleaning and filling missing values.
- For the Web Scraping: 1. BeautifulSoup to extract the table on the Wikipedia page 2. Parsed the data and converted it to a pandas dataframe.

Data Collection – SpaceX API

For rocket launch data: get request using API

Json_normalize to convert json to a dataframe

Datacleaning

Filling missing data values

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```
✓ 0.0s

```
response = requests.get(spacex_url)
```
✓ 0.6s

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```
✓ 0.0s

```
# Lets take a subset of our dataframe keeping only the features we want and the flight
# number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2
# extra rocket boosters and rows that have multiple payloads in a single rocket.
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value
# in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting
# the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

GitHub: https://github.com/DevinPoppe/IBM-Applied-Data-Science-Capstone/blob/main/jupyter-labs-spacex-data-collection-api-v2.ipynb

Data Collection - Scraping

Get the data from the URL via requests

- Creating a BeautifulSoup object from the response
- Find all tables and extract the information
- Parse the data

```
# use requests.get() method with the provided static_url
# assign the response to a object

data = requests.get(static_url)
```

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data.text, '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')
```


GitHub: https://github.com/DevinPoppe/IBM-Applied-Data-Science-Capstone/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

- 1. Data Loading
- 2. Handling missing values
- 3. Feature Analysis
- 4. Outcome Classification

```
df.isnull().sum()/len(df)*100
```

```
# Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for outcome in df["Outcome"]:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)

# Print the first 10 rows of the dataframe
df.head(10)
```


GitHub: https://github.com/DevinPoppe/IBM-Applied-Data-Science-Capstone/blob/main/labs-jupyter-spacex-Data%20wrangling-v2.ipynb

EDA with Data Visualization

- 1. Flight Number vs. Launch Site
- 2. Payload Mass vs. Launch Site
- 3. Success Rate by Orbit Type
- 4. Flight Number vs. Orbit Type
- 5. Payload Mass vs. Orbit Type
- 6. Yearly Success Trend

Scatter plots reveal relationships between numerical and categorical variables. Line plots show yearly trends and bar charts display success rates across different categories.

GitHub: https://github.com/DevinPoppe/IBM-Applied-Data-Science-Capstone/blob/main/jupyter-labs-eda-dataviz-v2.ipynb

EDA with SQL

- Performed SQL queries:

- Unique Launch Sites
- Launch Sites Starting with 'CCA'
- Total Payload Mass for NASA (CRS) Missions
- Average Payload Mass for Booster F9 v1.1
- First Successful Ground Pad Landing Date
- Boosters with Drone Ship Success (Payload 4k–6k kg)
- Mission Outcomes (Success/Failure Counts)
- Mission Outcomes (Success/Failure Counts)
- 2015 Drone Ship Failures
- Landing Outcome Rankings (2010–2017)

Used SQL functions:

DISTINCT, SUM, AVG, MAX, COUNT, LIKE, WHERE, ORDER BY, GROUP BY, strftime

GitHub: https://github.com/DevinPoppe/IBM-Applied-Data-Science-Capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- 1. Base Map Elements
  - Created a Folium Map
- 2. Launch Site Markers
  - Added circles around each launch site and displayed their names
- 3. Launch Outcome Visualization
  - Grouped markers by location added a color coding to quickly assess success or failure per site
- 4. Proximity Analysis
  - To identify positions of nearby infrastructure for logistical advantages (costal sites for easy rocket recovery)

GitHub: https://github.com/DevinPoppe/IBM-Applied-Data-Science-Capstone/blob/main/lab-jupyter-launch-site-location-v2.ipynb

Build a Dashboard with Plotly Dash

- Added a success pie chart: to display the success/failure ratio for each launch site. Provides immediate visual comparison
- Added a payload vs. success scatter plot: to display the correlation between payload mass and launch success. Helps analyze whether payload mass correlates with mission success and if booster versions perform differently.

- The dropdown menu allows for individual selection and the range slider enables filtering payload mass range, which dynamically updates.

GitHub: https://github.com/DevinPoppe/IBM-Applied-Data-Science-Capstone/blob/main/spacex_dash_app.py

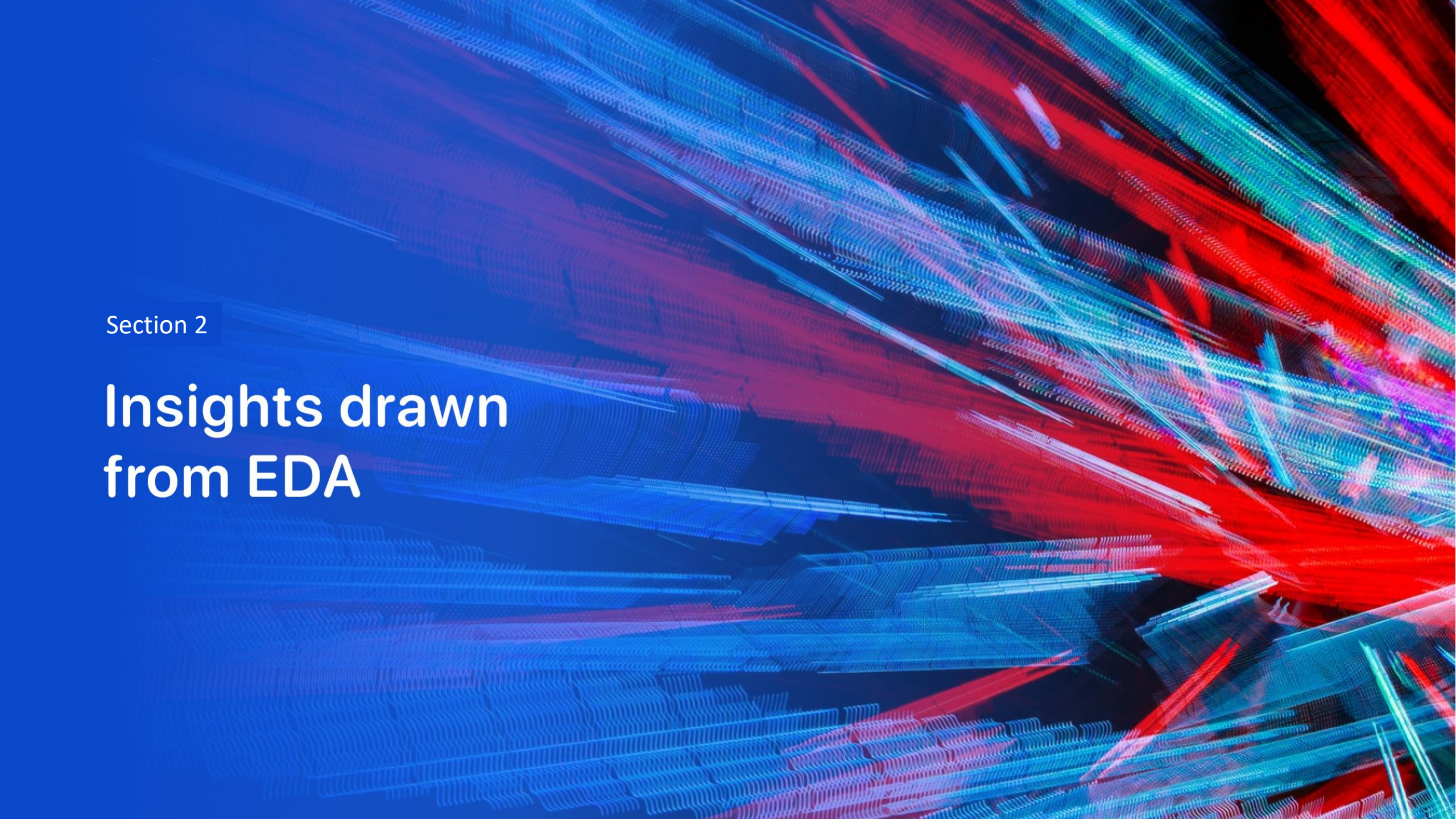
Predictive Analysis (Classification)

- 1. Data Preparation
  - Standardized the dataset and split data into train/test sets (80/20)
- 2. Model Selection & Tuning
  - Tested 4 classification algorithms (Logistic Regression, SVM, Decision Tree & KNM)
  - Used GridSearchCV with 10-fold CV for tuning
- 3. Evaluation
  - Compared accuracy scores, test set performance

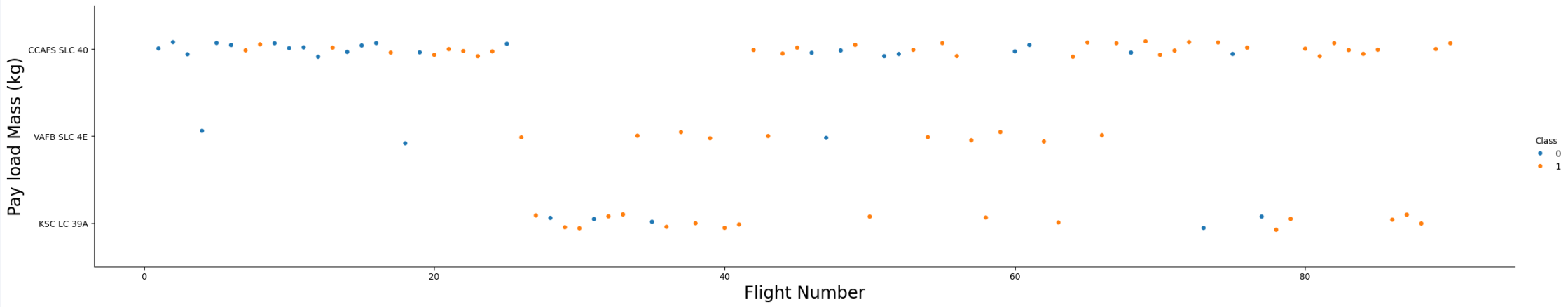
GitHub: https://github.com/DevinPoppe/IBM-Applied-Data-Science-Capstone/blob/main/SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb

Results

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


Section 2
Insights drawn from EDA

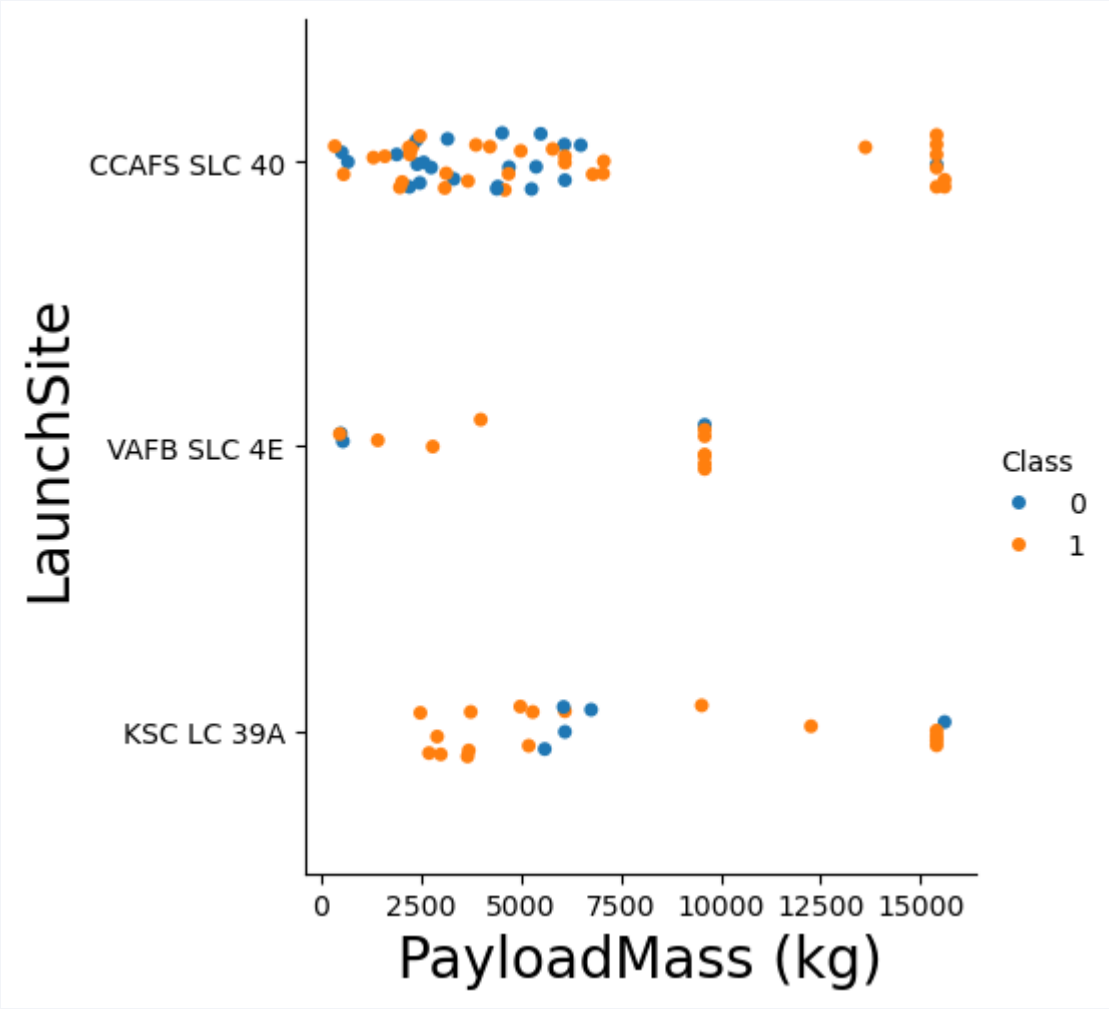
Flight Number vs. Launch Site



- As the times goes on more and more flights have a successful outcome
- CCAFS SLC 40 has the most amount of all launches, more than 50%
- VAFB SLC 4E and KSC LC 39A have a higher success rate than CCAFS SLC 40

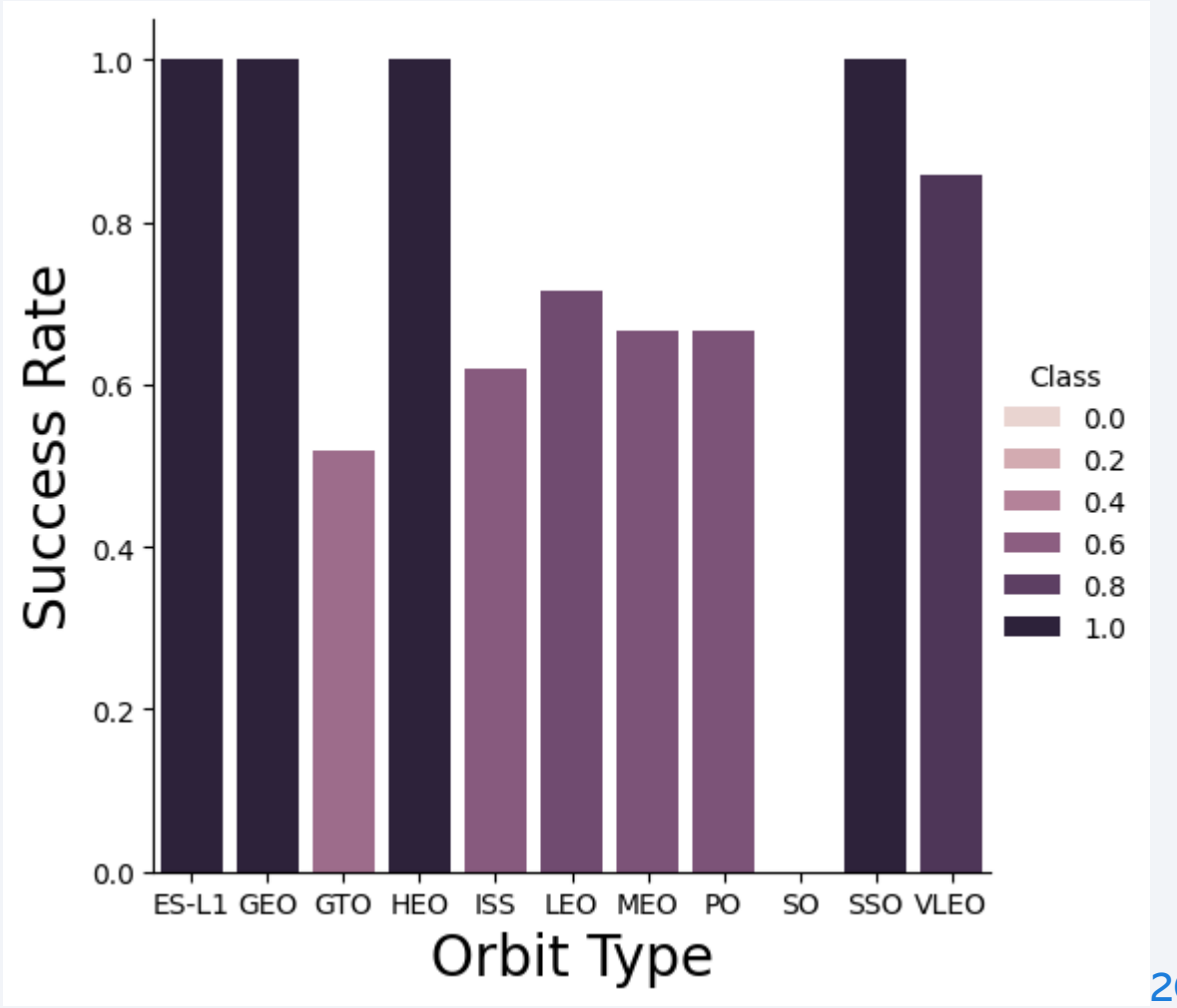
Payload vs. Launch Site

- Heavier payloads are launched from CCAFS SLC 40
- Most launches are carried out with a payload mass lighter than 7500 kg

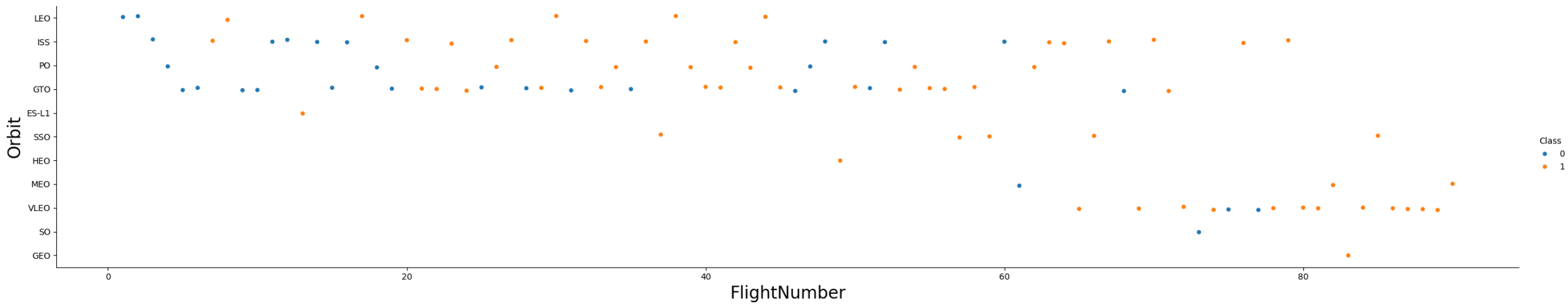


Success Rate vs. Orbit Type

- Orbits with 100% success rate: ES-L1, GEO, HEO, SSO
- Orbits with success rate between 50% and 85%: GTO, ISS, LEO, MEO, PO
- Orbits with 0% success rate: - SO



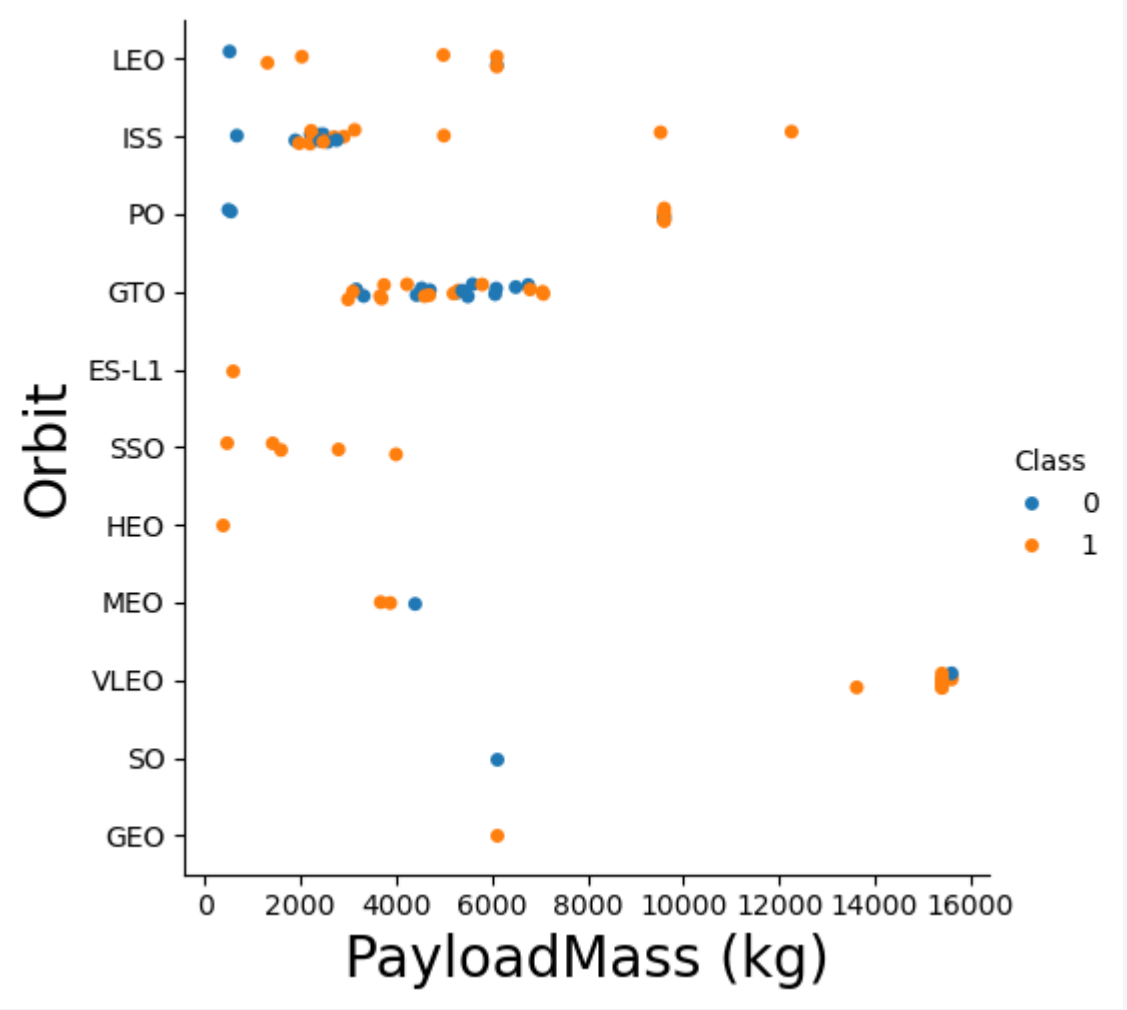
Flight Number vs. Orbit Type



- No relationship between flight number and Orbit Type

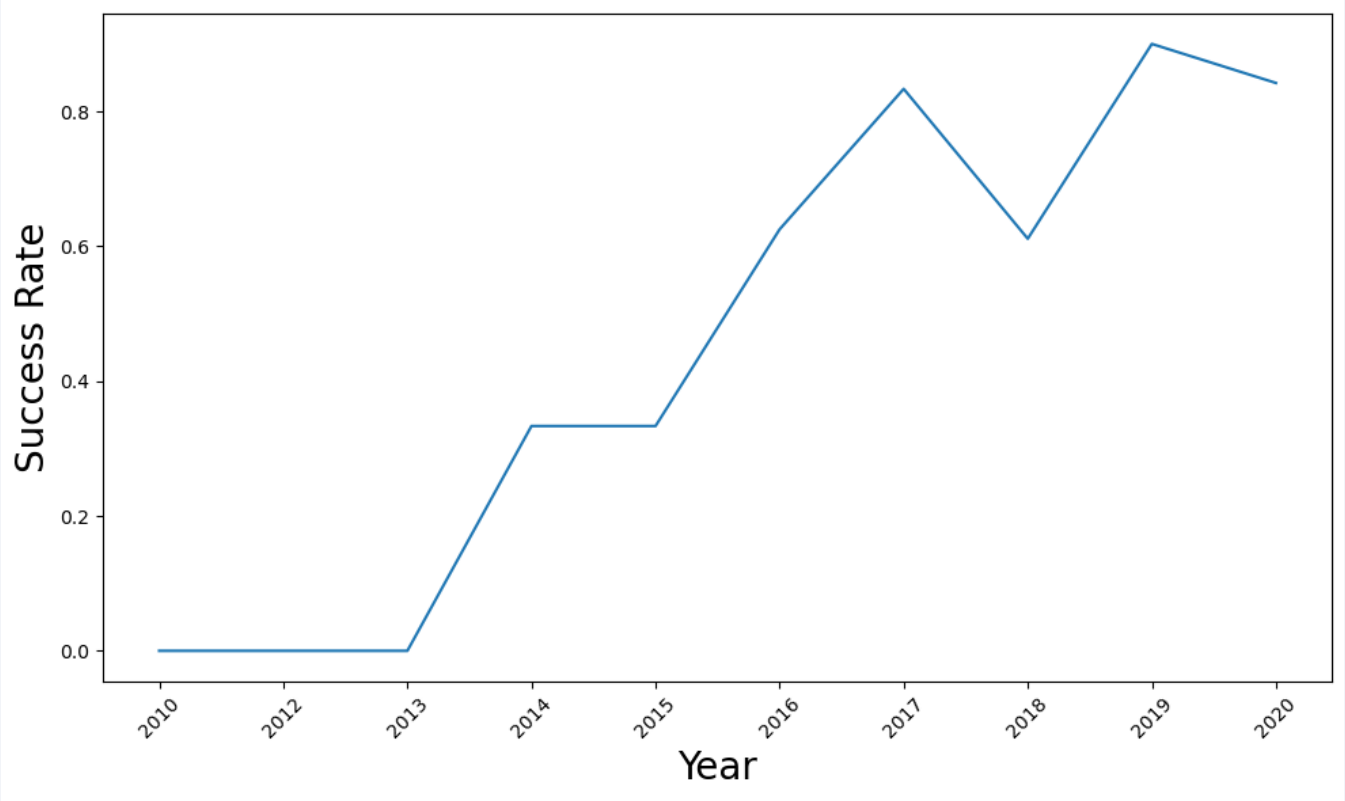
Payload vs. Orbit Type

- Heavier payloads tend to negatively influence GTO Orbit launches and positively influence ISS and LEO and PO Orbits.



Launch Success Yearly Trend

- As the year progresses the success rate increases.



All Launch Site Names

- Theses are the unique launch sites from the dataset.

```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTABLE;
```

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

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

Launch Site Names Begin with 'CCA'

- 5 records where the launch site begins with “CCA”

```
%sql SELECT  * FROM SPACEXTABLE 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

Total Payload Mass

- The calculated total payload carried by boosters from NASA is 45596 kg

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)';
```

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here
- The calculated average payload mass carried by booster version F9 v1.1 is 2534.66 kg

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version like '%F9 v1.1%';
```

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

AVG(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

- The first successful ground Landing was 2015-12-22

```
%sql SELECT Date FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)' ORDER BY Date ASC LIMIT 1;
```

✓ 0.0s

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

Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- The following boosters have successfully landed on drone ships and had payload mass greater than 4000 but less than 6000:

```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_    > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

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

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

Total Number of Successful and Failure Mission Outcomes

- Count function to count the outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) FROM SPACEXTABLE  WHERE Mission_Outcome LIKE '%Success%';
```

✓ 0.0s

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

Mission_Outcome	COUNT(Mission_Outcome)
Success	100

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) FROM SPACEXTABLE  WHERE Mission_Outcome LIKE '%Failure%';
```

✓ 0.0s

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

Mission_Outcome	COUNT(Mission_Outcome)
Failure (in flight)	1

Boosters Carried Maximum Payload

- Using a subquery in the Where clause and the max function

```
%sql SELECT booster_version FROM SPACEXTABLE WHERE payload_mass__kg_ = (SELECT max(payload_mass__kg_) FROM SPACEXTABLE)
```

```
 * 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

2015 Launch Records

```
%%sql
SELECT strftime('%m', date) AS month, date, Booster_Version, Launch_Site, Landing_Outcome
FROM SPACEXTABLE WHERE  Landing_Outcome = 'Failure (drone ship)' AND strftime('%Y', date) = '2015';
```

```
 * 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)

- strftime('%m', date) extracts the month from the date column
- strftime('%Y', date) extracts the year from the date column

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

The query groups the results by Landing_Outcome and sorts them by the count in descending order.

```
%%sql select Landing_Outcome, count(*) as count_outcomes from SPACEXTABLE
where date between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by count_outcomes desc;
```

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

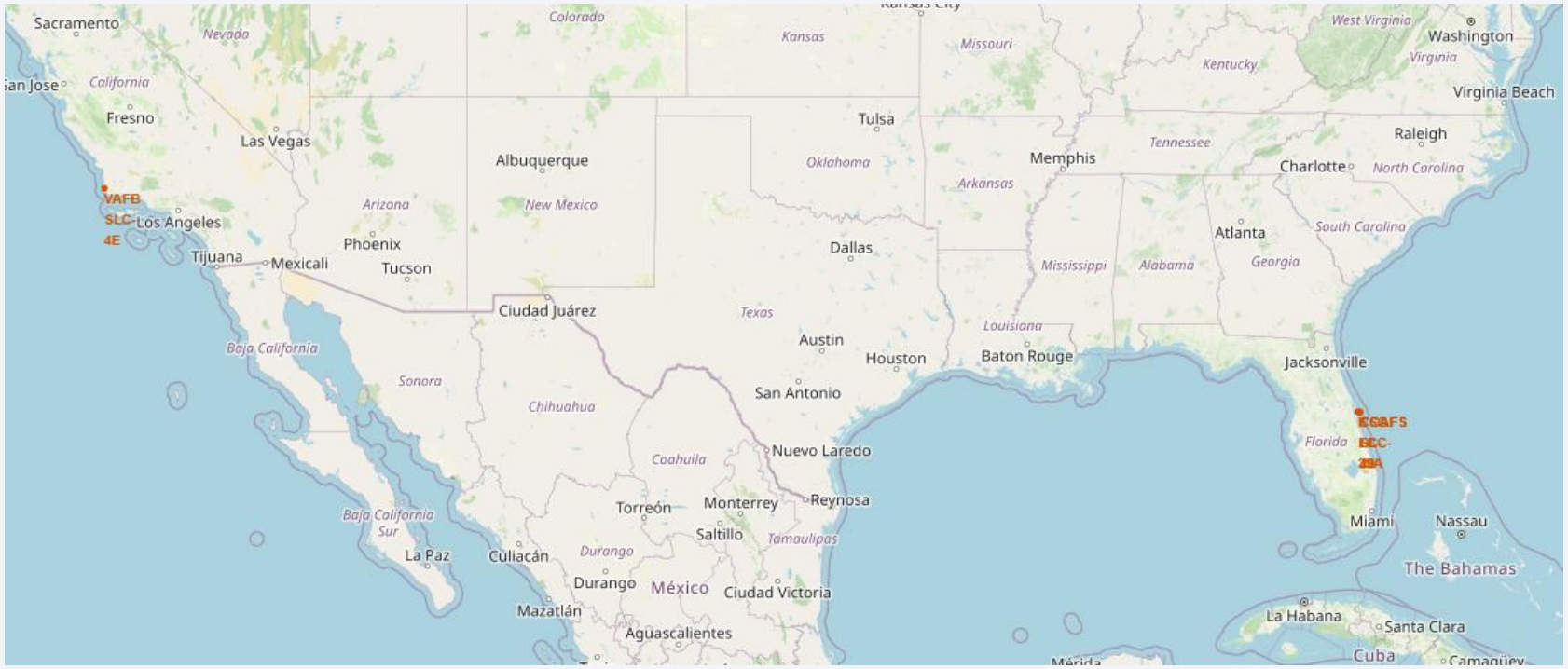
Landing_Outcome	count_outcomes
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

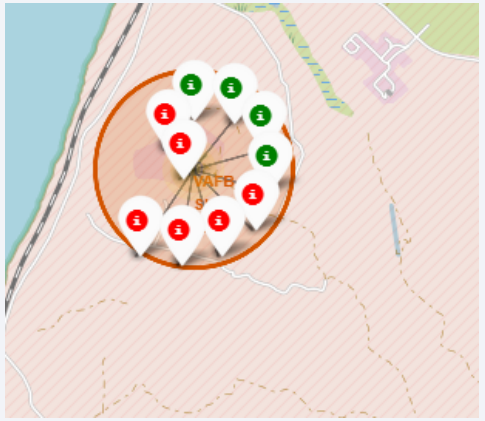
Location of all SpaceX launch sites

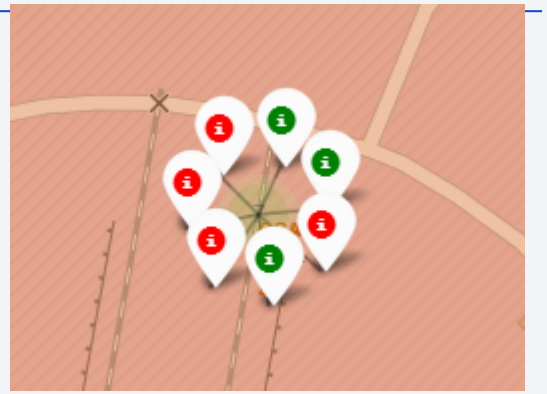
- They are rather in the south of the US then in the north part of the country.

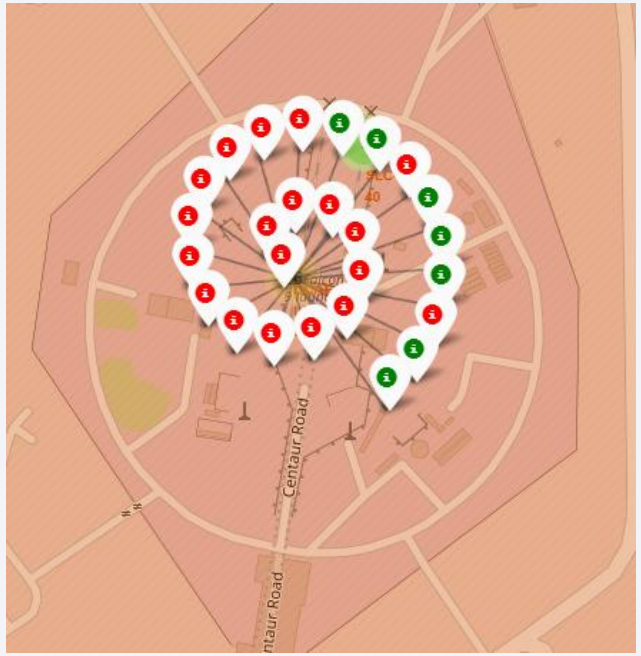


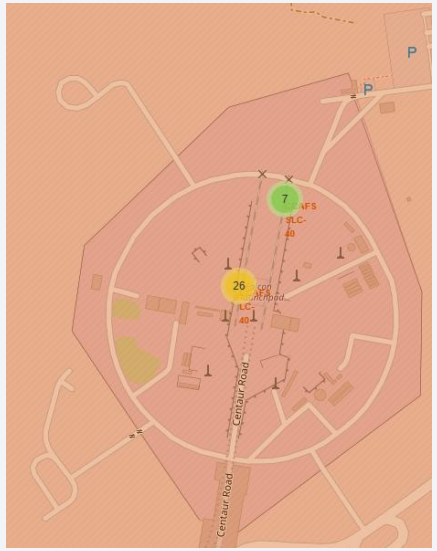
Landing outcomes from the launch sites

- The most failed launch outcomes
- are in Florida: CCAFS LC-40







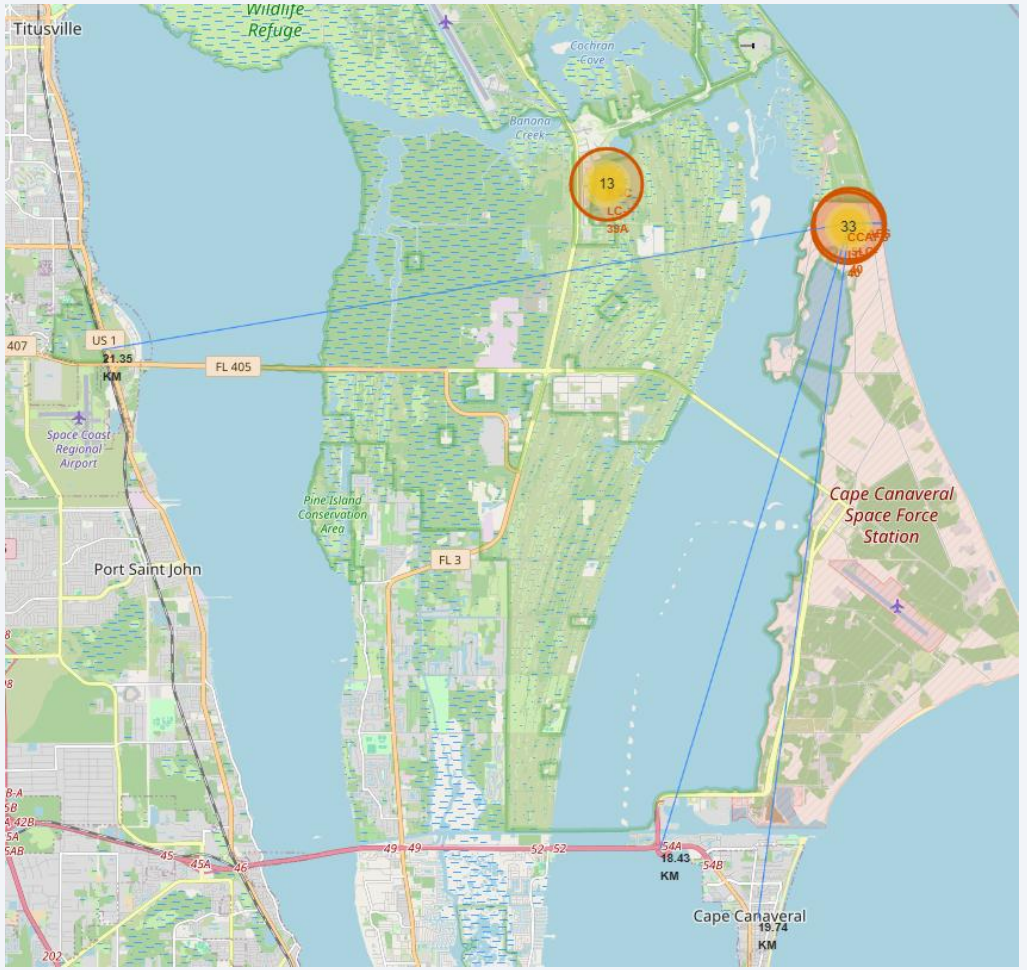


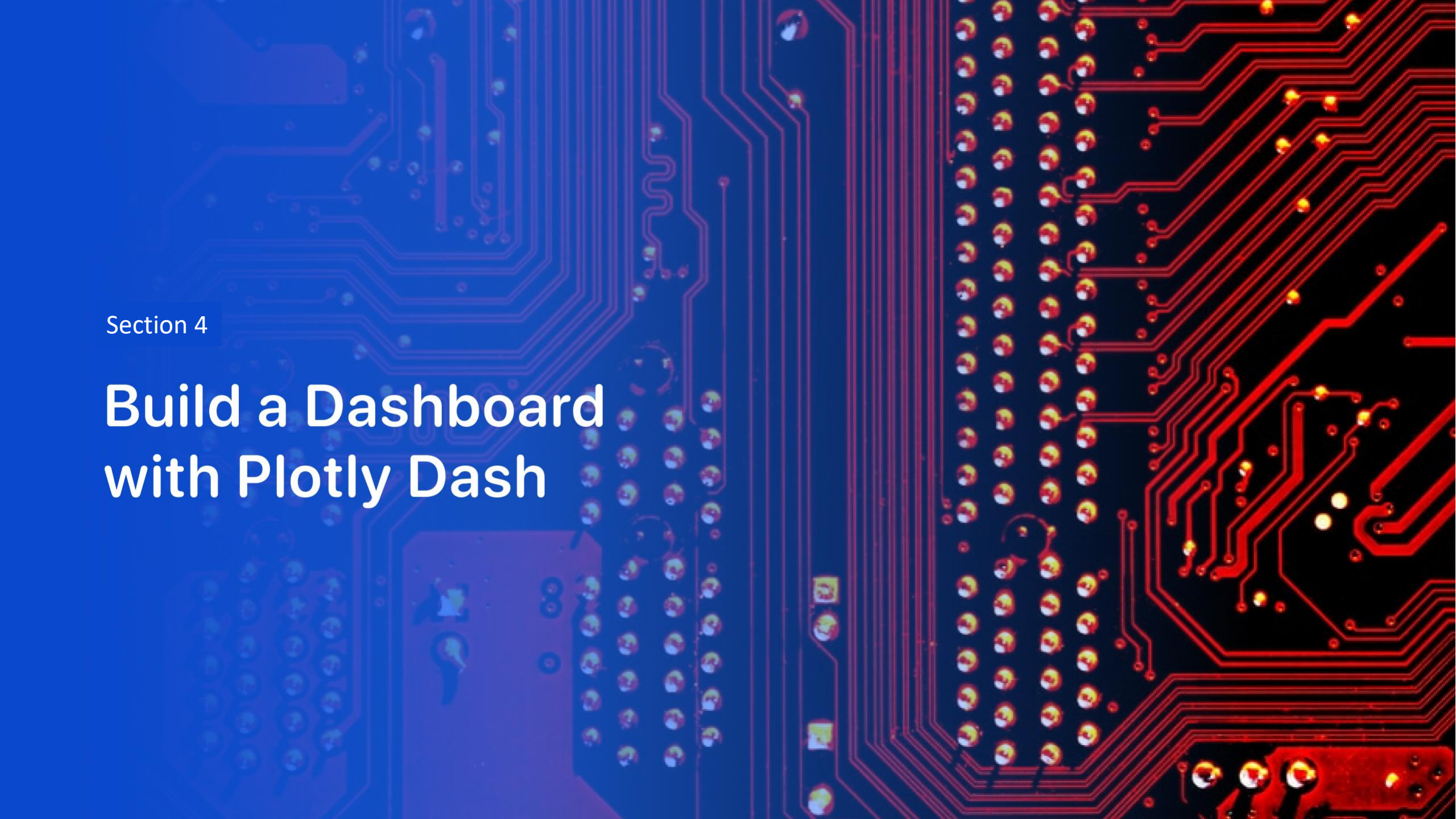
Green: Success

Red: Failure

SpaceX Launch Site Proximity

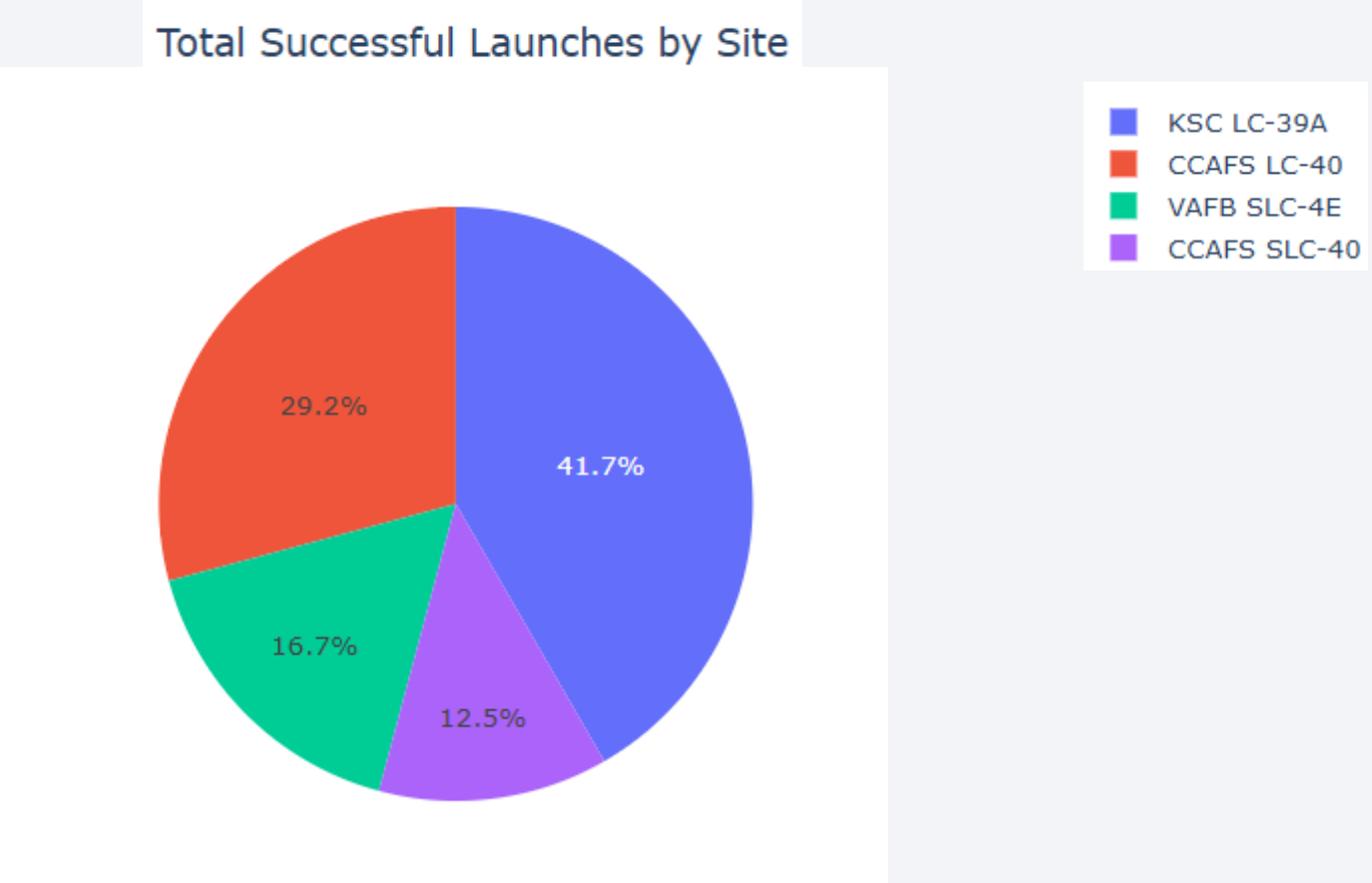
- The Distance to the coastline is 1 km.
- The Highway is 19 km away.
- The nearest city is Cape Canaveral about 19 km away.
- A railway-track is 21 km away.




Section 4
Build a Dashboard with Plotly Dash

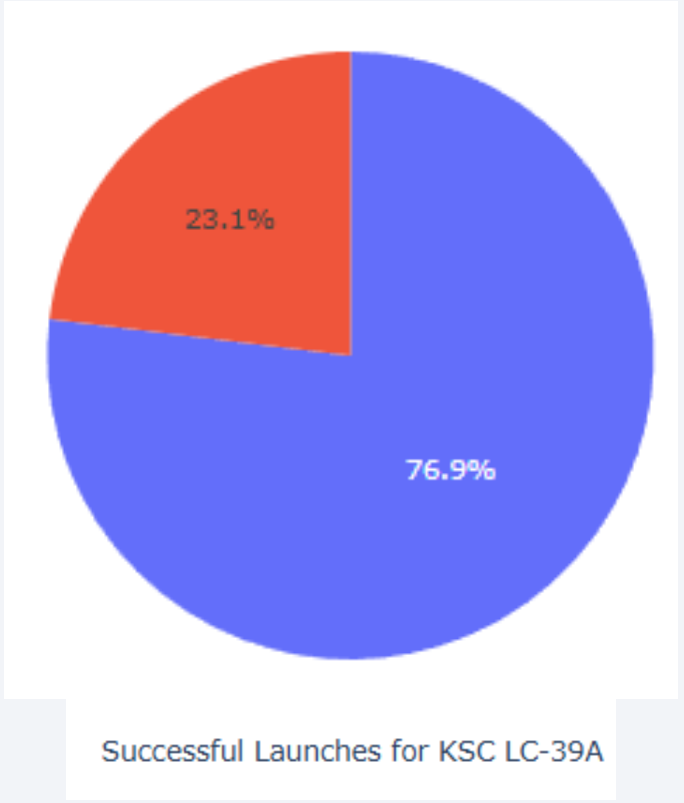
Successful launches by Launch Site

- KSC LC-39A has highest rate of success



Ratio of successful and failed launches for KSC LC-39A

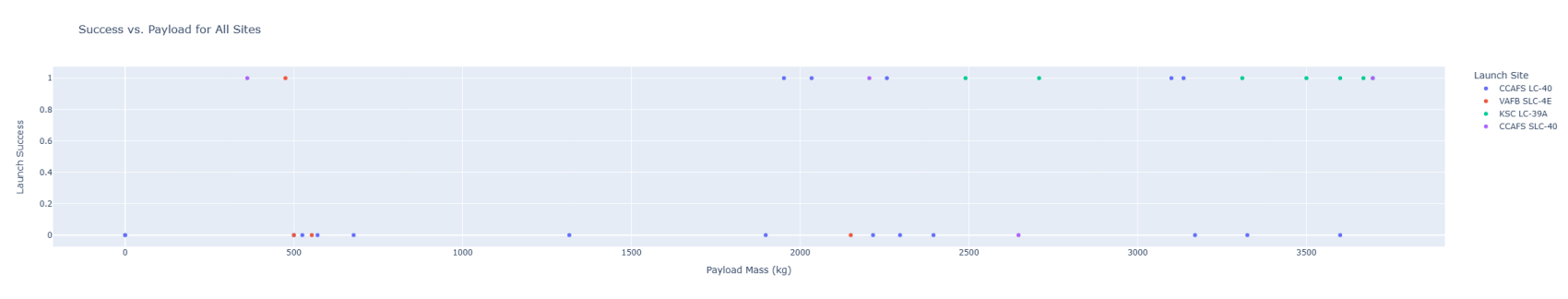
- 76.9 % of launches are successful
- 23.1 % of launches failed



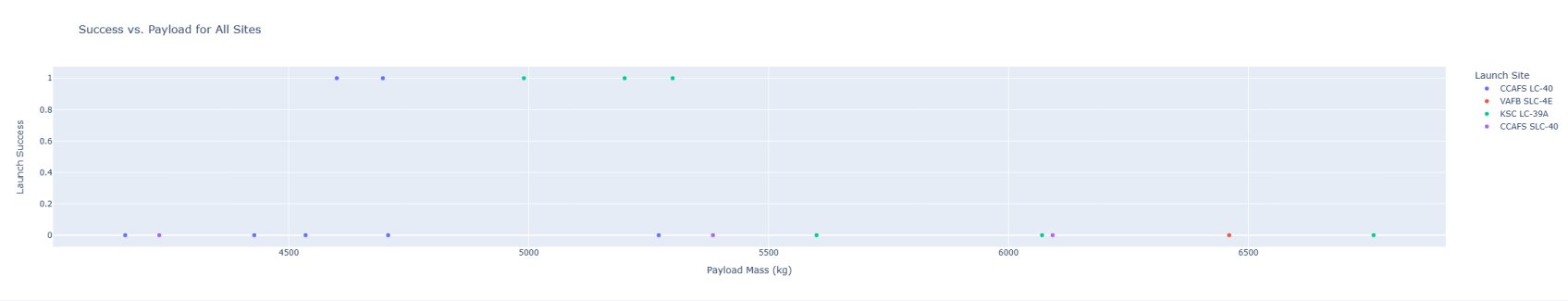
Payload mass against launch outcome

The success rate for lower weighted rockets is higher than heavy weighted rockets.

Payload: 0 – 4000kg



Payload: 4000 – 9000 kg

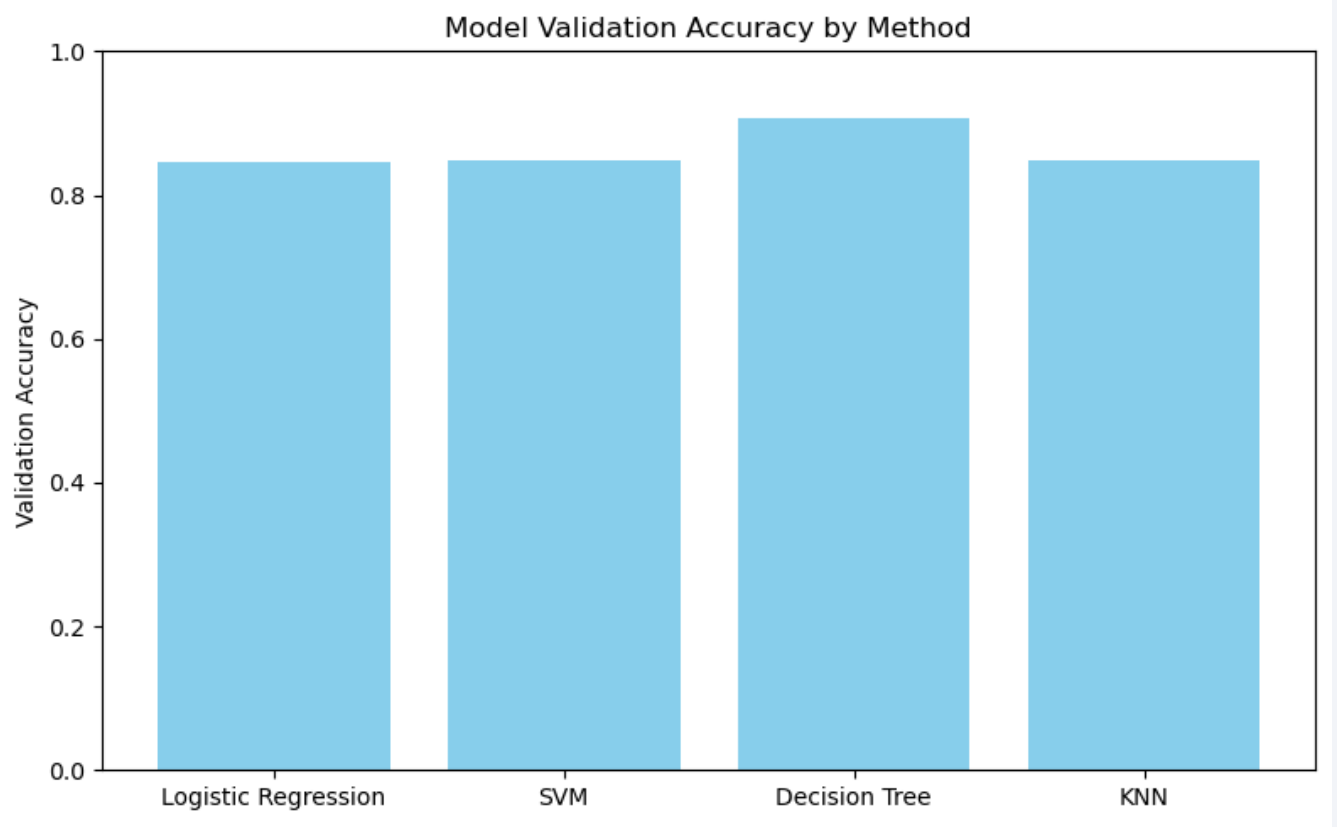


Section 5

Predictive Analysis (Classification)

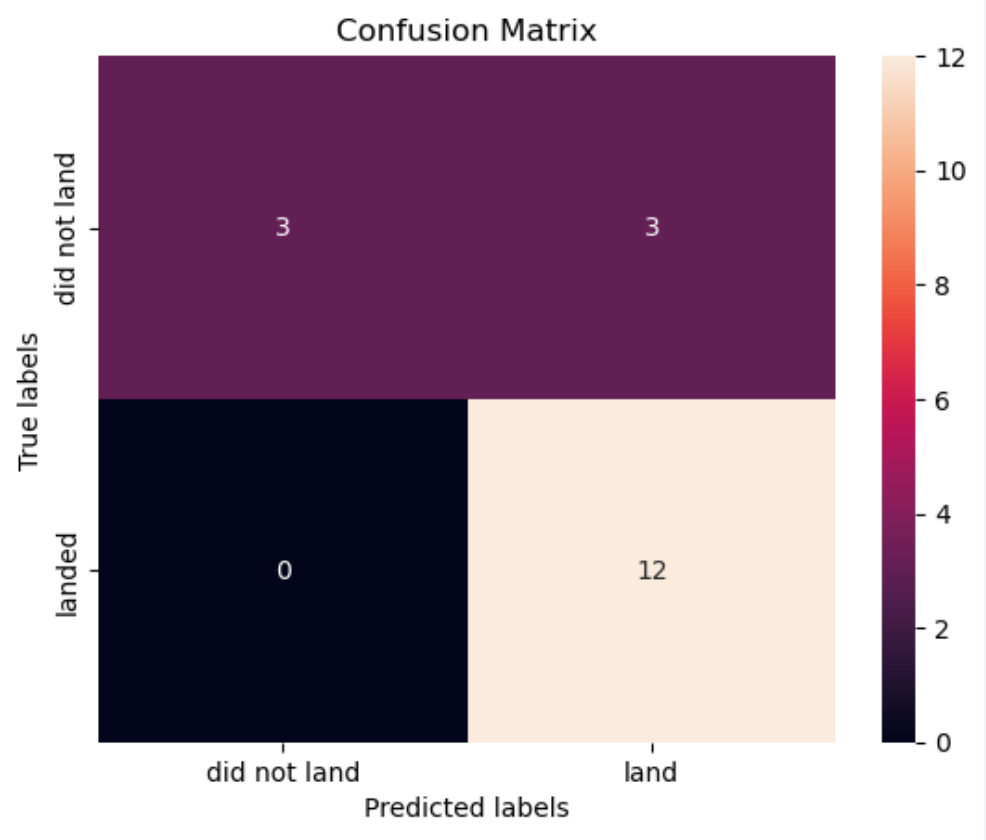
Classification Accuracy

- Decision Tree has the highest Model Validation Accuracy.



Confusion Matrix

- The confusion matrix for the decision tree has the best results.
- A major problem with this model are the false positives.



Conclusions

- A Decision Tree Model is the best choice for this particular dataset.
- Launches with a low payload show better outcomes that launches with a rather higher payload mass.
- The success rate increases over the years.
- All launch sites are located I proximity to the Equator with close proximity to the sea.
- Orbits ES-L1, GEO, HEO and SSO had a 100 % success rate.
- KSC LC-39A has highest rate of success of any launch sites.

Thank you!