IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Mohammed Khalil
May 19th, 2025

# Outline

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

# Executive Summary

- Methodology:
  - Data collection from SpaceX API and web scrapping from Wikipedia
  - Exploratory data analysis using visualization and SQL
  - Interactive visual analytics using Folium and Dash
  - Predictive analysis using machine learning classification models
- Summary of main results:
  - The data collected were enough to draw conclusions and build a model
  - The success rate increases with time and is higher for certain orbit types and booster versions
  - The best model is a decision tree classifier with 87.5% accuracy

# Introduction

- Rocket launches can cost around 165 million dollars
- SpaceX reuses the first stage to reduce the cost to around 62 million
- We need to determine if the first stage will land to determine the cost of a launch

Section 1

# Methodology

# Methodology

## Executive Summary

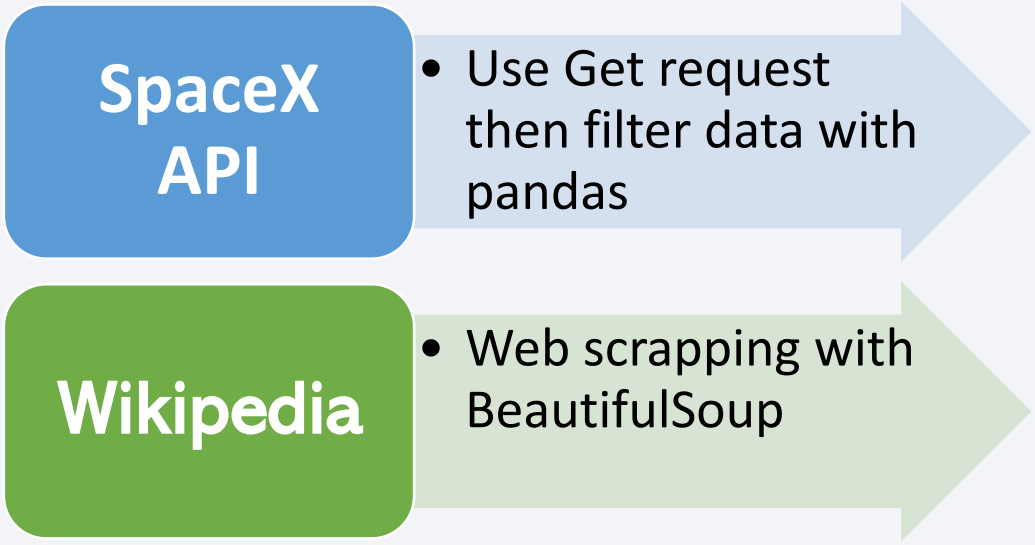
- Data collection methodology:
  - Obtain data directly from the SpaceX API
  - Scrape data from Wikipedia using BeautifulSoup
- Perform data wrangling
  - Use pandas in Python to determine the labels for training supervised models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Standardize data, split into training and testing data, build SVM, Classification Trees and Logistic Regression models, evaluate the accuracy of each model

# Data Collection

- Dataset sources:
  - SpaceX API for the rocket data, launchpad, payloads, and cores

    https://api.spacexdata.com/v4/launches/past
  - Wiki web scrapping for each launch and the outcome (success/ failure)

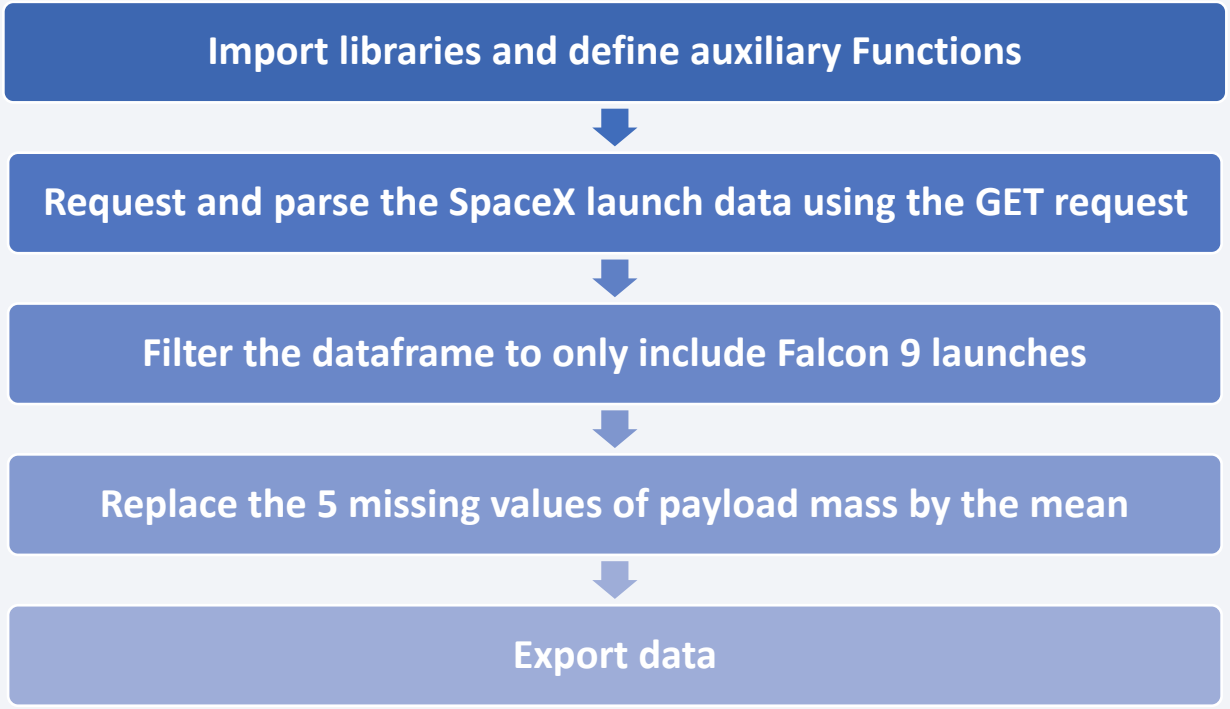
    https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches



# Data Collection – SpaceX API

- Requested data from the SpaceX API using a GET request in Python
- Cleaned data to include only Falcon 9 launches and replace missing values
- GitHub URL for notebook:

https://github.com/MKhalil-DS/DScapstone/blob/main/1-jupyter-labs-spacex-data-collection-api-v2.ipynb

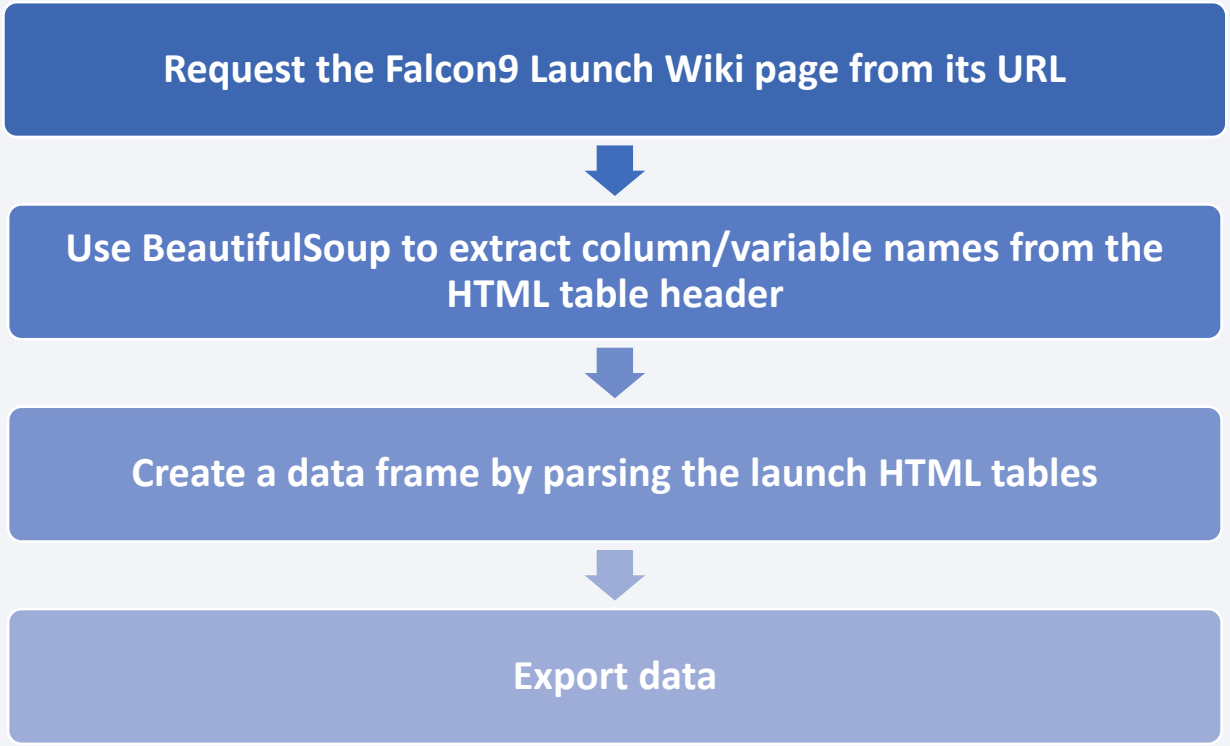


# Data Collection - Scraping

- Extract a Falcon 9 launch records HTML table from Wikipedia
- Parse the table and convert it into a Pandas data frame

- GitHub URL for notebook:

https://github.com/MKhalil-DS/DScapstone/blob/main/2-jupyter-labs-webscraping.ipynb

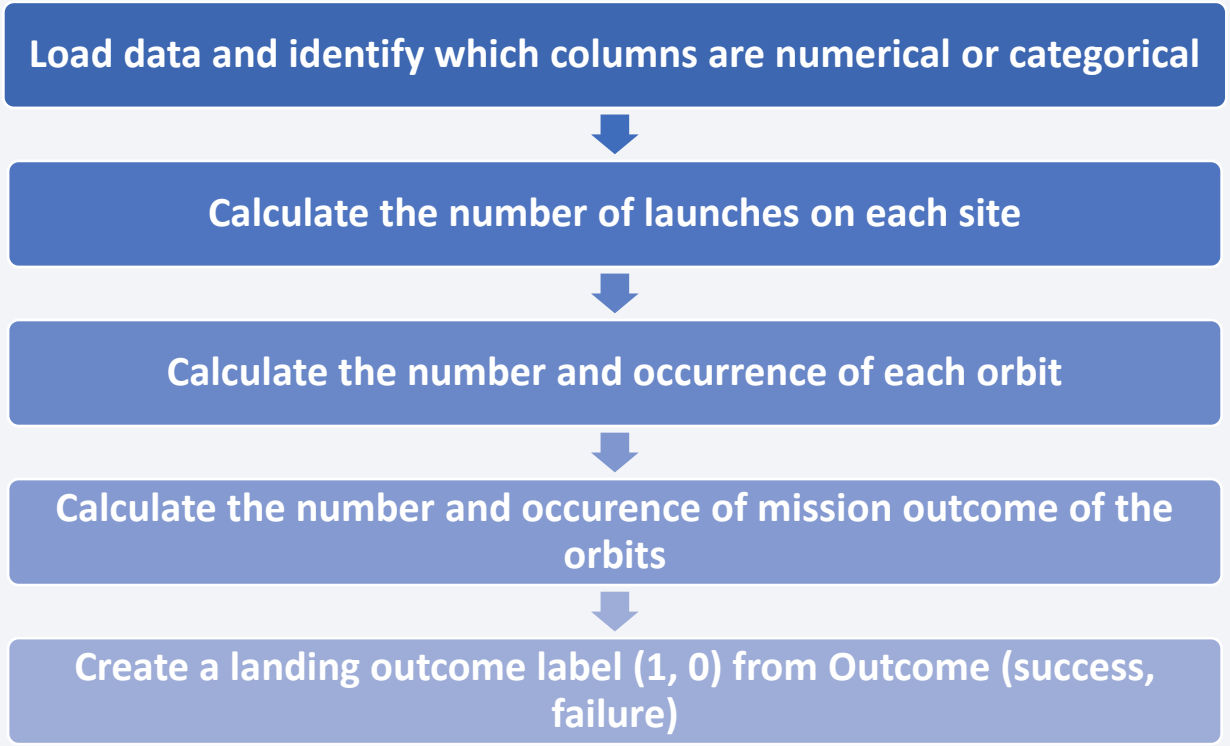


# Data Wrangling

- Use pandas in Python to process data
- Perform exploratory data analysis
- Determine training labels, including adding a ‘class’ column to represent the outcome (success/failure)

- GitHub URL for notebook:

https://github.com/MKhalil-DS/DScapstone/blob/main/3-labs-jupyter-spacex-Data%20wrangling-v2.ipynb



# EDA with Data Visualization

- Performed exploratory Data Analysis using Pandas, Matplotlib, and seaborn.
- Produced the following visualizations:
  - Scatter point chart of FlightNumber vs. PayloadMass overlaying the outcome, to see if mass is important in determining the outcome
  - Scatter point chart of visualize the relationship between FlightNumber and LaunchSite
  - Bar plot of success rate for each orbit to find the relationship between them
  - Scatter point chart to visualize the relationship between FlightNumber and Orbit type
  - Scatter point chart to visualize the relationship between PayloadMass and Orbit type
  - Line plot to visualize the launch success yearly trend (success rate vs year)
- Performed Feature Engineering to determine which variables are important in modeling the outcome, then cast categorical variables to numerical using one-hot encoding.
- GitHub URL for notebook: https://github.com/MKhalil-DS/DScapstone/blob/main/5-jupyter-labs-eda-dataviz-v2.ipynb

# EDA with SQL

- Performed some SQL queries to understand the dataset
  1. Display the names of the unique launch sites in the space mission
  2. Display 5 records where launch sites begin with the string 'CCA'
  3. Display the total payload mass carried by boosters launched by NASA (CRS)
  4. Display average payload mass carried by booster version F9 v1.1
  5. List the date when the first succesful landing outcome in ground pad was achieved
  6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  7. List the total number of successful and failure mission outcomes
  8. List all the booster_versions that have carried the maximum payload mass.
  9. List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015
  10. Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.
- GitHub URL for notebook: https://github.com/MKhalil-DS/DScapstone/blob/main/4-jupyter-labs-eda-sql-coursera_sqllite.ipynb

# Build an Interactive Map with Folium

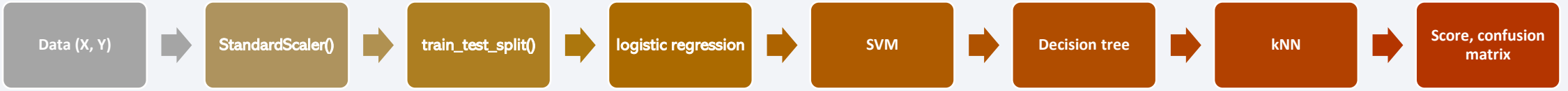
- Launch success rate could depend on the location of the launch site and its proximities.
- To discover which factors might affect the choice of a launch site, we use Folium to add the following markers to a map:
  - Marked all launch sites on a map with circles
  - Marked the success/failed launches for each site on the map using MarkerCluster
  - Calculated the distances between a launch site to some of its proximities (city, highway, …)
- GitHub URL for notebook: https://github.com/MKhalil-DS/DScapstone/blob/main/6-lab-jupyter-launch-site-location-v2.ipynb

# Build a Dashboard with Plotly Dash

- Built an interactive Dashboard with Plotly Dash containing the following:
  - Dropdown menu with launch sites, including an option for all sites
  - Pie chart for the success rate for all sites and one for each site
  - Range slider for payload mass
  - Scatter chart for the outcome vs payload mass for the mass in the range slider for each site
- These charts show which sites and payload ranges had the largest success rate
- GitHub URL for notebook: https://github.com/MKhalil-DS/DScapstone/blob/main/7-dash_spacex.ipynb

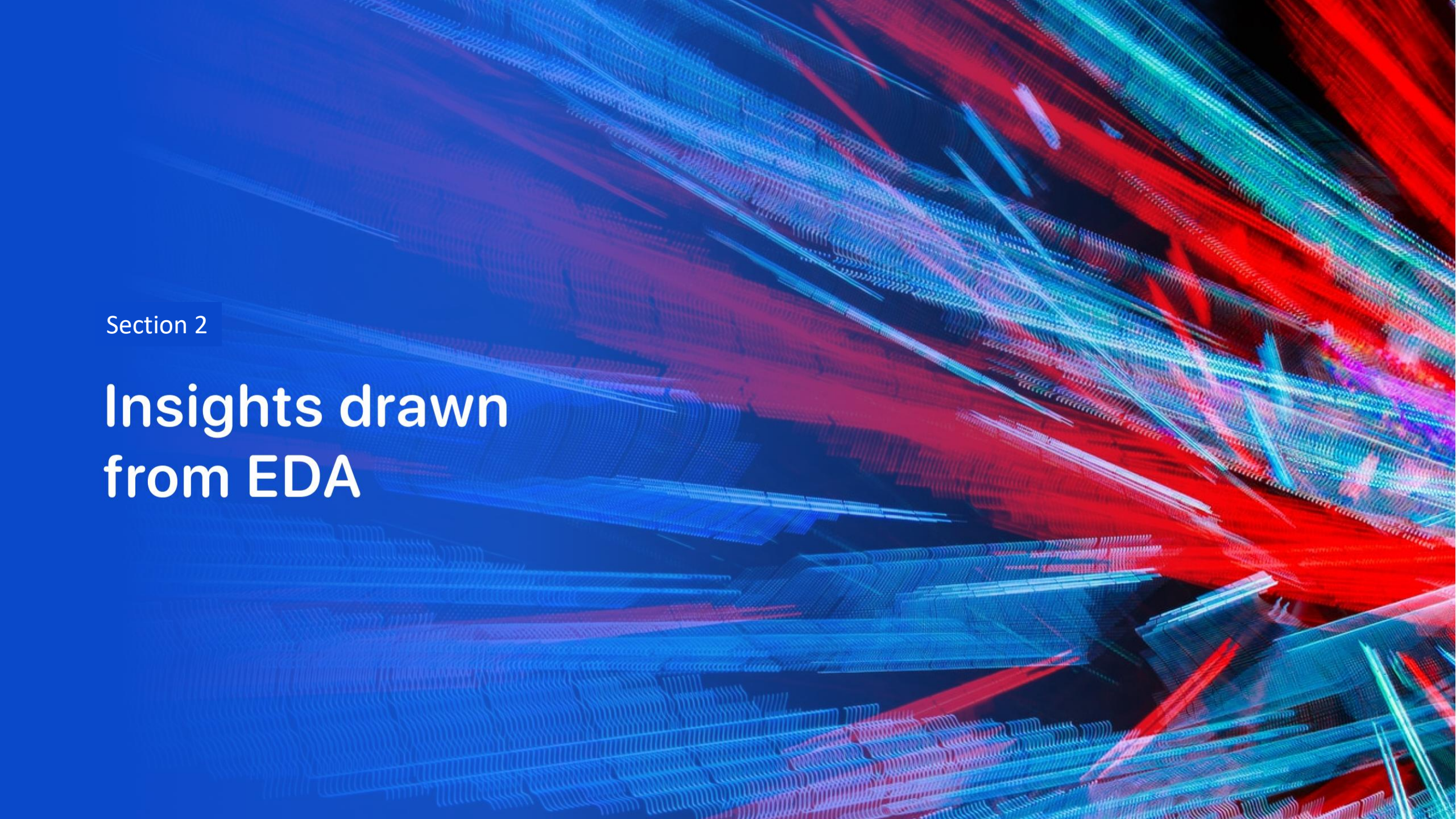
# Predictive Analysis (Classification)

- Built predictive models for the launch outcome using several machine learning algorithms
- The data were normalized using StandardScaler(), then split into training and testing subsets using the function train_test_split()
- Created a logistic regression object and a GridSearchCV object to find the best parameters
- Repeated the steps for SVM, Classification Trees and k Nearest Neighbors
- For each model, displayed the best parameters and best score, and plotted the confusion matrix
- The decision tree classifier performed best with an accuracy 87.5%
- GitHub URL for notebook: https://github.com/MKhalil-DS/DScapstone/blob/main/8-SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb



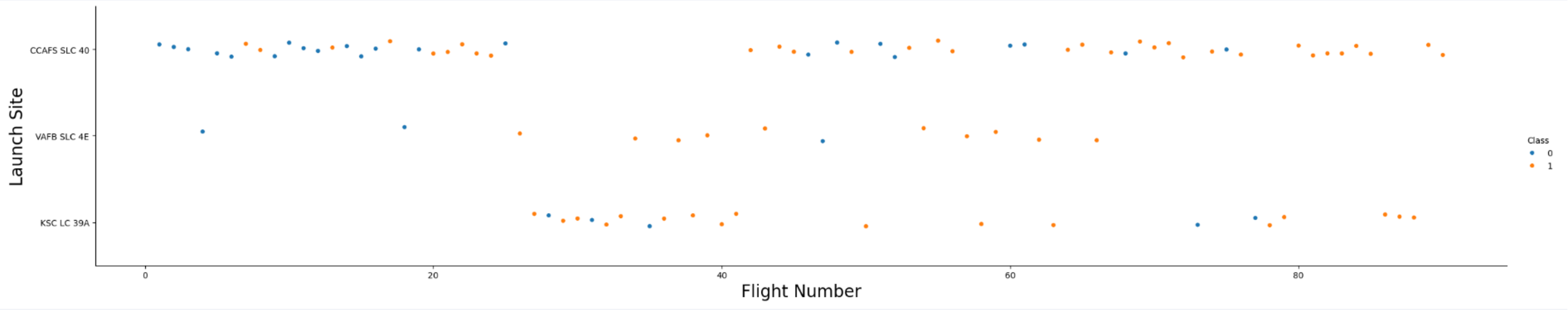
# Results

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


Section 2
Insights drawn from EDA

# Flight Number vs. Launch Site

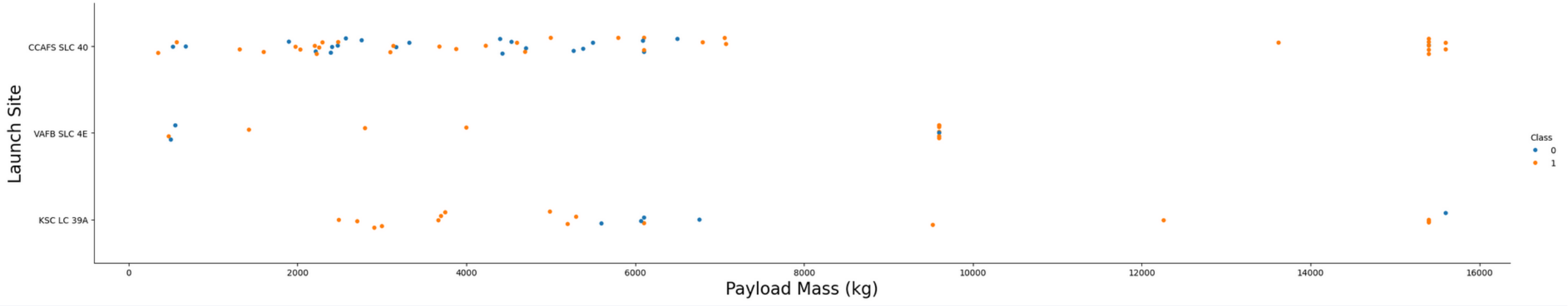
Scatter plot of Flight Number vs. Launch Site




- Success rate increases with flight number
- First 25 launches were mostly launched from CCAFS SLC 40
- The site VAFB SLC 4E was not used after flight number 66
- Launch from site KSC LC 39A started with flight number 27

# Payload vs. Launch Site

Scatter plot of Payload Mass vs. Launch Site

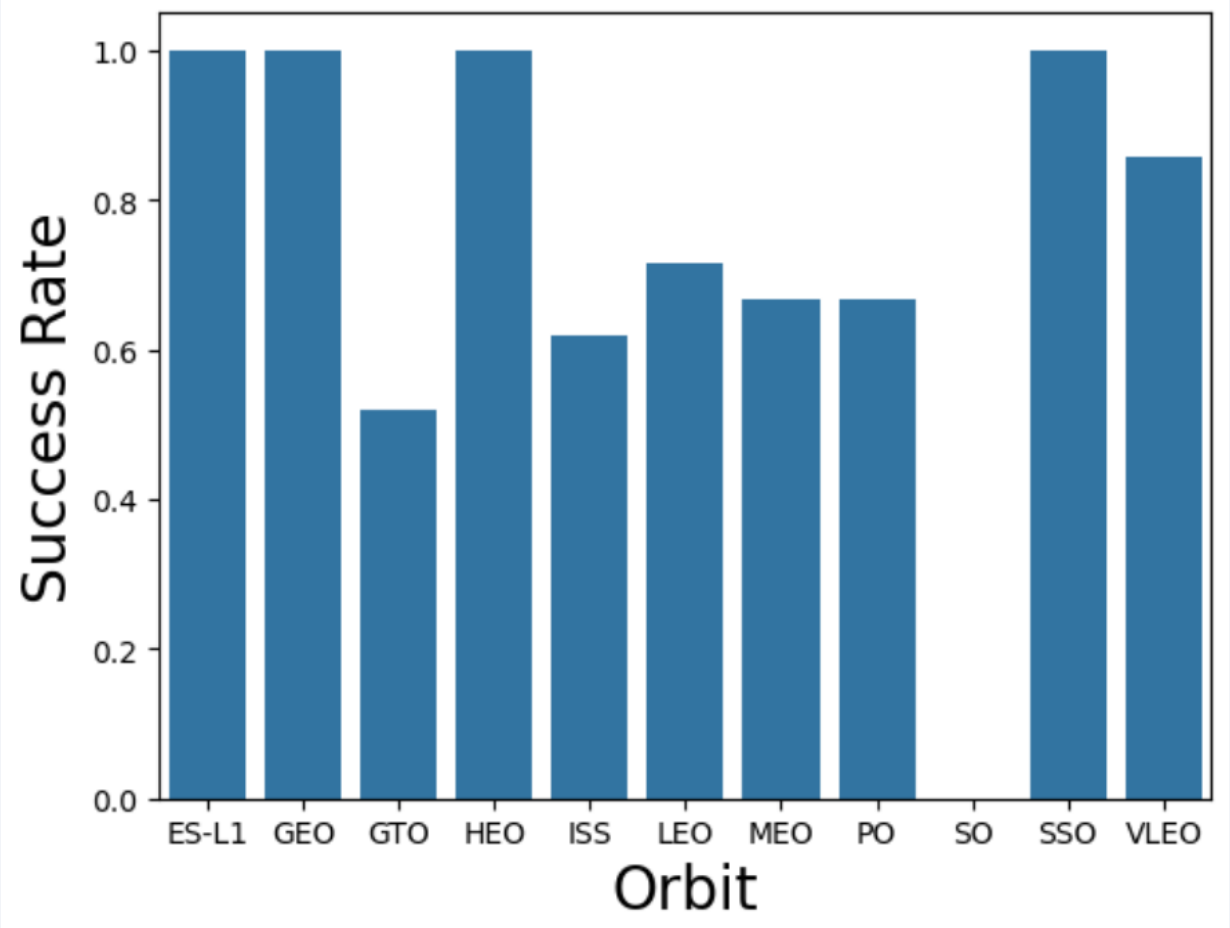



- For the VAFB-SLC launch site there are no rockets launched for heavy payload mass (greater than 10000)

# Success Rate vs. Orbit Type

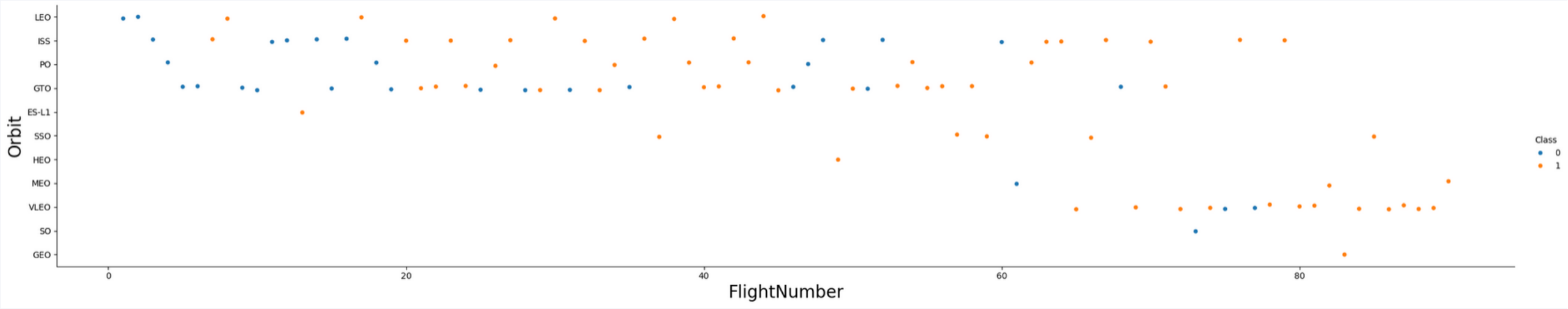
Bar chart for the success rate of each orbit type

- The success rate is high for orbits (ES-L1, GEO, HEO, SSO, VLEO)
- The success is lower for the other orbits (GTO, ISS, LEO, MEO, PO)



# Flight Number vs. Orbit Type

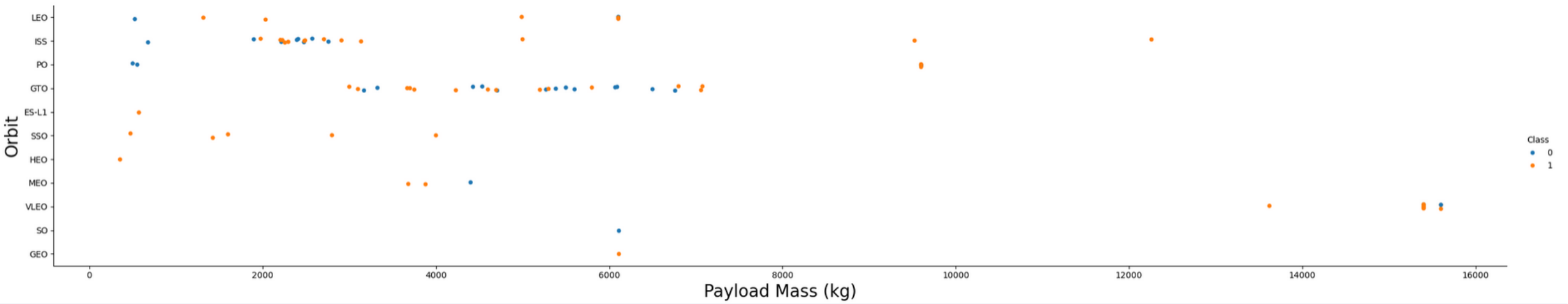
Scatter plot of Flight Number vs. Orbit Type




- For the LEO orbit, the Success appears related to the number of flights
- There seems to be no relationship between flight number when in GTO orbit
- Launches to orbits GEO, SO, VLEO, MEO, and HEO started more recently than the other orbits

# Payload vs. Orbit Type

Scatter plot of Payload Mass vs. Orbit Type

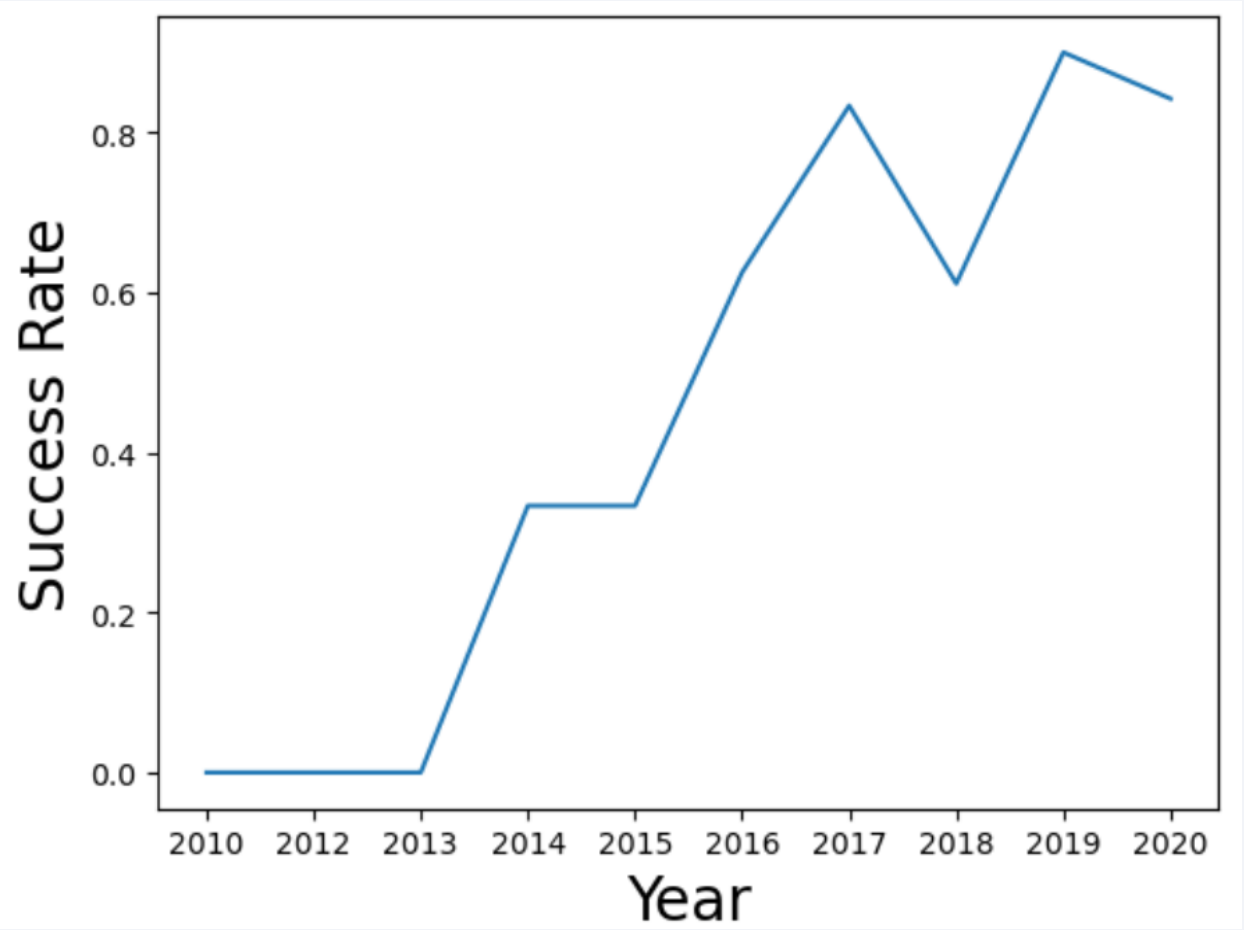



- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS
- For GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there

# Launch Success Yearly Trend

## Line plot for success rate vs year

- The success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.



# All Launch Site Names

- Names of the unique launch sites
- Obtained using the SQL query Select with Distinct on the Launch_Site column

```
%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 launch sites begin with `CCA`
- LIKE ‘CCA%’ is used to obtain the sites that begin 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 total payload carried by boosters from NASA
- Sum() is used to obtain the total mass for the selected Customer ‘NASA (CRS)’

```
%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

- Average payload mass carried by booster version F9 v1.1
- Avg() is used to obtain the average of the payload mass column for booster version F9 v1.1

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

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

| AVG(PAYLOAD_MASS__KG_) |
|---|
| 2928.4 |

# First Successful Ground Landing Date

- Date of the first successful landing outcome on ground pad
- Min() is used to obtain the first date with outcome ‘Success (ground pad)’

```
%sql SELECT MIN(Date) FROM SPACEXTABLE WHERE "Landing_Outcome" = 'Success (ground pad)'
```

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

| MIN(Date) |
| --- |
| 2015-12-22 |

# Successful Drone Ship Landing with Payload between 4000 and 6000

- Names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- To obtain masses in a range, we use Between 4000 and 6000

```
%%sql
SELECT DISTINCT "Booster_Version" FROM SPACEXTABLE WHERE
"Landing_Outcome" = 'Success (drone ship)' AND
"PAYLOAD_MASS__KG_" BETWEEN 4000 AND 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

- Total number of successful and failure mission outcomes
- Count() is used to obtain the number of successes or failures

```
%sql SELECT COUNT(*) FROM SPACEXTABLE WHERE "Mission_Outcome" LIKE 'Success%'
```

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

| COUNT(*) |
|---|
| 100 |

```
%sql SELECT COUNT(*) FROM SPACEXTABLE WHERE "Mission_Outcome" LIKE 'Failure%'
```

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

| COUNT(*) |
|---|
| 1 |

# Boosters Carried Maximum Payload

- Names of the booster versions which have carried the maximum payload mass
- Used the subquery

SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE

to find the max payload mass

```
%%sql
SELECT DISTINCT "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

- List of failed landing outcomes in drone ship, their booster versions, and launch site names in year 2015
- substr(Date, 6,2) was used to select the month, and substr(Date,0,5)='2015' for the year

```
%%sql
SELECT substr(Date,0,5) AS 'Year', substr(Date, 6,2) AS 'Month', "Booster_Version", "launch_site", "Landing_Outcome" FROM SPACEXTABLE WHERE
"Landing_Outcome" = 'Failure (drone ship)' AND substr(Date,0,5)='2015'
```

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

| Year | Month | Booster_Version | Launch_Site | Landing_Outcome |
|---|---|---|---|---|
| 2015 | 01 | F9 v1.1 B1012 | CCAFS LC-40 | Failure (drone ship) |
| 2015 | 04 | F9 v1.1 B1015 | CCAFS LC-40 | Failure (drone ship) |

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

- Rank the count of landing outcomes between the date 2010-06-04 and 2017-03-20, in descending order
- Used Group By Landing_Outcome and Order By Count(…) Desc to list them in descending order

```
%%sql
SELECT "Landing_Outcome", COUNT("Landing_Outcome") FROM SPACEXTABLE WHERE
Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing_Outcome"
ORDER BY COUNT("Landing_Outcome") DESC
```

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

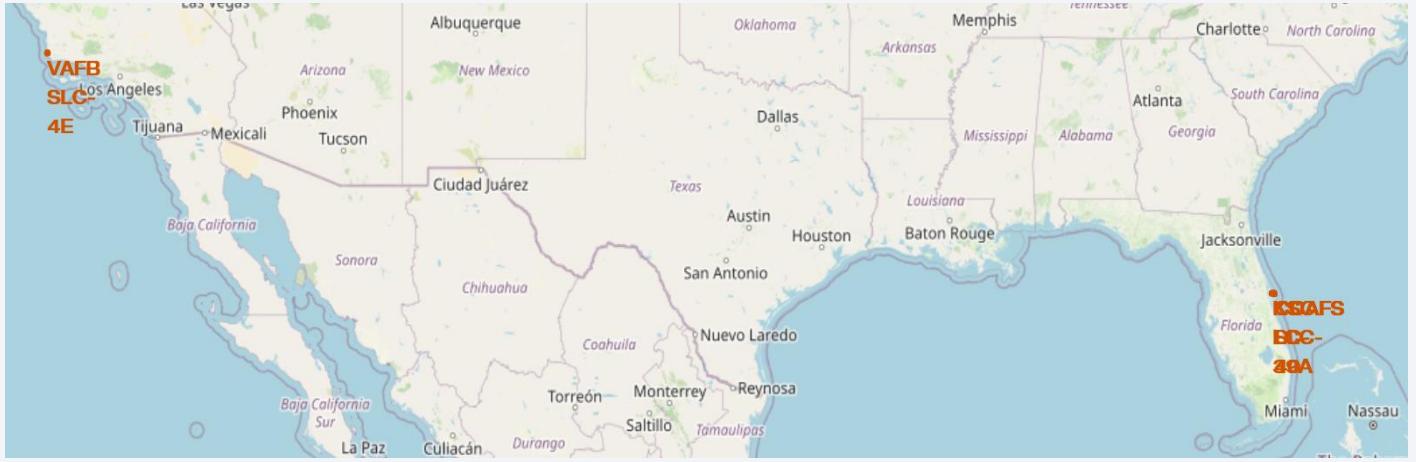
| Landing_Outcome | COUNT("Landing_Outcome") |
|---|---|
| No attempt | 10 |
| Success (drone ship) | 5 |
| Failure (drone ship) | 5 |
| Success (ground pad) | 3 |
| Controlled (ocean) | 3 |
| Uncontrolled (ocean) | 2 |
| Failure (parachute) | 2 |
| Precluded (drone ship) | 1 |

Section 3

# Launch Sites Proximities Analysis

# Launch Site Locations

- One launch site is in California near the west coast
- Three sites are in close proximity in Florida near the east coast

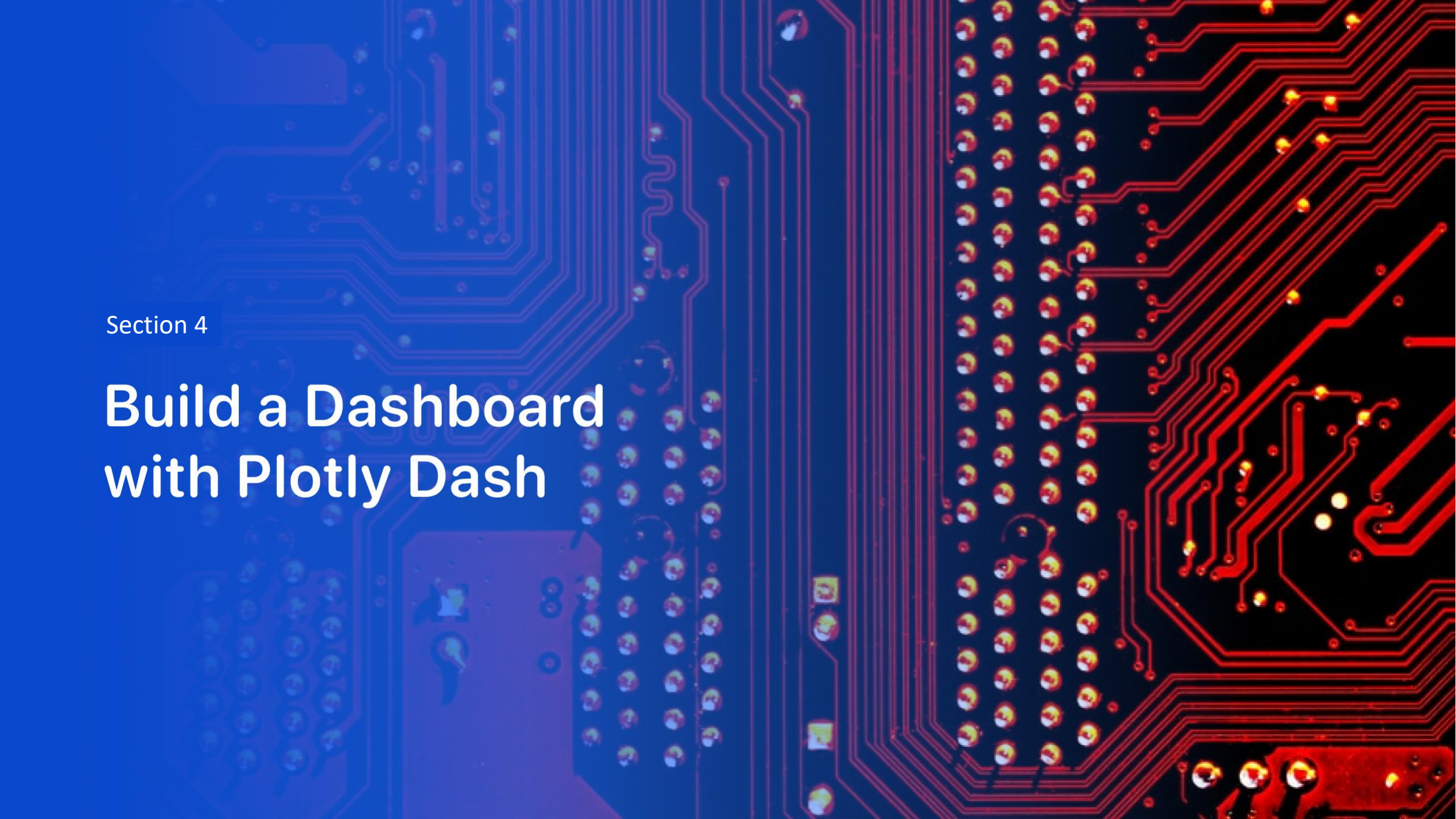



# Success/failed launches for each site

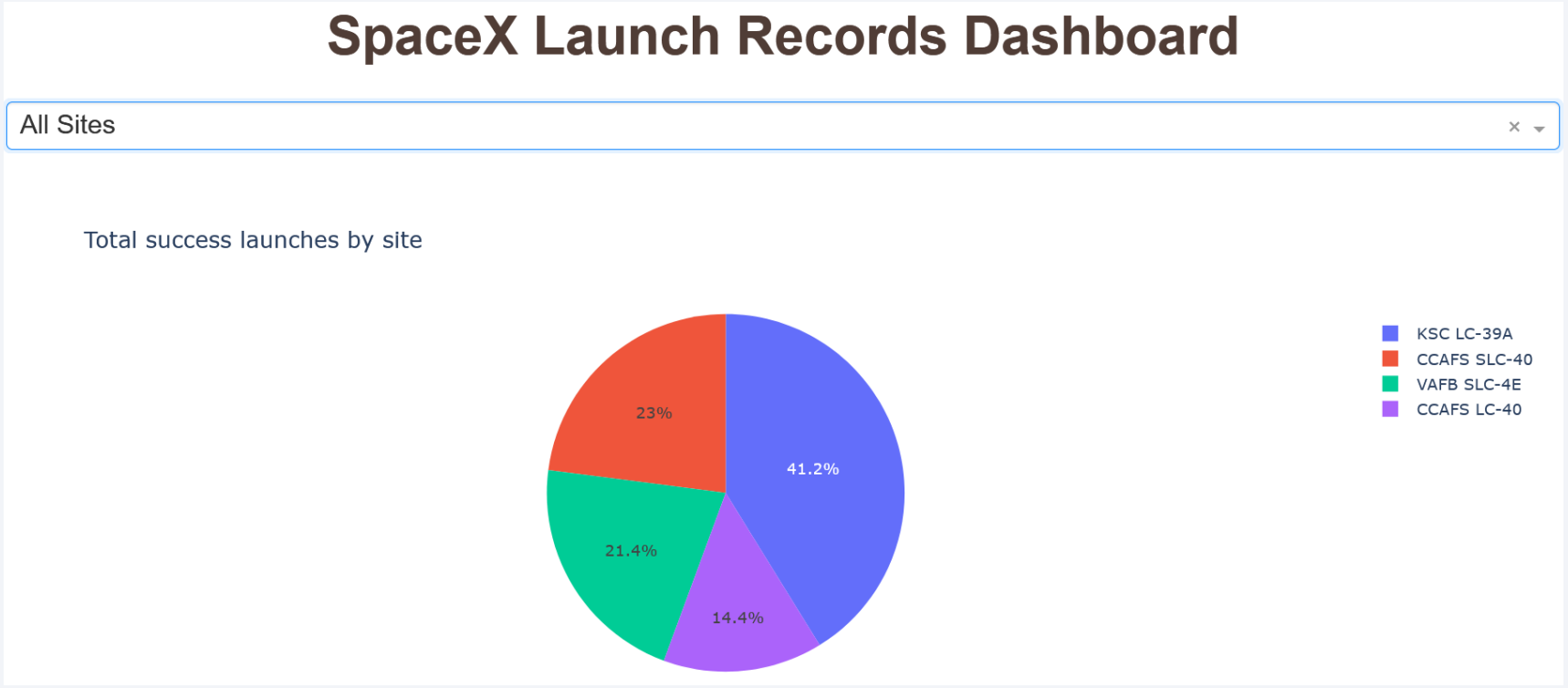

- Successful launches are presented by green markers and failures by red

# Distance between a launch site to its proximities

- Distance betwee the site CCAFS LC-40 and
  - the coast: 0.9 km
  - the city of Cape Canaveral: 19.5 km
  - Highway Samuel C Phillips: 0.7 km

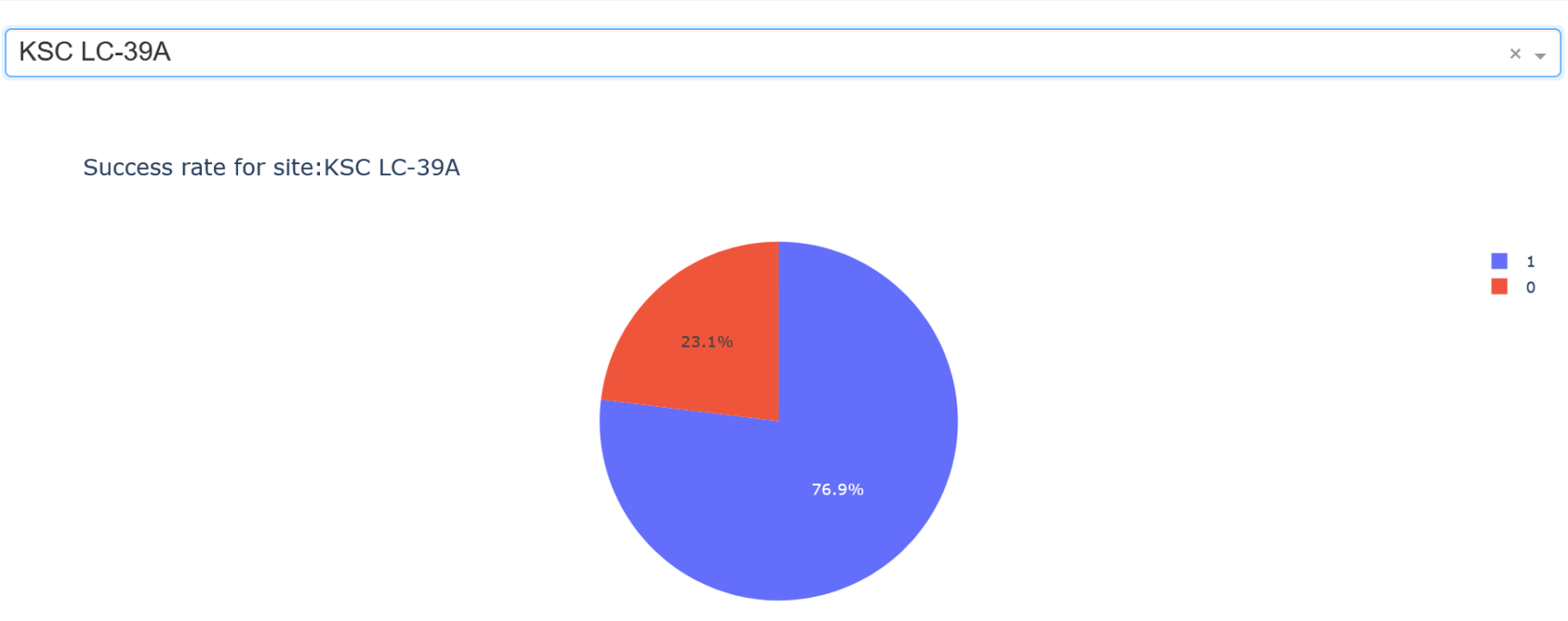

Section 4
Build a Dashboard with Plotly Dash

# Pie chart for total success launches by site




41.2% of successful launches were from the site KSC LC-39A, and smaller percentages from other sites.

# Success rate for site KSC LC-39A




Success rate for site KSC LC-39A is 76.9%, with 23.1% failure.

# Scatter plot for launch outcomes vs payload mass

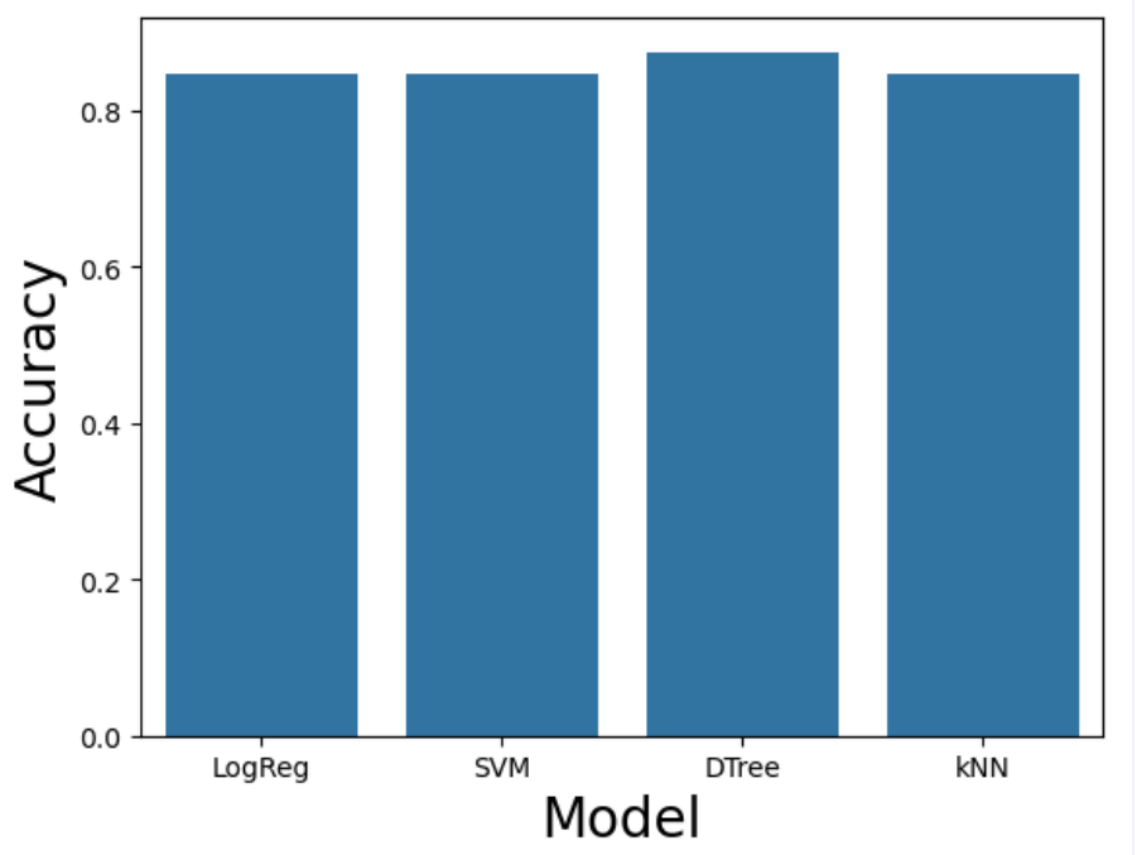
- There doesn't seem to be a correlation between payload mass and success rate
- Boosters v1.0 and v1.1 had low success rate, while booster FT has more successes
- There is no strong correlation for booster B4, and there is only one launch with booster B5


Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

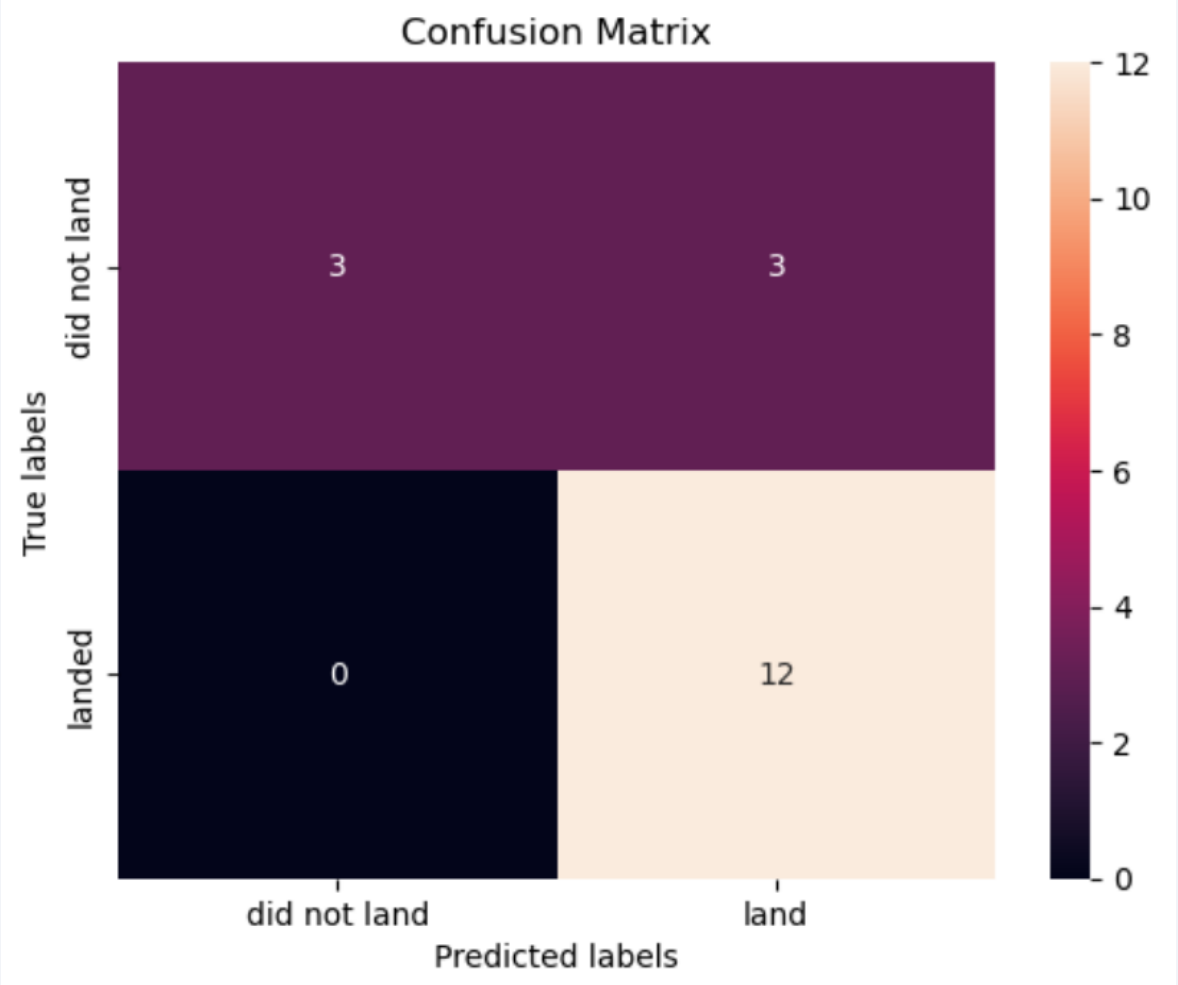
- The decision tree model has highest accuracy of 87.5%
- The other models (LogReg, SVM, and kNN) have accuracy a bit less than 85%



# Confusion Matrix

## Confusion matrix for the decision tree classifier

- has 12 true positive and 3 true negative
- has 0 false negative and 3 false positive
- leading to a score of 83.33%



# Conclusions

- Data were collected from SpaceX API and Wikipedia, which were enough to build a model
- The launch success rate increases with time / flight number and is highest for orbit types SSO and VLEO and for booster version B4
- Several machine learning classifiers were investigated. The decision tree classifier produces best accuracy of 87.5%.

# Appendix: Code snippet for Dash

```
# TASK 2:
# Add a callback function for `site-dropdown` as input, `success-pie-chart` as output
# Function decorator to specify function input and output
@app.callback(Output(component_id='success-pie-chart', component_property='figure'),
              Input(component_id='site-dropdown', component_property='value'))

def get_pie_chart(entered_site):
    if entered_site == 'ALL':
        data_rate = spacex_df[['Launch Site', 'class']].groupby(['Launch Site']).mean().reset_index()
        fig = px.pie(data_rate, values='class',
            names='Launch Site',
            title='Total success launches by site')
        return fig
    else:
        # return the outcomes piechart for a selected site
        site_df = spacex_df[spacex_df['Launch Site'] == entered_site]
        site_rate = site_df['class'].value_counts(normalize=True).reset_index()
        fig = px.pie(site_rate, values='proportion',
            names='class',
            title='Success rate for site:' + entered_site)
        return fig

# TASK 4:
# Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output
@app.callback(Output(component_id='success-payload-scatter-chart', component_property='figure'),
              [Input(component_id='site-dropdown', component_property='value'),
              Input(component_id='payload-slider', component_property='value')])

def get_scatter_chart(entered_site, payload_mass):
    if entered_site == 'ALL':
        allsites_mass_df = spacex_df[spacex_df['Payload Mass (kg)'].between(payload_mass[0], payload_mass[1])]
        fig = px.scatter(allsites_mass_df, x='Payload Mass (kg)', y='class', color="Booster Version Category",
            title='Correlation between payload and success for all sites')
        return fig
    else:
        site_df = spacex_df[spacex_df['Launch Site'] == entered_site]
        site_mass_df = site_df[site_df['Payload Mass (kg)'].between(payload_mass[0], payload_mass[1])]
        fig = px.scatter(site_mass_df, x='Payload Mass (kg)', y='class', color="Booster Version Category",
                    title='Correlation between payload and success for site ' + entered_site)
        return fig
```

# Thank you!