IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Joe Giacobbe
July 15th, 2024

Outline

Executive Summary

- Summary of methodologies
  - Data Collection (via REST API from SpaceX Wikipedia site + Webscraping)
  - Improve Data Quality: format, filter, & clean collected data, perform Data Wrangling
  - Conduct Exploratory Data Analysis (EDA) using SQL, Visualization using Pandas & MatPlotlib
  - Build Interactive Dashboard (Plotly Dash) and Visual Analytics (interactive map using Folium)
  - Run Predictive Analysis using Machine Learning (optimize method based on best performance)

Executive Summary (cont'd)

- Summary of results
  - EDA:
    - Launch success has progressively increased over time
    - KSC LC39A has highest success rate amongst launch sites
    - Orbits ES-L1, GEO, HEO, and SSO have 100% success rate
  - Interactive Visualization
    - Launch sites are near equator, and close to a seacoast, as well as in proximity to rail and other transport modes.
  - Predictive Analysis
    - Decision Tree model was most effective at predicting landing outcome.

Introduction

- Project background and context:
  - SpaceX is one of the most successful commercial space organizations, in large part because their rocket launches are relatively inexpensive. SpaceX advertises Falcon 9 rocket launches on its website at a cost of $62M; other providers cost upwards of $165M each – much of the savings is because SpaceX can reuse the first stage, which is fairly large and expensive.
  - Therefore, if we can predict if the first stage will land (and be reused), we can determine the cost of a launch.

- Problems to be solved:
  - Quantify price of each launch, by gathering information about Space X and formatting it for rigorous analysis.
  - Identify variables that have greatest impact on successfully landing the first stage, and whether any interactions between variables influence the landing; present information using dashboards and visuals
  - Build machine learning model, use public information to predict optimal conditions for a successful landing, and whether SpaceX will be able to reuse the first stage.

Section 1

Methodology

Methodology

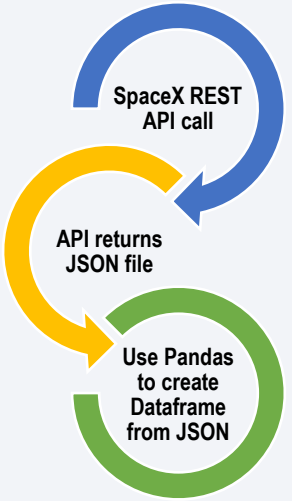
Executive Summary

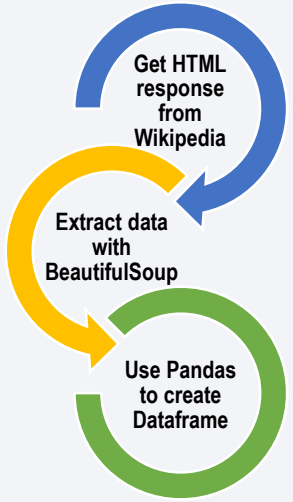
- Data collection methodology:
  - SpaceX API json data
  - Web Scraping from Wikipedia Falcon 9 page
- Data wrangling
  - Dropped unnecessary columns, handle missing values.
  - One Hot Encoding to prep for classification models
- Exploratory data analysis (EDA) using visualization and SQL
- Interactive visual analytics using Folium and Plotly Dash
- Predictive analysis using classification models
  - Used Scikit-Learn to build & train classification models; optimized effectiveness and ultimately identified most accurate model for prediction

Data Collection

The data sets were collected as follows:

1. A call to the SpaceX API
   - Information obtained by the API includes rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
   - Response is in the form of a JSON – a list of JSON objects which each represent a launch – which is then converted into a pandas dataframe using json_normalize

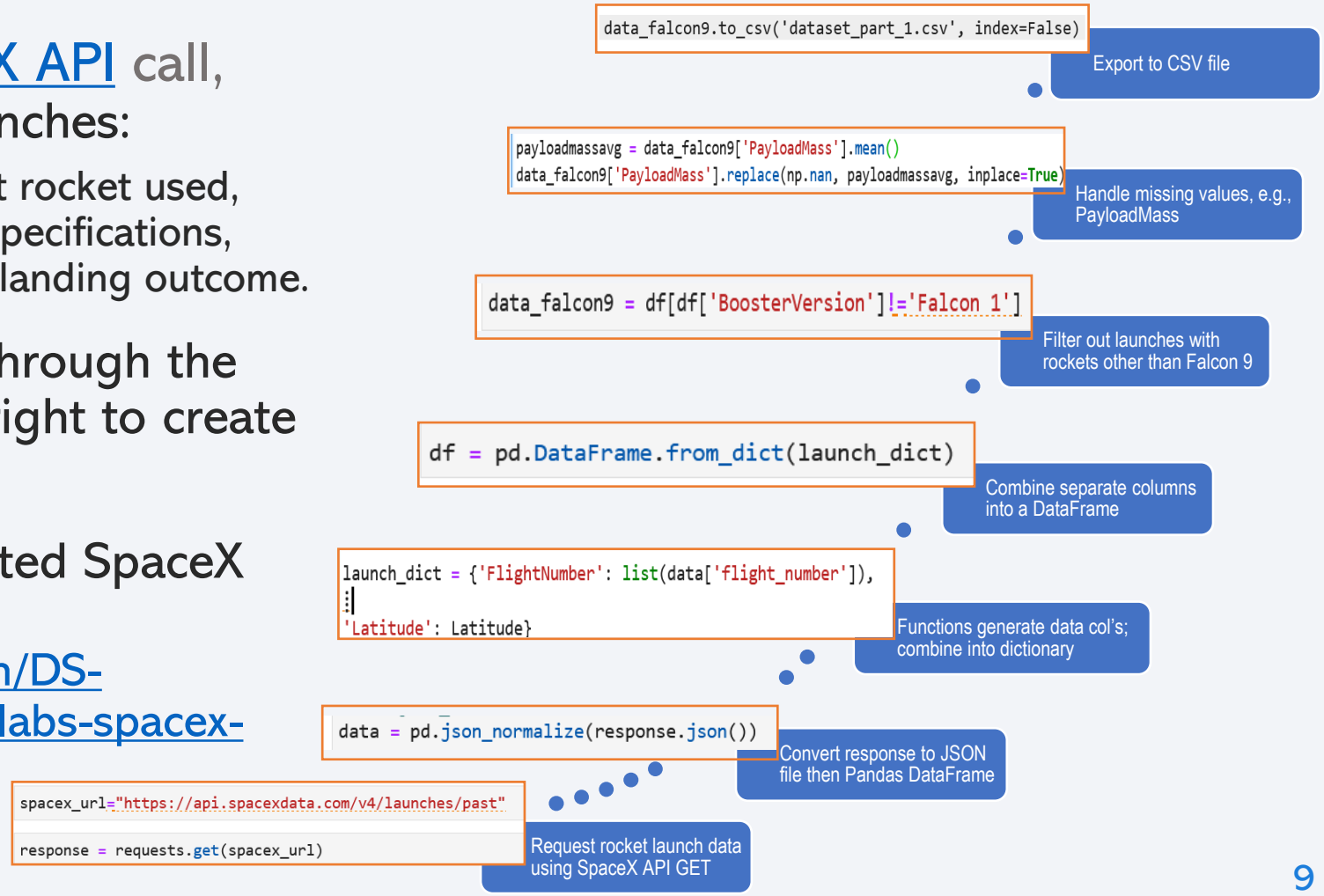




2. Web Scraping from Wikipedia Falcon 9 page
   - Information obtained by the webscraping of Wikipedia includes launches, landing, payload information.
   - Used Python BeautifulSoup package to web scrape HTML tables that contain Falcon 9 launch records, which were parsed and converted into a Pandas dataframe

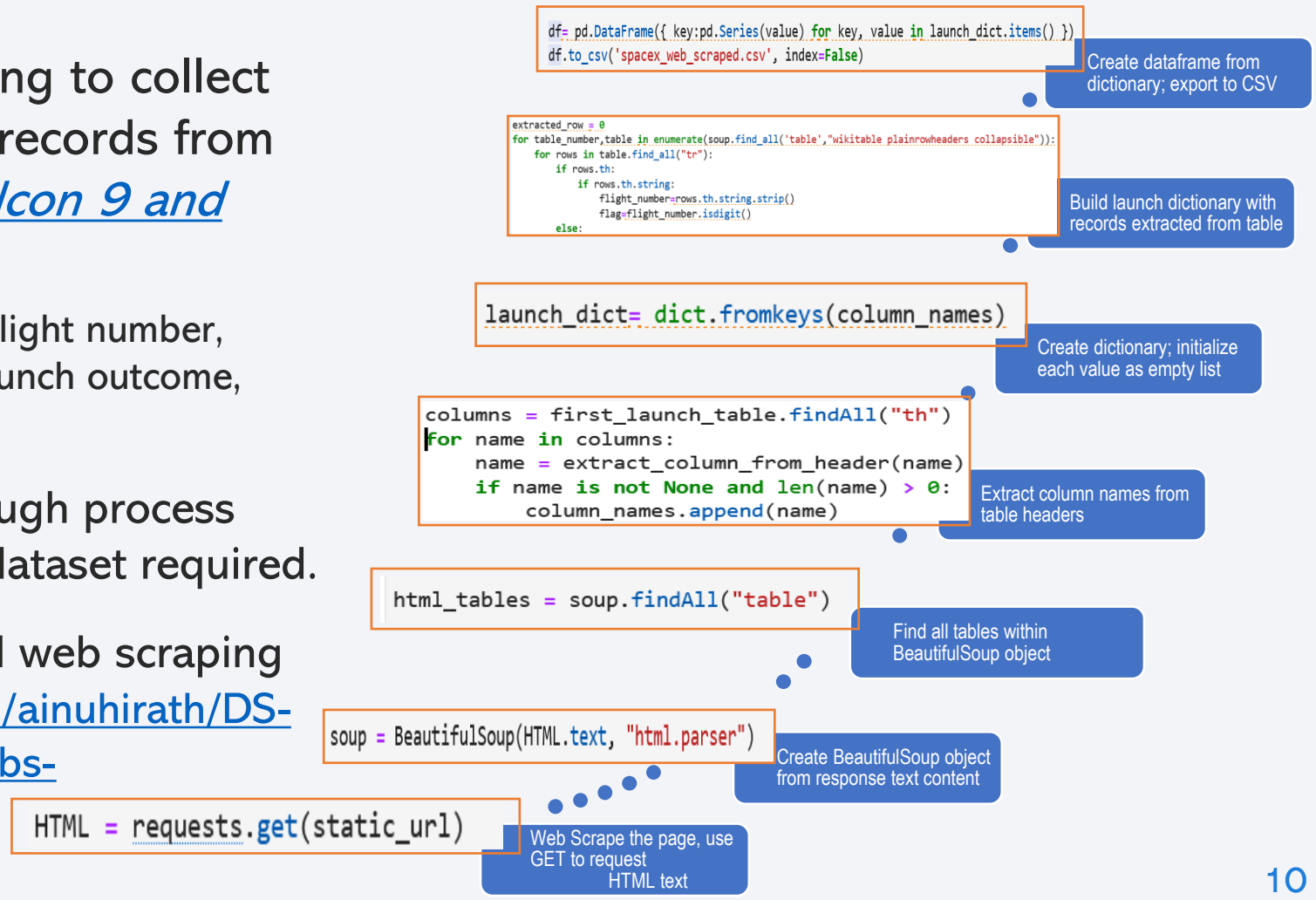
Data Collection – SpaceX API

- Initial step was a SpaceX API call, requesting data about launches:
  - Including information about rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
- Data received then went through the process flowchart on the right to create the dataset required.
- GitHub URL of the completed SpaceX API calls notebook: https://github.com/ainuhirath/DS-capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



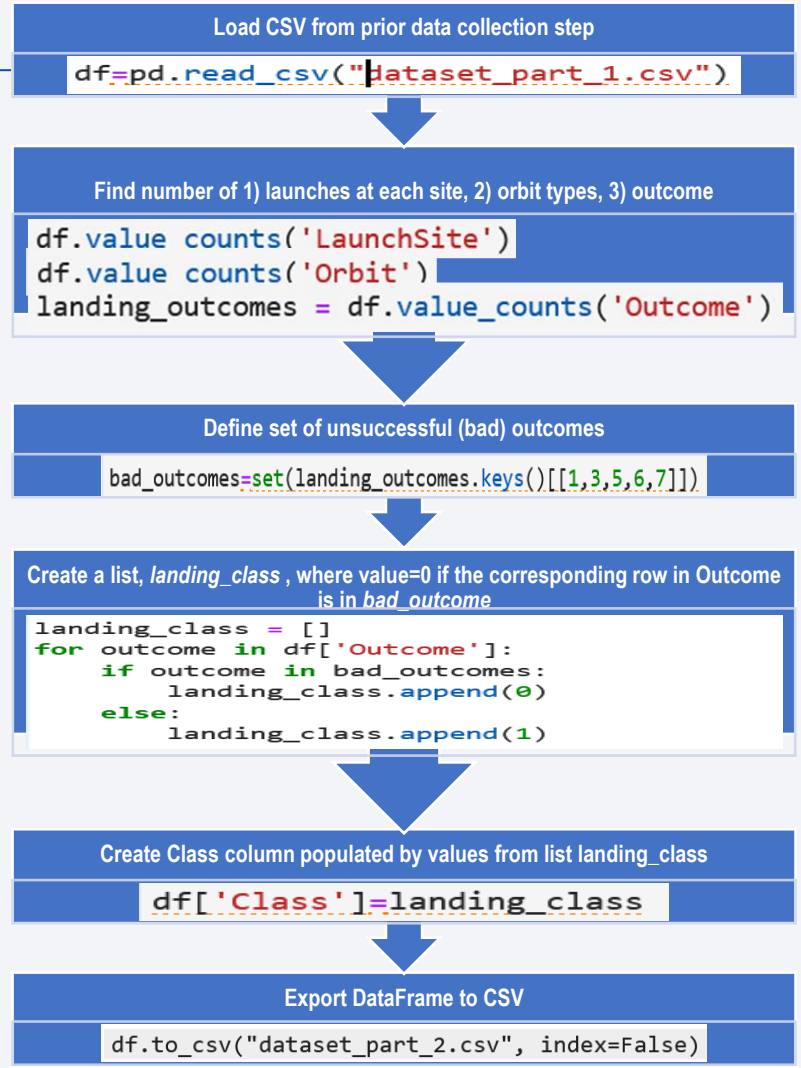
Data Collection - Scraping

- Initial step was web scraping to collect Falcon 9 historical launch records from Wikipedia page, *List of Falcon 9 and Falcon Heavy launches*:
  - Including information about flight number, launch site, payload, orbit, launch outcome, booster version.
- Data scraped then went through process flowchart on right to create dataset required.
- GitHub URL of the completed web scraping notebook : https://github.com/ainuhirath/DS-capstone/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

- Data Wrangling
  - Defined as process of transforming and structuring data from raw form into desired format, to improve data quality and make it more consumable and useful for analytics or ML.
  - Converted various outcomes into Training labels: 1 means the booster successfully landed, 0 means it was unsuccessful.
- Conducted preliminary Exploratory Data Analysis (EDA) to identify data patterns and determine likely label for training models.
- GitHub URL of completed data wrangling notebook:
  - https://github.com/ainuhirath/DS-capstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- Launch Site insights – impact to mission success
  - Scatterplot: Launch Site and Flight Number correlation to mission outcome
  - Scatterplot: Launch Site and Payload correlation to mission outcome
- Orbit Type insights – impact to mission success
  - Bar chart: Orbit Type correlation to mission outcome
  - Scatterplot: Orbit Type and Flight Number correlation to mission outcome
  - Scatterplot: Orbit Type and Payload correlation to mission outcome
- Timeline Insights
  - Line plot: Yearly trend correlation to mission outcome
- GitHub URL of completed data wrangling notebook:
  - https://github.com/ainuhirath/DS-capstone/blob/main/edadataviz-o.ipynb

EDA with SQL

- SQL Queries conducted provided insight on:
  - Launch sites
  - Payload mass – Max and Average,
  - Dates (of first successful mission outcome)
  - Booster types with max Payload
  - Mission outcomes – first success, success vs. failure, and for some specific criteria
- GitHub URL of completed EDA with SQL notebook:
  - https://github.com/ainuhirath/DS-capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

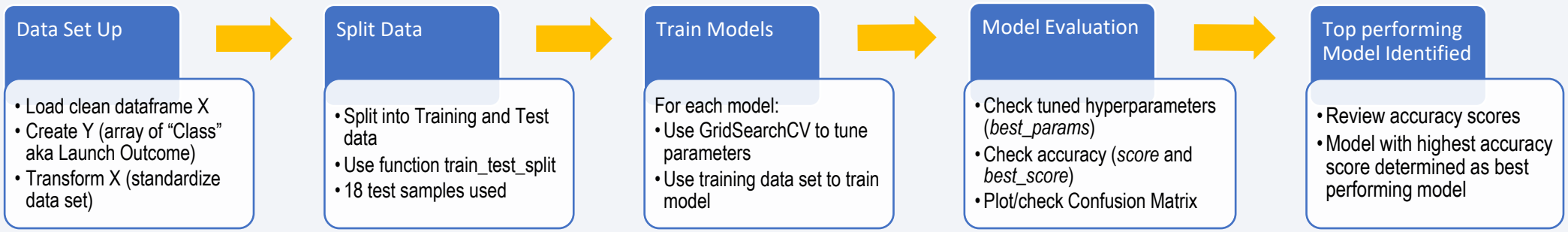
- Map objects created and added to a folium map:
  - Circle markers were added for NASA Johnson Space Center and for launch sites
  - Then added markers for successful (green) or failed (red) launches for each site
  - Colored Lines were added to show launch site proximity to nearby features:
    - Distance from CCAFS LC-40 to the coastline
    - Distance from CCAFS LC-40 to the rail line
    - Distance from CCAFS LC-40 to the perimeter road
    - Distance from CCAFS LC-40 to nearest city (Titusville, 50K population)
- Above objects were added so as to visually explore the data and begin to understand underlying issues.
- GitHub URL of completed interactive map with Folium map:
  - https://github.com/ainuhirath/DS-capstone/blob/main/lab_jupyter_launch_site_location-2.ipynb

Build a Dashboard with Plotly Dash

- Dropdown List: Launch Sites
  - User can select one or all launch sites
- Pie Chart: Successful Launches
  - User can see successful and unsuccessful launches as a percent of the total
  - Visually highlights which sites have most successful launches
  - Chart can also be filtered (using Dropdown) to see success/failure ratio for individual site
- Slider: Payload Mass Range
  - User can select payload mass (0 to 10K kg) for the scatter chart
- Scatter Chart: Payload Mass vs. Success Rate by Booster Version
  - User can see correlation between Payload and Launch Outcome
- GitHub URL of completed Plotly Dash lab:
  - https://github.com/ainuhirath/DS-capstone/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

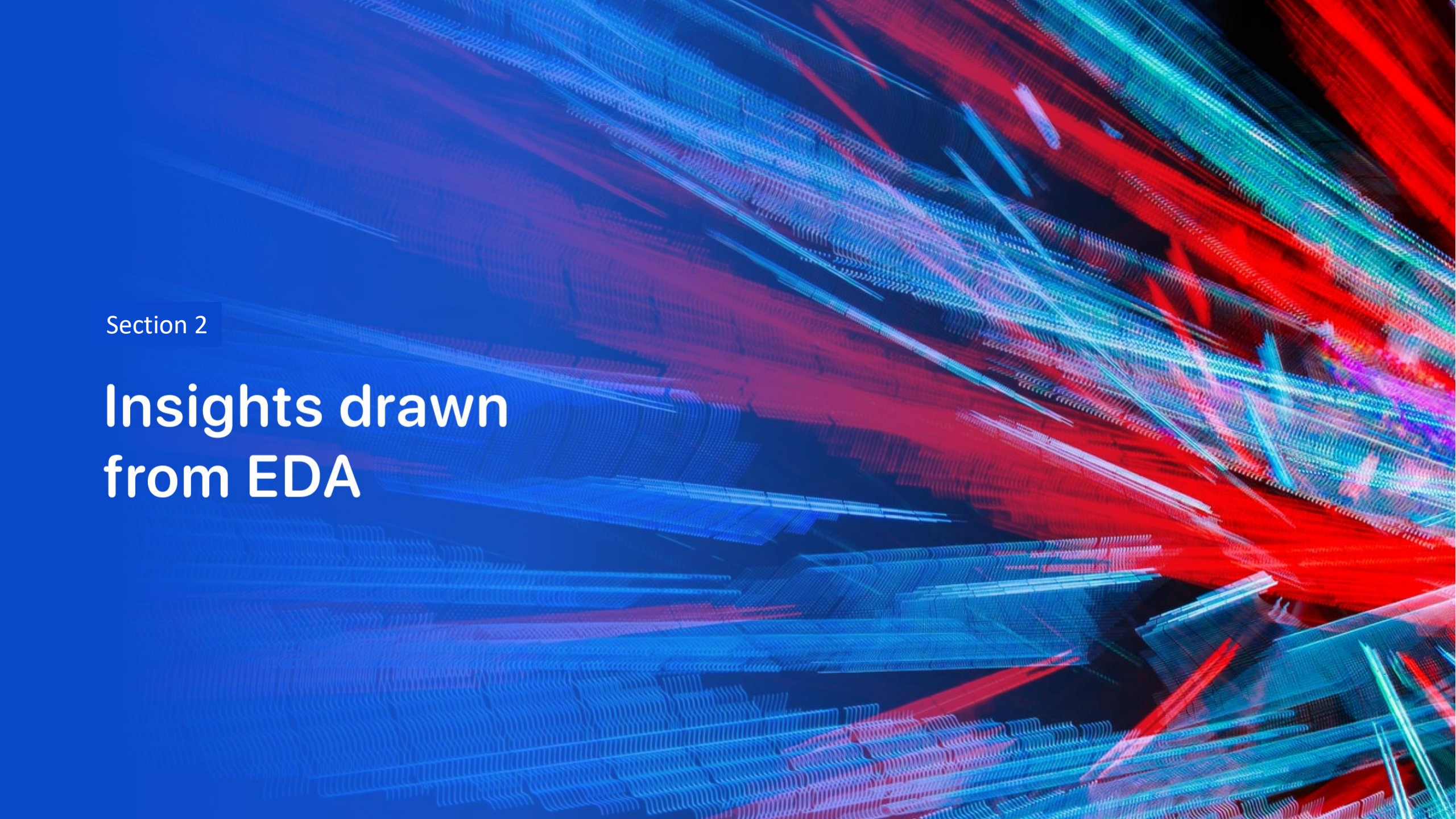
- How best performing classification model was built, improved, evaluated, identified:
  1. Prepared Data Set containing SpaceX launch outcomes and associated features/variables
  2. Selected Machine Learning Models:
     1. Logistic Regression
     2. Support Vector Machine
     3. Decision Tree
     4. K Nearest Neighbours
  3. Identified most accurate model to predict launch outcome
- GitHub URL of completed predictive analysis lab:
  - https://github.com/ainuhirath/DS-capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5-v2.ipynb



Results

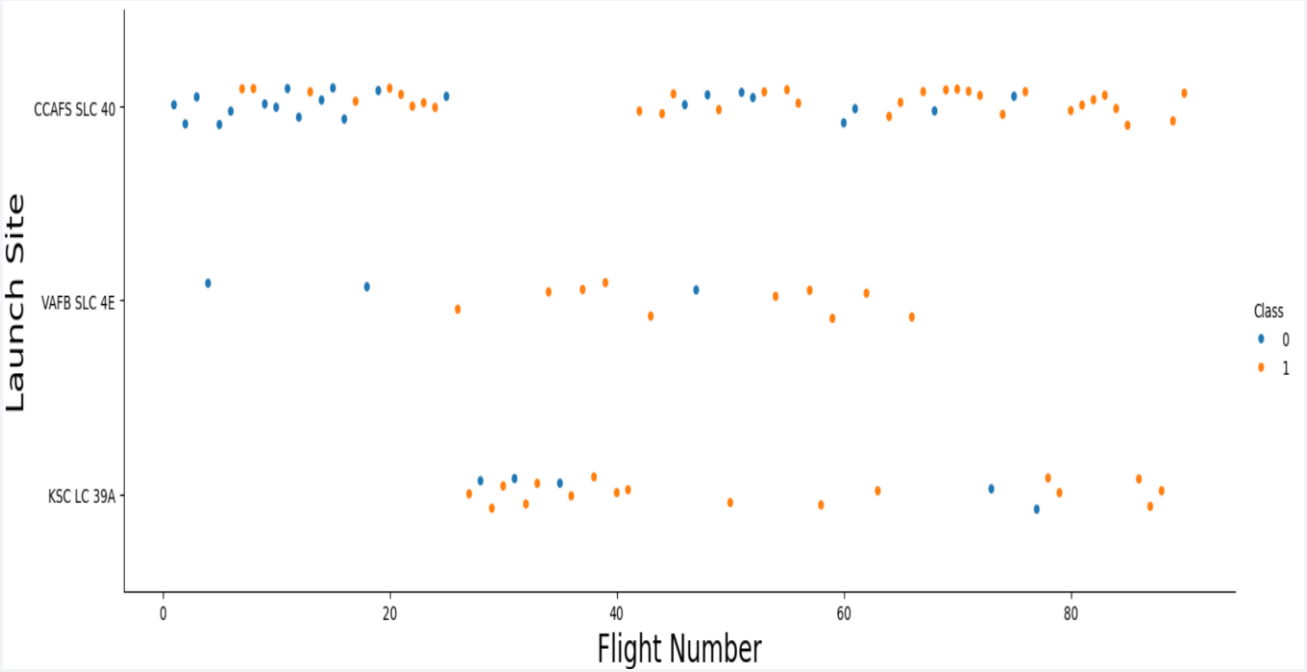
Sections that follow (#2 to #5) provide detail on:

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


Section 2
Insights drawn from EDA

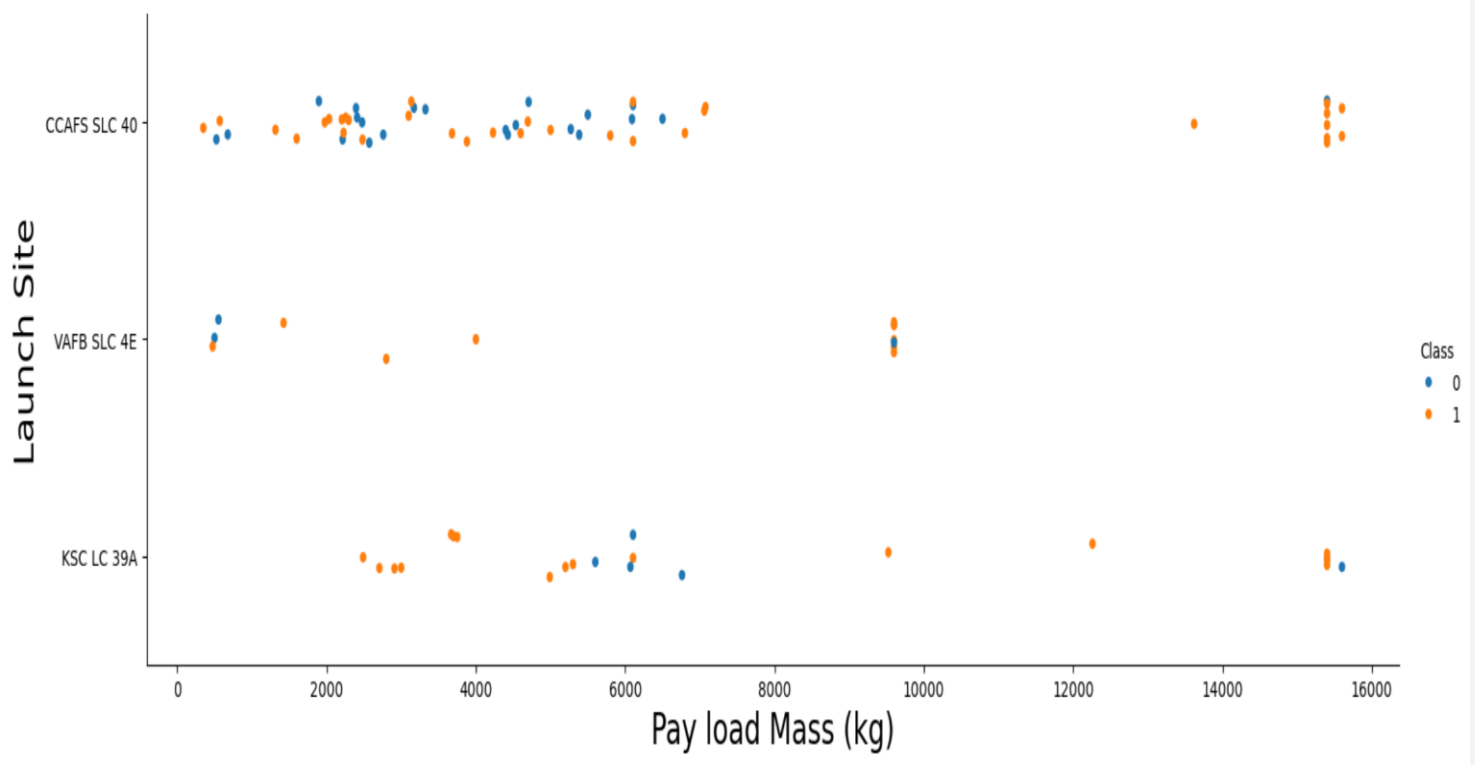
Flight Number vs. Launch Site

- Scatter plot displays launch outcomes, where:
  - 0 = Failed (blue)
  - 1 = Success (orange)
- Trend is that success rate generally improves over time.
- As site *CCAFS SLC 40* had most of the early launches, its overall success rate has been adversely impacted by large number of early failures.



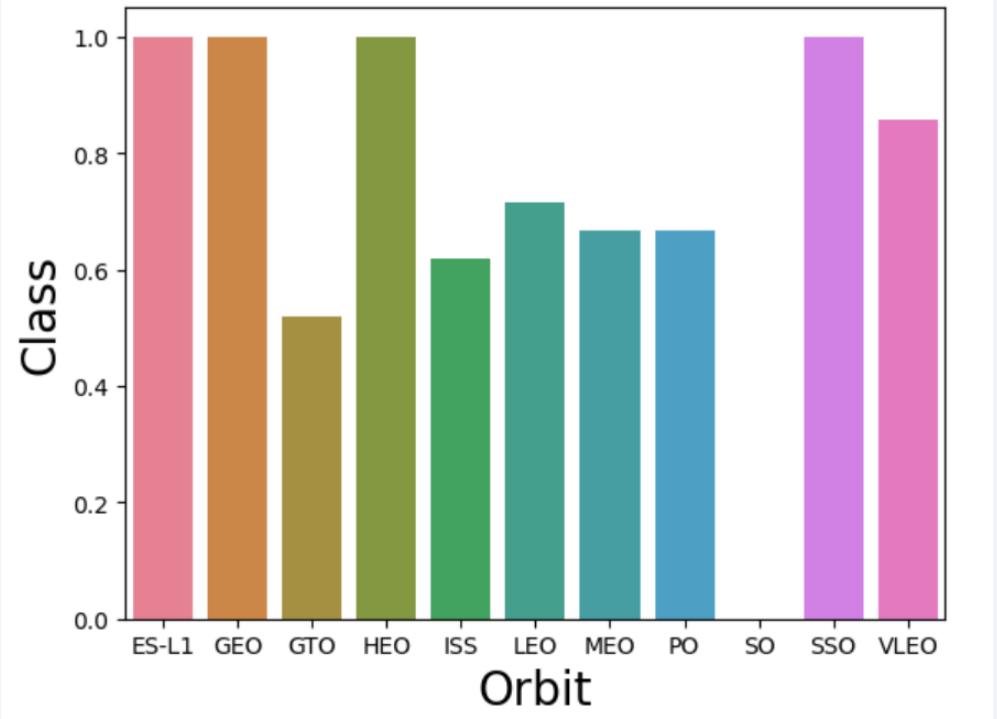
Payload vs. Launch Site

- Scatter plot displays launch outcomes, where:
  - 0 = Failed (blue)
  - 1 = Success (orange)
- No clear pattern between payload and success rate EXCEPT heavy loads above 9000 kg – these have a high success rate across all sites.



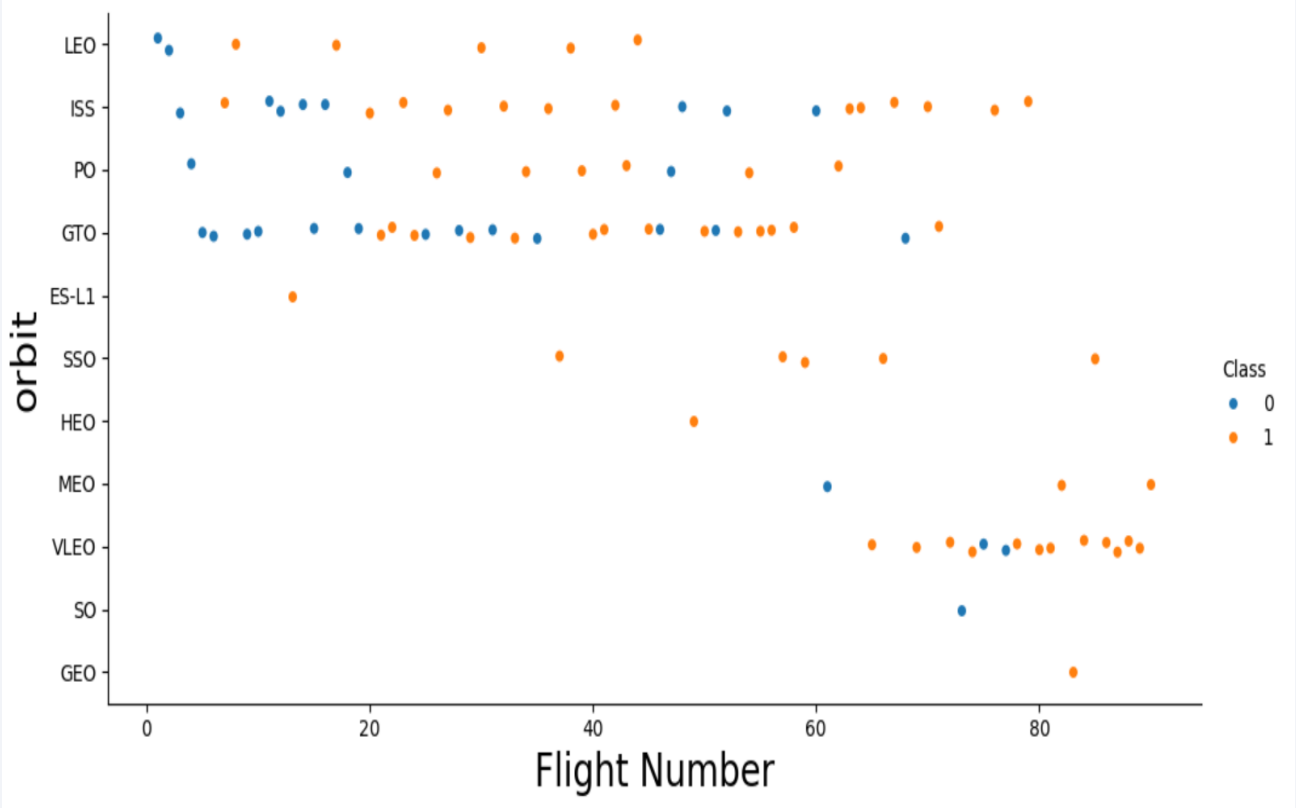
Success Rate vs. Orbit Type

- Bar chart displays launch success rate (“Class”) by orbit type
- Notable points from chart:
  - ES L1, GEO, HEO and SSO orbits have 100% Success Rate
  - SO orbit has 0% Success Rate
  - But these orbits with extremely low or high success rates may have low sample sizes so more assessment needed



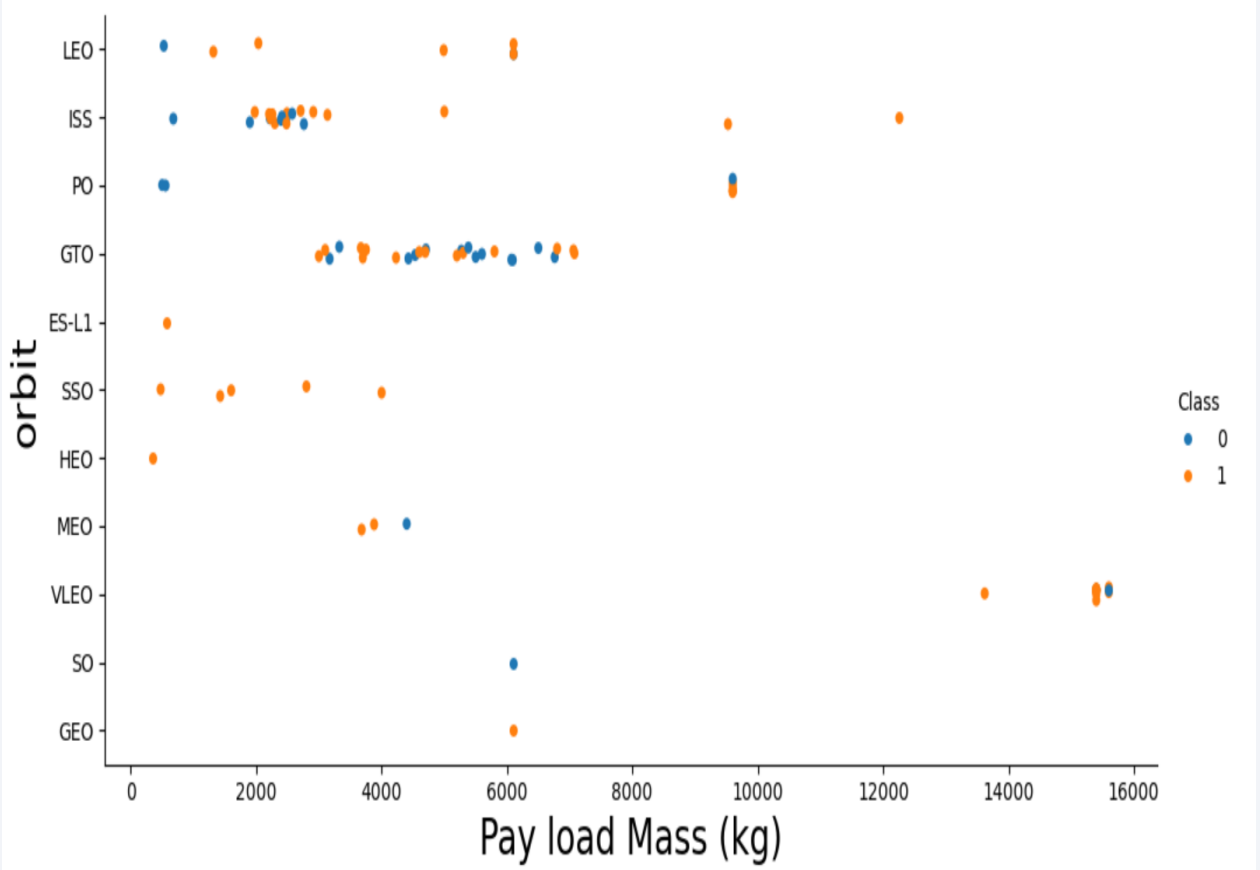
Flight Number vs. Orbit Type

- Scatter plot displays launch outcomes, where:
  - 0 = Failed (blue)
  - 1 = Success (orange)
- Trend: success rate for most orbit types generally improves over time.
- Important to note that success rate for specific orbit types is clearly impacted by when they began – e.g., SSO (100% success rate) started much later than ISS or GTO.



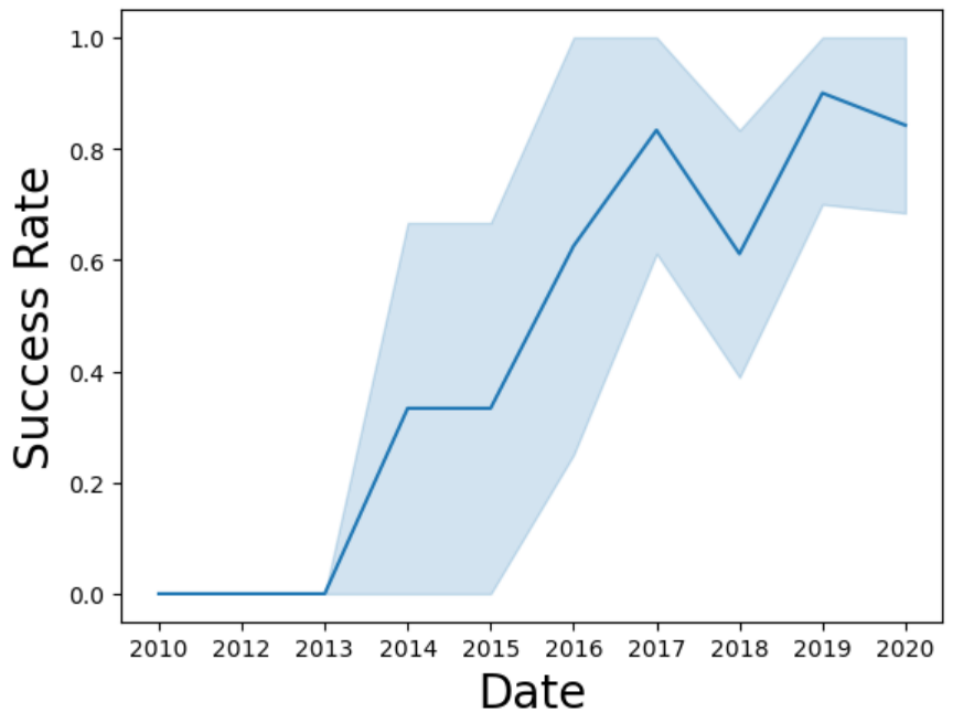
Payload vs. Orbit Type

- Scatter plot displays launch outcomes, where:
  - 0 = Failed (blue)
  - 1 = Success (orange)
- Some orbit types seem to consistently have low payloads (e.g., SSO), while others are high (e.g., VLEO).
- Overall (considering that some orbit types have extremely low counts) there's no clear pattern predicting success based on payload, esp. if one refers back to chart of orbit type success over time.



Launch Success Yearly Trend

- Overall, the launch success rate has increased dramatically over time, reaching 80-90% in the last 2 years displayed.
- 2018 appears to have been a “bad year” for launch outcomes – underlying reasons unclear.



All Launch Site Names

- Objective: ***Find the names of the unique launch sites.***
- Query used to generate result:

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

- Findings:

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

- Explanation: Used the key word DISTINCT to list unique launch sites in the SpaceX data set (SPACEXTABLE).

Launch Site Names Begin with 'CCA'

- Objective: ***Find 5 records where launch sites begin with `CCA`.***
- Query used to generate result:

```
%sql SELECT * FROM SPACEXTABLE WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

- Findings:

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

- Explanation: The WHERE clause followed by LIKE using a wild card identifies launch sites that contain the substring CCA. LIMIT 5 displays only 5 records from the result.

Total Payload Mass

- Objective: ***Calculate the total payload carried by boosters from NASA.***
- Query used to generate result:

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

- Findings:

SUM("PAYLOAD_MASS__KG_")
48213

- Explanation: As with prior query using LIKE, this one returns sum of all payload masses where the customer field contains the substring “NASA (CRS)”.

Average Payload Mass by F9 v1.1

- Objective: ***Calculate the average payload mass carried by booster version F9 v1.1***
- Query used to generate result:

```
%sql SELECT AVG("PAYLOAD_MASS__KG_") AS Average_PayloadMass FROM SPACEXTABLE WHERE Booster_Version LIKE 'F9 v1.1%';
```

- Findings:

Average_PayloadMass
2534.6666666666665

- Explanation: As with prior query using LIKE, this one returns the average of all payload masses where Booster Version field begins with the substring “F9 v1.1”.

First Successful Ground Landing Date

- Objective: ***Find the dates of the first successful landing outcome on ground pad.***
- Query used to generate result:

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

- Findings:

date_of_first_succesful_landing_outcome_in_ground_pad
2015-12-22

- Explanation: Used the MIN function to select the oldest date classified as successful (i.e., WHERE the Landing Outcome variable has value “Success (ground pad)”.

Successful Drone Ship Landing with Payload between 4000 and 6000

- Objective: ***List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000.***
- Query used to generate result:

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

- Findings:

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

- Explanation: Used WHERE keyword to filter results to include only those that satisfy all conditions (because AND keywords used), including only payload mass >4000 but < 6000.

Total Number of Successful and Failure Mission Outcomes

- Objective: ***Calculate the total number of successful and failure mission outcomes.***
- Query used to generate result:

```
%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL_MISSIONS FROM SPACEXTABLE GROUP BY MISSION_OUTCOME;
```

- Findings:

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

- Explanation: The COUNT keyword is used to calculate the total number of mission outcomes, and the GROUPBY keyword is used to group these results by the type of mission outcome.

Boosters Carried Maximum Payload

- Objective: ***List the names of the booster which have carried the maximum payload mass.***
- Query used to generate result:

```
%sql SELECT DISTINCT Booster_Version AS "Booster_versions which have carried the maximum payload mass" FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ == (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
```

- Findings:

Booster_versions which have carried the maximum payload mass
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

- Explanation: Used a subquery to filter data, MAX function returned heaviest payload mass. The main query uses subquery results and returns unique booster versions (SELECT DISTINCT) with the identifed heaviest payload mass.

2015 Launch Records

- Objective: ***List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015.***
- Query used to generate result:

```
%sql SELECT  substr(Date, 6,2) AS MONTH, LANDING_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTABLE WHERE Landing_Outcome = 'Failure (drone ship)' AND substr(Date,0,5) = '2015';
```

- Findings:

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

- Explanation: Used WHERE keyword to filter the results for only failed landing outcomes, AND only for the year of 2015. Used Substr function to process and display month.

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

- Objective: ***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.***
- Query used to generate result:

```
%sql SELECT LANDING_OUTCOME, COUNT(LANDING_OUTCOME) AS TOTAL_NUMBER FROM SPACEXTABLE WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING_OUTCOME ORDER BY TOTAL_NUMBER DESC;
```

- Findings:

Landing_Outcome	TOTAL_NUMBER
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

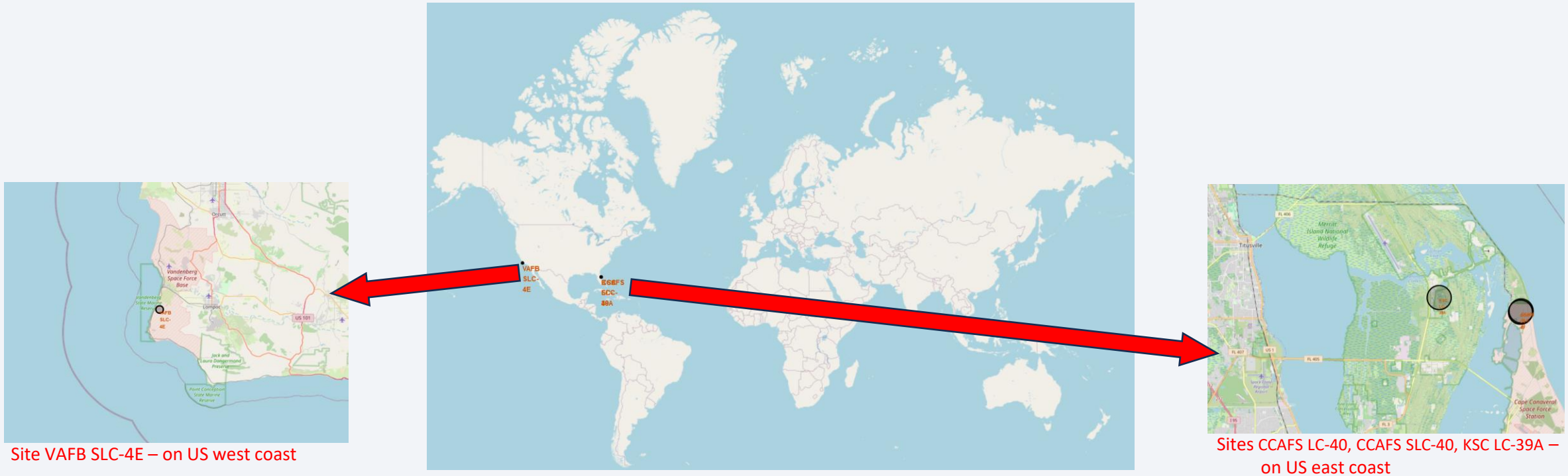
- Explanation: Used WHERE keyword with the BETWEEN keyword to filter the results to dates in the range specified. Results were then grouped and ordered, using the keywords GROUP BY and ORDER BY, respectively, where DESC is used to display a descending order.

Section 3

Launch Sites Proximities Analysis

Launch Site Locations

Global map with launch site locations



Site VAFB SLC-4E – on US west coast

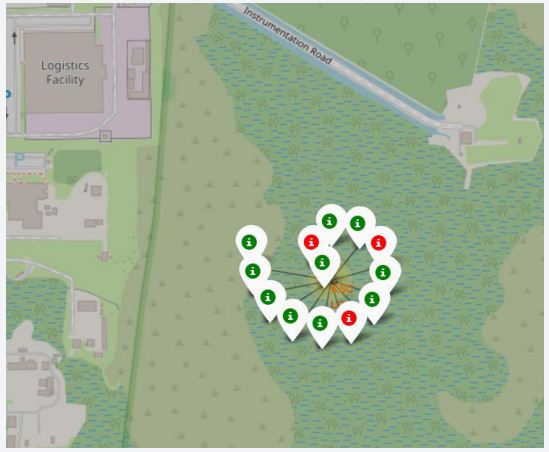
Sites CCAFS LC-40, CCAFS SLC-40, KSC LC-39A – on US east coast

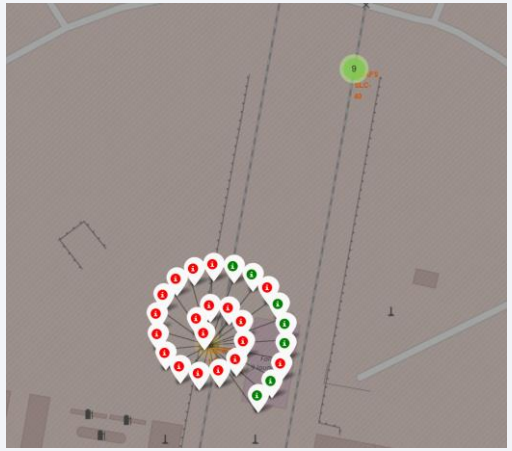
- SpaceX launch sites are along USA seacoast, specifically in Florida and California, as this reduces chances of accidental debris falling on urban areas and people.
- All sites are close to equator, as this reduces cost/fuel required to achieve orbit.

Success/failed launches for each site

Site VAFB SLC-4E

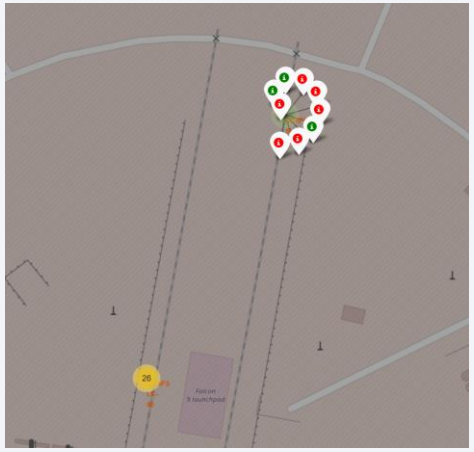
Site KSC LC-39A





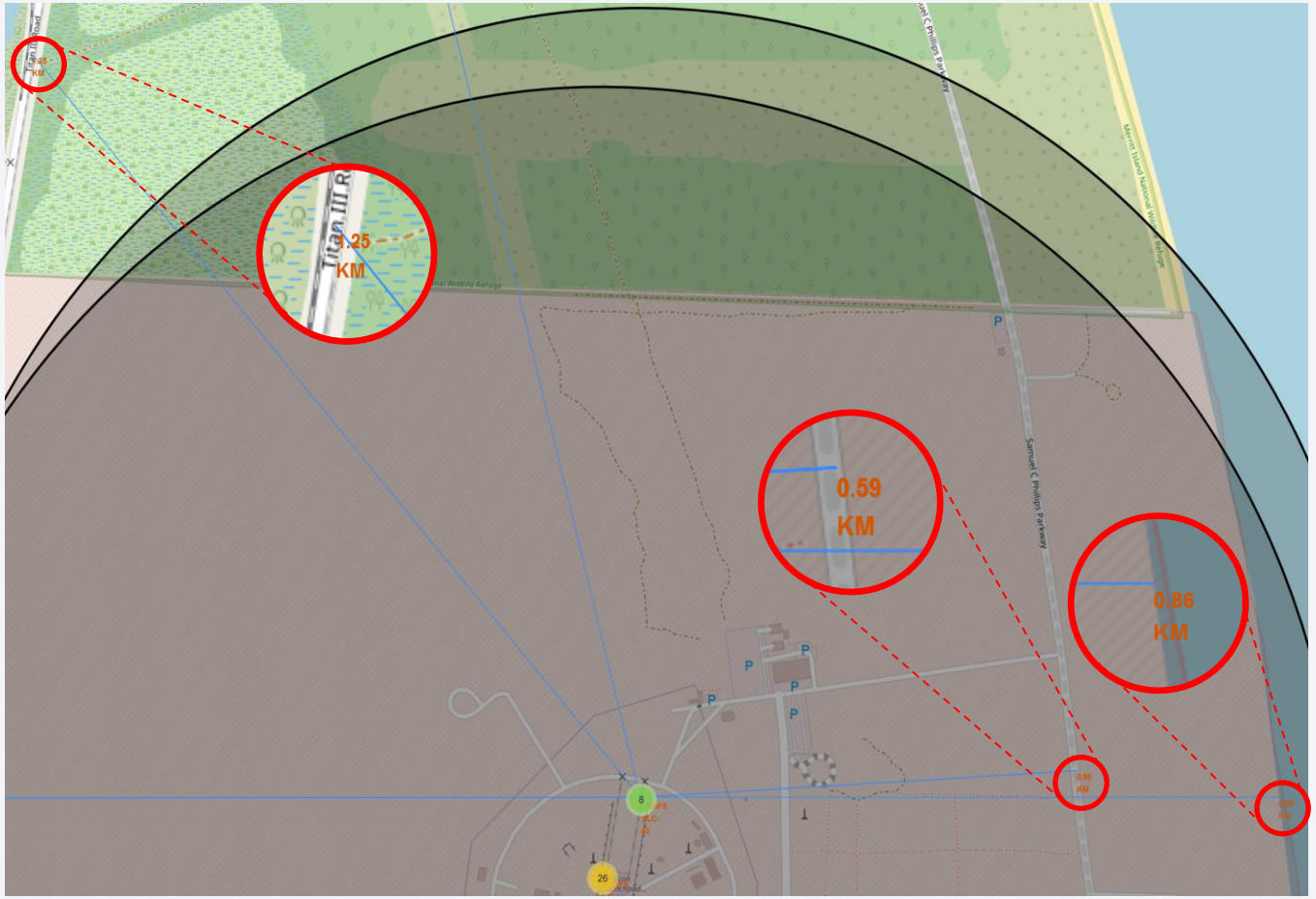
Site CCAFS LC-40

Site CCAFS SLC-40



- Launches grouped into site clusters; icons colored green for successful launches, red for failed launches.
- *KSC LC-39A* launch site has a relatively high success rate judging by preponderance of green icons.

Distances between CCAFS SLC 40 and local POI's



Are launch sites in proximity to the coast?

- YES. The coastline is 0.86 km away.
- Be over water during launch so (1) if launch aborts, attempt water landing; (2) minimize risk to people and property from possible accidents.

Are launch sites in proximity to highways?

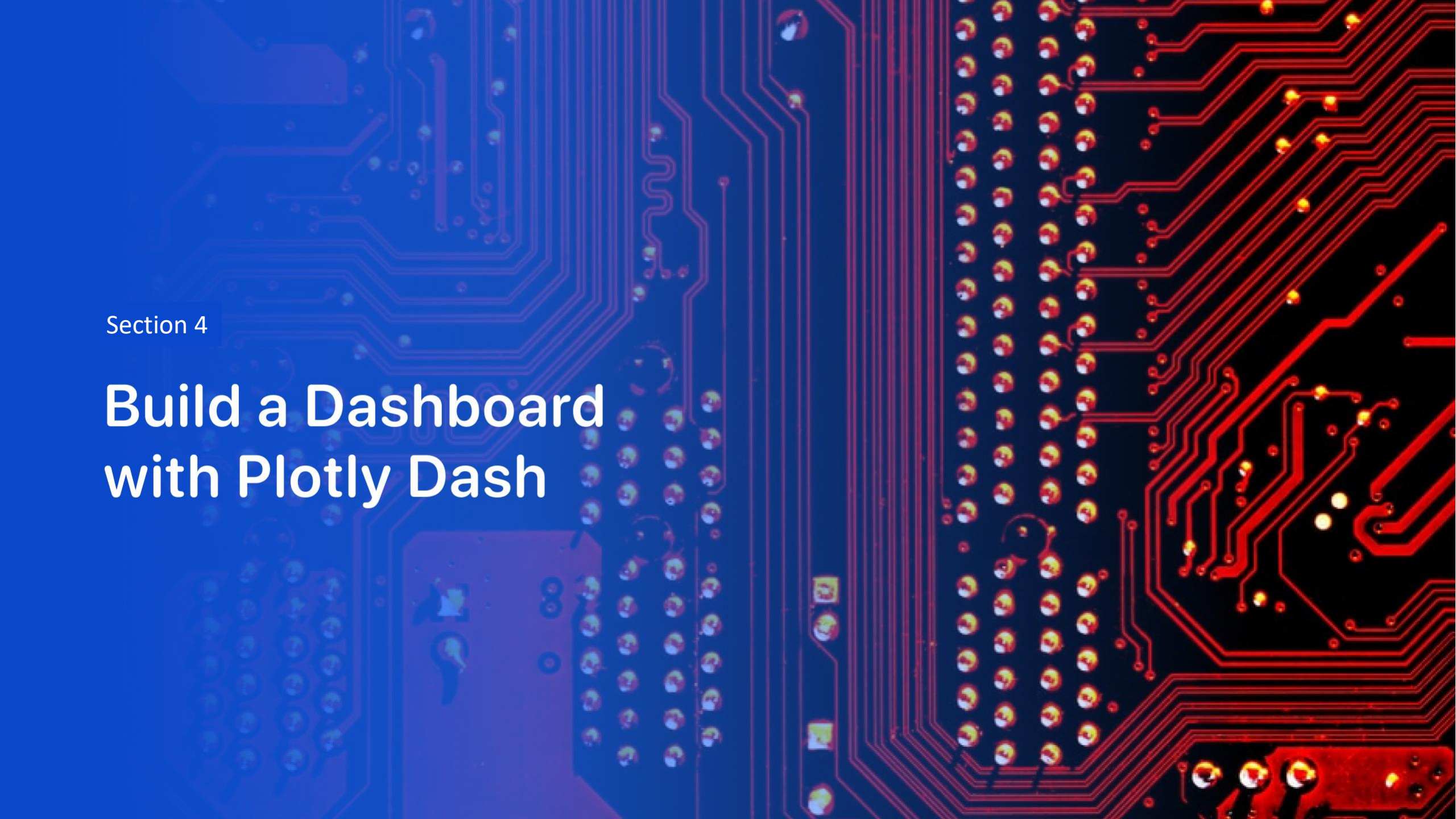
- YES. The nearest highway is 0.59 km away.
- Easily and rapidly move people and smaller items

Are launch sites in proximity to railways?

- YES. The nearest railway is 1.25 km away.
- Transport heavy loads to site

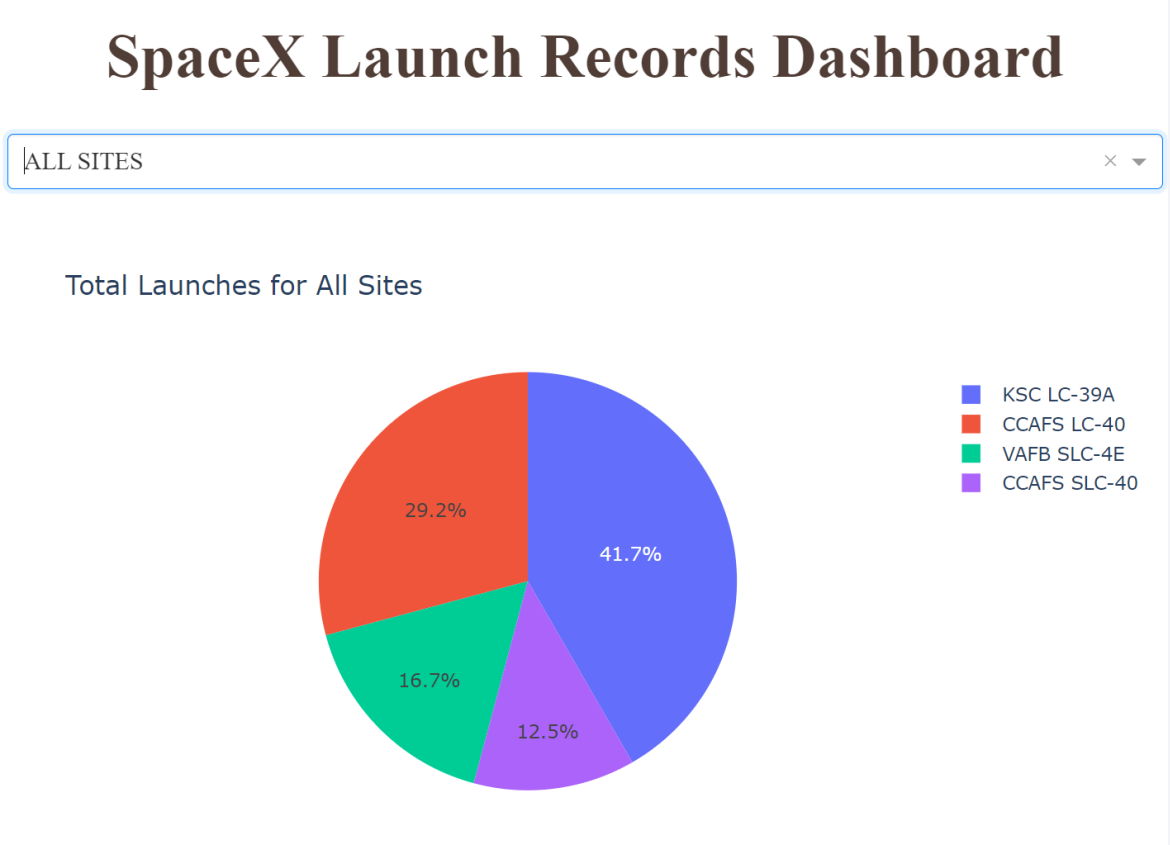
Do launch sites keep a certain distance away from cities?

- YES. Titusville is 21.8 km away.
- Minimize danger to population-dense areas


Section 4
Build a Dashboard with Plotly Dash

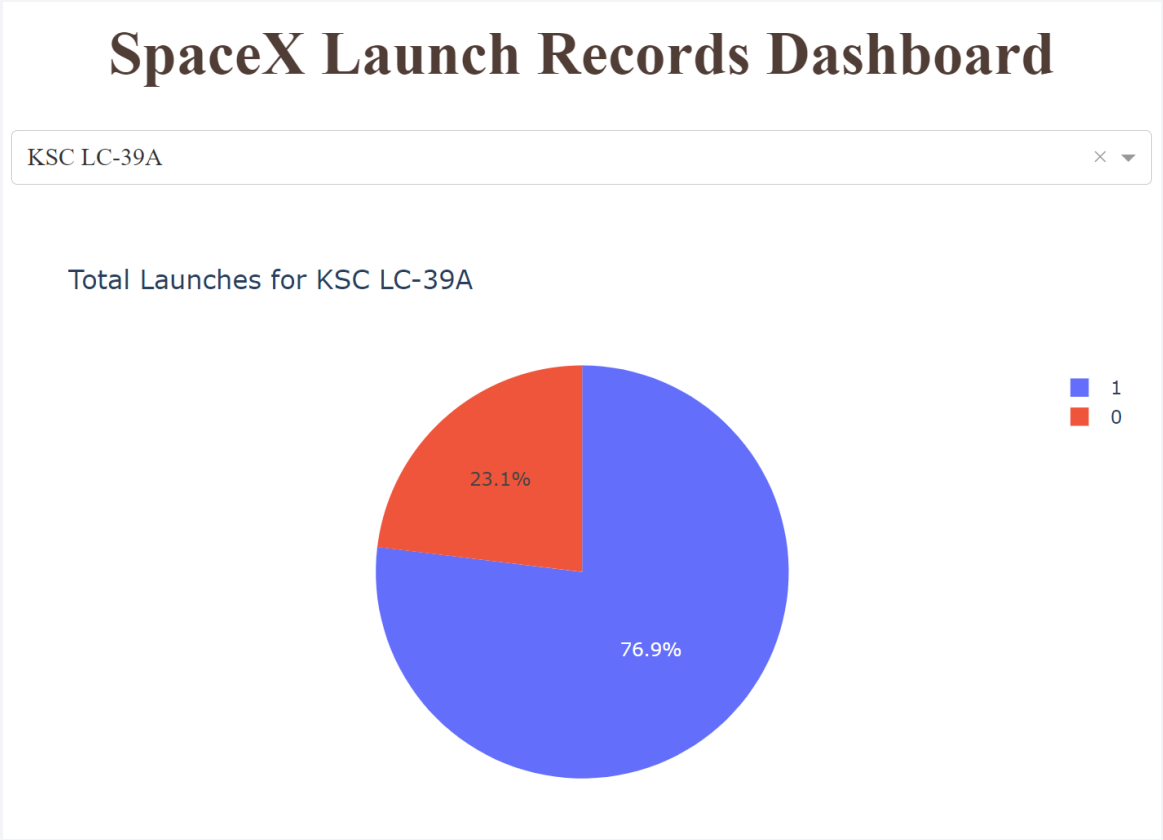
SpaceX Successful Launch Counts, Across Sites

- Pie chart to the right shows percentage of successful launches by launch site.
- Key Takeaway: *KSC LC-39A* is launch site with most successful launches – 41.7% of the total.



SpaceX Successful Launch Counts, Across Sites

- Pie chart to the right displays percentage of successful launches for *KSC LC-39A*, the launch site with the most successful launches.
- Blue = successful launch
- Key Takeaway: *KSC LC-39A* also has a high success rate – 76.9% of its launches were successful.



Launch Outcome by Payload Mass, All Sites

All Payloads, 0 to 10K kg

Payload 0 to 4K kg

Payload 4K to 10K kg

Scatter charts display launch success (class = 1) vs. failure (class = 0)

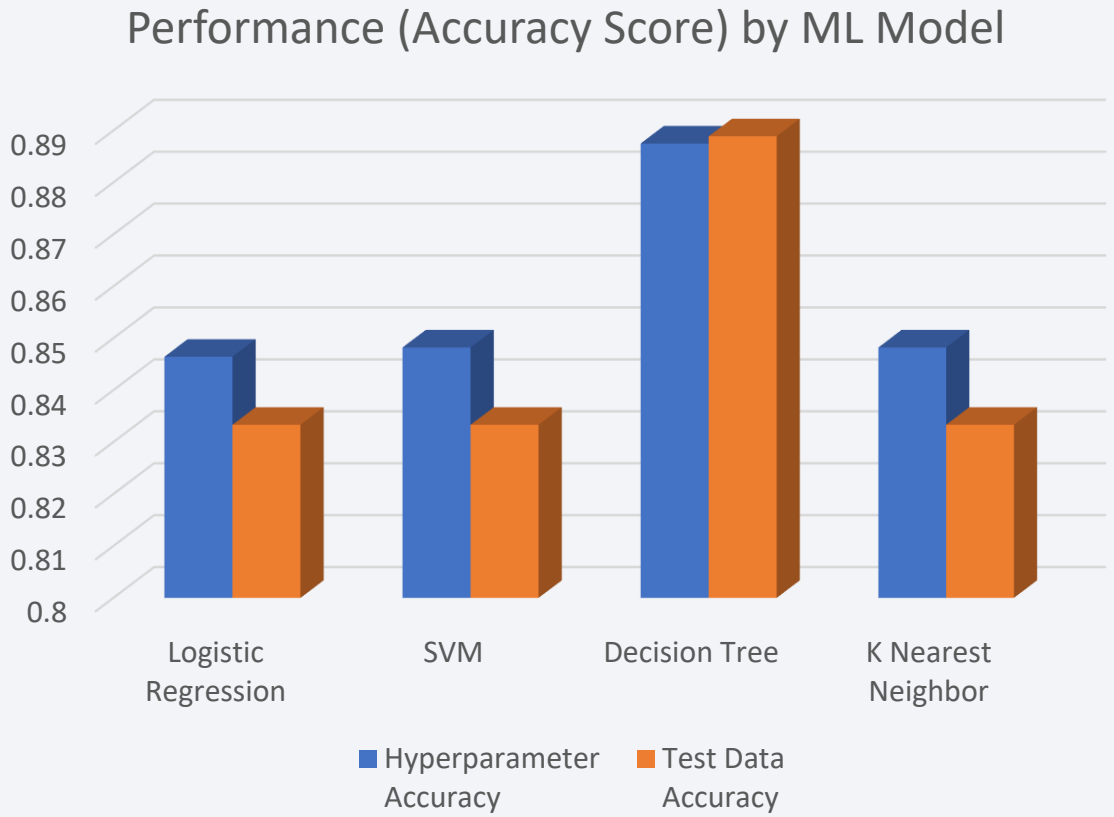
Lighter payloads (<4K kg) have higher success rate; conversely, heavier payloads (4 -10K kg) have lower success rate.

Section 5

Predictive Analysis (Classification)

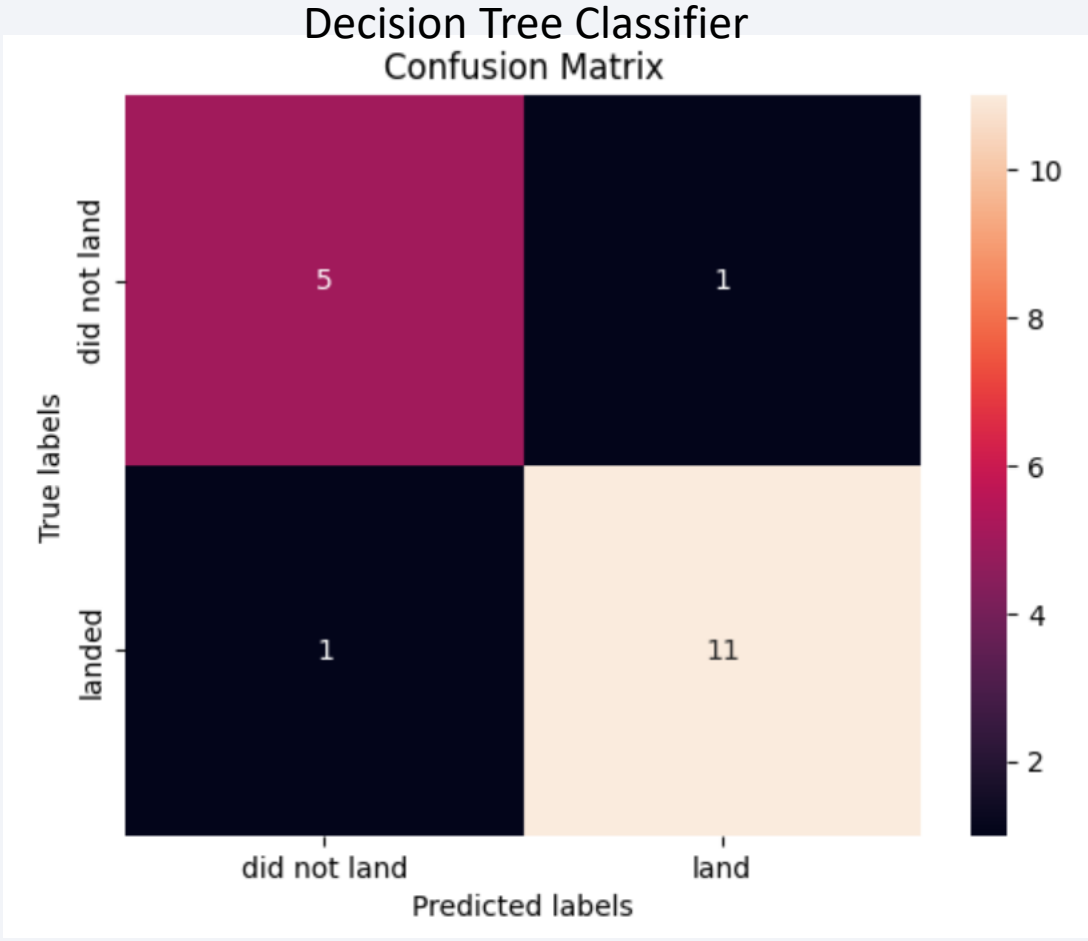
Classification Accuracy

- The 4 ML models tested are compared in the bar chart. The orange bars represent accuracy of predicting test data ("*Score*").
- Decision Tree model is the best-performing, with *Score* = 0.89

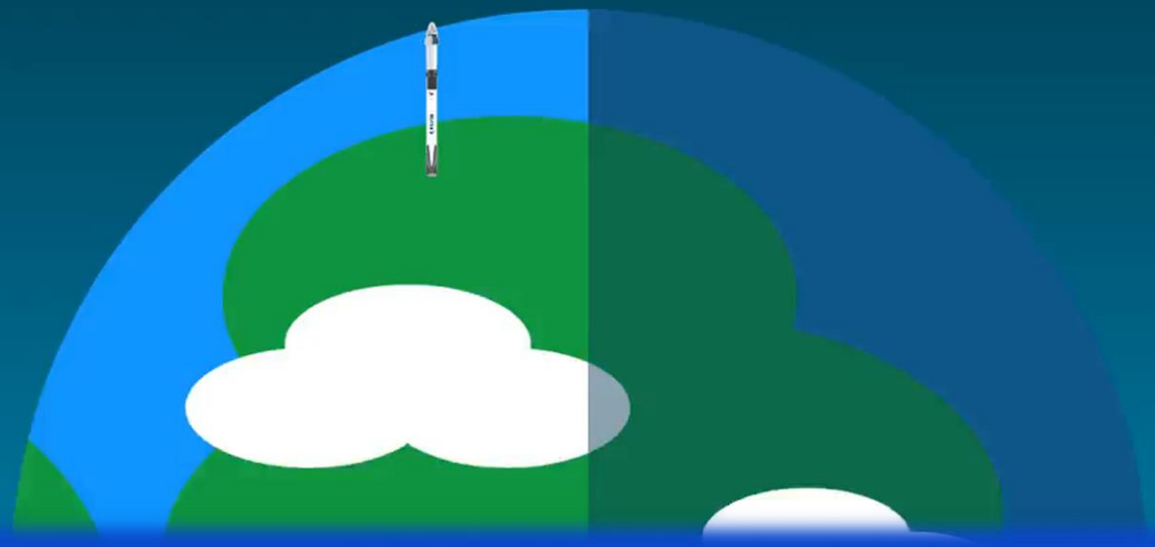


Confusion Matrix

- Confusion matrix: visual summary of the performance of a classification algorithm
- Decision Tree Classification Confusion Matrix to the right shows:
  - 11 True Positives and 5 True Negatives
  - 1 False Positive and 1 False Negative
  - So 16/18 correct predictions, producing an accuracy score of 0.89



Conclusions



Conclusions

- The success of a launch is affected by several factors, in particular:
  - Number of prior launches.
  - Launch site
  - Orbit type
  - Payload
- Success rates have increased over time; success rate of 90% achieved in 2019.
- The launch site with highest success rate is "*KSC LC 39A*"
- All launch sites are situated away from large population centers and close to coastlines to minimize risk; proximity to the equator and transport options reduces costs and time.
- Orbit Types "ES L1", "GEO", "HEO" , "SSO" have a 100% success rate.
- Payload mass under 4000 Kgs have a greater success rate than those 4000 to 10,000 kg
- The ML model best able to predict launch outcome is Decision Tree, with accuracy of 88.9%.

Appendix

- Other files not listed in prior slides:
  - Dataset created: https://github.com/ainuhirath/DS-capstone/blob/main/SPACEXTABLE.xlsx
  - Video file illustrating launch and landing: https://github.com/ainuhirath/DS-capstone/blob/main/SpaceX%20Falcon%209%20launch.mp4

Appendix

- Webscraping: full code to add data to keys:

```
[14]: extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
    # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            launch_dict['Flight No.'].append(flight_number)
            #print(flight_number)
            datatimelist=date_time(row[0])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`
            date = datatimelist[0].strip(',')
            launch_dict['Date'].append(date)
            #print(date)

            # Time value
            # TODO: Append the time into launch_dict with key `Time`
            time = datatimelist[1]
            launch_dict['Time'].append(time)
            #print(time)

            # Booster version
            # TODO: Append the bv into launch_dict with key `Version Booster`

            bv=booster_version(row[1])
            if not(bv):
                bv=row[1].a.string
            launch_dict['Version Booster'].append(bv)
            #print(bv)

            # Launch Site
            # TODO: Append the bv into launch_dict with key `Launch Site`
            launch_site = row[2].a.string
            launch_dict['Launch site'].append(launch_site)
            #print(launch_site)

            # Payload
            # TODO: Append the payload into launch_dict with key `Payload`
            payload = row[3].a.string
            launch_dict['Payload'].append(payload)
            #print(payload)

            # Payload Mass
            # TODO: Append the payload_mass into launch_dict with key `Payload mass`
            payload_mass = get_mass(row[4])
            launch_dict['Payload mass'].append(payload_mass)
            #print(payload)

            # Orbit
            # TODO: Append the orbit into launch_dict with key `Orbit`
            orbit = row[5].a.string
            launch_dict['Orbit'].append(orbit)
            #print(orbit)

            # Customer
            # TODO: Append the customer into launch_dict with key `Customer`
            #customer = row[6].a.string
            customer = list(row[6].strings)[0]
            launch_dict['Customer'].append(customer)
            #print(customer)

            # Launch outcome
            # TODO: Append the launch_outcome into launch_dict with key `Launch outcome`
            launch_outcome = list(row[7].strings)[0]
            launch_dict['Launch outcome'].append(launch_outcome)
            #print(launch_outcome)

            # Booster landing
            # TODO: Append the launch_outcome into launch_dict with key `Booster landing`
            booster_landing = landing_status(row[8])
            launch_dict['Booster landing'].append(booster_landing)
            #print(booster_landing)
```

Thank you!