IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Brielle Boomer>
<March 7, 2025>

Outline

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

Executive Summary

- Summary of methodologies used:
  - Collect Data Through API and Webscraping
  - Transform the Data (Data Wrangling)
  - Explore Data Analysis with SQL and Visuals
  - Folium Interactive Map to Analyze Launch Site
  - Interactive Dashboard to Analyze Launch Records (Plotly Dash)
  - Build Predictive Model to Assess if Falcon 9 Will Land Successfully
- Summary of all results:
  - The report will show that results were captured through:
    - Data Visuals
    - Interactive Dashboards
    - Predictive Model Analysis

Introduction

- Project Background
  - The goal of this project was to analyze and clean SpaceX data to determine the success rate of Falcon 9.
  - SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. If we can determine whether the first stage will land, we can determine the cost of a launch.

- What we are asking…
  - Will Falcon 9 land successfully?
  - Is there any correlations between site launches and success rates?
  - What is the impact of different variables on landing outcomes?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX API
  - Webscraping SpaceX/Falcon Data (Wikipedia)
- Perform data wrangling
  - Training of Supervised Models by Converting Mission Outcomes (0 = unsuccessful, 1 = successful)
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Compare classification models to determine the most effective (KNN, SVM, Classification Trees & Logistic Regression

Data Collection

- For this project, the data was collected through webscraping (wiki) for launch data & SpaceX API
- You need to present your data collection process use key phrases and flowcharts

API Request
- Send request to read SpaceX API Launch Data

Read API Response
- Read the .json response and parse the data frame

Data Wrangling
- Filter data frame for Falcon 9

Convert to CSV
- Convert data frame to CSV to save

[Github Data Collection API]

Perform HTTP GET
- Send GET request to Wikipedia page titled List of Falcon 9 and Falcon Heavy launches

Beautiful Soup
- Parse the data and extract relevant columns/headers

Extract to Data Frame
- Create a data frame by parsing the launch HTML tables

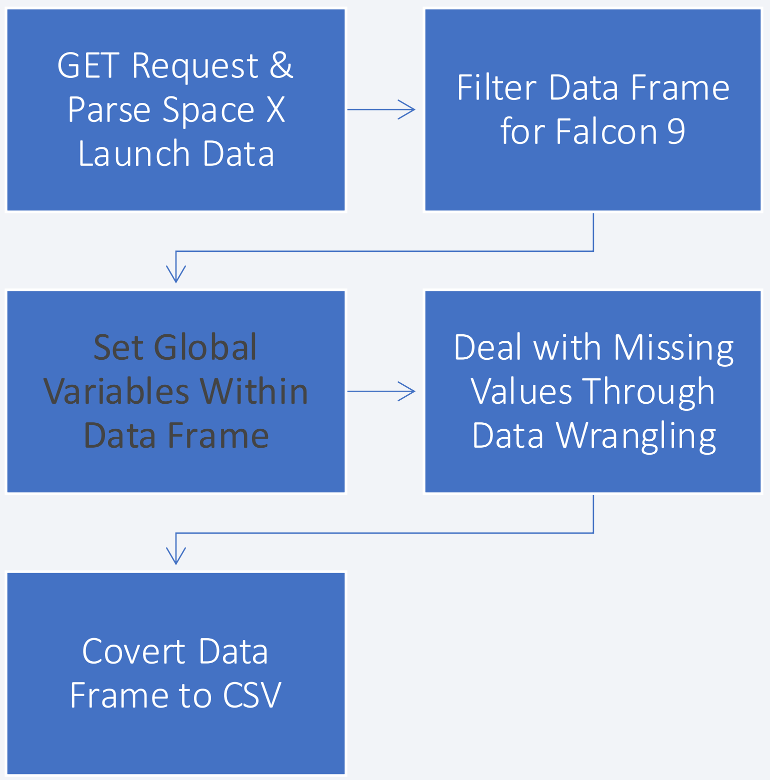
Convert to CSV
- Convert data frame to CSV to save

[Github SpaceX Webscraping]

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Github - Data Collection API



Data Collection - Scraping

- [GitHub - Webscraping]()

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

In [5]:

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

import requests

# URL of the Falcon9 Launch Wiki page
static_url = 'https://en.wikipedia.org/wiki/Falcon_9'  # Replace with the correct URL if needed

# Perform the HTTP GET request and store the response
response = requests.get(static_url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    print("Request successful")
else:
    print(f"Request failed with status code: {response.status_code}")
```

```
Request successful
```

Create a `BeautifulSoup` object from the HTML `response`

In [6]:

```python
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content

import requests
from bs4 import BeautifulSoup

# URL of the Falcon9 Launch Wiki page
static_url = 'https://en.wikipedia.org/wiki/Falcon_9'  # Replace with the correct URL if needed

# Perform the HTTP GET request and store the response
response = requests.get(static_url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    print("Request successful")

    # Create a BeautifulSoup object to parse the HTML content
    soup = BeautifulSoup(response.text, 'html.parser')

    # Now you can work with the 'soup' object to extract data from the HTML page
    # For example, print the title of the page
    print(soup.title.text)
else:
    print(f"Request failed with status code: {response.status_code}")
```

```
Request successful
Falcon 9 - Wikipedia
```

Print the page title to verify if the `BeautifulSoup` object was created properly

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

In [8]:

```python
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`

import requests
from bs4 import BeautifulSoup

# URL of the Falcon9 Launch Wiki page
static_url = 'https://en.wikipedia.org/wiki/Falcon_9'  # Replace with the correct URL if needed

# Perform the HTTP GET request and store the response
response = requests.get(static_url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    print("Request successful")

    # Create a BeautifulSoup object to parse the HTML content
    soup = BeautifulSoup(response.text, 'html.parser')

    # Find all tables on the page
    html_tables = soup.find_all('table')

    # Print the number of tables found and the first table to check
    print(f"Found {len(html_tables)} tables.")
    print(html_tables[0])  # Displaying the first table to check
else:
    print(f"Request failed with status code: {response.status_code}")
```

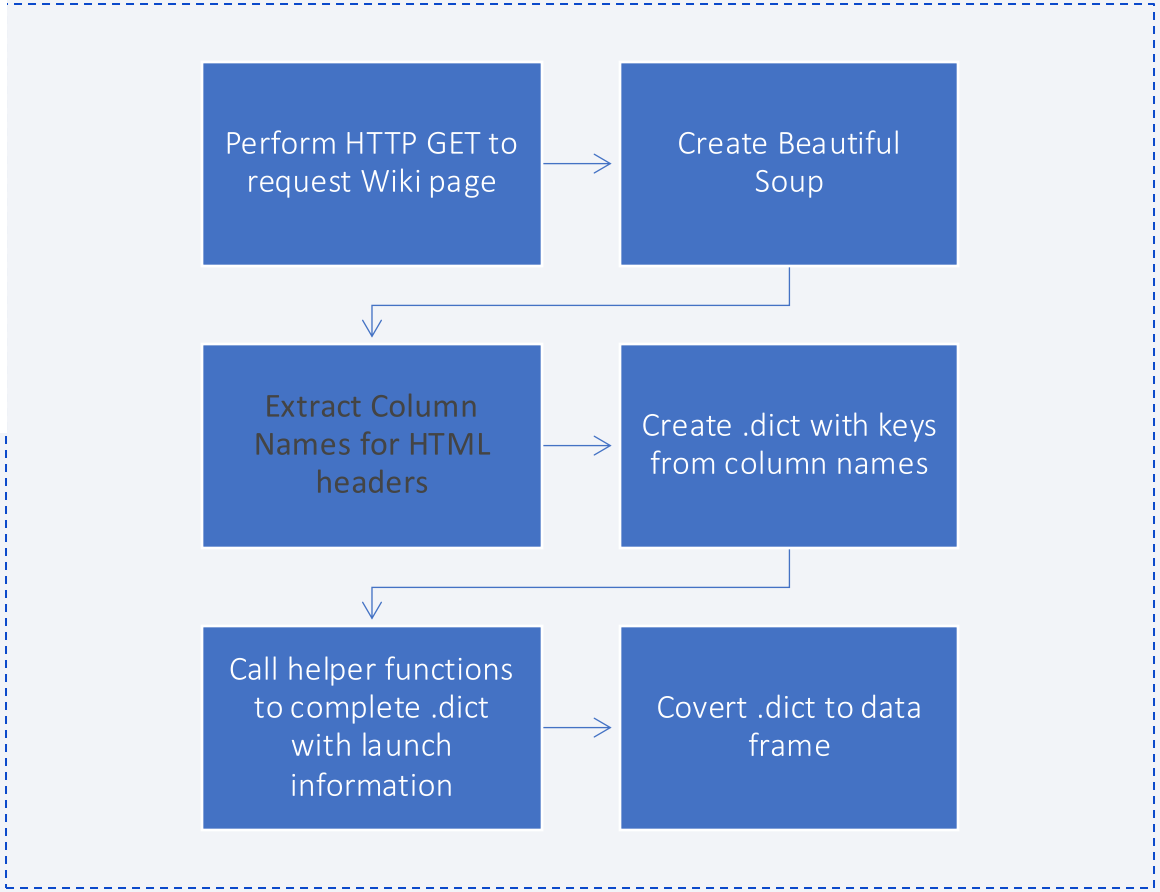
```
Request successful
Found 25 tables.
```

TASK 3: Create a data frame by parsing the launch HTML tables

We will create an empty dictionary with keys from the extracted column names in the previous task. Later, this dictionary will be converted into a Pandas dataframe

In [ ]:

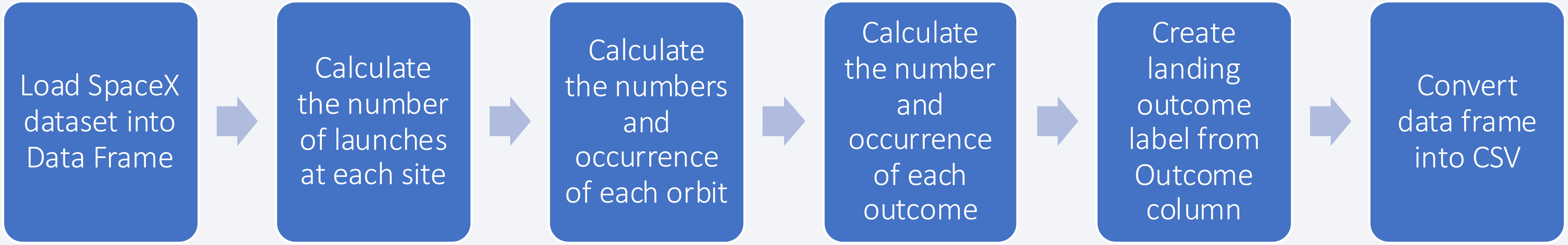
```python
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```



Data Wrangling

- How was the data processed?
  - Through EDA, there are patterns in the data
  - Define levels for training supervised models
  - The data set contained various outcomes that were converted into training labels (0,1). Those scenarios were then used to create labels:
    - True Ocean, False Ocean, RTLS, False RTLS, True ASDS, False ASDS

- [GitHub – Data Wrangling]



EDA with Data Visualization

- The following charts were used during EDA to gain further insights into the data:
  - Scatter Plot
    - Shows correlation between two variables to observe patterns
    - Charts Plotted:
      - Flight Number vs. Launch Site
      - Flight Number vs. Payload Mass
      - Payload Mass vs. Launch Site
      - Flight Number vs. Orbit Type
      - Payload Mass vs. Orbit Type
  - Bar Chart
    - Used to compare variables at a given time frame. Bar charts make it easier to see what is the most common result and compare the other groups as well.
      - Charts Plotted:
        - Success Rate for Each Orbit Type (ES-L1, GEO, GTO, HEO, ISS, LEO, MEO, PO, SO, SSO, VLEO)
  - Line Chart
    - Used to track changes over a period of time and depicts trends.
      - Charts Plotted:
        - Average Launch Success Yearly Trend (2010-2020)

GitHub EDA – Data Visualization

EDA with SQL

- Below is the list of SQL queries performed on the SpaceX data set:

[GitHub – EDA w: SQL]

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 successful landing outcome in ground pad was achieved
6. List the names of the boosters which have success in drone ship & have payload mass greater than 4000 but less than 6000
7. List the total number of successful and failure mission outcomes
8. List the names of the booster_versions which have carried the maximum payload mass (use a subquery)
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 & 2017-03-20 in descending order

Build an Interactive Map with Folium

- Folium mapping helps to analyze geospatial data with more interactive visual analytics. It is used to better understand factors in data such as location/proximity of launch sites and how it impacts success rates.
- The following map objects were created:
  1. Mark all launch sites on a map
     - Are all launch sites in proximity to the Equator line? NO
     - Are all launch sites in very close proximity to the coast? YES
  2. Mark the success/failed launches for each site on the map
     - Added a markercluster to show launch sites success and failure by using green and red
  3. Calculate the distances between a launch site and to its proximities
     - Are the launch sites in close proximity to railways? YES
     - Are the launch sites in close proximity to highways? YES
     - Are the launch sites in close proximity to coastlines? YES
     - Do launch sites keep certain distance away from cities? YES

GitHub - Folium

Build a Dashboard with Plotly Dash

[GitHub – Plotly Dash]

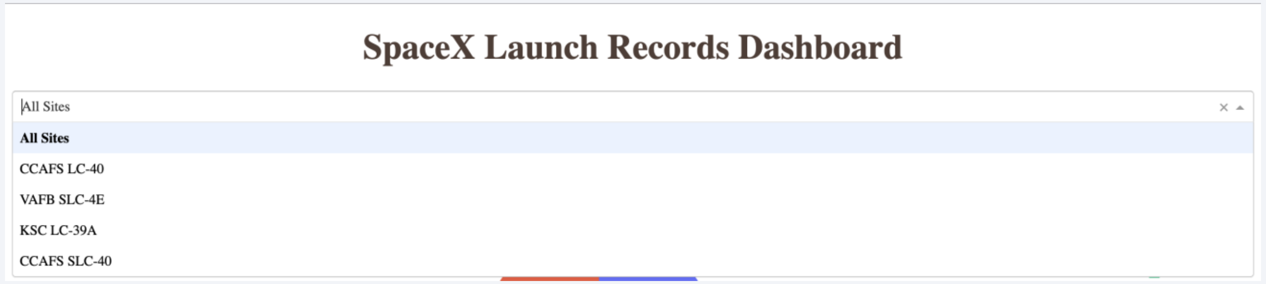
- A Plotly Dash was built to demonstrate interactive visual analytics of SpaceX launch data. A launch site drop down, pie chart, payload range slide and a scatter plot were added to the dashboard

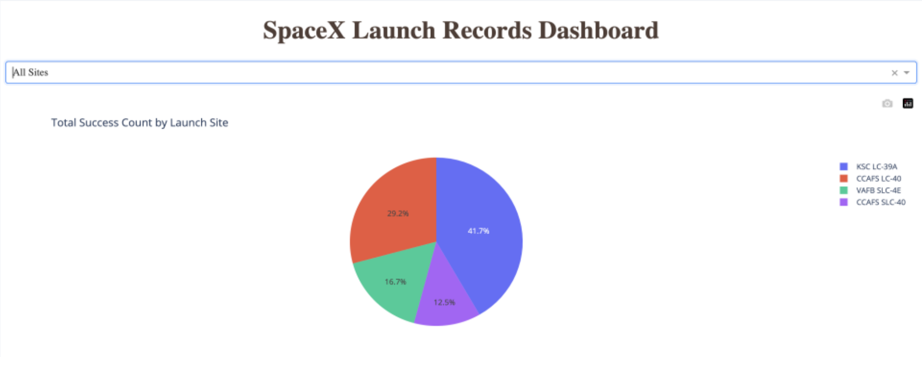
  1. Launch Site Drop Down: added the ability to filter by site and visualize success rates
  2. Pie Chart: showed total success of launches (when all sites is selected in drop down) or broke down the success and failures by launch sites (when selected)
  3. Payload Range Slide: used to find variable payload correlated to mission outcome. Made is easier to select different payloads to identify visual patterns.
  4. Scatter Plot: created a scatter plot with Payload (X-Axis) vs. Launch Outcome (Y-Axis) to visually observe the correlation between the two for all sites or selected site(s)

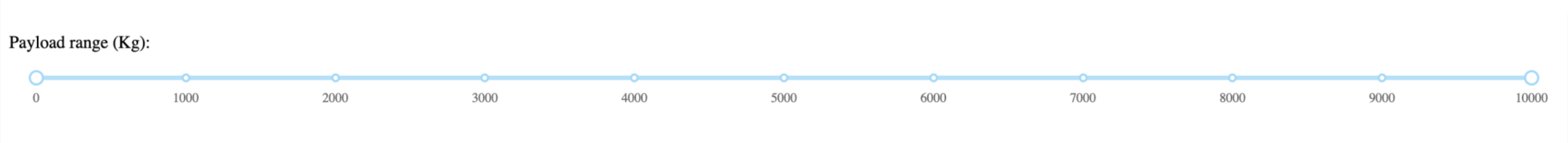
Build a Dashboard with Plotly Dash (Cont'd)

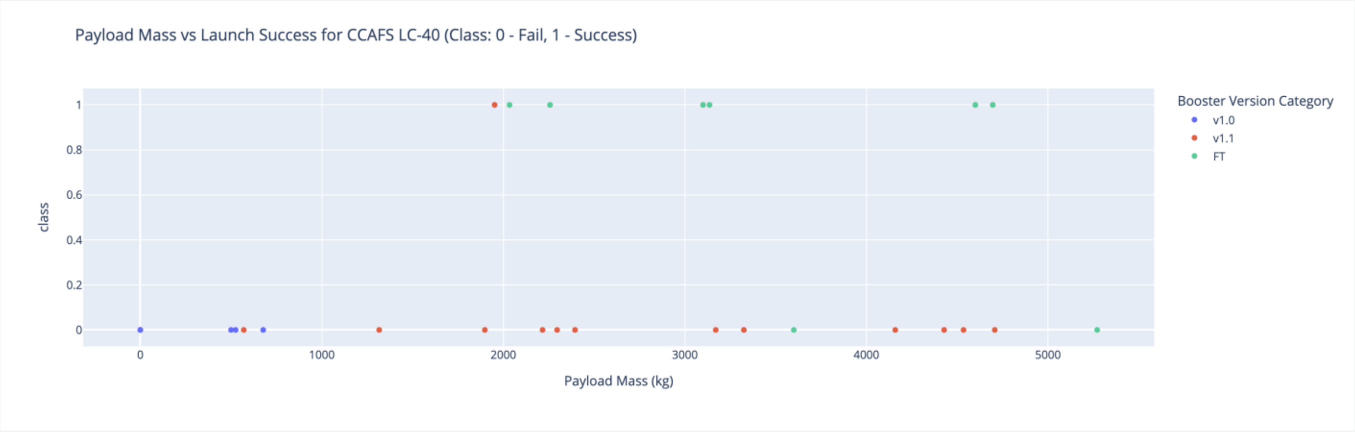
- Here are visuals of the dashboard created and outputs.

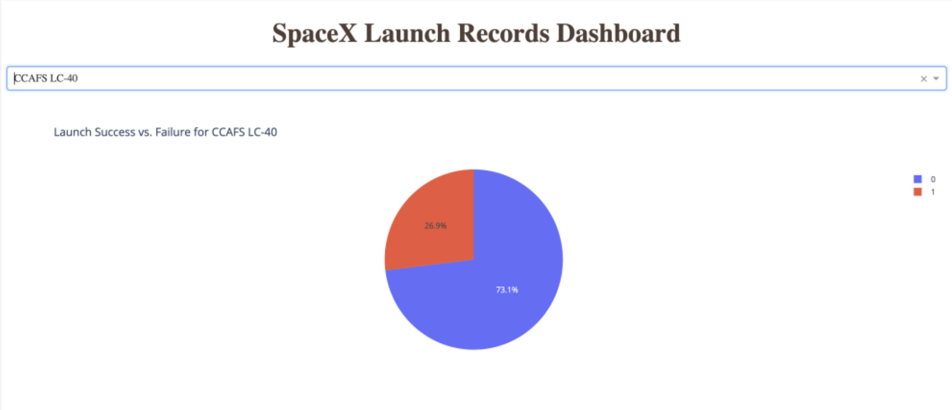
GitHub – Plotly Dash



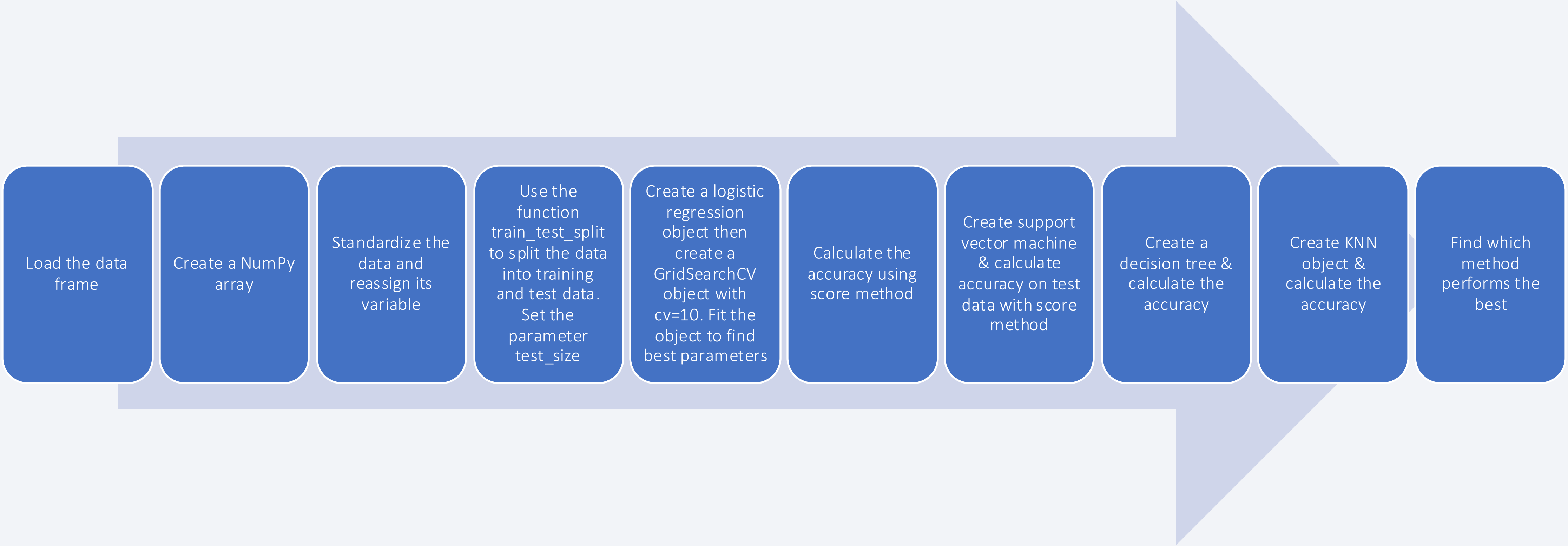








Predictive Analysis (Classification)



GitHub – Predictive

Results

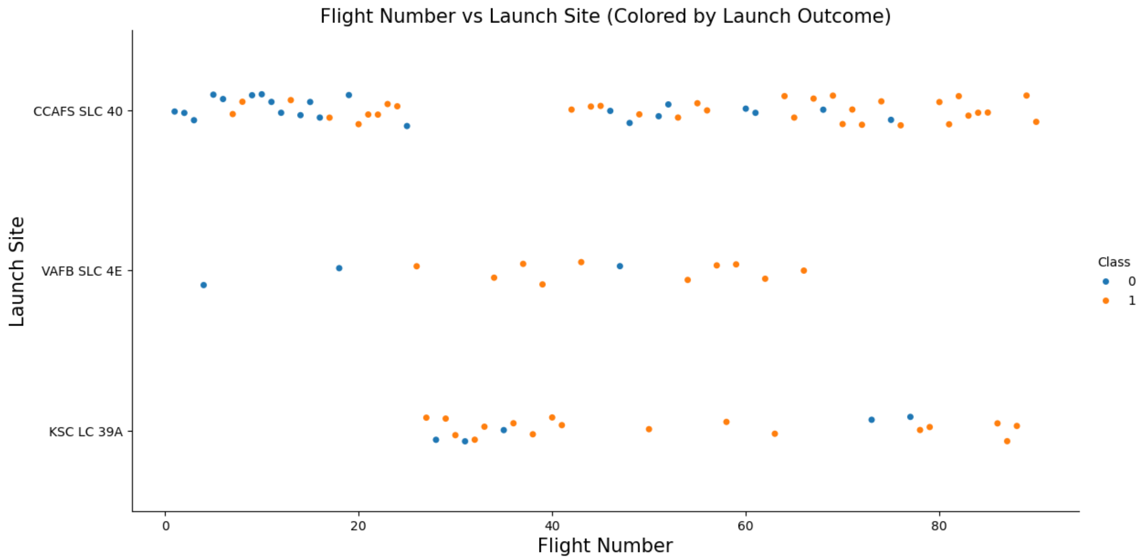
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

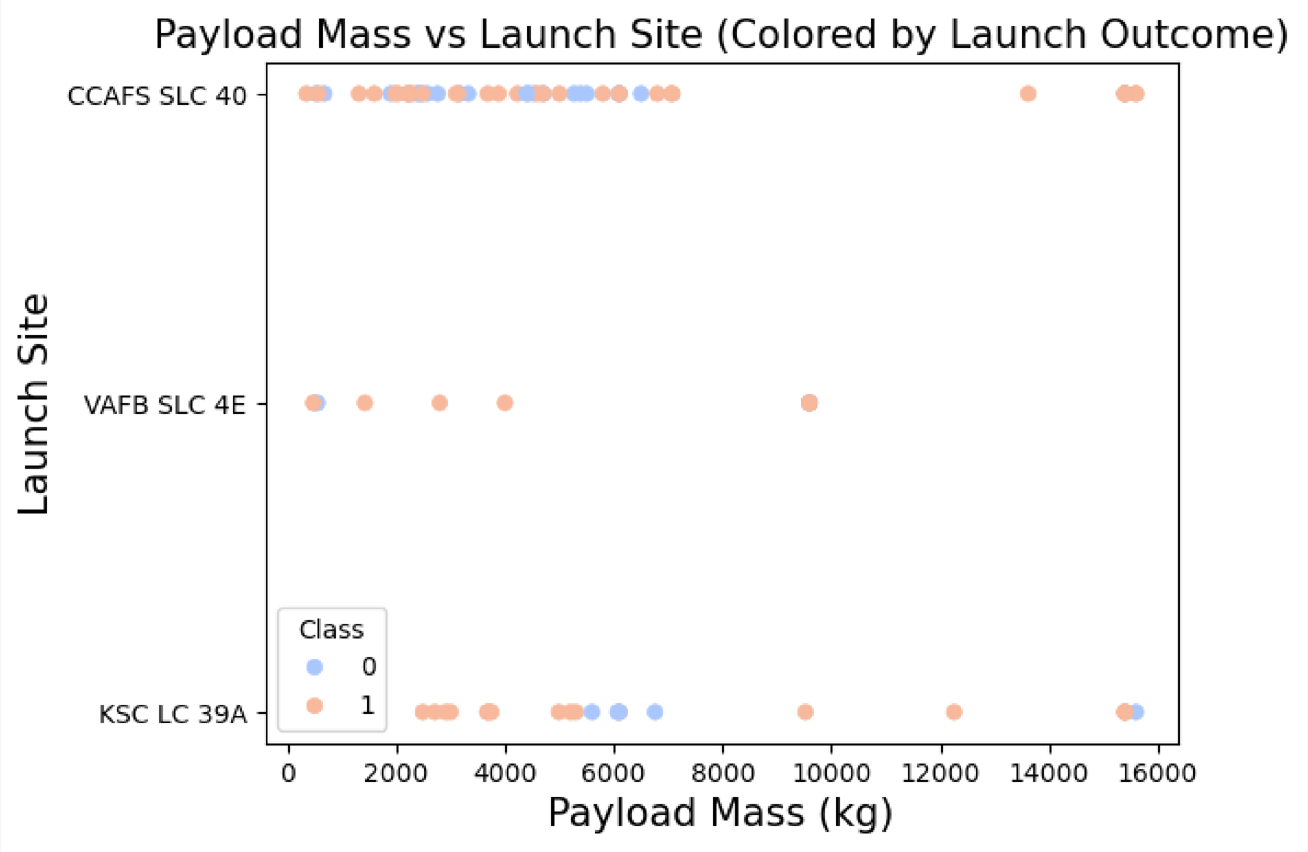
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

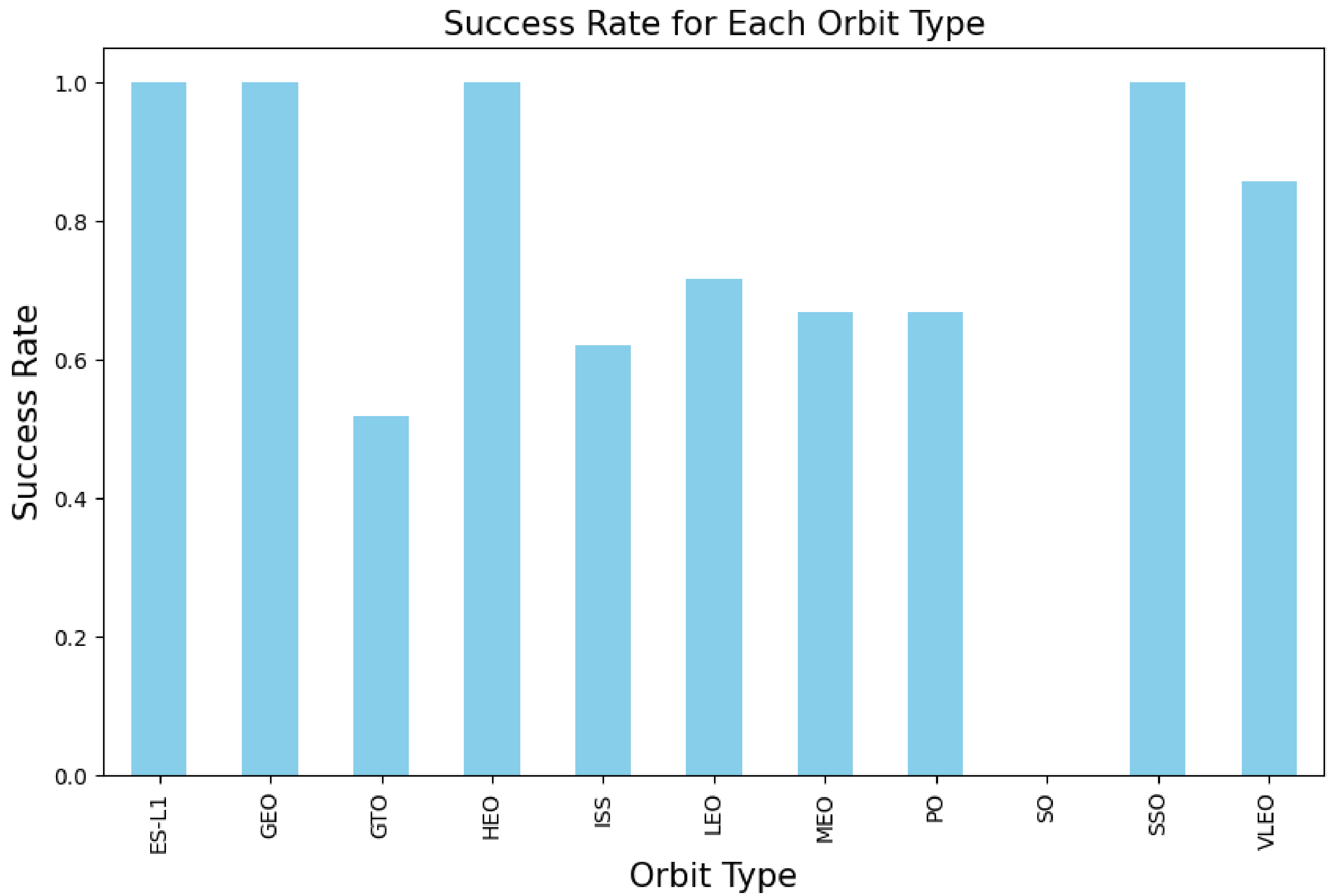
- Success rate increases as number of flights increases (Class 1, Orange)
- CCAFS SLC 40 has had the highest number of flights
- VAFB SLC 4E has had the least amount of flights





Payload vs. Launch Site

- No rockets where launched at VAFB SLC 4E with a payload greater than 10,000kg
- Successful launches increase as payload mass increases, specifically at launch site CCAFS SLC 40
- However, there is no clear correlation between payload and launch site

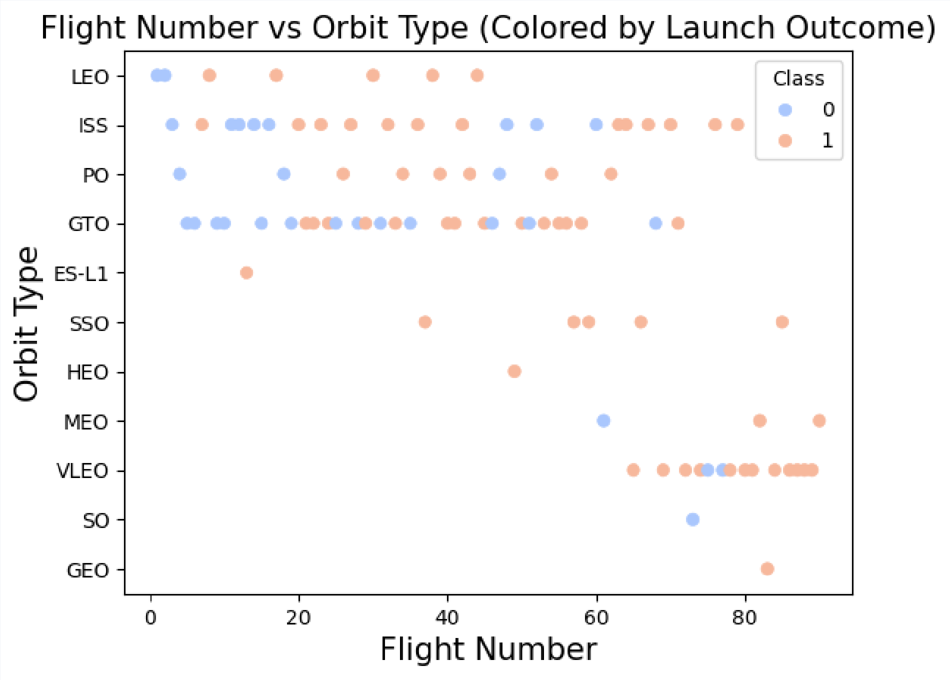


Success Rate vs. Orbit Type

- ES-L1, GEO, HEO & SSO have the highest success rates
- GTO has the lowest success rate while there is no data for SO

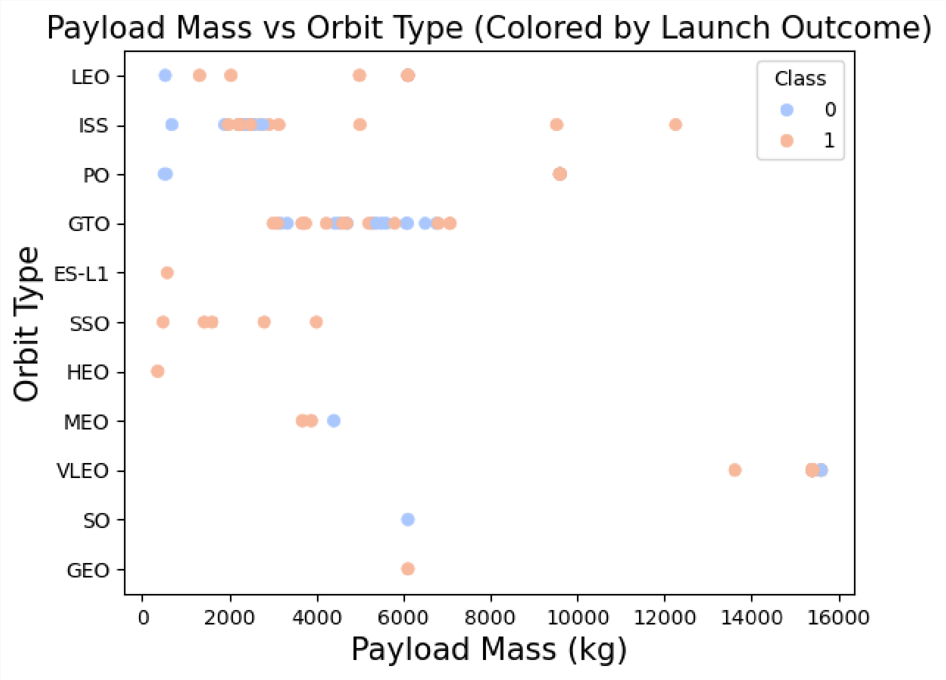
Flight Number vs. Orbit Type

- For LEO, orbit success is related to number of flights
- For GTO, there is no relation between orbit success and flight number
- It took over 80 flights for GEO to have its first success



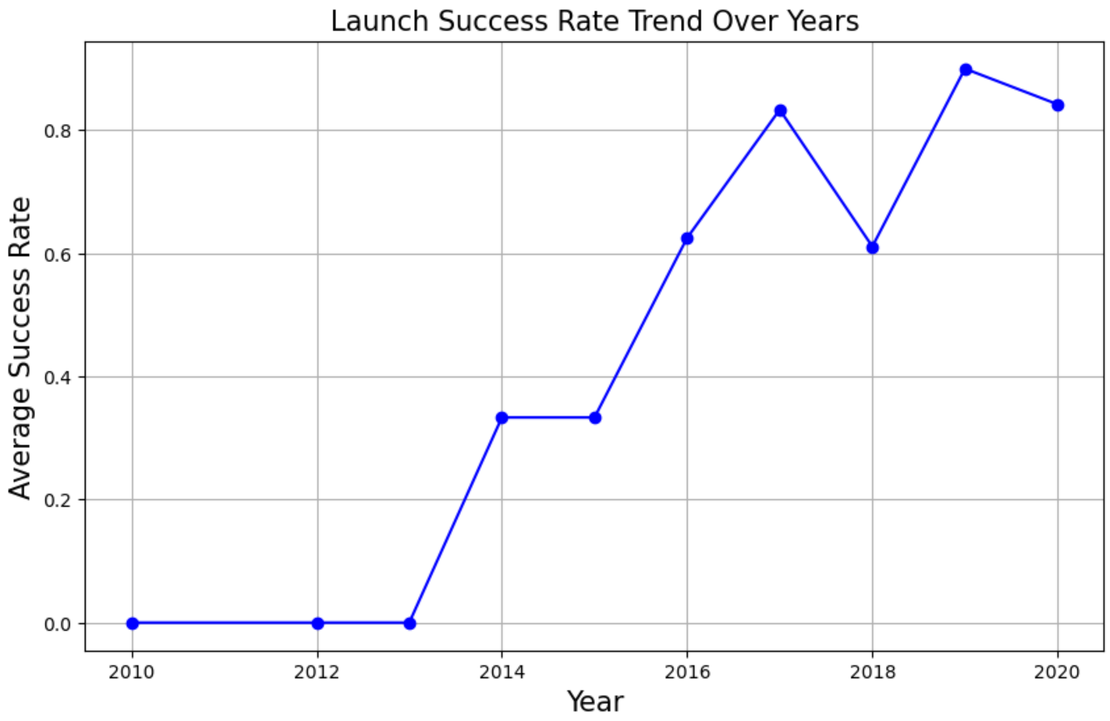
Payload vs. Orbit Type

- For GTO, it is hard to tell if payload mass effects the launch as it is difficult to tell the successful from unsuccessful launches apart
- Payload mass seems to have no effect on launch success at SSO, LEO
- Heavier loads lead to more successful launches for PO and ISS



Launch Success Yearly Trend

- Success rates continually increase from years 2013 to 2020
- Success rates hit 80% in 2020, with highest success rate being around 85% in 2019
- Success rates show a decrease between 2017 - 2018



All Launch Site Names

- Step 1 – Query:

```
%sql SELECT DISTINCT `Launch_Site` FROM SPACEXTBL;
```

- Step 2 – Explanation:
    - 'select distinct' returns only the unique values from the queried column
- Step 3 – Result:

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

Launch Site Names Begin with 'CCA'

- Step 1 – Query:

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

- Step 2 – Explanation:
  - The above query returns first 5 results (LIMIT) where Launch Site starts with CCA (LIKE)
- Step 3 – Result:

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

- Query:

```
%sql SELECT SUM("PAYLOAD_MASS__KG_") AS "Total Payload Mass (kg)" FROM SPACEXTBL WHERE "Customer" = 'NASA (CRS)';
```

- Explanation:
  - Function SUM() adds up the total of specified column while customer specifics which rows to pull from for the total

- Result:

Total Payload Mass (kg)
45596

Average Payload Mass by F9 v1.1

- Query:

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

- Explanation:
  - Function AVG() calculates the average of the specified column while WHERE … identifies which rows to include

- Result:

AVG("PAYLOAD_MASS__KG_")
2928.4

First Successful Ground Landing Date

- Query:

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

- Explanation:
  - MIN() identifies the oldest or first date in this query request
  - WHERE "landing outcome" = 'success (ground pad)' specifies which MIN date we want to use in the result
- Result:

first_successful_landing_date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Query:

```
%sql SELECT "Booster_Version" FROM SPACEXTBL WHERE (PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000) AND (LANDING_OUTCOME = 'Success (drone ship)')
```

- Explanation:
    - Using WHERE with a column header and range of greater than and less than (<>) along with AND function to locate the rows helps to provide the below results. The results show the successful landings via drone ship thanks to this query.

- Result:

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

Total Number of Successful and Failure Mission Outcomes

- Query:

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) AS 'counts' FROM SPACEXTBL GROUP BY Mission_Outcome
```

- Explanation:
  - GROUP BY allows similar data to be condensed to a specified column.
  - The number of mission outcomes are grouped by type and count

- Result:

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

Boosters Carried Maximum Payload

- Query:

```
%sql SELECT "Booster_Version" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);
```

- Explanation:
  - The query returns booster versions & payloads where MAX is met
- Result:

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

- Query:

```
%sql SELECT "Landing_Outcome", "Booster_Version", "Launch_Site" FROM SPACEXTBL WHERE "Landing_Outcome" = 'Failure (drone ship)' AND substr("Date", 1, 4) = '2015';
```

- Explanation:
    - The query returns landing outcome, booster version and launch site where failure (drone ship) occurred during the year 2015
    - The AND clause returns where variables are 'True'
    - The 'substr' "Date" helps the SQL read for 2015 only

- Result:

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

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

- Query:

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

- Explanation:
  - `"Landing_Outcome"`: This selects the column for the landing outcomes like `"Failure (drone ship)"` or `"Success (ground pad)"`.
  - `COUNT(*) AS outcome_count`: This counts the number of occurrences for each landing outcome.
  - `WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20'`: This filters the data to include only records where the `Date` falls between June 4, 2010, and March 20, 2017.
  - `GROUP BY "Landing_Outcome"`: This groups the data by landing outcome, allowing the count of each type to be calculated.
  - `ORDER BY outcome_count DESC`: This orders the results by the count of each outcome in descending order.

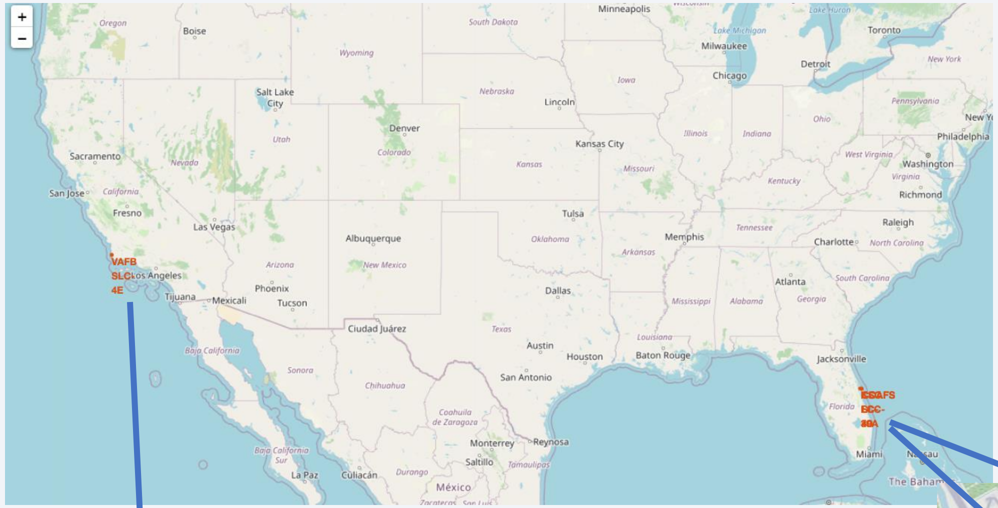
- Result:

Landing_Outcome	outcome_count
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

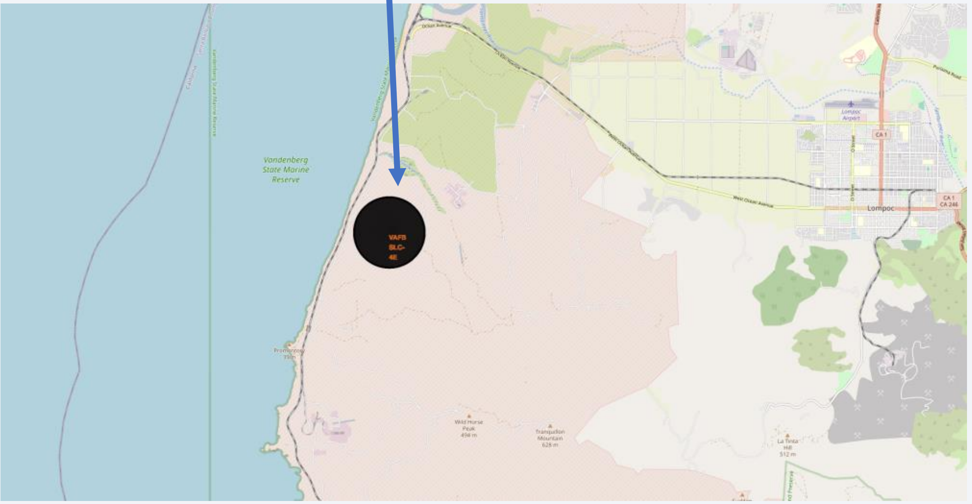
Section 3

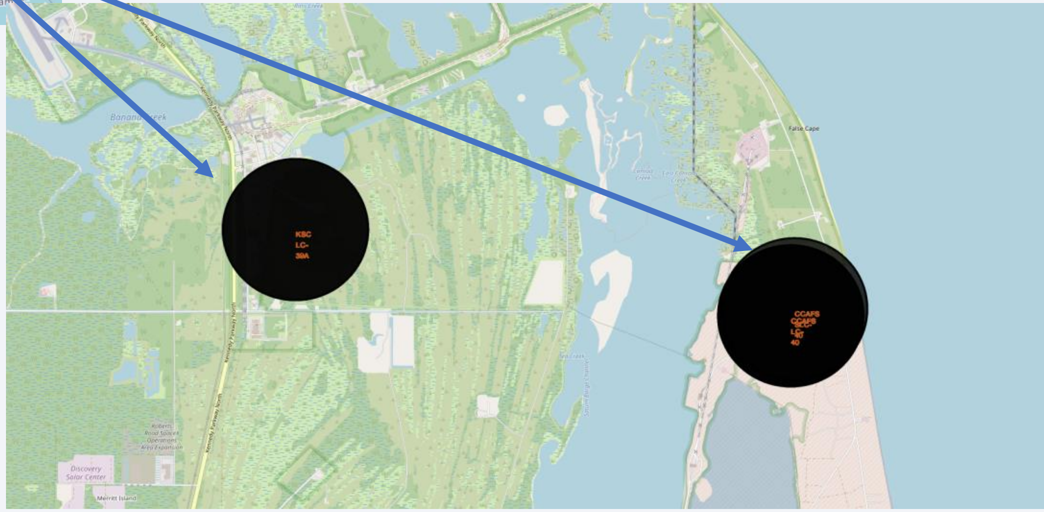
Launch Sites Proximities Analysis

SpaceX Falcon 9 Launch Sites Map

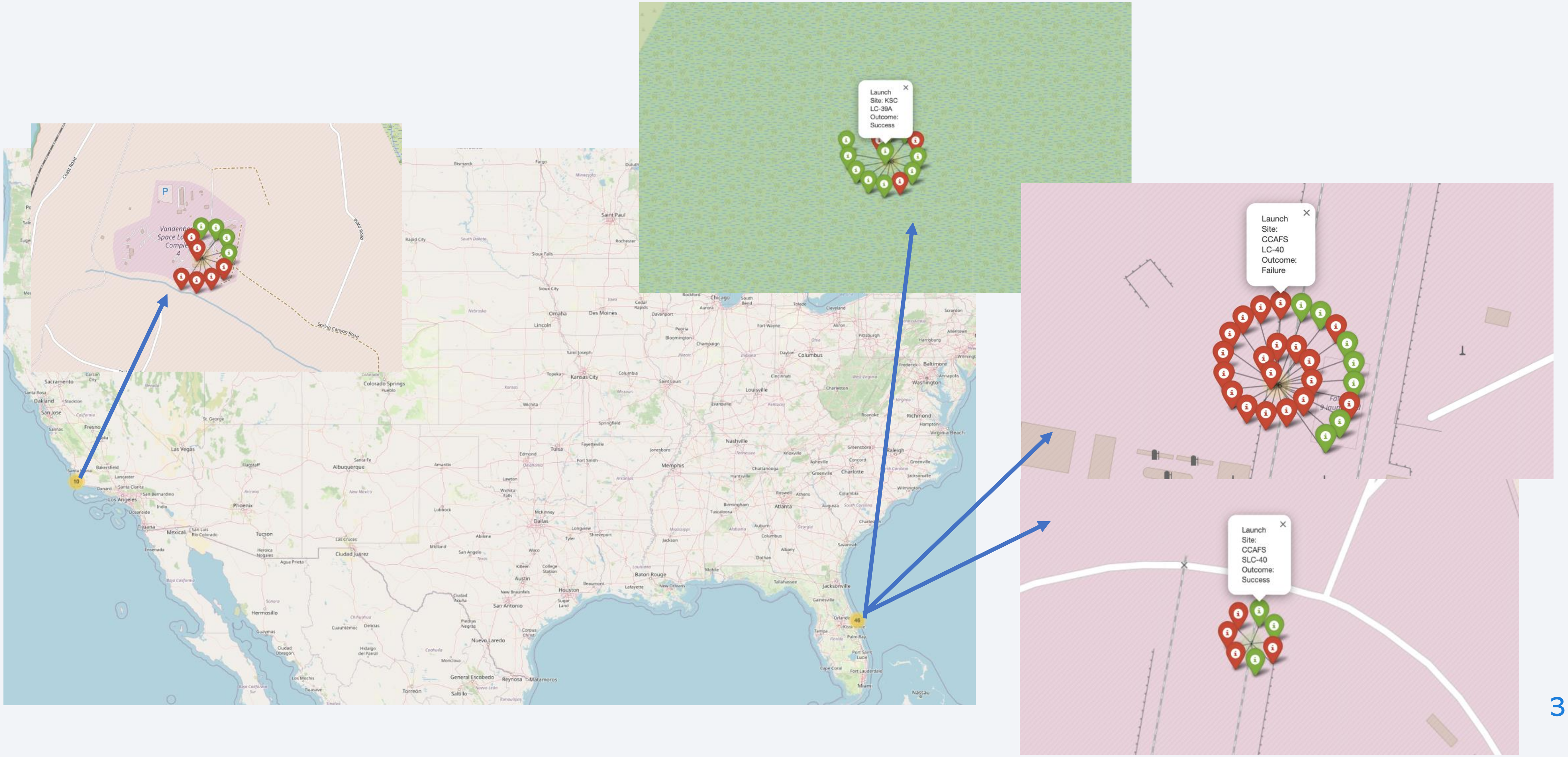


- The global map shows that the launch sites are primarily located in Florida and California on the coasts.
- The zoomed in maps show more exact locations of the launch sites.
- Some launch sites listed:
  - VAFB SLC 4E (CA)
  - CCAFS LC-40 (FL)
  - KSC LC-39A (FL)
  - CCAFS SLC-40 (FL)





SpaceX Falcon 9 – Success/Failure Launch Map For All Sites



SpaceX Falcon 9 – Success/Failure Launch Map For All Sites (Cont'd)

- As noted on the previous slide, the global map shows how many launches at each site (Florida & Cali coasts)
- The zoomed in versions (there are 4) show how many launches were done at each site
- The red color indicates failed launches. The green color indicates successful launches.
- When a launch is clicked you can see in the figures that it provides a popup with the name of the launch site

SpaceX Falcon 9 – Proximity Map

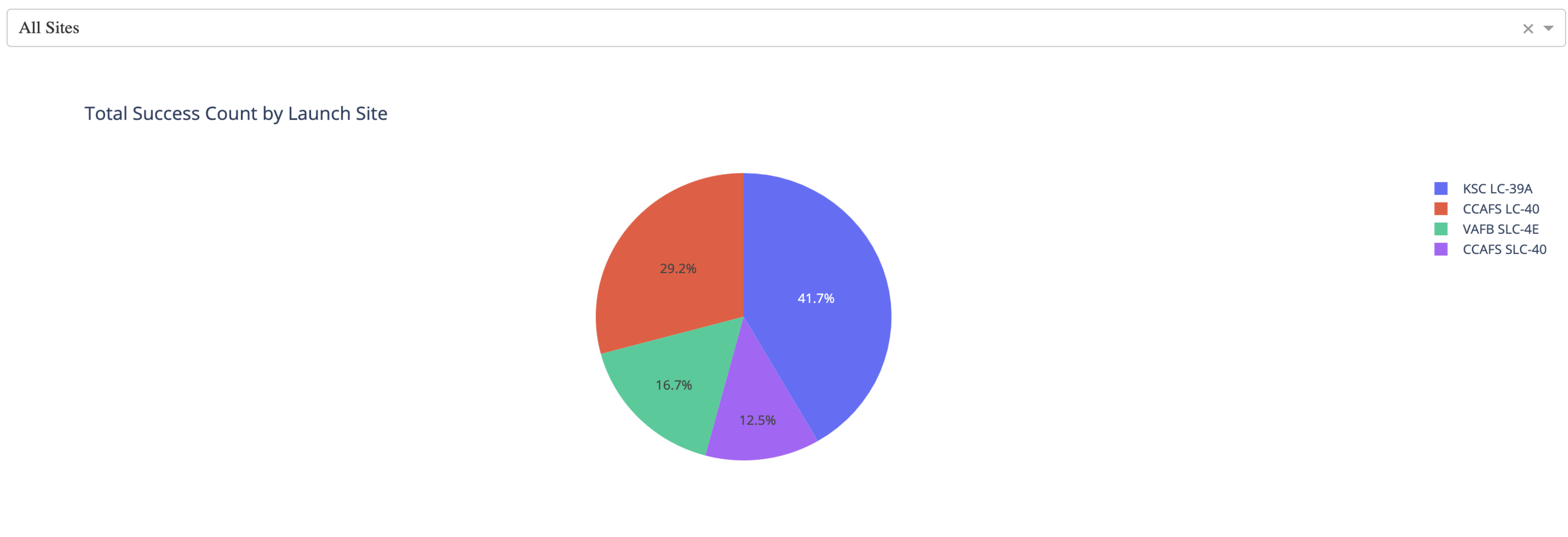
- As pictured, you can see the distance marker between the launch site and coastline
- It is general standing that launch sites be placed a far distance away from cities to mitigate any potential damage

Section 4

Build a Dashboard with Plotly Dash

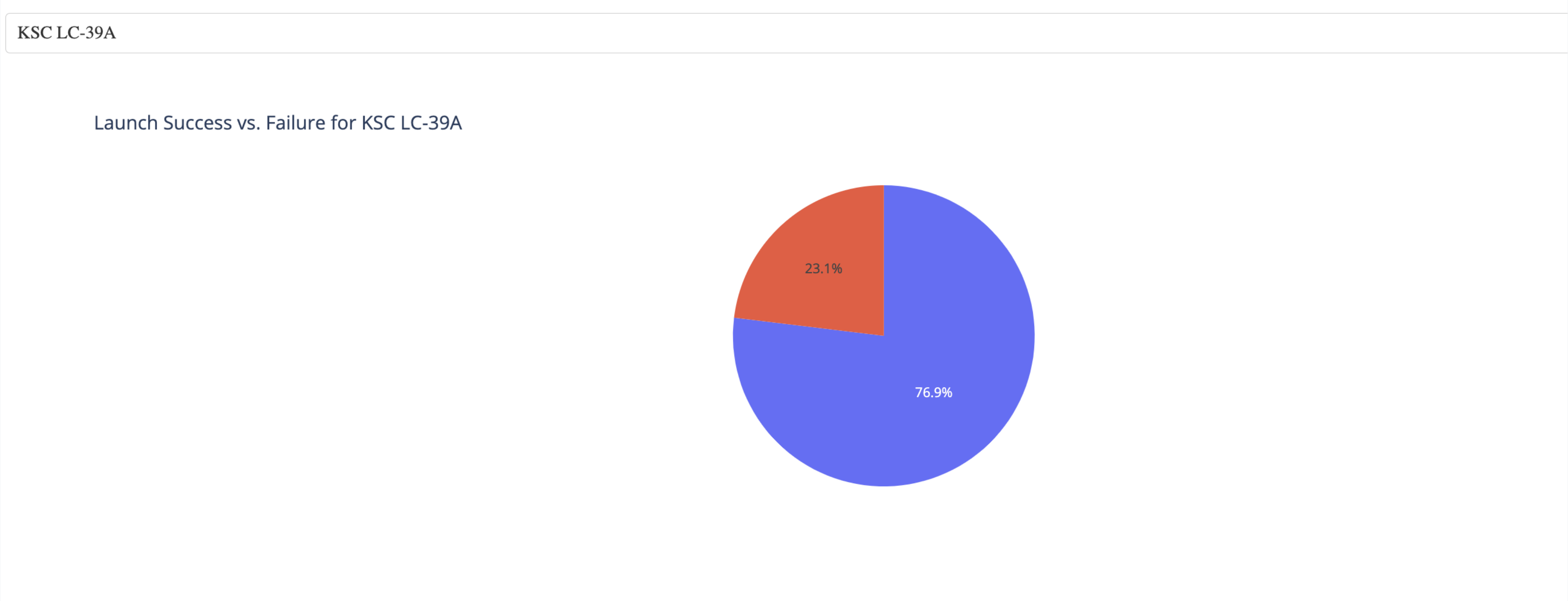
Launch Success Count – All Sites

- Launch site KSC LC-39A has the highest success rate of all sites
- Launch site CCAFS SLC-40 has the lowest success rate of all sites

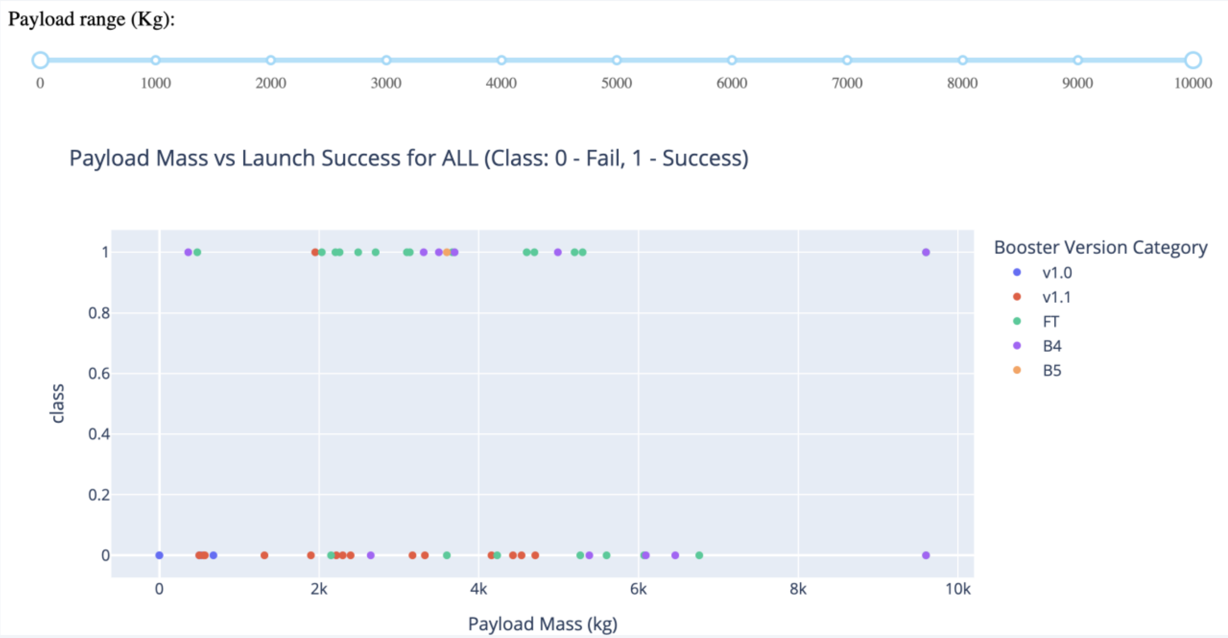


Launch Site With Highest Launch Success Ratio

- The launch site with the highest success ratio is KSC LC-39A
- Blue (Class 1) is the number of successful launches at 76.9%
- Red (Class 0) is the number of failed launches at 23.1%



Payload vs. Launch Outcome – All Sites



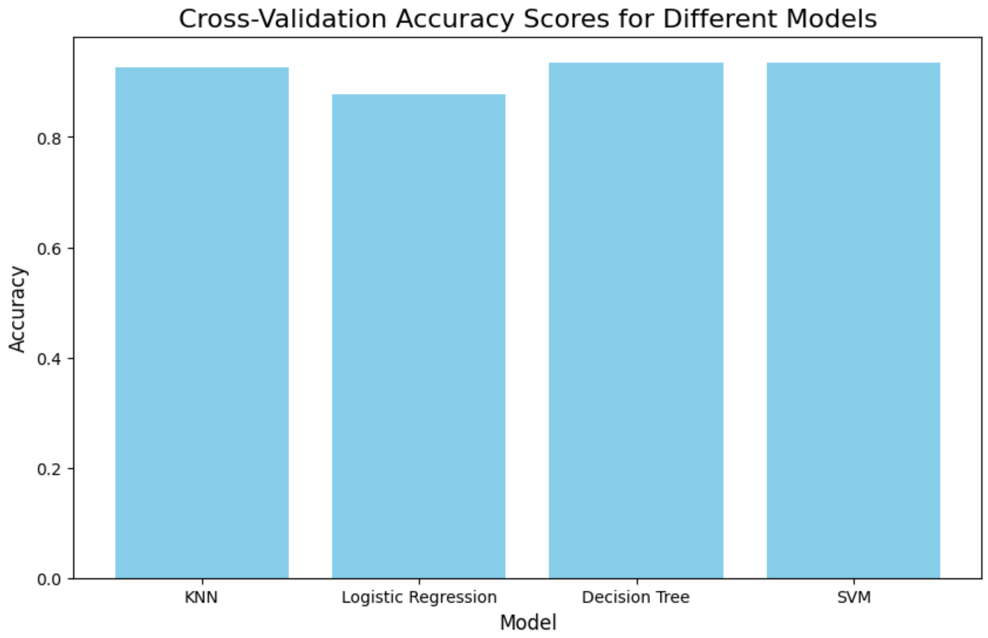
- The most successful launches fall between a payload range of 2000 to 6000
- Booster 'FT' has the most successful launches of all versions
- B4 is the only booster with successes above a 6000 payload

Section 5

Predictive Analysis (Classification)

Classification Accuracy

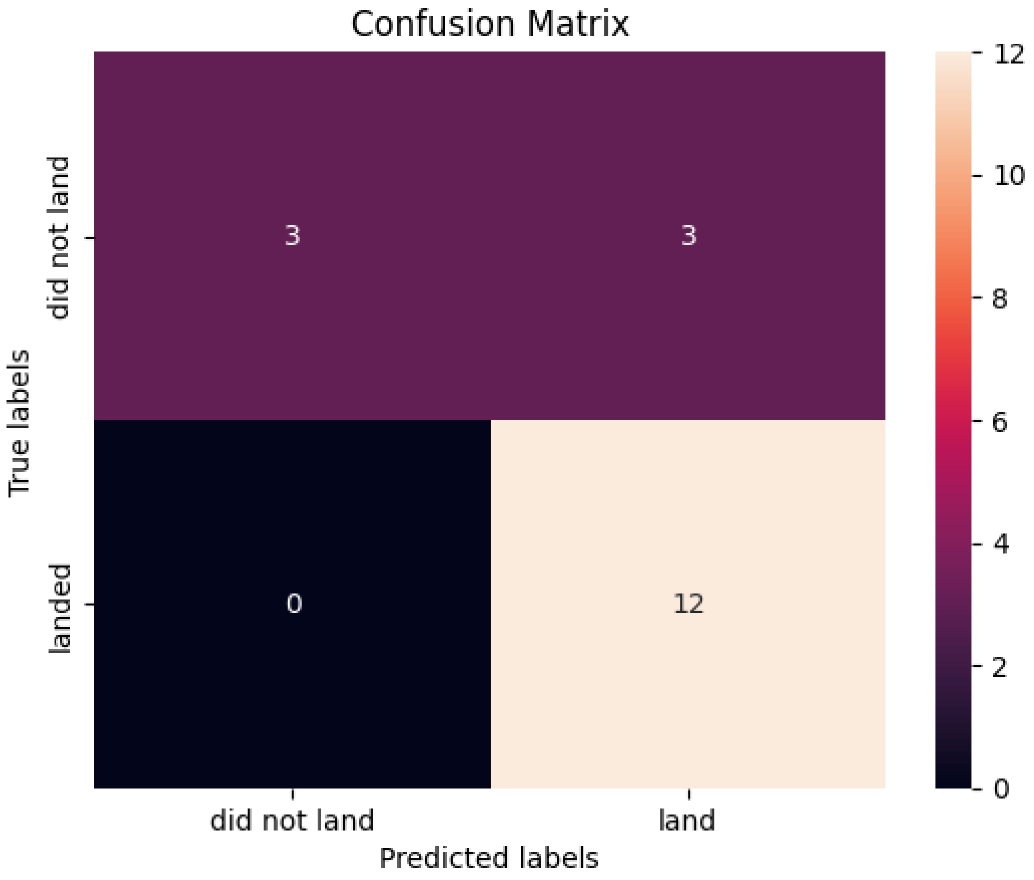
- According to analysis, SVM had the highest accuracy score at 94%
- Logistic Regression Model has the lowest accuracy score at 88%
- The test data accuracy was the same across all models at ~83%



Method Performance			
	Type	Accuracy	Test Data Accuracy
0	KNN	0.9264	0.833
1	SVM	0.9355	0.833
2	Decision Tree	0.9345	0.833
3	Logistic Regression	0.8773	0.833

Confusion Matrix

- The confusion matrix is the same for all models represented in predictive analysis (LR, SVM, KNN, Decision Tree)
- The classifier made 18 predictions
- 12 scenarios predicted at 'Yes' for landing (True Positive)
- 3 scenarios (top right) predicted at 'Yes' for landing and they did not (False Positive)
- 3 scenarios (top left) prediced at 'No' for landing and they did not land successfully (True Negative)
- The classifier is correct about 83% of the time with a misclassification/error rate of ~16.5%



Conclusions

- As flight rates increase, the number of successful landings increases
- Success rate improves as the payload increases although there is no correlation between payload mass & success rate
- Launch site success rate increased 80% in the past 8 years (2013-2020)
- Launch site KSC LC-39A has the highest success rate while Launch site CCAFS SLC-40 has the lowest
- Orbits ES-L1, GEO, HEO & SSO have the highest launch successes while GTO has the lowest
- Launch sites are strategically placed away from cities and closer to railways, coastlines or highways
- Decision Tree is the best performing machine learning model with 87.% accuracy rate. All three models score at an 83% accuracy. It is important to note, that more data may assist in tuning the models for a better fit.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!