IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Miguel Jaime Correa
June 21st, 2025

Outline

Executive Summary

1. Key Methodologies Employed

- For the preparation of this report, the following concepts and methods were used for data collection and analysis, building and evaluating machine learning models, and generating predictions:
- **Multifaceted Data Collection:** The information required for this study was obtained through two primary channels:
  - **API (Application Programming Interface):** An API was utilized to access structured and updated data on Falcon 9 launches, ensuring accuracy and efficiency in acquiring large volumes of relevant information.
  - **Web Scraping:** To complement the API-obtained information, web scraping techniques were implemented. This allowed for the extraction of data from unstructured web sources, such as historical pages or specific details not available via the API, thereby enriching the dataset.
- **Rigorous Data Transformation and Preparation:** Once collected, the raw data underwent an exhaustive process of **cleaning and transformation (Data Wrangling)**. This phase was crucial to ensure the quality and consistency of the information, addressing issues such as missing values, inconsistent formats, duplicates, and errors, thereby preparing the data for meaningful analysis and modeling.
  - **In-depth Exploratory Data Analysis (EDA):** The initial understanding of the data was achieved through detailed Exploratory Data Analysis. For this, the following were used:
  - **SQL Queries:** SQL was employed to manipulate, filter, and aggregate data directly within databases, allowing for quick and efficient exploration of relationships and trends.
  - **Data Visualizations:** Various graphical representations (scatter plots, histograms, box plots, etc.) were created to identify patterns, anomalies, and relationships between variables, facilitating an intuitive understanding of the dataset.
- **Geospatial Visualization with Folium:** To analyze the importance of launch site locations, an **interactive map was constructed using the Folium library**. This tool allowed for geographical visualization of launch sites and evaluation of their proximity to other relevant elements, such as landing zones, providing a crucial spatial perspective for the analysis.
- **Interactive Dashboard with Plotly Dash:** To enable dynamic, real-time exploration of launch records, an **interactive dashboard was developed using Plotly Dash**. This dashboard offers users the ability to filter, group, and visualize launch data in various ways, revealing trends and patterns that might not be apparent from static analysis.
- **Development of Advanced Predictive Models:** The culmination of the methodological process was the construction of a **predictive model** whose main objective is to forecast whether the first stage of the Falcon 9 rocket will achieve a successful landing. This model incorporates insights obtained from the previous analysis phases and is designed to offer robust and reliable predictions.

Executive Summary

2. Key Results Presentation

This report aims to share the findings and conclusions of this study clearly and accessibly, using various formats to optimize understanding:

- **Detailed Data Analysis Results:** Key discoveries obtained during exploratory data analysis and data preparation will be presented. This will include descriptive statistics, identified correlations, and any significant patterns found.
- **Data Visualizations and Interactive Dashboards:** Findings will be illustrated with a series of **static data visualizations** (charts, tables) that concisely summarize complex information. Additionally, **links or integrations to the interactive dashboard** developed with Plotly Dash will be included, allowing readers to explore the data themselves and delve into the details.
- **Exhaustive Predictive Model Analysis Results:** The results of the machine learning model will be detailed, including:
  - **Model Evaluation Metrics:** Key metrics such as precision, recall, F1-score, and the Area Under the Receiver Operating Characteristic (ROC) curve (AUC) will be presented to assess the model's performance.
  - **Feature Importance Analysis:** It will be explained which variables or features are most influential in predicting landing success.
  - **Discussion of Predictions:** Examples of predictions will be provided, and the model's behavior under different scenarios will be analyzed.

Introduction

- **Project background and context**

The burgeoning private space travel sector has brought the space industry increasingly into the mainstream, making it more accessible to the general population than ever before. Despite this rapid expansion and growing interest, the substantial cost of launching payloads into space remains a significant barrier for new entrants and existing competitors in the burgeoning space race.

In this competitive landscape, SpaceX stands out with a unique and powerful advantage: its pioneering capability for **first-stage rocket reuse**. While other launch providers typically incur costs upwards of $165 million per mission, a standard SpaceX Falcon 9 launch costs approximately **$62 million**. A crucial factor in this cost efficiency is the ability to recover and reuse the Falcon 9's first stage for subsequent launches. This innovative approach not only drastically reduces the operational costs for SpaceX but also positions the company with a distinct competitive edge, making space access more affordable and frequent.

- **Problems you want to find answers**

This report seeks to provide clear answers to several key questions that are critical for understanding the operational efficiency and future potential of reusable launch technology. Specifically, the primary objectives of this analysis are to:

    - **Determine the successful landing of the SpaceX Falcon 9's first stage:** The core problem is to develop a robust predictive model that can accurately forecast whether the first stage of a Falcon 9 rocket will achieve a successful vertical landing after mission completion.
    - **Identify the impact of various parameters on landing outcomes:** We aim to analyze and quantify how different variables influence the success or failure of a first-stage landing. This includes, but is not limited to, factors such as the **launch site (e.g., Cape Canaveral, Vandenberg)**, the **payload mass** carried, the specific **booster version** used, and environmental conditions at the time of launch.
    - **Explore correlations between launch sites and success rates:** A specific area of investigation will be to uncover any significant correlations or patterns between the chosen launch site and the historical success rates of first-stage landings, providing insights into geographical or operational advantages/disadvantages.

Section 1

Methodology

Data Collection

- Data collection methodology:
    - •SpaceX API
    - •Web scrap Falcon 9 and Falcon Heavy launch records from Wikipedia (link)

•Perform datawrangling

- •Determined labels for training the supervised models by converting mission outcomes in to training labels (0-unsuccessful, 1-successful)

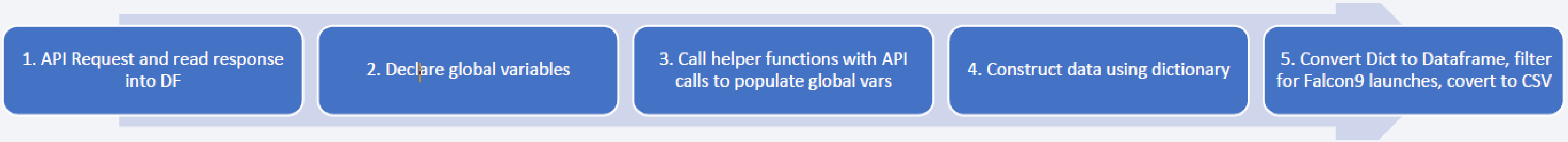
•Perform exploratory data analysis (EDA) using visualization andSQL

•Perform interactive visual analytics using Folium andPlotlyDash

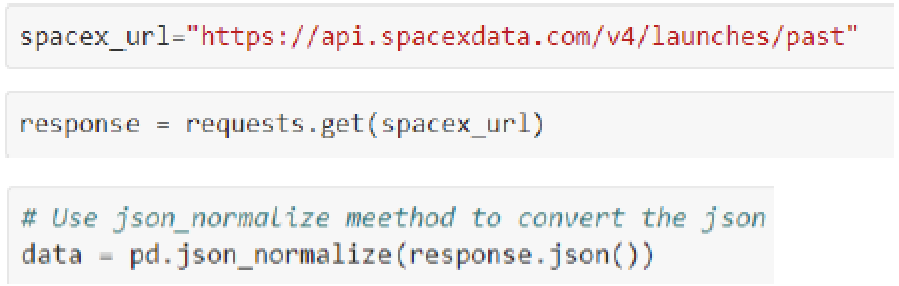
•Perform predictive analysis using classification models

- •Created a column for ‘class’; standardized and transformed data; train/test split data; find best classification algorithm (Logistic regression, SVM, decision tree, & KNN) using test data

Data Collection – SpaceX API


1. API Request and read response into DF
2. Declare global variables
3. Call helper functions with API calls to populate global vars
4. Construct data using dictionary
5. Convert Dict to Dataframe, filter for Falcon9 launches, covert to CSV

1. Create API GET request, normalize data and read in to a Dataframe:

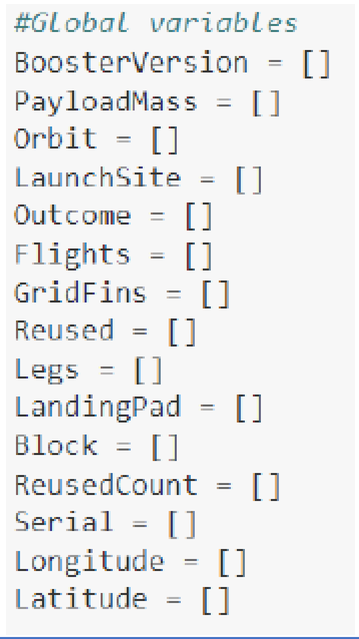


```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

```
# Use json_normalize meethod to convert the json
data = pd.json_normalize(response.json())
```

2. Declare global variable lists that will store data returned by helper functions with additional API calls to get relevant data



```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

Data Collection – SpaceX API

3. Call helper functions to get relevant data where columns have IDs (e.g., rocket column is an identification number)
   - getBoosterVersion(data)
   - getLaunchSite(data)
   - getPayloadData(data)
   - getCoreData(data)

4. Construct dataset from received data & combine columns into a dictionary:

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

4. Create Dataframe from dictionary and filter to keep only the Falcon9 launches:

```
# Create a data from launch_dict
df_launch = pd.DataFrame(launch_dict)
```

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = df_launch[df_launch['BoosterVersion']!= 'Falcon 1']
```

```
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```

Github URL

Data Collection - Scraping

1. Perform HTTP GET to request HTML page
2. Create Beautiful Soap object
3. Extract column names from HTML table header
4. Create Dictionary with keys from extracted column names
5. Call helper functions to fill up dict with launch records
6. Convert Dictionary to Dataframe

1. Create API GET method to request Falcon9 launch HTML page

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

```
html_data = requests.get(static_url).text
```

2. Create Beautiful Soap object

```
soup = BeautifulSoup(html_data,"html.parser")
```

3. Find all the tables on the Wiki page and extract relevant column names from the HTML table header

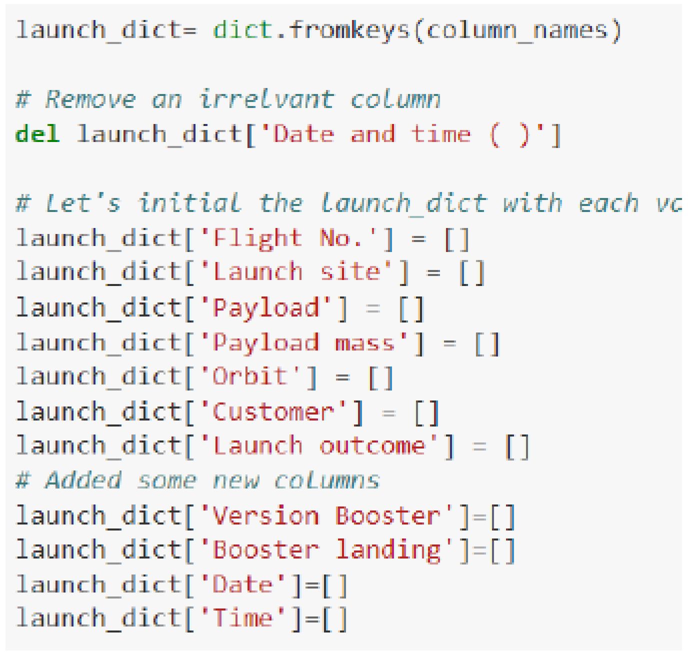
```
html_tables = soup.find_all ('table')
```

```
column_names = []

# Apply find_all() function with `th` element on firs
# Iterate each th element and apply the provided extr
# Append the Non-empty column name (`if name is not N
colnames = soup.find_all('th')
for x in range (len(colnames)):
    name2 = extract_column_from_header(colnames[x])
    if (name2 is not None and len(name2) > 3):
        column_names.append(name2)
```

Data Collection - Scraping

4. Create an empty Dictionary with keys from extracted column names:

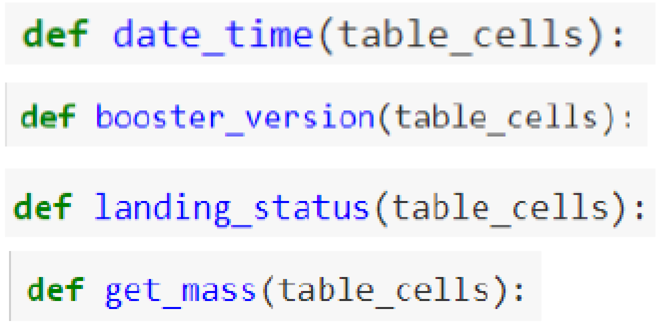


```
launch_dict= dict.fromkeys(column_names)

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

# Let's initial the launch_dict with each va
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']=[]
```

5. Fill up the launch_dict with launch records extracted from table rows.
   - Utilize following helper functions to help parse HTML data



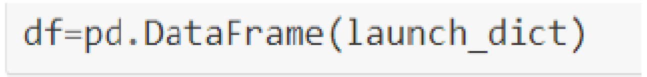
```
def date_time(table_cells):
```

```
def booster_version(table_cells):
```

```
def landing_status(table_cells):
```

```
def get_mass(table_cells):
```

6. Convert launch_dict to Dataframe:



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

[Github URL]

Data Wrangling

To prepare data for supervised model training, I conducted **Exploratory Data Analysis (EDA)** to uncover inherent patterns and establish appropriate labels. The dataset's diverse mission outcomes were transformed into **binary training labels**, where a value of '1' signifies a successful booster landing and '0' indicates an unsuccessful one.

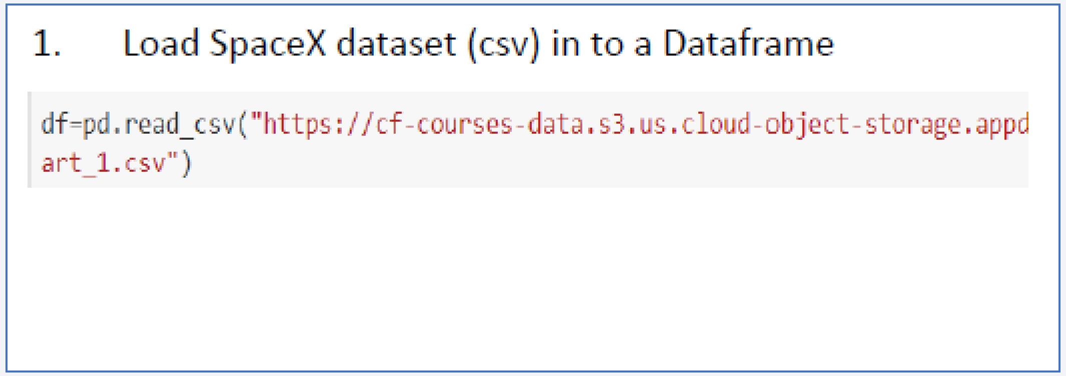
- **The following mission outcomes were specifically mapped to these labels:**
- **Successful Landings (Label: 1):**
  - **True Ocean:** Booster successfully landed in a specific ocean region.
  - **RTLS:** Booster successfully landed on a ground pad.
  - **True ASDS:** Booster successfully landed on a drone ship.
- **Unsuccessful Landings (Label: 0):**
  - **False Ocean:** Booster failed to land in a specific ocean region.
  - **False RTLS:** Booster failed to land on a ground pad.
  - **False ASDS:** Booster failed to land on a drone ship.

Data Wrangling

1. Load dataset in to Dataframe

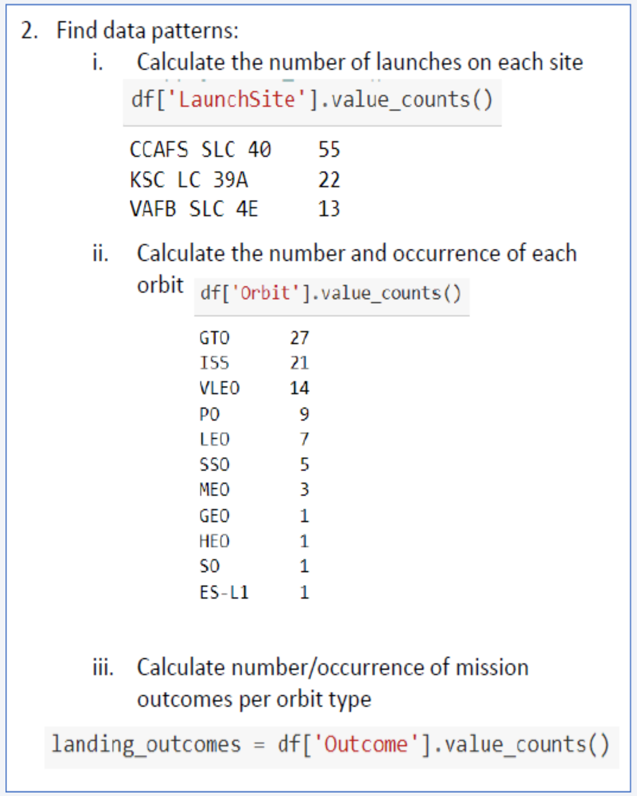
2. Find patterns in data

3. Create landing outcome label



1. Load SpaceX dataset (csv) in to a Dataframe

```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appd
art_1.csv")
```



2. Find data patterns:
   i. Calculate the number of launches on each site

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

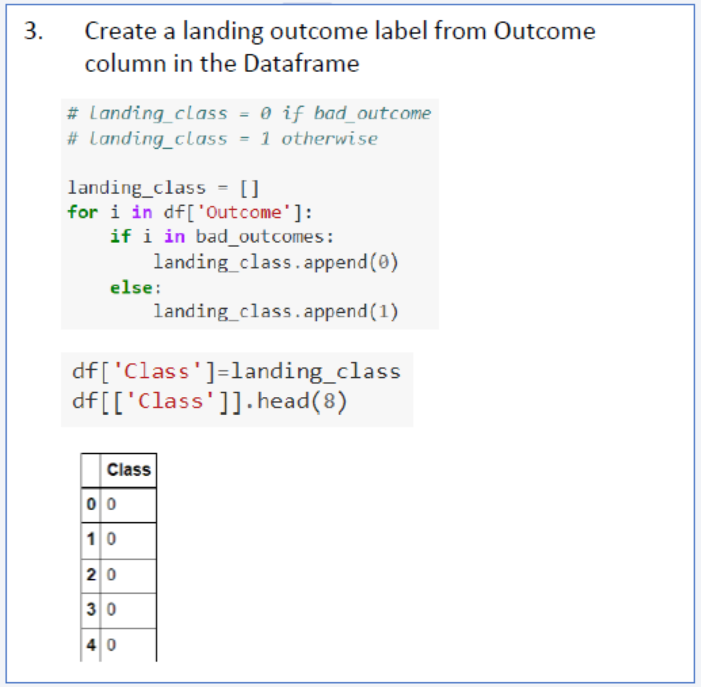
```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
```

   ii. Calculate the number and occurrence of each orbit `df['Orbit'].value_counts()`

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
GEO       1
HEO       1
SO        1
ES-L1     1
```

   iii. Calculate number/occurrence of mission outcomes per orbit type

```
landing_outcomes = df['Outcome'].value_counts()
```



3. Create a landing outcome label from Outcome column in the Dataframe

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise

landing_class = []
for i in df['Outcome']:
    if i in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

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

	Class
0	0
1	0
2	0
3	0
4	0

Github URL

EDA with Data Visualization

Following charts were plotted to gain further insights into the dataset:

1.Scatter plot:

- Shows relationship or correlation between two variables making patterns easy to observe
- Plotted following charts to visualize:
- Relationship between Flight Number and Launch Site
- Relationship between Payload and Launch Site
- Relationship between Flight Number and Orbit Type
- Relationship between Payload and Orbit Type

EDA with Data Visualization

Following charts were plotted to gain further insights into the dataset:

2.Bar Chart:

- Commonly used to compare the values of a variable at a given point in time. Barchartsmakes it easy to see which groups are highest/common and how other groups compare against each other. Length of each bar is proportional to the value of the items that it represents
- Plotted following Bar chart to visualize:
- Relationship between success rate of each orbit type

3.Line Chart:

- Commonly used to track changes over a period of time. It helps depict trends over time.
- Plotted following Line chart to observe:
- Average launch success yearly trend

Github URL

EDA with SQL

Following SQL queries/operations were performed:

Github URL

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA’
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first successful landing outcome in ground pad was achieved.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- 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

Build an Interactive Map with Folium

Folium interactive map helps analyze geospatial data to perform more interactive visual analytics and better understand factors such location and proximity of launch sites that impact launch success rate.

- Following map object were created and added to the map:
    - Mark all launch sites on the map. This allowed to visually see the launch sites on the map.
        - Added ‘folium.circle’ and ‘folium.marker’ to highlight circle area with a text label over each launch site.
    - Added a ‘MarkerCluster()’ to show launch success (green) and failure (red) markers for each launch site.
    - Calculated distances between a launch site to its proximities (e.g., coastline, railroad, highway, city)
        - Added ‘MousePosition() to get coordinate for a mouse position over a point on the map
        - Added ‘folium.Marker()’ to display distance (in KM) on the point on the map (e.g., coastline, railroad, highway, city)
        - Added ‘folium.Polyline()’ to draw a line between the point on the map and the launch site
        - Repeated steps above to add markers and draw lines between launch sites and proximities –coastline, railroad, highway, city)
- Building the Interactive Map with Folium helped answered following questions:
    - Are launch sites in close proximity to railways? YES
    - Are launch sites in close proximity to highways? YES
    - Are launch sites in close proximity to coastline? YES
    - Do launch sites keep certain distance away from cities? YES

Github URL

Build a Dashboard with Plotly Dash

Plotly Dash web application performs interactive visual analytics on SpaceX launch data in real-time. Added Launch Site Drop-down, Pie Chart, Payload range slide, and a Scatter chart to the Dashboard.

- Launch Site Drop-down Input component to the dashboard provides an ability to filter Dashboard visual by all launch sites or a particular launch site
- Pie Chart to the Dashboard shows total success launches when ‘All Sites’ is selected and show success and failed counts when a particular site is selected
- Payload range slider to the Dashboard permits to easily select different payload ranges to identify visual patterns
- Scatter chart permits to observe how payload may be correlated with mission outcomes for selected site(s). The color-label Booster version on each scatter point provided missions outcomes with different boosters

Build a Dashboard with Plotly Dash

Dashboard helps to answer following questions:

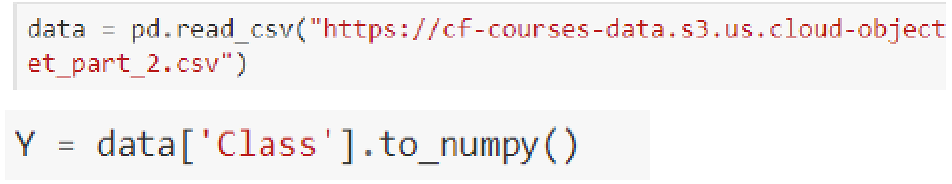
- Which site has the largest successful launches? KSC LC-39A with 10
- Which site has the highest launch success rate? KSC LC-39A with 76.9% success
- Which payload range(s) has the highest launch success rate? 2000 – 5000 kg
- Which payload range(s) has the lowest launch success rate? 0-2000 and 5500 -7000
- Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest launch success rate? FT

Github URL

Predictive Analysis (Classification)

1. Read dataset into Dataframe and create a ‘Class’ array
2. Standardize the data
3. Train/Test/Split data in to training and test data sets
4. Create and Refine Models
5. Find the best performing Model

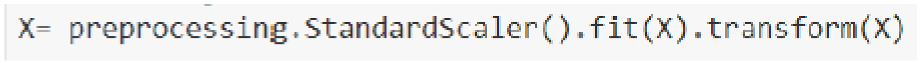
1. Load SpaceX dataset (csv) in to a Dataframe and create NumPy array from the column class in data



```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object
et_part_2.csv")
```

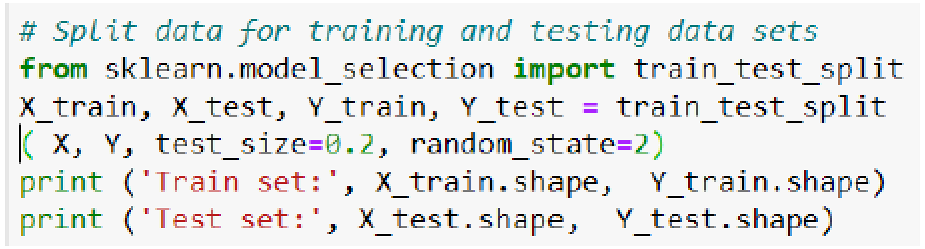
```
Y = data['Class'].to_numpy()
```

2. Standardize data in X then reassign to variable X using transform



```
X= preprocessing.StandardScaler().fit(X).transform(X)
```

3. Train/test/split X and Y in to training and test data sets.

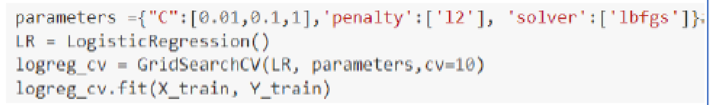


```
# Split data for training and testing data sets
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split
( X, Y, test_size=0.2, random_state=2)
print ('Train set:', X_train.shape,  Y_train.shape)
print ('Test set:', X_test.shape,  Y_test.shape)
```

Predictive Analysis (Classification)

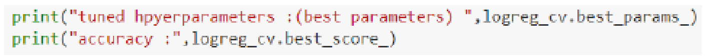
1. Read dataset into Dataframe and create a 'Class' array
2. Standardize the data
3. Train/Test/Split data in to training and test data sets
4. Create and Refine Models
5. Find the best performing Model

4. Create and refine Models based on following classification Algorithms: (below is LR example)
   i. Create Logistic Regression object and then create a GridSearchCV object
   ii. Fit train data set in to the GridSearchCV object and train the Model



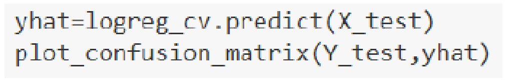
```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}
LR = LogisticRegression()
logreg_cv = GridSearchCV(LR, parameters,cv=10)
logreg_cv.fit(X_train, Y_train)
```

   iii. Find and display best hyperparameters and accuracy score



```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

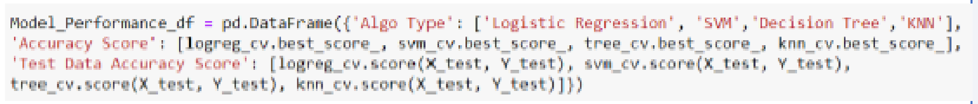
   iv. Check the accuracy on the test data by creating a confusion matrix



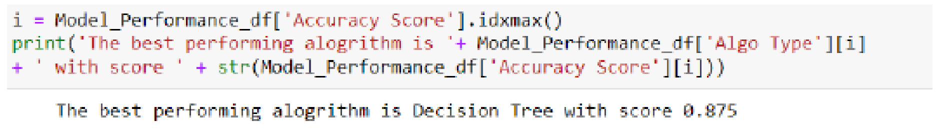
```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

   v. Repeat above steps for Decision Tree, KNN, and SVM algorithms

3. Find the best performing model

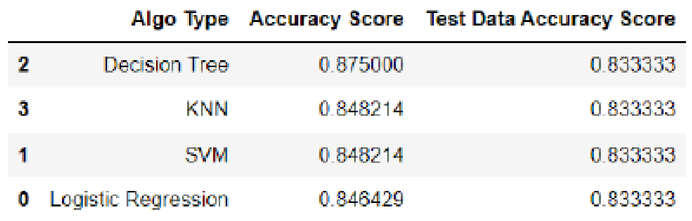


```
Model_Performance_df = pd.DataFrame({'Algo Type': ['Logistic Regression', 'SVM','Decision Tree','KNN'],
'Accuracy Score': [logreg_cv.best_score_, svm_cv.best_score_, tree_cv.best_score_, knn_cv.best_score_],
'Test Data Accuracy Score': [logreg_cv.score(X_test, Y_test), svm_cv.score(X_test, Y_test),
tree_cv.score(X_test, Y_test), knn_cv.score(X_test, Y_test)]})
```



```
i = Model_Performance_df['Accuracy Score'].idxmax()
print('The best performing alogrithm is '+ Model_Performance_df['Algo Type'][i]
+ ' with score ' + str(Model_Performance_df['Accuracy Score'][i]))
```

```
The best performing alogrithm is Decision Tree with score 0.875
```

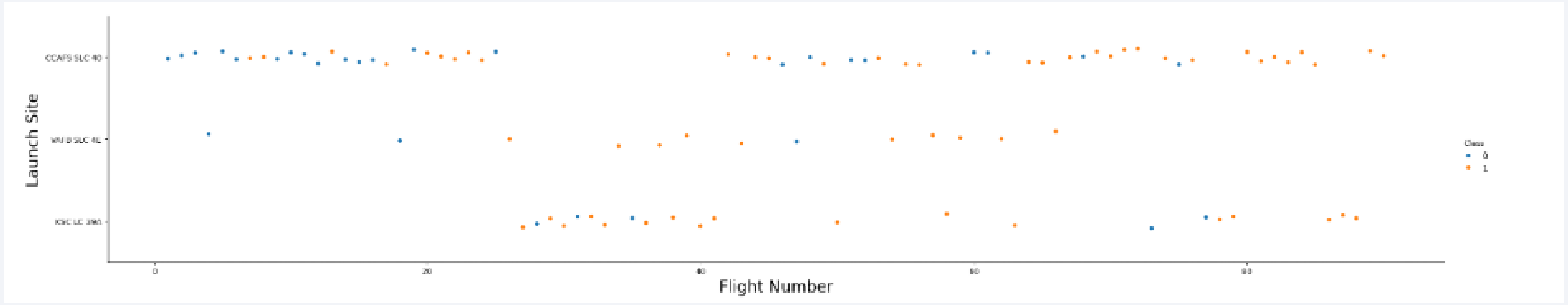


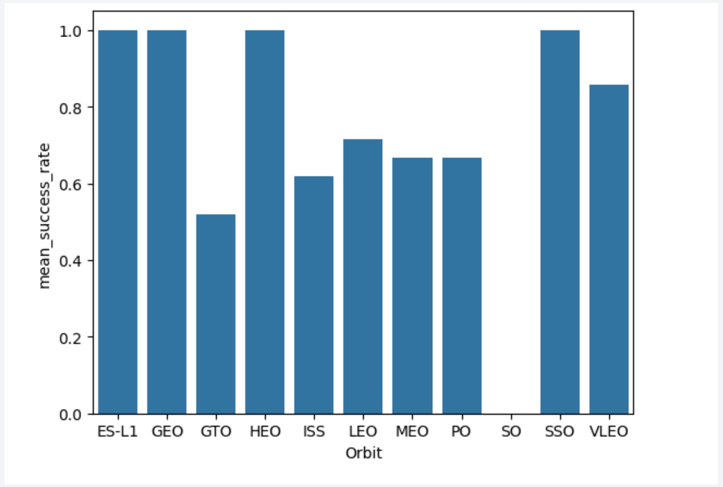
	Algo Type	Accuracy Score	Test Data Accuracy Score
2	Decision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333
1	SVM	0.848214	0.833333
0	Logistic Regression	0.846429	0.833333

Github URL

Results

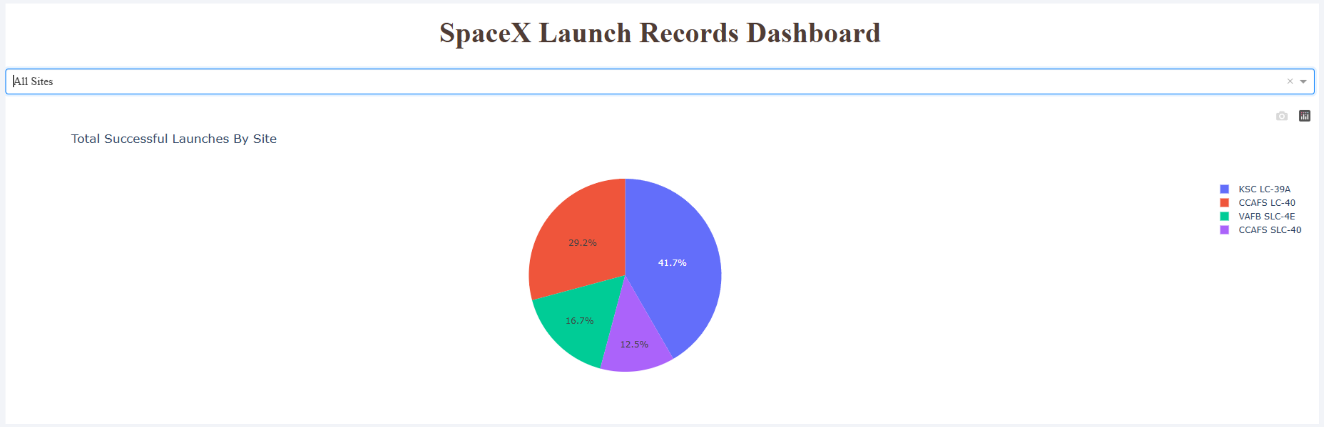
- Exploratory Data Analysis examples

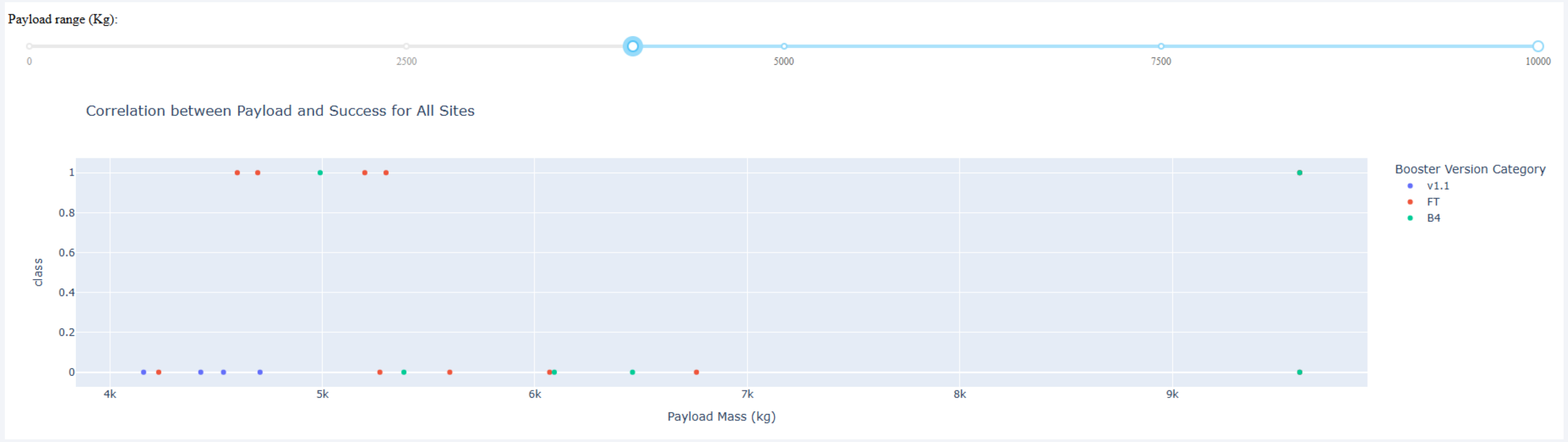




Results

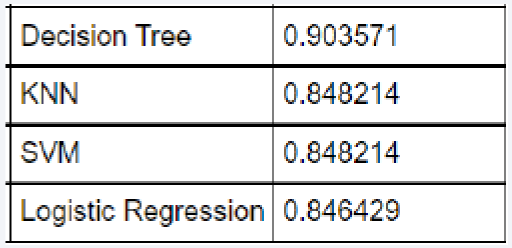
- Interactive analytic demo screenshots:





Results

- Predictive analytic (accuracy):

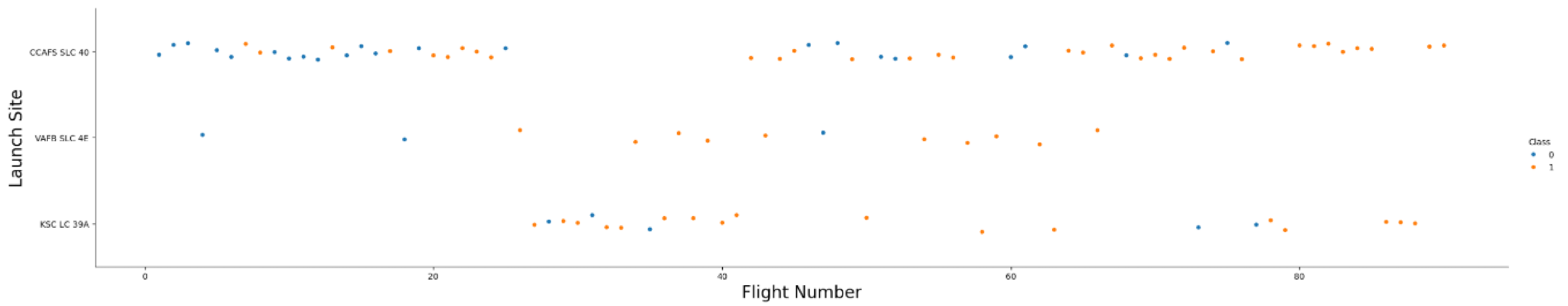


Decision Tree	0.903571
KNN	0.848214
SVM	0.848214
Logistic Regression	0.846429

Section 2

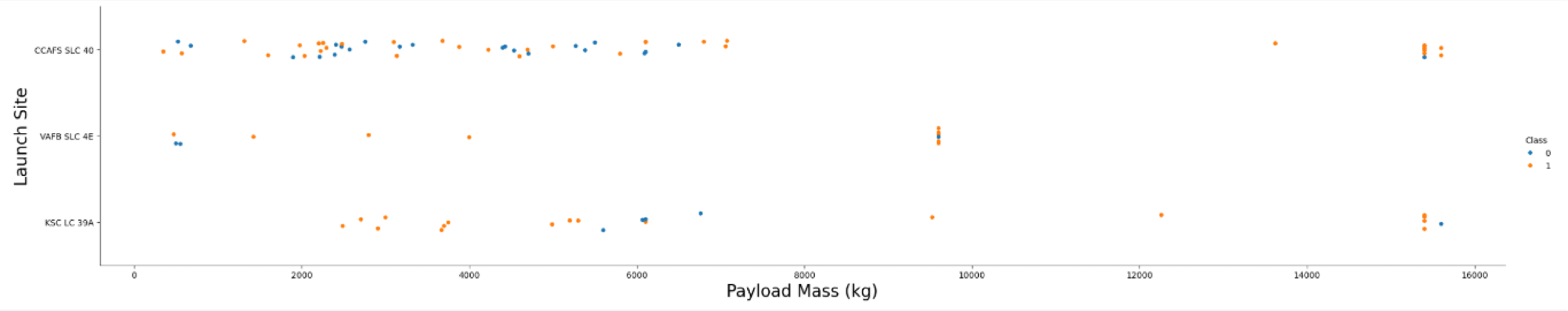
Insights drawn from EDA

Flight Number vs. Launch Site



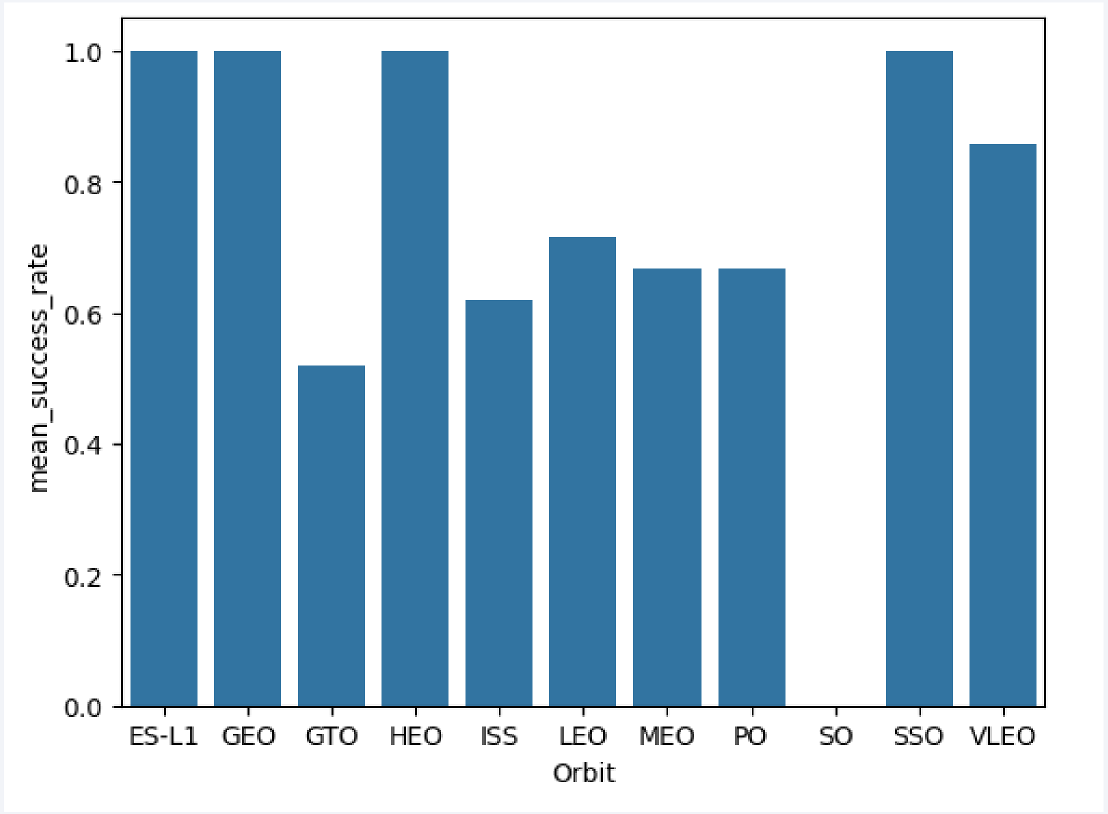
- Success rates increases as the number of flights increase
- For example, for launch site ‘CCAFS SLC-40’, it takes at least around 8 launches before a first successful launch

Payload vs. Launch Site



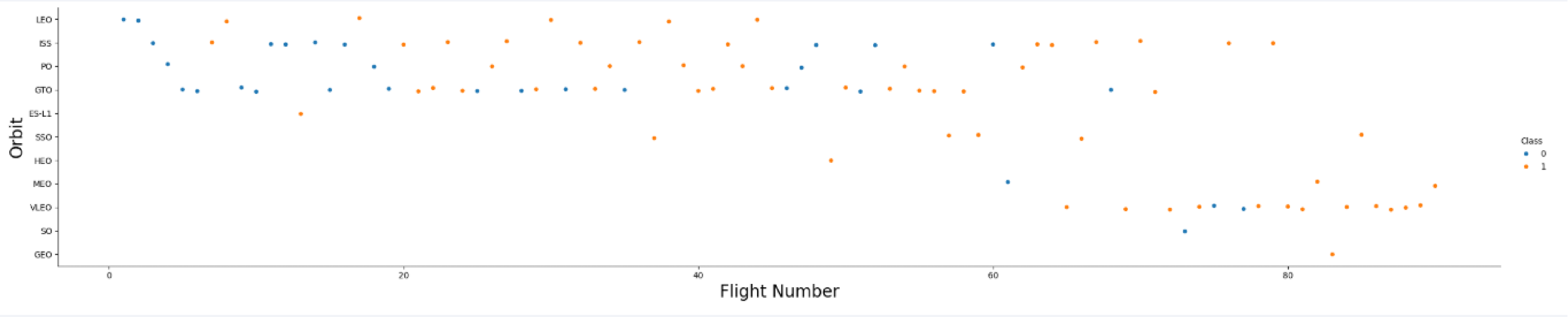
- For example, for launch site ‘VAFB SLC 4E’, there are no rockets launched for payload greater than 10,000 kg

Success Rate vs. Orbit Type



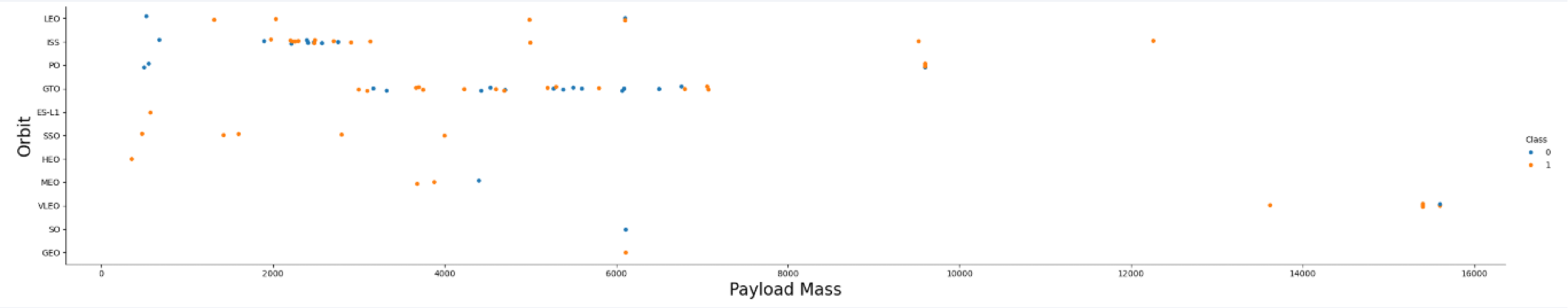
- Orbits ES-LI, GEO, HEO, and SSO have the highest success rates
- GTO orbit has the lowest success rate

Flight Number vs. Orbit Type



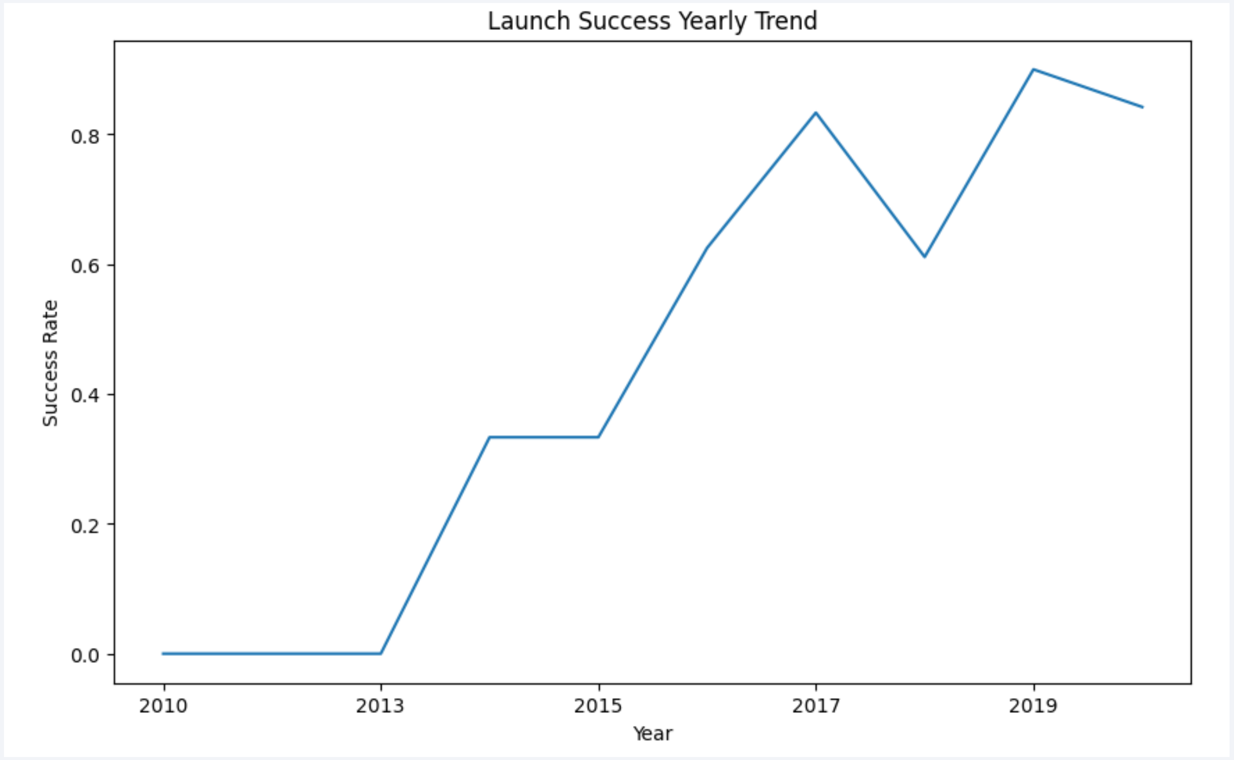
- For most orbits (LEO, ISS, PO, SSO, MEO, VLEO) successful landing rates appear to increase with flight numbers

Payload vs. Orbit Type



- Successful landing rates (Class=1) appear to increase with pay load for orbits LEO, ISS, PO, and SSO
- For GEO orbit, there is not clear pattern between payload and orbit for successful or unsuccessful landing

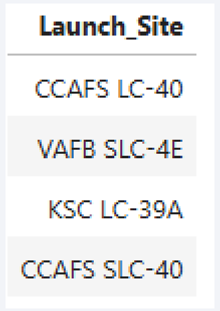
Launch Success Yearly Trend



- Success rate increased by about 80% between 2013 and 2020
- Success rates remained the same between 2010 and 2013 and between 2014 and 2015
- Success rates decreased between 2017 and 2018 and between 2019 and 2020

All Launch Site Names

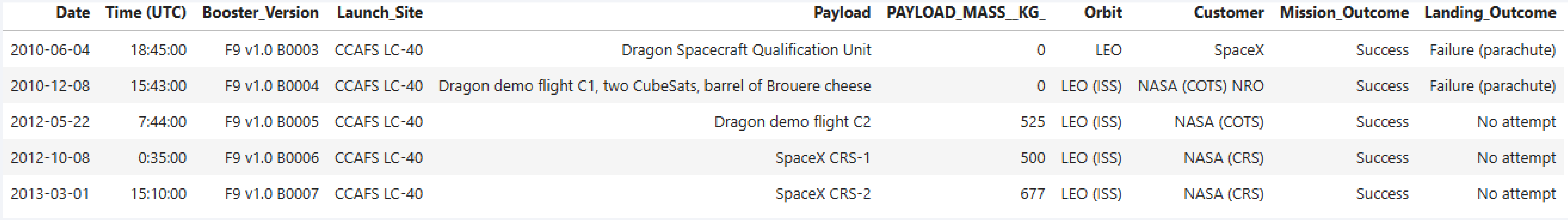
- SQL query:
  - SELECT DISTINCT(Launch_Site) FROM SPACEXTBL
- Result:



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

Launch Site Names Begin with 'CCA'

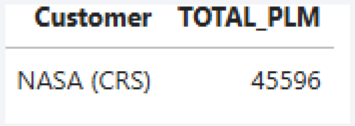
- SQL query:
  - SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE "CCA%" LIMIT 5
- Results:



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

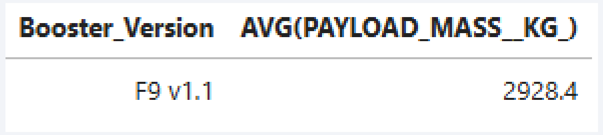
- SQL query:
  - SELECT Customer, SUM(PAYLOAD_MASS__KG_) AS TOTAL_PLM FROM SPACEXTBL WHERE Customer LIKE "NASA (CRS)"
- Result:



Customer	TOTAL_PLM
NASA (CRS)	45596

Average Payload Mass by F9 v1.1

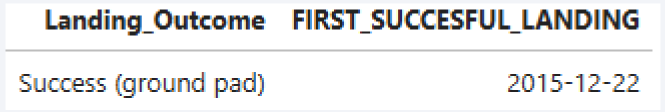
- SQL query:
  - SELECT Booster_Version, AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version LIKE "F9 v1.1"
- Result:



Booster_Version	AVG(PAYLOAD_MASS__KG_)
F9 v1.1	2928.4

First Successful Ground Landing Date

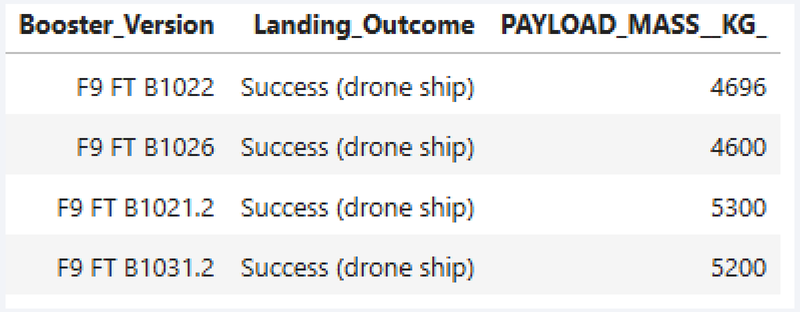
- SQL query:
  - SELECT Landing_Outcome, MIN(DATE) AS FIRST_SUCCESFUL_LANDING FROM SPACEXTBL WHERE Landing_Outcome LIKE "Success (ground pad)"
- Result:



Landing_Outcome	FIRST_SUCCESFUL_LANDING
Success (ground pad)	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

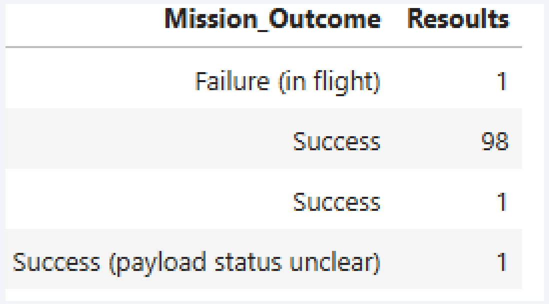
- SQL query:
  - SELECT Booster_Version, Landing_Outcome, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE Landing_Outcome LIKE "Success (drone ship)" AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000)
- Results:



Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

Total Number of Successful and Failure Mission Outcomes

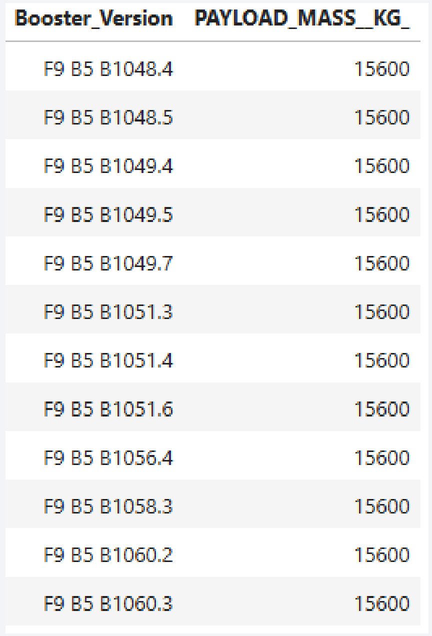
- SQL query:
  - SELECT Mission_Outcome, COUNT(Mission_Outcome) AS Resoults FROM SPACEXTBL GROUP BY Mission_Outcome
- Results:



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

Boosters Carried Maximum Payload

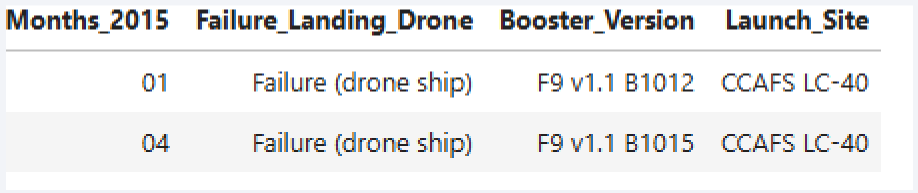
- SQL query:
  - SELECT DISTINCT(Booster_Version), PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL) ORDER BY Booster_Version
- Results:



Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600

2015 Launch Records

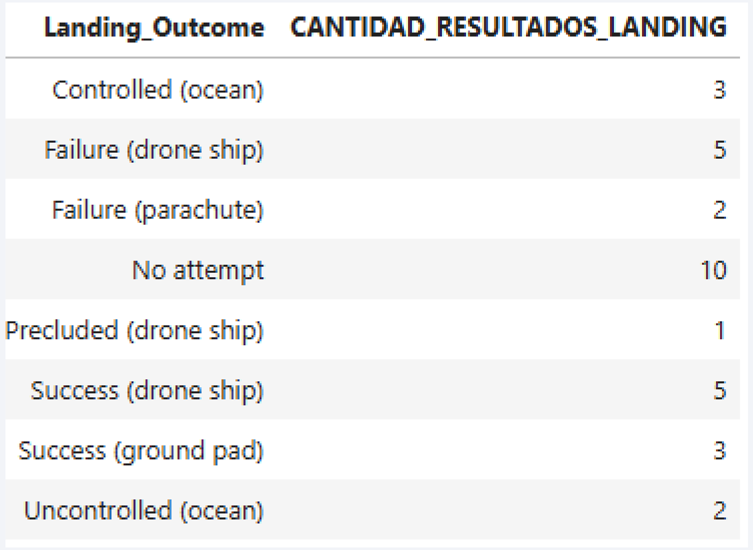
- SQL query:
  - SELECT substr(Date, 6, 2) AS Months_2015, Landing_Outcome AS Failure_Landing_Drone, Booster_Version, Launch_Site FROM SPACEXTBL WHERE Landing_Outcome LIKE "Failure (drone ship)" AND substr(Date, 0, 5) LIKE "2015"
- Results:



Months_2015	Failure_Landing_Drone	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

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

- SQL query:
  - SELECT Landing_Outcome, COUNT(Landing_Outcome) AS CANTIDAD_RESULTADOS_LANDING FROM SPACEXTBL WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome
- Results:

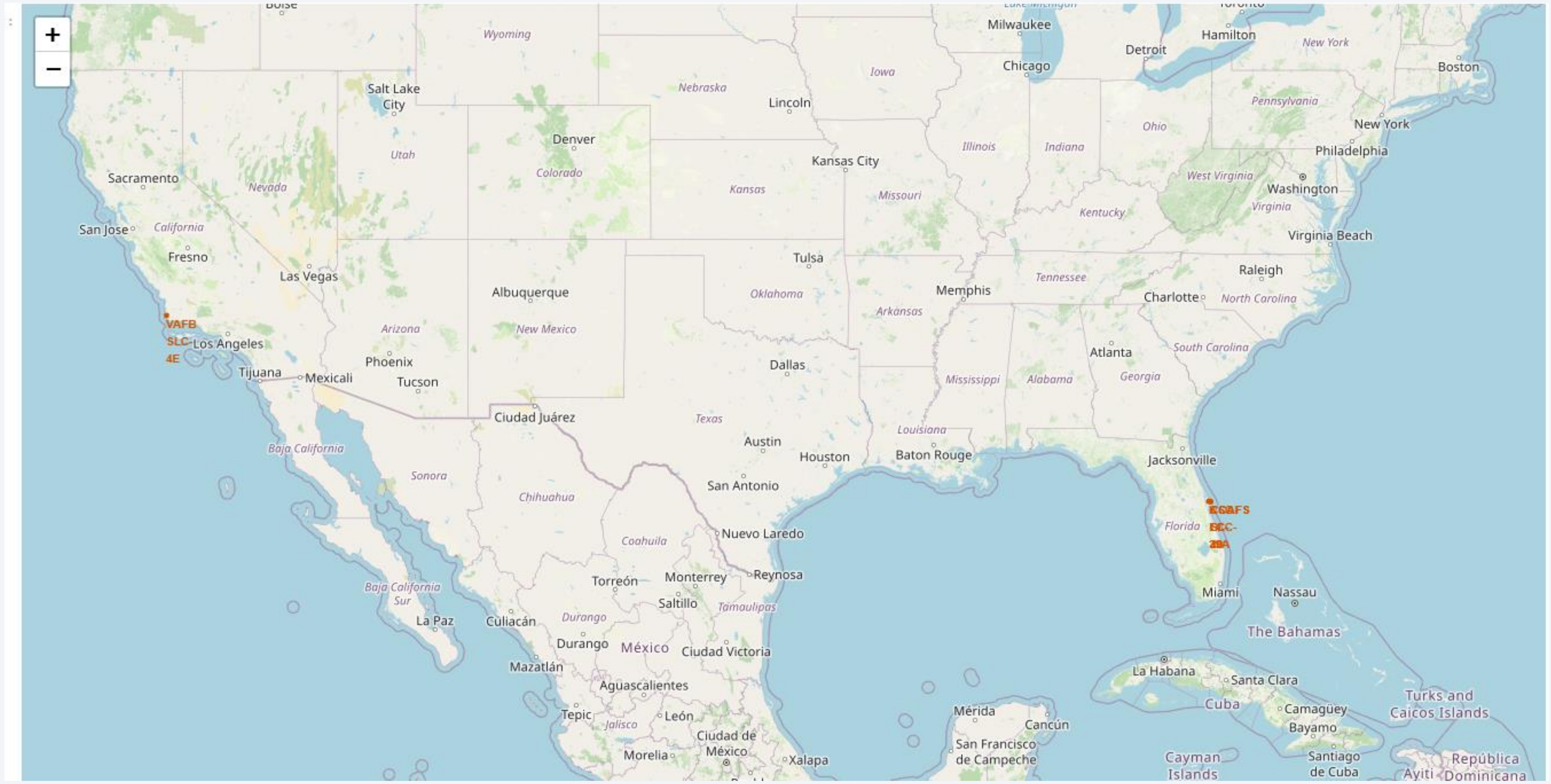


Landing_Outcome	CANTIDAD_RESULTADOS_LANDING
Controlled (ocean)	3
Failure (drone ship)	5
Failure (parachute)	2
No attempt	10
Precluded (drone ship)	1
Success (drone ship)	5
Success (ground pad)	3
Uncontrolled (ocean)	2

Section 3

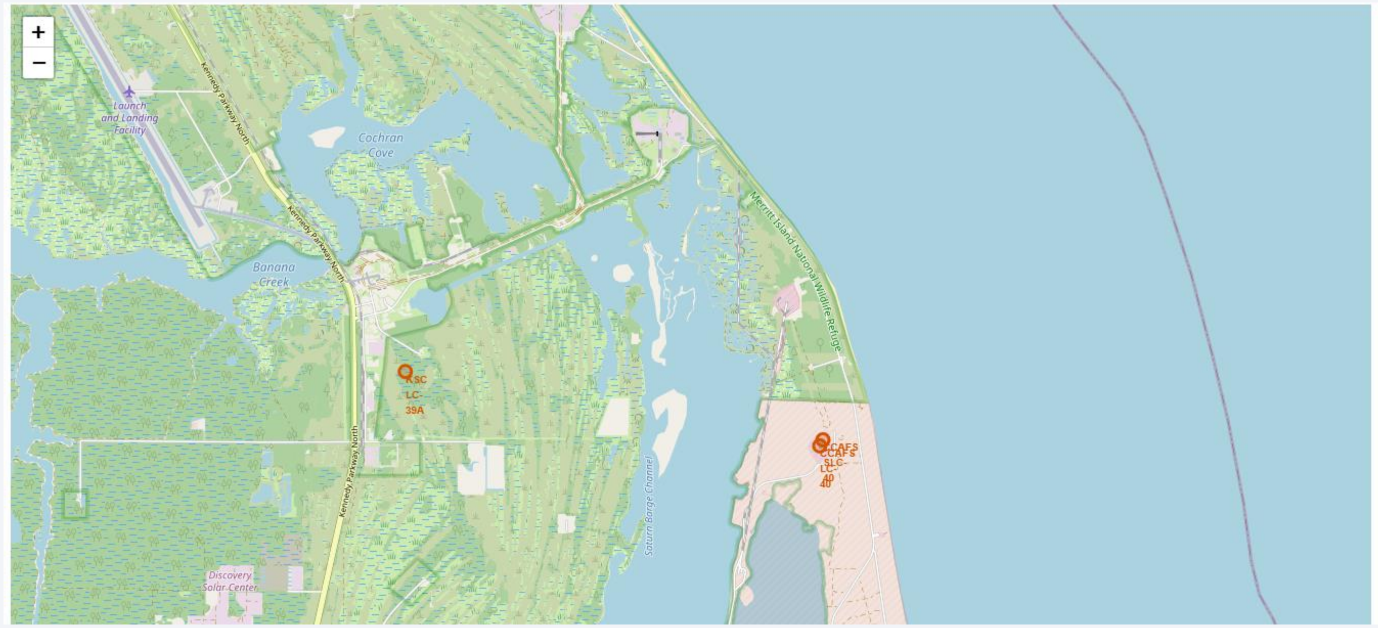
Launch Sites Proximities Analysis

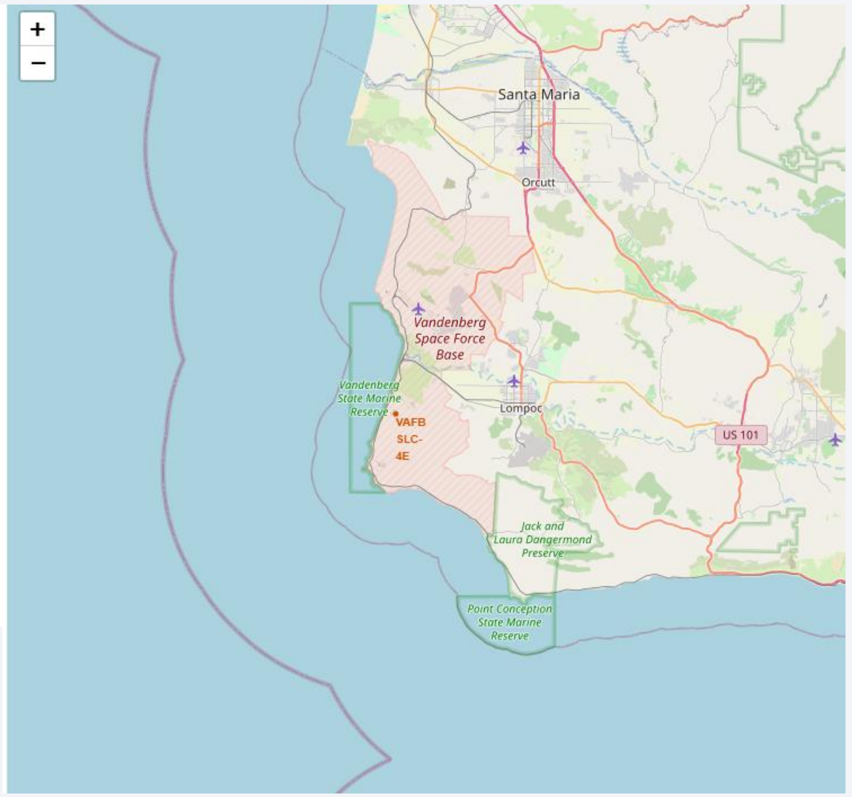
Laucn sites map



- Global map with Falcon 9 launch sites that are located in the United States (in California and Florida). Each launch site contains a circle, label, and a popup to highlight the location and the name of the launch site.

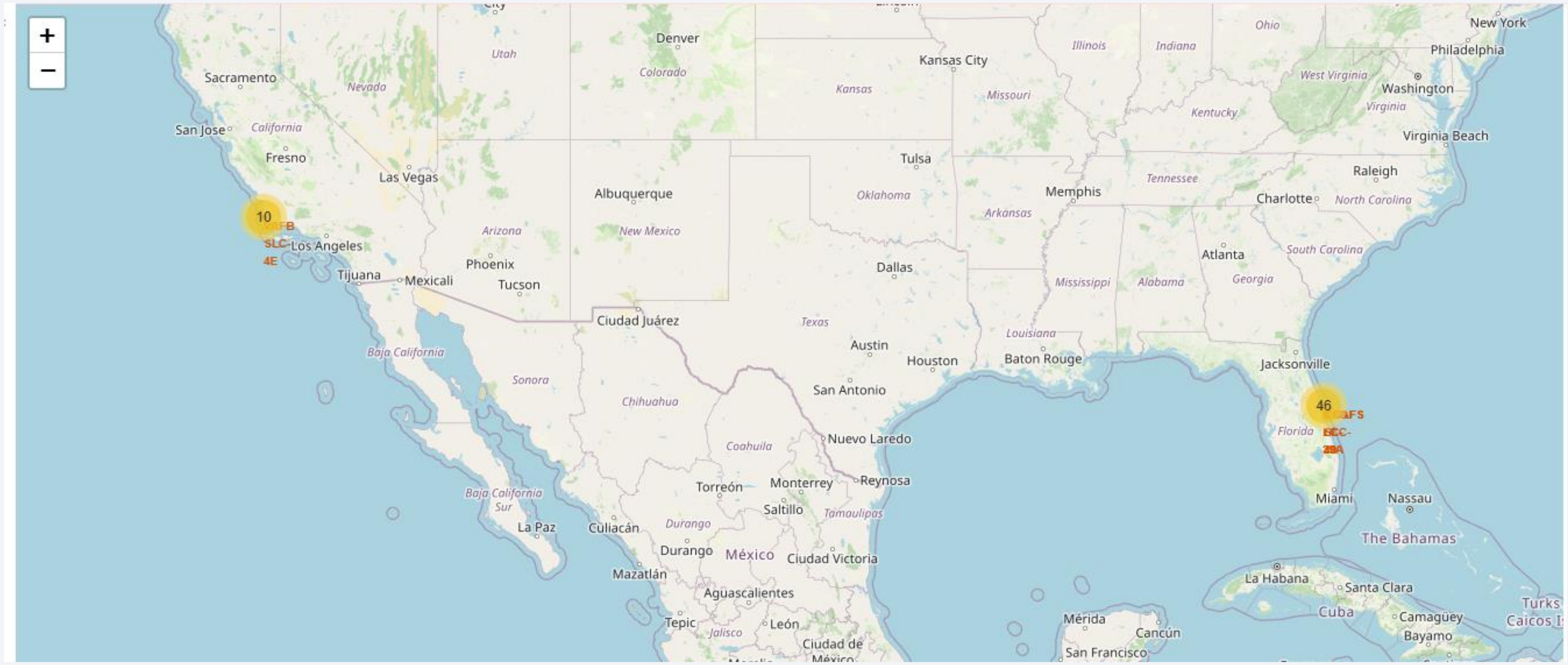
Laucn sites map




Santa Maria
Orcutt
Vandenberg Space Force Base
Vandenberg State Marine Reserve
VAFB SLC-4E
Lompoc
US 101
Jack and Laura Dangermond Preserve
Point Conception State Marine Reserve

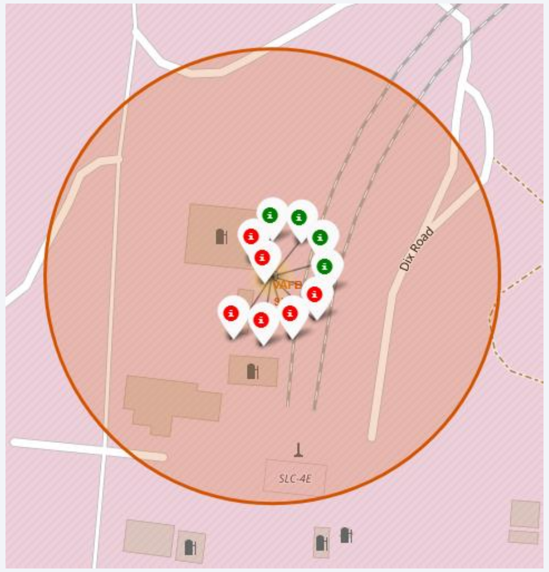
- Zoom in to the launch sites to display 4 launch sites

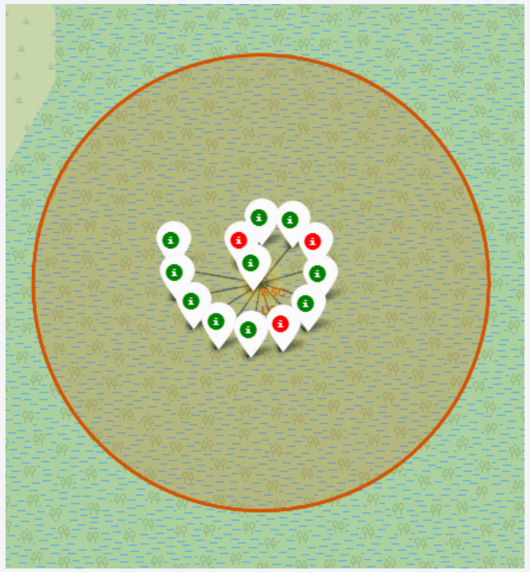
Success/Failed Launch Map for all Launch Sites

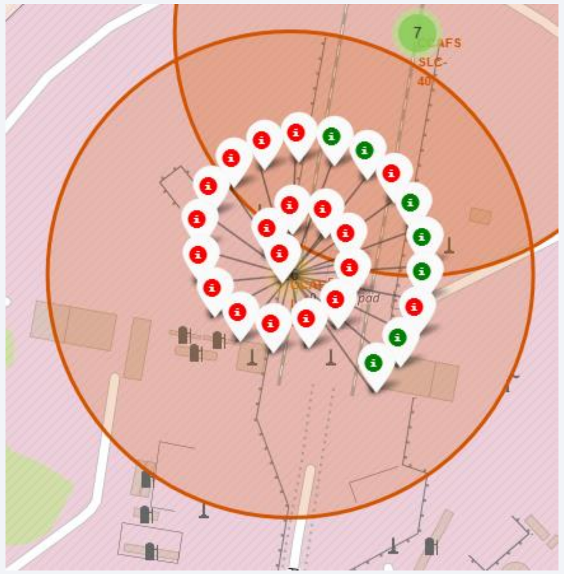


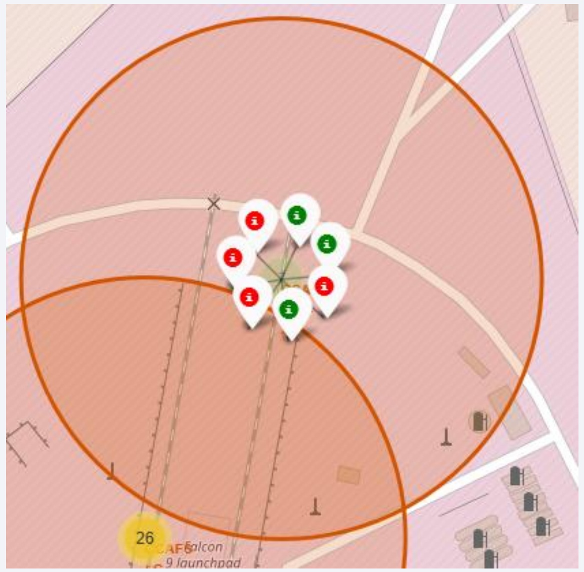
- Map with all success and failed launches

Success/Failed Launch Map for all Launch Sites



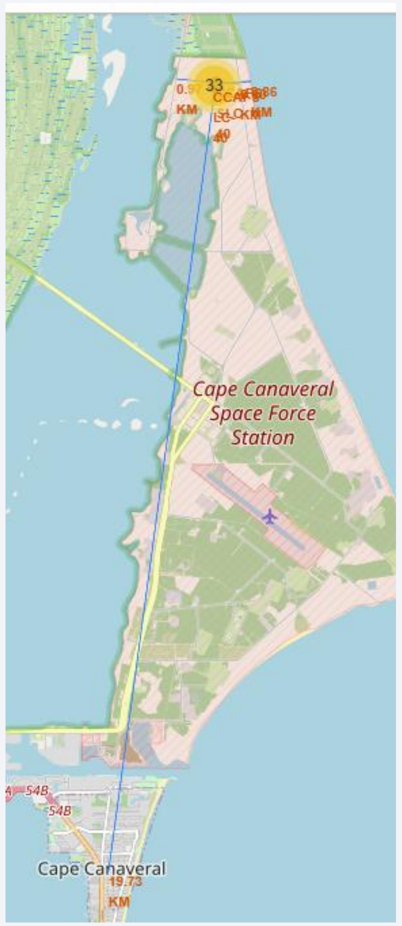


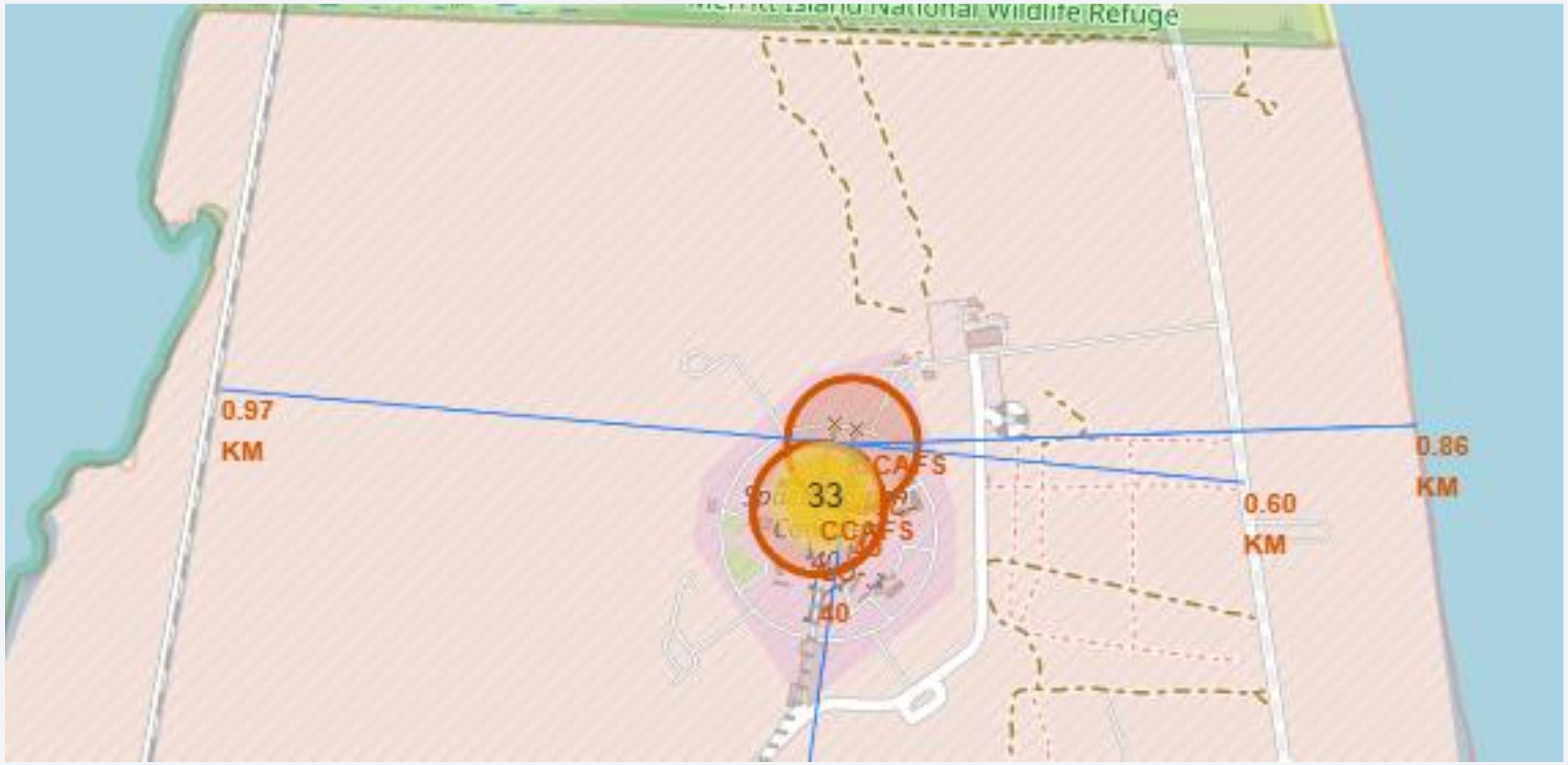




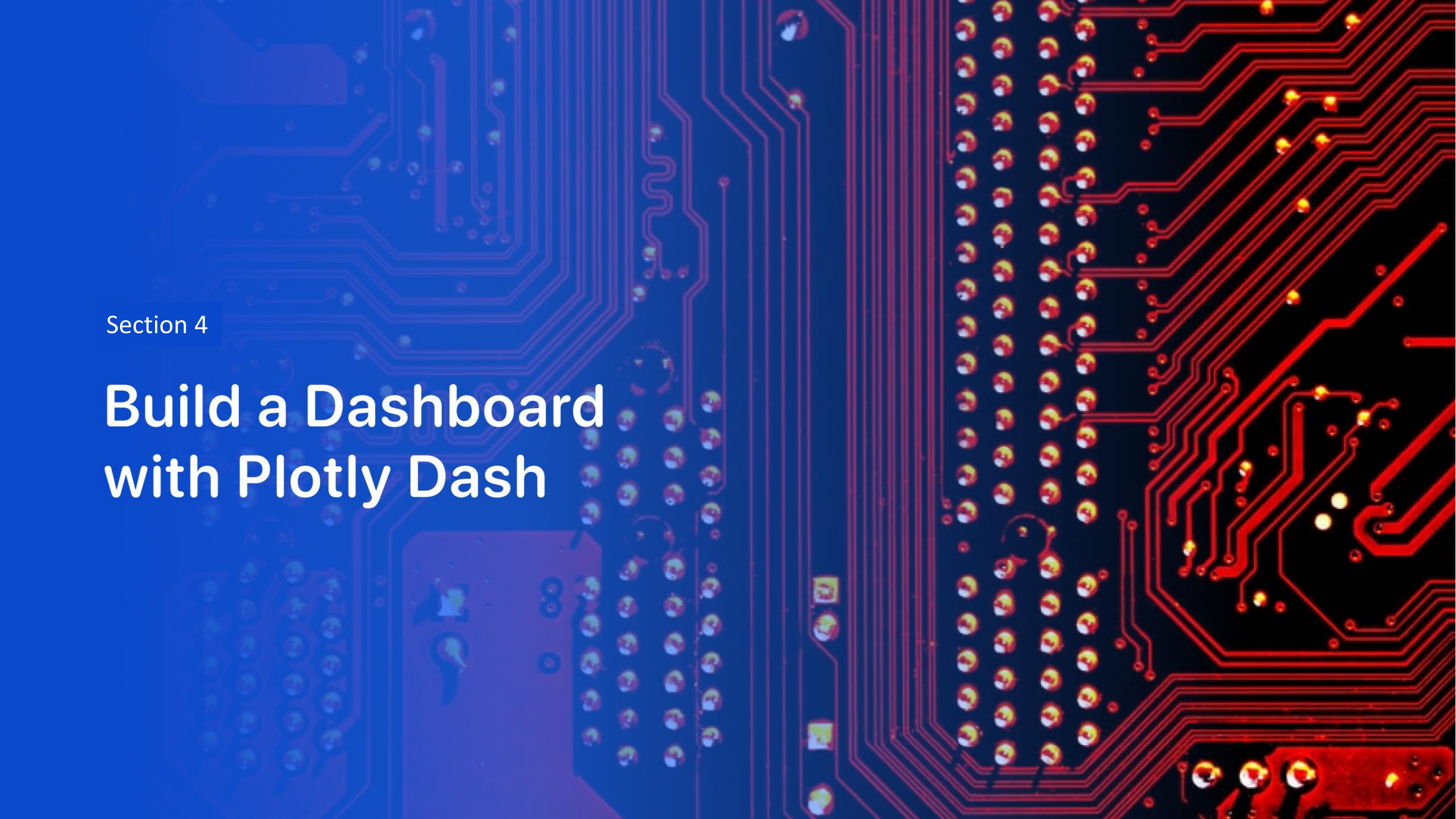
- Zoom in to each site and displays the success/fail markers with green as success and red as failed

Launch Site Proximity Distance Map





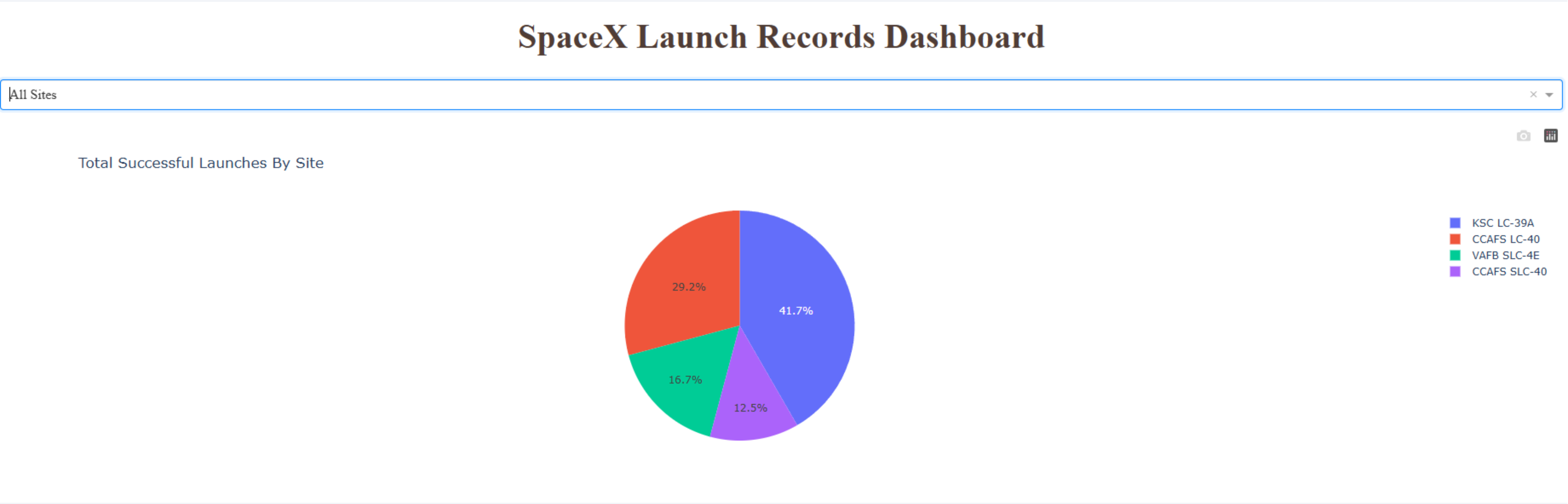
- Zoom on CCAFS SLC-40 Launch Site. In general there are close to coast, highway and railroad but far from cities



Section 4

Build a Dashboard with Plotly Dash

Launch success counts for all sites



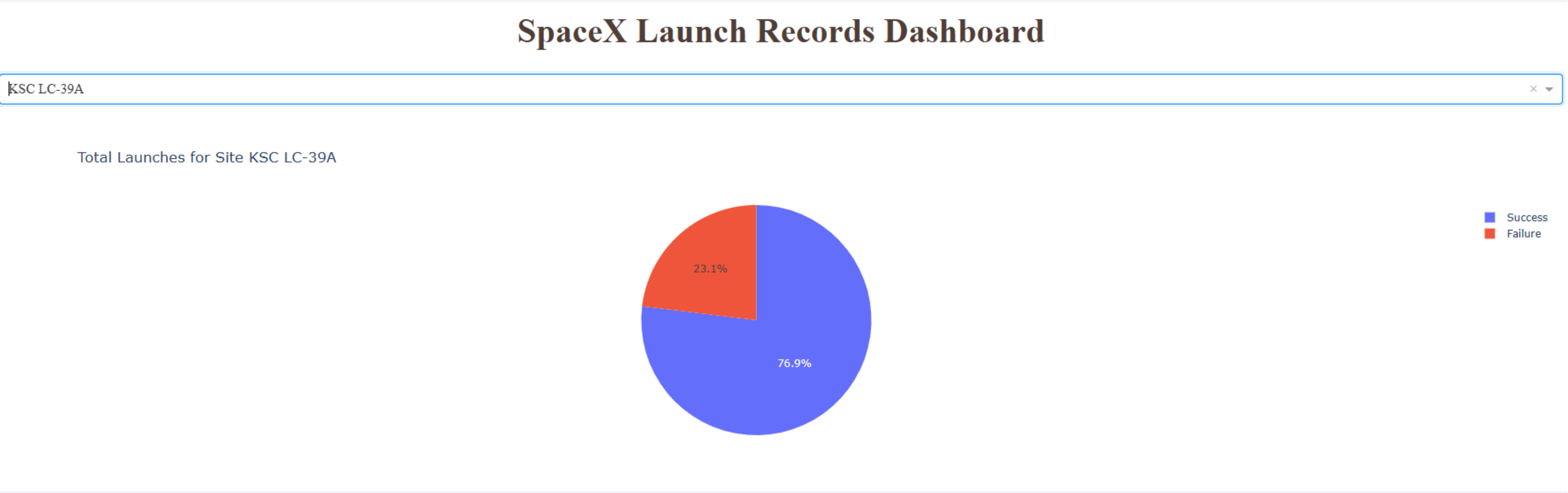
SpaceX Launch Records Dashboard

All Sites

Total Successful Launches By Site

- KSC LC-39A has the highest launch success rate
- CCAFS SLC-40 has the lowest launch success rate

Launch site with the highest launch success ratio



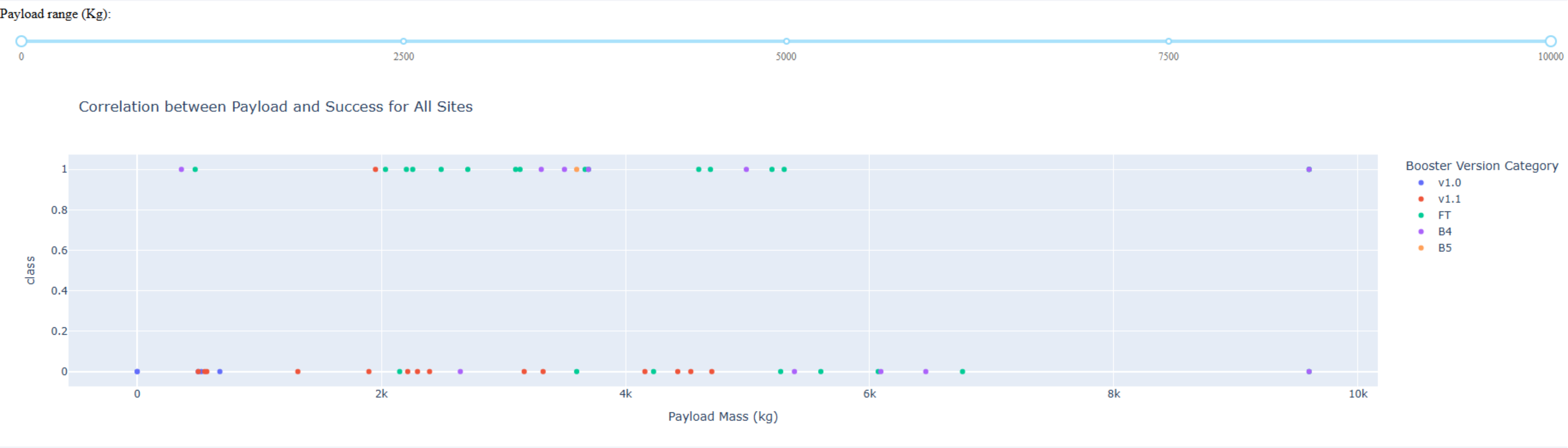
SpaceX Launch Records Dashboard

KSC LC-39A

Total Launches for Site KSC LC-39A

- KSC LC-39A is the launch site with the highest launch success ratio. Success 76.9% / Failure 23,1%

Payload vs. Launch Outcome Scatter Plot for All Sites

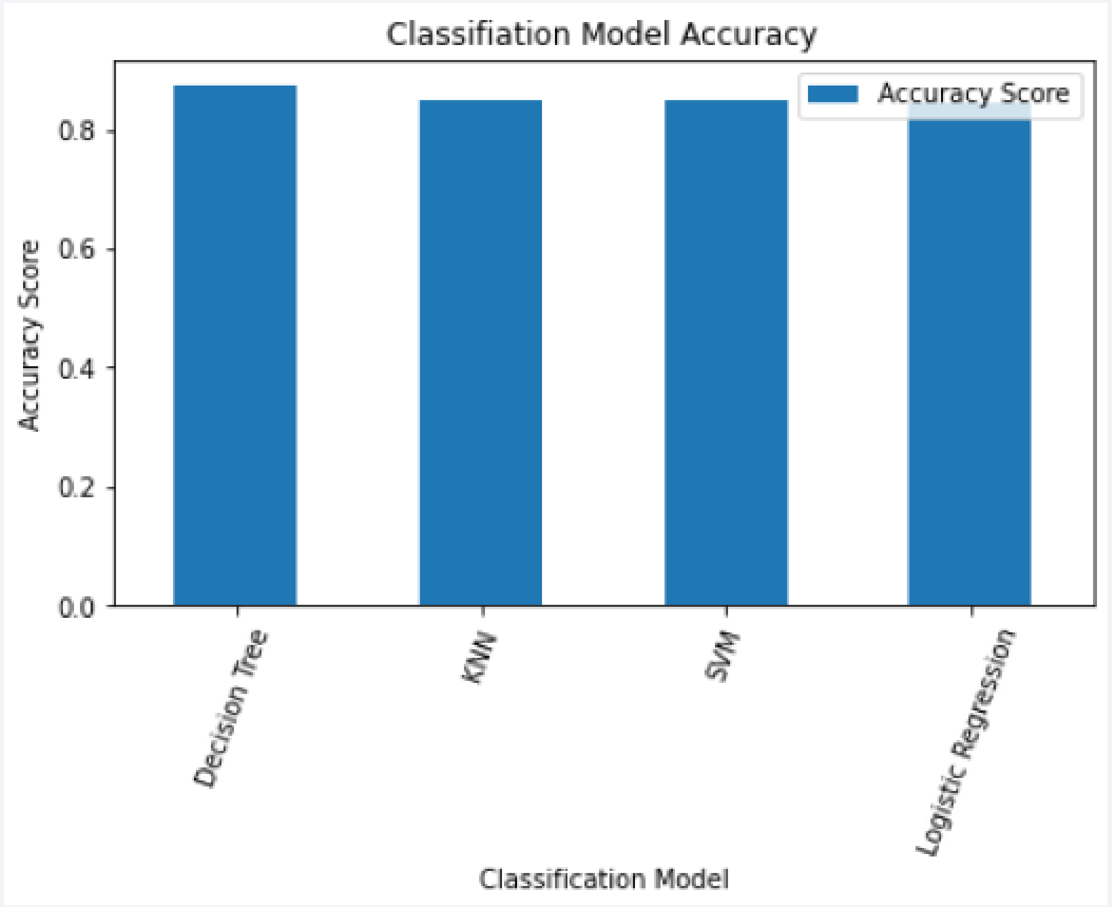

Payload range (Kg):

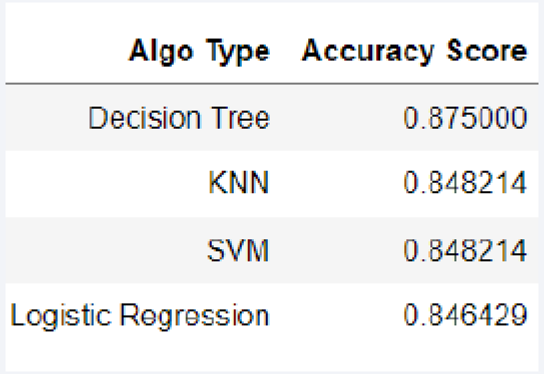
- Most successful launches are in the payload range from 2000 to about 5500

Section 5

Predictive Analysis (Classification)

Classification Accuracy



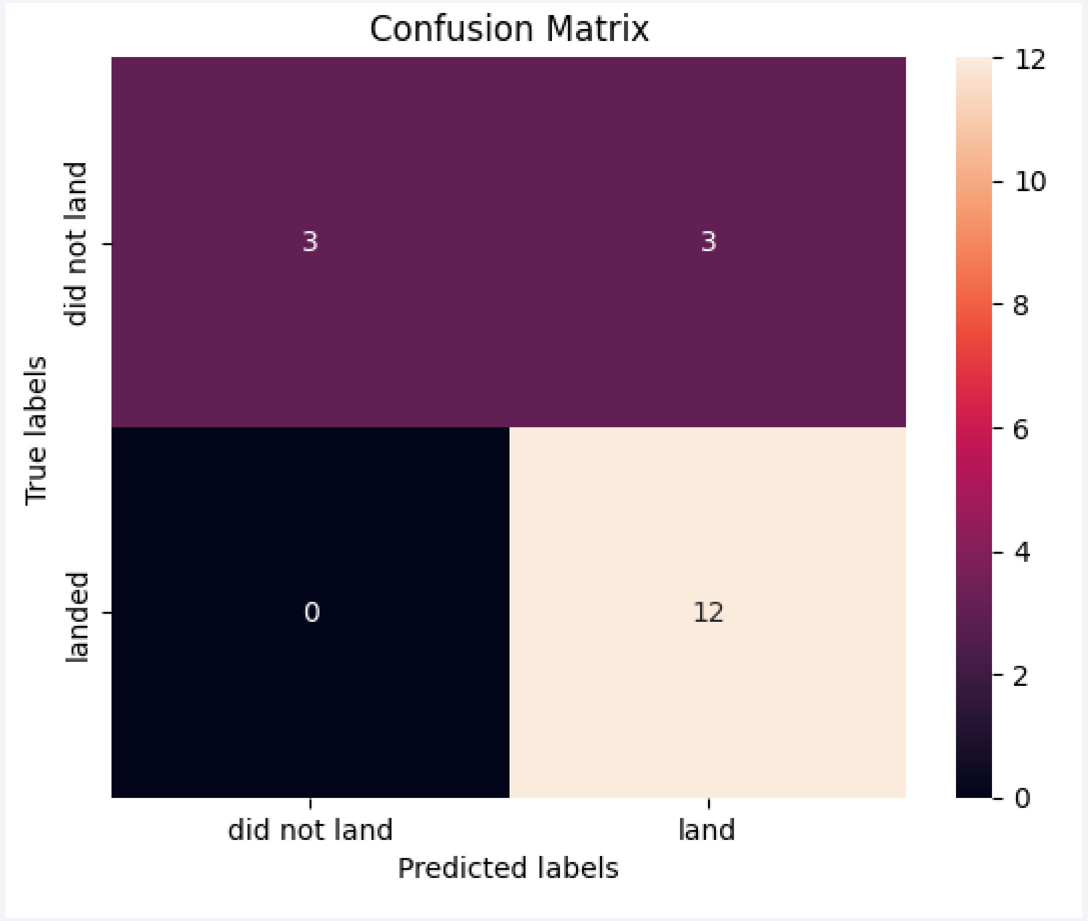


Algo Type	Accuracy Score
Decision Tree	0.875000
KNN	0.848214
SVM	0.848214
Logistic Regression	0.846429

- Decision Tree algorithm has the highest classification score with a value of .8750

Confusion Matrix

- The confusion matrix is same for all the models
- 18 predictions were made:
  - 12 true positive
  - 3 true negative
  - 3 false positive
  - 0 false negative



Conclusions

Launch Success Analysis

- **Increased flight frequency correlates with initial stage success.** As the number of flights rises, the first stage demonstrates a higher likelihood of successful landing.
- **Payload mass and success rates show an unclear relationship.** While success rates seem to increase with a heavier payload, a clear correlation between payload mass and success rates has not been established.
- **Significant increase in launch success rate from 2013 to 2020.** The launch success rate improved by approximately 80% within this seven-year period.
- **Launch site performance varies significantly.** "KSC LC 39A" boasts the highest launch success rate, whereas "CCAFS SLC 40" records the lowest.
- **Specific orbits yield higher success rates.** The orbits ES-L1, GEO, HEO, and SSO exhibit the highest launch success rates, with GTO showing the lowest.
- **Strategic placement of launch sites.** Launch sites are strategically located away from urban areas and are typically closer to coastlines, railroads, and highways.

Conclusions

Machine Learning Model Performance

- The **Decision Tree** model is the top performer among the machine learning classification models, achieving an accuracy of about 87.5%. However, when all models were evaluated on test data, their accuracy scores were consistently around 83%. Additional data may be necessary to further refine these models and potentially achieve a better fit.

Appendix

- All relevant auxiliary information are as links in slides

Thank you!