IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Cristina Morillo Leal
October 6th, 2024

Outline

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

Executive Summary

Objective of the Study

The aim of this study is to analyze SpaceX Falcon 9 data collected from multiple sources and leverage machine learning models to predict the success of its first stage landing. This could assist other space agencies in deciding whether to compete with SpaceX.

Methodology Overview

The research employed various techniques for data collection, analysis, model development, and prediction, including the following:

- Data collection through APIs and web scraping.
- Data transformation using data wrangling techniques.
- Exploratory data analysis using SQL and data visualization.
- Creation of an interactive map using Folium to study the proximity of launch sites.
- Development of an interactive dashboard with Plotly Dash for analyzing launch records.
- Building a predictive model to forecast the likelihood of successful Falcon 9 first-stage landings.

Summary of Findings

This report presents results in several formats, such as:

- Data analysis outcomes.
- Data visualizations and interactive dashboards.
- Predictive model results and insights.

Introduction

Project Background and Context

With the rise of private space travel, the space industry is becoming increasingly mainstream and accessible to the general public. However, the high costs associated with launches remain a significant barrier for new competitors entering the market.

SpaceX, with its ability to reuse the first stage of its rockets, holds a major competitive advantage. A SpaceX launch costs around $62 million, with the capability to reuse the first stage for future missions. In contrast, competitors often spend upwards of $165 million per launch, giving SpaceX a substantial edge.

Research Questions

Key issues this research aims to address include:

- Can the first stage of SpaceX Falcon 9 land successfully?
- How do different factors (e.g., launch site, payload mass, booster version) affect landing outcomes?
- What is the correlation between launch sites and success rates?

Section 1

Methodology

Methodology

Executive Summary

Data Collection Methods

- Utilized the SpaceX API.
- Web-scraped Falcon 9 and Falcon Heavy launch data from Wikipedia.

Data Wrangling

- Converted mission outcomes into training labels for supervised models (0 = unsuccessful, 1 = successful).

Exploratory Data Analysis (EDA)

- Conducted EDA using data visualizations and SQL queries.

Interactive Visual Analytics

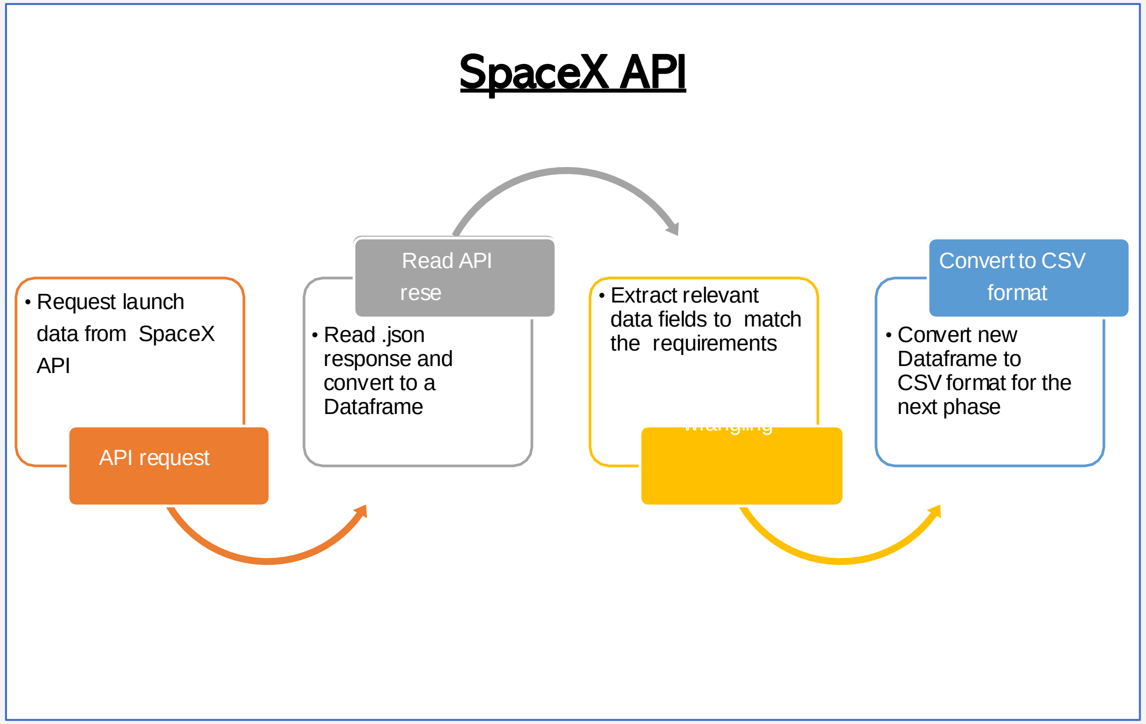
- Created interactive visualizations using Folium and Plotly Dash.

Predictive Analysis Using Classification Models

- Generated a ‘class’ column, standardized and transformed the data, performed a train-test data split, and evaluated different classification algorithms (Logistic Regression, SVM, Decision Trees, KNN) on test data to identify the best model.

Data Collection

Data collection involves gathering information from various sources, which can be structured, unstructured, or semi-structured. In this project, data was sourced through the SpaceX API and by web scraping Wikipedia pages to obtain relevant launch data.



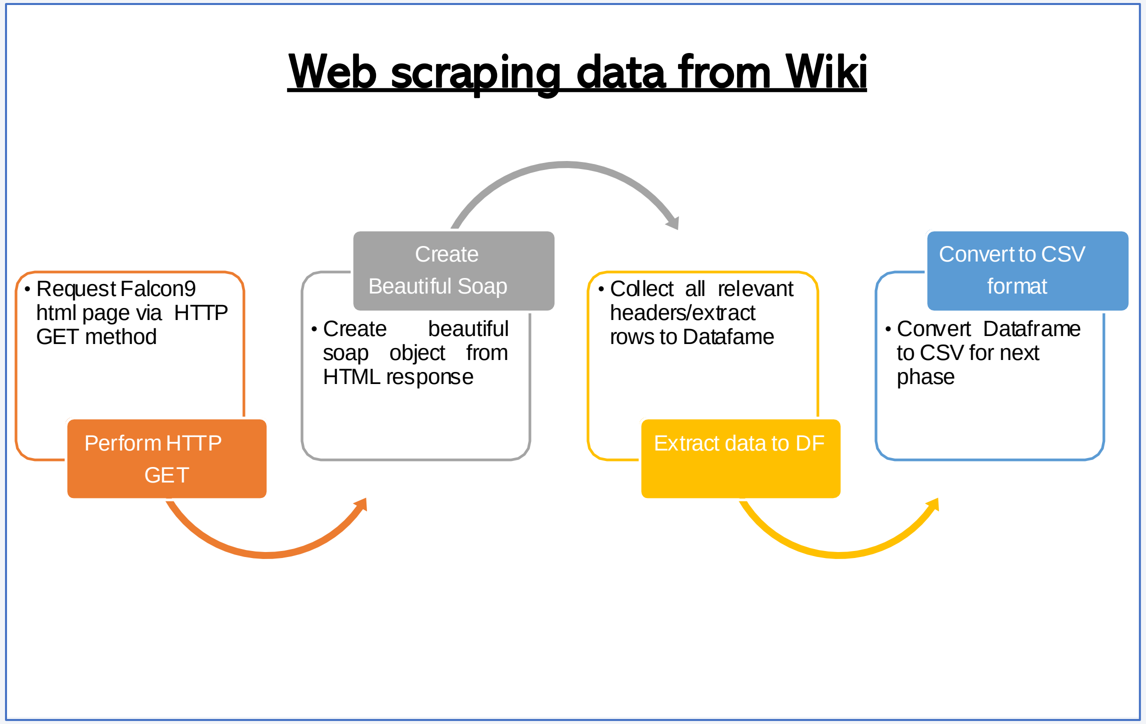
SpaceX API

API request
- Request launch data from SpaceX API

Read API rese
- Read .json response and convert to a Dataframe

Wrangling
- Extract relevant data fields to match the requirements

Convert to CSV format
- Convert new Dataframe to CSV format for the next phase



Web scraping data from Wiki

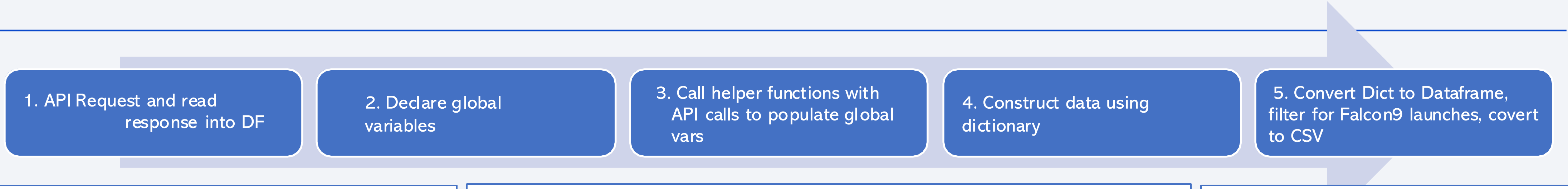
Perform HTTP GET
- Request Falcon9 html page via HTTP GET method

Create Beautiful Soap
- Create beautiful soap object from HTML response

Extract data to DF
- Collect all relevant headers/extract rows to Datafame

Convert to CSV format
- Convert Dataframe to CSV for next phase

Data Collection – SpaceX API



1. API Request and read response into DF

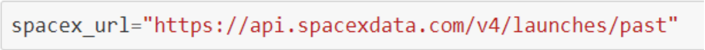
2. Declare global variables

3. Call helper functions with API calls to populate global vars

4. Construct data using dictionary

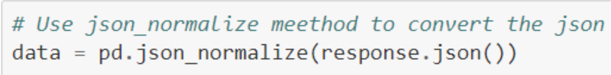
5. Convert Dict to Dataframe, filter for Falcon9 launches, covert to CSV

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



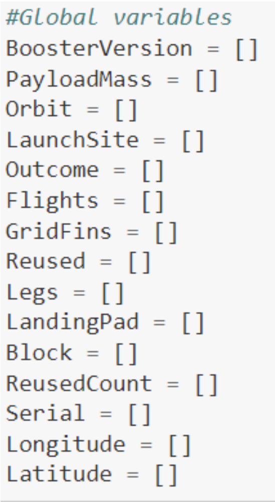
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
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 = []
```

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)

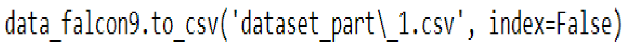
4. Construct dataset from received data & combine columns into a dictionary:

```
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:

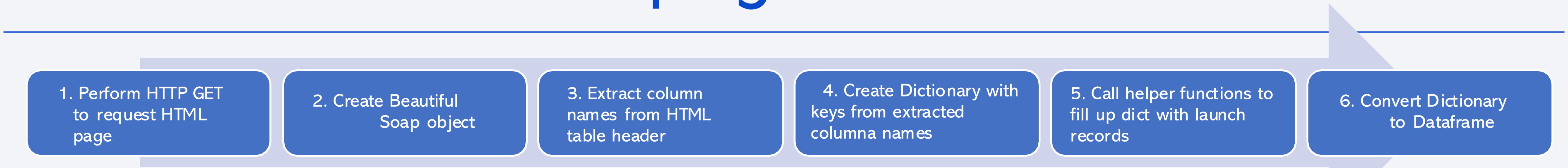
```
# Create a data from launch_dict
df_launch = pd.DataFrame(launch_dict)
```

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

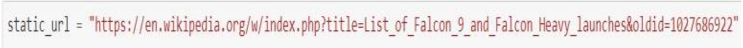


```
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```

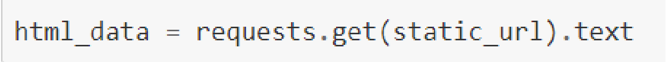
Data Collection - Scraping


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 columna 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

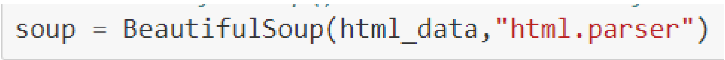


```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```



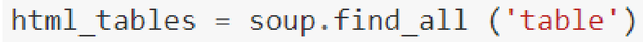
```
html_data = requests.get(static_url).text
```

2. Create Beautiful Soap object

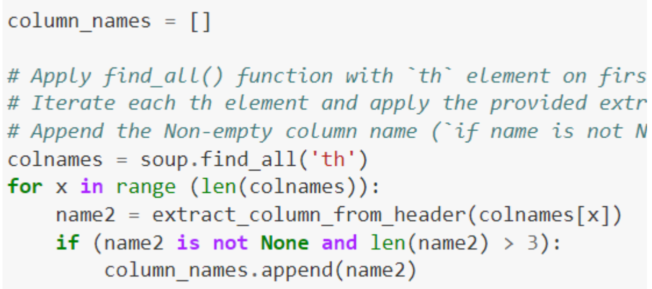


```
soup = BeautifulSoup(html_data,"html.parser")
```

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

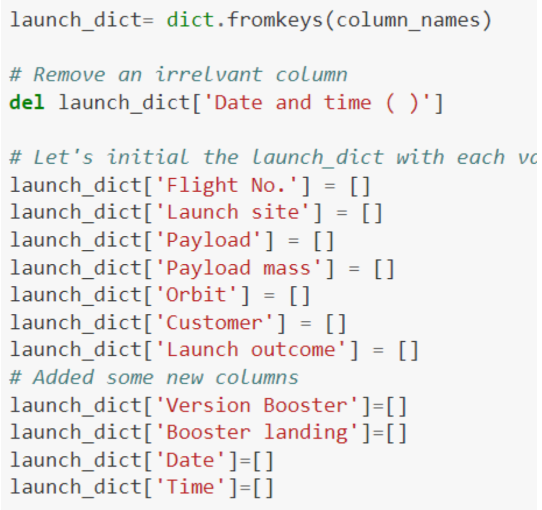


```
html_tables = soup.find_all ('table')
```



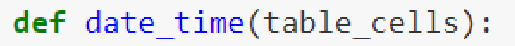
```
column_names = []

# 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)
```

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

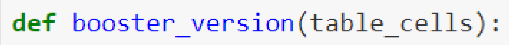


```
launch_dict= dict.fromkeys(column_names)

# 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']=[]
```

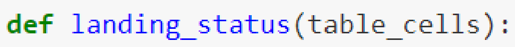
5. Fill up the launch_dict with launch records extracted from table rows.
   - Utilize following helper functions to help parse HTML data



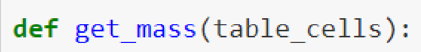
```
def date_time(table_cells):
```



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

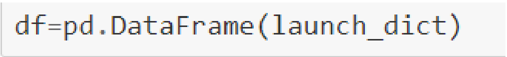


```
def landing_status(table_cells):
```



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

6. Convert launch_dict to Dataframe:



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

Data Wrangling

Exploratory Data Analysis (EDA)

EDA was performed to identify patterns in the data and define labels for training supervised models. The dataset contained various mission outcomes, which were converted into training labels: 1 indicating a successful booster landing and 0 indicating an unsuccessful landing. The following landing scenarios were used to create the labels:

- **True Ocean:** Successful landing in a designated ocean region.
- **False Ocean:** Unsuccessful landing in a designated ocean region.
- **RTLS (Return to Launch Site):** Successful landing on a ground pad.
- **False RTLS:** Unsuccessful landing on a ground pad.
- **True ASDS (Autonomous Spaceport Drone Ship):** Successful landing on a drone ship.
- **False ASDS:** Unsuccessful landing on a drone ship.

EDA with Data Visualization

Exploratory Data Analysis (EDA)

As part of the EDA process, the following charts were plotted to gain deeper insights into the dataset:

- **Scatter Plots:**
  These plots highlight the relationship or correlation between two variables, making patterns easier to detect. The following scatter plots were generated:
  - Flight Number vs. Launch Site
  - Payload vs. Launch Site
  - Flight Number vs. Orbit Type
  - Payload vs. Orbit Type
- **Bar Chart:**
  Bar charts are effective for comparing values of a variable at a specific point in time. The length of each bar is proportional to the value it represents, allowing easy comparison across different groups. The following bar chart was plotted:
  - Success rate by orbit type
- **Line Chart:**
  Line charts are useful for tracking changes over time, revealing trends in the data. The following line chart was created:
  - Yearly trend of average launch success

EDA with SQL

SQL Queries and Operations Performed on IBM DB2 Cloud Instance

To better understand the SpaceX dataset, the following SQL queries and operations were executed:

1. Display the unique launch sites involved in the space missions.
2. Retrieve 5 records where launch site names start with 'CCA’.
3. Calculate the total payload mass carried by boosters launched by NASA (CRS).
4. Find the average payload mass carried by the booster version F9 v1.1.
5. Identify the date of the first successful landing on a ground pad.
6. List the names of boosters that successfully landed on a drone ship and carried a payload mass greater than 4,000 but less than 6,000.
7. Display the total number of successful and failed mission outcomes.
8. List the booster versions that carried the maximum payload mass (using a subquery).
9. List the failed drone ship landings in 2015 along with their booster versions and launch site names.
10. Rank the landing outcomes

Build an Interactive Map with Folium

Interactive Map Using Folium

The Folium interactive map was used to analyze geospatial data, providing a more dynamic way to perform visual analytics. This helped in understanding factors like the location and proximity of launch sites, which may impact the success rate of launches.

The following map objects were created and added to the interactive map:

- Marked all launch sites to visualize their locations on the map.
- Used folium.circle and folium.marker to highlight each launch site with a circle and label.
- Implemented a MarkerCluster() to display launch outcomes, using green markers for successful launches and red markers for failures at each launch site.
- Calculated the distances between launch sites and nearby proximities (e.g., coastline, railroad, highway, city).
- Added MousePosition() to display coordinates dynamically as the mouse moves over points on the map.
- Used folium.Marker() to show distances (in kilometers) from a point (e.g., coastline, railroad, highway, city) to the launch site.
- Implemented folium.Polyline() to draw lines between the launch sites and their nearby proximities.
- Repeated the process of adding markers and drawing lines between the launch sites and proximities such as coastline, railroad, highway, and city.

Building the interactive map with Folium helped answer the following questions:

- Are launch sites close to railways? **Yes**
- Are launch sites near highways? **Yes**
- Are launch sites in proximity to the coastline? **Yes**
- Do launch sites maintain a certain distance from cities? **Yes**

Build a Dashboard with Plotly Dash

Plotly Dash Web Application for Interactive Visual Analytics

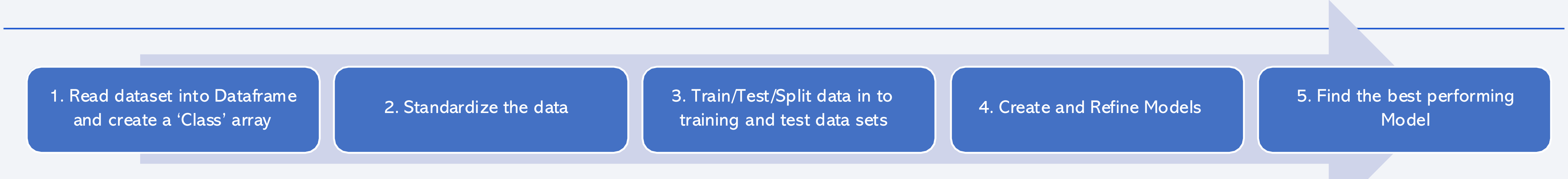
A Plotly Dash web application was developed to perform real-time, interactive visual analytics on SpaceX launch data. Several components were integrated into the dashboard for enhanced user interaction:

- **Launch Site Drop-down:** Allows users to filter visualizations by all launch sites or a specific launch site.
- **Pie Chart:** Displays total successful launches when "All Sites" is selected and shows success vs. failure counts when a particular site is chosen.
- **Payload Range Slider:** Enables users to select different payload ranges to identify patterns in the data.
- **Scatter Chart:** Used to observe correlations between payload and mission outcomes for the selected site(s). The scatter points are color-coded by booster version, allowing for visual differentiation of mission outcomes by different boosters.

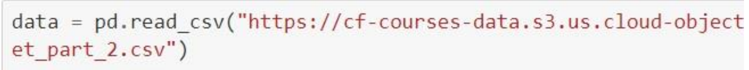
The dashboard helped answer the following key questions:

- **Which site has the largest number of successful launches?** *KSC LC-39A with 10 successful launches.*
- **Which site has the highest launch success rate?** *KSC LC-39A with a 76.9% success rate.*
- **Which payload range has the highest launch success rate?** *2000–5000 kg.*
- **Which payload ranges have the lowest launch success rates?** *0–2000 kg and 5500–7000 kg.*
- **Which Falcon 9 booster version has the highest launch success rate?** *FT (Full Thrust).*

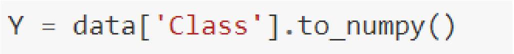
Predictive Analysis (Classification)


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

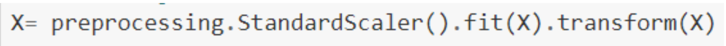


```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object
et_part_2.csv")
```



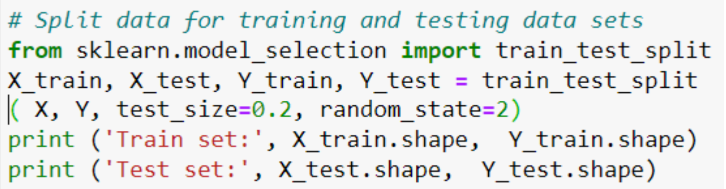
```
Y = data['Class'].to_numpy()
```

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



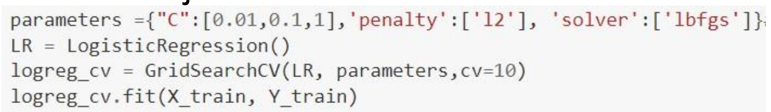
```
X= preprocessing.StandardScaler().fit(X).transform(X)
```

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



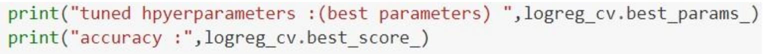
```
# Split data for training and testing data sets
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split
( X, Y, test_size=0.2, random_state=2)
print ('Train set:', X_train.shape,  Y_train.shape)
print ('Test set:', X_test.shape,  Y_test.shape)
```

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



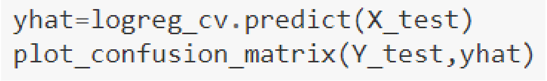
```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}
LR = LogisticRegression()
logreg_cv = GridSearchCV(LR, parameters,cv=10)
logreg_cv.fit(X_train, Y_train)
```

   iii. Find and display best hyperparameters and accuracy score



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

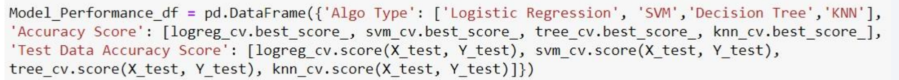
   iv. Check the accuracy on the test data by creating a confusion matrix



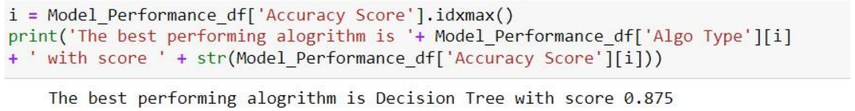
```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

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

3. Find the best performing model

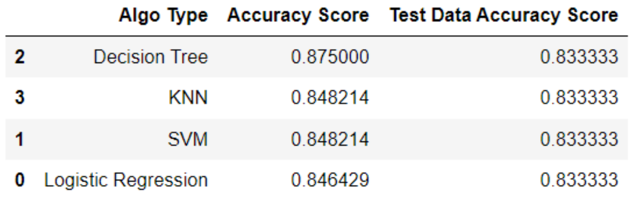


```
Model_Performance_df = pd.DataFrame({'Algo Type': ['Logistic Regression', 'SVM','Decision Tree','KNN'],
'Accuracy Score': [logreg_cv.best_score_, svm_cv.best_score_, tree_cv.best_score_, knn_cv.best_score_],
'Test Data Accuracy Score': [logreg_cv.score(X_test, Y_test), svm_cv.score(X_test, Y_test),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



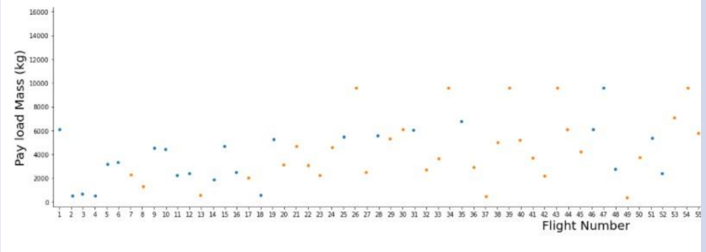
	Algo Type	Accuracy Score	Test Data Accuracy Score
2	Decision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333
1	SVM	0.848214	0.833333
0	Logistic Regression	0.846429	0.833333

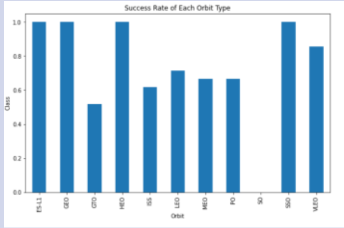
Results

Following sections and slides explain results for:

Exploratory data analysis results

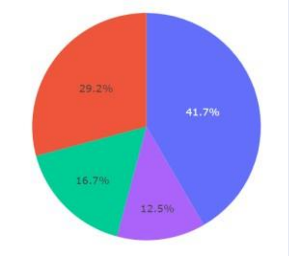
- Samples

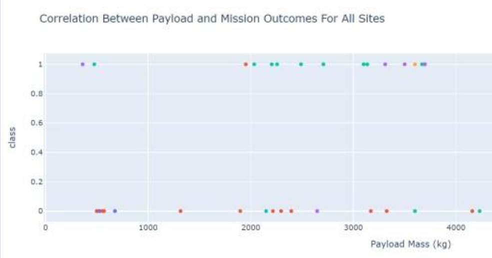




Interactive analytics demo in screenshots

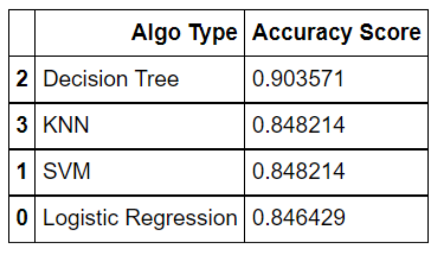
- Samples



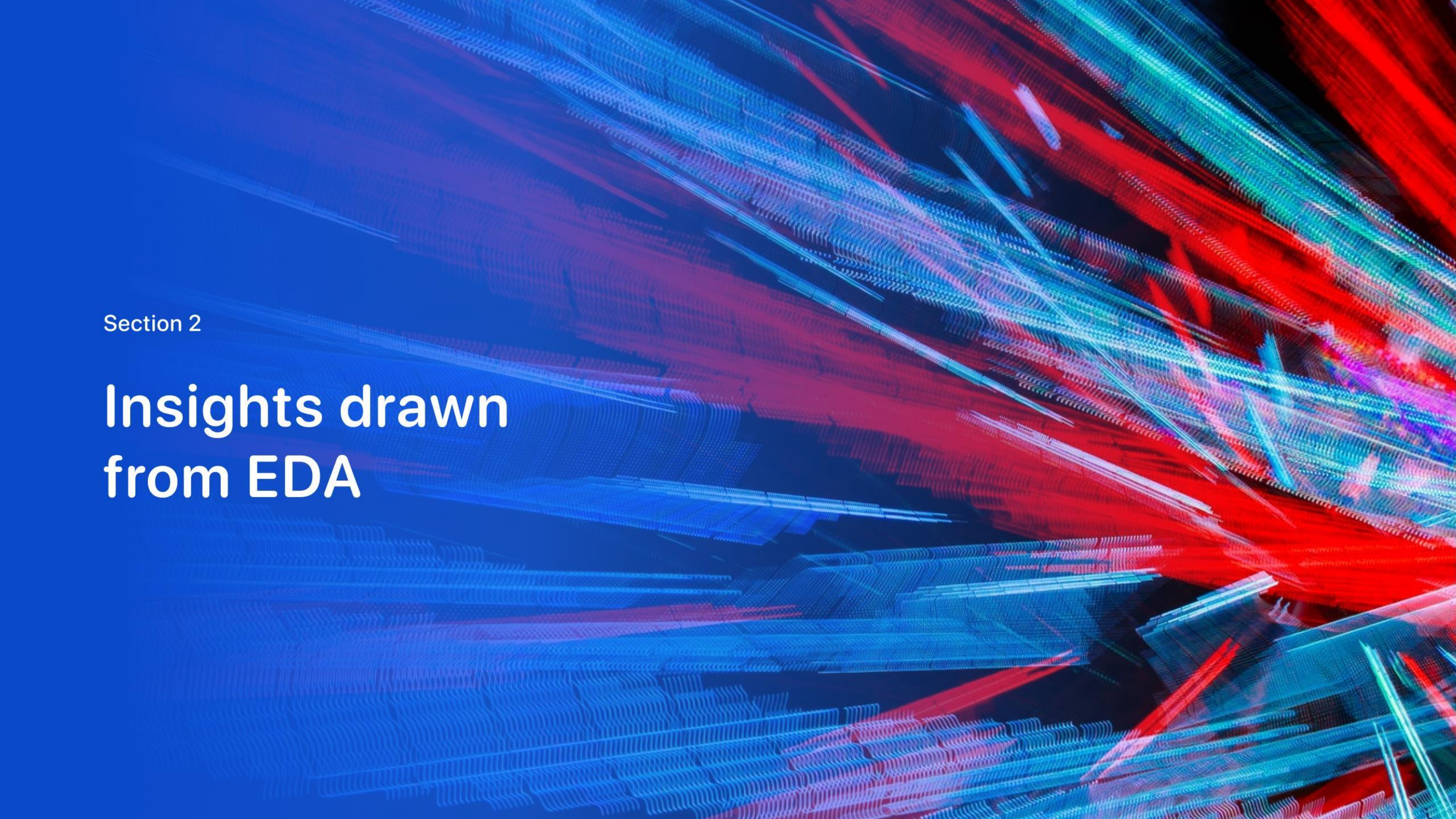


Predictive analysis results

- Samples



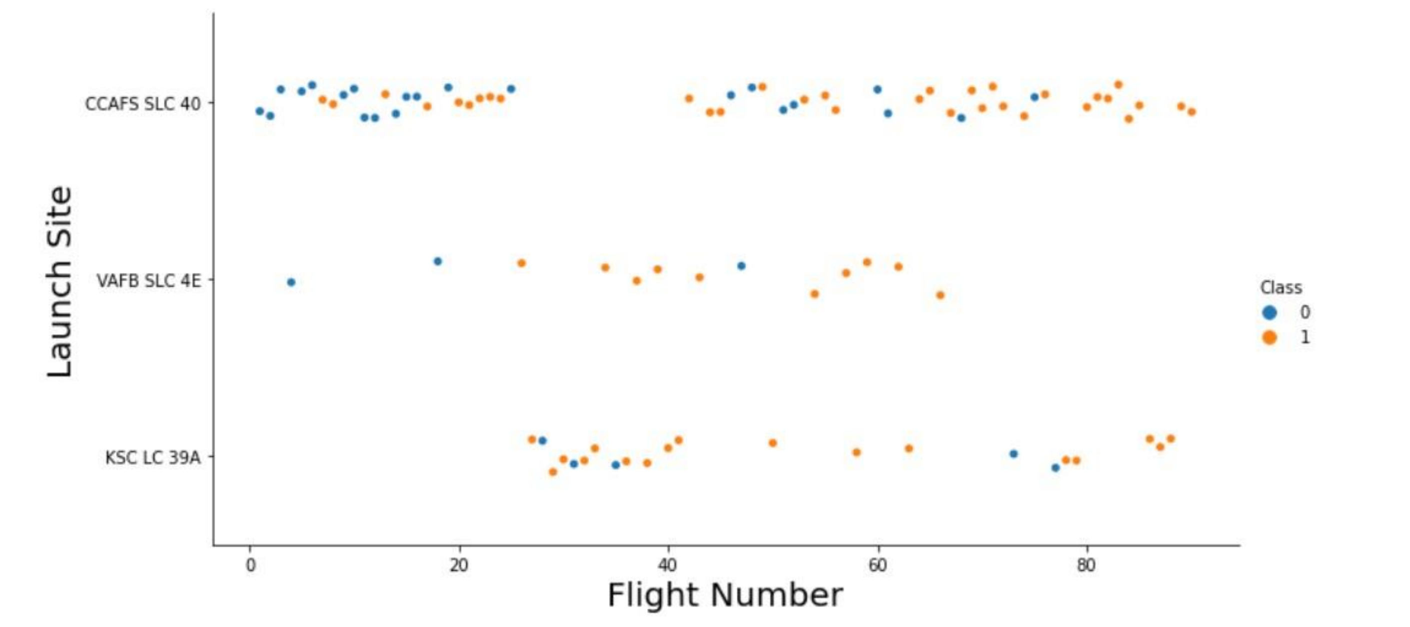
	Algo Type	Accuracy Score
2	Decision Tree	0.903571
3	KNN	0.848214
1	SVM	0.848214
0	Logistic Regression	0.846429



Section 2

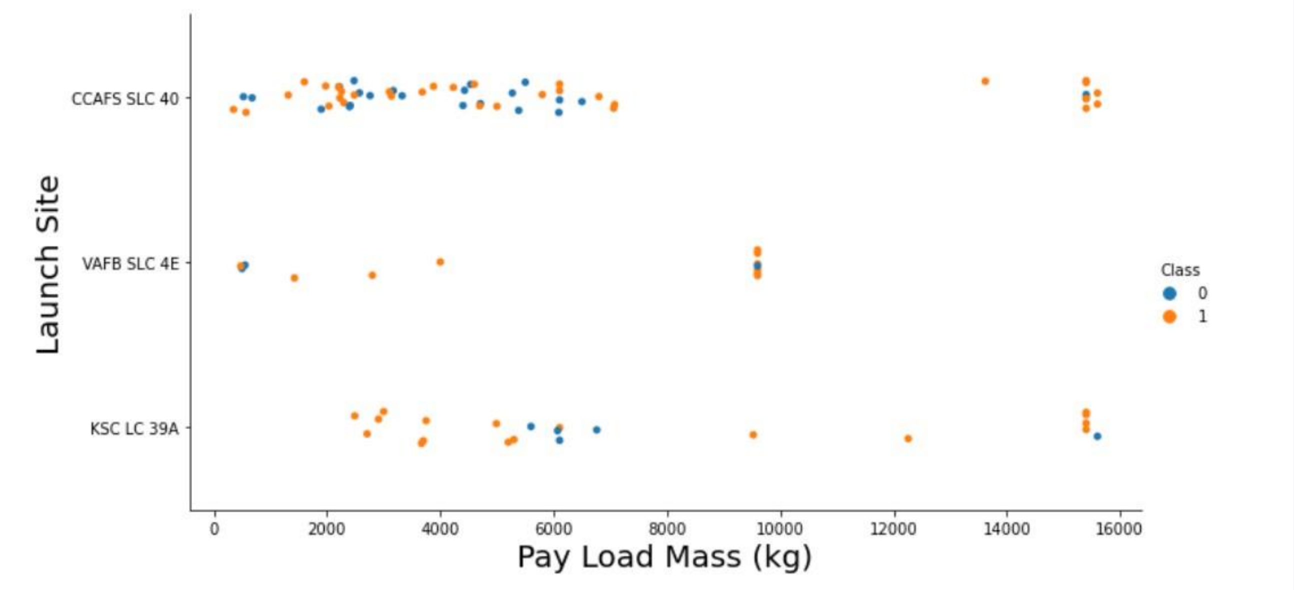
Insights drawn from EDA

Flight Number vs. Launch Site



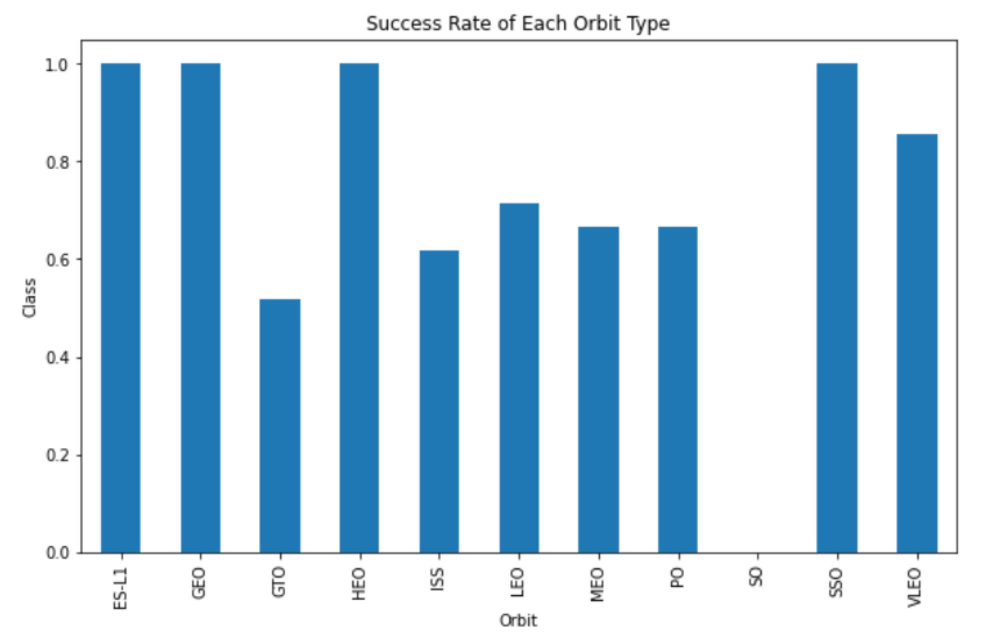
The success rates (Class=1) improve with an increasing number of flights. For the launch site 'KSC LC 39A', it typically requires around 25 launches to achieve the first successful launch.

Payload vs. Launch Site



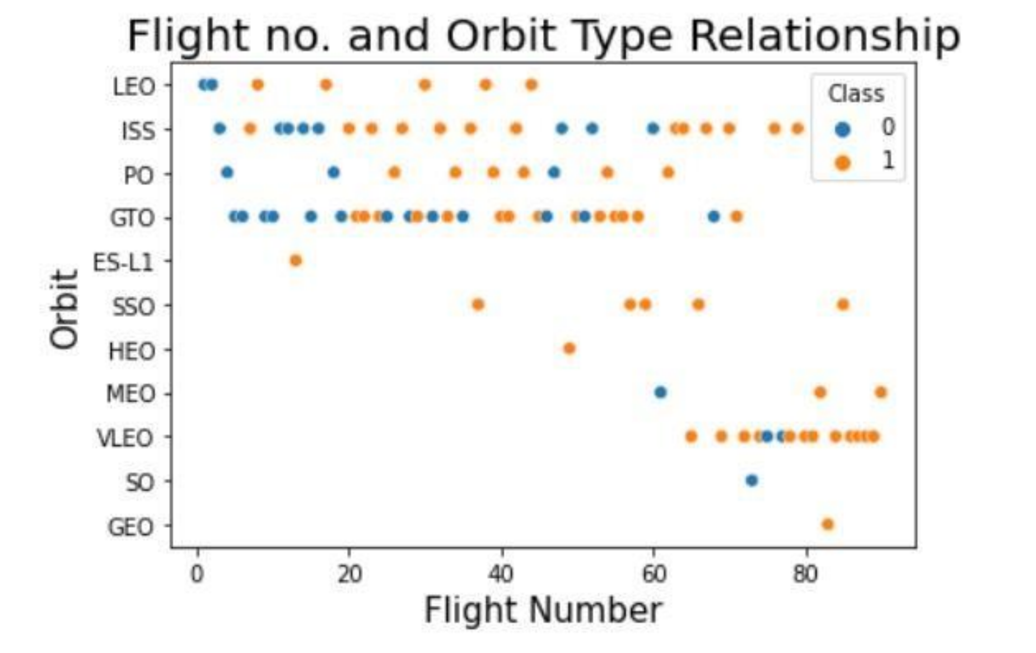
At the launch site ‘VAFB SLC 4E’, no rockets have been launched with payloads exceeding 10,000 kg. However, the percentage of successful launches (Class=1) at this site increases as the payload mass rises. Additionally, there is no discernible correlation or pattern between the launch site and the payload mass.

Success Rate vs. Orbit Type



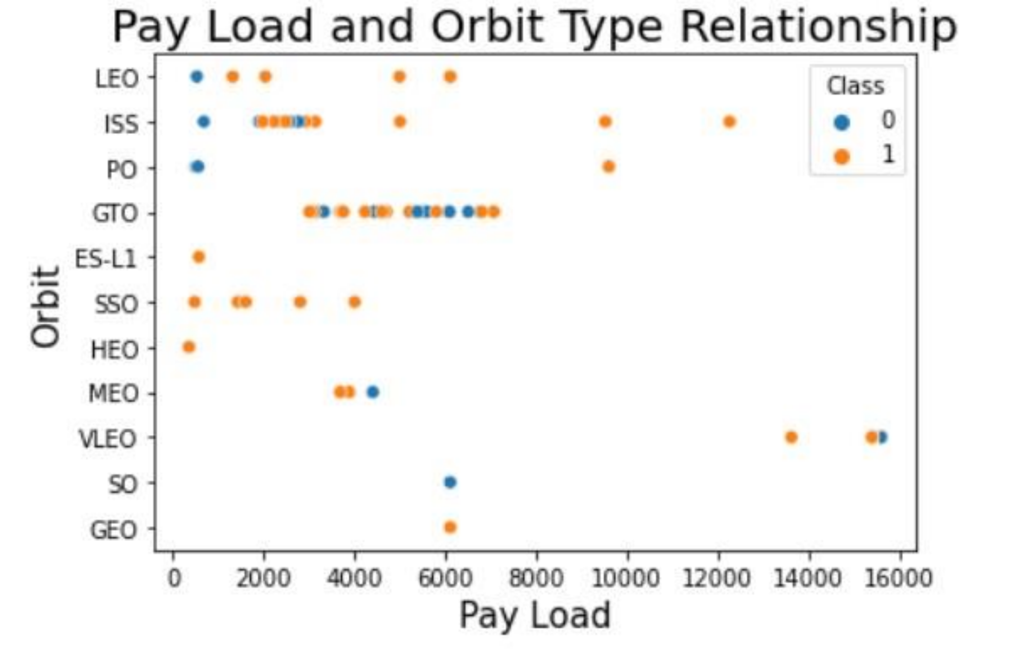
The orbits ES-LI, GEO, HEO, and SSO exhibit the highest success rates, while the GTO orbit has the lowest success rate.

Flight Number vs. Orbit Type



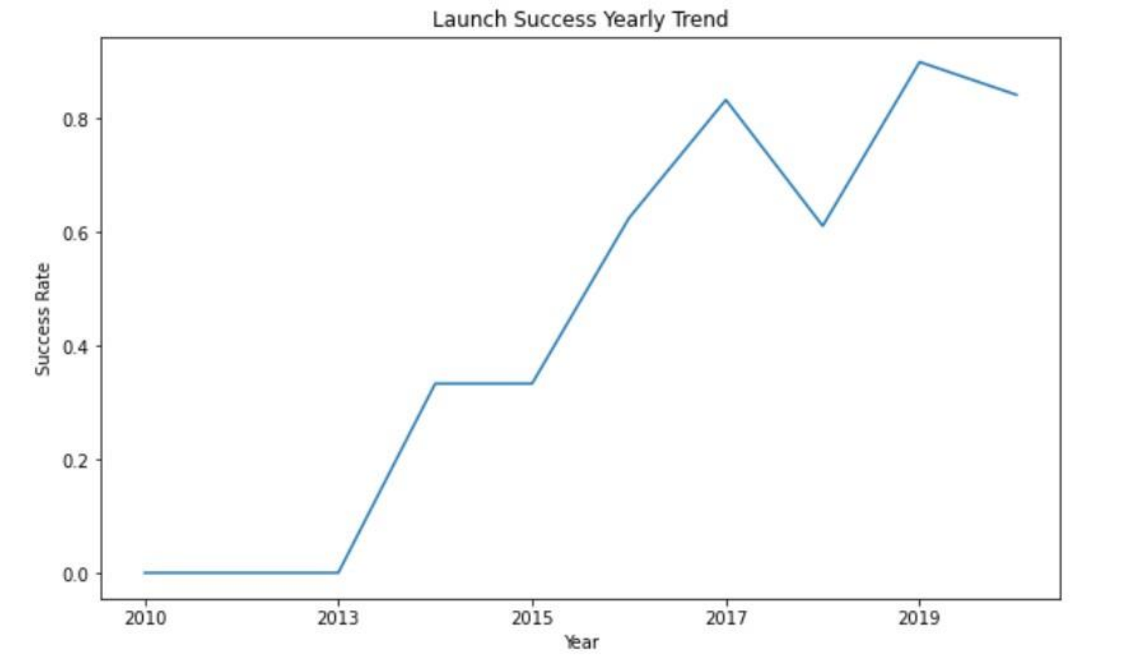
For the VLEO orbit, the first successful landing (Class=1) does not take place until after more than 60 flights. In contrast, for most orbits (LEO, ISS, PO, SSO, MEO, VLEO), the rates of successful landings tend to increase as the number of flights rises. However, there is no apparent relationship between the number of flights and the GTO orbit.

Payload vs. Orbit Type



Successful landing rates (Class=1) seem to rise with payload for the LEO, ISS, PO, and SSO orbits. In contrast, for the GEO orbit, there is no discernible pattern between payload and landing outcomes, whether successful or unsuccessful.

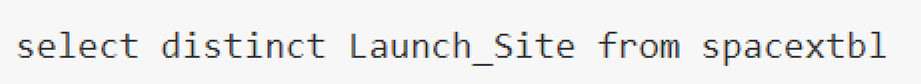
Launch Success Yearly Trend



The success rate (Class=1) rose by approximately 80% from 2013 to 2020. During the periods from 2010 to 2013 and from 2014 to 2015, the success rates remained unchanged. However, there was a decline in success rates between 2017 and 2018, as well as between 2019 and 2020.

All Launch Site Names

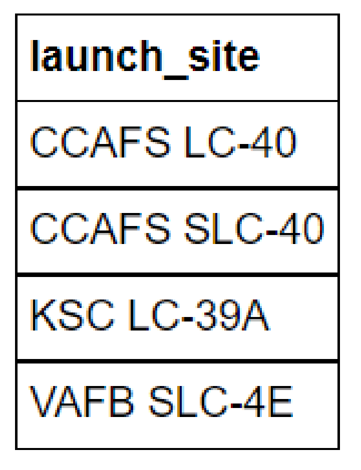
- Query:



```
select distinct Launch_Site from spacextbl
```

- Description:
  - The term 'distinct' retrieves only unique values from the queries column (Launch_Site). There are four unique launch sites in total.

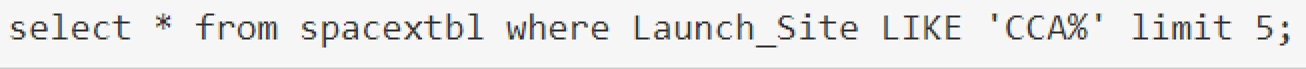
- Result:



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

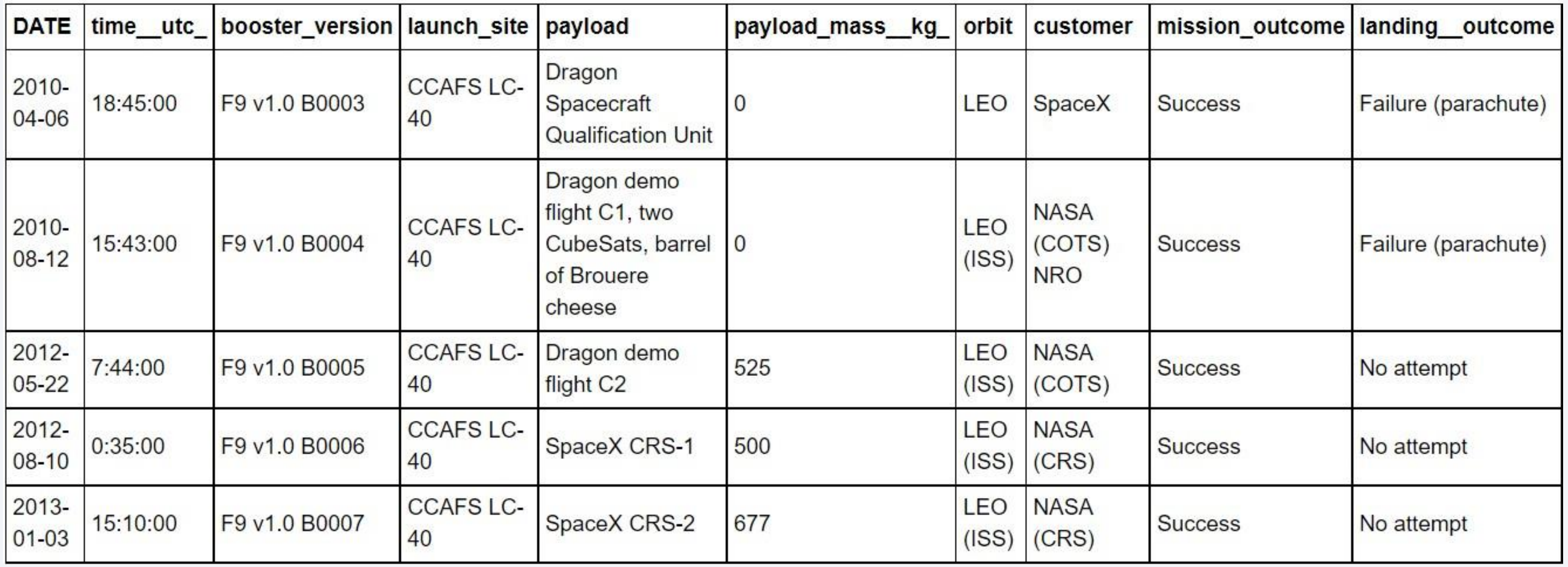
Launch Site Names Begin with 'CCA'

- <u>Query</u>:



```
select * from spacextbl where Launch_Site LIKE 'CCA%' limit 5;
```

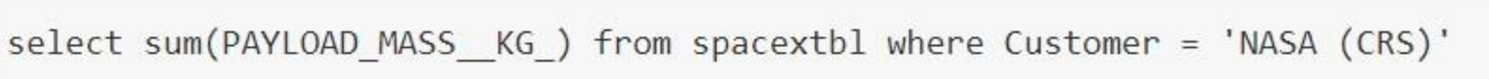
- <u>Description</u>:
  - Using the keyword 'Like' with the format 'CCA%', retrieves records from the 'Launch_Site' column where the values begin with "CCA." The 'Limit 5' clause restricts the number of returned records to five.
- <u>Result</u>:



DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

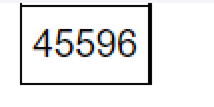
- Query:



```
select sum(PAYLOAD_MASS__KG_) from spacextbl where Customer = 'NASA (CRS)'
```

- Description:
  - The function 'sum' calculates the total payload mass by adding the values in the 'PAYLOAD_MASS_KG' column for customers named 'NASA (CRS).'

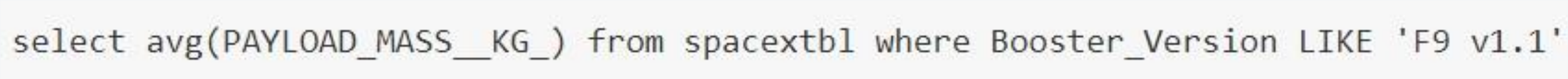
- Result:



45596

Average Payload Mass by F9 v1.1

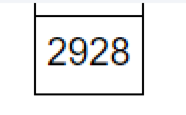
- Query:



```
select avg(PAYLOAD_MASS__KG_) from spacextbl where Booster_Version LIKE 'F9 v1.1'
```

- Description:
  - The 'avg' keyword computes the average payload mass from the 'PAYLOAD_MASS_KG' column for entries where the booster version is 'F9 v1.1.'

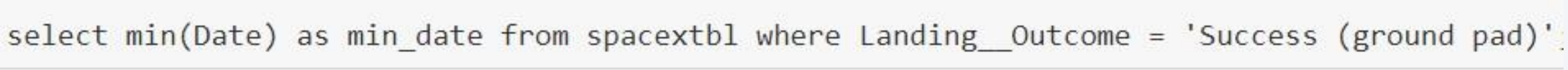
- Result:



2928

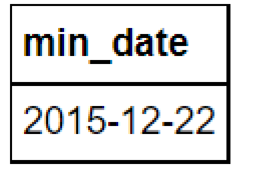
First Successful Ground Landing Date

- Query:



```
select min(Date) as min_date from spacextbl where Landing__Outcome = 'Success (ground pad)'
```

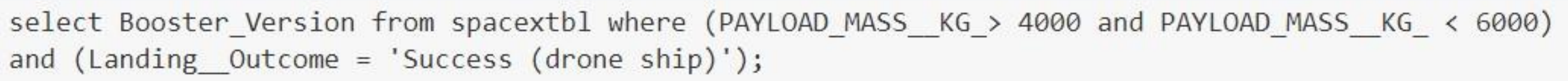
- Description:
  - The function 'min(Date)' selects the earliest or oldest date from the 'Date' column, indicating the first successful landing achieved on the group pad. The 'Where' clause specifies the criteria, filtering for scenarios where the 'Landing_Outcome' value is equal to 'Success (ground pad).'
- Result:



min_date
2015-12-22

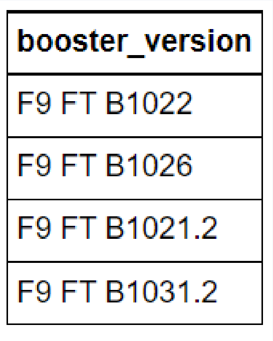
Successful Drone Ship Landing with Payload between 4000 and 6000

- Query:



```
select Booster_Version from spacextbl where (PAYLOAD_MASS__KG_> 4000 and PAYLOAD_MASS__KG_ < 6000)
and (Landing__Outcome = 'Success (drone ship)');
```

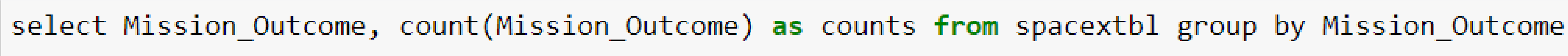
- Description:
  - The query identifies the booster versions for which the payload mass is greater than 4,000 but less than 6,000, and the landing outcome is a success on the drone ship. The 'and' operator in the 'where' clause ensures that both conditions must be true for the booster versions to be returned.
- 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

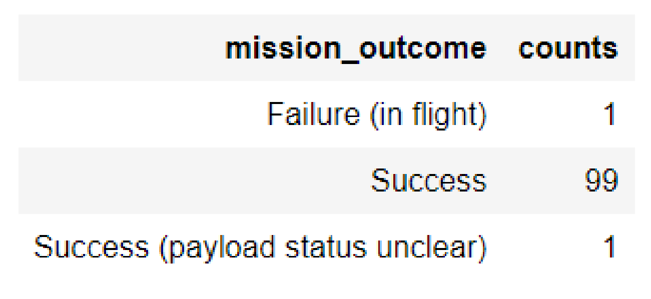
- Query:



```
select Mission_Outcome, count(Mission_Outcome) as counts from spacextbl group by Mission_Outcome
```

- Description:
  - The 'group by' keyword organizes identical data within a column into distinct groups. In this case, it groups the number of mission outcomes by their respective types, with the results being displayed in the 'counts' column.

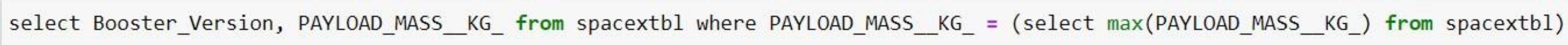
- Result:



mission_outcome	counts
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

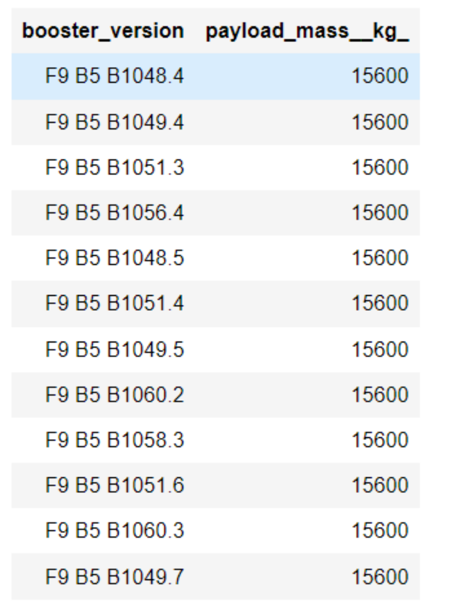
Boosters Carried Maximum Payload

- Query:



```
select Booster_Version, PAYLOAD_MASS__KG_ from spacextbl where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from spacextbl)
```

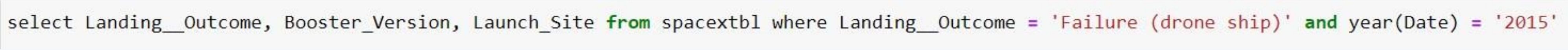
- Description:
  - The subquery retrieves the maximum payload mass by utilizing the keyword 'max' on the payload mass column. The main query then returns the booster versions along with their respective payload mass, specifically for entries where the payload mass equals the maximum value of 15,600.
- Result:



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

2015 Launch Records

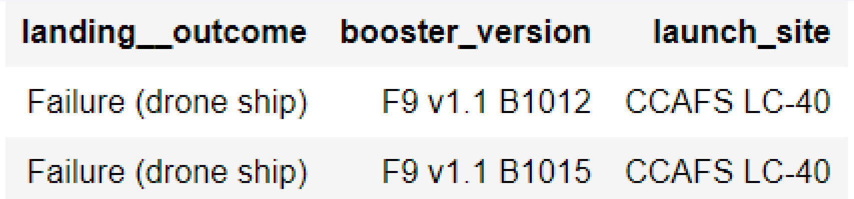
- Query:



```
select Landing__Outcome, Booster_Version, Launch_Site from spacextbl where Landing__Outcome = 'Failure (drone ship)' and year(Date) = '2015'
```

- Description:
  - The query retrieves the landing outcome, booster version, and launch site for instances where the landing outcome was a failure on the drone ship in the year 2015. The 'and' operator in the 'where' clause ensures that both conditions must be satisfied for the booster versions to be included. The 'year' keyword extracts the year from the 'Date' column. The results indicate that the launch site is 'CCAFS LC-40' and the booster versions are 'F9 v1.1 B1012' and 'B1015,' which experienced failed landing outcomes on the drone ship in 2015.

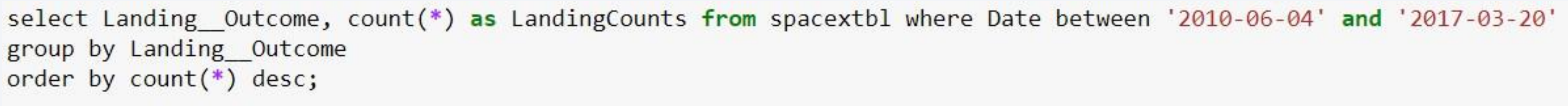
- Result:



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

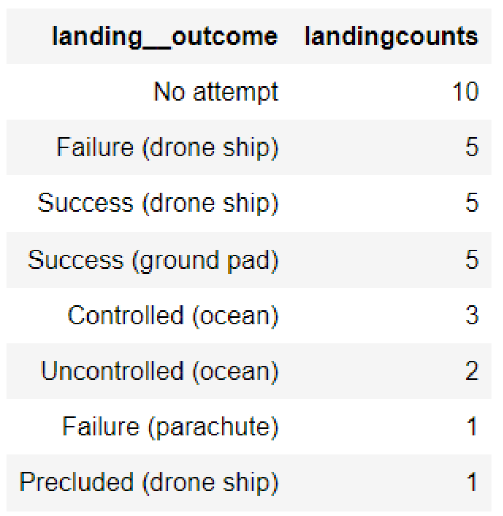
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- Query:



```
select Landing__Outcome, count(*) as LandingCounts from spacextbl where Date between '2010-06-04' and '2017-03-20'
group by Landing__Outcome
order by count(*) desc;
```

- Description:
  - The 'group by' keyword organizes the data in the 'Landing Outcome' column into distinct groups. The 'between' and 'and' keywords filter the data to include only records dated between June 4, 2010, and March 20, 2017. The 'order by' keyword sorts the counts column in descending order. The result of the query is a ranked list of landing outcome counts within the specified date range.
- Result:



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

Section 4

Launch Sites Proximities Analysis

SpaceX Falcon9 - Launch Sites Map

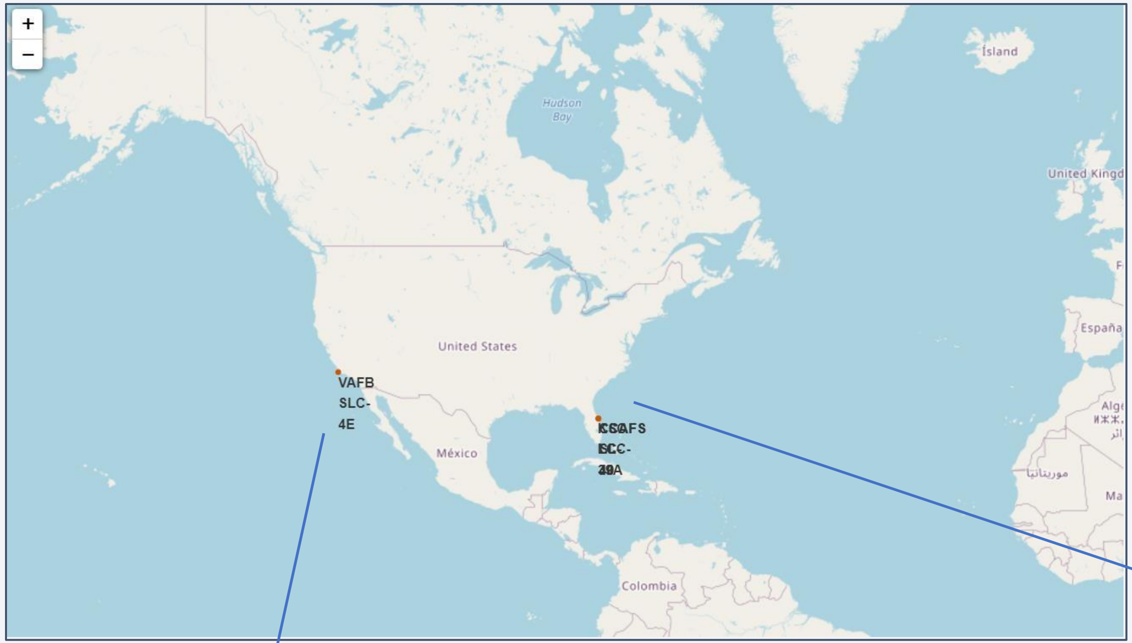


Fig 1 – Global Map

Figure 1 on the left presents a global map highlighting Falcon 9 launch sites located in the United States, specifically in California and Florida. Each launch site is represented by a circle, accompanied by a label and a popup that emphasizes the location and name of the launch site. Notably, all launch sites are situated near the coast.

Figures 2 and 3 provide a closer view of the launch sites, showcasing four specific locations:

- VAFB SLC-4E (CA)
- CCAFS LC-40 (FL)
- KSC LC-39A (FL)
- CCAFS SLC-40 (FL)

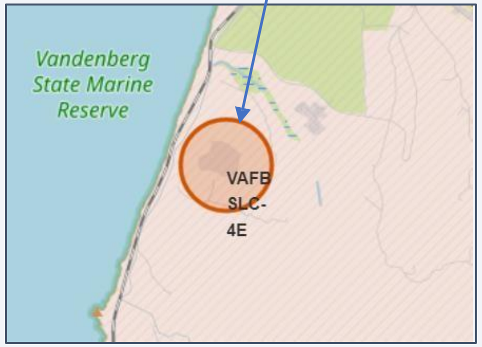


Fig 2 – Zoom 1

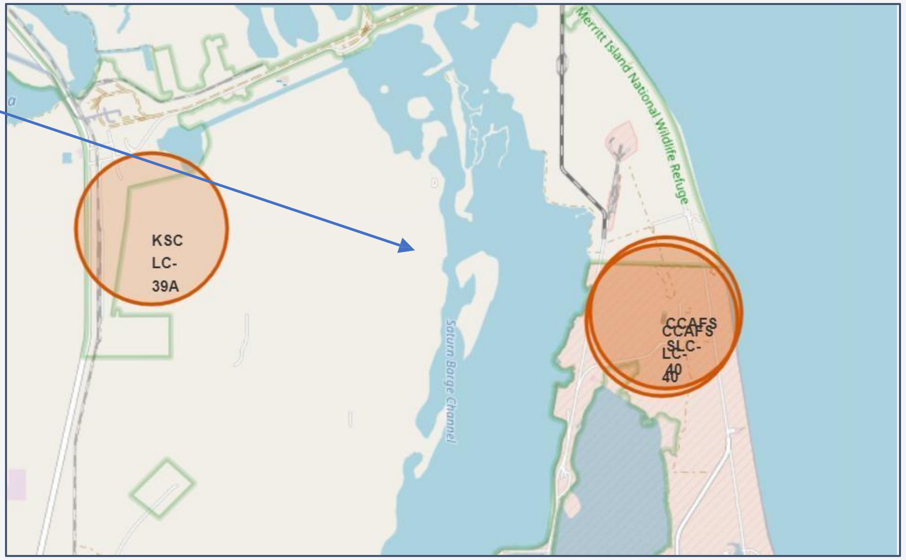


Fig 3 – Zoom 2

SpaceX Falon9 – Success/Failed Launch Map for all Launch Sites

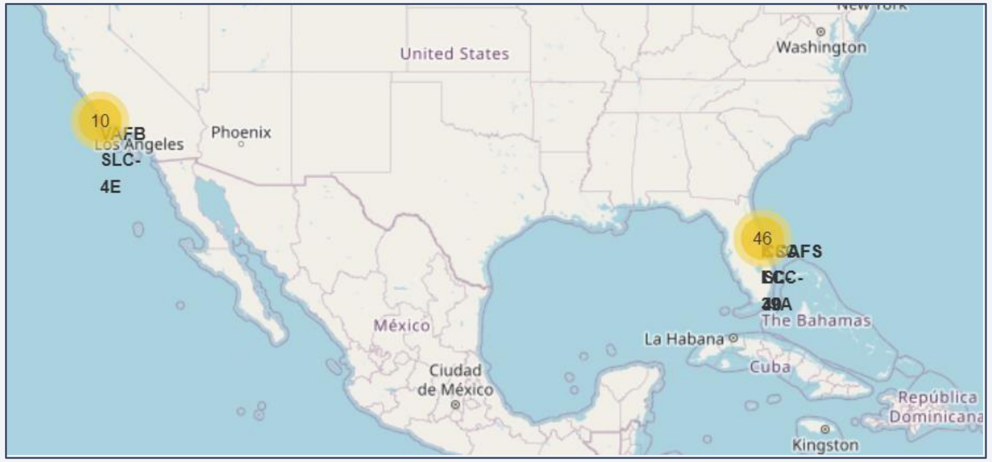


Fig 1 – US map with all Launch Sites

- Figure 1 shows a map of the United States with all the launch sites marked. The numbers at each site indicate the total count of successful and failed launches.
- Figures 2, 3, 4, and 5 provide a closer view of each site, displaying markers that indicate the outcomes of the launches: green markers represent successful launches, while red markers indicate failed launches.
- Upon reviewing the maps of each site, it is evident that the KSC LC-39A Launch Site has the highest number of successful launches.

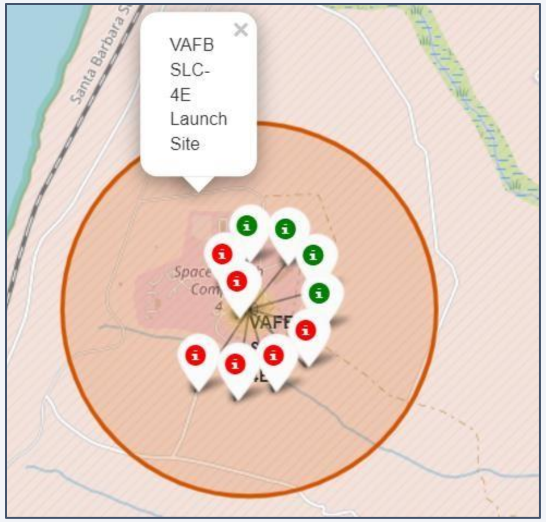


Fig 2 – VAFB Launch Site with success/failed markers

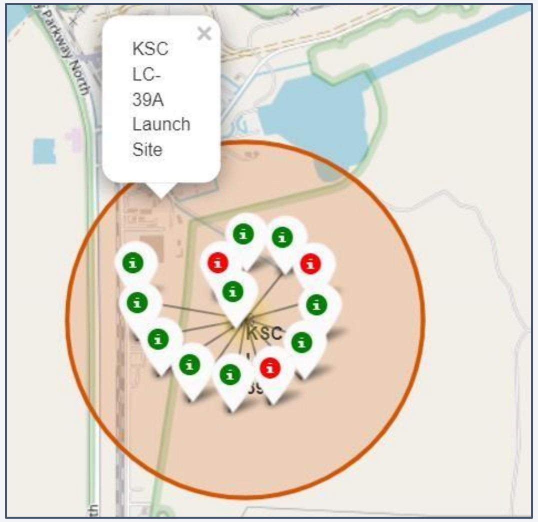


Fig 3 – KSC LC-39A success/failed markers

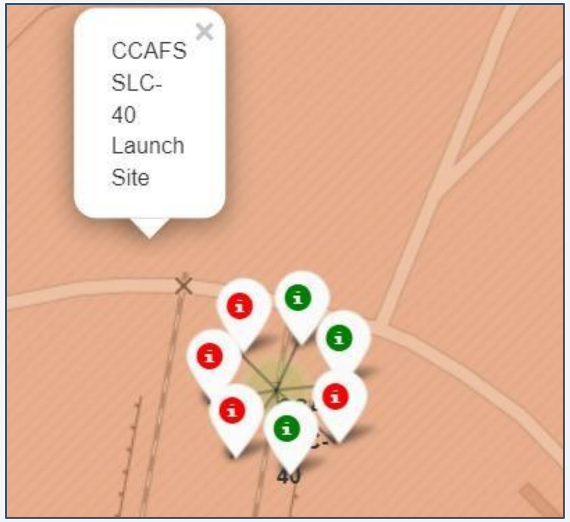


Fig 4 – CCAFS SLC-40 success/failed markers

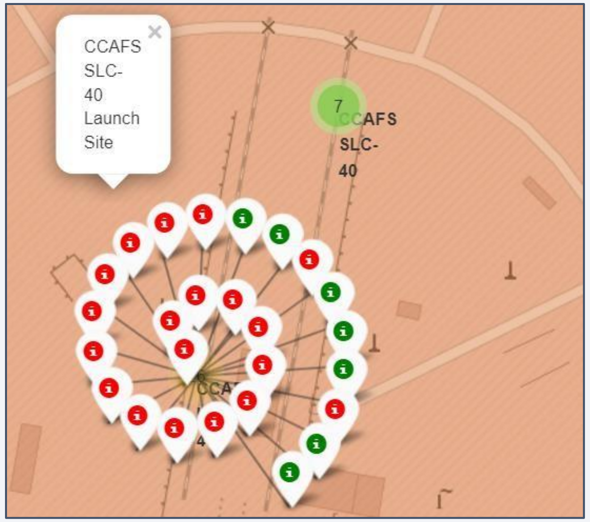


Fig 5 – CCAFS SLC-40 success/failed markers

SpaceX Falcon9 – Launch Site to proximity Distance Map

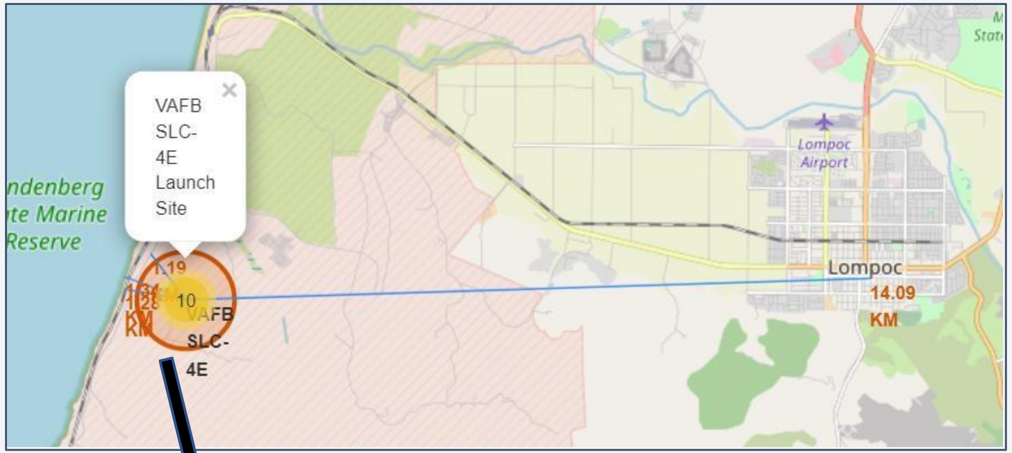


Fig 1 – Proximity site map for VAFB SLC-4E

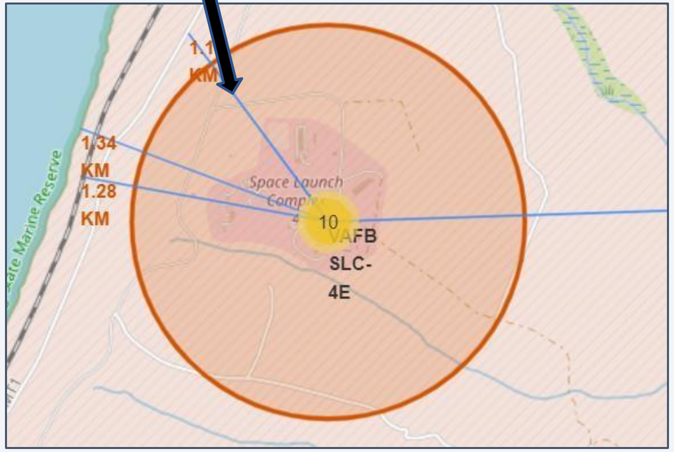
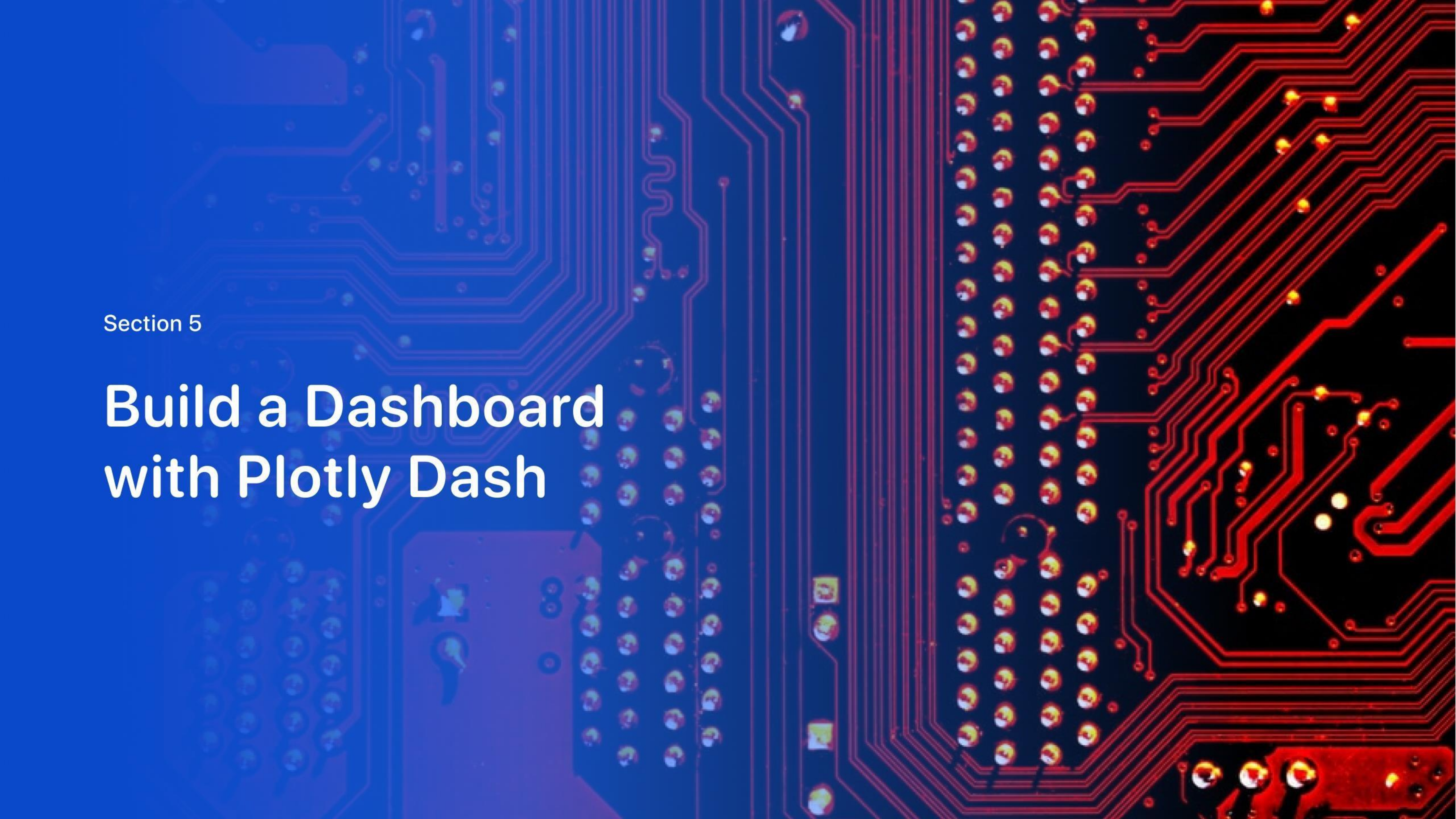


Fig 2 – Zoom in for sites – coastline, railroad, and highway

Figure 1 shows the proximity sites marked on the map for the VAFB SLC-4E Launch Site. Among these, the city of Lompoc is situated farther away from the Launch Site compared to other nearby features such as the coastline, railroad, and highway. The map also includes a marker indicating the distance from the Launch Site to Lompoc, which is 14.09 km.

Figure 2 offers a zoomed-in view of other proximity features, including the coastline, railroad, and highway, along with their respective distances from the Launch Site.

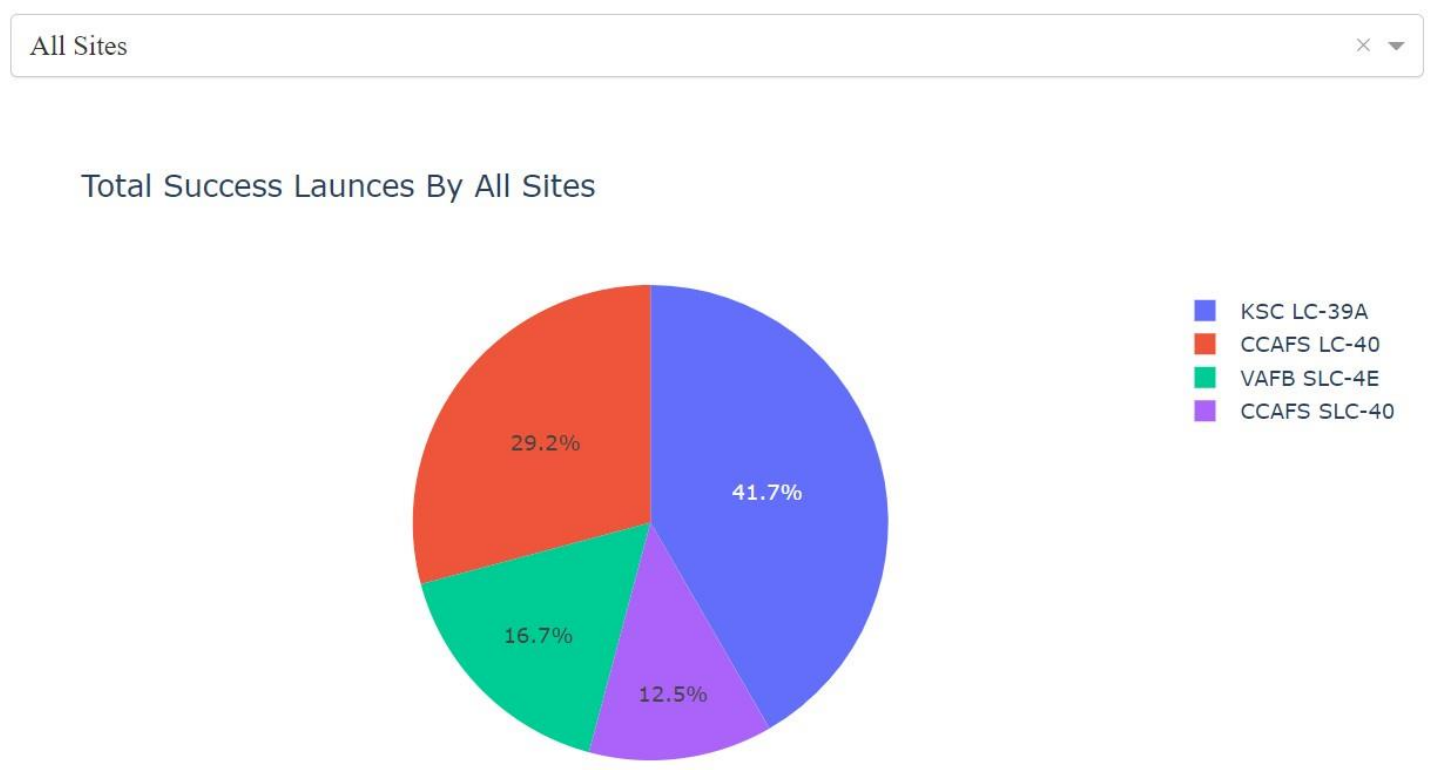
Overall, cities are intentionally located at a distance from Launch Sites to minimize the potential impact of any accidental incidents on the general public and existing infrastructure. Launch Sites are strategically positioned near coastlines, railroads, and highways to ensure easy access to necessary resources.



Section 5

Build a Dashboard with Plotly Dash

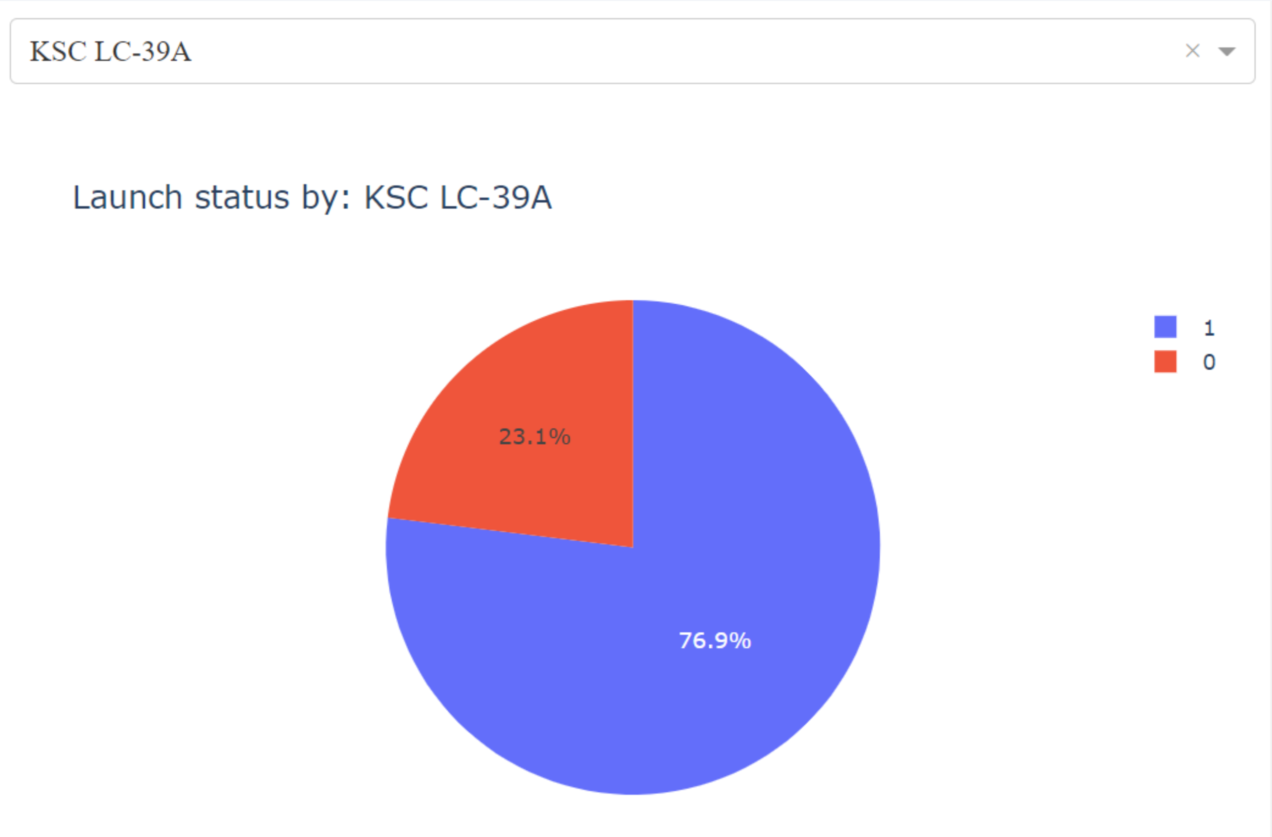
Launch Success Counts For All Sites



All Sites

- Launch Site ‘KSC LC-39A’ boasts the highest launch success rate, while Launch Site ‘CCAFS SLC-40’ has the lowest launch success rate.

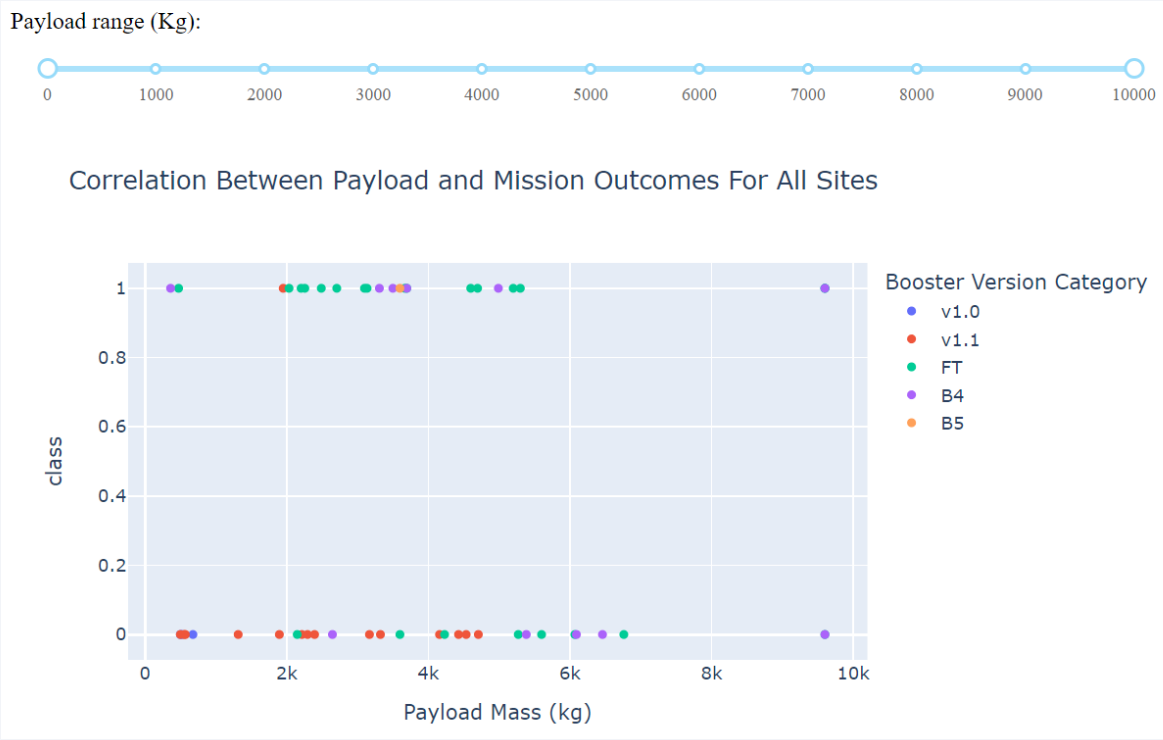
Launch Site with Highest Launch Success Ratio

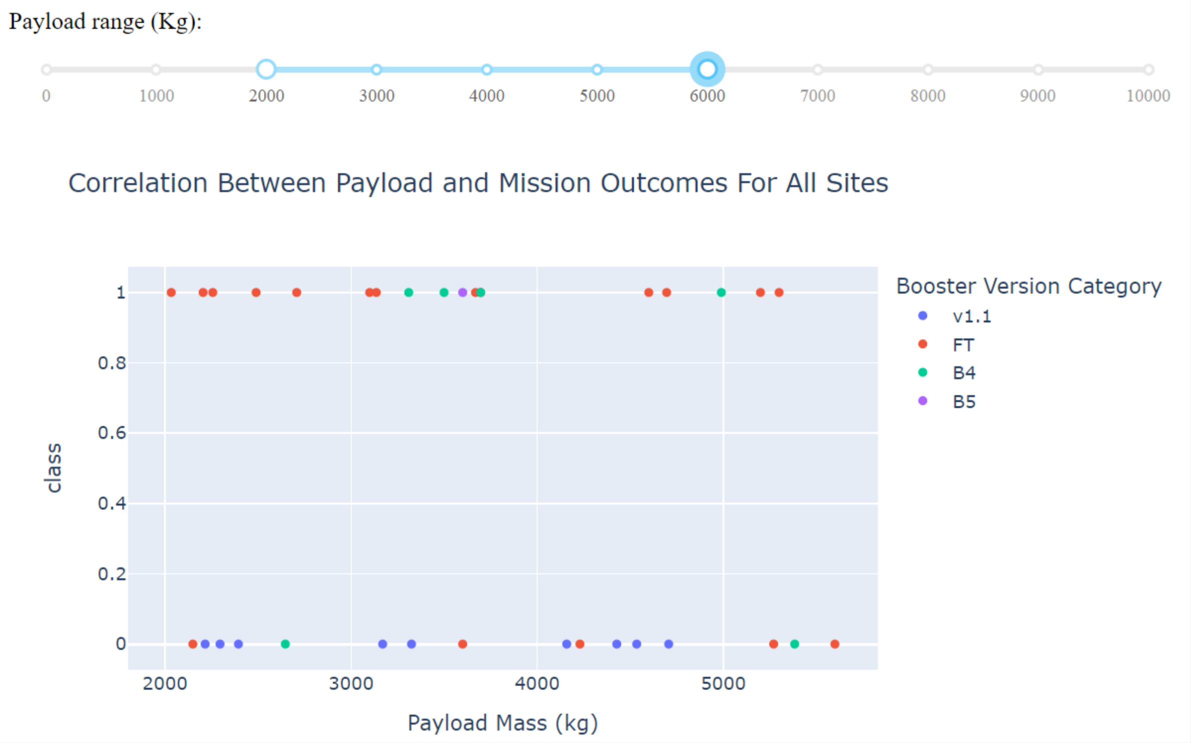

KSC LC-39A

Launch status by: KSC LC-39A

- The KSC LC-39A Launch Site has the highest launch success rate and count, with a launch success rate of 76.9% and a launch failure rate of 23.1%.

Payload vs. Launch Outcome Scatter Plot for All Sites



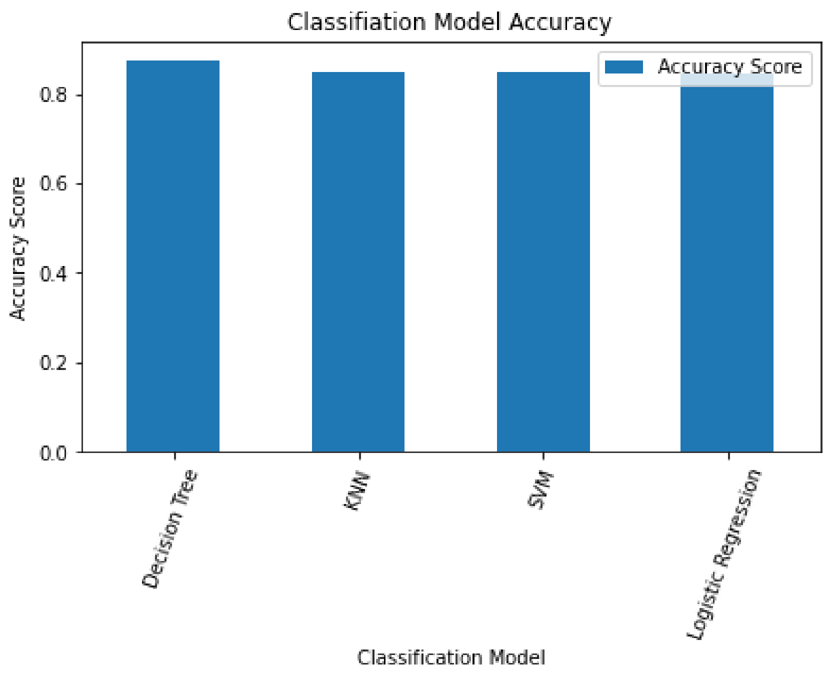


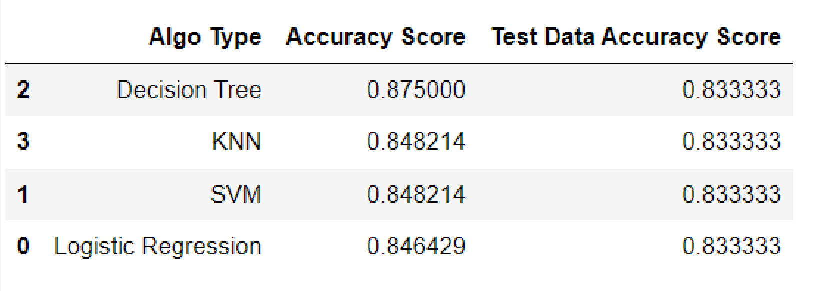
The majority of successful launches fall within the payload range of 2,000 to approximately 5,500 kg. The booster version category 'FT' has recorded the highest number of successful launches. Notably, the only booster that achieved a successful launch with a payload exceeding 6,000 kg is 'B4.'

Section 6

Predictive Analysis (Classification)

Classification Accuracy

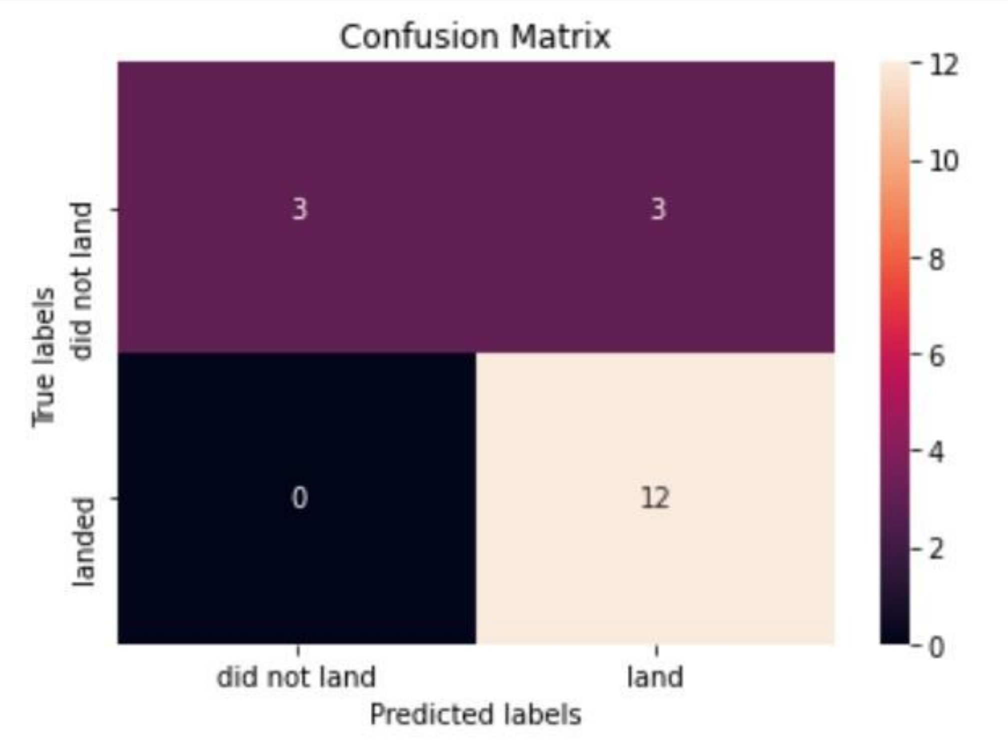




	Algo Type	Accuracy Score	Test Data Accuracy Score
2	Decision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333
1	SVM	0.848214	0.833333
0	Logistic Regression	0.846429	0.833333

- Based on the accuracy scores, as illustrated in the bar chart, the Decision Tree algorithm achieves the highest classification score, with a value of 0.8750. The accuracy score on the test data is consistent across all classification algorithms, each scoring 0.8333. Given that the accuracy scores for the classification algorithms are quite similar and the test scores are identical, it may be beneficial to utilize a broader data set to further refine and optimize the models.

Confusion Matrix



- The confusion matrix is identical for all models (LR, SVM, Decision Tree, KNN). According to the confusion matrix, the classifier made a total of 18 predictions:
  - **True Positives (TP):** 12 scenarios were predicted as "Yes" for landing, and these landings were successful.
  - **True Negatives (TN):** 3 scenarios (top left) were predicted as "No" for landing, and these did not land (successfully).
  - **False Positives (FP):** 3 scenarios (top right) were predicted as "Yes" for landing, but they did not land successfully.
- Overall, the classifier is correct approximately 83% of the time, calculated as *(TP + TN) / Total*, while the misclassification or error rate, defined as *(FP + FN) /Total*, is around 16.5%.

Conclusions

As the number of flights increases, the likelihood of successful landings for the first stage also rises. Success rates tend to improve with increasing payload; however, there is no clear correlation between payload mass and success rates.

From 2013 to 2020, the launch success rate increased by approximately 80%. Among the launch sites, ‘KSC LC-39A’ boasts the highest launch success rate, while ‘CCAFS SLC-40’ has the lowest.

In terms of orbits, ES-L1, GEO, HEO, and SSO exhibit the highest launch success rates, while the GTO orbit has the lowest. Launch sites are strategically located away from urban areas and are positioned closer to coastlines, railroads, and highways for better accessibility.

The best-performing machine learning classification model is the Decision Tree, achieving an accuracy of about 87.5%. When evaluated on the test data, all models recorded an accuracy score of approximately 83%. To further optimize the models and identify a potentially better fit, additional data may be required.

Thank you!