IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vik Thakur

January 19th, 2022

Outline

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

Executive Summary

The objective of this study is to examine SpaceX Falcon 9 data gathered from diverse channels and utilize Machine Learning algorithms to forecast the probability of successful first stage landings. This analysis aims to offer insights to other space agencies, aiding them in making informed decisions regarding competitive bidding against SpaceX..

- Summary of methodologies
  - The following methodologies and techniques were employed to gather and analyze data, develop and assess machine learning models, and forecast outcomes:
  - Data acquisition via API integration and web scraping techniques
  - Data preprocessing and manipulation through data wrangling procedures
  - Exploratory data analysis utilizing SQL queries and visualization tools
  - Creation of an interactive map using Folium to assess the proximity of launch sites
  - Development of a dashboard employing Plotly Dash for interactive analysis of launch records
  - Implementation of a predictive model to forecast the successful landing of the first stage of Falcon 9 rockets.Summary of all results
  - This report will share results in various formats such as:
    - Data analysis results
    - Data visuals, interactive dashboards
    - Predictive model analysis results

Introduction

- Project background and context
  - With the recent achievements in private space exploration, the space industry is increasingly becoming popular and within reach of the general public. However, the cost of launching remains a significant hurdle for new players to join the competition.
  - SpaceX stands out due to its ability to reuse the first stage of its rockets, giving it a crucial edge over its rivals. While other companies spend approximately $165 million per launch, SpaceX only incurs around $62 million by reusing its first stage for future missions. This cost-efficient approach grants SpaceX a distinct advantage in the market.
- Problems you want to find answers
  - Assessing the likelihood of a successful landing for the first stage of SpaceX's Falcon 9 rocket.
  - Exploring how various factors such as launch site, payload weight, booster model, and others influence the outcome of the landing.
  - Investigating potential connections between launch sites and the rates of successful landings.

Section 1

Methodology

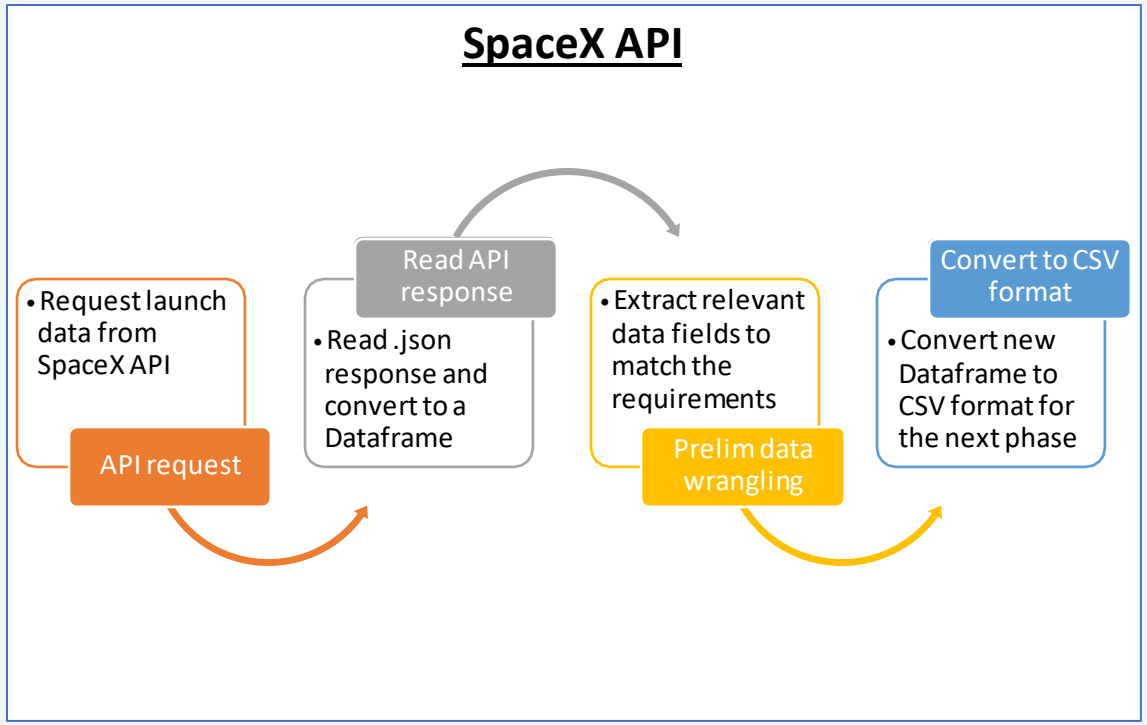
Methodology

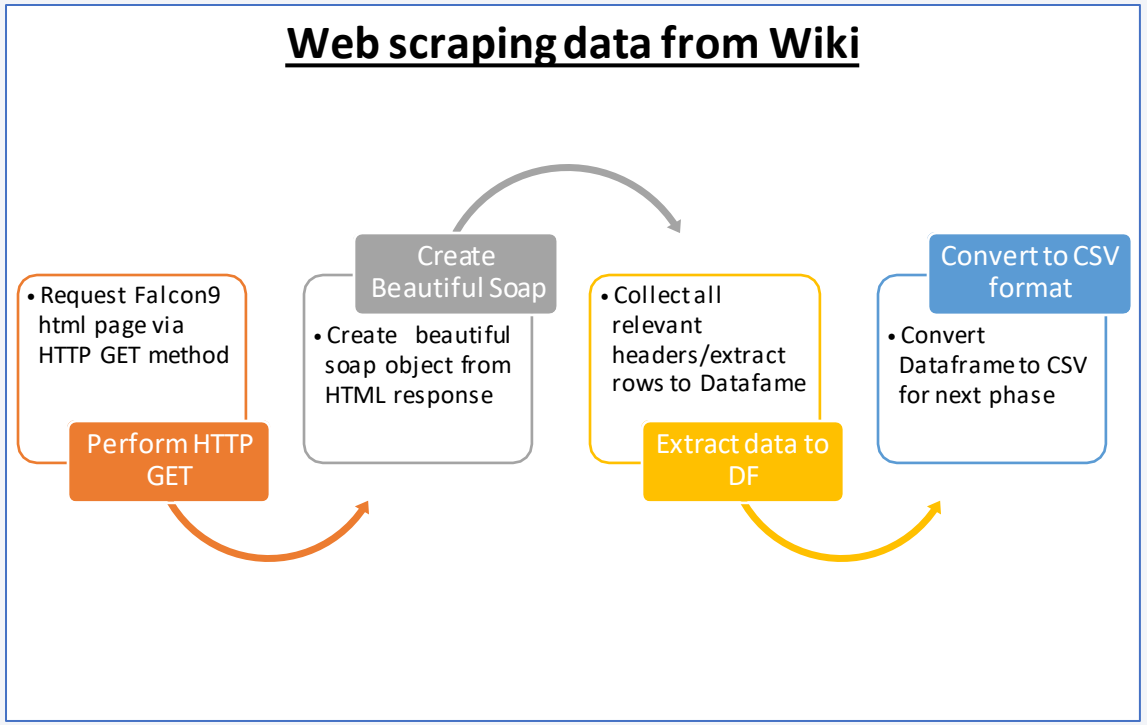
Executive Summary

- Data collection methodology:
  - SpaceX API
  - Web scrap Falcon 9 and Falcon Heavy launch records from Wikipedia (link)
- Perform data wrangling
  - 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 and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- 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

- Data collection is the process of gathering data from available sources. This data can be structured, unstructured, or semi-structured. For this project, data was collected via SpaceX API and Web scrapping Wiki pages for relevant launch 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 = []
```

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

Data Collection - Scraping

1. Perform HTTP GET to request HTML page
2. Create Beautiful Soup 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 Soup 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)
```

GitHub

Data Wrangling

- Conducted Exploratory Data Analysis (EDA) to find patters in data and define labels for training supervised models
- The data set contained various mission outcomes that were converted into Training Labels with 1 meaning the booster successfully landed and 0 meaning booster was unsuccessful in landing. Following landing scenarios were considered to create labels:
  - False Ocean signifies an unsuccessful mission outcome with a landing failure in a specified ocean region.
  - RTLS denotes a successful landing on a ground pad.
  - False RTLS indicates an unsuccessful landing attempt on a ground pad.
  - True ASDS signifies a successful landing on a drone ship.
  - False ASDS indicates an unsuccessful landing attempt on a drone ship.

Data Wrangling - cont'd

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

EDA with Data Visualization

- As part of the Exploratory Data Analysis (EDA), 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
2. Bar Chart:
   - Commonly used to compare the values of a variable at a given point in time. Bar charts makes 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

EDA with SQL

- To better understand SpaceX data set, following SQL queries/operations were performed on an IBM DB2 cloud instance:

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 and 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 failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
10. Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

GitHub

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

Build a Dashboard with Plotly Dash

- Built a Plotly Dash web application to perform 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.
  1. Added a Launch Site Drop-down Input component to the dashboard to provide an ability to filter Dashboard visual by all launch sites or a particular launch site
  2. Added a Pie Chart to the Dashboard to show total success launches when 'All Sites' is selected and show success and failed counts when a particular site is selected
  3. Added a Payload range slider to the Dashboard to easily select different payload ranges to identify visual patterns
  4. Added a Scatter chart 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

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

GitHub

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

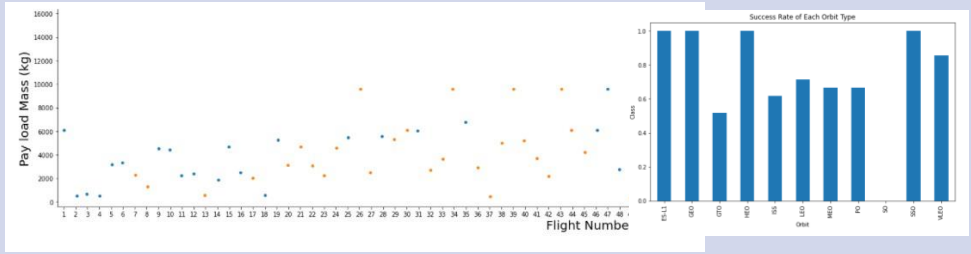
GitHub

Results

Following sections and slides explain results for:

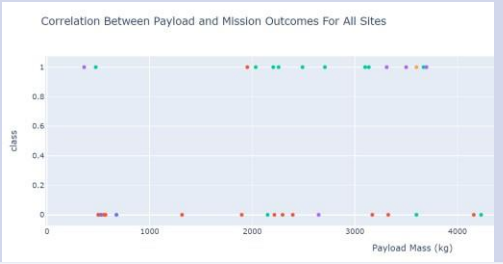
Exploratory data analysis results

- Samples:



Interactive analytics demo in screenshots

- Samples



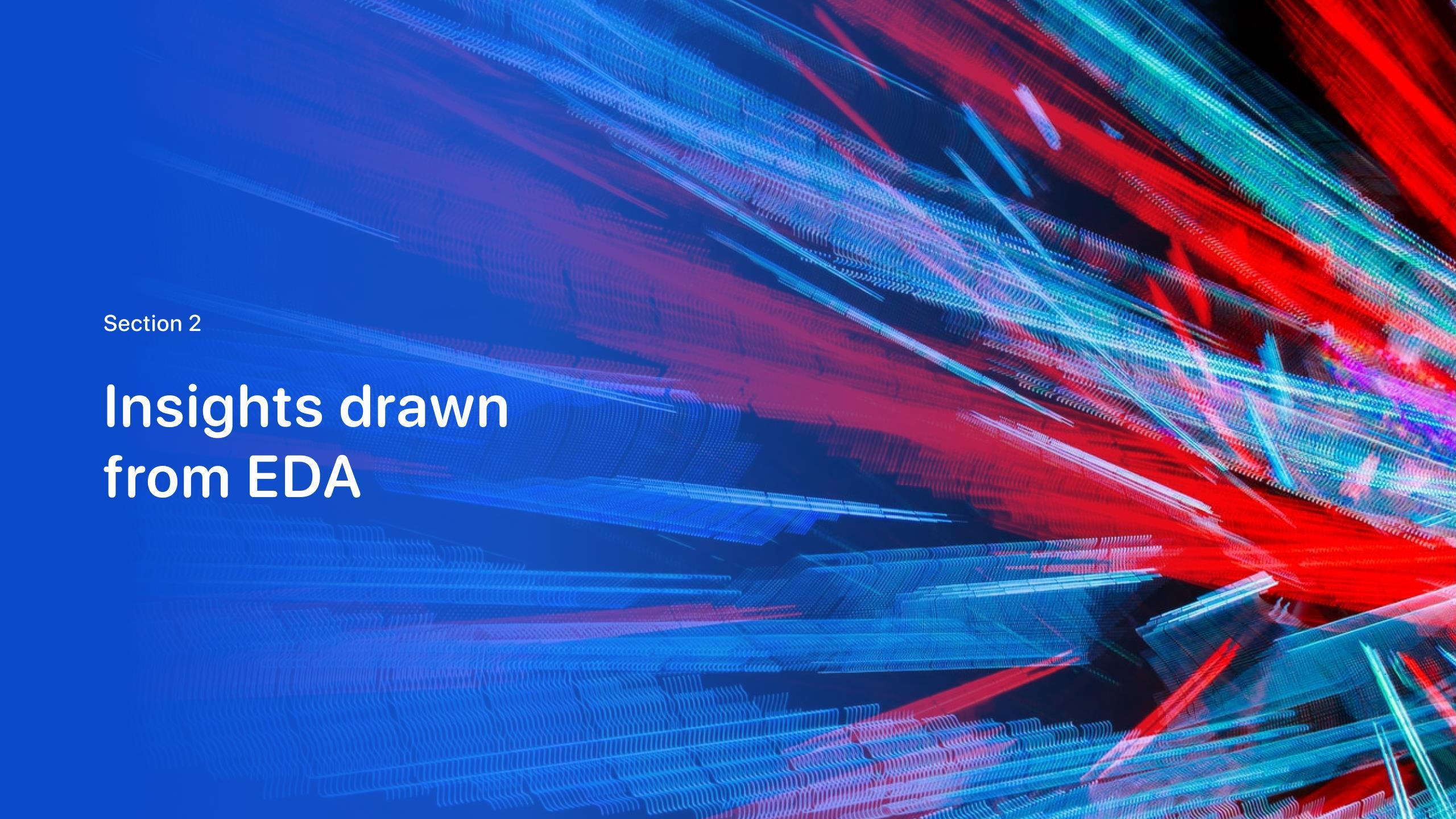
Predictive analysis results

- Samples

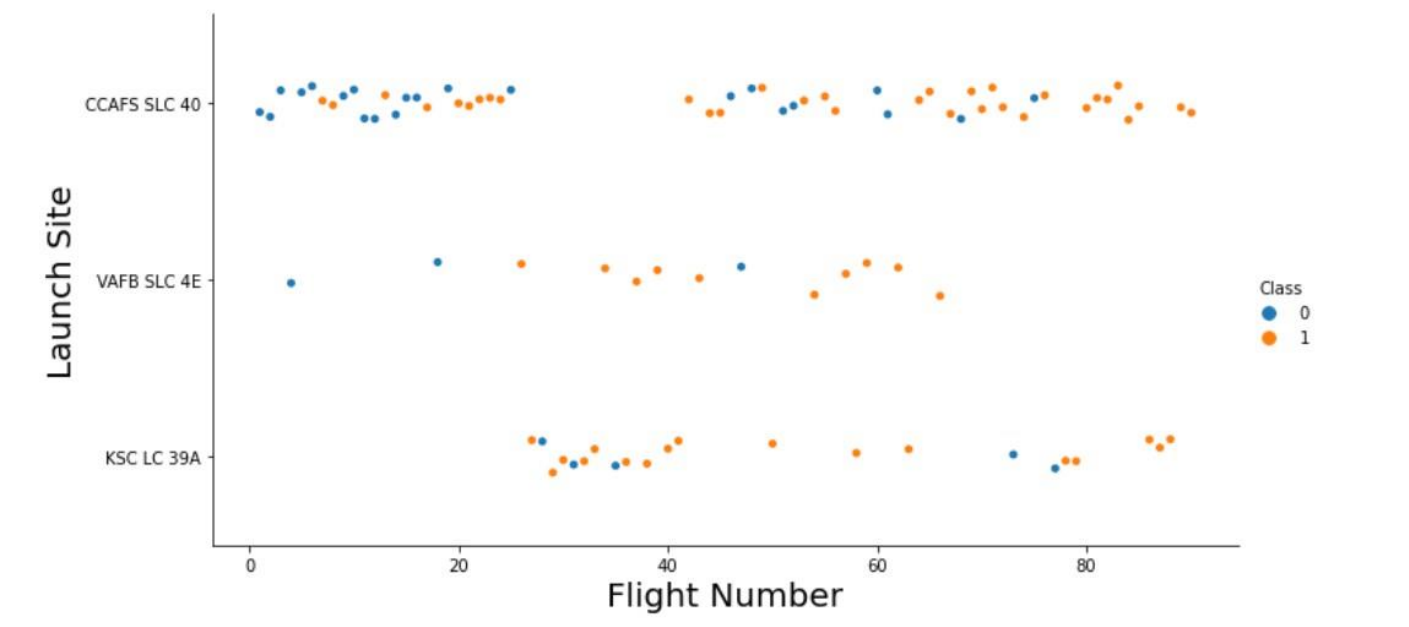
	Algo Type	Accuracy Score
2	Decision Tree	0.903571
3	KNN	0.848214
1	SVM	0.848214
0	Logistic Regression	0.846429

Section 2

Insights drawn from EDA

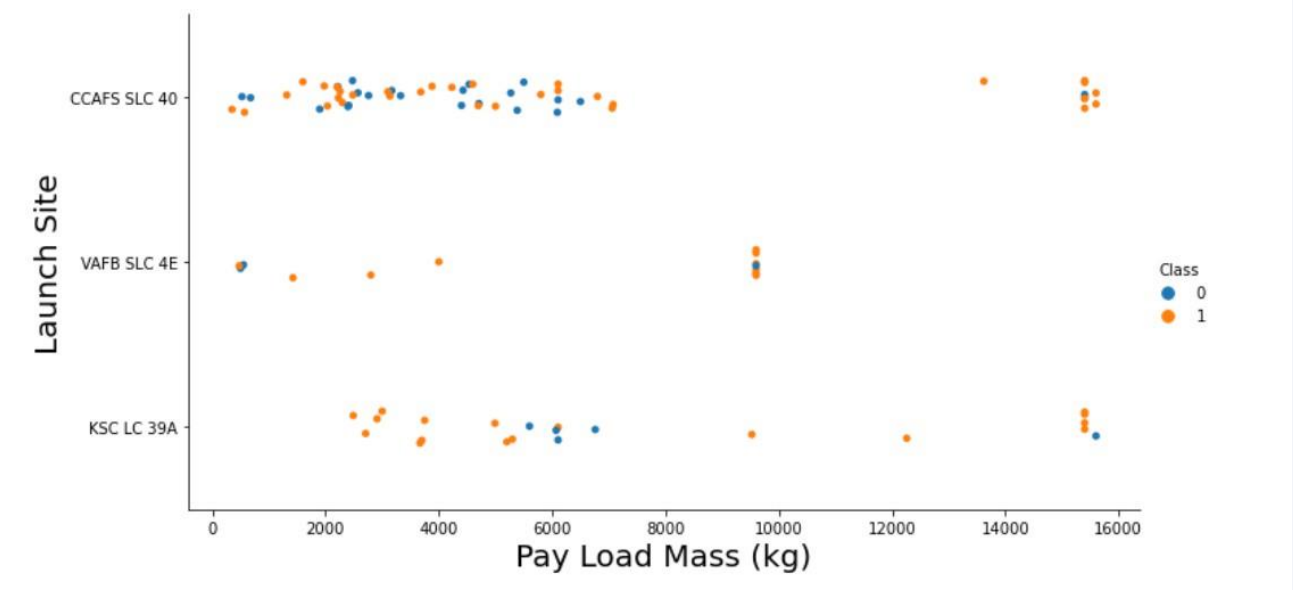


Flight Number vs. Launch Site



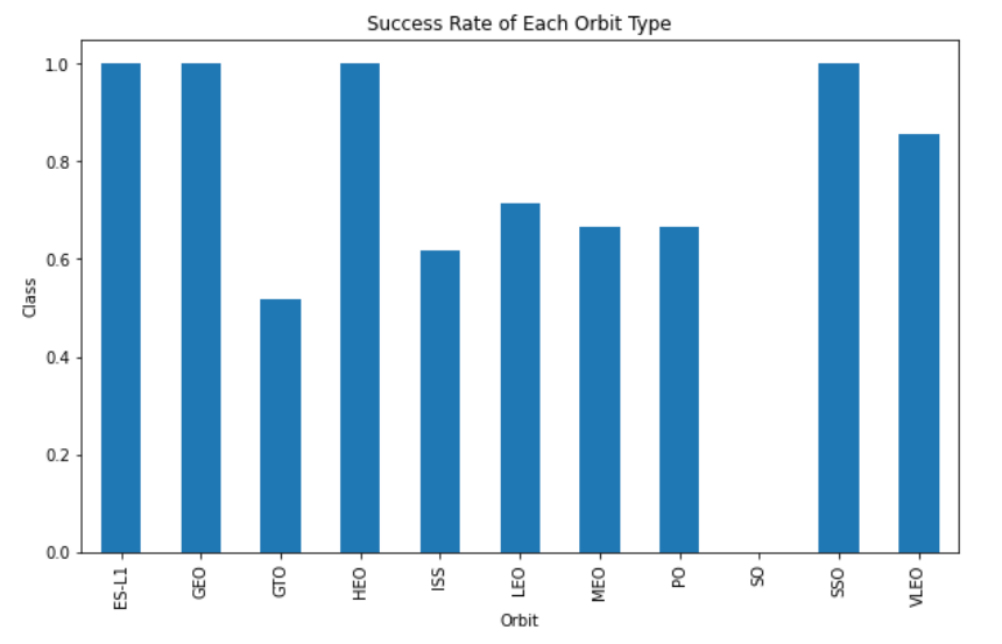
- Success rates (Class=1) tend to increase as the number of flights increases.
- Specifically for the launch site 'KSC LC 39A', it typically takes approximately 25 launches before achieving the first successful launch.

Payload vs. Launch Site



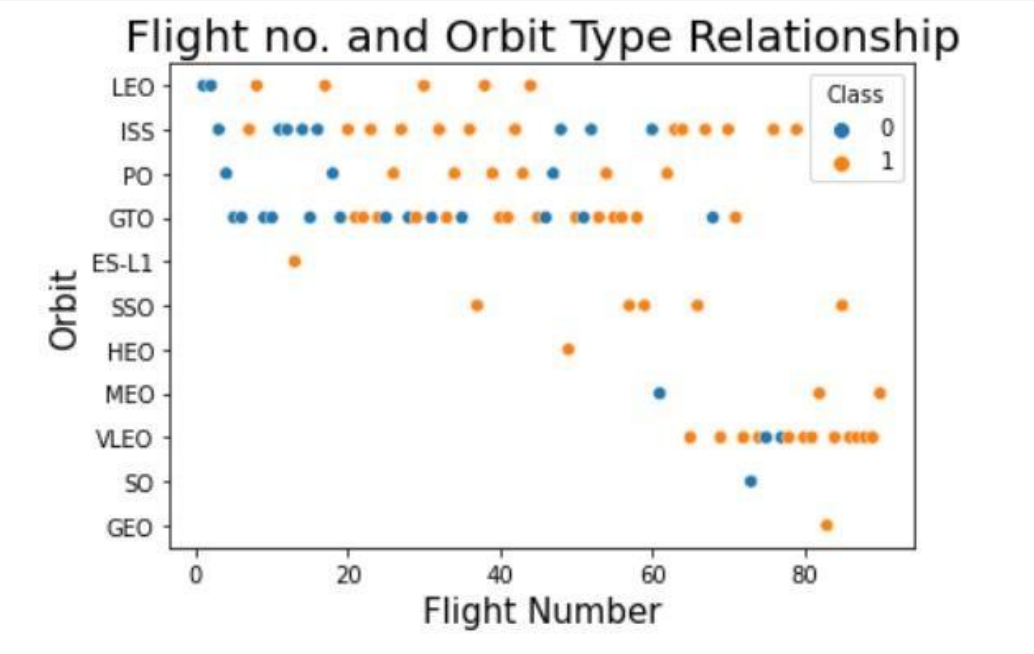
- For the launch site 'VAFB SLC 4E', no rockets have been launched for payloads greater than 10,000 kg.
- Additionally, the percentage of successful launches (Class=1) tends to increase for the launch site 'VAFB SLC 4E' as the payload mass increases.
- However, there is no apparent correlation or discernible pattern between the launch site and payload mass.

Success Rate vs. Orbit Type



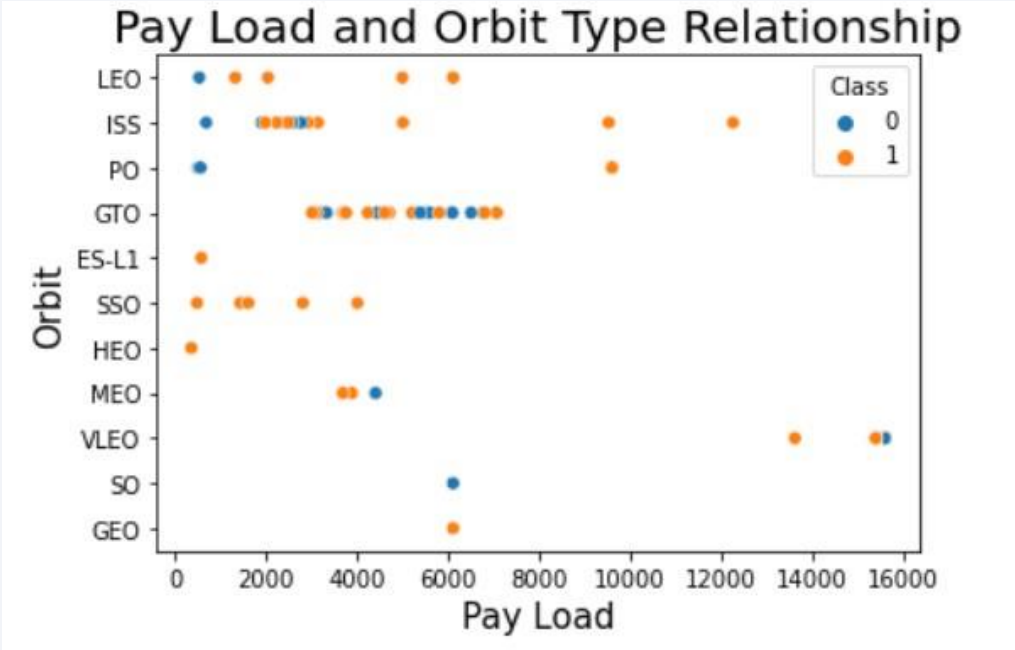
- Orbits ES-LI, GEO, HEO, and SSO exhibit the highest success rates.
- On the contrary, the GTO orbit demonstrates the lowest success rate.

Flight Number vs. Orbit Type



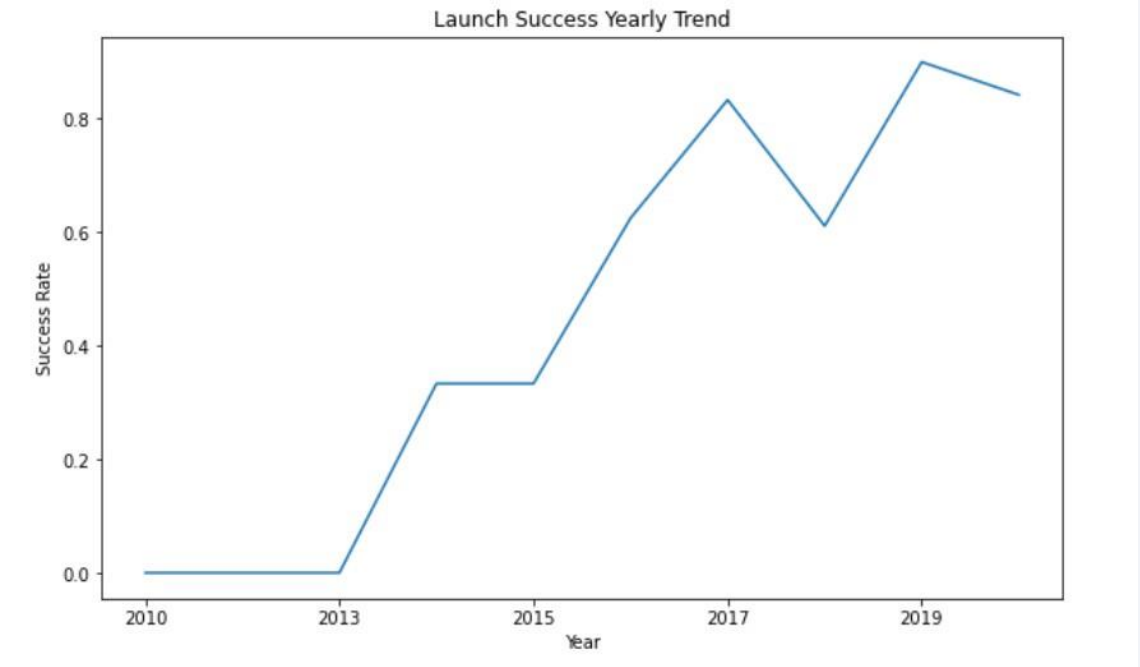
- For orbit VLEO, the first successful landing (class=1) doesn't occur until after 60 flights or more.
- In general, for most orbits such as LEO, ISS, PO, SSO, and MEO, successful landing rates seem to increase with the number of flights.
- However, there is no discernible relationship between the number of flights and the orbit for GTO.

Payload vs. Orbit Type



- Successful landing rates (Class=1) seem to rise with payload for orbits such as LEO, ISS, PO, and SSO.
- However, for the GEO orbit, there isn't a clear pattern between payload and the success or failure of landing.

Launch Success Yearly Trend



- The success rate (Class=1) witnessed an increase of approximately 80% between 2013 and 2020.
- Success rates remained consistent between 2010 and 2013, as well as between 2014 and 2015.
- However, success rates experienced a decline between 2017 and 2018, and again between 2019 and 2020.

All Launch Site Names

- Query:

```
select distinct Launch_Site from spacextbl
```

- Description:
  - The keyword 'DISTINCT' retrieves only unique values from the queried column (Launch_Site). In this case, there are 4 unique launch sites.

- Result:

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

Launch Site Names Begin with 'CCA'

- Query:

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

- Description:
  - Using the keyword 'LIKE' and the format 'CCA%', the query retrieves records where the 'Launch_Site' column starts with "CCA". The 'LIMIT 5' clause restricts the number of returned records to 5.
  - Result:

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 'sum' keyword adds the values in the column 'PAYLOAD_MASS_KG' and returns the total payload mass 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 of the payload mass in the 'PAYLOAD_MASS_KG' column 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 expression 'min(Date)' selects the earliest or oldest date from the 'Date' column where the first successful landing on a ground pad occurred.
  - The WHERE clause specifies the criteria to retrieve dates 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 where the payload mass is greater than 4000 but less than 6000, and the landing outcome is a success on a drone ship.
  - The 'and' operator in the WHERE clause retrieves booster versions where both conditions specified in the WHERE clause are simultaneously met.

- 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 in a column into groups. In this instance, the number of mission outcomes categorized by types of outcomes are grouped into the column labeled 'counts'.

- 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 employs the 'max' keyword on the payload mass column to retrieve the maximum payload mass.
  - The main query then selects booster versions and their corresponding payload masses where the payload mass equals the maximum value of 15600.
- 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 displays landing outcomes, booster versions, and launch sites where the landing outcome is classified as failed on a drone ship, specifically in the year 2015.
  - The 'and' operator in the WHERE clause retrieves booster versions where both conditions specified in the WHERE clause are met.
  - 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' that experienced failed landing outcomes on a drone ship in the year 2015.

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 data in the 'Landing Outcome' column into groups.
  - The 'between' and 'and' keywords retrieve data that falls between the dates 2010-06-04 and 2017-03-20.
  - The 'order by' keyword sorts the counts column in descending order.
  - The outcome of the query yields 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

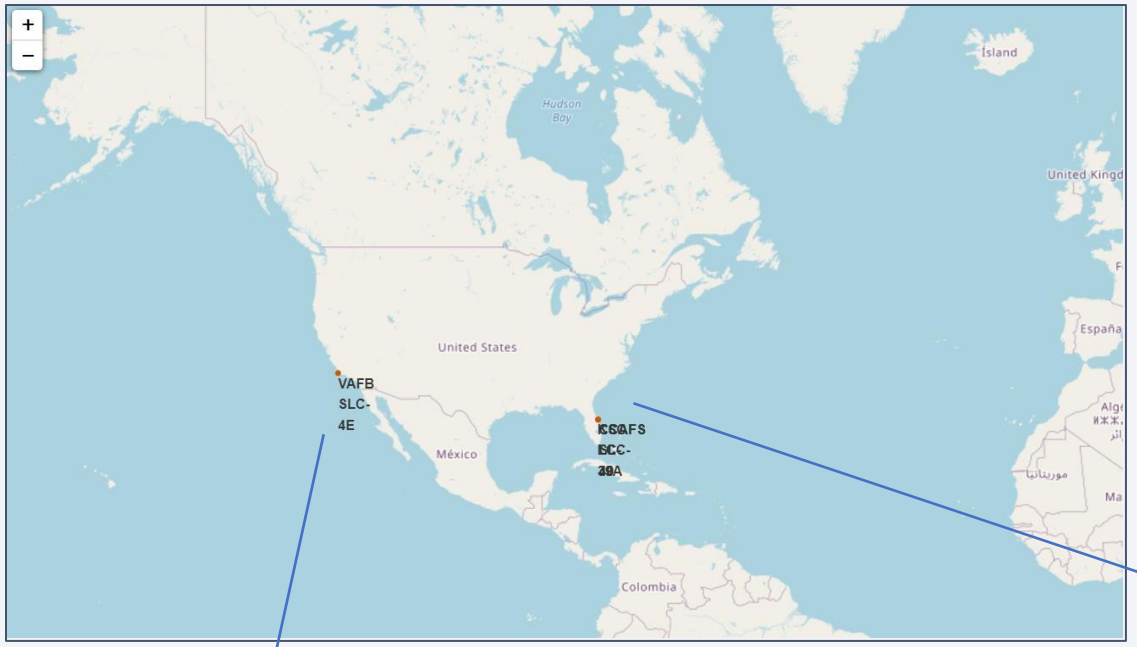


Fig 1 – Global Map

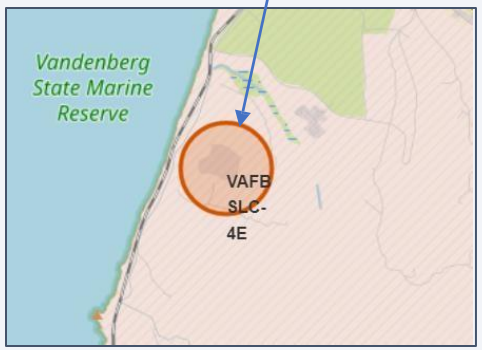


Fig 2 – Zoom 1

Figure 1 on the left showcases a global map featuring Falcon 9 launch sites situated within the United States, specifically in California and Florida. Each launch site is denoted by a circle with a label, accompanied by a popup indicating the site's location and name. Furthermore, it is noticeable that all launch sites are positioned near the coastline.

Figures 2 and 3 provide zoomed-in views of four specific launch sites:

VAFB SLC-4E (California)
CCAFS LC-40 (Florida)
KSC LC-39A (Florida)
CCAFS SLC-40 (Florida)

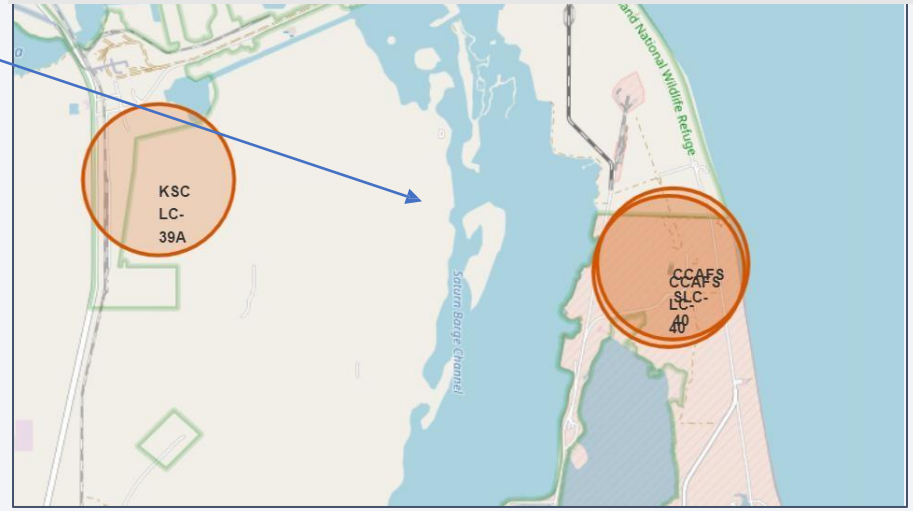


Fig 3 – Zoom 2

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

Fig 1 – US map with all Launch Sites

- Figure 1 represents a map of the United States showcasing all the Launch Sites. Each site is marked with numbers indicating the total count of successful and failed launches.
- Figures 2, 3, 4, and 5 provide closer views of each individual site, displaying markers indicating success (green) and failure (red) for each launch.
- Upon examining the maps of each site, it is evident that the KSC LC-39A Launch Site boasts the highest number of successful launches.

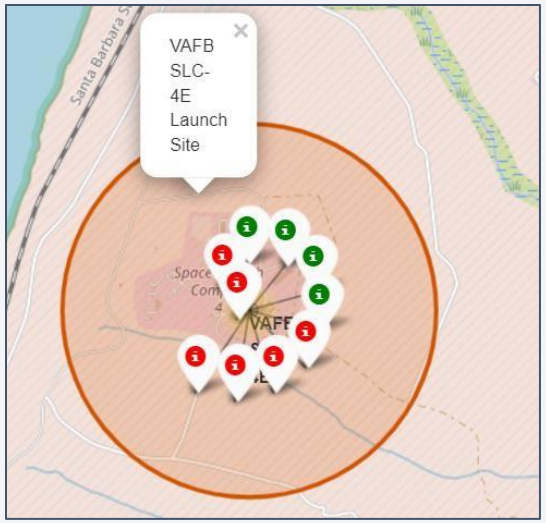


Fig 2 – VAFB Launch Site with success/failed markers

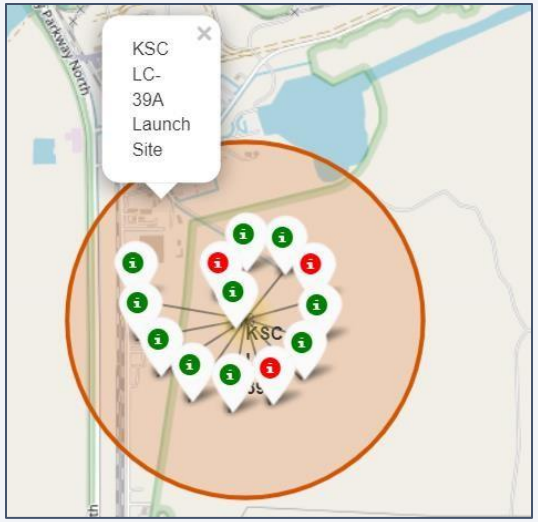


Fig 3 – KSC LC-39A success/failed markers

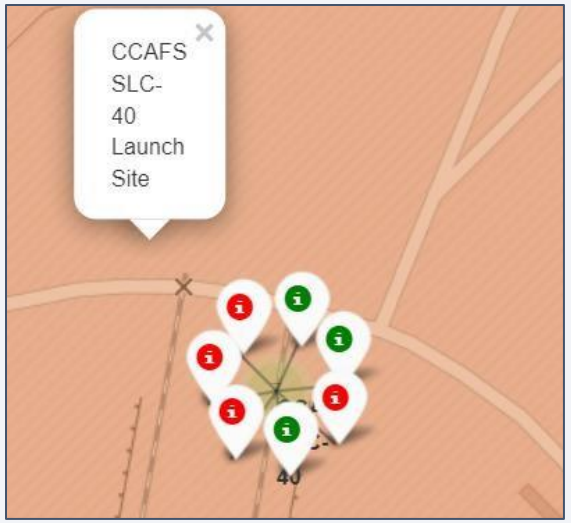


Fig 4 – CCAFS SLC-40 success/failed markers

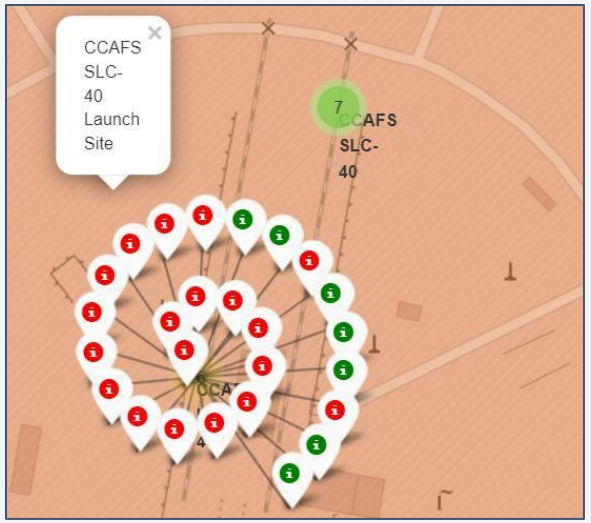


Fig 5 – CCAFS SLC-40 success/failed markers

SpaceX Falcon9 – Launch Site to proximity Distance Map

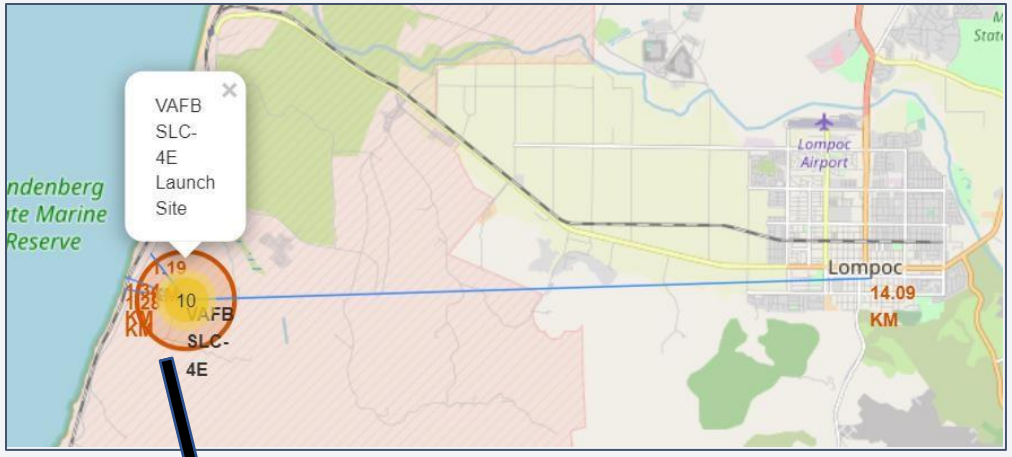


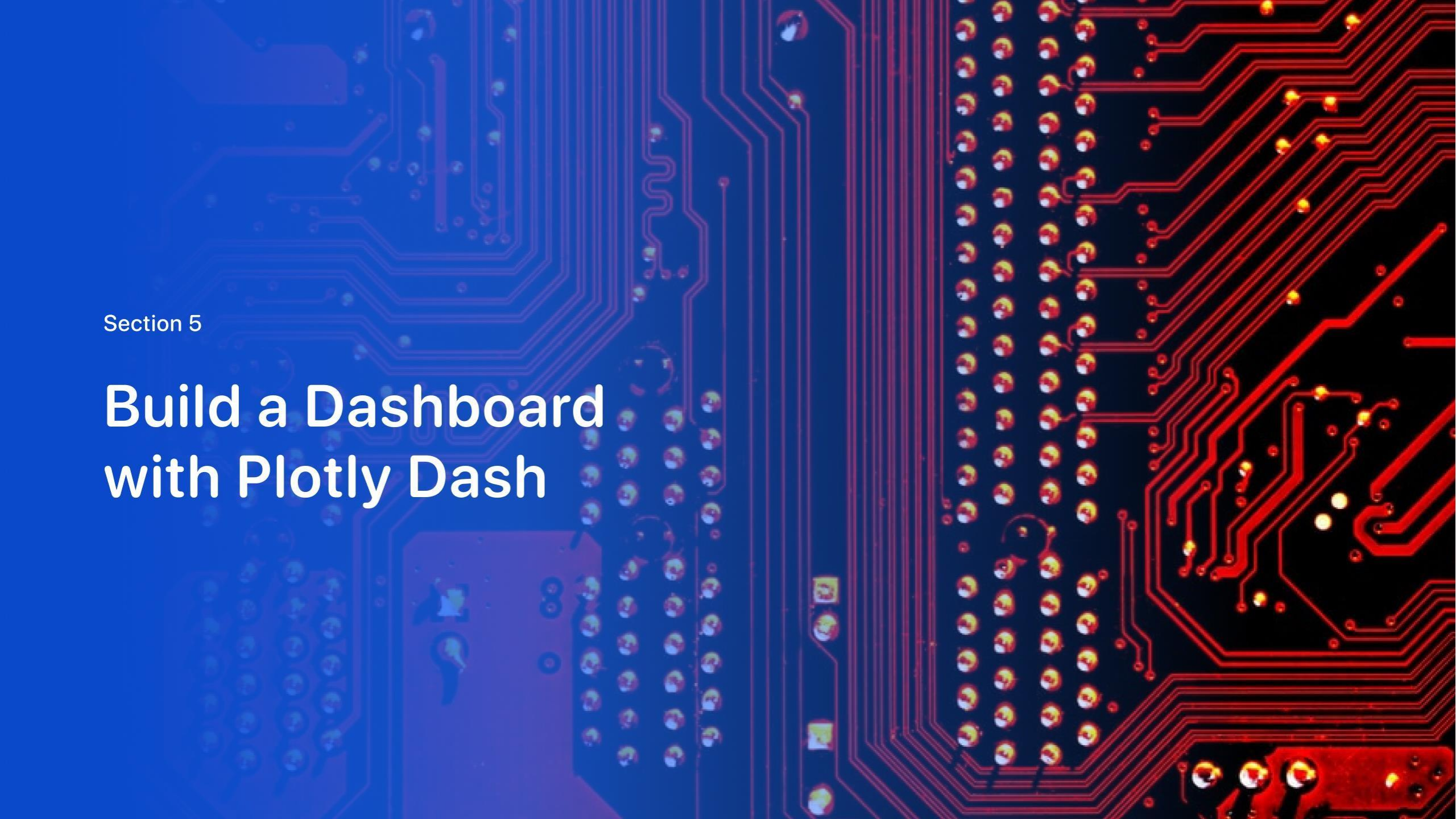
Fig 1 – Proximity site map for VAFB SLC-4E

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

Figure 1 illustrates all the nearby sites plotted on the map for Launch Site VAFB SLC-4E. Notably, City Lompoc is situated farther from the Launch Site compared to other proximities like the coastline, railroad, and highway. The map also indicates a marker denoting the distance of the city from the Launch Site, which measures approximately 14.09 kilometers.

Figure 2 offers a closer view of other proximities such as the coastline, railroad, and highway, along with their respective distances from the Launch Site.

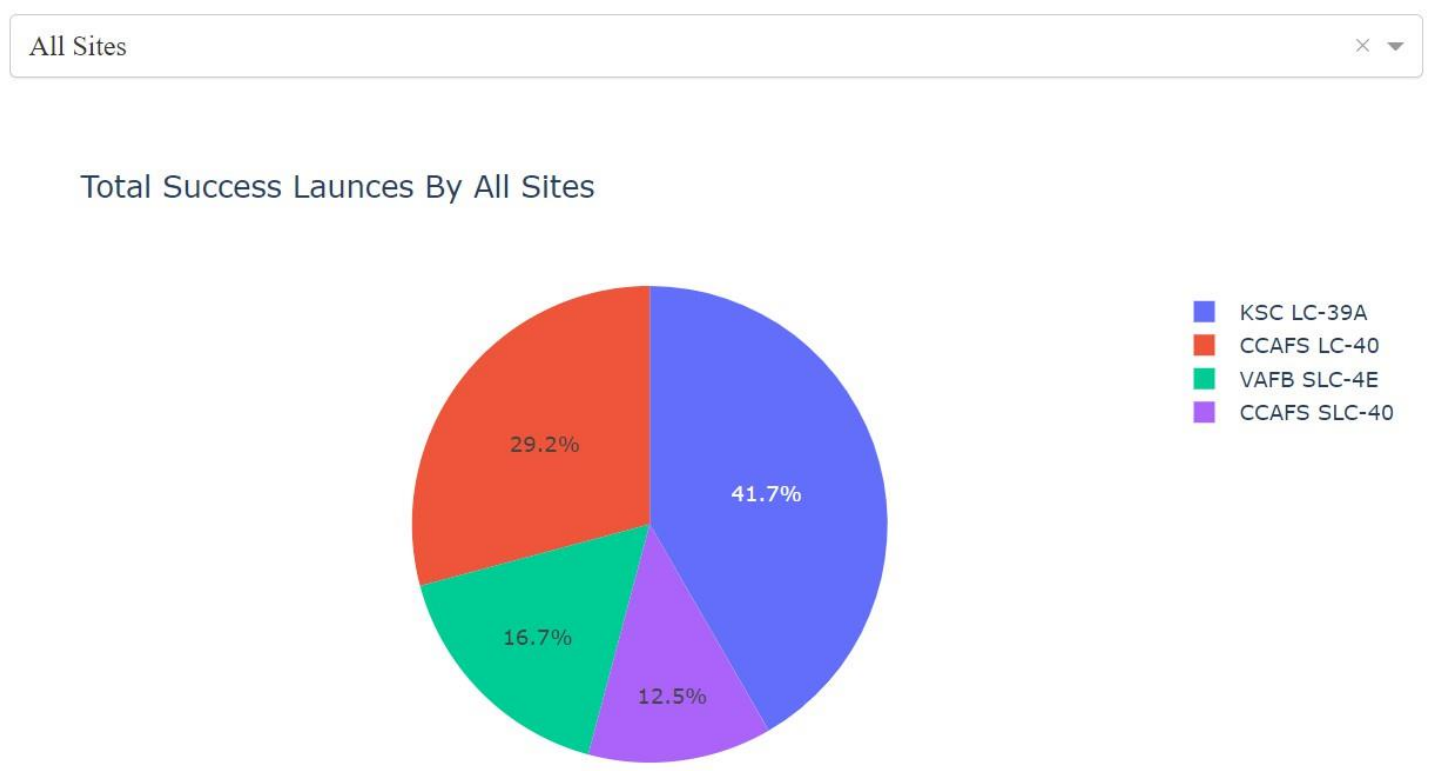
In general, cities are strategically positioned away from Launch Sites to minimize potential impacts on the general public and infrastructure in the event of any accidents. Launch Sites are strategically situated near the coastline, railroad, and highways to facilitate easy access to essential resources.



Section 5

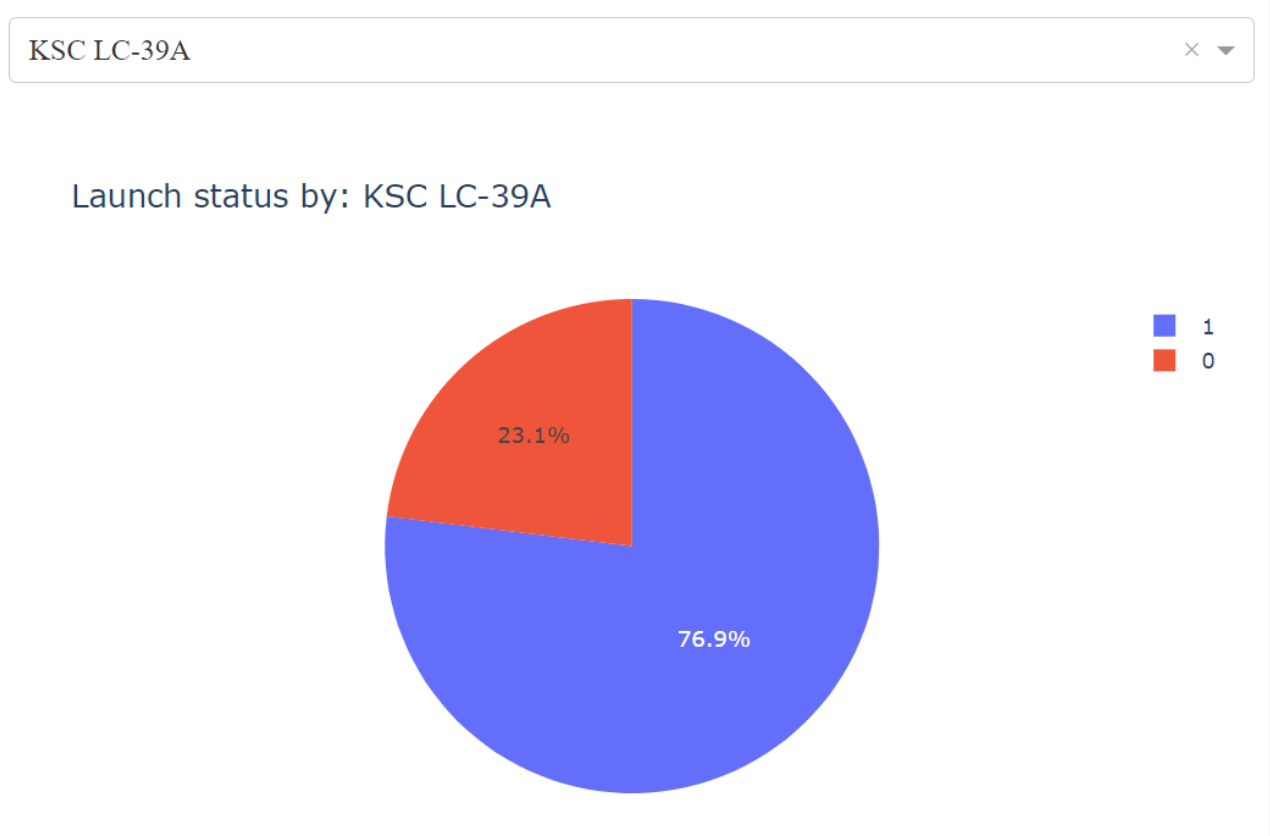
Build a Dashboard with Plotly Dash

Launch Success Counts For All Sites



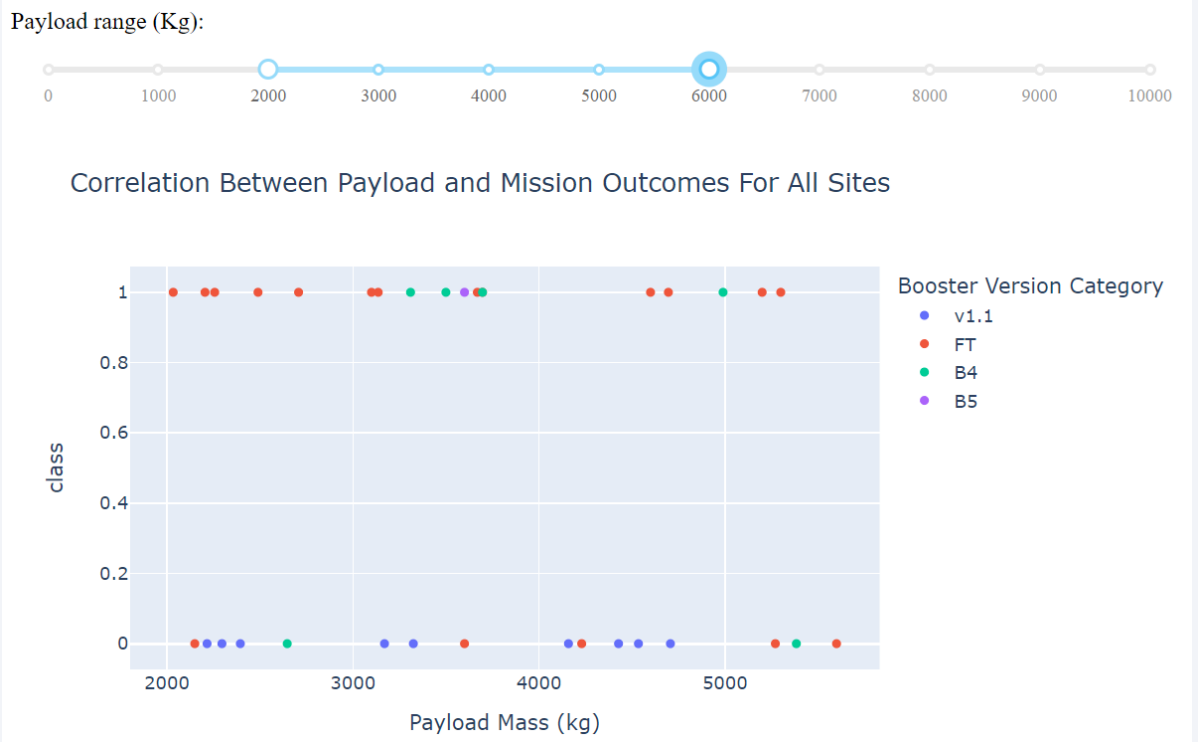
- Launch Site 'KSC LC-39A' has the highest launch success rate
- Launch Site 'CCAFS SLC-40' has the lowest launch success rate

Launch Site with Highest Launch Success Ratio



- KSC LC-39A Launch Site has the highest launch success rate and count
- Launch success rate is 76.9%
- Launch success failure rate is 23.1%

Payload vs. Launch Outcome Scatter Plot for All Sites

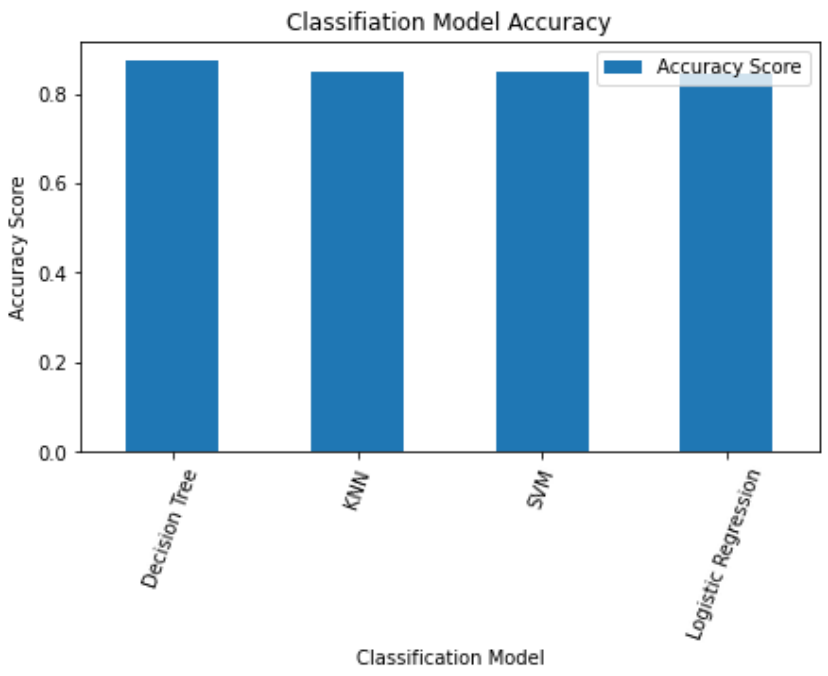


- Most successful launches are in the payload range from 2000 to about 5500
- Booster version category ‘FT’ has the most successful launches
- Only booster with a success launch when payload is greater than 6k is ‘B4’

Section 6

Predictive Analysis (Classification)

Classification Accuracy

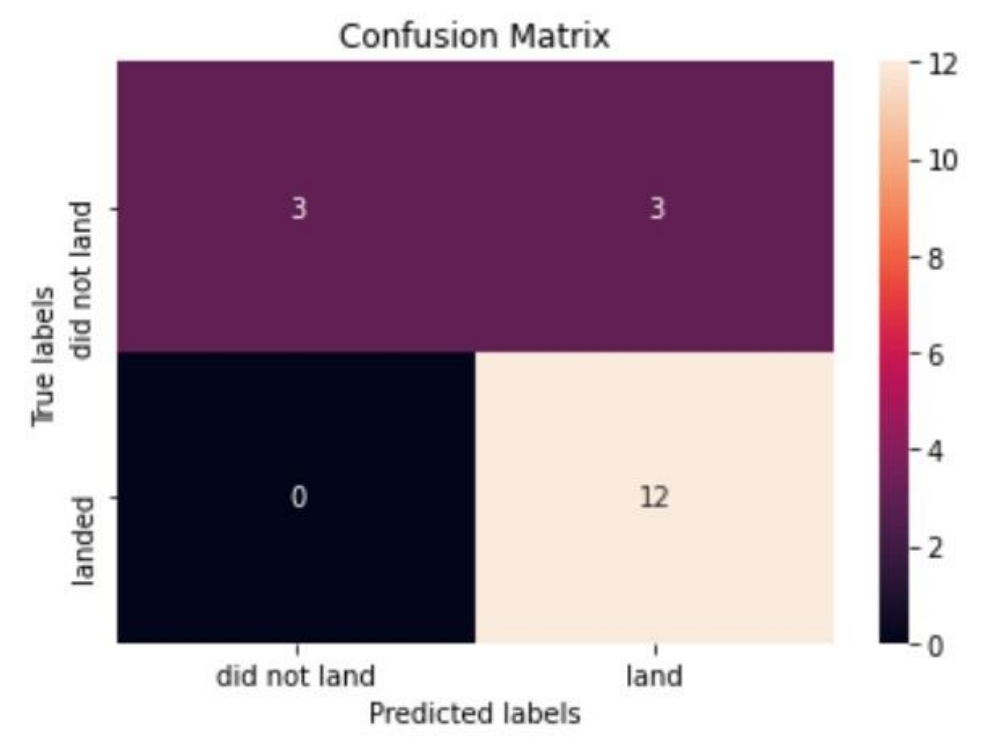


	Algo Type	Accuracy Score	Test Data Accuracy Score
2	Decision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333
1	SVM	0.848214	0.833333
0	Logistic Regression	0.846429	0.833333

- Considering the accuracy scores and as depicted in the bar chart, the Decision Tree algorithm exhibits the highest classification score, standing at a value of .8750. Additionally, the accuracy score on the test data remains consistent across all classification algorithms, indicating a value of .8333.

- Given the marginal differences in accuracy scores among classification algorithms and the uniformity in test scores, it suggests the necessity for a more expansive dataset to refine and optimize the models further

Confusion Matrix



- The confusion matrix exhibits uniformity across all models (LR, SVM, Decision Tree, KNN).
- According to the confusion matrix, the classifier generated 18 predictions:
- 12 instances were predicted as Yes for landing, and indeed, they did land successfully (True positive).
- 3 scenarios (located at the top left of the matrix) were predicted as No for landing, and correspondingly, they did not land (True negative).
- 3 scenarios (located at the top right of the matrix) were predicted as Yes for landing, but unfortunately, they did not land successfully (False positive).
- Overall, the classifier's accuracy stands at approximately 83% ((TP + TN) / Total), with a misclassification or error rate of about 16.5% ((FP + FN) / Total).

Conclusions

- With an increasing number of flights, the likelihood of successful first stage landings tends to rise.
- There seems to be a tendency for success rates to increase with higher payload masses, although no definitive correlation between payload mass and success rates is evident.
- The launch success rate experienced an approximately 80% increase from 2013 to 2020.
- Launch Site 'KSC LC-39A' boasts the highest launch success rate, while Launch Site 'CCAFS SLC-40' exhibits the lowest launch success rate.
- Orbits ES-L1, GEO, HEO, and SSO demonstrate the highest launch success rates, whereas orbit GTO displays the lowest.
- Launch sites are strategically situated away from urban areas and closer to coastlines, railroads, and highways.
- The most effective Machine Learning Classification Model identified is the Decision Tree, achieving an accuracy of around 87.5%. Upon testing the models with the test data, the accuracy score averaged about 83% for all models. Further data might be necessary to fine-tune the models and potentially enhance their performance.

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!