Winning Space Race with Data Science

IBM Data Science Professional Certificate
Capstone project

Nuttasit Phasukdee
October 29, 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)
- Summary of all results
  - Exploratory data analysis results
  - Geospatial analytics
  - Interactive dashboard
  - Predictive analysis of classification models

Introduction

- Project background and context
  - SpaceX has gained worldwide attention for a series of historic milestones. It is the only private company ever to return a spacecraft from low-earth orbit, which it first accomplished in December 2010.
  - SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars wheras other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage.
  - Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.
- Problems you want to find answers
  - This project's objective is predicting the success rate in landing of the first stage.

Section 1

Methodology

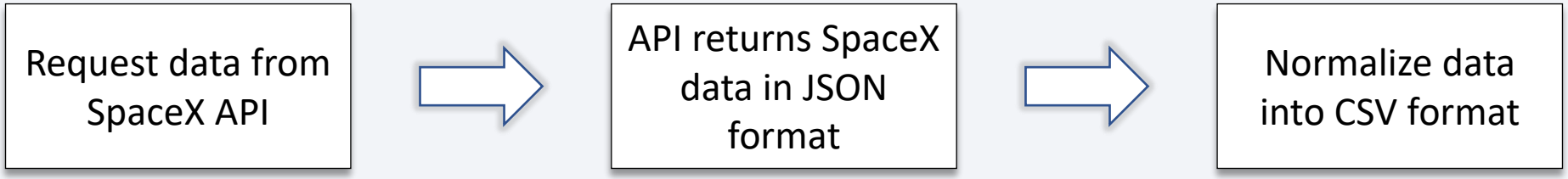
Methodology

Executive Summary

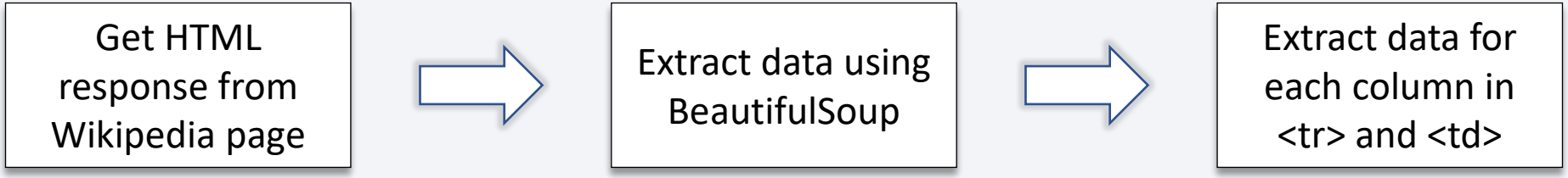
1. Data collection methodology
    - Making GET requests to the SpaceX API
    - Web Scraping launch records on a Wiki page
2. Perform data Wrangling
    - Converting landing outcomes into labels for training labels
3. Perform exploratory data analysis (EDA) using visualization and SQL
4. Perform interactive visual analytics using Folium and Plotly Dash
5. Perform predictive analysis using classification models
    - Find best Hyperparameter for SVM, Classification Trees and Logistic Regression

Data Collection

- The data collection process includes a combination of API requests from SpaceX API and web scraping data from a table in Wikipedia page of SpaceX, Falcon 9 and Falcon Heavy Launches Records.
  - SpaceX API Data Columns: FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude



  - Web scraping Data Columns: Flight No., Launch site, Payload, Payload mass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time



Data Collection – SpaceX API

1. Requesting data from SpaceX API
2. Converting Response to JSON format and create a DataFrame
3. Using helper functions to extract only interested information to lists of strings
4. Combining the lists into a dictionary and create a DataFrame from the dictionary
5. Filtering DataFrame and exporting to a CSV file

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-stora
response = requests.get(static_json_url)
```

```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url)
data = pd.json_normalize(response.json())
```

```
# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)
```

```
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}
```

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

```
# Save a CSV file
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```

GitHub

Data Collection - Scraping

1. Requesting response from HTML page as an HTTP response and creating a BeautifulSoup object from it
2. Finding all tables
3. Extracting column names from <th> element
4. Create an empty dictionary with keys
5. Filling up the launch_dict with launch records from <tr> and <td> element
6. Creating a DataFrame and exporting it to a CSV file

```
response = requests.get(static_url)
soup = BeautifulSoup(response.text, 'html5lib')
```

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

```
column_names = []
for row in first_launch_table.find_all("th"):
    col_name = extract_column_from_header(row)
    if col_name != None and len(col_name) > 0:
        column_names.append(col_name)
```

```
launch_dict= dict.fromkeys(column_names)

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

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

```
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
  # get table row
  for rows in table.find_all("tr"):
      #check to see if first table heading is as number corresponding to launch a number
      if rows.th:
          if rows.th.string:
              flight_number=rows.th.string.strip()
              flag=flight_number.isdigit()
      else:
          flag=false
      #get table element
      row=rows.find_all('td')
      #if it is number save cells in a dictonary
      if flag:
          extracted_row += 1

          # Flight Number value
          # TODO: Append the flight_number into launch_dict with key `Flight No.`
          #print(flight_number)
          launch_dict['Flight No.'].append(flight_number)

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

```
df=pd.DataFrame(launch_dict)
df.to_csv('spacex_web_scraped.csv', index=False)
```

GitHub

Data Wrangling

1. Performing Exploratory Data Analysis (EDA) by implementing **value_counts()** method on 3 columns (LaunchSite, Orbit, Outcome)
2. From Outcome column, there are several cases of landing outcomes
   - True Ocean: successfully landed to a specific region of the ocean while
   - False Ocean: unsuccessfully landed to a specific region of the ocean.
   - True RTLS: successfully landed to a ground pad
   - False RTLS: unsuccessfully landed to a ground pad.
   - True ASDS: successfully landed to a drone ship
   - False ASDS: unsuccessfully landed to a drone ship.
   - None ASDS and None None: a failure to land.
3. Converting these landing outcomes into training labels
   - If the first stage landed successfully, the label value is 1.
   - If the first stage did not land successfully, the label value is 0.
4. Calculating the average success rates of landing outcomes

```
df["LaunchSite"].value_counts()
df["Orbit"].value_counts()
df["Outcome"].value_counts()
```

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

True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

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

```
df["Class"].mean()

0.6666666666666666
```

GitHub

EDA with Data Visualization

- Scatter chart: Plotting two columns or featues as x and y and the outcome in Class column as color, so we can see the effect of two features on the outcome
  - Flight Number vs. Launch Site
  - Payload vs. Launch Site
  - Flight Number vs. Orbit Type
  - Payload vs. Orbit Type
- Bar chart: By grouping by a column or feature, we can observe the relationship of the feature and the outcome
  - Orbit Types vs. Class (Success rate)
- Line chart: We can observe the trend from the plot
  - Year vs. Class (Success rate)

GitHub

EDA with SQL

- Loading the dataset into the corresponding table in a Db2 database, and executing SQL queries to answer following questions:
  - Displaying the names of the unique launch sites in the space mission
  - Displaying 5 records where launch sites begin with the string 'CCA'
  - Displaying the total payload mass carried by boosters launched by NASA (CRS)
  - Displaying average payload mass carried by booster version F9 v1.1
  - Listing the date when the first successful landing outcome in ground pad was achieved
  - Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - Listing the total number of successful and failure mission outcomes
  - Listing the names of the booster_versions which have carried the maximum payload mass
  - Listing the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - Ranking 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

- Objects created and added to a folium map
  - Markers and Circles showing all launch sites' locations on a map
  - Markers' colors that indicate the success and fails launches for each site on the map
  - Lines showing the distances between a launch site to its proximities
- By adding these map objects, we can answer the 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

- The dashboard application contains a pie chart and scatter chart
  - Pie chart
    - For showing total success launches by sites
    - This pie chart can be selected to indicate a successful landing distribution across all launch sites or to indicate the success rate of individual launch sites
  - Scatter chart
    - For showing the relationship between Outcomes and Payload mass (Kg) by different boosters
    - Has 2 inputs: All sites/individual site & Payload mass on a slider between 0 and 10000 kg
    - This chart help how success depending on the launch point, payload mass and booster version.

GitHub

Predictive Analysis (Classification)

- Perform EDA and determine training labels
  - Create a list of targets from Class columns
  - Standardize the features
  - Split data into training data and test data
- Finding best Hyperparameters for 4 models (SVM, Classification Trees, Logistic Regression and K-Nearest Neighbor)
- Calculate each model's accuracy
- Comparing the accuracies of 4 models

GitHub

```
Y = data["Class"].to_numpy()
transform = preprocessing.StandardScaler()
X = transform.fit_transform(X)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

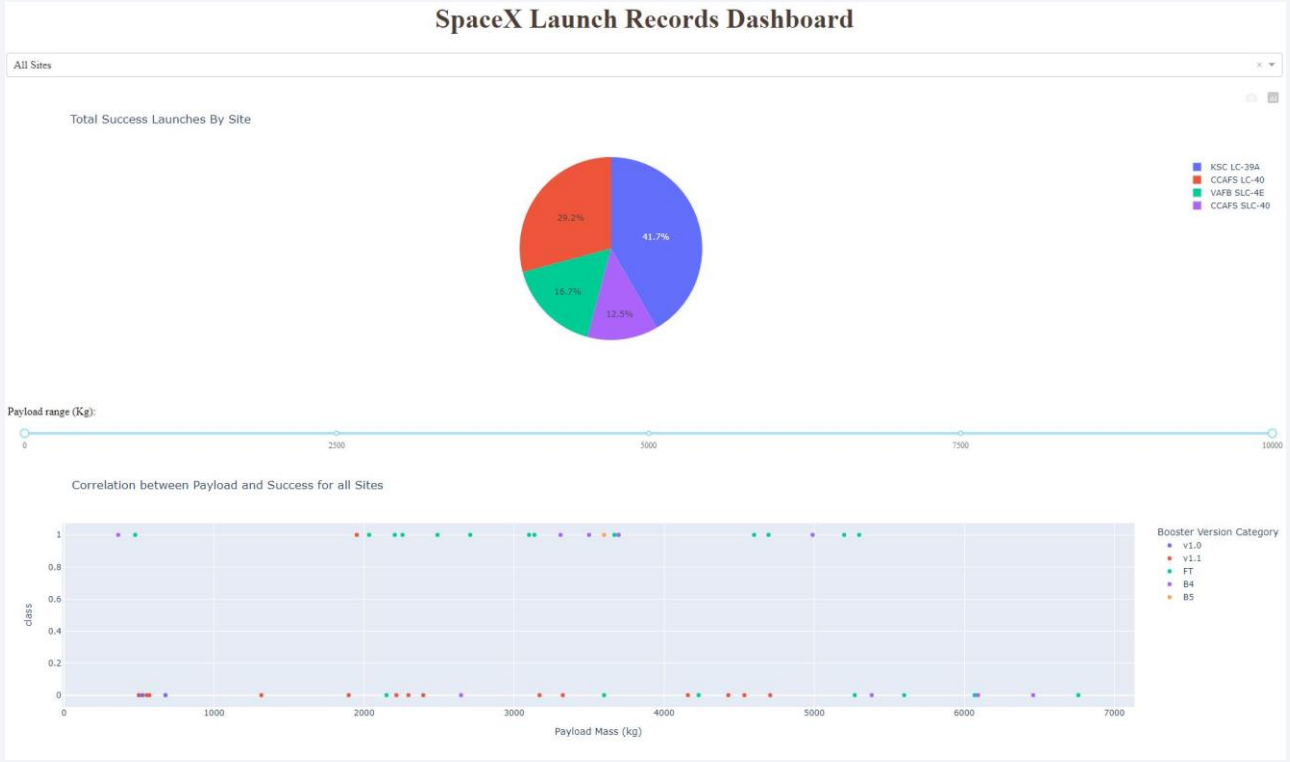
```
parameters ={'C':[0.01,0.1,1],
             'penalty':['l2'],
             'solver':['lbfgs']}


lr=LogisticRegression()

logreg_cv = GridSearchCV(estimator=lr,
                         param_grid=parameters,
                         cv=10,
                         verbose=1)
logreg_cv.fit(X_train, Y_train)
```

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

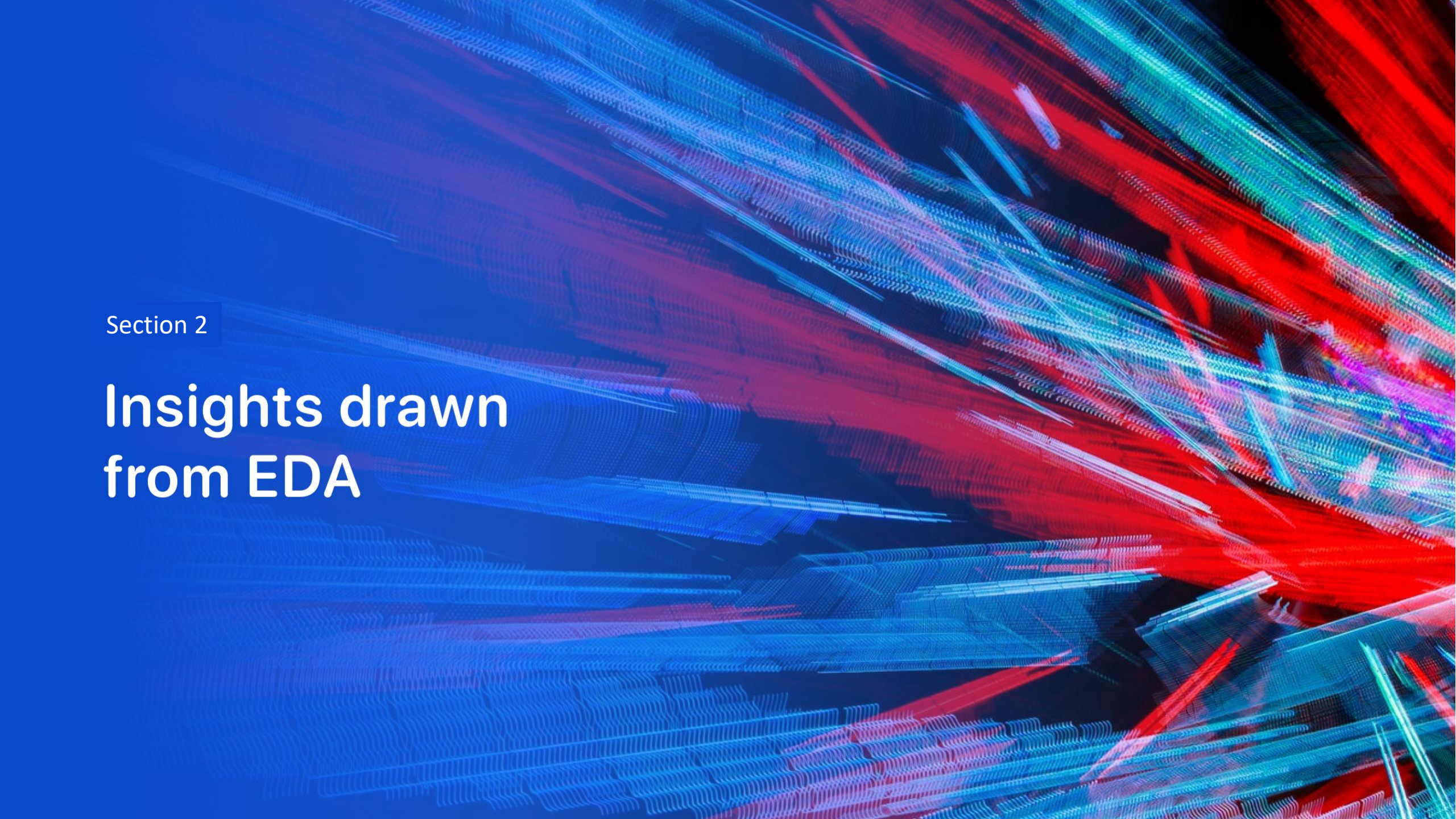
```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8472222222222222
```

```
score = logreg_cv.score(X_test, Y_test)
```

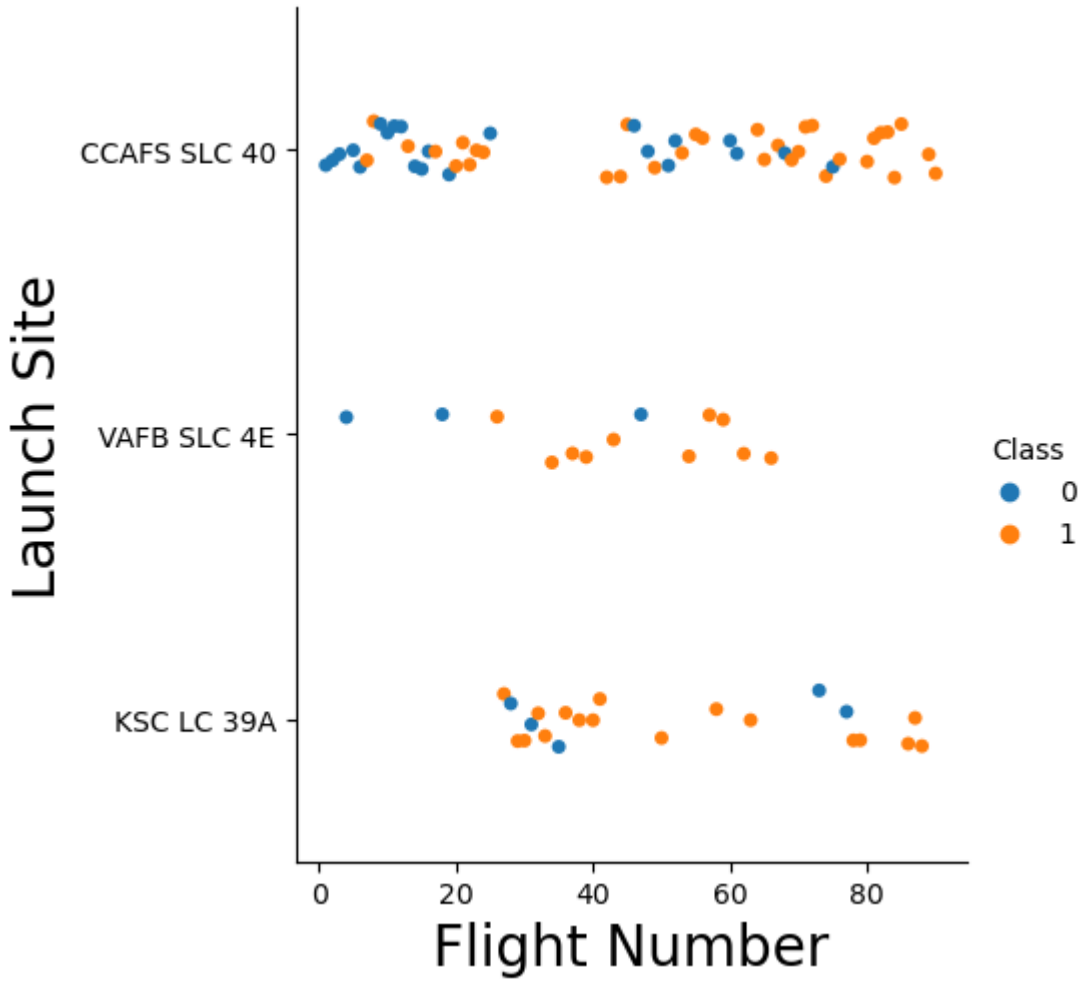
Results



- The left image is a preview of the Dashboard with Plotly Dash.
- The results of EDA with visualization, EDA with SQL, Interactive Map with Folium, and Interactive Dashboard will be shown in the next slides.
- The accuracies of the 4 models in predicting the landing outcome in test set are the same which is 83.3333%

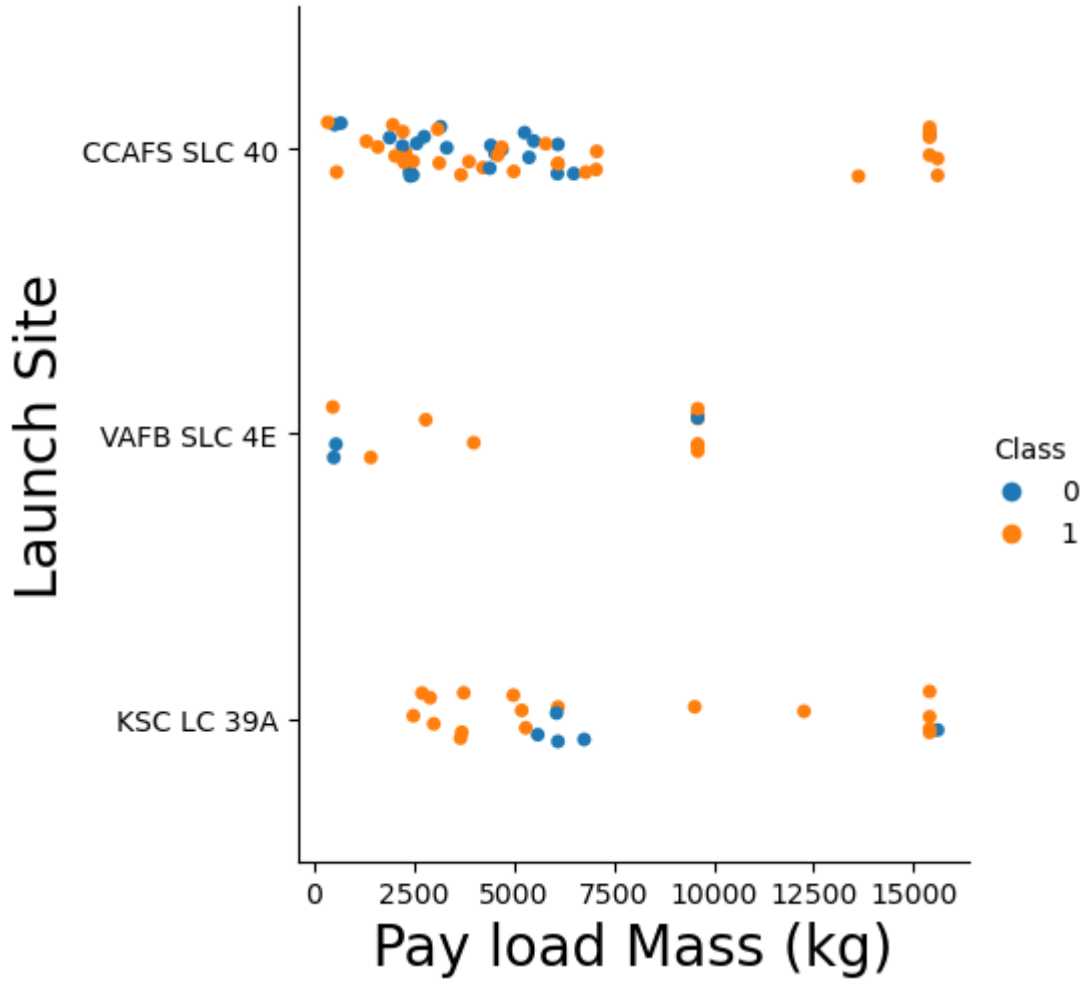

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



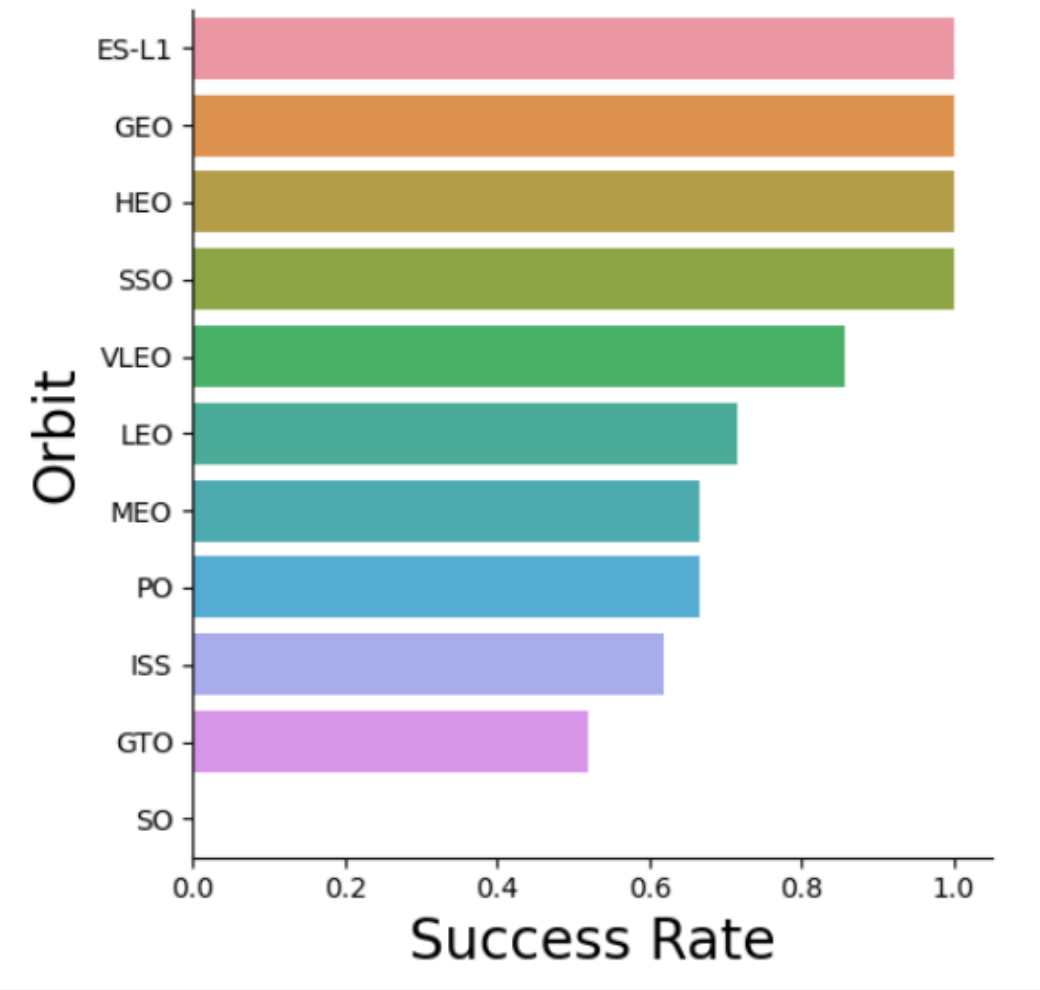
- Class 0 (Blue) represents unsuccessful launch, and Class 1 (Orange) represents successful launch.
- This figure shows that the success rate of each launch site increased as the number of flights of each launch site increased
- Overall, the success rate has significant increased since the 20th flight.

Payload vs. Launch Site



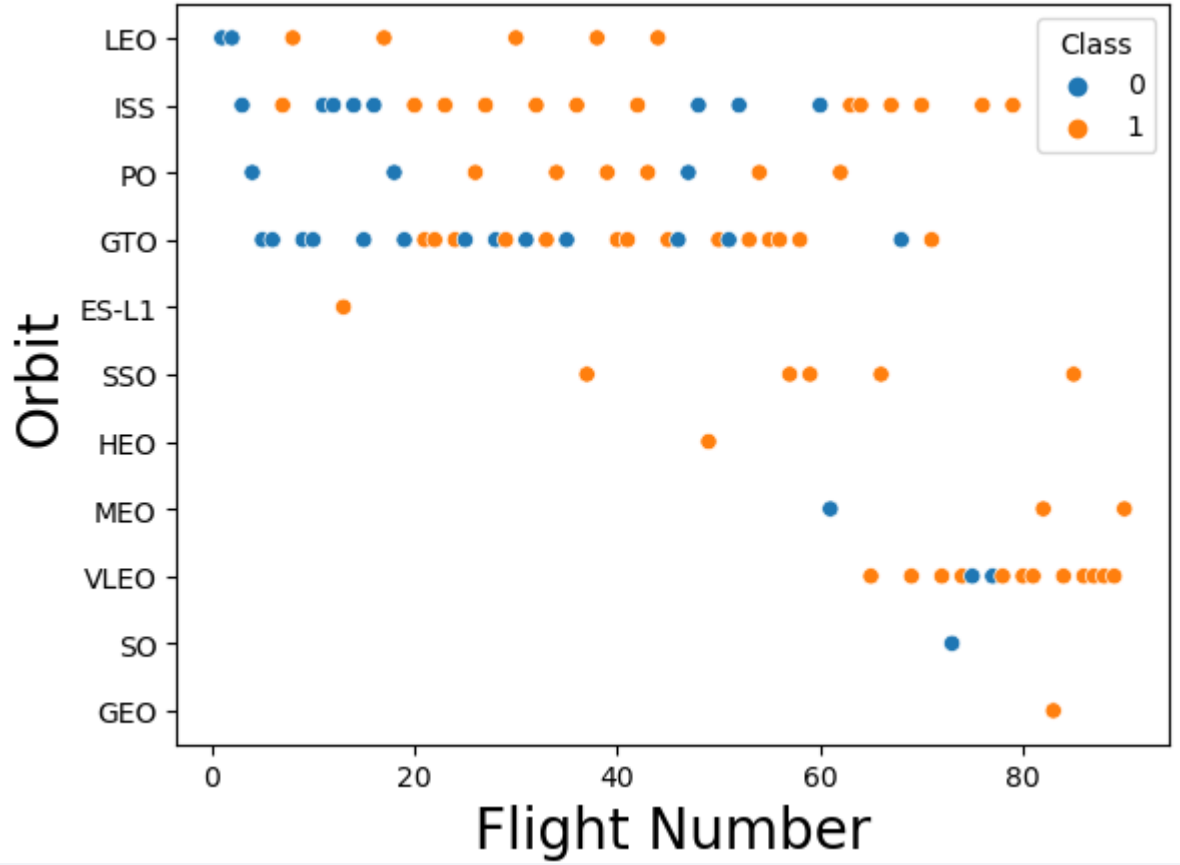
- Class 0 (Blue) represents unsuccessful launch, and Class 1 (Orange) represents successful launch.
- There is no payload's mass more than 10,000 kg launching from the VAFB-SLC launch site.
- Overall, there is no clear pattern or a breakthrough payload mass that separate the successful or failed outcome.

Success Rate vs. Orbit Type



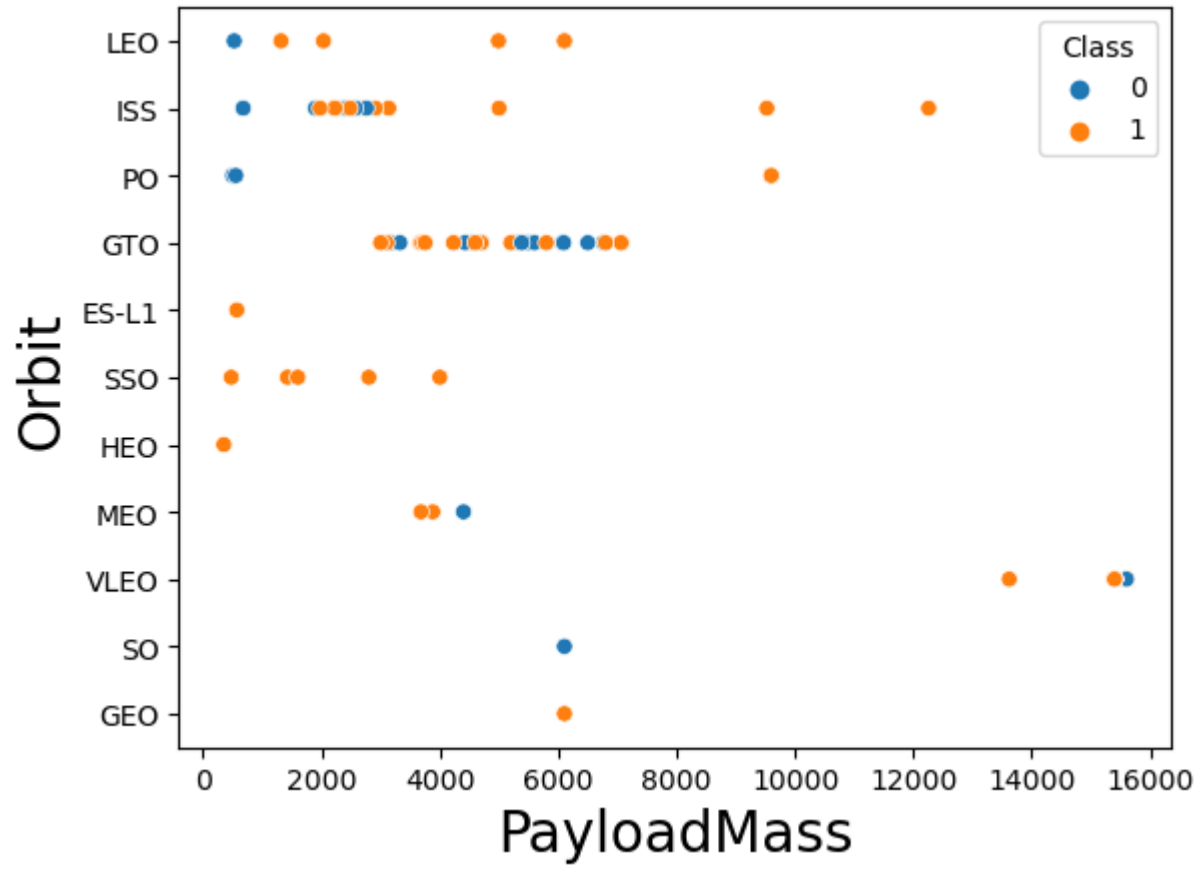
- Orbit types (**ES-L1, GEO, HEO and SSO**) have the highest success rate at 100%.
- The orbit type with lowest success rate (0%) is **SO**.
- The rest of orbit types have the success rate between 50% and 85%.

Flight Number vs. Orbit Type



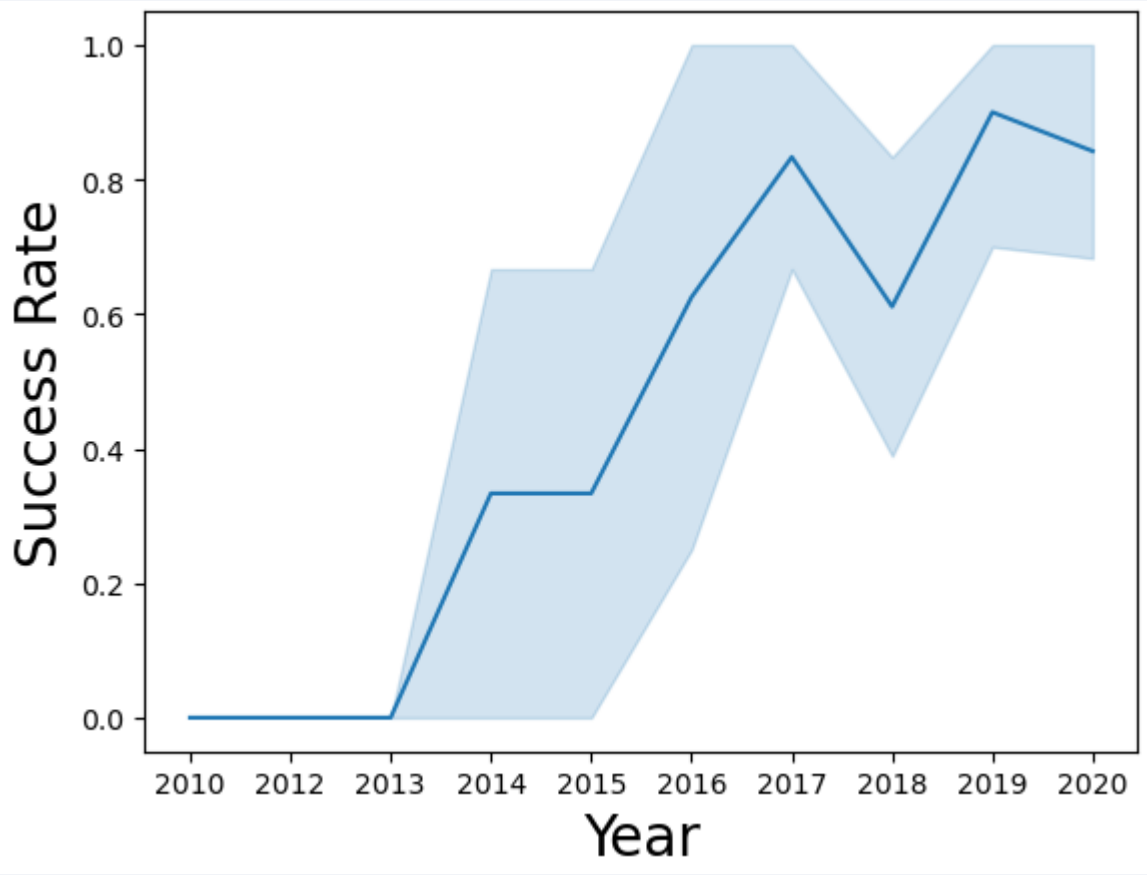
- Class 0 (Blue) represents unsuccessful launch, and Class 1 (Orange) represents successful launch.
- The successes in most of orbit types except GTO become more successful as the number of flight increased (Positive correlation)
- After the 40th flight, most of launch outcomes of every orbit type are successful landing.

Payload vs. Orbit Type



- Class 0 (Blue) represents unsuccessful launch, and Class 1 (Orange) represents successful launch.
- There are only three orbit types that launches with payload more than 10,000 kg (**ISS**, **PO**, **VLEO**).
- **SSO** orbit type has a high success landing outcome with payload's mass less than 5,000 kg

Launch Success Yearly Trend



- The average success rate is significantly increased since 2013.
- Between 2017 and 2019, there is a decrease in a landing outcome's success rate.
- In recent years, the success rate is above 80%.

All Launch Site Names

- SQL

```
%%sql
SELECT DISTINCT LAUNCH_SITE
FROM SPACE_X;
```

- Result

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

- DISTINCT LAUNCH_SITE is used in a SQL quest to return unique launch sites from LAUNCH_SITE column.
- There are 4 unique launch sites which are **CCASF LC-40**, **CCAGS SLC-40**, **KSC LC-39A**, **VAFB SLC-4E**.

Launch Site Names Begin with 'CCA'

- SQL

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

- Result

DATE	time_utc_	booster_version	launch_site	payload	payload_mass_kg_	orbit	customer	mission_outcome	landing_outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

- WHERE clause is used to filter and extract only rows that pass a specified condition.
- LIKE operator extract rows that contain a determined string
- LAUNCH_SITE LIKE ‘CCA%’ tells SQL to find rows that strings in LAUNCH_SITE start with ‘CCA’.
- LIMIT clause is used to limit the number of returned outcomes.

Total Payload Mass

- SQL

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) AS Total
FROM SPACE_X
WHERE CUSTOMER = 'NASA (CRS)';
```

- Result

total
45596

- WHERE clause is used to filter and extract only rows that pass a specified condition.
- SUM function returns the total sum of a numeric column.
- The total sum of payload mass of a customer, NASA (CRS) is **45,596**.

Average Payload Mass by F9 v1.1

- SQL

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) AS Average
FROM SPACE_X
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

- Result

average
2928

- WHERE clause is used to filter and extract only rows that pass a specified condition.
- AVG function returns the average value of a numeric column.
- The average of payload mass carried by booster version, F9 v1.1, is **2,928**.

First Successful Ground Landing Date

- SQL

```
%%sql
SELECT DATE AS FIRST_SUCCESSFUL_GROUND_LANDING
FROM SPACE_X
WHERE LANDING__OUTCOME = 'Success (ground pad)'
ORDER BY DATE ASC, TIME__UTC_ ASC
LIMIT 1
```

- Result

first_successful_ground_landing
2015-12-22

- WHERE clause is used to filter and extract only rows that pass a specified condition.
- ORDER BY keyword is used to sort the result-set in ascending (ASC) or descending (DESC) order
- LIMIT clause is used to limit the number of returned outcomes.
- With the combination of ORDER BY keyword, WHERE and LIMIT clause, we found that the first successful ground landing is **2015-12-22**.

Successful Drone Ship Landing with Payload between 4000 and 6000

- SQL

```
%%sql
SELECT BOOSTER_VERSION
FROM SPACE_X
WHERE LANDING_OUTCOME  = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

- Result

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

- The combination of WHERE clause and AND operator is used to filer extract only rows that pass multiple conditions
- The booster versions that has successful landing outcomes on drone ship are **F9 FT B1022**, **F9 FT B1026**, **F9 FT B1021.2**, **F9 FT B1021.3**.

Total Number of Successful and Failure Mission Outcomes

- SQL

```
%%sql
SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL_NUMBER
FROM SPACE_X
GROUP BY MISSION_OUTCOME;
```

- Result

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

- GROUP BY statement groups rows that have the same values into summary rows and often used with aggregate functions (COUNT, MAX, MIN, SUM, AVG).
- COUNT function returns the number of rows
- From the SQL query, The success rate of landing outcomes is 99%.

Boosters Carried Maximum Payload

- SQL

```
%%sql
SELECT DISTINCT BOOSTER_VERSION
FROM SPACE_X
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_)
                           FROM SPACE_X)
```

- Result

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

- MAX function returns the maximum value of a numeric column.
- In SQL query, it implements sub-query before main-query.
- The sub-query is used to find the maximum value of payload mass.
- All booster versions that have carried the maximum payload mass are a part of **F9 B510xx.x** version.

2015 Launch Records

- SQL

```
%%sql
SELECT PAYLOAD, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACE_X
WHERE LANDING__OUTCOME LIKE '%Failure%' AND YEAR(DATE) = '2015';
```

- Result

payload	booster_version	launch_site
SpaceX CRS-5	F9 v1.1 B1012	CCAFS LC-40
SpaceX CRS-6	F9 v1.1 B1015	CCAFS LC-40

- WHERE clause and LIKE operator with a condition, '%Failure%' is used to extract every row that have a word 'Failure' as a part of string.
- YEAR function returns the year part for a given date, e.g., YEAR(2015-12-22) returns "2015".
- Both failed landing outcomes in drone ship's launch site is **CCAFS LC-40**.

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

- SQL

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

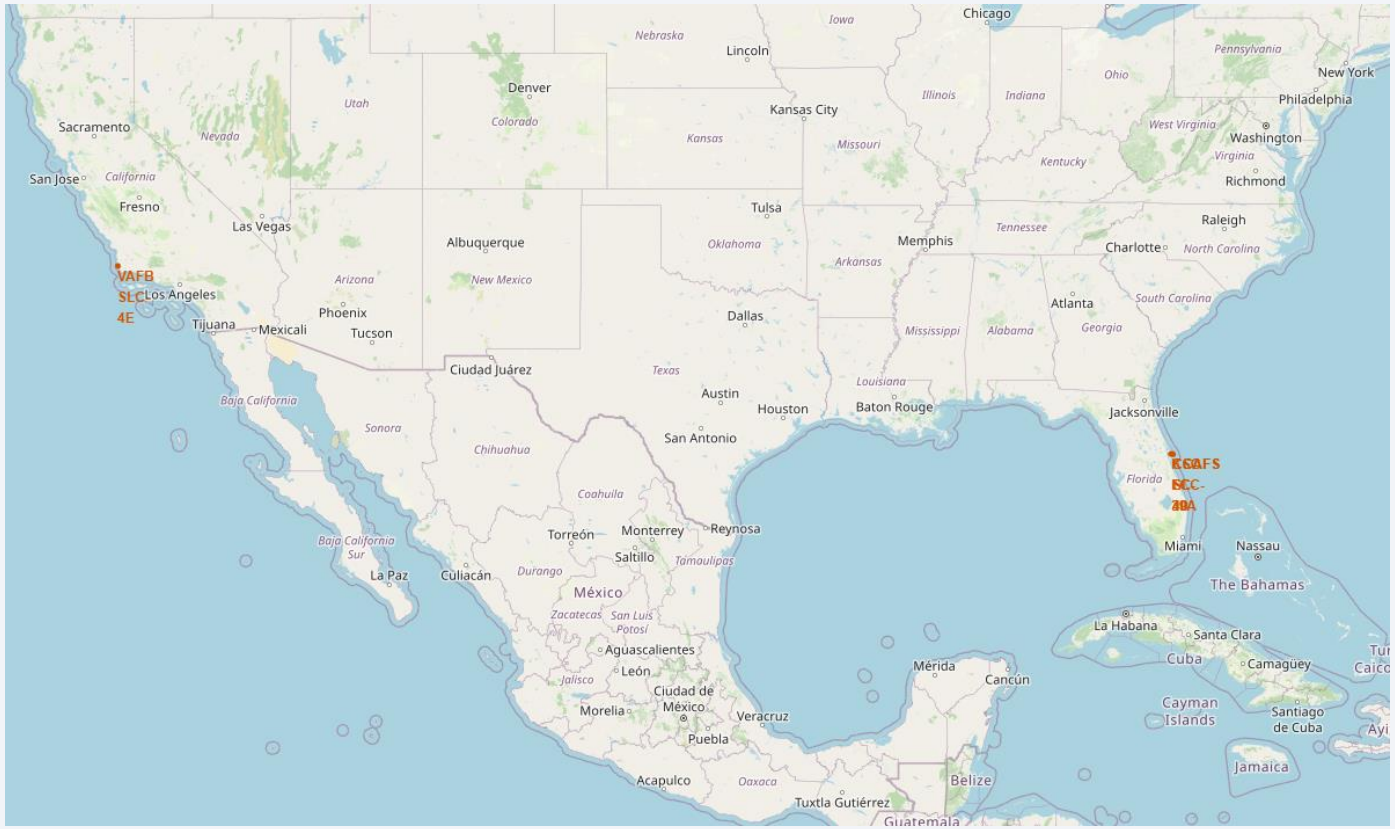
- Result

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

- The combination of WHERE clause and BETWEEN operator on DATE column is used to filter and returns only rows that have dates between the condition.
- ORDER BY keyword with descending (DESC) order on TOTAL_NUMBER column returns sorted rows with highest total number as the first row and lowest total number as the last row.
- The number of success and failed outcomes are about the same.


Section 3
Launch Sites
Proximities Analysis

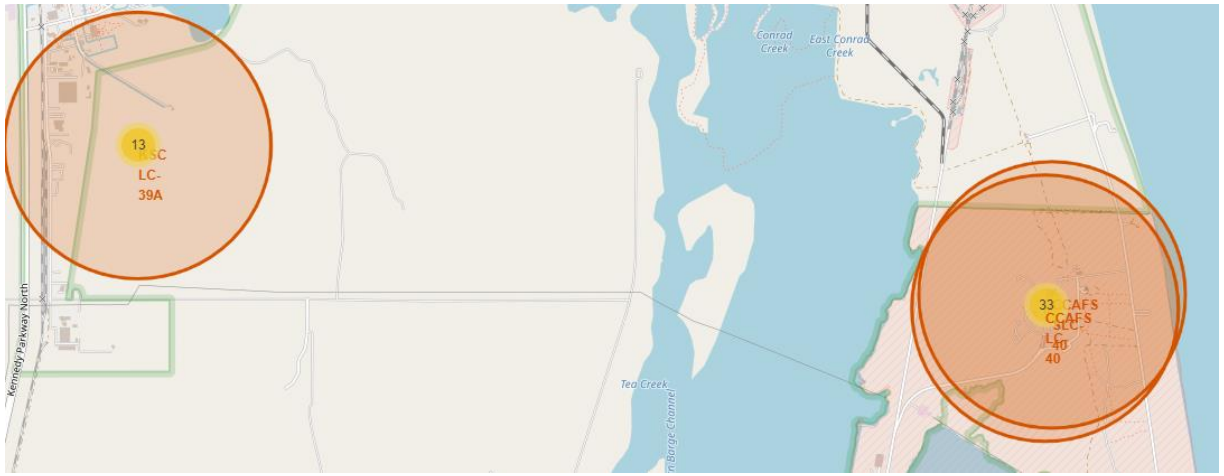
All Launch Sites' Locations

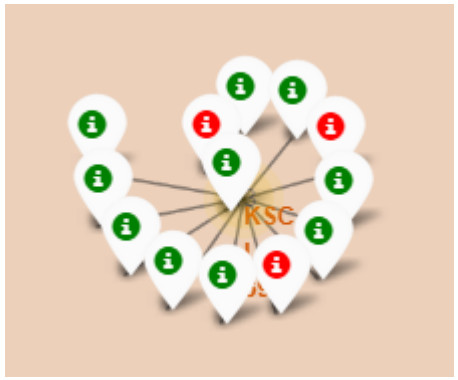


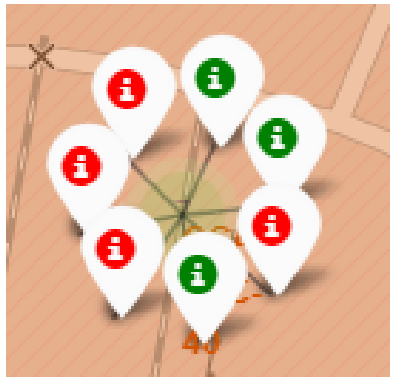
- The left map shows all SpaceX launch sites in the United States.
- All launch sites are near the coast.

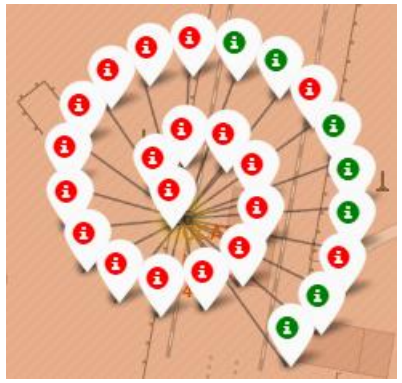
Color-labeled Launch Outcomes

Launch sites in Florida

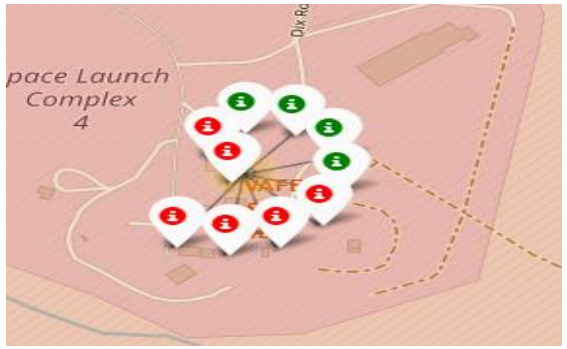






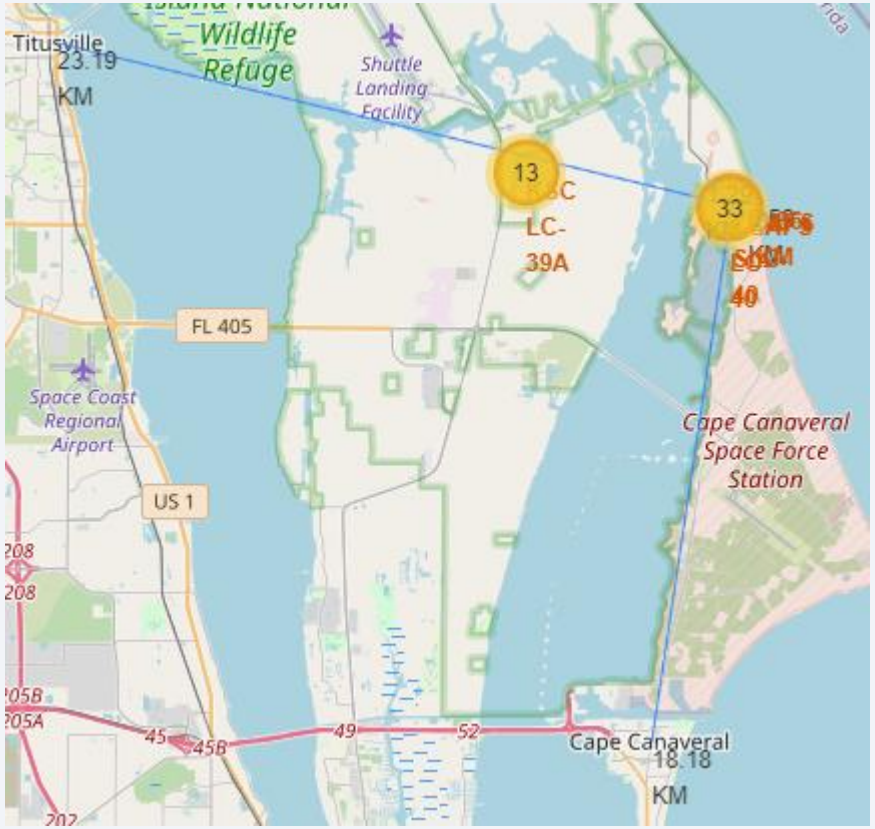


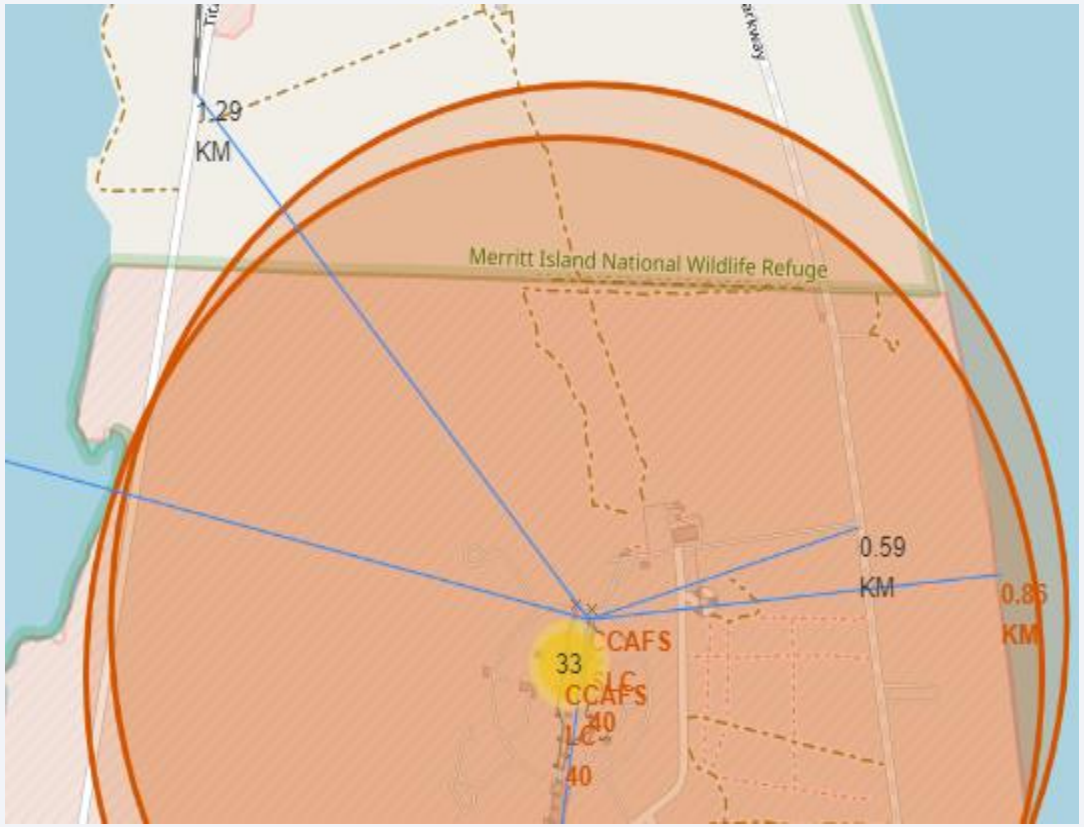
Launch sites in California



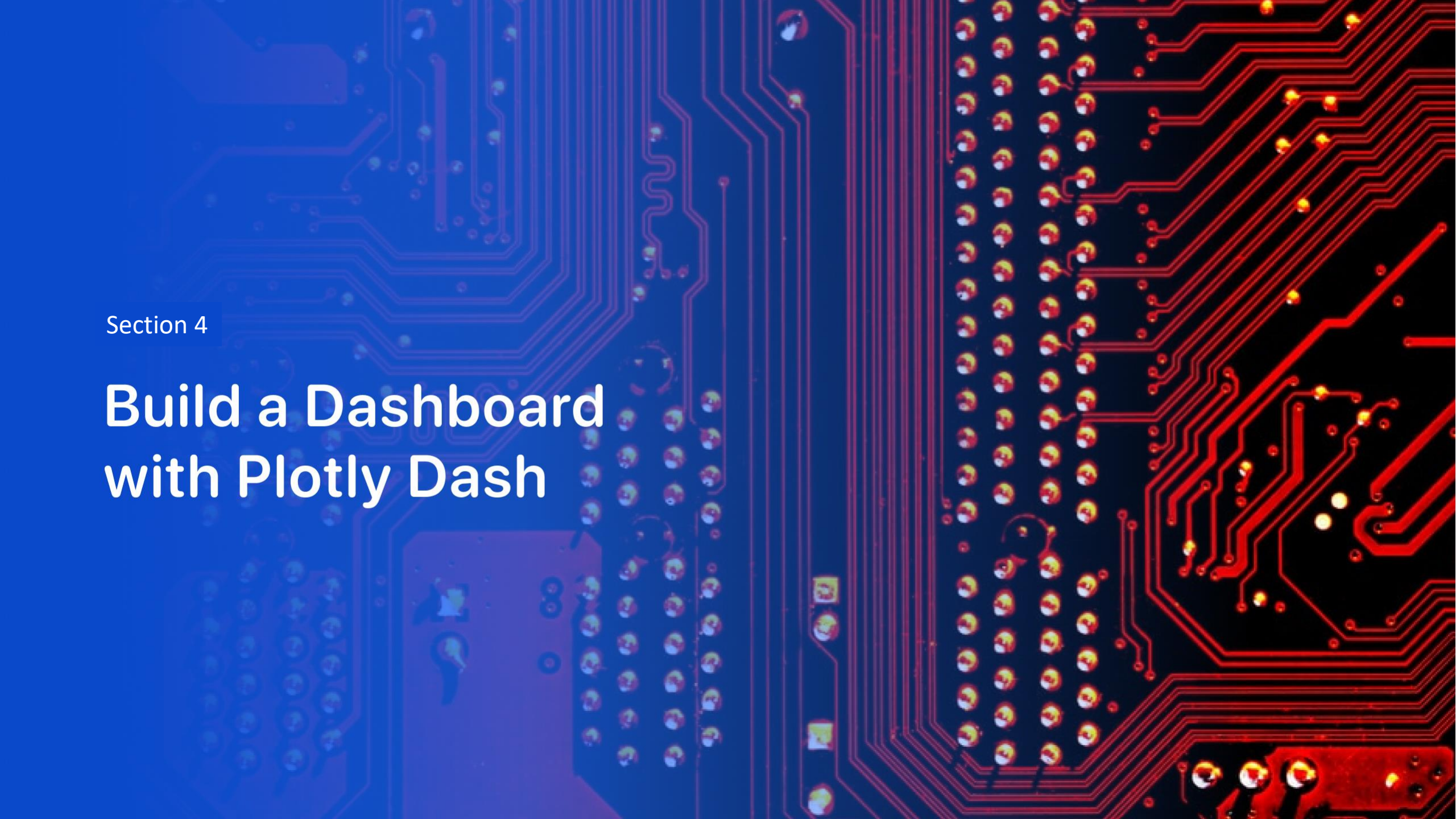
- By clicking on the marker clusters (representing with number), it will pop-up multiple marker representing each outcome.
- The green marker represents a successful landing.
- The red marker represents a failed landing.

Proximities of Launch Sites

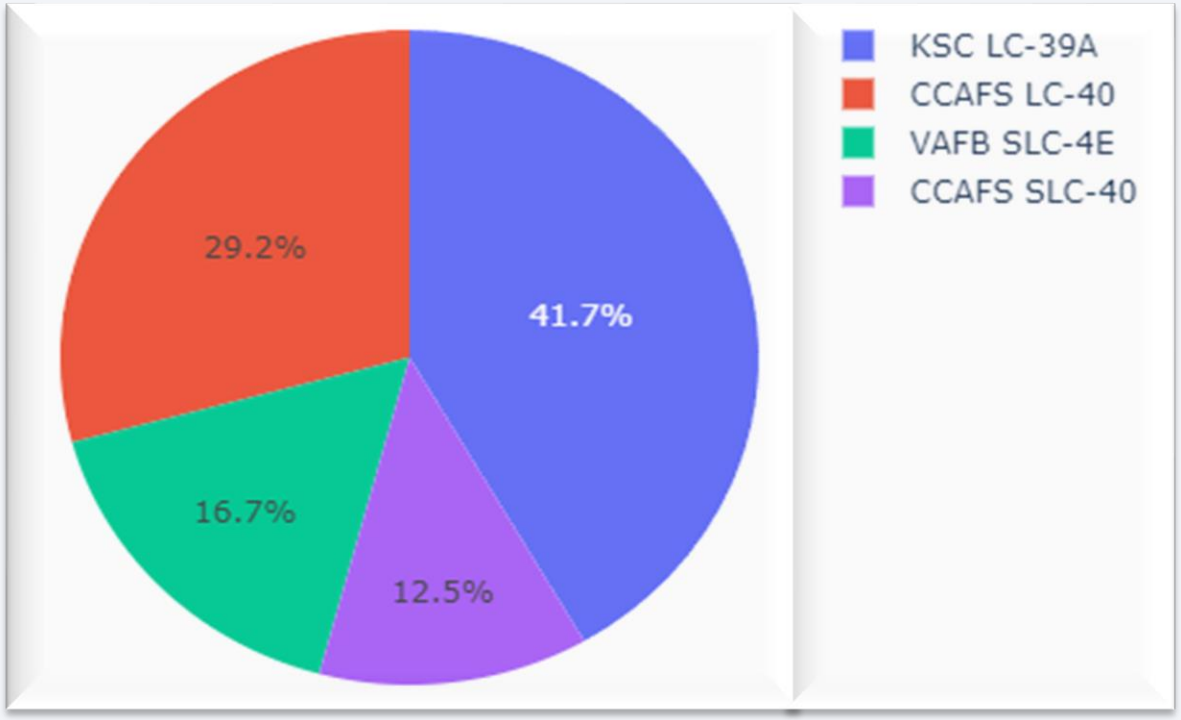




- Launch sites are close proximity to railways, highways and coastline with the distance from a launch site around 1.29, 0.59 and 0.85 km respectively.
- Launch sites keep certain distance away from cities with the distance from a launch site more than 15 km.

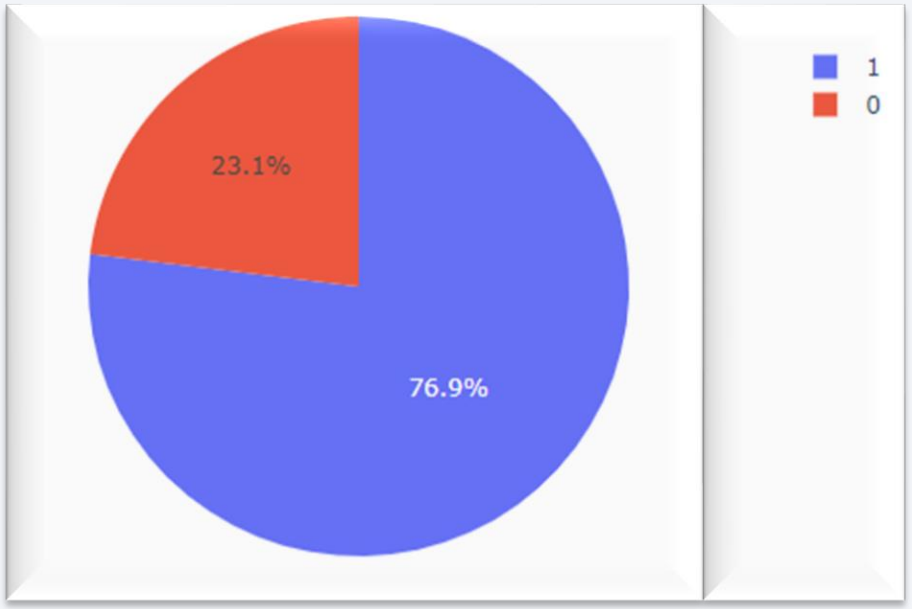

Section 4
Build a Dashboard with Plotly Dash

Total Success Launches by All Sites



- KSL LC-39A launch site has the most launch success among all sites.
- VAFB SLC-4E launch site has the least launch success among all sites. Probably because of
  - The small-size sample data
  - Its locations (West coast and East coast)

Launch Site with Highest Launch Success Ratio



- KSC LC-39A has the highest success ratio with 10 successful landing and 3 failed landing.

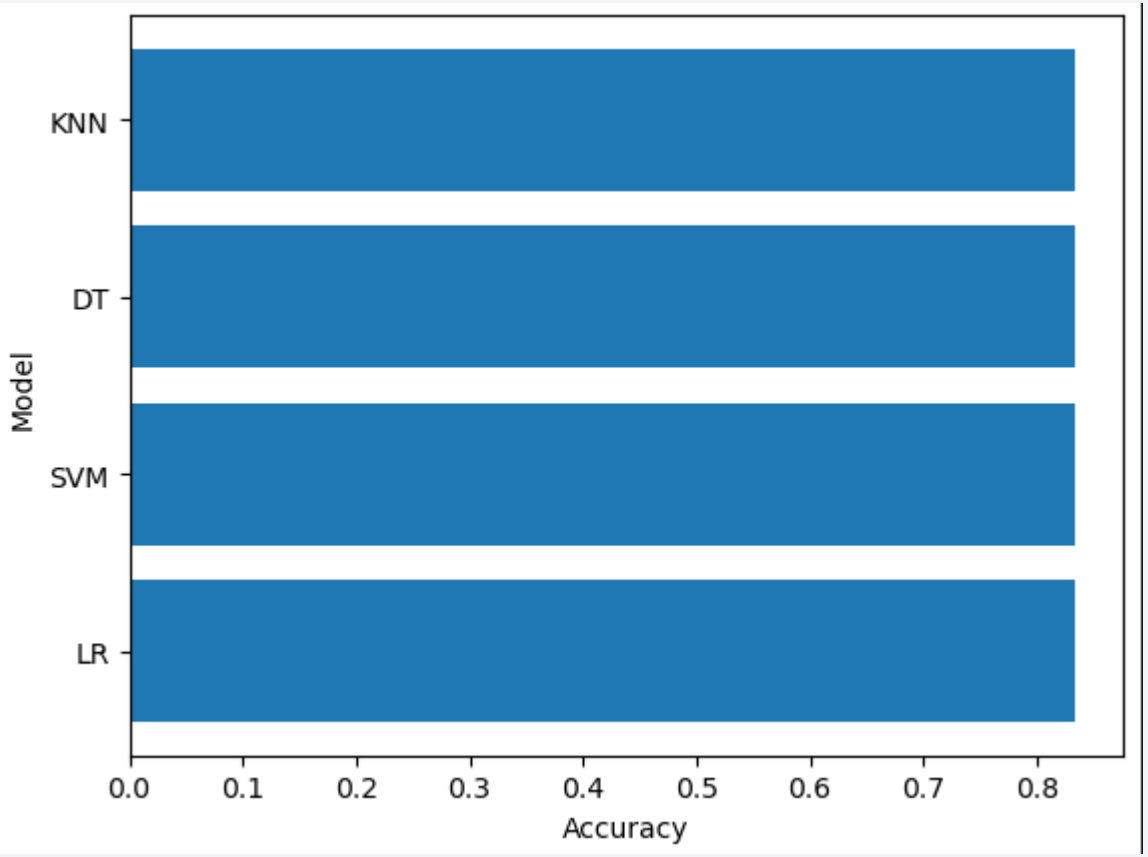
Scatter plot of Payload vs. Launch Outcome

- Most of successful launch outcome occur with payload mass between 0 and 5,000 kg.
- Considering successful launch, FT booster version has the highest launch success rate.

Section 5

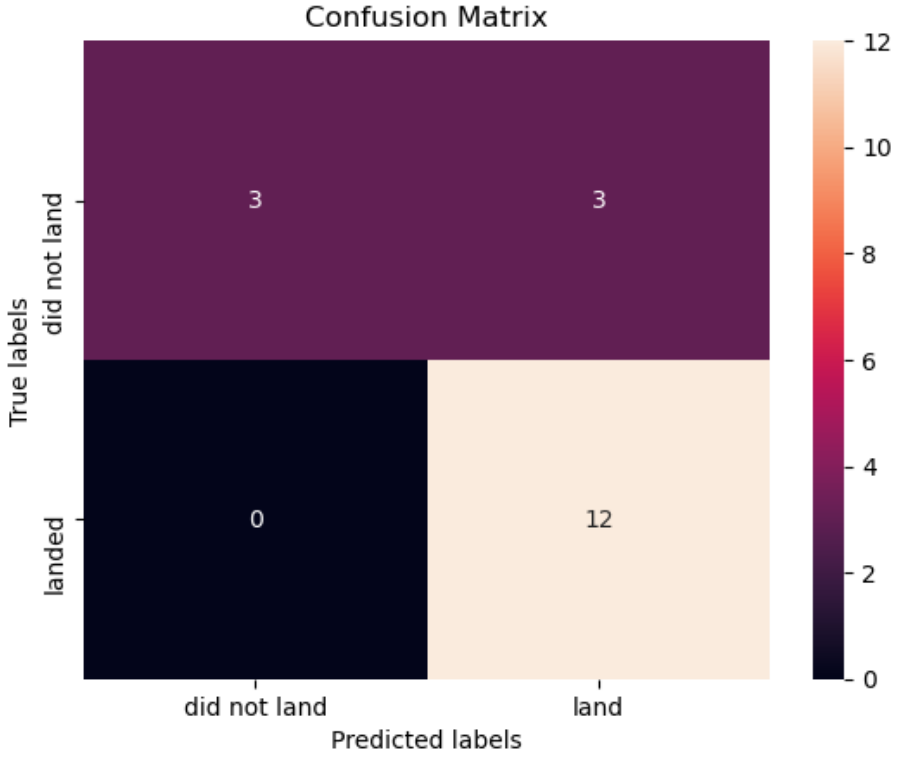
Predictive Analysis (Classification)

Classification Accuracy



- The accuracies of four models (K-Nearest Neighbors, Decision Tree Classifier, SVM, and Logistic Regression) which evaluate with test set are about the same at **83.3333%**.
- These accuracies have a high chance that they are wrong since the test size is only 18 samples.

Confusion Matrix



- The confusion matrix is the same for all models because the size of test set is small (90 samples, 72 training samples, 18 test samples)
- The models successfully predict 15 times (12 for successful landing and 3 for failed landing)
- There is 3 times that the models fail to predict so there is 16.7777% chance that these models will fail.

Conclusions

- As the number of flights increased, the success rate in landing increased and the trend of success rate is increased since 2013.
- Orbit types (ES-L1, GEO, HEO and SSO) have the highest success rate at 100%.
- The launch site is close to railways, coastline, and highways; however, it far from cities.
- KSL LC-39A launch site has the most launch success among all sites.
- Most of successful launch outcome occur with payload mass between 0 and 5,000 kg.
- Considering successful launch, FT booster version has the highest launch success rate.
- The best accuracy from four models (K-Nearest Neighbors, Decision Tree Classifier, SVM, and Logistic Regression) which evaluate with test set, 18 samples, for predicting the landing outcome is 83.3333%.

Appendix

- Coursera: Applied Data Science Capstone offered by IBM
- GitHub

Thank you!