Winning Space Race with Data Science

Hossein Salehi
August 27, 2022

Github URL:
https://github.com/hosseinsalehii/AppliedDataScienceCapstone.git

Outline

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

Executive Summary

- In this project, the successful rate of landing of Falcon 9 first stage is investigated. The process consists of the following steps:
  - Collecting data from SpaceX Wikipedia webpage and SpaceX API.
  - Cleaning and wrangling data (handling the missing values, re-categorizing the categorical variables to binary variables using One-Hot encoding, etc.).
  - Conducting explanatory data analysis (EDA) using SQL.
  - Visualizing the collected data through Interactive Visual Analytics (Folium maps) and dashboards (Plotly Dashboard).
  - Standardizing the collected data and finding the best parameters for machine learning models by using classification models (GridSearchCV).
  - Comparing the accuracy of the results of four machine learning models including: Logistic Regressions, Support Vector Machine, Decision Tree Classifier, and K Nearest Neighbors.

  All predictive results indicate that their accuracy is around 83.33%, overpredicting successful landings. Therefore, more data and additional analysis are necessary.

Introduction

- Project background and context:
  - As one of the leaders of commercial space traveling, SpaceX offers a competitive ticket price($62 MM USD vs. $165 MM USD) mainly due to reducing their expenses through recovering part of rocket in the first stage.
  - SpaceY is interested in entering the competition of commercial space travelling.
  - This project explores the machine learning approach to predict successful stage one recovery by performing predictive analysis using classification models.

- Problems you want to find answers
  - Determining the factors that are associated with a successful landing of rocket in the stage one?
  - Predicting stage one recovery?

Section 1

Methodology

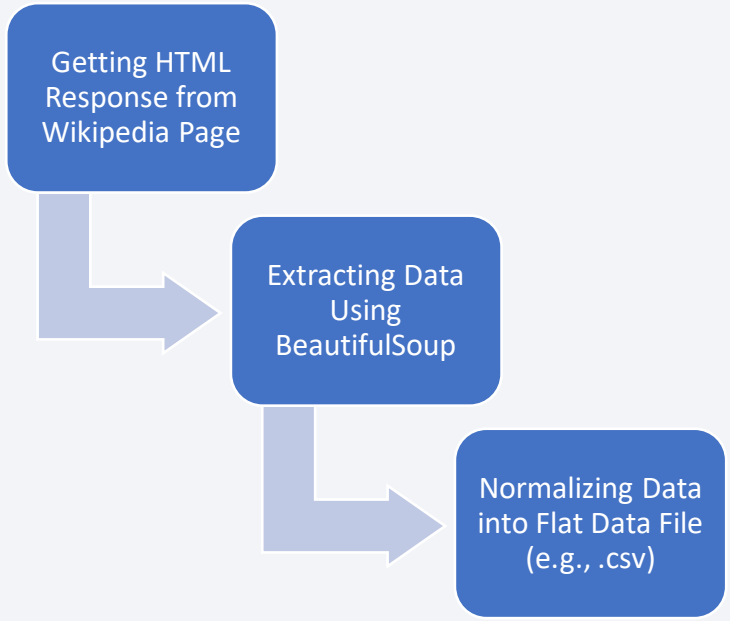
Methodology

- Data collection methodology:
  - The utilized data is collected from two main resources: Public API and SpaceX Wikipedia page.
- Performing data wrangling:
  - The categorical features have been encoded (successful vs. unsuccessful landing) by using one-hot module.
- The following steps have been performed and will be presented in this presentation:
  - Exploratory data analysis (EDA) using visualization and SQL
  - Interactive visual analytics using Folium and Plotly Dash
  - Predictive analysis using classification models (GridSearchCV)

Data Collection

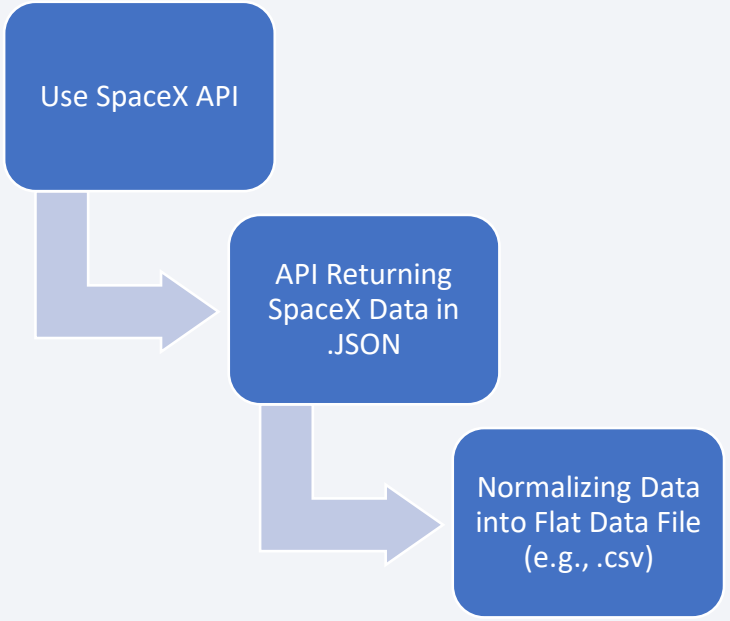
- The project data has been collected through a combination SpaceX public API and SpaceX Wikipedia page (web scraping method using BeahtifulSoup) as follows:

Web Scraping Process



[GitHub URL]

API Process



[GitHub URL]

Data Collection – Web Scraping (Part I)

1. Receiving Response from HTML

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

2. Creating BeautifulSoup Object

```
soup = BeautifulSoup(html_data.text, 'html.parser')
```

3. Finding all Tables

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

4. Getting column names

```
column_names = []
table_headers = first_launch_table.find_all('th')
for i, row in enumerate(table_headers):
    col_name = extract_column_from_header(row)
    if (col_name != None and col_name != ''):
        column_names.append(col_name)
```

GitHub URL

Data Collection – Web Scraping (Part II)

5. Creating a Dictionary

```
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']=[]
```

6. Converting Dictionary to DataFrame

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

7. Converting DataFrame to .CSV

```
df.to_csv('myversion_spacex_web_scraped.csv', index=False)
```

Data Collection – SpaceX API (Part I)

1. Getting Response from API

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

2. Converting the Response to a .JSON File

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Cleaning the Data by using pre-built functions

```
# Call getBoosterVersion
getBoosterVersion(data)
# Call getLaunchSite
getLaunchSite(data)
# Call getPayloadData
getPayloadData(data)
# Call getCoreData
getCoreData(data)
```

GitHub URL

Data Collection – SpaceX API (Part I)

4. Assigning List to 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}
```

5. Filtering DataFrame and Exporting to a .CSV

```
data_falcon9 = df.loc[df['BoosterVersion'] != "Falcon 1"]
data_falcon9.to_csv('myversion_dataset_part_1.csv', index=False)
```

Data Collection

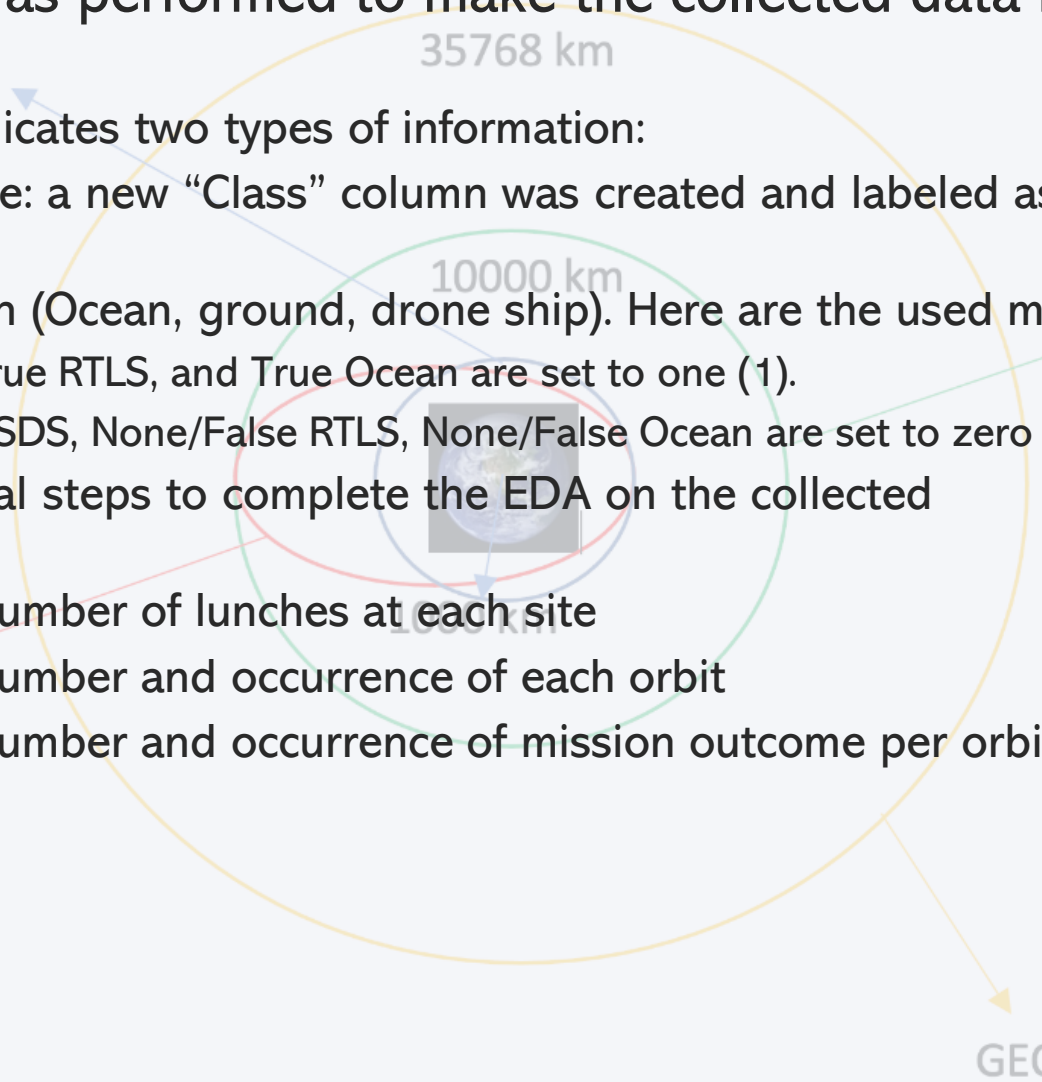
- The following are the data columns that were extracted by SpaceX Wikipedia web scraping and SpaceX API methods:

  - **SpaceX API:**
    - Flight Number, Date, Booster Version, Payload Mass, Orbit, Launch Site, Outcome, Flights,
    - Grid Fins, Reused, Legs, Landing, Block, Reused Count, Serial, Longitude, Latitude

  - **SpaceX Wikipedia Web scraping:**
    - Flight No., Launch site, Payload, Payload Mass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time

Data Wrangling

- The following steps was performed to make the collected data ready for further analysis:
  - The outcome data indicates two types of information:
    1. Mission outcome: a new “Class” column was created and labeled as 1 if successful and zero if otherwise,
    2. Landing location (Ocean, ground, drone ship). Here are the used mapping values:
       - True ASDS, True RTLS, and True Ocean are set to one (1).
       - None/False ASDS, None/False RTLS, None/False Ocean are set to zero (0).
  - Here are the additional steps to complete the EDA on the collected date:
    - Calculating the number of lunches at each site
    - Calculating the number and occurrence of each orbit
    - Calculating the number and occurrence of mission outcome per orbit type.



GitHub URL

EDA with Data Visualization

- In this step, the relationship/association between two variables are presented by performing EDA with data visualization. If there is an association between the variables of interest, then the next step is to use those variables to train the machine learning models. The following graphs were plotted:
  - Scatter Graphs:
    - Flight Number vs. Payload Mass
    - Flight Number vs. Launch Site
    - Flight Number vs. Orbit
    - Payload Mass vs. Orbit
  - Bar Graph:
    - Orbit vs. Mean
  - Line Graph:
    - Success rate of mission vs. Year

GitHub URL

EDA with SQL

- The second step of performing explanatory data analysis is using SQL queries to have a deeper comprehension of the collected data. Here are the completed steps in this stage:

  1. Loading the dataset into IBM DB2 Database.
  2. Loading SQL extension and establishing a connection with the database.
  3. Executing 10 different SQL queries to view:
     - The name of unique launch sites.
     - 5 records of launch sites starting with 'CCA'.
     - The total payload mass carried by boosters launched by NASA (CRS).
     - The average payload mass carried by booster version F9 v1.1.
     - The date when the first successful landing outcome in ground pad.
     - The names of the boosters which have success in drone ship and have payload mass >4000 but <6000.
     - The total number of successful and failure mission outcomes.
     - The list the names of the booster versions which have carried the maximum payload mass.
     - The failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015.
     - 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 URL

Build an Interactive Map with Folium

- Interactive Map with Folium is created to visualize the launch data considering the latitude and longitude coordinates at each launch site. Next, a Circle Maker is used to label each launch site by its name. Moreover, the outcomes of each launch (**success** vs. **failure**) are color-coded with **green** and **red**, respectively.

- By building an interactive map with Folium, the approximate distance from specific locations (e.g., coast, highway, city, etc.) is examined. This may help scientists better understand whether different characteristics of the location of a launch site are associated with a higher success rate of landing.

GitHub URL

Build a Dashboard with Plotly Dash

- In this section, two plots are created:

1. A Pie Chart: Showing the number of total and successful launches in a certain location or all locations. The size of the pie represents the total number of launches from a specific launch site. In other words, it represents the success rate of landing of each launch site.

2. A Scatter Plot: Showing the relationship between the launch site and the payload mass (which can be selected from 0 kg up to 10K kg).

The previews of these two plots are displayed in the result slide.

GitHub URL

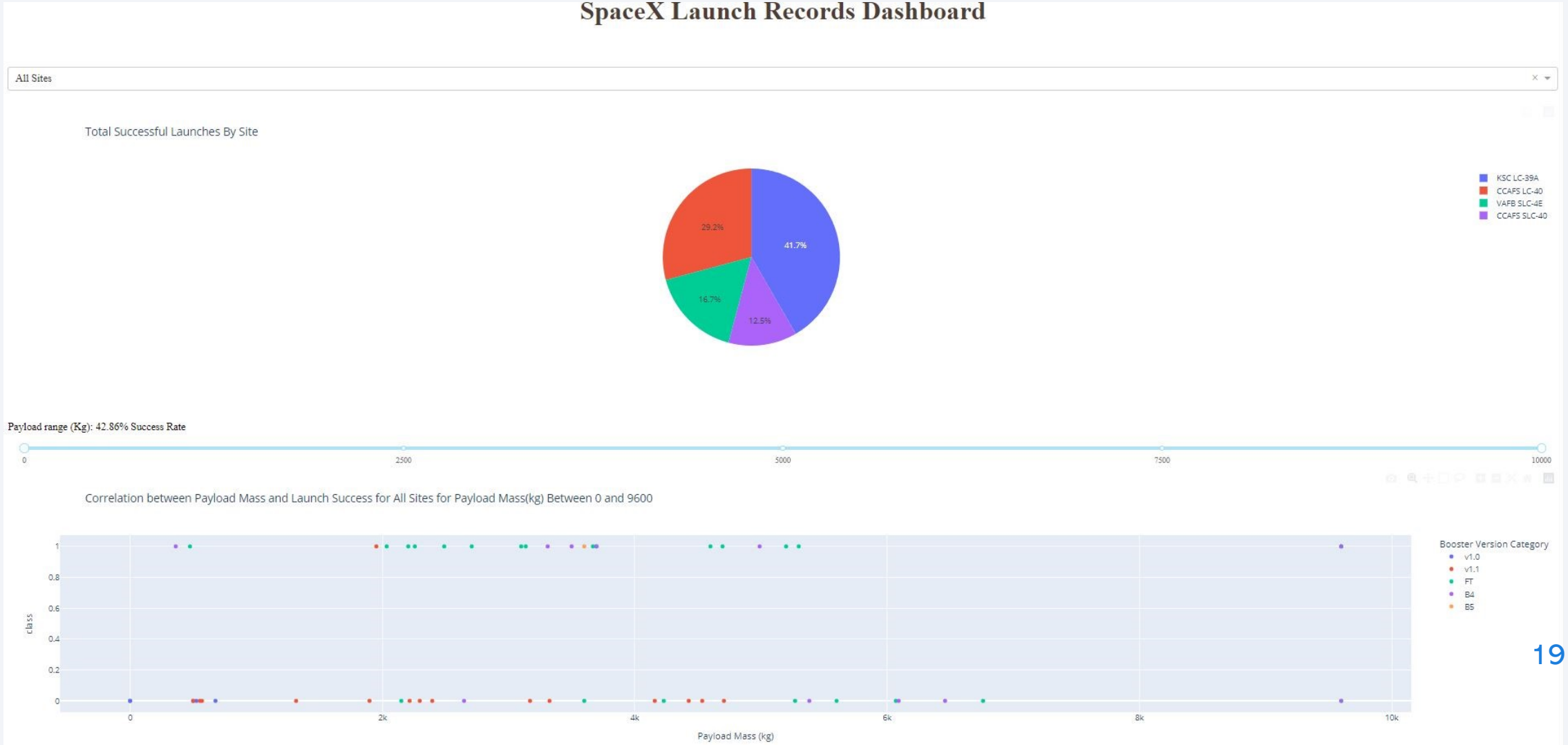
Predictive Analysis (Classification)

- In order to perform predictive analysis, the following steps are completed:
    1. Loading Numpy and Pandas libraries.
    2. Splitting the dataset into two sets: training and testing.
    3. Constructing four different machine learning models and finding the optimal parameters using GridSearchCV for producing the best outcomes.
    4. Utilizing accuracy metric for each model and comparing confusion matrix of all models.
    5. Finding the best classification model with the highest accuracy.

GitHub URL

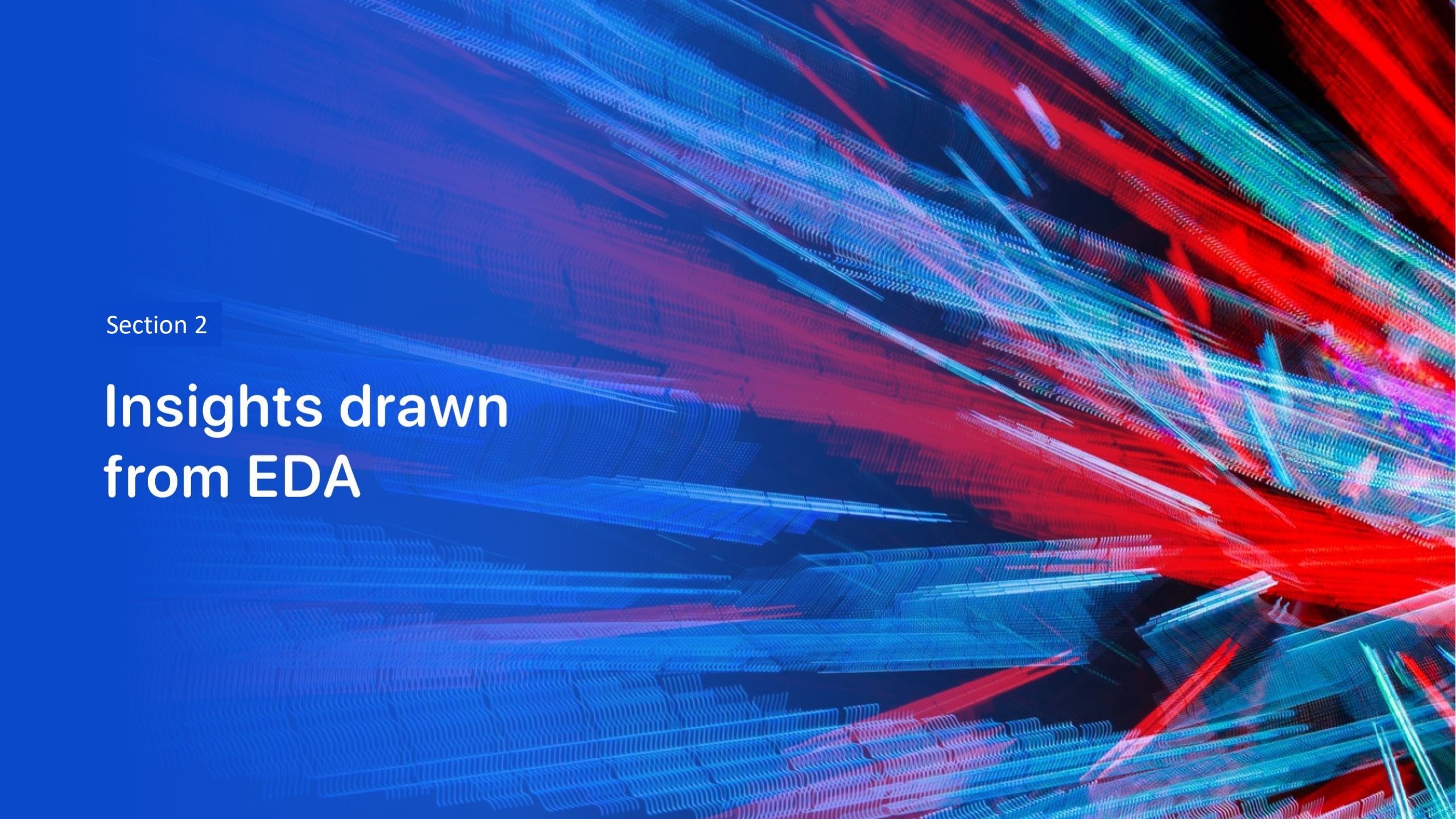
Results

- The interactive Plotly dashboard is as follows:



Section 2

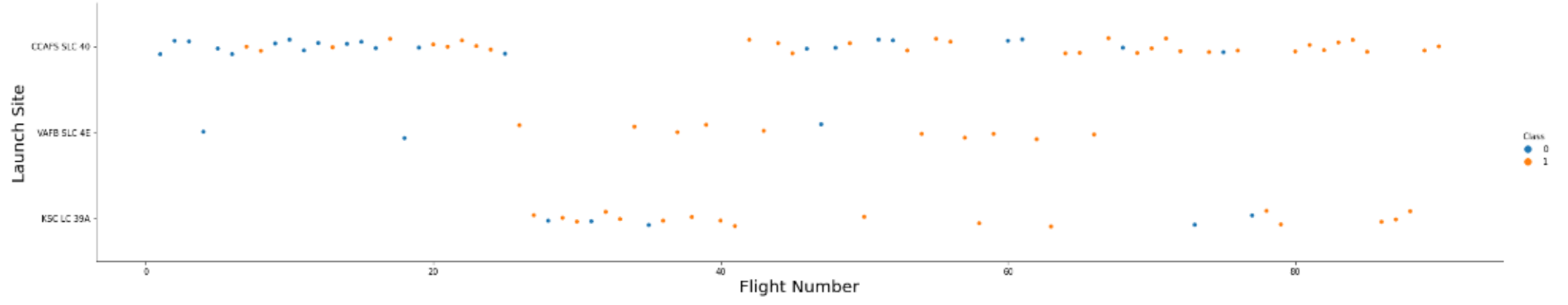
Insights drawn from EDA



Flight Number vs. Launch Site

- The scatter plot of Flight Number vs. Launch Site

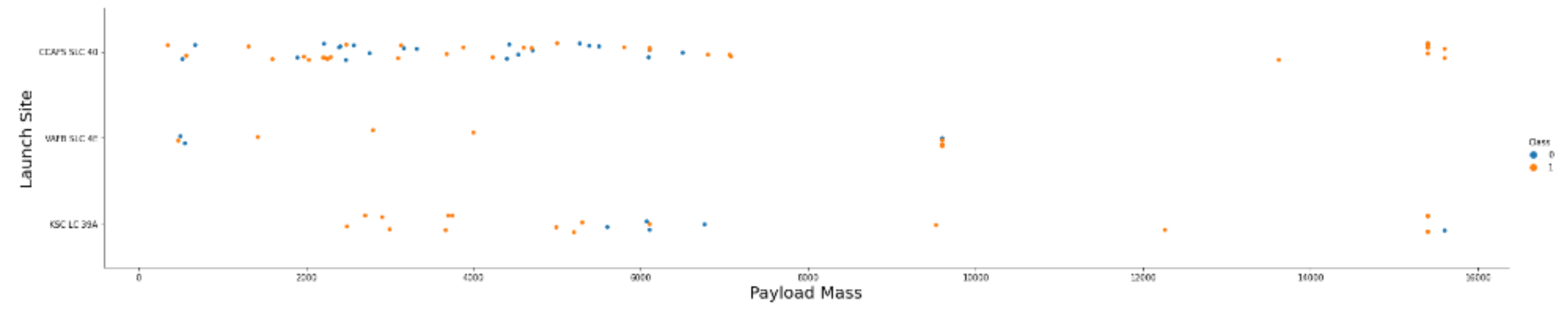
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```



- Blue dots and orange dots represent failed (zero) and successful (one) launches respectively.
- CCAFS SLC 40 has the highest number of launches and has the highest number of successful launches. (positive association between the two factors)

Payload vs. Launch Site

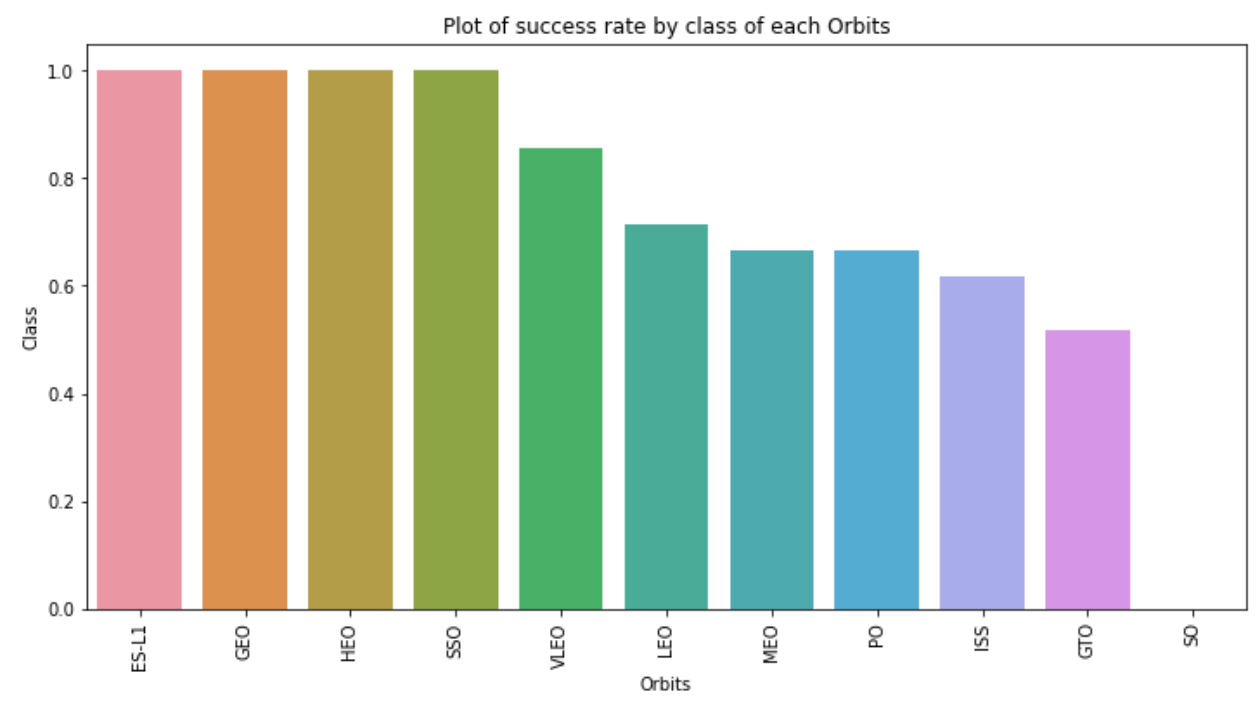
- The scatter plot of Payload vs. Launch Site



- Blue dots and orange dots represent failed (zero) and successful (one) launches respectively.
- Payloads below 6K kg are mostly launched from CCAFS SLC 40 launch sitehas the highest number of launches and has the highest number of successful launches. (positive association between the two factors)

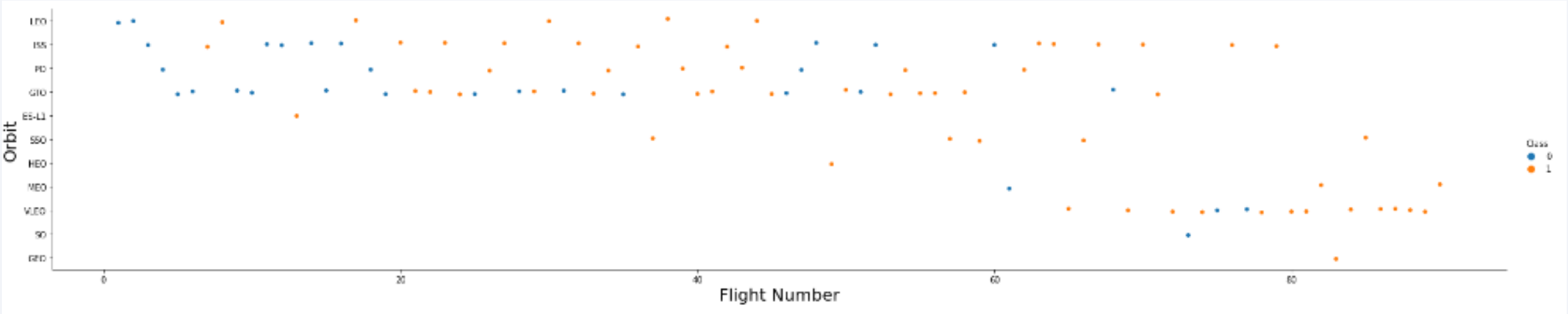
Success Rate vs. Orbit Type

- The bar plot of Success Rate vs. Orbit Type
- The scale of the success rate is between 0 and 0.6 success rate which is encoded between % and 100%.
- Four orbits (ES-L1, GEO, HEO, and SSO) have 100% success rate, and only one orbit (SO) has no success.
- The largest sample size is for GTO, with 27 data values, and 50% success rate.



Flight Number vs. Orbit Type

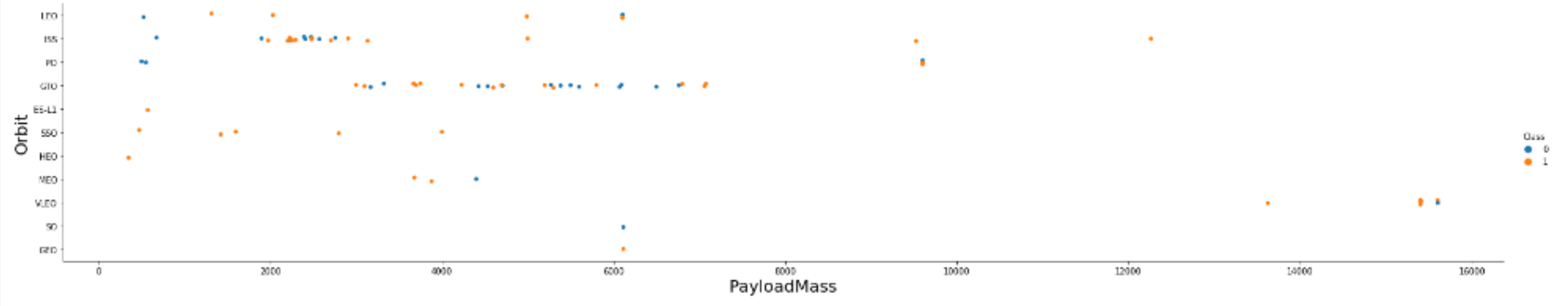
- The scatter plot of Flight Number vs. Orbit Type



- Blue dots and orange dots represent failed (zero) and successful (one) launches respectively.
- The pattern on the switch between different orbit types are clear in this scatter plot. Also, it presents a more successful launches to the lower orbits (the orange dots)

Payload vs. Orbit Type

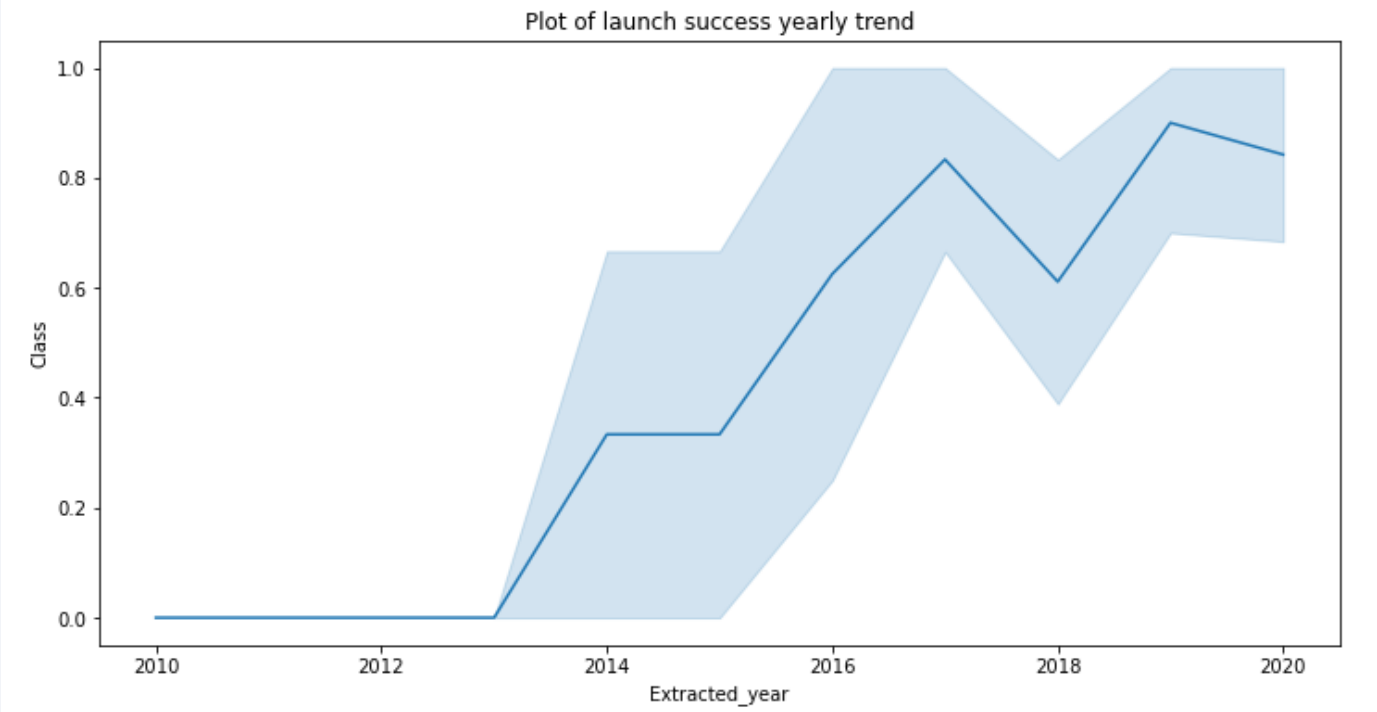
- The scatter plot of Payload vs. Orbit Type



- While heavy payloads are not as common as lighter payloads, it seems heavy payloads have negative association on GTO orbit as most failed launches happened for this orbit.

Launch Success Yearly Trend

- The line chart of yearly average success rate:
- As time passes, the success rate increases between 2013 and 2018.
- In 2018, there is a small set back which SpaceX was recovered from in the following year.



All Launch Site Names (SQL Query)

- SQL Query:
- Result: There are four launch sites:

```
%sql select distinct Launch_Site from SPACEXTBL
```

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

Launch Site Names Begin with 'CCA' (SQL Query)

- SQL Query and Result: retrieving all the data that the launch site name includes CCA.

```
%sql select * from SPACEXTBL where Launch_Site like 'CCA%' limit 5
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass (SQL Query)

- SQL Query and Result: This query gives back the total payload (KG) that the customer is NASA, sending the payloads to the International Space Station.

```
%sql select sum(payload_mass__kg_) from SPACEXTBL WHERE customer = 'NASA (CRS)'
```

sum(payload_mass__kg_)
45596

Average Payload Mass by F9 v1.1 (SQL Query)

- SQL Query and Result: This calculates the average payload mass that was carried by booster F9 v 1.1

```
%sql select avg(payload_mass__kg_) from SPACEXTBL WHERE booster_version = 'F9 v1.1'
```

avg(payload_mass__kg_)
2928.4

First Successful Ground Landing Date (SQL Query)

- SQL Query and Result: This returns the dates of first successful landing outcome on the ground pad, which is in late December 2015.

```
%sql select min(DATE) from SPACEXTBL WHERE landing__outcome = 'Success (ground pad)'
```

2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000 (SQL Query)

- SQL Query and Result: This return the name of the boosters with successful drone ship landing that carried a payload between 4K and 6K.

```
%sql select booster_version from SPACEXTBL where landing__outcome = 'Success (drone ship)'\
and payload_mass__kg_ between 4000 and 6000
```

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

Total Number of Successful and Failure Mission Outcomes (SQL Query)

- SQL Query and Result: This returns the total number of all successful and failed mission outcomes. While there is only one failed mission, among all 99 successful outcomes, one of them had unclear payload information.

```
%sql select mission_outcome, count(mission_outcome) from SPACEXTBL GROUP BY mission_outcome
```

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

Boosters Carried Maximum Payload (SQL Query)

- SQL Query and Result:

This returns the version of boosters that carried the max payload of 15.6K kg. As shown, the boosters are all F9 B5 series.

```
%sql select booster_version, payload_mass__kg_ from SPACEXTBL\
where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL)
```

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 (SQL Query)

- SQL Query and Result:

This returns the booster version and lunching site of the two launches that failed to land on drop ship.

```
%sql select booster_version, launch_site from SPACEXTBL where landing__outcome = 'Failure (drone ship)' and year(DATE) = 2015
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20 (SQL Query)

- SQL Query and Result:

This returns a list the count of landing outcomes (e.g., failed drone ship and successful ground pad) between the given two dates, in descending order.

```
%sql select count(landing__outcome), landing__outcome from SPACEXTBL \
where DATE between '2010-06-04' and '2017-03-20' group by landing__outcome\
order by count(landing__outcome) desc
```

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

Section 3

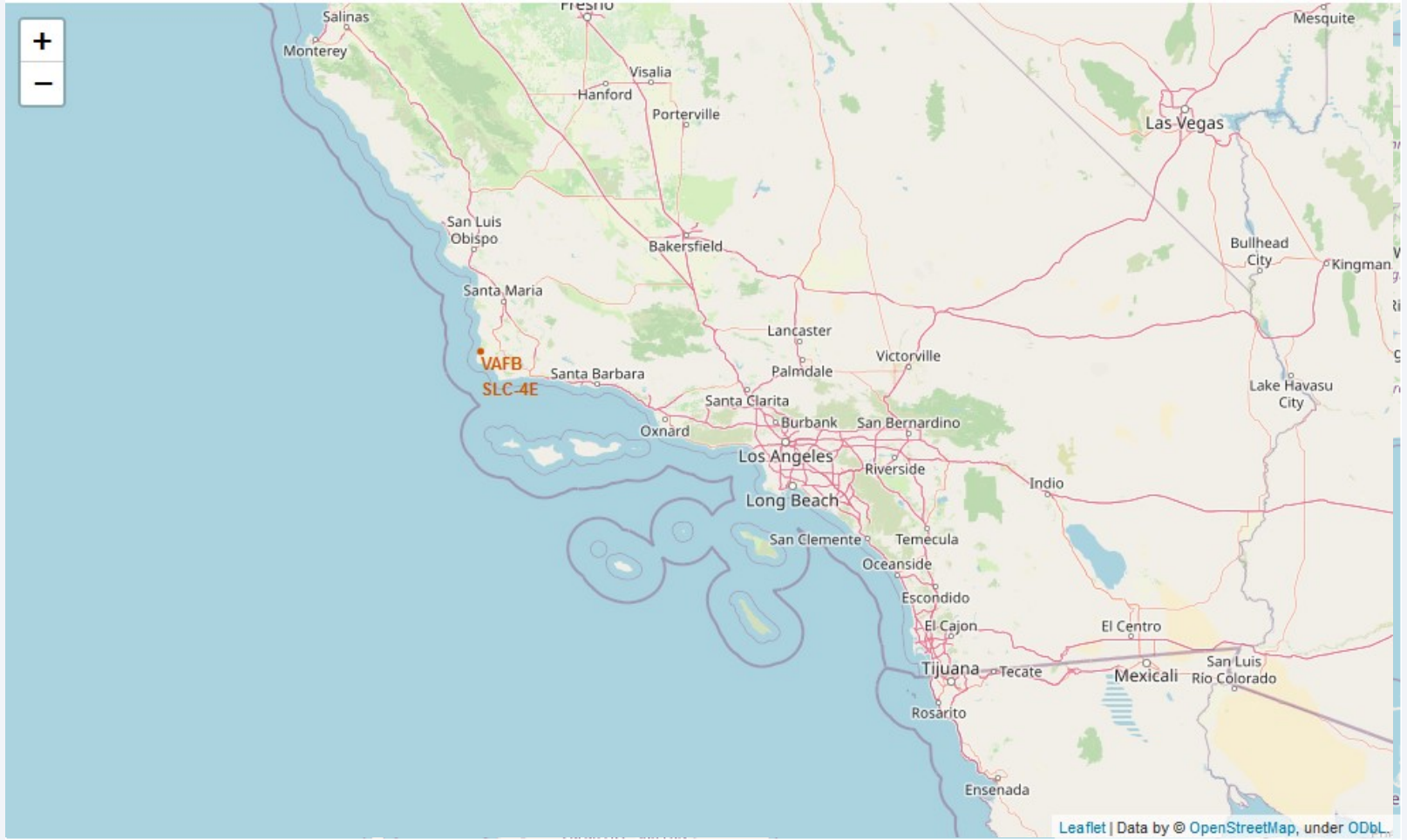
Launch Sites Proximities Analysis

Launch Sites on Map

- In the next few slides all launch sites are presented on the map of US as well as states of California and Florida.

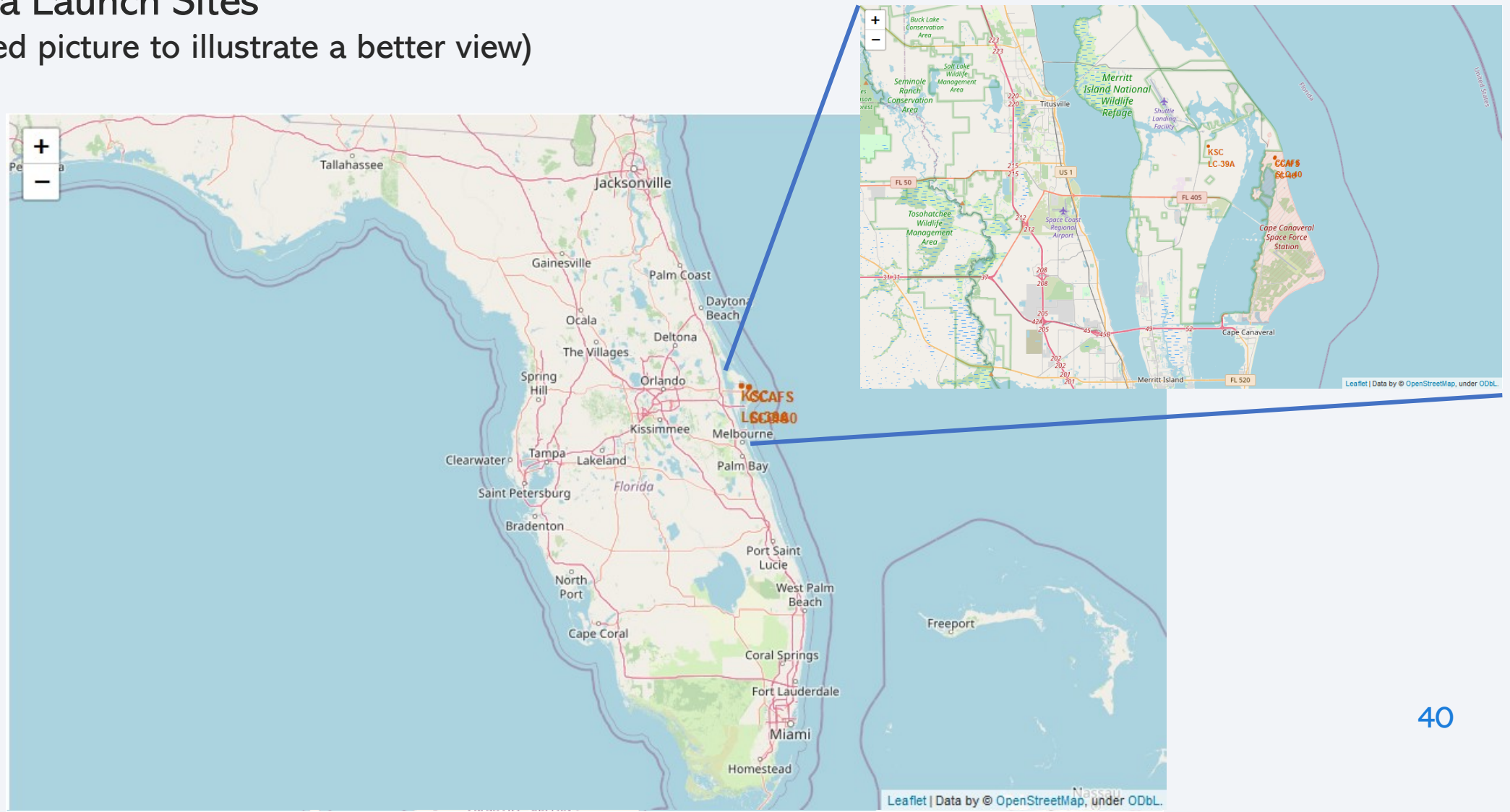
Launch Sites on Map

- California Launch Site



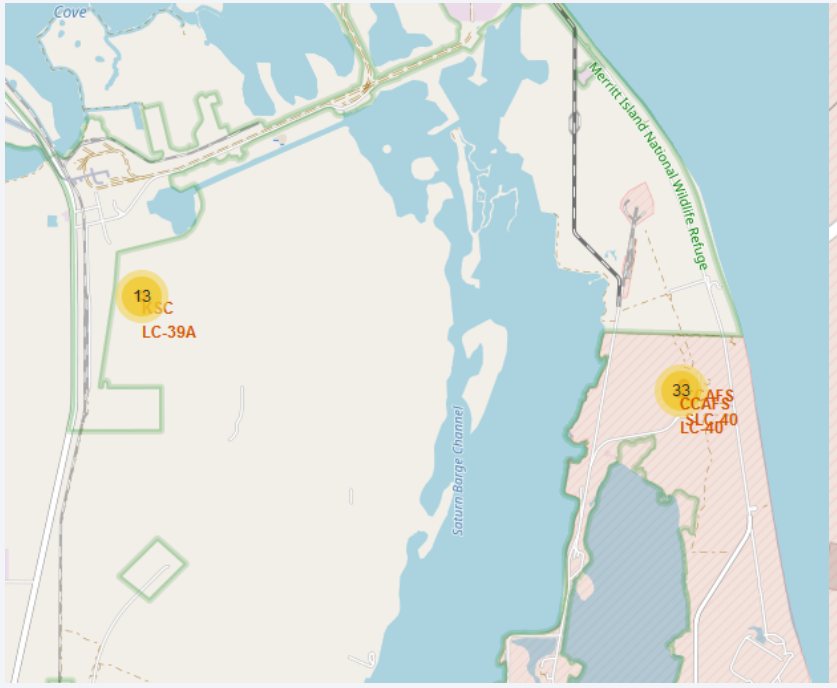
Launch Sites on Map

- Florida Launch Sites
  (Zoomed picture to illustrate a better view)



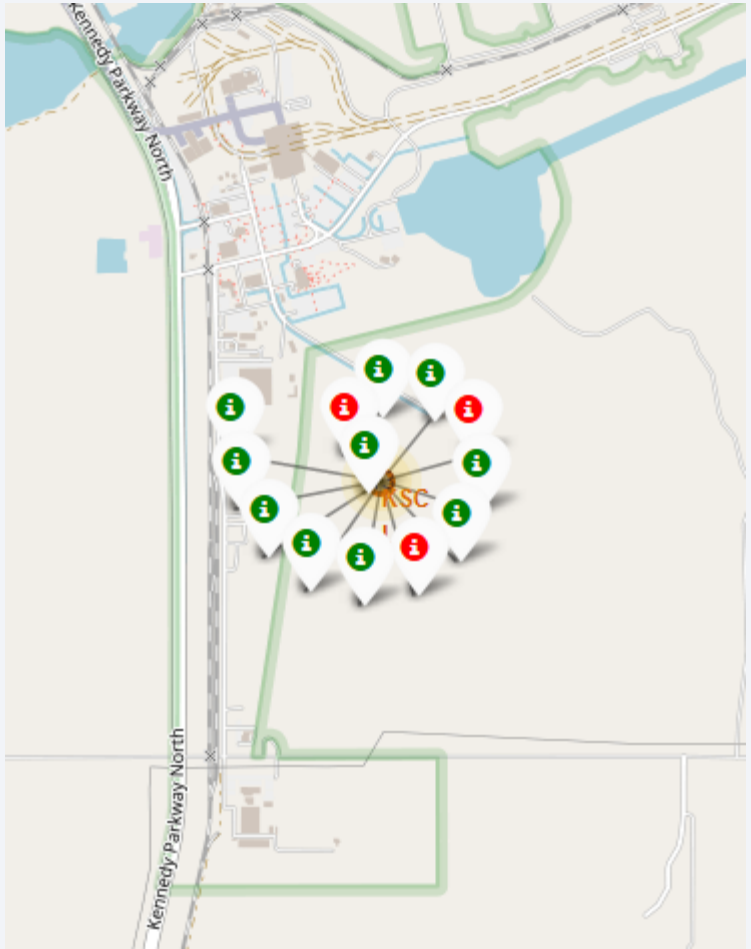
Launch Sites with Color-Coded Labels

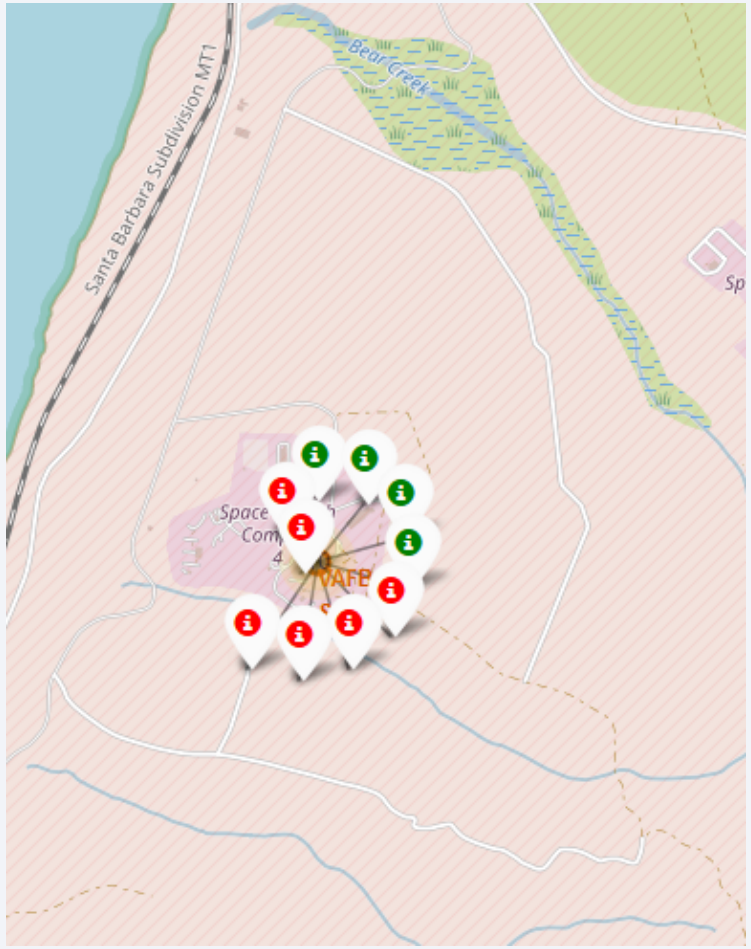
- All failed (labeled red) and successful (labeled green) landing are shown on the Folium map. For instance, all the landing outcomes of CCAFS SLC-40 launch site are presented in the following pictures.



Launch Sites with Color-Coded Labels

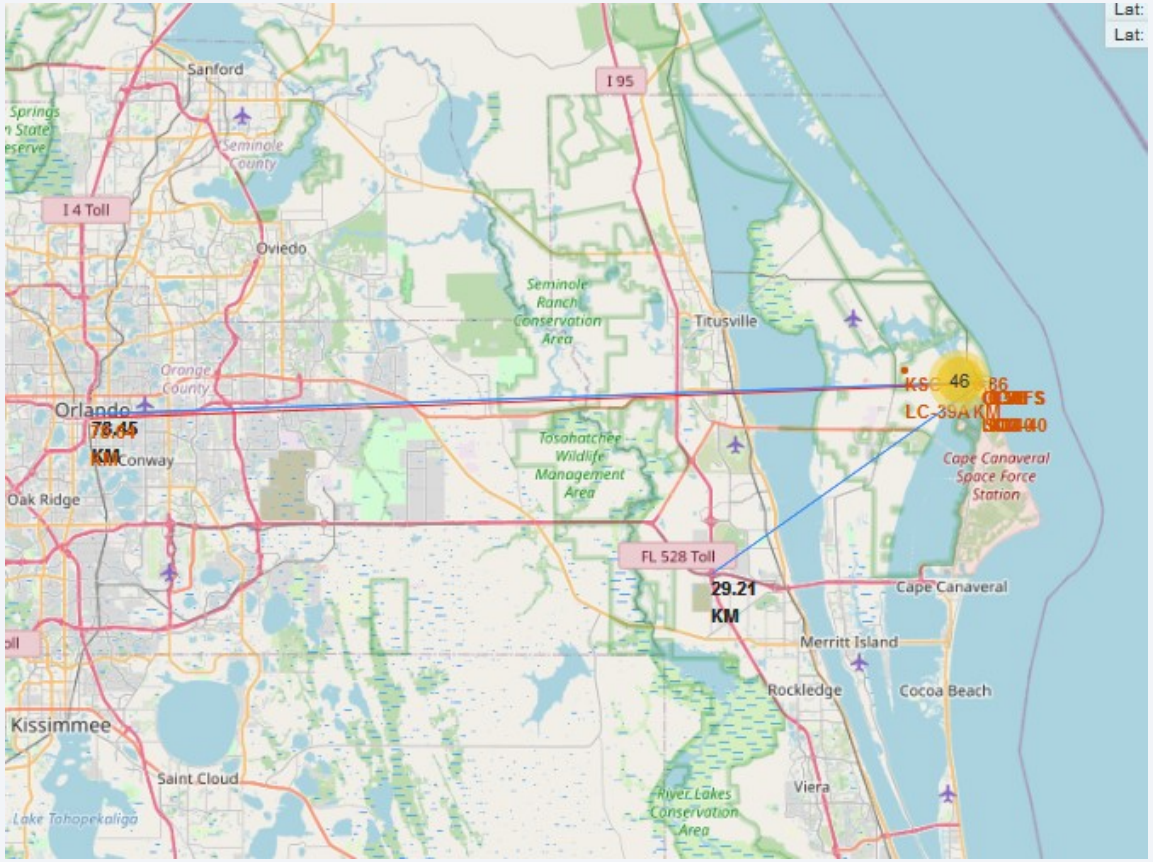
- And, here are the landing information on other two sites: KSC LC-39A in Florida (on the left) and VAFB SLC-4E in California (on the right).

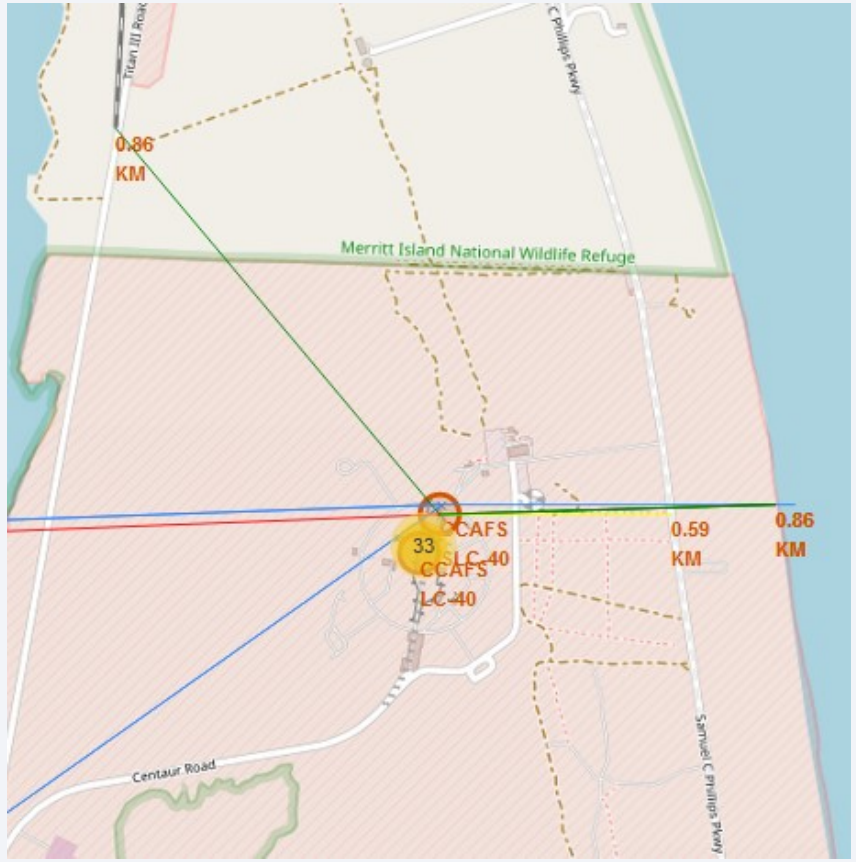




Distance between Launch Sites to Landmarks

- Pictures below show the distance of the launch sites in Florida from landmarks such as the city of Orlando, railroad, coastline, and highway. Please visit the github link to interact with the maps.

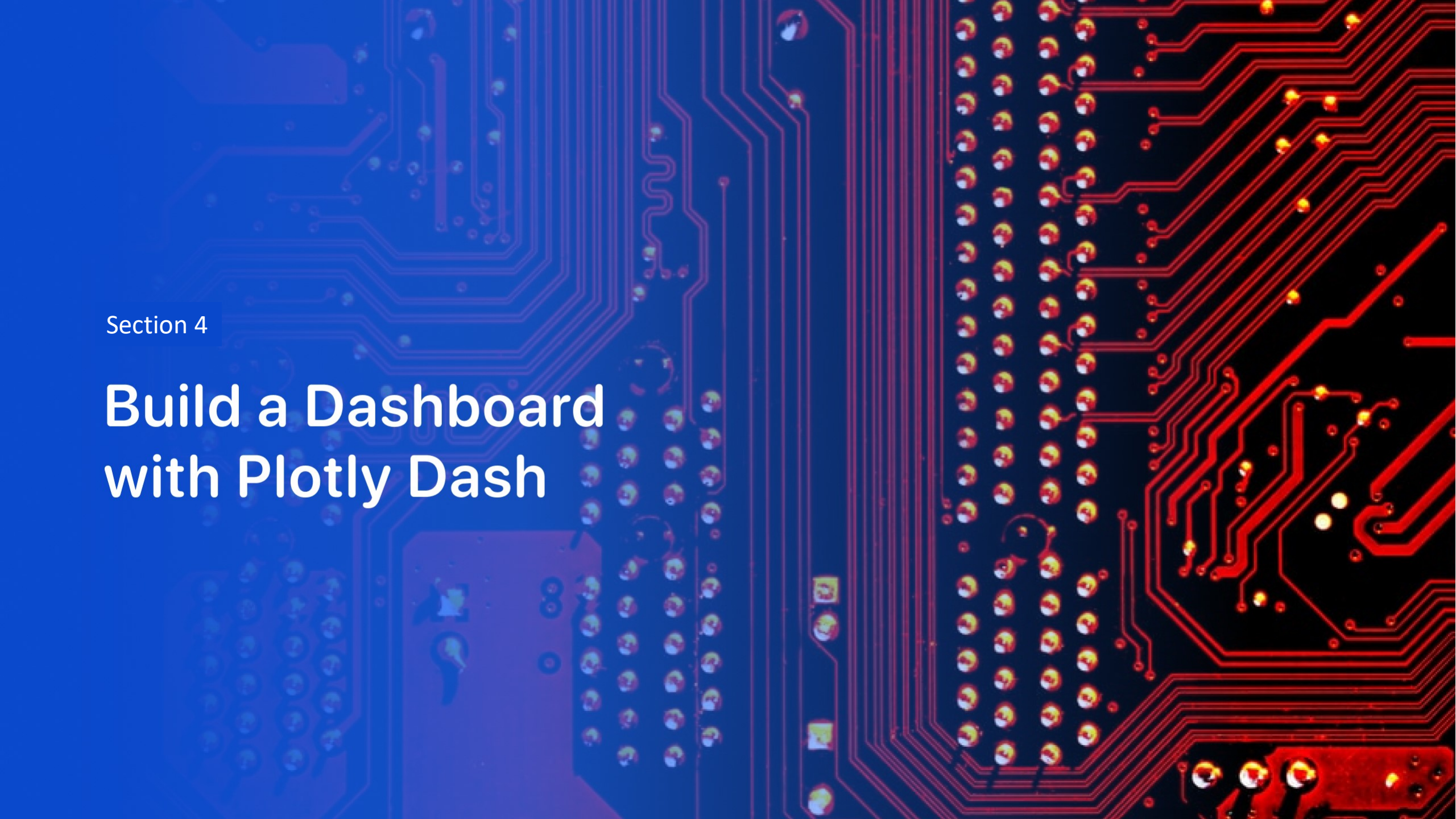




Distance between Launch Sites to Landmarks

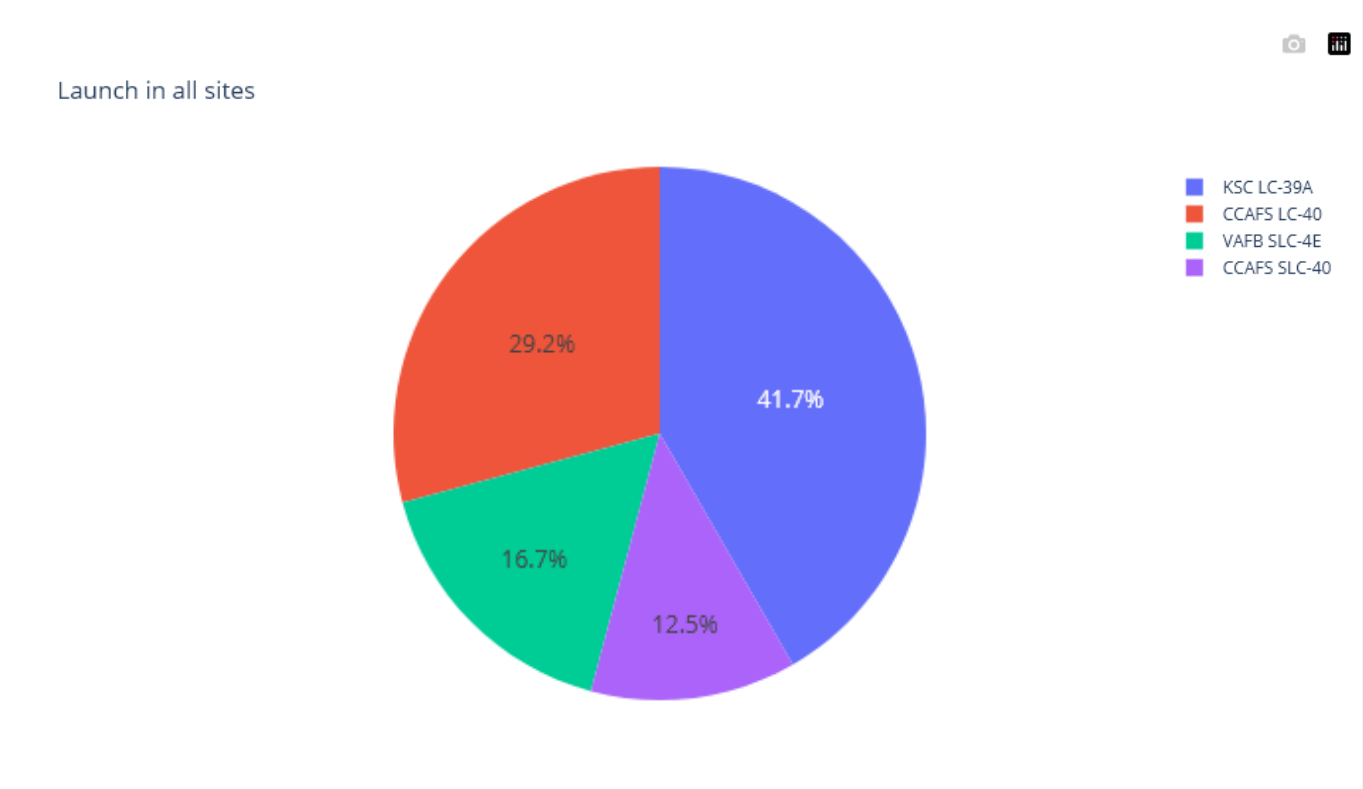
- Here are the answers at the end of the notebook:
  - After you plot distance lines to the proximities, you can answer the following questions easily:
  - Are launch sites in close proximity to railways? Yes
  - Are launch sites in close proximity to highways? No
  - Are launch sites in close proximity to coastline? Yes
  - Do launch sites keep certain distance away from cities? Yes

  The launch sites are far from the highways and cities and close to coastline as the safety of civilians is a goal.


Section 4
Build a Dashboard with Plotly Dash

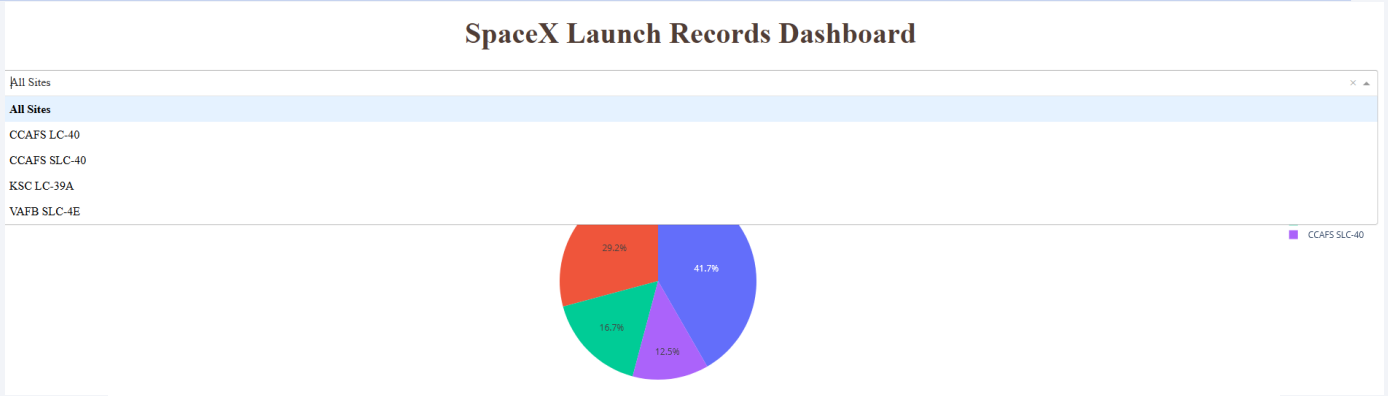
Pie Chart of Success Rates of All Launch Sites

- The pie chart below compares the success rate of the launch sites.
- KSC LC-39A has the highest success rate of landing among all four launch sites. While CCAF SLC-40 has the lowest success rate with only 12.5%.
- CCAFS LC-40 and VAFB SLC-4E are ranked 2nd and 3rd.

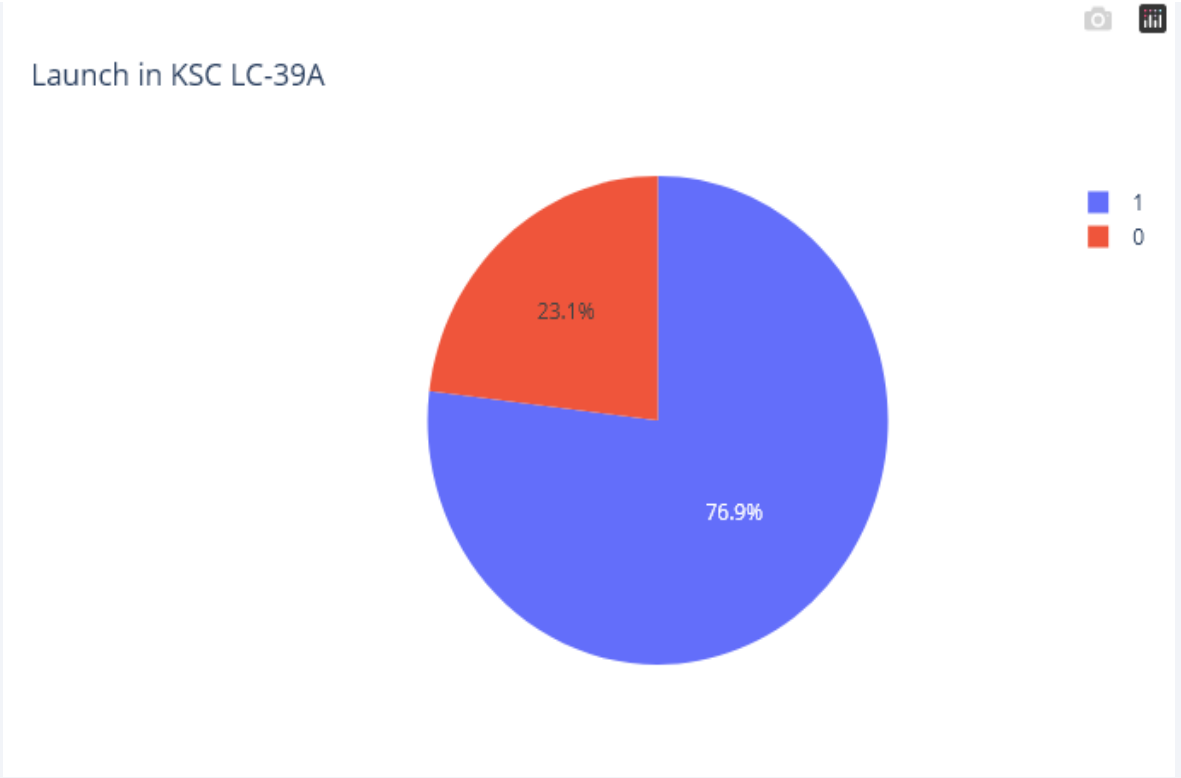


<Dashboard Screenshot 1>

- By choosing different launch sites using the dropdown menu, the success rate of each launch site is presented as below:

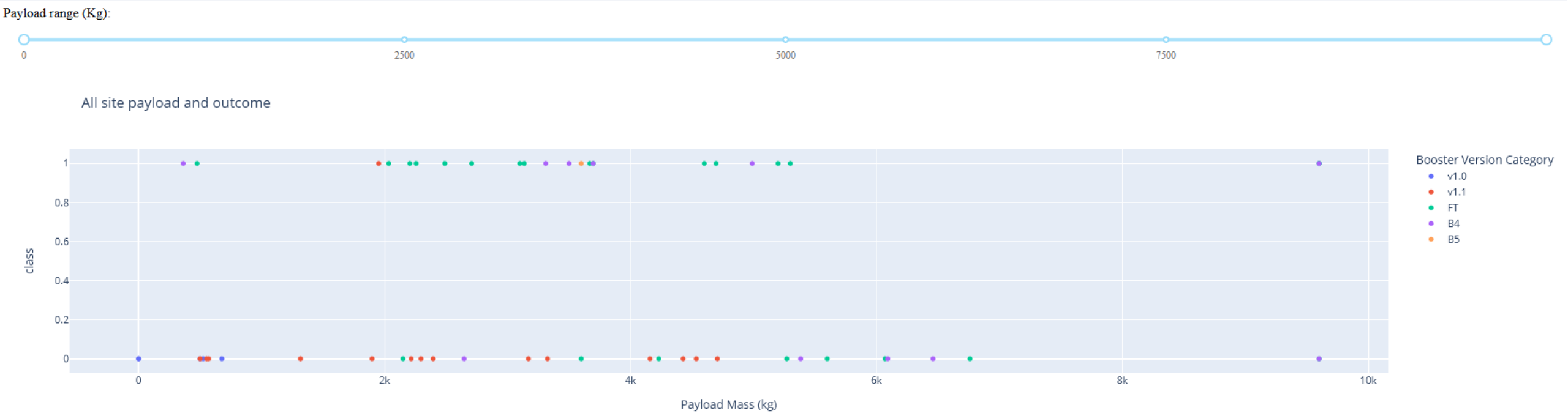


- For instance, KSC LC-39A launch site has a success rate of 76.9% and failure rate of 23.1%.



Dashboard Scatter Plot

- The scatter plot below illustrates the Payload vs. Launch Outcome of all sites. The payload can be changed by using the “Payload Rang (Kg)” bar.



Dashboard Records: Conclusion

- KSC LC-39A has the highest launch success rate, carrying 2,000 Kg – 10,000 Kg payloads.
- Launches with a payload between 0 Kg and 1000 Kg are more likely to failed. This can be explained by noting that initial tests that had a higher rate of failure may also be lighter with less payload.
- Finally, among all F9 Boosters, FT has the highest success rate.

Section 5

Predictive Analysis (Classification)

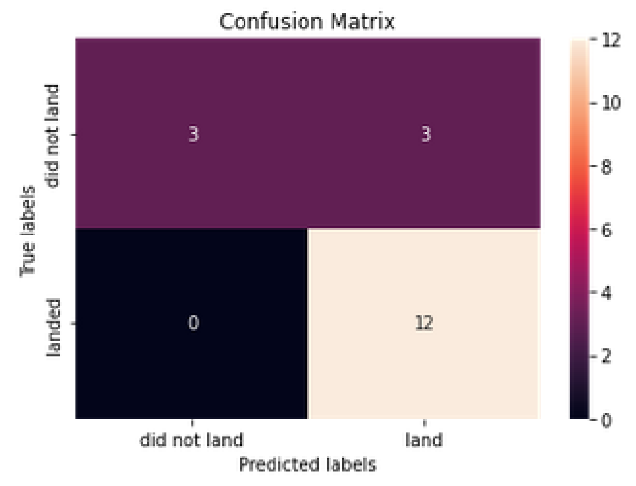
Classification Accuracy

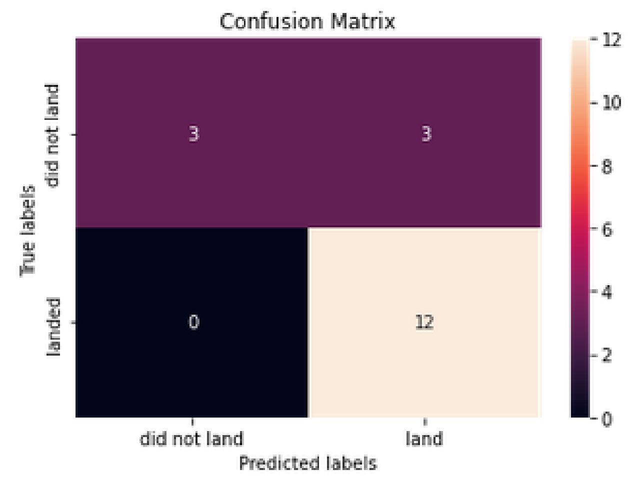
- The outcomes of all four classification methods and their confusion matrixes are illustrated here:

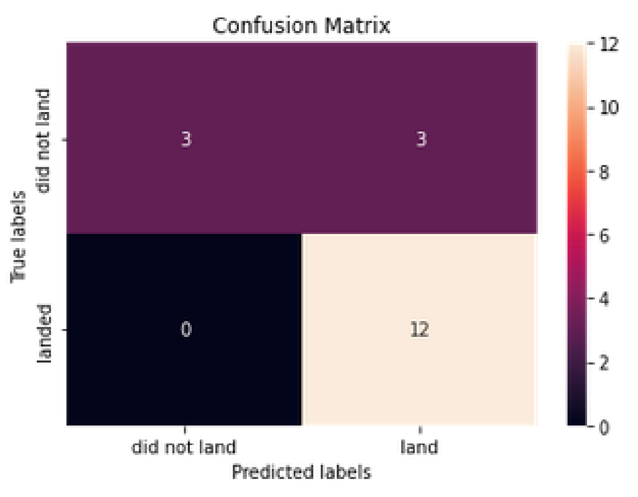
- All four models have very close accuracy (83,33%) on the test set. Having a small sample size of 18 might be the reason.

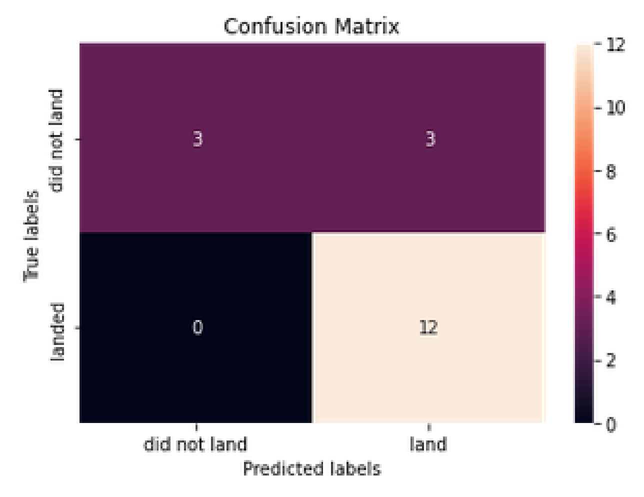
- However, this may cause a large variance in the accuracy results.

- Collecting more data samples can help resolving this issue.



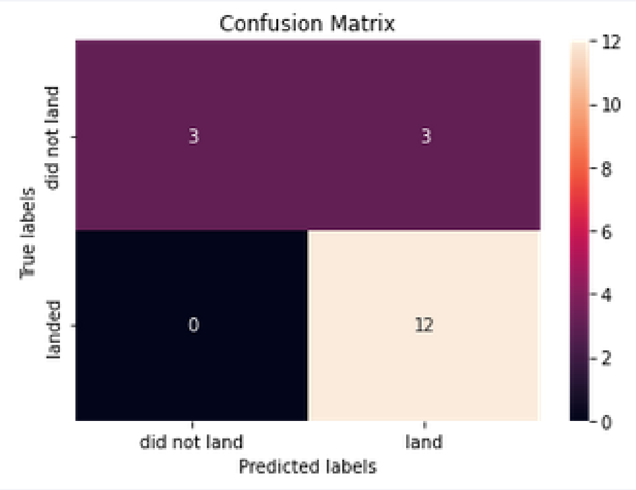


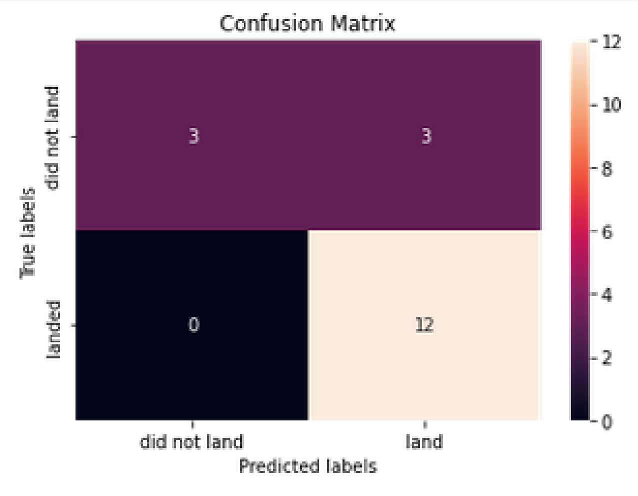


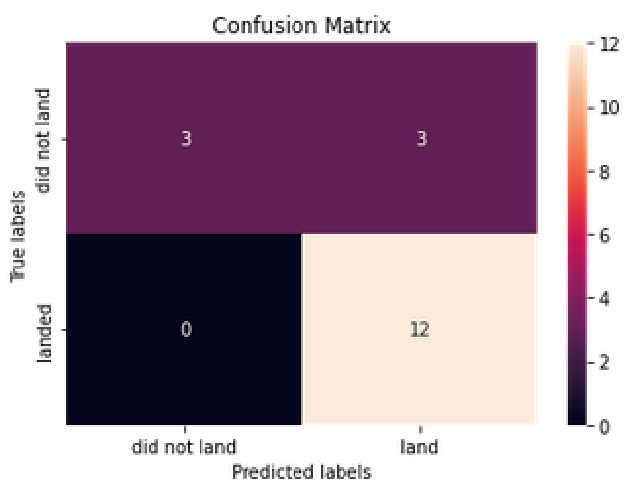


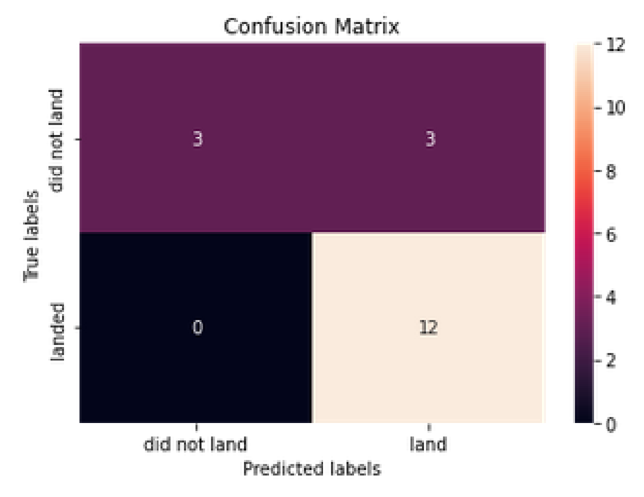
Classification Accuracy

- The outcomes of all four classification methods and their confusion matrixes are illustrated here:









Confusion Matrix

- The confusion matrix of Decision Tree shows the classifier can differentiate between the classes. However, a high rate of false positive (unsuccessful landing marked as successful) can be problematic and requires further investigation.

Conclusions

- Here are the main points of this machine learning project:

1. Among all tested machine learning models, the Decision Tree algorithm produces the most accurate outcome given the collected dataset.
2. KSC LC-39A had the highest success rate of landing compared to other three launch sites.
3. There is a positive association between the number of lunches and success rate of a launching site. This is not a causal relationship!

Appendix

- Python Code Connecting to Database (SQLlite):

Connect to the database

Let us first load the SQL extension and establish a connection with the database

```
%load_ext sql
```

```
import csv, sqlite3

con = sqlite3.connect("my_data1.db")
cur = con.cursor()
```

```
%sql sqlite:///my_data1.db
```

```
import pandas as pd
df = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/labs/module_2/data/Spacex.csv")
df.to_sql("SPACEXTBL", con, if_exists='replace', index=False,method="multi")
```

Appendix

- Python Code for Finding the Best (most accurate) Machine Learning Model:

Find the method performs best:

```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

Thank you!