IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Nuno Ferreira
August 2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection (using API's and web scraping)
  - Data wrangling
  - EDA with SQL
  - EDA with data visualization
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

Project background and context

In this project, we aim to predict the successful landing of the Falcon 9 first stage.

SpaceX advertises Falcon 9 rocket launches at a cost of 62 million dollars, significantly lower than the industry average of 165 million dollars offered by other providers.

The cost-effectiveness is primarily attributed to SpaceX's ability to reuse the first stage. By accurately predicting the success of the first stage landing, we can determine the potential cost of a launch.

This information can be invaluable for companies looking to bid against SpaceX in the competitive rocket launch market.

Problems you want to find answers

- Identify the correlation between rocket launch success and various launch characteristics such as launch site and rocket type.
- Identify the optimal predictive model for determining rocket landing outcome.

Section 1

Methodology

Methodology

Executive Summary

- **Data collection methodology**
  - SpaceX API and web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled *List of Falcon 9 and Falcon Heavy launches.*
- **Perform data wrangling**
  - Augmented data with a Boolean feature that identifies the landing outcome.
- **Perform exploratory data analysis (EDA) using SQL and visualization**
  - Visualization using scatter charts and bar charts to show relationships between features to identify patterns in data.
- **Perform interactive visual analytics using Folium and Plotly Dash**
  - Folium to plot all launch sites on a map. Plotly Dash to create an interactive dashboard to visualize success rate in a pie chart and a scatter chart to show the relationship between outcome and payload for the different booster versions.
- **Perform predictive analysis using classification models**
  - Determine optimal hyperparameter for SVM, Classification Trees and Logistic Regression models.

Data Collection

The data was collected using 2 different methods:

- SpaceX API:
  - Using get request to the SpaceX API, we retrieved the data for the rockets, launchpads, payloads, cores, flight numbers and dates. Decoded the response content as a Json using json() function call and parsed it to a pandas dataframe using json_normalize(). Cleaned the data, checked for missing values and filled in missing values in payloadmass with its mean. Saved the data frame to a csv file for later analysis.
- Wikipedia web scraping:
  - Using BeautifulSoup performed web scraping from Wikipedia for Falcon 9 launch records. Extracted the launch records as HTML tables, parsed the core table and converted it to a pandas data frame. Similarl to the API data collection process, saved the data frame to a csv file for later analysis.

Data Collection – SpaceX API

https://github.com/[...]spacex-data-collection-api.ipynb

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	
89	86	2020-09-03	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3(
90	87	2020-10-06	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	3	True	True	True	5e9e3(
91	88	2020-10-18	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	6	True	True	True	5e9e3(
92	89	2020-10-24	Falcon 9	15600.0	VLEO	CCSFS SLC 40	True ASDS	3	True	True	True	5e9e3(
93	90	2020-11-05	Falcon 9	3681.0	MEO	CCSFS SLC 40	True ASDS	1	True	False	True	5e9e3(

Partial picture of the final dataframe

Request and parse the SpaceX launch data

GET request to https://api.spacexdata.com/v4/launches/past
Convert result into a Pandas dataframe using pd.json_normalize()

Request and parse detailed launch information

Using the launch ID's get detailed data for each rocket launch. For each feature create a list.
Finally create a new dataframe with the combined information

Data Wrangling - filter data

Filter on "Falcon 9" launches

Data Wrangling – identify and deal with missing values.

Replace missing "Payload Mass" values with its mean

Save dataframe to CSV file.

Data Collection - Scraping

https://github.com/[...]-webscraping.ipynb

```
extracted_row = 0
#Extract each table
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.`
            launch_dict['Flight No.'].append(flight_number)
            #print(flight_number)
            datatimelist=date_time(row[0])

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

            # Time value
            # TODO: Append the time into launch_dict with key `Time`
            time = datatimelist[1]
            launch_dict['Time'].append(time)
            #print(time)
```

Partial picture of the parsing process using a launch_dict *dictionary*

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

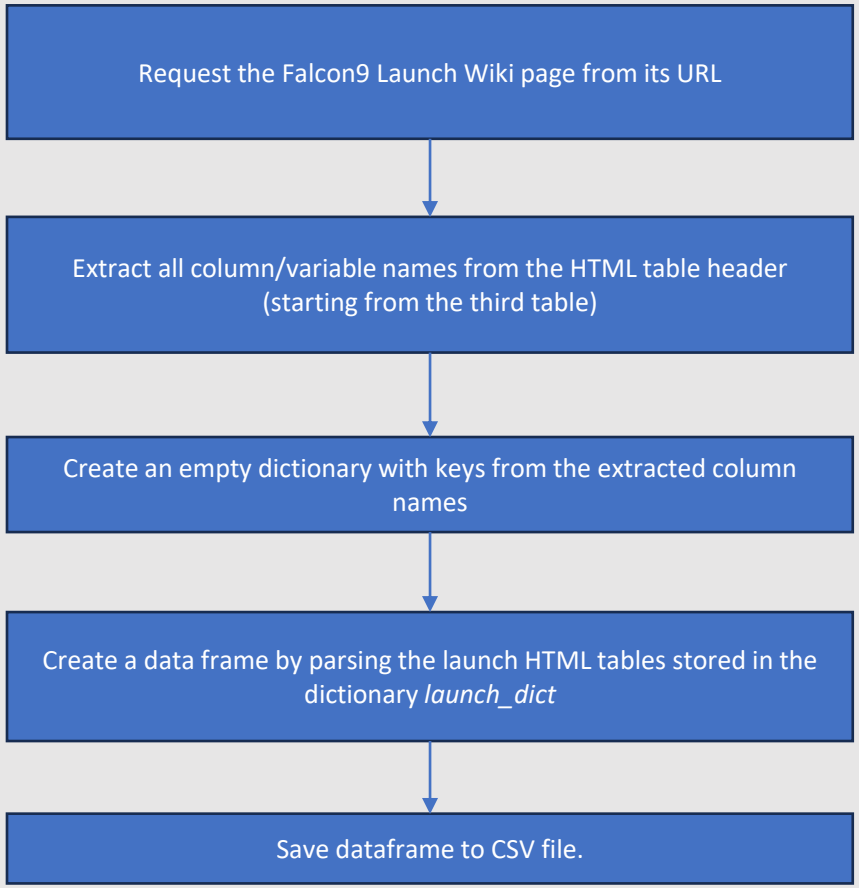
```
df.shape
```

```
(121, 11)
```

Picture of the final dataframe shape

Extract a Falcon 9 launch records HTML table from Wikipedia

Parse the table and convert it into a Pandas data frame



Data Wrangling

https://github.com/[...] spacex-Data%20wrangling.ipynb

```
df.head(5)
```

Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

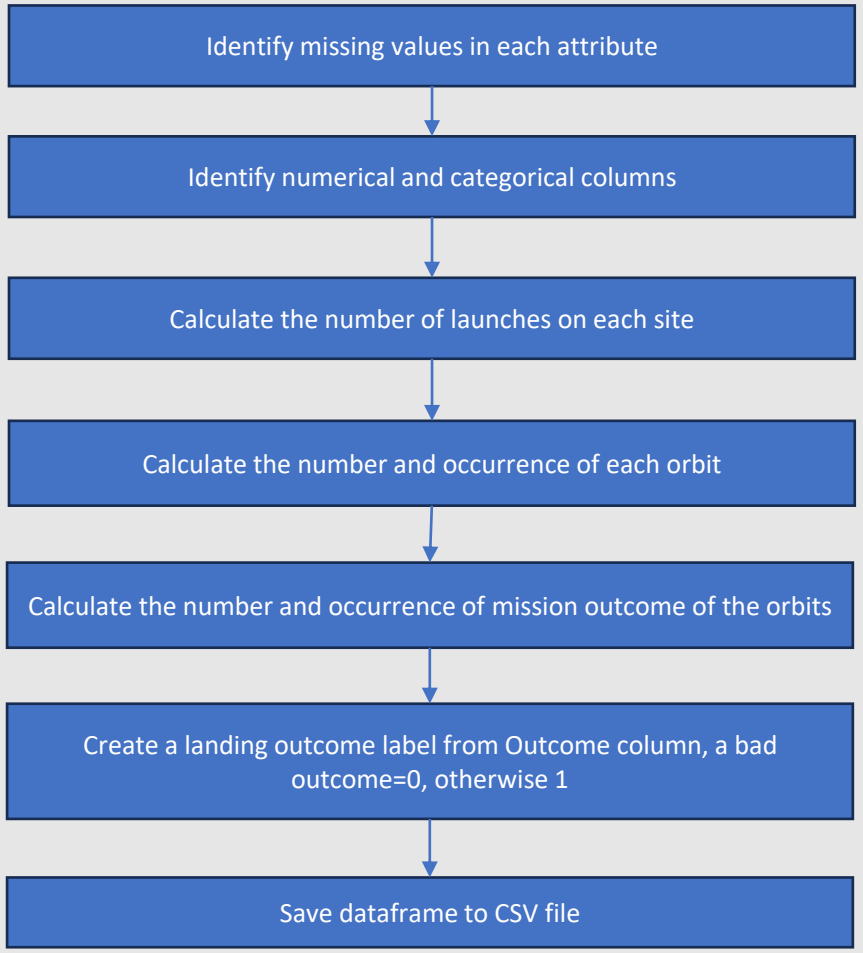
We can use the following line of code to determine the success rate:

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

```
0.6666666666666666
```

Partial picture of the final data frame with an additional outcome label named class

Data Analysis



Create a landing outcome label

EDA with SQL

https://github.com/[...] eda-sql-coursera_sqllite.ipynb

Summarize the SQL queries you performed:

- Names of unique launch sites
- 5 records where launch sites begin with the string 'CCA'
- Total payload mass carried by boosters launched by 'NASA (CRS)'
- Average payload mass carried by booster version 'F9 v1.1'
- Date when the first successful landing outcome in ground pad was achieved.
- Names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- Total number of successful and failure mission outcomes
- Names of the booster versions which have carried the maximum payload mass.
- Records which will display the month names, failure landing outcomes in drone ship, booster versions, launch site for the months in year 2015
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad) between the date 2010-06-04 and 2017-03-20, in descending order.

EDA with Data Visualization

https://github.com/[...] eda dataviz.ipynb

Summarize what charts were plotted and why you used those charts:

- Visualize how the Flight Number (indicating the continuous launch attempts.) and Payload variables would affect the launch outcome. Plot a scatter point chart 'Flight Number' vs. 'Payload Mass' and overlay the outcome of the launch.
- Visualize the relationship between Flight Number and Launch Site. Plot a scatter point chart 'Flight Number' vs. 'Launch Site', and hue to be the 'class' value (overlay of the outcome of the launch)
- Visualize the relationship between Payload Mass and Launch Site. Plot a scatter point chart 'Pay Load Mass (kg)' vs. 'Launch Site', and hue to be the 'class' value (overlay of the outcome of the launch)
- Visualize the relationship between success rate of each orbit type. Plot a bar chart 'Orbit' vs. 'Class' mean, grouped by 'Orbit'
- Visualize the relationship between Flight Number and Orbit type. Plot a scatter point chart 'Flight Number' vs. 'Orbit', and hue to be the class value
- Visualize the relationship between Payload Mass and Orbit type. Plot a scatter point chart 'Payload Mass' vs. 'Orbit', and hue to be the class value
- Visualize the launch success yearly trend. Plot a line chart 'Year' vs 'Class' sum, grouped by 'Year'

Build an Interactive Map with Folium

https://github.com/[...] site_location.ipynb

Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map:

- Using folium.Circle and folium.Marker marked each launch site on a map, to visualize their location. Labeled the site name using folium.Popup.
- Marked the success/failed launches for each site using a green or red marker
- Calculated the distances between a launch site to its proximities, using folium.PolyLine drew lines between points and labeled the distance using folium.Popup.

Explain why you added those objects:

Having identified and labeled on the map each launch sites, the outcome of each rocked launch, and the distance to notable points in their proximities, we can identity patterns and create hypothesis to test .e.g.:

- Nearest coastline: 1km
- Nearest railway: 1km
- Nearest highway: ~24 km
- Nearest airport: ~15 km

Build a Dashboard with Plotly Dash

https://github.com/[...] spacex_dash_app.py

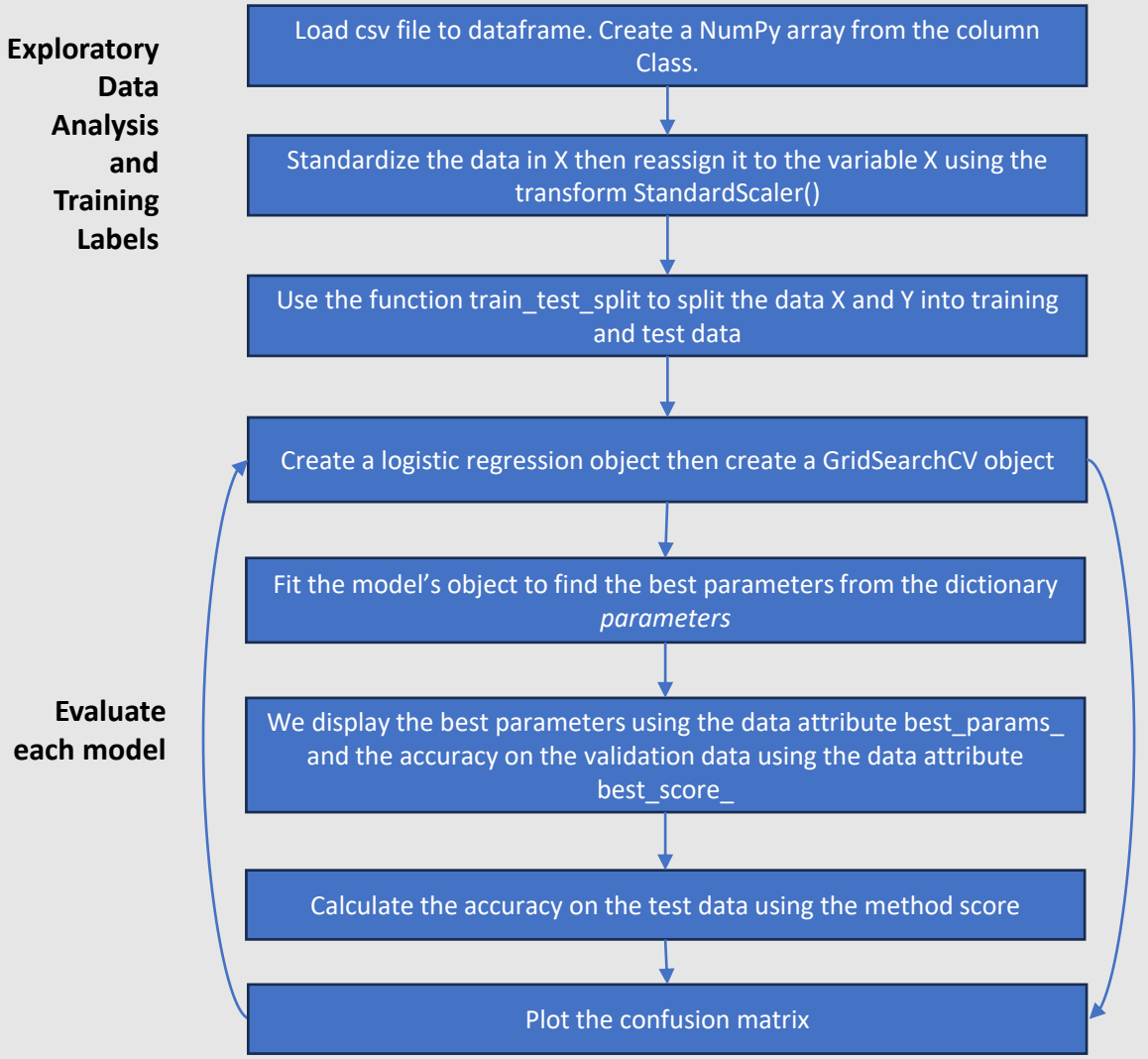
- Summarize what plots/graphs and interactions you have added to a dashboard:
  - Created a dropdown button to select ALL or a single launch site. Created a linked pie chart to show the successful launches per site (when “all” sites are selected), or to show the success/fail proportion (when a single site is selected).
  - Created a payload range selector between 0 Kg and 10,000 Kg. Created a linked scatter chart to show the relationship between outcome and payload for the different booster versions. This chart allows an interactive and dynamic analysis of relationship between outcome, payload and booster version.

Predictive Analysis (Classification)

https://github.com/[...] MachineLearningPrediction_Part_5.ipynb

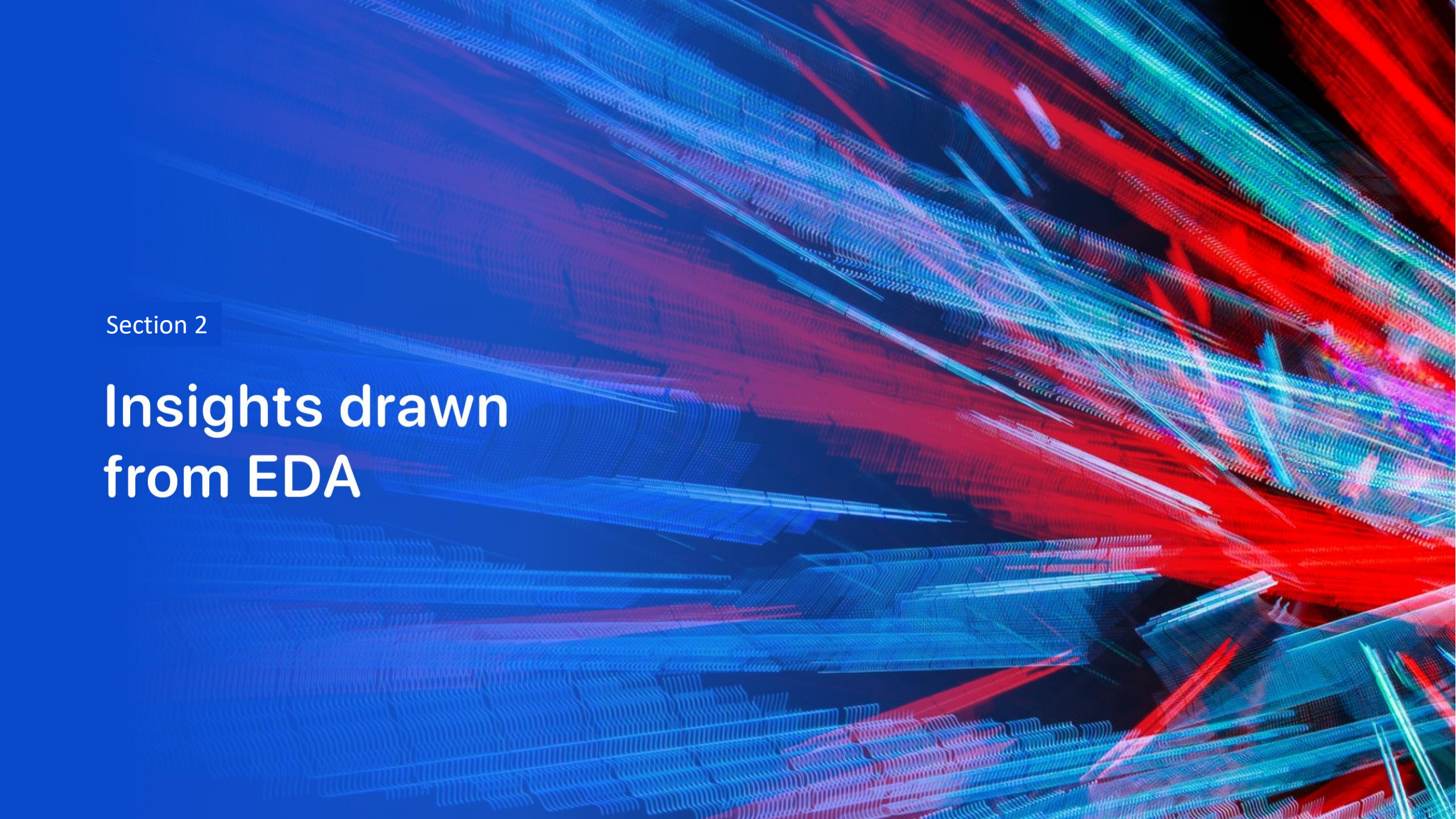
Summarize how you built, evaluated, improved, and found the best performing classification model:

- Performed exploratory Data Analysis and determined Training Labels:
  - created an array from the landing outcome “class”
  - standardized the data
  - split data into training data and test data
- Using the GridSearchCV, found the best hyperparameter for Logistic Regression, SVM, Decision Tree and K-Nearest Neighbors.
- Using the test data, applied the method score to determine the model accuracy and plotted a confusion matrix to evaluate and compare the models.



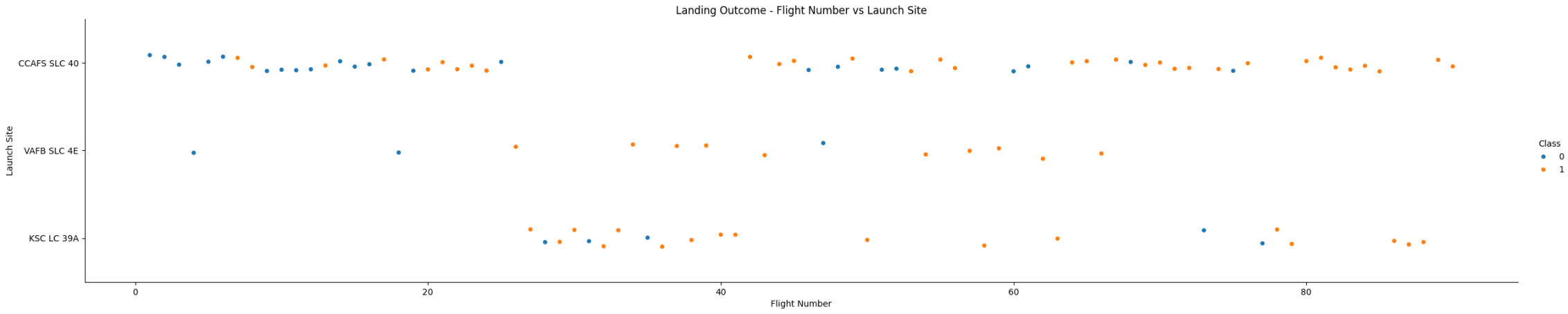
Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results


Section 2
Insights drawn from EDA

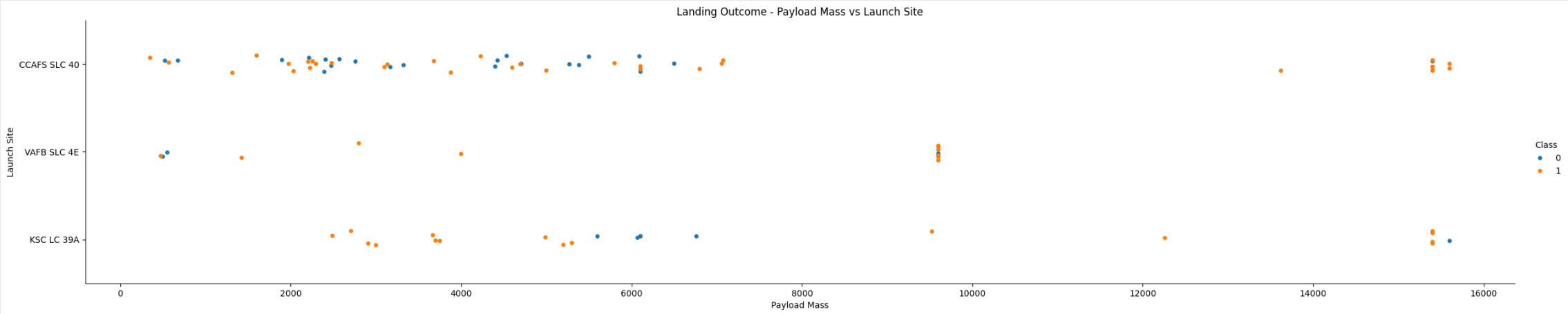
Flight Number vs. Launch Site

- With time (seen by the flight number), the success rate increased at all sites



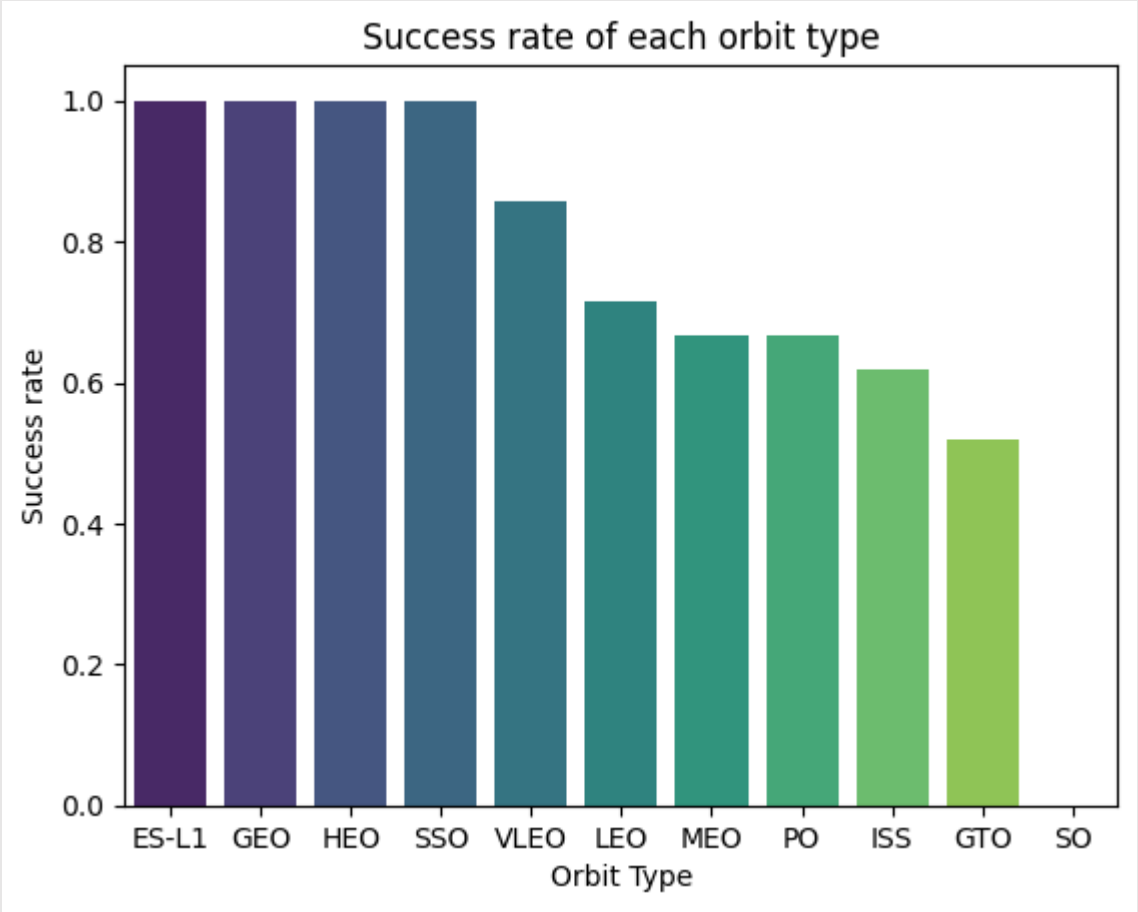
Payload vs. Launch Site

- At CCAFS SLC 40 the success rate is mixed at low payloads (less than 8000), no pattern. However, at heavy load around 15500 the success rate is very high.
- At VAFB-SLC 4E there are no rockets launched for heavy payload mass (greater than 10000). At the highest payload around 9600, the success rate is very high.
- At KSC LC 39A there is a range between ~5500 and 7000 where the success rate is null, however outside that range, the success rate is high.



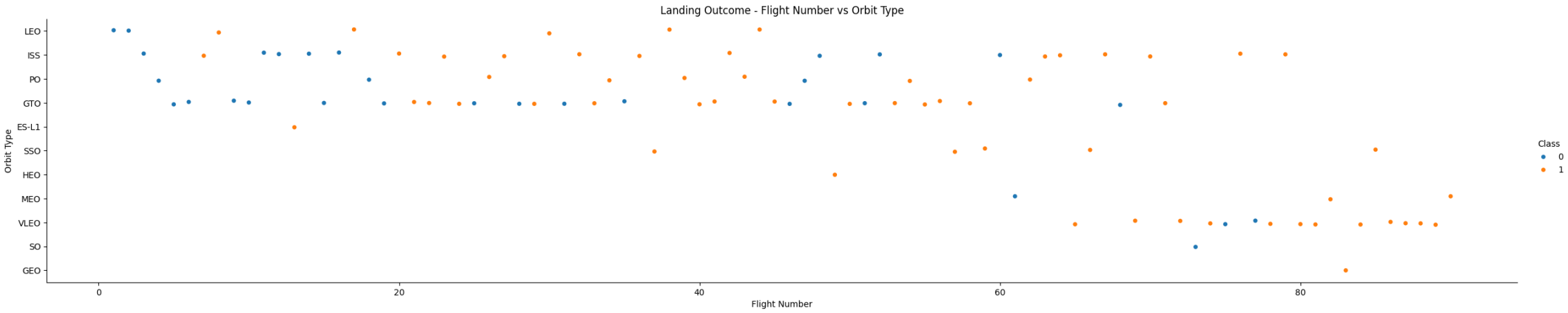
Success Rate vs. Orbit Type

- Orbits ES-L1, GEO, HEO and SSO have the highest success rate.



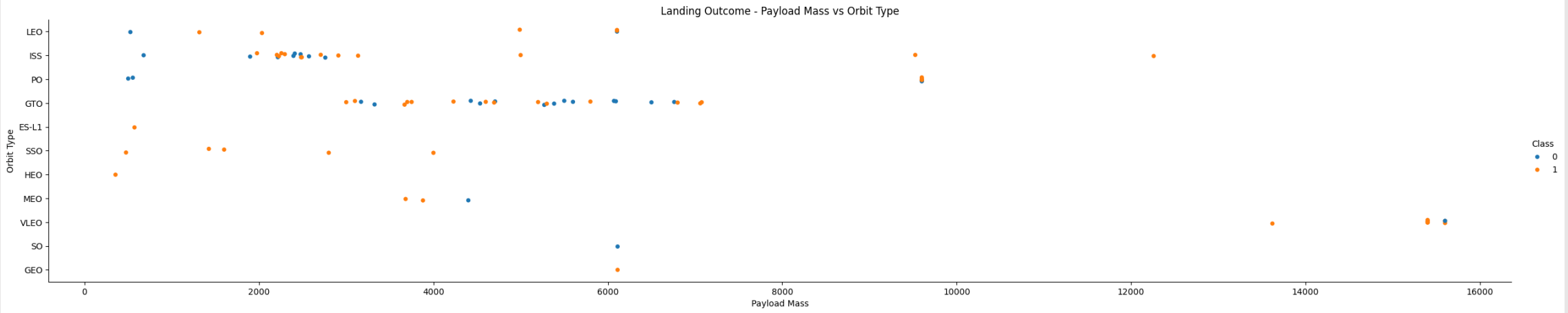
Flight Number vs. Orbit Type

- In the LEO orbit, success seems to be related to the number of flights.
- In the GTO orbit, there appears to be no relationship between flight number and success.
- SSO had few launches but is has a 100% success rate
- VLEO has a high success rate and was the preferred launch pad in the later launches.



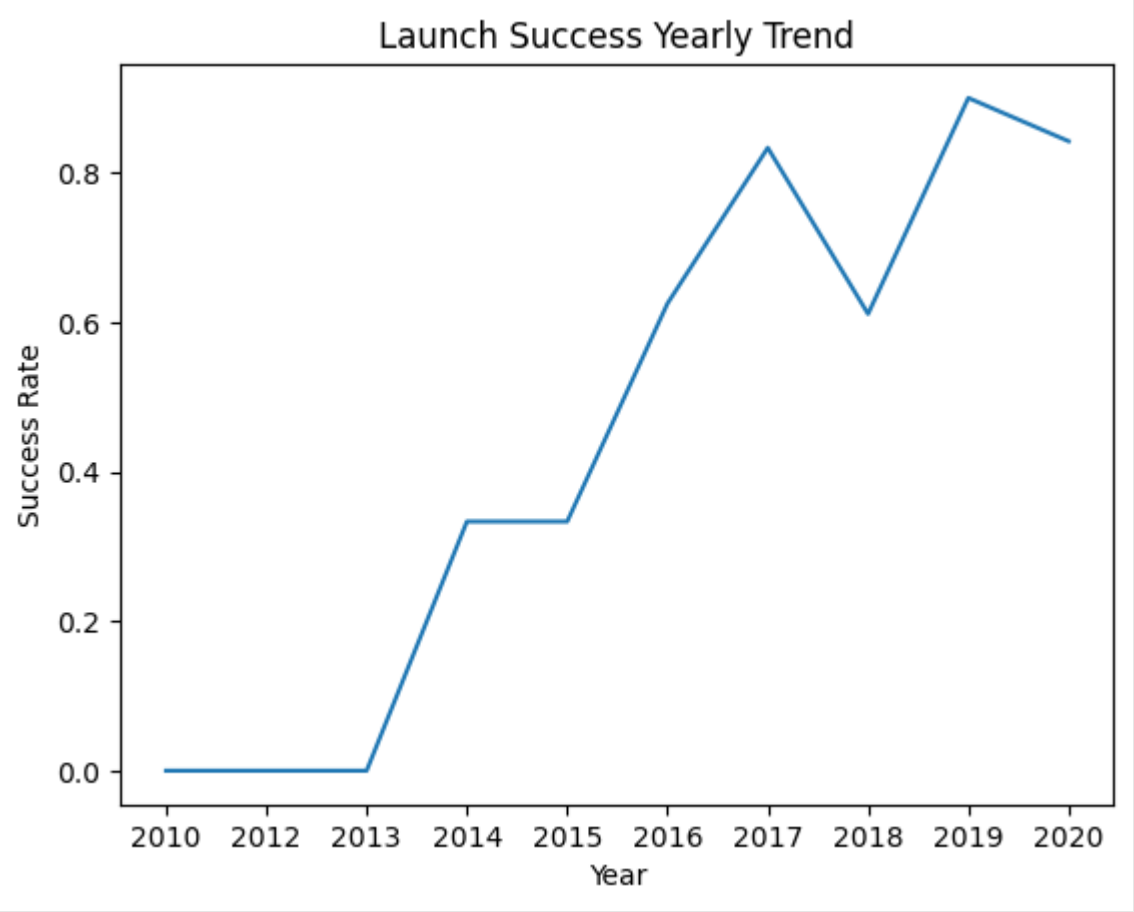
Payload vs. Orbit Type

- With heavy payloads the successful landing rate is higher for LEO, PO and ISS.
- For GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.



Launch Success Yearly Trend

- Success rate increased since 2013 until it picked in 2017
- Then the success rate decreased slightly in 2018
- In later years success rate was about 80%



All Launch Site Names

- Query:

```
select distinct Launch_Site
from SPACEXTABLE
```

- DISTINCT clause is used in the query to select only unique values

- There are four unique launch sites:

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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA` - query:

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

- Use of **LIKE 'CCA%'** ensures the **Launch_Site** begins with 'CCA'. Use of **LIMIT 5** outputs only 5 records.

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Calculate the total payload carried by boosters from NASA – query:

```
select sum("PAYLOAD_MASS__KG_") Total_Payload from SPACEXTABLE
where customer ='NASA (CRS)'
```

- Use of SUM("PAYLOAD_MASS__KG_") calculates the sum of the payload of all records WHERE customer ='NASA (CRS)' condition is true.

Total_Payload
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1- query:

```
select AVG("PAYLOAD_MASS__KG_") Avg_Payload from SPACEXTABLE
where "Booster_Version" ='F9 v1.1'
```

- Use of **AVG("PAYLOAD_MASS__KG_")** calculates the average of the payload of all records **WHERE "Booster_Version" ='F9 v1.1'** condition is true.

Avg_Payload
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad – query:

```
select min(Date) First_Landing from SPACEXTABLE
where "Landing_Outcome" ='Success (ground pad)'
```

- Use of MIN(Date) function selected the earliest date, WHERE "Landing_Outcome" ='Success (ground pad)' condition was true.

First_Landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```
select distinct Booster_Version  from SPACEXTABLE
where "Landing_Outcome" ='Success (drone ship)’ and ("PAYLOAD_MASS__KG_">4000) and
("PAYLOAD_MASS__KG_"<6000)
```

- Use of DISTINCT ensured only unique values were selected. The WHERE condition ensured that only "Landing_Outcome" ='Success (drone ship)’ were included and limited the payload to the desired range ("PAYLOAD_MASS__KG_">4000) AND ("PAYLOAD_MASS__KG_"<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

- Calculate the total number of successful and failure mission outcomes

```
select rtrim(Mission_Outcome) Mission_Outcome, count(*) Total from
SPACEXTABLE
Group by rtrim(Mission_Outcome)
```

- Combining the use of **COUNT(*)** with **GROUP BY** outputs the count of the field in the group by **Mission_Outcome** or **RTRIM(Mission_Outcome)**.
- Use of RTRIM was used to remove an extra “space” existent in one “Success “ entry

Mission_Outcome	Total
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass – query:

```
select  Booster_Version   from SPACEXTABLE
where "PAYLOAD_MASS__KG_" = (select max("PAYLOAD_MASS__KG_") from SPACEXTABLE)
```

- This is a 2-step query performed using a sub-query:

```
SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE
```

This sub-query retrieves the maximum value of the payload. The result is used as input for the WHERE condition on the main query:

```
WHERE "PAYLOAD_MASS__KG_" =
```

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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
select  substr(Date, 6,2) Month, Landing_Outcome, Booster_Version ,Launch_Site from SPACEXTABLE
where substr(Date,0,5)='2015' and "Landing_Outcome"='Failure (drone ship)'
```

- The function **SUBSTR(Date, 6,2)** retrieves the month from the **Date** field.
- Similarly, in the **WHERE** condition, using **SUBSTR(Date,0,5)='2015'** ensures we only select the year 2015. Finally, we ensure **"Landing_Outcome"='Failure (drone ship)'**

Month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- 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 – query:

```
select  Landing_Outcome, count(Landing_Outcome) Total from SPACEXTABLE
where Date>'2010-06-04' and Date<'2017-03-20'
group by Landing_Outcome
order by Total desc
```

- **COUNT(Landing_Outcome) Total** defines the aggregation function and the field used, 'Total' is the name of the new column.
- Condition **WHERE Date>'2010-06-04' AND Date<'2017-03-20'** selects only results in the data range desired.
- **GROUP BY Landing_Outcome** allows the aggregation of the count results by **Landing_Outcome**
- **ORDER BY Total DESC** lists results in descending order of the count "Total"

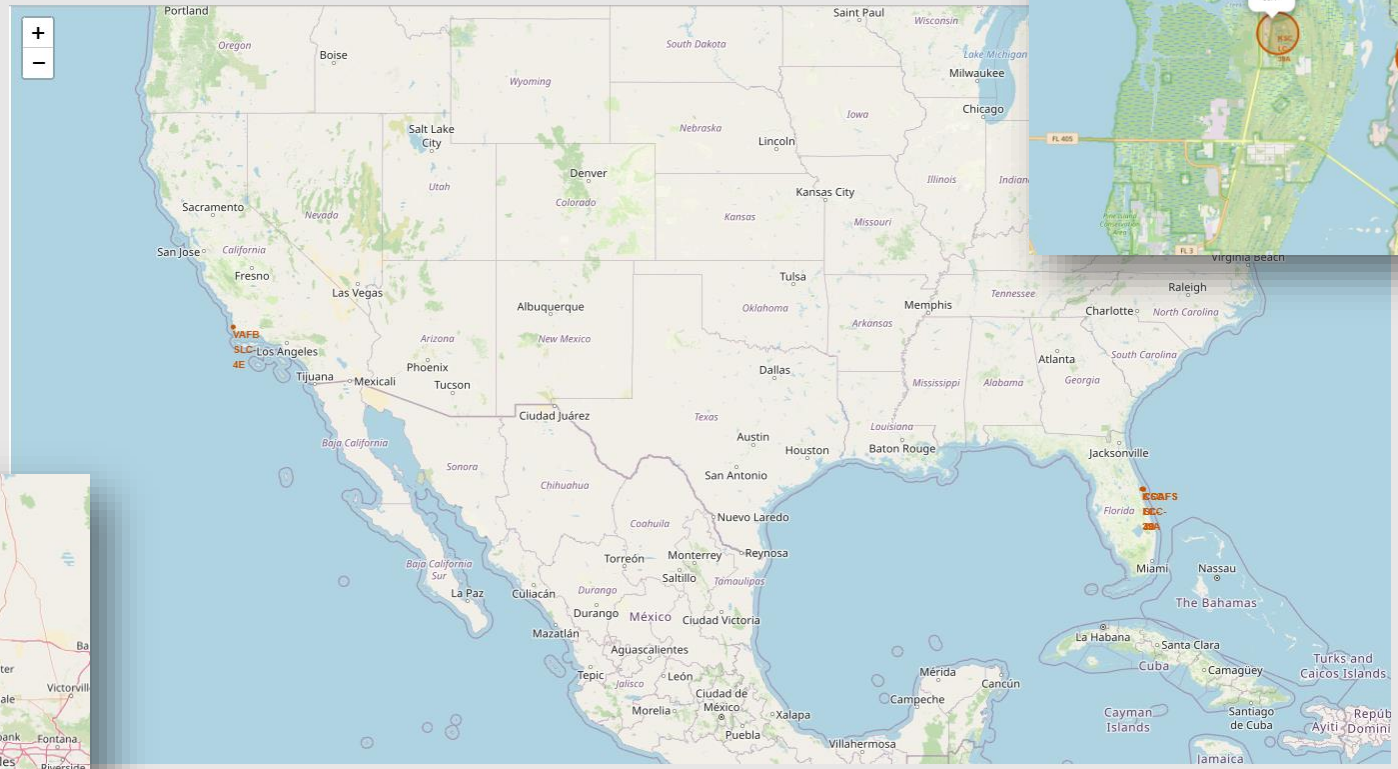
Landing_Outcome	Total
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

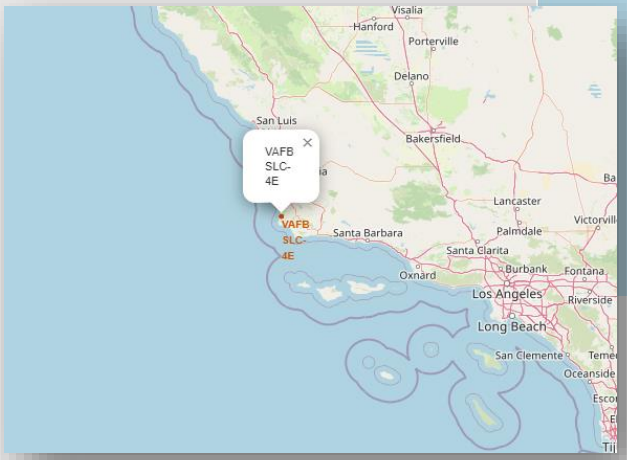
Section 3

Launch Sites Proximities Analysis

Launch Site Locations

- Map shows the locations of the four launch sites
- All launch sites are in the United States of America, nearby the coast.
- VAFB SLC-4E is located in California
- The other three sites are located in Florida, in close proximity.

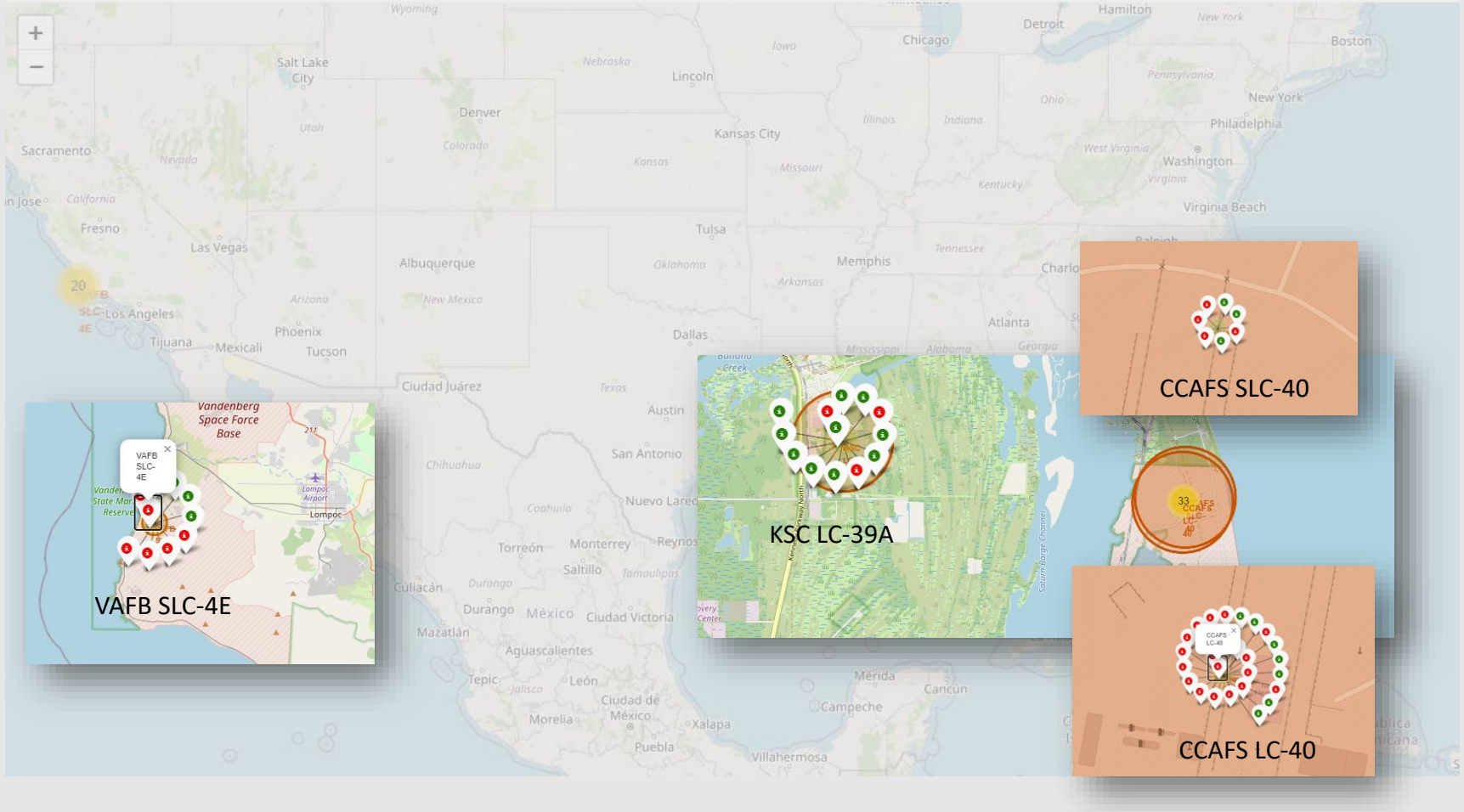




Launch Outcomes on a map

Map shows the 4 launch sites with a colored marker for each launch/outcome:

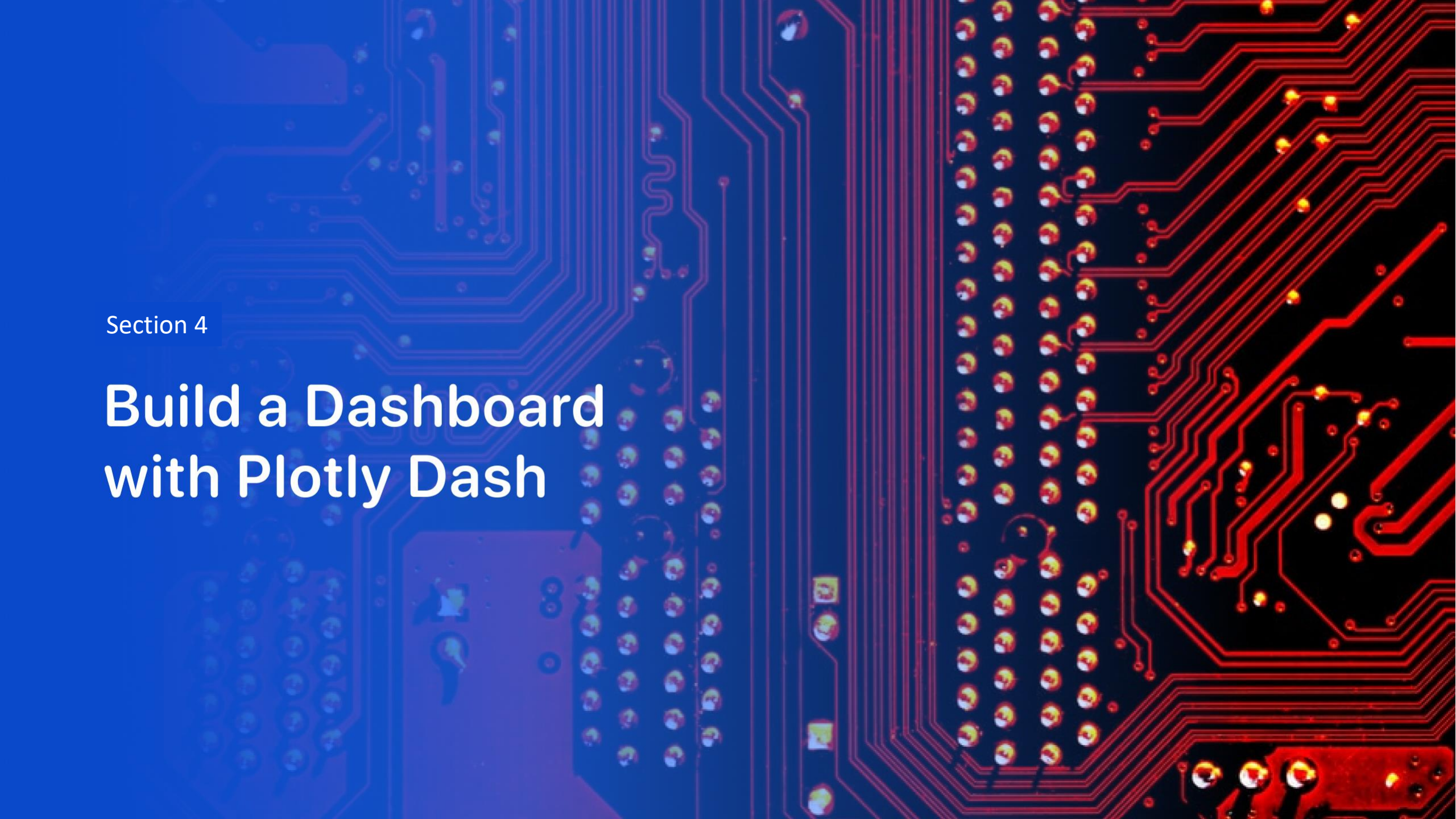
- Green indicate a success launch
- Red indicates a failed launch



Exploring Launch Sites proximities

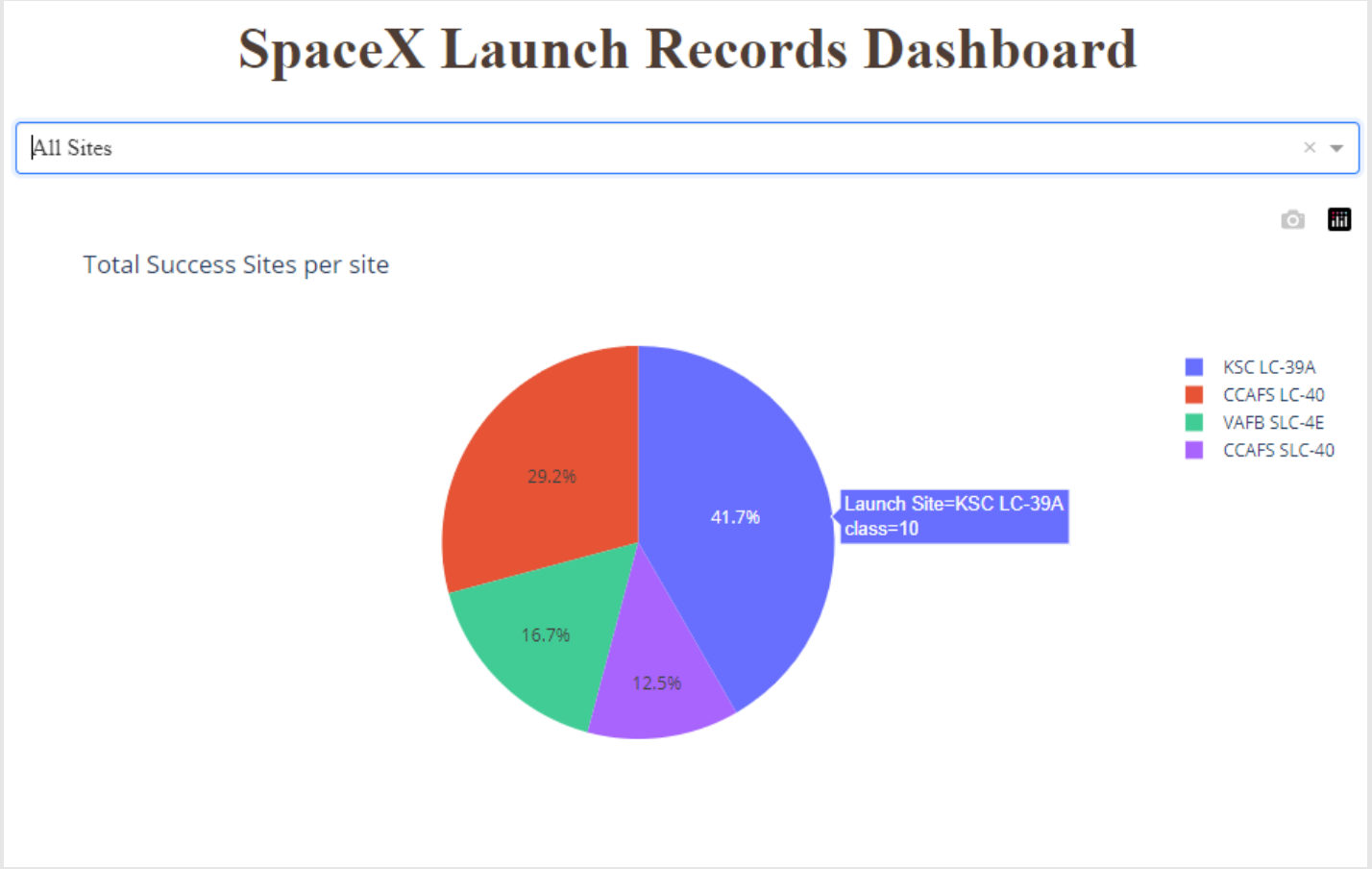
Exploring the CCAFS SLC-40 launch site:

- it is close to railways and highways, relevant for transportation of equipment and personnel,
- it is close to coastline and relatively far from cities, important for safety reasons.


Section 4
Build a Dashboard with Plotly Dash

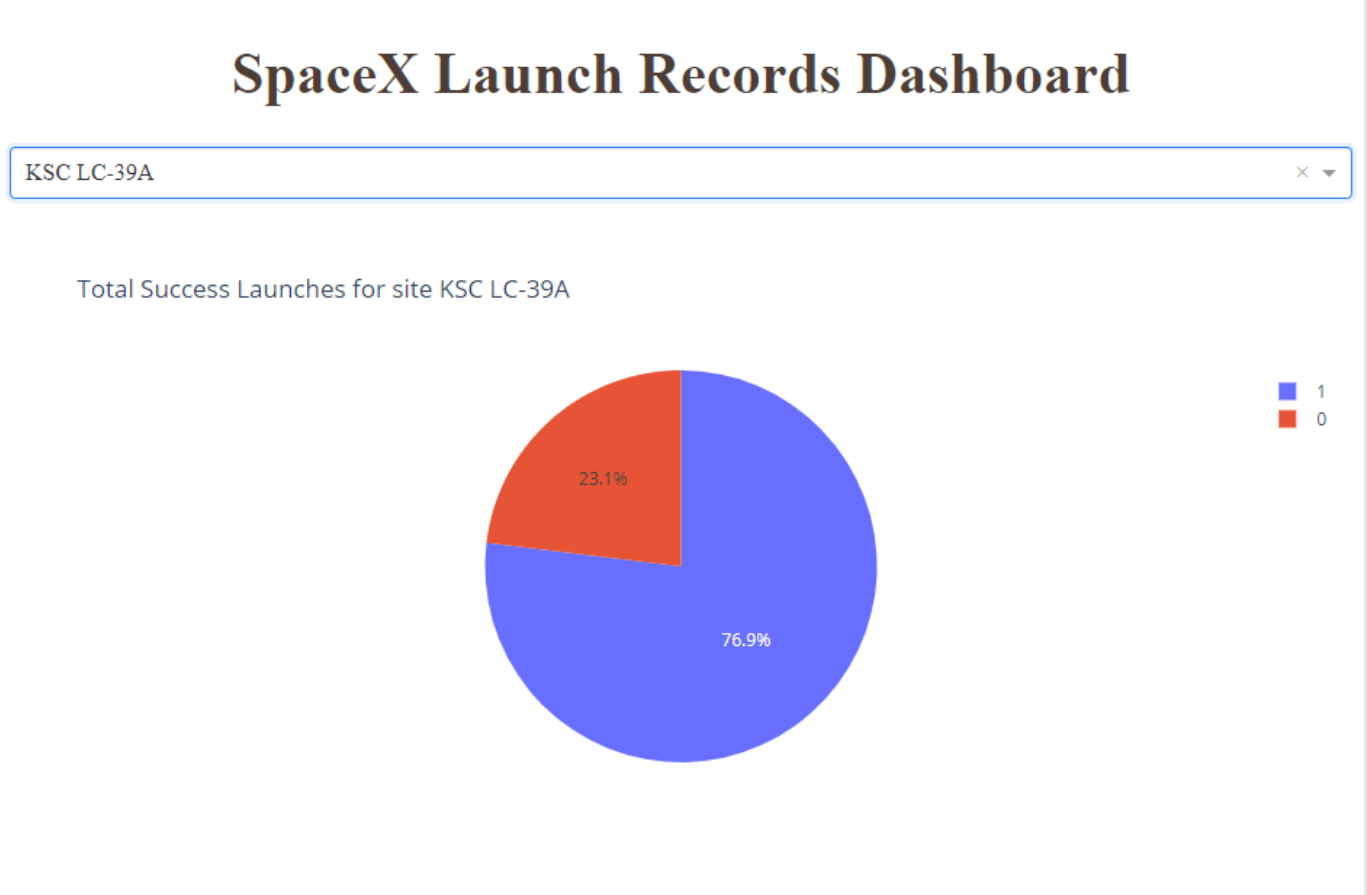
Total Success Launches per launch site

- 41.7% of the success launches (10) occurred at the KSC LC-39A
- CCAFS SLC-40 has the fewest success launches



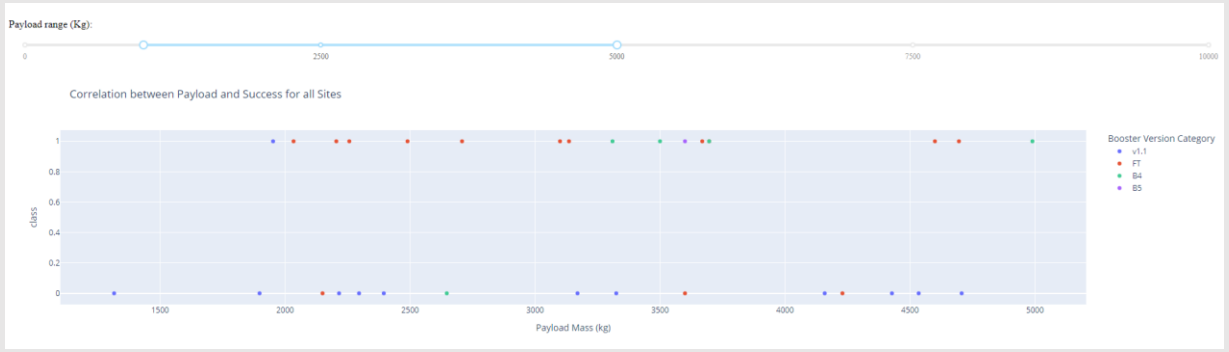
Launch site with highest launch success ratio

- KSLC-39A has the highest success rate with 10 success launches (76.9%) and 3 failed launches (23.1%).

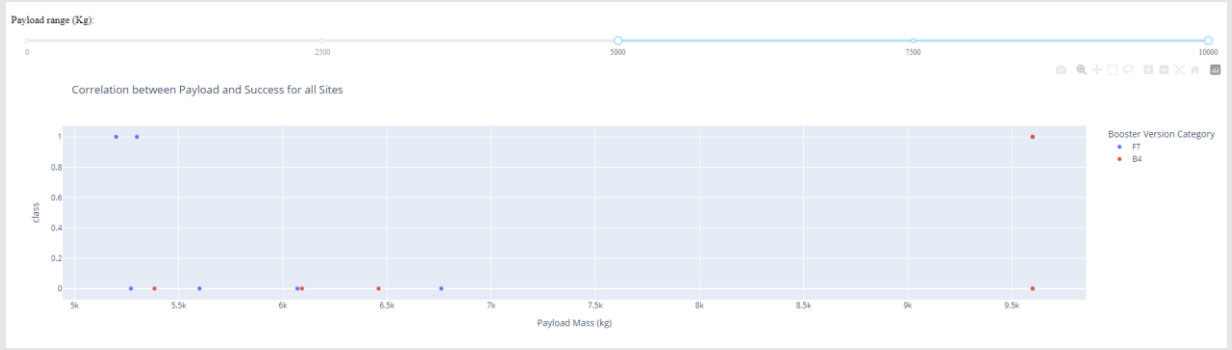


Payload vs. Launch Outcome

- The FT booster has the largest success rate with payloads up to 5000kg.
- Launch success rates decrease with payload
- Newer versions of the booster have higher success rates



- No booster performed well at heavy payloads.

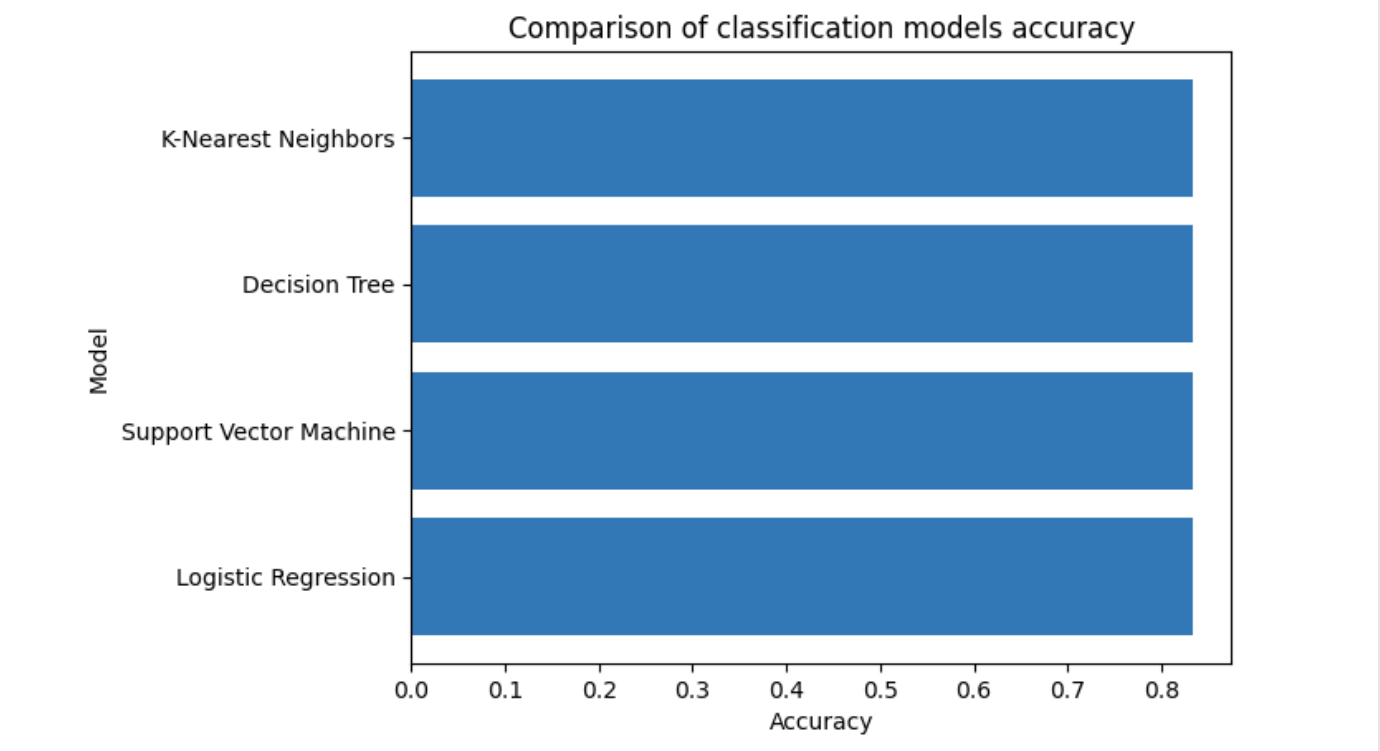


Section 5

Predictive Analysis (Classification)

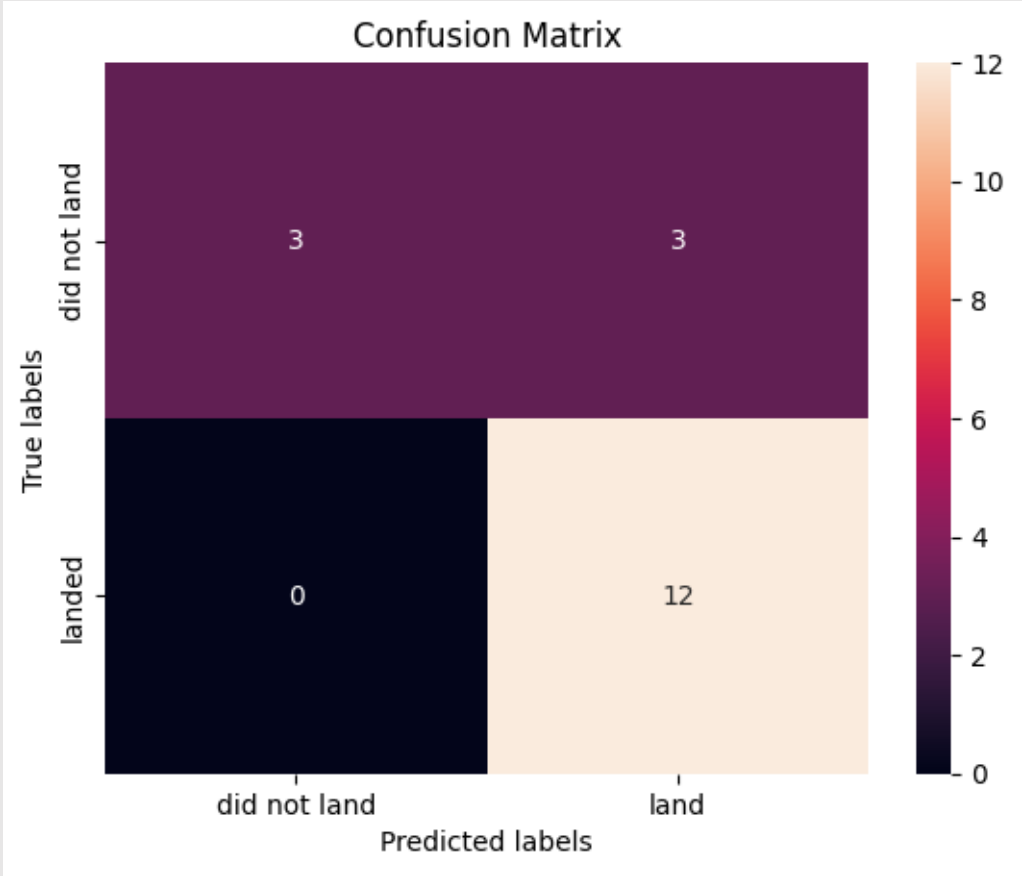
Classification Accuracy

- All models analyzed have the same accuracy with test data, 83.33%.



Confusion Matrix

- All models have the same confusion matri.x
- The models can distinguish between the different classes.
- The problem are false positives:
  - False Positive - 3 (True label is not landed, Predicted label is landed)
  - True Positive - 12 (True label is landed, Predicted label is also landed)



Conclusions

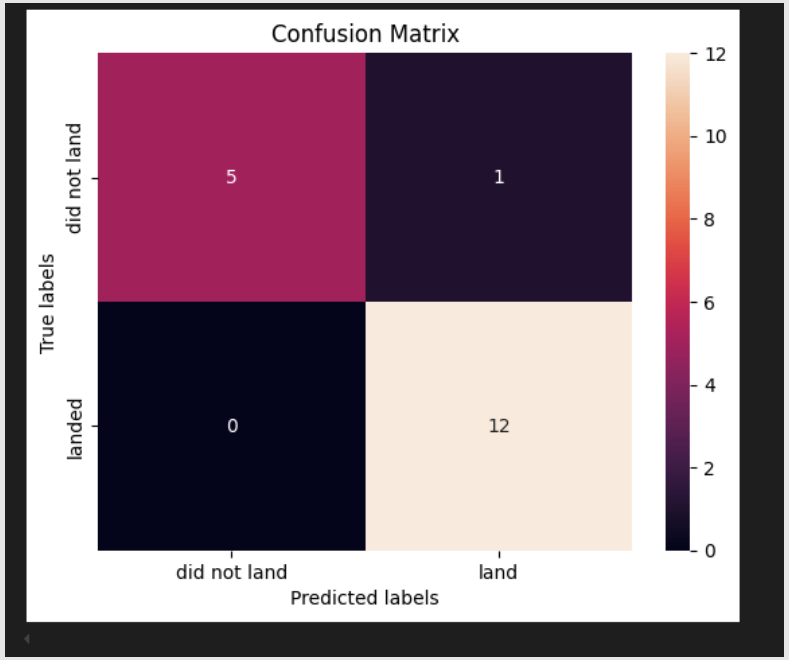
- All classification models analyzed have the same accuracy of 83.33% and the same confusion matrix, the problem are the problem is false positives, which considering the subject of the analyze would have a significant impact on costs.
- Lower then 5000kg payloads have considerable better success rate.
- Newer booster versions have higher success rates with all payloads.
- KSC LC-39A in Florida, has the largest success rates of all 4 sites utilized.
- Orbits: SSO had few launches but is has a 100% success rate. VLEO has a high success rate and was the preferred launch pad in the later launches.

Appendix

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
accuracy : 0.7464285714285713
```

```
accuracy = tree_cv.score(X_test, Y_test)
print("Accuracy on test data:", accuracy)
```

```
Accuracy on test data: 0.9444444444444444
```



- Tuning the hyperparameters of the decision tree model it was possible to achieve an accuracy of 94% and an improved confusion matrix.
- However, the results were not consistent, and the accuracy varies on each run.

Appendix

Tuning the hyperparameters

[60]:
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
accuracy : 0.875
```

TASK 9

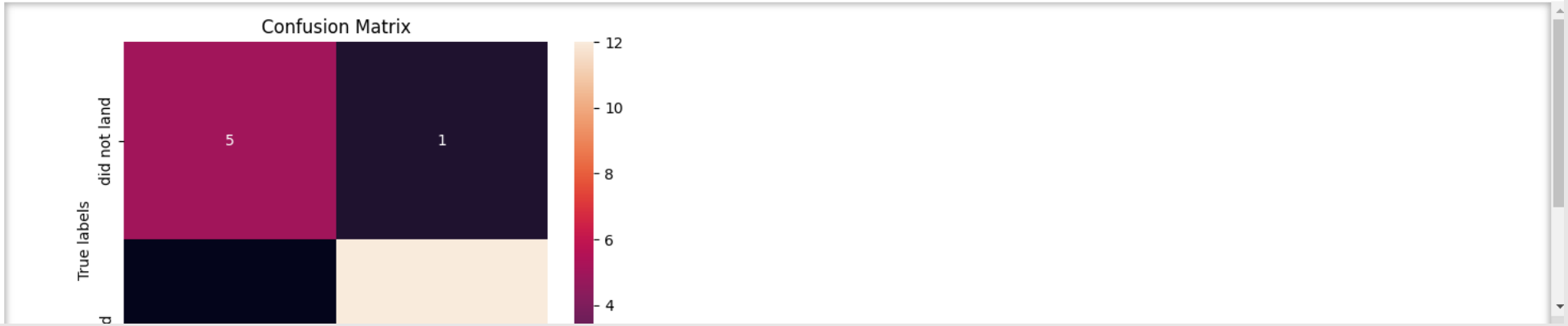
Calculate the accuracy of tree_cv on the test data using the method `score` :

[61]:
```
accuracy = tree_cv.score(X_test, Y_test)
print("Accuracy on test data:", accuracy)
```

```
Accuracy on test data: 0.9444444444444444
```

We can plot the confusion matrix

[62]:
```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Thank you!