IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Pradipta Nandi
26/12/2023

Outline

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

Executive Summary

- Summary of methodologies
  - SpaceX Data Collection using SpaceX API
  - SpaceX Data Collection with Web Scaping
  - SpaceX Data Wrangling
  - SpaceX Exploratory Data Analysis using SQL
  - SpaceX EDA DataViz using Python Pandas and Matplotlib
  - SpaceX Launch Sites Analysis with Folium-Interactive Visual Analytics and Plotly Dash
  - SpaceX Machine Learning Landing Prediction
- Summary of all results
  - EDA results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis (Classification)

Introduction

- Project background and context

  Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; 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 space X for a rocket launch. In this lab, you will create a machine learning pipeline to predict if the first stage will land given the data from the preceding labs.

- Problems you want to find answers

  In this capstone, we will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

Section 1

Methodology

Methodology

Executive Summary

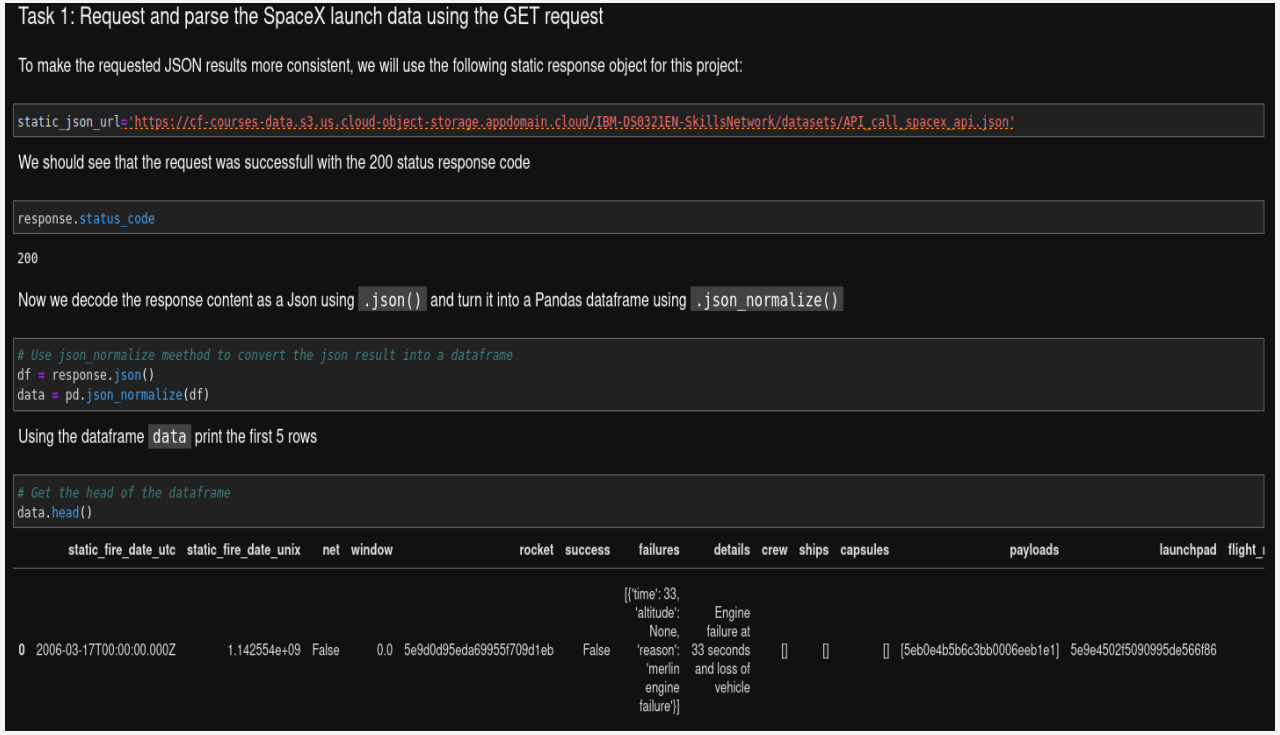
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
  - Data was first collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API. This was done by first defining a series helper functions that would help in the use of the API to extract information using identification numbers in the launch data and then requesting rocket launch data from the SpaceX API URL.
  - Finally, to make the requested JSON results more consistent, the SpaceX launch data was requested and parsed using the GET request and then decoded the response content as a JSON result which was then converted into a Pandas data frame.
  - Also performed web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy Launches of the launch records are stored in a HTML. Using BeautifulSoup and request Libraries, I extract the Falcon 9 launch HTML table records from the Wikipedia page, Parsed the table and converted it into a Pandas data frame.

Data Collection – SpaceX API

- Data collected using SpaceX API (a RESTful API) by making a GET request to the SpaceX API then requested and parsed the SpaceX launch data using the GET request and decoded the response content as a JSON result which was then converted into a Pandas data frame.
- GitHub Link



Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

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

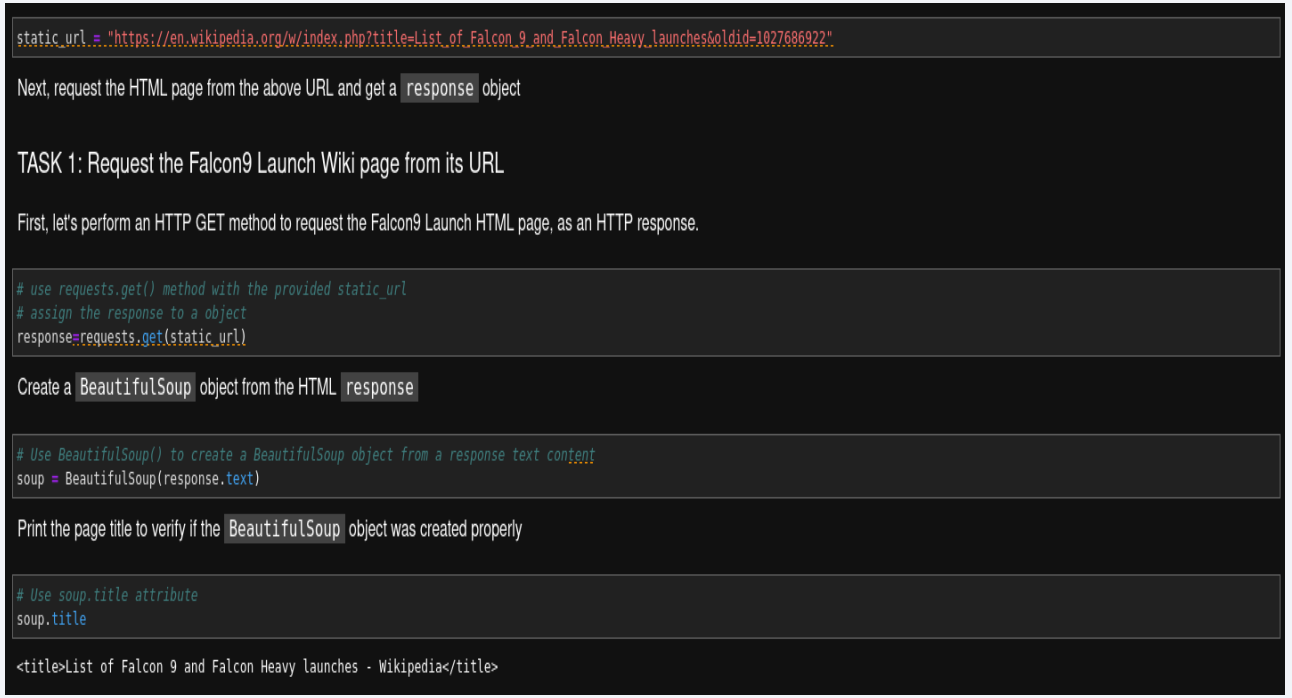
Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head()
```

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules	payloads	launchpad	flight_
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]	[5eb0e4b5b6c3bb0006eeb1e1]	5e9e4502f5090995de566f86	

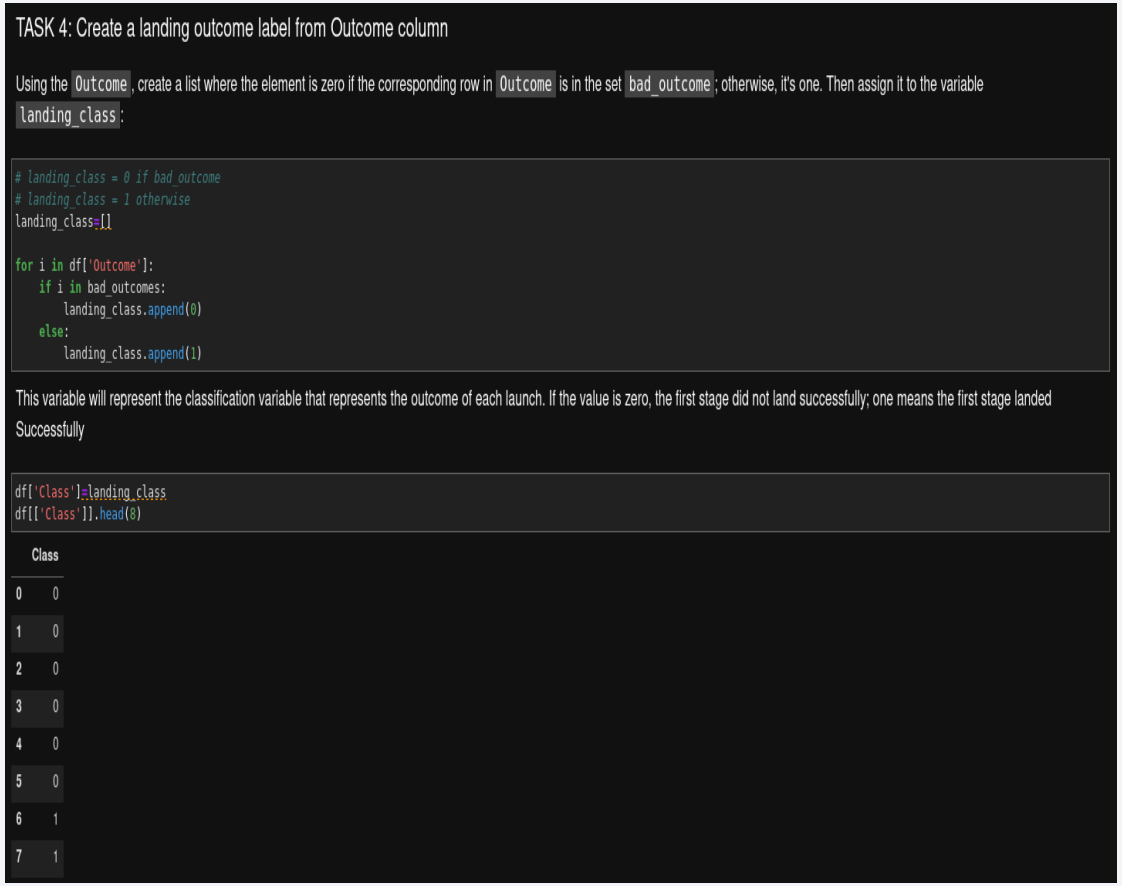
Data Collection - Scraping

- Performed web scraping to collect Falcon 9 historical launch records from a Wikipedia using BeautifulSoup and request, to extract the Falcon 9 launch records from HTML table of the Wikipedia page, then created a data frame by parsing the launch HTML.
- GitHub Link



Data Wrangling

- After obtaining and creating a Pandas data frame from the collected data, data was filtered using the **BoosterVersion** column to only keep the Falcon 9 launches, then dealt with the missing data values in the **LandingPad** and **PayloadMass** columns. For the **PayloadMass**, missing data values were replaced using the mean value of column.
- Also performed Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
- GitHub Link



TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class=[]

for i in df['Outcome']:
    if i in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

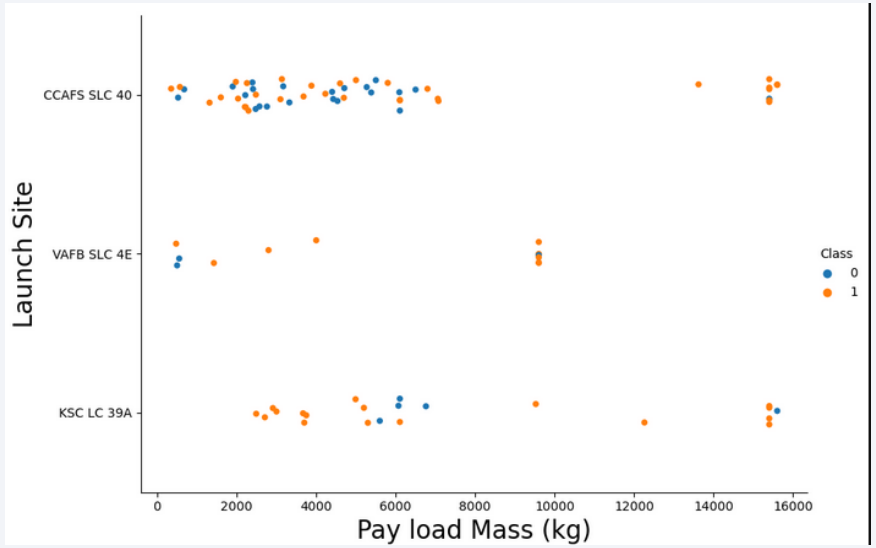
```
df['Class']=landing_class
df[['Class']].head(8)
```

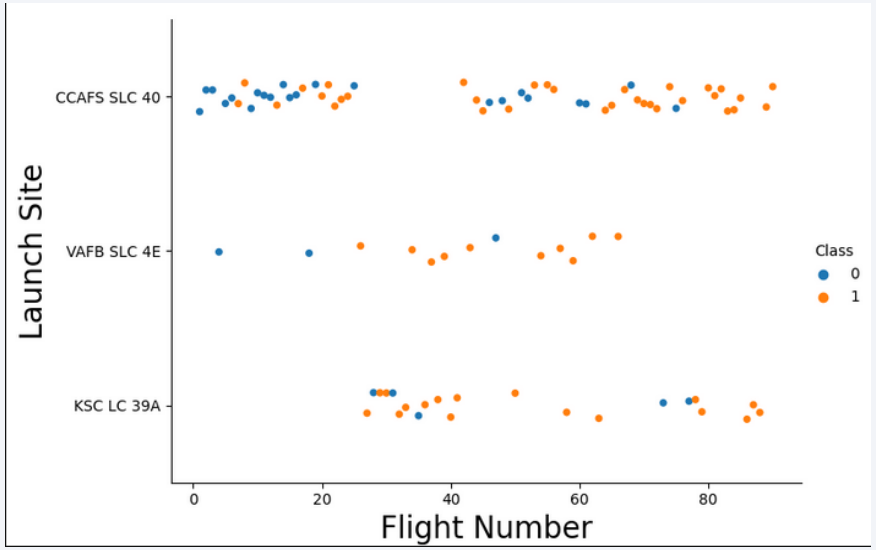
	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

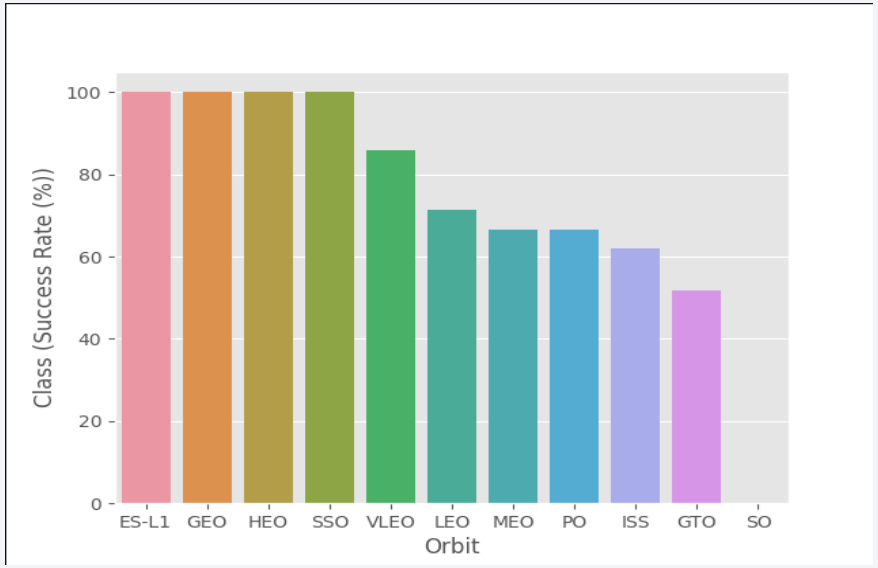
EDA with Data Visualization

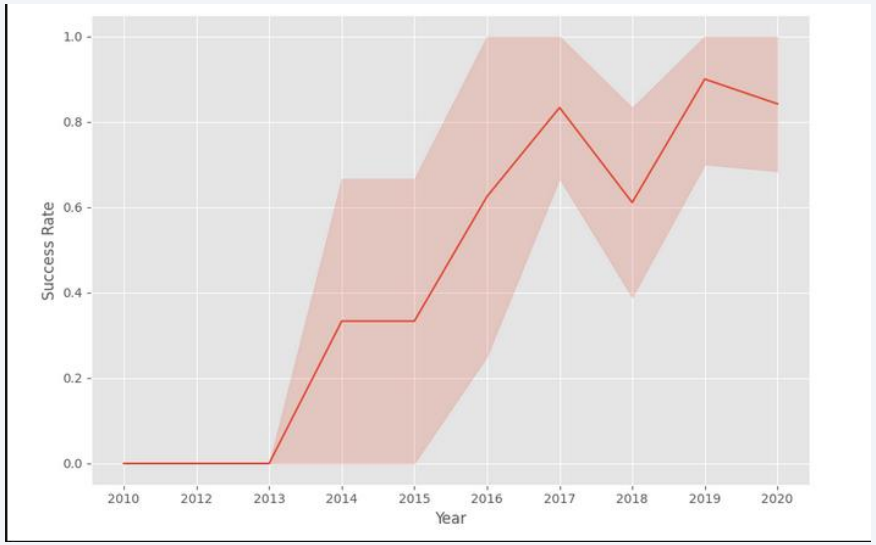
- Performed Data Analysis and Feature Engineering using Pandas and Matplotlib i.e.
  - Exploratory Data Analysis
  - Preparing Data Feature Engineering
- Used scatter plots to Visualize the relationship between Flight Number and Launch Site, Payload and Launch Site, FlightNumber and Orbit Type, Payload and Orbit type.
- Used Bar Chart to visualize the relationship between success rate of each orbit type.
- Line plot to visualize the launch success yearly trend.
- GitHub Link

EDA with Data Visualization (Plots contd...)









EDA with SQL

- The following SQL queries were performed for EDA
  - Display the names of the unique launch sites in the space mission

```
%sql select distinct(Launch_Site) from SPACEXTBL;
```

  - Display 5 records where launch sites begin with the string 'CCA'

```
%sql SELECT LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

  - Display the total payload mass carried by boosters launched by NASA (CRS)

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL
```

  - Display average payload mass carried by booster version F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL;
```

EDA with SQL

- List the date when the first successful landing outcome in ground pad was achieved

```
%sql select min(Date) from spacextbl;
```

- List the names of the boosters which success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000;
```

- List the total number of successful and failure mission outcomes

```
%sql select count(Mission_Outcome) from SPACEXTBL group by Mission_Outcome;
```

GitHub Link

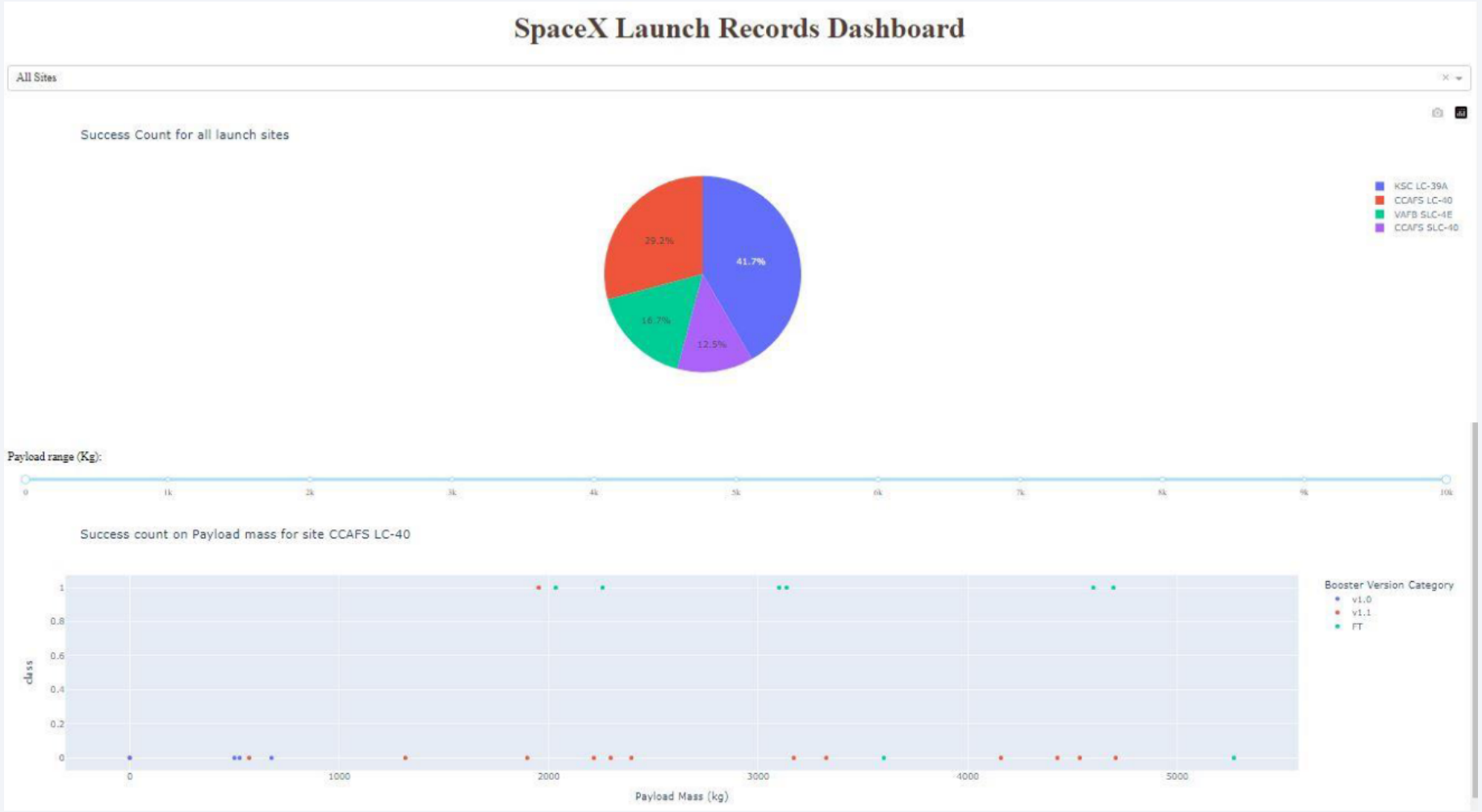
Build an Interactive Map with Folium

- Created folium map to marked all the launch sites, and created map objects such as markers, circles, lines to mark the success or failure of launches for each launch site.
- Created a launch set outcomes (failure=0 or success=1)
- GitHub Link

Build a Dashboard with Plotly Dash

- Built an interactive dashboard application with Plotly dash by:
  - Adding a Launch Site Drop-down Input component
  - Adding a callback function to render success-pie-chart based on selected site dropdown
  - Adding a Range Slider to 'Select Payload'
  - Adding a callback function to render the success-payload-scatter-chart scatter plot
- GitHub Link

SpaceX Dash App



Predictive Analysis (Classification)

- Summary of how I built, evaluated, improved, and found the best performing classification model
- After loading the data as a Pandas Data frame, I set out to perform exploratory Data Analysis and determine Training Labels by:
  - Creating a NumPy array from the column Class in data, by applying the method to_numpy() then assigned it to the variable Y as the outcome variable.
  - Then standardized the feature dataset (x) by transforming it using preprocessing.StandardScaler() function from Sklearn.
  - After which the data was split into training and testing sets using the function train_test_split from sklearn.model_selection with the test_size parameter set to 0.2 and random_state to 2.

Predictive Analysis (Classification)

- In order to find the best ML model/method that would perform best using the test data between SVM, Classification Trees, k nearest neighbors and Logistic Regression:
  - First created an object for each of the algorithms then created a GridSearchCV object and assigned them a set of parameters for each model.
  - For each of the models under evaluation, the GridSearchCV object was created with cv=10, then fit the training data into the GridSearchCV object for each to Dinf Best Hyperparameter.
  - After fitting the training set, we output GridSearchCV object for each of the models, then displayed the best parameters using the data attributes best_params_ and the accuracy on the validation data using the data attribute best_score.
  - Finally using the method score to calculate the accuracy on the test data for each model and plotted a confusion matrix for each using the test and predicted outcomes.

Predictive Analysis (Classification)

- The table below shoes the test data accuracy score for each of the methods comparing them to show which performed best using the test data between SVM, Classification Trees, k nearest neighbors and Logistic Regression:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

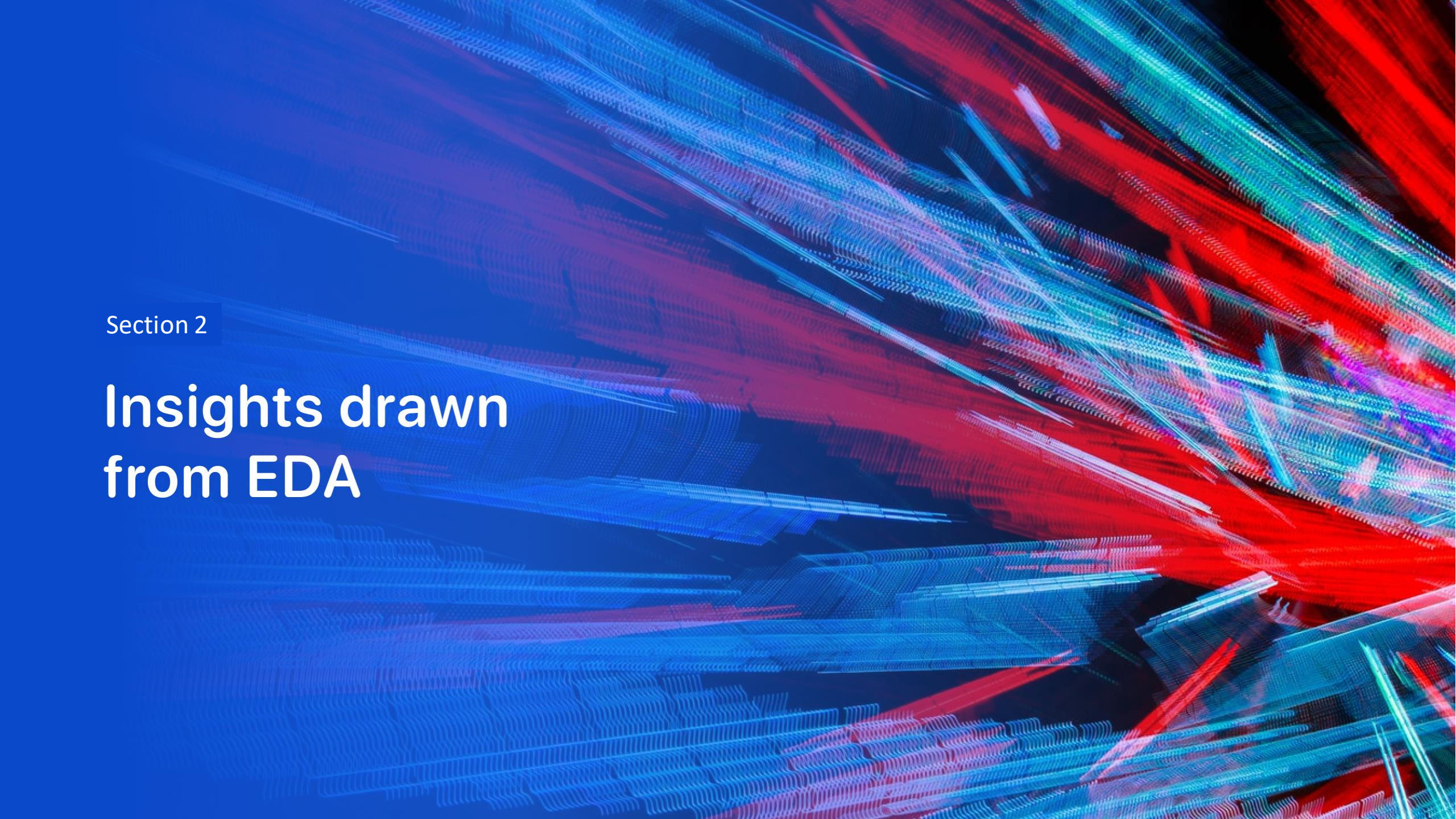
- GitHub Link

Results

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

Section 2

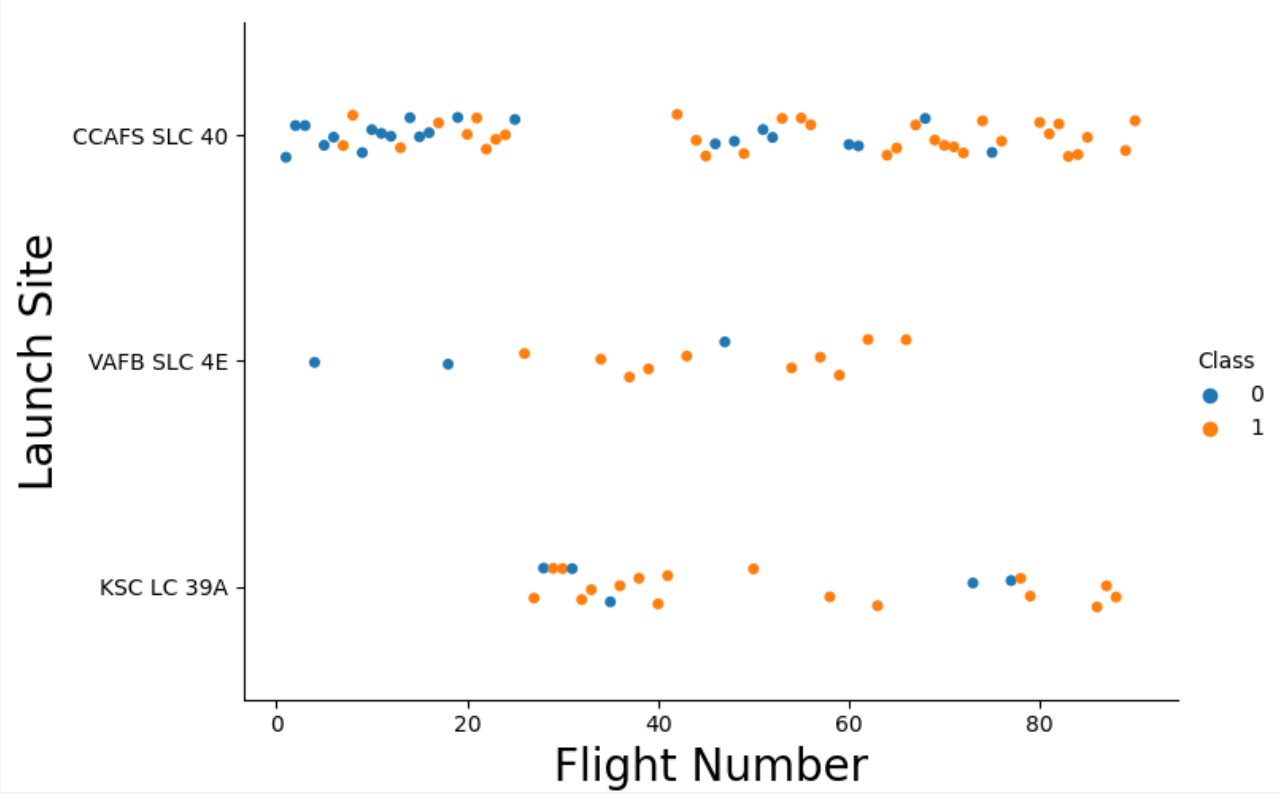
Insights drawn from EDA



Flight Number vs. Launch Site

Scatter Plot

Flight Number vs. Launch Site

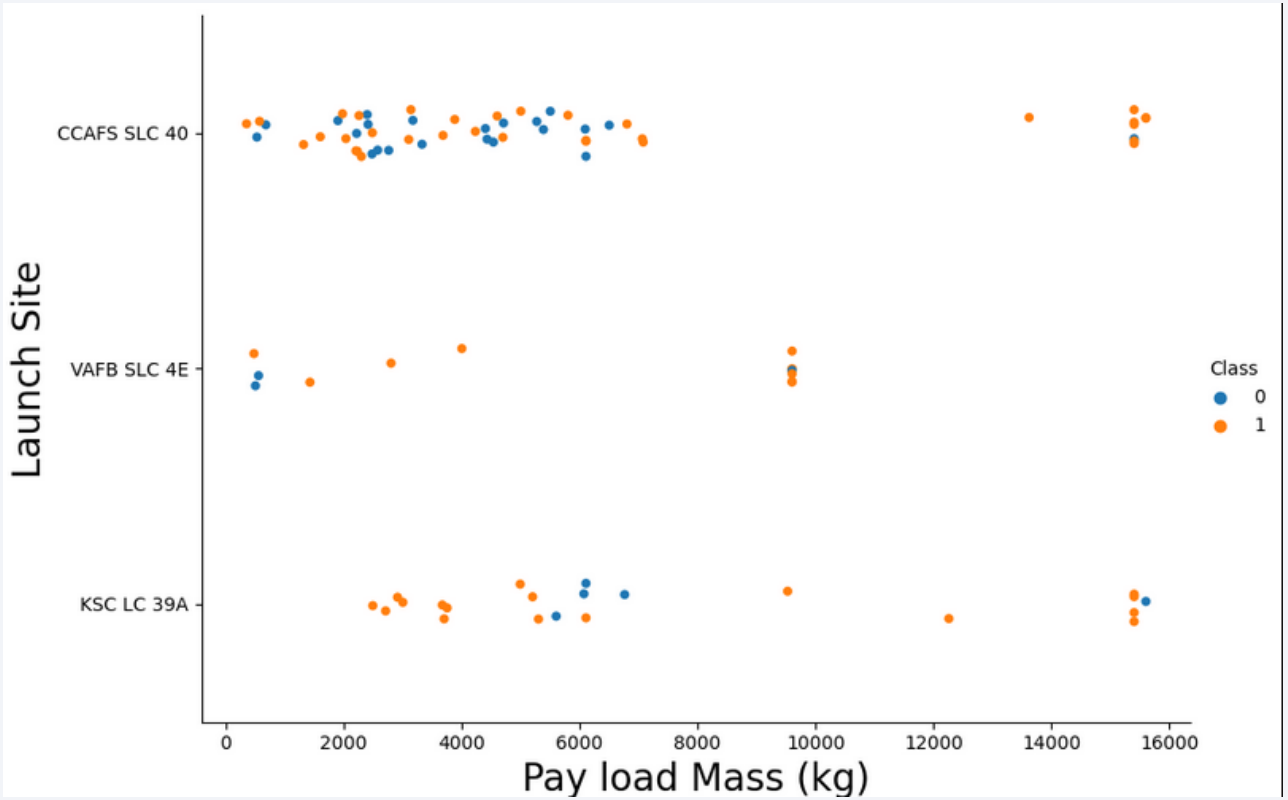


We can deduce that, as the flight number increases in each of the 3 launch sites, so does the success rate. For instance, the success rate for the VAFB SLC 4E launch site is 100% after the Flight number 50. Both KSC LC 39A and CCAFS SLC 40 have a 100% success rates after 80th flight.

Payload vs. Launch Site

Scatter Plot

Payload vs. Launch Site

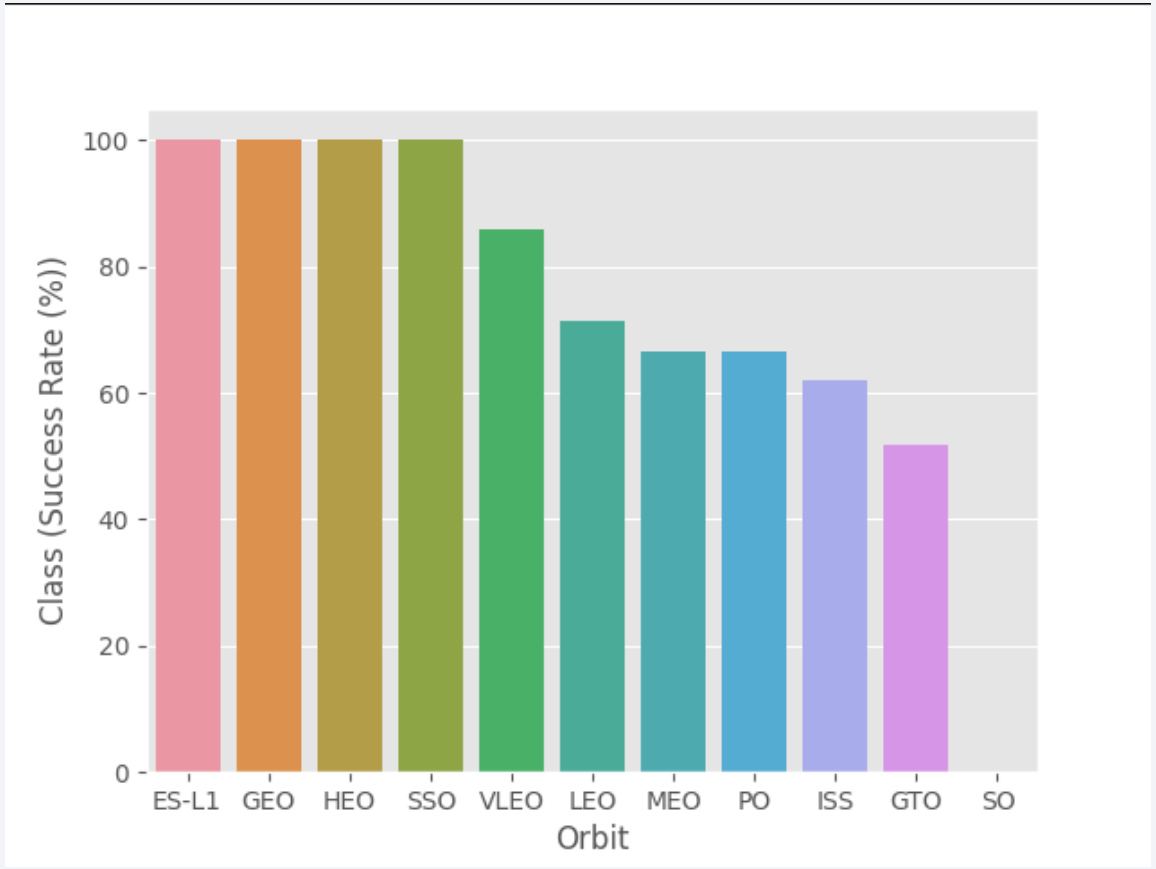


Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

Success Rate vs. Orbit Type

Bar Plot

Success Rate vs. Orbit Type

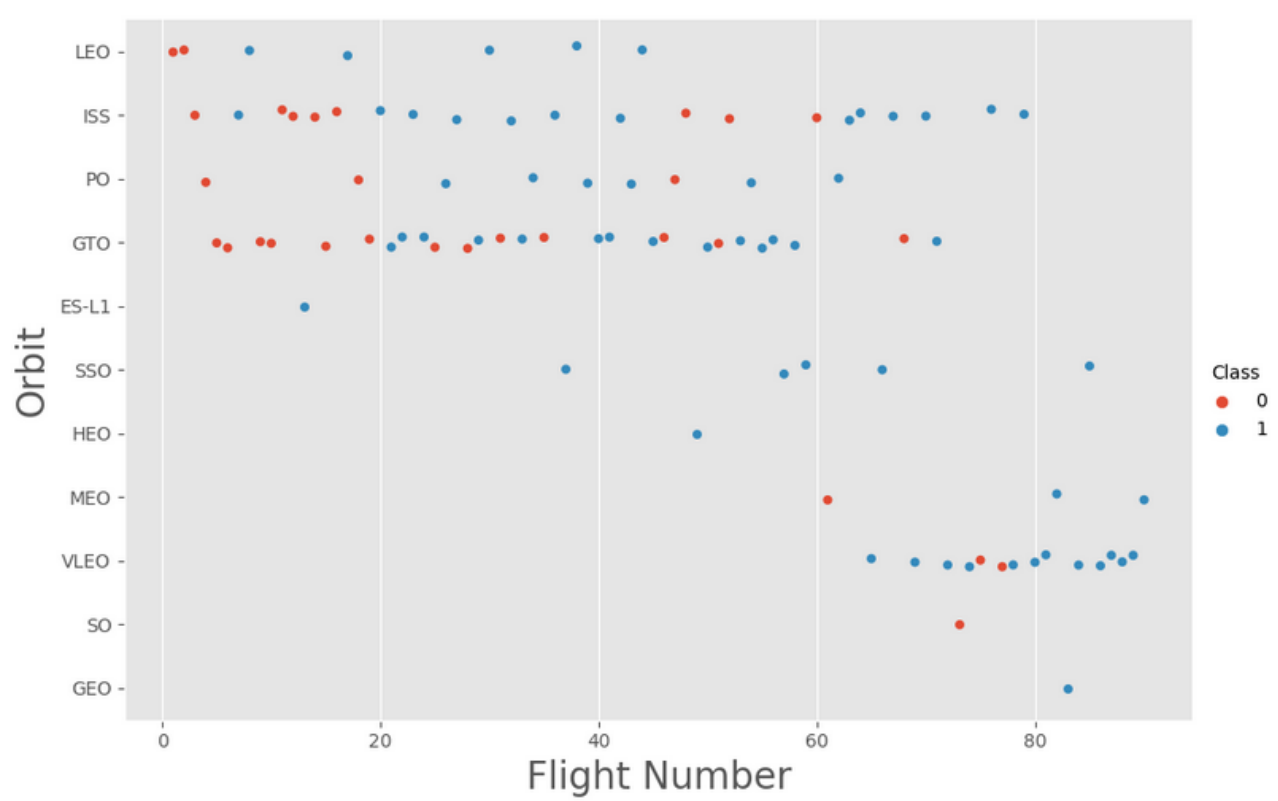


Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, with SO orbit having the lowest success rate at ~50%. Orbit SO has 0% success rate.

Flight Number vs. Orbit Type

Scatter Plot

Flight Number vs. Orbit Type



You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type

Scatter Plot

Payload vs. Orbit Type

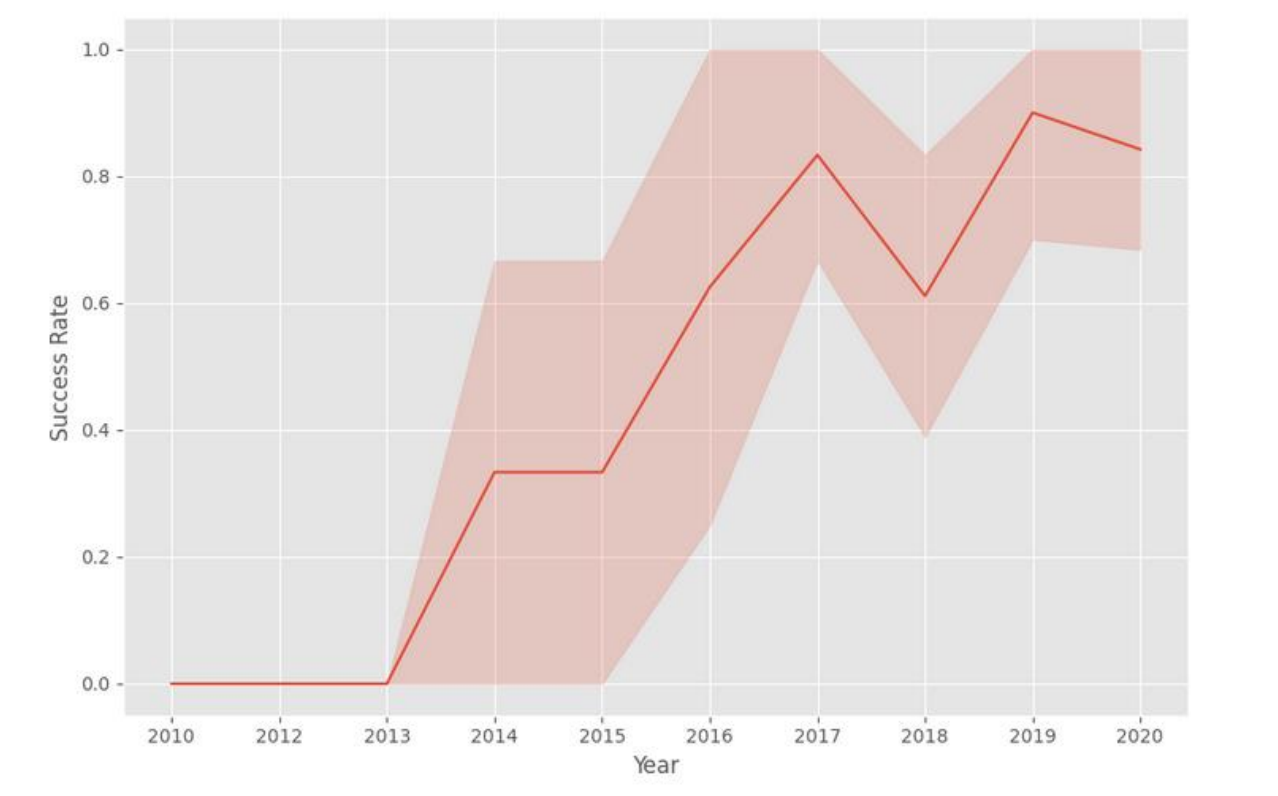
With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS

However, for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend

Line Plot

Yearly Average Success Rate



Since 2013, success rate kept increasing till 2020

All Launch Site Names

- Find the names of the unique launch sites

Task 1

Display the names of the unique launch sites in the space mission

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

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

- Used 'SELECT DISTINCT' statement to return only the unique launch sites from 'LAUNCH_SITE' column of the SPACEXTBL table.

Launch Site Names Begin with 'CCA'

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

Task 2

Display 5 records where launch sites begin with the string 'CCA'

```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

```
 * sqlite:///my_data1.db
Done.
```

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

- Used 'LIKE' command with '%' wildcard in 'WHERE' clause to select and display a table of all records where launch sites begin with the string 'CCA'.

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL
 * sqlite:///my_data1.db
Done.
sum(PAYLOAD_MASS__KG_)
619967
```

- Used the 'SUM' function to return and display the total sum of 'PAYLOAD_MASS_KG' column for Customer 'NASA(CRS)'

Average Payload Mass by F9 v1.1

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

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL;
 * sqlite:///my_data1.db
Done.
avg(PAYLOAD_MASS__KG_)
6138.287128712871
```

- Used the 'AVG' function to return and display the average payload mass carried by booster version F9 v1.1.

First Successful Ground Landing Date

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

```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
 * sqlite:///my_data1.db
Done.
 MIN(DATE)
 01-05-2017
```

- Used the 'MIN' function to return and display the first (oldest) date when first successful landing outcome on ground pad 'Success (ground pad)' happened.

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

```
%sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000;
 * sqlite:///my_data1.db
Done.
```

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

- Used 'SELECT DISTINCT' statement to return and list the 'unique' names of boosters with operators >4000 and <6000 to list booster with payloads between 4000-6000 with landing outcome of 'Success (drone ship)'.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
%sql select count(Mission_Outcome) from SPACEXTBL group by Mission_Outcome;
 * sqlite:///my_data1.db
Done.
```

count(Mission_Outcome)
1
98
1
1

- Used the 'COUNT' together with the 'GROUP BY' statement to return total number of mission outcomes.

Boosters Carried Maximum Payload

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

```
%sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_= (select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
```

```
 * sqlite:///my_data1.db
Done.
```

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

- Used a subquery to return and pass the Max Payload and used it to list all the boosters that have carried the Max payload of 15600kgs.

2015 Launch Records

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

```
%sql  SELECT substr(Date, 6, 2),Mission_Outcome,Booster_Version,LAUNCH_SITE FROM SPACEXTBL where substr(Date, 0, 5)='2015';
 * sqlite:///my_data1.db
Done.
```

substr(Date, 6, 2)	Mission_Outcome	Booster_Version	Launch_Site
01	Success	F9 v1.1 B1012	CCAFS LC-40
02	Success	F9 v1.1 B1013	CCAFS LC-40
03	Success	F9 v1.1 B1014	CCAFS LC-40
04	Success	F9 v1.1 B1015	CCAFS LC-40
04	Success	F9 v1.1 B1016	CCAFS LC-40
06	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

- Used the 'substr' in the select statement to get the month and year from the date column where substr(Date,7,4)='2015' for year and Landing_outcome was ‘Failure (drone ship)’ and return the records matching the filter.

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

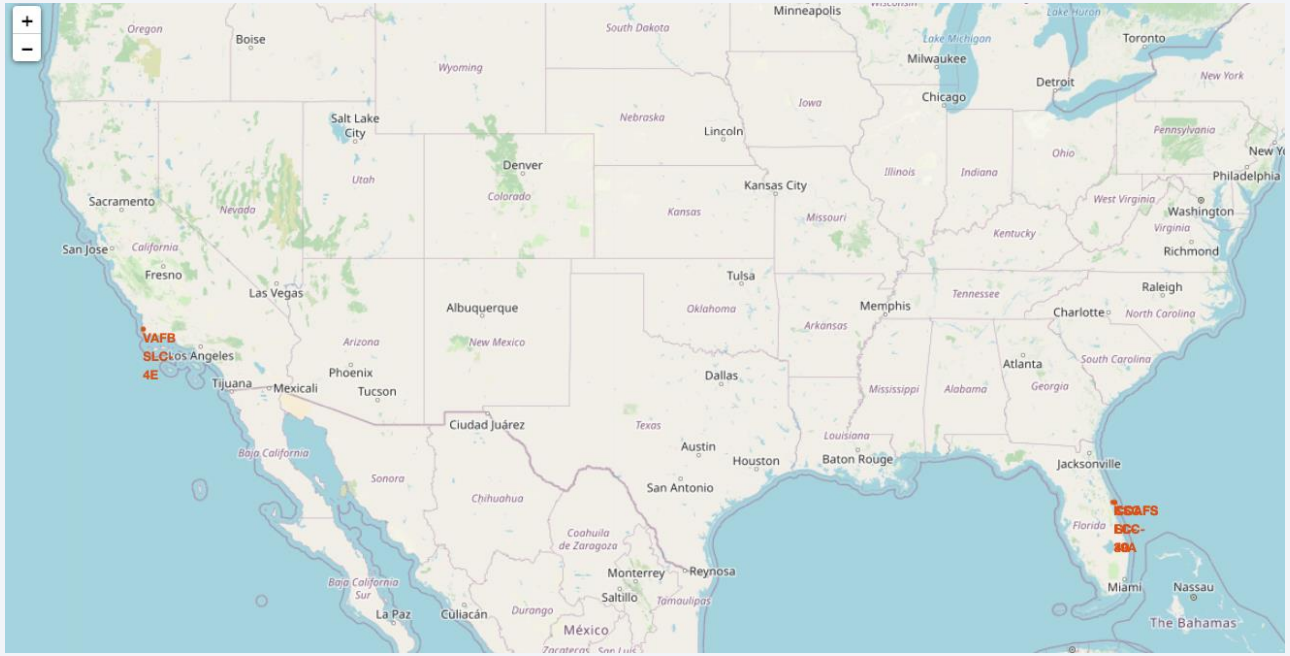
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

```
%sql SELECT Landing_Outcome FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY DATE DESC;
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome
No attempt
Success (ground pad)
Success (drone ship)
Success (drone ship)
Success (ground pad)
Failure (drone ship)
Success (drone ship)
Success (drone ship)
Success (drone ship)
Failure (drone ship)


Section 3
Launch Sites
Proximities Analysis

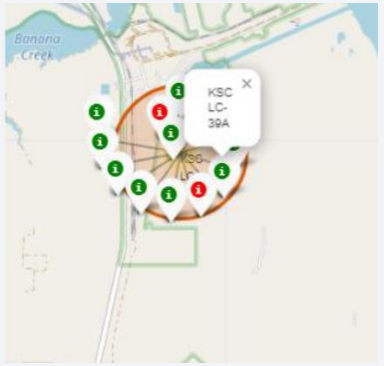
Markers of all launch sites on global map



All launch sites are in proximity to the Equator, (located southwards of the US map). Also all the launch sites are in very close proximity to the coast.

Launch outcomes for each site on the map with color markers

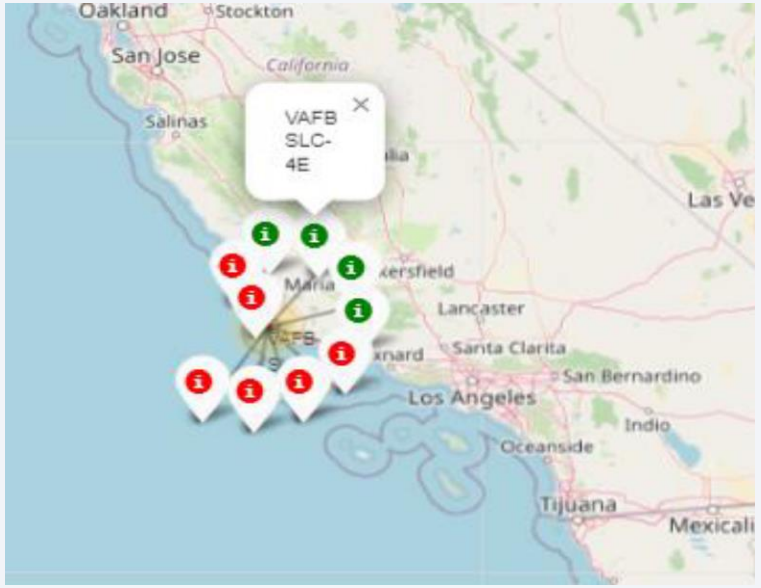
Florida Sites



In the Eastern coast (Florida) Launch site KSC LC-39A has relatively high success rates compared to CCAFS SLC-40 & CCAFS LC-40.

Launch outcomes for each site on the map with color markers

West Coast/California

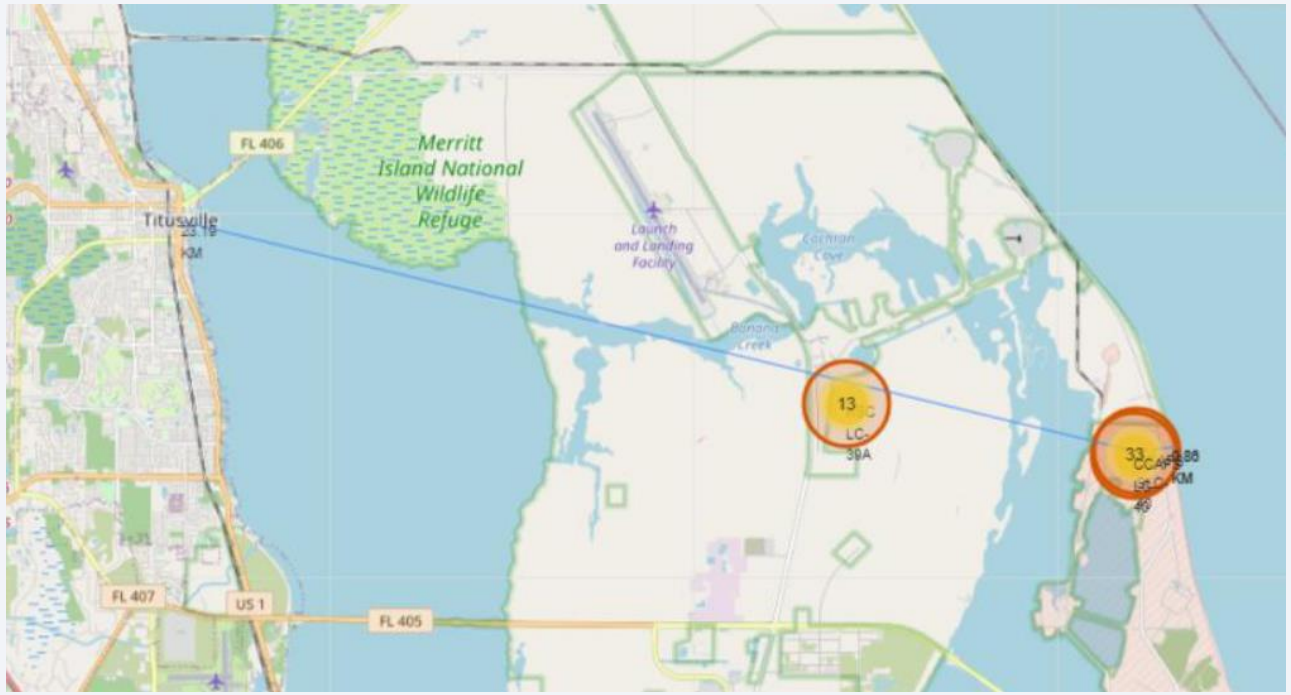


In the West coast (California) Launch site VAFB SLC-4E has relatively lower success 4/10 compared to KSC LC-39A launch site in the Eastern Coast of Florida.

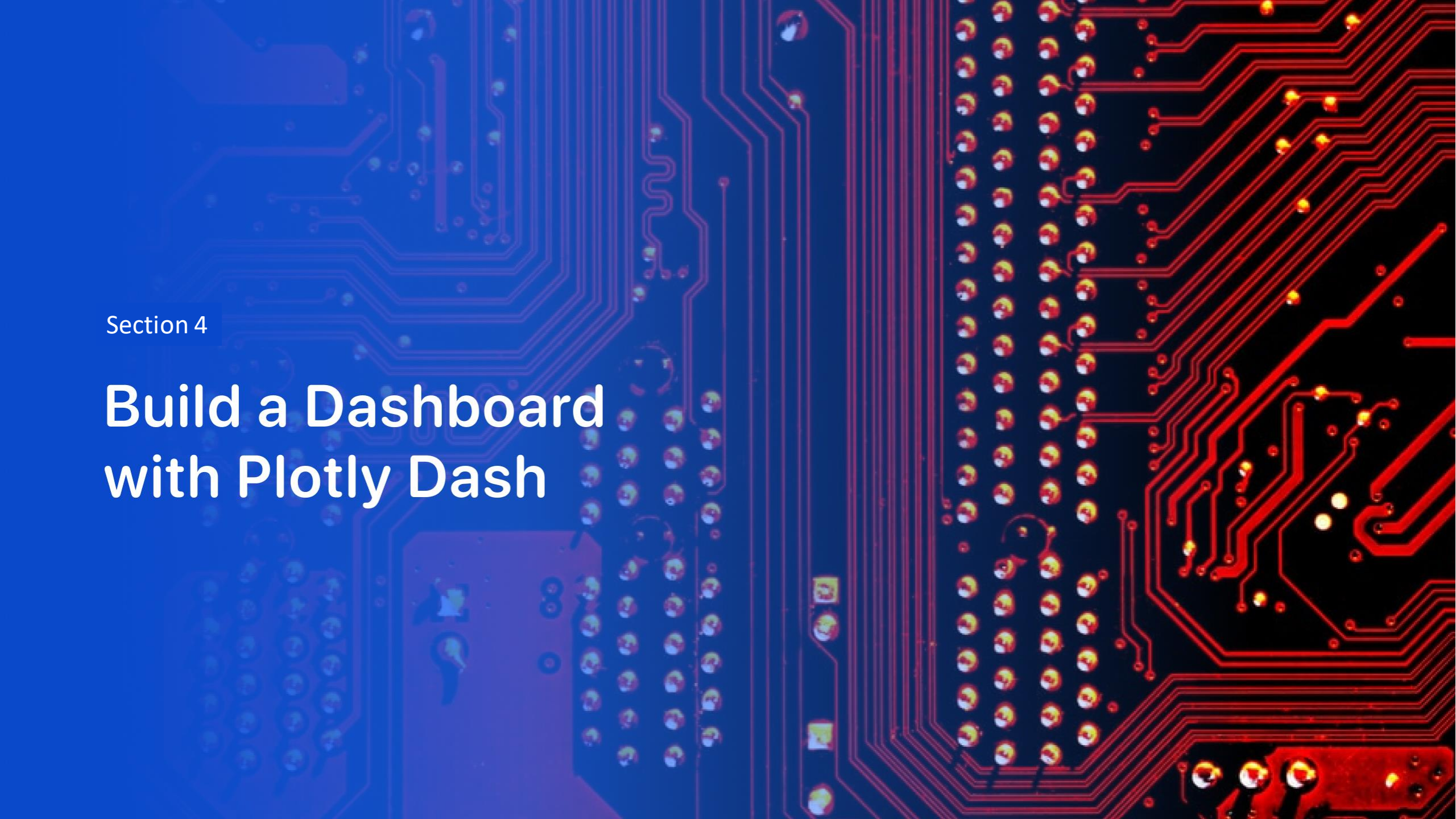
Distances between a launch site to its proximities

Launch site CCAFS SLC-40 proximity to coastline is 0.86 km.

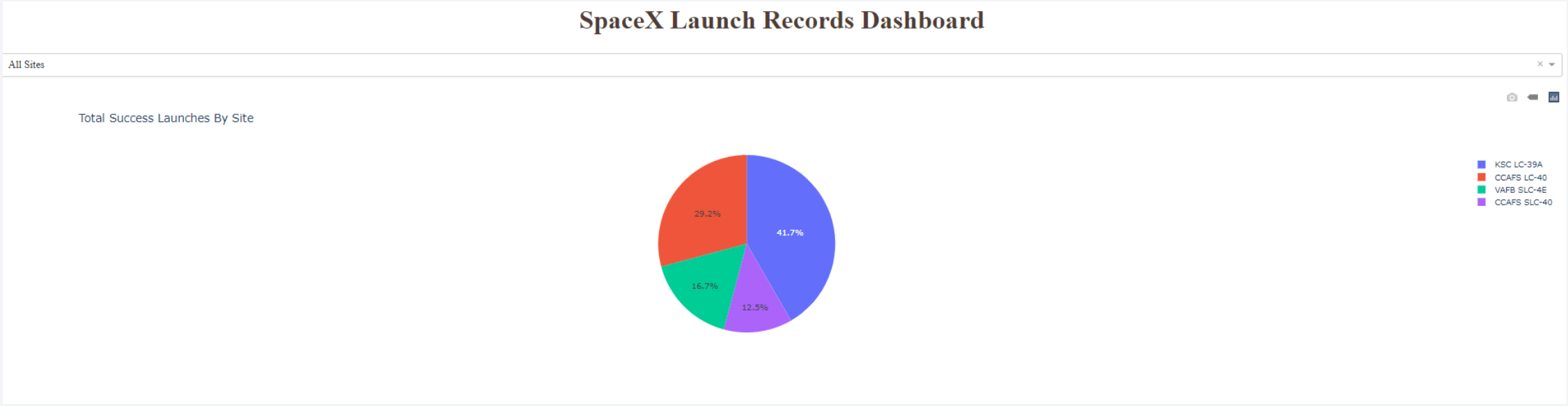
Distances between a launch site to its proximities



Launch site CCAFS SLC-40 closest highway (Washington Avenue) proximity is 23.19 km.

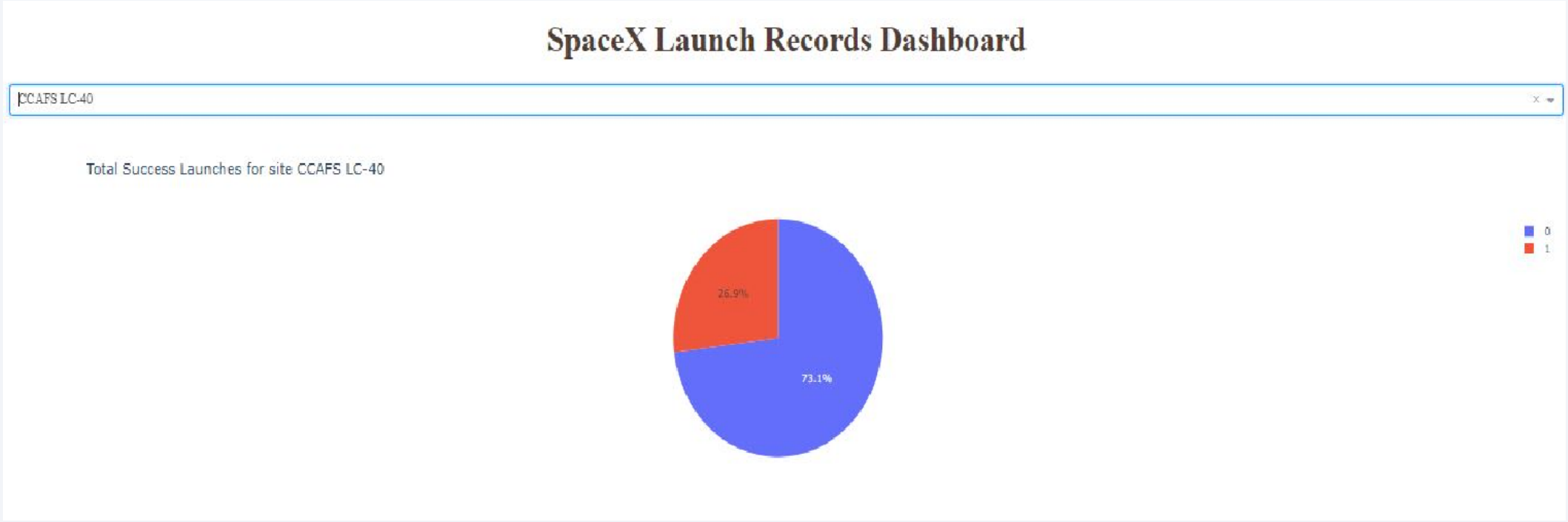

Section 4
Build a Dashboard with Plotly Dash

Pie-Chart for launch success count for all sites



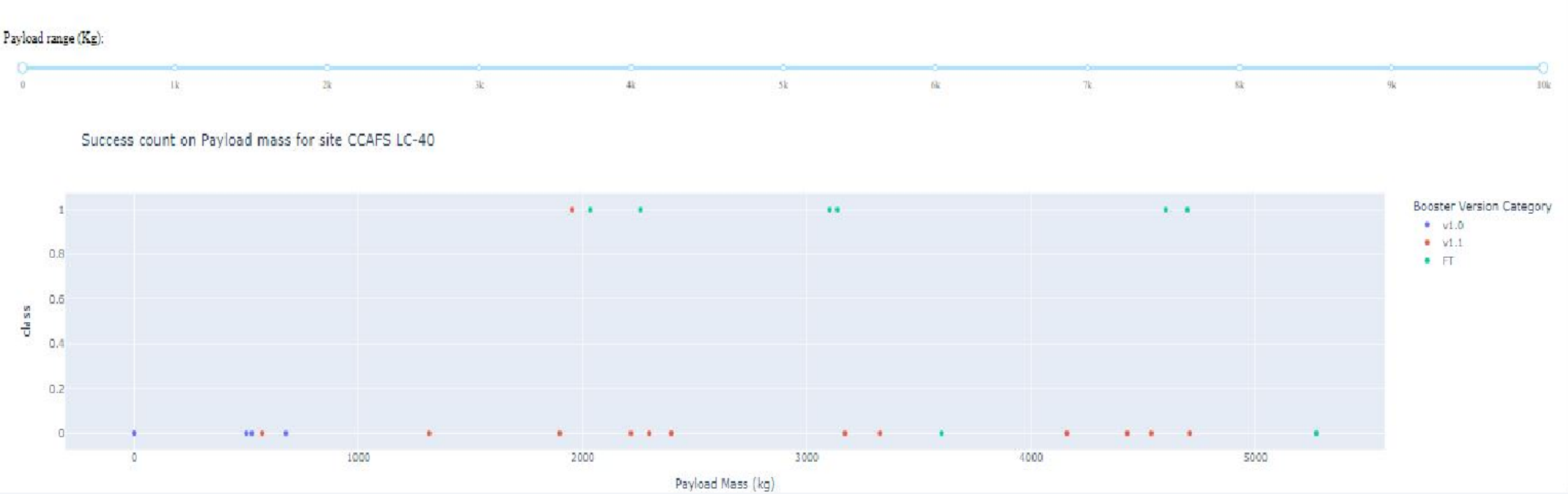
Launch site LSC LC-39A has the highest launch success rate at 42% followed by CCAFS LC-40 at29%, VAFB SLC-4E at 17% and lastly launch site CCAFS SLC-40 with a success of 13%.

Pie-Chart for the launch with 2nd highest launch success ratio



Launch site CCAFS LC-40 had the 2nd highest success ratio of 73% success against 27% failed launches.

Payload vs. Launch Outcome scatter plot for all sites



For Launch site CCAFS LC-40 the booster version FT has the largest success rate from a payload mass of >2000kg.

Section 5

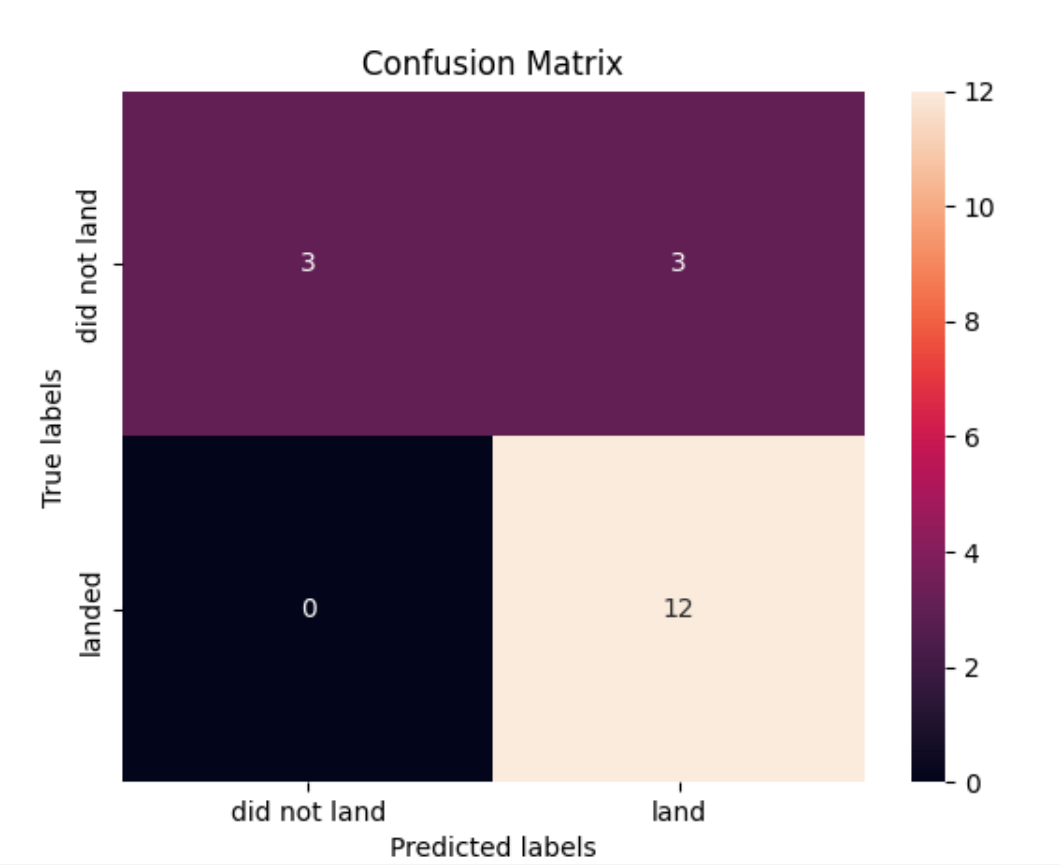
Predictive Analysis (Classification)

Classification Accuracy

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

All the methods perform equally on the test data, i.e., they all have the same accuracy of 0.833333 on the test data

Confusion Matrix



All the four-classification models had the same confusion matrixes and were able to equally distinguish between the different classes. The major problem is false positives for all the models.

Conclusions

- Different launch sites have different success rates. CCAFS LC-40 has a success rate of 60%, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.
- We can deduce that, as the flight number increases in each of the three launch sites, so does the success rate. For instance, the success rate for the VAFB SLC 4E launch site is 100% after the Flight Number 50. Both KSC LC 39A and CCAFS SLC 40 have a 100%success rates after 80th flight.
- If you observe Payload vs. Launch Site scatter plot chart you will find for the VAFB-SLC launch site, there are no rockets launched for heavy payload mass (greater than 10000).
- Orbits ES-1,GEO, HEO and SSO have the highest success rates at 100%, with SO orbit having the lowest success rate at ~50%. Orbit SO has 0% success rate.
- In LEO orbit, the success appears related to the number of flights, on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Conclusions contd...

- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However, for GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there here.
- And finally the success rate since 2013 kept increasing till 2020.

Thank you!