IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

TAMELAH Fabrice
February, 2025

Outline

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

Executive Summary

In this project, a rival company to SpaceX (i.e., SpaceY) uses SpaceX Falcon 9 rocket data to determine the rocket first stage landing successes and the cost of a launch. A summary for the methodologies and results described in this report is outlined below

Summary of methodologies

- Data Collection
- Data wrangling
- Exploratory data analysis with data visualization and SQL
- Building an interactive map with Folium
- Building Dashboard with Plotly Dash
- Predictive analysis (classification)

Summary of all results

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

Introduction

- This capstone project, part of the IBM Data Science Professional certificate, aims to demonstrate skills in data science and machine learning through the analysis of real data. The fictional company SpaceY uses data from SpaceX's Falcon 9 rocket to assess the success of the first stage landing and estimate the cost of a launch. SpaceX puts the cost at $62 million, while other companies exceed $165 million.

- The project uses Jupyter notebooks for data collection and analysis, with a final report available on GitHub. Key sections of the report include data collection methodology, data manipulation and exploratory analysis, interactive visualization, development and evaluation of machine learning classification models. Finally, the accuracy of different algorithms is compared to predict future Falcon 9 landings.

Section 1

Methodology

Methodology

- Data used in this project were collected from SpaceX Rest API and from Wikipedia launch table.
- The wrangling of the collected data included cleaning, preparation for visualization and information extraction for usage in ML predictive models such as logistic regression, support vector machine (SVM), decision tree, and K-nearest neighbors (KNN).
- In addition, exploratory data analysis (EDA) was performed using visualization and SQL. Lastly, Folium and Plotly Dash Python libraries were use in data representation and in the interactive visual analytics of the data.
- Finally, predictive analysis was performed using classification models for predicting if the first stage of Falcon 9 rocket will land successfully using Skikit-learn and also the accuracy of the model was determined.

Data Collection: **Overview**

Data collection and visualization major steps:

Step # 1: Collect Data from SpaceX API and Convert data to .jason file

Step# 2: Scrap and filter data to include Falcon 9 data, assign data to dataf rame and dictionary, and export data to a csv file

Step 3: Plot and visualize the data

Portion of generated output data file: dataset_part1.csv:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	Landin
0	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	

GitHub URL: IBM_cours/dataset_part_1.csv at main · indomptablelion/IBM_cours

Data Collection – SpaceX API

- Request response from SpaceX API using get request and covert data to .Json file
- Use custom functions to clean data
- Clean data and assign data to dictionary and data frame
- Filter data to include only Falcon 9 launches and export data to a csv file: dataset_part1
- GitHub URL: [IBM_cours/jupyter-labs-spacex-data-collection-api.ipynb at main · indomptablelion/IBM_cours]

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/data
```

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

In [10]:
```
response=requests.get(static_json_url)
```

In [11]:
```
response.status_code
```

Out[11]: 200

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

In [12]:
```
# Use json_normalize meethod to convert the json result into a dataframe
# Decode the response content as JSON
json_data = response.json()

# Convert the JSON data into a Pandas DataFrame
data = pd.json_normalize(json_data)
```

Using the dataframe `data` print the first 5 rows

In [13]:
```
# Get the head of the dataframe
data.head()
```

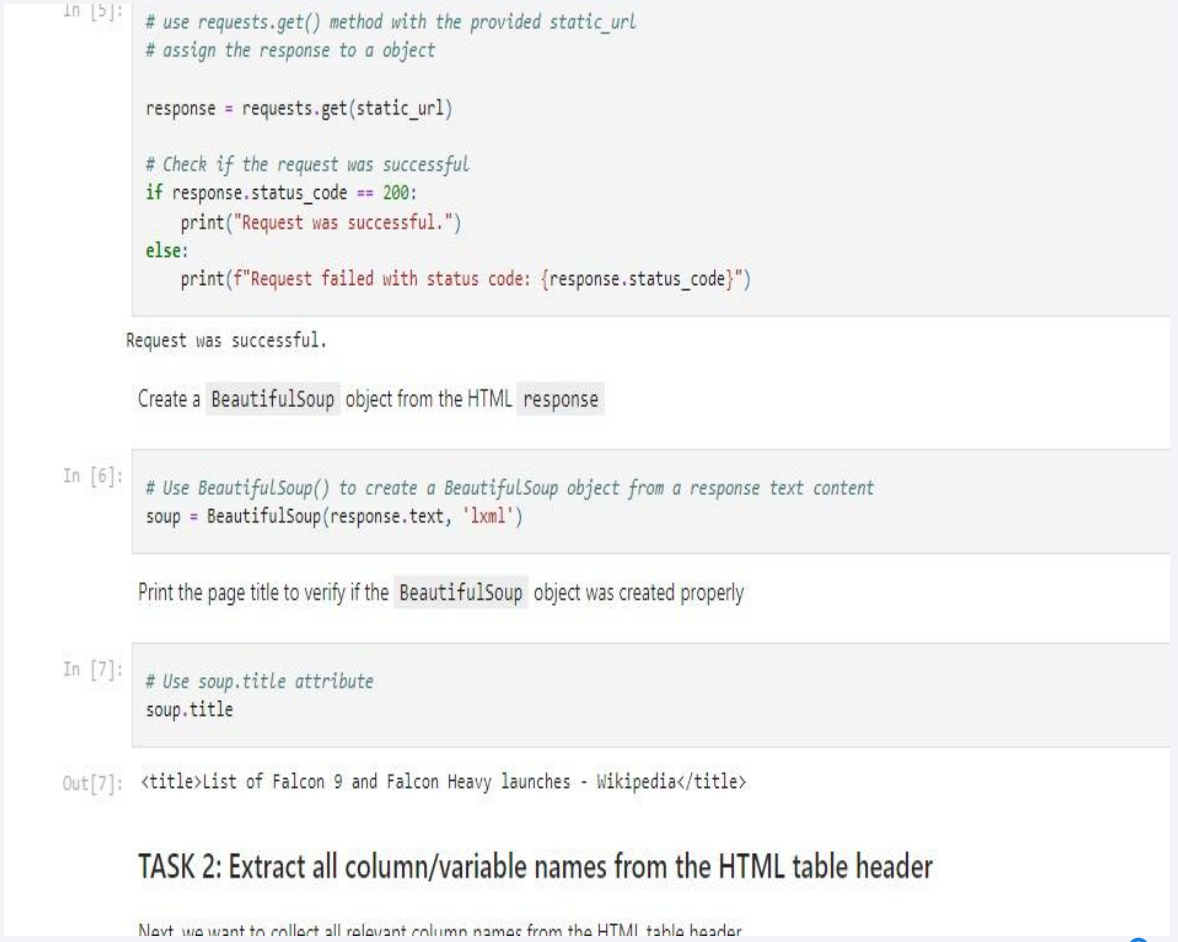
Out[13]:

	static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	crew	ships

Data Collection - Scraping

- Step 1: Perform HTTP get to request Falcon 9 HTML page and create Beautiful Soup object from HTML
- Step 2: Extract all column/variable names from the HTML table header
- Step3: Create a data frame by parsing the launch HTML tables
- Step 4: export data into CSV file (spacex_web_scraped.csv

GitHub URL: IBM_cours/jupyter-labs-webscraping.ipynb at main · indomptablelion/IBM_cours



In [5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object

response = requests.get(static_url)

# Check if the request was successful
if response.status_code == 200:
    print("Request was successful.")
else:
    print(f"Request failed with status code: {response.status_code}")
```

```
Request was successful.
```

Create a `BeautifulSoup` object from the HTML `response`

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, 'lxml')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Data Wrangling

- Step 1: Load data from dataset_part1.csv file and calculate the number of launches on each site
- Step 2: Calculate the number and the occurrence of each orbit
- Step3: Calculate the number and occurrence of mission outcome of the orbits
- Step 4: Create a landing outcome label from outcome column and export data into dataset_part2.csv file
- GitHub URL: [IBM_cours/labs-jupyter-spacex-Data wrangling.ipynb at main · indomptablelion/IBM_cours]

Out[13]:

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

EDA with Data Visualization

Use Matplotlib and Seaborn for data visualization

- Step 1: Visualize the relationship between flight number and launch site
- Step 2: Visualize the relationship between payload and launch site
- Step3: Visualize the relationship between success rate of each orbit type Use Matplotlib and Seaborn for data visualization
- Step 4: Visualize the relationship between flight number and orbit type
- Step 5: Visualize the relationship between payload and orbit type
- Step 6: Visualize the launch success yearly trend
- Step 7: Create dummy variable to categorical columns
- Step 8: Cast all numeric columns to float64

GitHub URL:
IBM_cours/edadataviz.ipynb at main · indomptablelion/IBM_cours

EDA with SQL

- Step 1: Display the names of the unique launch sites in the space mission
- Step 2: Display 5 records where launch sites begin with the string ‘CCA’
- Step 3: Display average payload mass carried by booster launched by NASA (CRS)
- Step 4: Display average payload mass carried by booster version F9 v1.1
- Step 5: List the date when the first successful landing outcome in ground pad was achieved
- Step 6: List the names of the boosters which have success in drone ship and mass > 4000 & <6000
- Step 7: List the total number of successful and failure mission outcomes
- Step 8: List the names of the booster_ve which have carried the max payload mass
- Step 9: List the records which display the month, failure landing, booster version ..etc.
- Step 10: Rank the count of landing outcomes or success

GitHub URL: IBM_cours/jupyter-labs-eda-sql-coursera_sqllite.ipynb at main · indomptablelion/IBM_cours

Build an Interactive Map with Folium

-Step 1: Mark all launch sites on a map created using Folium by adding markers* with circle , popup label and text label to each site using its longitude and latitude coordinates to show the geographical location approximately to the equator

-Step 2: Mark the success/failed launches for each site on the map using colored markers --

- Step 3: Calculate the distance between a launch site to its proximities

GitHub URL: IBM_cours/lab_jupyter_launch_site_location.ipynb at main · indomptablelion/IBM_cours

Explanation:

From the visual analysis of the launch site KSC LC-39A we can clearly see that it is:

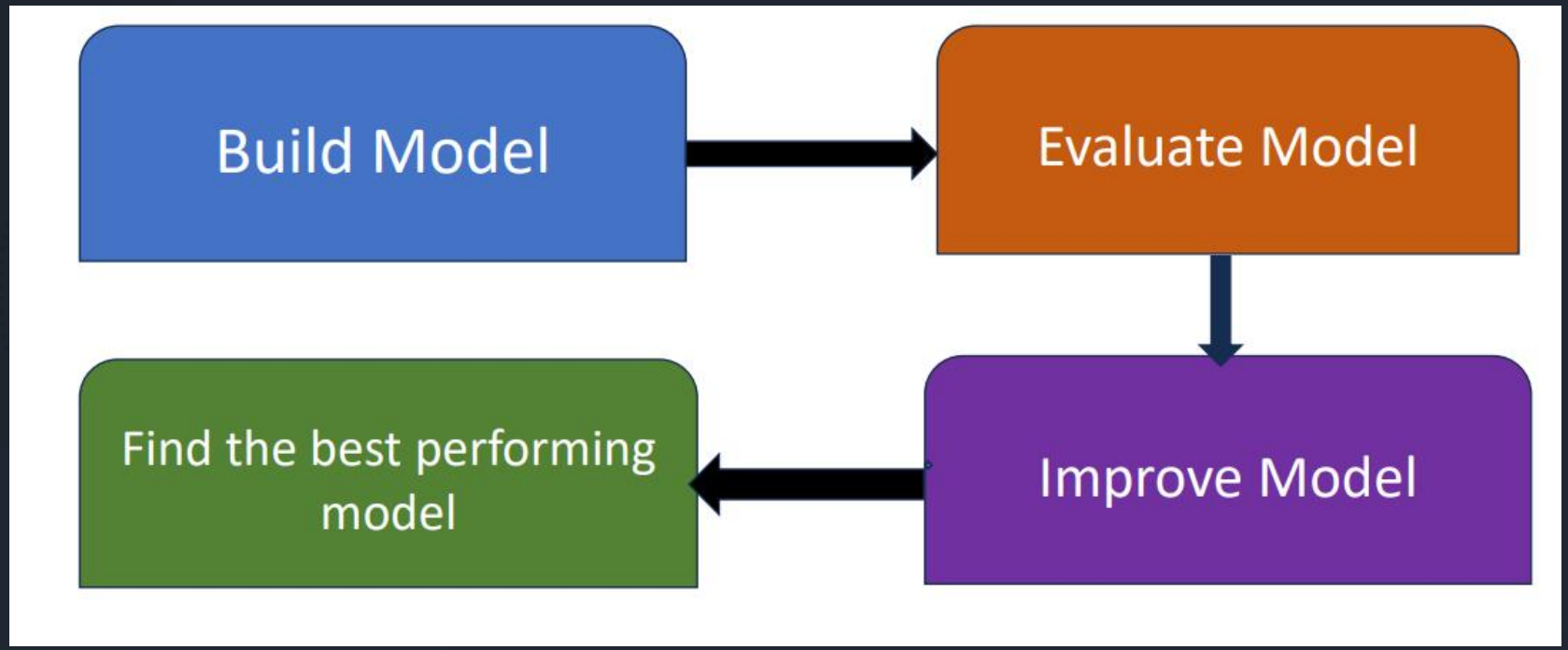
- relative close to railway (15.23 km)
- relative close to highway (20.28 km)
- relative close to coastline (14.99 km)
- Also the launch site KSC LC-39A is relative close to its closest city Titusville (16.32 km).
- Failed rocket with its high speed can cover distances like 15-20 km in few seconds. It could be potentially dangerous to populated areas.

Build a Dashboard with Plotly Dash

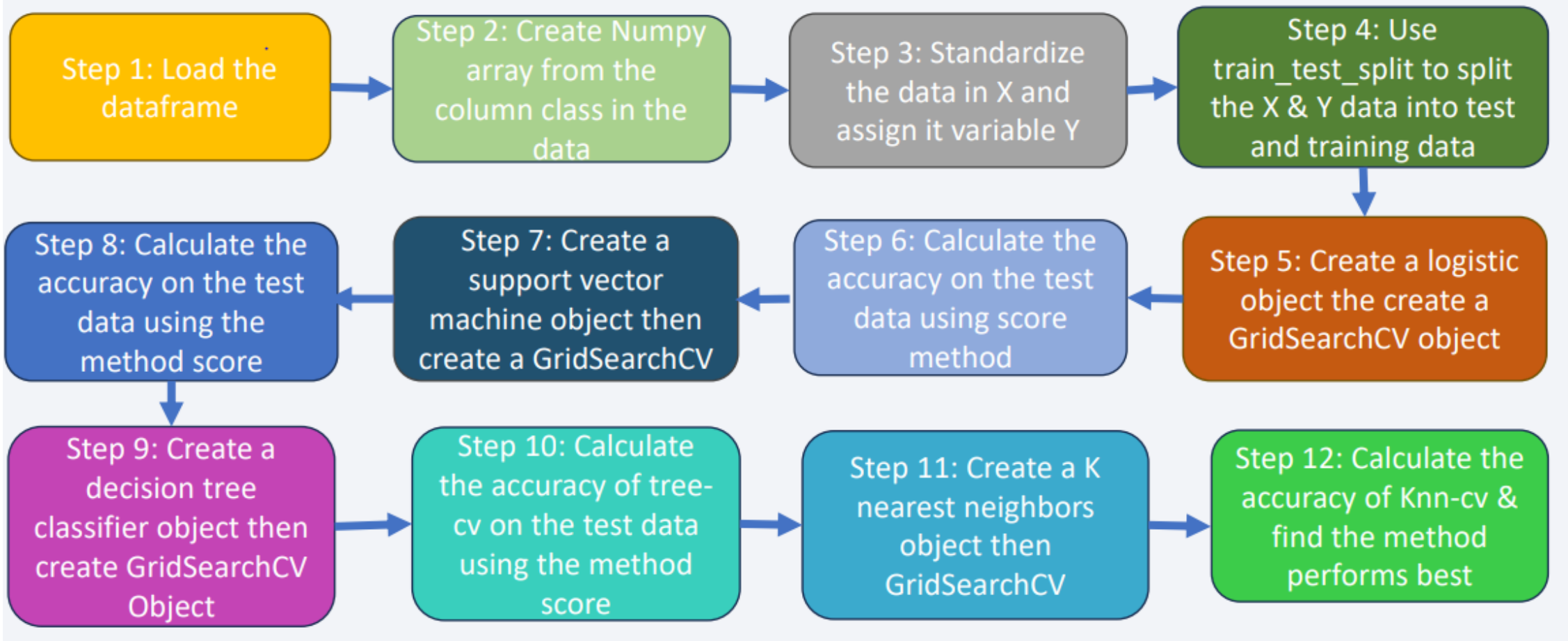
- Step 1: Add dropdown list to enable launch site selection
- Step 2: Add pie chart to show the total successful launches count for all sites and the success vs. failed counts
- Step 3: Add a range slider to select payload
- Step 4: Add a scatter chart of payload mass vs. success rate of different booster versions
  The dashboard is built using Dash web

GitHub URL: IBM_cours/spacex_dash_app.py at main · indomptablelion/IBM_cours

Predictive Analysis (Classification) : Overview



Predictive Analysis (Classification) Steps



GitHub URL: [IBM_cours/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · indomptablelion/IBM_cours]

Results

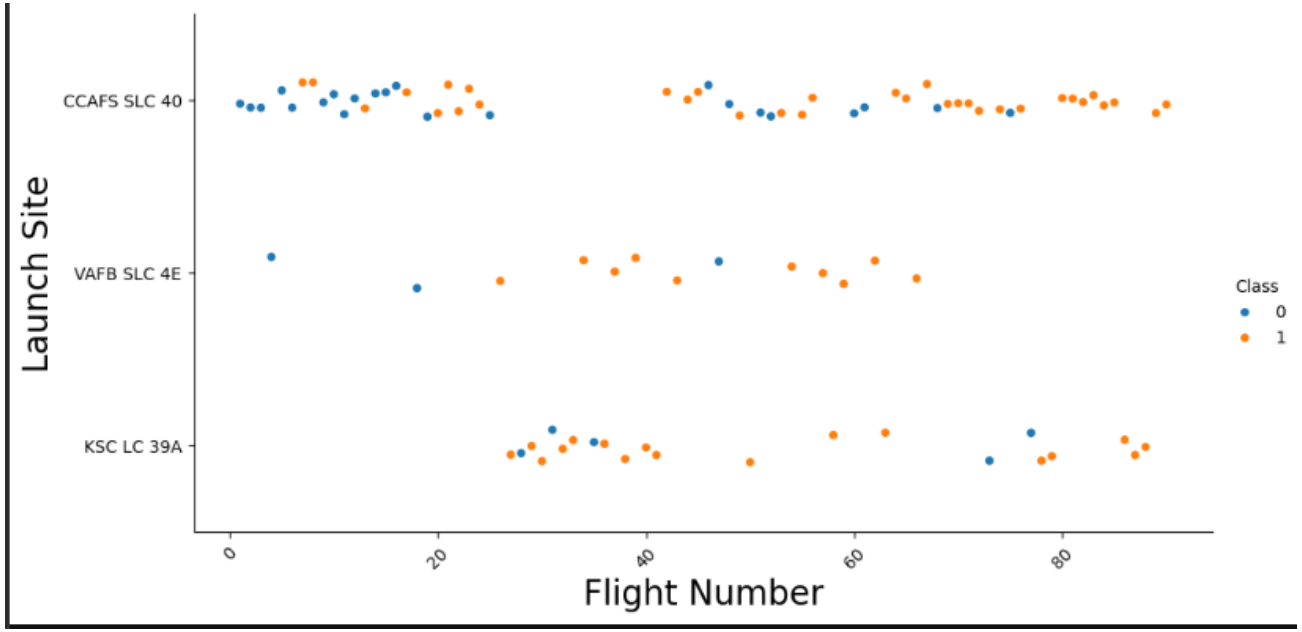
- Exploratory data analysis results.
- Interactive analytics demo in screenshots.
- Predictive analysis results.

Section 2

Insights drawn from EDA

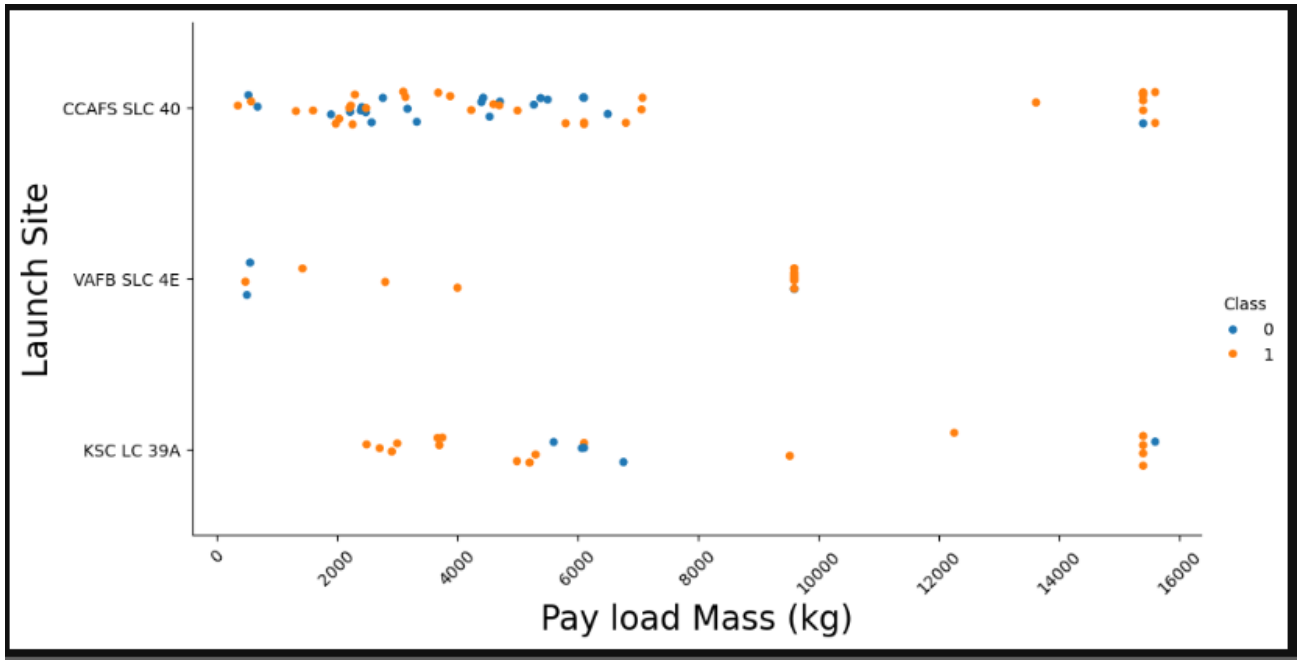
Flight Number vs. Launch Site

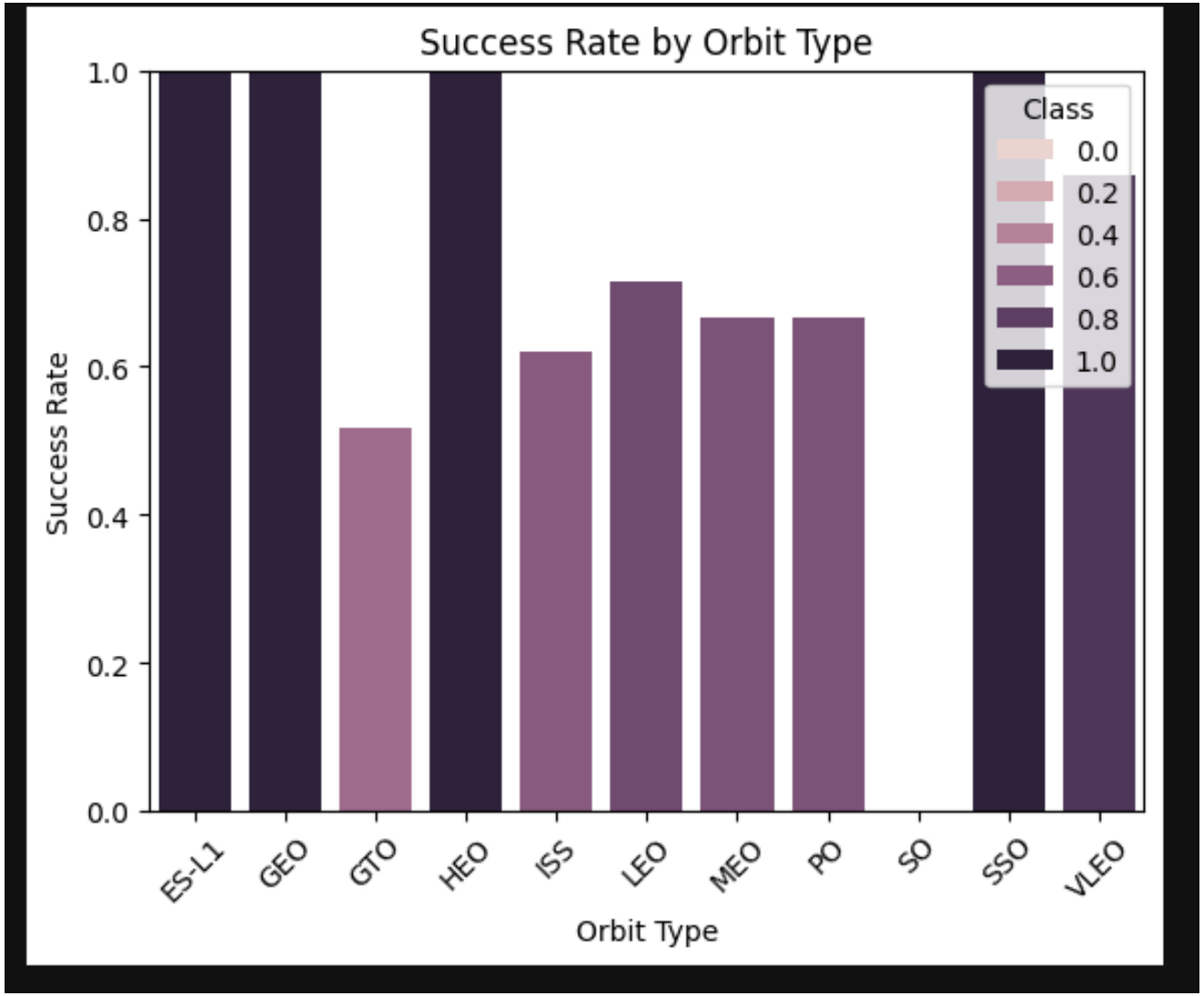
- he majority of the flights were launched from the CCAFS SLC 40 sites.
- The VAFB SLC 4E and KSC LC 39A sites have higher success rates than other sites.
- Newer flights have higher success rates than older flights



Payload vs. Launch Site

- • The majority of the flights with payload mass above 7000 Kg were successful.
- • KSC LC 39A success rate for payload mass under 5500 kg is 100%.
- • For all launch sites the success rate is proportional to the payload mass.



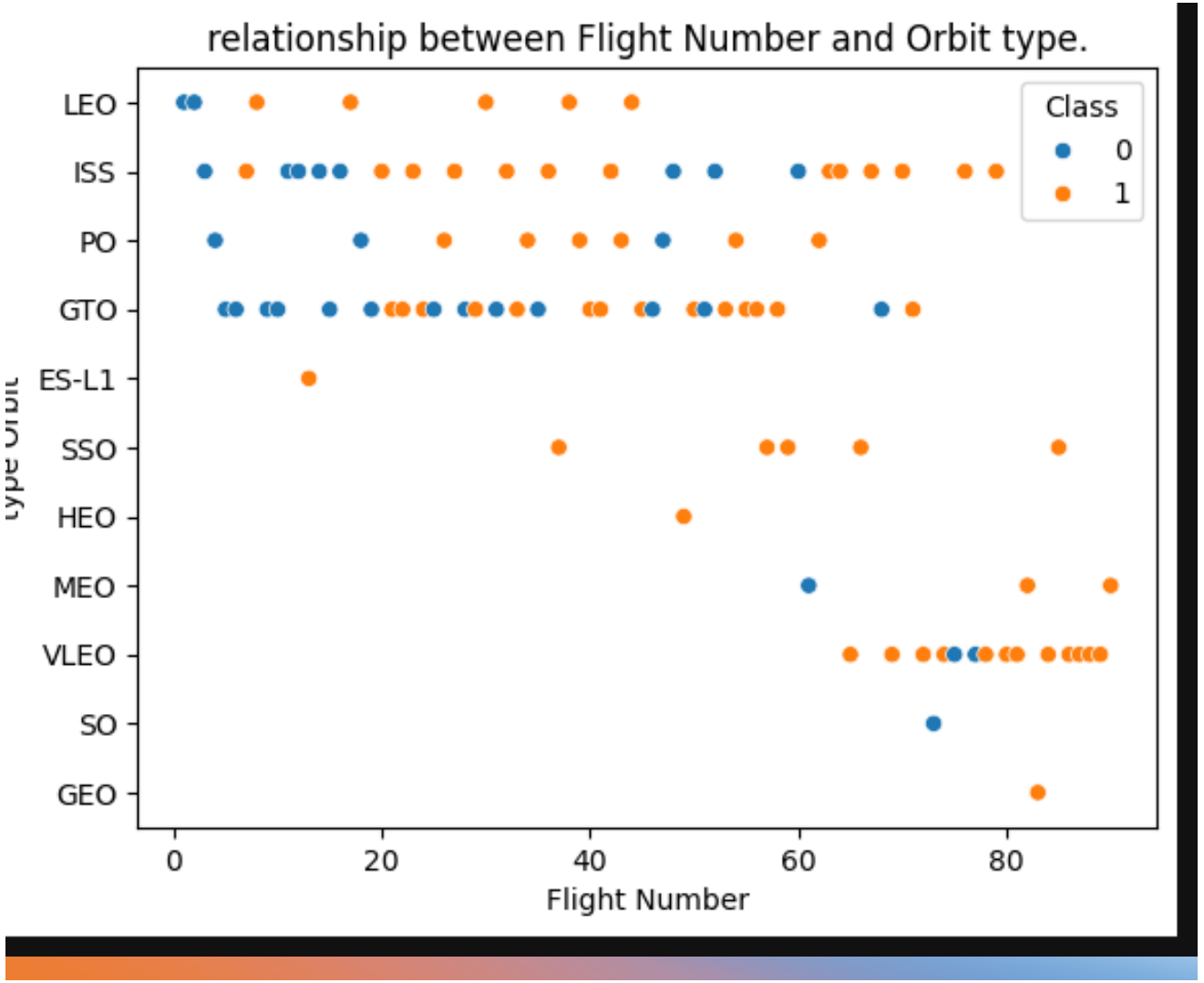


Success Rate vs. Orbit Type

- The OS orbit has 0% success rate.
- The ELS-1, GEO, HEO and SSO orbits have 100% success rate.
- Orbits GTO, ISS, LEO, MEO and PO success rate is higher than 50% and less than 75%

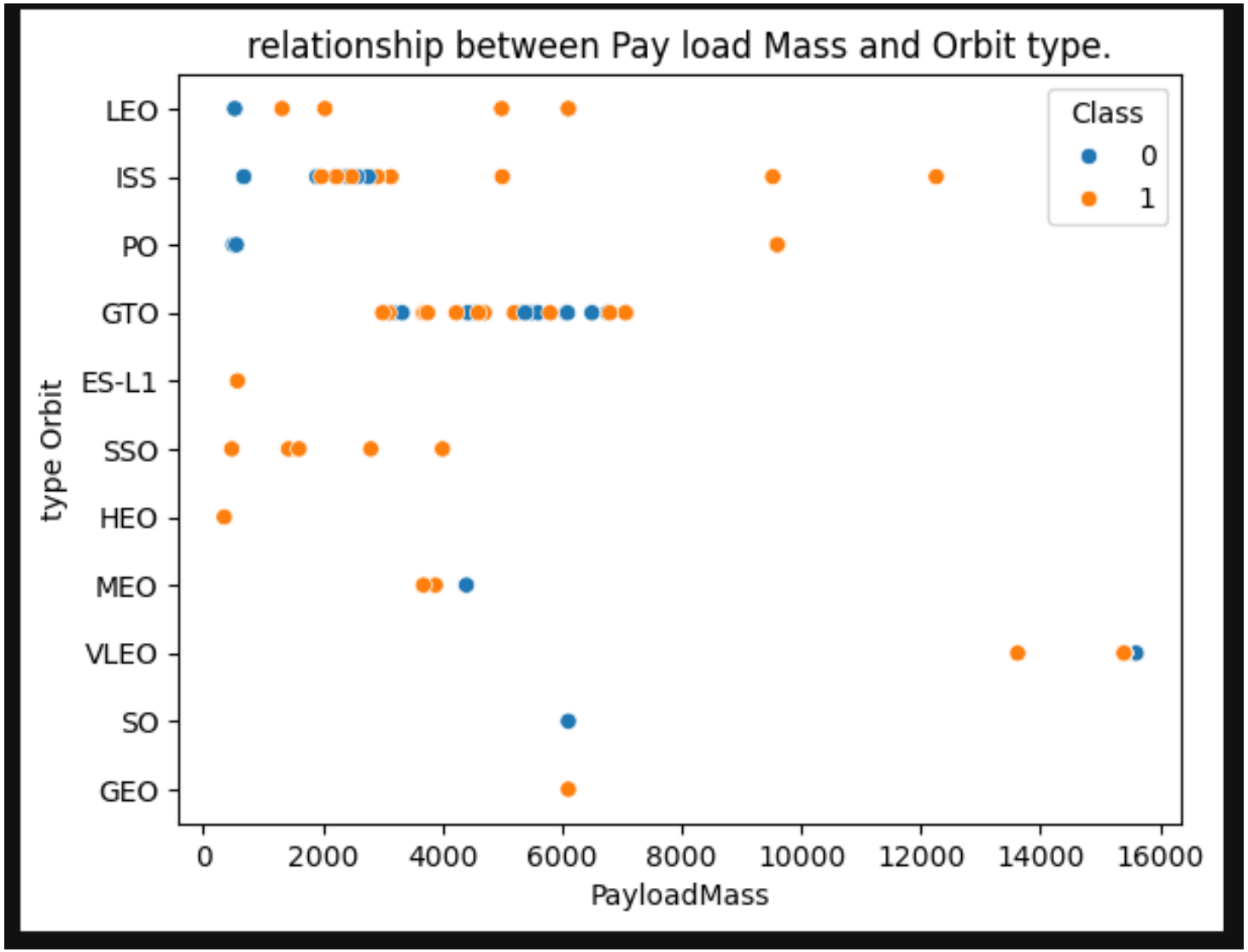
Flight Number vs. Orbit Type

- The majority of the flights were launches to the ISS and GTO orbits.
- The data suggests that there is no relationship between the flight number and the orbit type.



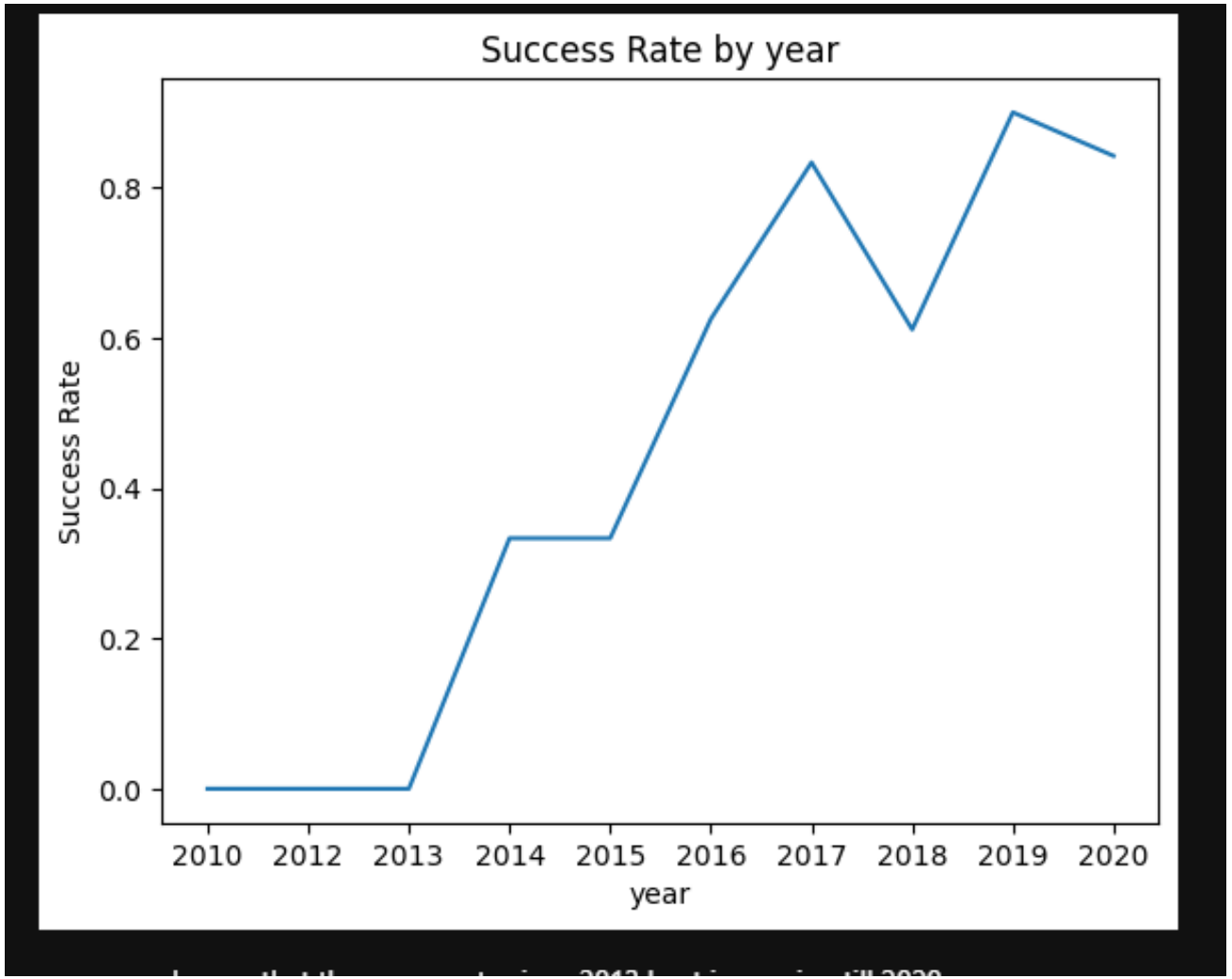
Payload vs. Orbit Type

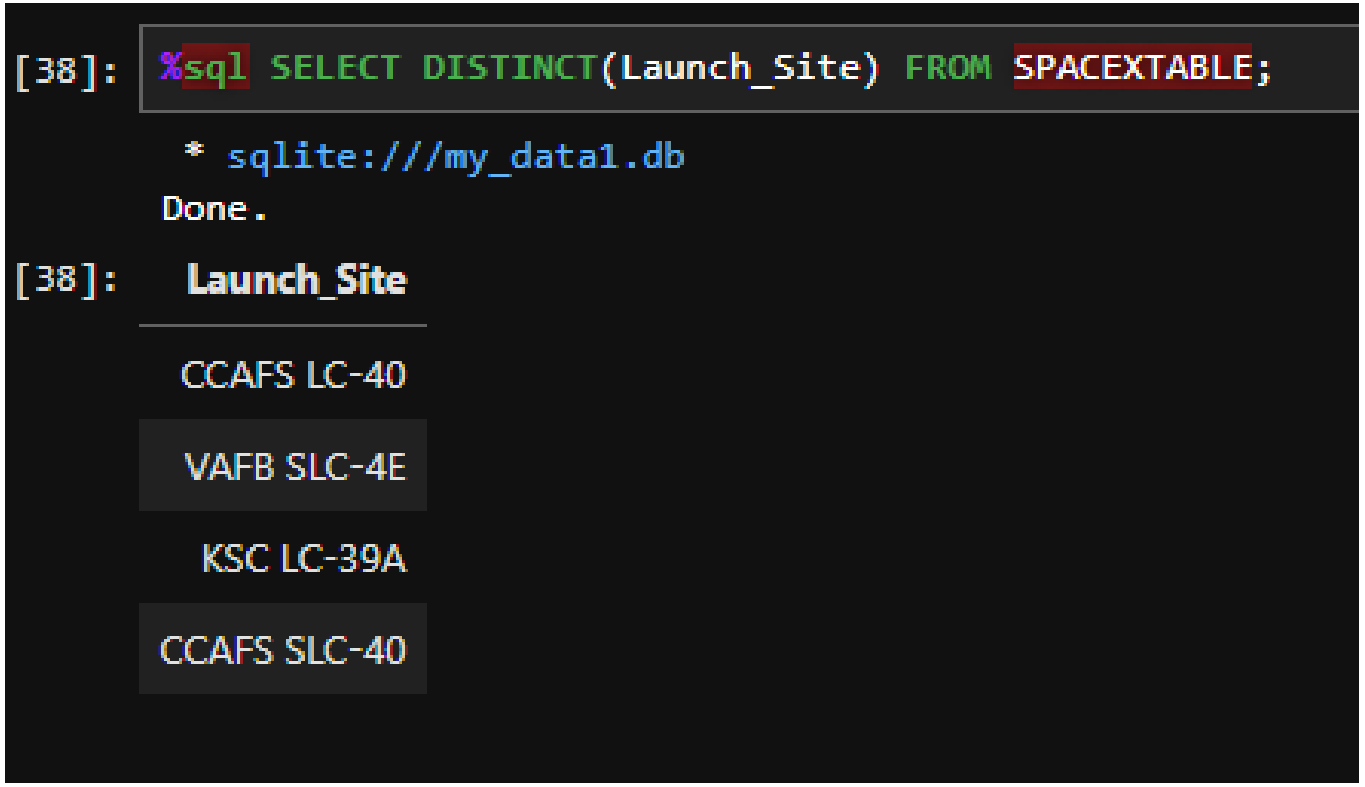
- Payload masses above 10000 Kg were placed in PO, ISS and LEO orbits.
- Payload masses above 4000 and less than 8000 Kg were placed in the GTO orbit.



Launch Success Yearly Trend

- The launches success rate increased steadily since 2013.
- The increase in the success rate between 2013 and 2017 was linear.
- During 2018 there was a drop in the launches success rate.





All Launch Site Names

- The names of the unique launch sites and the query structure to obtain these sites are shown opposite.

```
[34]: %sql SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
 * sqlite:///my_data1.db
Done.
```

[34]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landin
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure
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
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	

Launch Site Names Begin with 'CCA'

- 5 records for launch sites begin with the string “CCA”.
- The method used to obtain the information is shown opposite.

Total Payload Mass

- The total calculated mass of the payload carried by the boosters from the NASA site is 45596 kg. The query to obtain the total payload mass is as follows

Task 3

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

```
[35]: %sql SELECT Customer, SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE GROUP BY Customer HAVING Customer='NASA (CRS)';
```

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

[35]:

Customer	SUM(PAYLOAD_MASS__KG_)
NASA (CRS)	45596

Task 4

Average Payload Mass by F9 v1.1

The average payload mass carried by booster version F9 v1.1=2928.4 Kg.

Task 4

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

```
[20]: %sql SELECT Booster_Version, AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE GROUP BY Booster_Version HAVING Booster_Versi
```

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

[20]:

Booster_Version	AVG(PAYLOAD_MASS__KG_)
F9 v1.1	2928.4

First Successful Ground Landing Date

The first successful landing outcome on a ground pad was in 2015-12-22.

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
[22]: %sql SELECT min(Date) AS Date_min, Mission_Outcome FROM SPACEXTABLE WHERE Mission_Outcome='Success';
```

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

[22]:

Date_min	Mission_Outcome
2010-06-04	Success

Successful Drone Ship Landing with Payload between 4000 and 6000

- List of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 is shown below.

Task 6

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

```
[40]: %sql SELECT Booster_Version, Landing_Outcome from SPACEXTABLE WHERE Landing_Outcome='Success (drone ship)' AND PAY
```

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

[40]:

Booster_Version	Landing_Outcome
F9 FT B1022	Success (drone ship)
F9 FT B1026	Success (drone ship)
F9 FT B1021.2	Success (drone ship)
F9 FT B1031.2	Success (drone ship)

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failed missions is as follows:
- Failure (in flight)= 1
- Successful number of flights= 98

Task 7

List the total number of successful and failure mission outcomes

```
%sql SELECT Mission_Outcome, COUNT(Mission_Outcome) FROM SPACEXTABLE GROUP BY 1;
```

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

Mission_Outcome	COUNT(Mission_Outcome)
Failure (in flight)	1
Success	98

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

[47]:
```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_=(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE);
```

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

[47]:

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

Boosters Carried Maximum Payload

List of the boosters which have carried the maximum payload mass are shown below.

2015 Launch Records

- List of the failed ”landing_outcomes” in drone ship, their booster version, and the launch site name during year 2015 is shown below.

Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
%sql SELECT substr(Date, 6,2) AS months, substr(Date,0,5) AS years, Mission_Outcome,Booster_Version,Launch_Site FROM SPACEXTABLE where substr(Date,0,5)='2015';
```

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

months	years	Mission_Outcome	Booster_Version	Launch_Site
01	2015	Success	F9 v1.1 B1012	CCAFS LC-40
02	2015	Success	F9 v1.1 B1013	CCAFS LC-40
03	2015	Success	F9 v1.1 B1014	CCAFS LC-40
04	2015	Success	F9 v1.1 B1015	CCAFS LC-40
04	2015	Success	F9 v1.1 B1016	CCAFS LC-40
06	2015	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	2015	Success	F9 FT B1019	CCAFS LC-40

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

- A rank of the count of landing outcomes (such as Failure (drone ship) or success (ground pad)) between the dates 2010-06-04 and 2017-03-20, in descending order is shown below.

Task 10

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

```
[55]: %sql SELECT Landing_Outcome, COUNT(Landing_Outcome), Date FROM SPACEXTABLE GROUP BY Landing_Outcome HAVING Date BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY 2
```

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

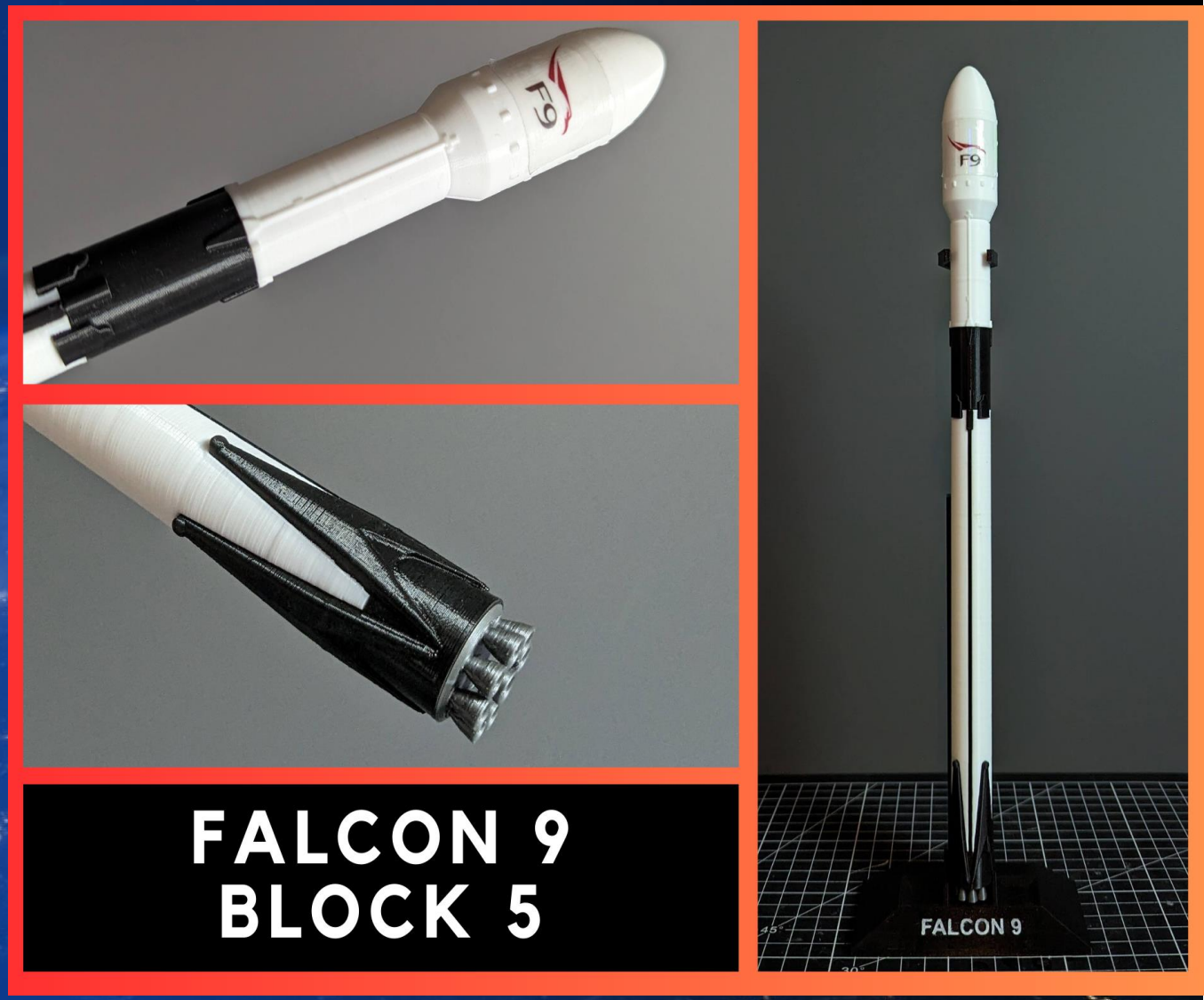
[55]:

Landing_Outcome	COUNT(Landing_Outcome)	Date
No attempt	21	2012-05-22
Success (drone ship)	14	2016-04-08
Success (ground pad)	9	2015-12-22
Failure (drone ship)	5	2015-01-10
Controlled (ocean)	5	2014-04-18
Uncontrolled (ocean)	2	2013-09-29
Failure (parachute)	2	2010-06-04
Precluded (drone ship)	1	2015-06-28

Reference Links

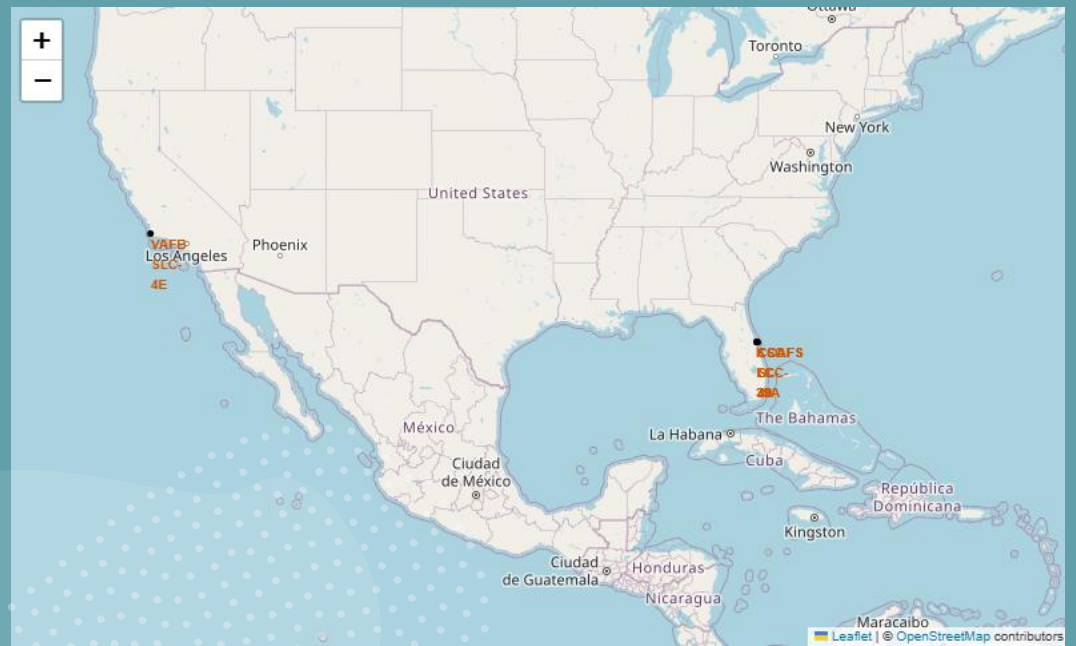
Section 3

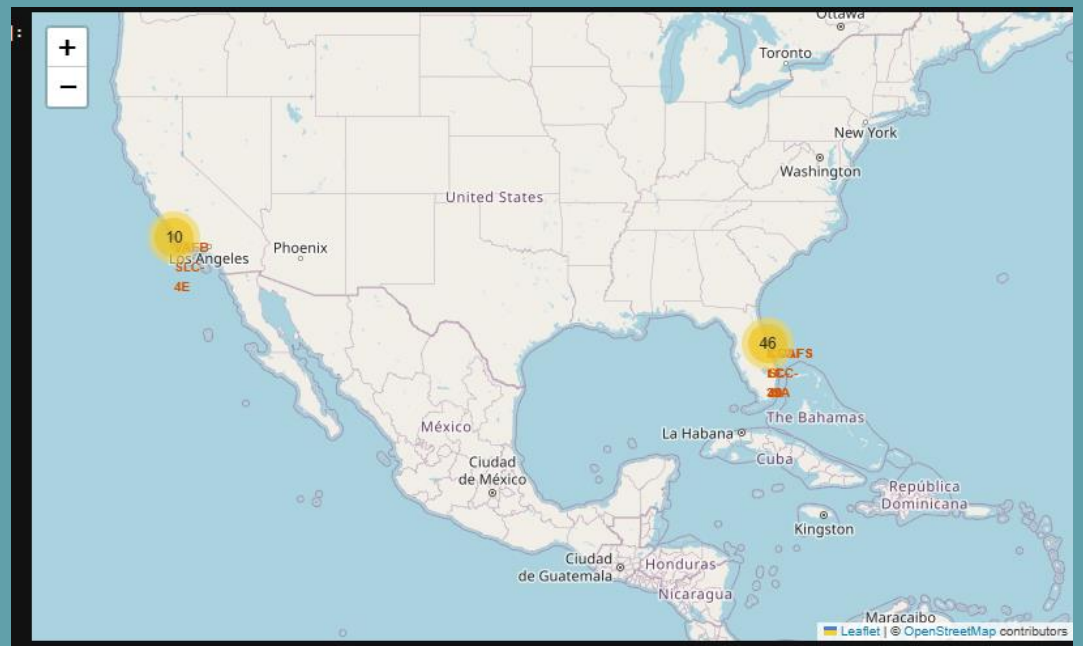
Launch Sites Proximities Analysis



USA Launch Sites in California and Florida

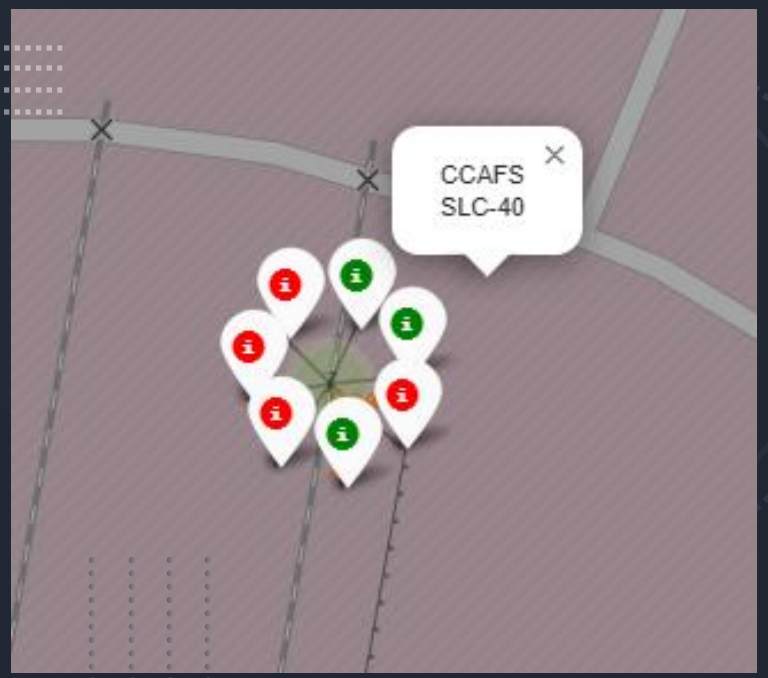
- Most of Launch sites considered in this project are in proximity to the Equator line. Launch sites are made at the closest point possible to Equator line, because anything on the surface of the Earth at the equator is already moving at the maximum speed (1670 kilometers per hour). For example launching from the equator makes the spacecraft move almost 500 km/hour faster once it is launched compared half way to north pole.
- All launch sites considered in this project are in very close proximity to the coast While starting rockets towards the ocean we minimize the risk of having any debris dropping or exploding near people.

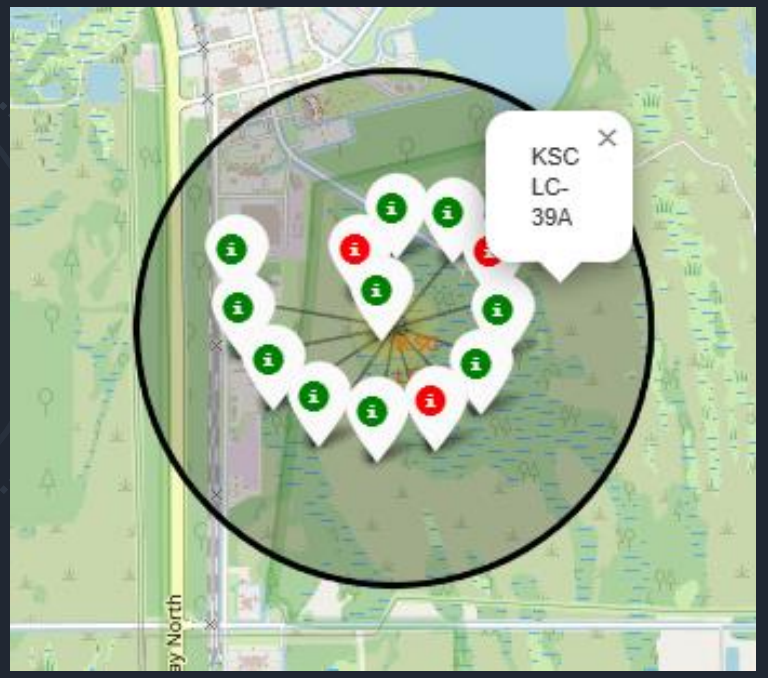


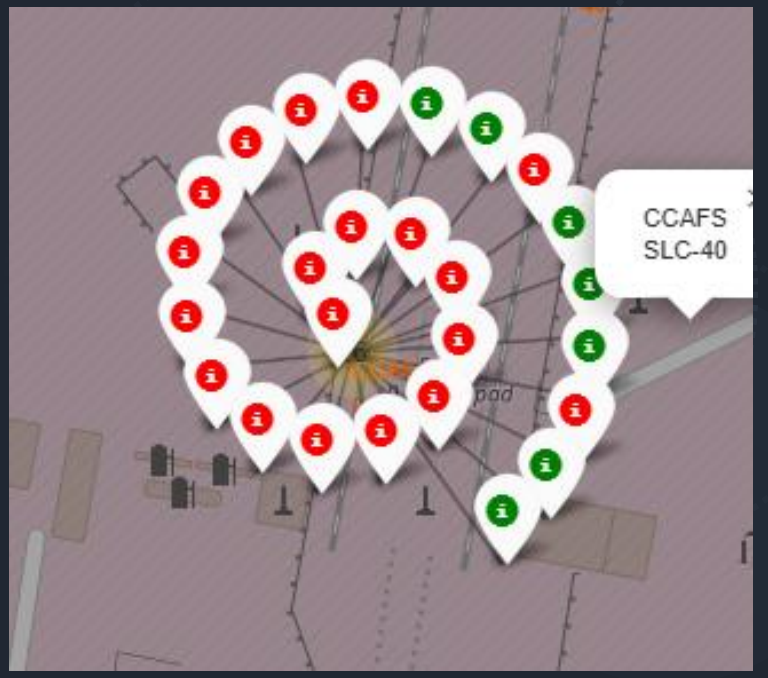


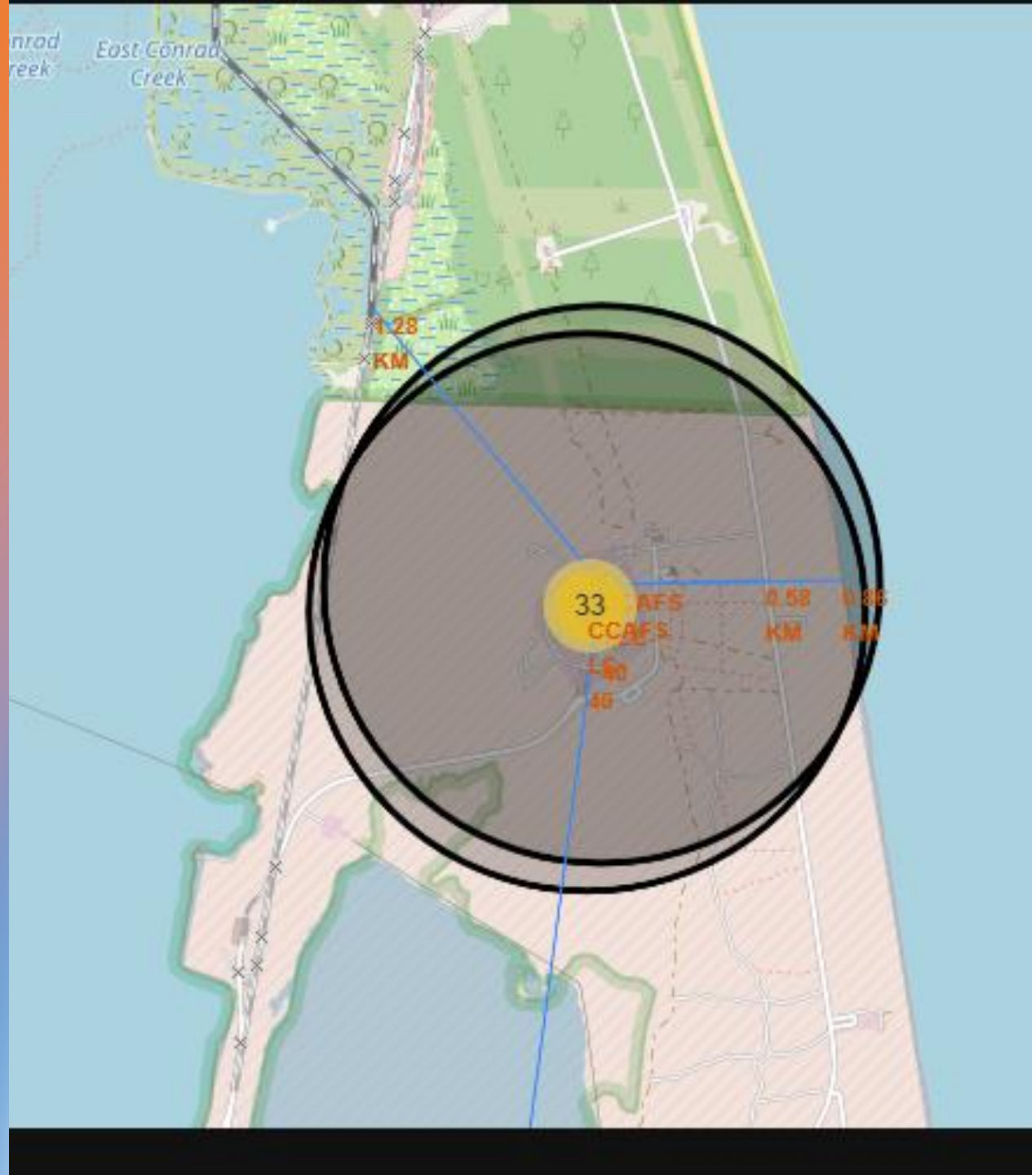
Color Labels Showing the Launch Sites on a Map

- **Green= Successful Launch**
- **Red= Failed Launch**



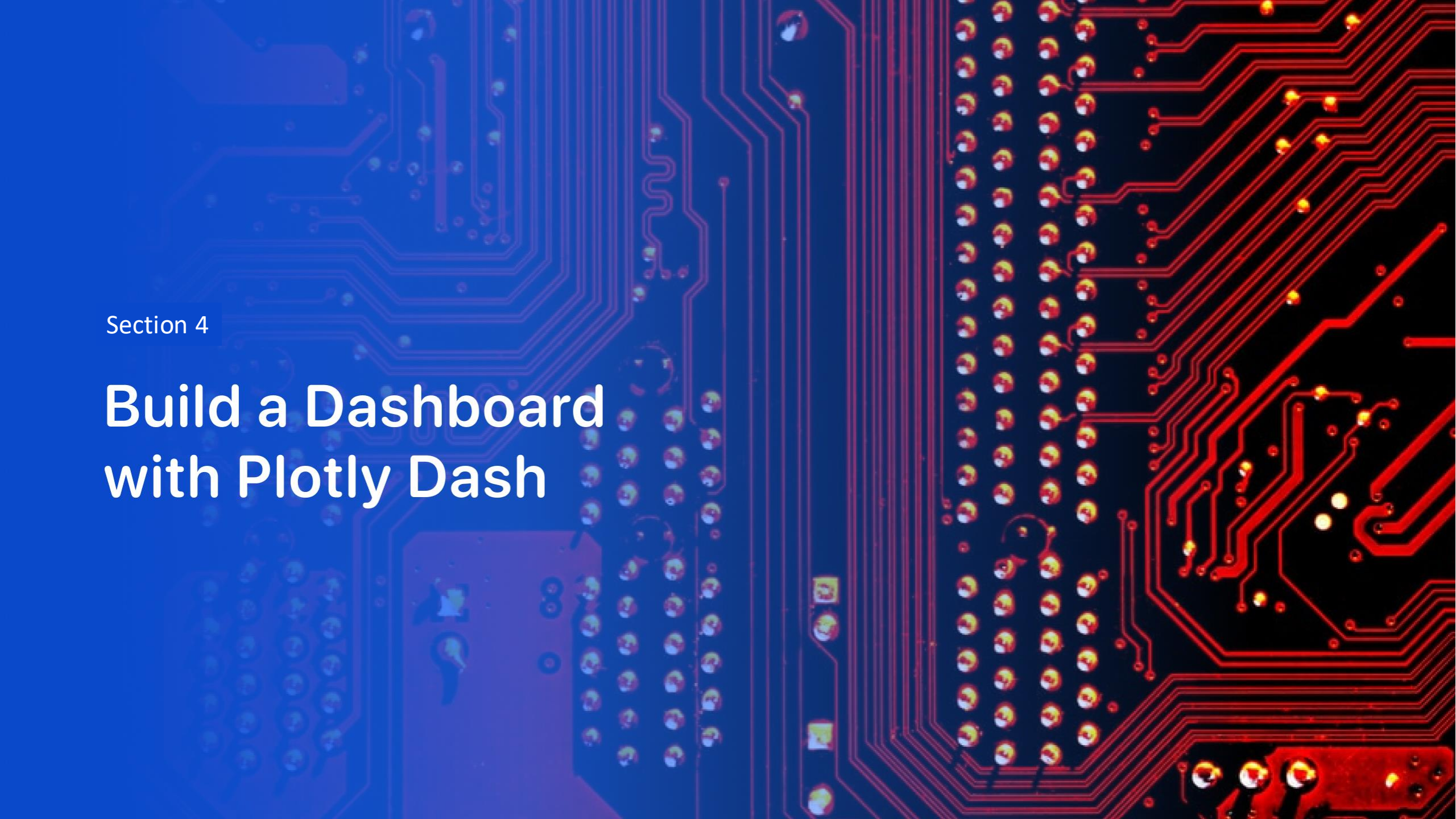


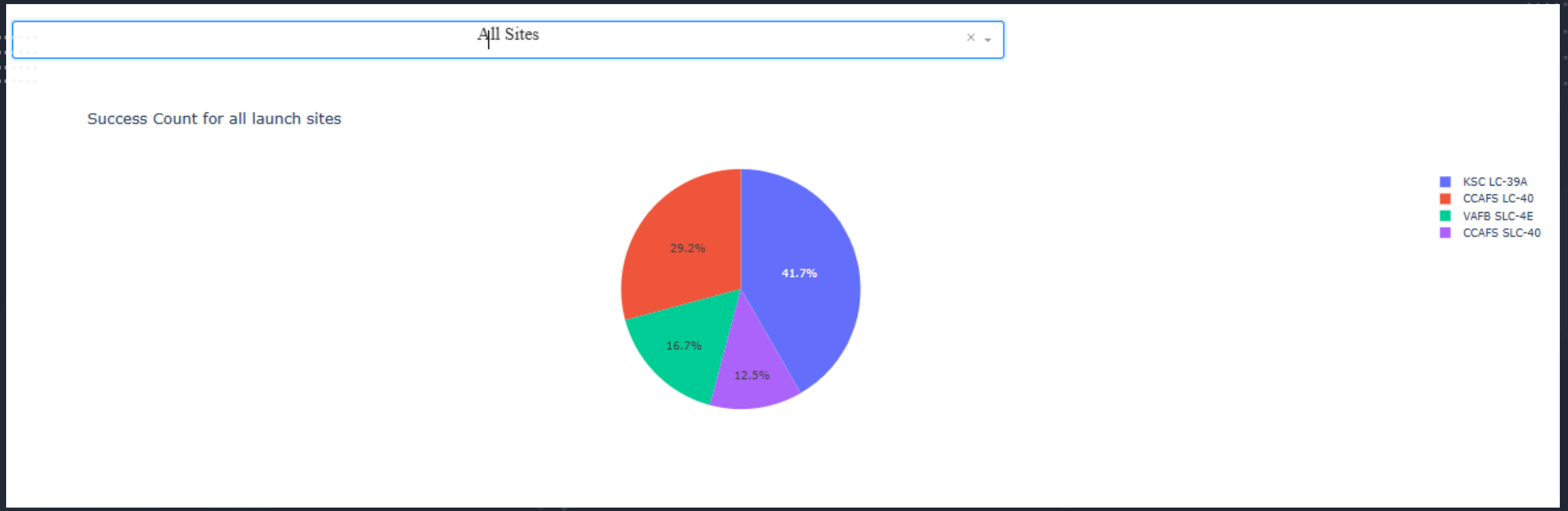




Safe Distance to Launch Site

- **The obtained results indicate that all launch sites are at safe distance from railway lines and cities.**


Section 4
Build a Dashboard with Plotly Dash

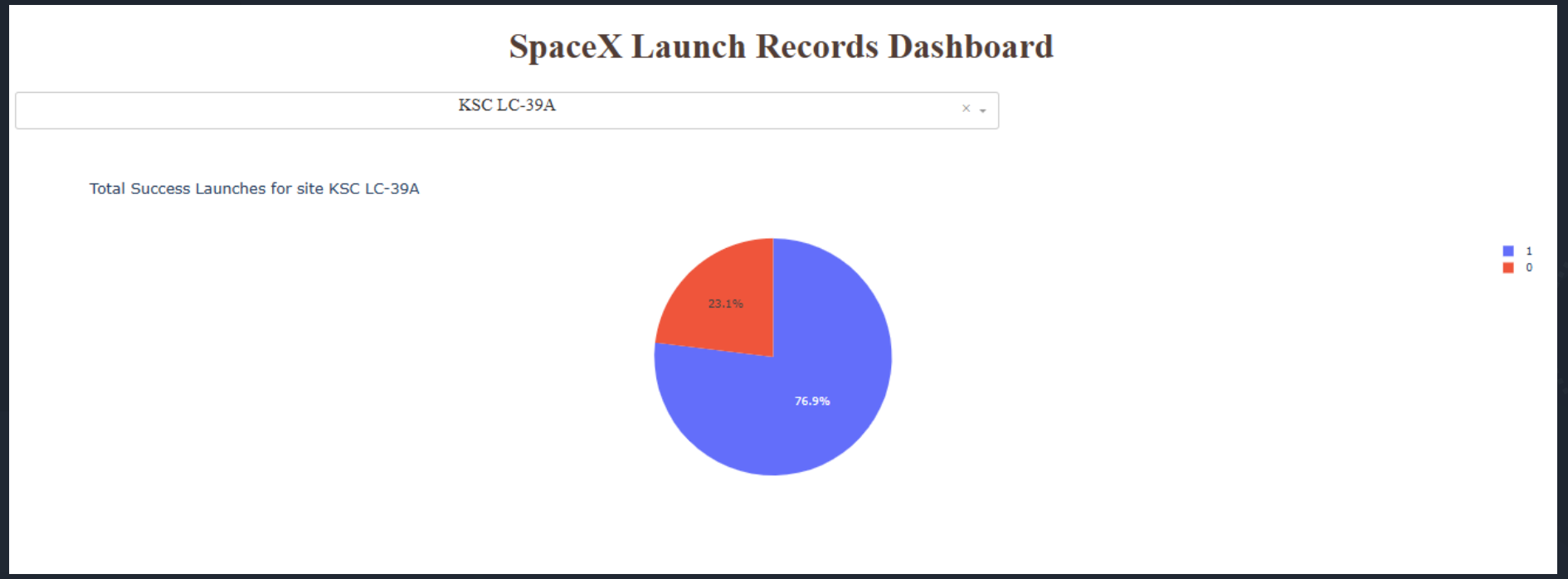


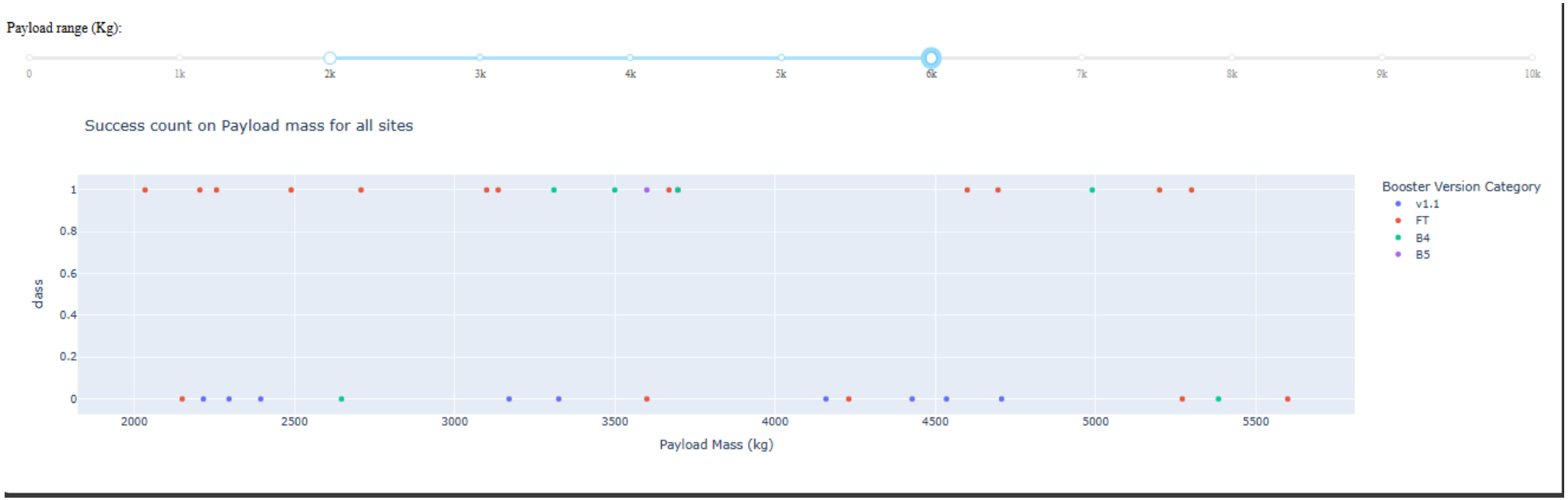
Total Launch Success for All Sites

- The highest success launch rates were recorded at these sites :
- 1.KSC LC-39A (41.7%)
- 2.CCAFS LC-40 (29.2%)

KSC LC-39 Launch Site Success Rate

Site KSC LC-39 success rate is 76.9%



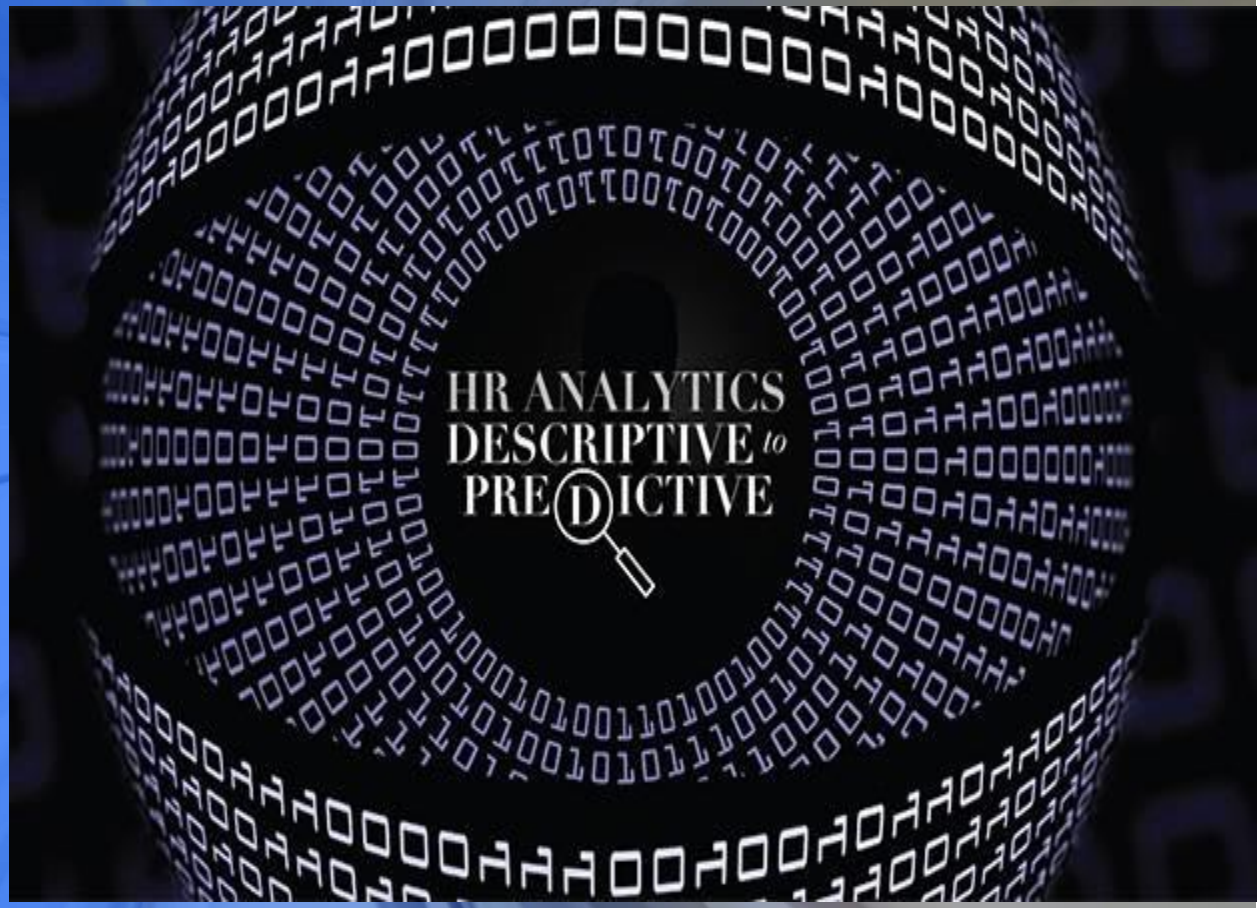


Payload vs. Launch Outcome for All Sites

Highest success rate for payloads is between 2000 and 5500 Kgs

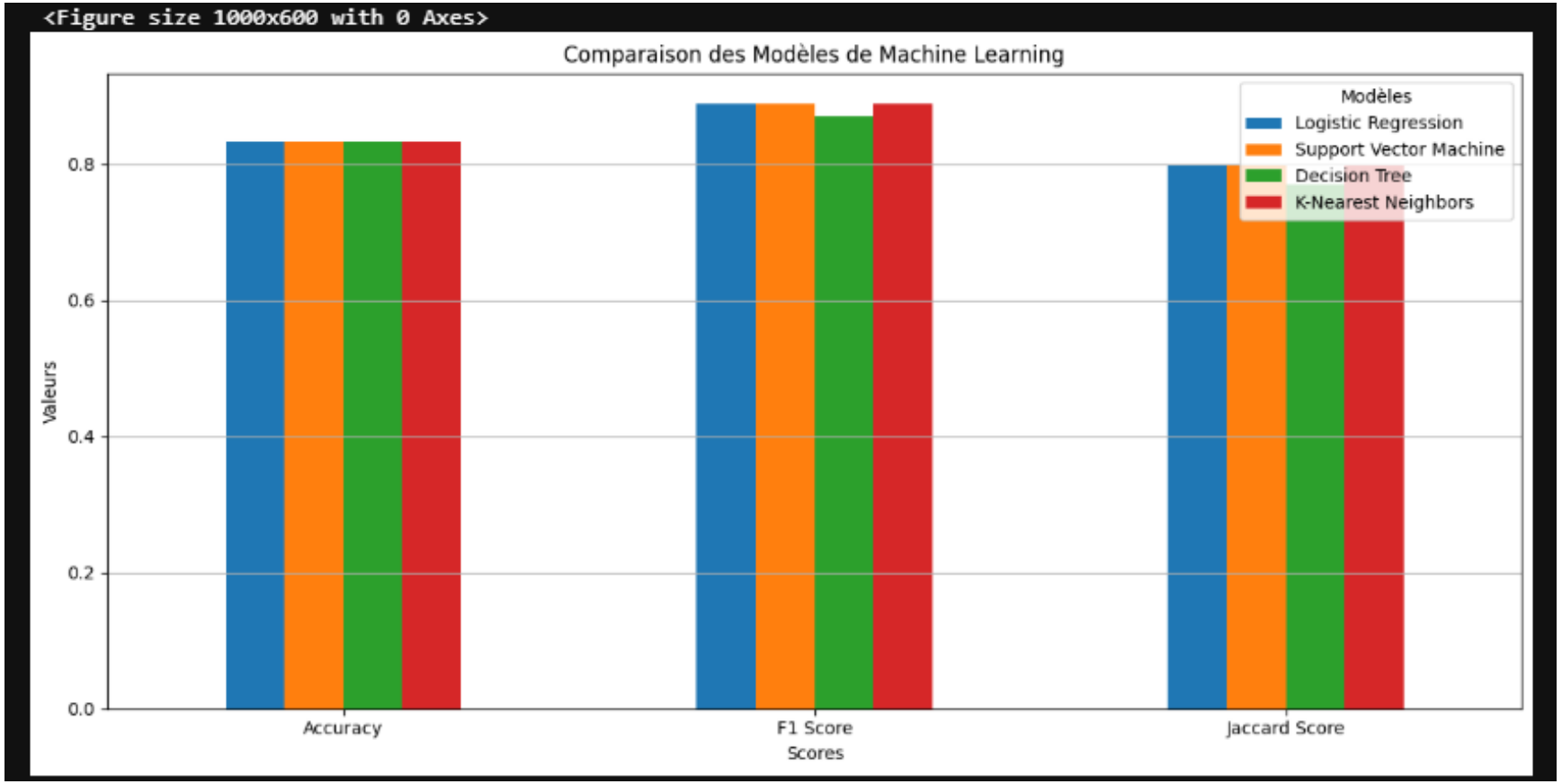
Section 5

Predictive Analysis (Classification)



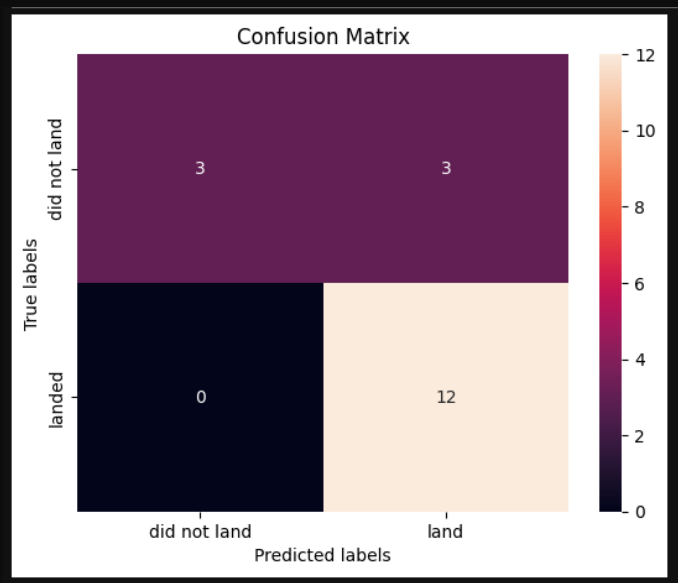
Classification Accuracy

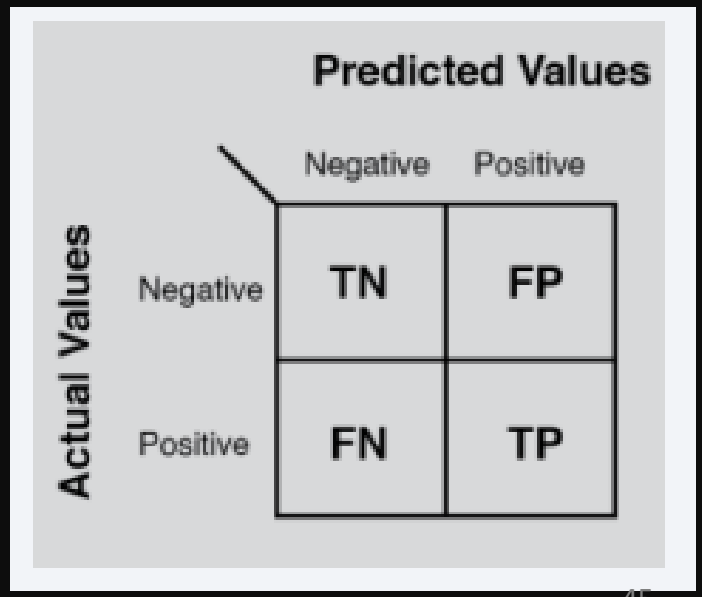
we find that the K-Nearest Neighbors model is the best for our classification.



Confusion Matrix

- The confusion matrix analysis suggests that the best performing model is the K-Nearest Neighbors model.
- The confusion matrix predicts 12 true positives, 3 false positives, 3 true positive, and 0 false negative.





Conclusions

- The success rate for the rocket launches increased
- after 2013.
- Orbits GEO, HEO, ES-L1 and SSO have 100% launch success rate.
- Launch site KSC LC-39A has the highest success rate.
- The K-Nearest Neighbors model is the best ML algorithm for analyzing the SpaceX data set and provided the best accuracy results.

Appendix

indomptablelion/IBM_cours: dans le cadre de la formation de data scientist IBM de cousera

Science des données IBM Certificat Professionnel | Coursera

Ocean2024/IBM_DS_Certificate Capstone Project_Sami Alaruri.pdf at main · ocean2024/Ocean2024

Thank you!