IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vaclav Chomat
Apr 10 2025

Outline

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

Executive Summary

- Summary of methodologies

  - SpaceX Data Collection using SpaceX API
  - SpaceX Data Collection with Web Scraping
  - SpaceX Data Wrangling
  - SpaceX Exploratory Data Analysis using SQL
  - Space-X Exploratory Data Analysis DataViz Using Python Pandas and Matplotlib
  - Space-X Launch Sites Analysis with Folium-Interactive Visual Analytics and PlotyDash
  - SpaceX Machine Learning Landing Prediction

- Summary of all results

  - Exploratory Data Analysis results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis

Introduction

- Project background and context

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars, while other providers start at 165 million dollars each.

Essentially, much of the savings is because SpaceX can reuse the first stage.

We want to determine if the first stage lands. Based on that, we can determine the cost of a launch, and we can use it to compare the cost per a launch with other companies.

- Problems we want to find answers

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

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

- Data was collected using a SpaceX RESTful API by making a get request to the SpaceX API. This was achieved by defining a series of helper functions to extract information using identification numbers from the launch data.
- Next, SpaceX launch data was parsed using the GET request and then the content was saved in a JSON format, that was converted into a Python-Pandas data frame.
- Finally, web scraping methods have been used to collect Falcon 9 historical launch records from a Wikipedia page **List of Falcon 9 and Falcon Heavy launches**. We have used BeautifulSoup and Request Python modules to extract the Falcon 9 launch records from the Internet, parsed the table, and converted it into a Python-Pandas data frame.

Data Collection – SpaceX API

- Data was collected using a SpaceX RESTful API by making a get request to the SpaceX API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

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

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

```
response=requests.get(static_json_url)
```

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

GitHub URL of the completed notebook SpaceX API calls

https://github.com/Vac9430502138/IBM_DS_2025/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

- Web scraping methods have been used to collect Falcon 9 historical launch records from a Wikipedia page **List of Falcon 9 and Falcon Heavy launches**

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=102768
```

```
# use requests.get() method with the provided static_url
# assign the response to a object

response = requests.get(static_url)
```

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`

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

GitHub URL of the completed notebook SpaceX web scraping

https://github.com/Vac9430502138/IBM_DS_2025/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

- A Pandas dataframe was created from the collected data. Then, data was filtered using the Booster Version column to keep the Falcon 9 launches only. Next, we have used Exploratory Data Analysis to find patterns in the data. This was essential to determine what would be the labels for Python models.

Identify and calculate the percentage of the missing values in each attribute

```
df.isnull().sum()/len(df)*100
```

Identify which columns are numerical and categorical:

```
df.dtypes
```

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
df['Class'] = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
df['Class'].value_counts()
```

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

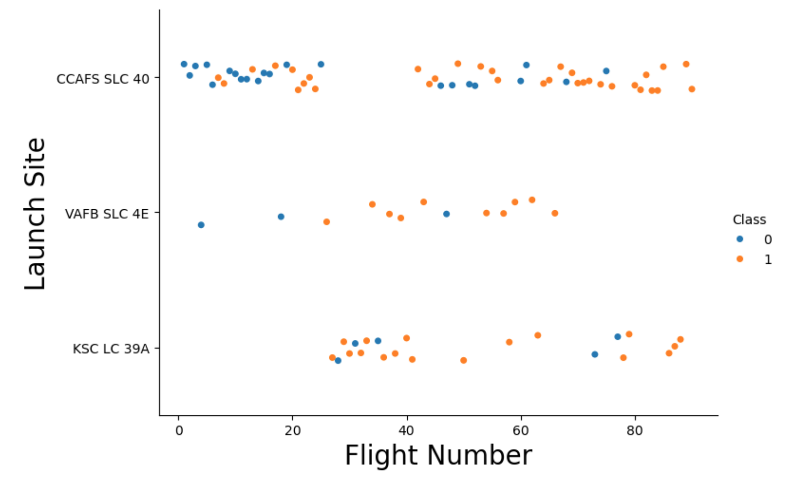
```
landing_class=df['Class']
df[['Class']].head(8)
```

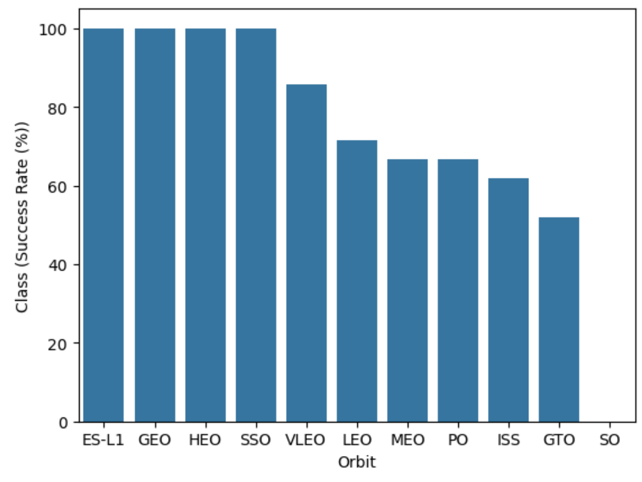
GitHub URL of the completed notebook SpaceX data wrangling

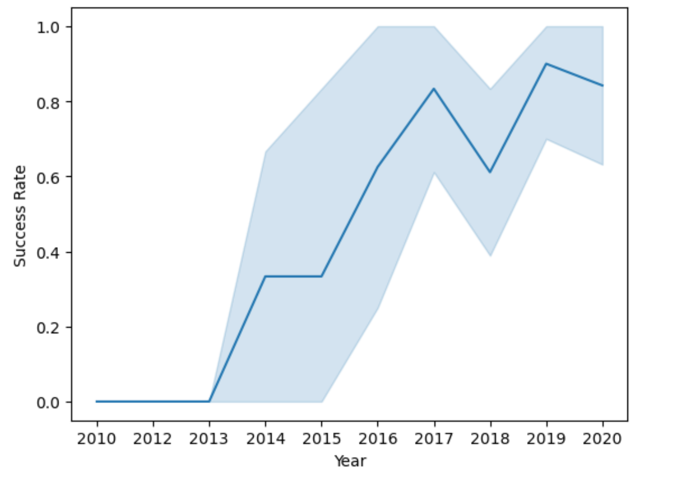
https://github.com/Vac9430502138/IBM_DS_2025/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

We have used Python Matplotlib scatter plots to Visualize the relationship between the Flight Number and a Launch Site, Payload and Launch Site, FlightNumberand Orbit type, Payload and Orbit type. Next, the bar chart was used to visualize the relationship of each orbit type. A line plot was used to visualize the launch success trend.







GitHub URL of the completed notebook SpaceX data visualization

https://github.com/Vac9430502138/IBM_DS_2025/blob/main/edadataviz.ipynb

EDA with SQL

SQL queries have been used for additional analysis:

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

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

List the total number of successful and failure mission outcomes

```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE Landing_Outcome = "Success (drone ship)" AND F
```

List the total number of successful and failure mission outcomes

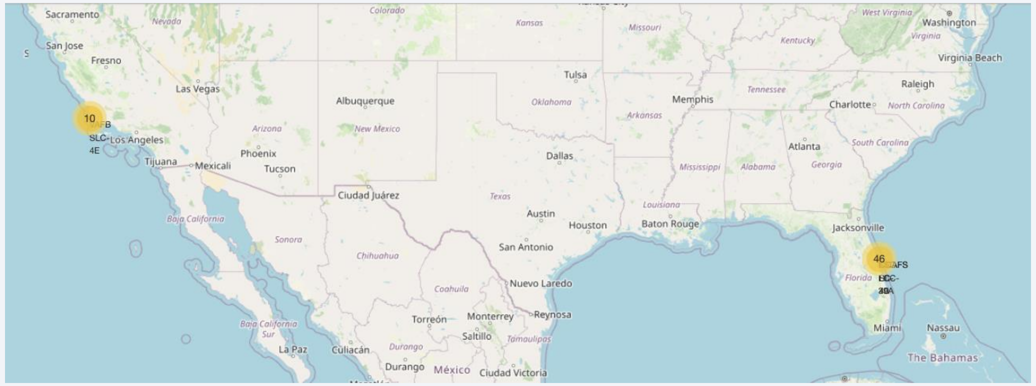
```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

GitHub URL of the completed notebook SpaceX SQL analysis

https://github.com/Vac9430502138/IBM_DS_2025/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We have used the folium maps to show all launch sites. Map objects such as markers and circles, have been used to mark the success or failure of launches for each launch site.

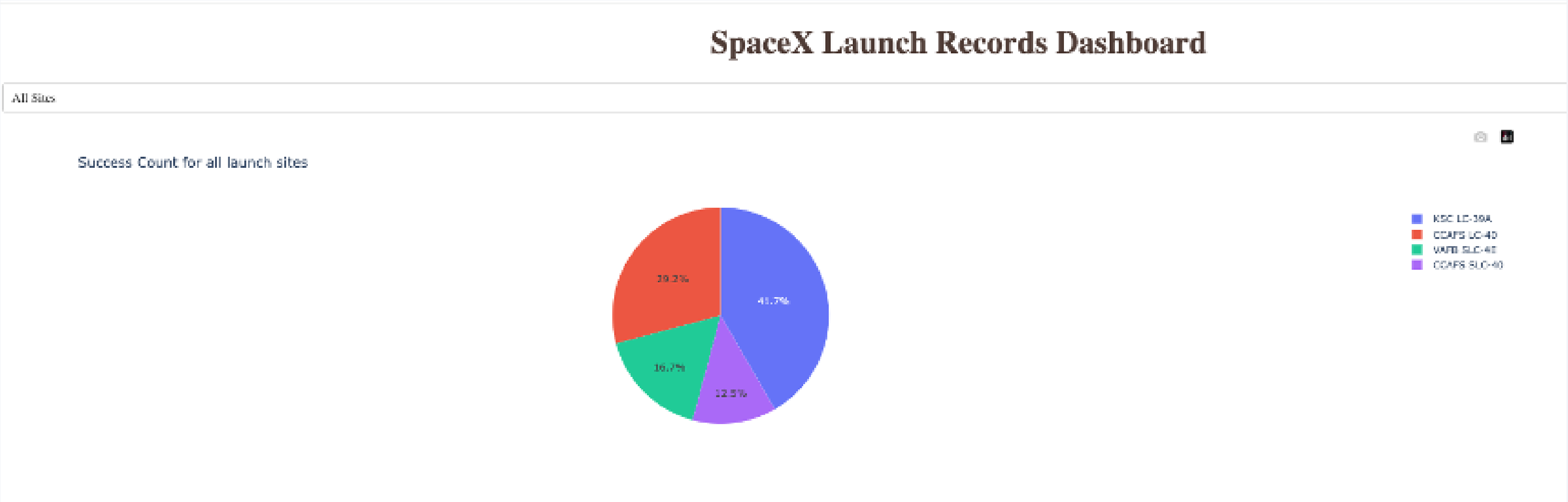


GitHub URL of the completed notebook SpaceX maps

https://github.com/Vac9430502138/IBM_DS_2025/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Plotly Dash has been used to create an interactive dashboard.



GitHub URL of the completed dashboard

https://github.com/Vac9430502138/IBM_DS_2025/blob/main/spacex-dash-app.py

Predictive Analysis (Classification)

First, we create 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. Next, we can standardize the feature dataset (x) by transforming it using preprocessing StandardScaler() function from Sklearn. Then, 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.

Create a NumPy array from the column `Class` in `data`, by applying the method `to_numpy()` then assign it to the variable `Y`,make sure the output is a Pandas series (only one bracket df['name of column']).

```
Y = data['Class'].to_numpy()
Y.dtype
```

Standardize the data in `X` then reassign it to the variable `X` using the transform provided below.

```
# students get this
transform = preprocessing.StandardScaler()

X = transform.fit_transform(X)
X
```

GitHub URL of the model

https://github.com/Vac9430502138/IBM_DS_2025/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

```
X_train, X_test, Y_train, Y_test = train_test_split( X, Y, test_size=0.2, random_state=2)
```

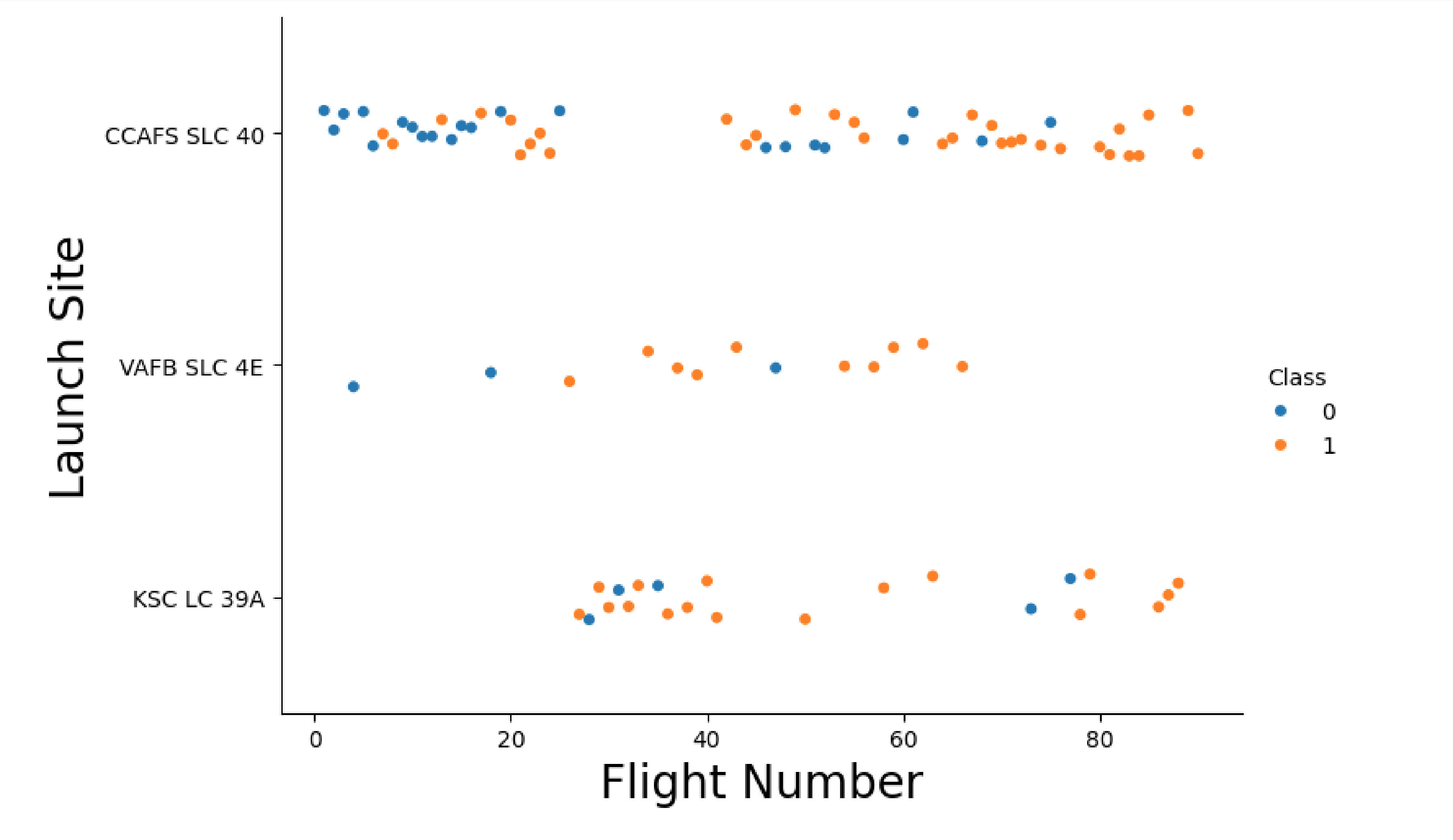
Results

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

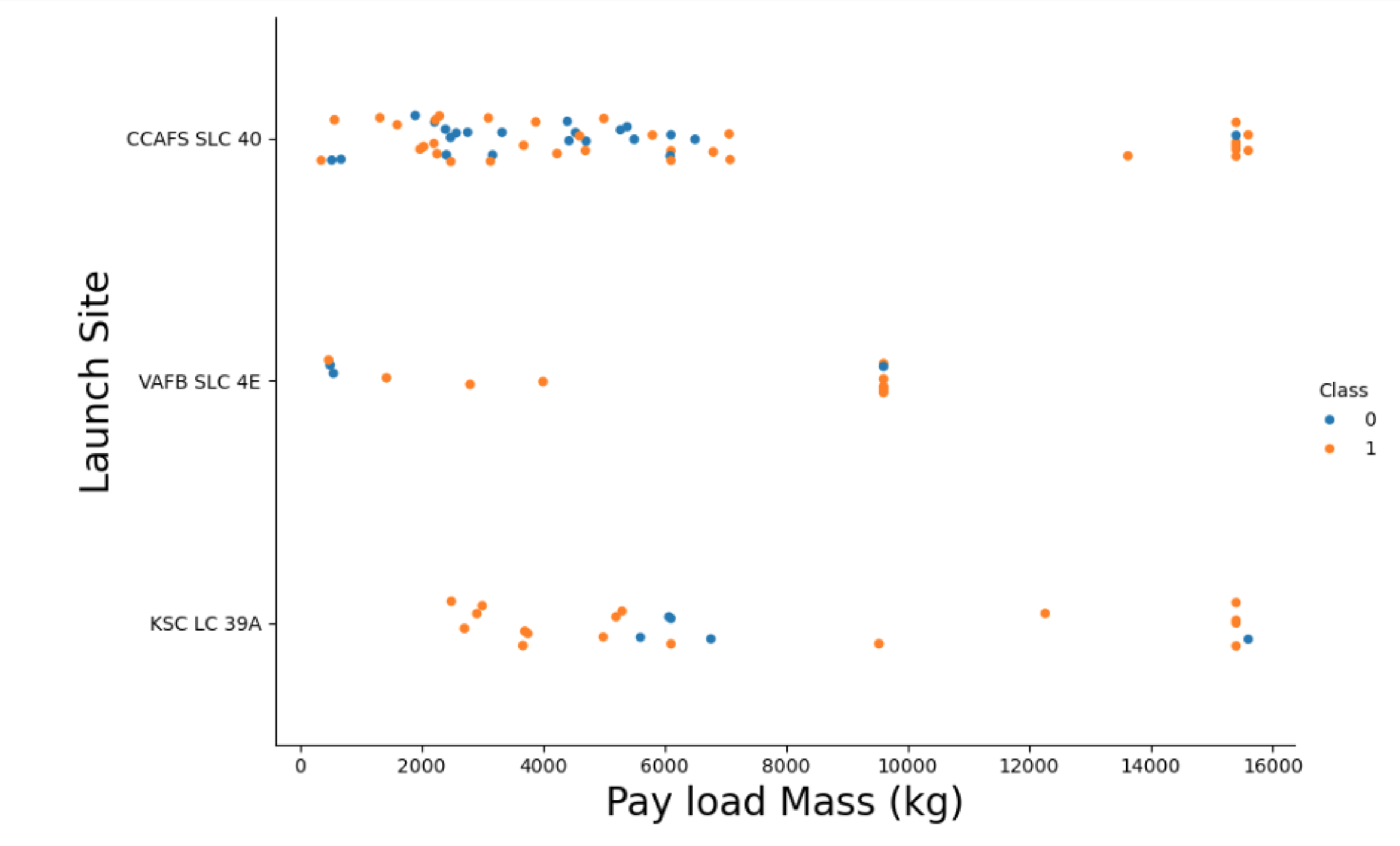
Section 2

Insights drawn from EDA

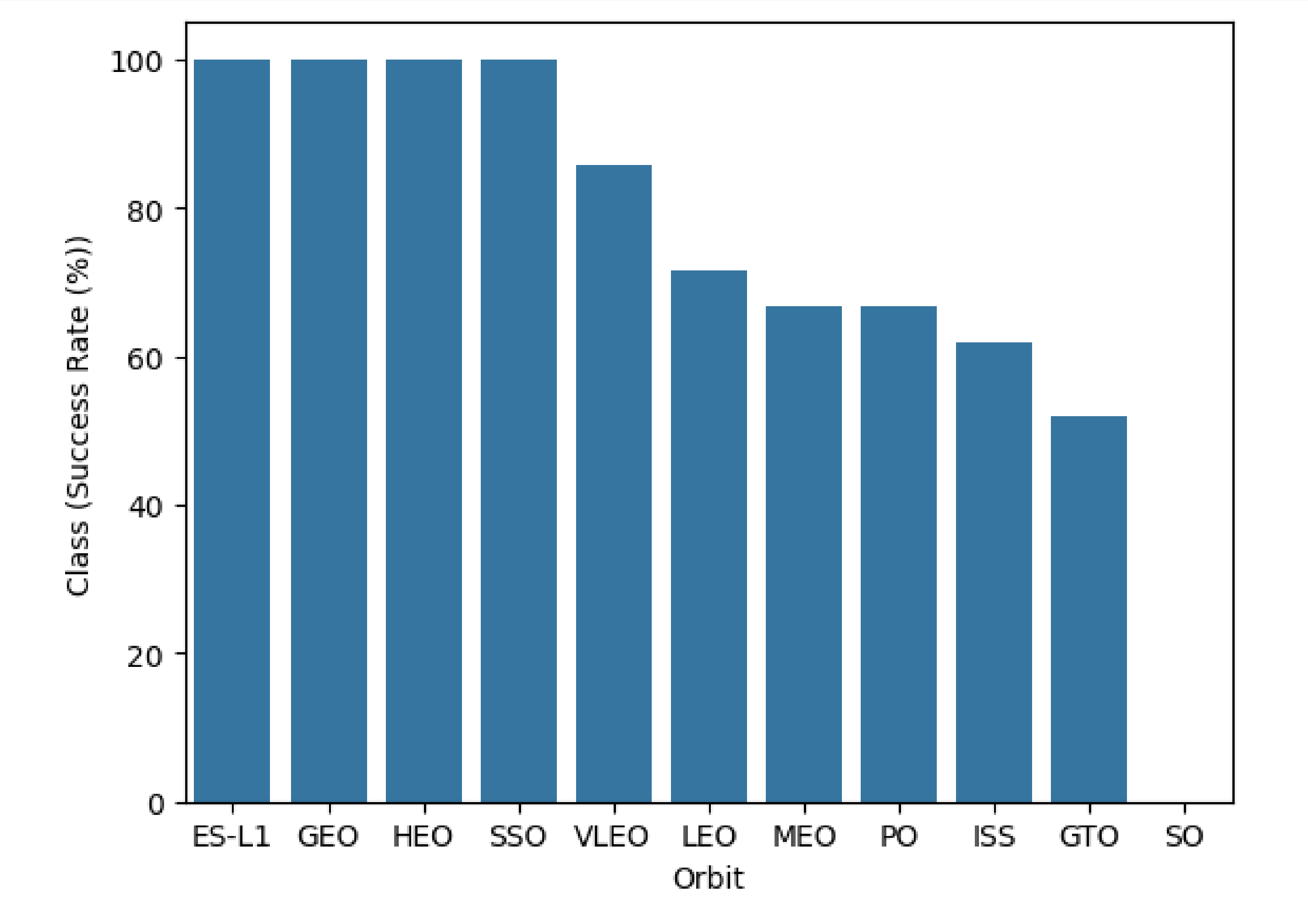
Flight Number vs. Launch Site



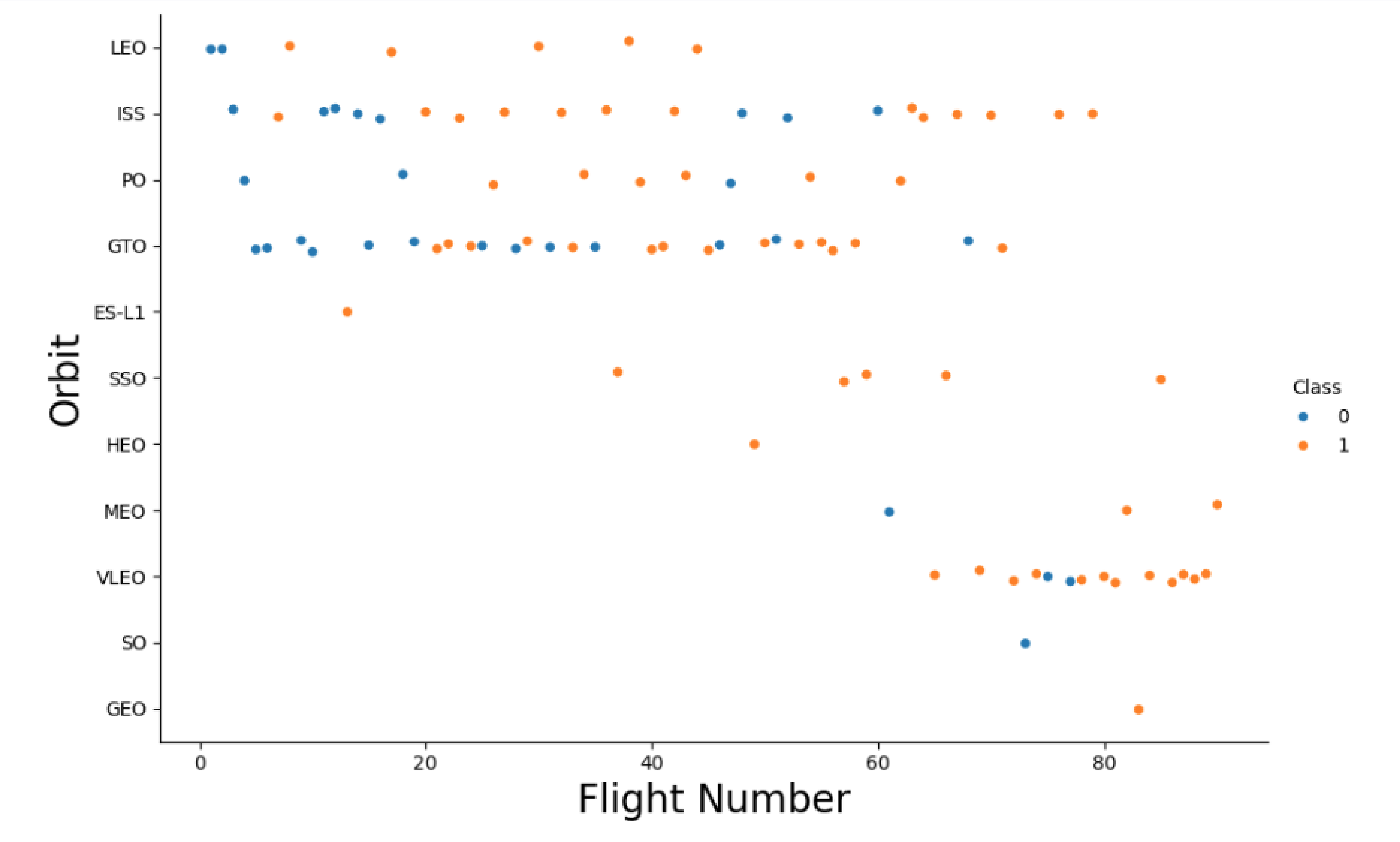
Payload vs. Launch Site



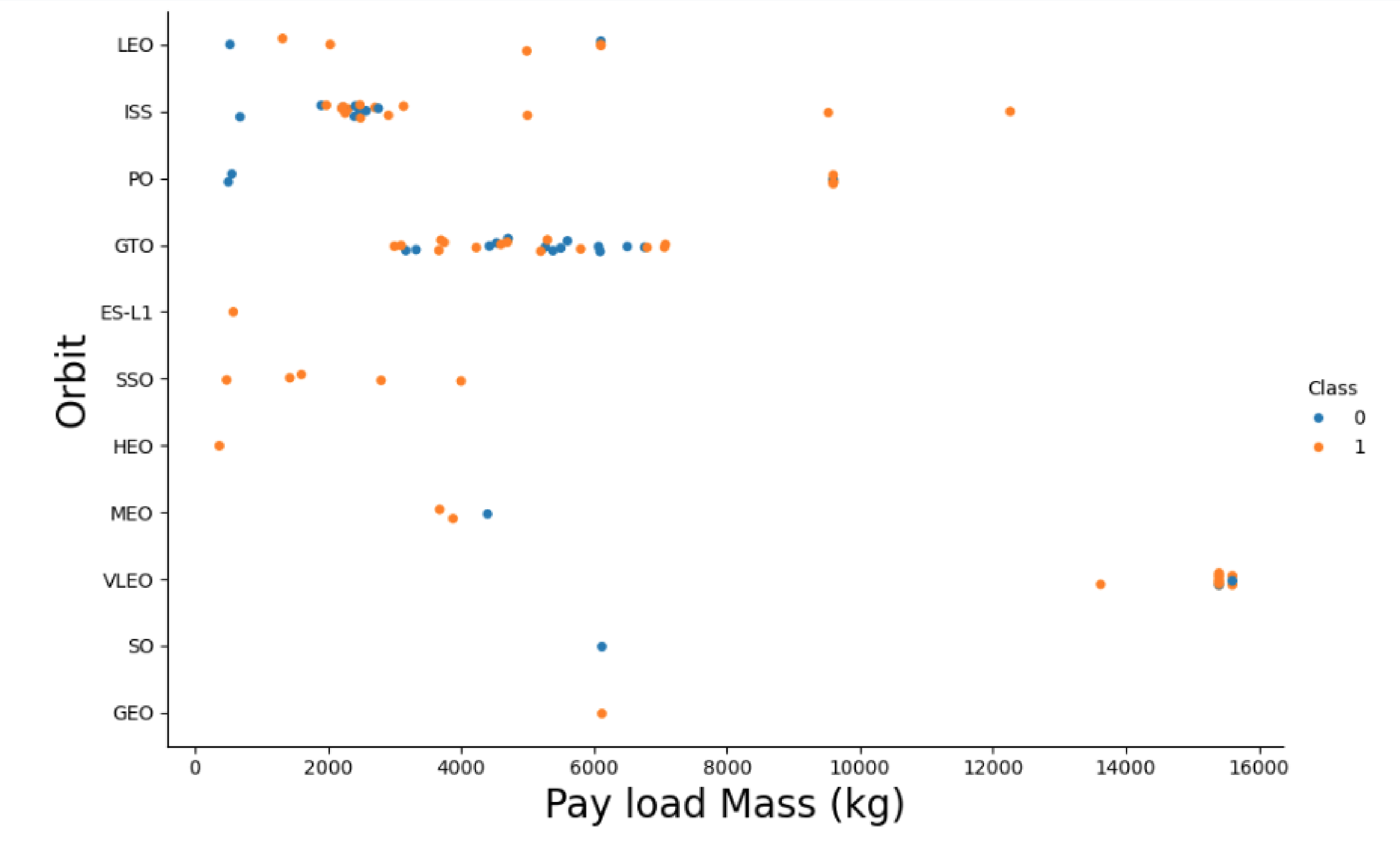
Success Rate vs. Orbit Type



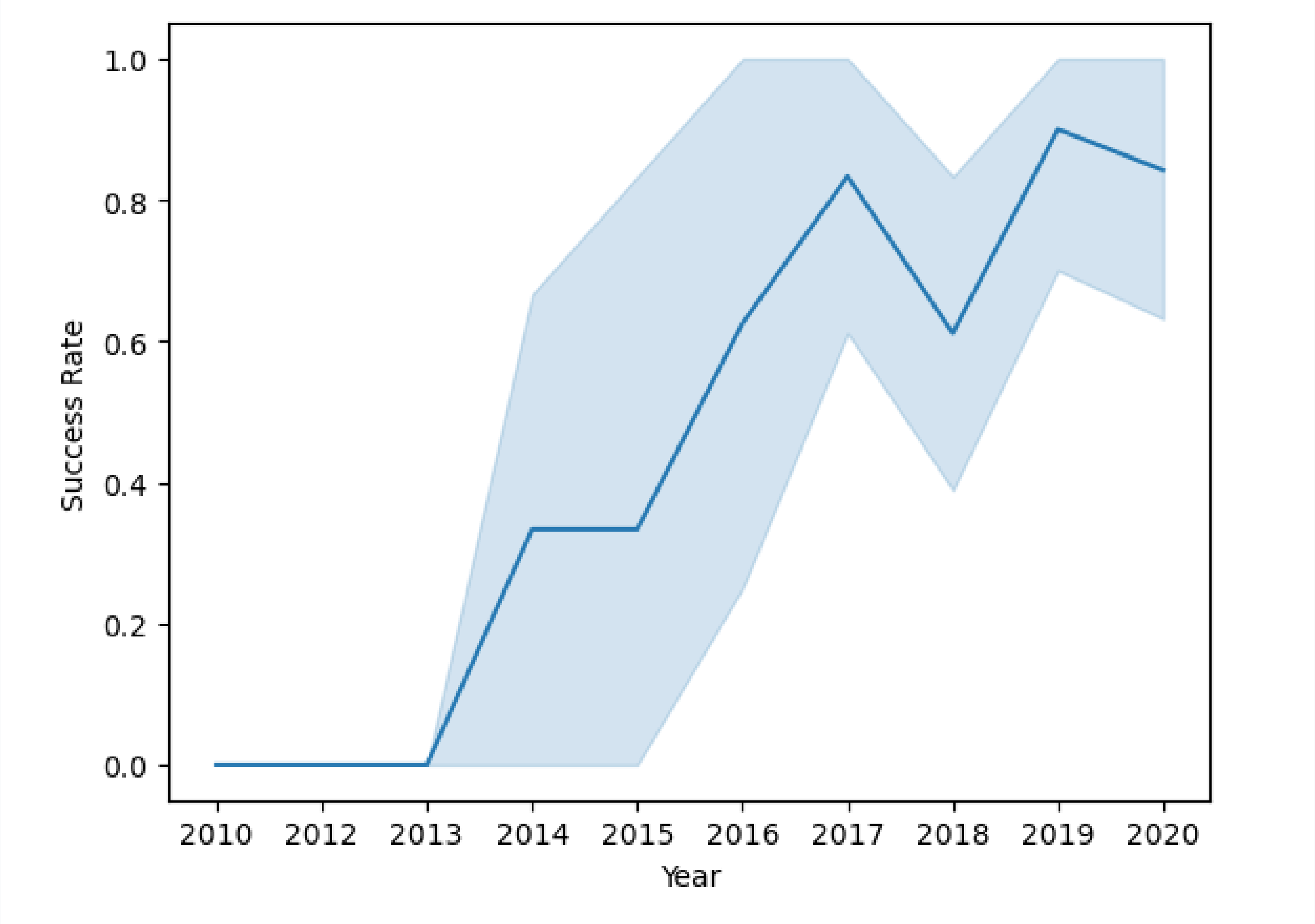
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

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

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

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

Launch Site Names Begin with 'CCA'

Display 5 records where launch sites begin with the string '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_
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (p
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 (p
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	N
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	N
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	N

Total Payload Mass

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA
```

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booste
```

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

First Successful Ground Landing Date

```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE Landing_Outcome = "Success (ground pad)";
```

MIN(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE Landing_Outcome = "Success (drone ship)" AND P
```

Booster_Version	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

Boosters Carried Maximum Payload

```
%sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX
```

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

2015 Launch Records

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.

Month	Year	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing_Outcome
01	2015	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
04	2015	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	Success	Failure (drone ship)

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.

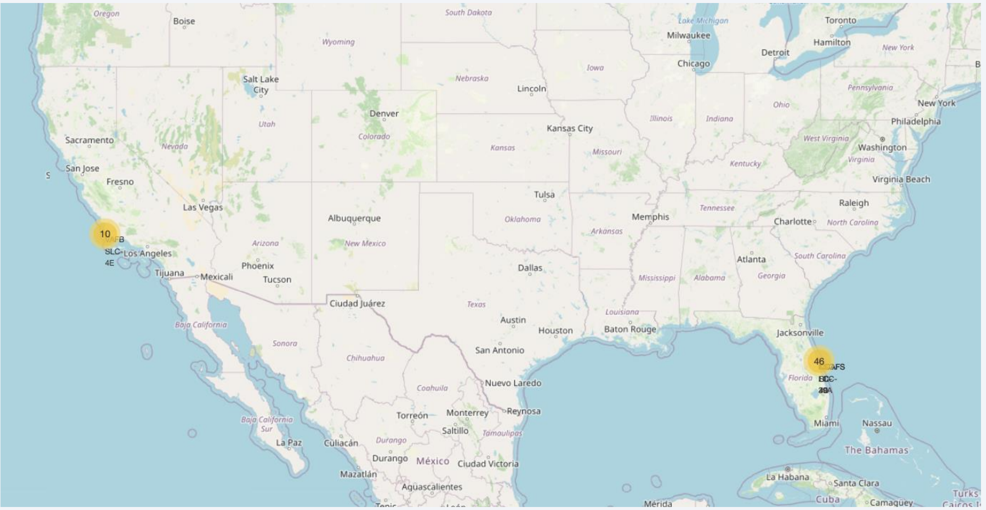
```
%sql SELECT * FROM SPACEXTBL WHERE "Landing_Outcome" LIKE 'Success%' AND Date BETWEEN '2010-06-04' AND '2017-03-2
```

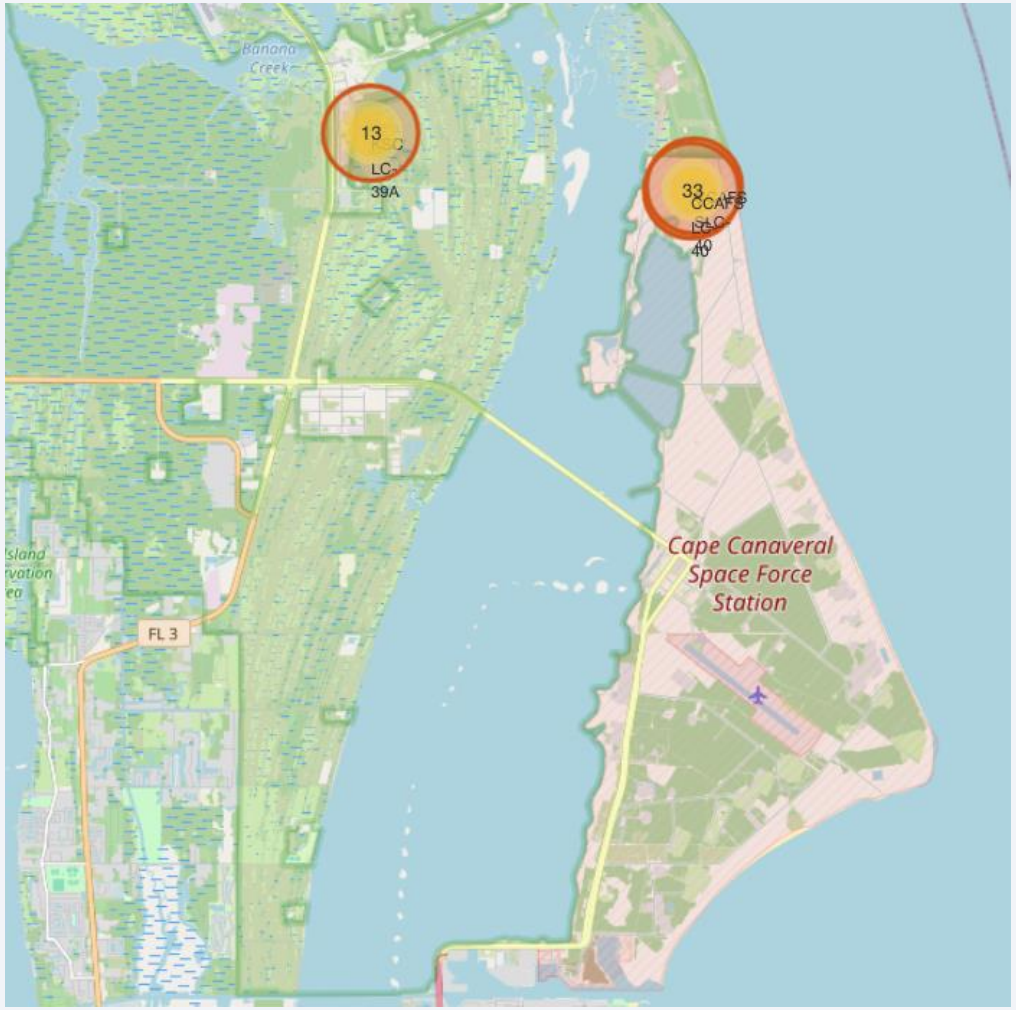
Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Lan
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Su
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	S
2016-08-14	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	S
2016-07-18	4:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Su
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	S
2016-05-06	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	S
2016-04-08	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	S
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Su

Section 3

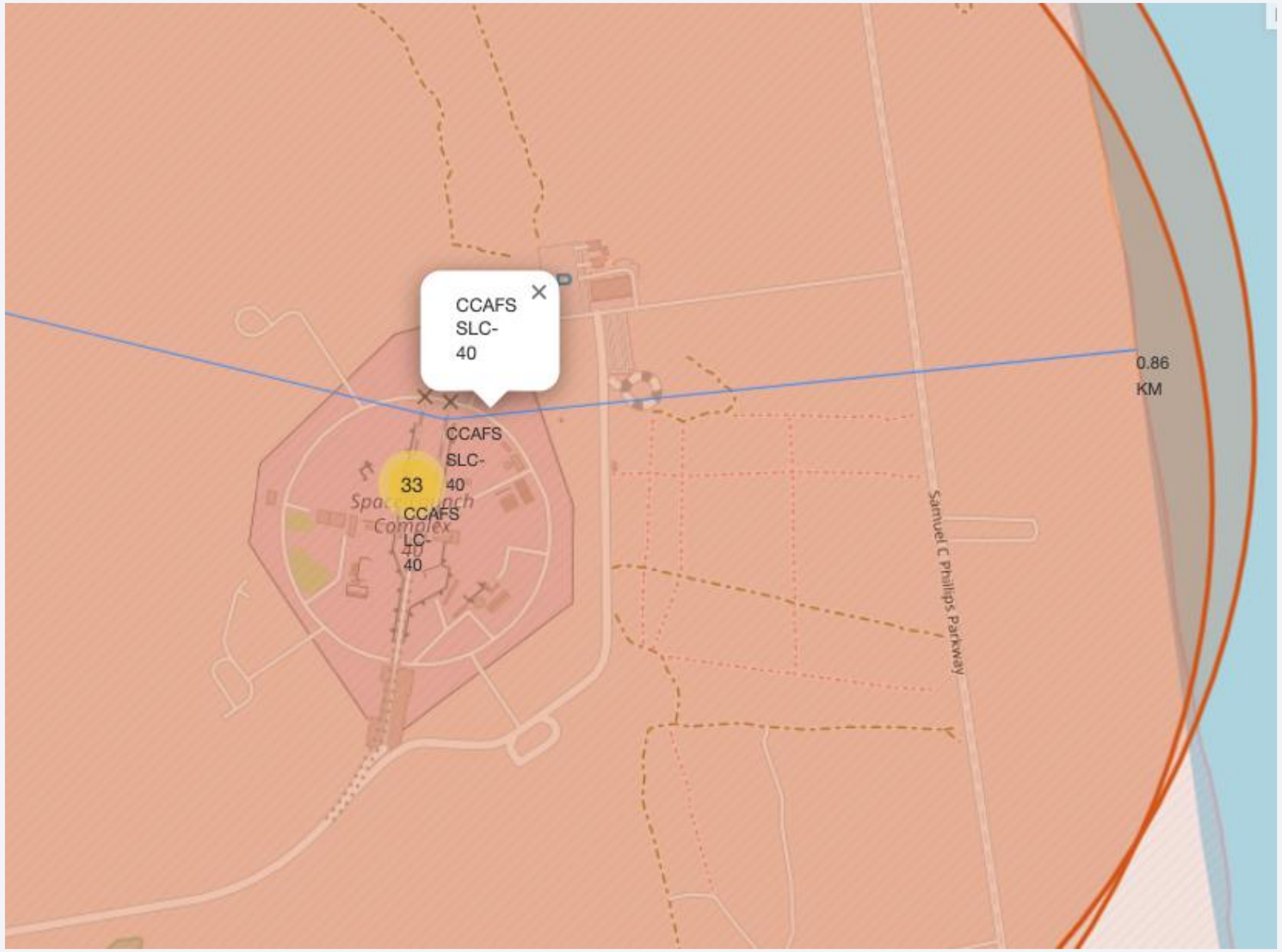
Launch Sites Proximities Analysis

Launch sites in California and Florida

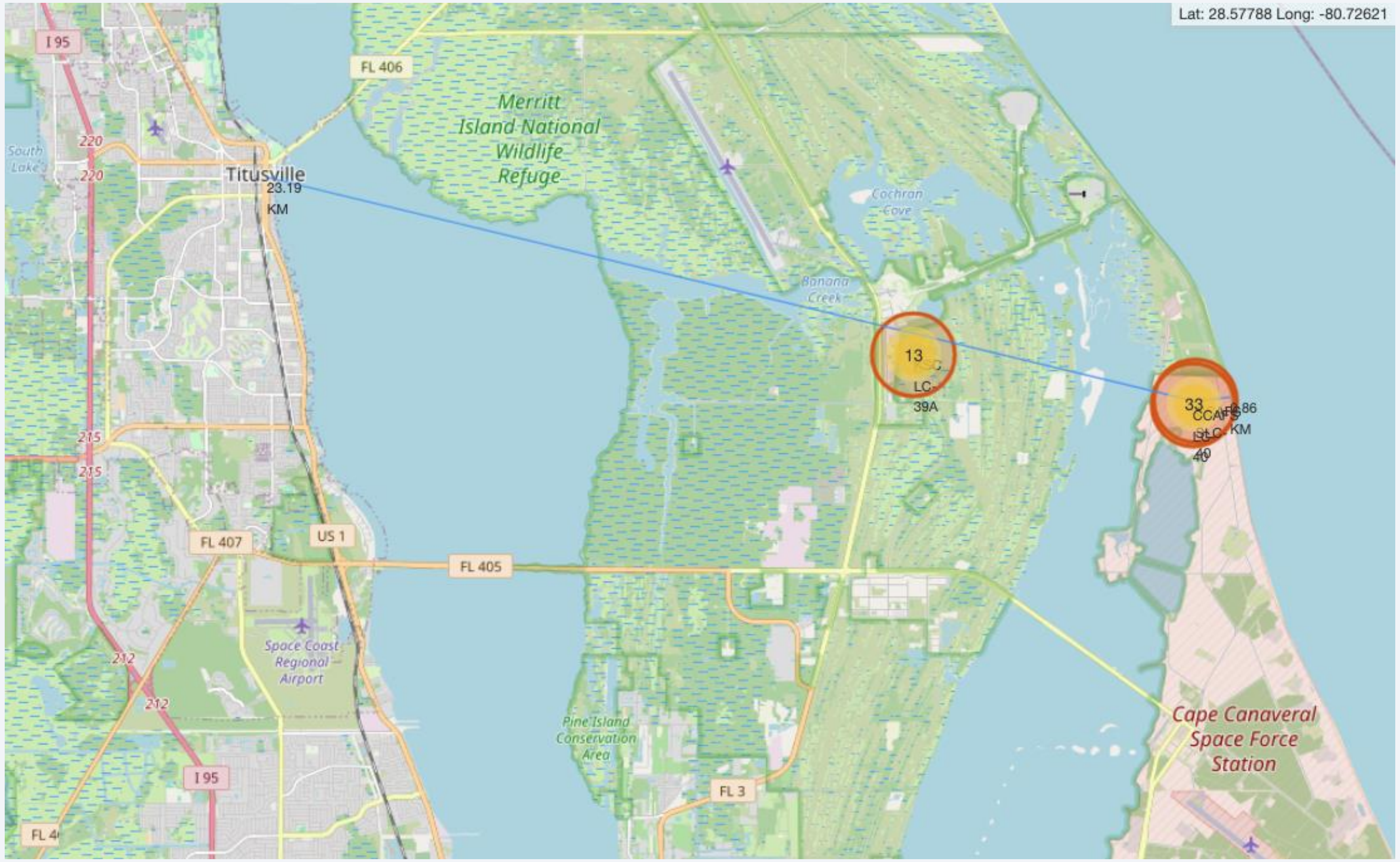




Launch site CCAFS SLC40 is 0.86 km from the coastline.



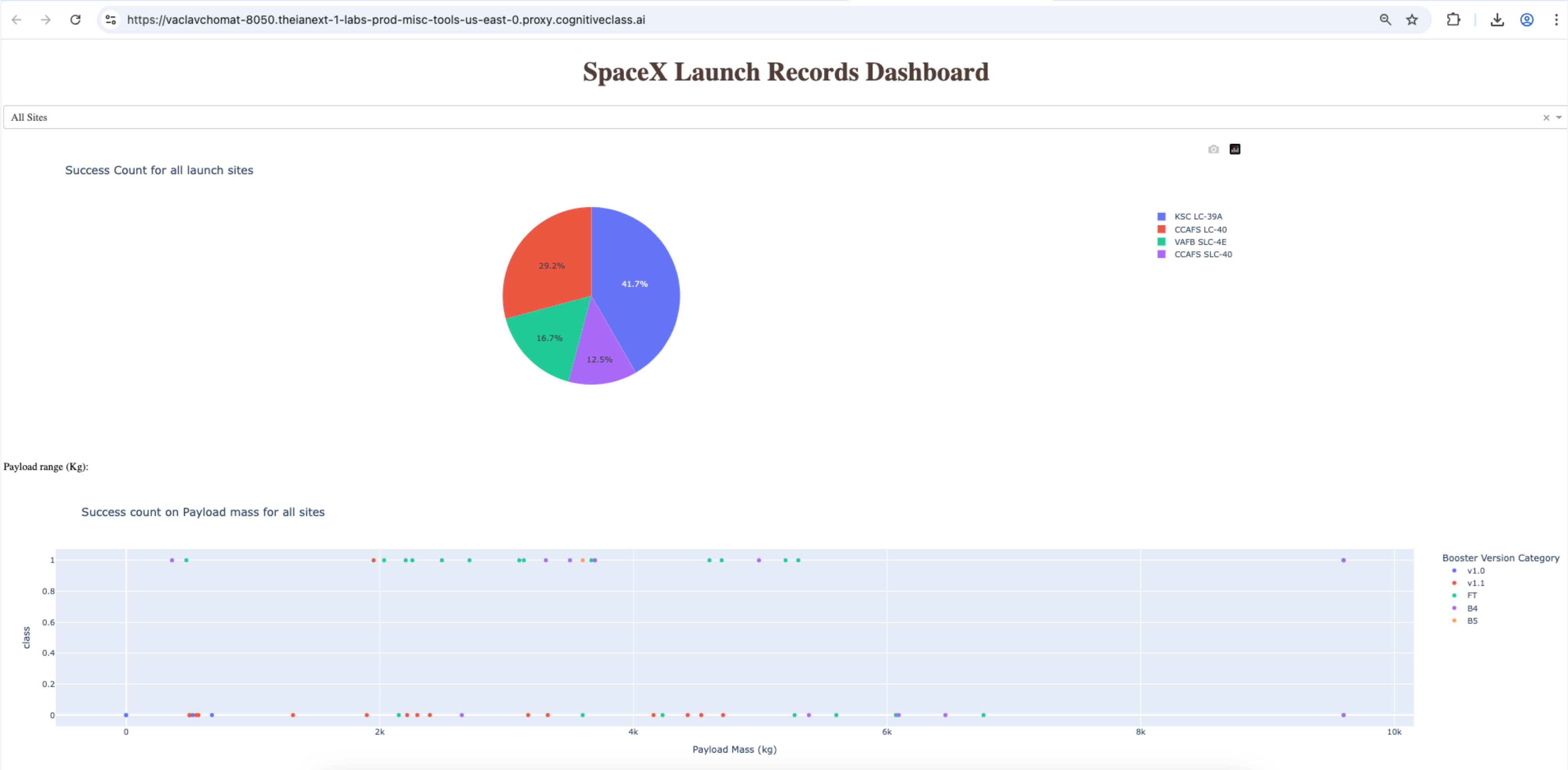
Distance between a launch site and the nearest town is 23 km.



Section 4

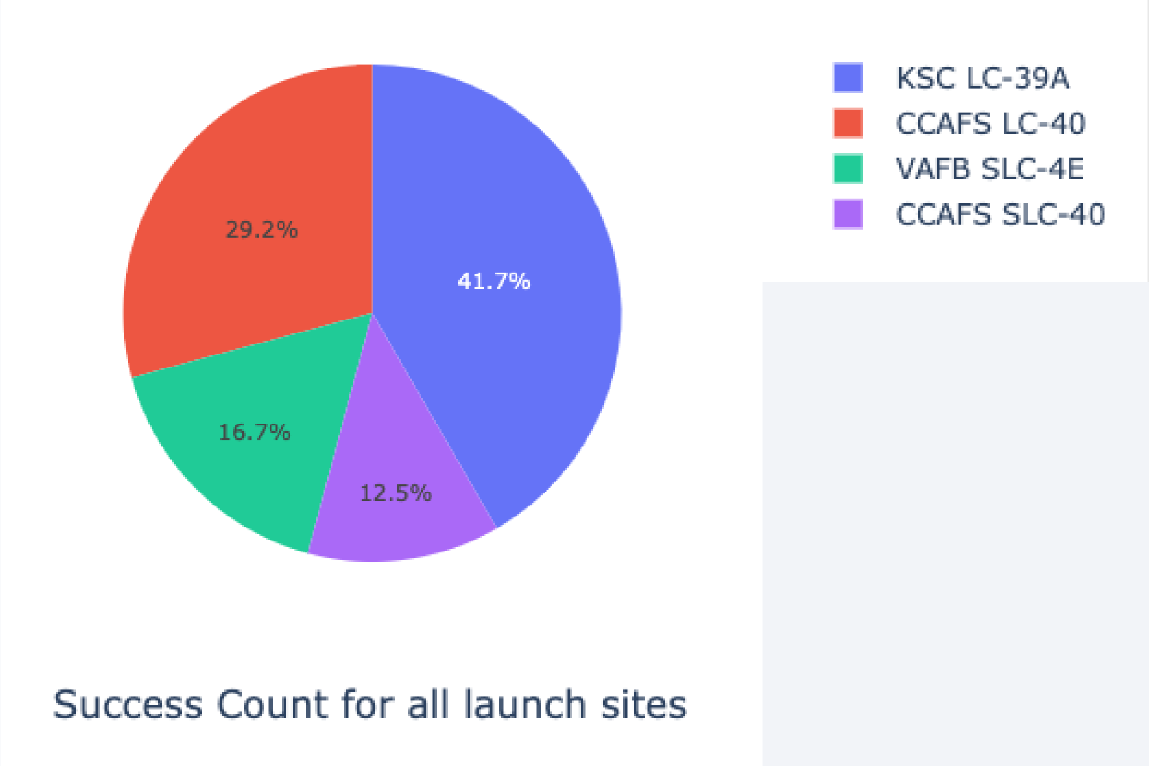
Build a Dashboard with Plotly Dash

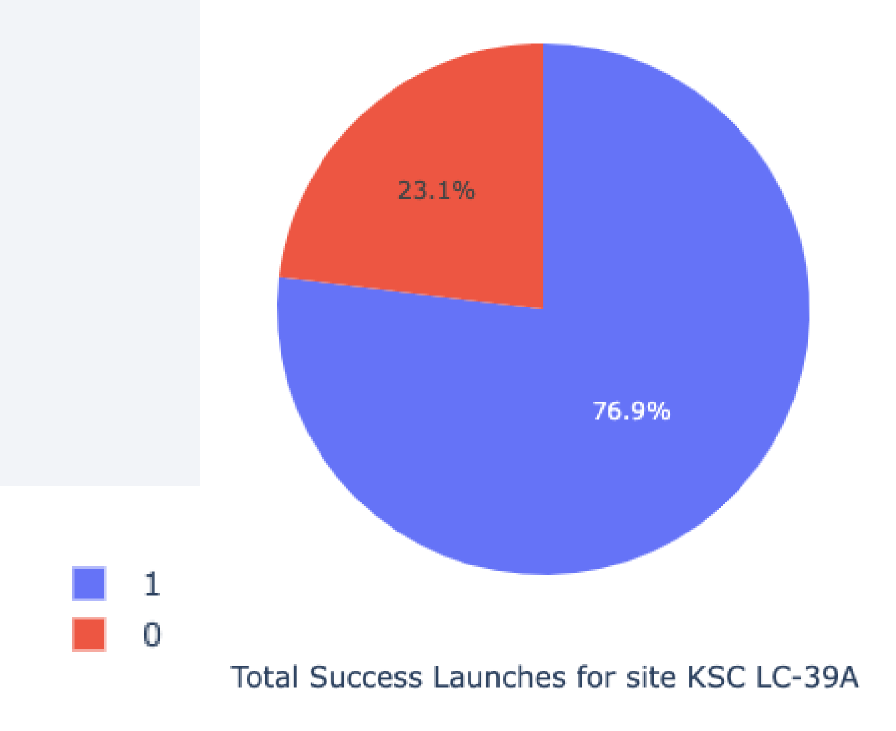
SpaceX Launch Records Dashboard



The best launch site

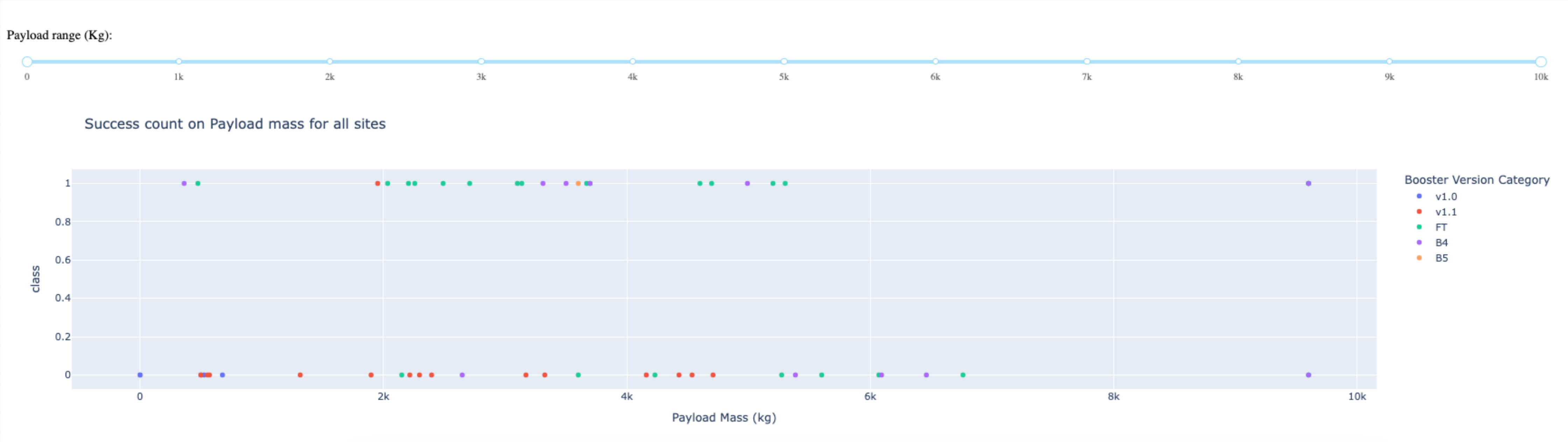
- The launch site with the highest launch success ratio is KSC LC-39A at 76.9%. This site is followed by CCAFS LC-40 at 73%.





Payload vs. Launch Outcome scatter plot for all sites

- At the launch site CCAFS LC-40 achieves the booster version FT the largest success rate for a payload mass of greater than 2000kg.



Section 5

Predictive Analysis (Classification)

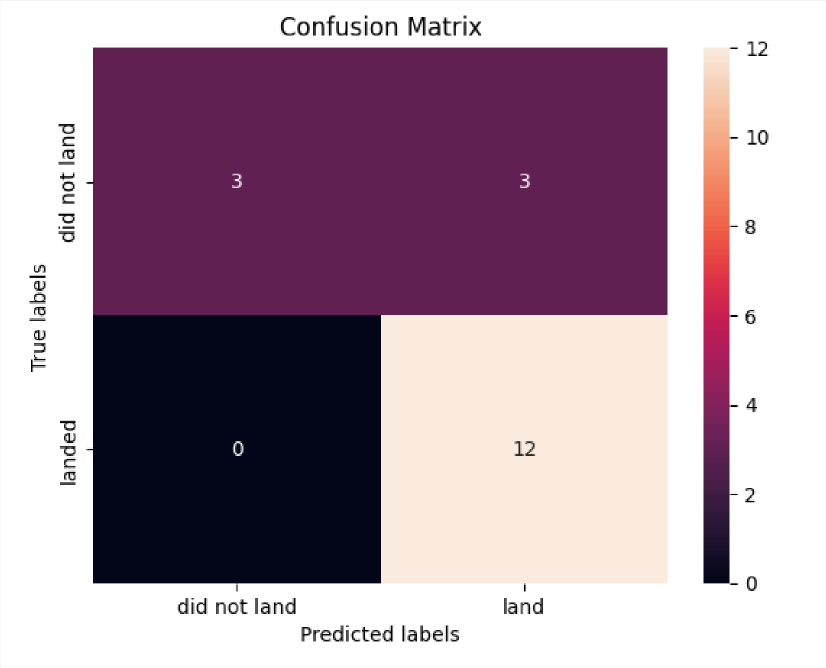
Classification Accuracy

- ✓ Logistic_Reg
- ✓ SVM
- ✓ Decision Tree
- ✓ KNN

- There is not much difference between the models. They perform equally on the test data with the accuracy of 0.83333.

Confusion Matrix

Examining the confusion matrix, we can observe that logistic regression can distinguish between the different classes. We see that the problem is false positives. There is not much difference between other techniques.



Conclusions

- Different launch sites have different success rates. Unfortunately, CCAFS LC-40, has achieved a success rate of only 60 %, while KSC LC-39A and VAFB SLC 4E have been much better at a success rate of 77%.
- Orbits ES-L1, GEO, HEO & SSO have the highest success rates , while the SO orbit has the lowest success rate.
- As the flight number increases in each of the 3 launch sites, the success rate increases.
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

Thank you!