IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

31/08/2023

Outline

Executive Summary

Summary of methodologies:

- Data collection
- Data wrangling
- EDA with Data visualization
- EDA with SQL
- Build an Interactive Map with Folium
- Build a Dashboard with Plotly Dash
- Predictive Analysis (Classification)

Summary of all results:

- EDA results (Data visualization and SQL)
- Interactives analytics results (Folium and Ploty Dash)
- Predictive Analysis results (Classification)

Introduction

Project background and context:

SpaceX 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 SpaceX 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 SpaceX for a rocket launch.
In this capstone, by using SpaceX datas from severals sources, we will predict if the Falcon 9 first stage will land successfully, and make Data Analysis to figure out data correlations on success rates.

Problems you want to find answers:

- Wich kind of rockets or Launch Site have more successful landing ?
- Wich kind of Orbits is more relevant for us to use to launch our rockets ?
- What are the accuracies of our predictions models for a successful landing ?
- How to avoid launch and landing failures ?
- How to reduces the cost of launches by Prediction and data analysis ?

Section 1

Methodology

Methodology

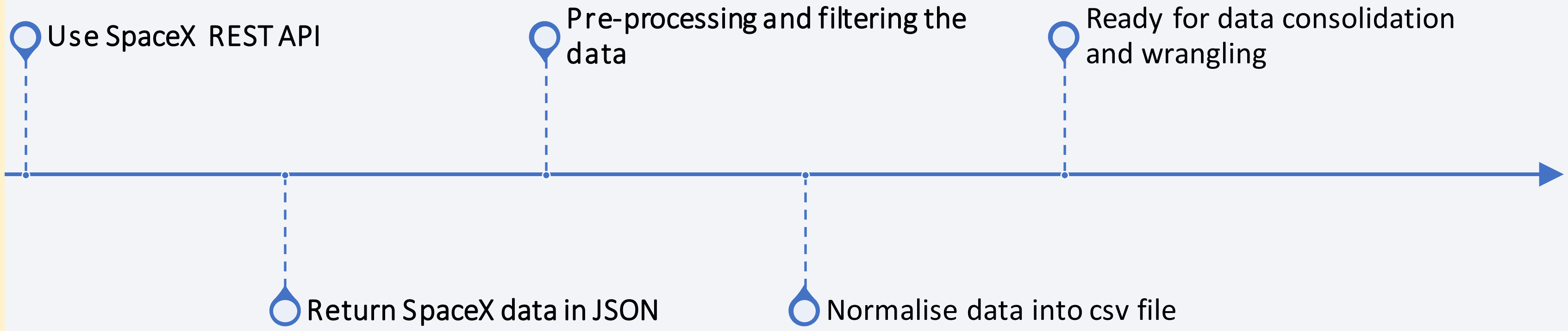
Executive Summary

- Data collection methodology:
  - Request to the SpaceX API and Web scraping from a Wikipedia page
- Perform data wrangling
  - Perform exploratory Data Analysis and determine Training Labels
- 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 Standardization, training data and test data, accuracy calculation for models, predictions on test data and confusions matrix

Data Collection

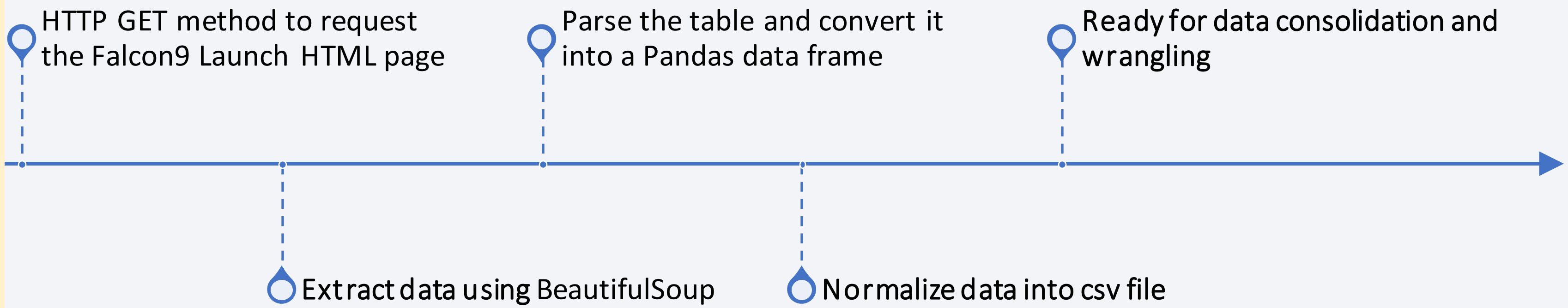
SpaceX API :

- Request to the SpaceX API : https://api.spacexdata.com/v4/
- This API provide data about rockets launches, rockets infos, launch site specifications, payload mass, and landing infos



Web Scrapping :

- Collect Falcon 9 historical launch records from a Wikipedia page : https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches



Data Collection – SpaceX API

1/ Get response from the SpaceX API

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

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

2/ Convert JSON from response to DataFrame

```
# Use json_normalize meethod to convert the
data = pd.json_normalize(response.json())
```

3/ Cleaning & filtering DF

4/ Get others data from API

```
# Call getBoosterVersion
getBoosterVersion(data)
```

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

5/ Create a DF from gathered datas

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

```
# Create a data from launch_dict
launch_df = pd.DataFrame(launch_dict)
```

6/ Filtering DF to Falcon 9 launches

```
# Filter out Falcon 1 launches and keep only Falcon 9 launches
data_falcon9 = launch_df[launch_df['BoosterVersion'] != 'Falcon 1'].copy()
```

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
```

7/ Dealing with missing values

```
payload_mass_mean = data_falcon9['PayloadMass'].mean()
data_falcon9['PayloadMass'].replace(np.nan, payload_mass_mean, inplace=True)
```

8/ Export DF to CSV

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```


GITHUB = jupyter-labs-spacex-data-collection-api.ipynb · GitHub

Data Collection - Scraping

1/ HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
falson9_lauch = requests.get(static_url)
```

2/ Create a BeautifulSoup object from the HTML response

```
soup = BeautifulSoup(falson9_lauch.text)
```

3/ Target the third table from it

```
html_tables = soup.find_all('table')
first_launch_table = html_tables[2]
```

4/ Extract column names from the table

```
th_elements = first_launch_table.find_all('th')
for th_element in th_elements:
        name = extract_column_from_header(th_element)
        if name is not None and len(name) > 0:
          column_names.append(name)
```

5/ Create empty dictionary with keys

```
launch_dict= dict.fromkeys(column_names)
del launch_dict['Date and time ( )']
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6/ Fill up dictionary from wikitable

```
Length of Flight No.: 121
Length of Launch site: 121
Length of Payload: 121
Length of Payload mass: 121
Length of Orbit: 121
Length of Customer: 121
Length of Launch outcome: 121
Length of Version Booster: 121
Length of Booster landing: 121
Length of Date: 121
Length of Time: 121
```

7/ Create DF from it

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

8/ Export DF to a CSV

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


GITHUB = [jupyter-labs-webscraping.ipynb · GitHub]

Data Wrangling

Data Wrangling is the process of converting data from the initial format to a format that may be better for analysis.

1/ Load Dataset from CSV file already created : dataset_part_1.csv

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

3/ Identify which columns are numerical and categorical (with df.dtypes)

4/ Calculate the number of launches on each site (with value_counts)

5/ Calculate the number and occurrence of each orbit (with value_counts)

6/ Calculate the number and occurence of mission outcome of the orbits (with value_counts)

7/ Create a set of outcomes where the second stage did not land successfully (bad_outcomes)

8/ Create a classification variable that represents the outcome of each launch using bad_outcomes and Outcome column.

9/ Attribute this variable to new columns 'Class' and determine the success rate from it (with mean)

10/ Export the dataframe to a CSV : df.to_csv("dataset_part_2.csv", index=False)

GITHUB = [labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb · GitHub]

EDA with Data Visualization

Scatter point chart:

- **Categorical scatterplots with catplot** to plot out the **FlightNumber vs. PayloadMassand** overlay the outcome of the launch. *From that we see that as the flight number increases, the first stage is more likely to land successfully. The payload mass is also important; it seems the more massive the payload, the less likely the first stage will return.*
- **Categorcial scatterplots with catplot** to visualize the relationship between **Flight Number and Launch Site**.
- **Categorcial scatterplots with catplot** to visualize the relationship between **Payload and Launch Site**. *From that we observe for the Payload Vs. Launch Site scatter point chart that for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).*
- **Categorcial scatterplots with catplot** to visualize the relationship between **FlightNumber and Orbit type**. *From that we 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.*
- **Categorcial scatterplots with catplot** to visualize the relationship between **Payload and Orbit type**. *From that we see that 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.*

Bar chart :

- **Bar chart** to visualize the relationship between **success rate of each orbit type**. *From that we can see that orbits that have high sucess rate are ES-L1, GEO, HEO and SSO.*

Line chart :

- **Line chart** to visualize the **launch success yearly trend**. *From that we can observe that the sucess rate since 2013 kept increasing till 2020.*

GITHUB = [jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb · GitHub]()

EDA with SQL

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 SpaceX for a rocket launch. This dataset includes a record for each payload carried during a SpaceX mission into outer space.

Here all SQL queries that I performed:

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- 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.
- 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.

GITHUB = [jupyter-labs-eda-sql-coursera_sqllite.ipynb · GitHub]

Build an Interactive Map with Folium

The Launch success rate may depend on the location and proximities of a launch site. Finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations.

In order to find some geographical patterns about launch sites and perfom launch sites Locations analysis, I used this follows map object:

- **Folium.Circle() :** to add a highlighted circle area with a text label on a specific coordinate.
  - *Used to mark each launch site from a dataframe with a circle on a folium map*
- **MarkerCluster() :** to group and cluster map markers for better visualization of markers.
  - *Used to mark and group each launch site or success/failed launches for each site on the map*
- **Folium.Marker() :** to mark a specific point on the map, which can by customized by objects such as icon and popup information.
  - *Used to mark each launch site or success/failed launches for each site on the map*
- **Folium.Popup() :** to creates a popup label that can be associated with markers or other map elements.
  - *Used to display names of launch sites on markers*
- **Folium.Icon() :** to defines the icon used for markers.
  - *Use to display color marker for success/failed launches for each site on the map*
- **MousePosition() :** to get the coordinate (Lat, Long) for a mouse over on the map.
  - *Used to be able to see the coordinate of the mouse position on the map to get coastlines,city or highway coordinate around launch sites*
- **Folium.PolyLine() :** to draw a line on the map between two coordinates.
  - *Used to draw line betweean each launch site and the nearest coastlines or the nearest City for exemple*

GITHUB = lab_jupyter_launch_site_location.jupyterlite.ipynb · GitHub

Build a Dashboard with Plotly Dash

Build a Plotly Dash application for users to perform interactive visual analytics on SpaceX launch data in real-time. From that We will be able to the determine for sites the largest successful launches and the highest launch success rate or for payload range the highest launch success rate.

Here what plots/graphs and interactions I have added to a dashboard:

- **Drop-down Input Component :** *dcc.Dropdown() from library dash_core_components* : to display interactive drop-down list to select items.
    - Used to let us select different launch sites, and interact with fisplayed graphs.
- **Pie Chart :** *px.pie() from library plotly.express* : to display interactive Pie Chart wich is divided into sectors to illustrate numerical proportion
    - Used to render a pie chart visualizing launch success counts for selected launch site in drop-down list
    - Used to render and return a pie chart graph to show the total success launches if 'ALL sites' items is selected in drop-down list
- **Range slider :** *dcc.RangeSlider() from library dash_core_components* : to display interactive range slider, to select areas on the rail or by dragging handles.
    - Used to be able to easily select different payload mass range, and interacte with the scatter plot graph.
- **Scatter plot :** *px.scatter() from library plotly.express* : to display interactive Scatter plot graph, which each point are markers
    - Used to plot a scatter plot with the x axis to be the payload and the y axis to be the launch outcome.
    - We can visually observe how payload, linked to payload range slider, may be correlated with mission outcomes for selected site(s), linked to drop-down lauch sites list.

- *dcc.Graph() from library dash_core_components : it's the components to display previous pie chart and scatter plot.*

GITHUB = [spacex_dash_app.py · GitHub]

Predictive Analysis (Classification)

If we can determine if the first stage will land, we can determine the cost of a launch. A machine learning pipeline is used to predict if the first stage will land given previous generated datas. To process, follows classification models can be trained : SVM, Decision Tree, Logistic Regression and KNN.

1/ Create a NumPy array from the column Class in data, by applying the method to_numpy() then assign it to the variable Y

2/ Standardize the data in X then reassign it to the variable X

3/ Use the function train_test_split to split the data X and Y into training and test data

4/ Create GridSearchCV object on each model and fit the object to find the best parameters on each model

5/ Calculate accuraries on the test data using the method score on each model's object

6/ Create predictions on tests data using method predict on each model's object

7/ Plot confusion matrix on each model's object with Y test data and previous calculated predictions data

8/ Figure out the best model with highest accuracy

9/ Figure out the best model regarding confusion matrix, interpretability and computational complexity

10/ Confirm the optimal model after comparison

* All models : logistic regression / support vector machine (SVM) / decision tree classifier / k nearest neighbors (KNN)

GITHUB = [SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb · GitHub]

Results

EXPLORATY DATA ANALYSIS RESULTS

Data Visualisation Results :

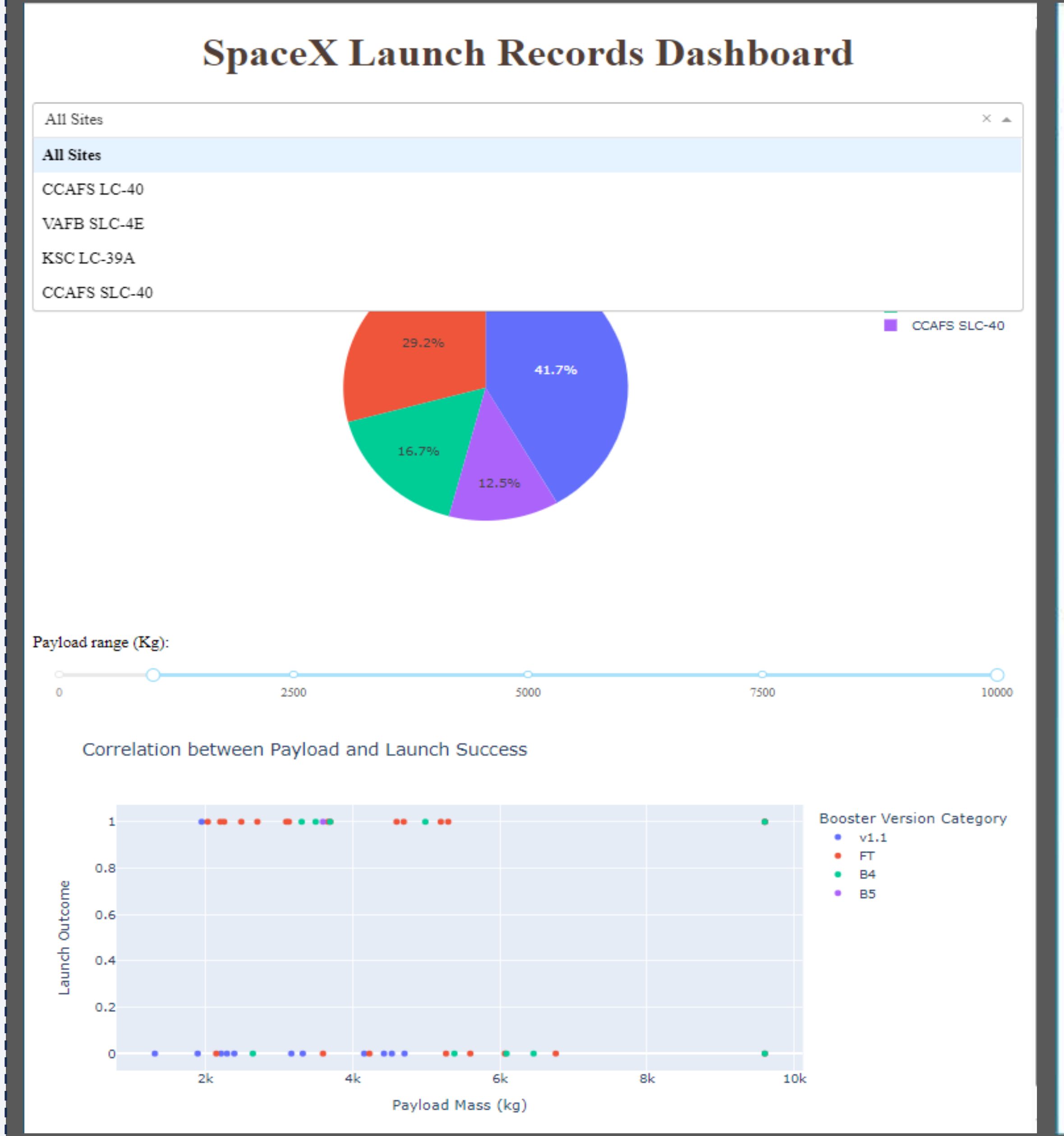
- Our analysis revealed that the choice of launch site significantly influences mission success, with KSC LC-39A and VAFB SLC 4E demonstrating higher success rates.
- We observed a clear learning curve in the LEO orbit, where success rates improved with an increasing number of flights.
- Missions targeting ES-L1, GEO, HEO, and SSO orbits exhibited consistently high success rates, indicating the desirability of these orbits for future missions.
- Heavy payloads show higher success rates for Polar, LEO, and ISS orbits.
- VAFB-SLC launch site has not been used for heavy payloads (greater than 10,000). This could indicate site-specific limitations or a deliberate choice based on payload characteristics.
- The data indicates an increasing success rate from 2013 to 2020, suggesting improvements in mission planning, technology, or operational procedures over the years.

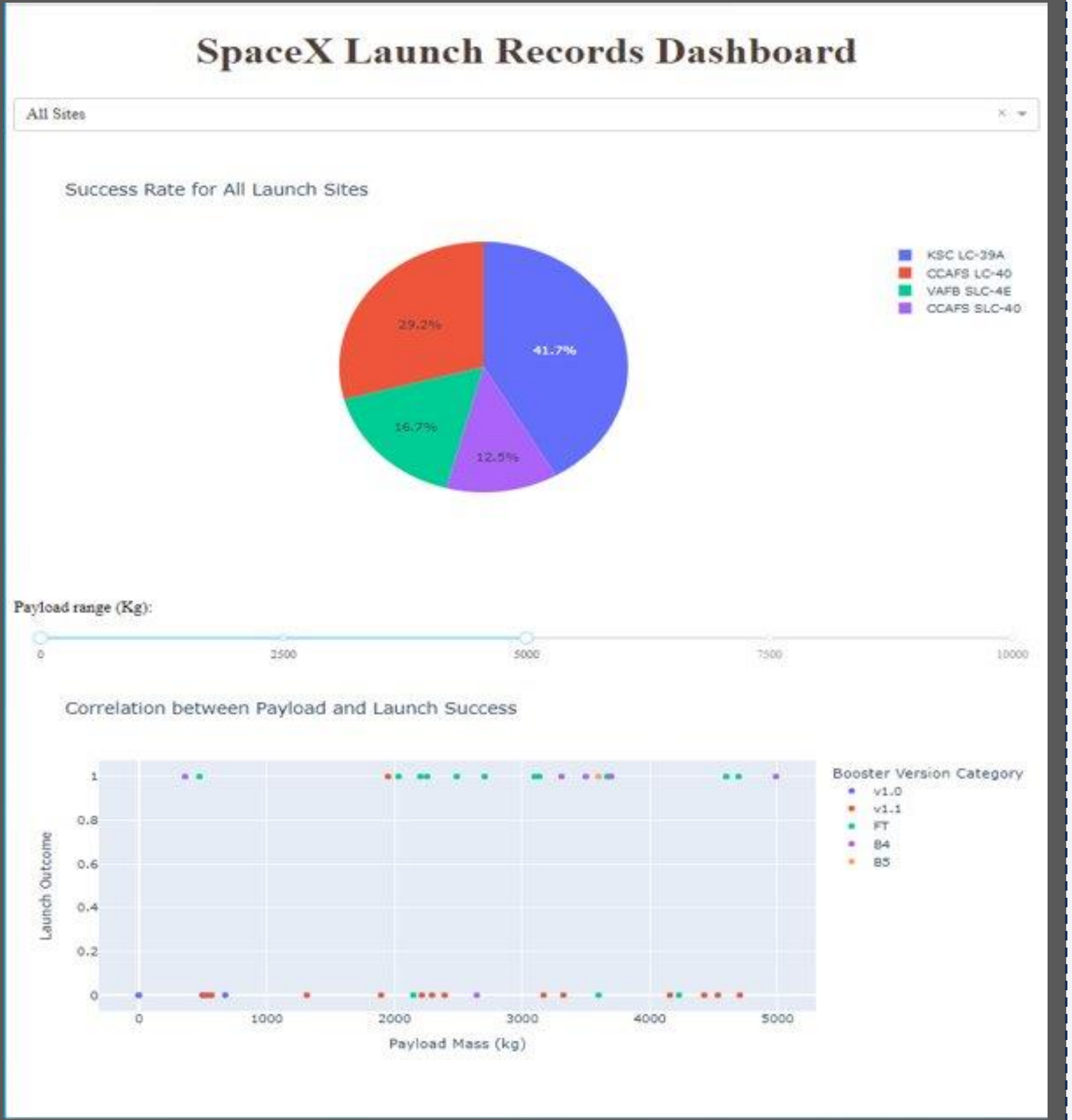
SQL queries Results :

- We figure out a lot of landing outcome and mission outcome regarding booster version, launch site, date of launch, customer, payload and orbit.

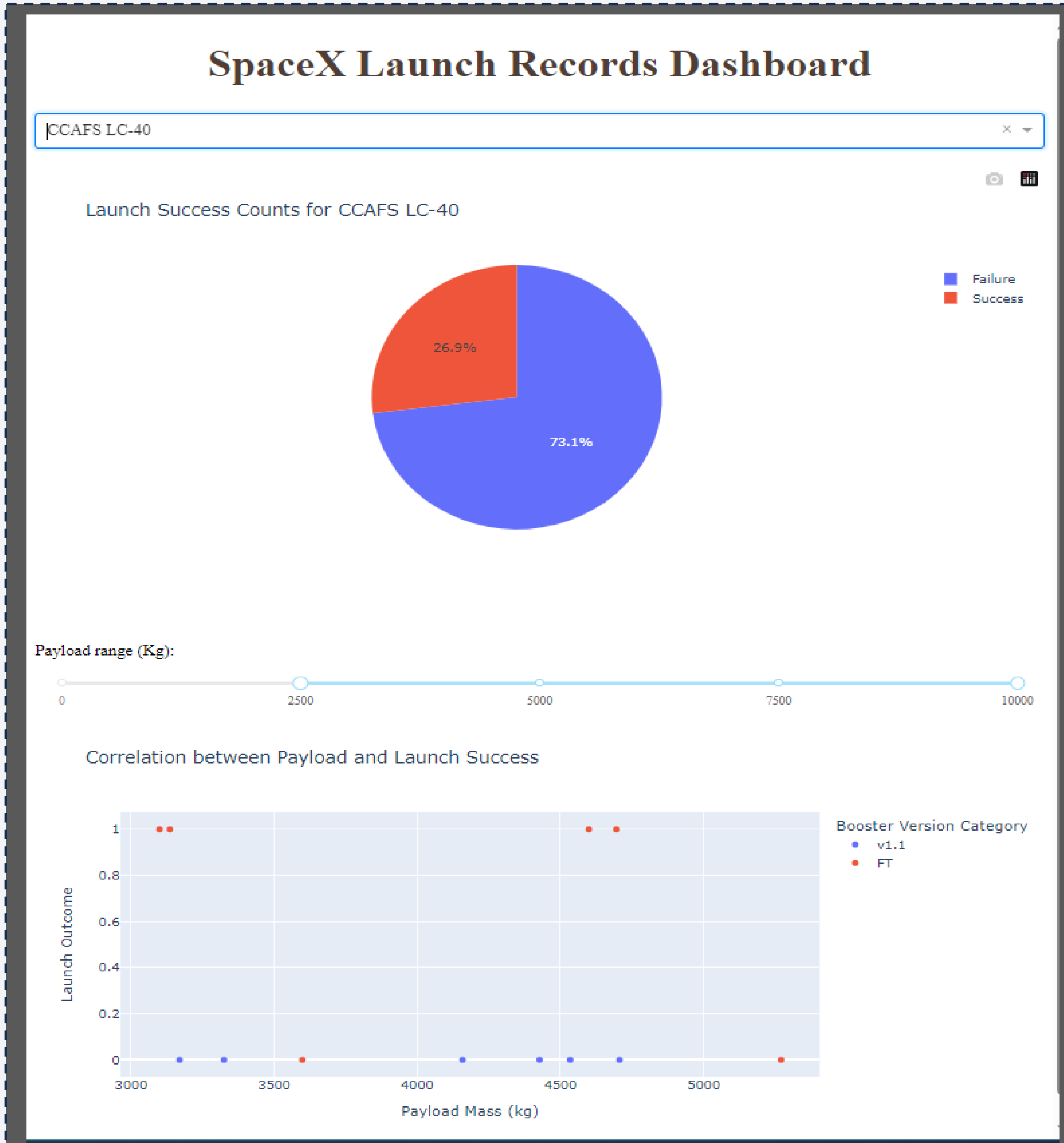
Conclusion : The analysis of launch site success rates, orbit choices, and payload characteristics highlights the significance of data-driven decision-making for optimizing space missions and achieving higher success rates over time.

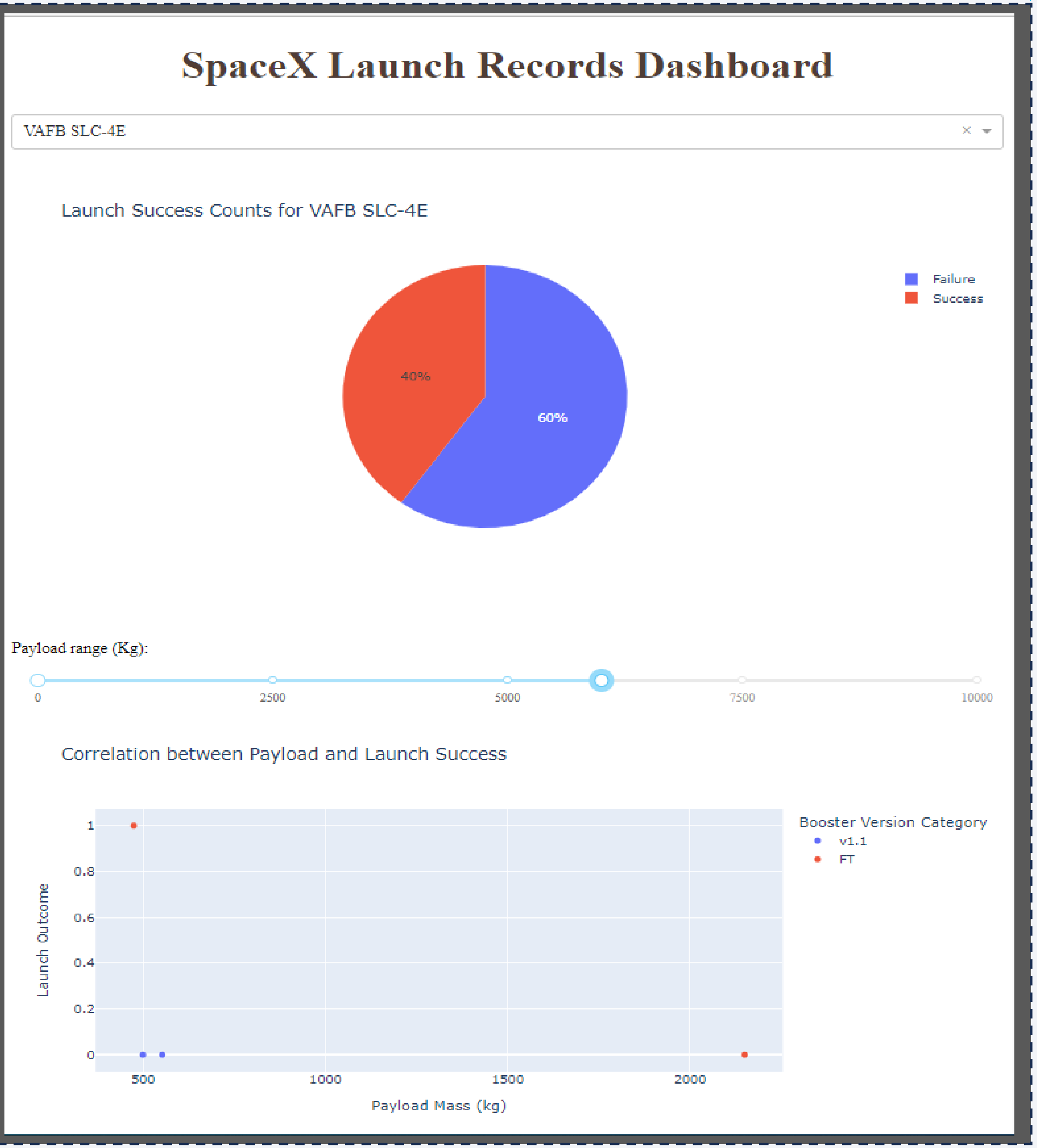
Results - Interactive analytics demo



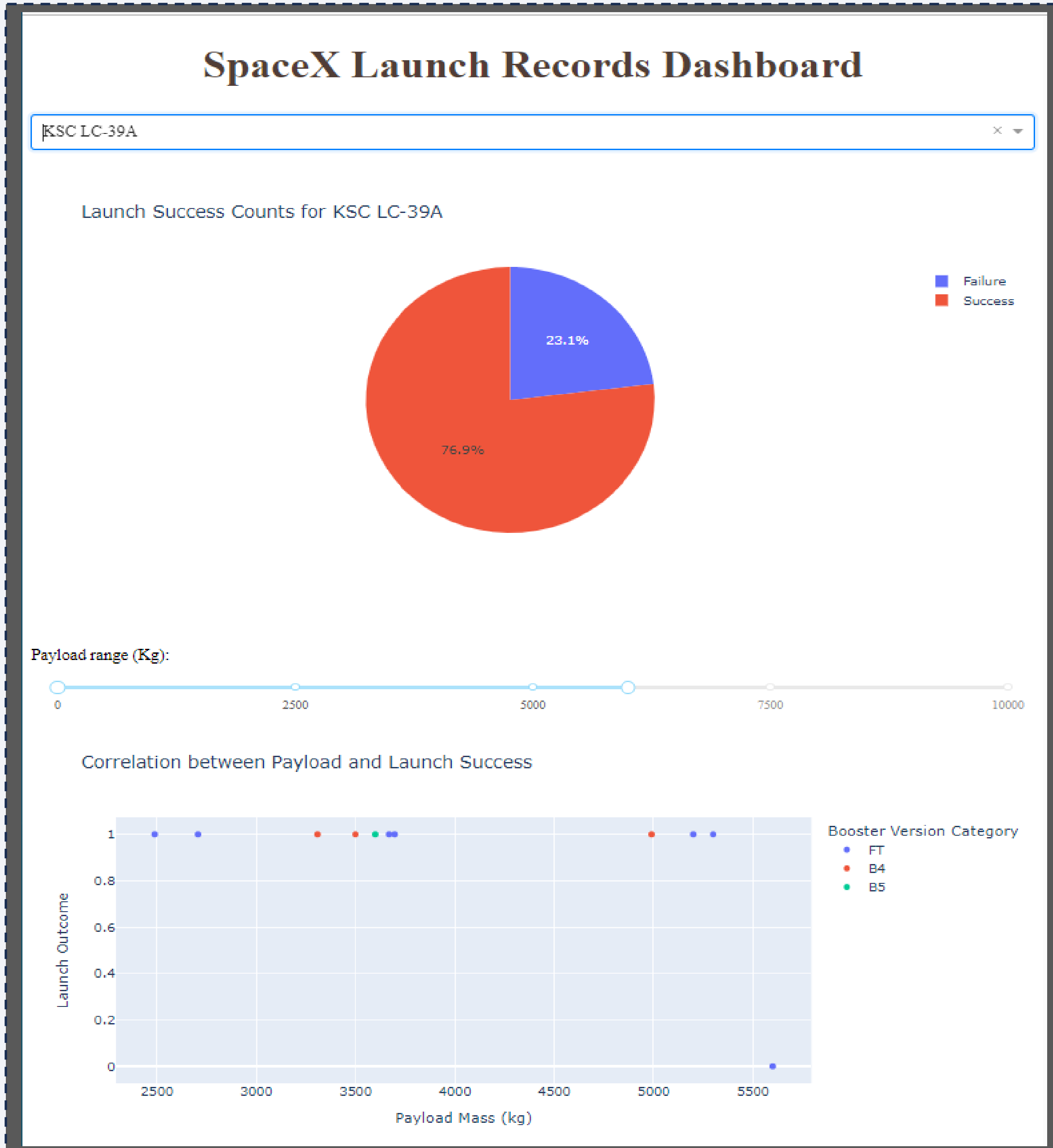


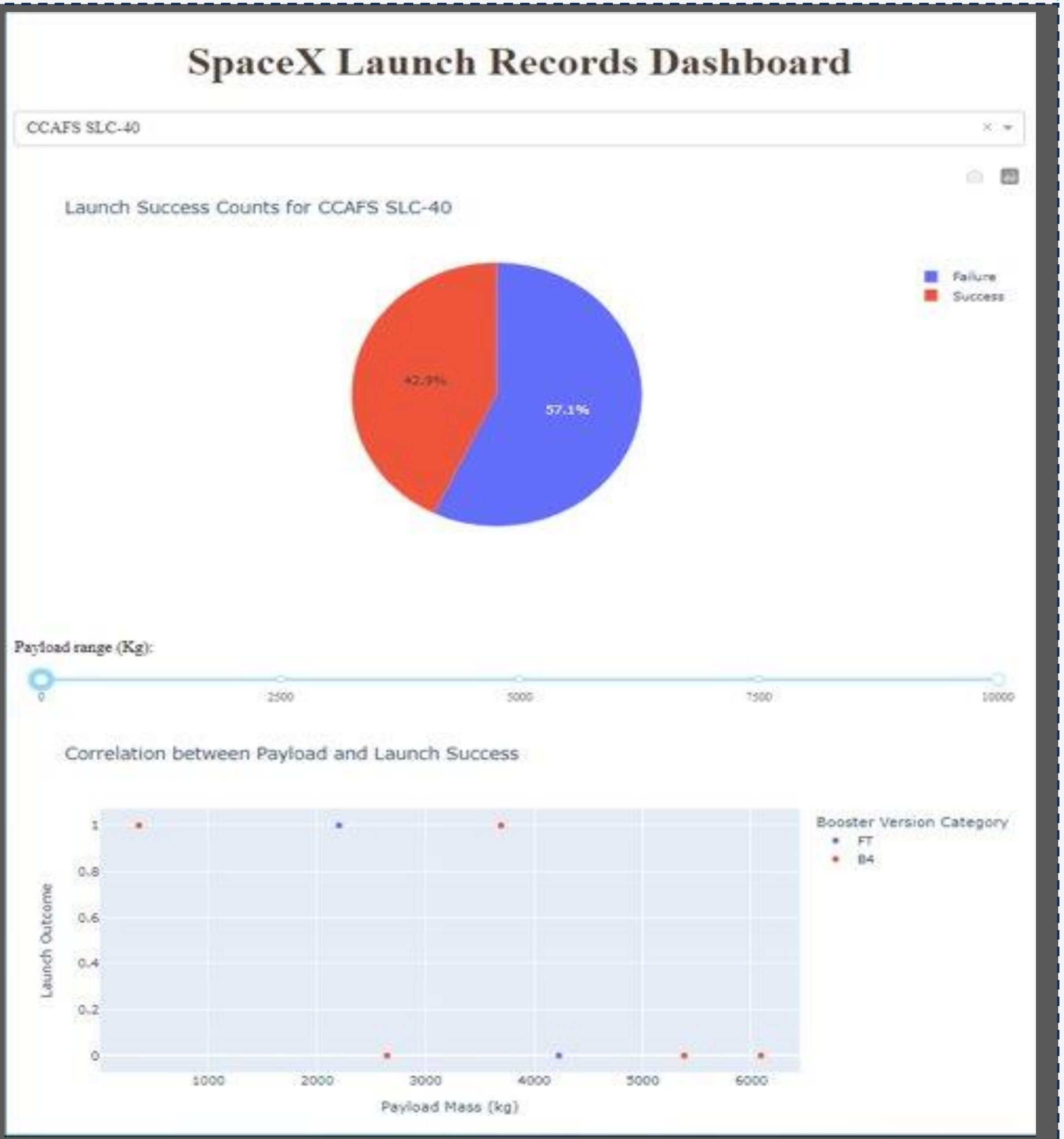
Results - Interactive analytics demo





Results - Interactive analytics demo





Results

PREDICTIVE ANALYSIS RESULTS

The SVM, KNN and Logistic Regression models are the best in terms of prediction accuracy with around 0.83 for this dataset. For the Decision Tree model i found around 0.72 of accuracy.

Regarding confusion matrix, all of three have the same table. So the calculation of all metrics are the same.

The result with another tool sklearn.metrics i tried show me that all four SVM, KNN , Decision Tree and Logistic Regression models have around 0.83 accuracy too.

We notice that accuracy can change randomly after severals retry tests, specially for the Decision Tree.

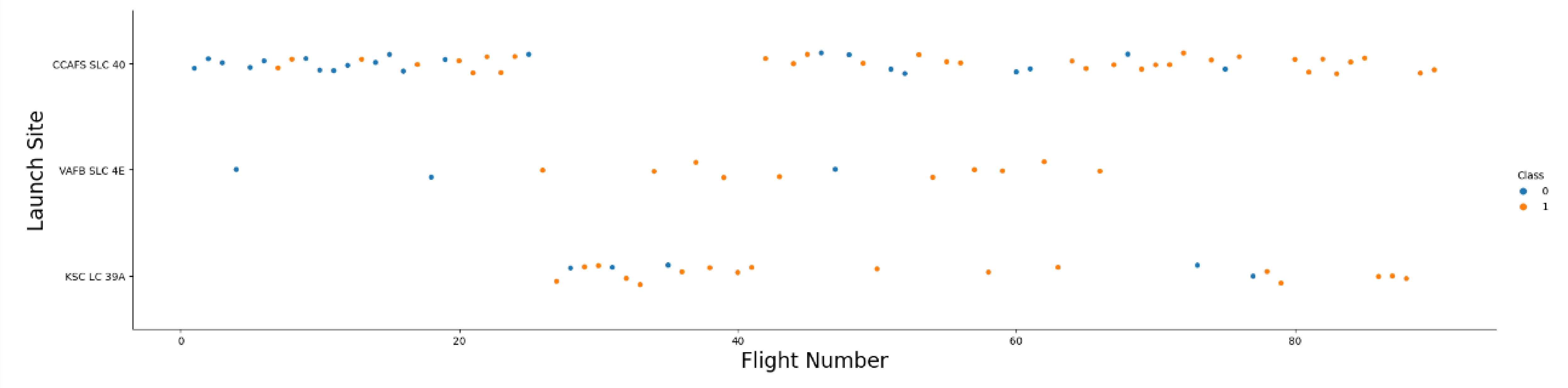
To choose the best method among several models with similar accuracy and confusion matrix, we should consider other criteria such as interpretability, computational complexity, or specific requirements of our problem.

Conclusion : We should consider other factors like interpretability, computational resources, and whether false positives/negatives have a significant impact on modle performance. However SVM, KNN and Logistic Regression models are the one performs best regarding our dataset.

Section 2

Insights drawn from EDA

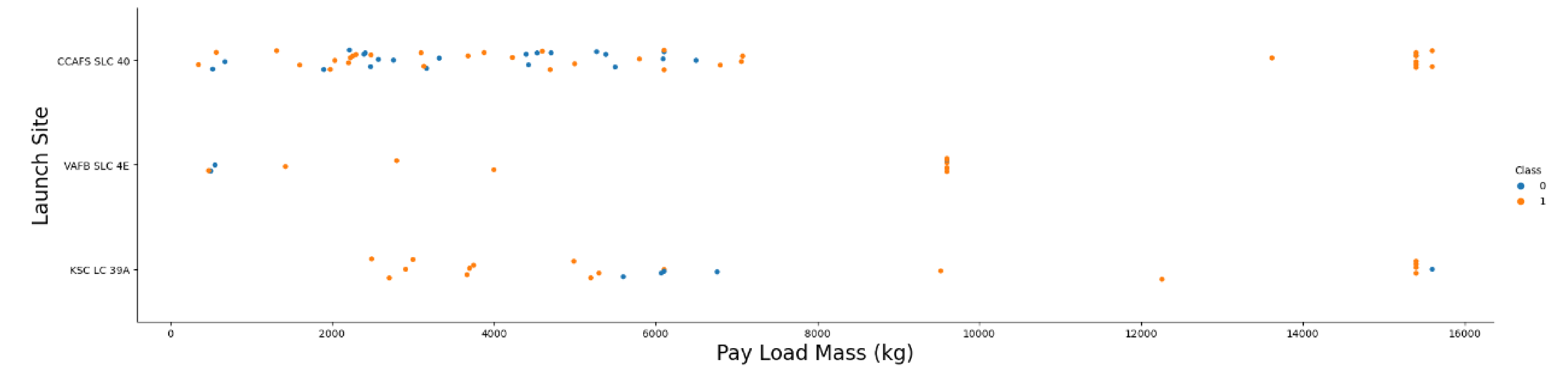
Flight Number vs. Launch Site



Explanation :

Launches from the site of CCAFS SLC 40 are significantly higher than launches from other sites. We can add that the more flight there are, the more success rate increase.

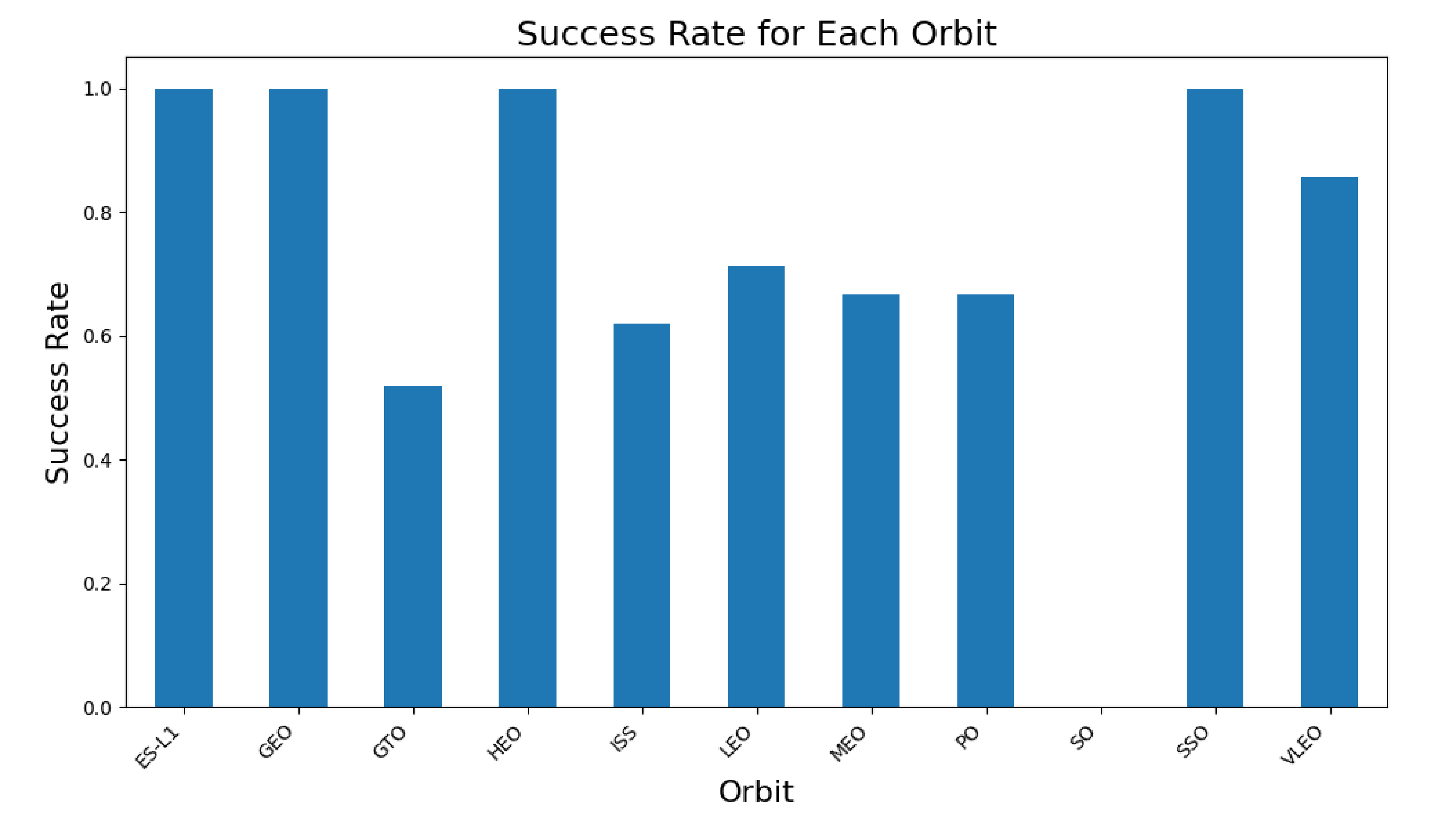
Payload vs. Launch Site



Explanation :

For the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000). We see that the majority of Pay Load with lower Mass have been lauched from CCAFS SLC 40 launch site, and we can add that from this launch site rockets launched for heavypayload are all successful.

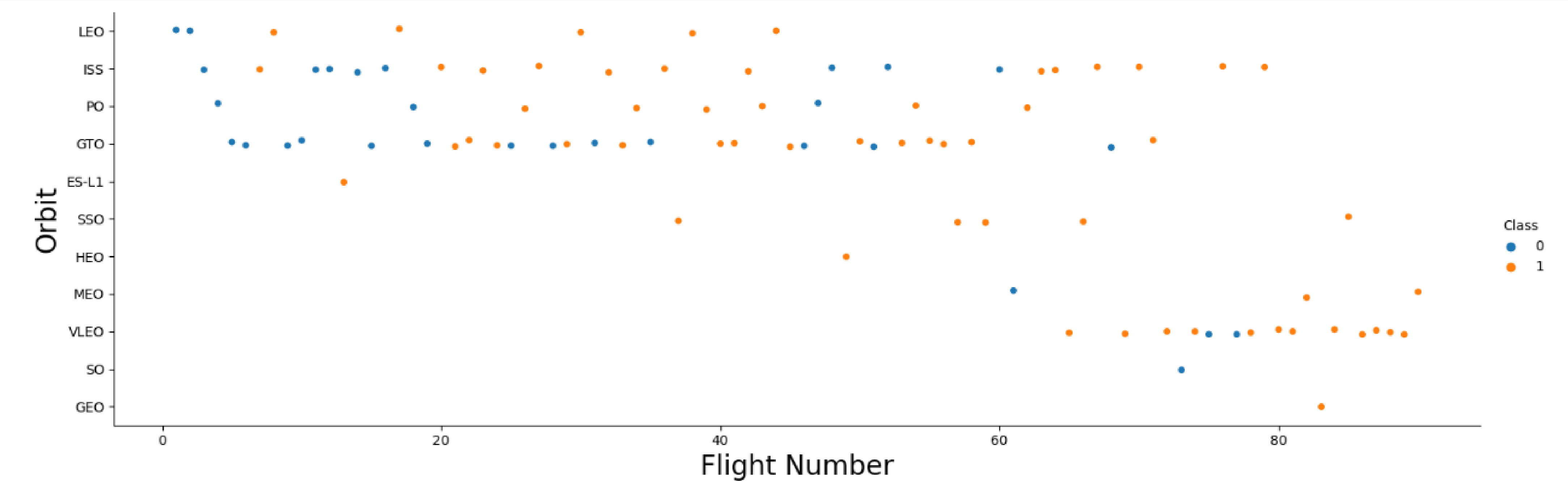
Success Rate vs. Orbit Type



Explanation :

The orbit types of ES-L1, GEO, HEO and SSO are among the highest success rate with 100%. The orbit type SO is the only one with no success at all.

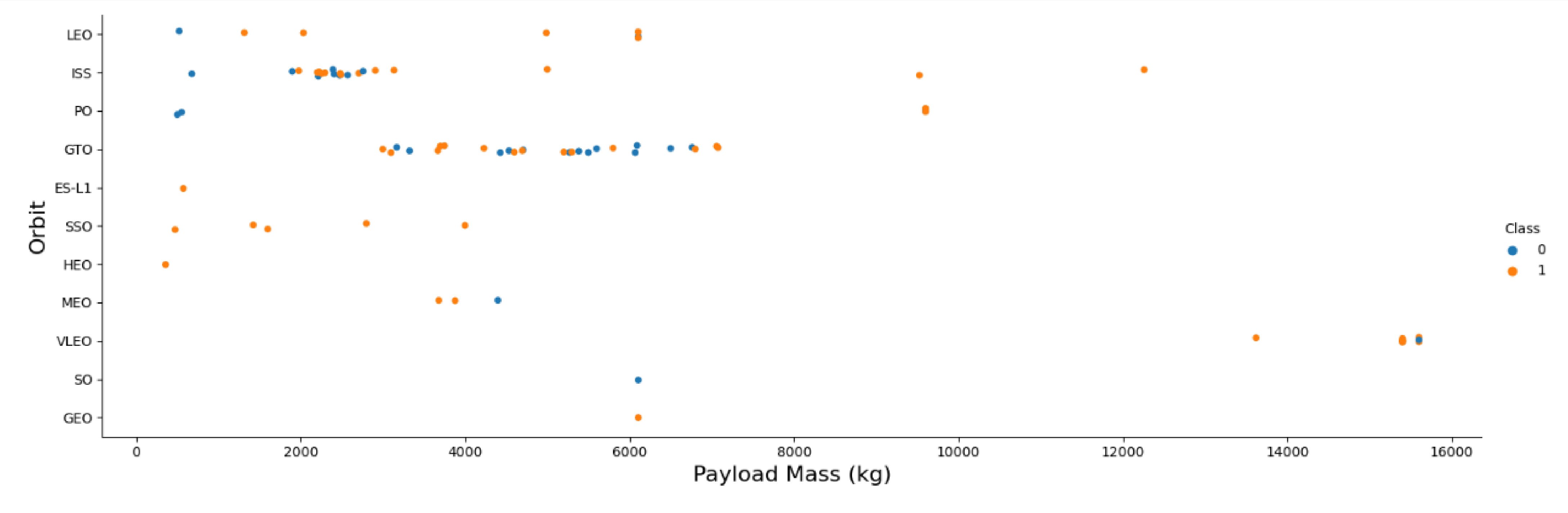
Flight Number vs. Orbit Type



Explanation :

In the Low Earth Orbit (LEO), success appears to be positively correlated with the number of flights, while in the Geostationary Transfer Orbit (GTO), there appears to be no discernible relationship between the number of flights and success.

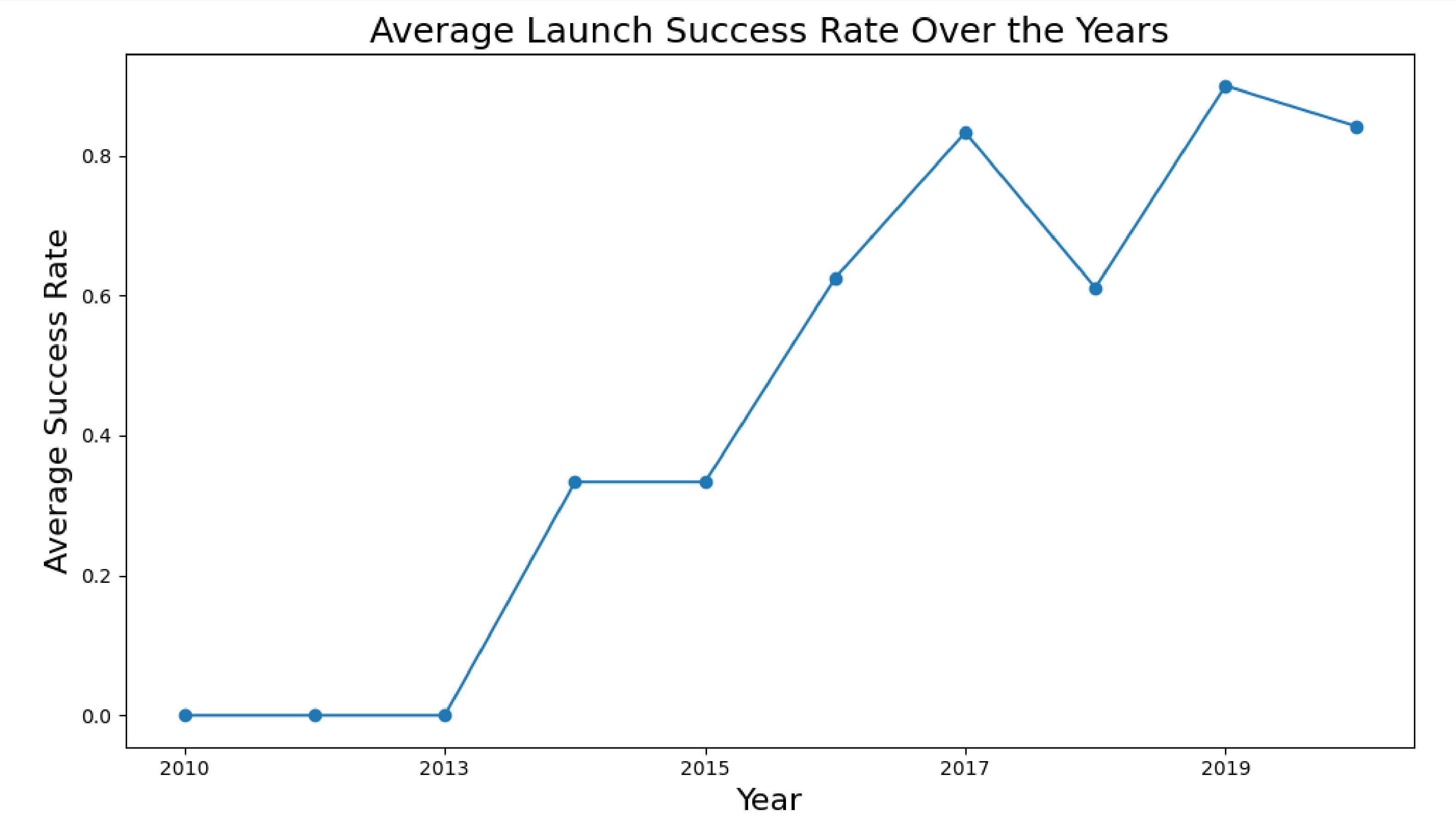
Payload vs. Orbit Type



Explanation :

For heavy payloads, the likelihood of successful or positive landings is higher in Polar, LEO, and ISS orbits. However, in the GTO orbit, it's challenging to differentiate, as both positive and negative landing outcomes (unsuccessful missions) are prevalent.

Launch Success Yearly Trend



Explanation :

The sucess rate since 2013 kept increasing till 2020.
May suggest improvements in mission planning, technology, or operational procedures over the years.

All Launch Site Names

```
%%sql
SELECT DISTINCT launch_site FROM SPACEXTABLE
```

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

Explanation :

There is only this 4 launch site in SpaceX dataset.

Launch Site Names Begin with 'CCA'

```
%%sql
SELECT * FROM SPACEXTABLE WHERE launch_site LIKE 'CCA%' LIMIT 5
```

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

Explanation :

Here an overview of all results data in dataset for Launch Site beginning with 'CCA'.

Average Payload Mass by F9 v1.1

```
%%sql

SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version = 'F9 v1.1'
```

AVG(PAYLOAD_MASS__KG_)
2928.4

Explanation :

The average payload mass carried by booster version F9 v1.1 is 2928.4 KG.

Total Payload Mass

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer = 'NASA (CRS)'
```

SUM(PAYLOAD_MASS__KG_)
45596

Explanation :

The total payload carried by boosters from NASA is 45596 KG.

First Successful Ground Landing Date

```
%%sql

SELECT MIN(DATE) FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)'
```

MIN(DATE)
2015-12-22

Explanation :

The data of the first successful landing outcome on ground pad is 2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql

SELECT Booster_Version FROM SPACEXTABLE WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

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

Explanation :

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

Total Number of Successful and Failure Mission Outcomes

```
%%sql

SELECT Mission_Outcome, COUNT(*) FROM SPACEXTABLE GROUP BY Mission_Outcome
```

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

Explanation :

Here the total number of successful and failure mission outcomes, we can see that outcomes are great majority success with just one Failure in flight.

Boosters Carried Maximum Payload

```
%%sql
SELECT DISTINCT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ IN (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE )
```

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

Explanation :

Here a list of booster names which have carried the maximum payload mass, all of them are the category of booster version beginning with F9 B5 B10....

2015 Launch Records

```
%%sql

SELECT substr(DATE,1,4) as year, substr(Date, 6, 2) as month, Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTABLE WHERE substr(DATE,1,4) = '2015' AND Landing_Outcome = 'Failure (drone ship)'
```

year	month	Landing_Outcome	Booster_Version	Launch_Site
2015	10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015	04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Explanation :

The failed landing outcomes in drone ship in year 2015 are 2 in and number, for the same launch site CCAFS LC-40 and the booster version F9 v1.1 B1012 and B1015.

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

```
%%sql

SELECT Landing_Outcome, COUNT() as total_count FROM SPACEXTABLE
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY total_count DESC
```

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

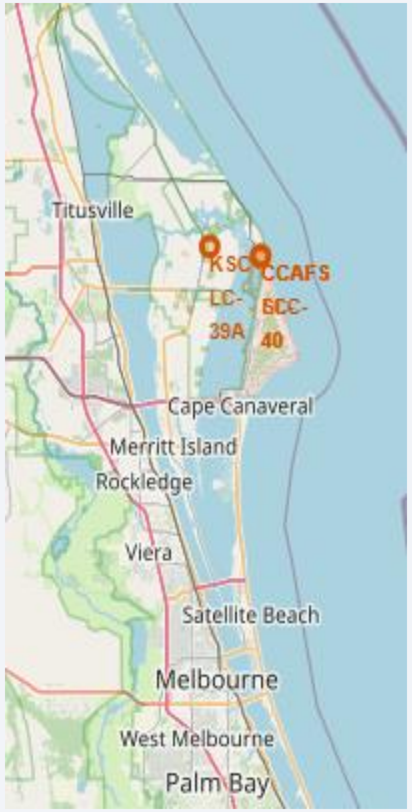
Explanation :

Between this two dates, we count 10 not attempt, 10 success, 6 failures, and 6 others landing outcome types. We see that Succes are greater than Failures. We can that success and failure for drone ship are equals too.

Section 3

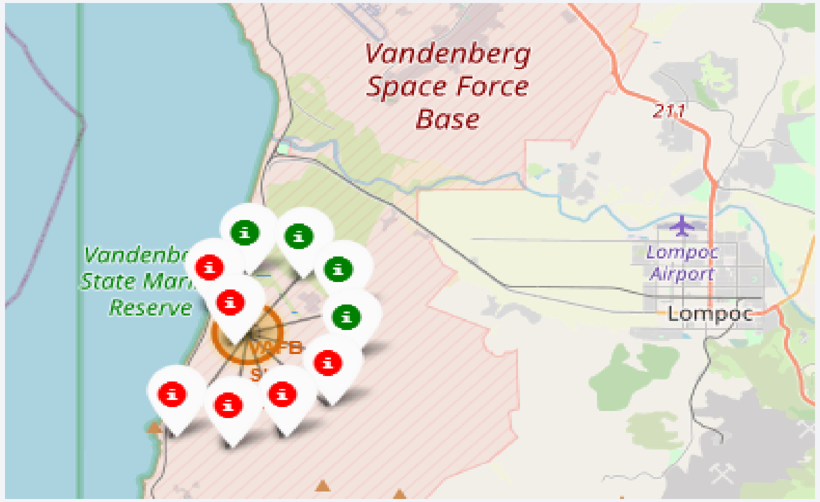
Launch Sites Proximities Analysis

All launch sites' location on the map

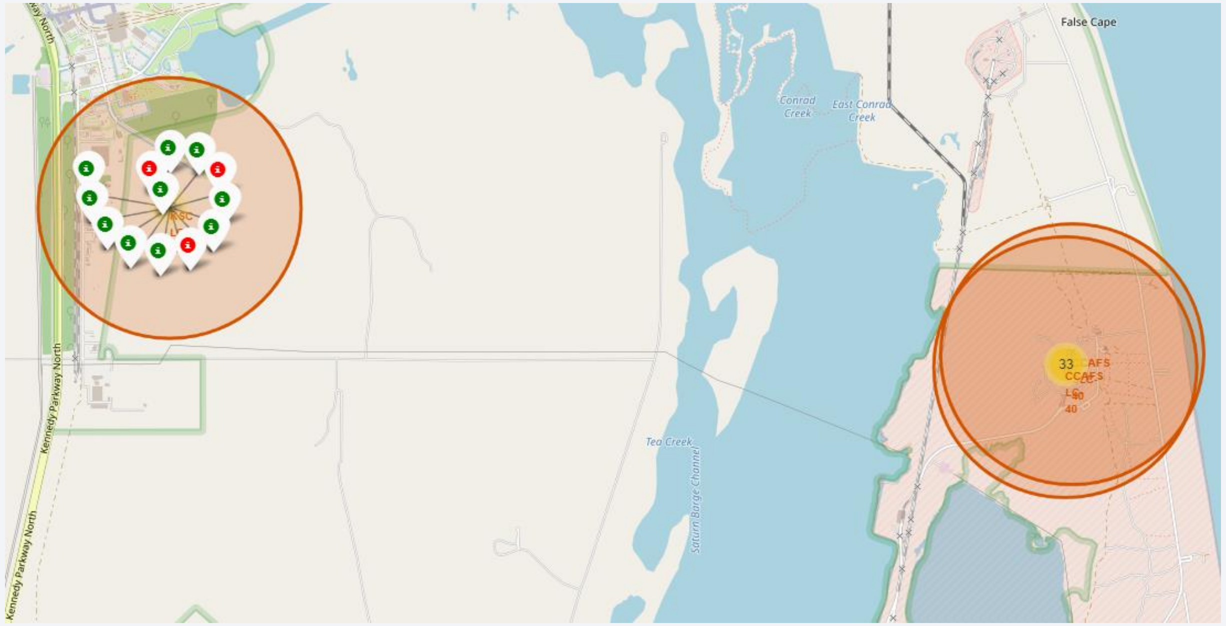


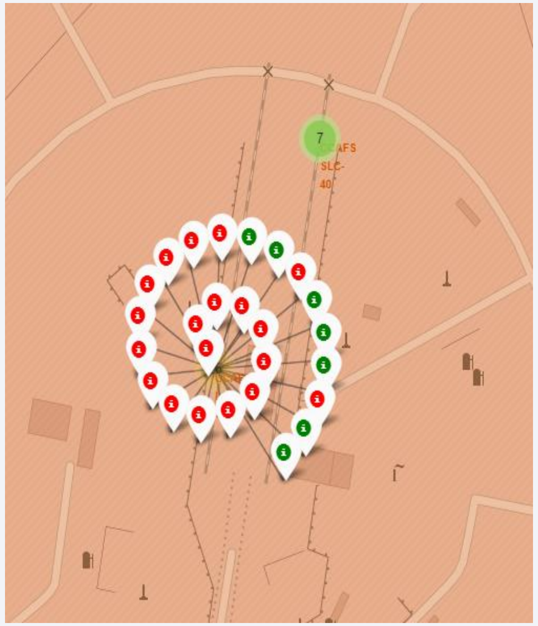
Here the four launch site, one in west coast near Los Angeles and others in east coast near Miami. We notice all of them are very close proximity to the coast. Not all launch sites in proximity to the Equator line.

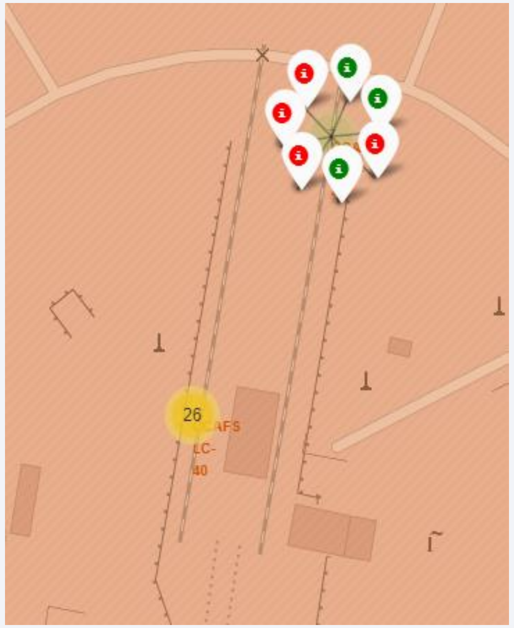
Success/Failures outcomes of launches



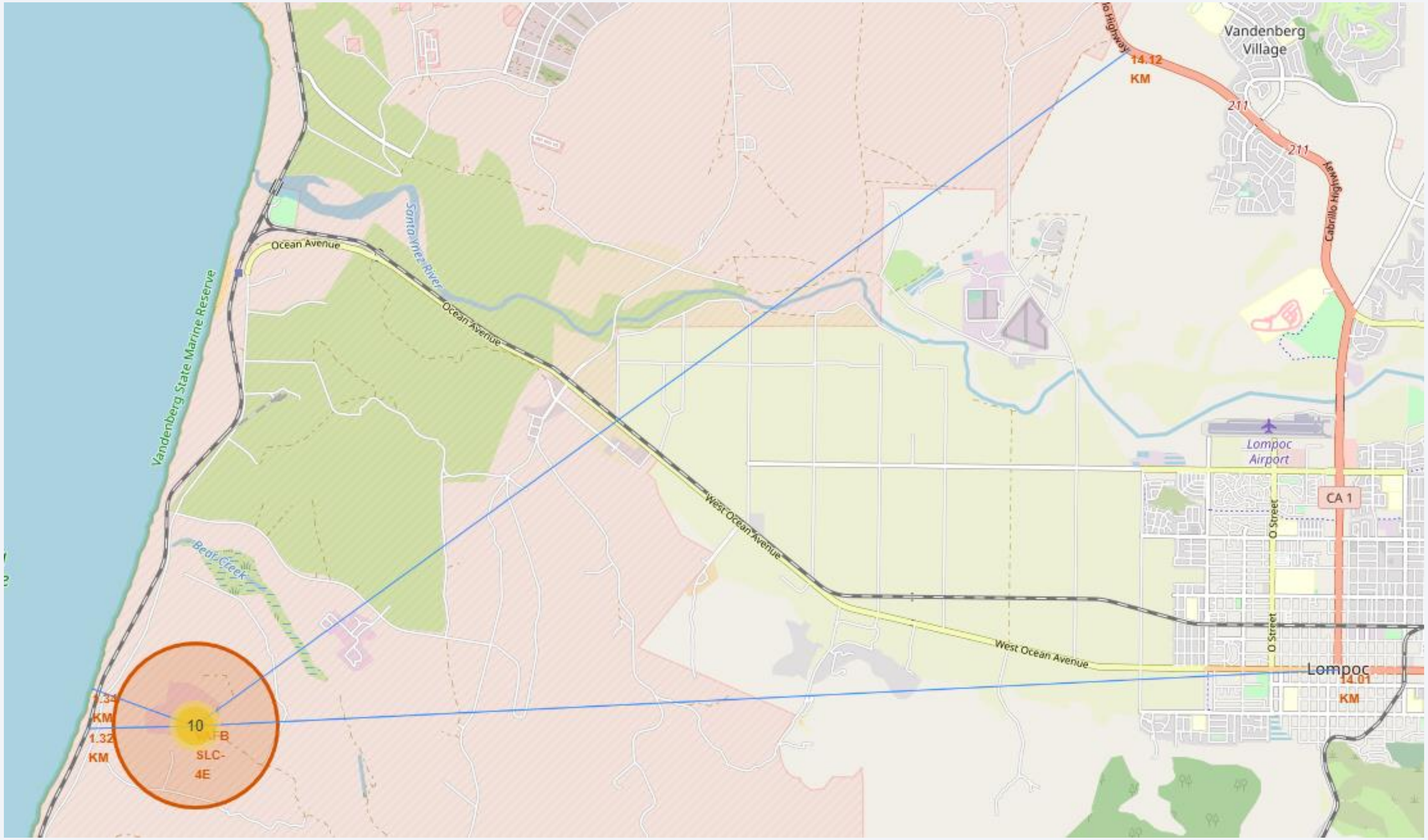
From all this color-labeled markers, we can identify which launch sites have relatively high success rates. It's by large the laucnh site **KSC LC 39-A** who contains the most in proportion of green markers.







Launch site distances to its proximities

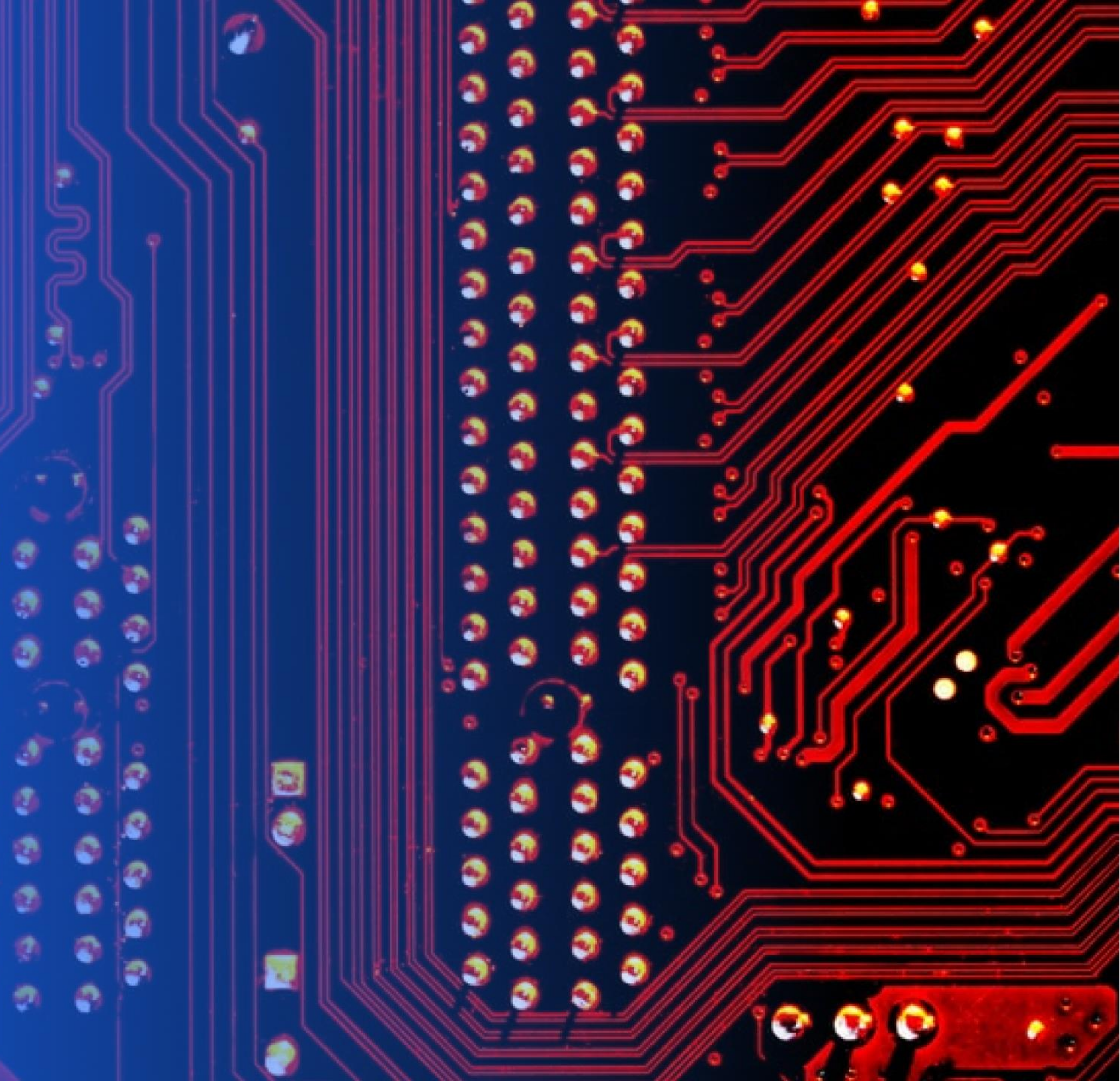


The selected launch site for proximity checking is **VAFB SLC-4E** on the West Coast. We calculated the distance to the nearest **coastline**, which is approximately 1.34 kilometers, the nearest **railway**, which is about 1.32 kilometers away, the nearest **city**, Lompoc, which is approximately 14 kilometers away, and the nearest **highway**, which is about 14.12 kilometers away.

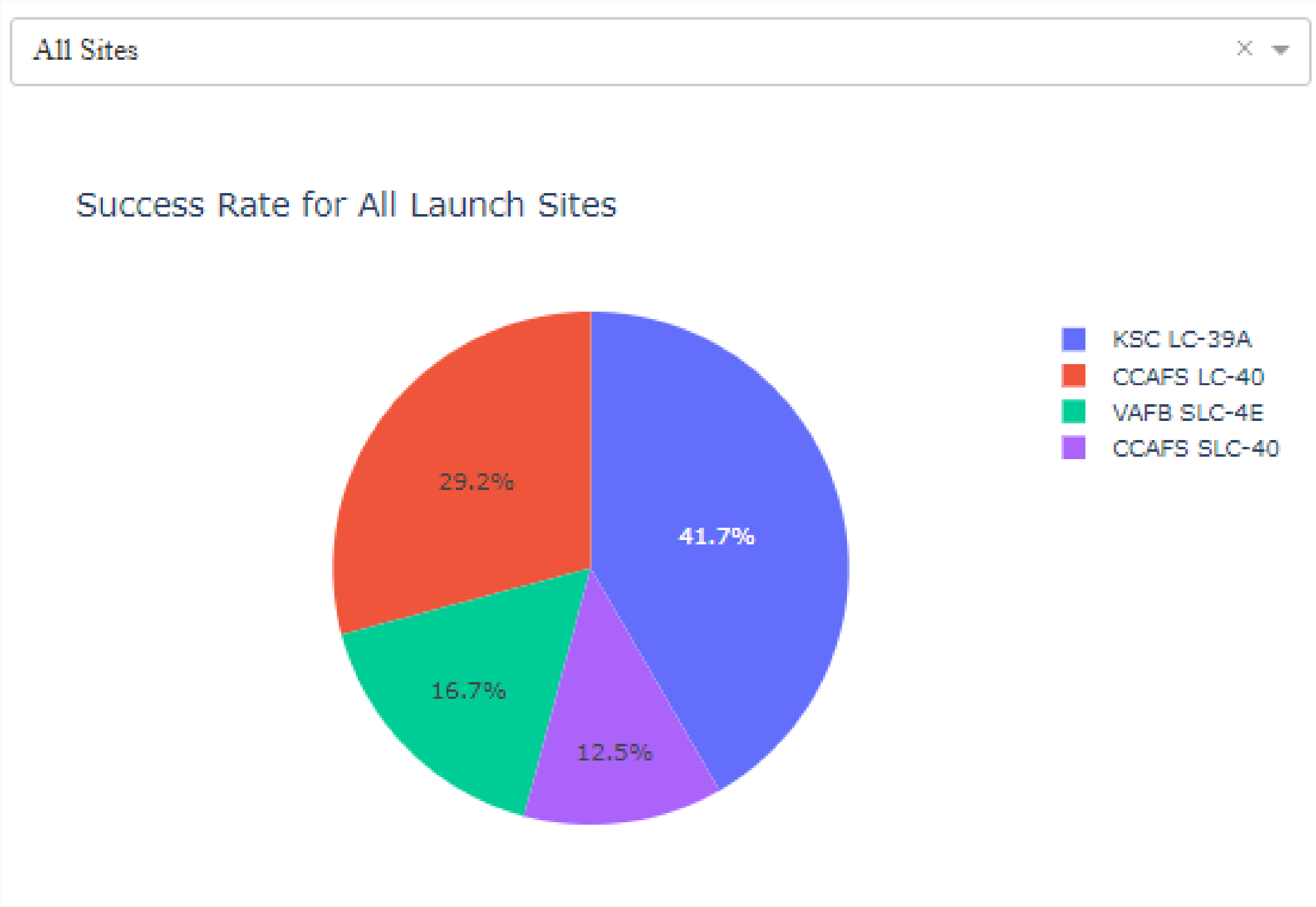
This indicates that the launch site is relatively close to coastlines and railways but is not in close proximity to highways. It also maintains a certain distance away from cities. We see also a distance maintains from cities for others launch sites.

Section 4

Build a Dashboard with Plotly Dash

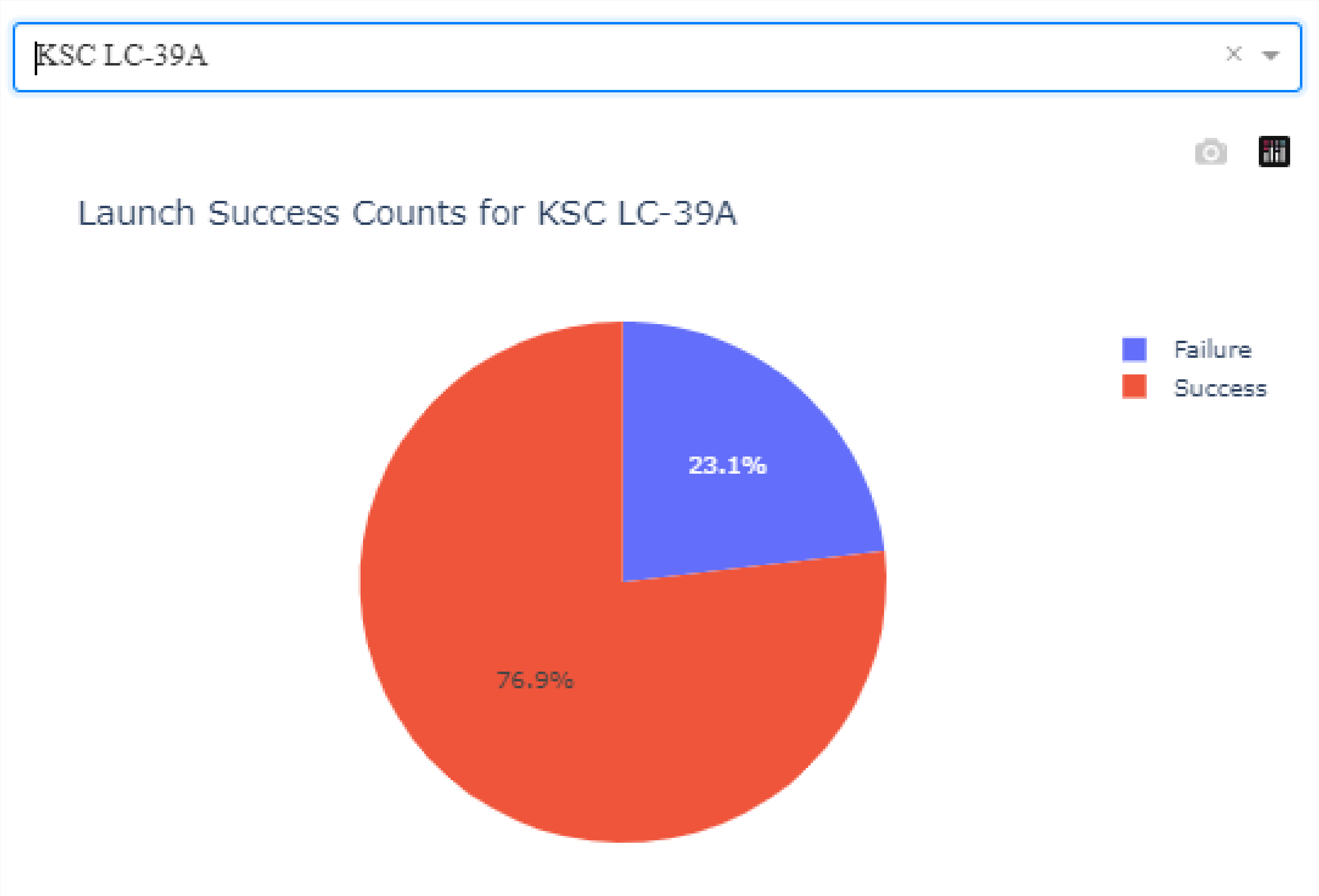


Success rate for all Launch Sites



This is an interactive Pie Chart that show for all lauch sites the success rate of launches. With this Pie Chart we see easily wich launch site have the more or least success rate and compare them.
Here this is **KSC LC-39A** wich have the more success rate, it represent **41.7%** in all launch sites success rates.
And this is **CCAFS SLC-40** launch site wich have the least success rate with **12.5%**.

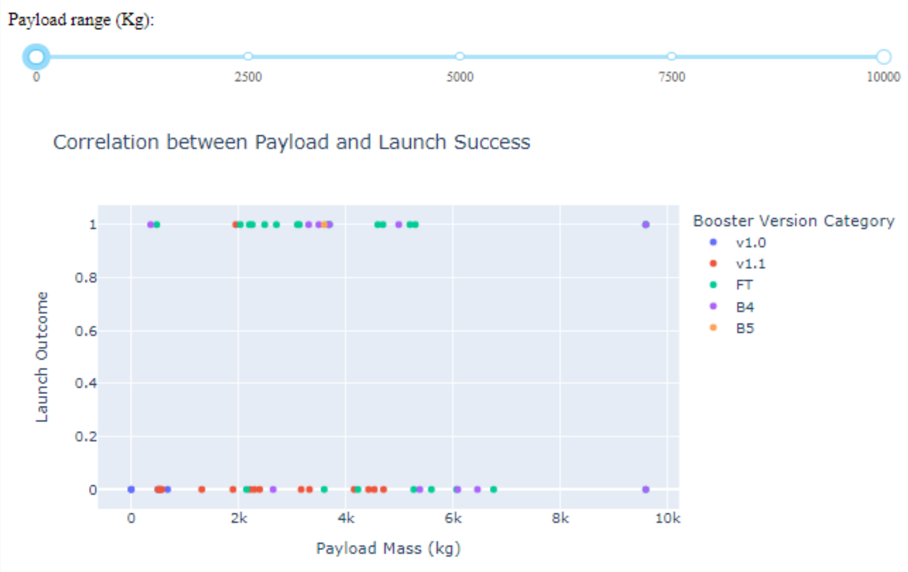
Launch site with highest launch success ratio

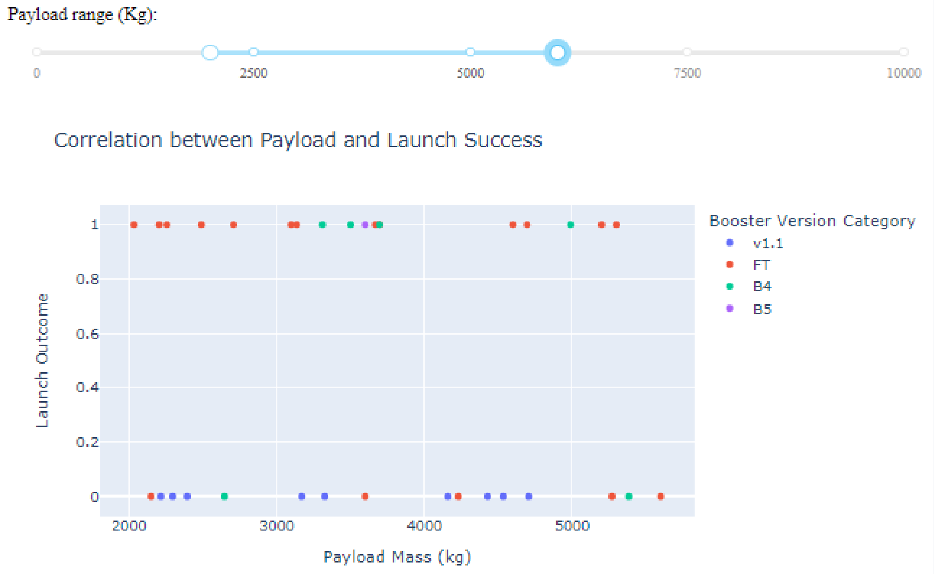


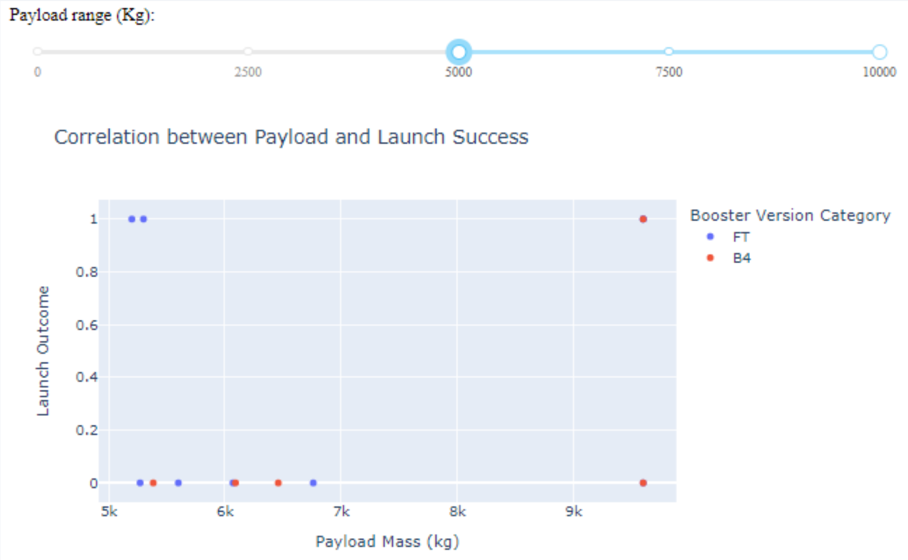
This is an interactive Pie Chart that show Success count rate and Failure counts rate of the launch site, **KSC LC-39A**, wich have the highest launch success ratio.

The success rate of this launch site is **76.9%**.

Payload vs. Launch Outcome scatter plot for all sites





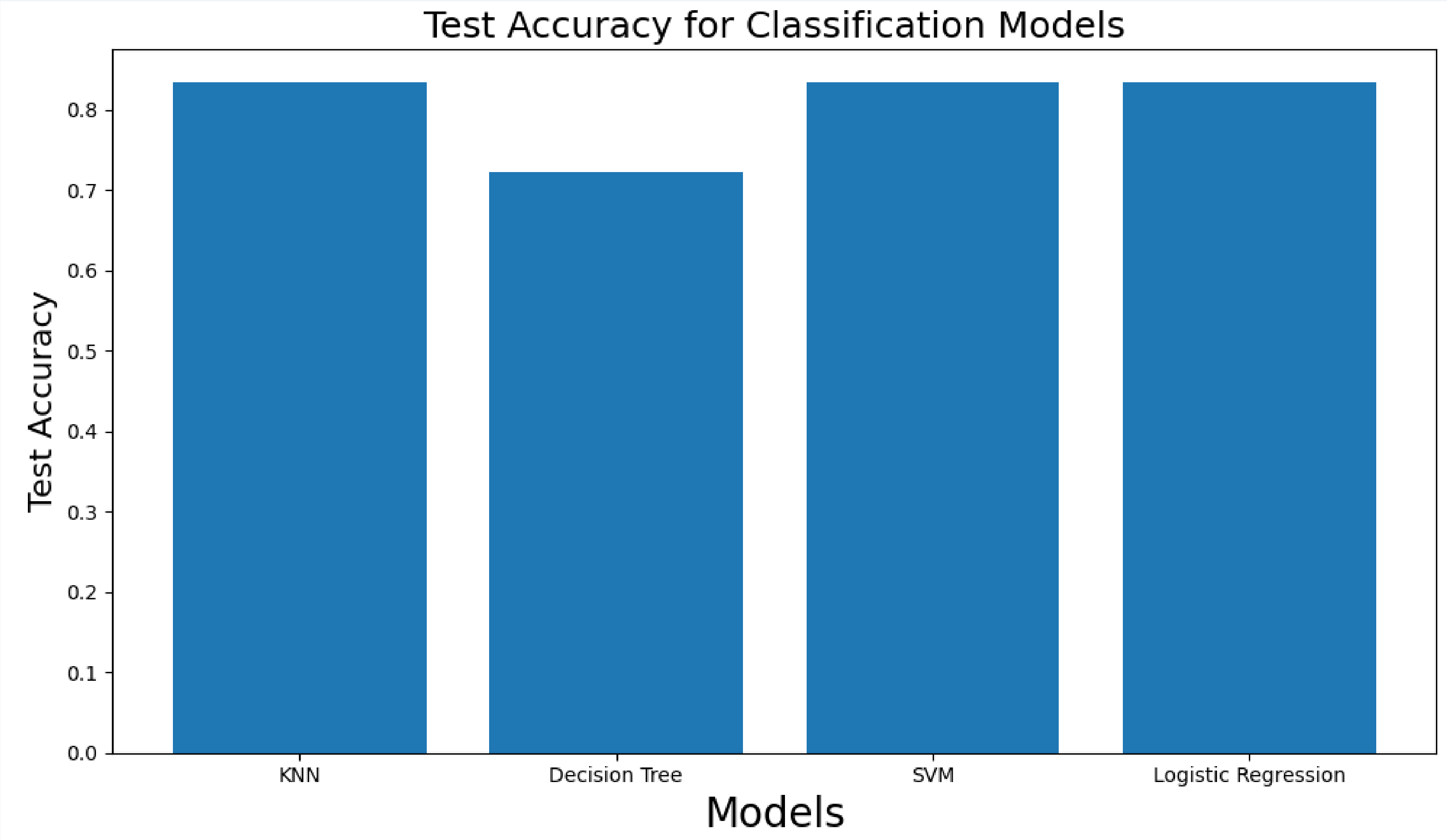


Here samples of scatter plot for all sites with different payload range, they show that the booster version which have the largest success rate is **FT** category and the payload range with largest succes rate is from **2000 Kg to 6000 Kg,** before and after this payload range we count more failure rate.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

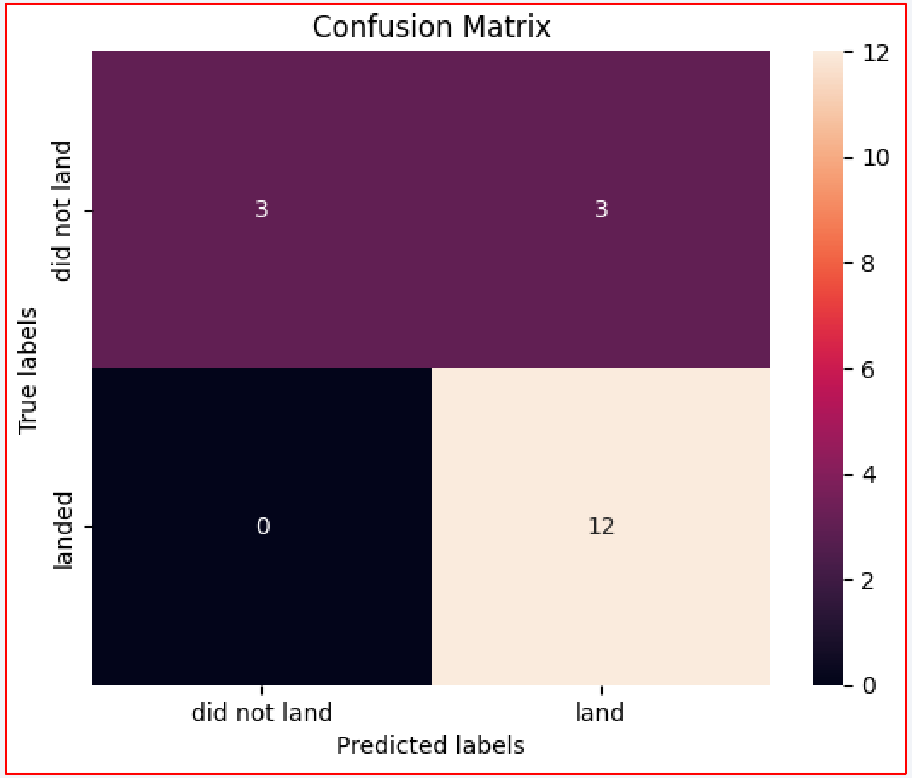


Here a bar chart who Visualize the built model accuracy for all built classification models.

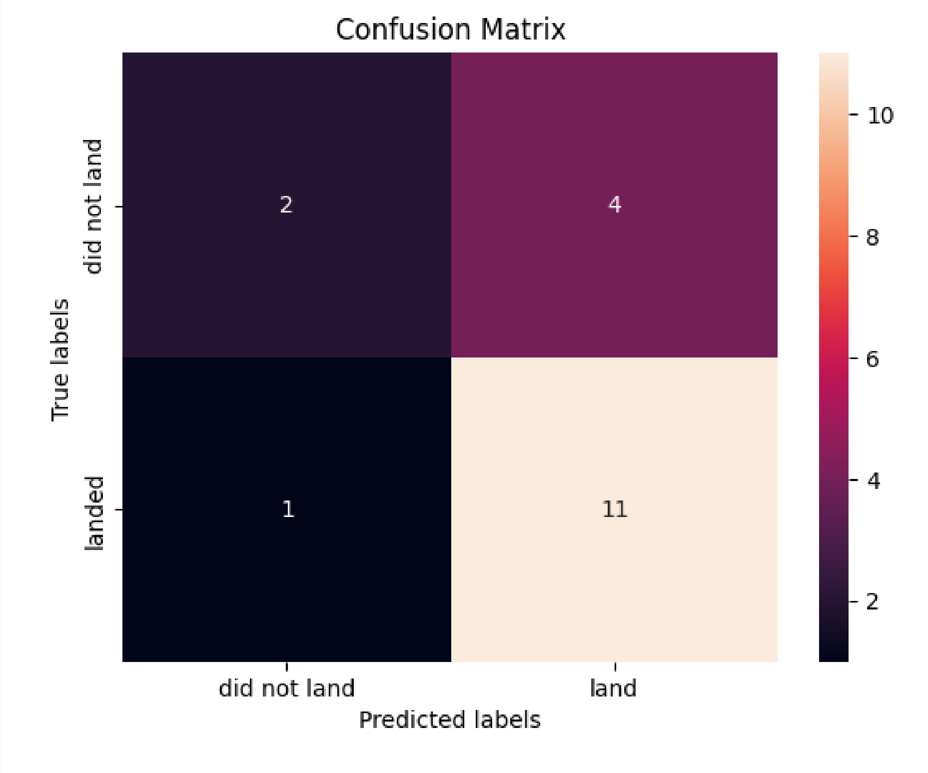
The classification models **KNN, SVM and Logistic Regression** have higher accuracy (**0.83**) than the la classification model Decision Tree (0.72).

I can obtain others result if I use the library **sklearn.metrics** instead (Appendix), like sometime depending of the run Kernel Decision Tree was higher than others models (**0.88**) but it's very random event, the last run show 0.77 for Decision Tree and 0.83 too for others.

Confusion Matrix



Model Logistic Regression, KNN and SVM



Model Decision Tree

Here it's the **Left Confusion Matrix** who is the best, so for the models **Logistic Regression, KNN and SVM** because they have the same confusion matrix. We see the **True Positif is Higher** for the left matrix (12) than the right (11), and there is **no False Negatif** for the left Matrix, but 1 False Negatif for the right One (Decision Tree model).

Conclusions

- The analysis of launch site success rates, orbit choices, and payload characteristics highlights the significance of data-driven decision-making for optimizing space missions and achieving higher success rates over time. The data indicates an **increasing success rate** from 2013 to 2020.

- Regarding Orbits we noticed that **Heavy payloads** show higher success rates for **Polar, LEO, and ISS** orbits. However missions targeting **ES-L1, GEO, HEO, and SSO** orbits exhibited consistently high success rates.

- Regarding lauch sites, **KSC LC-39A** is the one wich have the highest launch success ratio with **76.9%** of success.

- The booster version which have the largest success rate is **FT category** and the payload range with largest succes rate is from **2000 Kg to 6000 Kg**

- To figure out wich classification models perform bets for predictive analysis, we should consider **accuracies** and other factors like interpretability, computational resources, and whether false positives/negatives have a significant impact on modle performance. However **SVM, KNN and Logistic Regression models** are the one performs best regarding our dataset.

Appendix

```python
from sklearn.metrics import accuracy_score

# Get the best hyperparameters
best_svm_params = svm_cv.best_params_
best_tree_params = tree_cv.best_params_
best_logreg_params = logreg_cv.best_params_
best_knn_params = knn_cv.best_params_

# Train models with best hyperparameters on training data
best_svm_model = SVC(**best_svm_params)
best_tree_model = DecisionTreeClassifier(**best_tree_params)
best_logreg_model = LogisticRegression(**best_logreg_params)
best_knn_model = KNeighborsClassifier(**best_knn_params)

best_svm_model.fit(X_train, Y_train)
best_tree_model.fit(X_train, Y_train)
best_logreg_model.fit(X_train, Y_train)
best_knn_model.fit(X_train, Y_train)

# Predictions on test data
svm_predictions = best_svm_model.predict(X_test)
tree_predictions = best_tree_model.predict(X_test)
logreg_predictions = best_logreg_model.predict(X_test)
knn_predictions = best_knn_model.predict(X_test)

# Calculate accuracies
svm_accuracy = accuracy_score(Y_test, svm_predictions)
tree_accuracy = accuracy_score(Y_test, tree_predictions)
logreg_accuracy = accuracy_score(Y_test, logreg_predictions)
knn_accuracy = accuracy_score(Y_test, knn_predictions)

# Print accuracies
print("SVM Accuracy:", svm_accuracy)
print("Decision Tree Accuracy:", tree_accuracy)
print("Logistic Regression Accuracy:", logreg_accuracy)
print("KNN Accuracy:", knn_accuracy)

# Determine which method performs best
print("")
best_method = max(svm_accuracy, tree_accuracy, logreg_accuracy, knn_accuracy)
if best_method == svm_accuracy:
    print("Best method: SVM")
if best_method == tree_accuracy:
    print("Best method: Decision Tree")
if best_method == logreg_accuracy:
    print("Best method: Logistic Regression")
if best_method == knn_accuracy:
    print("Best method: KNN")
```

Here a Python Code and Result Output of Classification accuracies calculation of every models using **sklearn.metrics**.

```
SVM Accuracy: 0.8333333333333334
Decision Tree Accuracy: 0.7777777777777778
Logistic Regression Accuracy: 0.8333333333333334
KNN Accuracy: 0.8333333333333334

Best method: SVM
Best method: Logistic Regression
Best method: KNN
```

Thank you!