IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

RIRHANDZU NOVELA
05-02-2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection using SpaceX API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis using SQL
  - DataViz Using Python Pandas and Matplotlib
  - Launch Sites Analysis with Folium-Interactive Visual Analytics and Ploty Dash
  - Machine Learning Landing Prediction
- Summary of all results
  - EDA results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis(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.
- Problems you want to find answers
- To predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on it’s 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.**
- REST API

➢ The data was collected using SpaceX API by making a get request to the SpaceX API.

➢ The response content was decoded as a Json result which was then converted into a Pandas data frame.

- WEB SCRAPPING

➢ Falcon 9 historical launch records were collected through web crapping from a Wikipedia page.

➢ BeautifulSoup and request Libraries were used to extract the Falcon 9 launch HTML table.

➢ The table was parsed and converted it into a Pandas data frame

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Add the GitHub URL of the completed SpaceX API calls notebook.

- https://github.com/Rirhandzu-Novela/Cousera-Capstone-project/blob/90294ee8569c239982fca0834214d934faabee60/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

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:

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

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

```
[10]: response.status_code
```

```
[10]: 200
```

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

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

Using the dataframe `data` print the first 5 rows

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts
- https://github.com/Rirhandzu-Novela/Cousera-Capstone-project/blob/90294ee8569c239982fca0834214d934faabee60/jupyter-labs-webscraping.ipynb

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

[5]:
```
# 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`

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

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

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

[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

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

[8]:
```
# 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')
```

Data Wrangling

- BoosterVersioncolumn was used to filter the collected data to only keep the Falcon 9 launches.
- Processed missing values in LandingPad and PayloadMass columns.
- For the PayloadMass, missing data values were replaced using mean value of column.
- Exploratory Data Analysis (EDA) was used to find some patterns in the data and determine the label for training supervised models
- https://github.com/Rirhandzu-Novela/Cousera-Capstone-project/blob/90294ee8569c239982f ca0834214d934faabee60/labs-jupyter-spacex-Data%20wrangling.ipynb

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`:

[10]:
```
# 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()
```

[10]:
```
1    60
0    30
Name: Class, dtype: int64
```

[12]:
```
landing_class=df['Class']
df[['Class']].head(8)
```

[12]:

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

EDA with Data Visualization

- Pandas and Matplotlib were used for Exploratory Data Analysis and preparing Data Feature Engineering
- DATA VISUALIZATION

➢ The relationship between Flight Number and Launch Site, Payload and Launch Site, FlightNumber and Orbit type, Payload and Orbit type were visualized using scatter plots.

➢ The relationship between success rate of each orbit type was visualized using barplots.

➢ The yearly trend of launching success was visualized using line graph.

https://github.com/Rirhandzu-Novela/Cousera-Capstone-project/blob/90294ee8569c239982fca0834214d934faabee60/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

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 as "Launch_Sites" FROM SPACEXTBL;
```

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

```
%sql SELECT * 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_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

- 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 Booster_Version LIKE 'F9 v1.1%';
```

EDA with SQL

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

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

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

```
[13]: %sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

- 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";
```

- https://github.com/Rirhandzu-Novela/Cousera-Capstone-project/blob/e65dfba72fcd1e413594cc991998574fb2d7ca4f/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Created folium map and mark all the launch sites
- Created map objects to mark the success or failure of launches for each launch site.
- Created a launch set outcomes
- https://github.com/Rirhandzu-Novela/Cousera-Capstone-project/blob/e65dfba72fcd1e413594cc991998574fb2d7ca4f/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- Built an interactive dashboard application with Plotly dash

➢ Add a Launch Site Drop-down Input Component

➢ Add a callback function to render success-pie-chart based on selected site dropdown

➢ Add a Range Slider to Select Payload

➢ Add a callback function to render the success-payload-scatter-chart scatter plot

- https://github.com/Rirhandzu-Novela/Cousera-Capstone-project/blob/e65dfba72fcd1e413594cc991998574fb2d7ca4f/spacex_dash_app.py

Predictive Analysis (Classification)

- To determine Training Labels
  - NumPy array was created from the column Class in data.
  - Standardized the feature dataset (x).
  - Split the data into training and testing sets.
- To find the best model
  - Created a GridSearchCV object and fit the training data
  - Get the best hyper parameters for the model
  - Use the best parameters to predict the test data
  - Calculate the accuracy on the test data for each model
  - Plot a confusion matrix for each using the test and predicted outcomes

Predictive Analysis (Classification)

- https://github.com/Rirhandzu-Novela/Cousera-Capstone-project/blob/e65dfba72fcd1e413594cc991998574fb2d7ca4f/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

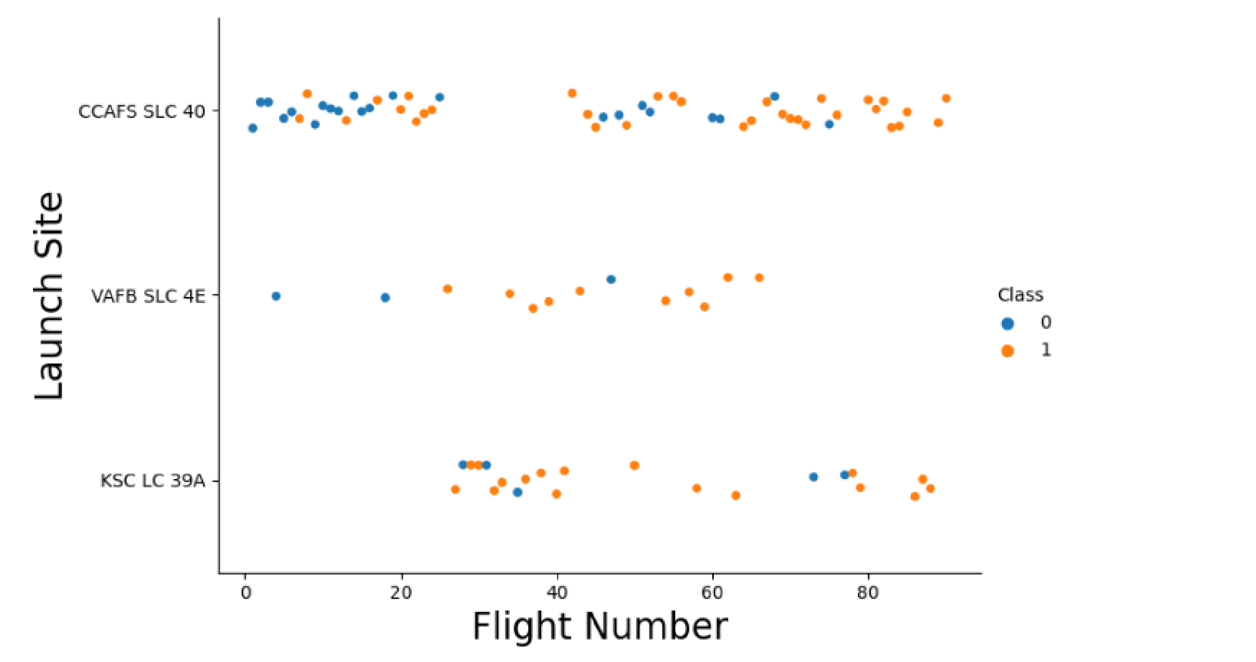
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

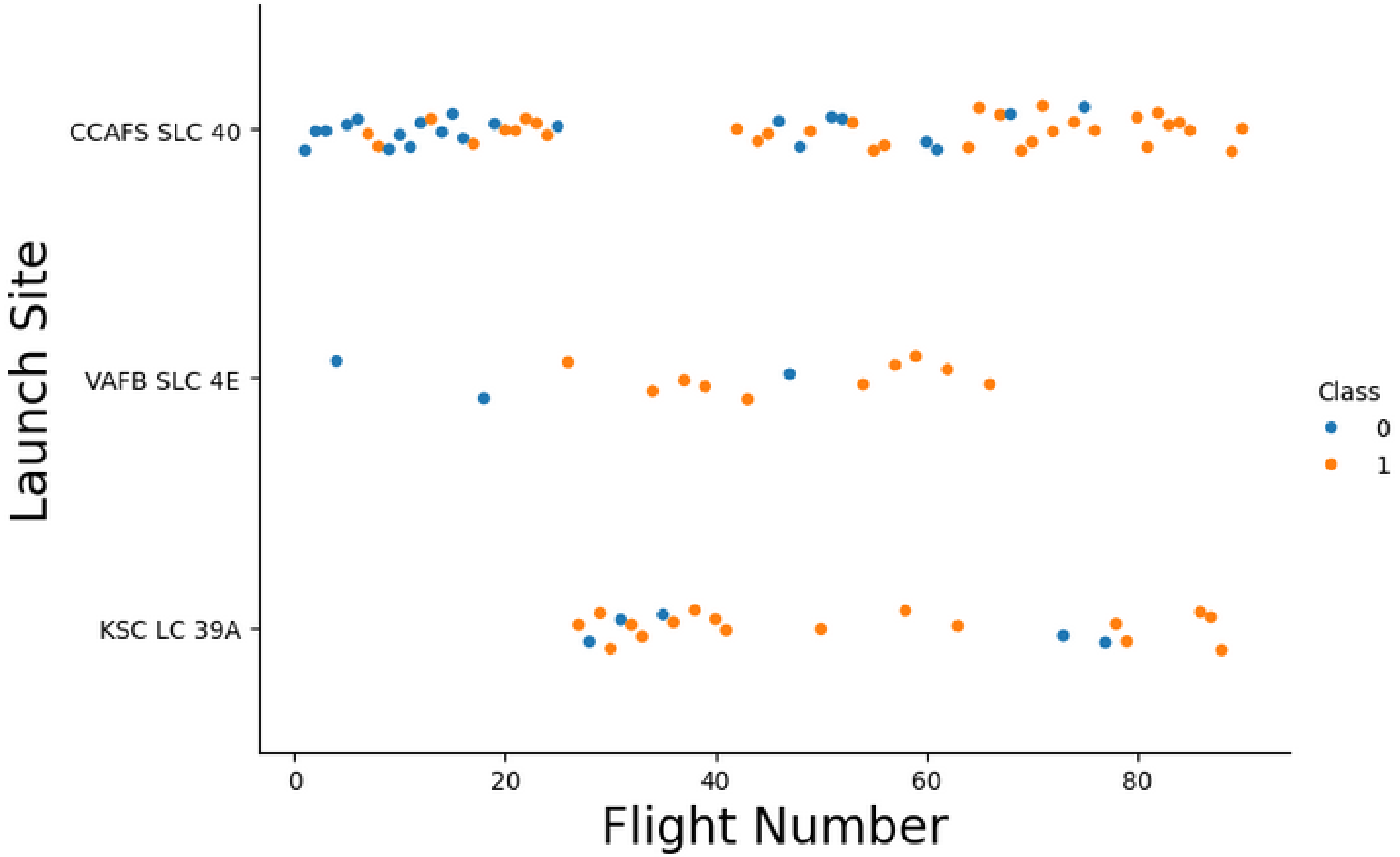
Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site



Flight Number vs. Launch Site

- Show the screenshot of the scatter plot with explanations

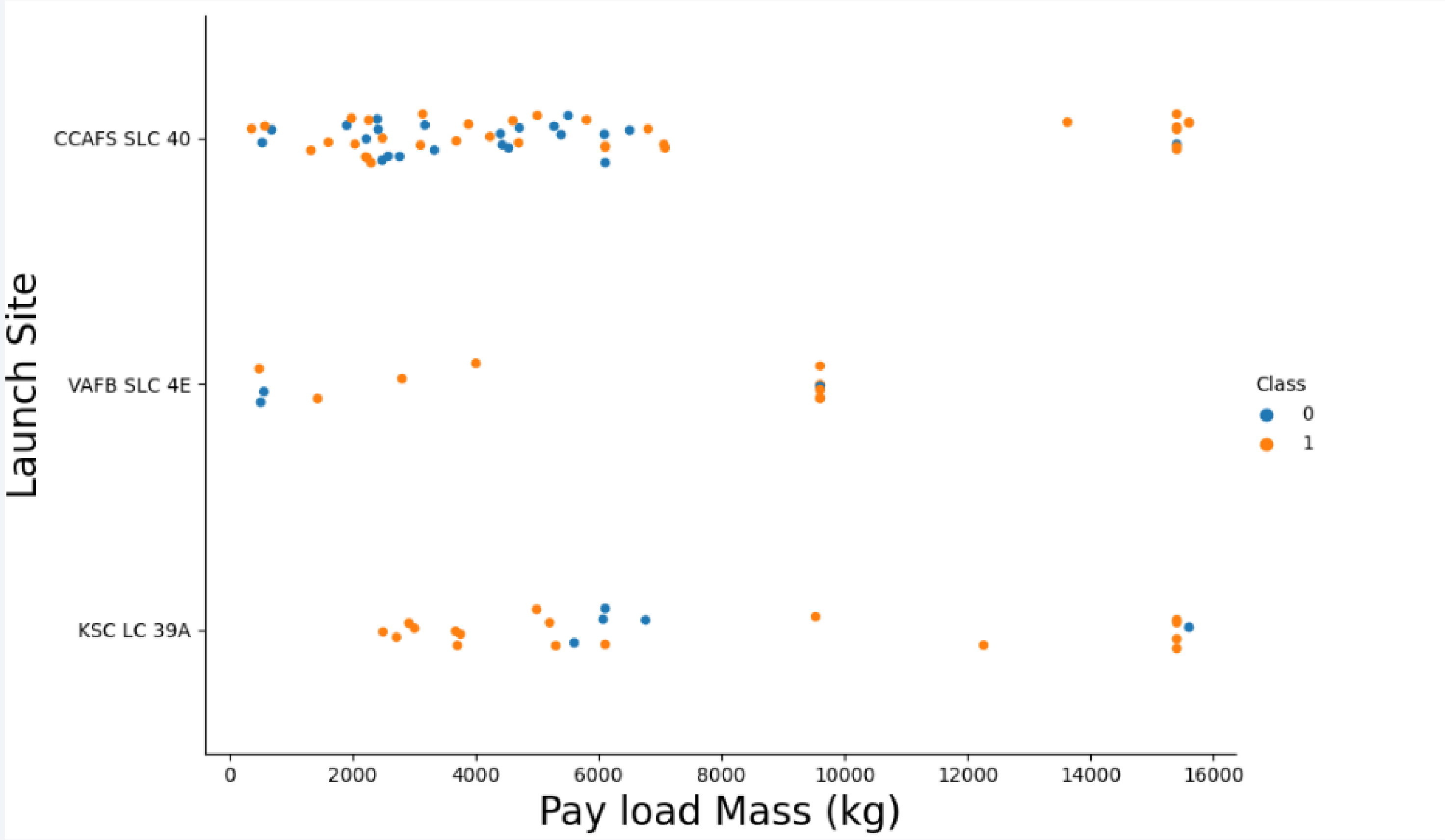


Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

```
Site VAFB SLC 4E achhieved 100% success rate first and has minimun number of failures. Site CCAFS SLC 40 has high number of failures.
```

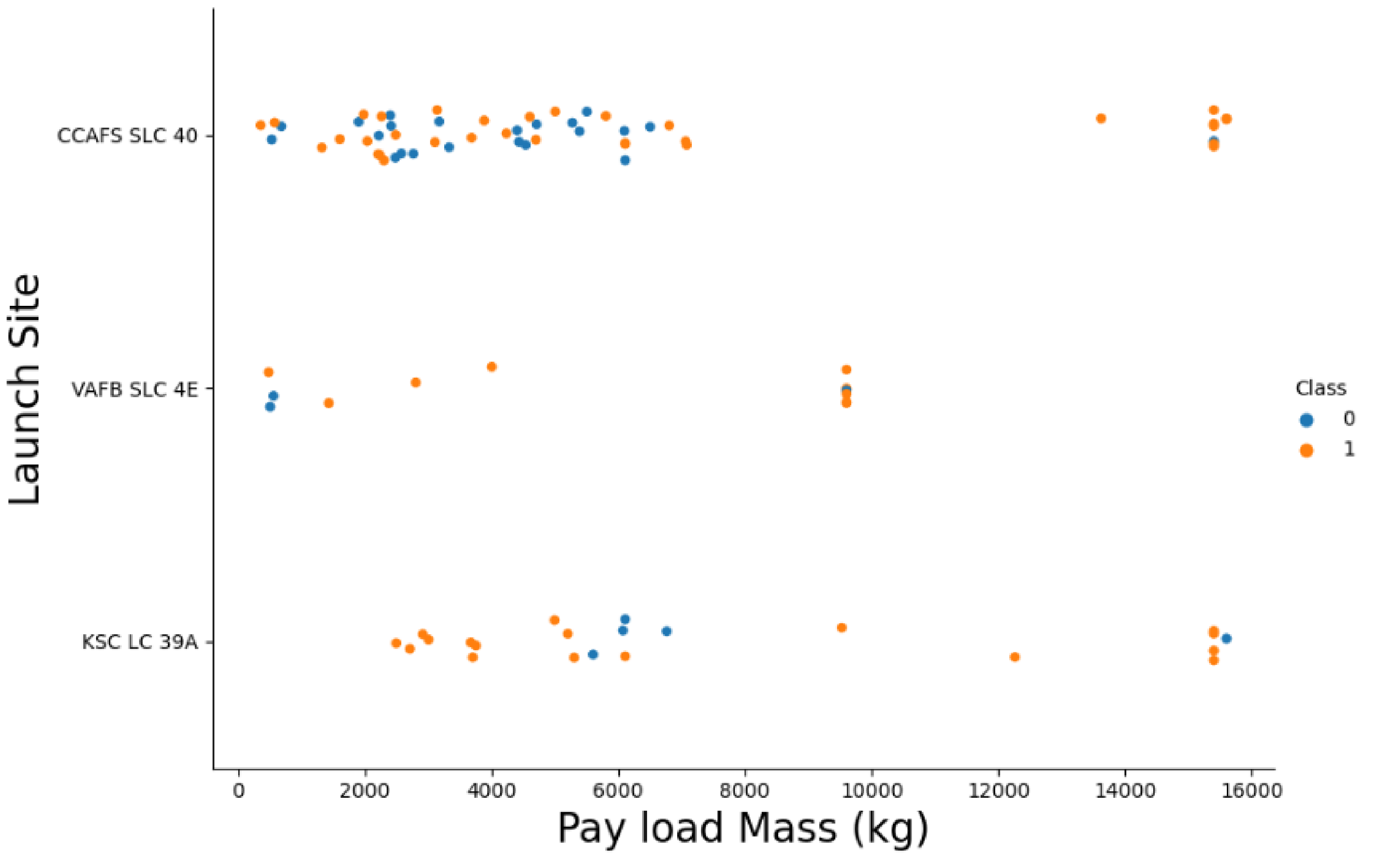
Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site



Payload vs. Launch Site

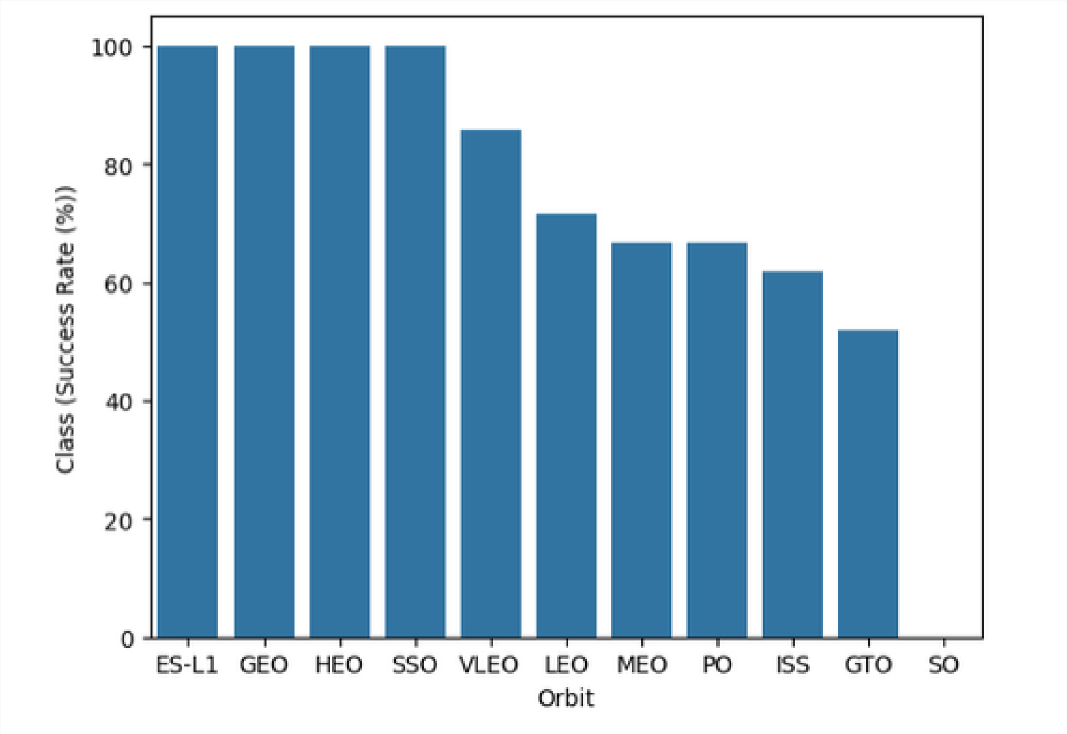
- Show the screenshot of the scatter plot with explanations



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).

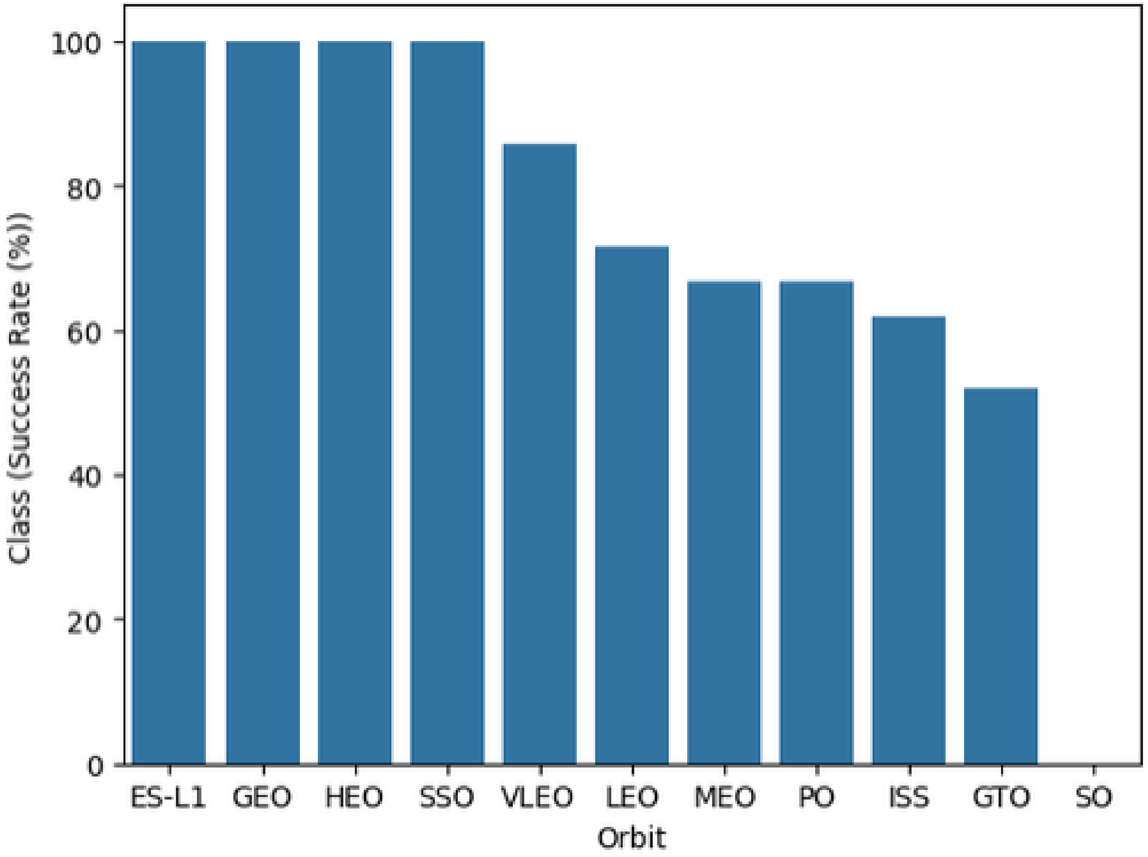
Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type



Success Rate vs. Orbit Type

- Show the screenshot of the scatter plot with explanations

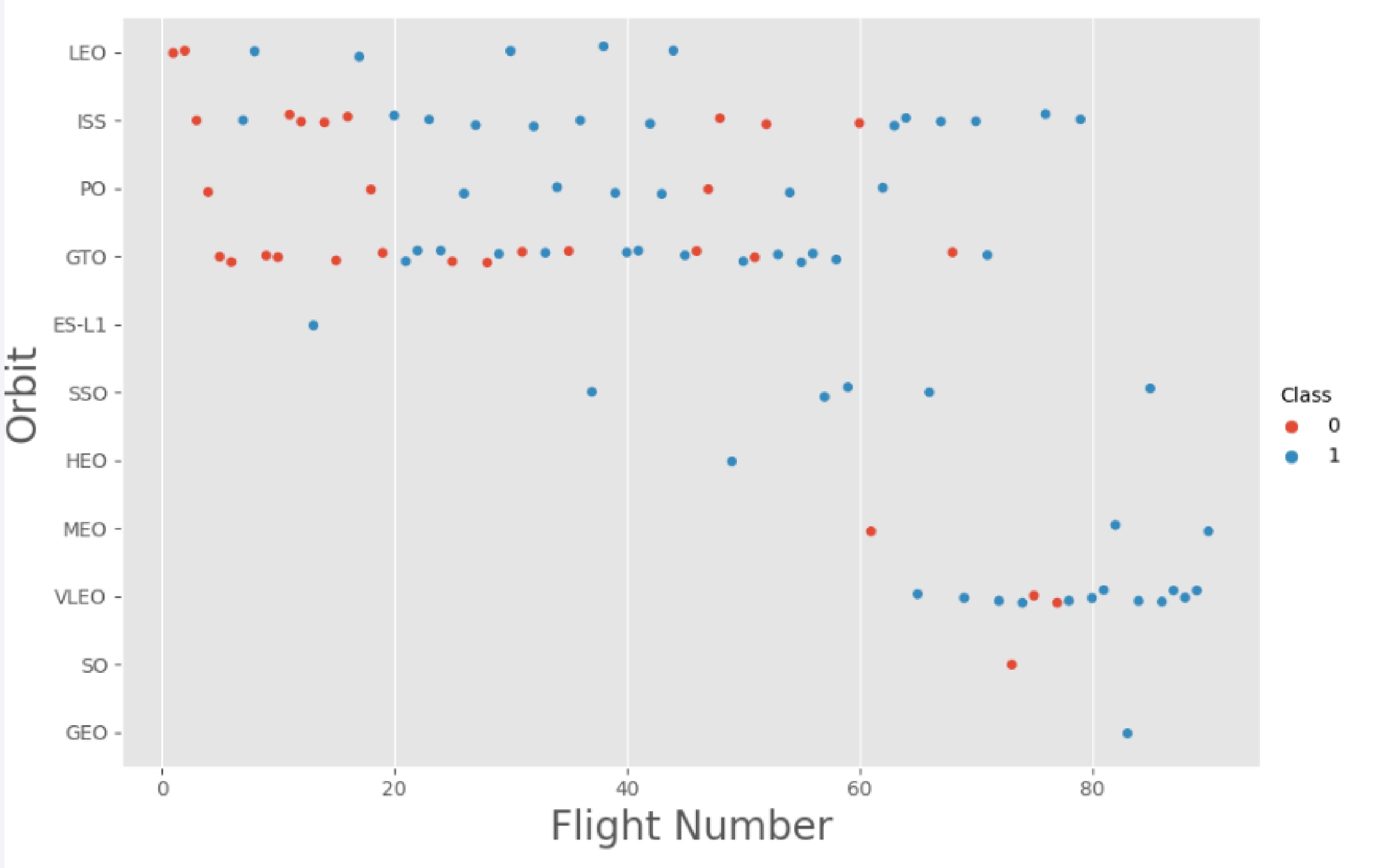


Analyze the ploted bar chart try to find which orbits have high sucess rate.

```
[ ]: ES-L1, GEO, HEO, SSO have high success rate.
```

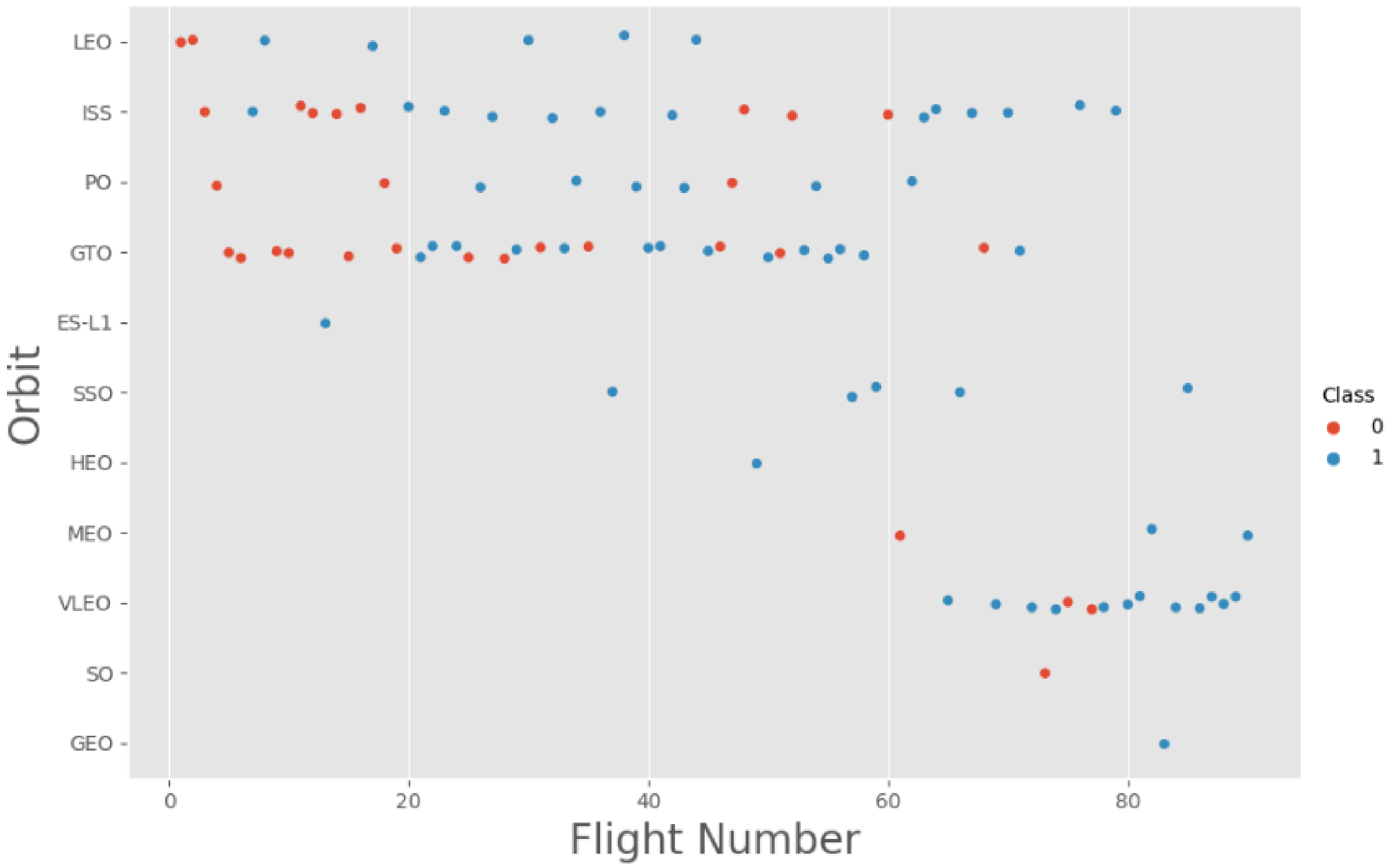
Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type



Flight Number vs. Orbit Type

- Show the screenshot of the scatter plot with explanations



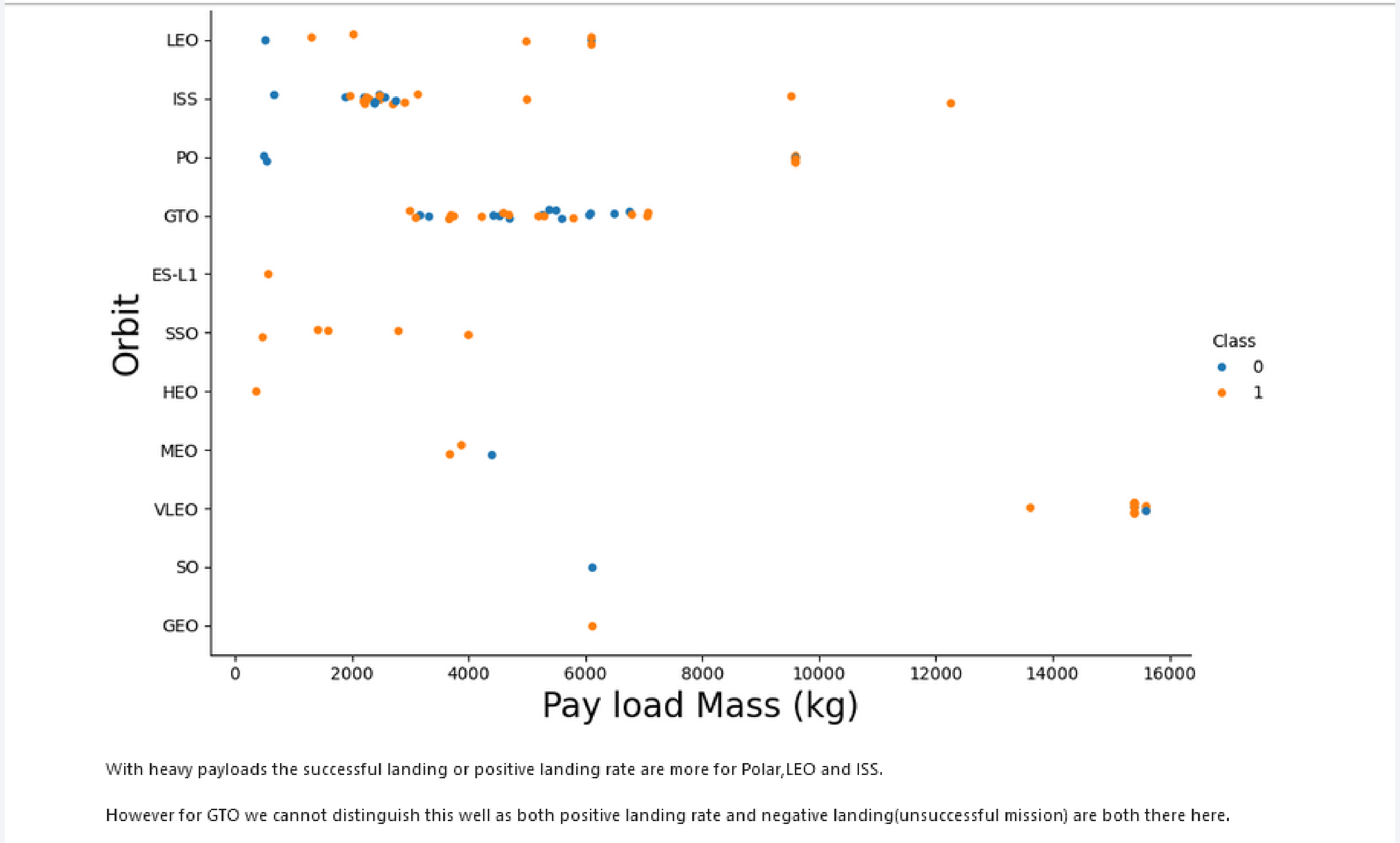
You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type

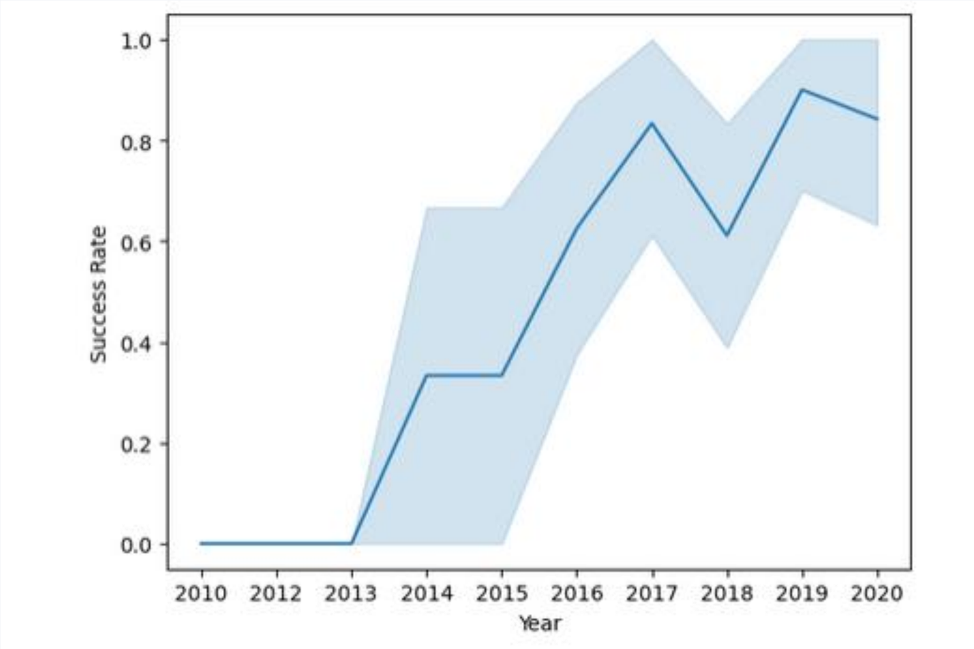
Payload vs. Orbit Type

- Show the screenshot of the scatter plot with explanations



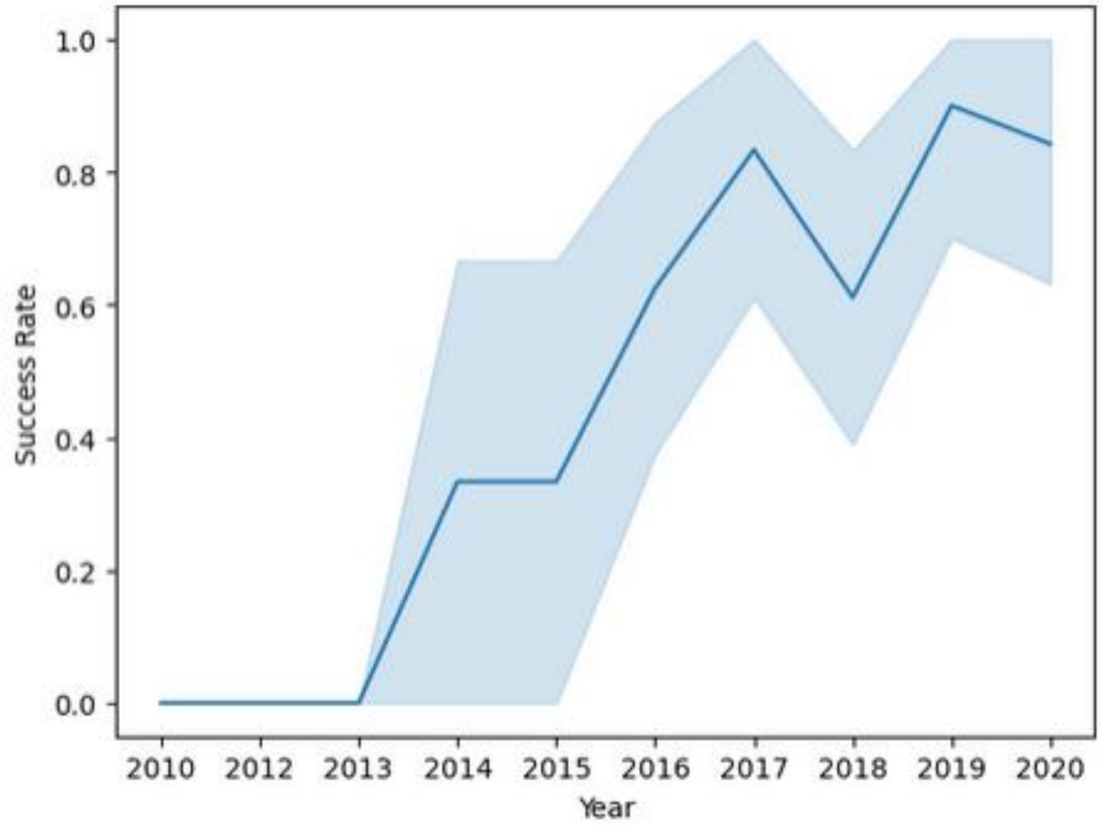
Launch Success Yearly Trend

- Show a line chart of yearly average success rate



Launch Success Yearly Trend

- Show the screenshot of the scatter plot with explanations



you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

- Find the names of the unique launch sites

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

Task 1

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

```
[7]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

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

[7]:

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

Launch Site Names Begin with 'CCA'

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

- ‘LIKE’ command with ‘%’ wildcard in ‘WHERE’ clause select and dispay a table of all records where launch sites begin with the string 'CCA'

Task 2

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

```
[9]: %sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

* sqlite:///my_data1.db
Done.

[9]:

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

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

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

Task 3

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

[8]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

[8]:

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

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- ‘AVG()’ function return and display the average payload mass carried by booster version F9 v1.1

Task 4

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

```
[11]: %sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

[11]:

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

First Successful Ground Landing Date

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

- ‘MIN()’ function return and display the first date when first successful landing outcome on ground pad ‘Success (ground pad)’ happened.

Task 5

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

Hint:Use min function

```
[14]: %sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";

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

[14]:

MIN(DATE)
None

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- ‘Select Distinct’ statement return and list the ‘unique’ names of boosters with operators >4000 and <6000 to only list booster with payloads between 4000-6000 with landing outcome of ‘Success (drone ship)’.

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

```
[15]: %sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

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

[15]:

Booster_Version	Payload

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

- ‘COUNT()’ and ‘GROUP BY’ statement return total number of missions outcomes

Task 7

List the total number of successful and failure mission outcomes

[16]:
```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

[16]:

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

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Subquerry return and pass the Max payload and list all the boosters that have carried the Max payload of 15600kgs

Task 8

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

```
[17]: %sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL)

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

[17]:

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 failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- ‘subsrt()’ in the select statement get the month and year from the date column where substr(Date,6,2)='2015' for year and Landing_outcome was ‘Failure (drone ship’) and return the records nmatching the filter.

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.

```
[20]: %sql SELECT substr(Date,6,2), substr(Date,0,5),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing_Outcome"
```

```
 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: Date
[SQL: SELECT substr(Date,6,2), substr(Date,0,5),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing_Outcome"]
(Background on this error at: http://sqlalche.me/e/13/e3q8)
```

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
- SELECT From SPACESTBL WHERE “Landing_Outcome” LIKE “success” AND DATE BETWEEN 2010-06-04 and 2017-03-20, ORDER BY DATE

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.

[21]:
```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date
```

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

[21]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome

Section 3

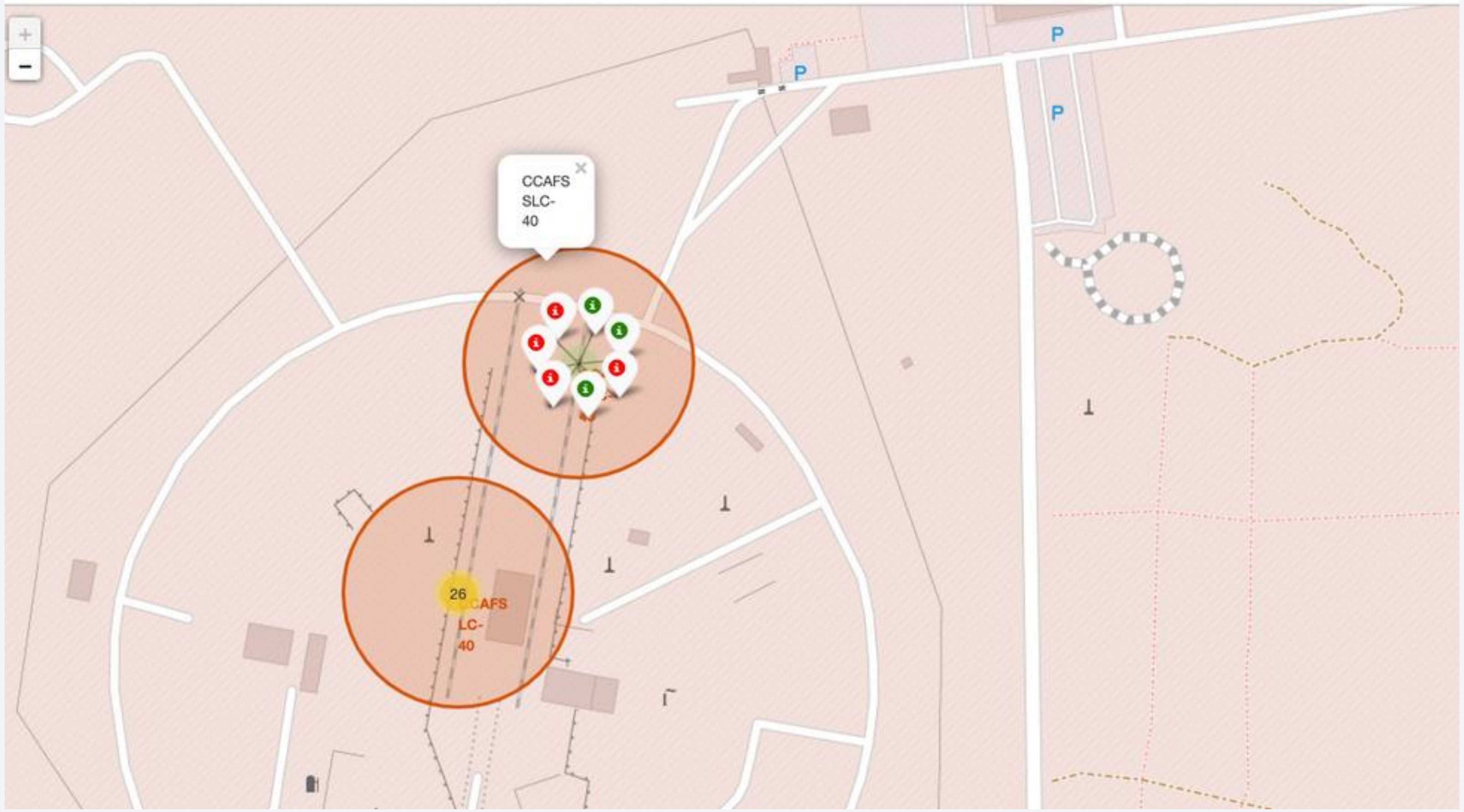
Launch Sites Proximities Analysis

Folium Map of all launching sites

- All the launch sites are along the coast.

Markers showing the success and failure rate

- CCAFS SLC-40 has low success rate compared to failure rate



The distance between the coast and the launch site

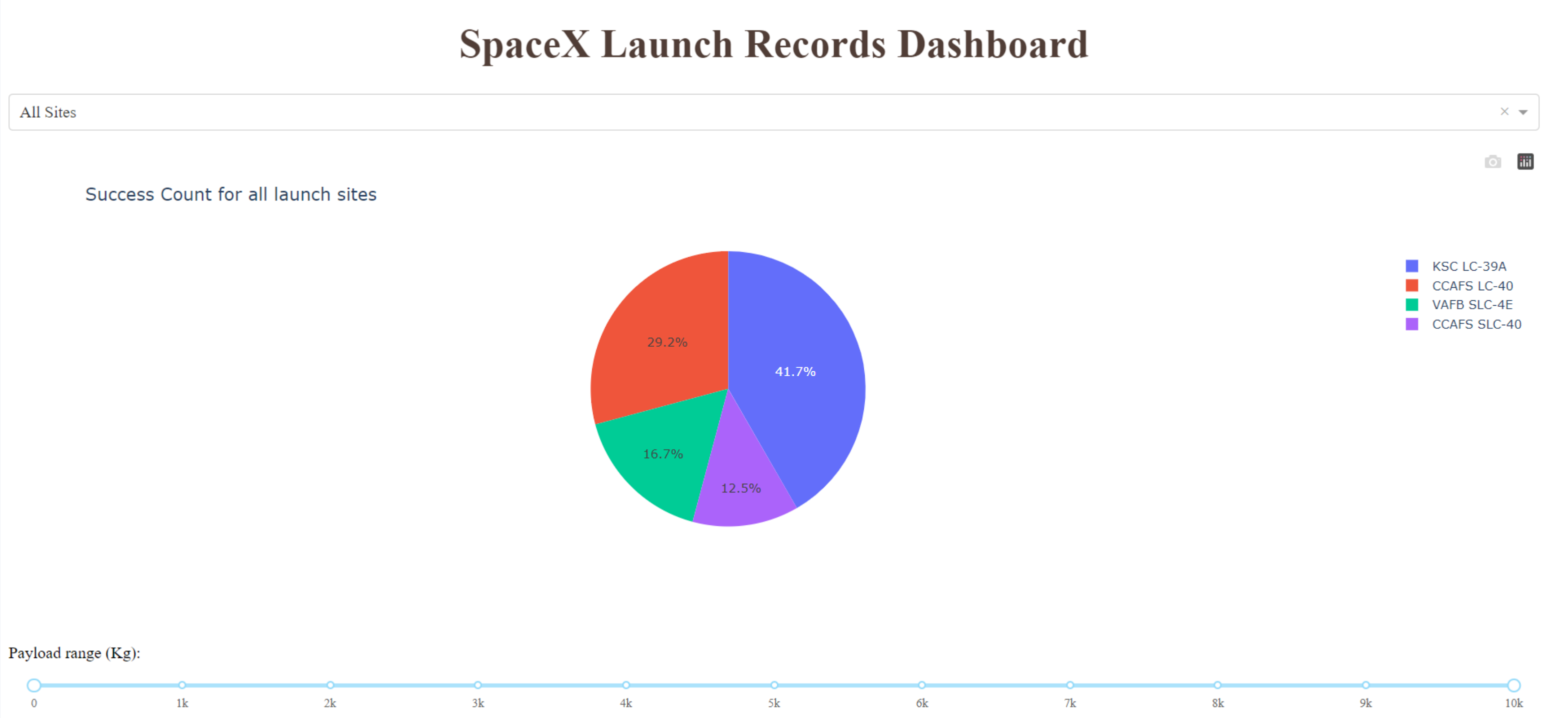
- Launch site CCAFS SLC-40 is 0.90km away from the coast

Section 4

Build a Dashboard with Plotly Dash

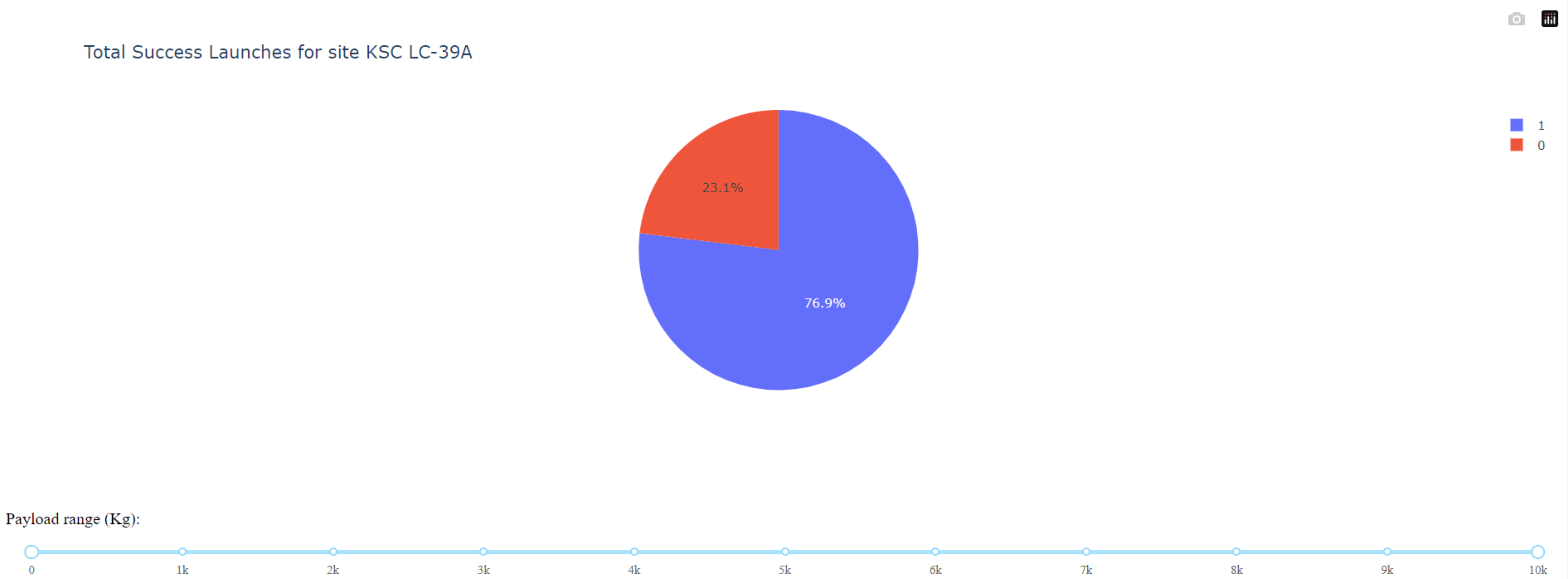
The comparison between success rate per site

- Launch site CCAFS SLC-40 has the lowest success rate of 12,5% and KSC LC-39A has the highest launch success rate at 41,7%. CCAFS LC-40 comes second with 29,2% and VAFB SLC-4E comes third with 16,7%



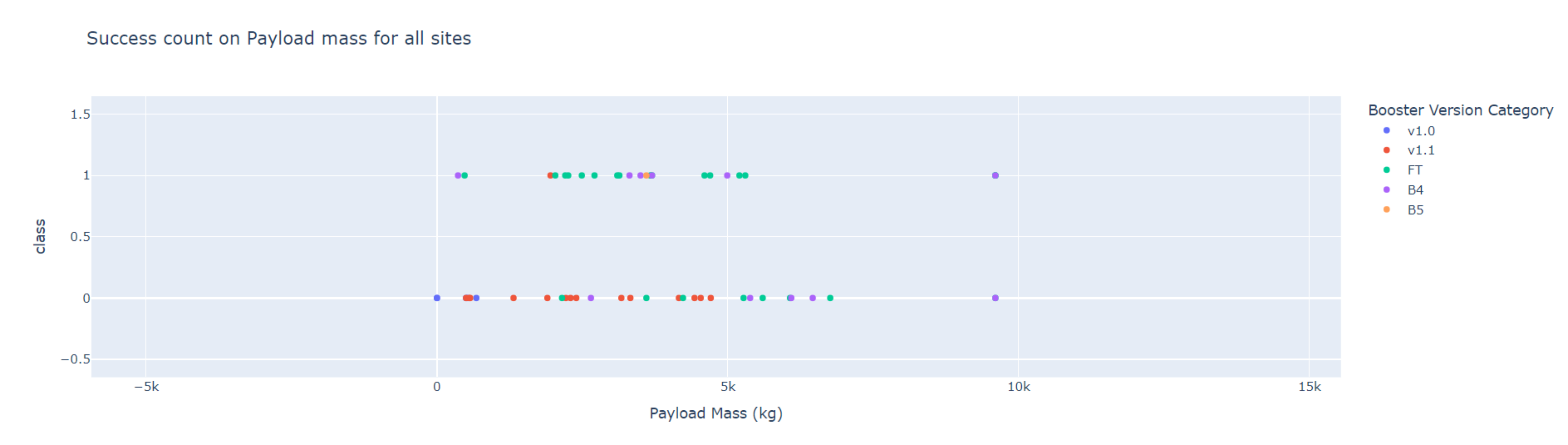
The ratio between success and failure of the site with highest success

- Launch site KSC LC-39A had the highest success ratio of 76,9% success launches



Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

- 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

- All the models have the same accuracy

Out[68]:

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

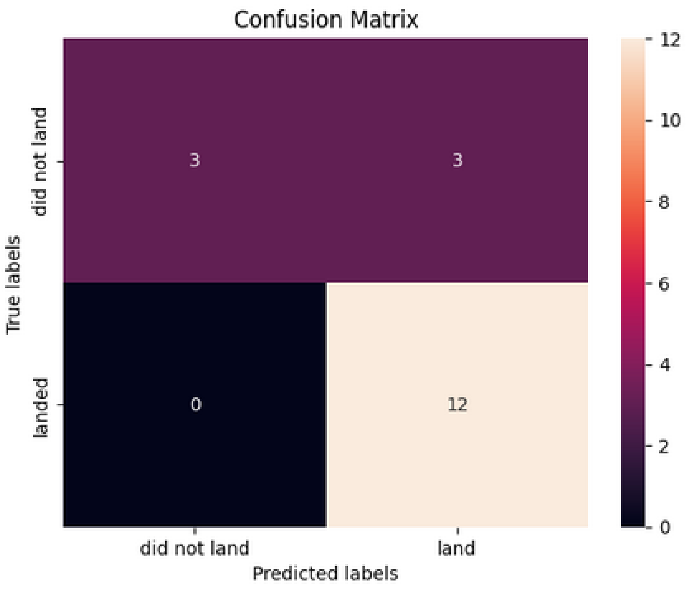
All the methods perform equally on the test data: i.e. They all have the same accuracy of 0.833333 on the test Data

Confusion Matrix

- All the 4 classification model were able equally distinguish between the different classes. The major problem is false positives for all the models.

Find the method performs best:

```
[34]:
yhat = knn_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
plt.show()
```



Conclusions

- Launch site KSC LC-39A have the highest success rate. It is recommended to use this site.
- The success rate increase with an increase in flight number
- Orbits ES-L1, GEO, HEO & SSO have the highest success rates
- The success rate since 2013 kept increasing till 2020.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!