IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Abanoub George
04.09.2023

Outline

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

Executive Summary

- Summary of methodologies:
    - Data Collection using SpaceX API.
    - Data Collection using Web Scraping.
    - Data Wrangling using Python.
    - Exploratory Data Analysis using SQL.
    - Exploratory Data Analysis & Data Visualization using Python.
    - Interactive Dashboards using Python.
    - Predictive Analysis using Machine Learning.

- Summary of all results:
    - Data after Data Wrangling.
    - EDA Results.
    - Dashboard.
    - Classification Results.

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.

- Problems you want to find answers:
  - In this project, we will predict if the Falcon 9 first stage will land successfully.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using the SpaceX API.
- Perform data wrangling
  - We did convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.
- 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 were collected using the RESTful GET Request on the SpaceX data API. Then we requested and parsed the data. This occurred by defining a series of helper functions that will help us use the API to extract information using identification numbers in the launch data.

- In the second phase, we decoded the response content as a JSON using the .json() method and turned it into a Pandas dataframe.

- Finally, we did some web scraping to collect Falcon 9 historical values using BeautiflSoap and request to extract the data from the HTML table.

Data Collection – SpaceX API

- Data were collected using the RESTful GET Request on the SpaceX data API. Then we requested and parsed the data. This occurred by defining a series of helper functions that will help us use the API to extract information using identification numbers in the launch data.

- Here is the link to the Data Collection Notebook: [GitHub URL]

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:

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

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

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

Out[10]: 200

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

In [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

In [12]:
```
# Get the head of the dataframe
data.head()
```

Data Collection - Scraping

- We did some web scraping to collect Falcon 9 historical values using BeautiflSoap and request to extract the data from the HTML table.
- Here is the link to the Data Collection Notebook: [GitHub URL]()

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.

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

response = requests.get(static_url).text
```

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, 'html.parser')
```

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

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

In [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

- We performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models. Also, we will mainly converted those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

- Here is the link to the Data Collection Notebook: [GitHub URL]()

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`:

In [10]:
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = [0 if x in bad_outcomes else 1 for x in df['Outcome']]
```

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

In [11]:
```
df['Class']=landing_class
df[['Class']].head(8)
```

Out[11]:

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

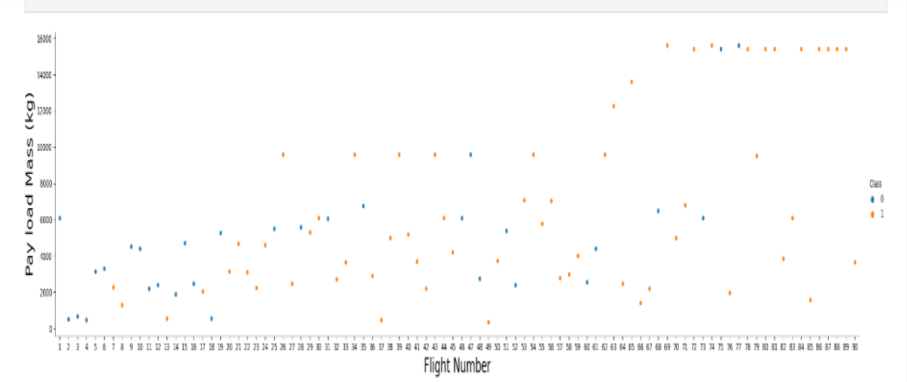
EDA with Data Visualization

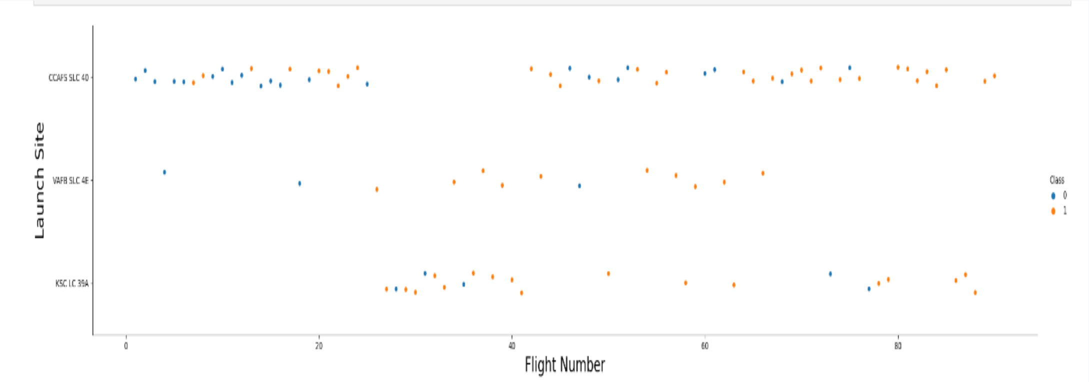
- In this assignment, we did predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is due to the fact that SpaceX can reuse the first stage. We also performed Exploratory Data Analysis and Feature Engineering.
- Here is the link to the Data Collection Notebook: GitHub URL

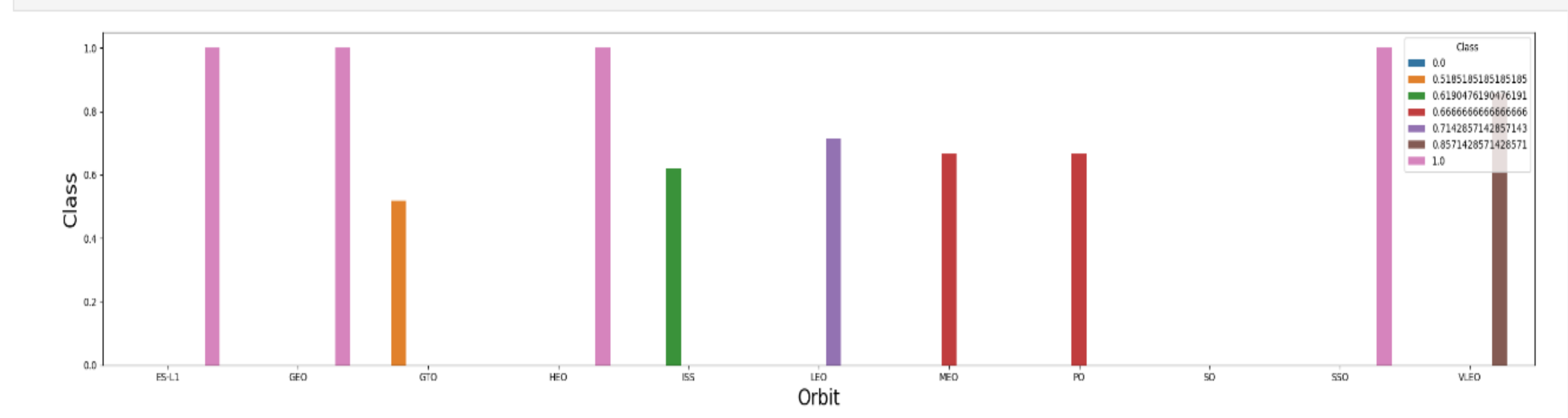
Objectives:

- Perform exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib
- Exploratory Data Analysis
- Preparing Data Feature Engineering

Cont EDA with Data Visualization







EDA with SQL

- Here are some of the queries used for EDA, link: GitHub URL:
  - Display the names of the unique launch sites in the space mission:

```
%sql SELECT DISTINCT(Launch_Site) FROM SPACEXTBL;
```

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

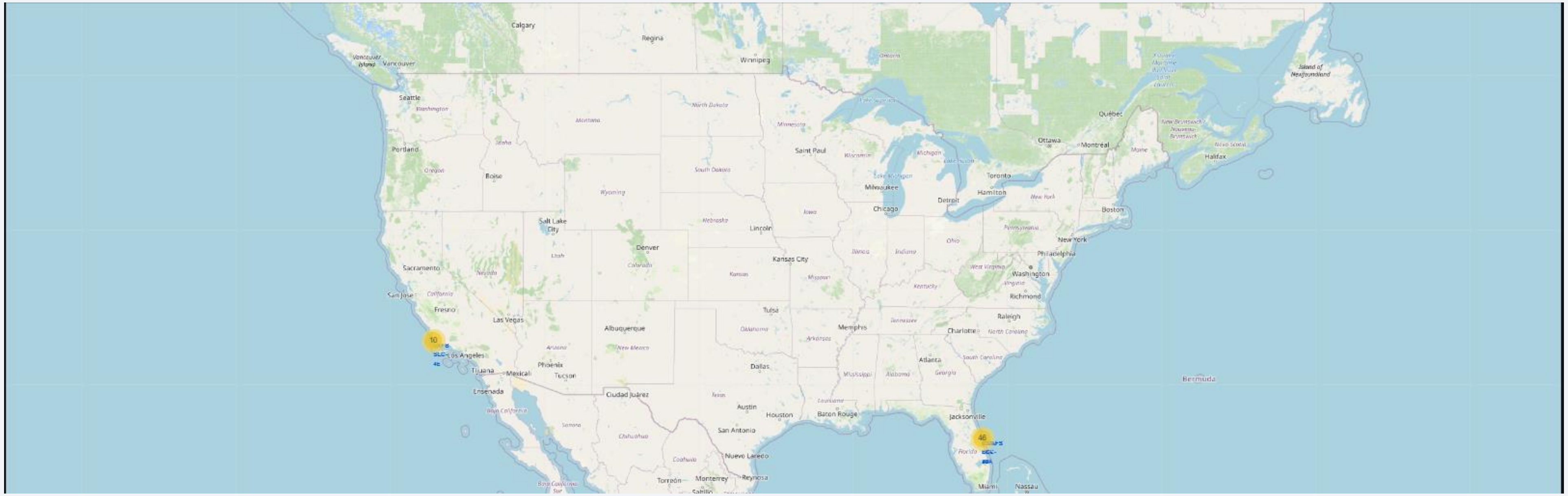
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Customer = 'NASA (CRS)';
```

  - List the names of the booster_versions which have carried the maximum payload mass:

```
%sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

Build an Interactive Map with Folium

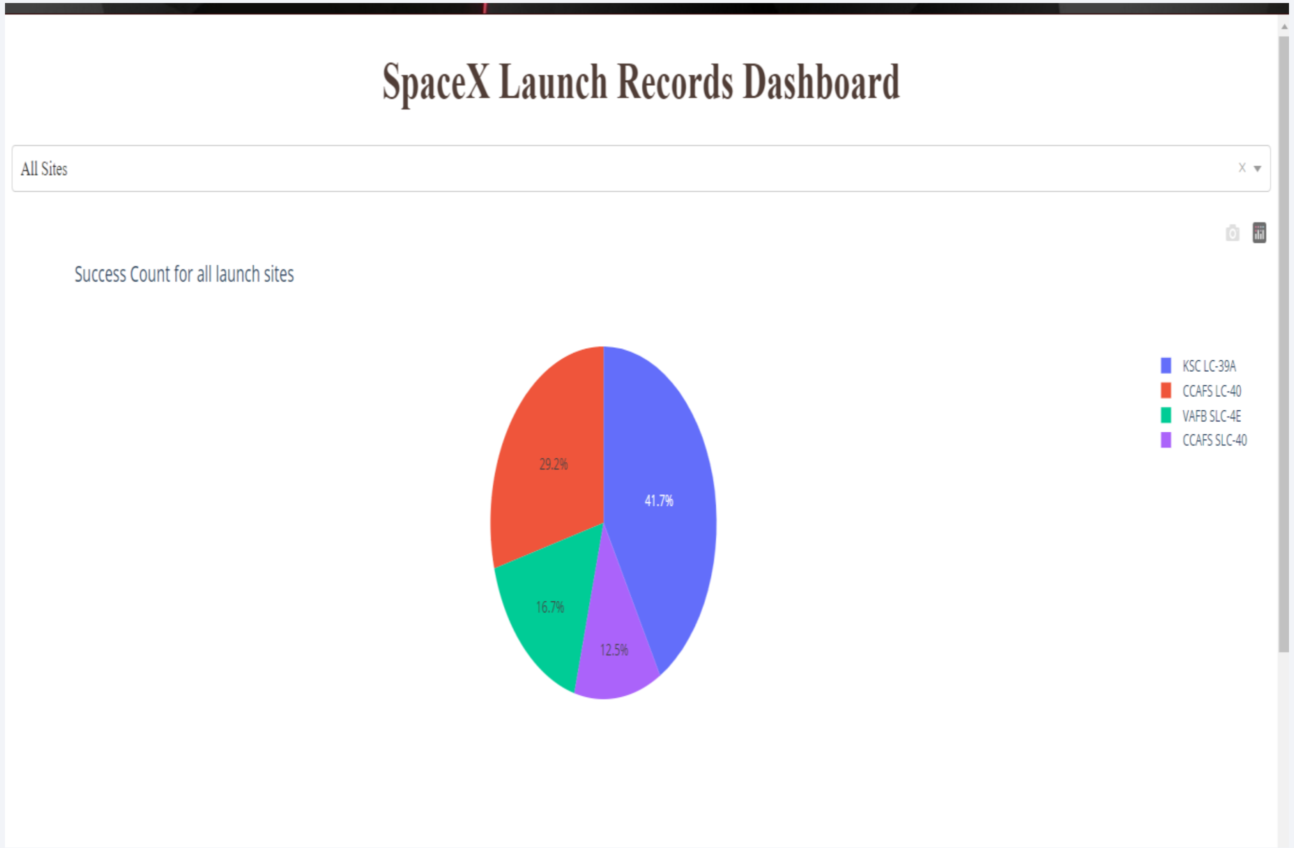
- Created an interactive Folium map that marked all the launch sites, and created map objects such as markers, circles, lines to mark the success or failure of launches for each launch site. Link to the notebook: [GitHub URL]



Build a Dashboard with Plotly Dash

- Built an interactive dashboard application with Plotly Dash.

- Link to the python file: [GitHub URL]



Predictive Analysis (Classification)

- After loading the data as a Pandas Dataframe, I performed some Exploratory Data Analysis and determined the Training Labels by:
  - Creating a NumPy array from the column Class in data, by applying the method to_numpy() then assigned it to the variable Y as the outcome variable.
  - Then standardized the feature dataset (x) by transforming it using preprocessing.StandardScaler() function from Sklearn.
  - After that 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 .2 and random_state to 2.

Cont 1 Predictive Analysis (Classification)

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

Cont 2 Predictive Analysis (Classification)

- Here are the models results, they all have the same accuracy and confusion matrix. Here is also the link: GitHub URL

	0
Model	Accuracy
Logistic Regression:	83.333333
SVM:	83.333333
Decision Tree:	83.333333
KNN:	83.333333

Results

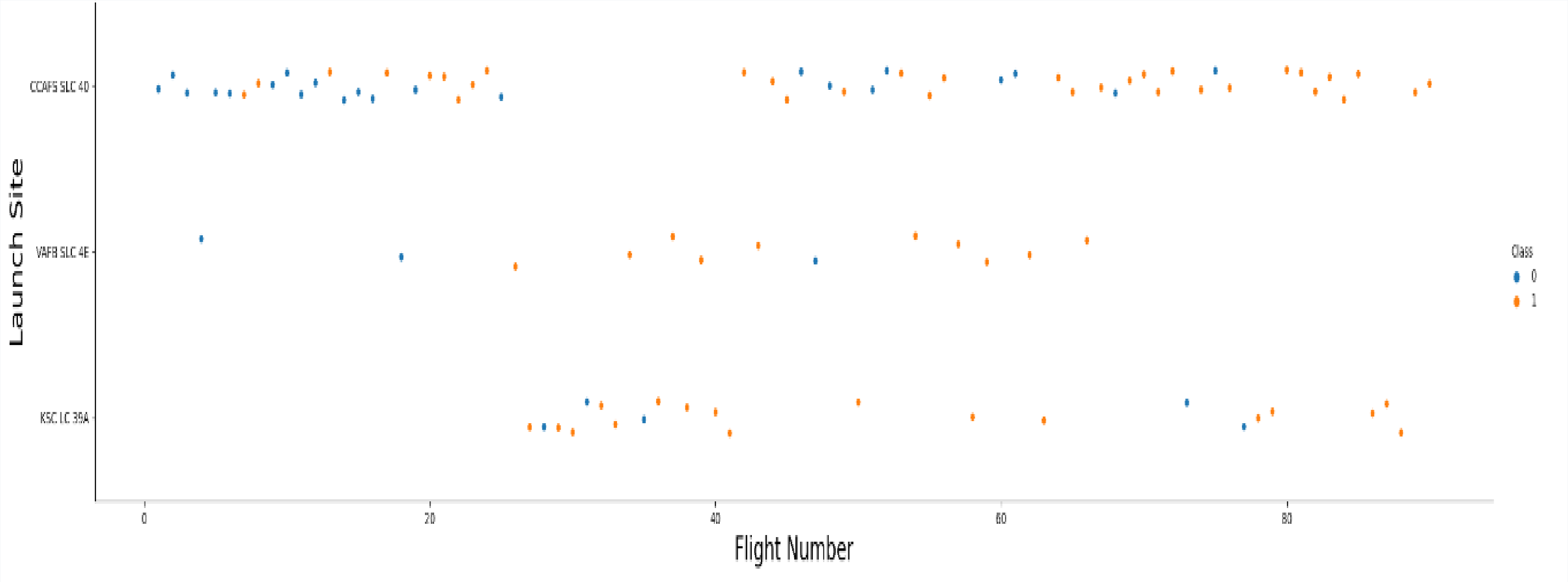
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

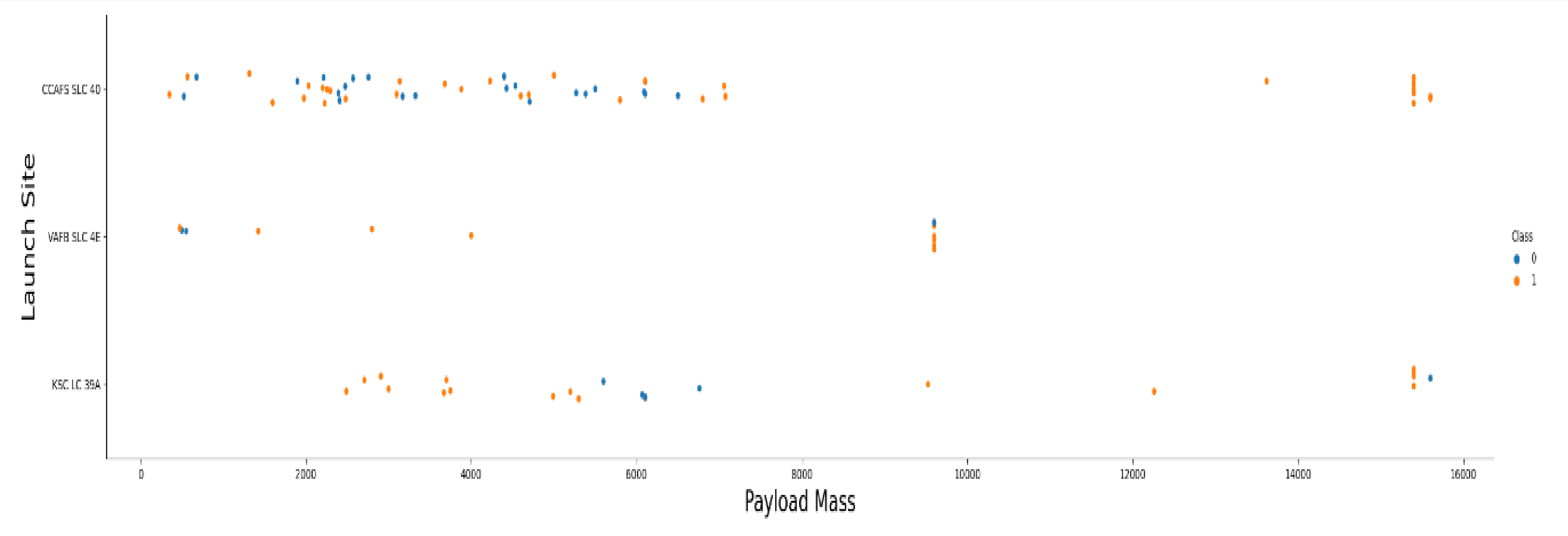
Flight Number vs. Launch Site

- We see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.



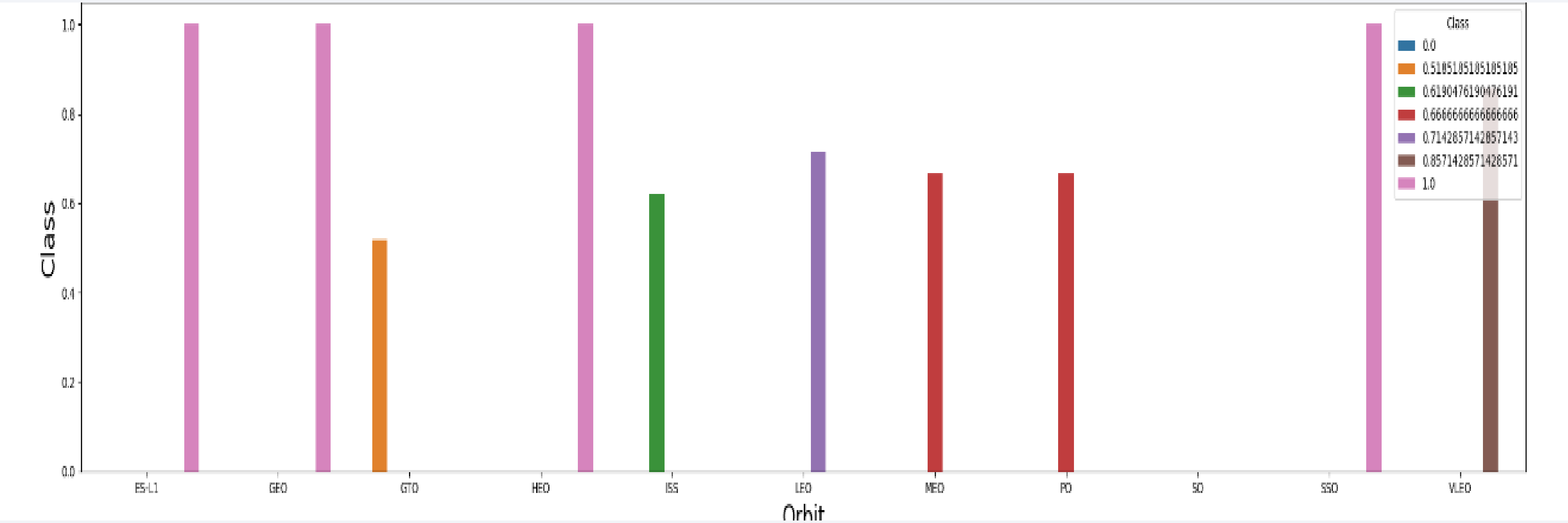
Payload vs. Launch Site

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



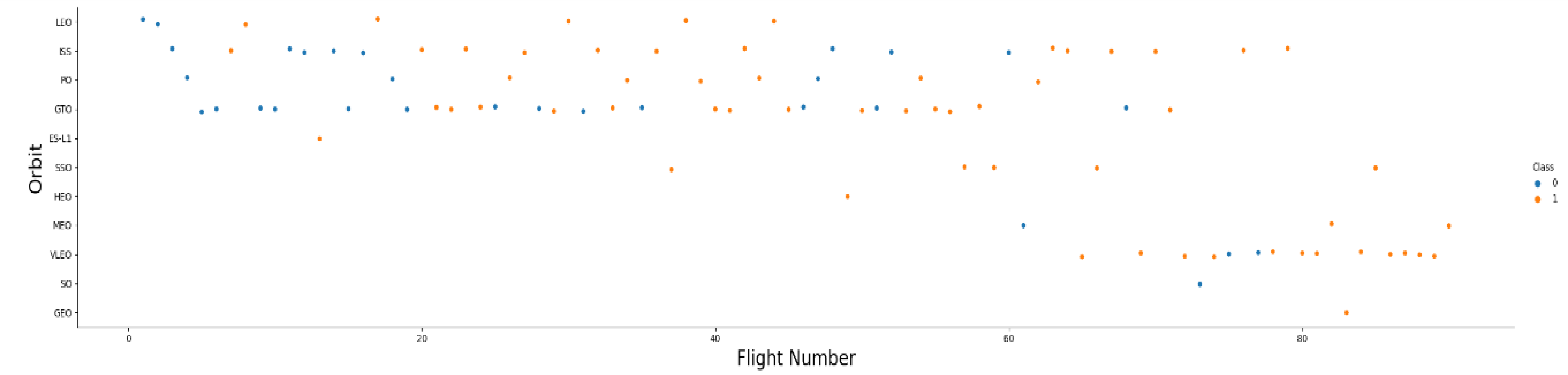
Success Rate vs. Orbit Type

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



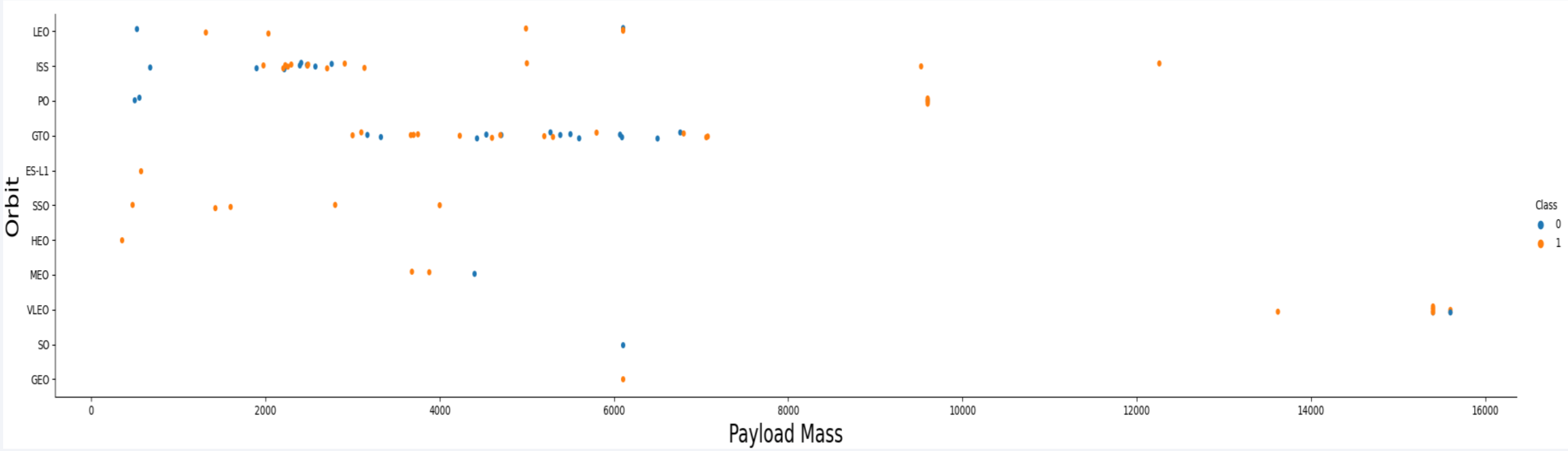
Flight Number vs. Orbit Type

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



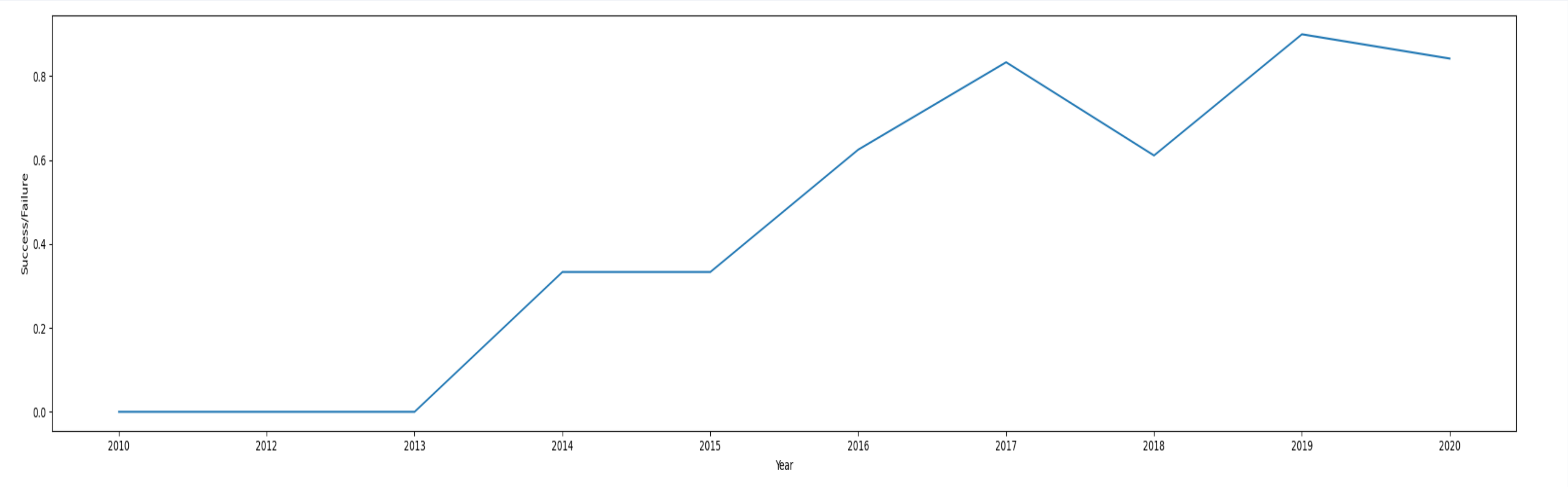
Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.



Launch Success Yearly Trend

- You can observe that the success rate since 2013 kept increasing till 2020.



All Launch Site Names

- Display the names of the unique launch sites in the space mission. We used the SELECT DISTINCT to select all unique launch sites names.

In [7]:
```
%sql SELECT DISTINCT(Launch_Site) FROM SPACEXTBL;
```

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

Out[7]:

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

Launch Site Names Begin with 'CCA'

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

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

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

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

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

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1';
```

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

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

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

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

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

MIN(DATE)
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

- Used ‘Select Distinct’ statement to 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)’.

```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOA
```

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

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

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

```
%sql SELECT Mission_Outcome, Count(*) as Total FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

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

Boosters Carried Maximum Payload

- Using a Subquery to return and pass the Max payload and used it list all the boosters that have carried the Max payload of 15,600KG.

```
%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.
```

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

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

```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing _Outcome"
```

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

substr(Date,7,4)	substr(Date, 4, 2)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing _Outcome
2015	01	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
2015	04	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 successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT [Landing _Outcome], count(*) as Total_Outcomes FROM SPACEXTBL WHERE DATE BETWEEN '04-06-2010' AND '20-03-2017' GROUP BY [Landing _Outcome]
```

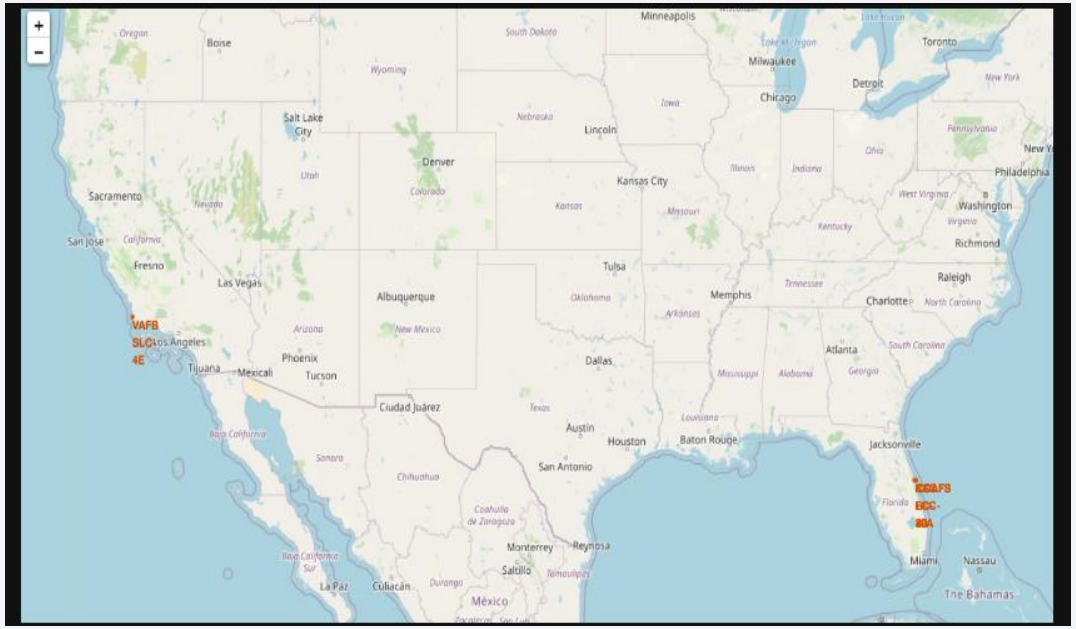
```
 * sqlite:///my_data1.db
Done.
```

Landing _Outcome	Total_Outcomes
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

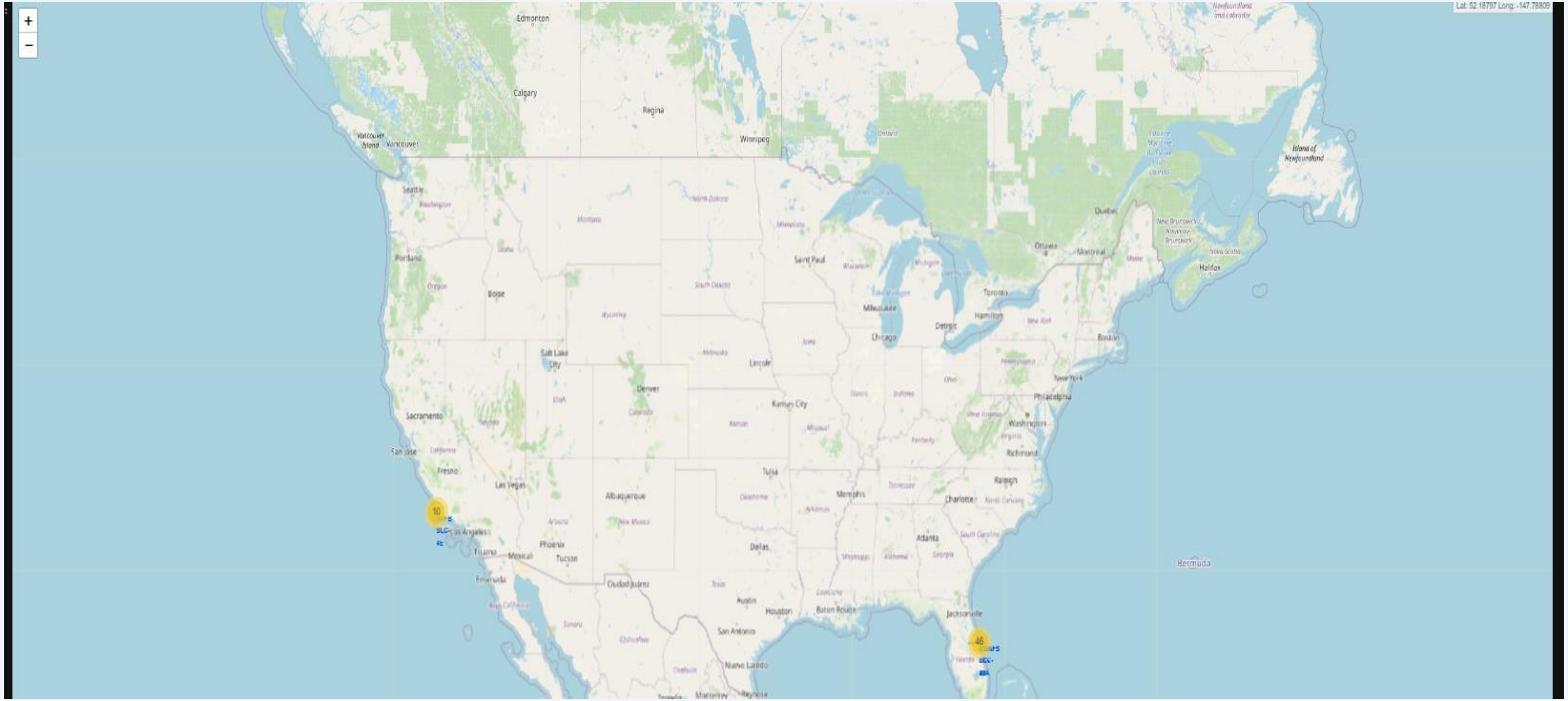
Section 3

Launch Sites Proximities Analysis

All Launch Sites.



Launch Outcomes.

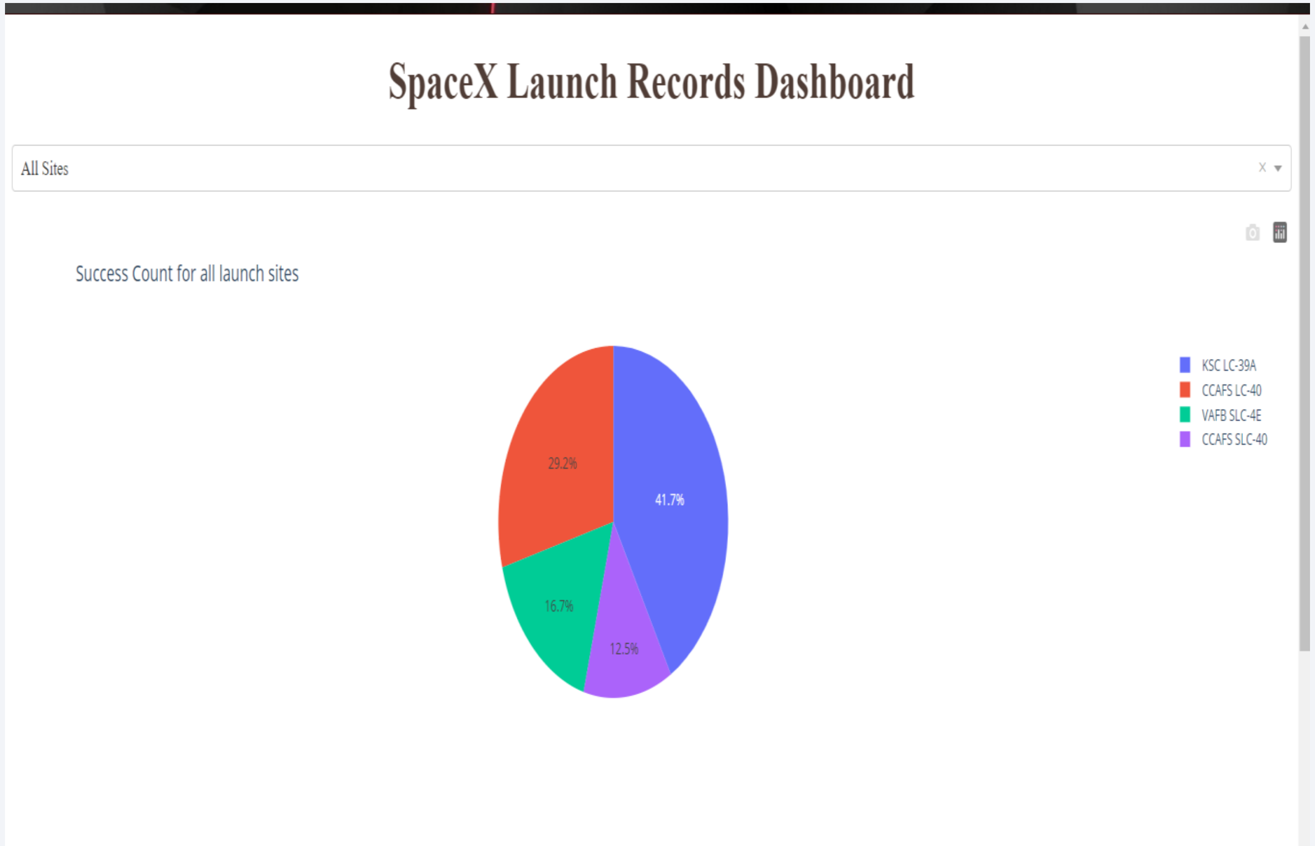


Section 4

Build a Dashboard with Plotly Dash

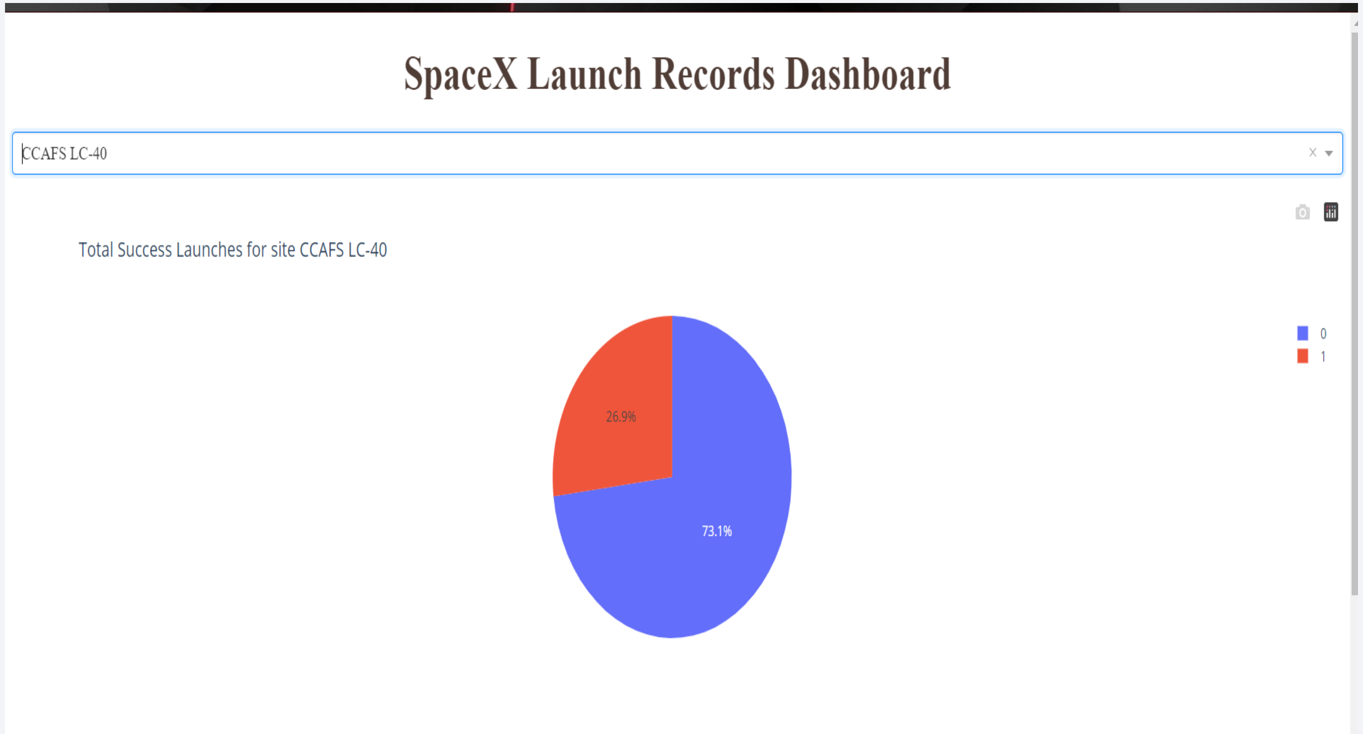
Pie Chart for all Launch Sites.

- Launch site KSC LC-39A has the highest launch success rate at 42% followed by CCAFS LC-40 at 29%, VAFB SLC-4E at 17% and lastly launch site CCAFS SLC-40 with a success rate of 13%.



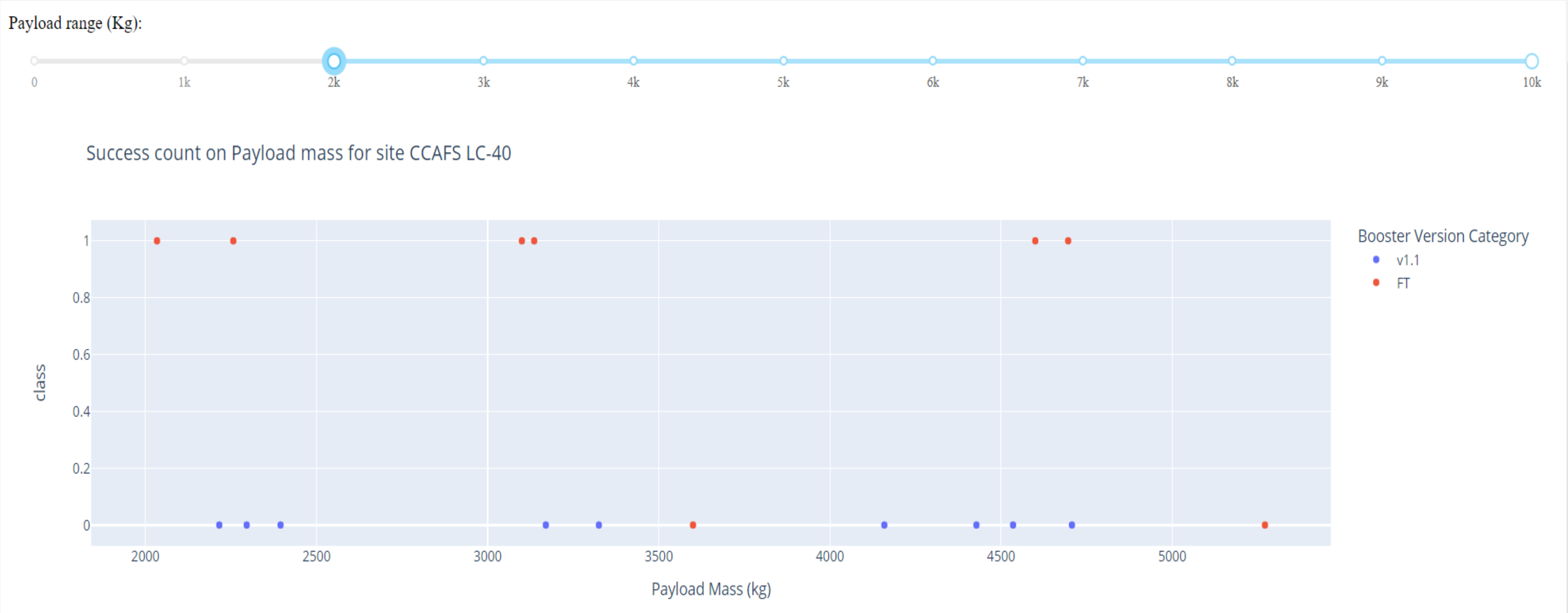
Pie Chart for the 2nd highest success ate.

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



Payload vs. Launch Outcome for all launch sites.

- For the Launch Site CCAFS LC-40 the booster version FT has the largest success rate from a payload mass of more than 2000KG.



Section 5

Predictive Analysis (Classification)

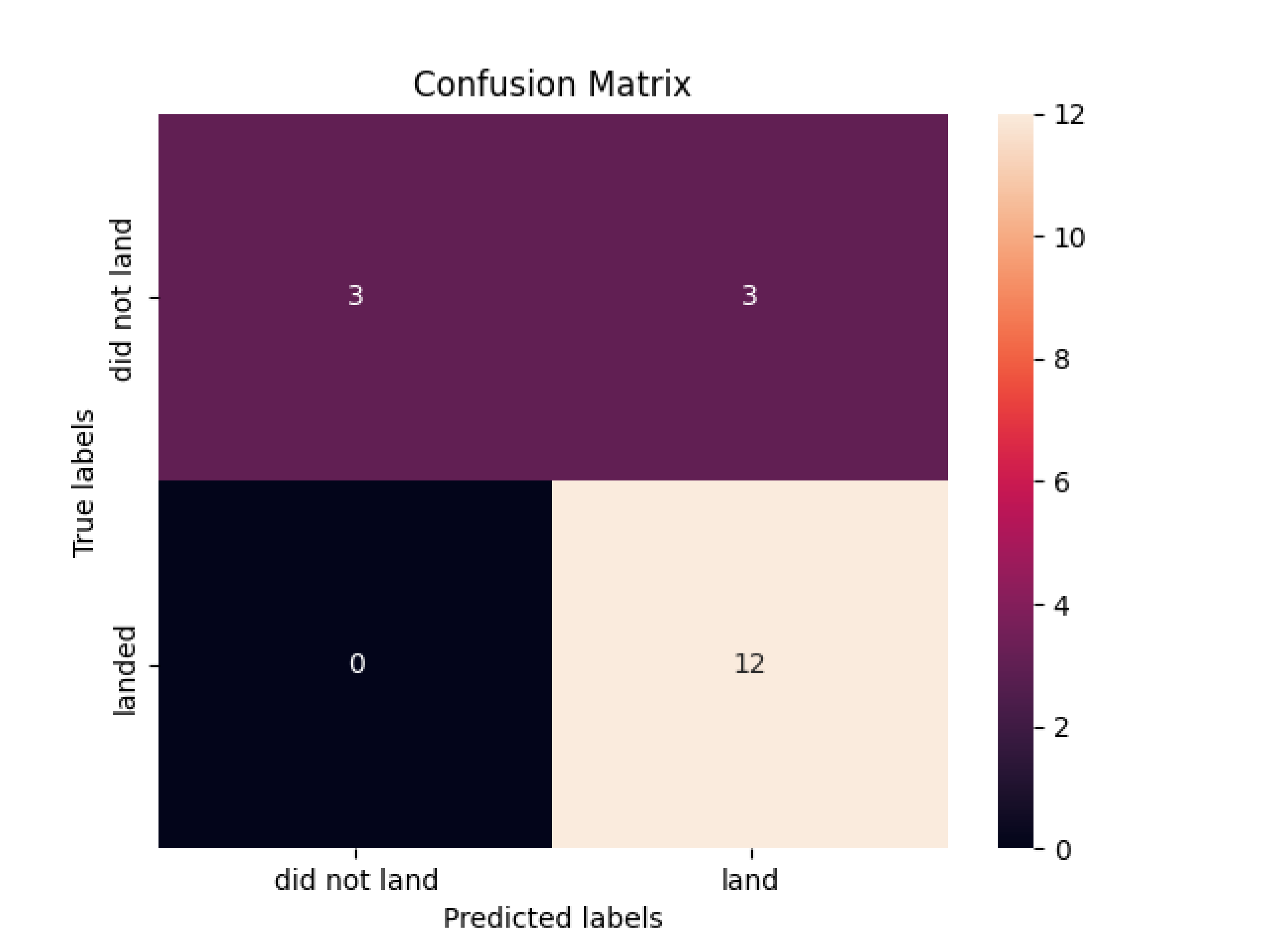
Classification Accuracy

- All the models had the same accuracy value.

	0
Model	Accuracy
Logistic Regression:	83.333333
SVM:	83.333333
Decision Tree:	83.333333
KNN:	83.333333

Confusion Matrix

All of the models had the same Confusion Matrix and were equally distinguish between the different classes. The major problem was the false positives for all the models.



Conclusions

- We see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.

- 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).

- 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.

Cont Conclusions

- 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.
- You can observe that the success rate since 2013 kept increasing till 2020.
- By now, you should obtain some preliminary insights about how each important variable would affect the success rate, we will select the features that will be used in success prediction in the future module.

Thank you!