Winning Space Race with Data Science

By Kaushik Reddy
1/10/2023

Outline

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

Executive Summary

- Methodologies

1. Data collection using SpaceX API & web scrapping
2. Data wrangling
3. Exploratory data analysis using SQL
4. EDA Dataviz using Python pandas & Matplotlib
5. Launch site analysis with Folium and Plotly Dash
6. Machine learning landing prediction

- Results

1. EDA results
2. Predictive analysis
3. Interactive visual analytics and dashboards

Introduction

Synopsis

- The space race is in full swing, there are many notable companies providing affordable space travel. Among them, perhaps the most famous is SpaceX.
- One of the features that make SpaceX stand out is their inexpensive rockets, where other companies cost upwards of 165million dollars each, SpaceX makes them for just 65million dollars, 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. Spaces X's Falcon 9 launch like regular rockets.

What are the questions that needs answers?

- Our goal is to determine the price of each launch and also determine if the first stage will land successfully.
- We will use machine learning to predict if the first stage will land successfully with public information on their 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

1. Data was initially collected using SpaceX API using a GET request
2. The data was made more consistent after it was phrased using GET request by making is a json result and turning it into a panda dataframe.
3. Web scraping was done to retrive historical launch data from wikipedia and stored in HTML file. Using Beautifulsoup and libraries the data was phrashed from wiki HTML and stored in a pandas dataframe.

Data Collection – SpaceX API

1. Using GET request on SpaceX API we collect data and then to make the results more consistent we make it a JSON result and store it in a pandas dataframe

2. GitHub link of completed call. https://github.com/KaushikTR/DataScience-capstone-project/blob/main/1.SpaceX-Data-Collection.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:

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

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

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

Out[10]: 200

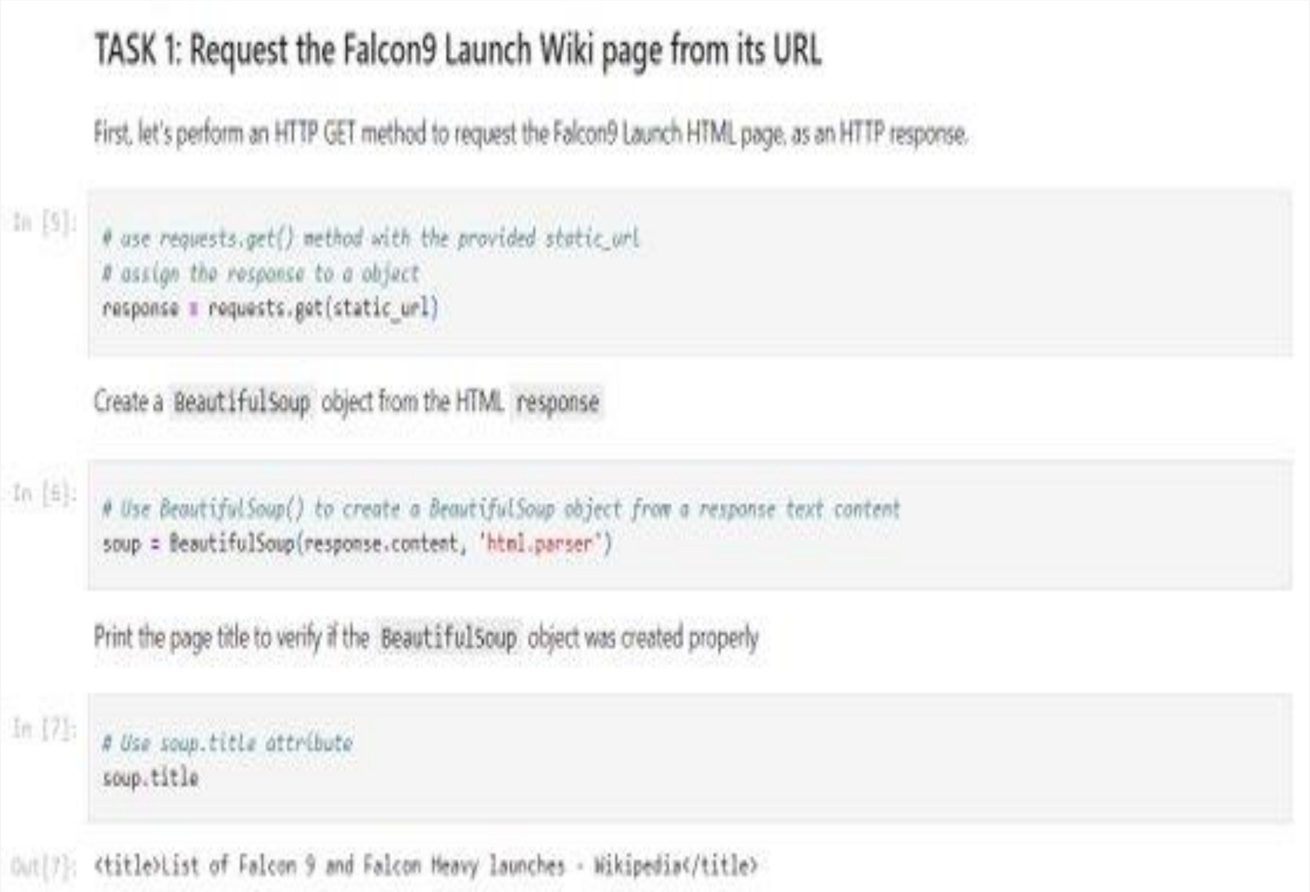
Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

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

Data Collection - Scraping

1. Using web scraping to collect Falcon-9 Historical data from wikipedia via Beautifulsoup and request libraries. Then creating a dataframe after parshing the HTML result.

2. GitHub link of completed call. https://github.com/KaushikTR/DataScience-capstone-project/blob/main/1.SpaceX.Web-Scraping.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.

In [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`

In [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

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

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

Data Wrangling

1. After a Pandas dataframe was created upon obtaining data, it was filtered to show the 'BoosterVersion' column to show only the launches. The missing values in 'LandingPad' and 'PayloadMass' columns were dealt with. Missing values in 'PayloadMass' were replaced by mean values of the column.
2. An EDA analysis was performed to find patterns and conclude the label which would be used in training supervised models.
3. Link to completed results: https://github.com/KaushikTR/DataScience-capstone-project/blob/main/1.SpaceX-Data-Wrangling.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`:

```python
# landing_class = 0 if bad_outcome

# landing_class = 1 otherwise
landing_class = []
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(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

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

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

EDA with Data Visualization

- Exploratory data analysis and feature engineering performed using Matplotlib and pandas

Scatter plots used to visualize the relationship between

- Launch site and Payload mass
- Flight number and launch site
- Flight number and orbit type
- Payload and orbit type

Bar charts used to visualize success rate of each orbit type

Line plot used to see yearly trend

Github link to notebook https://github.com/KaushikTR/DataScience-capstone-project/blob/main/2.SpaceX-EDA-DataViz.ipynb

EDA with SQL

EDA Queries performed in SQL as as follows

- 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 achieved
- 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 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.

Completed notebook GitHub link: https://github.com/KaushikTR/DataScience-capstone-project/blob/main/2.SpaceX-EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- A folium map was created with marked launch sites, map objects such as markers, circles, lines indicating success and failure of each launch site were created.
- Launch set outcome was created, 1= success, 0= failure
- Folium map notebook Github link: https://github.com/KaushikTR/DataScience-capstone-project/blob/main/3-SpaceX-Launchsite-Locations.ipynb

Build a Dashboard with Plotly Dash

Interactive dashboard application created with plotly contsaining:

- Range slider to select payload
- A callback function to display the payload success scatter plot
- A launchsite dropdown input component
- A callback function to display success piechart depending on the selected site.
- GitHub link of dash app: https://github.com/KaushikTR/DataScience-capstone-project/blob/main/3.SpaceX-Dash-app.py

Predictive Analysis (Classification)

Summary of process

- Start of by creating NumPy array from the column 'Class' in data and output set to be in pandas series, assign it to variable Y.
- Standardize data in X and transform the data into variable X
- Using the function train_test_split to split the data X and Y into training and test data, parameter test_size to 0.2 and random_state to 2, The training data and test data is assigned to the following labels, `X_train, X_test, Y_train, Y_test.`

`To determine the best model to fit test data from` SVM, Classification Trees, k nearest neighbors and Logistic Regression

Predictive Analysis (Classification)

To find the model out of SVM, Classification Trees, k nearest neighbors and Logistic Regression that would best fit the test data we:

- Create an object for use in each of the algorithms (eg, logistic regression object) and GridSearchCV object, then assign them a set of parameters
- GridSearchCV object has a cv = 10 for each evaluation method, training data was fit into GridSearch object for each method to determine hyperparameter.
- We output GridSearchCV object for each of the models after fitting the training set. Then display the best parameters using the data attribute best_params_ and the accuracy on the validation data using the data attribute best_score_ respectively.
- We conclude by using 'Score' to calculate the accuracy of each model with test data and plot a confusion matrix for each.
- GitHub link of completed notebook: https://github.com/KaushikTR/DataScience-capstone-project/blob/main/4-SpaceX-Machine-Learning-Prediction.ipynb

Results

Exploratory data analysis results

- Launch success rate increases over time
- Higher success rate for higher orbits

Interactive analytics demo in screenshots

- Higher success rate for higher payload mass
- Low success rate for booster versions v1.0, v1.1, high success rate for FT, B4, B5
- Higher success rate for Kennedy Space center and recent starts at Cape Canaveral

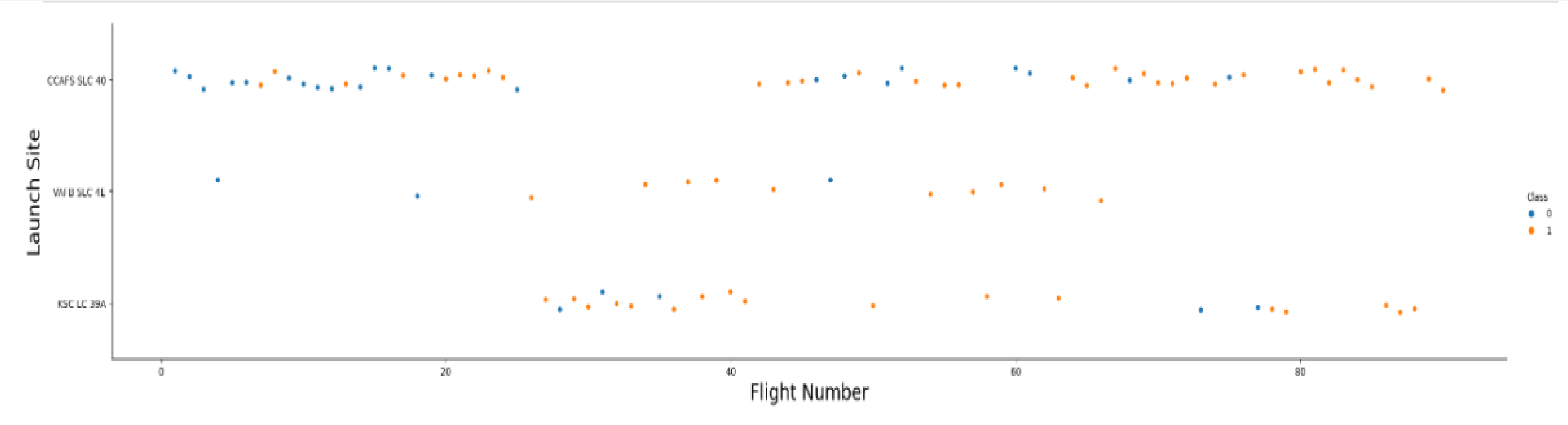
Predictive analysis results

- Best prediction results with Logistic Regression and SVM

Section 2

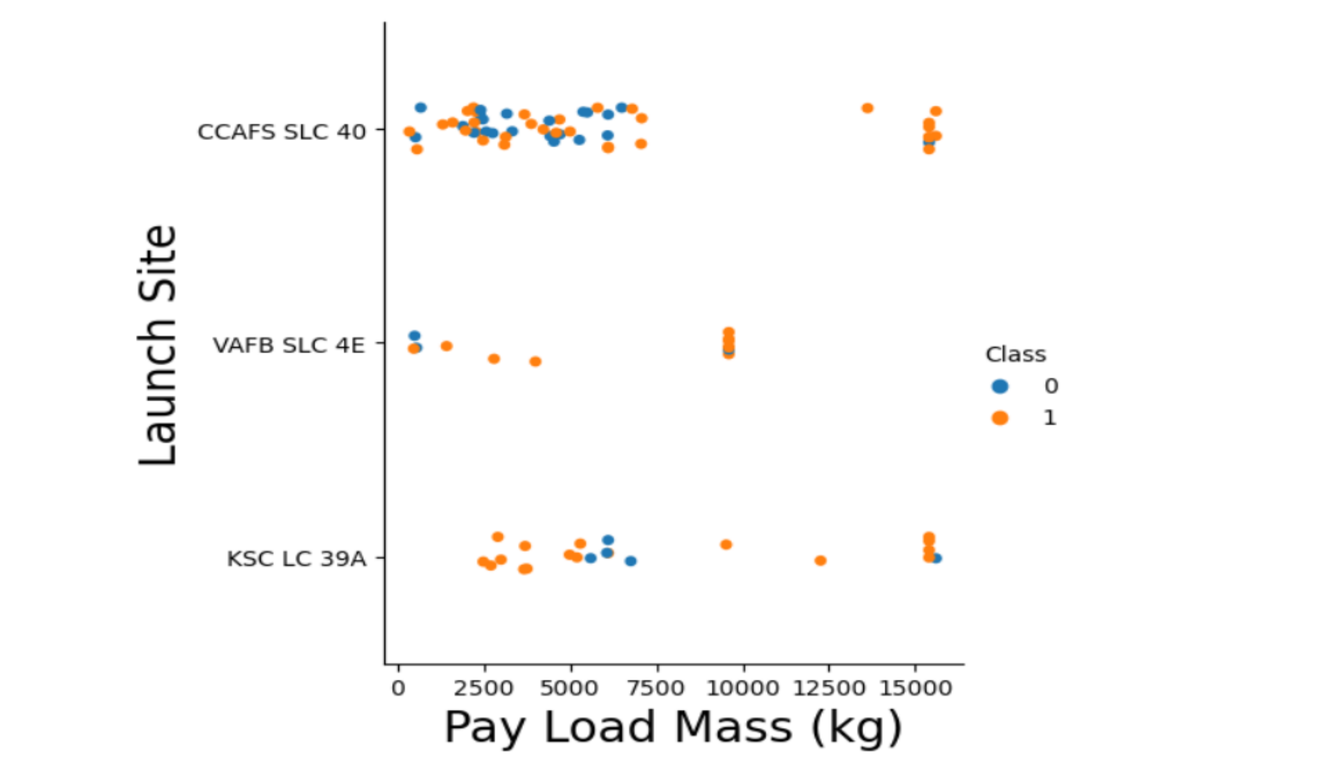
Insights drawn from EDA

Flight Number vs. Launch Site



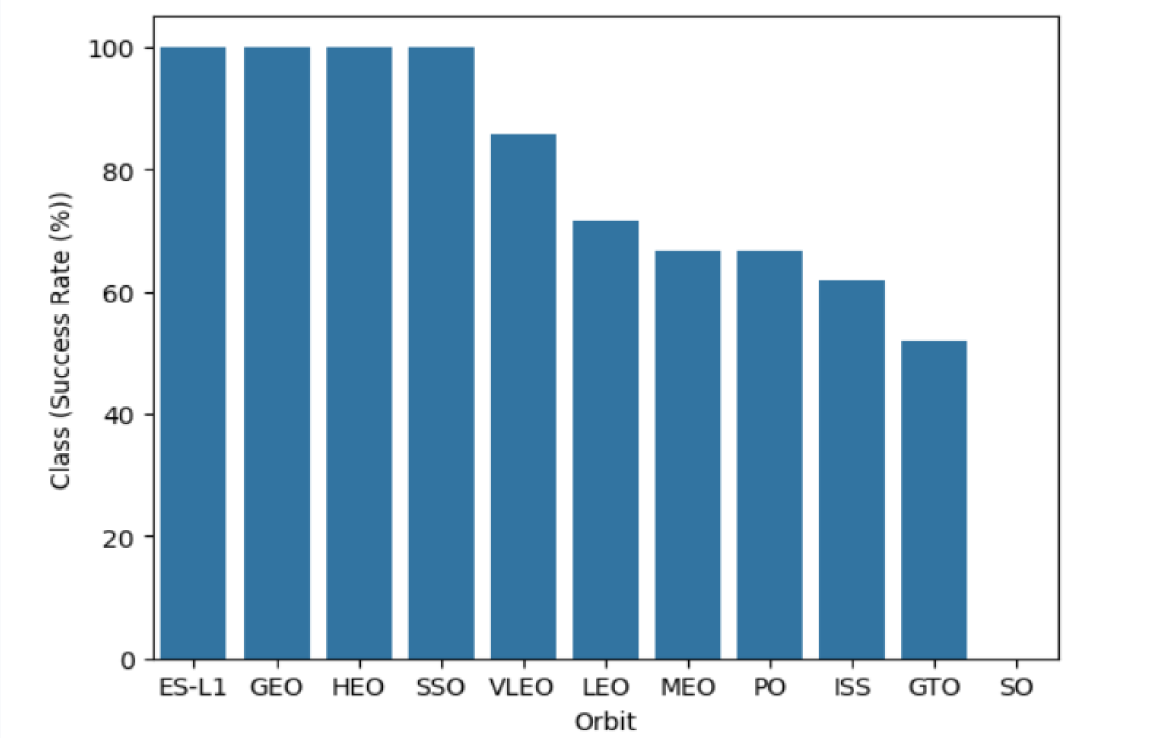
From the scatter plot we can see that the number of flights increases as launch site increases and success rate increases aswell.

Payload vs. Launch Site



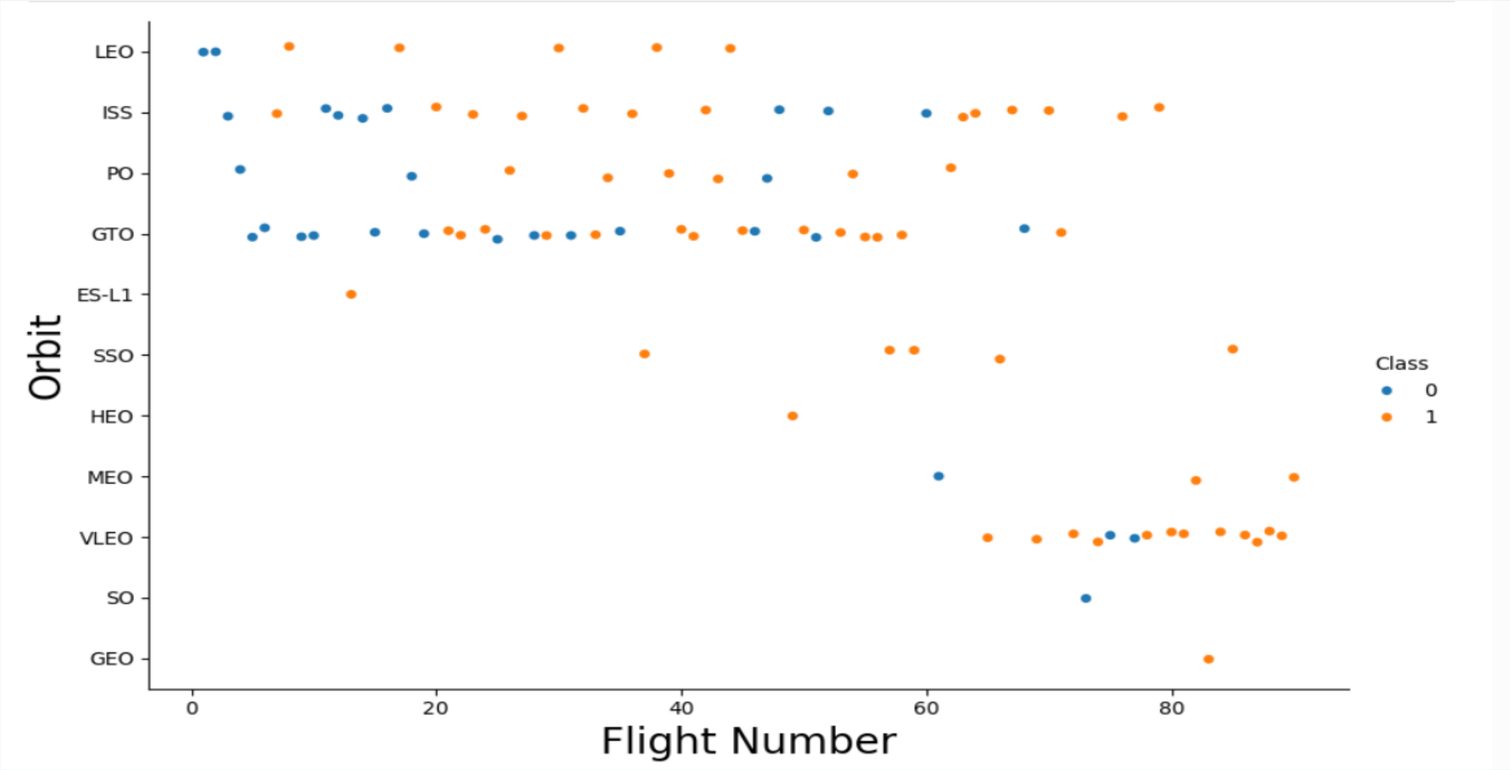
We can see that the number of flights decreases as payload mass increases and VAFB SLC 4E stops flights all together after mass exceeds 10,000kg

Success Rate vs. Orbit Type



As seen above ES-L1, GEO, HEO, SSO all have 100% success rate

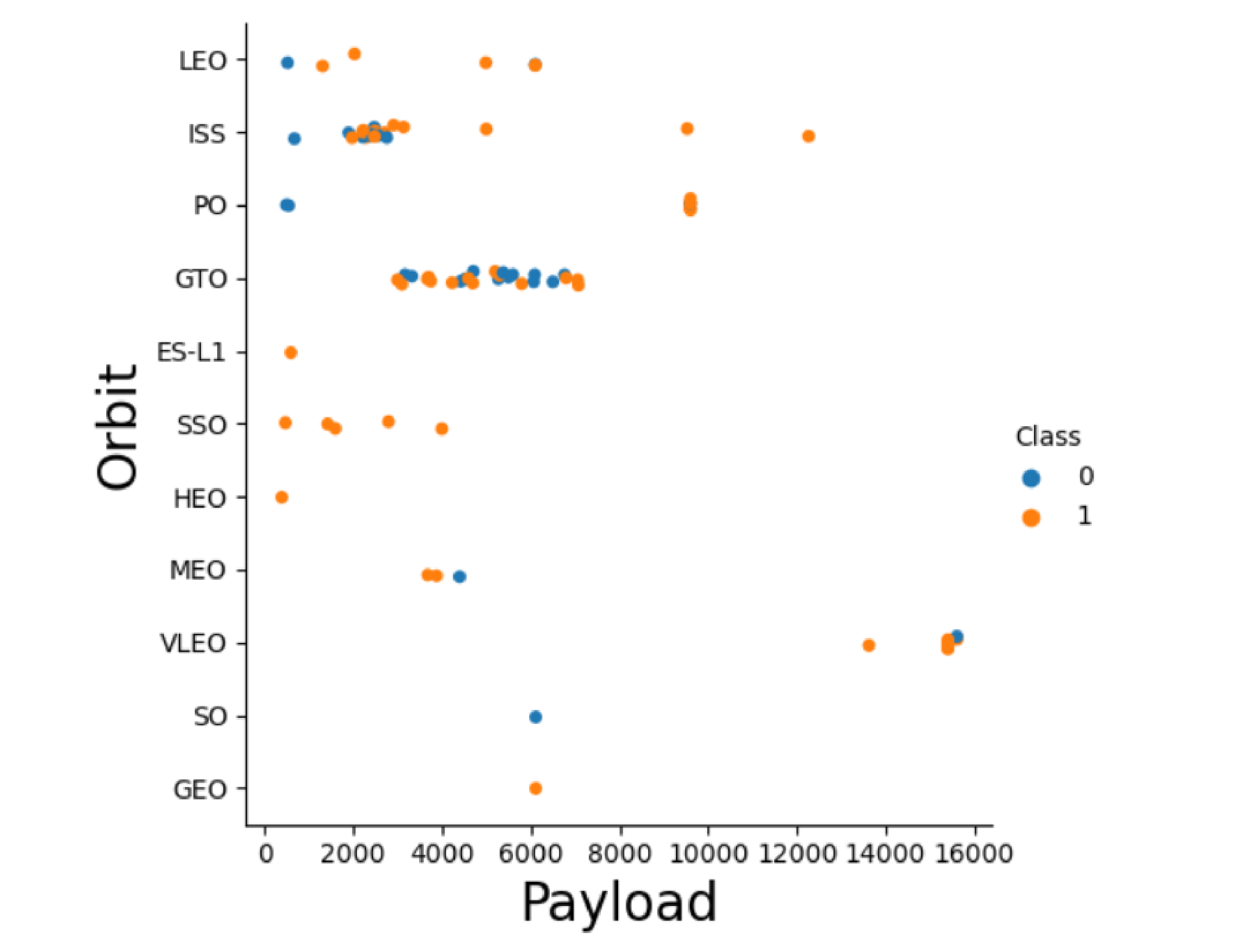
Flight Number vs. Orbit Type



Success rate seems to related to number of flights launched in LEO, ISS, PO and GTO orbits.

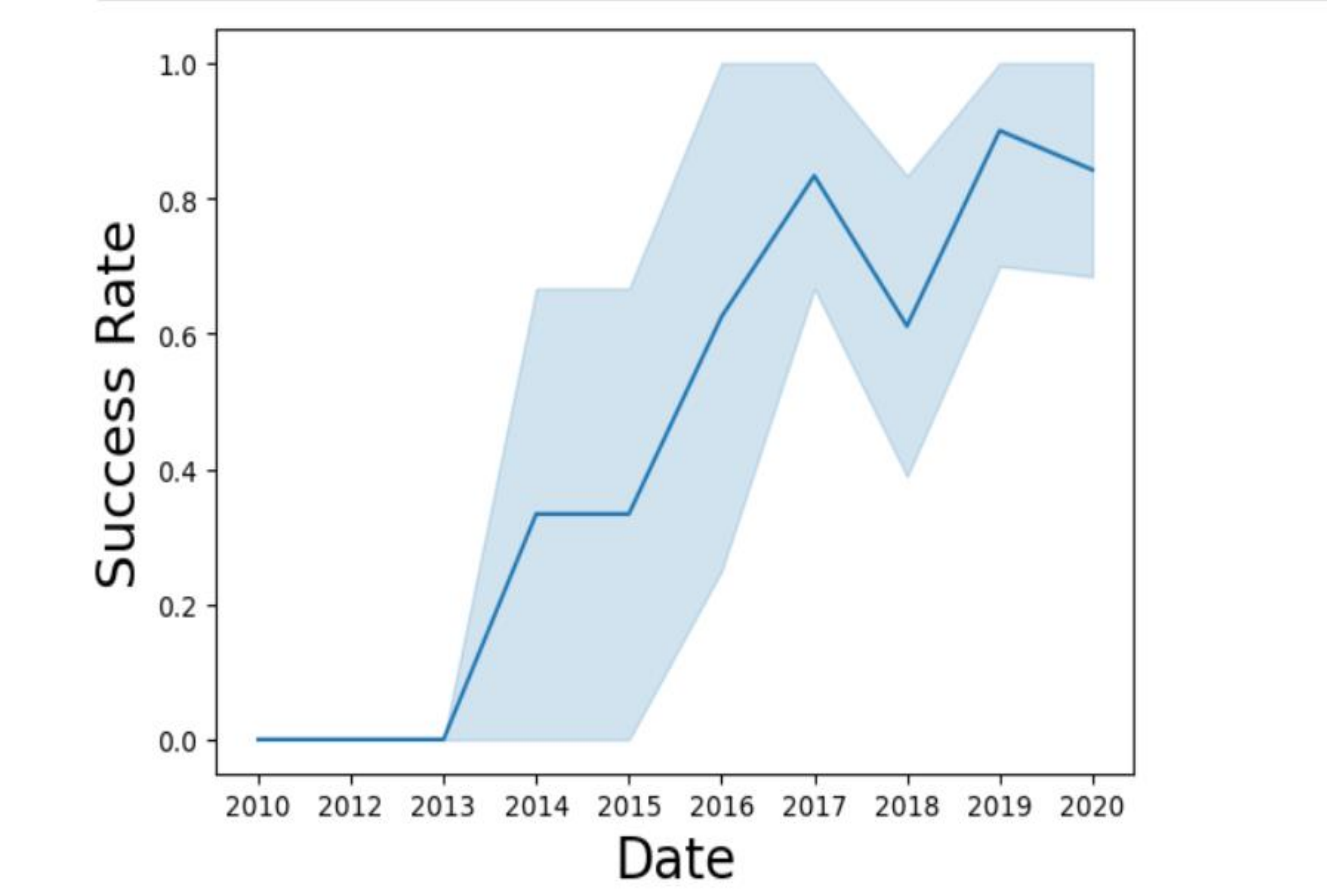
Payload vs. Orbit Type

- VLEO seems to be successful at carrying heavy payload
- GTO and ISS
  have relatively good success rates at low to medium payloads



Launch Success Yearly Trend

- It is clear the as time passes the success rate has a positive increase with a dip in 2018 but rises again in 2019 and slight dip in 2020.



All Launch Site Names

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

In [11]:
```
%sql select Unique(LAUNCH_SITE) from SPACEX;
```

```
 * ibm_db_sa://qwq42142:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
   sqlite:///my_data1.db
Done.
```

Out[11]:

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

1. KSC = Kennedy Space Center
2. CCA = Cape Canaveral Launch Center
3. VAFB = Vandenburg Air Force Base

Launch Site Names Begin with 'CCA'

1) Used ‘LIKE’ command with ‘%’ wildcard in ‘WHERE’ clause to select and display a table of all records where launch sites begin with the 'CCA

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

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

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

Out[9]:

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

Total Payload Mass

Task 3

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

In [10]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

Out[10]:

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

Used the 'SUM()' function to dispaly the total sum of payload mass.

Average Payload Mass by F9 v1.1

Task 4

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

In [12]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version I
```

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

Out[12]:

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

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

First Successful Ground Landing Date

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

Hint:Use min function

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

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

MIN(DATE)
01-05-2017

Used 'Min()' function to find out the oldest date where a successful landing happened.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
# %sql SELECT * FROM 'SPACEXTBL'
```

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

```
 * 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

Used 'SELECT DISTINCT' to calculate the names of boosters that had success in drone ship and payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

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

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

Out[16]:

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

Used the functions 'Count()' along with with 'Group By' to calculate the amount of successful and failure missions

Boosters Carried Maximum Payload

Task 8

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

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

```
 * 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

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, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

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

```
 * 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)

Using the function 'Subsrt()' in the select column to retrive the year and month from date column where subsrt (Date, 7,4) ='2015' for year and landing outcome was failure.

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

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDE
```

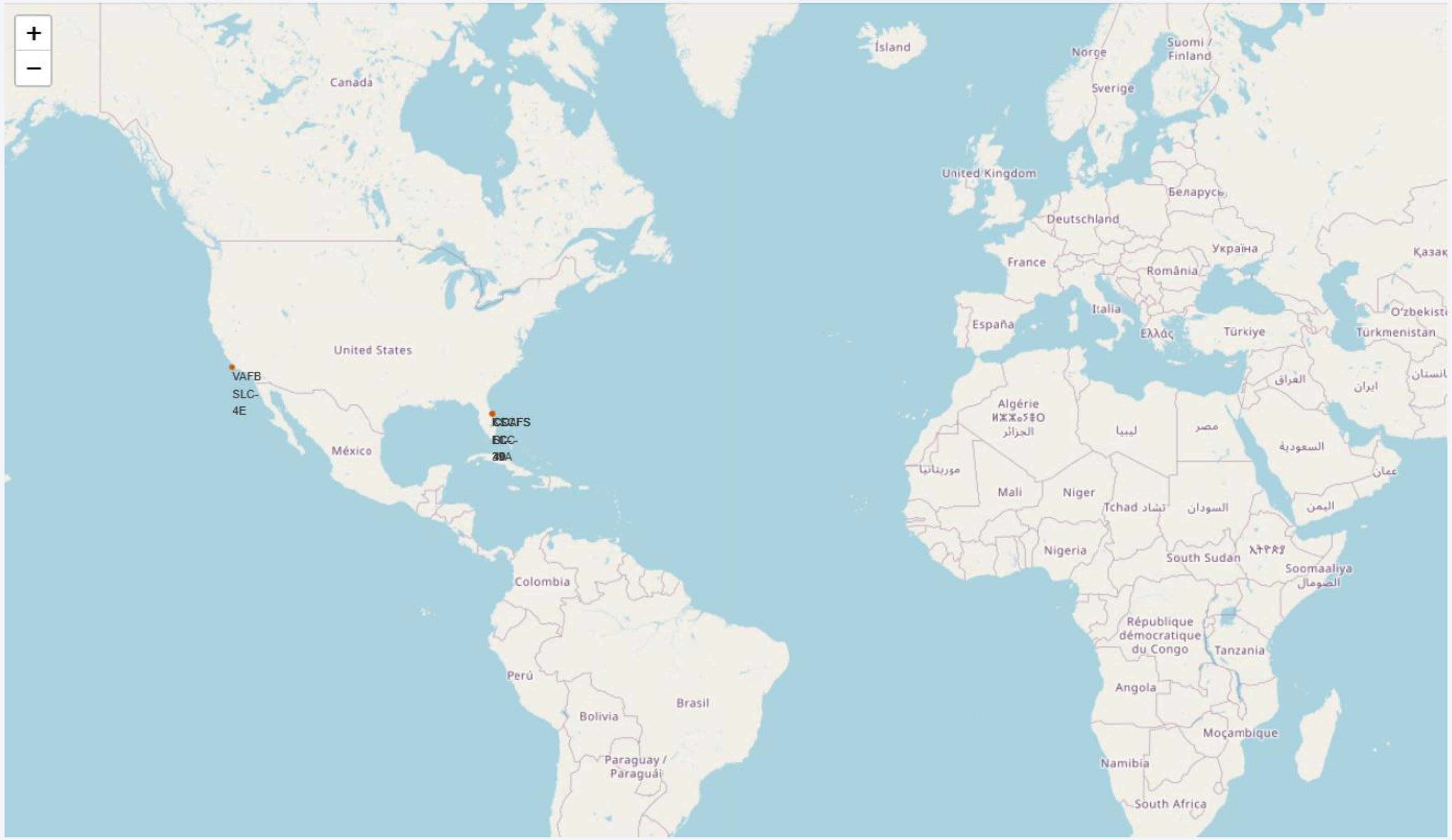
```
* sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
19-02-2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-10-2020	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600	LEO	SpaceX	Success	Success
18-08-2020	14:31:00	F9 B5 B1049.6	CCAFS SLC-40	Starlink 10 v1.0, SkySat-19, -20, -21, SAOCOM 1B	15440	LEO	SpaceX, Planet Labs, PlanetIQ	Success	Success
18-07-2016	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-04-2018	22:51:00	F9 B4 B1045.1	CCAFS SLC-40	Transiting Exoplanet Survey Satellite (TESS)	362	HEO	NASA (LSP)	Success	Success (drone ship)
17-12-2019	00:10:00	F9 B5 B1056.3	CCAFS SLC-40	JCSat-18 / Kacific 1, Starlink 2 v1.0	6956	GTO	Sky Perfect JSAT, Kacific 1	Success	Success
16-11-2020	00:27:00	F9 B5B1061.1	KSC LC-39A	Crew-1, Sentinel-6 Michael Freilich	12500	LEO (ISS)	NASA (CCP)	Success	Success
15-12-2017	15:36:00	F9 FT B1035.2	CCAFS SLC-40	SpaceX CRS-13	2205	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
15-11-2018	20:46:00	F9 B5 B1047.2	KSC LC-39A	Es hail 2	5300	GTO	Es hailSat	Success	Success
14-08-2017	16:31:00	F9 B4 B1039.1	KSC LC-39A	SpaceX CRS-12	3310	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)

Section 3

Launch Sites Proximities Analysis

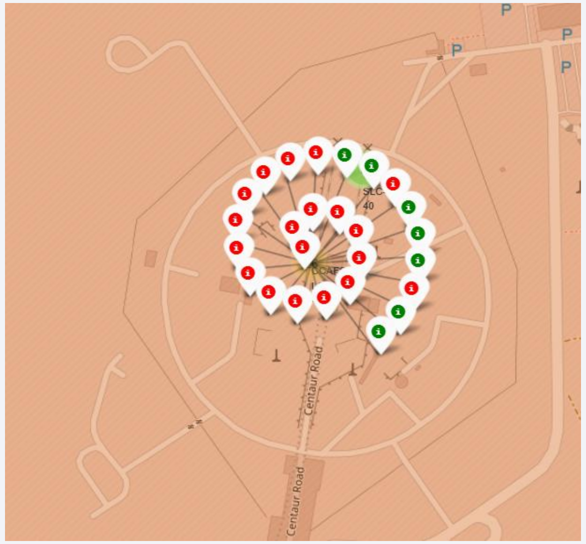
Launch site markers on global map



Launch sites are close to the equator and close to the coast lines of the US.

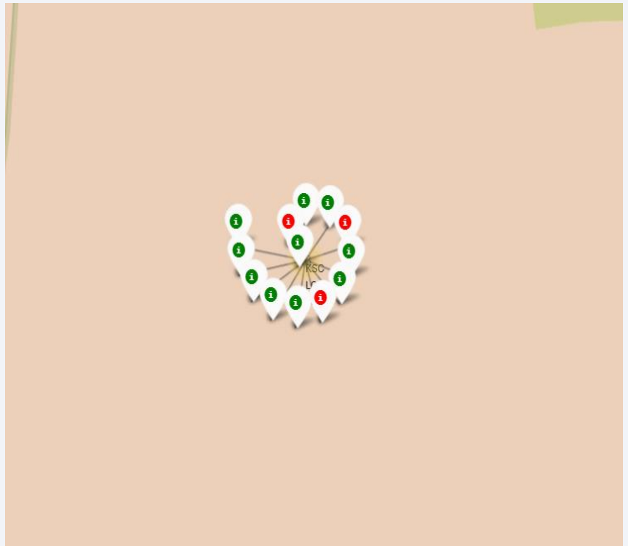
Launch site success/failure rates - Florida

CCAFS LC-40



CCAFS SLC-40

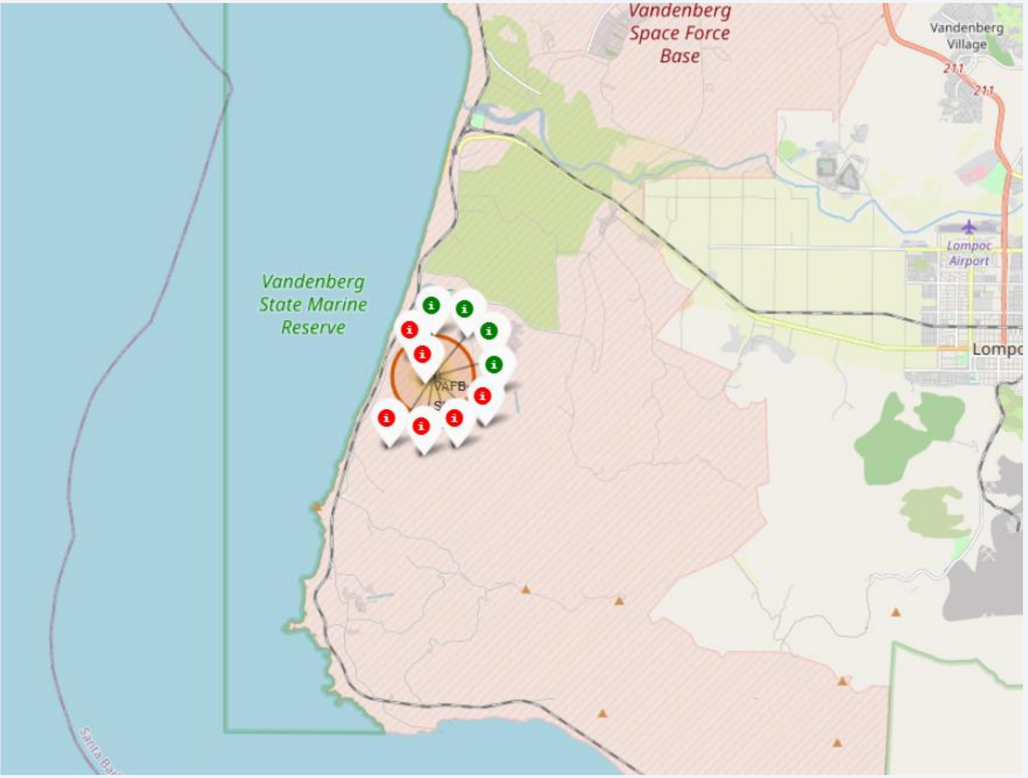
KSC LC-39A



In the Eastern coast (Florida) Launch site KSC LC-39A has relatively high success rates compared to CCAFS SLC-40 & CCAFS LC-40.

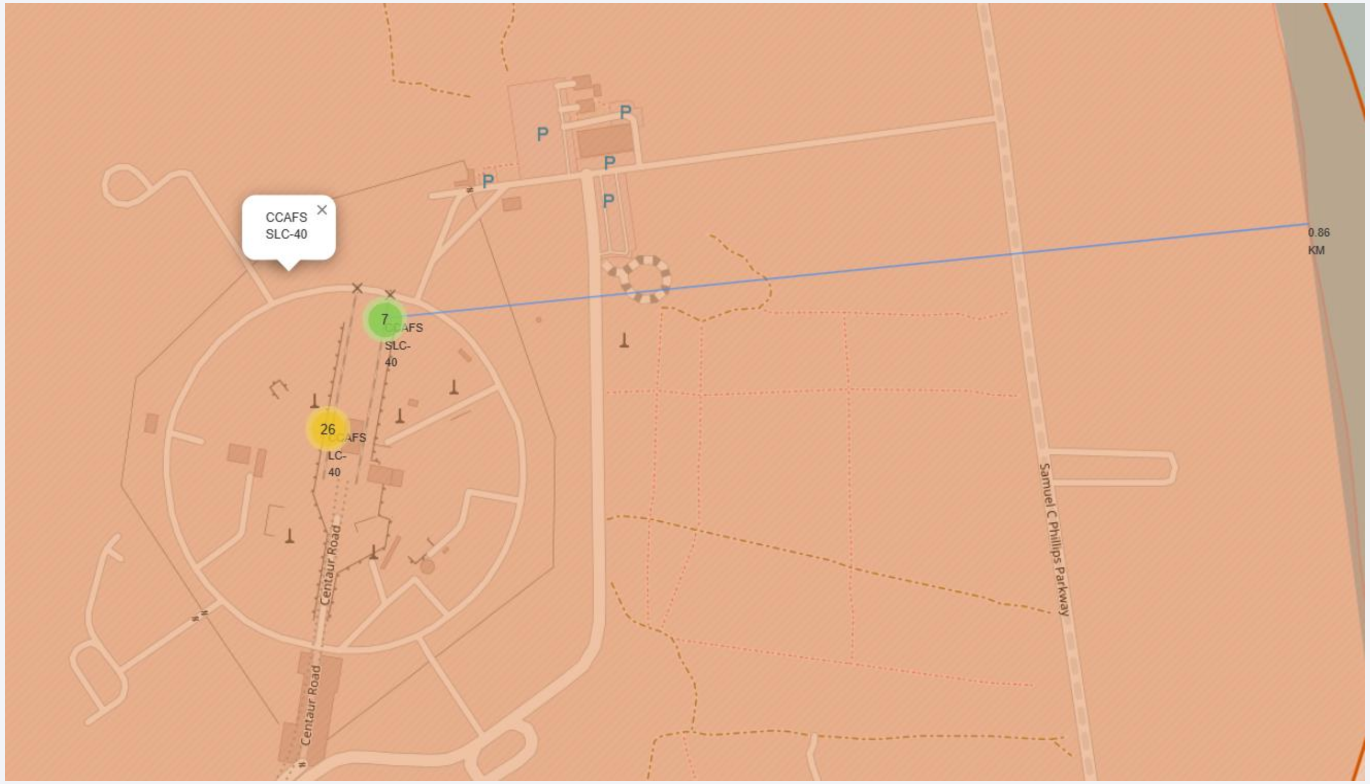
Launch site success/failure rates - Westcoast

VAFB SLC-4E



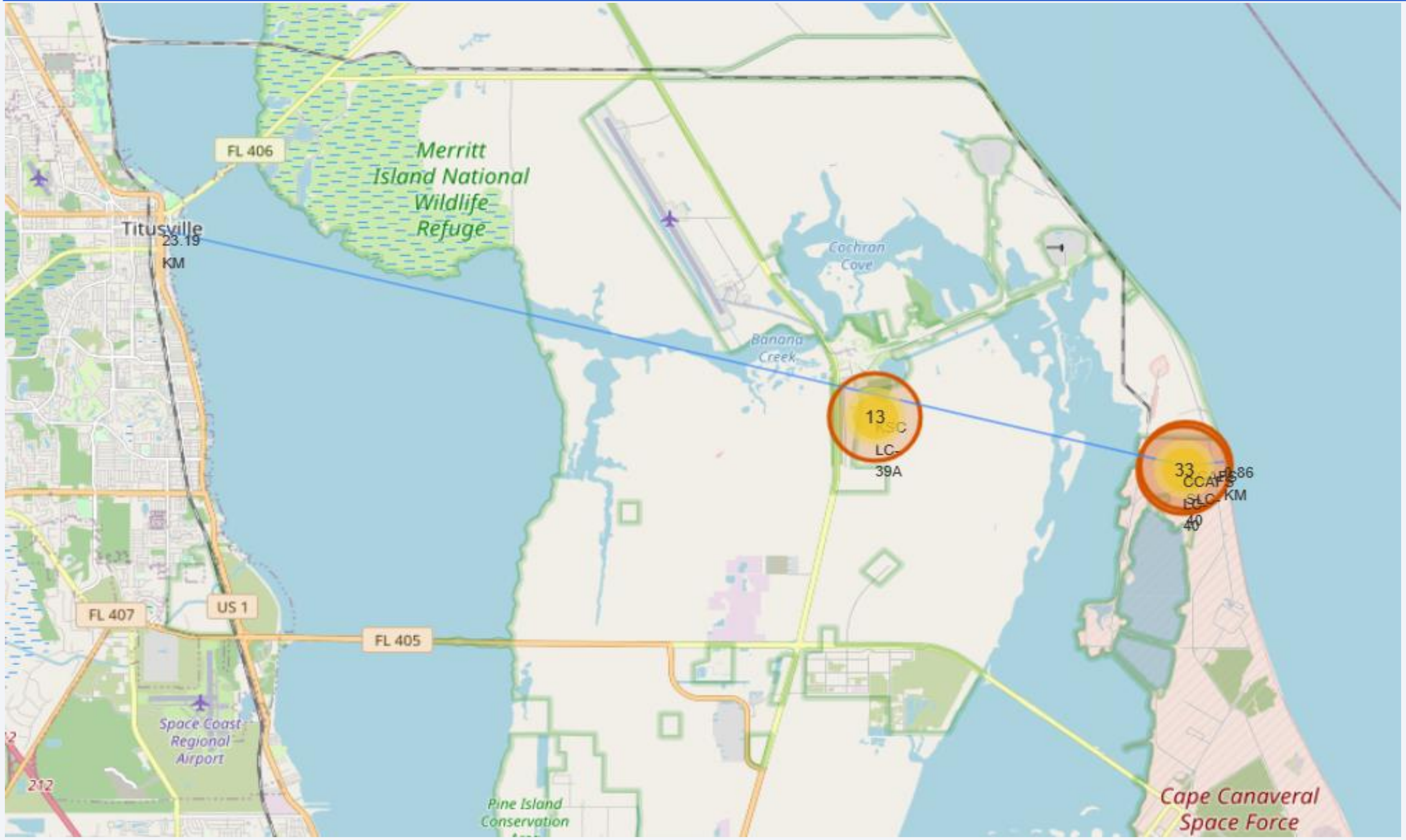
Launch site in California/westcoast has lower success rate as compared to KSC LC-39A .

Launch site distance to its proximities



Launch site CCAFS SLC-40 is about 0.86km from the coastline

Launch site distance to its proximities

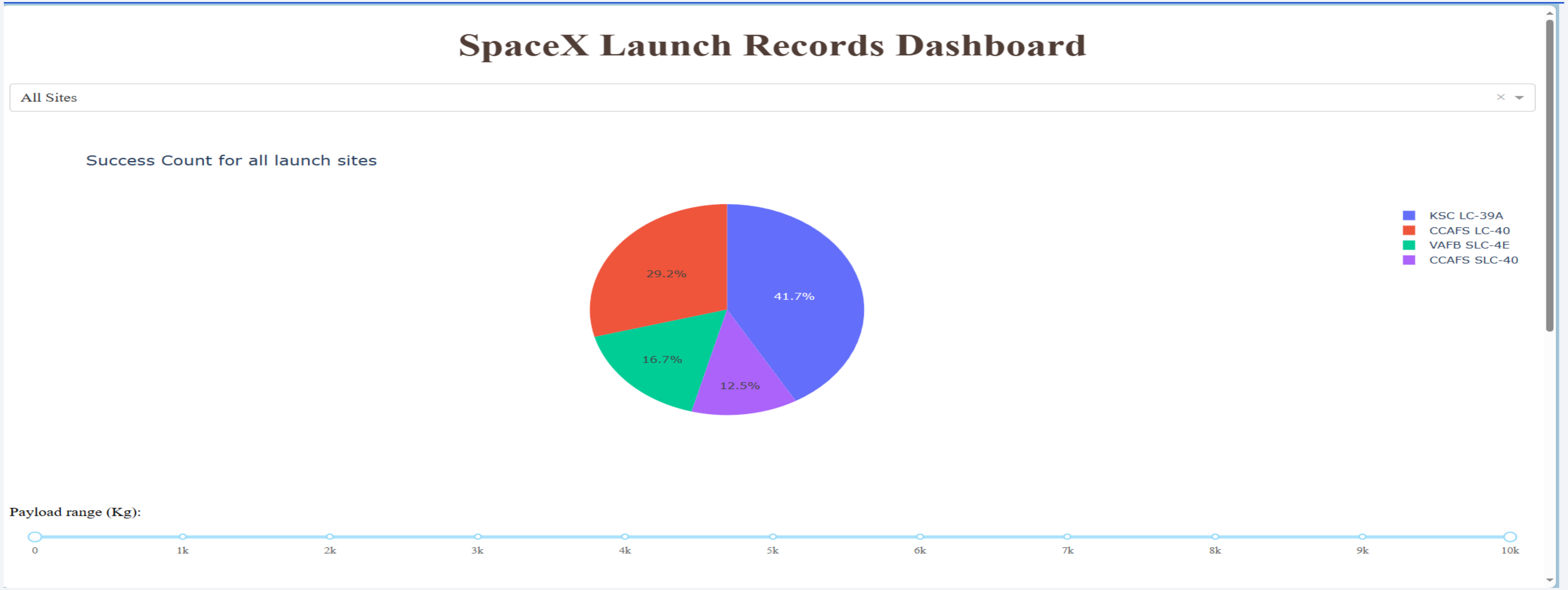


Closest highway to launch site CCAFS SLC-40 is about 23.19km

Section 4

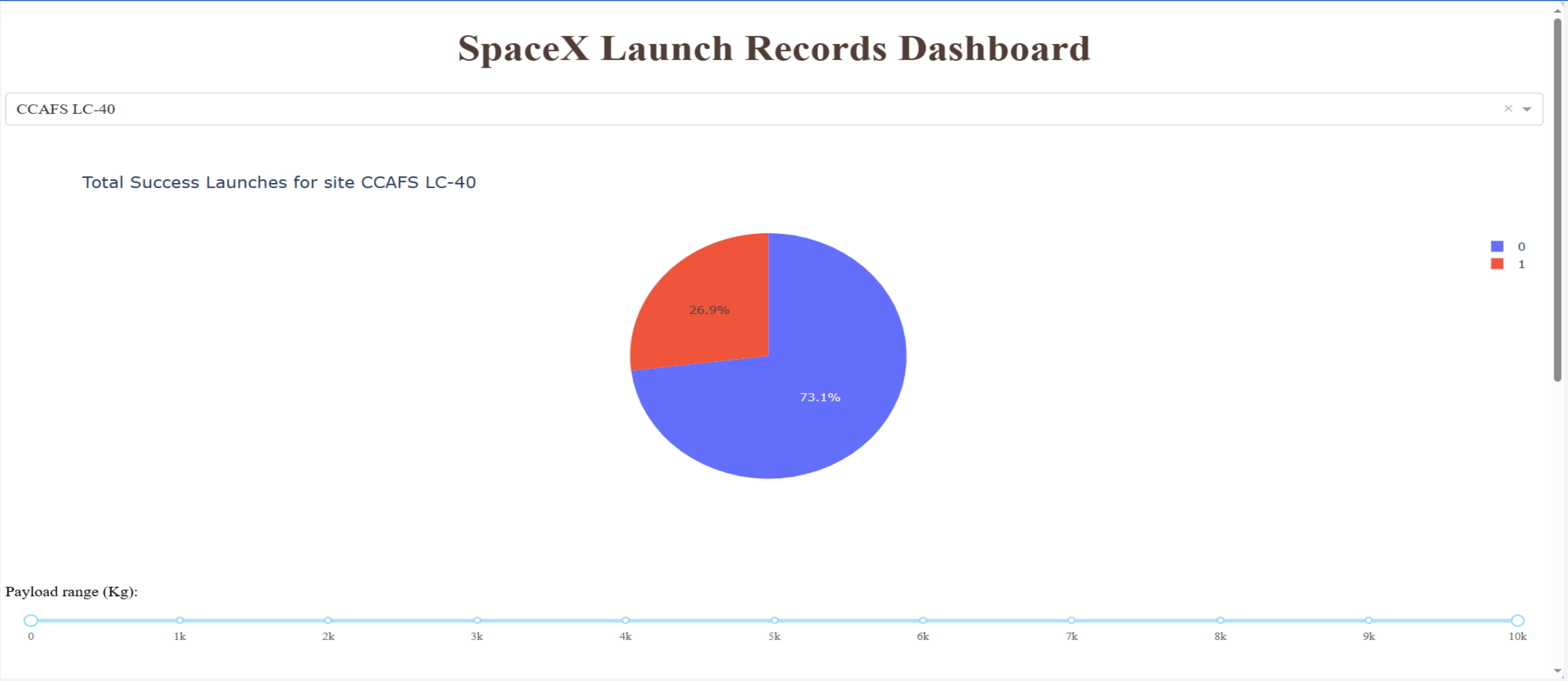
Build a Dashboard with Plotly Dash

Launch success piechart- All sites



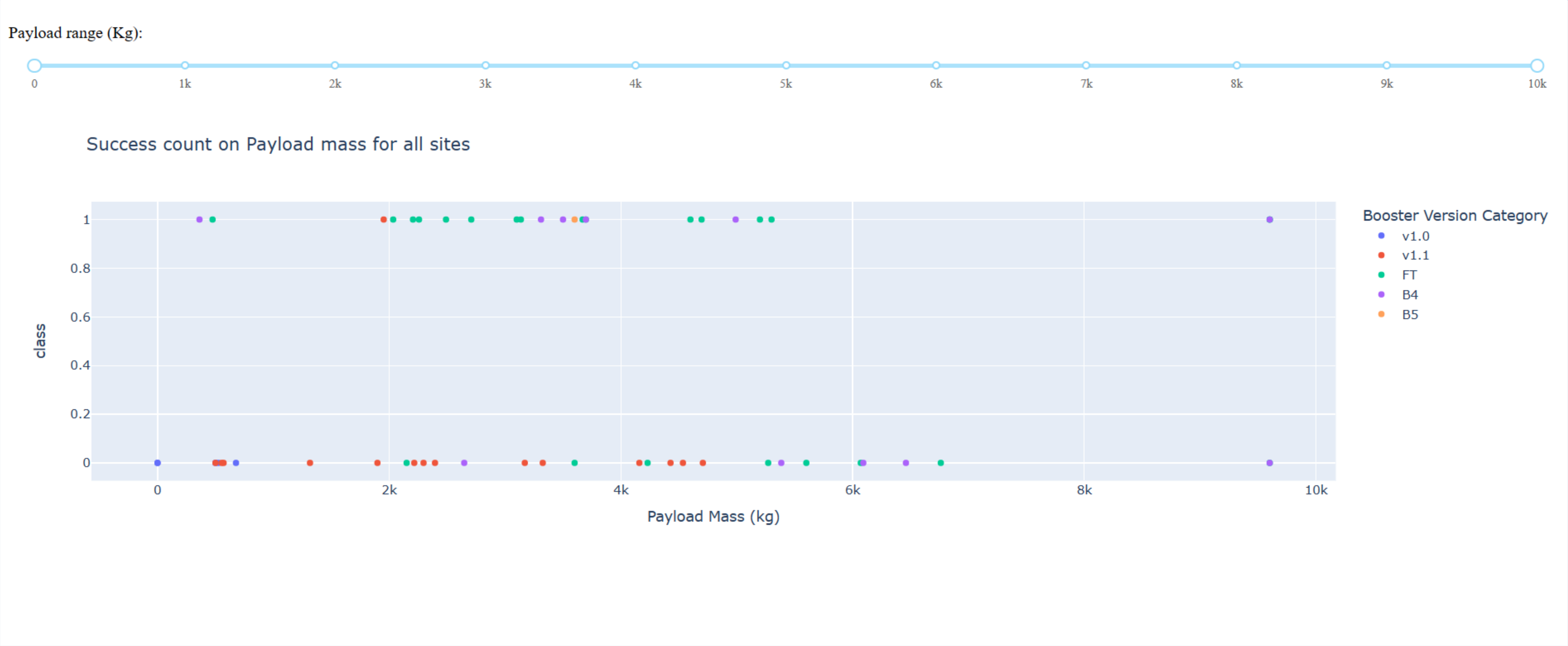
Launch site KSC LC-39A has the highest success rate at 41.7%, followed by CCAFS LC-40 at 29.2. The other two sites VAFB SLC-4E and CCAFS SLC-40 have comparatively lower success at 16.7% and 12.5% respectively

Launch site success piechart- CCAFS LC-40



CCAFS LC-40 had the second highest success ratio with 73.1% with 26.9% failure

Payload vs Launch outcome – All sites



Booster version FT has high success for payload between 2000kg and 6000kg

Section 5

Predictive Analysis (Classification)

Classification Accuracy

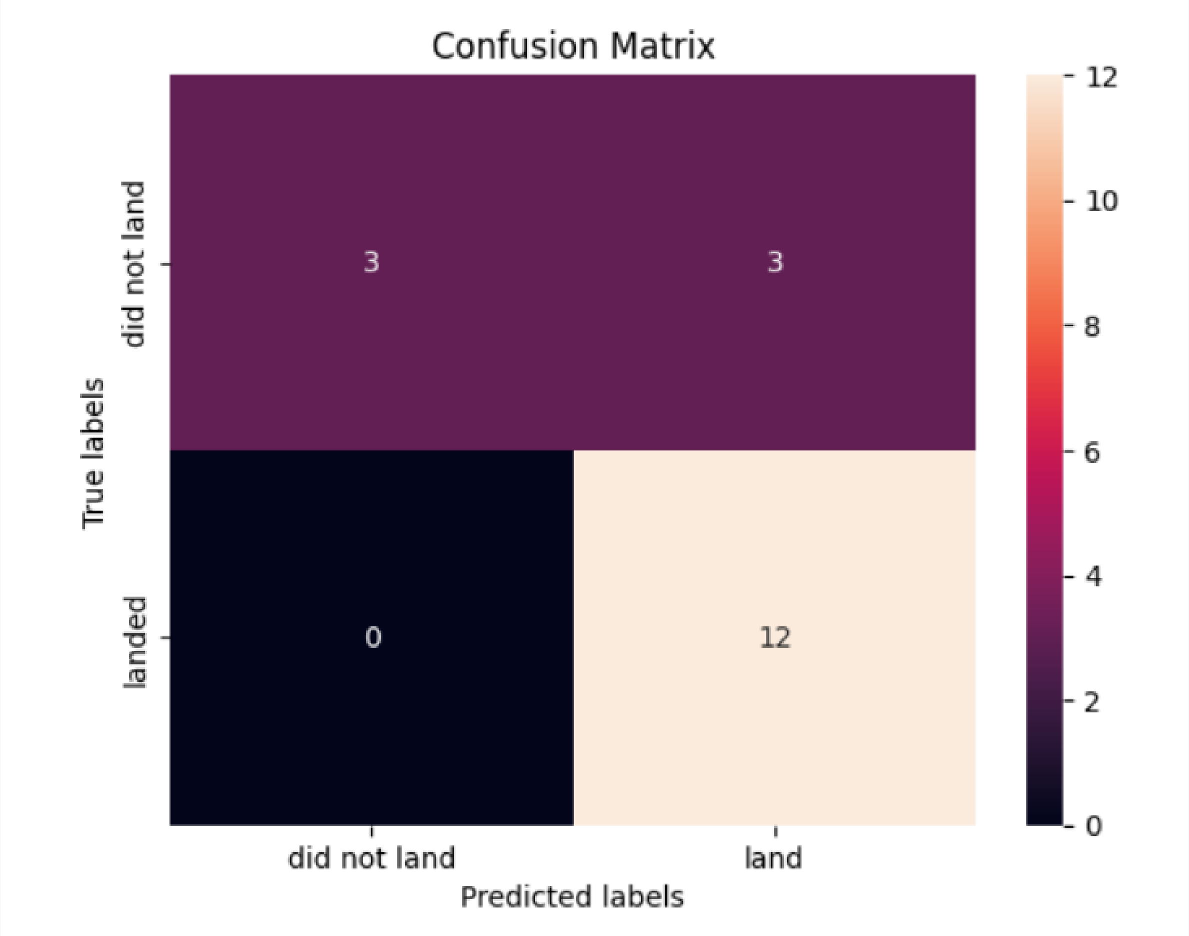
- All methods had about the same accuracy, except Decision tree with 0.8888 as compared to all others of 0.8333

Out[34]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.888889
KNN	0.833333

Confusion Matrix

- Best performing confusion matrix was this, and were able to distinguish between models. Biggest problems were false positives for all models.



Conclusions

1) The location of launch sites are close to coastal areas and away from civillan towns, both of which are precautions incase of an accident.
2) Success rate of launch sites increases with number of flights, as evidenced in EDA scatter plot where VAFB SLC 4E had a success rate of 100% after it's 50th launch, followed by KSC LC 39A and CCAFS SLC 40 which had 100% success rate after their 80th launch
3) Increases in payload results in decrease in success rate of launches, as evidenced by EDA scatter plot of Payload vs Launch site. VAFB SLC 4E stops flights all together after mass exceeds 10,000kg
4) Orbit type has a big influence on the success rate with ES-L1, GEO, HEO, SSO all having 100% success rate
5) Succes rate has a steady increase since 2013 until 2020

Appendix

1) SpaceX dataset used https://github.com/KaushikTR/DataScience-capstone-project/blob/main/Spacex.csv

Thank you!