IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Chandrasekar Chinnathambi
10/08/2024

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

Introduction

- Project background and context

  SpaceX has gained worldwide attention for a series of historic milestones. It is the only private company ever to return a spacecraft from low-earth orbit, which it first accomplished in December 2010.

  SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars whereas other providers cost upward of 165 million dollars each, much of the savings is because Space X 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 capstone, we will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its 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.

Data was first collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API. This was done by first defining a series helper functions that would help in the use of the API to extract information using identification numbers in the launch data and then requesting rocket launch data from the SpaceX API url.

Finally, to make the requested JSON results more consistent, the SpaceX launch data was requested and parsed using the GET request and then decoded the response content as a Json result which was then converted into a Pandas data frame.

Also, performed web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launches of the launch records are stored in a HTML. Using BeautifulSoup and request Libraries, I extract the Falcon 9 launch HTML table records from the Wikipedia page, Parsed the table and converted it into a Pandas data frame

- You need to present your data collection process use key phrases and flowcharts

Data Collection – SpaceX API

- Data collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API then requested and parsed the SpaceX launch data using the GET request and decoded the response content as a Json result which was then converted into a Pandas data frame
- GitHub URL of the completed SpaceX API calls notebook

(https://github.com/Chandrubi/SpaceX-Falcon-9-Success-Landing-Prediction/blob/main/Spacex-data-collection-api.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:

```
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

```
response.status_code
```

```
200
```

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

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

Using the dataframe `data` print the first 5 rows

Data Collection - Scraping

- Performed web scraping to collect Falcon 9 historical launch records from a Wikipedia using BeautifulSoup and request, to extract the Falcon 9 launch records from HTML table of the Wikipedia page, then created a data frame by parsing the launch HTML.
- GitHub URL of the completed web scraping notebook, as an external reference and peer-review

(https://github.com/Chandrubi/SpaceX-Falcon-9-Success-Landing-Prediction/blob/main/SpaceX-webscraping.ipynb)

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

Next, request the HTML page from the above URL and get a `response` object

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.

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

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
data=requests.get(static_url).text

soup = BeautifulSoup(response.content, 'html.parser')
```

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

```
# Use soup.title attribute
soup.title
```

```
<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 th

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
html_tables = soup.find_all('table')
# First we isolate the body of the table which contains all the information
```

Data Wrangling

- Performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
- There are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad. True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.
- So, convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful
- GitHub URL of your completed data wrangling related notebooks

(https://github.com/Chandrubi/SpaceX-Falcon-9-Success-Landing-Prediction/blob/main/SpaceX-Data%20wrangling.ipynb)

`RTLS` means the mission outcome was successfully landed to a ground pad `False RTLS` means the mission outco
successfully landed to a drone ship `False ASDS` means the mission outcome was unsuccessfully landed to a drone

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

We create a set of outcomes where the second stage did not land successfully:

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

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_`

```
# landing_class = 0 if bad_outcome
landing_class = [0 if x in bad_outcomes else 1 for x in df['Outcome']]
# landing_class
df['Class']=landing_class
print(df[['Class']].head(8))
print(df["Class"].mean())  # probability of positive outcome 2/3
print(df.head(5))
```

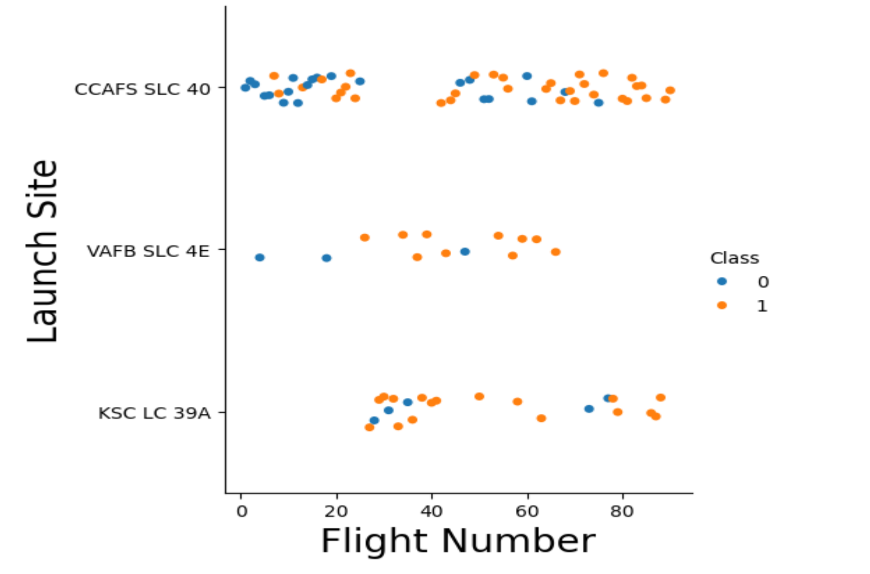
```
   Class
0      0
1      0
2      0
```

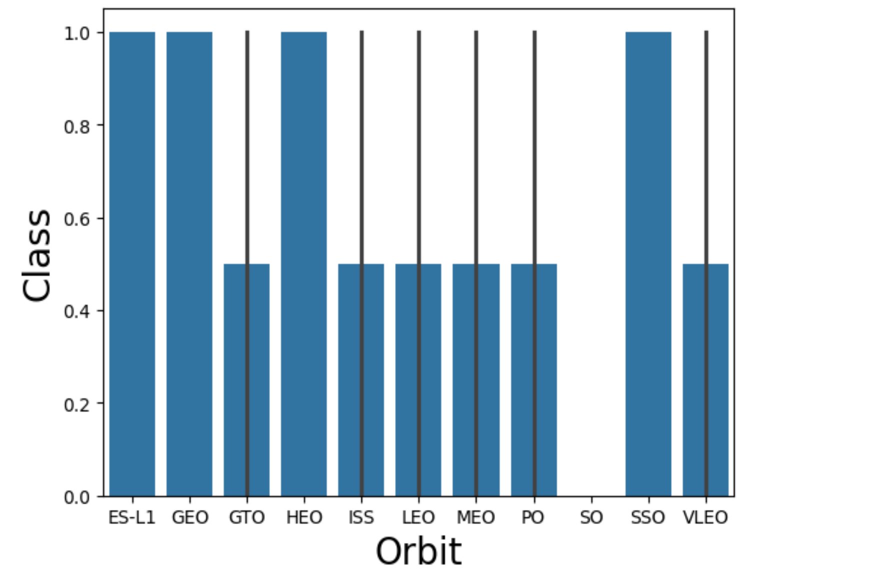
EDA with Data Visualization

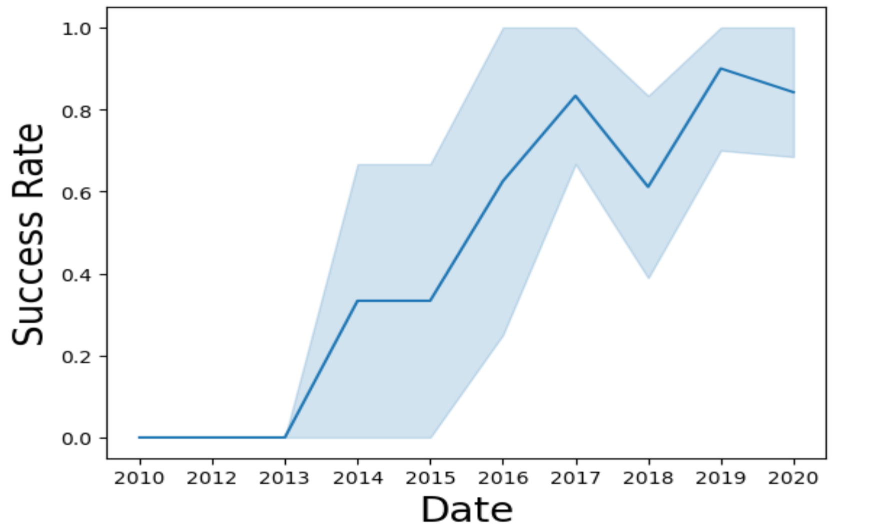
- Perform exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib
  - Exploratory Data Analysis
  - Preparing Data Feature Engineering
- Used scatter plots to Visualize the relationship between Flight Number and Launch Site, Payload and Launch Site, Flight Number and Orbit type, Payload and Orbit type.
- Used Bar chart to Visualize the relationship between success rate of each orbit type
- Line plot to Visualize the launch success yearly trend
- GitHub URL of your completed EDA with data visualization notebook

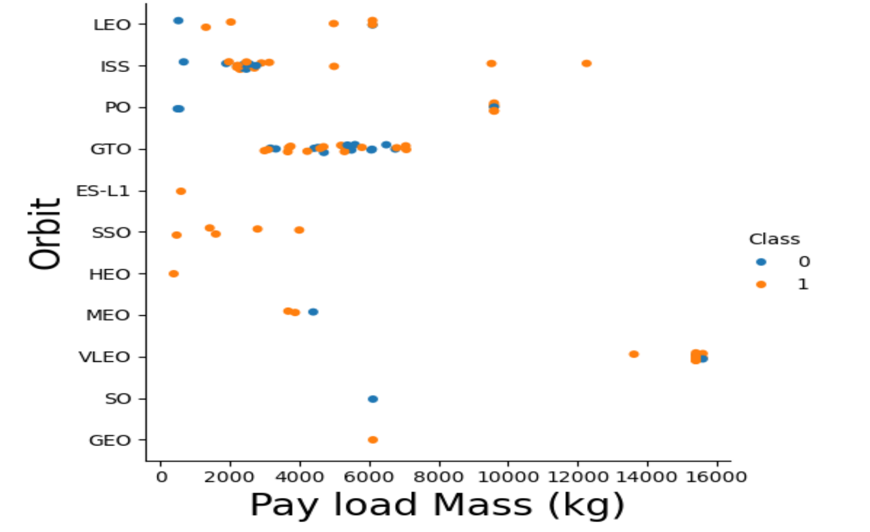
(https://github.com/Chandrubi/SpaceX-Falcon-9-Success-Landing-Prediction/blob/main/SpaceX-EDA%20DataViz.ipynb)

EDA with Data Visualization(Cont..)









EDA with SQL

Various SQL queries were performed and some of them are

- **Display the names of the unique launch sites in the space mission and number of Launches**

```
%sql select "Launch_Site",Count(*) from SPACEXTABLE group by "Launch_Site"
```

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

```
%sql select * from SPACEXTABLE 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_) from SPACEXTABLE where Customer='NASA (CRS)'
```

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

```
%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**

```
%sql select Booster_Version from SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

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

```
%sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

- **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**

```
%sql SELECT SUBSTR(Date,6,2) AS Month, Booster_Version, Launch_site FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Failure%drone%' AND SUBSTR(Date,0,5) = '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.**

```
%sql SELECT Landing_Outcome, COUNT(*) AS Numbers FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success%' AND Date BETWEEN '2010-06-04' and '2017-03-20' GROUP BY Landing_Outcome ORDER BY Numbers D
```

- **GitHub URL of your completed EDA with SQL notebook (https://github.com/Chandrubi/SpaceX-Falcon-9-Success-Landing-Prediction/blob/main/SpaceX-EDA-sql%20using%20sqllite.ipynb)**

Build an Interactive Map with Folium

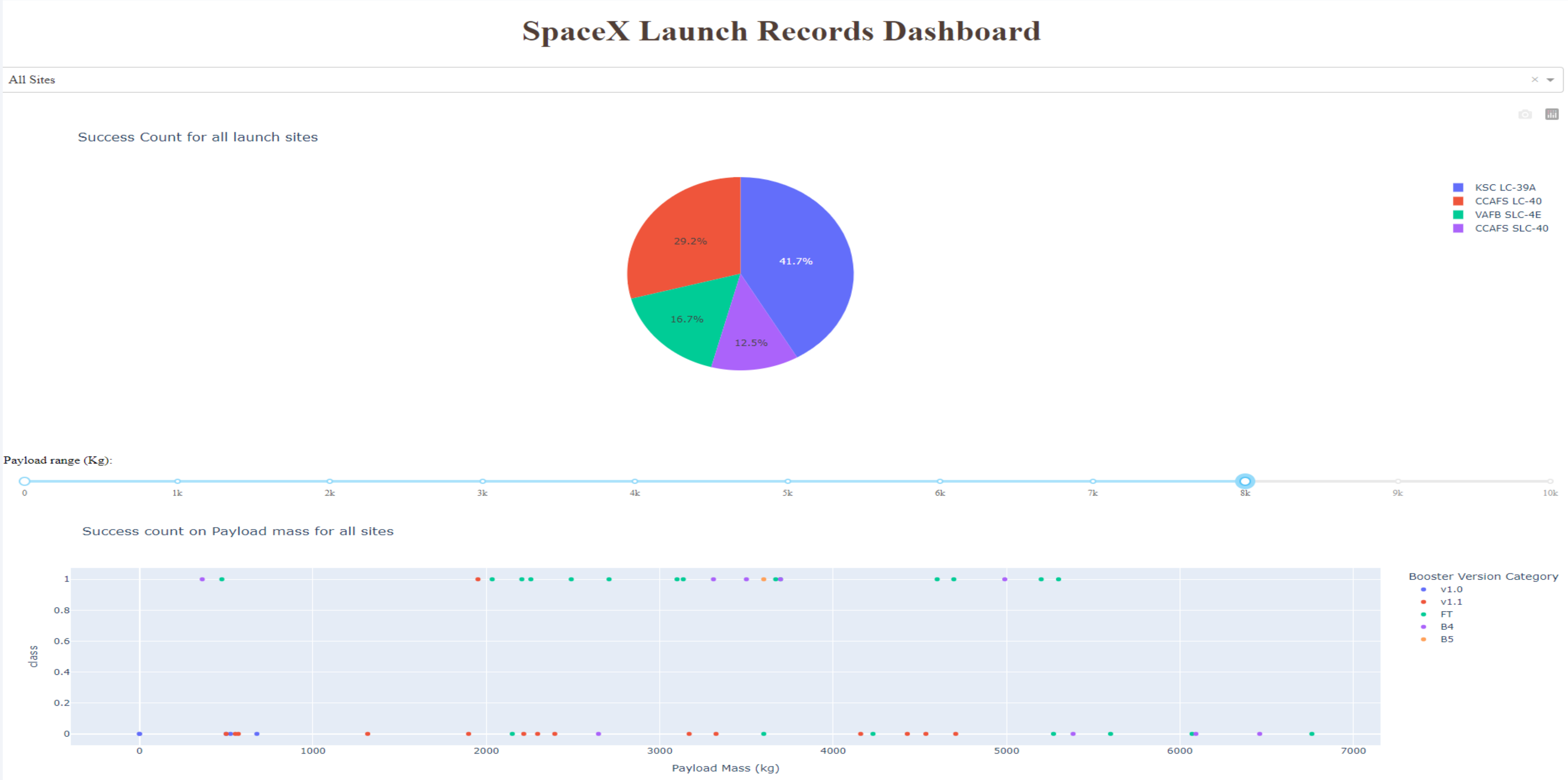
- Performed more interactive visual analytics using Folium
    - Mark all launch sites on a map
    - Mark the success/failed launches for each site on the map
    - Calculate the distances between a launch site to its proximities
- Added folium.Circle and folium.Marker for each launch site on the site map
- Created markers for all launch records. If a launch was successful (class=1), then we use a green marker and if a launch was failed, we use a red marker (class=0)
- Calculated the distance between the coastline point and the launch site.
- GitHub URL of your completed interactive map with Folium map (https://github.com/Chandrubi/SpaceX-Falcon-9-Success-Landing-Prediction/blob/main/SpaceX-launch_site_location%20Analysis%20using%20folium.ipynb)

Build a Dashboard with Plotly Dash

Built an interactive dashboard application with Plotly dash by:

- Adding a Launch Site Drop-down Input Component
- Adding a callback function to render success-pie-chart based on selected site dropdown
- Adding a Range Slider to Select Payload
- Adding a callback function to render the success-payload-scatter-chart scatter plot
- GitHub URL of your completed Plotly Dash lab (https://github.com/Chandrubi/SpaceX-Falcon-9-Success-Landing-Prediction/blob/main/SpaceX%20Intractive%20Dashboard%20Spacex_dash_app.txt)

Dashboard with Plotly Dash



Predictive Analysis (Classification)

- Created a machine learning pipeline to predict if the first stage will land with the given data. It summarize, how the model is built, evaluated, improved, and found the best performing classification model
  - Create a NumPy array from the column Class in data, by applying the method to_numpy() then assign it to the variable Y
  - Standardize the data in X then reassign it to the variable X
  - Use the function train_test_split to split the data X and Y into training and test data. Set the parameter test_size to 0.2 and random_state to 2
  - Create a logistic regression object then create a GridSearchCV object logreg_cv with cv = 10
  - Calculate the accuracy on the test data using the method score
  - Examining the confusion matrix, we see that logistic regression can distinguish between the different classes
- Follow the above steps for SVM, Decision Tree and KNN

Predictive Analysis (Classification)

- Find best method of SVM, Classification Trees, Logistic Regression and KNN

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

- GitHub URL of your completed predictive analysis lab(https://github.com/Chandrubi/SpaceX-Falcon-9-Success-Landing-Prediction/blob/main/SpaceX_Machine%20Learning%20Prediction.ipynb)

Results

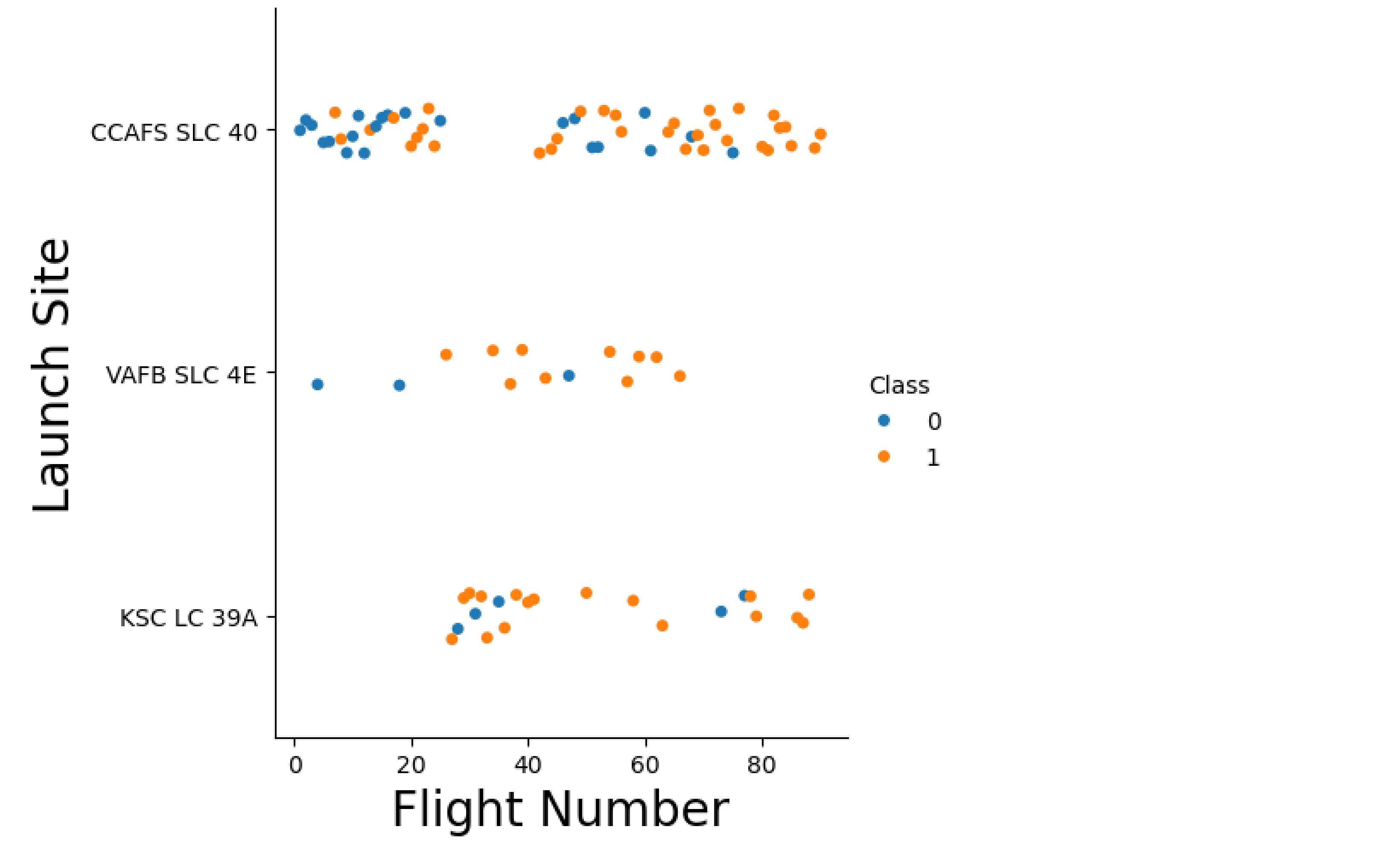
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

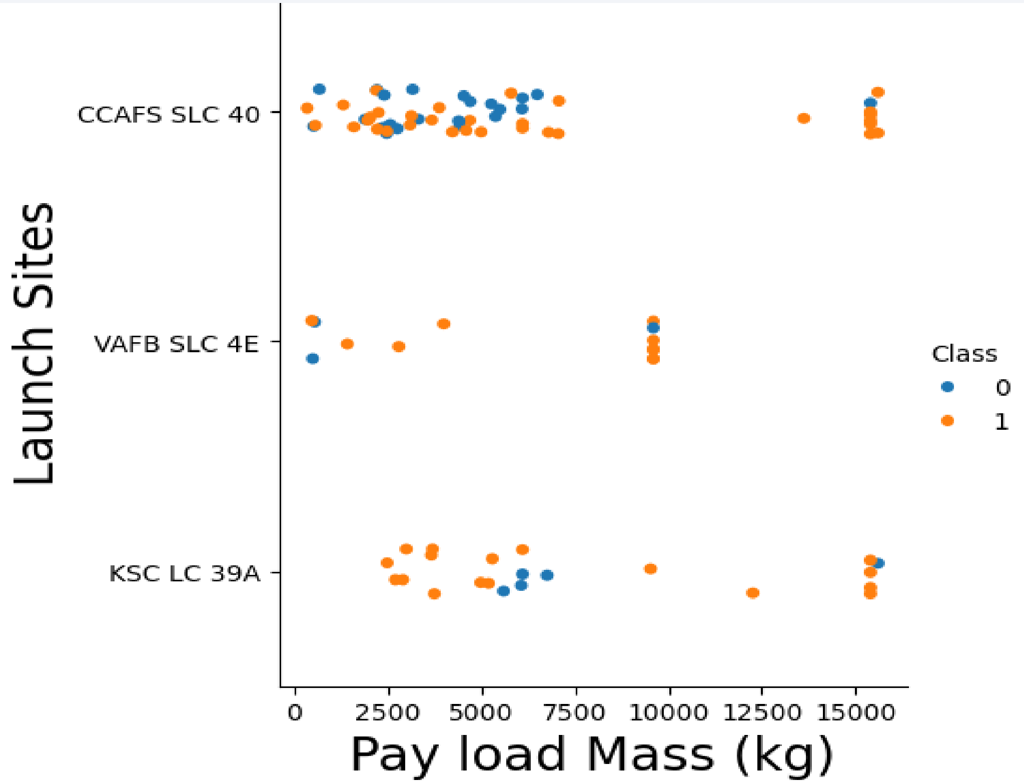
Flight Number vs. Launch Site

Scatter plot of Flight Number vs. Launch Site



Payload vs. Launch Site

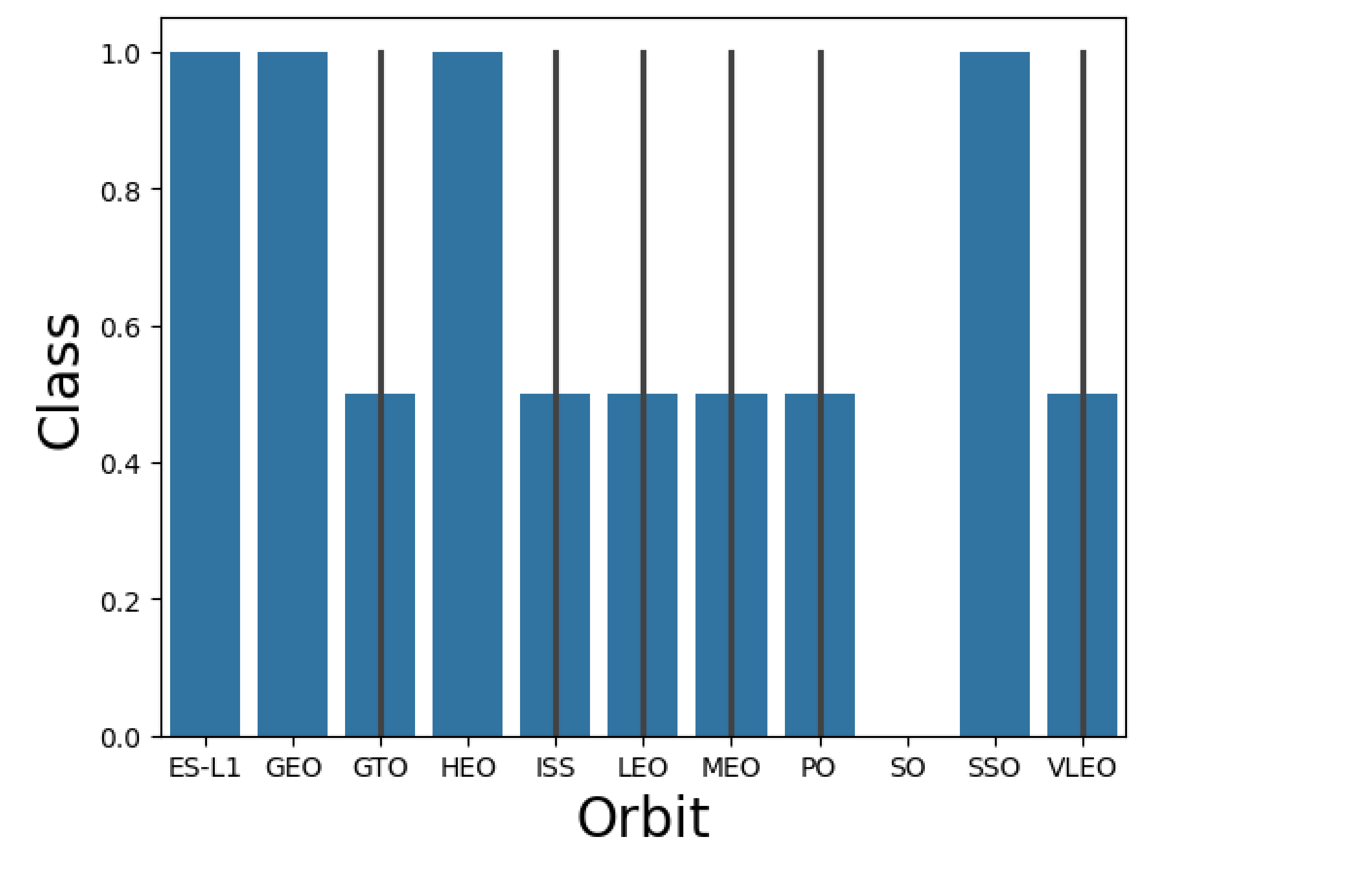
Scatter plot of Payload vs. Launch Site



Now if you observe Payload Mass 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).

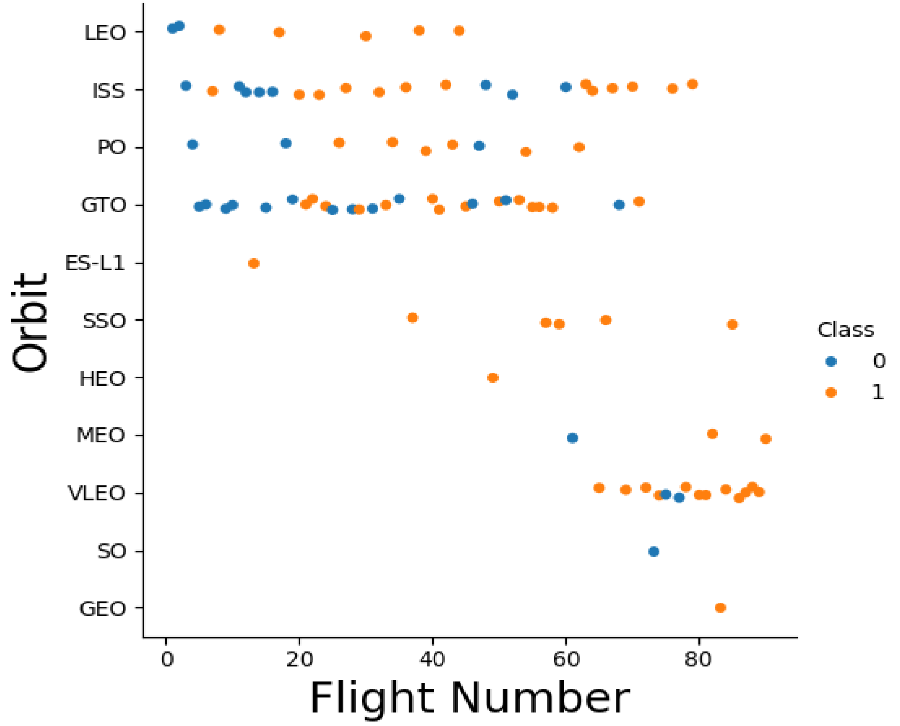
Success Rate vs. Orbit Type

Bar chart for the success rate of each orbit type



Flight Number vs. Orbit Type

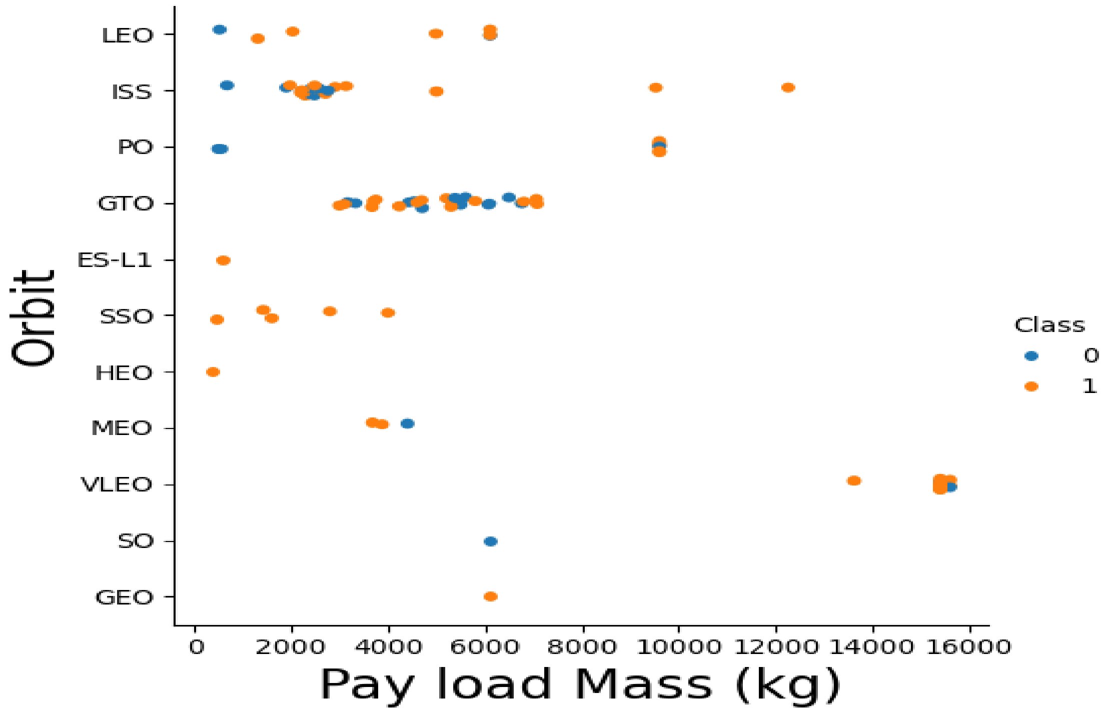
Scatter point of Flight number vs. Orbit type



You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

Payload vs. Orbit Type

Scatter point of payload vs. orbit type

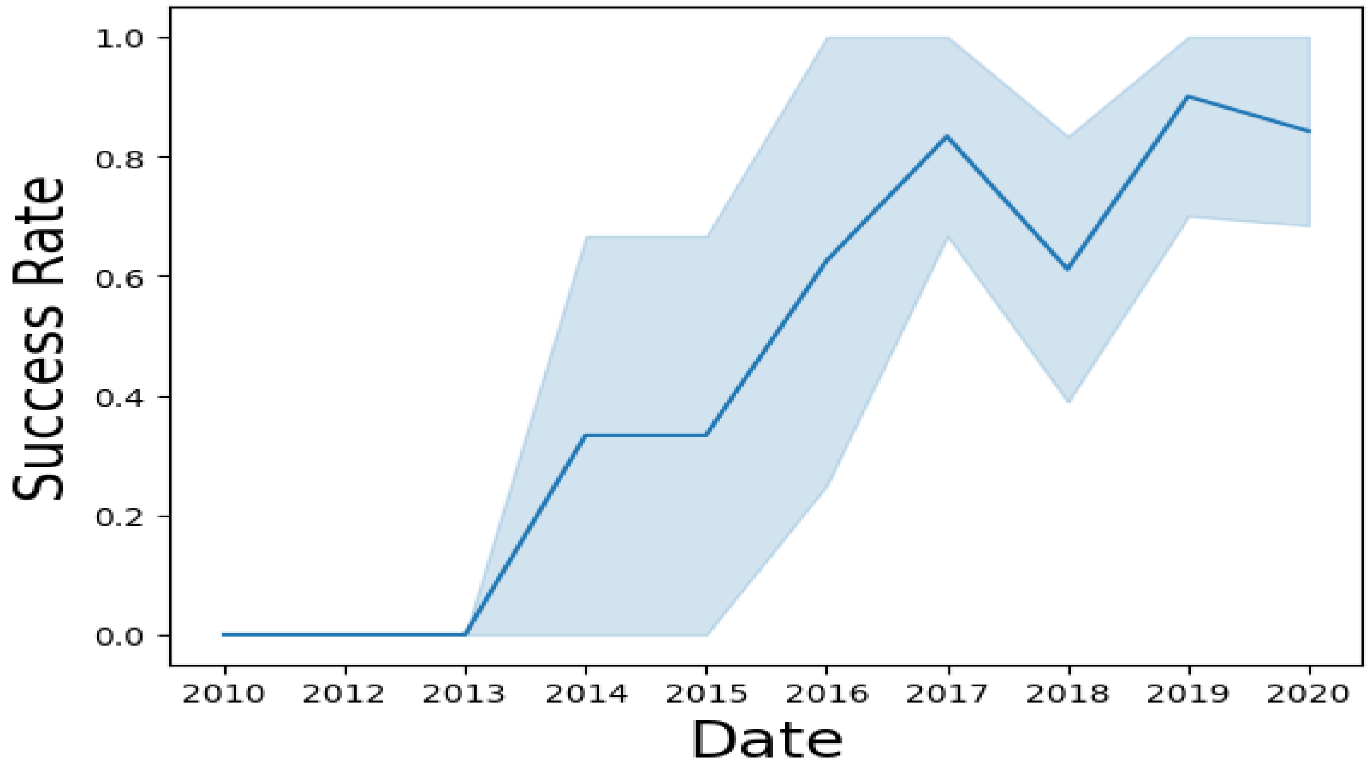


With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

Launch Success Yearly Trend

Line chart of yearly average success rate



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

```
%sql select "Launch_Site",Count(*) from SPACEXTABLE group by "Launch_Site"
```

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

Launch_Site	Count(*)
CCAFS LC-40	26
CCAFS SLC-40	34
KSC LC-39A	25
VAFB SLC-4E	16

Above sql gives unique launch sites with number of launches in each site.

Launch Site Names Begin with 'CCA'

Find 5 records where launch sites begin with `CCA`

```
%sql select * from SPACEXTABLE 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
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

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer='NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596

Used Sum() aggregate function to get the total payload and the WHERE condition filters the data only for NASA

Average Payload Mass by F9 v1.1

Calculate the average payload mass carried by booster version F9 v1.1

```
%sql select Avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where Booster_Version like 'F9 v1.1%'
```

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

Avg(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

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

```
%sql select min(DATE) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
```

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

min(DATE)
2015-12-22

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

```
%sql select Booster_Version from SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

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

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

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

```
%sql select count(Mission_Outcome) from SPACEXTBL WHERE Mission_Outcome = 'Success' or Mission_Outcome = 'Failure (in flight)'
```

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

count(Mission_Outcome)
99

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass

```
%sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

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

2015 Launch Records

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
%sql SELECT SUBSTR(Date,6,2) AS Month, Booster_Version, Launch_site FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Failure%drone%' AND SUBSTR(Date,0,5) = '2015'
```

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

Month	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

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

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

```
%sql SELECT Landing_Outcome, COUNT(*) AS Numbers FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success%' AND Date BETWEEN '2010-06-04' and '2017-03-20' GROUP BY Landing_Outcome ORDER BY Numbers D
```

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

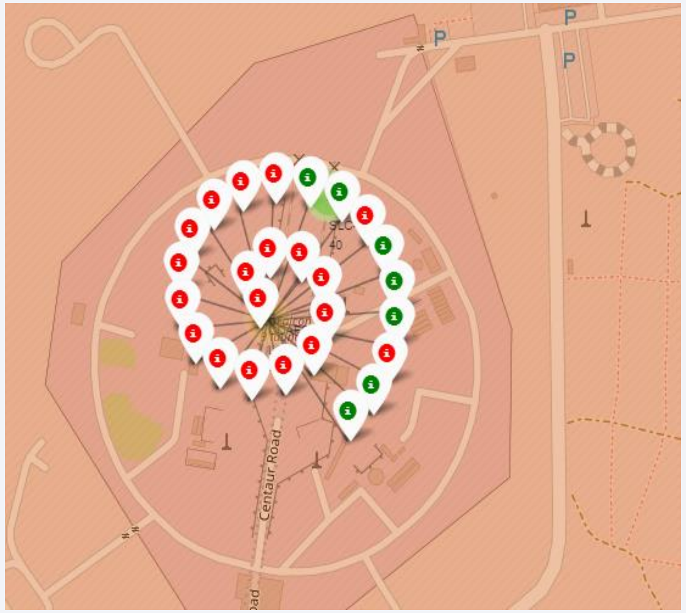
Landing_Outcome	Numbers
Success (drone ship)	5
Success (ground pad)	3

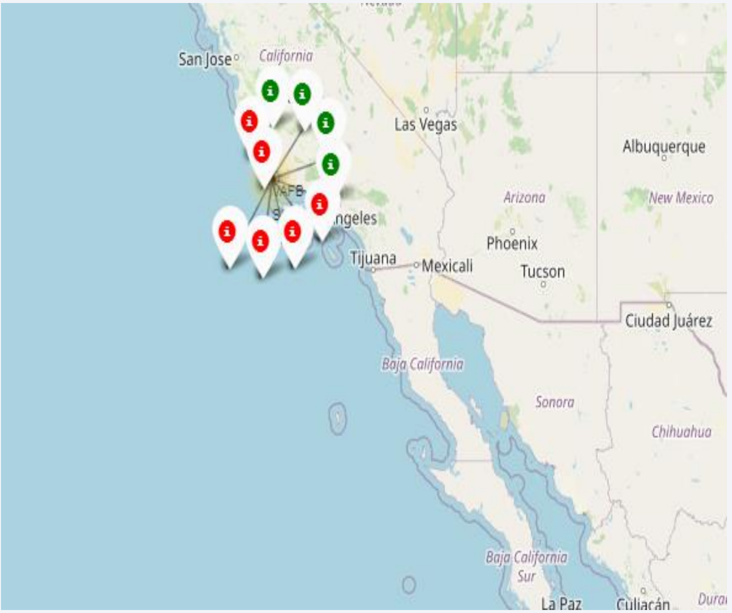
Section 3

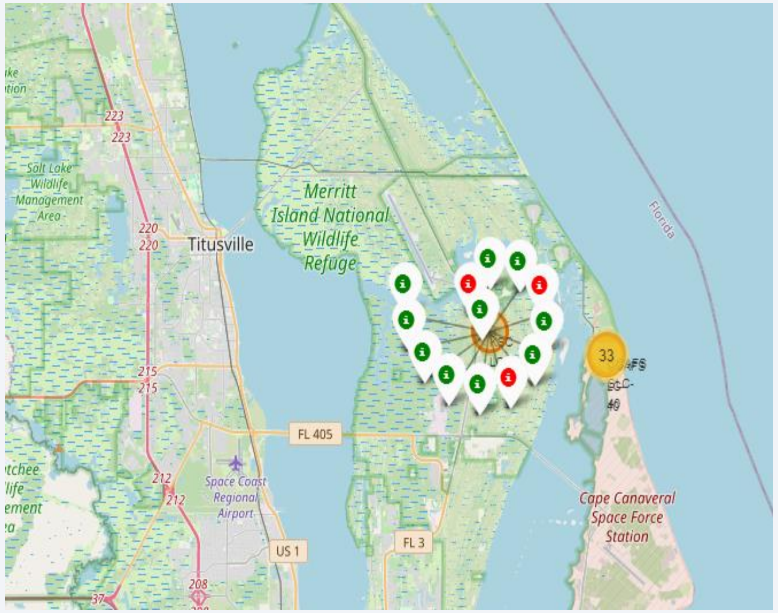
Launch Sites Proximities Analysis

Launch Sites in Folium Global Map

Launch Outcomes on Each Sites

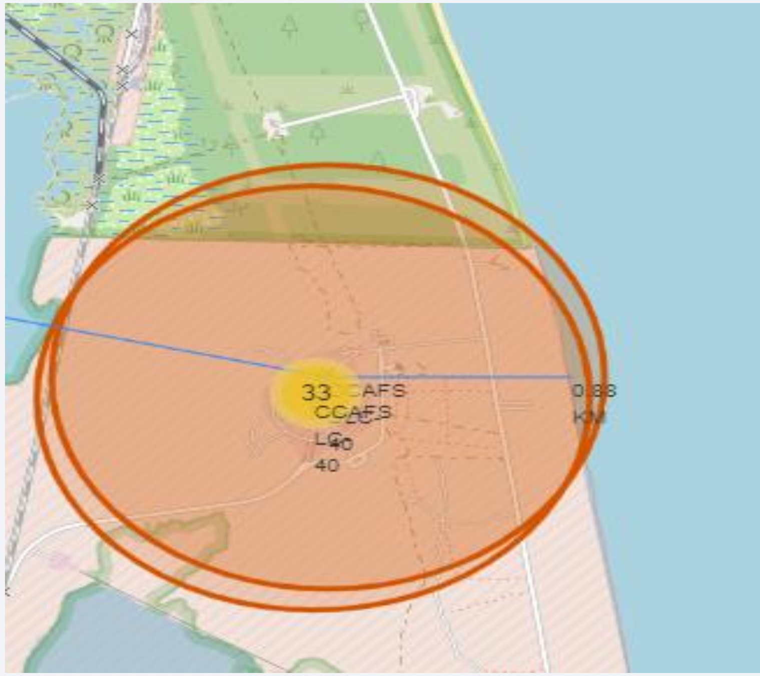


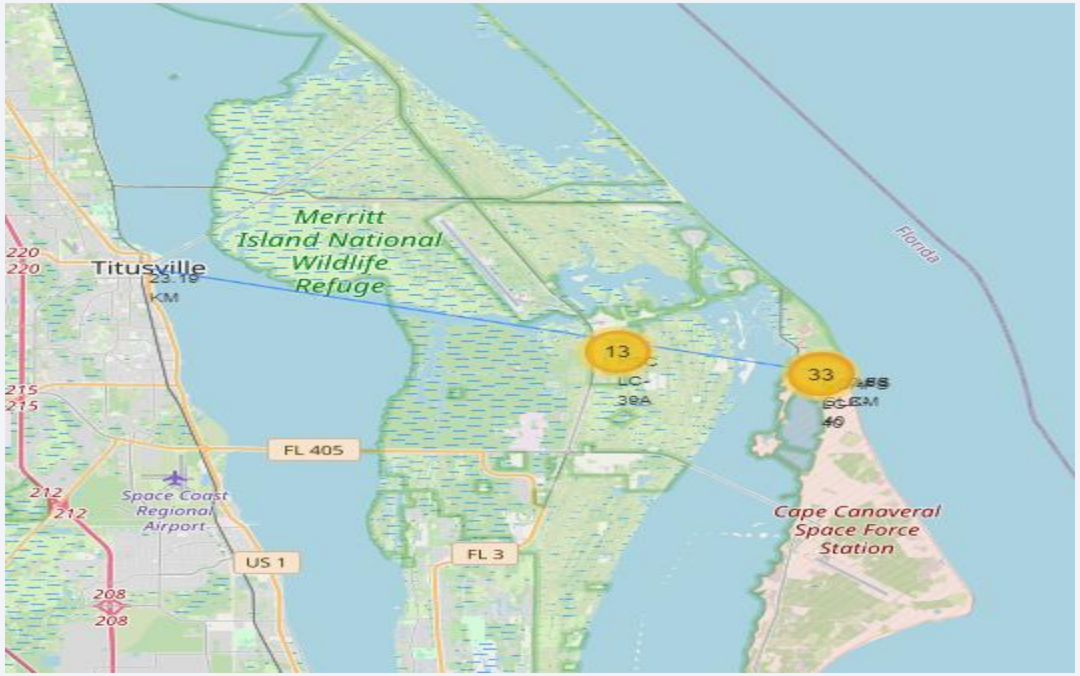




In the Eastern coast (Florida) Launch site KSC LC-39A has relatively high success rates compared to CCAFS SLC-40 & CCAFS LC-40.

Distance b/w a Launch site to its Proximities





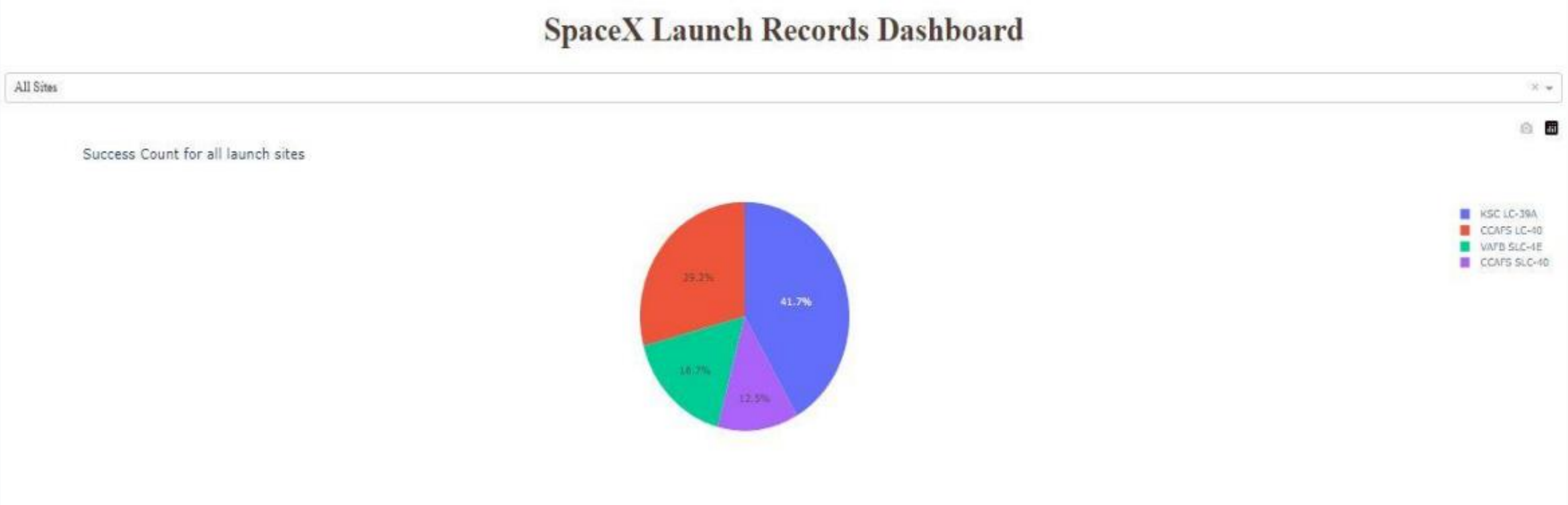
Launch site CCAFS SLC-40 proximity to coastline is 0.86km

Launch site CCAFS SLC-40 closest to highway (Washington Avenue) is 23.19km

Section 4

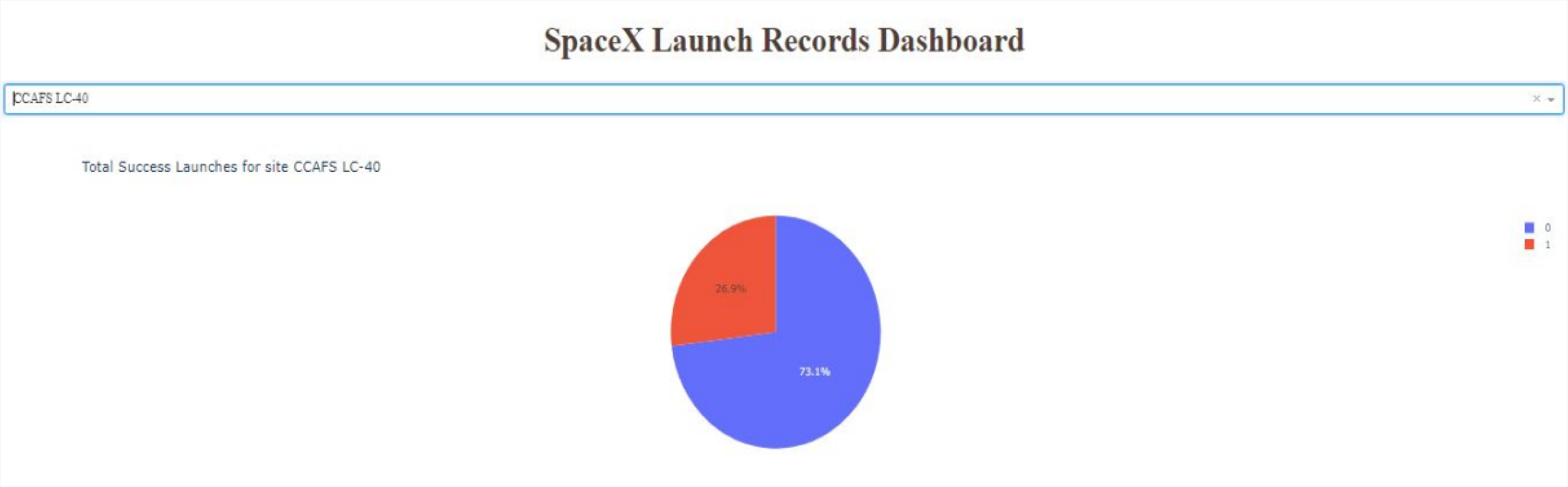
Build a Dashboard with Plotly Dash

Launch Success counts by Sites



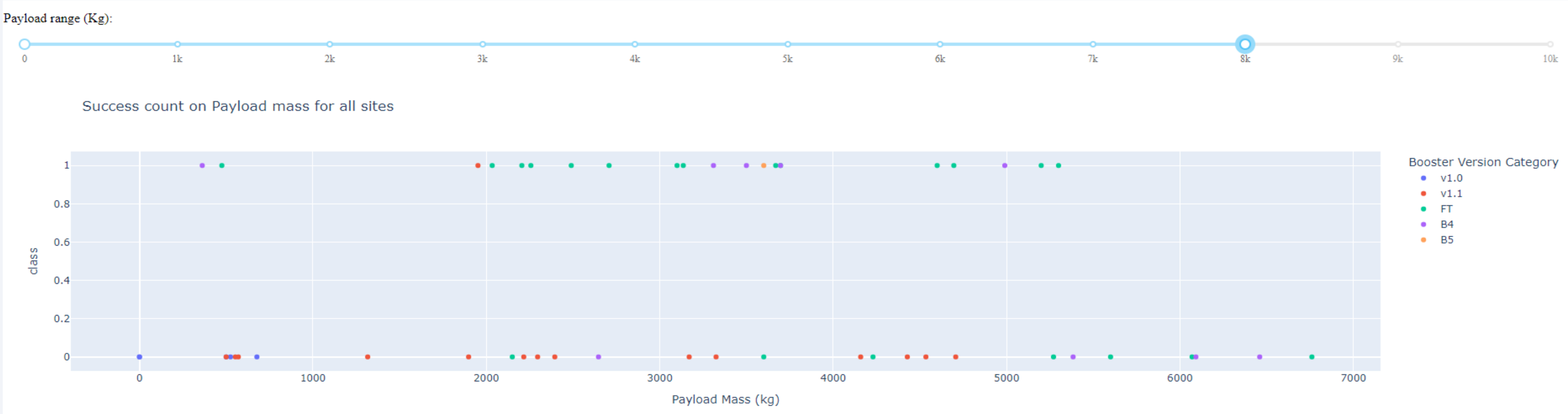
Launch site KSC LC-39A has the highest launch success rate at 42% followed by CCAFS LC-40 at 29%

Success Ratio for Site CCAFS LC-40



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

Payload vs. Launch Outcome



This interactive Dashboard let you select payload ranges

Section 5

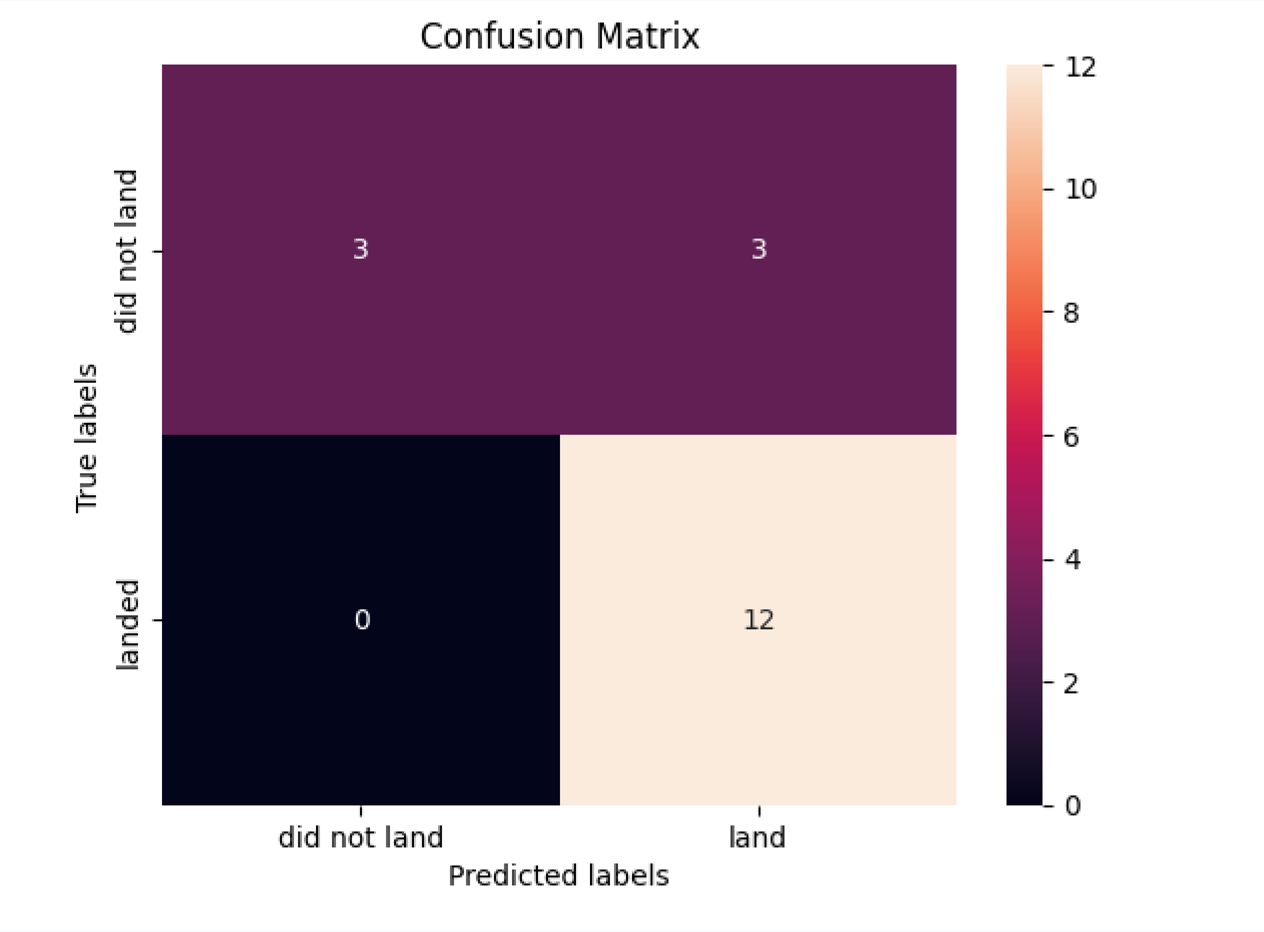
Predictive Analysis (Classification)

Classification Accuracy

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



- Show the confusion matrix of the best performing model with an explanation

Conclusions

- Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, Orbits GTO, ISS, LEO, MEO, PO and VLEO has 50% success rate. Orbit SO has 0% success rate.
- In the LEO orbit, success seems to be related to the number of flights
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS
- 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%
- As the flight number increases in each of the 3 launch sites, so does the success rate. For instance, the success rate for the VAFB SLC 4E launch site is 100% after the Flight number 50. Both KSC LC 39A and CCAFS SLC 40 have a 100% success rates after 80th flight.
- Success rates are kept increasing from Year 2013 onwards

Thank you!