IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Pengcheng Zhao
2024.03.25

Outline

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

Executive Summary

- Summary of methodologies

Data Collection with API and Web Scraping

Data Wrangling

EDA with SQL/Data Visualization(Pandas and Seaborn)

Interactive Visual Analytics with Folium

Prediction with Machine Learning

- Summary of all results

Exploratory Data Analysis result

Interactive analytics with screenshots

Predictive Analytics Results

Introduction

- Project background and context

The commercial space age is here, companies are making space travel affordable for everyone. Perhaps the most successful is SpaceX——Sending manned missions to Space——One reason SpaceX can do this is the rocket launches are relatively inexpensive. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million 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.

- Problems you want to find answers

To determine the price of each launch;

To determine if SpaceX will reuse the first stage using a machine learning model and use public information to predict if SpaceX will reuse the first stage.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected from SpaceX RESTful API and Web scraping from Wikipedia.
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Using scikit-learn to build, tune, evaluate classification models

Data Collection

·Data collection using get request to the SpaceX API

·Decoded the response content as a Json object using .json() function call and turn it into a pandas dataframe using pd.json_normalize().

·Cleaned the data, checked for missing values and fill in missing values where necessary.

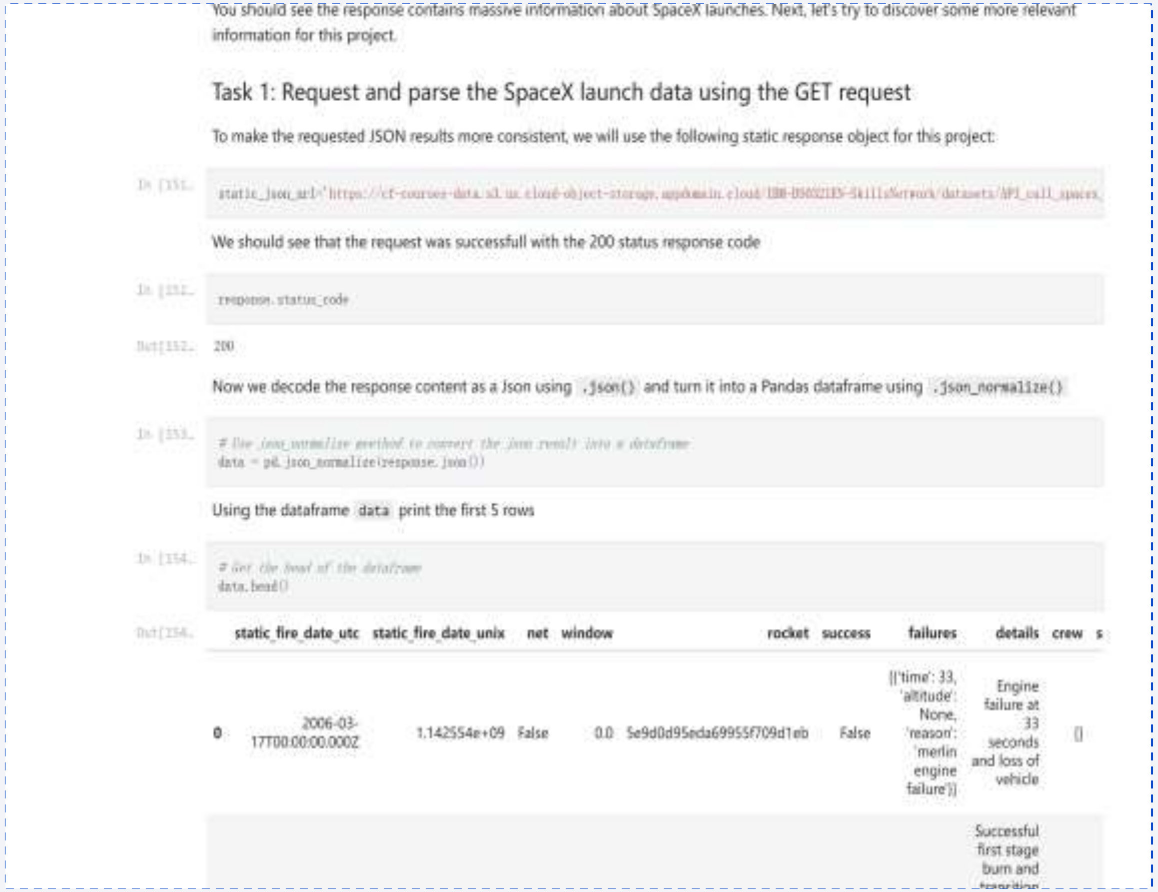
· Performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup tool.

· The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas Dataframe for future analysis.

Data Collection – SpaceX API

- Data collection using get request to the SpaceX API

- https://github.com/pagesys/coursera_exam/blob/main/capstone_jupyter-labs-spacex-data-collection-api.ipynb



You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

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_call_spacex
```

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 method to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

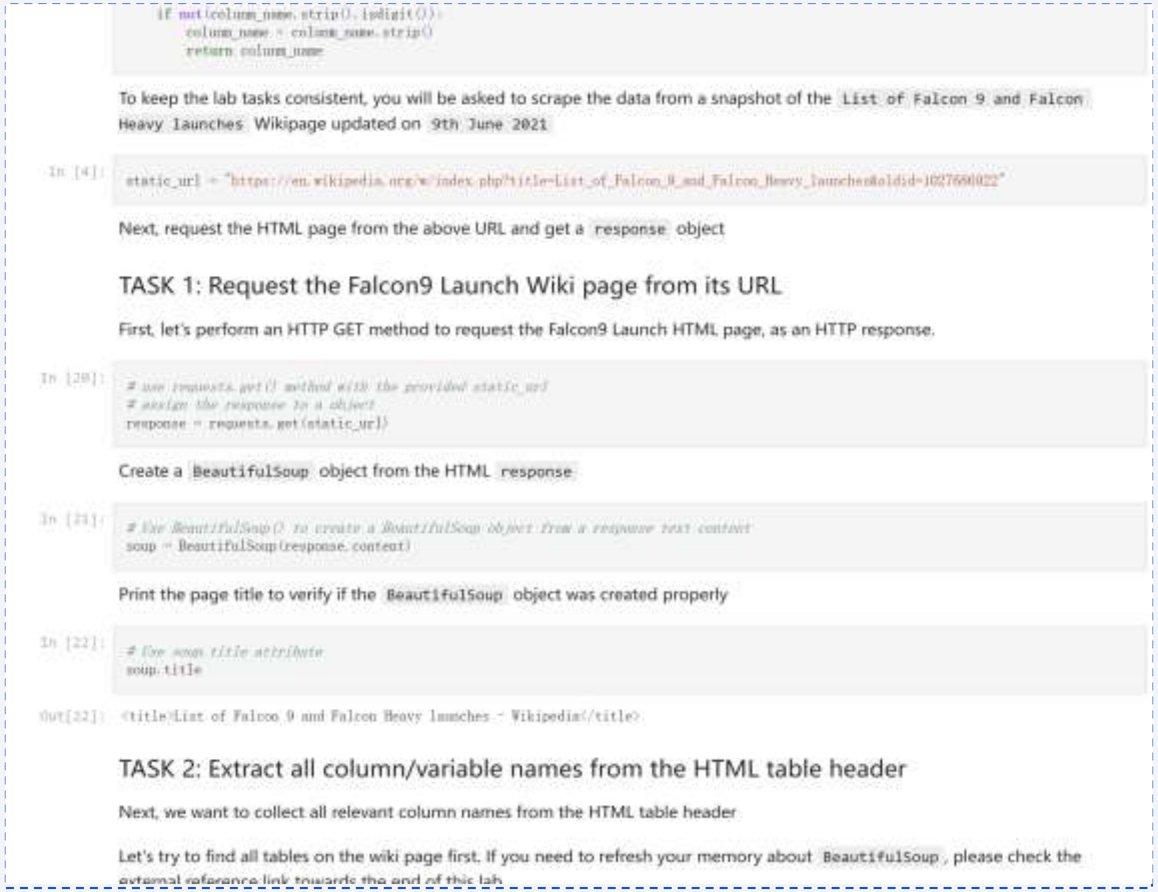
Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head()
```

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	s
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle	[]	
								Successful first stage burn and transition		

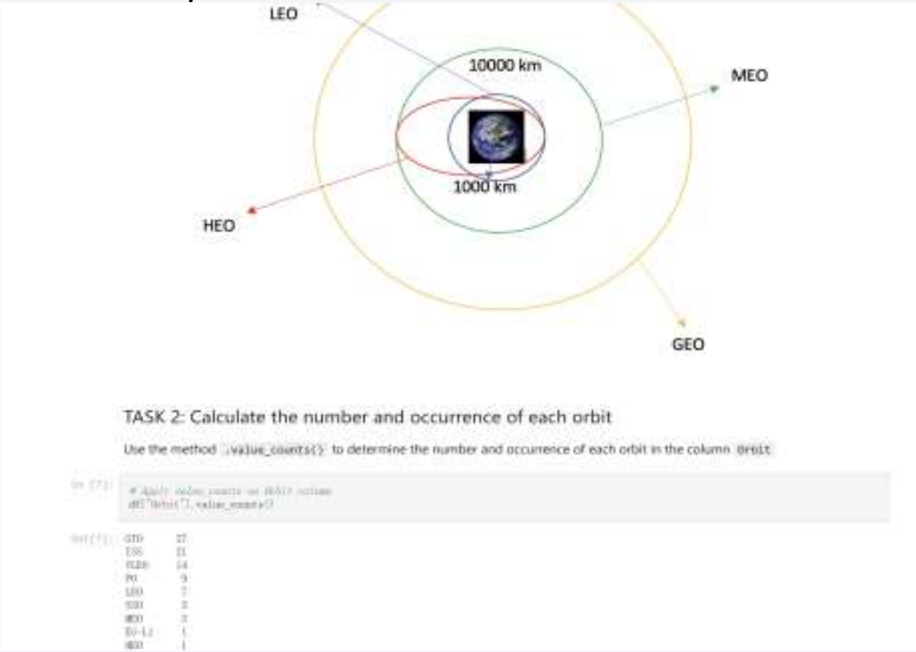
Data Collection - Scraping

- Performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup tool.
- https://github.com/pagesys/coursera_exam/blob/main/capstone_jupyter-labs-webscraping.ipynb



Data Wrangling

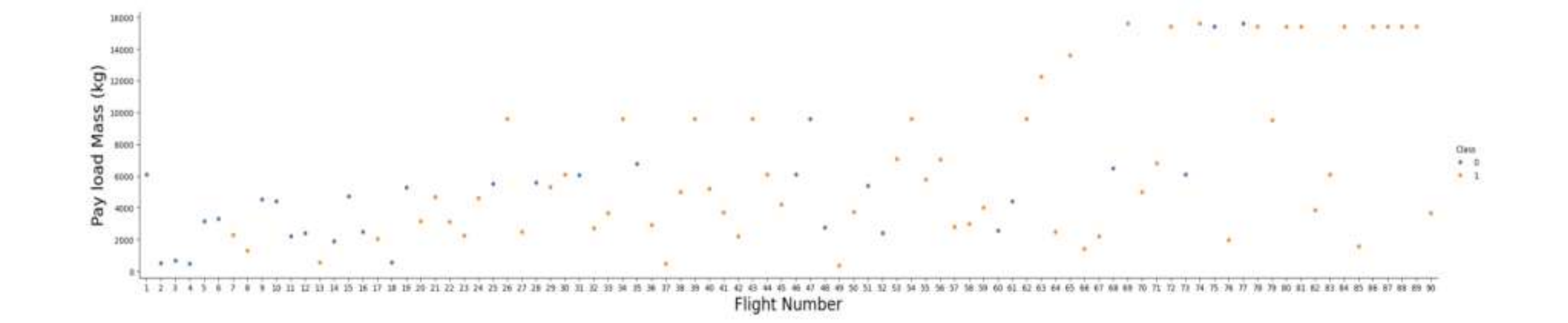
- I performed exploratory data analysis and determined the training labels, then calculated the number of launches at each site, and the number and occurrence of each orbits . And created landing outcome label from outcome column and exported the results to csv file.



- https://github.com/pagesys/coursera_exam/blob/main/capstone_labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Plot list: Catplot, bar chart, Line chart
- Explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.
- https://github.com/pagesys/coursera_exam/blob/main/capstone_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb



EDA with SQL

- Loading the SpaceX table into SQLite database in the jupyter notebook.
- Using EDA with SQL to get insight from the data, such as:
  - **Get the total payload mass carried by boosters launched by NASA (CRS)**
  - **Get average payload mass carried by booster version F9 v1.1**
  - **List the date when the first succesful landing outcome in ground pad was achieved**
  - **Get the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000**
  - et. al
- https://github.com/pagesys/coursera_exam/blob/main/capstone_jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

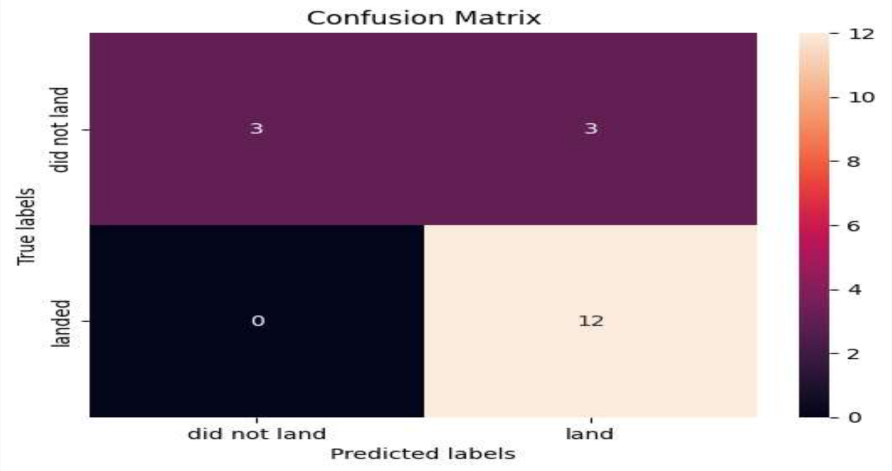
- marking all launch sites, and adding map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- Using the color-labeled marker clusters to identifie which launch sites have relatively high success rate.
- https://github.com/pagesys/coursera_exam/blob/main/capstone_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- Building an interactive dashboard with Plotly dash
- Plotting piecharts to show the total launches by a certain sites, and scatter graph showing the relationship with Outcome and PayloadMass (Kg) for the different booster version
- https://github.com/pagesys/coursera_exam/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- Building different machine learning models and tune different hyperparameters using GridSearchCV, then used accuracy as the metric for machine learning models, improved the model using feature engineering and algorithm tuning



- https://github.com/pagesys/coursera_exam/blob/main/Capstone_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

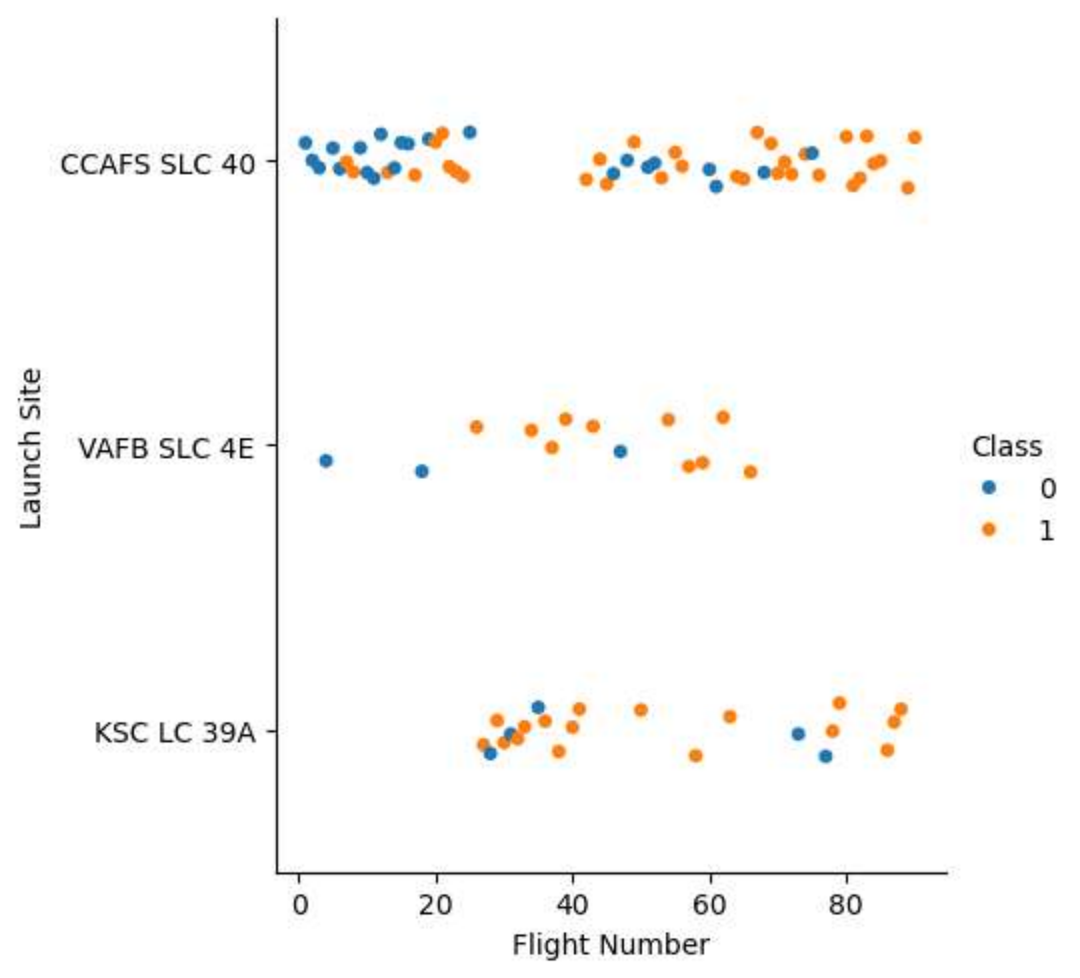
Results

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

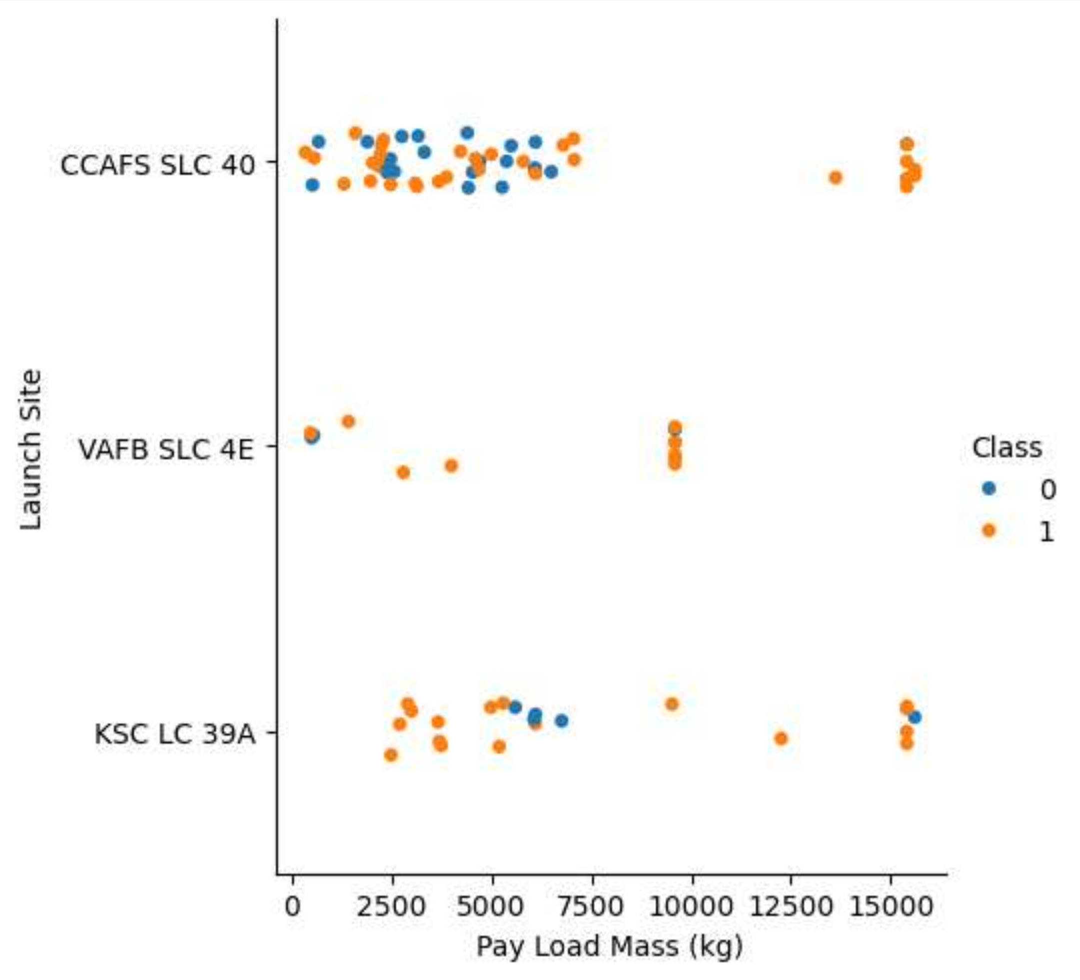
Section 2

Insights drawn from EDA

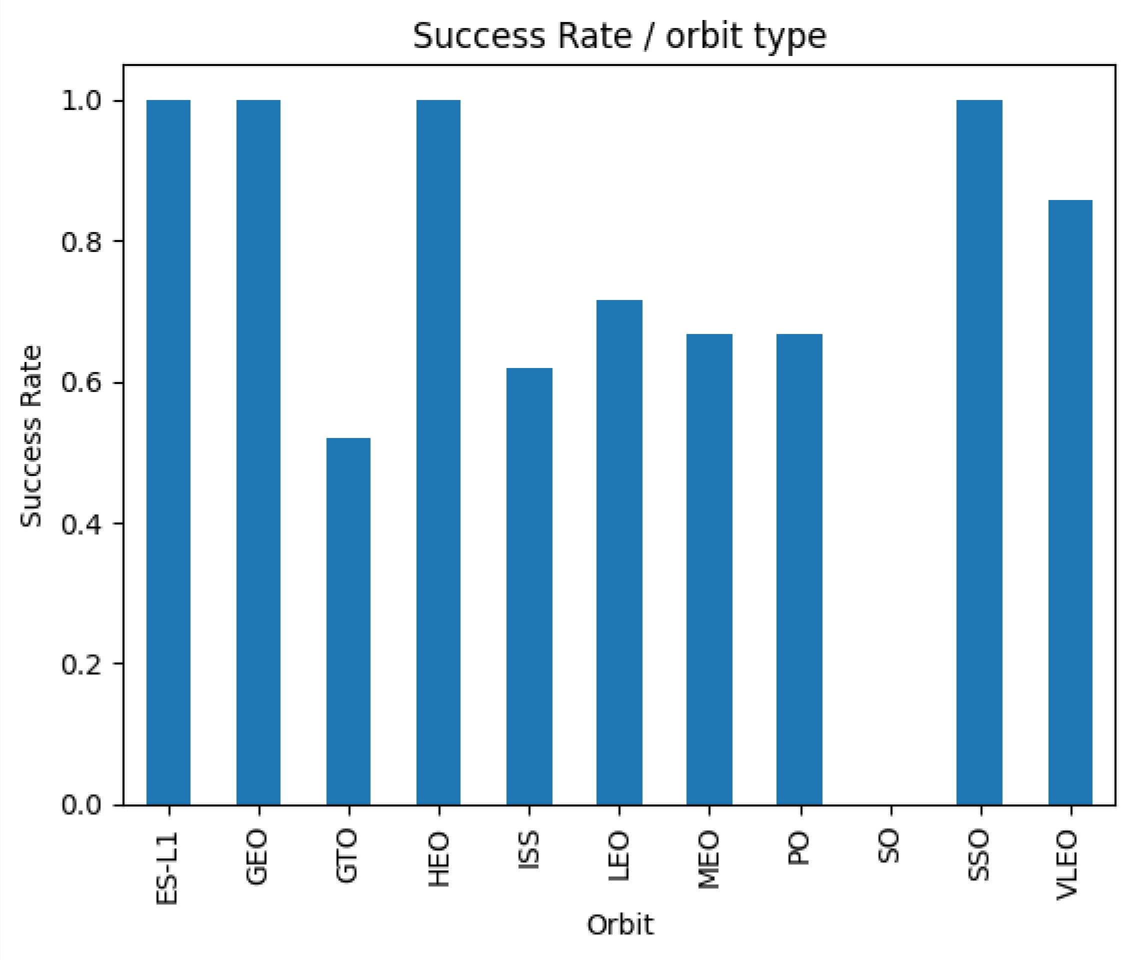
Flight Number vs. Launch Site



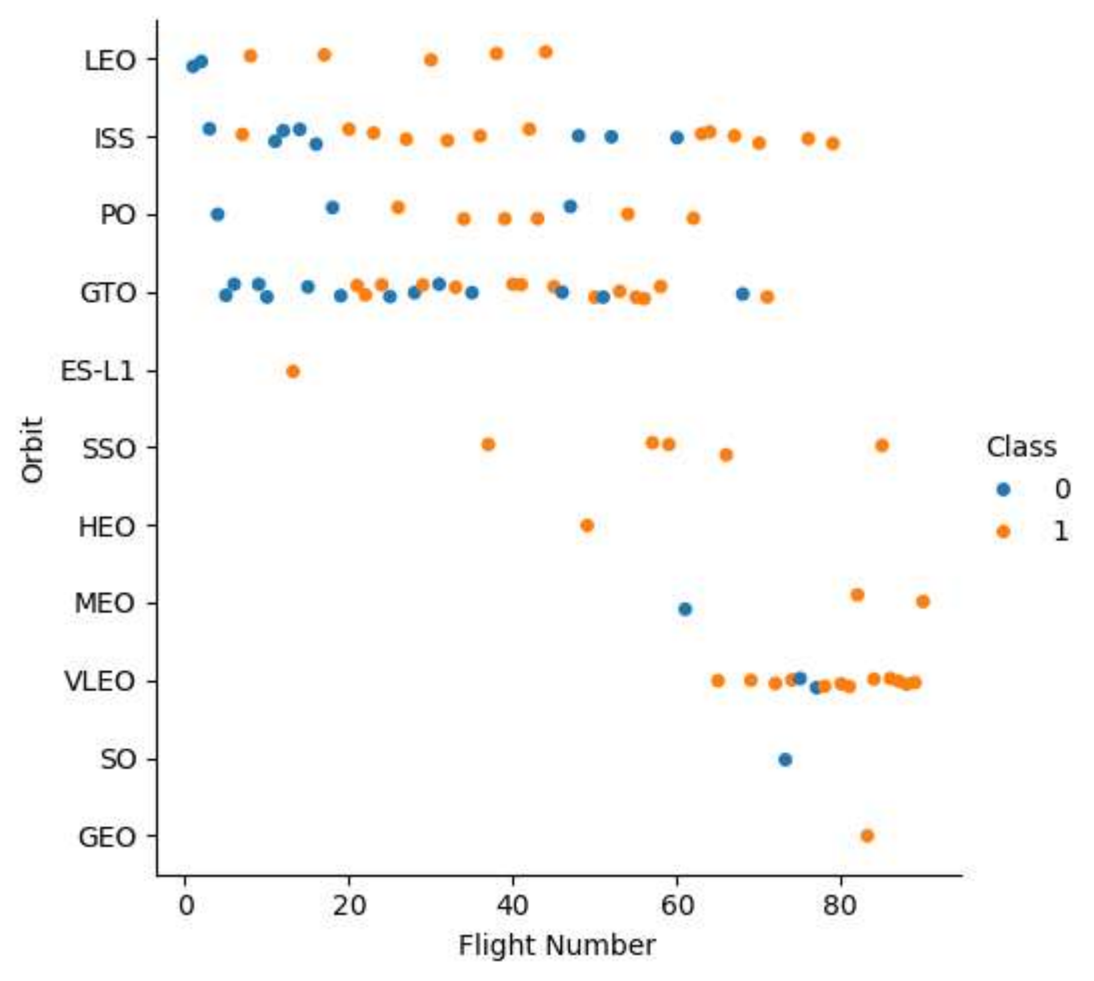
Payload vs. Launch Site



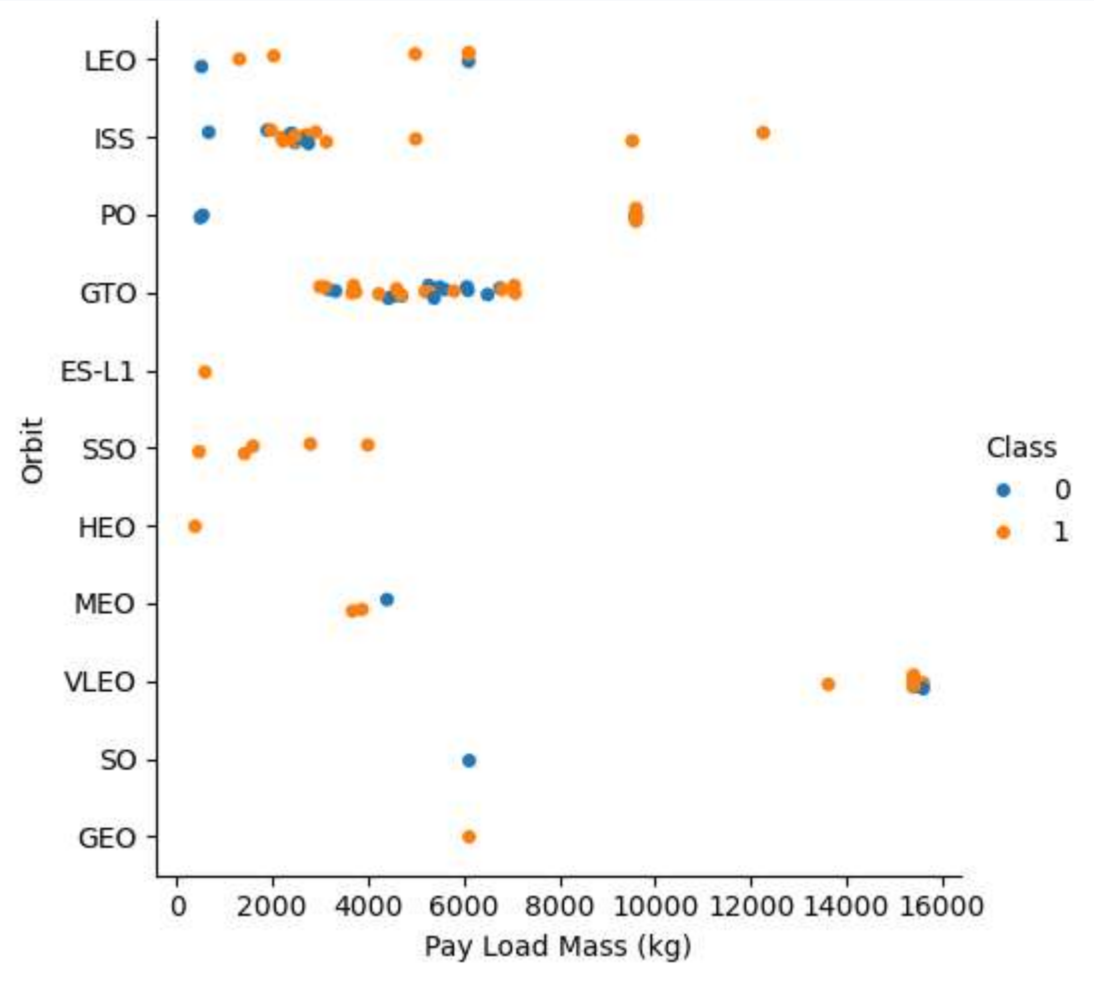
Success Rate vs. Orbit Type



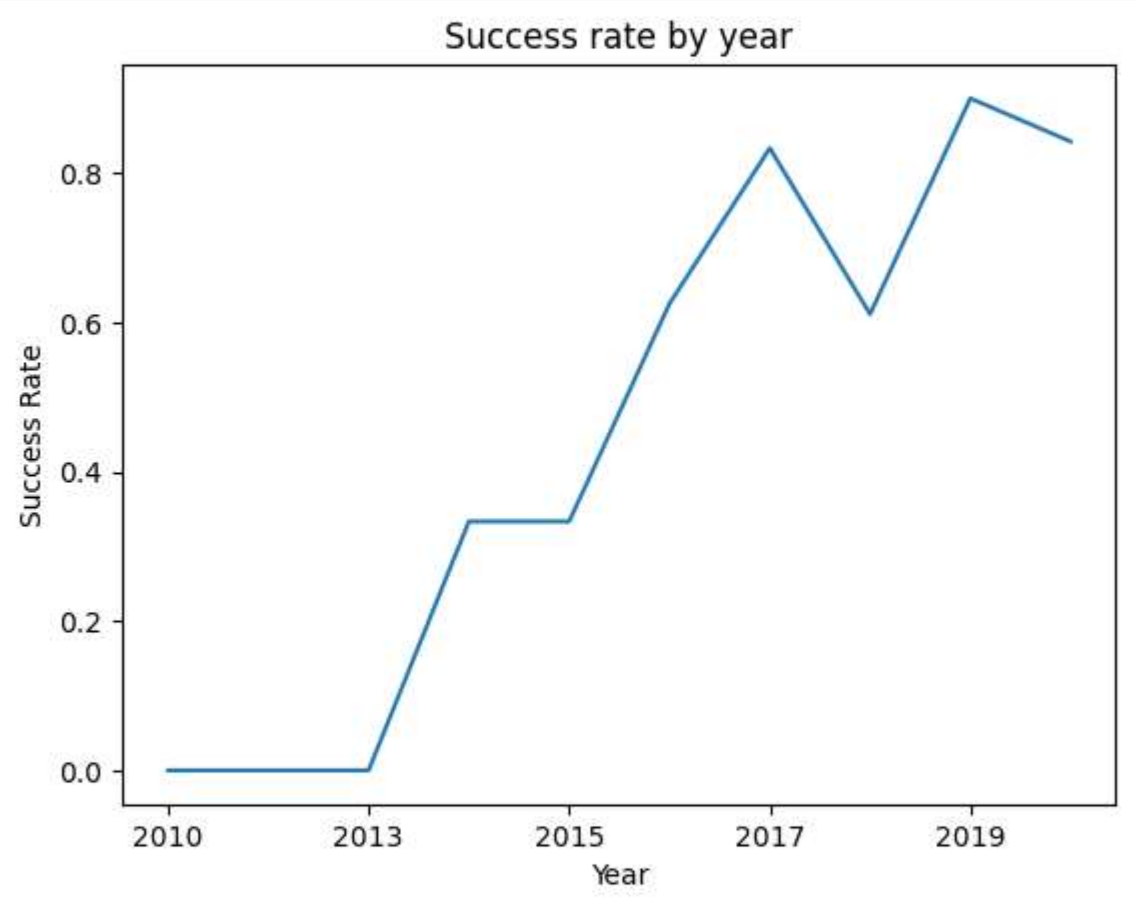
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

Task 1

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

In [34]:

```
%sql select distinct Launch_Site from spacextable;
```

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

Out[34]:

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

Launch Site Names Begin with 'CCA'

Task 2

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

In [35]:
```
%sql select * from spacextable where Launch_Site like 'CCA%' limit 5;
```

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

Out[35]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_C
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (pa
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 (pa
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No

Total Payload Mass

Task 3

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

In [36]:

```
%sql select sum(PAYLOAD_MASS__KG_) from spacextable where Customer = "NASA (CRS)";
```

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

Out[36]:

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

Task 4

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

In [37]:

```
%sql select avg(PAYLOAD_MASS__KG_) from spacextable where Booster_Version = "F9 v1.1"
```

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

Out[37]:

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

In [38]:

```
%sql select min(Date) from spacextable where Landing_Outcome = "Success (ground pad)";
```

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

Out[38]:

min(Date)
2015-12-22

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

In [39]:

```
%sql select Booster_Version from spacextable where Landing_Outcome = "Success (drone ship)" a
```

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

Out[39]:

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

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

In [43]:

```
%sql select count(Mission_Outcome) from spacextable where Mission_Outcome = "Success" or Miss
```

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

Out[43]:

count(Mission_Outcome)
98

Boosters Carried Maximum Payload

Task 8

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

In [45]:

```
%sql select Booster_Version from spacextable where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MA
```

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

Out[45]:

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

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.

In [49]:

```
%sql select substr(Date, 6,2) as Month,Booster_Version,Launch_Site,Landing_Outcome from space
```

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

Out[49]:

Month	Booster_Version	Launch_Site	Landing_Outcome
01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

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.

In [52]:

```
%sql select count(*) as total, landing_outcome from spacextable where date between "2010-06-0
```

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

Out[52]:

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

Section 3

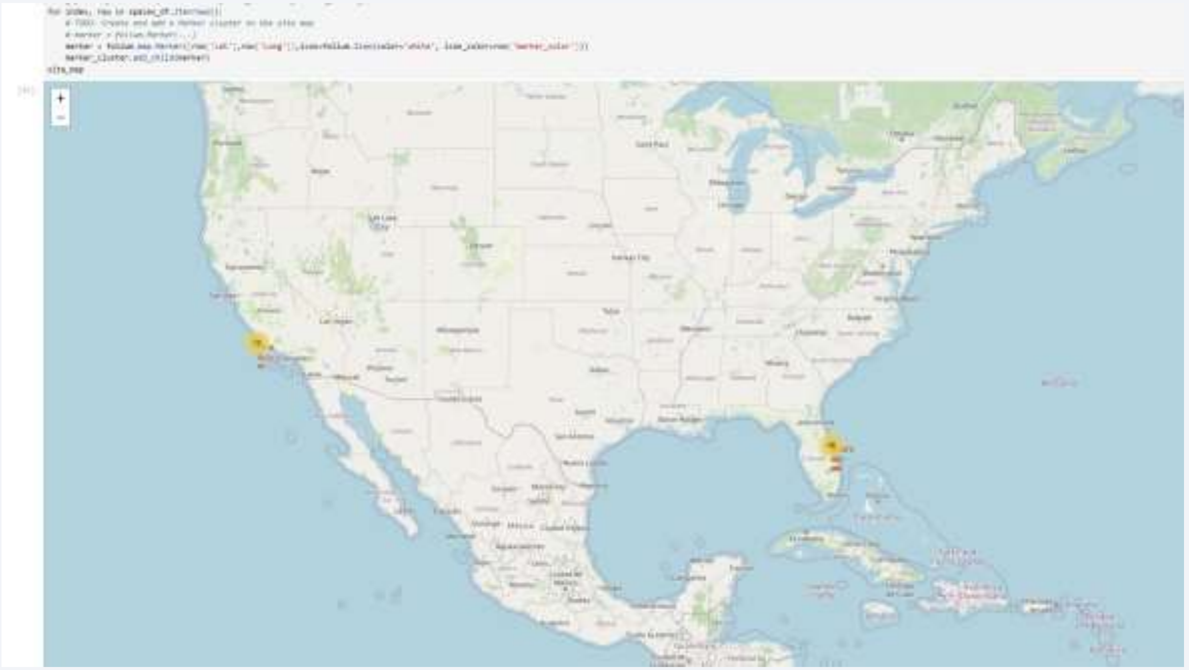
Launch Sites Proximities Analysis

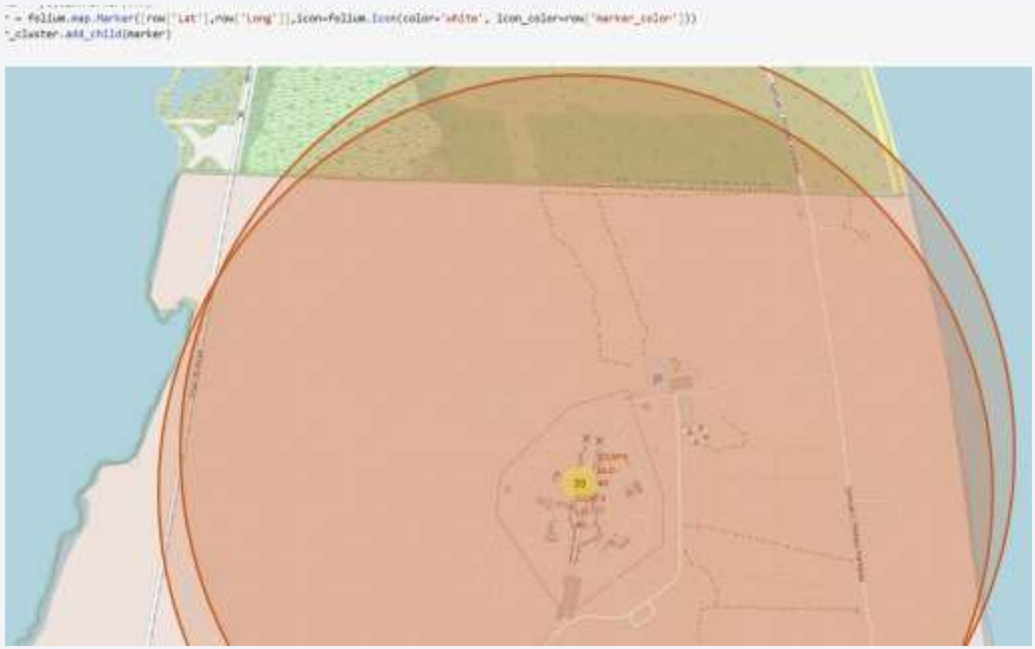
All launch sites on a map

```python
        [row['Lat'],row['Long']],
        # Create an icon as a text label
        icon=DivIcon(
            icon_size=(20,20),
            icon_anchor=(0,0),
            html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % row['Launch Site'],
            )
        )
    site_map.add_child(circle)
    site_map.add_child(marker)
site_map
```

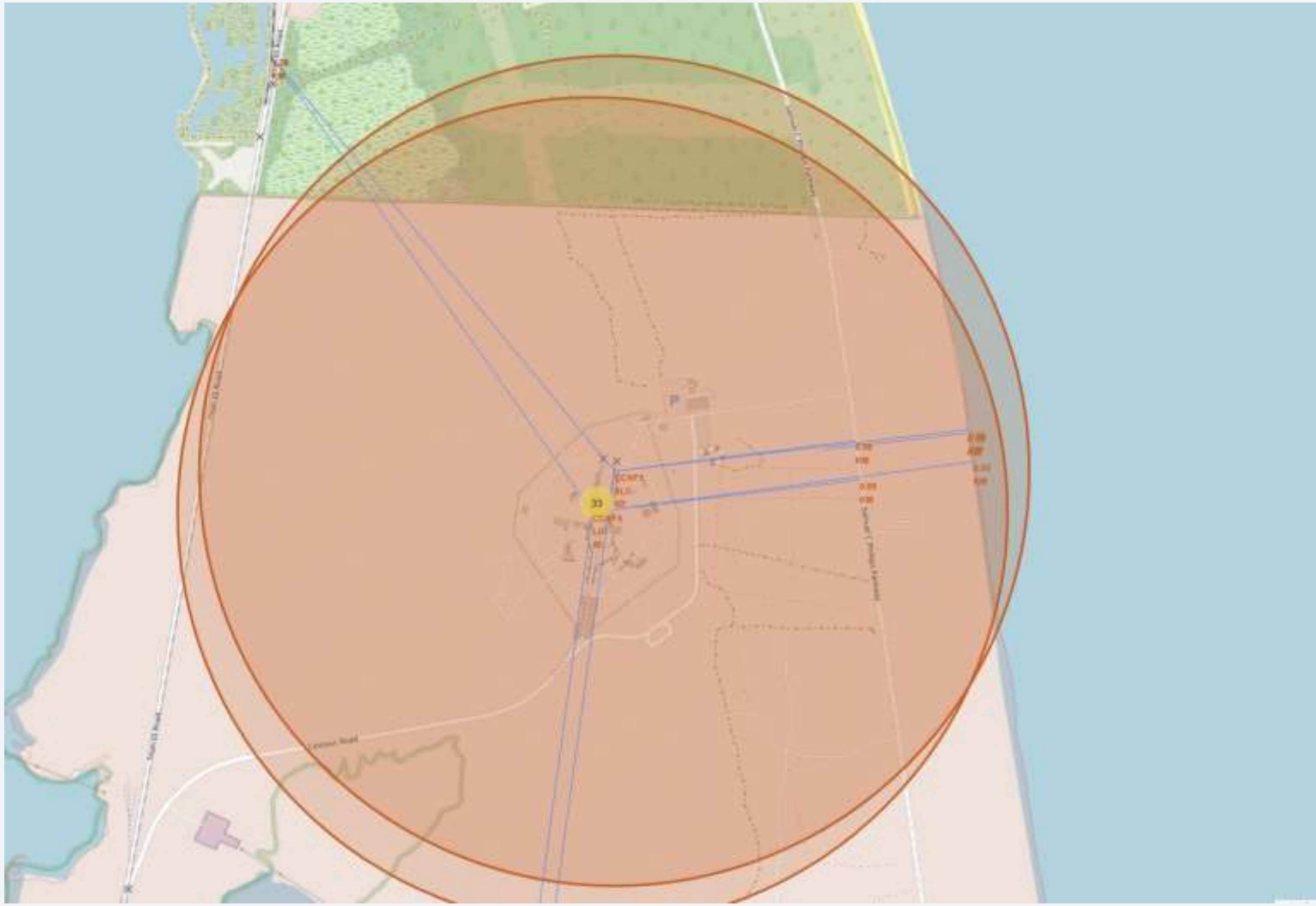
[36]:

Success/failed launches for each site on the map





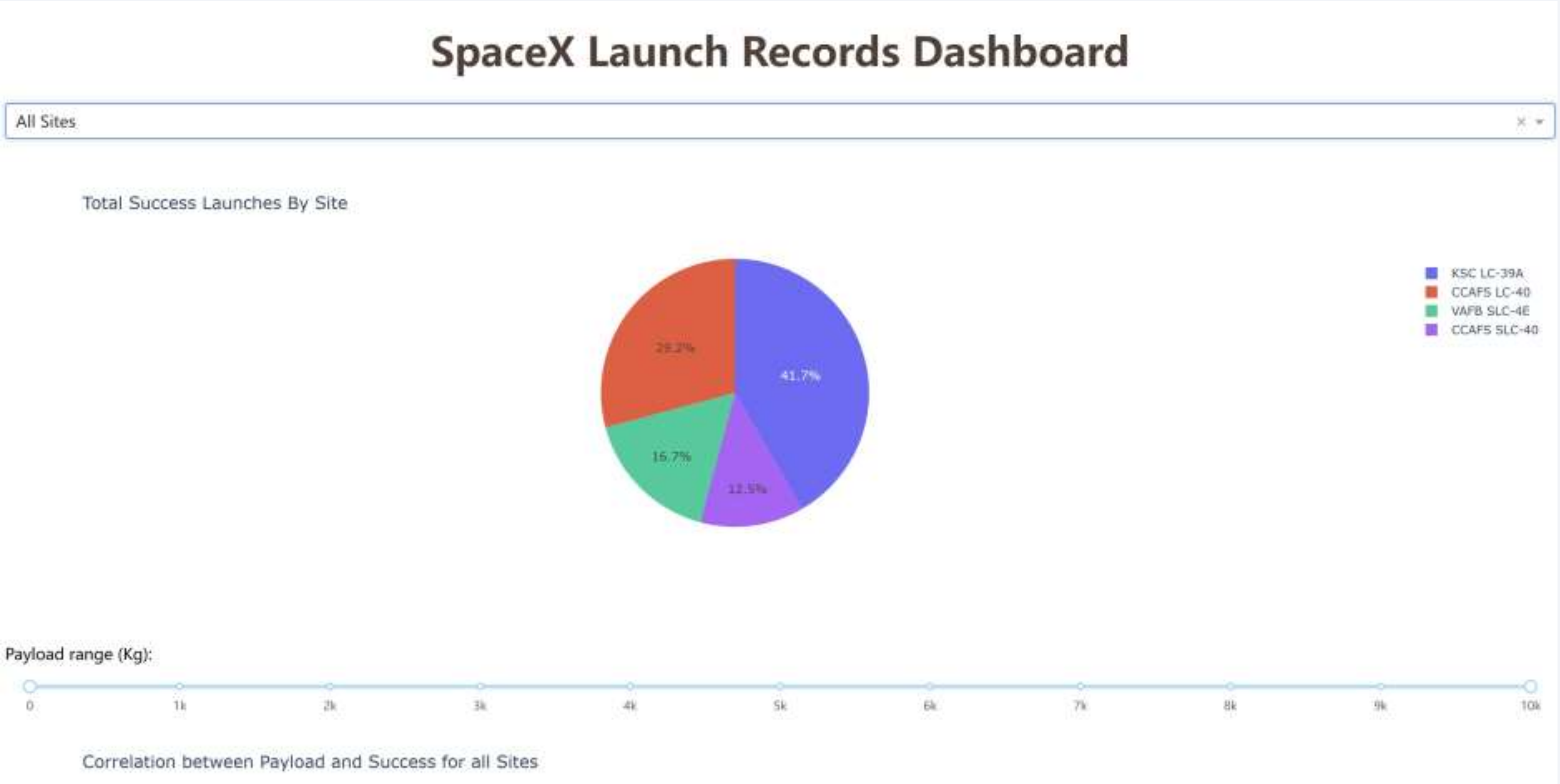
Distances between a launch site to its proximities



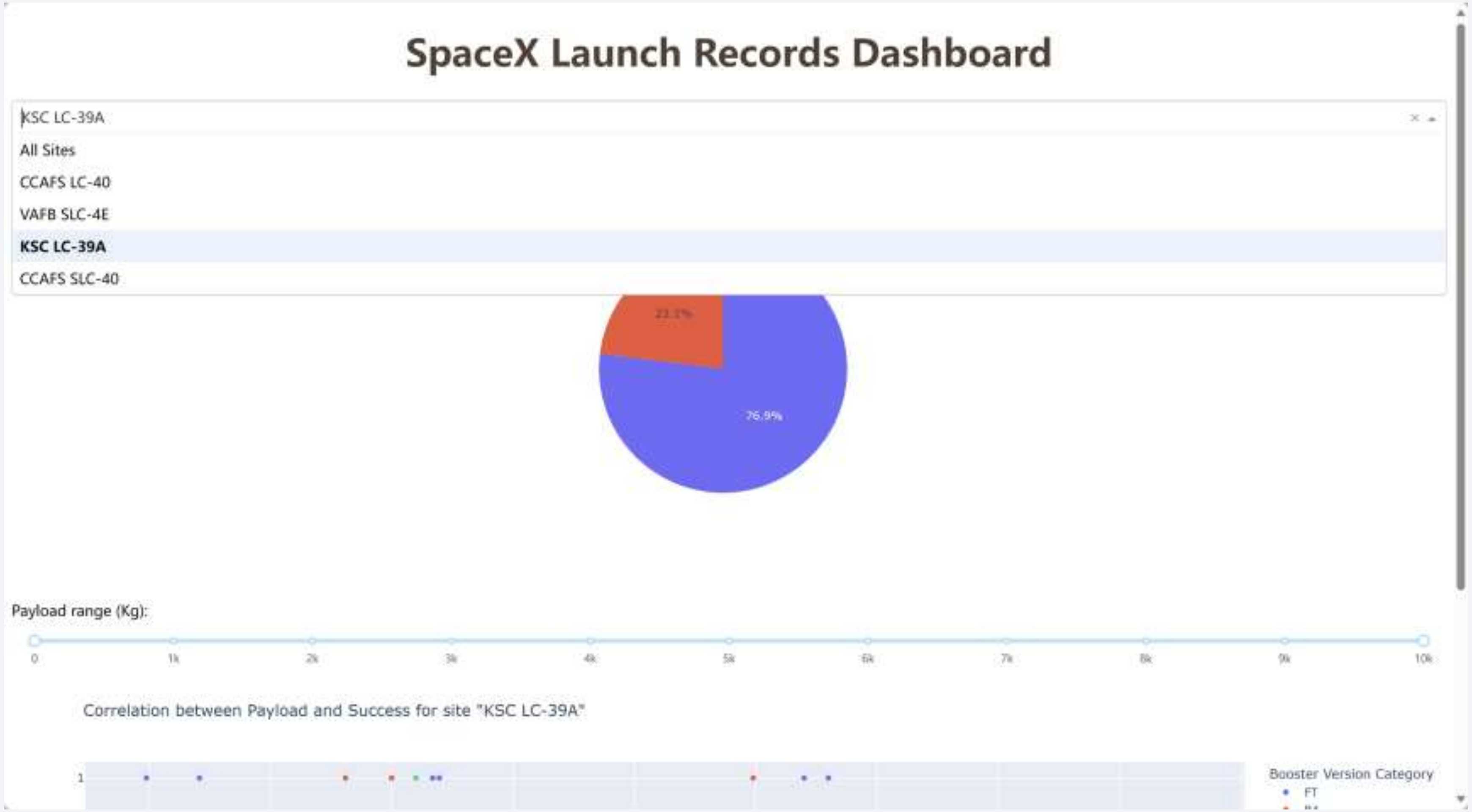
Section 4

Build a Dashboard with Plotly Dash

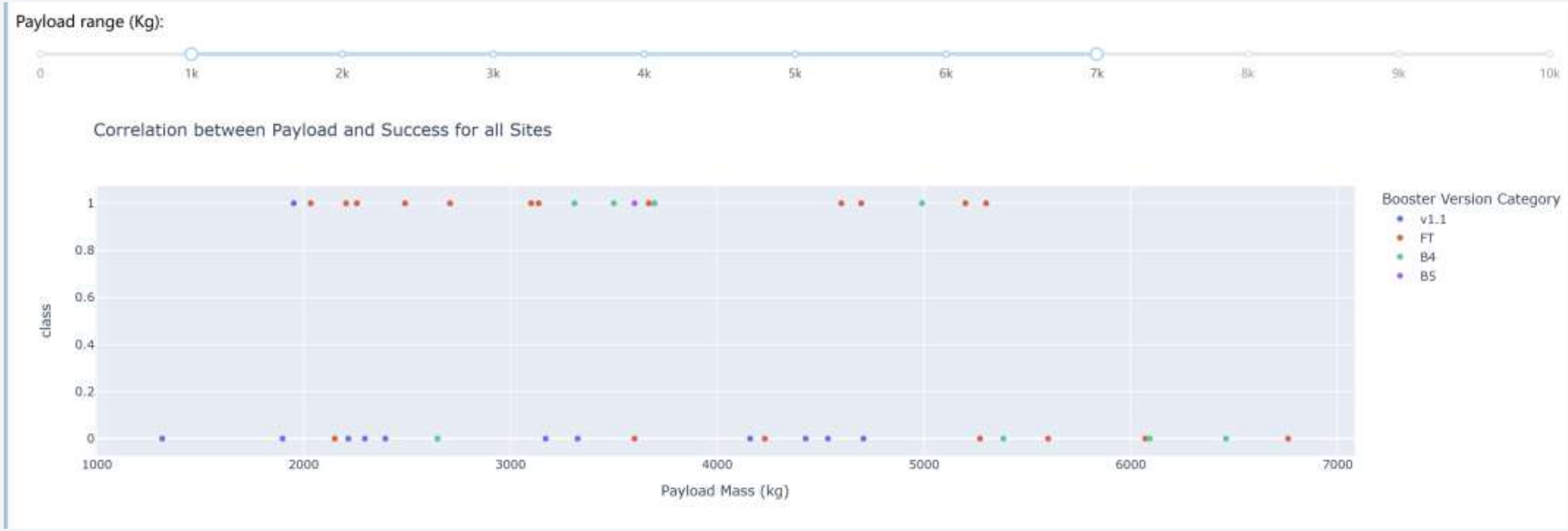
Launch success count for all sites in a piechart



Screenshot of the piechart for the launch site with highest launch success ratio



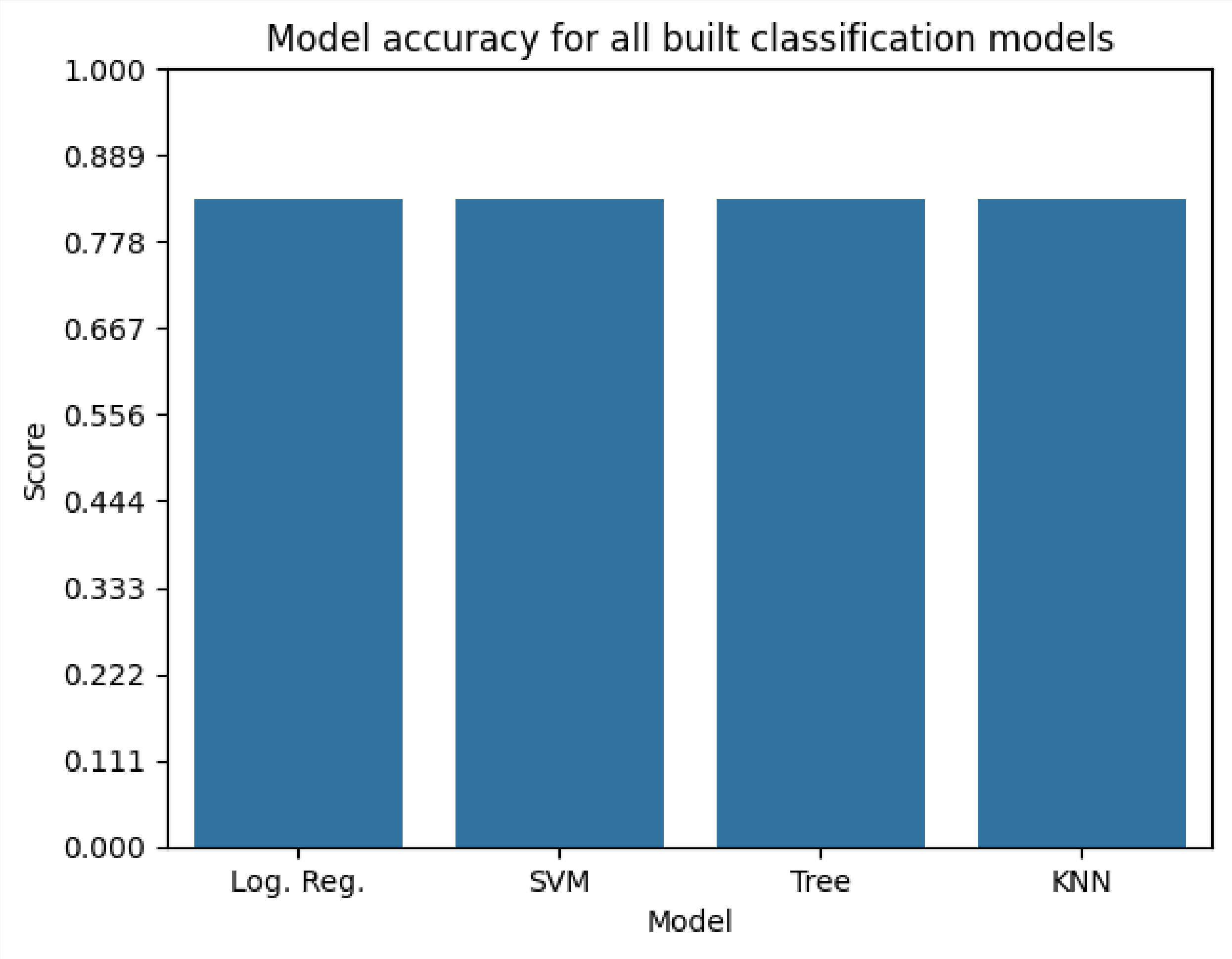
Screenshots of Payload vs. Launch Outcome scatter plot for all sites



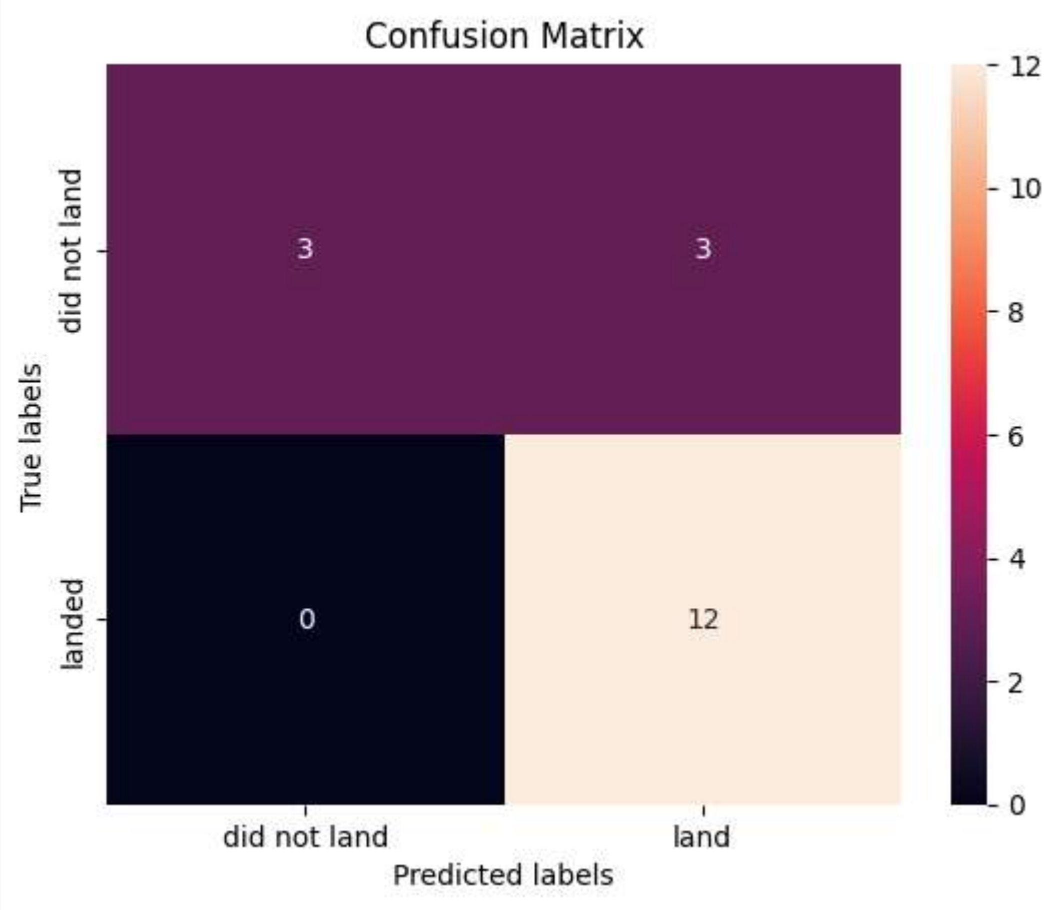
Section 5

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix



Conclusions

1. Launch success rate started to increase in 2013 end with 2020.
2. Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
3. The Decision tree classifier is the best machine learning algorithm for this task.

Appendix

- https://github.com/pagesys/coursera_exam/tree/main

Thank you!