Winning Space Race with Data Science

Ratih Nur Puspita Dewi
22 February 2024

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a dashboard with Plotly Dash
  - Predictive analysis
- **Summary of all results**
  - EDA results
  - Interactive analytics
  - Predictive analysis

Introduction

- Project background and context

  Space X 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 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers
  - What factors determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

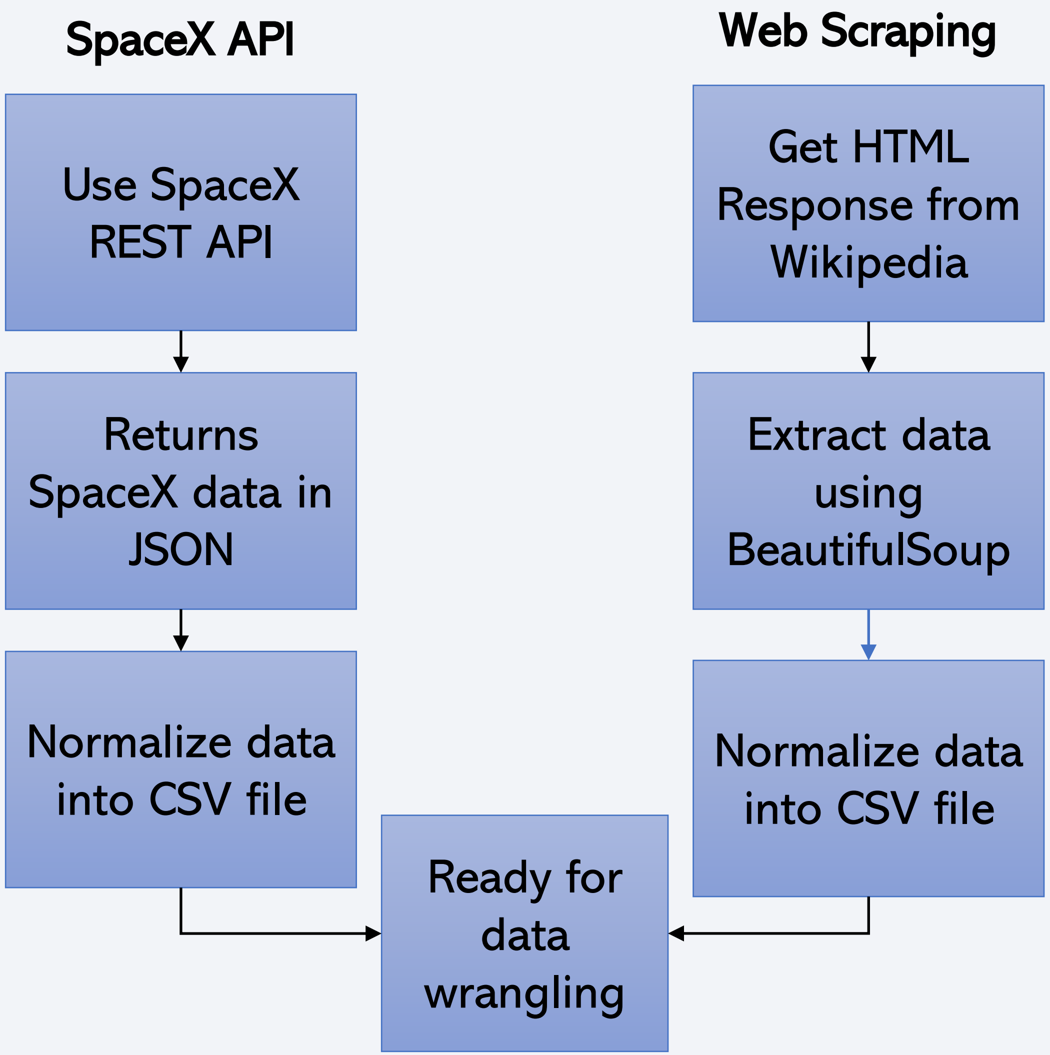
Methodology

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

- How data sets were collected:
  - SpaceX Launch data collected from the SpaceX REST API.
  - The SpaceX API give us data about: rocket used, launch site, landing outcome, and payload.
  - The SpaceX API endpoints, or URL, starts with api.spacexdata.com/v4/
  - Another way for obtaining Falcon 9 Launch data is by scraping Wikipedia website using BeautifulSoup.



Data Collection – SpaceX API

- We used the get requests to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.
- The link to the notebook is

https://github.com/ratihnpd/AppliedDataScienceCapstone/blob/a24e5794d7debcc73eae5d585ccf23beb5d1fade/jupyter-labs-spacex-data-collection-api.ipynb

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

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

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

```
# Create a data from launch_dict
df = pd.DataFrame(launch_dict)
```

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = df[df['BoosterVersion']!='Falcon 1']
```

Data Collection - Scraping

- Applied web scrapping to scrap Falcon 9 launch records from Wikipedia using BeautifulSoup.
- Then, parsed the table and converted it into a pandas dataframe.
- The link to the notebook is

https://github.com/ratihnpd/AppliedDataScienceCapstone/blob/a24e5794d7debcc73eae5d585ccf23beb5d1fade/jupyter-labs-webscraping.ipynb

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

soup = BeautifulSoup(data, 'html.parser')
```

```
temp = first_launch_table.find_all('th')

for col in range(len(temp)):
    name = extract_column_from_header(temp[col])
    if (name is not None and len(name)>0):
        column_names.append(name)
    else:
        pass
```

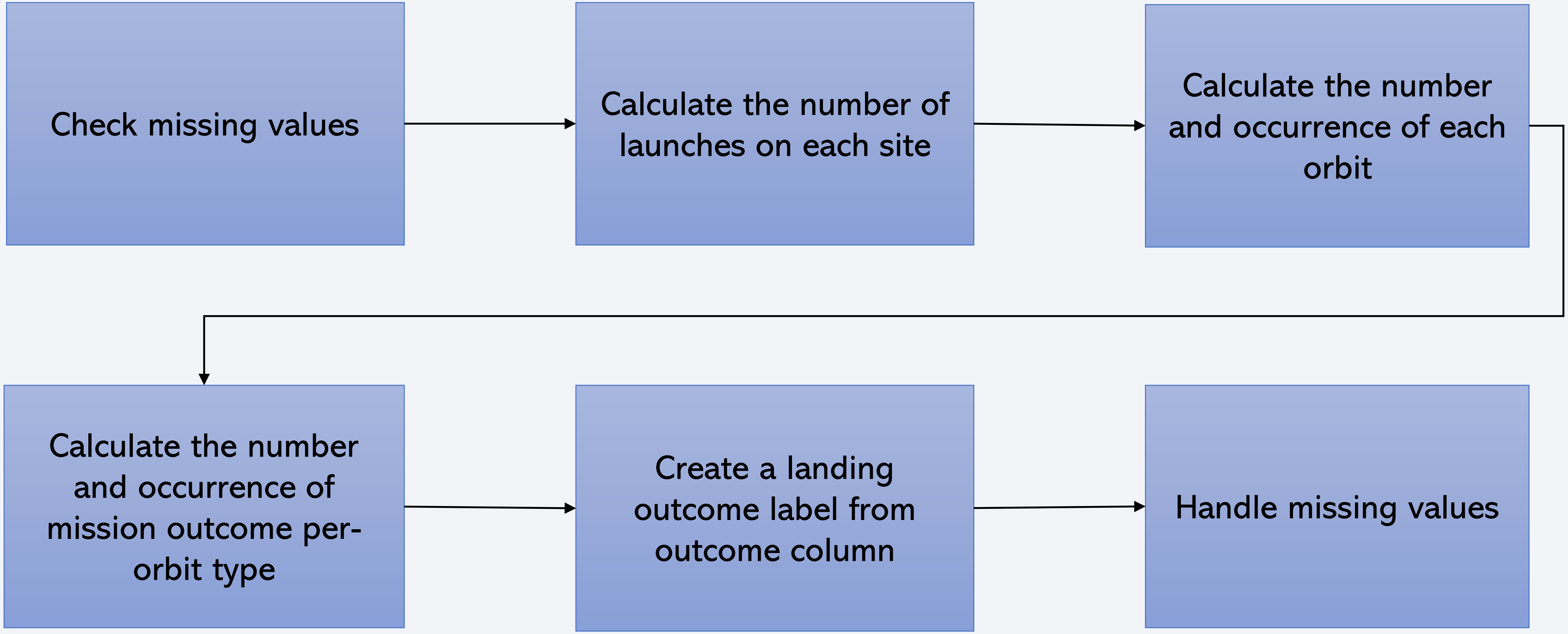
```
[ ]  launch_dict = dict.fromkeys(column_names)

     # Remove an irrelvant column
     del launch_dict['Date and time ( )']

     # Let's initial the launch_dict with each v
     launch_dict['Flight No.'] = []
     launch_dict['Launch site'] = []
     launch_dict['Payload'] = []
     launch_dict['Payload mass'] = []
     launch_dict['Orbit'] = []
     launch_dict['Customer'] = []
     launch_dict['Launch outcome'] = []
     # Added some new columns
     launch_dict['Version Booster']=[]
     launch_dict['Booster landing']=[]
     launch_dict['Date']=[]
     launch_dict['Time']=[]
```

```
[ ]  extracted_row = 0
     #Extract each table
     for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowh
        # get table row
         for rows in table.find_all("tr"):
             #check to see if first table heading is as number corresponding to lau
             if rows.th:
                 if rows.th.string:
                     flight_number=rows.th.string.strip()
                     flag=flight_number.isdigit()
             else:
                 flag=False
```

```
[ ]  df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items()
```

```
[ ]  df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling



Link to the notebook:
https://github.com/ratihnpd/AppliedDataScienceCapstone/blob/a24e5794d7debcc73eae5d585ccf23beb5d1fade/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- We explored the data to see the relationship between variables by visualizing it using scatter plot. What we explore is relationship between flight number & launch site, payload & launch site, flight number & orbit type.
- We use bar chart to visualize the success rate of each orbit type and line chart to visualize the launch success yearly trend.
- Link to the notebook:

  https://github.com/ratihnpd/AppliedDataScienceCapstone/blob/a24e5794d7debcc73eae5d585ccf23beb5d1fade/jupyter_labs_eda_dataviz_ipynb_jupyterlite.ipynb

EDA with SQL

- Applied EDA with SQL to get insight from the data, for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS).
  - The average payload mass carried by booster version F9 v1.1.
  - The total number of successful and failure mission outcomes.
  - The failed landing outcomes in drone ship, their booster version, and launch site names.
- Notebook link: https://github.com/ratihnpd/AppliedDataScienceCapstone/blob/4ea7b49b1206613ff52e06ebdee9994391fcf023/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Marked all launch sites, and added map objects such as markers and circles to mark the success or failure of launches for each site on the folium map.
- Use color-labeled marker clusters to identified launch outcomes (success or failure).
- Calculated the distances between a launch site to coastlines, railways, highways, and cities.

- Notebook link:

https://github.com/ratihnpd/AppliedDataScienceCapstone/blob/4ea7b49b1206613ff52e06ebdee9994391fcf023/lab_jupyter_launch_site_location_jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- Built an interactive dashboard with Plotly dash that include pie chart and scatter plot.
- We plotted pie charts to show the total launches by a certain launch sites.
- We plotted scatter plot to show the relationship between Launch Outcome and Payload Mass (Kg) for the different booster version.

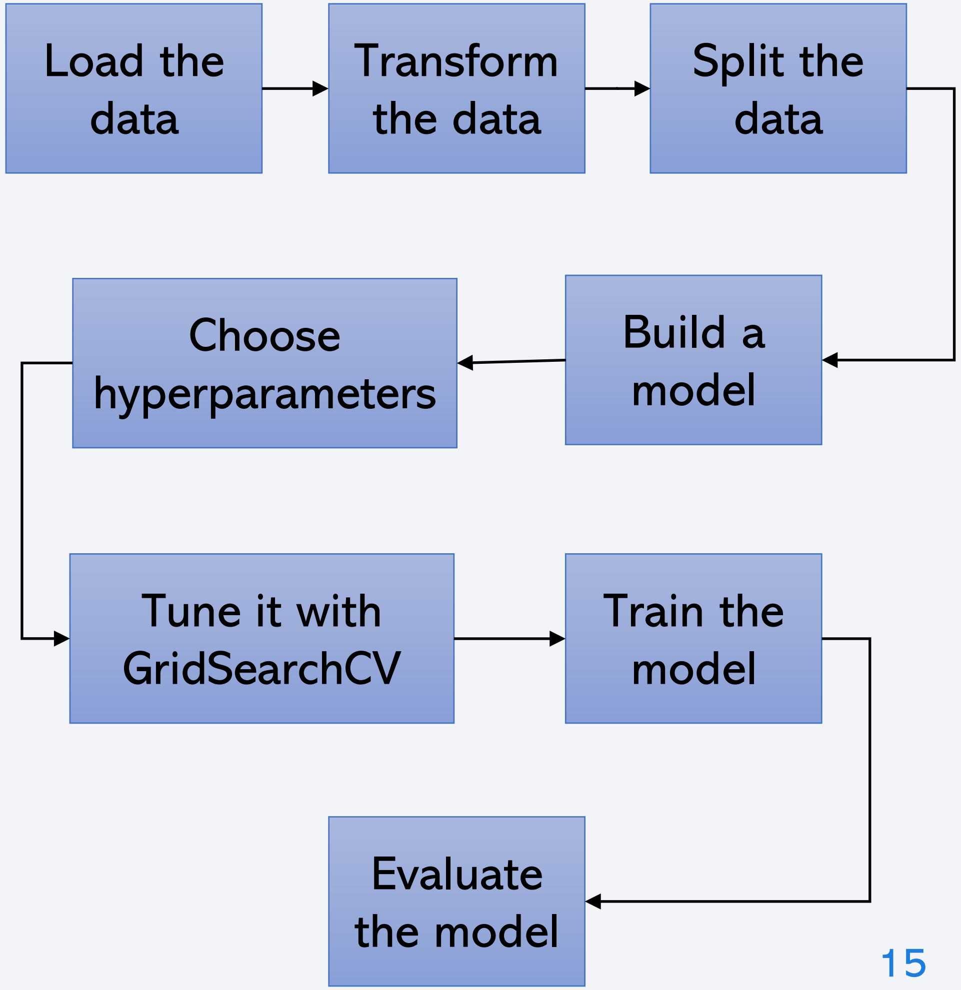
- Notebook link:

https://github.com/ratihnpd/AppliedDataScienceCapstone/blob/a24e5794d7debcc73eae5d585ccf23beb5d1fade/spacex_dash_app.py

Predictive Analysis (Classification)

- Load the data using numpy and pandas, transformed the data, split the data into training and testing data.
- Built different machine learning models and tune different hyperparameters using GridSearchCV.
- Evaluate our models and found the best performing classification model.
- Notebook link:

https://github.com/ratihnpd/AppliedDataScienceCapstone/blob/c80c7938cb8557fb1cc4b034d6a80fec36a75a3b/SpaceX_Machine_Learning_Prediction_Part_5_jupyterlite.ipynb



Results

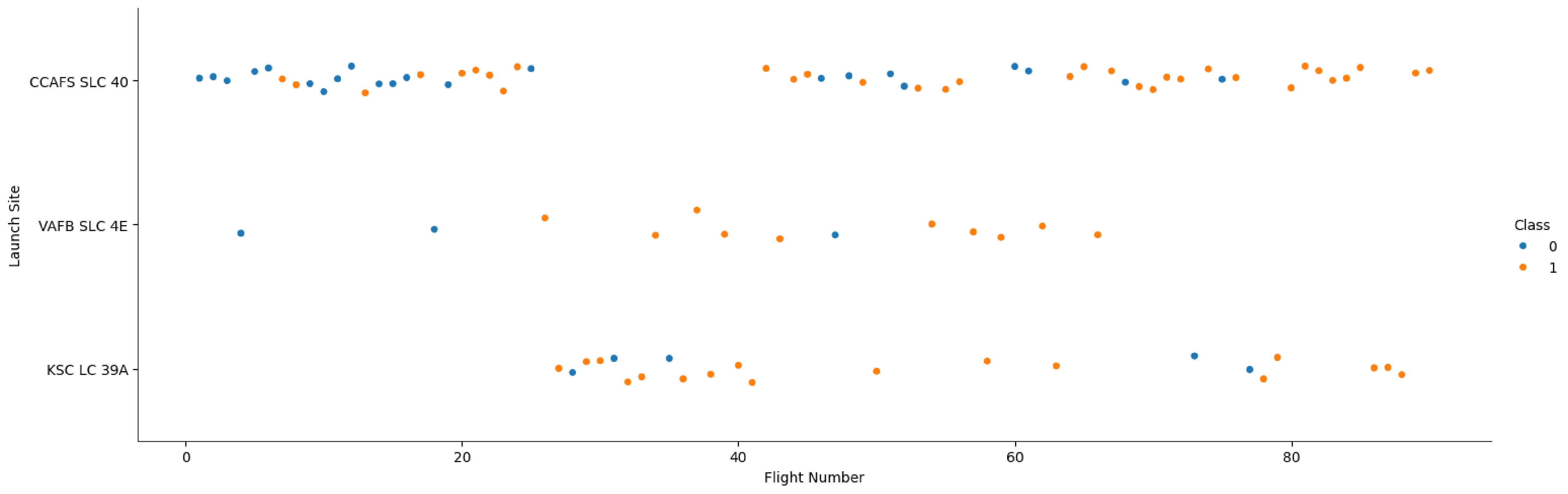
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

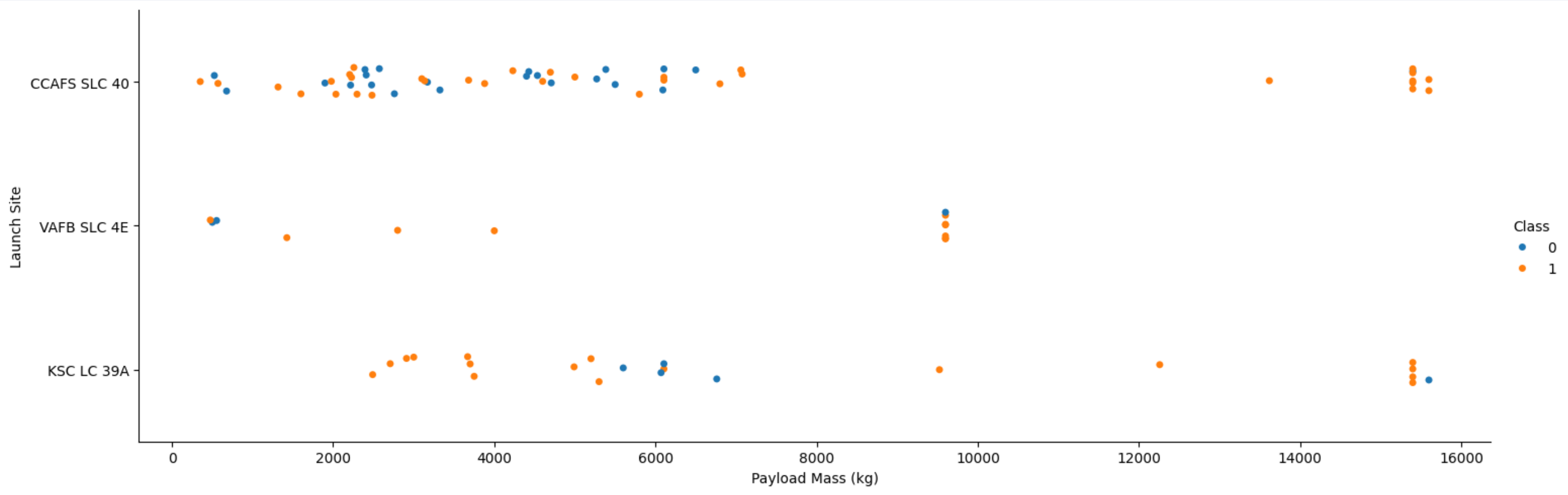
Flight Number vs. Launch Site

- From the plot, we found that the larger the flight number at a launch site, the greater the success rate at a launch site.



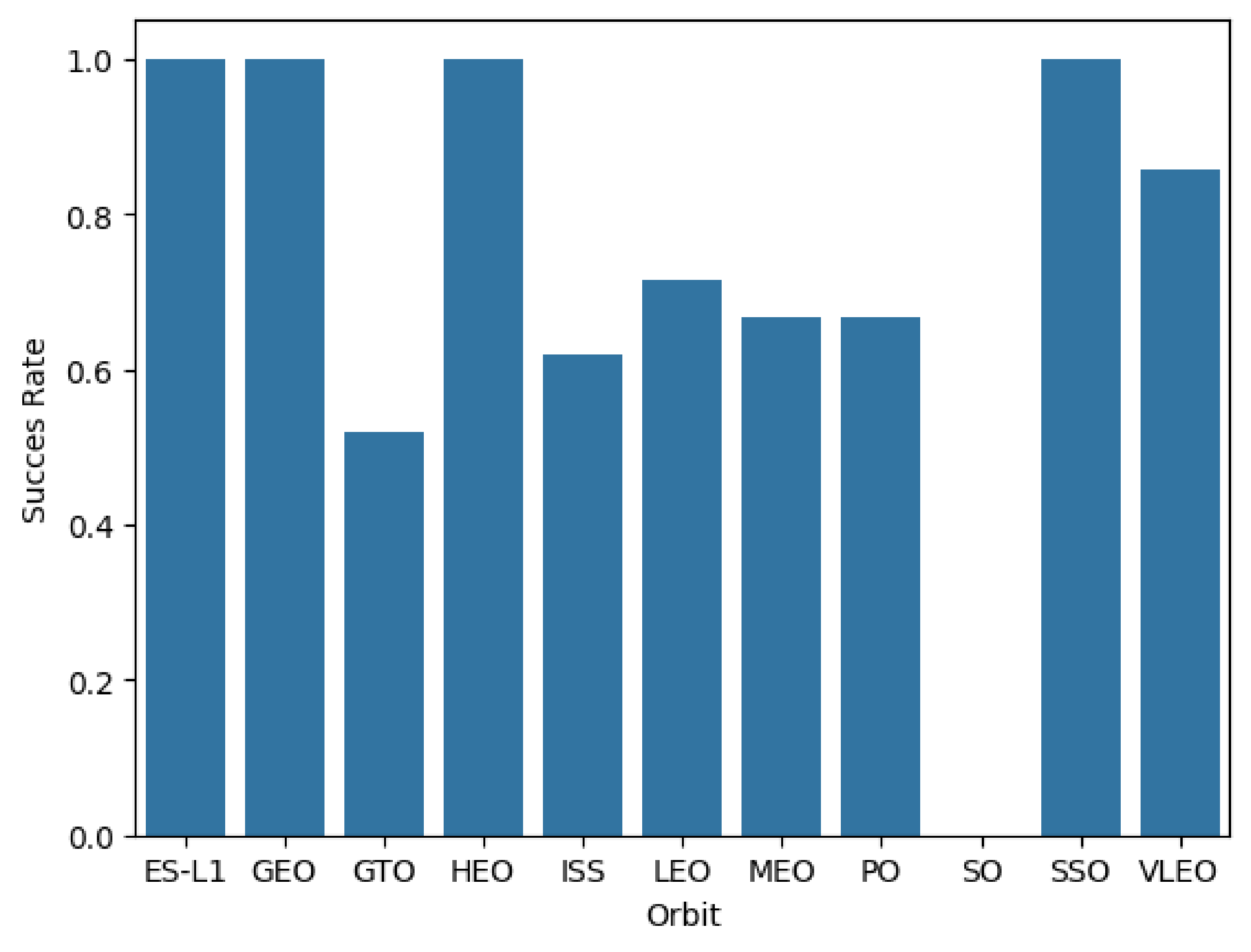
Payload vs. Launch Site

- The greater the payload mass for launch site CCAFS SLC 40, the higher the success rate for the rocket launch.



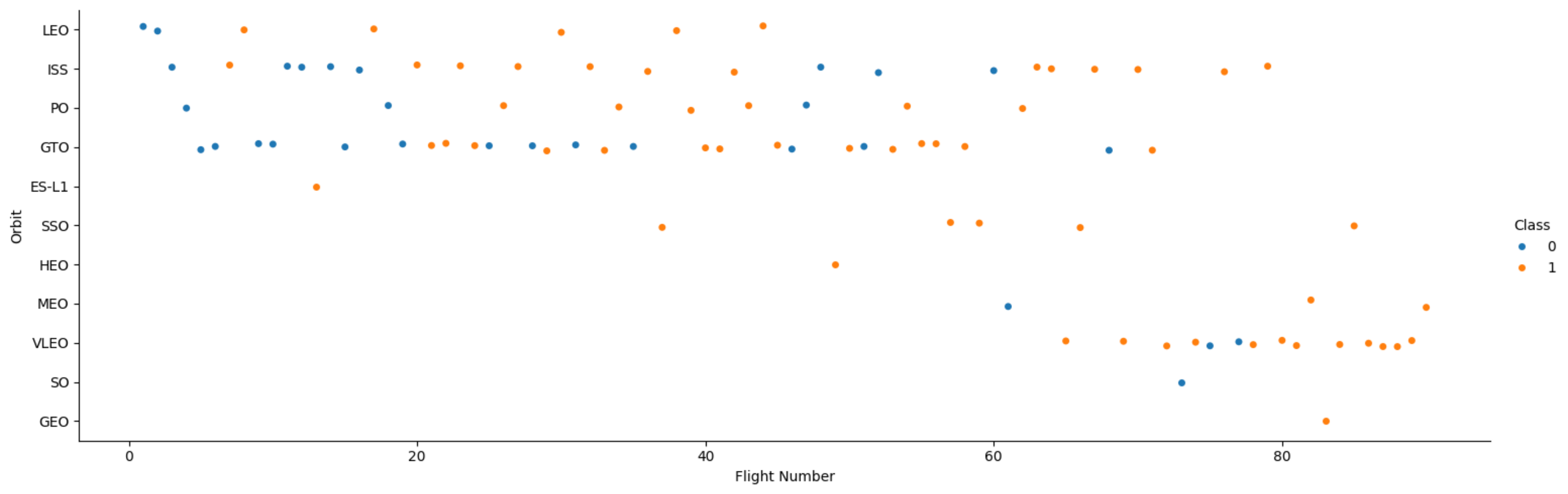
Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, and SSO orbit types have higher success rates.



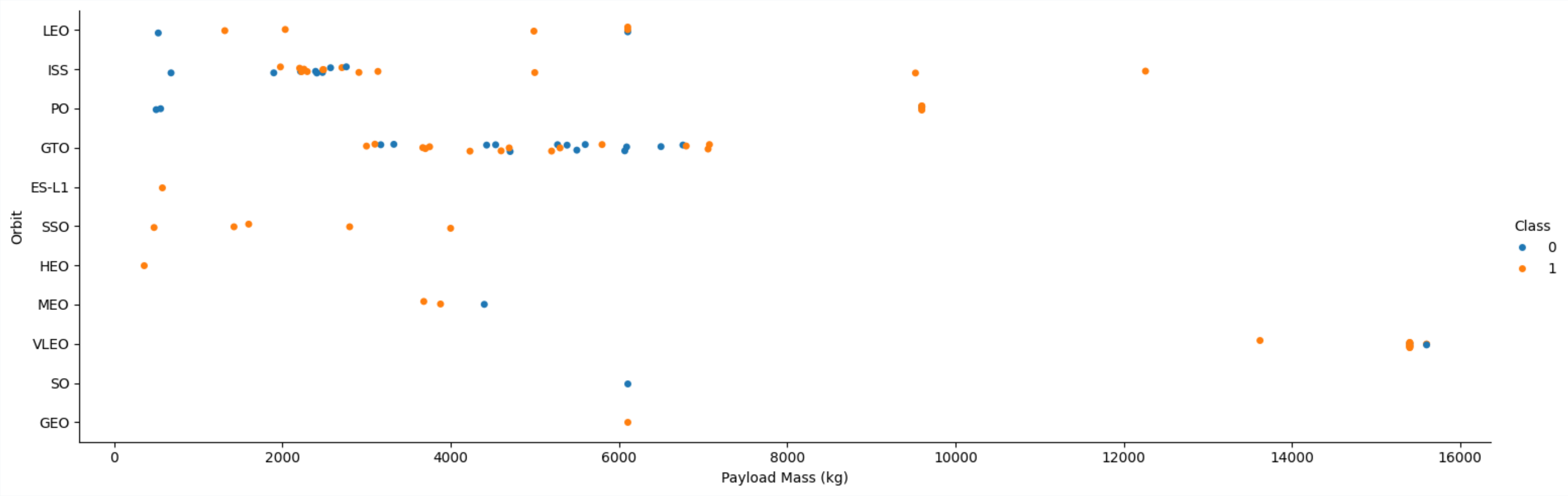
Flight Number vs. Orbit Type

- We observe that in the LEO orbit, success is related to the flight number whereas in the GTO orbit, there is no relationship between flight number and the orbit.



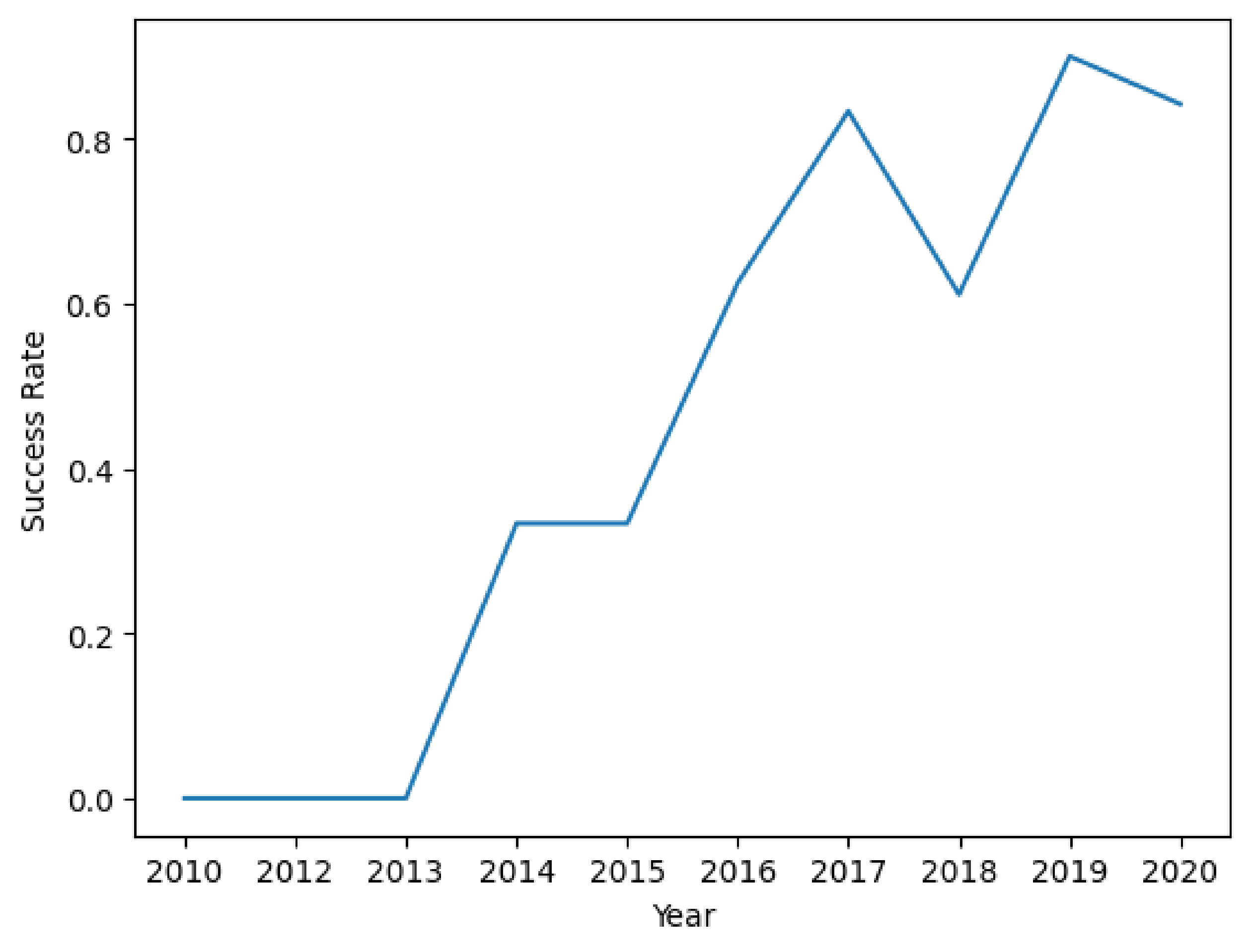
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landings are more for PO, LEO, and ISS orbits. And with light payloads, the successful landing is more for SSO orbit.



Launch Success Yearly Trend

- From the plot, we can observe that the success rate since 2013 kept on increasing till 2020.



All Launch Site Names

- We used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

```
%%sql

SELECT DISTINCT(Launch_Site) FROM SPACEXTABLE;
```

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

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

Launch Site Names Begin with 'CCA'

- We used the query below to display 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

- The total payload mass carried by boosters from NASA (CRS) is 45596.

```
%%sql

SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE
    WHERE Customer = 'NASA (CRS)';
```

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is 2928.4.

```
%%sql

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

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

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- The dates of the first successful landing outcome on ground pad was 22^{nd} December 2015

```
%%sql

SELECT MIN(Date) FROM SPACEXTABLE
    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

- Use the **WHERE** clause to filter for boosters that have successfully landed on a drone ship and apply the **AND** condition to determine a successful landing with a payload mass greater than 4000 but less than 6000.

```
%%sql

SELECT DISTINCT Booster_Version FROM SPACEXTABLE
    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

- From the table total number of successful mission outcomes is 100 and for failed mission outcomes is 1.
- We use **GROUP BY** to group the same outcomes.

```
%%sql

SELECT Mission_Outcome, Count(Mission_Outcome) FROM SPACEXTABLE
    GROUP BY Mission_Outcome;
```

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

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

Boosters Carried Maximum Payload

- We determined the booster that has carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

```
%%sql
SELECT DISTINCT Booster_Version FROM SPACEXTABLE
    WHERE Payload_Mass__Kg_ = (SELECT MAX(Payload_Mass__Kg_) FROM SPACEXTABLE);
```

```
 * 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

- We used a combination of the **WHERE** clause and **AND** conditions to filter for failed landing outcomes in drone ships for the year 2015.

```
%%sql

SELECT substr(Date,6,2) AS Month, Landing_Outcome,
    Booster_Version, Launch_Site FROM SPACEXTABLE
    WHERE substr(Date,0,5)='2015'
    AND Landing_Outcome = 'Failure (drone ship)';
```

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

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

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

- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcomes in descending order.

```
%%sql

SELECT Date, Landing_Outcome,
    COUNT(Landing_Outcome) AS Total_Landing FROM SPACEXTABLE
    WHERE Date BETWEEN '2010-6-04' AND '2017-03-20'
    GROUP BY Landing_Outcome
    ORDER BY Total_Landing DESC;
```

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

Date	Landing_Outcome	Total_Landing
2012-05-22	No attempt	10
2016-04-08	Success (drone ship)	5
2015-01-10	Failure (drone ship)	5
2015-12-22	Success (ground pad)	3
2014-04-18	Controlled (ocean)	3
2013-09-29	Uncontrolled (ocean)	2
2015-06-28	Precluded (drone ship)	1

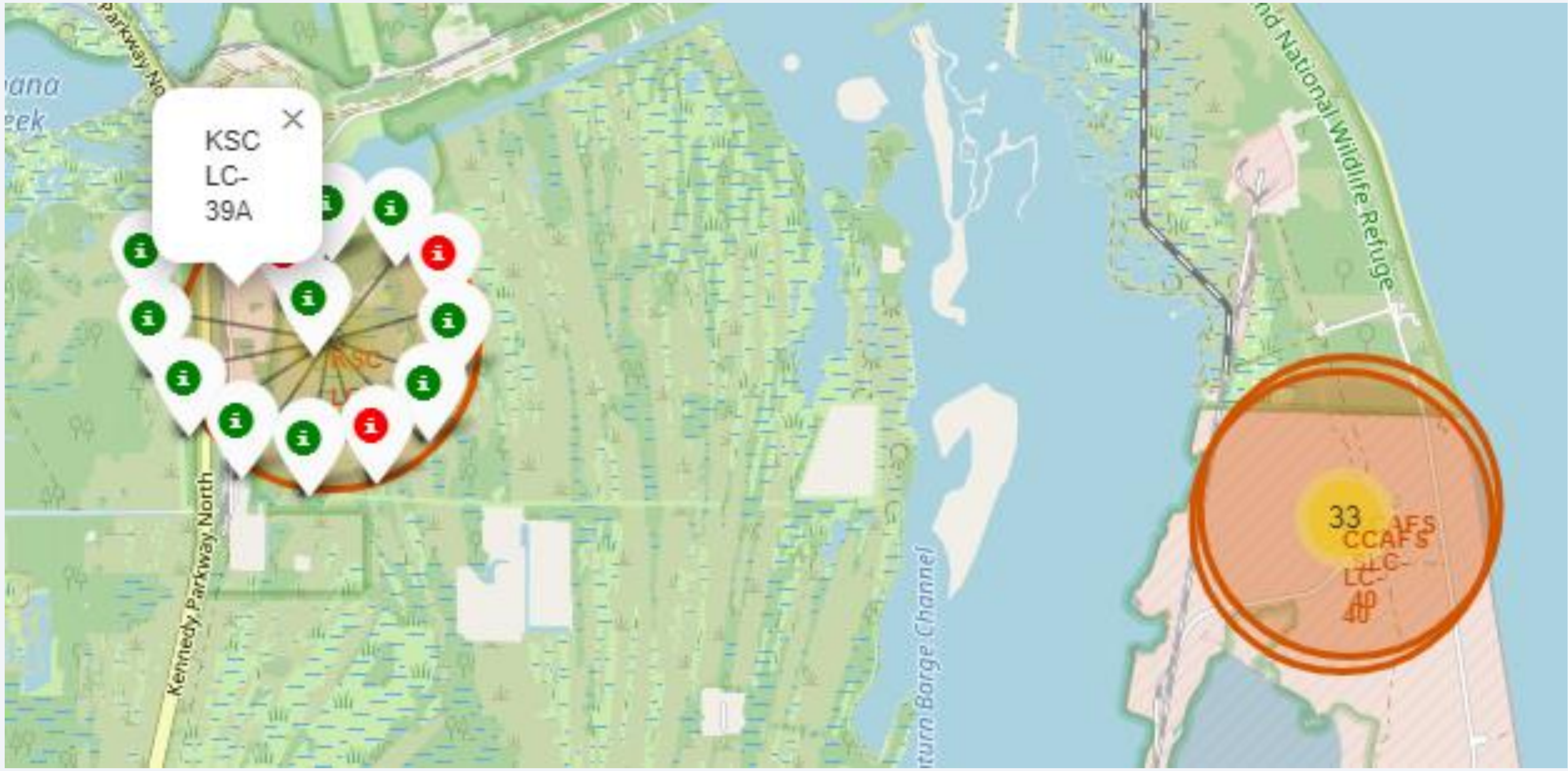
Section 3

Launch Sites Proximities Analysis

All launch sites global map markers

- We can see, that all launch sites are on the coast.

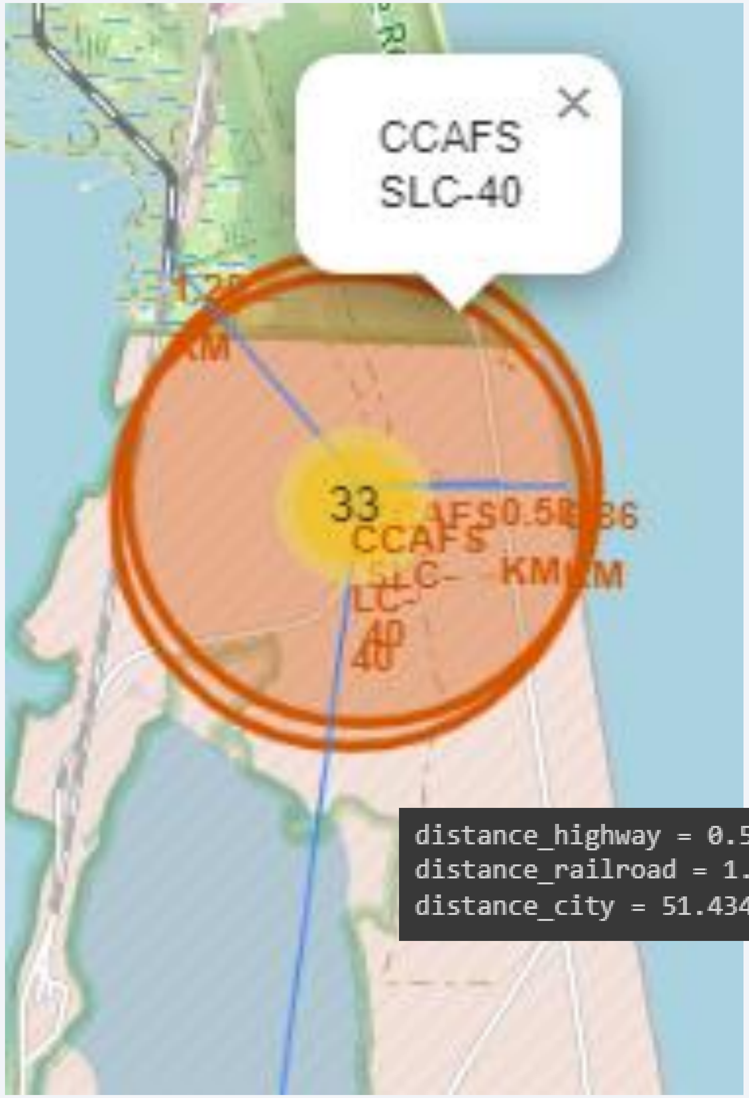
Markers showing launch sites with color labels



- The green marker shows successful launches and the red marker shows failed launches.

Launch Site distance to its proximities

- Launch sites are in close proximity to equator to minimize fuel consumption by using Earth's ~ 30km/sec eastward spin to help spaceships get into orbit.
- Launch sites are in close proximity to coastline so they can fly over the ocean during launch for safety reasons.
- Launch sites are in close proximity to highways, which allows for easily transport required people and property.
- Launch sites are in close proximity to railways, which allows transport for heavy cargo.
- Launch sites are not in close proximity to cities, which minimizes danger to population dense areas.

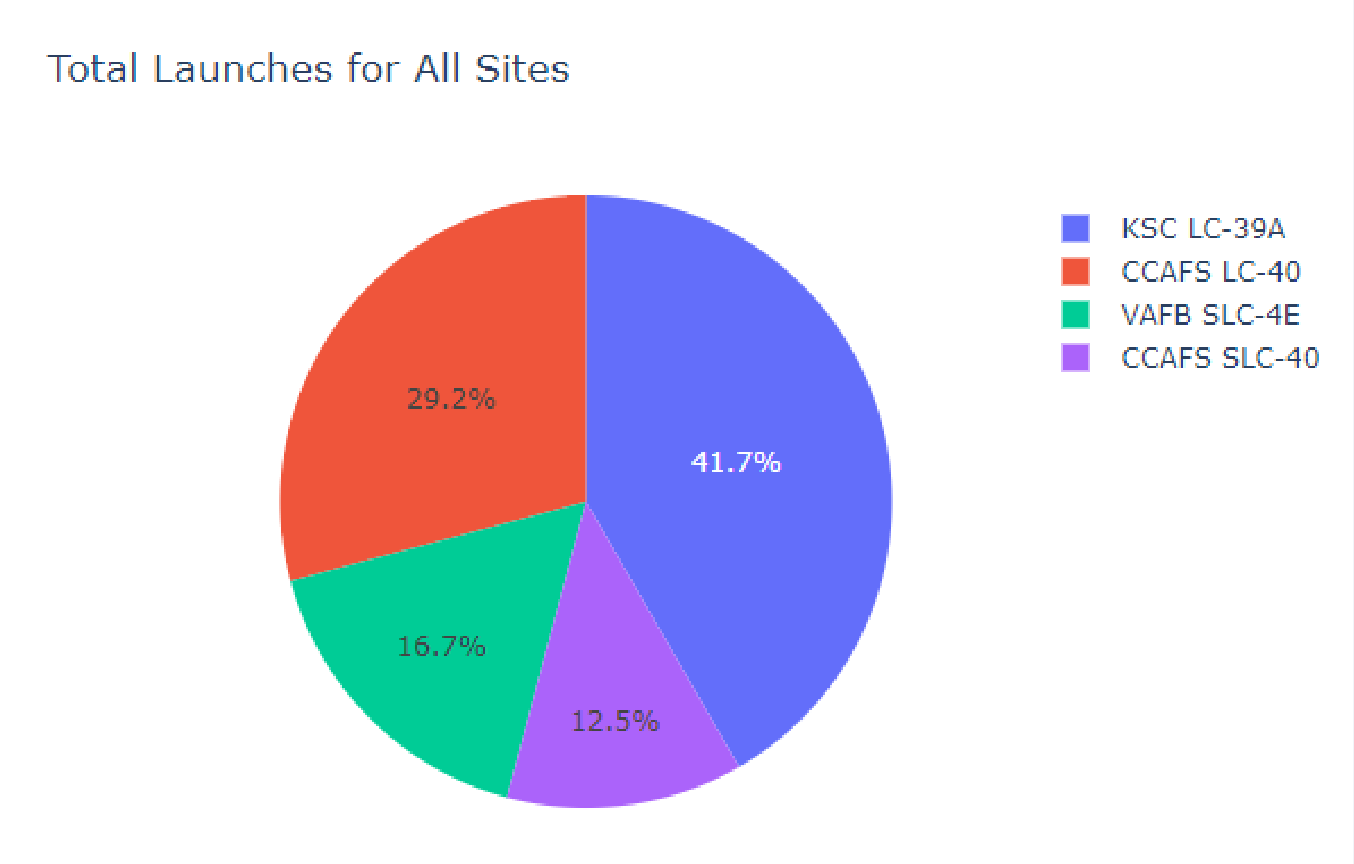



```
distance_highway = 0.5834695366934144  km
distance_railroad = 1.2845344718142522  km
distance_city = 51.43416999517233  km
```

Section 4

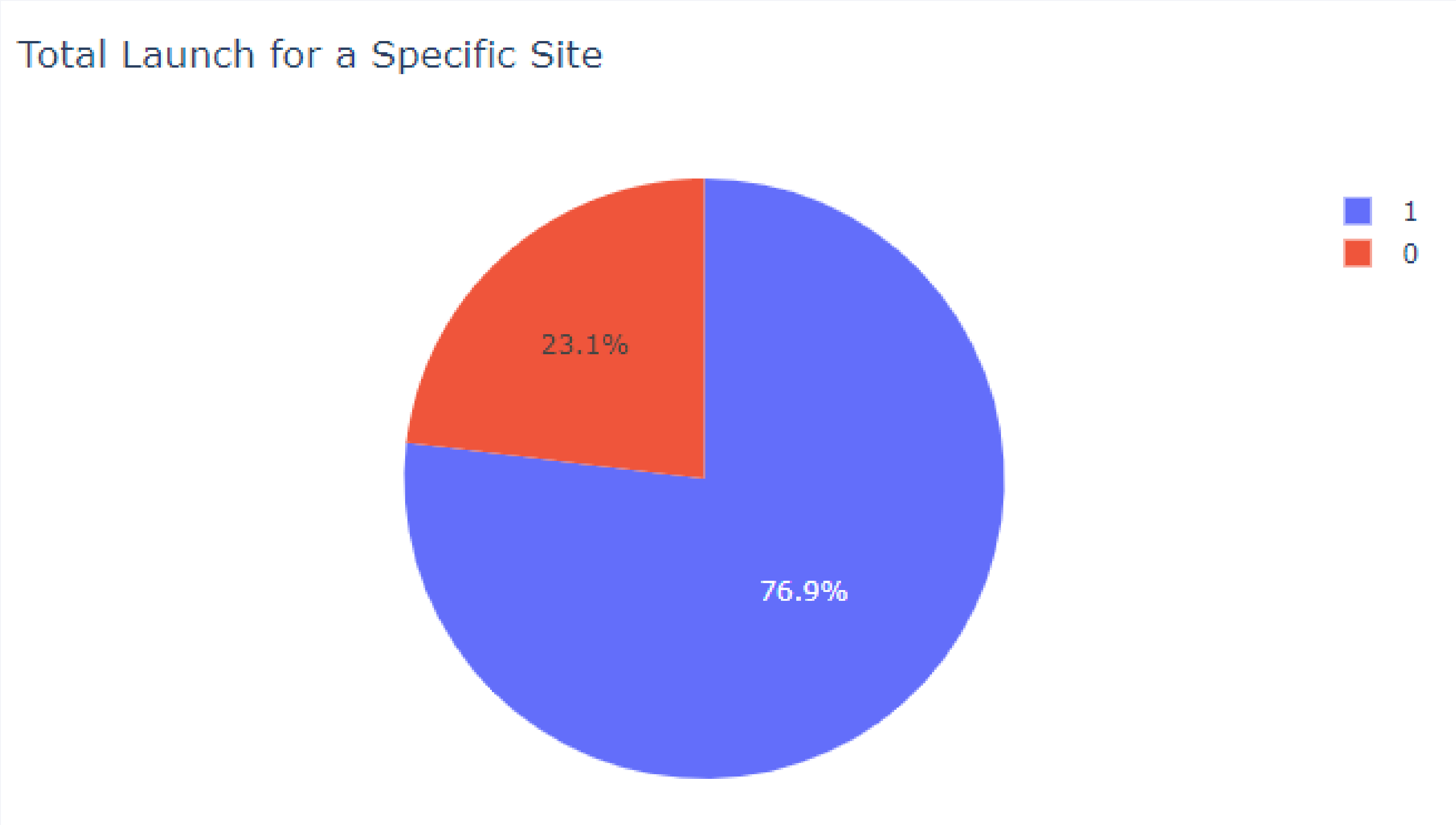
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



- We can see that KSC LC-39A has the most successful launches from all sites.

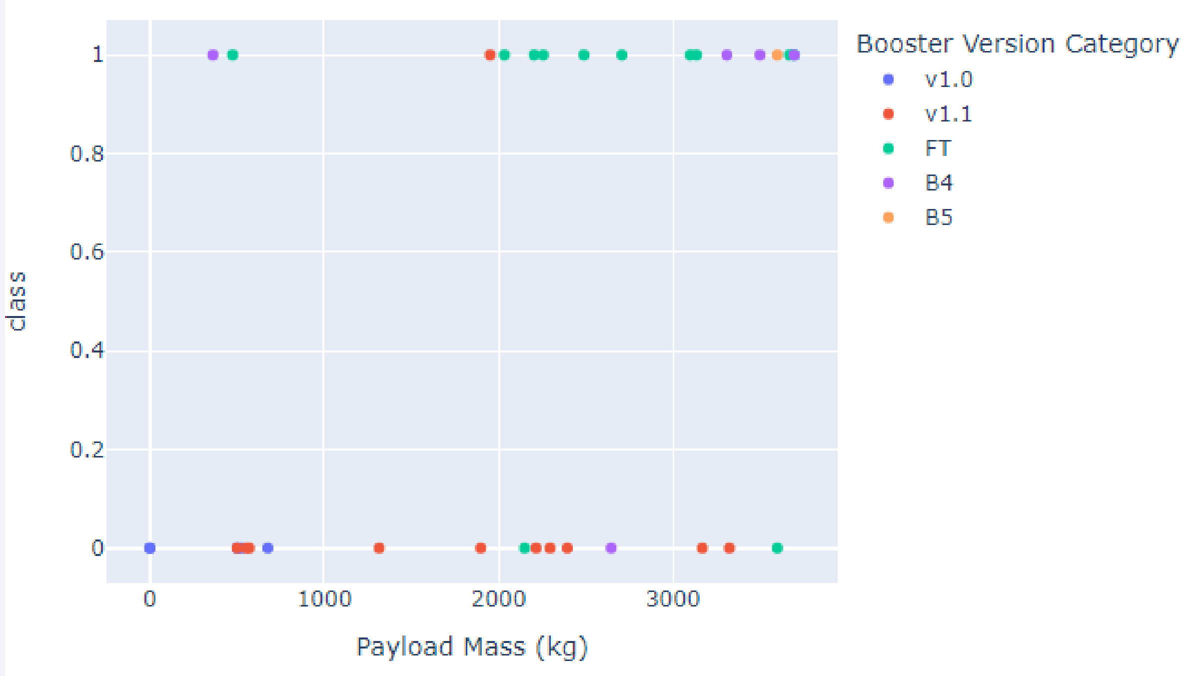
Pie chart showing the Launch site with the highest launch success ratio



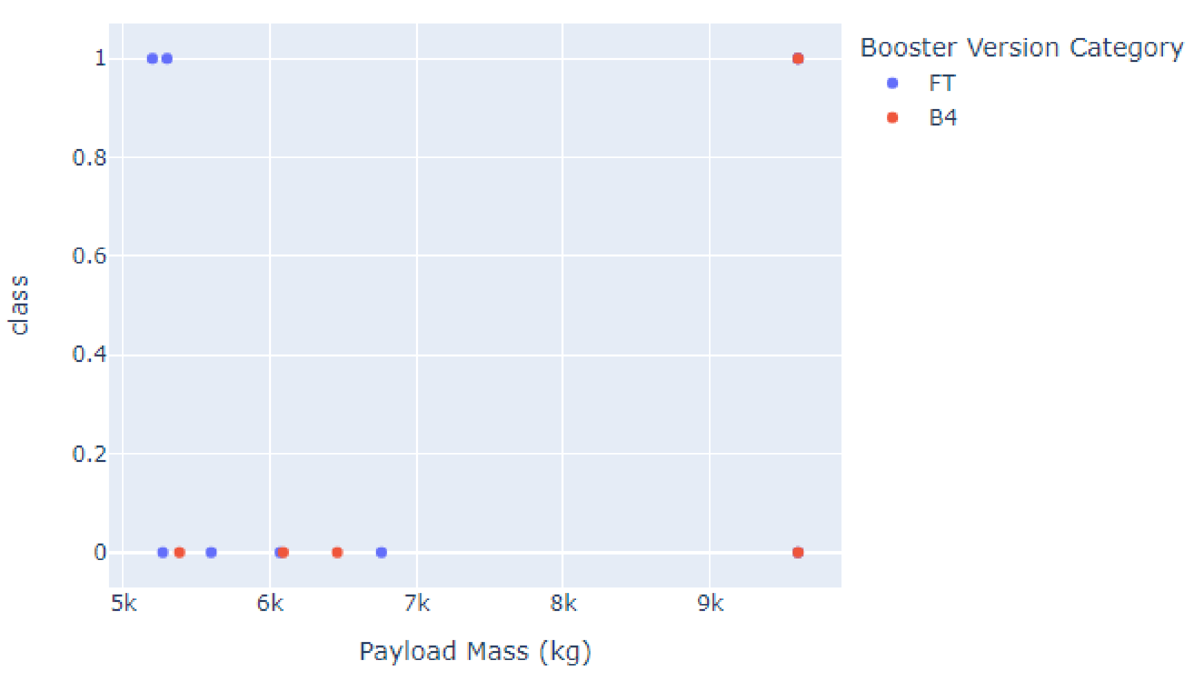
- KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate.

Scatter plot of Payload vs Launch Outcome for all sites

Low Weighted Payload 0kg-4000kg



Heavy Weighted Payload 5000kg-10000kg



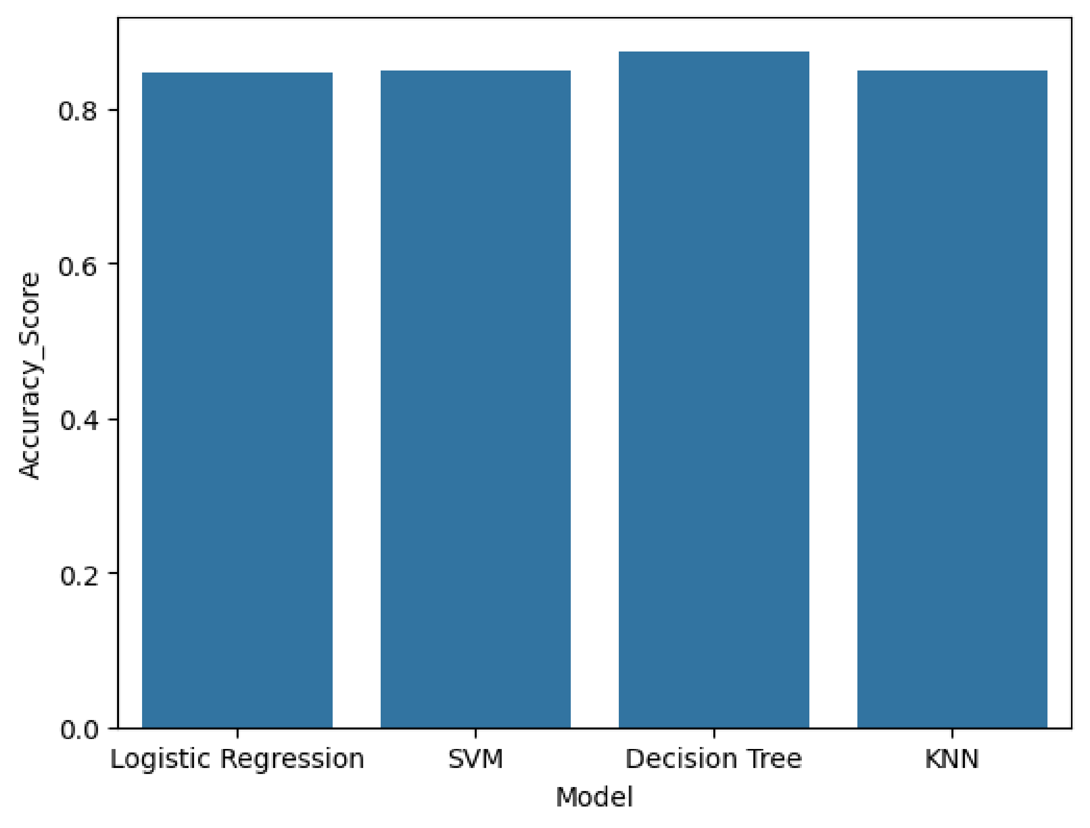
- We can see that the success rate for low-weighted payloads is higher than the heavy heavy-weighted payloads.

Section 5

Predictive Analysis (Classification)

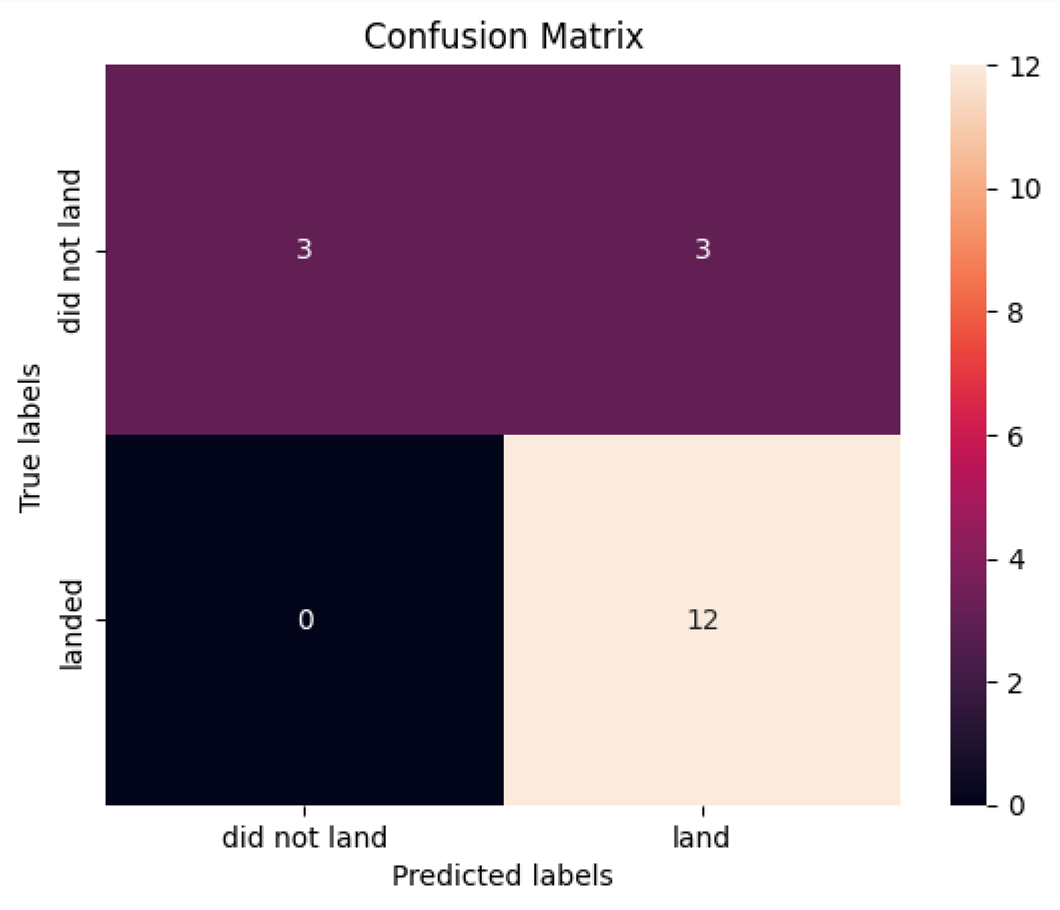
Classification Accuracy

- The decision tree classifier is the model with the highest classification accuracy.



Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives. i.e., an unsuccessful landing is marked as a successful landing by the classifier.



Conclusions

We can conclude that:

- The larger the flight number at a launch site, the greater the success rate at a launch site.
- The launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, and VLEO had the highest success rate.
- KSC LC-39A had the most successful launches of any site.
- The Decision tree classifier is the best machine-learning algorithm for this task.

Thank you!