IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Shrikala Kadam
28th August 2024

Outline

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

Executive Summary

- Summary of methodologies
  - This project involved comprehensive process starting with data collection through APIs and web scraping, followed by data wrangling to clean and prepare the data.
  - Exploratory Data Analysis (EDA) using Python was conducted, complemented by data visualization techniques to uncover insights. The analysis also involved creating interactive visual analytics with Folium.
  - Finally, machine learning models were applied to make predictive analyses, showcasing the integration of these methodologies for thorough data-driven decision-making.
- Summary of all results
  - This project analyzed SpaceX Falcon 9 launch data to predict the likelihood of successful first-stage landings. By leveraging data from multiple sources, including launch outcomes and weather conditions, we performed extensive data wrangling and exploratory analysis.
  - Using machine learning models, we achieved a high accuracy in predicting landing success, providing valuable insights into the factors influencing mission outcomes. The interactive visualizations further highlighted key trends and patterns, aiding in data-driven decision-making for future launches.

Introduction

- Project background and context

SpaceX Falcon 9 aims to make space travel more affordable by reusing rocket boosters. To make this possible, it's important to understand what helps or hinders the successful landing of these boosters. This project uses past launch data to find patterns and predict outcomes, helping SpaceX improve the success of future missions.

- Problems you want to find answers

o What factors affect whether the Falcon 9 booster lands successfully?

o How can we use machine learning to predict successful landings?

o What can the data tell us about improving future launches?

o How does weather impact landing success?

o Can we use predictions to reduce risks and improve launch planning?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

Data was gathered from SpaceX's public launch records, Wikipedia and other relevant databases using APIs and web scraping. This included information on launch details, rocket specs, weather conditions, and landing results.

- Perform data wrangling:

Data was cleaned and organized by removing duplicates, handling missing values, and standardizing formats. Cleaned data was processed by converting categories into numbers, normalizing values, and creating new features. The data was then split into training and testing sets for machine learning model development.

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models:

Selecting and training different classification models, fine-tuned these models by adjusting hyperparameters through techniques like Grid Search. Evaluated the models' accuracy, precision, recall, and F1-score to determine their effectiveness in predicting successful landings.

Data Collection

- **Describe how data sets were collected.**
- Data collection was done using get request to the SpaceX API.
- Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
- Then we cleaned the data, checked for missing values and fill in missing values where necessary.
- In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
- 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

- We used the get request to the SpaceX API to collect data, clean the data and did some data wrangling and formatting.

- Link for Notebook:

https://github.com/Shrikala28/IBMDS1/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

In [5]:
```
# Takes the dataset and uses the cores column to call the API and append the data to the lists
def getCoreData(data):
    for core in data['cores']:
            if core['core'] != None:
                response = requests.get("https://api.spacexdata.com/v4/cores/"+core['core']).json()
                Block.append(response['block'])
                ReusedCount.append(response['reuse_count'])
                Serial.append(response['serial'])
            else:
                Block.append(None)
                ReusedCount.append(None)
                Serial.append(None)
            Outcome.append(str(core['landing_success'])+' '+str(core['landing_type']))
            Flights.append(core['flight'])
            GridFins.append(core['gridfins'])
            Reused.append(core['reused'])
            Legs.append(core['legs'])
            LandingPad.append(core['landpad'])
```

Now let's start requesting rocket launch data from SpaceX API with the following URL:

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

Check the content of the response

In [8]:
```
print(response.content)
```

b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://imag
es2.imgbox.com/94/f2/NN6Ph45r_o.png","large":"https://images2.imgbox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"la
unch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.c

Data Collection - Scraping

- Web Scraping was applied to webscrap Falcon 9 launch records with BeautifulSoup.
- The table was parsed and converted it into a pandas dataframe.
- Link for Notebook:

https://github.com/Shrikala28/IBMDS1/blob/main/jupyter-labs-webscraping.ipynb

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
print(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 the external reference link the end of this lab

In [9]:
```
# 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")
print(html_tables)
```

```
[<table class="col-begin" role="presentation">
```

Data Wrangling

- Calculated the number of launches at each site, and the number and occurrence of each orbits.
- Created landing outcome label from outcome column and exported the results to csv.
- Link for Notebook:

https://github.com/Shrikala28/IBMDS1/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

```
In [7]: # Apply value_counts on Orbit column
        df.Orbit.value_counts()

Out[7]: GTO      27
        ISS      21
        VLEO     14
        PO        9
        LEO       7
        SSO       5
        MEO       3
        ES-L1     1
        HEO       1
        SO        1
        GEO       1
        Name: Orbit, dtype: int64
```

TASK 3: Calculate the number and occurence of mission outcome of the orbits

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then assign it t
landing_outcomes.

```
In [8]: # landing_outcomes = values on Outcome column
        landing_outcomes = df.Outcome.value_counts()
        print(landing_outcomes)

        True ASDS      41
        None None      19
        True RTLS      14
        False ASDS      6
```

EDA with Data Visualization

- Explored the data by visualizing the relationship between:

o Flight number and launch Site

o Payload and launch site

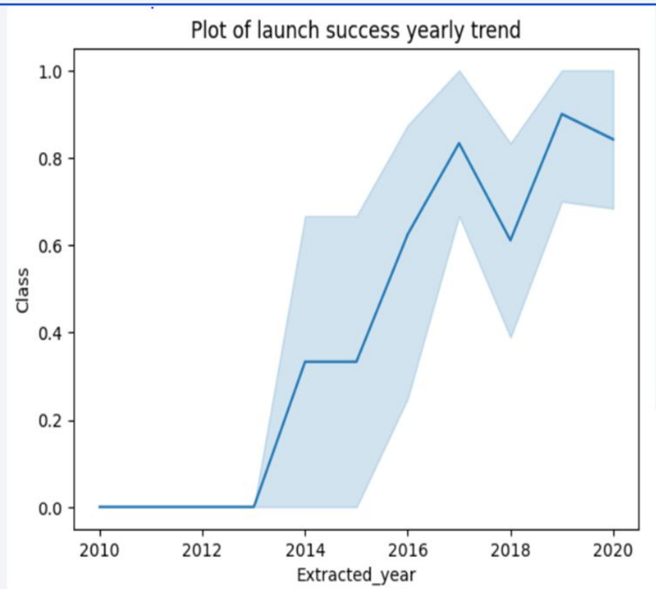
o Success rate of each orbit type

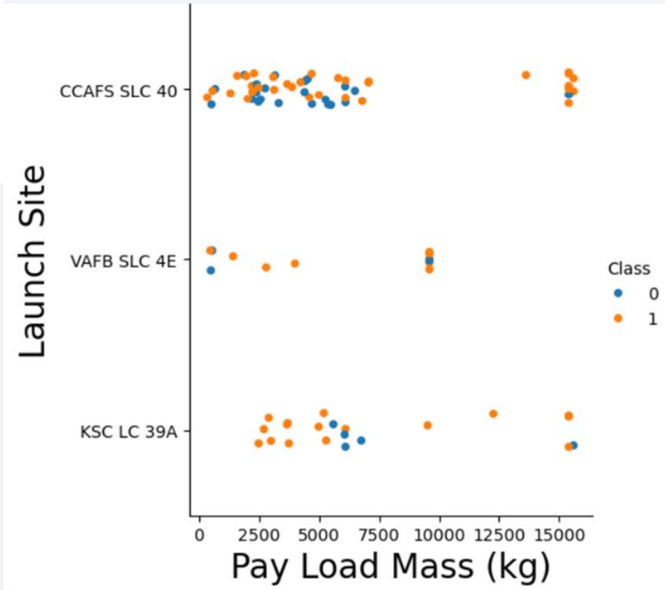
o Flight number and orbit type

o The launch success yearly trend.

- Link for Notebook:

https://github.com/Shrikala28/IBMDS1/blob/main/edadataviz.ipynb





EDA with SQL

- Explored the data using SQL commands in python within the Jupyter Environment.
- EDA was done using SQL commands and these following findings were noted:
    - 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.
- Link for Notebook:
- https://github.com/Shrikala28/IBMDS1/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- All launch sites were added as well as map objects such as markers, circles, lines to mark the success or failure of launches for each site were added on the folium map.
- Assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, launch sites having relatively high success rate were identified.
- Calculated the distances between a launch site to its proximities. Some of the questions that were answered are :
  - o Are launch sites near railways, highways and coastlines.
  - o Do launch sites keep certain distance away from cities.
- Link for Notebook:
- https://github.com/Shrikala28/IBMDS1/blob/main/lab_jupyter_launch_site_location%20(1).ipynb

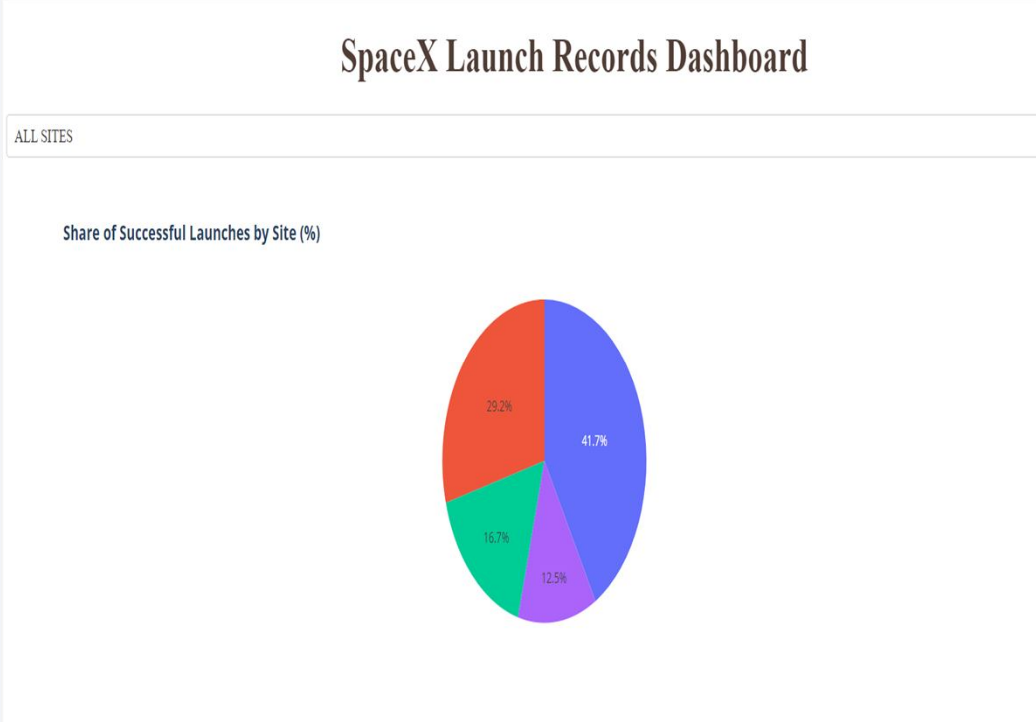
Build a Dashboard with Plotly Dash

An interactive dashboard with Plotly is built including:

- Dropdown menu for selecting launch sites
- Pie charts displaying success rate.
- Scatter chart displaying launch site, payload mass, success/failure
- Range slider for selecting range of payload mass (kg).

This dashboard is built for analyzing SpaceX launch records features:

- Site with largest successful launches.
- Site with highest launch success rate
- Payload range(s) with highest launch success rate
- Payload range(s) with lowest launch success rate
- F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) with highest launch success rate.



- Link for Notebook:
- https://github.com/Shrikala28/IBMDS1/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- The data was loaded using Numpy and Pandas, transformed and split into training and testing.
- Different machine learning models were built and tuning was done using different hyperparameters.
- Accuracy as metric was used for the model and the model was improved using feature engineering and algorithm tuning.
- The Best performing classification model was found by this process.
- Link for Notebook:

https://github.com/Shrikala28/IBMDS1/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

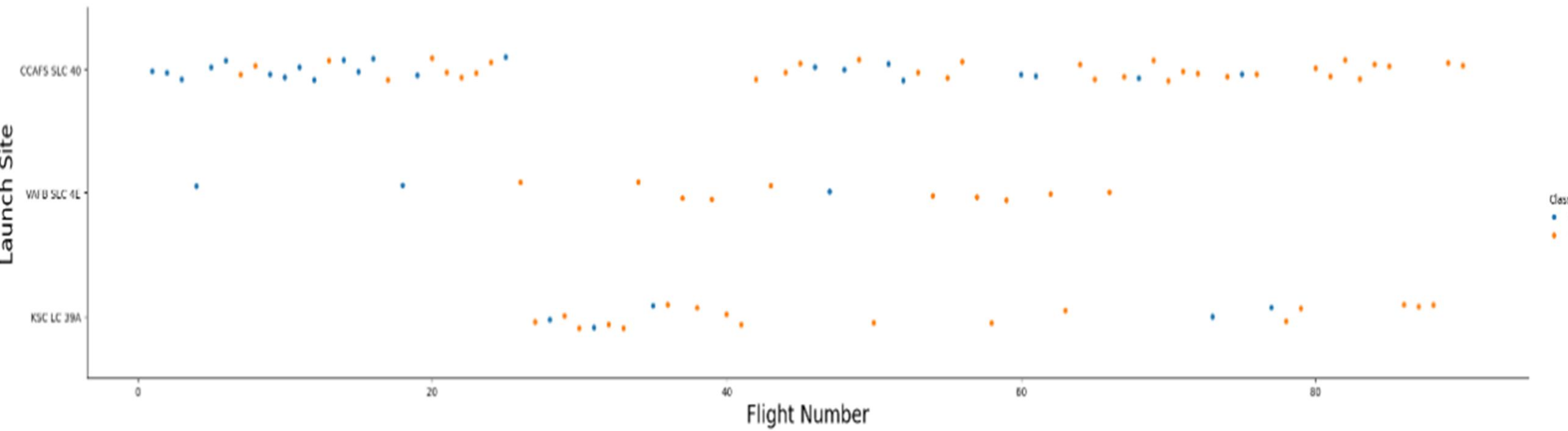
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

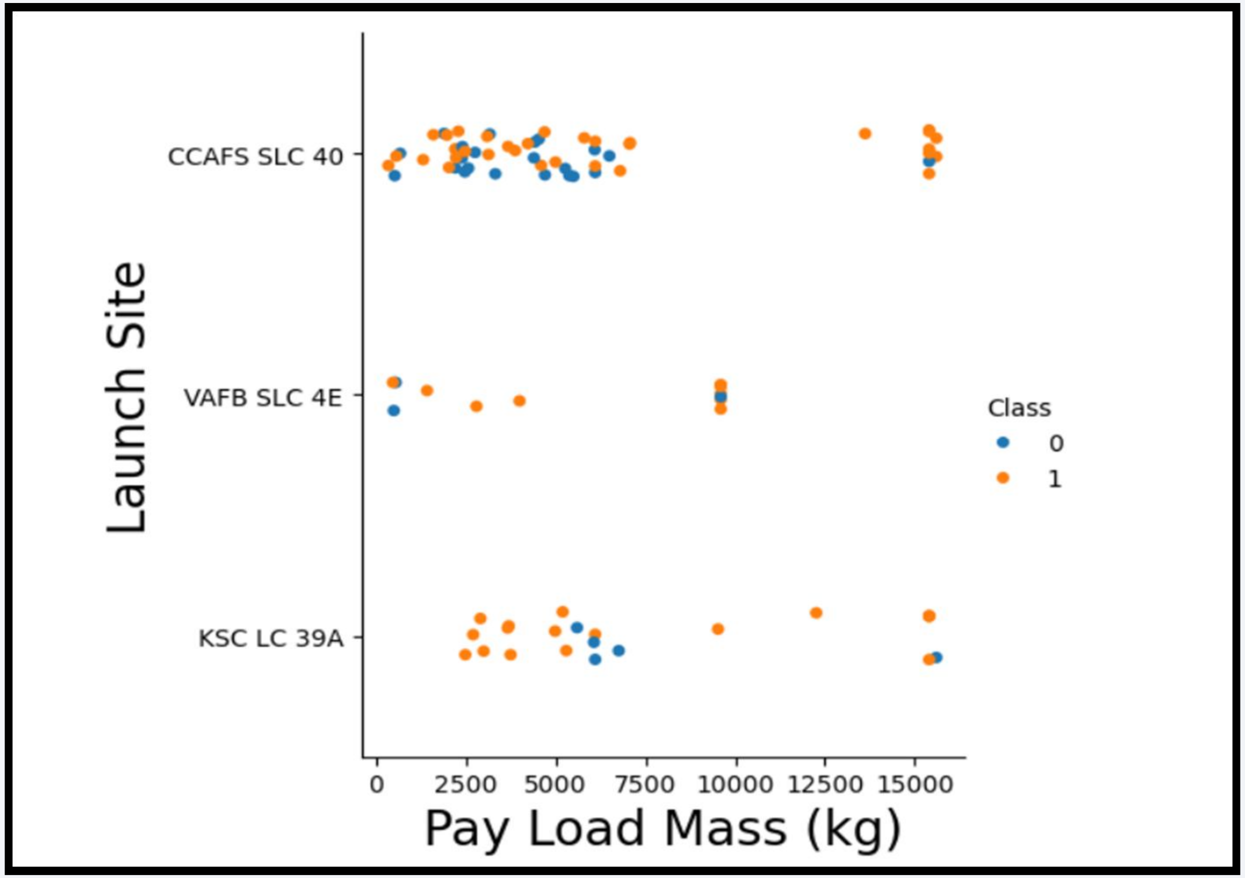
Flight Number vs. Launch Site

- Based on the plot, we observed that higher flight volumes at a launch site correlate with increased success rates.



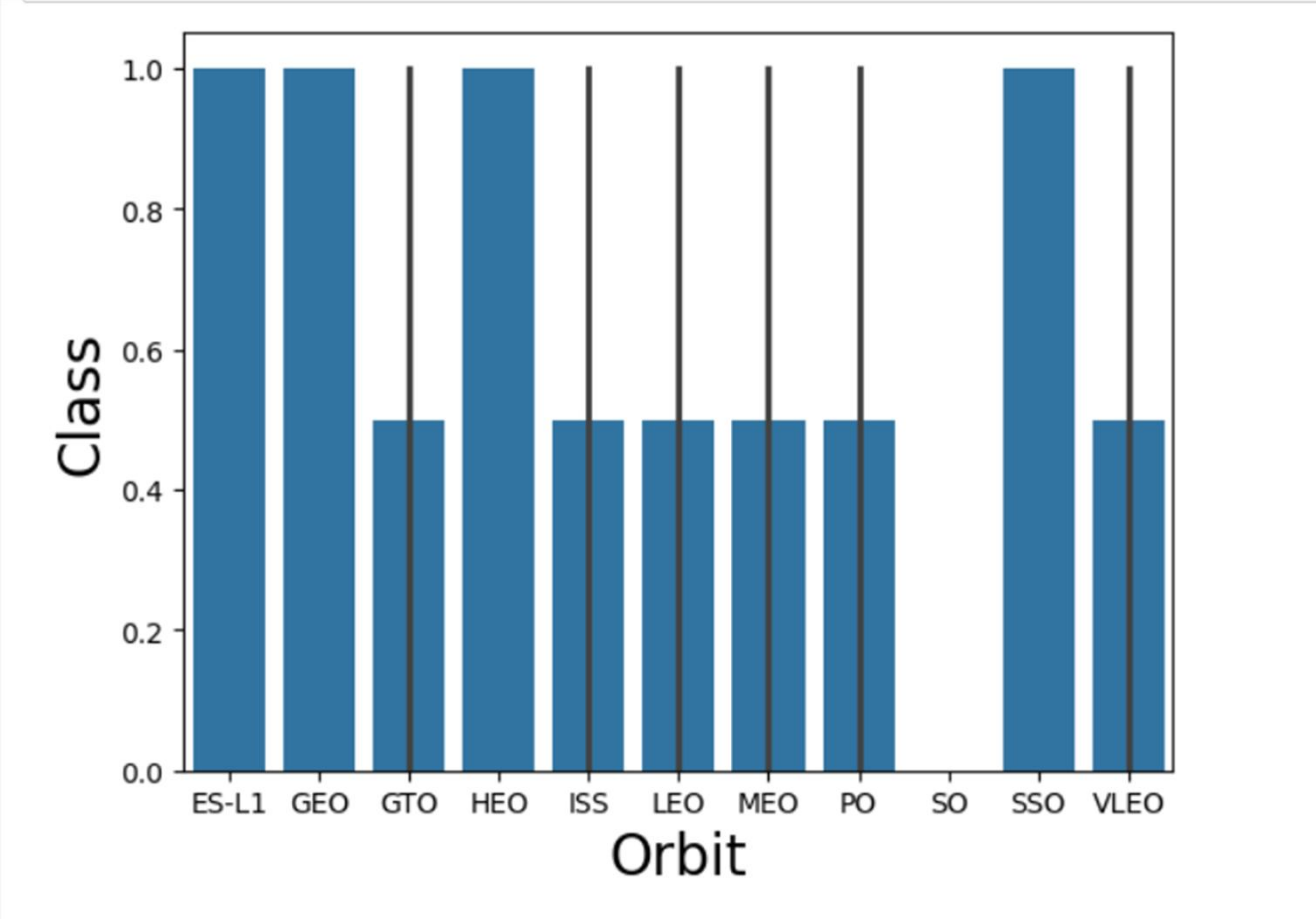
Payload vs. Launch Site

- This plot conveys that for the VAFB-SLC launchsite there are no rockets launched for heavy payload mass(greater than 10000).



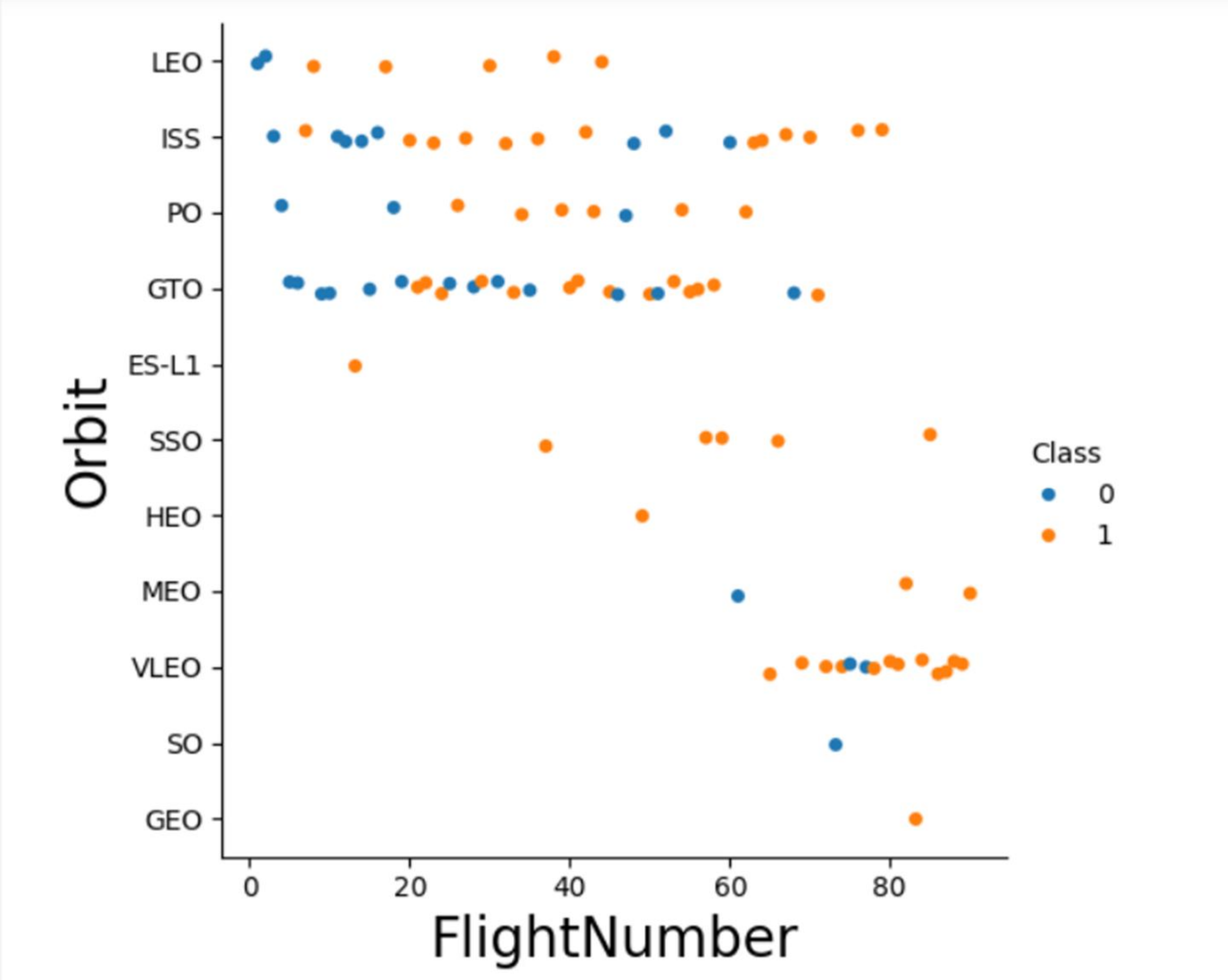
Success Rate vs. Orbit Type

- This plot conveys that among all these orbit types, GTO has the lowest success rate and SSO has the highest one.



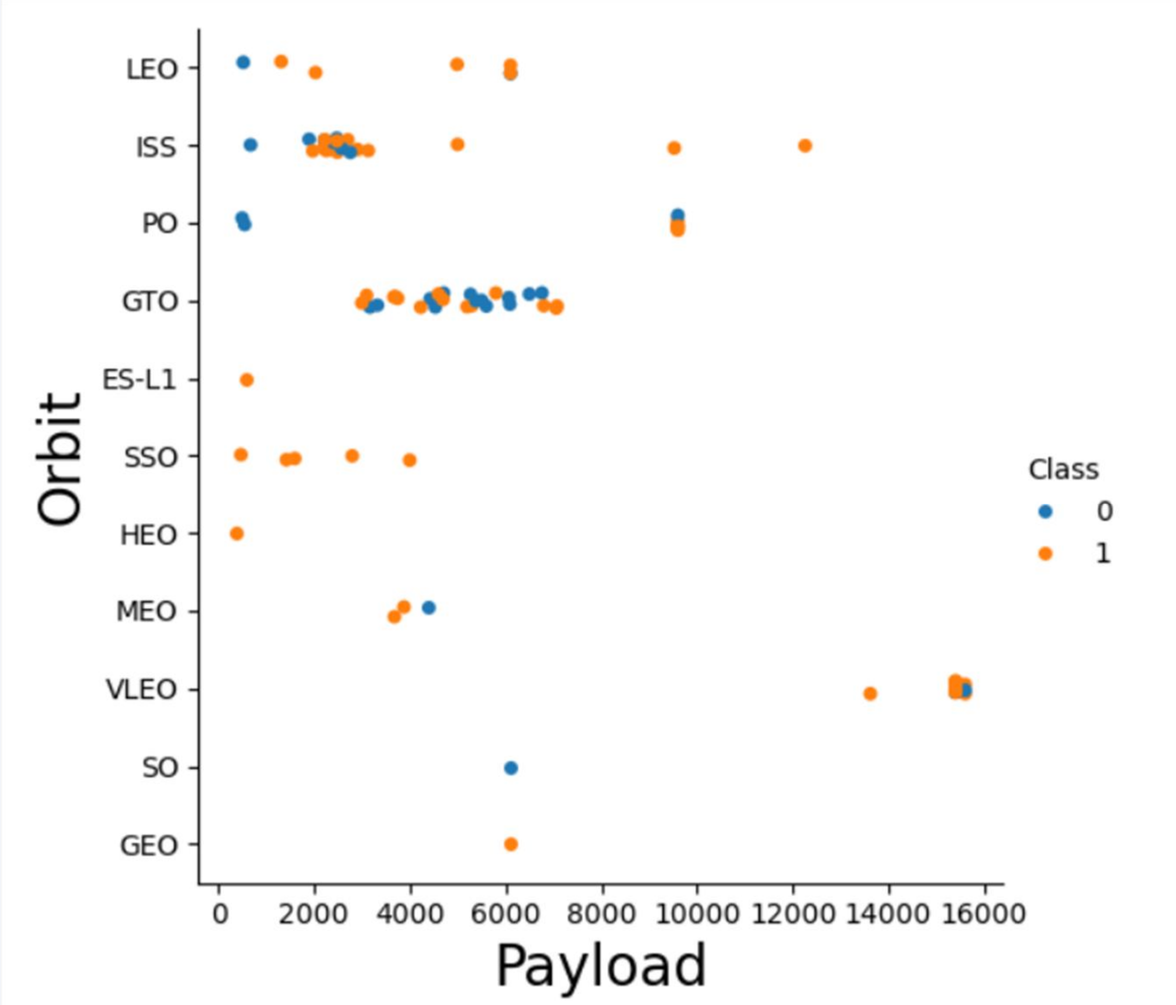
Flight Number vs. Orbit Type

- This plot conveys that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



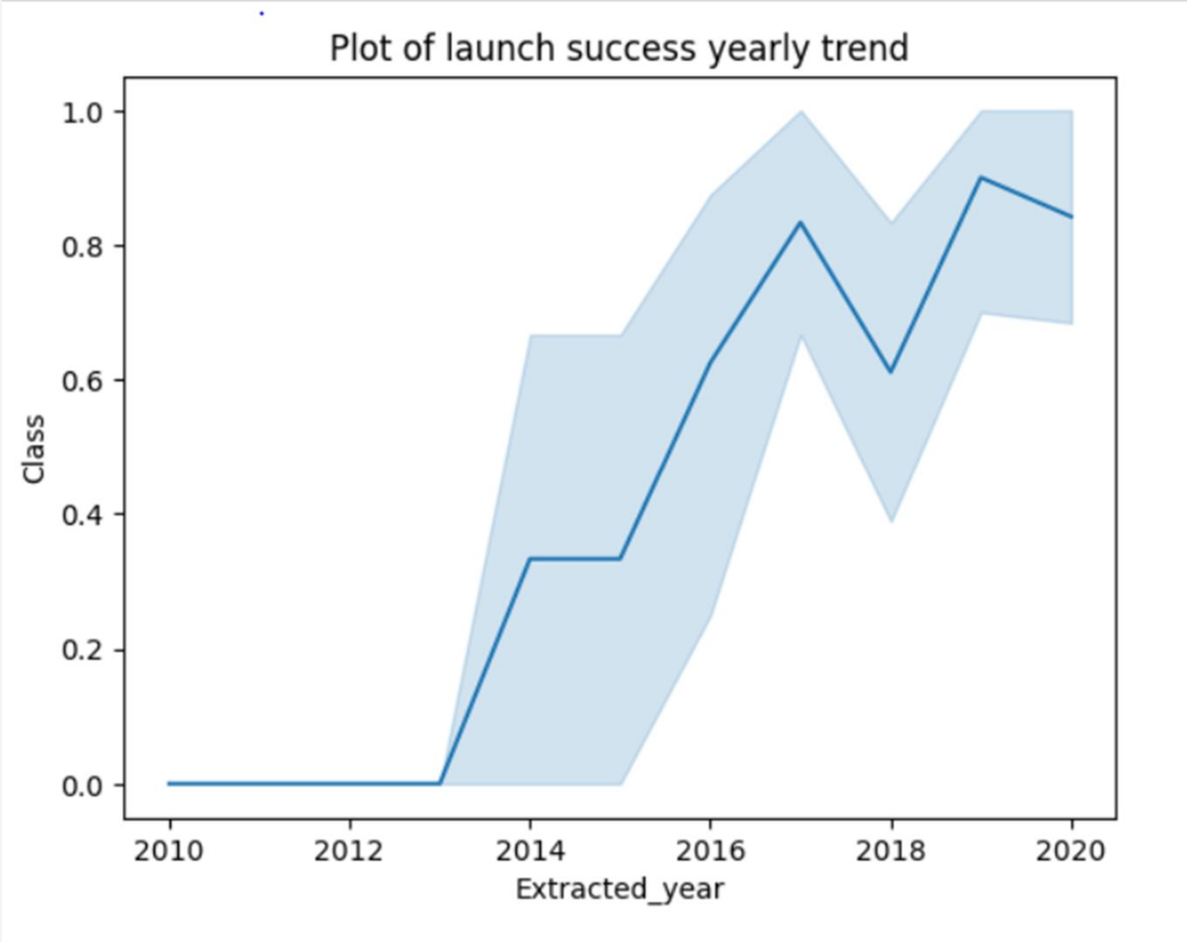
Payload vs. Orbit Type

- This plot conveys that with heavy payloads, successful landings are more frequent for PO, LEO, and ISS orbits.



Launch Success Yearly Trend

- This plot conveys that success rate has been increasing since 2013 until 2020.



All Launch Site Names

- The names of the unique launch sites are:

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

Task 1

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

```
In [26]: %sql select distinct(LAUNCH_SITE) from SPACEXTBL

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

Out[26]:

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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

Task 2

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

In [27]:
```
%sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

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

Out[27]:

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

Task 3

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

```
In [28]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'

 * sqlite:///my_data1.db
Done.

Out[28]:
```

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

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

Task 4

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

In [29]:
```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
```

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

Out[29]:

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

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

Task 5

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

Hint:Use min function

```
In [32]: %sql select min(DATE) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
```

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

Out[32]:

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.

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 [34]: %sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOA
```

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

Out[34]:

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.

Task 7

List the total number of successful and failure mission outcomes

```
In [37]: %sql select count(MISSION_OUTCOME) from SPACEXTBL where MISSION_OUTCOME like '%Success%' or MISSION_OUTCOME = '%Failure%'
```

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

Out[37]:

count(MISSION_OUTCOME)
100

Boosters Carried Maximum Payload

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

Task 8

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

In [36]:
```
%sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

Out[36]:

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 query
q_failed_landing= """ Select Date, Booster_Version, Launch_Site, Landing_Outcome
                    from spacex_v11
                    where Landing_Outcome = 'Failure (drone ship)'
                    and Date like '%2015%'   """
```

```
fail_drone= pd.read_sql_query(q_failed_landing,conn)
fail_drone.head(5)
```

	Date	Booster_Version	Launch_Site	Landing_Outcome
0	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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.

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 [40]:
```
%sql select * from SPACEXTBL where Landing_Outcome like 'Success%' and (DATE between '2010-06-04' and '2017-03-20') order by date
```

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

Out[40]:

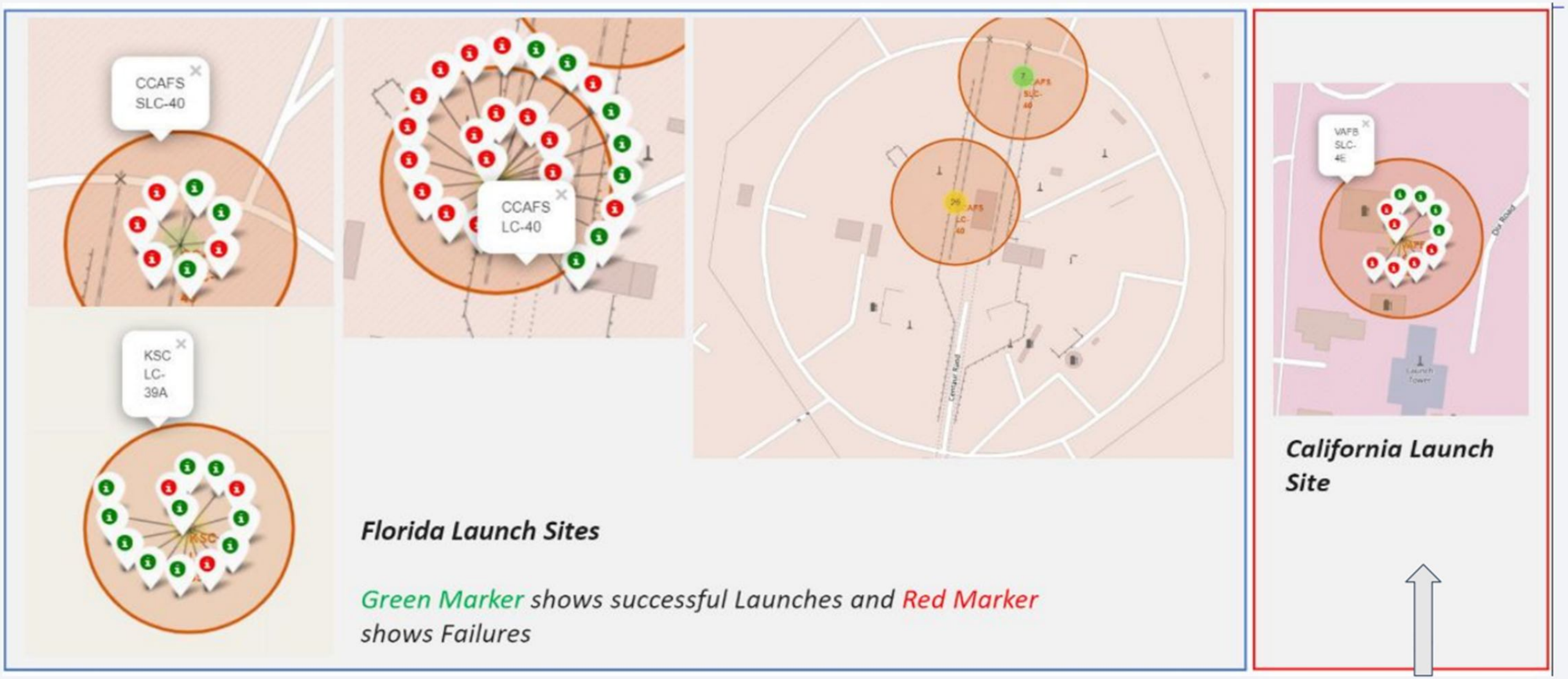
Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Success (drone ship)
2016-08-14	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-07-18	4:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Success (drone ship)
2016-05-06	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-04-08	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (drone ship)
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

Section 3

Launch Sites Proximities Analysis

Launch Sites

Success and Failure Rates



Florida Launch Sites

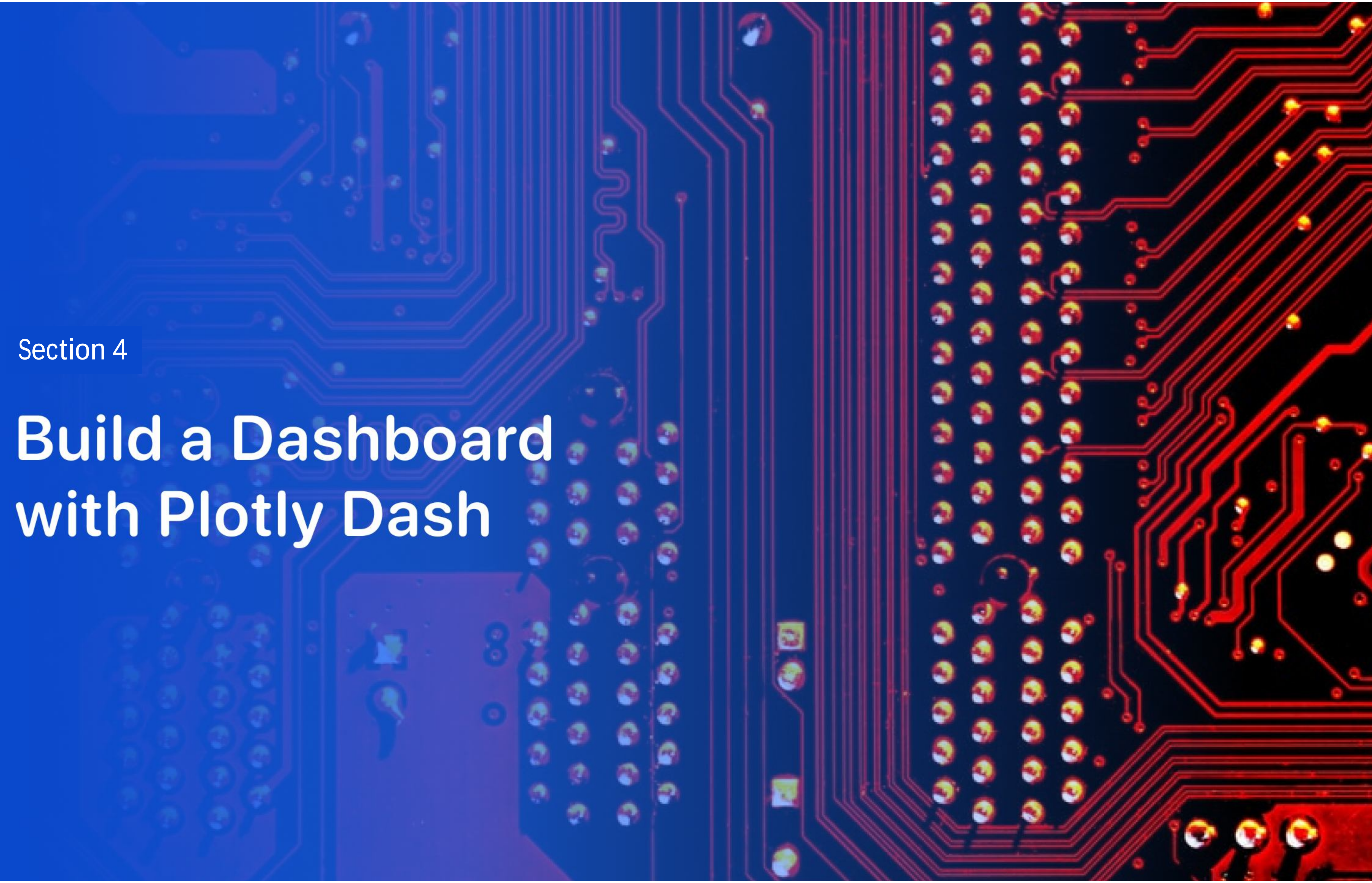
Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

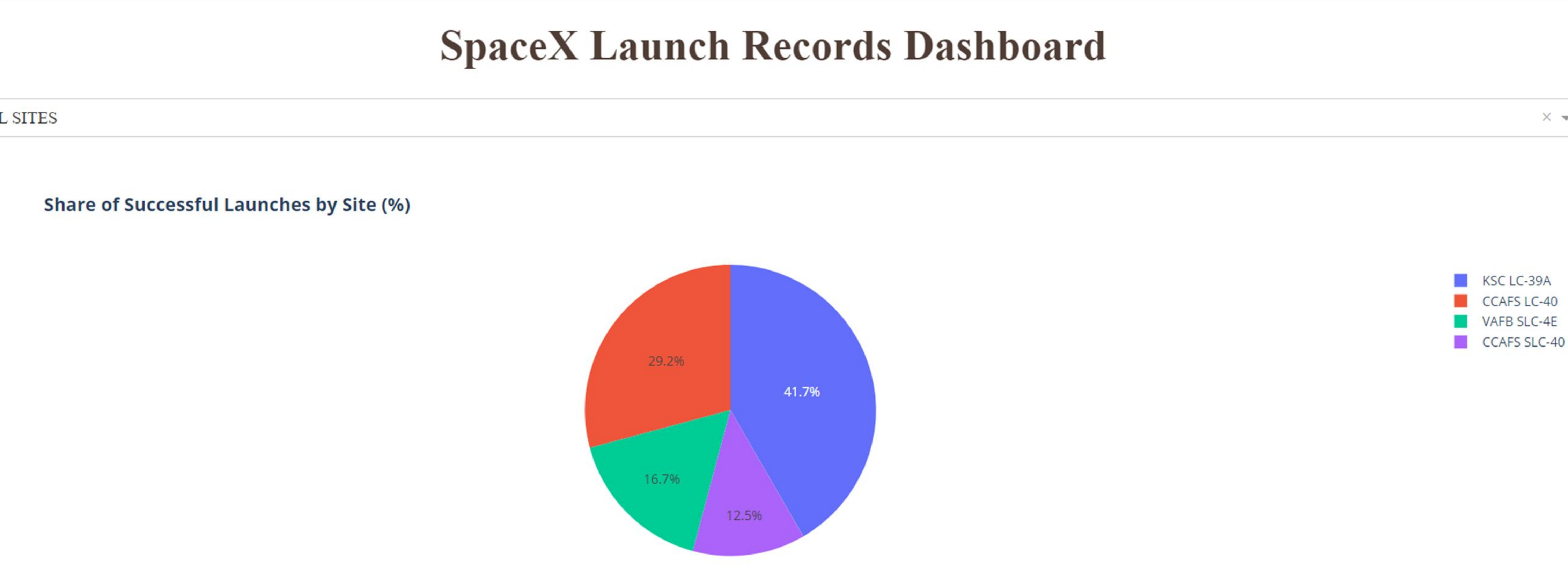
Distance between launch sites and areas

Section 4

Build a Dashboard with Plotly Dash

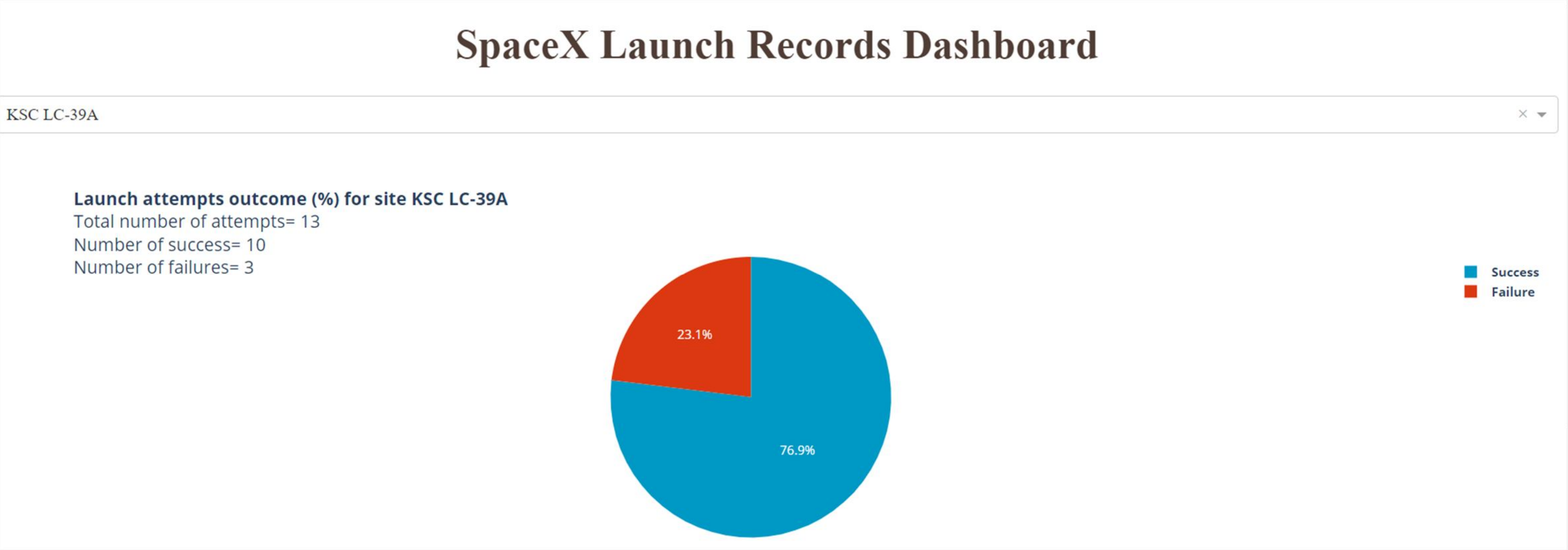


Launch Success Count for Sites



Launch Site with Highest Launch Success Ratio

- This plot shows that for the highest launch success ratio there were total 13 attempts in which 3 were failures and others were success.



Payload vs. Launch Outcome Scatter Plot

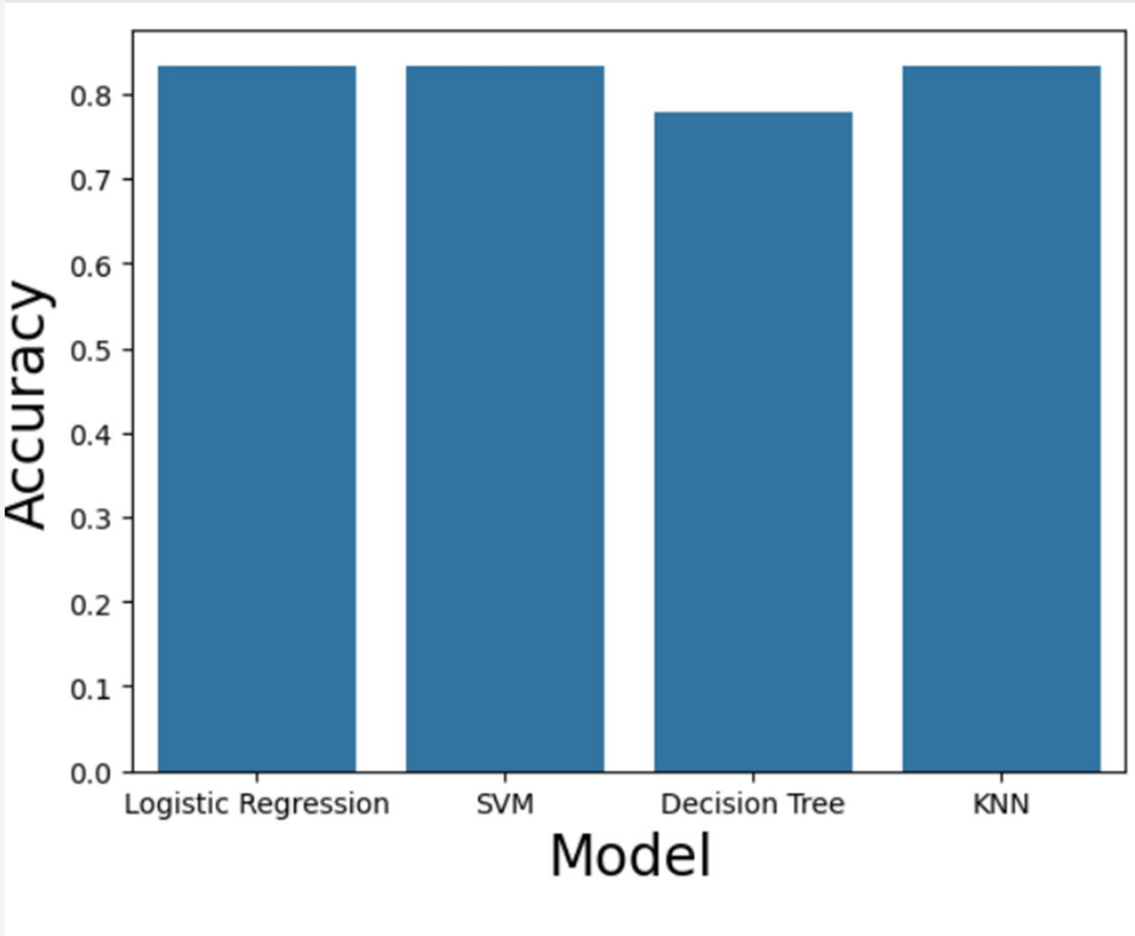
- The data suggests that as payload mass increases, particularly in the range of 6,000 to 10,000 kg, the success rate generally remains high across different booster versions. Also , the FT and B4 versions show consistent success in the higher payload ranges.

Section 5

Predictive Analysis (Classification)

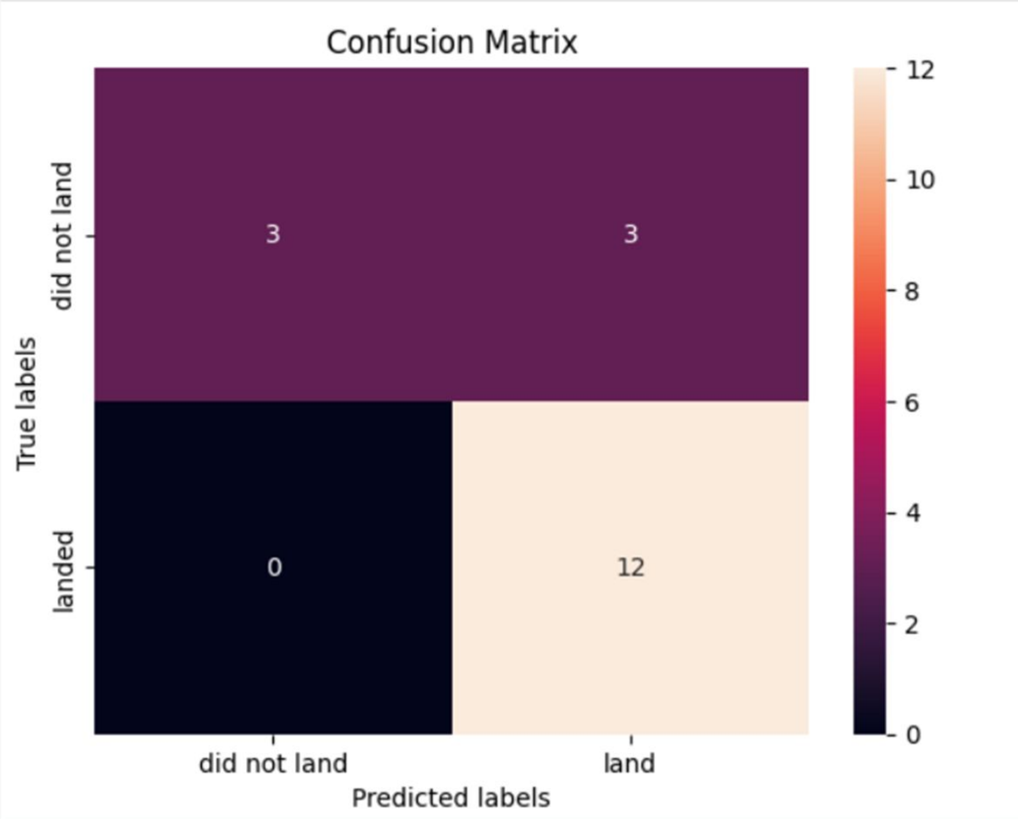
Classification Accuracy

- The KNN is the model with the highest classification accuracy.



Confusion Matrix

- The matrix reveals that the model accurately predicts 12 successful landings, with no false negatives, indicating strong reliability in identifying successful landings.



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- KSC LC-39A had the most successful launches of any sites.
- KNN is the best machine learning algorithm for this task.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project


Thank you!