IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Navya Oberoi

10th June, 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection using API
  - Data Collection using Web Scraping
  - Data Wrangling
  - EDA with SQL
  - EDA with Data Visualization
  - Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics result in maps
  - Predictive Analytics result

Introduction

- Project background and context

  In this project, I have tried to predict the first stage landing of the Falcon 9 rocket launch. 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.

- Problems you want to find answers

  - Featutes which will determine whether the rocket will land successfully or not?
  - The relationship amongst various features that determine the success rate of a successful landing.
  - What conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - In the data acquisition phase, i have mainly collected the data using APIs and using WebScraping (using BeautifulSoup) where i extracted the information through the Wikipedia page of SpaceX.
- Perform data wrangling
  - In the data pre-processing, i have applied feature engineering (or basically one-hot encoding) on the dataset.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Here i have applied different classification models.

Data Collection

- The data were collected through many different ways :
  - 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().
  - We then 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

- Here, we need to request and parse the SpaceX launch data using the GET request and then decode the request using .json() and dataframe conversion using .json_normalize(). Also i did a bit of data wrangling and formatting where i replaced the missing values with the mean value.
- Link to the notebook : Data Collection API

Task 1: Request and parse the SpaceX launch data using the GET request

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

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

Task 2: Dealing with Missing Values

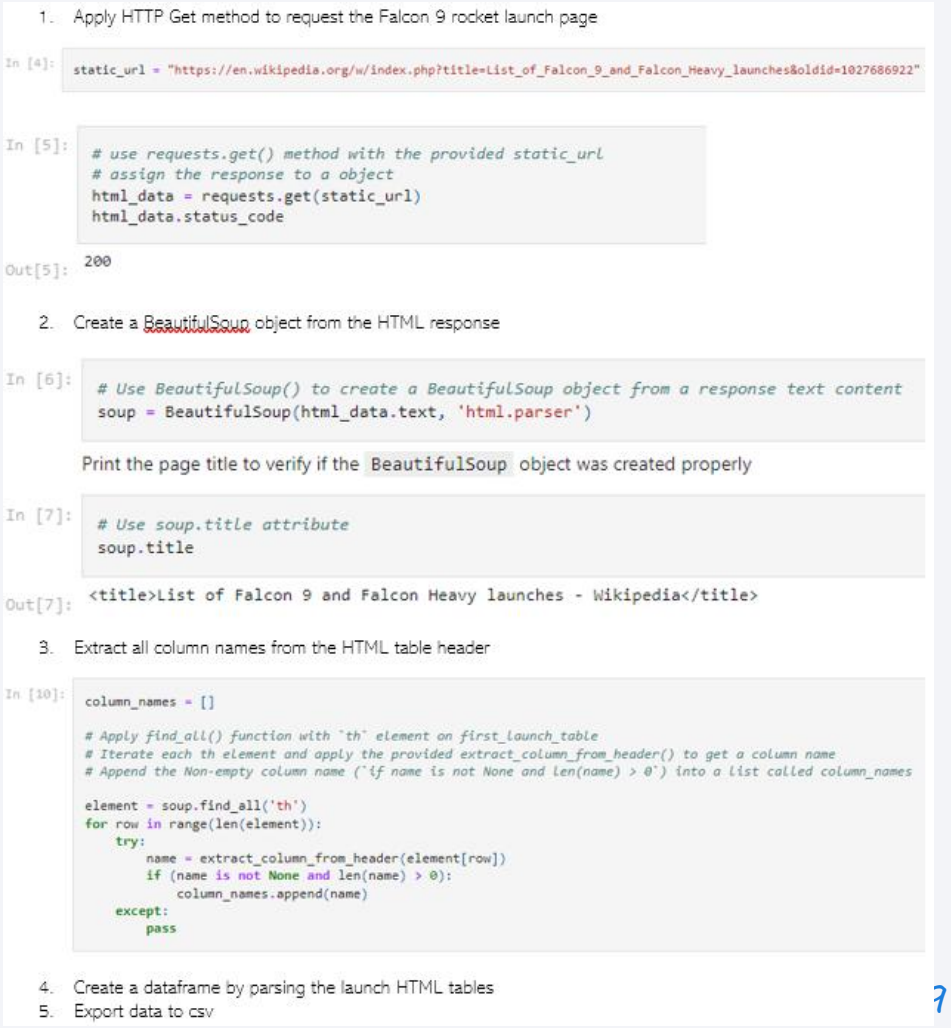
```
[31] data_falcon9.isna().sum()
```
Python

```
FlightNumber      0
Date              0
BoosterVersion    0
PayloadMass       5
Orbit             0
LaunchSite        0
Outcome           0
Flights           0
GridFins          0
Reused            0
Legs              0
LandingPad       26
Block             0
ReusedCount       0
Serial            0
Longitude         0
Latitude          0
dtype: int64
```

```
[32] # Calculate the mean value of PayloadMass column
     pm = df.PayloadMass.mean()
     # Replace the np.nan values with its mean value
     data_falcon9['PayloadMass'].replace(np.nan, pm, inplace = True)
```
Python

Data Collection - Scraping

- Here, I have extracted the Falcon 9 launch records HTML table from the Wikipedia source where i have mainly performed to 2 tasks : HTTP GET method and created a Beautiful Soup object for extracting data.
- Link to the Notebook : Data-Collection-Webscraping



1. Apply HTTP Get method to request the Falcon 9 rocket launch page

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

In [5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url)
html_data.status_code
```

Out[5]:
```
200
```

2. Create a BeautifulSoup object from the HTML response

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text, '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>
```

3. Extract all column names from the HTML table header

In [10]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

4. Create a dataframe by parsing the launch HTML tables
5. Export data to csv

Data Wrangling

- We performed exploratory data analysis and determined the training labels.
- Also here we calculated number of launches on each site and calculated the number and occurrence of each orbit.
- Also, we created landing outcome label from outcome column and exported the results to csv.
- Link to the notebook : Data-Wrangling

```python
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

Python

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

```python
# Apply value_counts on Orbit column
df.Orbit.value_counts()
```

Python

```
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
```

```python
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for i in df.Outcome:
    if i in bad_outcomes:
        landing_class.append(0)

    else:
        landing_class.append(1)
```

Python

```python
landing_class
```

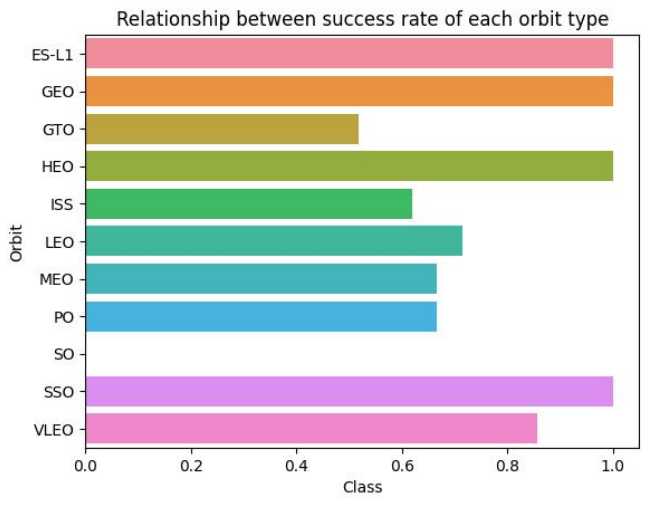
Python

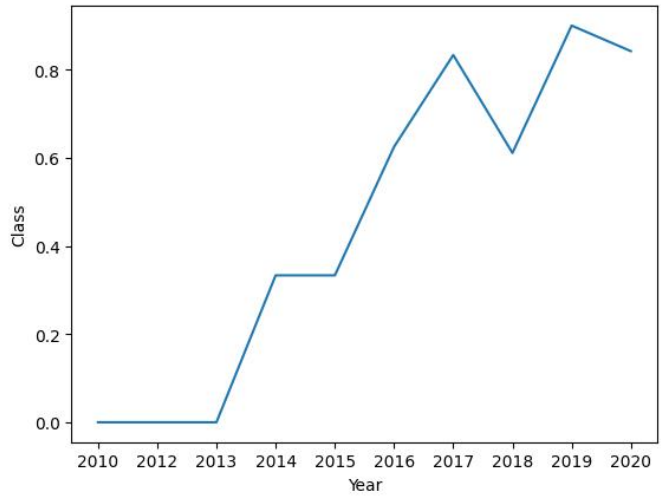
EDA with SQL

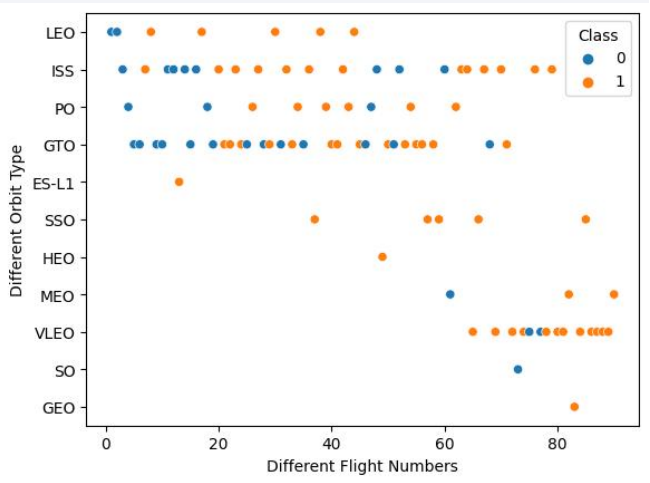
- The SQL Queries performed were :
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- Link to the notebook : EDA-SQL

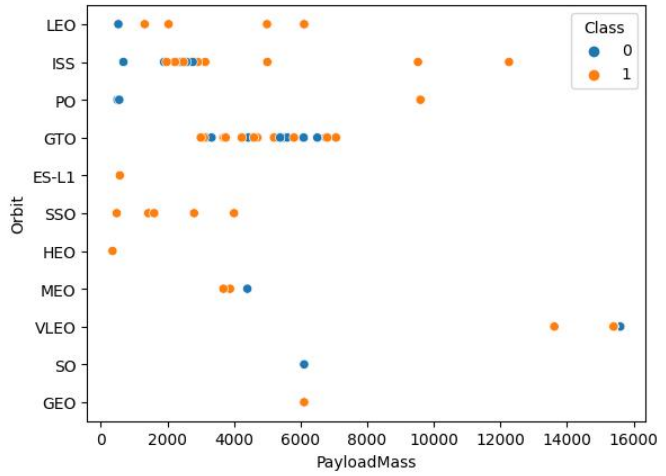
EDA with Data Visualization

- We 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.
- Also i have applied feature engineering (basically one hot encoding) to the dataset for data modelling.
- Link to the notebook : EDA-Data-Viz









Build an Interactive Map with Folium

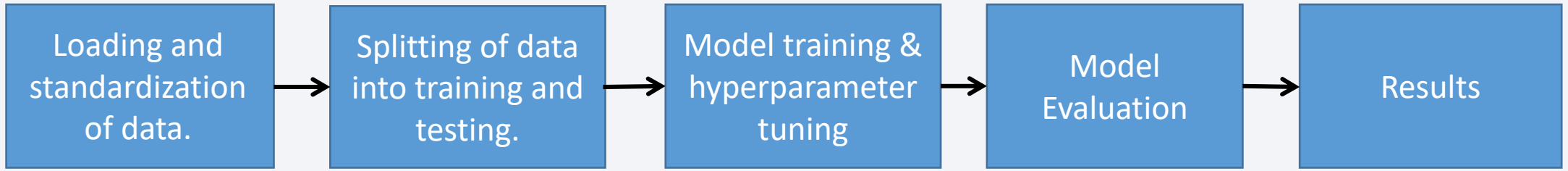
- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.
- Link to the notebook : Visual-Analytics-using-Folium

Build a Dashboard with Plotly Dash

- Built an interactive Dashboard using Plotly Dash having the following features:
- Added a dropdown for the site input, and plotted a pie chart to view the relative success and failures in launches for each site.
- Added a Range Slider for the Payload range to view the scatterplot between Payload and Class, according to the input site and within the provided payload range.
- These plots and interactions allowed to quickly visualize and analyze the relation between payloads and launch sites, which helped to find the best site for launching Falcon9.
- Link to the Notebook : SpaceXDashApplication.py

Predictive Analysis (Classification)

- The dataset's features and target variables were loaded in their respective dataframes. Then data standarization is also done using StandardScaler().
- The data was split into training and test data. The labels are the data that we want to predict. The labels are in the column labeled "class". The features are all the other columns.
- For the problem statement, classification models such as logistic regression, SVM, Decision Trees, and KNN were used. We determined the best hyperparameters using GridSearchCV with 10 fold cross validation We also plotted the confusion matrices for each model.
- We will use the test data to determine which machine learning model performs the best using confusion matrices and scores, then compile the results.
- Link to the notebook : ML-Prediction.ipynb



Results

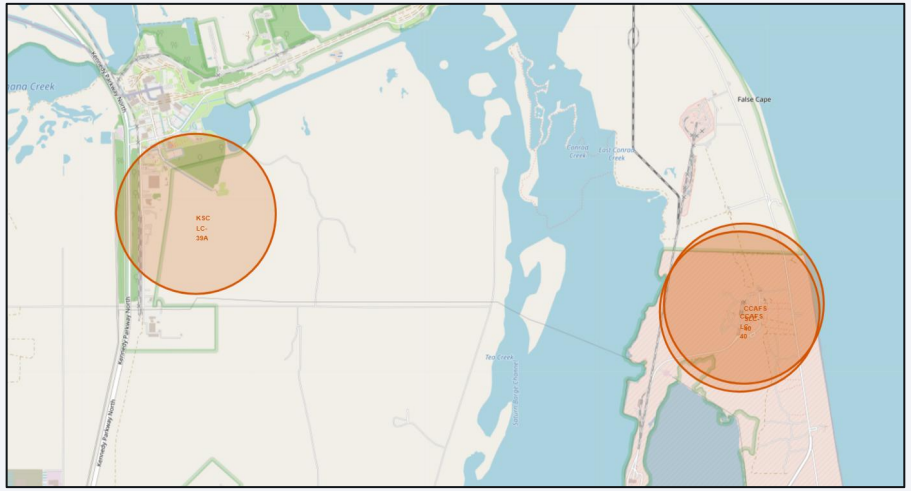
Exploratory data analysis results

- Newer rockets could carry more payload
- Payloads over 8000kg have high success rate
- the sucess rate since 2013 kept increasing till 2020
- 2010-2013 period had no success rate
- Space X uses 4 different launch sites;
- VLEO orbit has 14 launches and 85% success rate
- The first successful landing outcome was in 2015, five years after the first launch.
- With booster F9, almost every mission outcome was successful.
- around 70 landing outcomes were successful, while there were 22 no attempts, and around 10 failed.
- With time, the success increased mostly due to advancement in technology.

Results

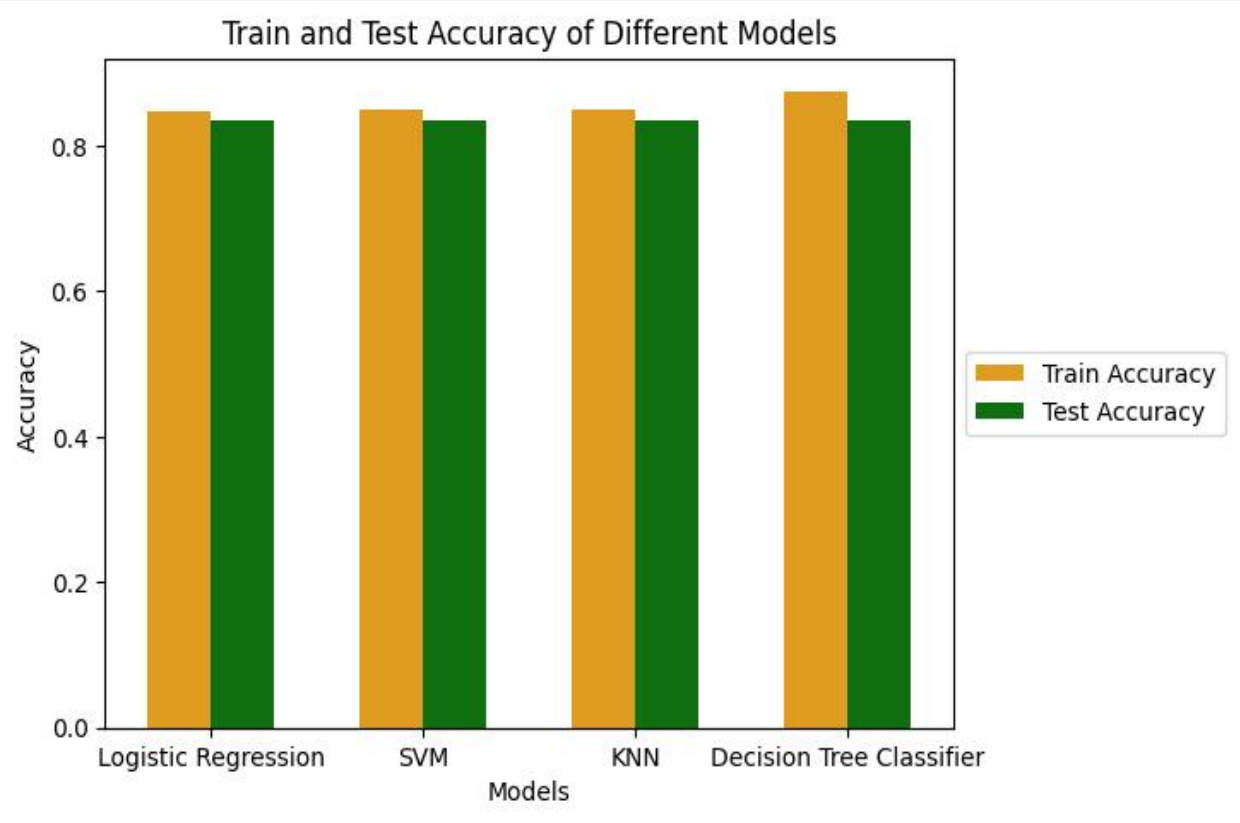
Interactive analytics results:

- launch sites are close to the equator
- Most launches happens at east cost launch sites
- launch sites are in close proximity to railways
- Launch sites are in close proximity to highways
- Launch sites in close proximity to coastline
- Launch sites keep certain distance away from cities



Results

- **Predictive analysis results:**
  - Here all the models had the same accuracy on testing data.
  - In case of training data, all the models had the same accuracy except Decision Tree Classifier which had 87.5% accuracy.
  - So, Here we can say that all the models performed equally well during the testing but in the training phase the **Decision Tree Classifier** performed exceptionally well when compared with others.

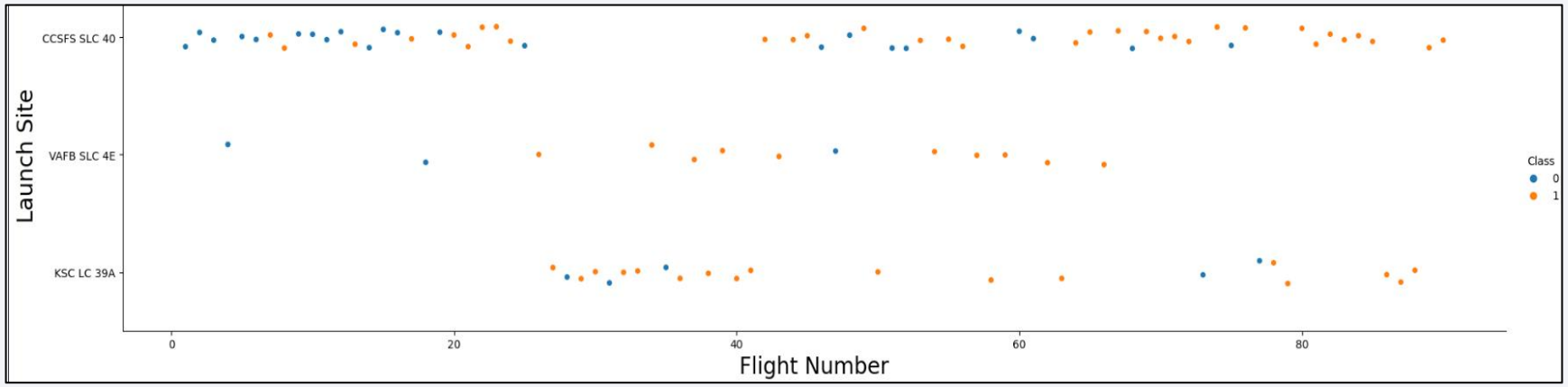


Section 2

Section 2

Insights drawn from EDA

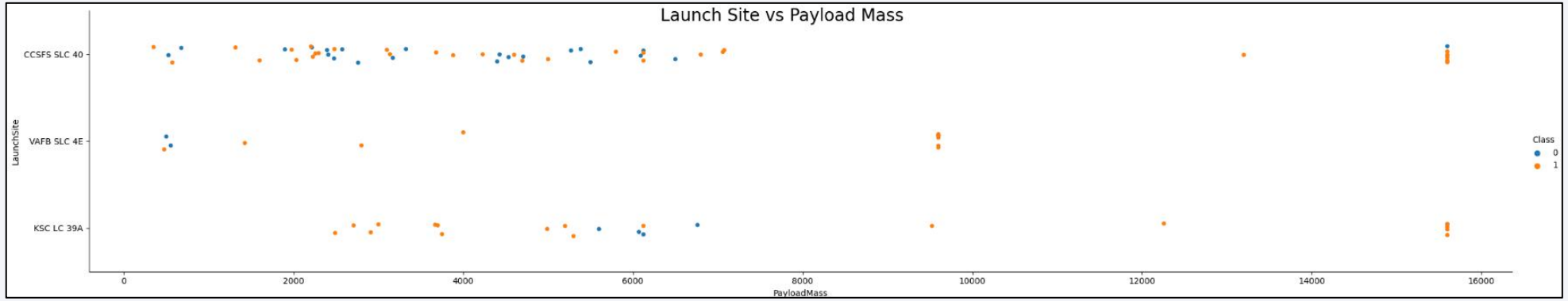
Flight Number vs. Launch Site



Observations :

- CCAFS LC-40 has overall lower success rate than other two as it failed a lot during initial flights
- KSC LC-39A and VAFB SLC 4E have almost same success rate, and they have a relatively higher flight number so failure rate is low.
- Best Site is CCAFS SLC-40 as it has very high success rate in recent times.

Payload vs. Launch Site

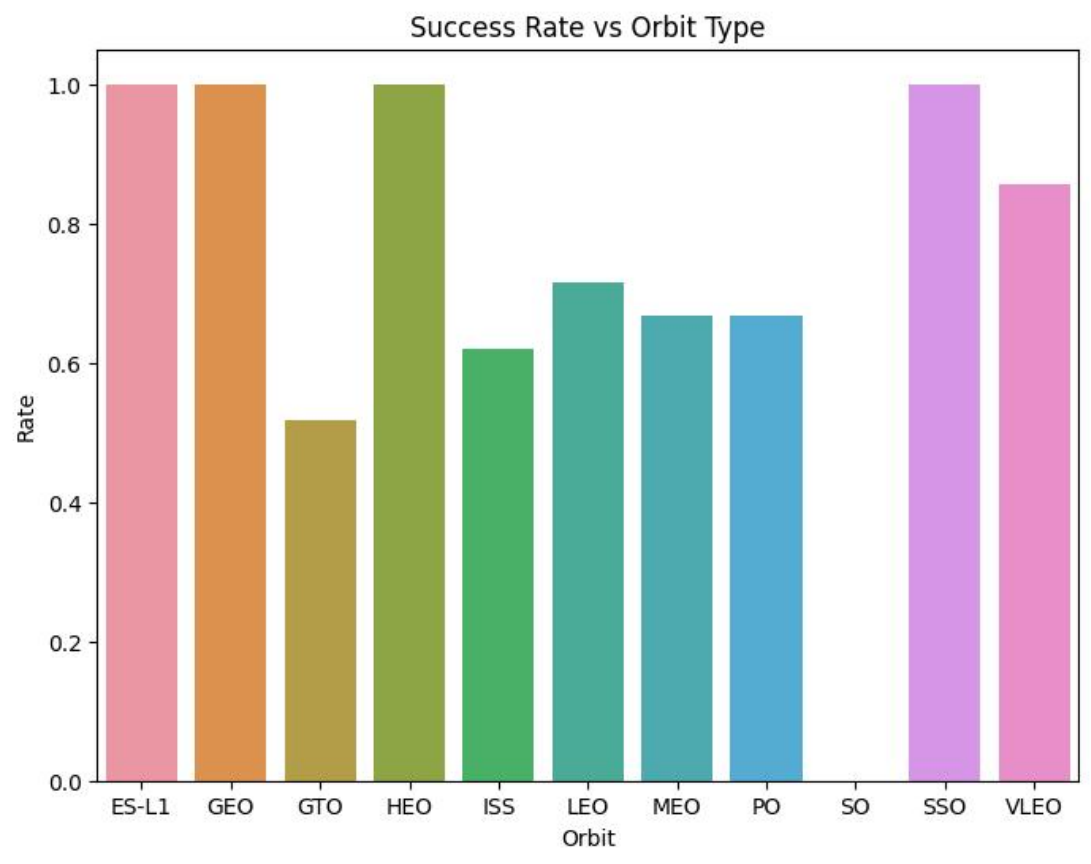


Observations:

- There are no rockets launched for heavy payload mass (greater than 10000) for the VAFB-SLC launchsiteA
- Payloads over 8000kg have high success rate
- Payloads less than 6000kg have high failure rate for the CCAFS SLC-40 launch site

Success Rate vs. Orbit Type

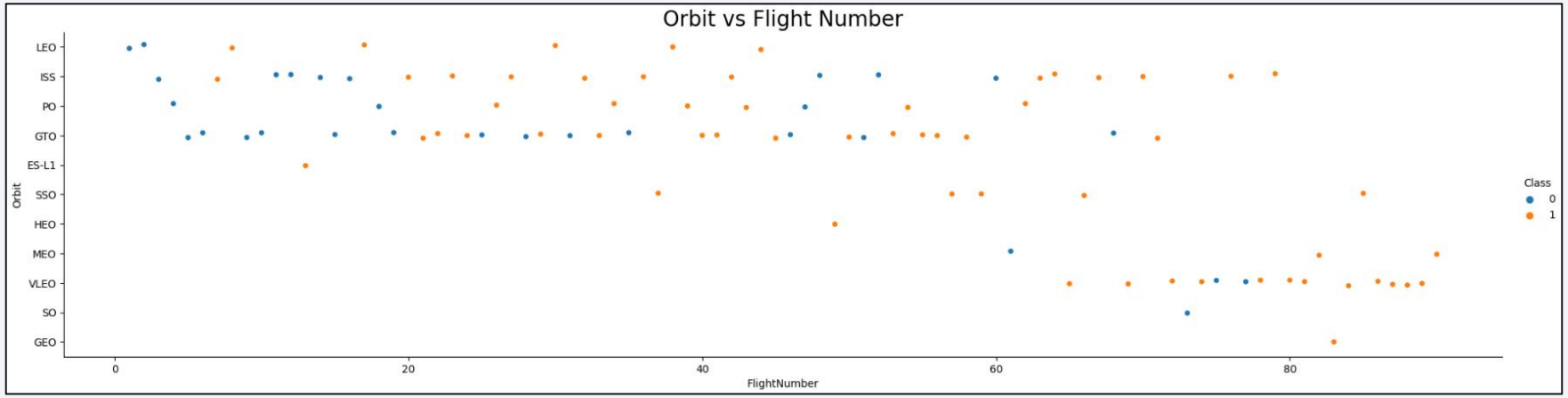
	Orbit	Class
0	ES-L1	1.000000
1	GEO	1.000000
2	GTO	0.518519
3	HEO	1.000000
4	ISS	0.619048
5	LEO	0.714286
6	MEO	0.666667
7	PO	0.666667
8	SO	0.000000
9	SSO	1.000000
10	VLEO	0.857143



Observations from data and bar chart:

- GEO, HEO, ES-L1, SSO have 1 launches and 100% success rate
- SO has 1 launch and 0% succes rate
- ISS has 21 launches 61% success rate
- VLEO has 14 launches and 85% success rate
- PO has 9 launches and ~65% success rate

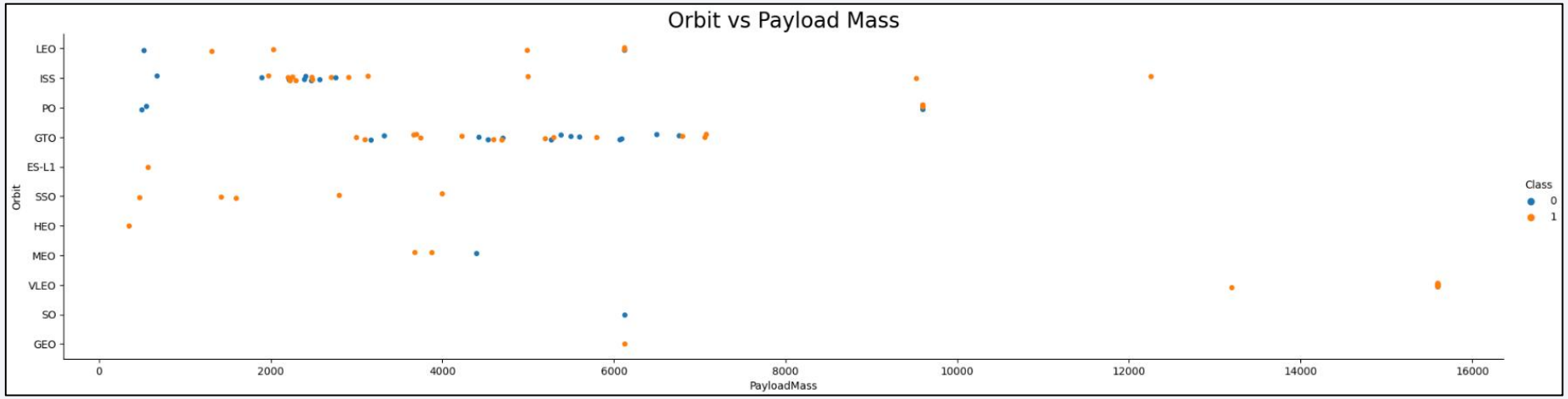
Flight Number vs. Orbit Type



Observations:

- We can see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.
- Success rate is low for the first flight and it increases as the flight number increases

Payload vs. Orbit Type



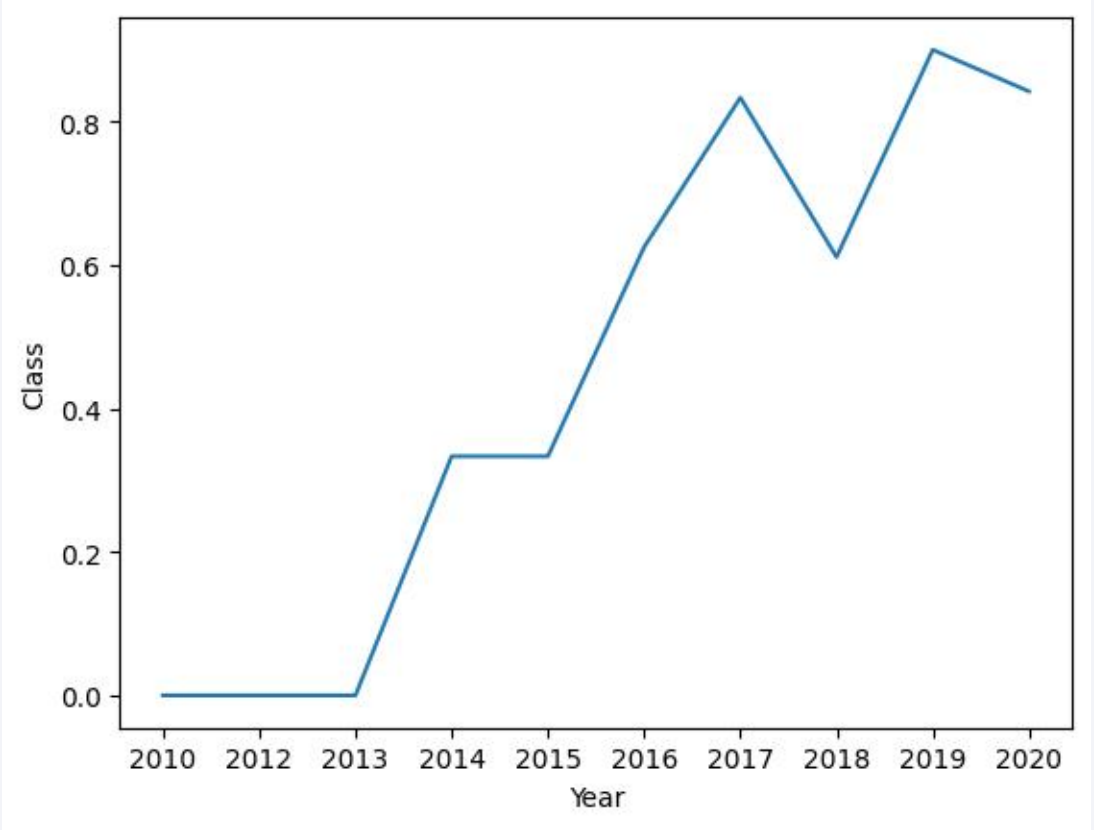
Observations:

- With heavy payloads the successful landing or positive landing rate are more for PO, LEO and ISS orbits.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend

Observations:

- The sucess rate since 2013 kept increasing till 2020
- 2010-2013, 2014-2015 period had no success rate



All Launch Site Names

- Here I have used **distinct** clause to display the names of unique launch sites.

```
%sql select distinct launch_site from spacextbl;
```

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

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Launch Site Names Begin with 'CCA'

```
%sql select * from spacextbl where launch_site like 'cca%' limit 5
```

Python

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (parachute)
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No attempt
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No attempt
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No attempt

- Here i have used **LIKE** predicate to select launch site names beginning with 'CCA' and **LIMIT** clause to display only first 5 records.

Total Payload Mass

- Here i have used **SUM()** on PAYLOAD_MASS__KG_ column to calculate the total payload mass and used the **LIKE** predicate to filter out which are carried by boosters launched by NASA (CRS).

```
# cur.execute('select sum("PAYLOAD_MASS__KG_") from SPACEXTBL where Customer = \'NASA (CRS)\'')
# cur.fetchall()
%sql select sum(PAYLOAD_MASS__KG_) as SUM from spacextbl where customer like 'NASA (CRS)'
```

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

SUM
45596.0

Average Payload Mass by F9 v1.1

- Used the **AVG()** on PAYLOAD_MASS__KG_ column and used **WHERE** clause in where the average payload mass carried by booster version F9 v1.1.

```
%sql select avg(payload_mass__kg_) as AVG from spacextbl where booster_version = 'F9 v1.1'

 * sqlite:///my_data1.db
Done.

AVG
2928.4
```

First Successful Ground Landing Date

- Used the **MIN()** function on Date column and used **WHERE** clause in which landing outcome is successful on ground pad.

```
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (ground pad)'

 * sqlite:///my_data1.db
Done.

MIN(DATE)
01/08/2018
```

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select booster_version as name from spacextbl where landing_outcome like '%drone%' and payload_mass__kg_ > 4000 and payload_mass__kg_ < 6000
```

Python

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

name
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

- We used the **where** clause to filter for boosters which have successfully landed on drone ship and applied the and condition to find the query result.

Total Number of Successful and Failure Mission Outcomes

- Here i have used the **COUNT()** function and **WHERE** clause to calculate the frequency of successful and failure missions.

```
%sql select count(*) as successful_launches from spacextbl where mission_outcome like '%Success%';
```

✓ 0.0s

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

successful_launches
100

```
%sql select count(*) as failed_launches from spacextbl where mission_outcome like '%Failure%'
```

✓ 0.0s

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

failed_launches
1

Boosters Carried Maximum Payload

- Used a subquery to calculate the maximum payload using **MAX()** and then used **= operator** in the subquery to filter out the data.

```
%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

```
%sql select substr(Date, 4, 2), booster_version, launch_site, landing_outcome from spacextbl where substr(Date,7,4)='2015' and landing_outcome like '%Failure%drone%'
```

✓ 0.0s

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

substr(Date, 4, 2)	Booster_Version	Launch_Site	Landing_Outcome
10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- The failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015 is found out using the **substr()** function, **AND** conditional statement and **LIKE** predicate.

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

- The query to 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 is shown beside:

  - Here i have used the count() the count the frequency of landing landing outcome then used groupby clause to group the data by all possible landing outcomes and atlast for ranking i have used order by clause

```
%%sql select landing_outcome, count(landing_outcome) as COUNT from spacextbl
    group by landing_outcome order by count(landing_outcome) desc
```

✓ 0.0s

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

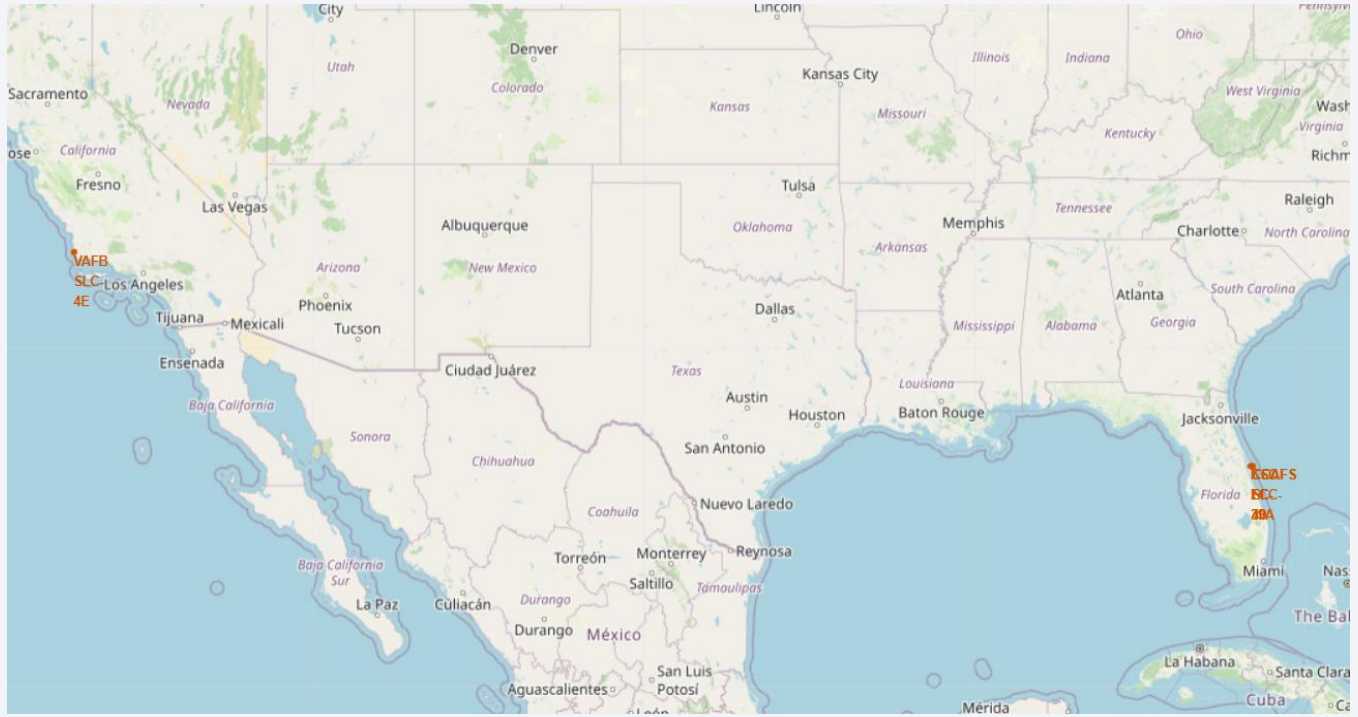
Landing_Outcome	COUNT
Success	38
No attempt	21
Success (drone ship)	14
Success (ground pad)	9
Failure (drone ship)	5
Controlled (ocean)	5
Failure	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1
No attempt	1
None	0


Section 3
Launch Sites
Proximities Analysis

Map showing All Launch Sites

The map shows all the sites of Falcon 9 launch.

- We can see that the sites are located very close to the coastline as the failure rates of rockets are high (about 5-10%) and civilian areas must be avoided.
- 3 launch sites are located on the east coast of the US and 1 on the west coast
- all the sites are located close to the equator as it takes less fuel to launch a rocket from the equator

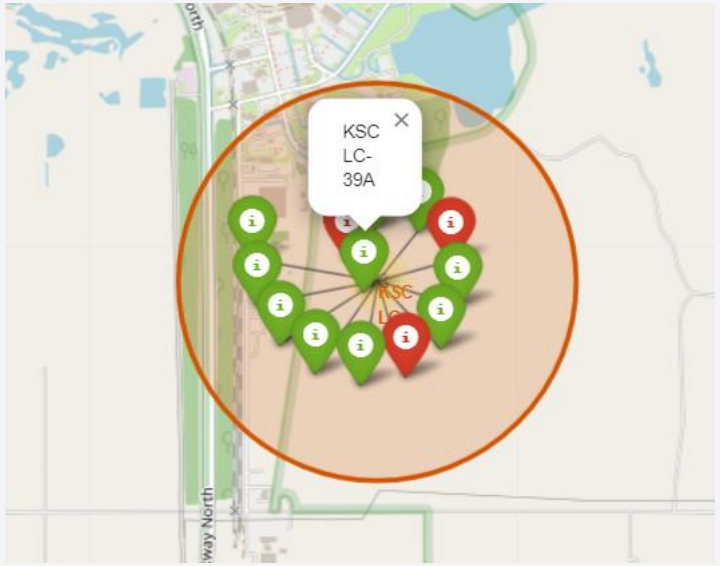


Map Showing the no. of launches for each site

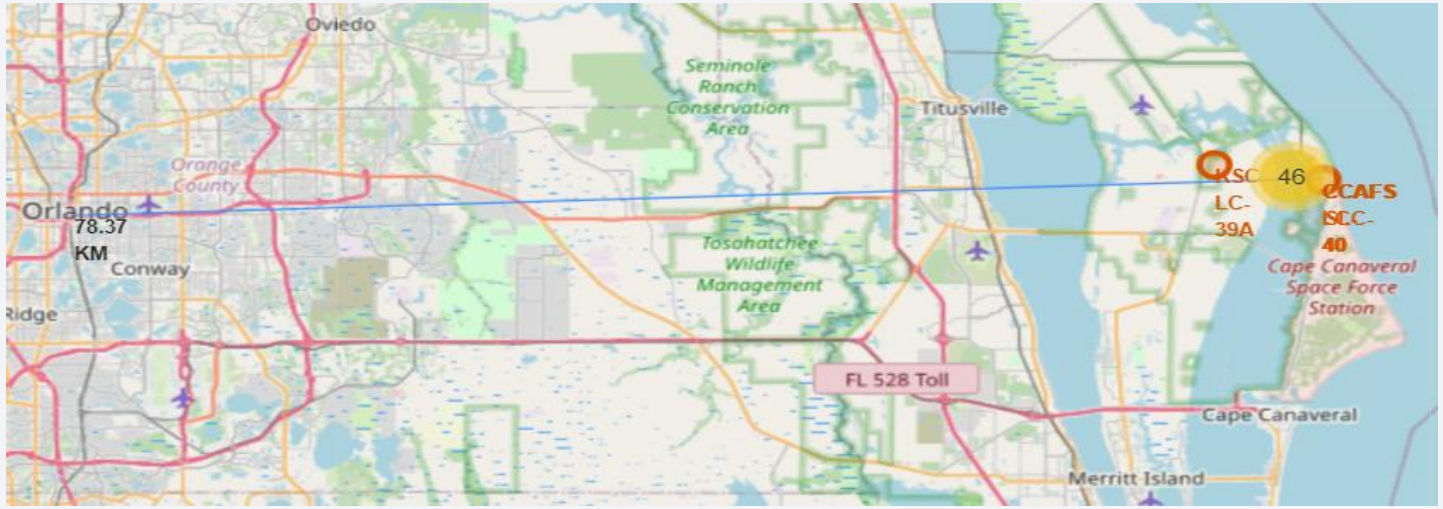
From the color-labeled markers in marker clusters, it can be easily identified which launch sites have relatively high success rates.

The green markers show successful launches and red markers show unsuccessful ones.

Most launches happen on the east coast



Map showing distance of sites from nearby physical features



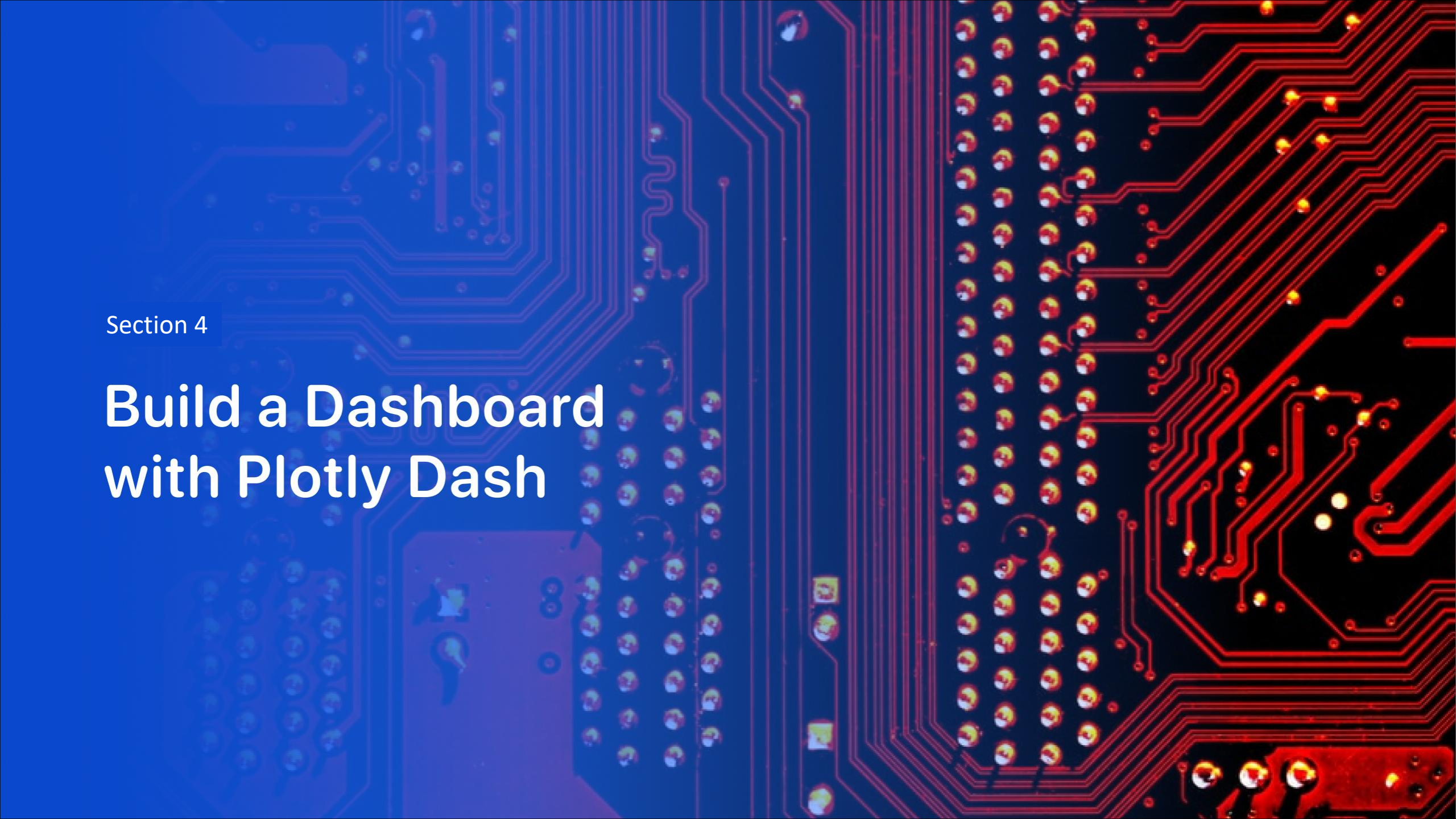
Marked the distance of site with the west coast, and calculated the distance of site from Orlando

We can observe that:

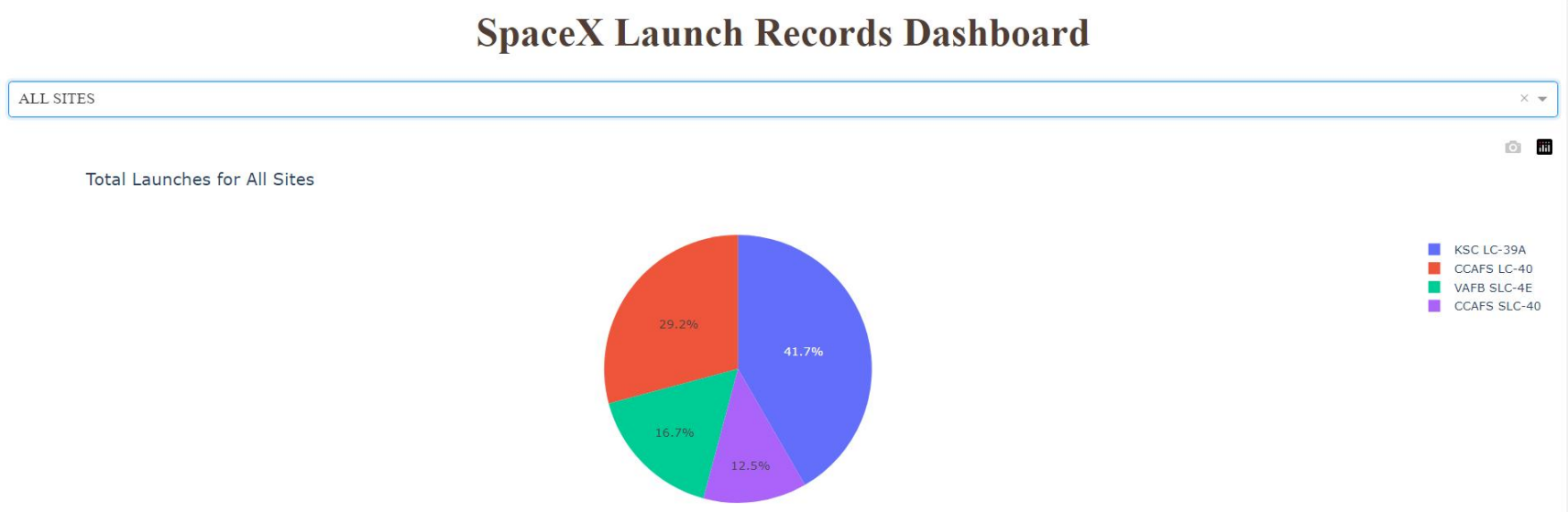
- launch sites are in close proximity to railways, highways, coastline
- Launch sites keep certain distance away from cities

Section 4

Build a Dashboard with Plotly Dash

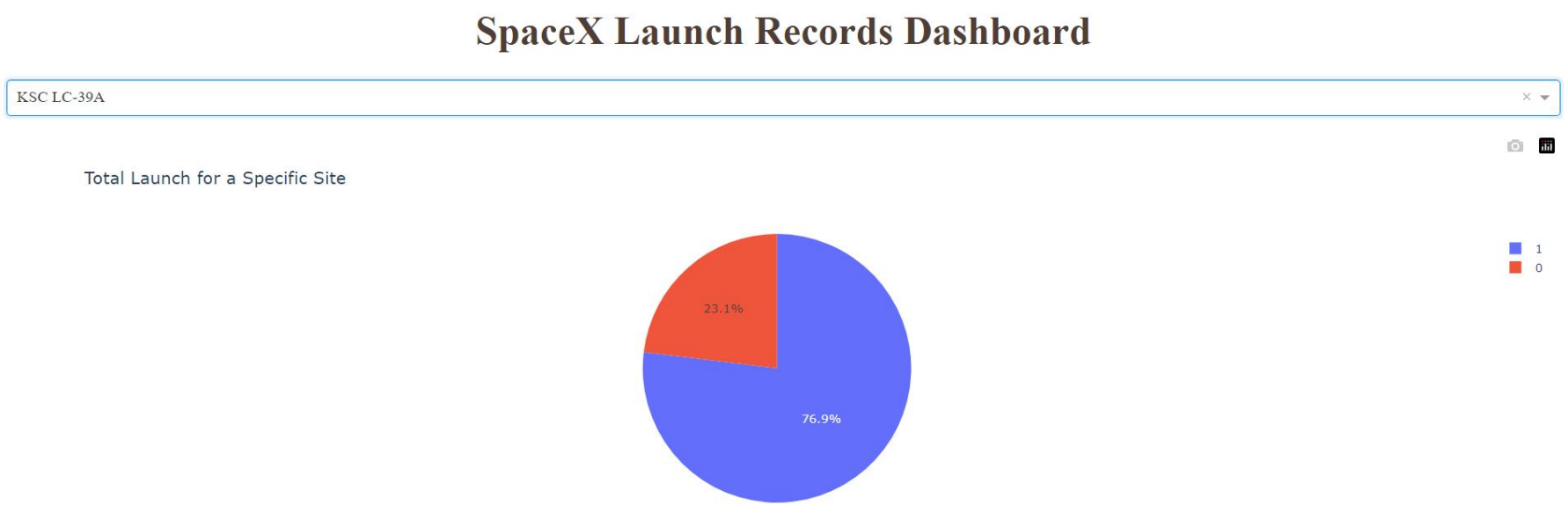


Launch success count for all sites



- Here we can see that **KSC LC-39A** has the highest percentage of launches of all the sites whereas **CCAFS SLC-40** has the lowest percentage of launches.

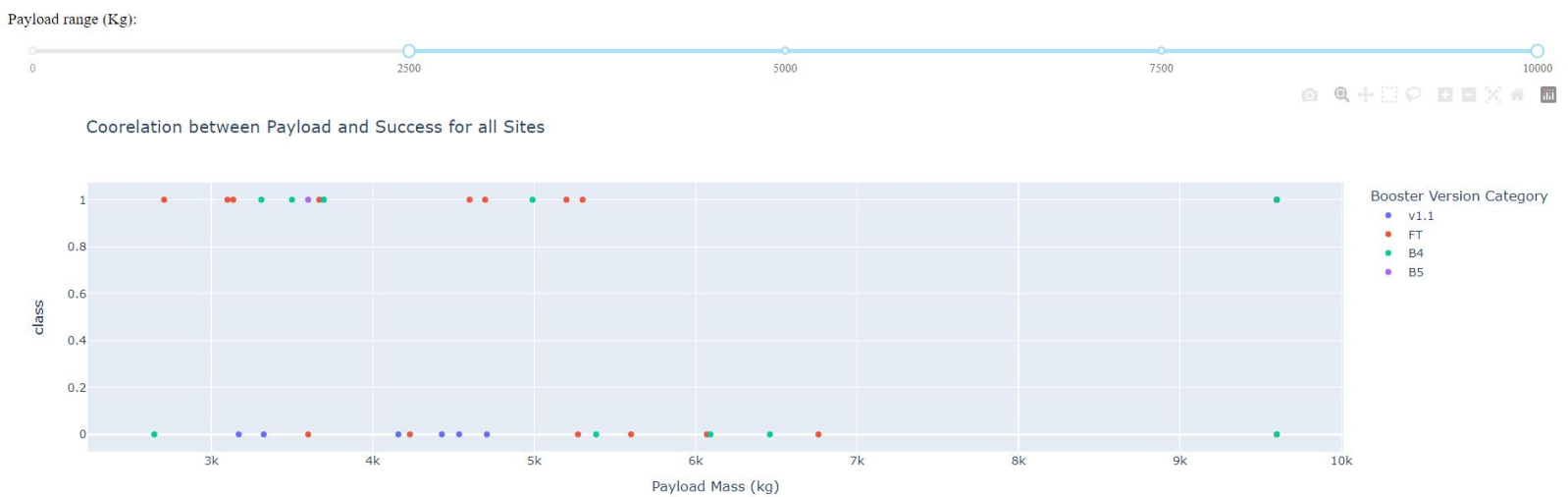
Launch site with highest launch success ratio



In this pie-chart, **KSC LC-39A** has the highest launch success ratio of all sites.
Here, we can also see that it has 76.9% success rate whereas 23.1% failure rate.

Payload vs Launch Outcome for all sites

Payload Range : 2500 - 10000kg



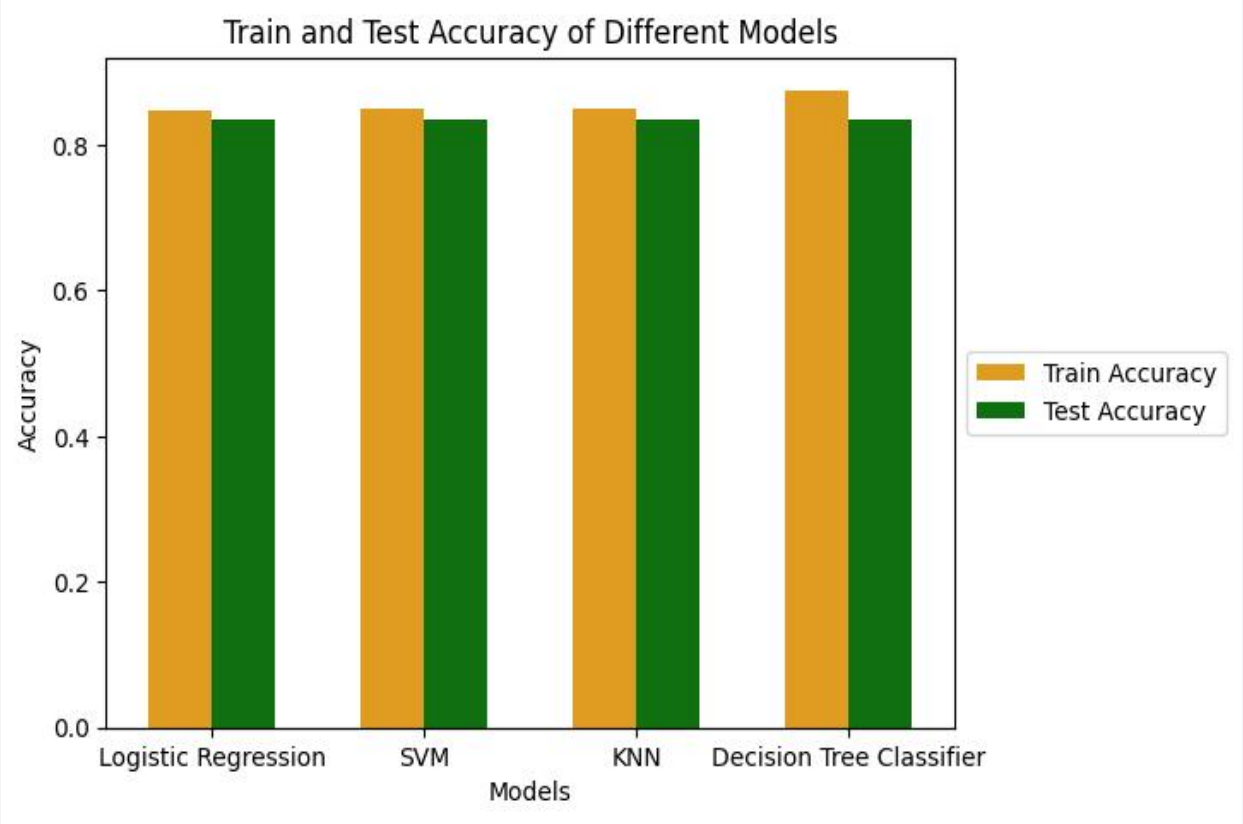
- Here we can see that B5 has the highest success rate and when Payload Mass goes after 6000kg failure rate increases.

Section 5

Predictive Analysis (Classification)

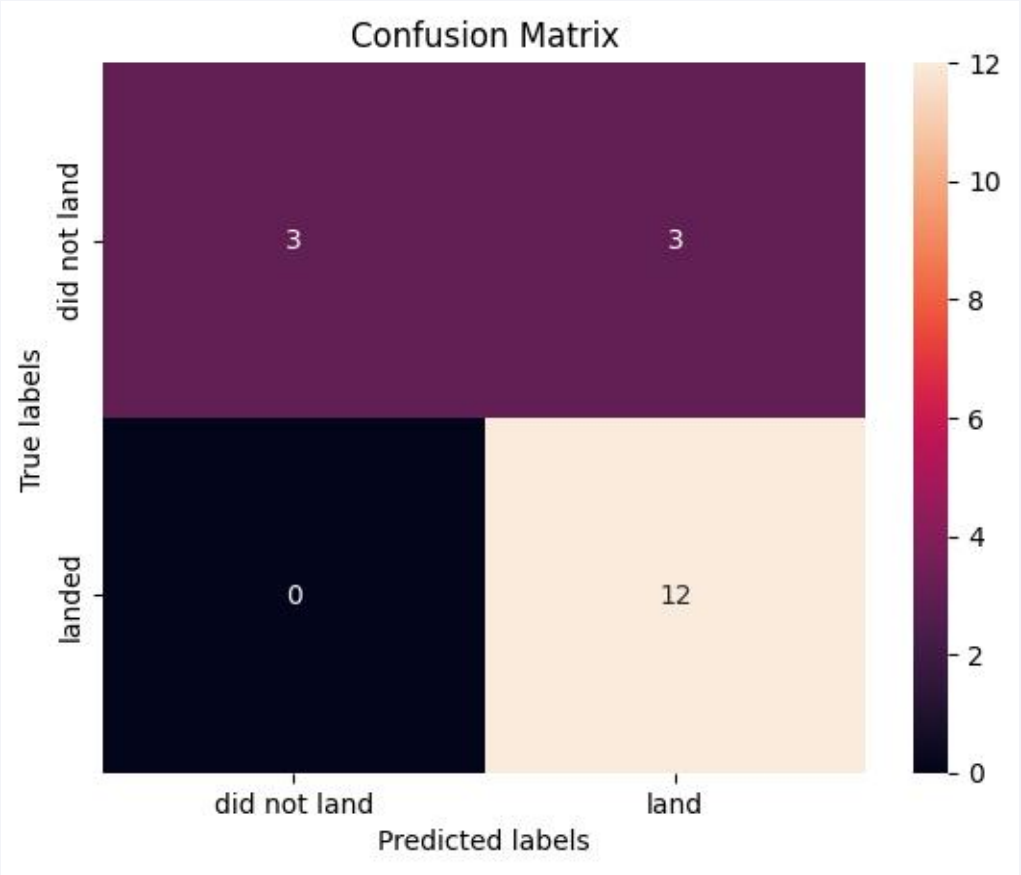
Classification Accuracy

- In this bar plot for Train and Test Accuracy of different classification models see that Decision Tree Classifier has the highest train accuracy, whereas all of the classifier have the same test accuracy (83.34%).



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., unsuccessful landing marked as successful landing by the classifier.



Conclusions

We can conclude that:

- 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.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!