IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science


Kiran Chalancharla

28 March 2023

# Outline

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

# Executive Summary

- Summary of methodologies
  - Data Collection from different sources API's , Web Scraping
  - Exploratory Data Analysis: Identify Patterns using Data wrangling, Data Visualization with Charts and SQL to analyze data and find relationships
  - Visual Analytics using Folium and Interactive Dashboards using python for team and stakeholder engagement
  - Predictive Analysis using difference Machine Leaning models to find the best model for train and predict
- Summary of all results
  - After analyzing the collected data, the Decision Tree classification model is perfect for prediction, further sections provide the explanation of each methodology to the concluded results

# Introduction

- The commercial space age is here, companies are making space travel affordable for everyone. Different existing companies like Virgin Galactic, Rocket Lab, Blue Origin, and SpaceX are competing each other in this journey.
- The company Space Y would like to compete with SpaceX, company assigned the project to create the machine learning pipeline to predict the landing outcome of the first stage
- Space X is the most successful company among companies, using the public data of SpaceX  to find the answer with machine learning models to predict the landing outcome of the first stage

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - Data collection from different sources:
    - Launch data with vehicle information from SpaceX public API
    - Falcon9 Launch data from Wiki
- Perform data wrangling
  - Data cleansing and Data transformation on collected Data
- 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

Data sets are collected from SpaceX Public API https://api.spacexdata.com/v4/launches/past

*(for Project purpose: https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json)*

Lookup data is collected form corresponding APIs as below:

for rocket details: https://api.spacexdata.com/v4/rockets/
for Launch Pad details: https://api.spacexdata.com/v4/launchpads/
for Payload details: https://api.spacexdata.com/v4/payloads
for core details: https://api.spacexdata.com/v4/cores/

Falcon 9 launch Dataset is also collected from Wiki pages
https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

# Data Collection – SpaceX API

1. Data collection through API Get request
   `response = requests.get(spacex_url)`
2. Normalize json data to dataframe
   `data = pd.json_normalize(response.json())`
3. Update ID columns with details using respective API calls
4. Filter data and remove columns
   `data_falcon9 = data[(data['BoosterVersion']!='Falcon 1')]`
5. Deal with missing data
   Replace the null values with mean

GitHub URL:
jupyter-labs-spacex-data-collection-api.ipynb

# Data Collection - Scraping

Collect data using get request

response = requests.get(static_url)

→

Parse html Data

soup = BeautifulSoup(response.text, "html.parser")

→

Extract Required Data table from list

→

Parse the data to Dataframe

GitHub URL:
[jupyter-labs-webscraping.ipynb]

# Data Wrangling

In Data Wrangling below tasks are performed

- Exploratory Data Analysis
- Determine Training Labels

Explore the Data: group & summarize

- percentage of nulls on each column
- number of launches on each site
- number and occurrence of each orbit
- number and occurence of mission outcome per orbit type

Training Label:

landing outcome label from Outcome column

GitHub URL:

labs-jupyter-spacex-Data wrangling.ipynb

# EDA with Data Visualization

- Data explored visually using various charts like Scatter, bar and line to identify the relationships, patterns, trends and outcomes
  - **relationship between Flight Number and Launch Site**
  - **relationship between Payload and Launch Site**
  - **relationship between success rate of each orbit type**
  - **relationship between Payload and Orbit type**
  - **launch success yearly trend**

GitHub URL:
[jupyter-labs-eda-dataviz.ipynb]

# EDA with SQL

- Data is further explored using SQLite for outcomes
  - Identify names of the launch sites.
  - Top 5 records launch sites begin with the string ‘CCA’.
  - total payload mass carried by booster launched by NASA (CRS).
  - Average payload mass carried by booster version F9 v1.1.
  - Date when the first successful landing outcome in ground pad was achieved.
  - Names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - Total number of successful and failure mission outcomes.
  - Names of the booster_versions which have carried the maximum payload mass.
  - Failed landing_outcomes in drone ship, their booster versions, and launch sitesnames for in year 2015.
  - Rank the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20, in descending order.

GitHub URL:
[jupyter-labs-eda-sql-coursera_sqllite.ipynb]

# Build an Interactive Map with Folium

- Integrative Folium maps are created with objects:
  - Circle to show Launch site
  - Makers to show Launch site names, distance measure, Launch outcome success of failure using folium icon
  - Marker Cluster to show all success and failures at launch site
  - Polyline to show distance of launch site proximities

GitHub URL:
lab_jupyter_launch_site_location.ipynb

# Build a Dashboard with Plotly Dash

Created dashboard for interactive data visualization using Plotly Dash:

- Pie chart showing the success outcome of all launch sites or individual site
- Scatter chart showing the relationship between payload and success outcome of booster versions with dynamic payload range selection

The interactive dashboard is created to find more insights from the SpaceX dataset

GitHub URL:
spacex_dash_app.py

# Predictive Analysis (Classification)

**Prepare the data for Model**

- Standardize and transpform the data
- Split the data into training and test datasets

**Train the different models**

- Train the models with different hyperparameters using GridSearchCV
  - Logistic Regression, support vector machine, Decision Tree KNN

**Validate model**

- Find Build Accuracy, Test Accuracy using Test Data
- Draw Confusion Matrix

**Compare Models**

- Compare Models using build accuracy
- Choose the best model for predict the data

GitHub URL:
SpaceX_Machine Learning Prediction_Part_5.ipynb

# Results

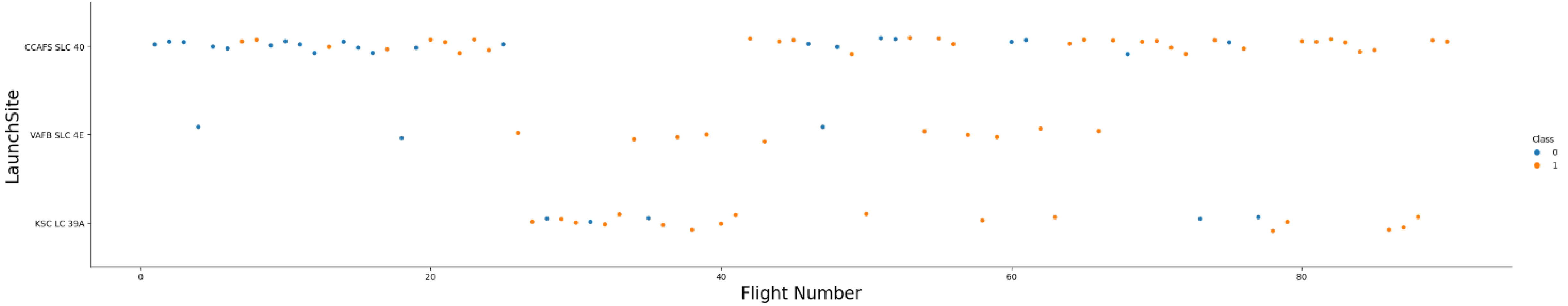
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

# Insights drawn from EDA

# Flight Number vs. Launch Site

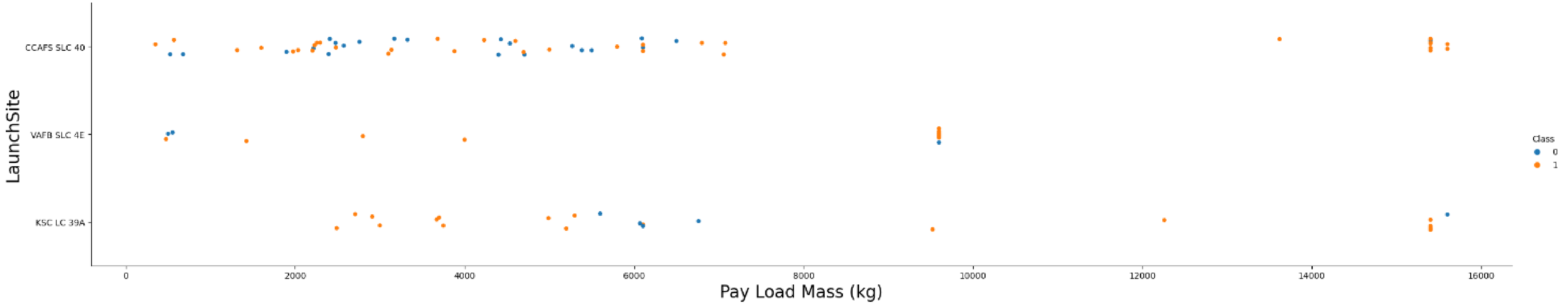
```
[4]: # Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```




- Scatter plot shows that success rate is high over the years on all launch sites.
- Launch sites are initially with experimental until success is high, later with increase in demand the launch sites are added to meet the demand

# Payload vs. Launch Site

```
[5]: # Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay Load Mass (kg)",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```

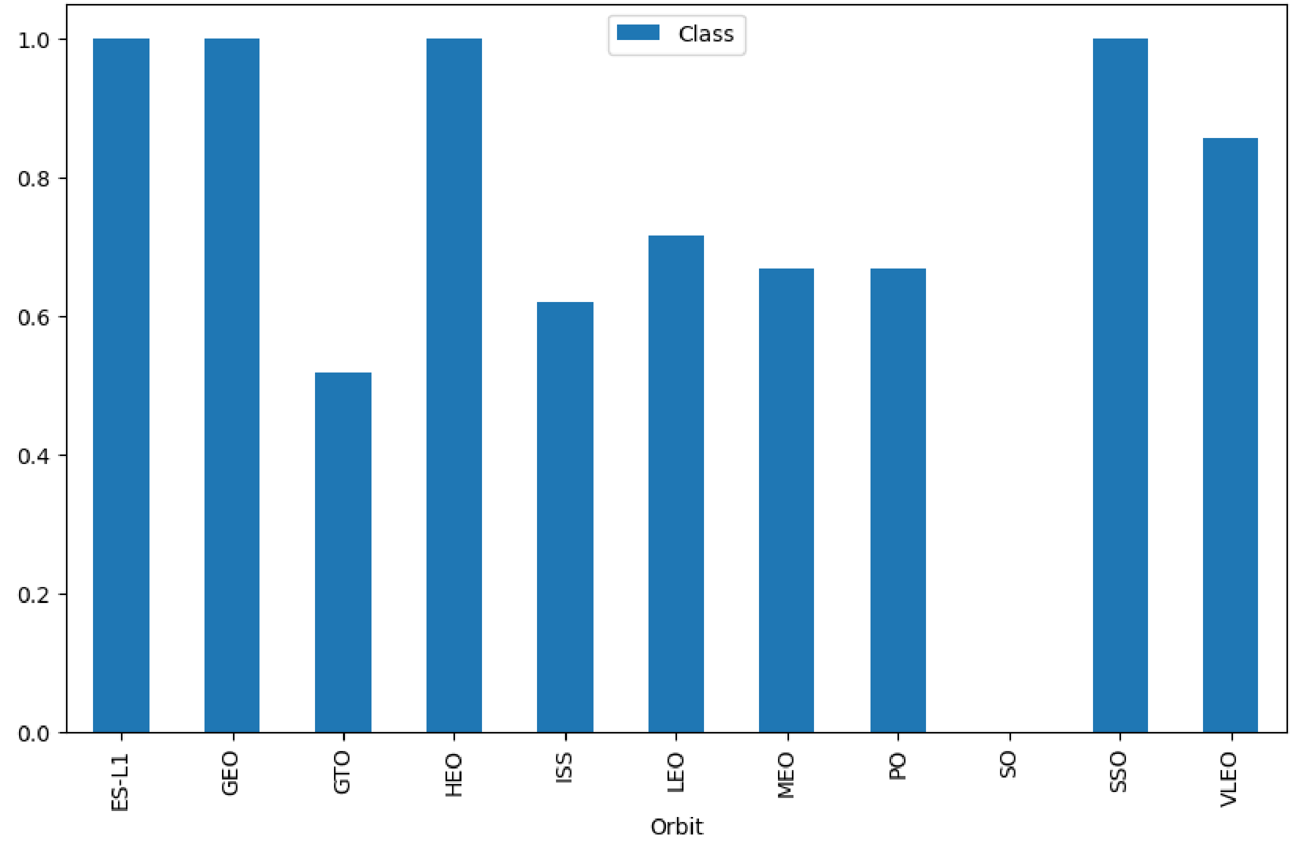



- Scatter plot shows that most of the launches are with payload mass below 7500
- Major launches are planned from Launch site CCAFS SLC-40
- Very few are planned with higher payload mass exceeding 15000

# Success Rate vs. Orbit Type

```
[6]: # HINT use groupby method on Orbit column and get the mean of Class column
     dfbar = df.groupby(['Orbit']).aggregate({'Class':'mean'})
     dfbar.plot(kind='bar', figsize=(10, 6))
```

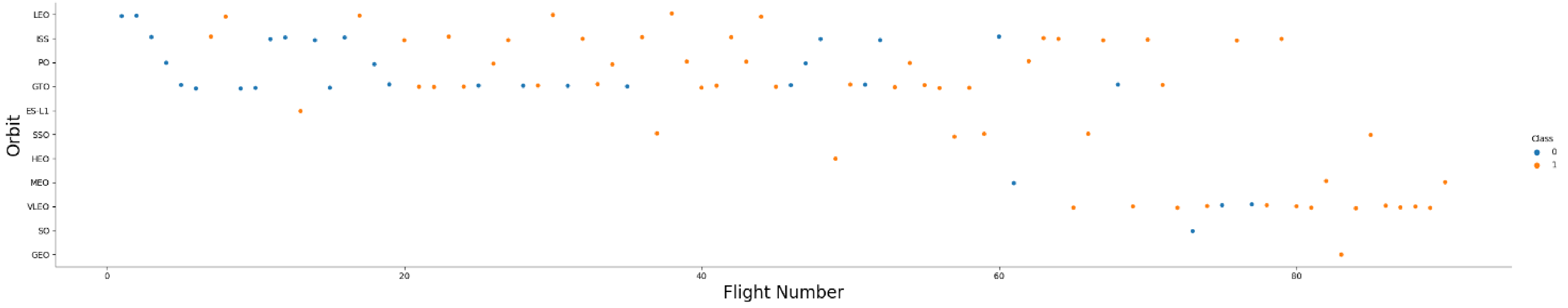
```
[6]: <AxesSubplot: xlabel='Orbit'>
```




- Bar Chart shows that Success rate for orbit's ES-L1, GEO, HEO and SSO is high
- Success Rate is very low for SO and GTO

# Flight Number vs. Orbit Type

```
[7]: # Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

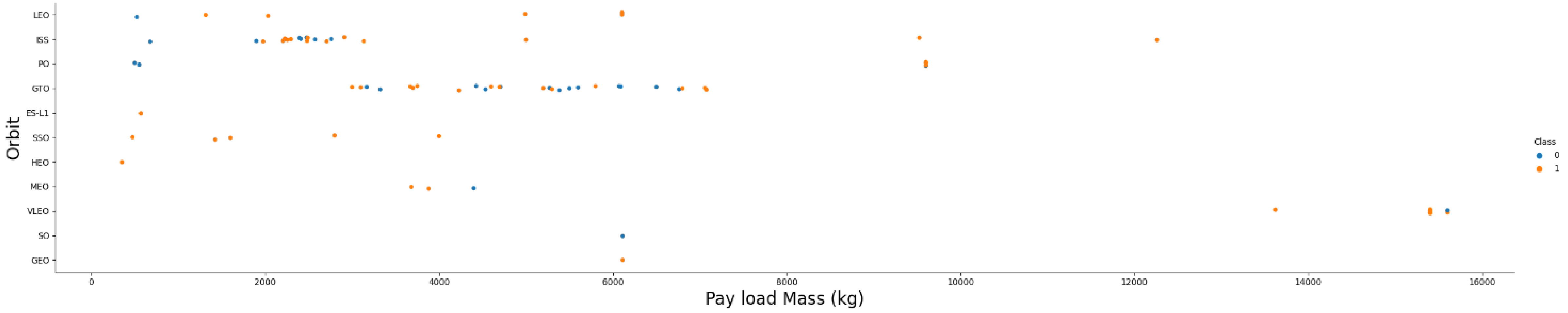



- Scatter plot shows that the higher success rate is achieved over the years (based on the experience)
- Initial launches are targeted between Low Earth Orbit to Geosynchronous Earth Orbit
- latest launches with heavy payload are targeted below Low Earth Orbit, i.e., VLEO (Very Low Earth Orbit)

# Payload vs. Orbit Type

[8]:

```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



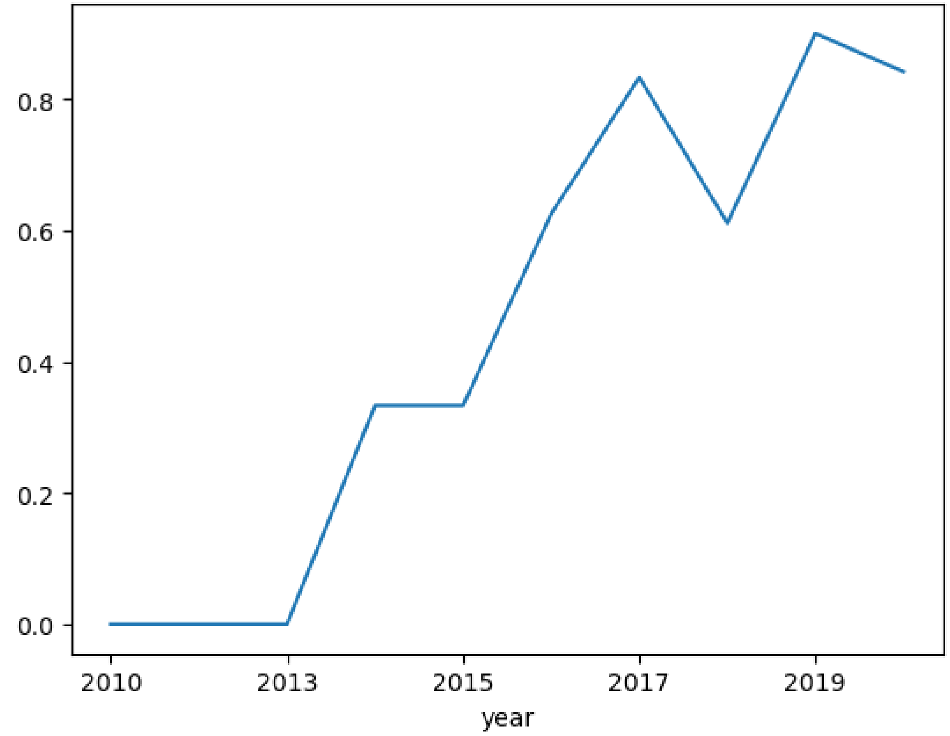

- Scatter chart shows that heavy payloads are launched targeting below low earth orbits
- Payload ranging from 3000 to 7500 are  launched targeting GTO (Geosynchronous Earth Orbit) and MEO

# Launch Success Yearly Trend

```
[11]: # Plot a line chart with x axis to be the extracted year and y axis to be the success rate

      dfline= pd.DataFrame(columns=['class','year'])
      dfline['class'] = df['Class'].values
      dfline['year'] = years
      dfline.groupby('year')['class'].mean().plot()
```

```
[11]: <AxesSubplot: xlabel='year'>
```




- Line chart of yearly average success rate
- The trend line shows that the success rate is steady from 2013

# All Launch Site Names

```
[6]: %sql SELECT DISTINCT "Launch_Site" FROM SPACEXTBL

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

[6]:

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

- The launch data presented is collection from 5 launch sites

# Launch Site Names Begin with 'CCA'

```
[7]: %sql SELECT * from SPACEXTBL where  "Launch_Site" like 'CCA%' LIMIT 5

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

[7]:

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome | Landing _Outcome |
|---|---|---|---|---|---|---|---|---|---|
| 04-06-2010 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success | Failure (parachute) |
| 08-12-2010 | 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) |
| 22-05-2012 | 07:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success | No attempt |
| 08-10-2012 | 00:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500 | LEO (ISS) | NASA (CRS) | Success | No attempt |
| 01-03-2013 | 15:10:00 | F9 v1.0 B0007 | CCAFS LC-40 | SpaceX CRS-2 | 677 | LEO (ISS) | NASA (CRS) | Success | No attempt |

- The data shows that, some launches are planned even without any payload mass (0 payload mass)

# Total Payload Mass

```
[8]: %sql select sum(PAYLOAD_MASS__KG_) "Total_Payload" from SPACEXTBL where  "Customer" = 'NASA (CRS)'

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

[8]:

| Total_Payload |
| --- |
| 45596 |

- Total payload mass carried by boosters from NASA is 45596

# Average Payload Mass by F9 v1.1

```
In [10]: %sql select AVG(PAYLOAD_MASS__KG_) "Average_Payload"  from SPACEXTBL where  "Booster_Version" = 'F9 v1.1'

 * sqlite:///my_data1.db
Done.

Out[10]:
```

| Average_Payload |
|---|
| 2928.4 |

- Average payload of booster version F9 v1.1 is 2928.4

# First Successful Ground Landing Date

```
[12]: %sql select min(Date) from SPACEXTBL where  "Landing _Outcome" = 'Success (ground pad)'

 * sqlite:///my_data1.db
Done.

[12]:  min(Date)
       01-05-2017
```

- The first success on ground pad is on 1st May 2017

# Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

[13]:
```
%sql select DISTINCT 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.
```

[13]:

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

- Data shows that 4 booster versions have successful landing with payload mass between 4000 and 6000

# Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
[16]: %sql select UPPER(TRIM(Mission_Outcome)) Mission_Outcome, count(1) nums from SPACEXTBL group by UPPER(TRIM(Mission_Outcome))
```

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

[16]:

| Mission_Outcome | nums |
|---|---|
| FAILURE (IN FLIGHT) | 1 |
| SUCCESS | 99 |
| SUCCESS (PAYLOAD STATUS UNCLEAR) | 1 |

- Data shows that the mission success as per plan are 99, failure is 1 and unclear is 1

# Boosters Carried Maximum Payload

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

```
[17]: %sql select DISTINCT Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)

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

[17]:

| 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 |

- Data shows that 12 booster versions carried the maximum pay load

# 2015 Launch Records

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, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.**

```
[18]: %sql select substr(Date, 4, 2) as month, "Landing _Outcome", Booster_Version, Launch_Site from SPACEXTBL where "Landing _Outcome" = 'Failure (drone ship)'  and substr(Date,7,4)='2015'
```

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

[18]:

| 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 |

- Data shows that there are 2 failures during year 2015 while landing on Drone ship launched from site CCAFS LC-40

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

[24]:
```
%%sql
select "Landing _Outcome", count(1) rank from SPACEXTBL where date(substr(Date,7,4) || '-' || substr(Date, 4, 2) || '-' ||  substr(Date, 1, 2))
between date('2010-06-04') and date('2017-03-20') group by "Landing _Outcome"  order by rank desc
```

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

[24]:

| Landing _Outcome | rank |
|---|---|
| No attempt | 10 |
| Success (drone ship) | 5 |
| Failure (drone ship) | 5 |
| Success (ground pad) | 3 |
| Controlled (ocean) | 3 |
| Uncontrolled (ocean) | 2 |
| Failure (parachute) | 2 |
| Precluded (drone ship) | 1 |

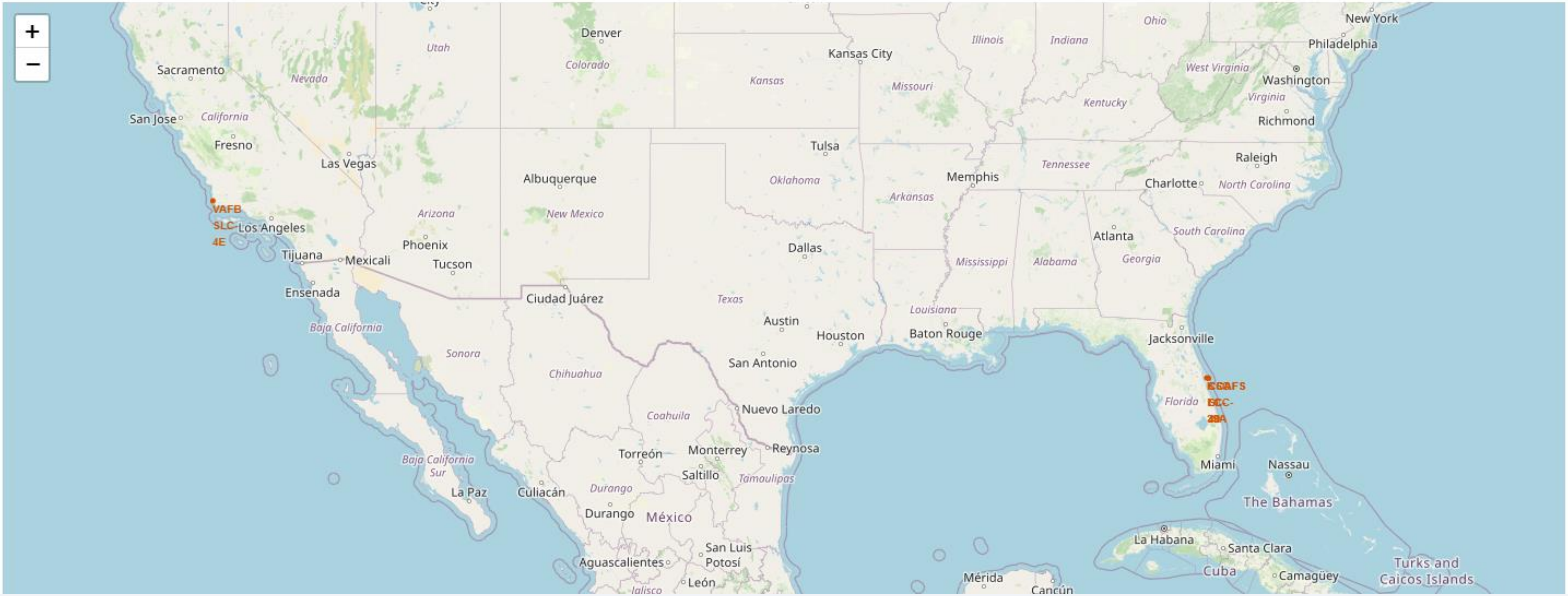
- Ranking count of all landing outcomes between the date 2010-06-04 and 2017-03-20, in descending order
- Ranking represents that the data include the Launches without landing attempts.

Section 3

# Launch Sites Proximities Analysis

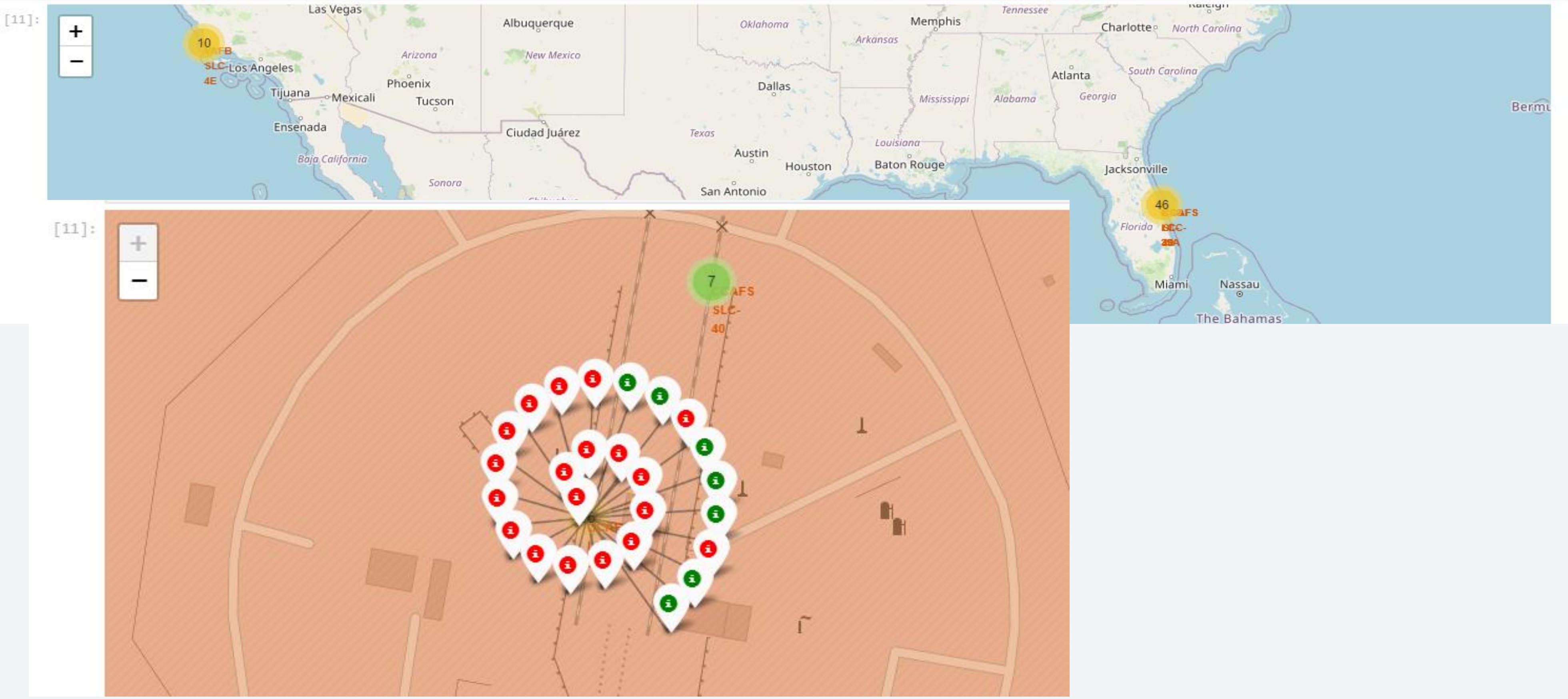
# Geolocation of All Launch sites

[7]:



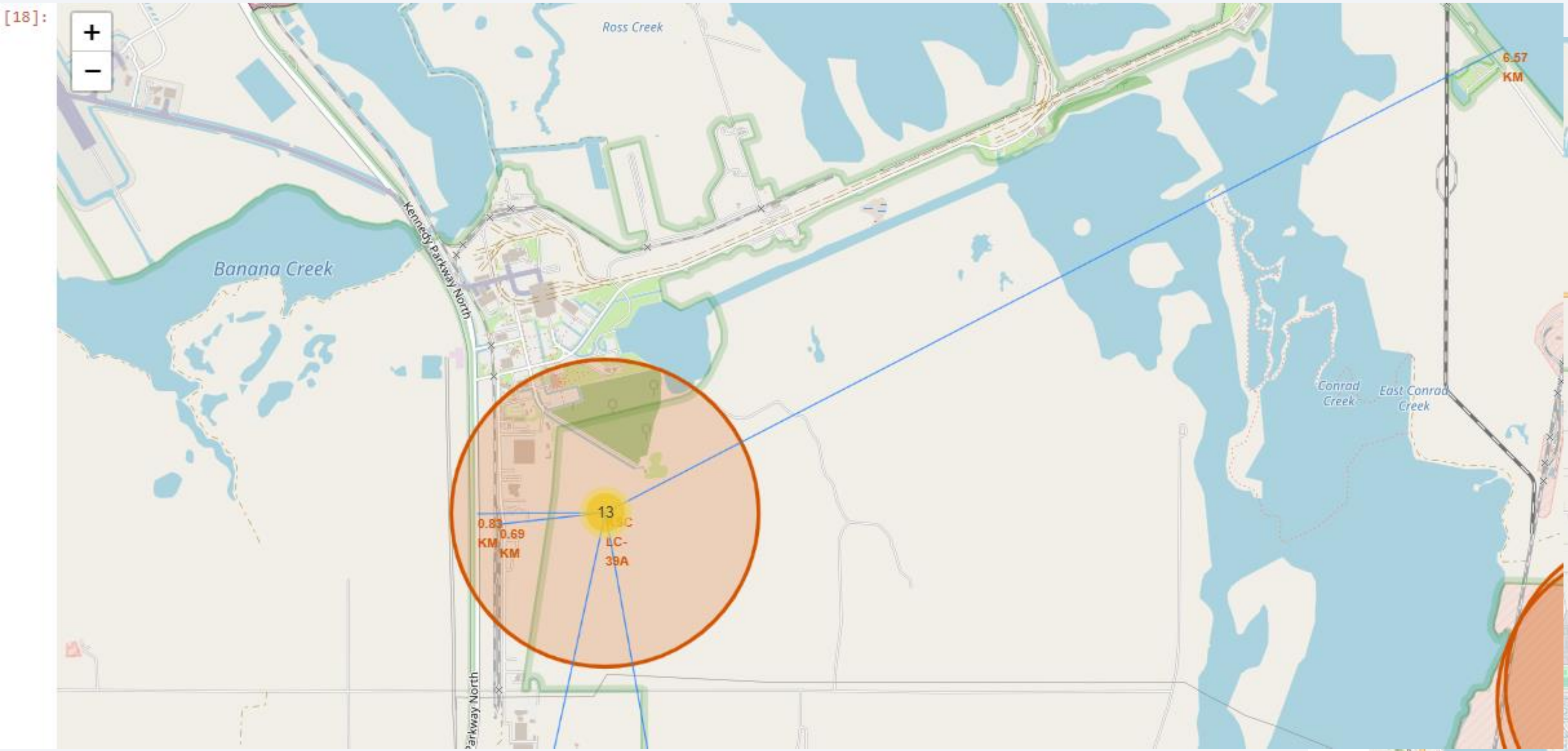

- Map showing the Launch sites geolocation.
- All Launch sites are located near to coastline

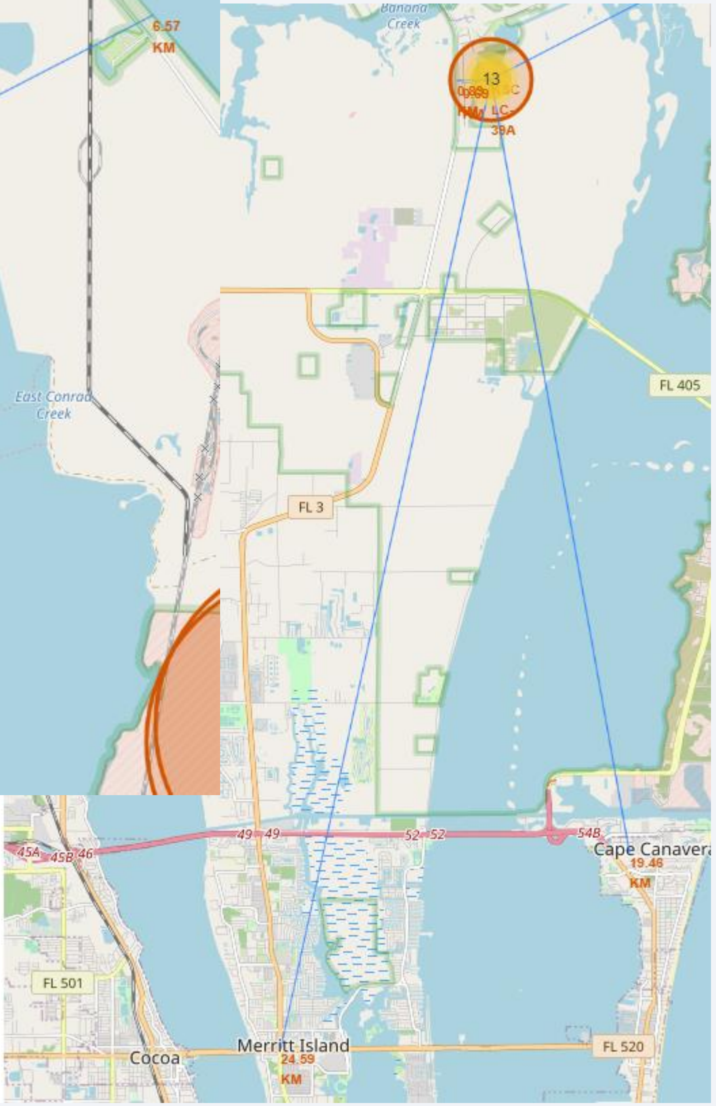
# Launch Outcomes of Launch Site - CCAFS LC-40




- Maps shows the cluster of success and failure launces of sites
- Further zooming to launch site, shows each launch success or failure using marker icon indicating green as success, red as failure.

# Proximities for Launch Site KSC LC-39A



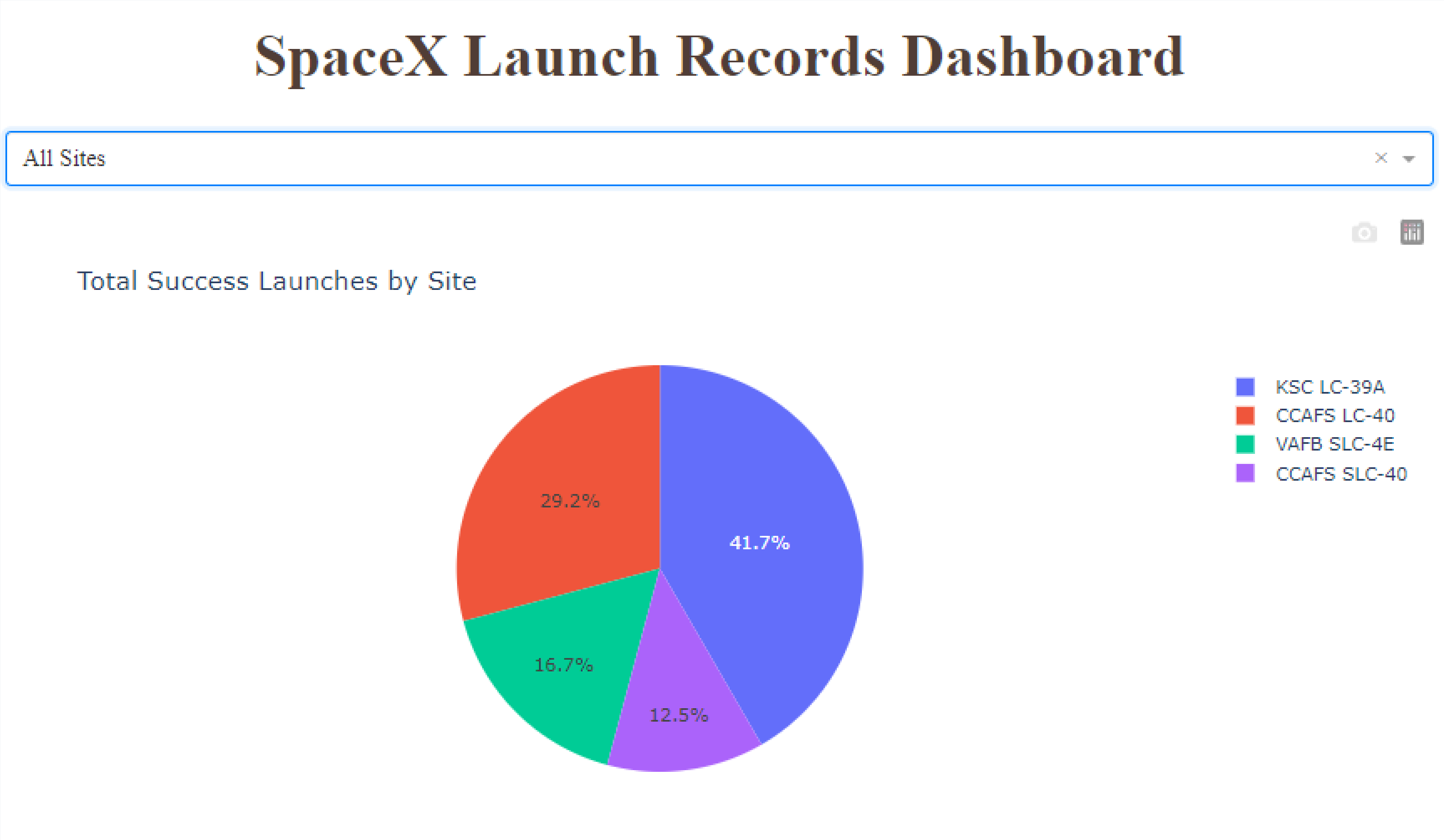




- Maps clearly shows that the launch sites are near to coasts and far away from human settlements (cities, towns)
- Launch sites are near to connectivity network like Roadways, railways for logistics

Section 4

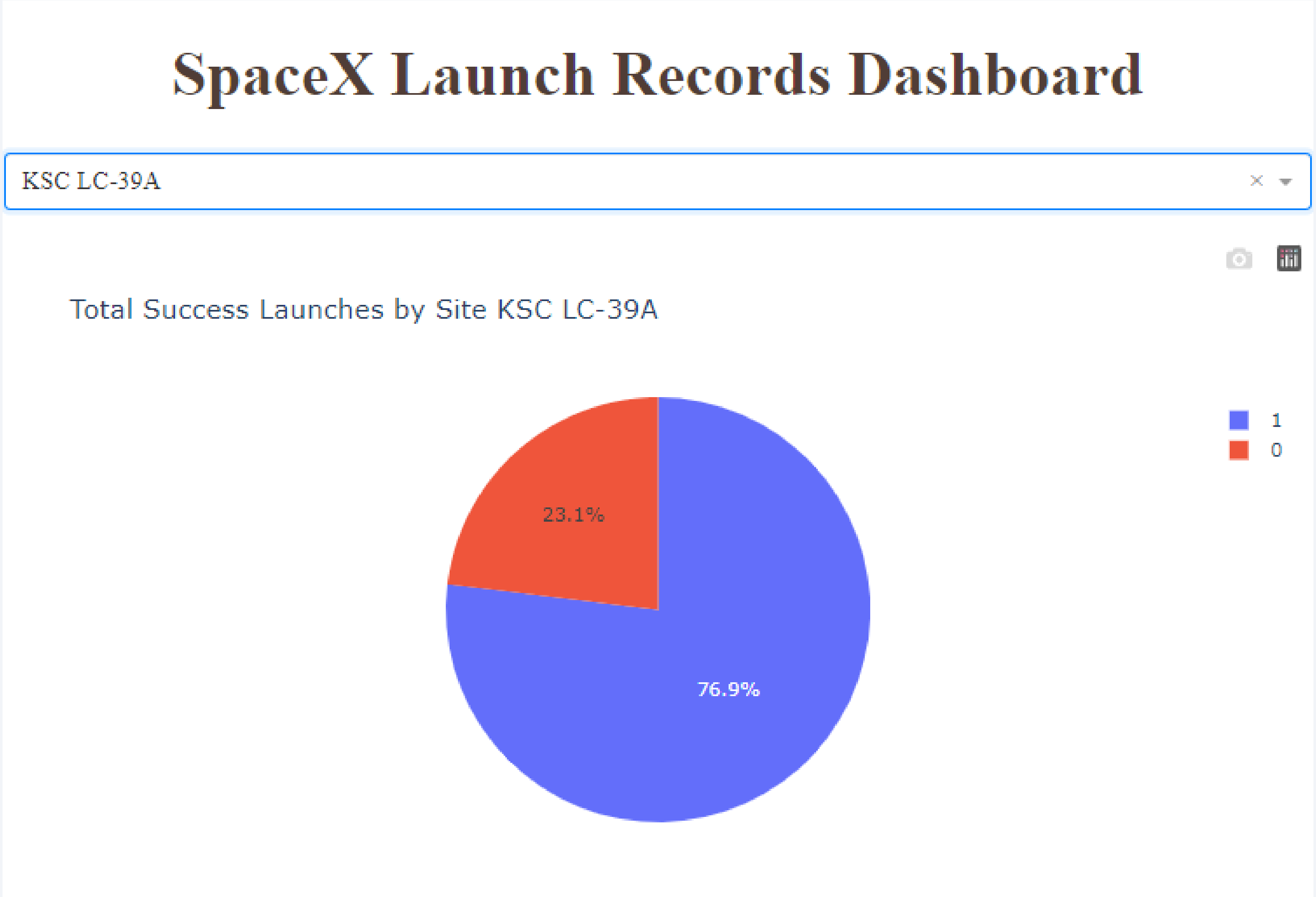
# Build a Dashboard with Plotly Dash

# Launch Success Distribution for all Sites



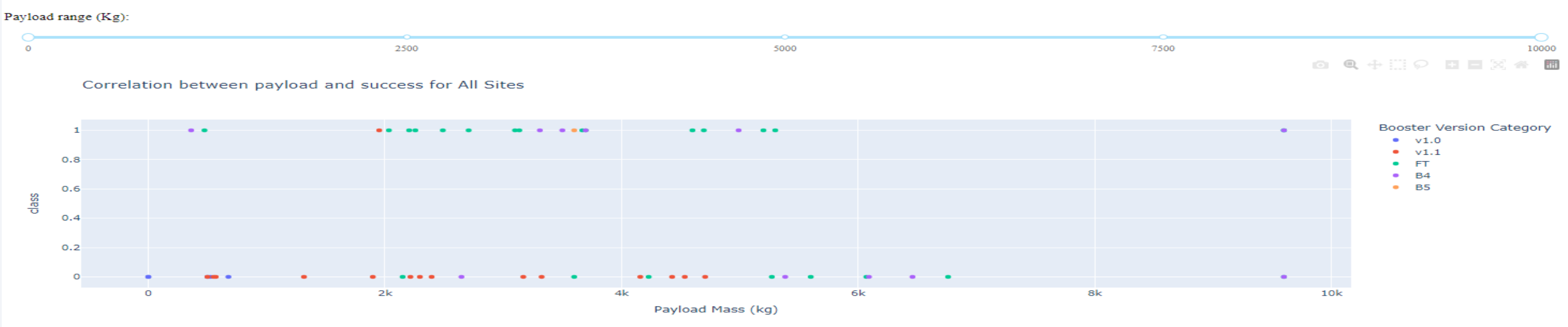

- From chart it is quite evident that Launch site KSC LC-39A has the highest success ratio of 41.7%

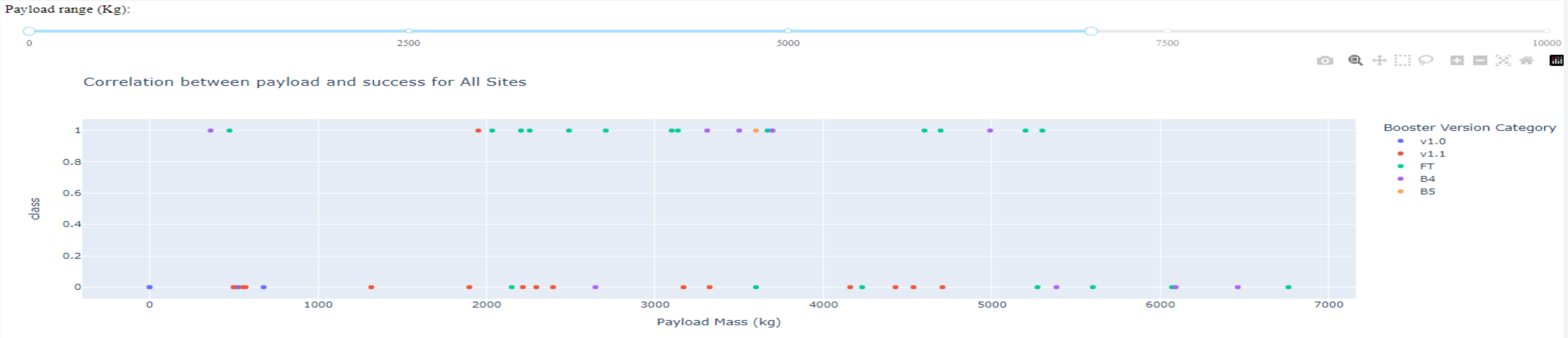
# Highest Launch Success Ratio Site : KSC LC-39A

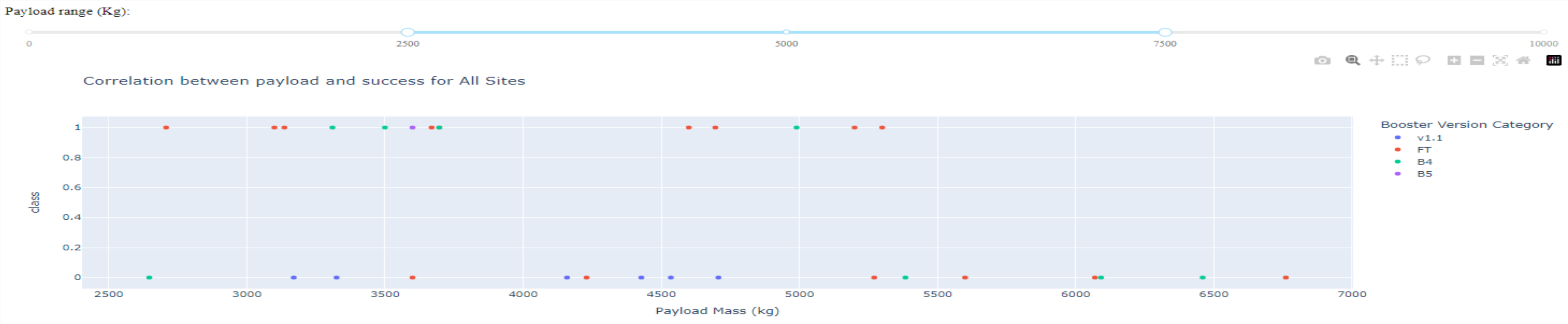



- From the launch site wise visualization of success ratio, KSC LC-39A launch site clearly shows that the success ratio is higher compared to other Launch site.
- Success ration is 76.9% and Failure Ratio is 23.1%

# Launch Outcome at Different Range of Payloads




- Success rate is high for payloads between 2000 to 5500
- FT booster version has higher success rate for payloads between 2500 to 5500

Section 5

# Predictive Analysis (Classification)

# Classification Accuracy

[95]:
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 2, 'sp
litter': 'best'}
accuracy : 0.875
```

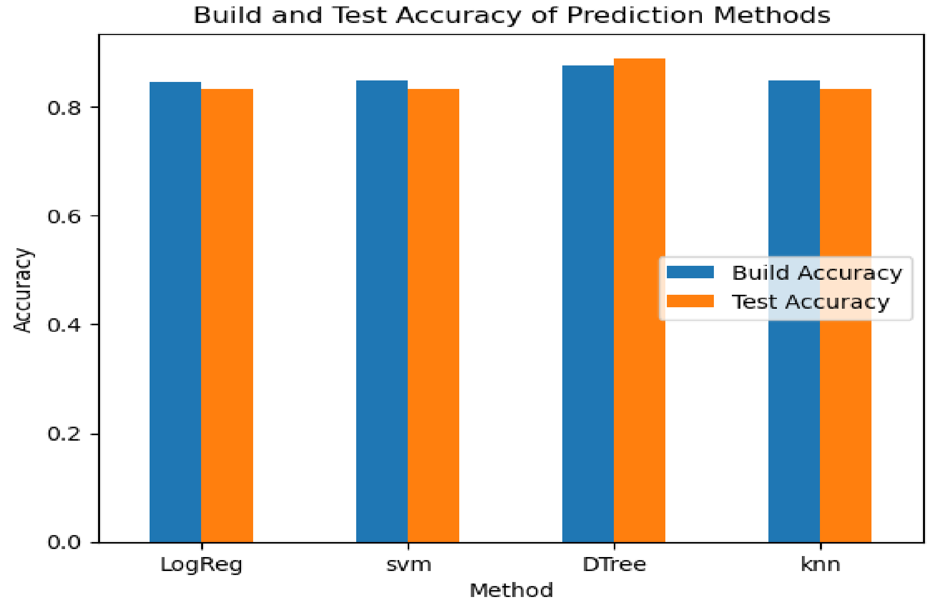
[99]:

| | Method | Method-Abr | Build Accuracy | Test Accuracy |
|---|---|---|---|---|
| 0 | Logistic Regression | LogReg | 0.846429 | 0.833333 |
| 1 | support vector machine | svm | 0.848214 | 0.833333 |
| 2 | Decision Tree | DTree | 0.875000 | 0.888889 |
| 3 | k nearest neighbors | knn | 0.848214 | 0.833333 |

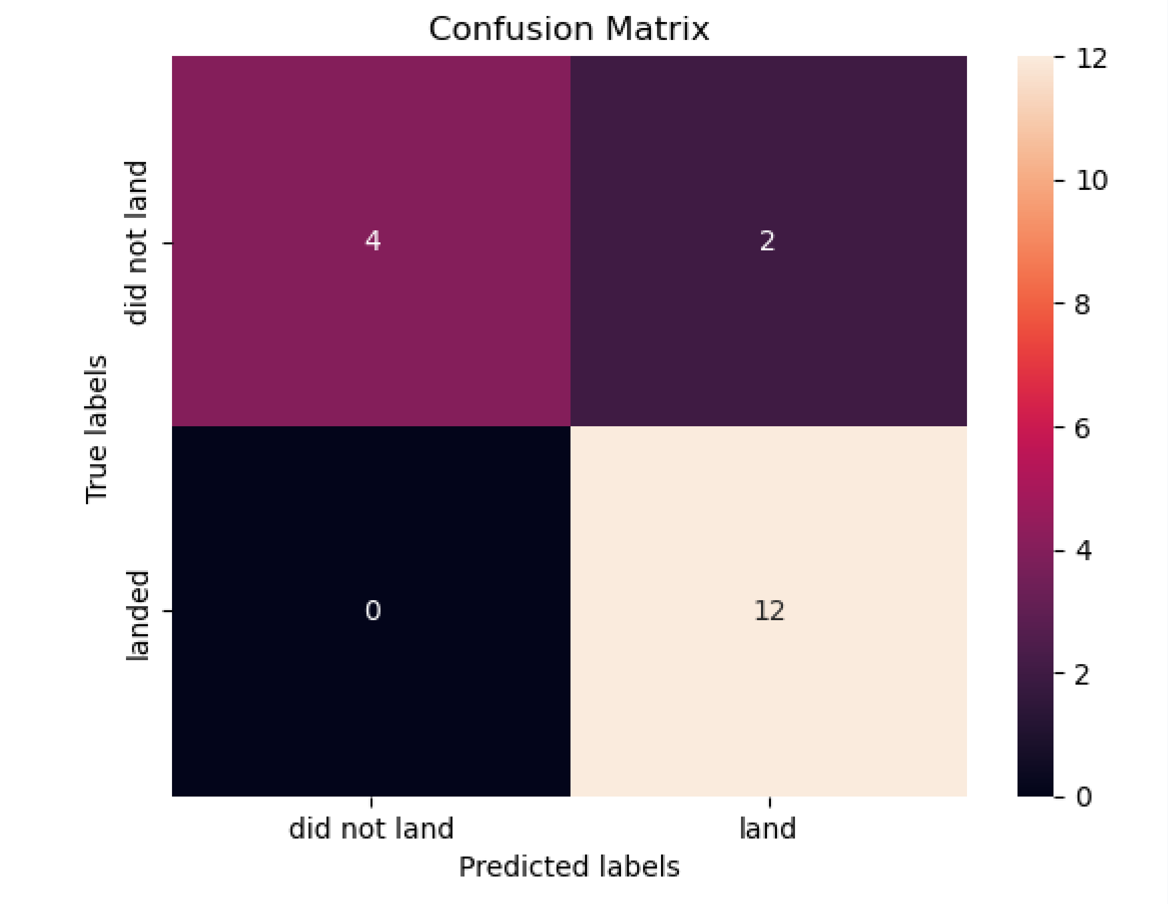
- From the table, showing the build accuracy and test accuracy of each model, it is evident that Decision Tree classification has higher accuracy.
- During the training of Decision Tree model, on multiple iterations of the accuracy is achieved (this is the due to max_features “sqrt” restriction)

[100]:
```
#using Pandas
barplt=methoddf.plot.bar(x='Method-Abr')
plt.xlabel("Method", rotation=0)
plt.ylabel("Accuracy")
plt.xticks(rotation=0, horizontalalignment="center")
plt.legend(loc='right')
plt.title("Build and Test Accuracy of Prediction Methods")
```

[100]:
```
Text(0.5, 1.0, 'Build and Test Accuracy of Prediction Methods')
```



# Confusion Matrix




Confusion Matrix of Decision Tree:

1. Landings are predicted successfully
2. Failure are predicted with accuracy more than 50% which is more than other models.

# Conclusions

- Launch sites should be located near to coastline and away from human settlements, proximities like roadways, railways should be near to launch site for logistics
- Based on the data collected:
  - Launches with Heavy payload have good success rate up to Low earth orbits
  - Launches below 8000 kg payload mass have good success rate up to Geosynchronous Earth orbit
  - Higher Launch success rate can be achieved over the subsequent years
  - For predicting the success rate, Decision tree classifier gives better results based on the past launches of public data.

# Appendix

GitHub reference for this project:

https://github.com/csnvkiran/IBMDataScience-AppliedDataScience-Capstone

# Appendix

Python HTML code:

```
app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard',
                                        style={'textAlign': 'center', 'color': '#503D36',
                                               'font-size': 40}),
                                # TASK 1: Add a dropdown list to enable Launch Site selection
                                # The default select value is for ALL sites
                                # dcc.Dropdown(id='site-dropdown',...)
                                dcc.Dropdown(id='site-dropdown', options=dcc_sites, value='ALL'),

                                html.Br(),

                                # TASK 2: Add a pie chart to show the total successful launches count for all sites
                                # If a specific launch site was selected, show the Success vs. Failed counts for the site
                                html.Div(dcc.Graph(id='success-pie-chart')),
                                html.Br(),

                                html.P("Payload range (Kg):"),
                                #html.P("Minimum Payload: " + str(min_payload)),
                                #html.P("Maximum Payload: " + str(max_payload)),
                                # TASK 3: Add a slider to select payload range
                                #dcc.RangeSlider(id='payload-slider',...)
                                dcc.RangeSlider(id='payload-slider', min=0, max=10000, step=1000, marks={0: '0', 2500: '2500', 5000: '5000',
7500: '7500', 10000: '10000'},value=[min_payload, max_payload]),

                                # TASK 4: Add a scatter chart to show the correlation between payload and launch success
                                html.Div(dcc.Graph(id='success-payload-scatter-chart')),
                                ])
```

# Appendix

SQL to find the table description:

```
In [5]: %sql SELECT name,type,length(type) FROM PRAGMA_TABLE_INFO('SPACEXTBL');

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

Out[5]:

| name | type | length(type) |
|---|---|---|
| Date | TEXT | 4 |
| Time (UTC) | TEXT | 4 |
| Booster_Version | TEXT | 4 |
| Launch_Site | TEXT | 4 |
| Payload | TEXT | 4 |
| PAYLOAD_MASS__KG_ | INTEGER | 7 |
| Orbit | TEXT | 4 |
| Customer | TEXT | 4 |
| Mission_Outcome | TEXT | 4 |
| Landing _Outcome | TEXT | 4 |

# Thank you!