IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ankit Singh Chauhan
2nd September

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection using API, Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL and Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data analysis performed on SpaceX features to identify relationships and visualization.
  - Determine best model for prediction.

Introduction

- Project background and context
  - The goal is to evaluate SpaceX historical launch data and predict if the first stage will land.
  - This information can be used if an alternate company wants to big against SpaceX for launch.
- Problems you want to find answers
  - Identify factors that aid in successful launch.
  - Conditions and locations ideal for launch.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected from SpaceX API
  - Data scraped from Wikipedia
- Perform data wrangling
  - We applied one-hot encoding to transform data for better outcome.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Evaluated hyperparameters and tuning to develop the best classification model.

Data Collection

- SpaceX API
  - Data requested using SpaceX API using get request
  - Convert json data to a dataframe for processing
  - We filtered the dataframe to include only falcon 9 launches.
  - We then replaced the missing data with mean values.
- Web Scraping
  - We also collected launch data by web scraping Wikipedia page for falcon 9 launches.
  - We parsed the HTML tables to extract data and organized it as a dataframe.

Data Collection – SpaceX API

```
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 getBoosterVersion
getBoosterVersion(data)
```

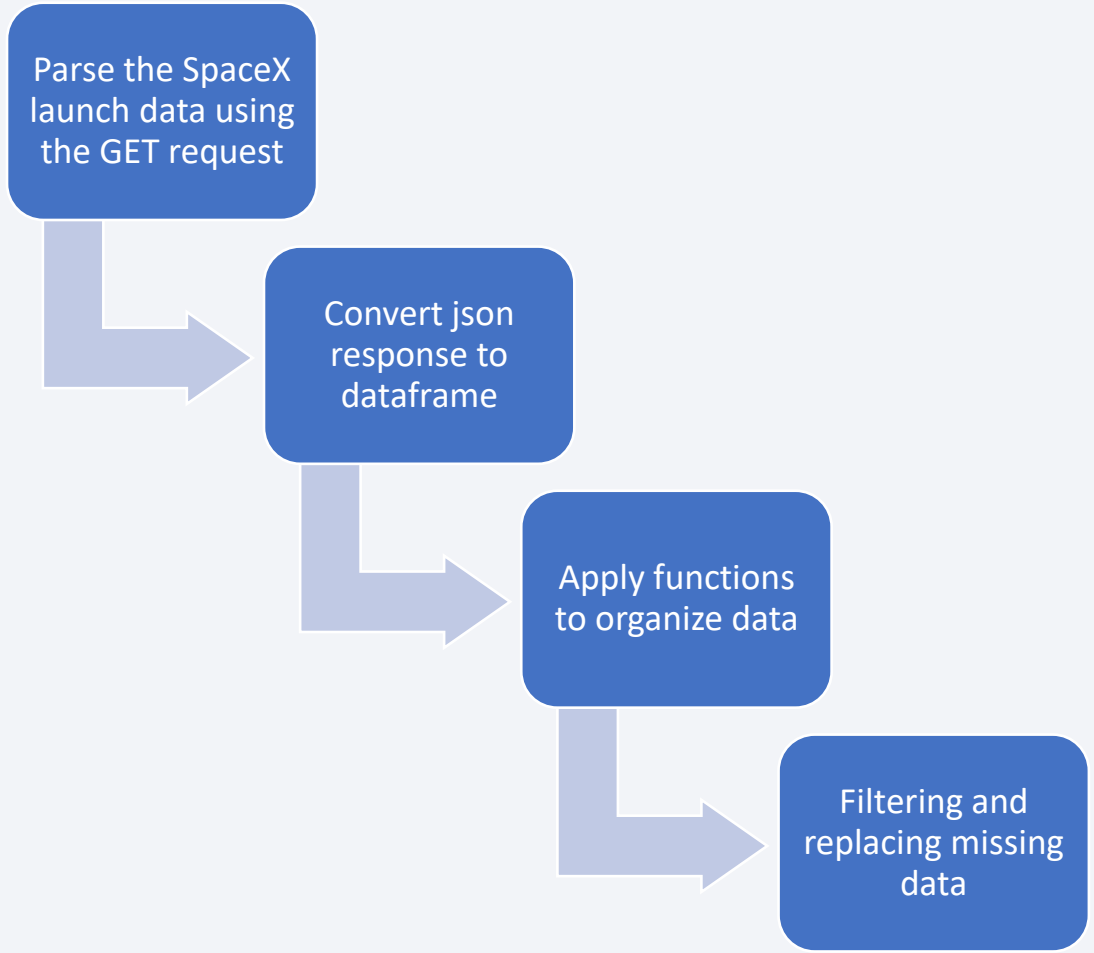
```
# Call getLaunchSite
getLaunchSite(data)
```

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

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

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = launch_data[launch_data['BoosterVersion']=='Falcon 9']
data_falcon9.head()
```

```
# Calculate the mean value of PayloadMass column
mass = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan, mass)
data_falcon9
```



Click here for Github URL

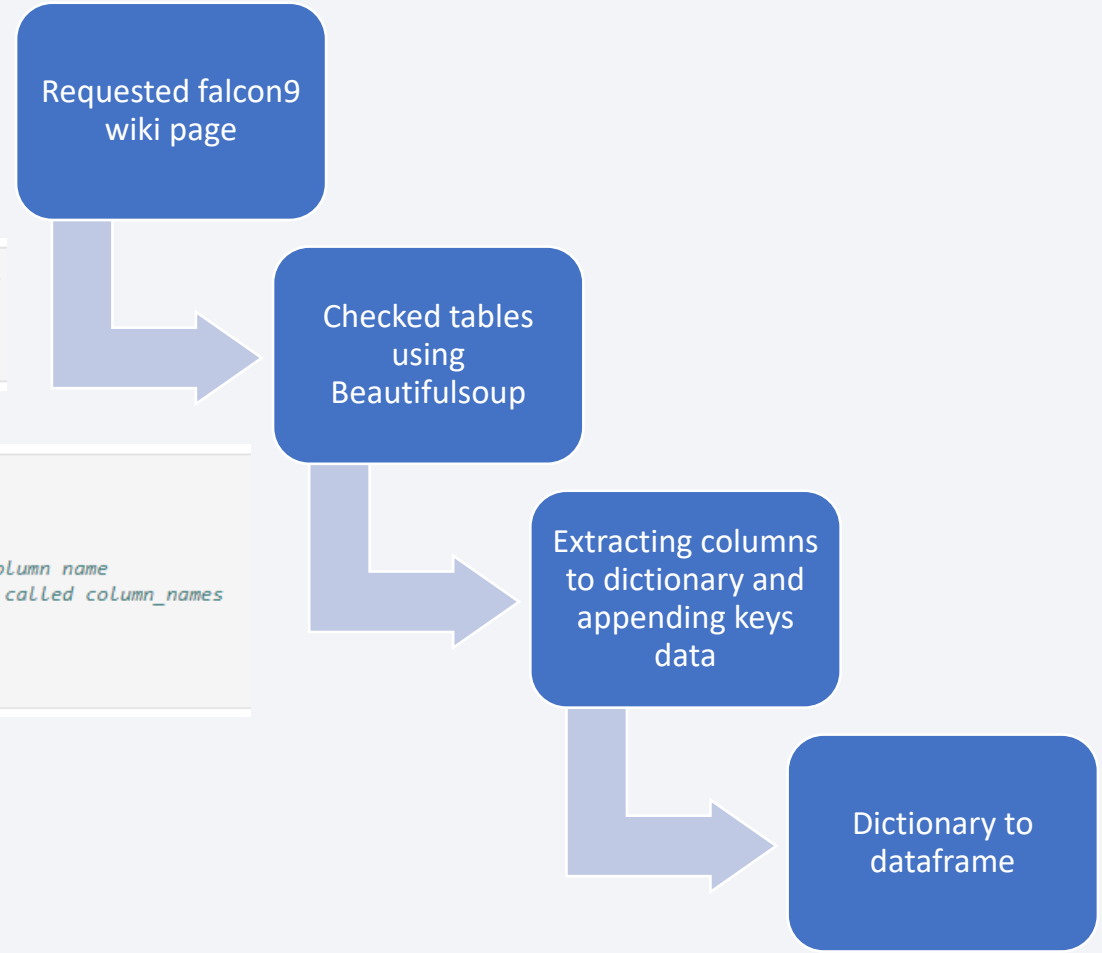
Data Collection - Scraping

```
# use requests.get() method with the provided static_url
# assign the response to a object
wp = page=requests.get(static_url)
```

```
# 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')
```

```
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
for i in first_launch_table.find_all('th'):
    if extract_column_from_header(i)!= None and len(extract_column_from_header(i)) > 0:
            column_names.append(extract_column_from_header(i))
```

```
df=pd.DataFrame(launch_dict)
```



Click here for Github URL

Data Wrangling

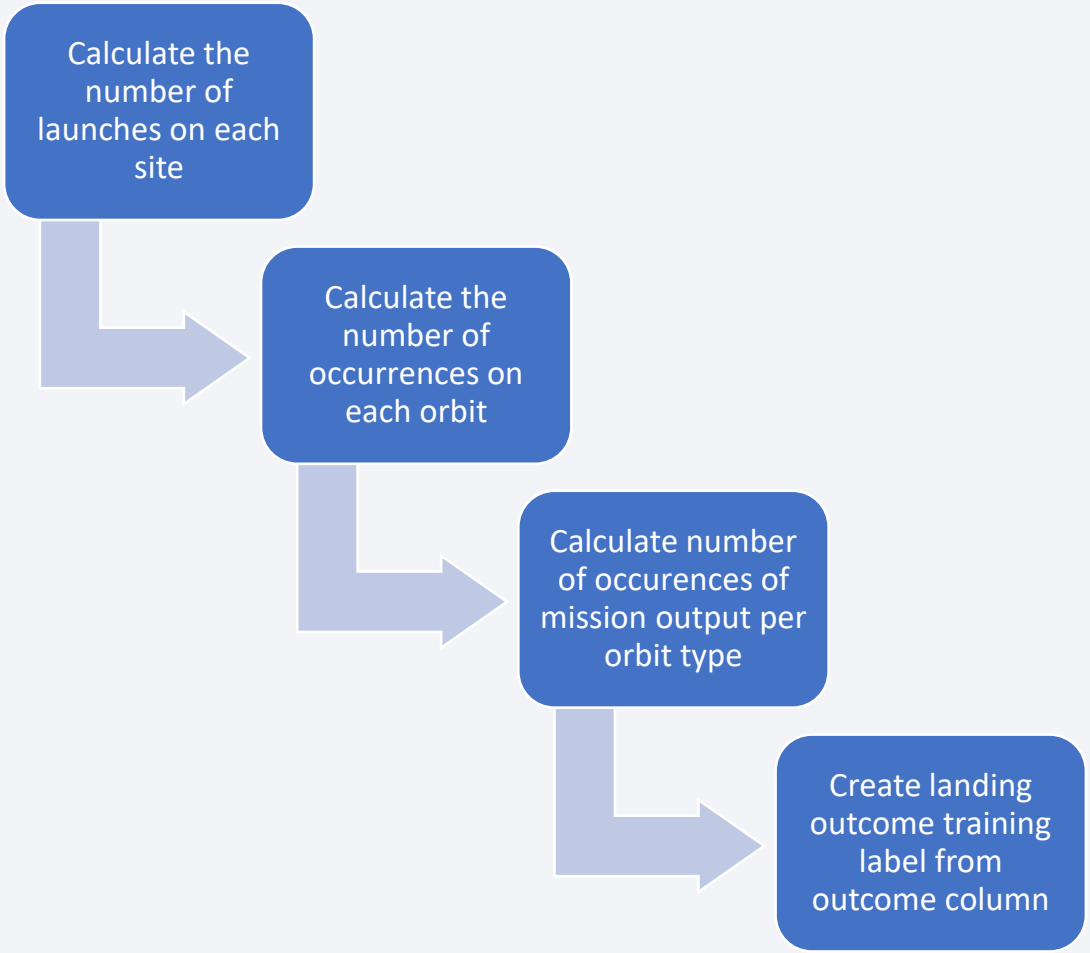
```
# Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

```
# Apply value_counts on Orbit column
df["Orbit"].value_counts()
```

```
# landing_outcomes = values on Outcome column
landing_outcomes = df["Outcome"].value_counts()
```

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
def onehot(item):
    if item in bad_outcomes:
        return 0
    else:
        return 1
landing_class = df["Outcome"].apply(onehot)
```

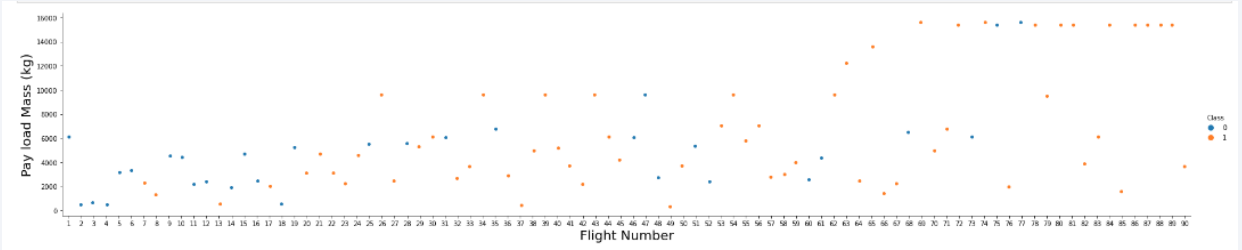
```
df['Class']=landing_class
df[['Class']].head(8)
```

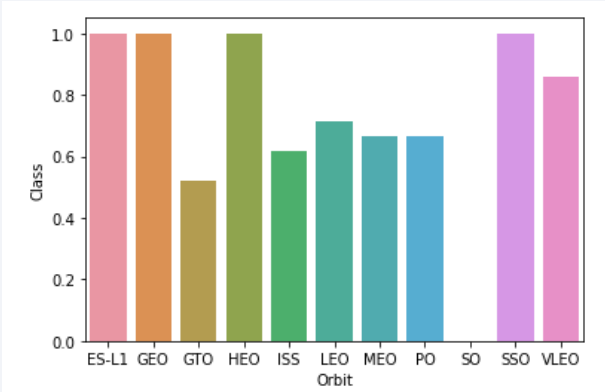


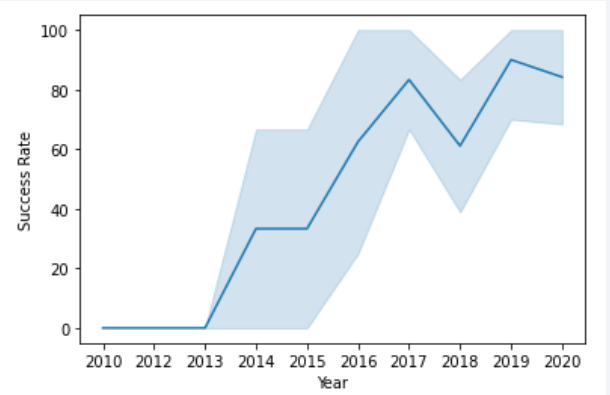
Click here for Github URL

EDA with Data Visualization

- We used various charts to present our findings:
  - Scatter plot to visualize:
    - Flight Number vs Payload Mass to outcome of the launch.
    - Relationship between Flight Number and Launch Site.
    - Relationship between Payload and Launch Site.
    - Relationship between Flight Number and Orbit type.
    - Relationship between Payload and Orbit type.
  - Bar chart to visualize relationship between success rate of each orbit
  - Line chart to visualize success rate with year







Click here for Github URL

EDA with SQL

- Summary of the SQL queries performed
  - Display the names of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version F9 v1.1
  - List the date when the first successful landing outcome in ground pad was achieved.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - List the total number of successful and failure mission outcomes.
  - List the names of the booster versions which have carried the maximum payload mass. Use a subquery.
  - 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.
  - Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

Click here for Github URL

Build an Interactive Map with Folium

- Using the launch site coordinates we highlighted launch sites with a Circle Marker to get objects within its proximity.
- Successful and failed launches were marked with Cluster Marker.
- Distances between launch sites and nearest coastlines/railways were measured with Poly line.

Click here for Github URL

Build a Dashboard with Plotly Dash

- We built the following plots/graphs:
- Pie Chart to display success rate for a selected launch site.
- Range slider for payload range.
- Scatter chart to display outcome related to payload mass.

- The dashboard allows users select a site from dropdown and seek through a relevant payload range to display data.

Click here for Github URL

Predictive Analysis (Classification)

```
Y = data["Class"].to_numpy()
```

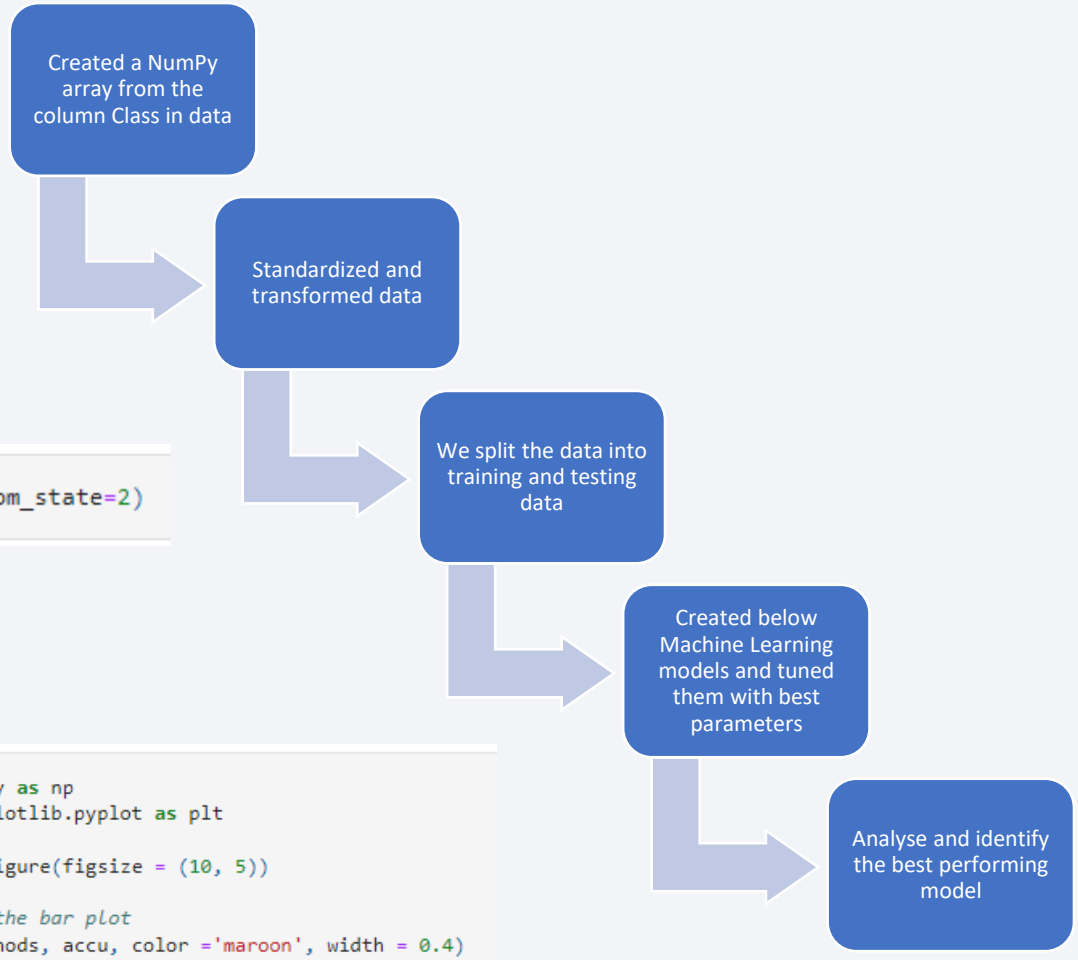
```
# students get this
transform = preprocessing.StandardScaler()
```

```
x_scaled = transform.fit_transform(X)
X = pd.DataFrame(x_scaled)
X.head()
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}
lr=LogisticRegression()
logreg_cv = GridSearchCV(lr, parameters, cv = 10)
logreg_cv.fit(X_train, Y_train)
```

```
accu=[]
methods=[]
accu.append(logreg_cv.score(X_test,Y_test))
methods.append('logistic regression')
logreg_cv.score(X_test,Y_test)
```

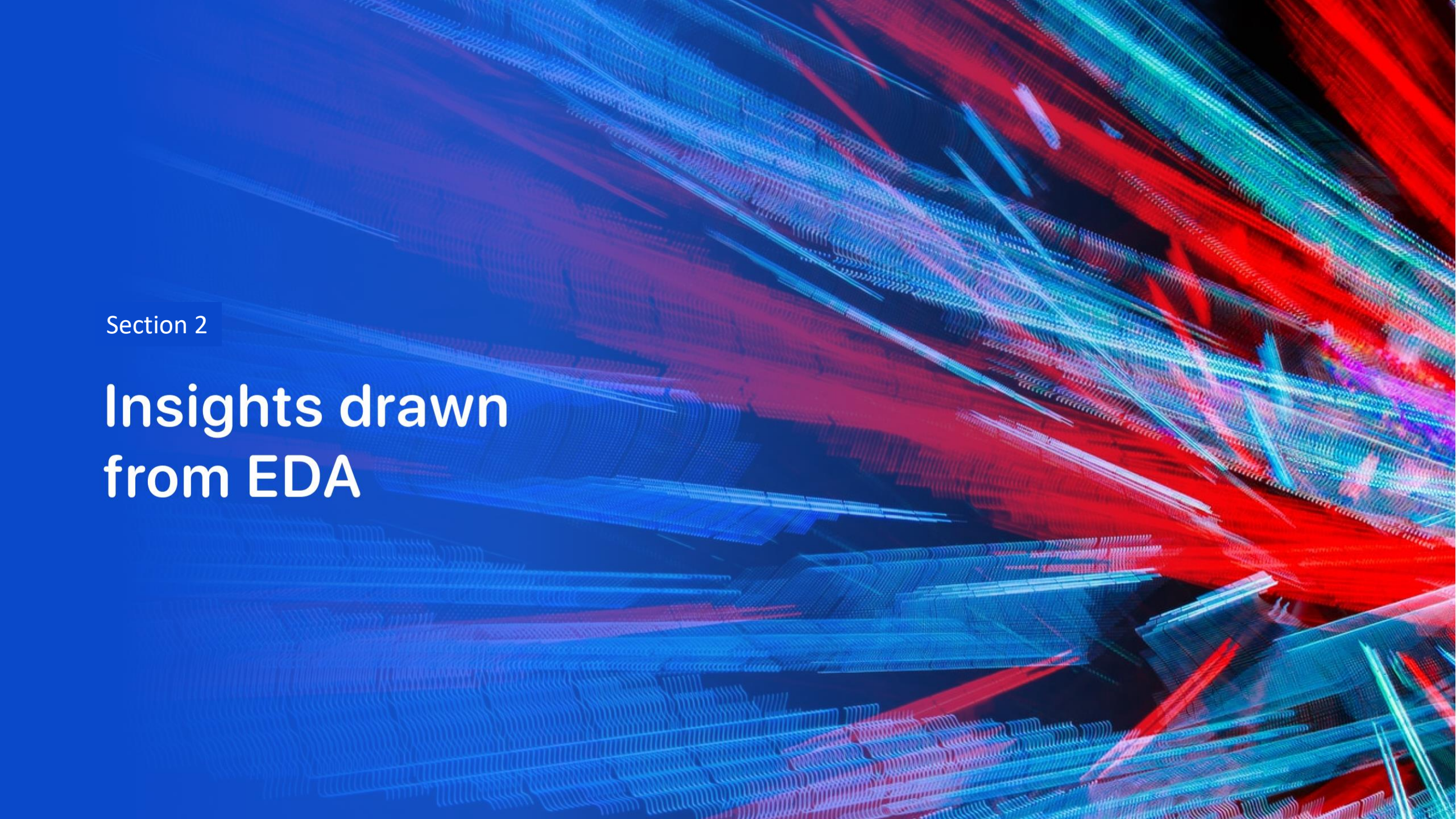



```
import numpy as np
import matplotlib.pyplot as plt

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(methods, accu, color ='maroon', width = 0.4)

plt.xlabel("Methods")
plt.ylabel("Accuracy")
plt.title("Best Perfomed Method")
plt.show()
```

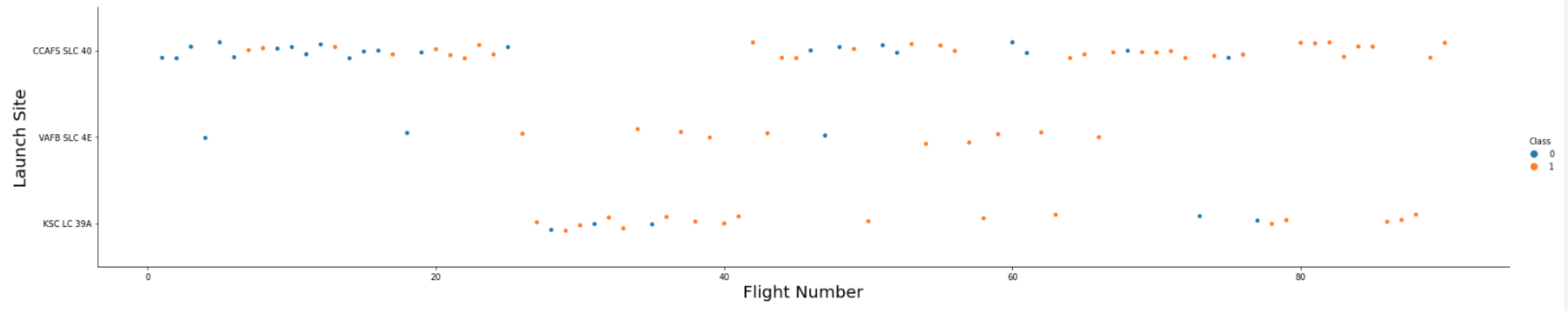

Click here for Github URL

Results

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

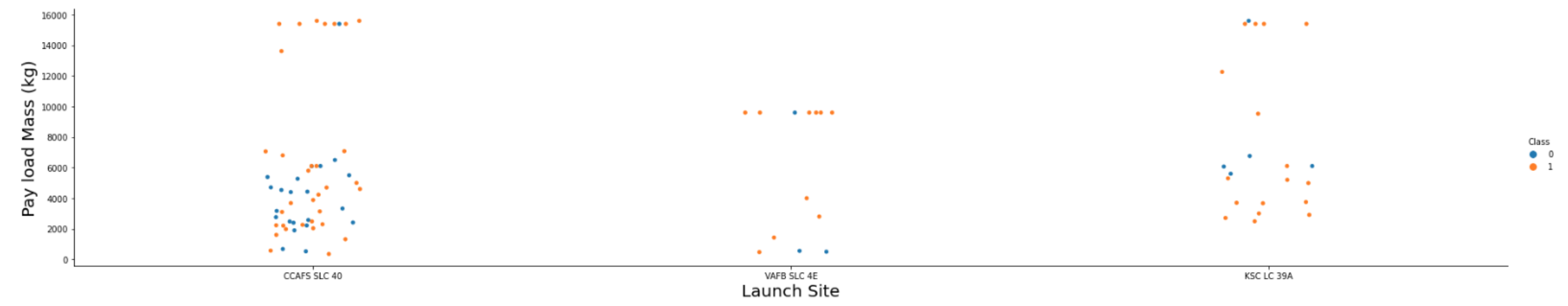

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



- With more flight numbers (continuous launch attempts) the rate of success increased.
- CCAFS SLC 40 remained the primary test site in early stages where most failures took place.
- KSC LC 39A appears to have taken over after initial failures at CCAFS site and shows great success over this period.
- Over the final stages CCAFS SLC 40 site appears to have most successful launches indicating the amount of progress made on this site.

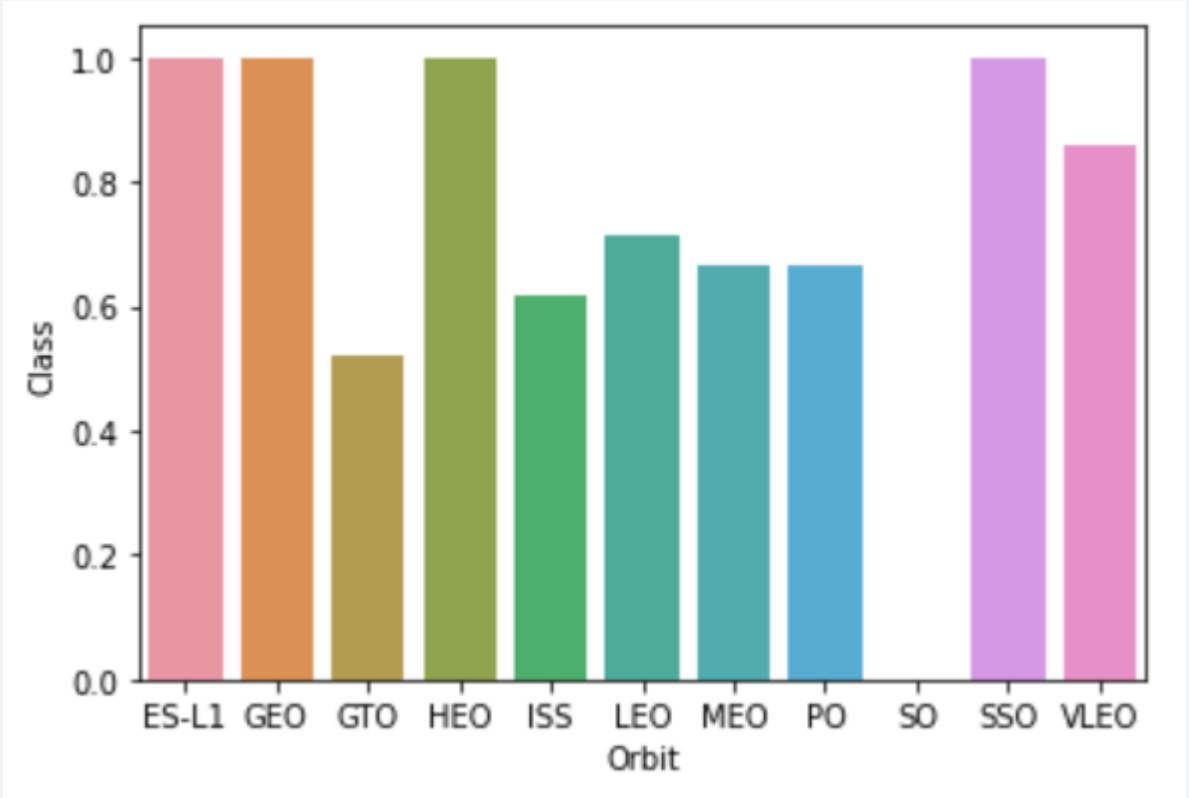
Payload vs. Launch Site



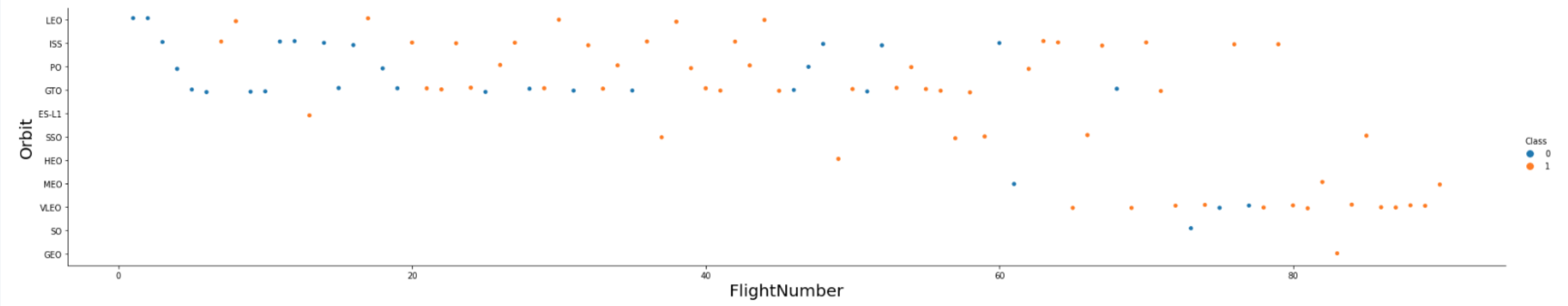
- CCAFS SLC-40 is compatible for both light and heavy payloads and has most launches.
- VAFB SLC 4E and KSC LC 39A has better success rate.
- VAFB SLC 4E supports light payloads as well as payloads of 10,000 KG. These might be allocated for specific purpose.
- Heavier payloads seems to have most success rate for launches however there is no clarity about when was this achieved in the progress curve.

Success Rate vs. Orbit Type

- Orbit type ES-L1, GEO, HEO, SSO has 100% success rate.
- Orbit type SO have had no success.

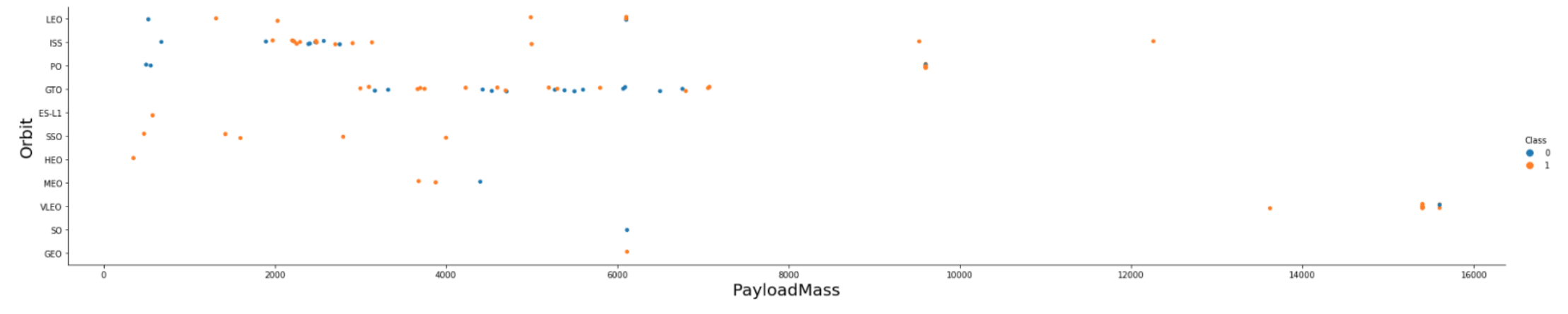


Flight Number vs. Orbit Type



- For LEO orbit success appears related to the number of flights(continuous launch attempts)
- There seems to be no relationship between flight number when in GTO orbit.

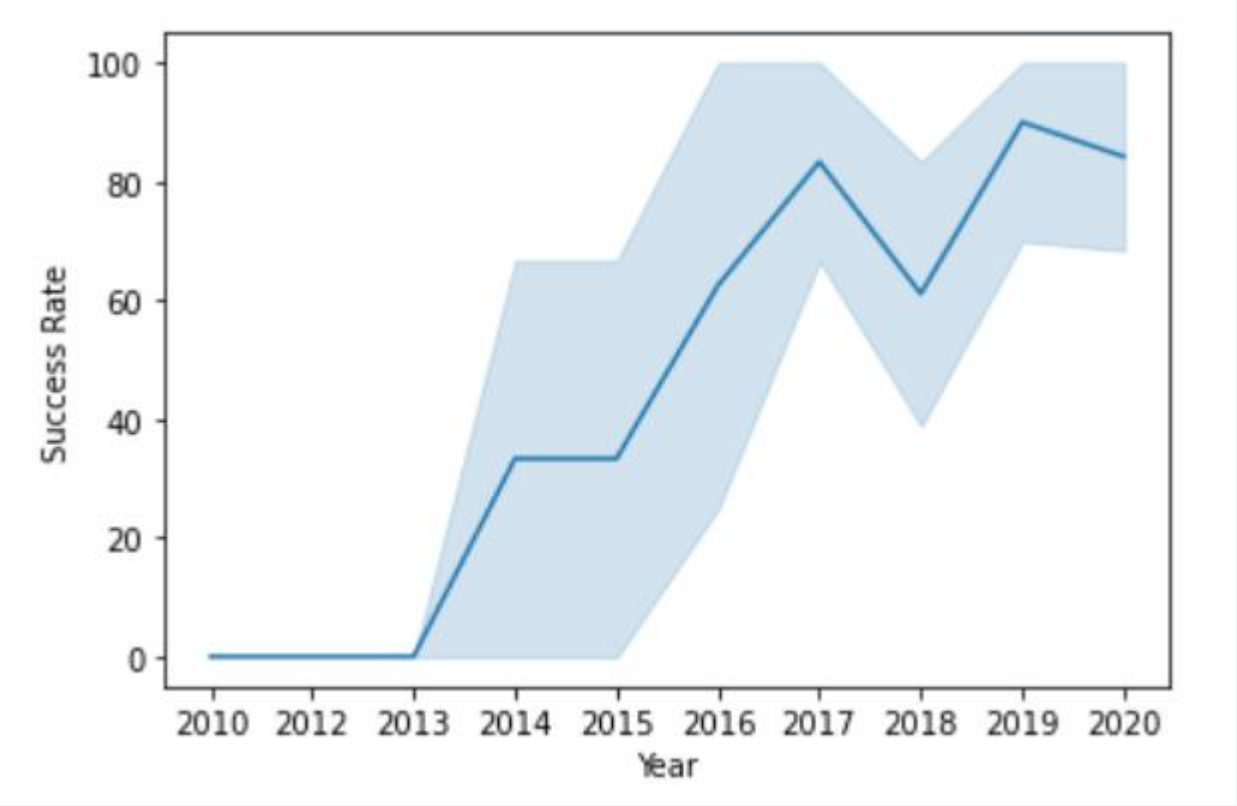
Payload vs. Orbit Type



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

Launch Success Yearly Trend

- The average success rate has been trending upwards since 2013 with slight dip in 2018.



All Launch Site Names

- Names of the unique launch sites

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

- We use the DISTINCT SQL keyword to obtain unique examples.

```
%sql select DISTINCT LAUNCH_SITE from SPACEXTBL
```

Launch Site Names Begin with 'CCA'

- 5 records where launch sites begin with `CCA`

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

- We filter results using where clause and set filter parameter to 'CCA%'. Limit 5 produces only 5 results.

```
%sql select * from SPACEXTBL where launch_site like 'CCA%' limit 5
```

Total Payload Mass

- Total payload mass carried by boosters from NASA

SUM
45596

- We use the sum function on ‘payload_mass__kg_’ column along with where clause on customer column to find all occurrence of ‘NASA (CRS)’.

```
%sql select sum(payload_mass__kg_) as sum from SPACEXTBL where customer like 'NASA (CRS)'
```

Average Payload Mass by F9 v1.1

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

average
2534

- We use the avg function on ‘payload_mass__kg_’ column along with where clause on booster_version column to find all occurrence of ‘F9 v1.1%’.

```
%sql select avg(payload_mass__kg_) as Average from SPACEXTBL where booster_version like 'F9 v1.1%'
```

First Successful Ground Landing Date

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

First Successful landing
2015-12-22

- We use the min function on date column along with where clause on Landing__outcome column to find first occurrence of ‘Success (ground pad)’.

```
%sql select min(date) as "First Successful landing" from SPACEXTBL where Landing__Outcome like 'Success (ground pad)'
```

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

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

- We use where clause on columns mission_outcome, payload_mass_kg_ and landing__outcome to find occurrences ‘Success’, between ‘4000 and 6000’ and ‘Success (drone ship)’.

```
%%sql select booster_version from SPACEXTBL
where (mission_outcome like 'Success') AND (payload_mass__kg_ BETWEEN 4000 AND 6000) AND (landing__outcome like 'Success (drone ship)')
```

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

mission_outcome	COUNT
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- We used group by to group missions_outcomes based on count.

```
%sql SELECT mission_outcome, count(*) as Count FROM SPACEXTBL GROUP by mission_outcome
```

Boosters Carried Maximum Payload

- List of booster_versions which have carried the maximum payload mass
- We wrote a subquery to determine the max payload using the function max. Used where clause on column payload_mass__kg_ with subquery as parameter.

```
%%sql select booster_version from SPACEXTBL
where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL)
```

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

MONTH	landing__outcome	booster_version	launch_site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- We used monthname function to extract month. Using where clause we queried Date and landing__outcome to '2015%' and 'Failure (drone ship)'.

```
%%sql select MONTHNAME(DATE) as Month, landing__outcome, booster_version, launch_site
from SPACEXTBL where DATE like '2015%' AND landing__outcome like 'Failure (drone ship)'
```

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

- Rank of count of landing outcomes between the date 2010-06-04 and 2017-03-20, in descending order

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

- We filtered date using where clause and grouped landing__outcome based on count. The count is arranged in descending order.

```
%%sql select landing__outcome, count(*) as count from SPACEXTBL
where Date >= '2010-06-04' AND Date <= '2017-03-20'
GROUP by landing__outcome ORDER BY count Desc
```

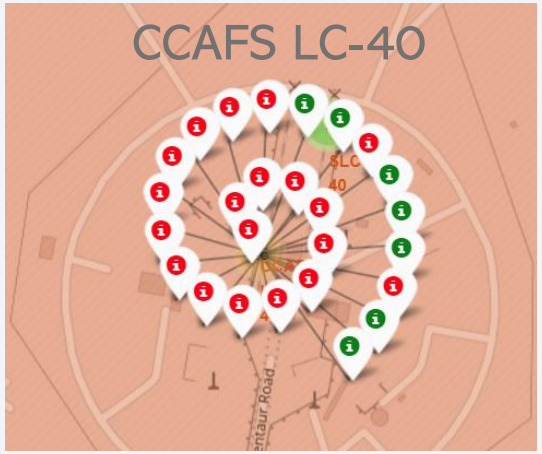
Section 3

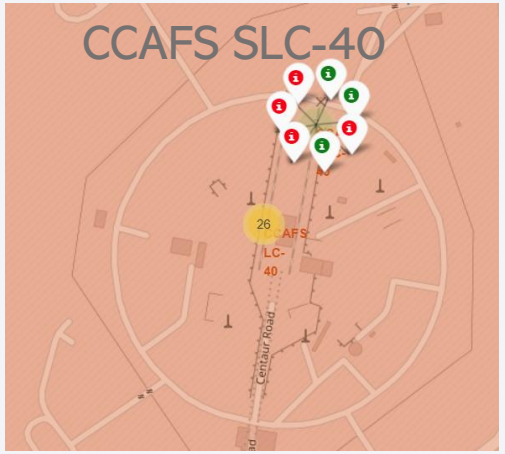
Launch Sites Proximities Analysis

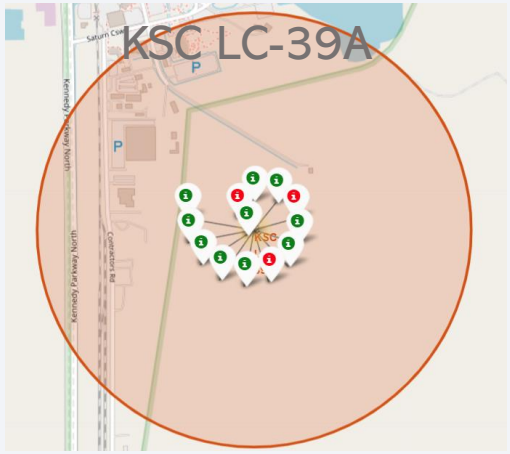
Launch Sites

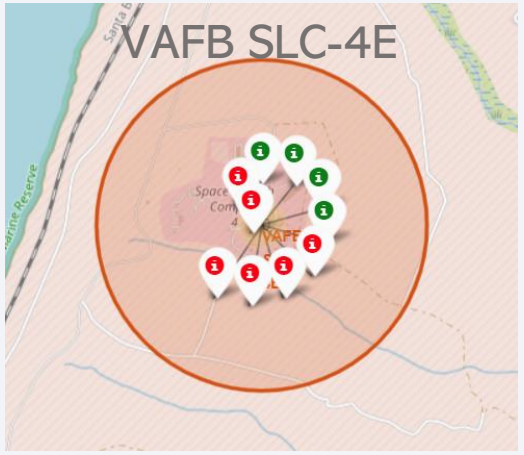
Color-labeled launch outcomes

- Green markers show successful launches.
- Red markers show failed launches.









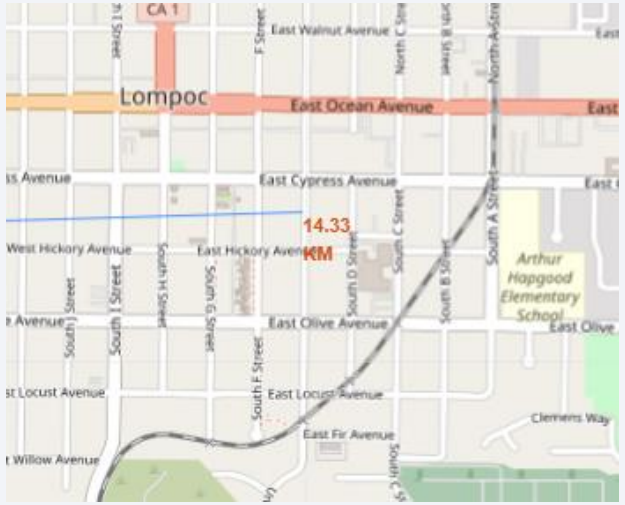
Launch site proximities

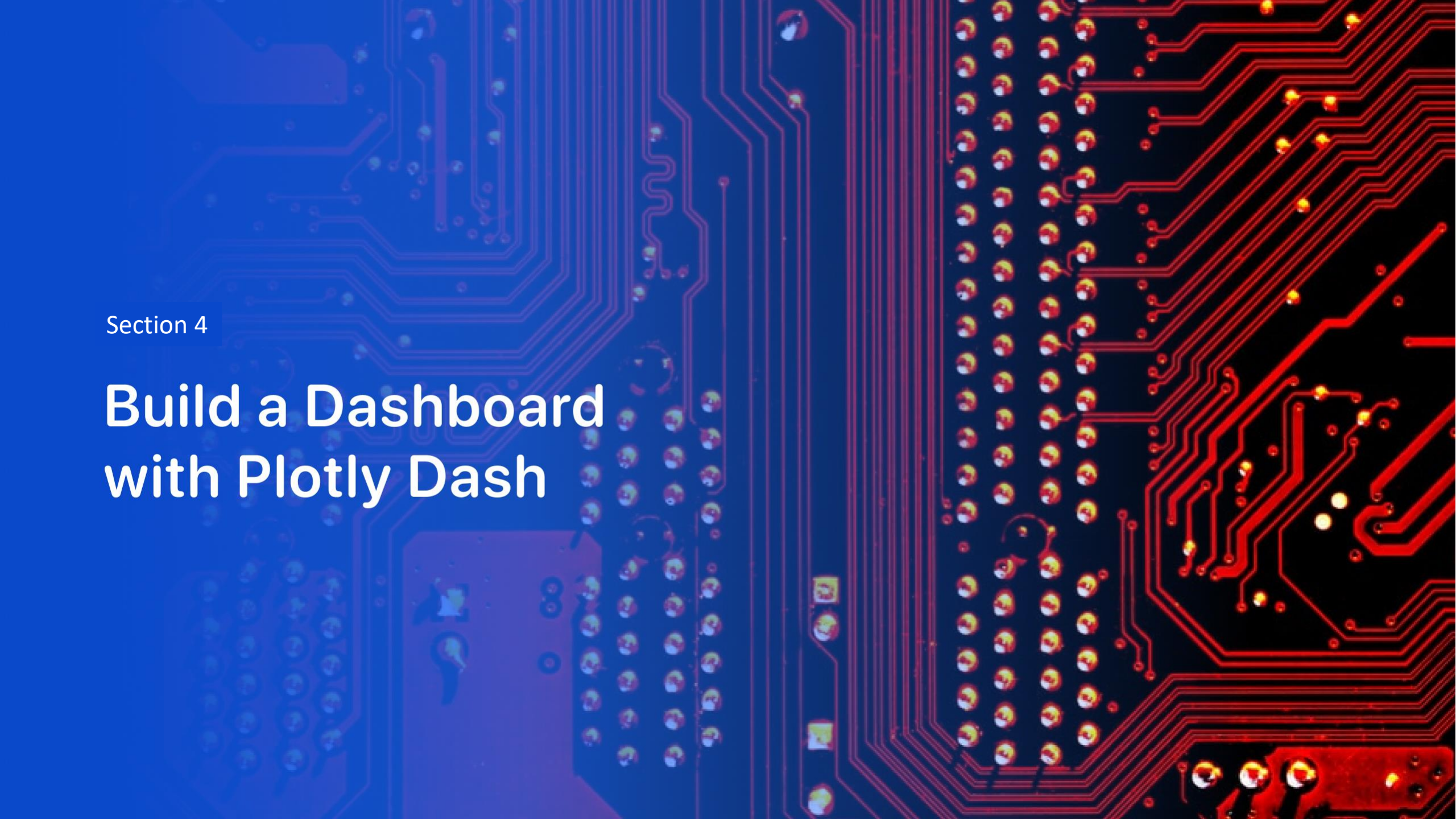
- Launch site to railway

- Launch site to coastline

- Launch site to highway

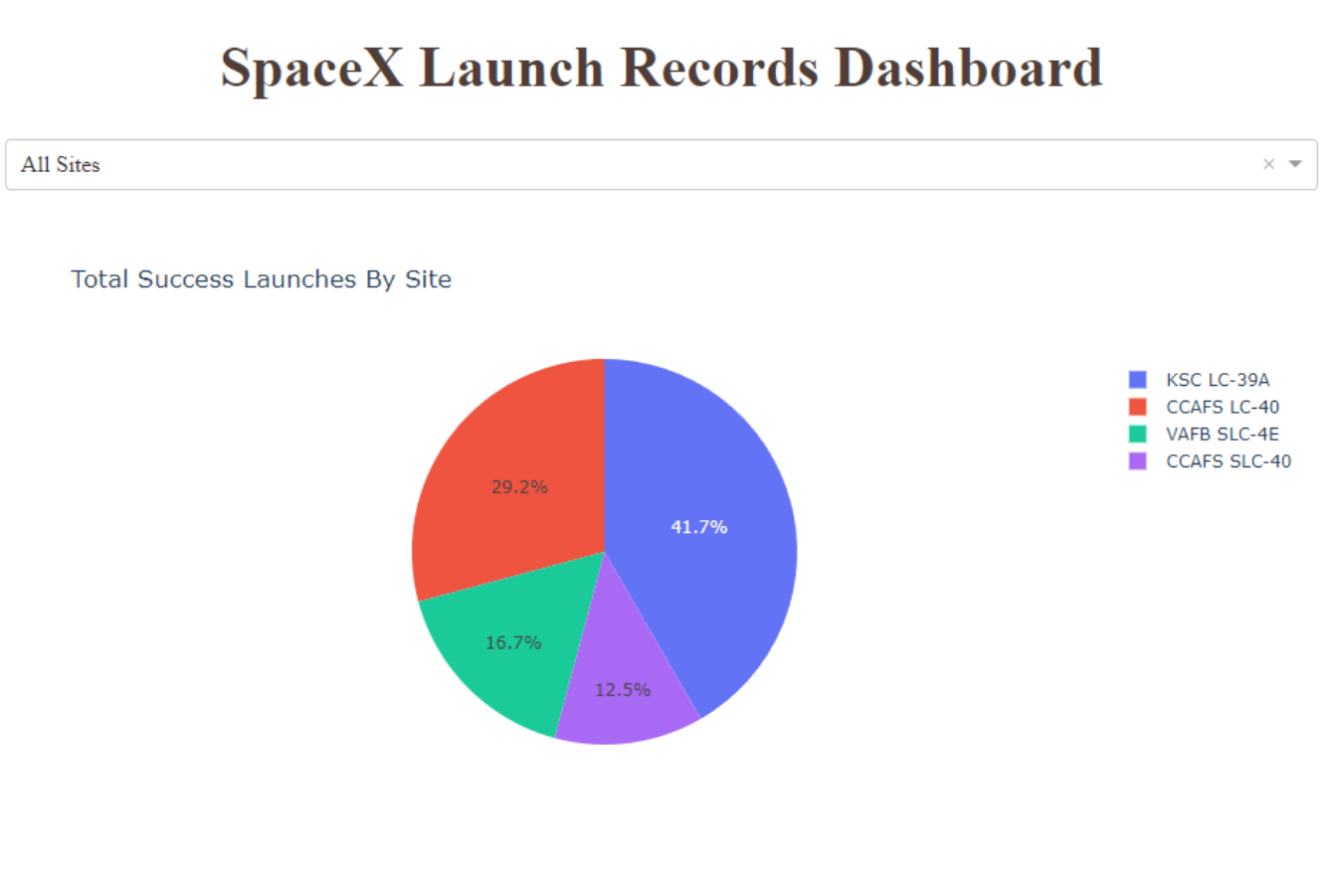
- Launch site to city




Section 4
Build a Dashboard with Plotly Dash

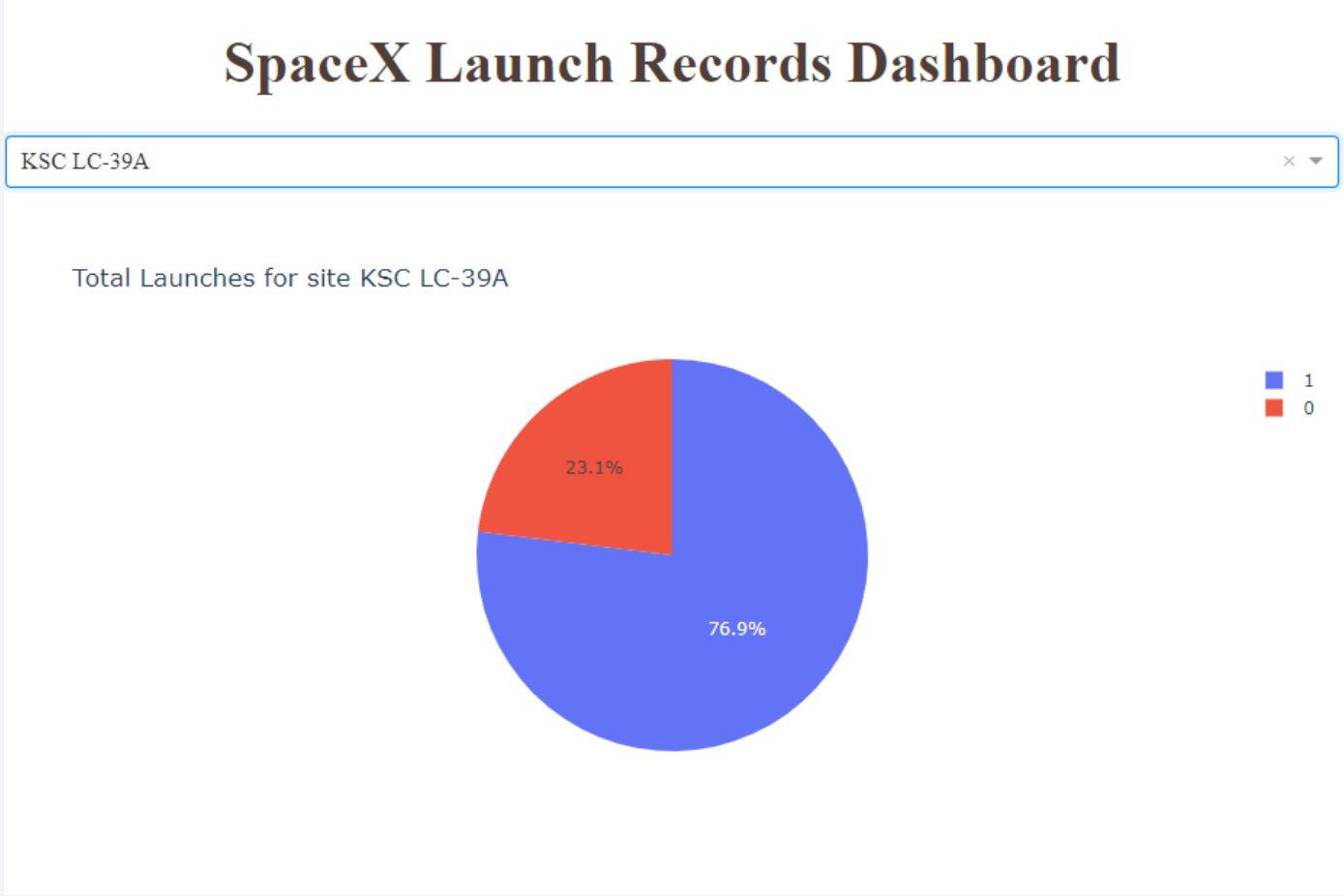
Success Percentage by site

- KSC LC-39A achieved more successful launches followed by CCAFS LC-40



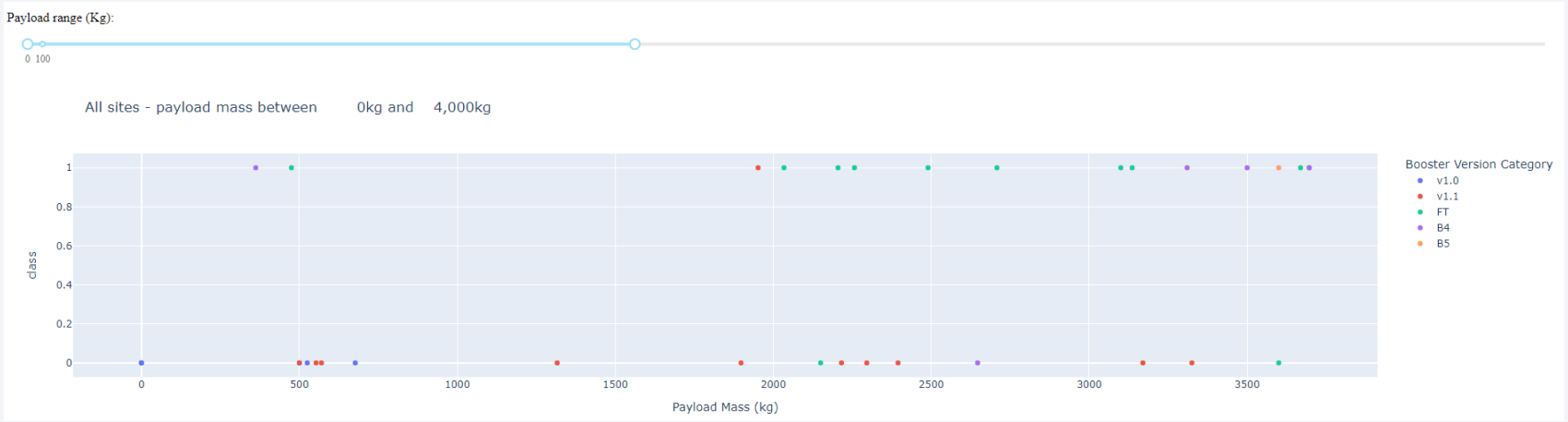
Launch Site with highest success rate

- KSC LC-39A achieved a 76.9 percent success rate.

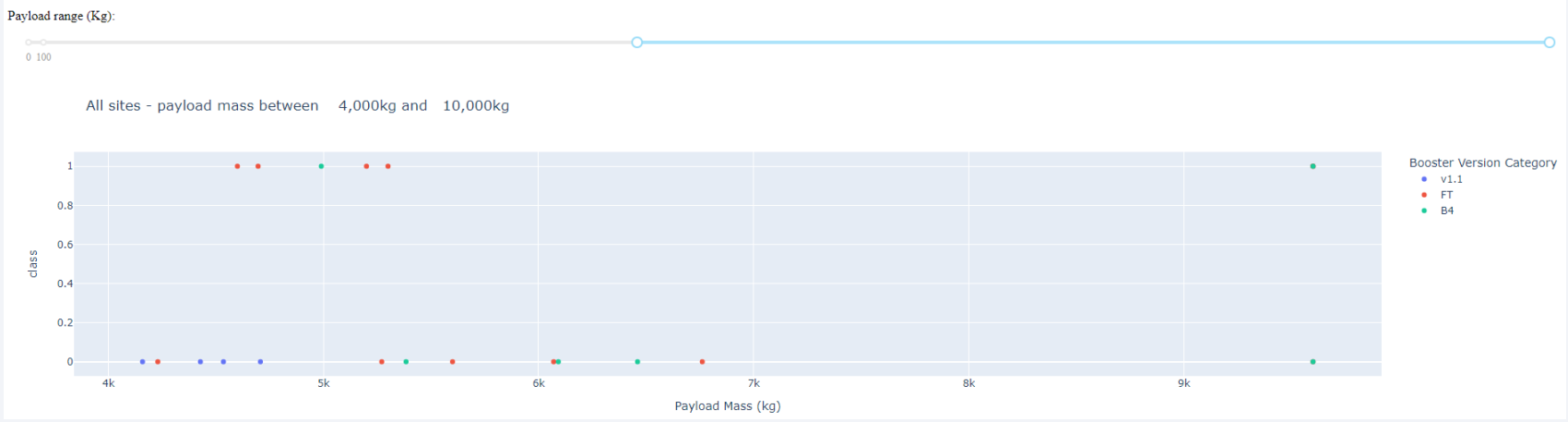


<Dashboard Screenshot 3>

- Payload between 0 to 4,000 KG has higher success rate



- Payload between 4,000 to 10,000 KG has lower success rate

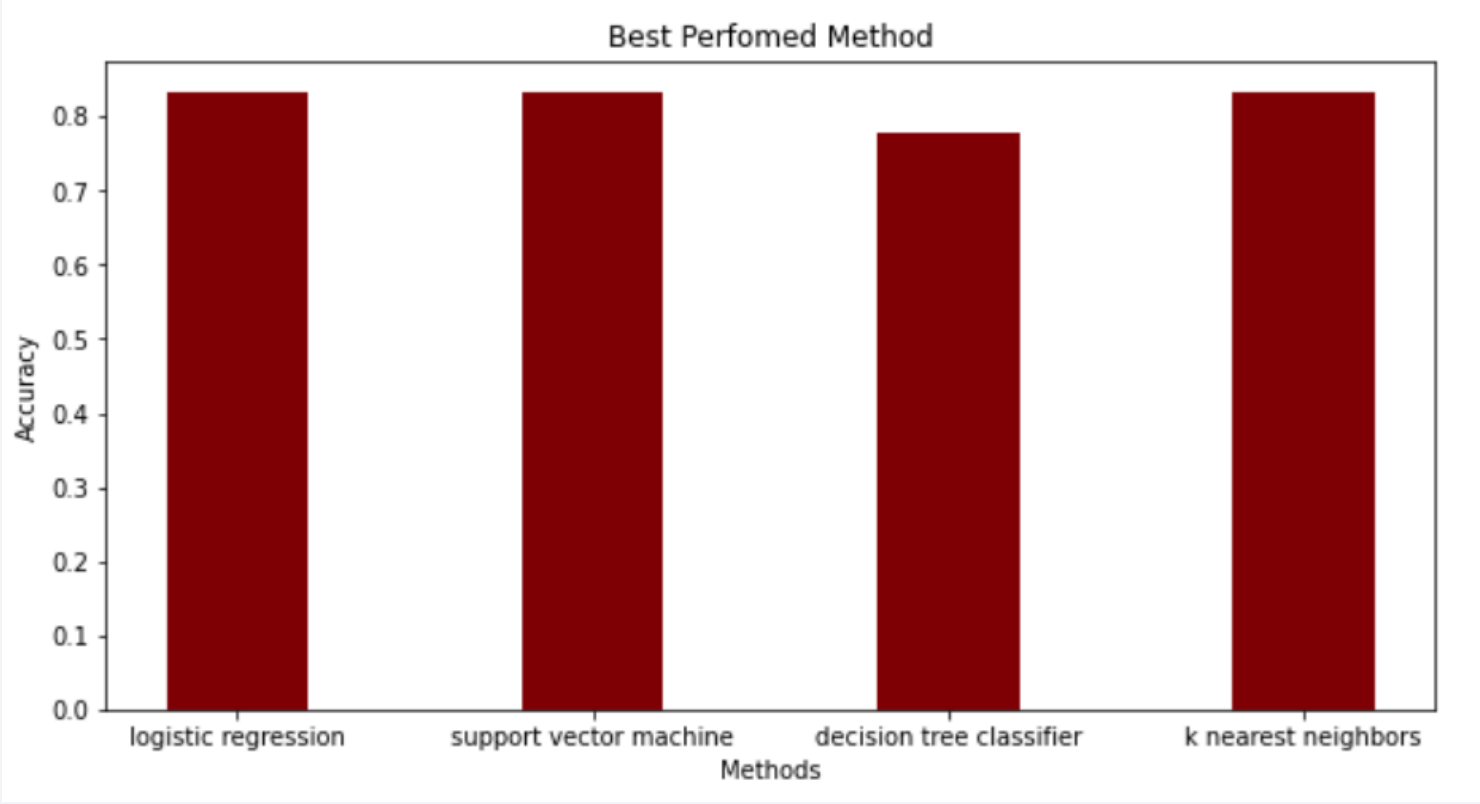


Section 5

Predictive Analysis (Classification)

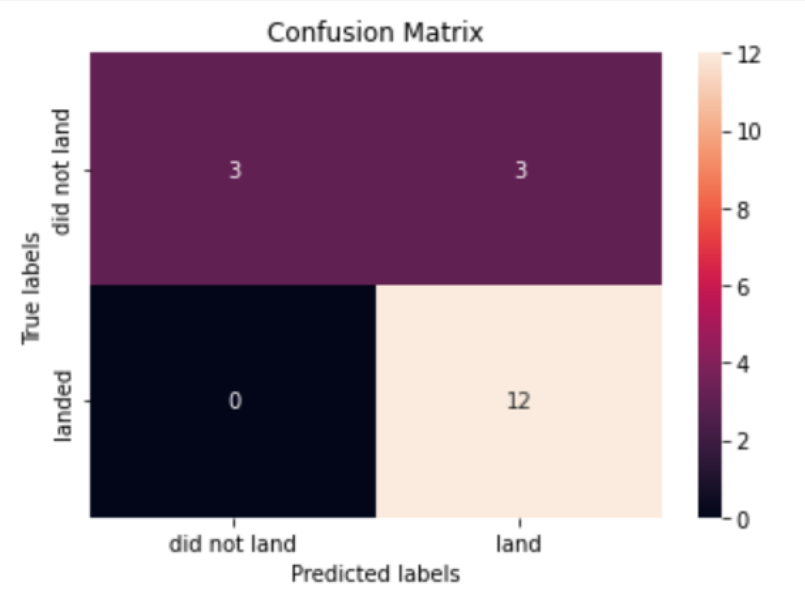
Classification Accuracy

- All models except decision tree classifier performed with an accuracy of 83%.
- Decision tree scored performance of 77% approx



Confusion Matrix

- The confusion matrix for all three best performing models are the same.
- Some of the failed landings were predicted to be successful in the test set.



Conclusions

- Success rate of SpaceX increased with flight numbers.
- Though KSC LC-39A had more successful launches with optimum payloads, CCAFS SLC-40 gradually increased its success rate for both light and heavy payloads.
- Orbit type ES-L1, GEO, HEO, SSO had 100% success rates.
- All models except decision tree performed with an accuracy of 83%.

Appendix

- Notebook References:
  - Data Collection API
  - Data Collection with Web Scraping
  - Data Wrangling
  - EDA with SQL
  - EDA with Visualization
  - Interactive Visual analytics with folium
  - Machine Learning Prediction
  - Plotly Dash App

Thank you!