IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Pranay Moluguri
July 19th 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection - API
  - Data Collection – Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis
    - SQL
    - Data Visualization
  - Interactive Learning Predictions with Folium
  - Predictions using Machine Learning
- Summary of all results
  - Exploratory Data Analysis Result
  - Interactive Analytics Screenshots
  - Predictive Analytics Results

Introduction

- **Project background and context**

SpaceX promotes Falcon 9 rocket launches on its website at a competitive cost of 62 million dollars, while other providers charge upwards of 165 million dollars per launch. This significant price difference is largely due to SpaceX's ability to reuse the first stage of the rocket. Therefore, accurately predicting whether the first stage will land successfully becomes crucial in determining the overall cost of a launch. This predictive information holds valuable potential for other companies interested in bidding against SpaceX for a rocket launch contract. The ultimate aim of this project is to develop a machine learning pipeline that can efficiently forecast the likelihood of a successful first stage landing.

- **Problems you want to find answers**

1. What are the factors that determine if a rocket will land successfully?
2. How do different features interact to influence the success rate of a rocket's landing?
3. What operating conditions must be in place to ensure a successful landing program for rockets?
4. How important is the proper management of operating conditions for the successful return of a rocket's first stage?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data has been collected from
    - SpaceX API
    - Web scraping from Wikipedia
- Perform data wrangling
  - One Hot Encoding was applied and other launch vehicles were filtered out.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - LR, KNN, SVM, Decision Tree models have been used.

Data Collection

- Data Collection was done using the SpaceX API and Wikipedia Web Scrapping.
- The Data was converted as a JSON using a .json function and was used to read into a Pandas Data Frame.
- The unnecessary data was filtered out.
- Missing values were found and filled using methods like mean etc.
- Web Scraping from Wikipedia was done using Beautiful Soup from the records of the HTML table.

Data Collection – SpaceX API

- We utilized the GET request to access the SpaceX API for data collection. Afterward, we performed data cleaning, basic data wrangling, and formatting on the retrieved data.

- https://github.com/pranaymoluguri/DataScience_Capstone/blob/42bee169f9e1f177417f68601738a8526f0e2f2f/1.Data%20Collection%20API%20SpaceX.ipynb

```
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
       if x:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])
```

```
# Takes the dataset and uses the launchpad column to call the API and append the data to the list
def getLaunchSite(data):
    for x in data['launchpad']:
       if x:
         response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
         Longitude.append(response['longitude'])
         Latitude.append(response['latitude'])
         LaunchSite.append(response['name'])
```

```
# 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'])
```

Data Collection - Scraping

- Web Scraping from Wikipedia page.
- Beautiful Soup has been used, converted to Panda DataFrame.

- https://github.com/pranaymoluguri/DataScience_Capstone/blob/6db5ba040372eb1c6e449b7670e7bd7c15919aad/2.%20WebScraping%20SpaceX.ipynb

```
def date_time(table_cells):
    return [data_time.strip() for data_time in list(table_cells.strings)][0:2]

def booster_version(table_cells):
    out=''.join([booster_version for i,booster_version in enumerate( table_cells.strings) if i%2==0][0:-1])
    return out

def landing_status(table_cells):
    out=[i for i in table_cells.strings][0]
    return out

def get_mass(table_cells):
    mass=unicodedata.normalize("NFKD", table_cells.text).strip()
    if mass:
        mass.find("kg")
        new_mass=mass[0:mass.find("kg")+2]
    else:
        new_mass=0
    return new_mass
```

```
html_data = requests.get(static_url)
html_data.status_code
```

```
# Use BeautifulSoup() to create a BeautifulSoup object
soup = BeautifulSoup(html_data.text, 'html.parser')
```

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

Data Wrangling

- The Data was copied into a Pandas Data Frame and Analysed.
- Null Values were found, replaced using methods like mean etc.

- https://github.com/pranaymoluguri/DataScience_Capstone/blob/6db5ba040372eb1c6e449b7670e7bd7c15919aad/3.%20Data%20Wrangling.ipynb

```
from js import fetch
import io

URL = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_1.csv'
resp = await fetch(URL)
dataset_part_1_csv = io.BytesIO((await resp.arrayBuffer()).to_py())
```

```
df=pd.read_csv(dataset_part_1_csv)
df.head(10)
```

```
# Apply value_counts() on column LaunchSite
df.LaunchSite.value_counts()
```

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

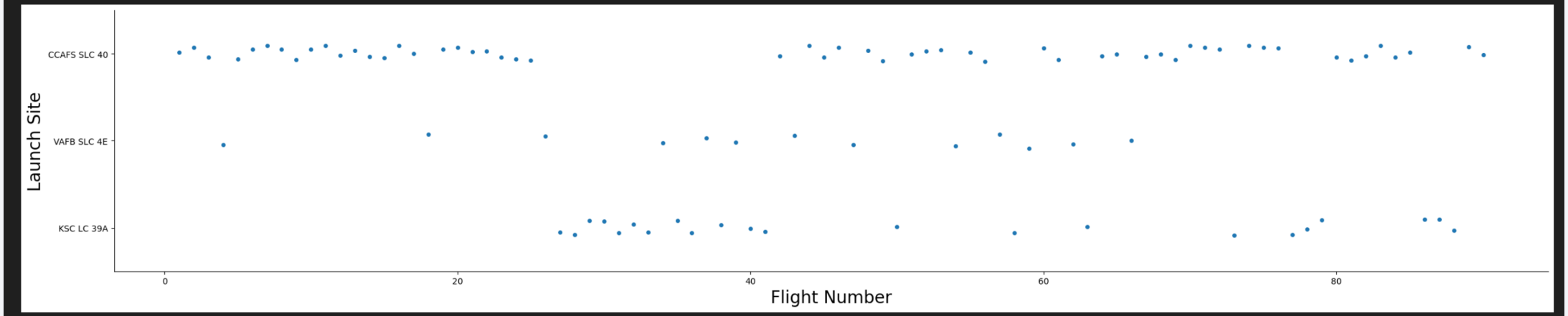
```
df.isnull().sum()/df.shape[0]*100
```

```
FlightNumber      0.000000
Date              0.000000
BoosterVersion    0.000000
PayloadMass       0.000000
Orbit             0.000000
LaunchSite        0.000000
Outcome           0.000000
Flights           0.000000
GridFins          0.000000
Reused            0.000000
Legs              0.000000
LandingPad       28.888889
Block             0.000000
ReusedCount       0.000000
Serial            0.000000
Longitude         0.000000
Latitude          0.000000
dtype: float64
```

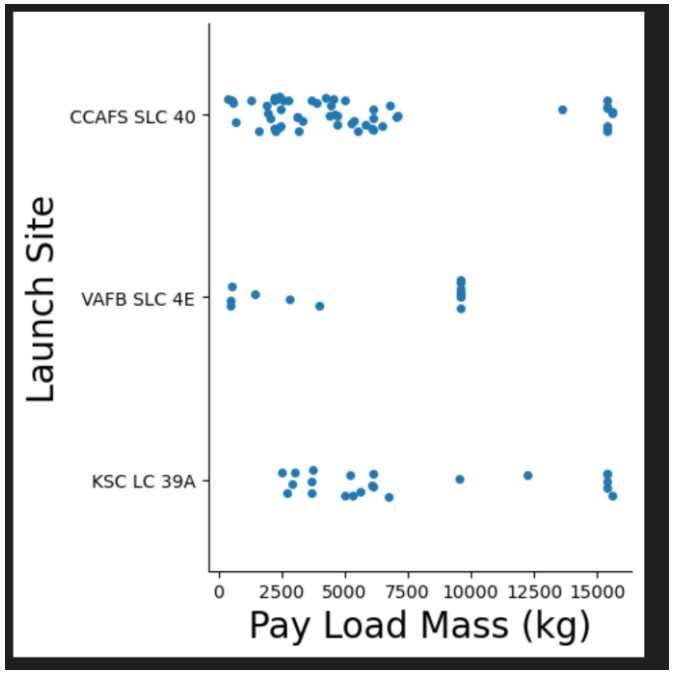
EDA with Data Visualization

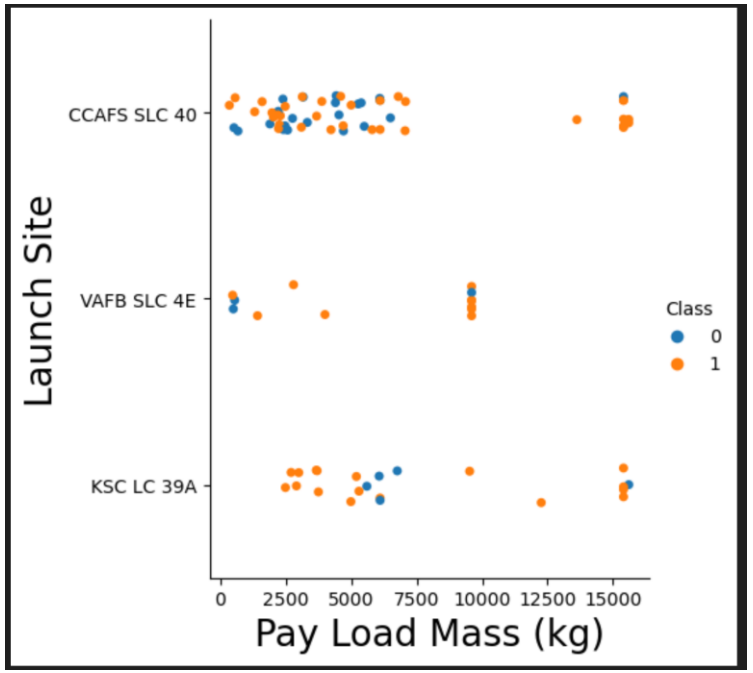
```python
### TASK 1: Visualize the relationship between Flight Number and Launch Site
sns.catplot(y="LaunchSite", x="FlightNumber", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

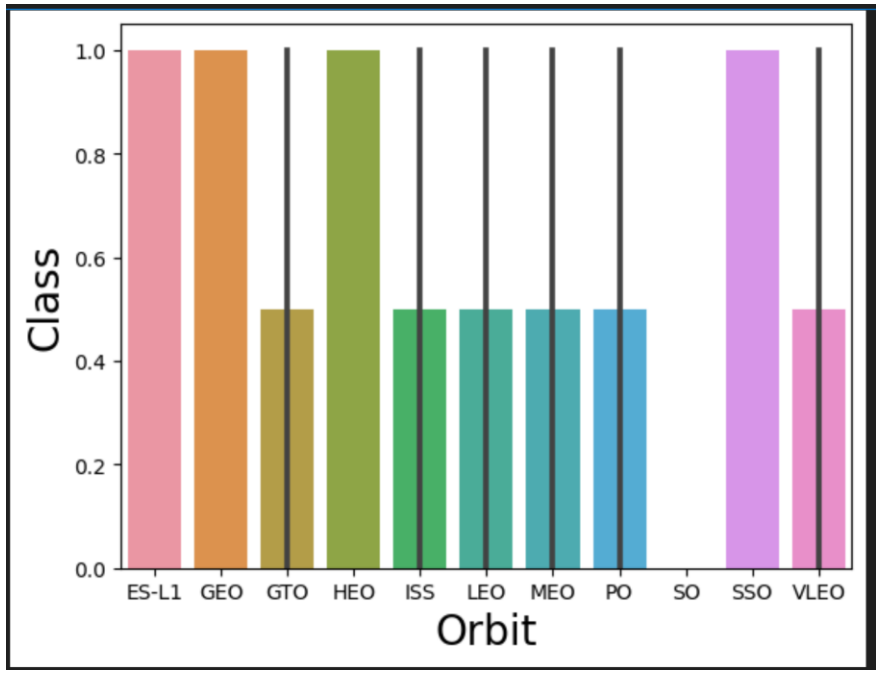
Python

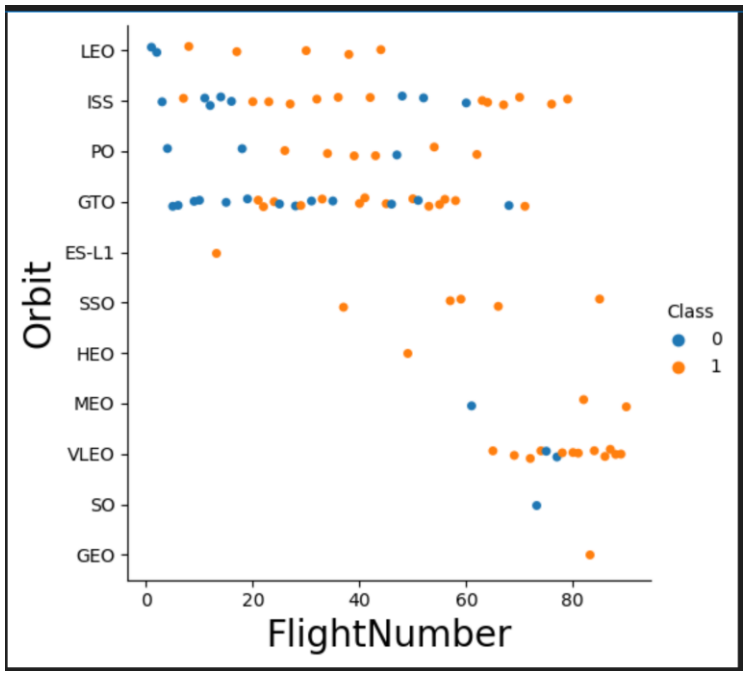


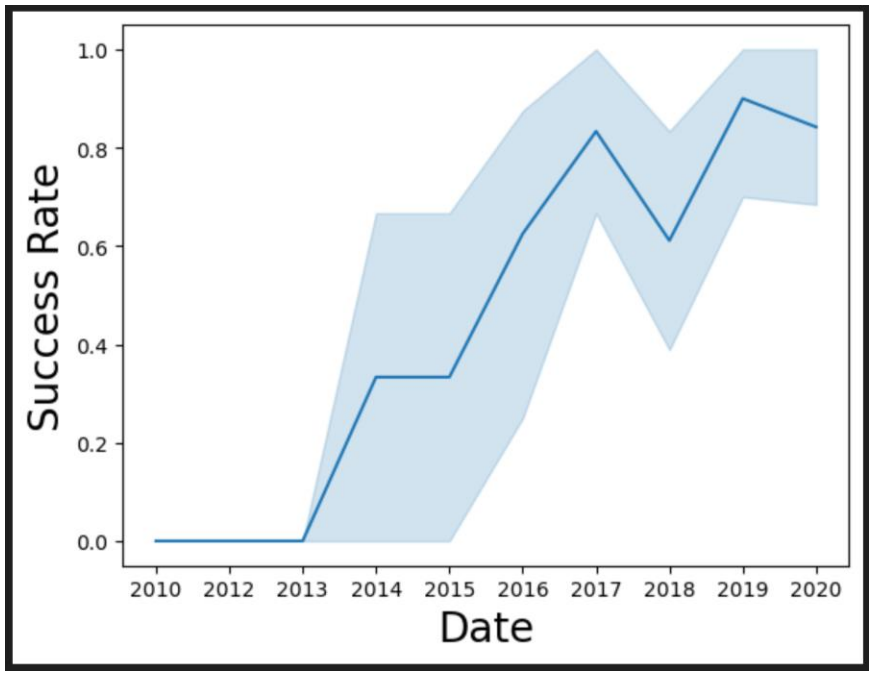
https://github.com/pranaymoluguri/DataScience_Capstone/blob/e9c7da384d409e19e483ccf99fcbd0cf3ddc42db/5.%20EDA%20Data%20Visualization.ipynb

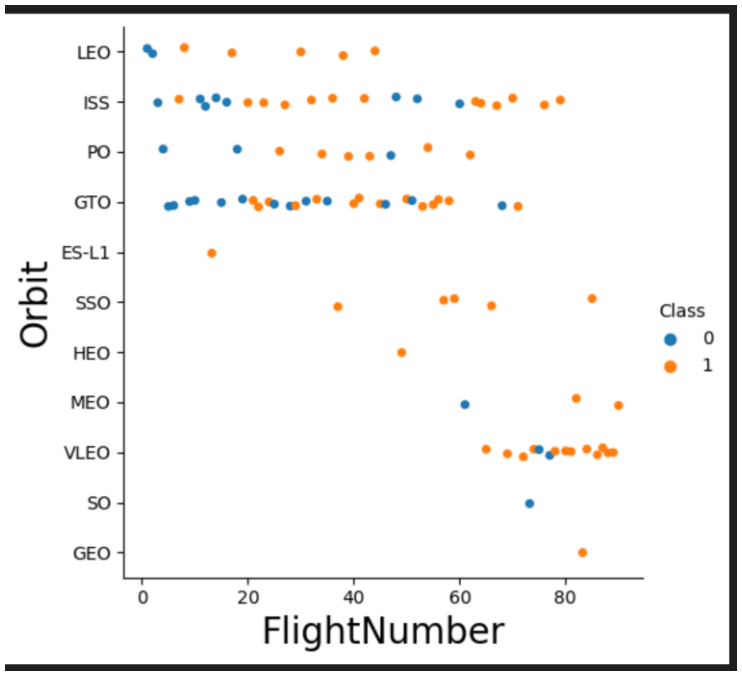

Launch Site
CCAFS SLC 40
VAFB SLC 4E
KSC LC 39A
0
2500
5000
7500
10000
12500
15000
Pay Load Mass (kg)


Launch Site
CCAFS SLC 40
VAFB SLC 4E
KSC LC 39A
Class
0
1
0
2500
5000
7500
10000
12500
15000
Pay Load Mass (kg)


Class
1.0
0.8
0.6
0.4
0.2
0.0
ES-L1
GEO
GTO
HEO
ISS
LEO
MEO
PO
SO
SSO
VLEO
Orbit


Orbit
LEO
ISS
PO
GTO
ES-L1
SSO
HEO
MEO
VLEO
SO
GEO
Class
0
1
0
20
40
60
80
FlightNumber


Success Rate
1.0
0.8
0.6
0.4
0.2
0.0
2010
2012
2013
2014
2015
2016
2017
2018
2019
2020
Date


Orbit
LEO
ISS
PO
GTO
ES-L1
SSO
HEO
MEO
VLEO
SO
GEO
Class
0
1
0
20
40
60
80
FlightNumber

EDA with SQL

```
import pandas as pd
df = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/labs/
df.to_sql("SPACEXTBL", con, if_exists='replace', index=False,method="multi")
```

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

```
%sql SELECT LAUNCH_SITE FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

List the total number of successful and failure mission outcomes

```
%sql SELECT SUM(CASE WHEN MISSION_OUTCOME LIKE '%Success%' THEN 1 ELSE 0 END) AS SUCCESS,
SUM(CASE WHEN MISSION_OUTCOME LIKE '%FAIL%' THEN 1 ELSE 0 END) AS FAIL FROM SPACEXTBL;
```

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

SUCCESS	FAIL
100	1

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

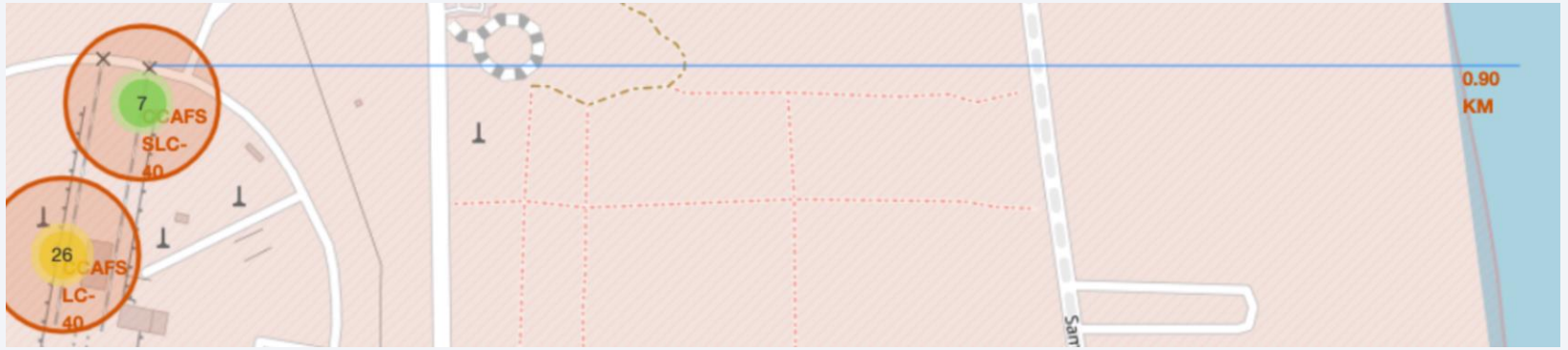
```
%sql select SUM(PAYLOAD_MASS__KG_) AS PAYLOADMASS FROM SPACEXTBL
```

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

PAYLOADMASS
619967.0

https://github.com/pranaymoluguri/DataScience_Capstone/blob/e9c7da384d409e19e483ccf99fcbd0cf3ddc42db/4.%20EDA%20Sql%20SpaceX.ipynb

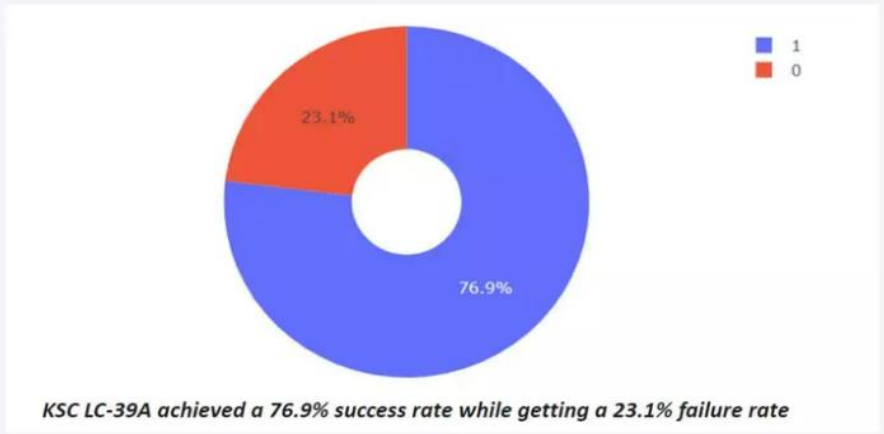
Build an Interactive Map with Folium



https://github.com/pranaymoluguri/DataScience_Capstone/blob/9f7fdcaaaf528f84e75918d64de5d6eff8f45f9b/6.%20Maps%20Launch%20Site%20Location.jupyterlite.ipynb

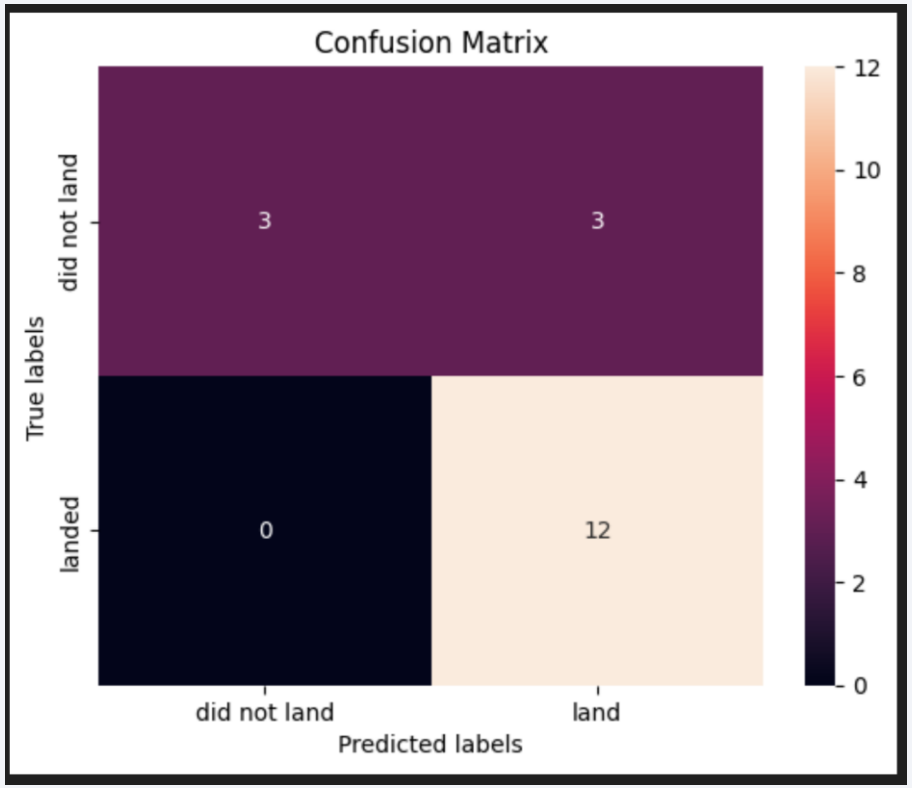
Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.



https://github.com/pranaymoluguri/DataScience_Capstone/blob/30c747f29fddbe9fd573412905ea53065864bde3/7.%20Plotly%20Dashboard.py

Predictive Analysis (Classification)

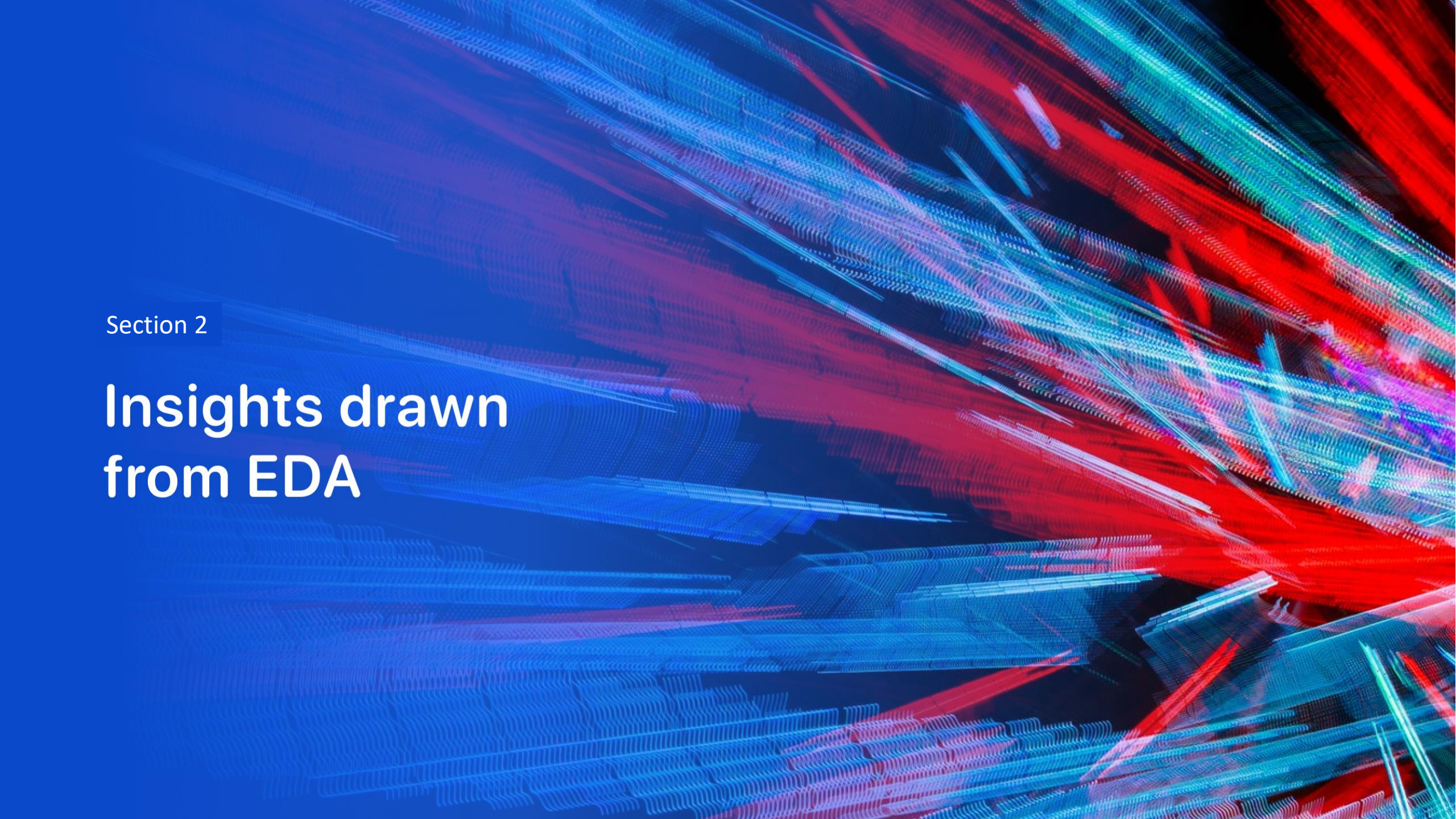


Logistic Regression and KNN, achieved the best accuracy at 83.33%, and Decision Tree at 77.8% while SVM performs the best Area under curve at 0.958.

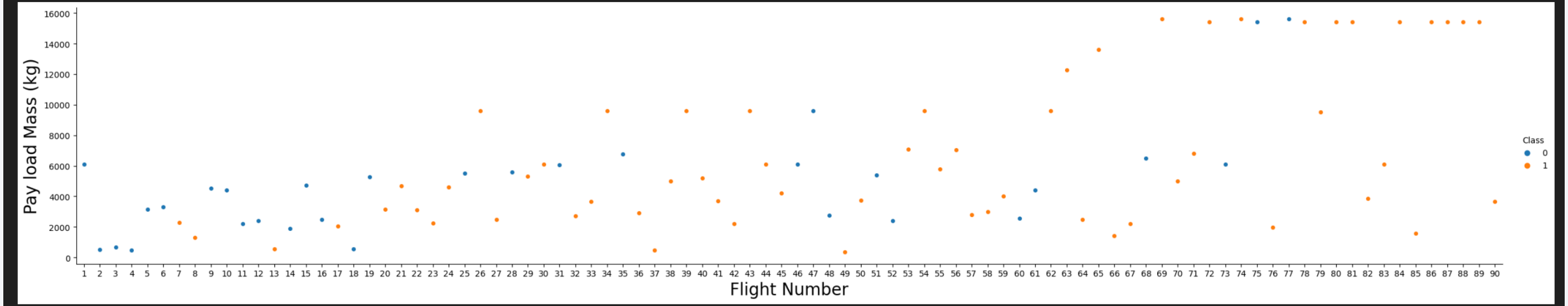
https://github.com/pranaymoluguri/DataScience_Capstone/blob/adb9d578242440cb00ee24220be46ccbaec84ece/8.%20Predictive%20Analysis.ipynb

Results

- Logistic Regression, KNN and SVM were the best models with a very high accuracy of 83.33% for this dataset.
- Higher payloads did not perform as good as the Low weighted payloads.
- Most Successful launch Site : KSC LC 39A
- Orbit GEO, HEO, SSO, ES L1 has the best success rate.
- Increase in success rate of launches as the time is passing by.

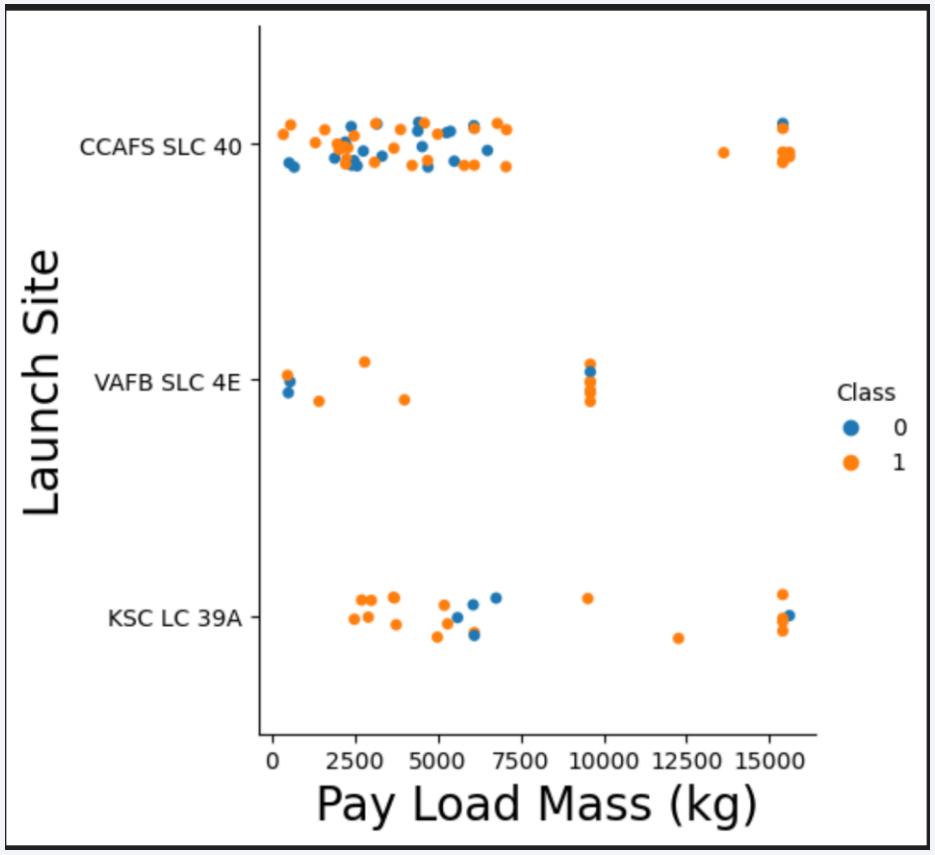

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



We see that different launch sites have different success rates. `CCAFS LC-40`, has a success rate of 60 %, while `KSC LC-39A` and `VAFB SLC 4E` has a success rate of 77%.

Payload vs. Launch Site

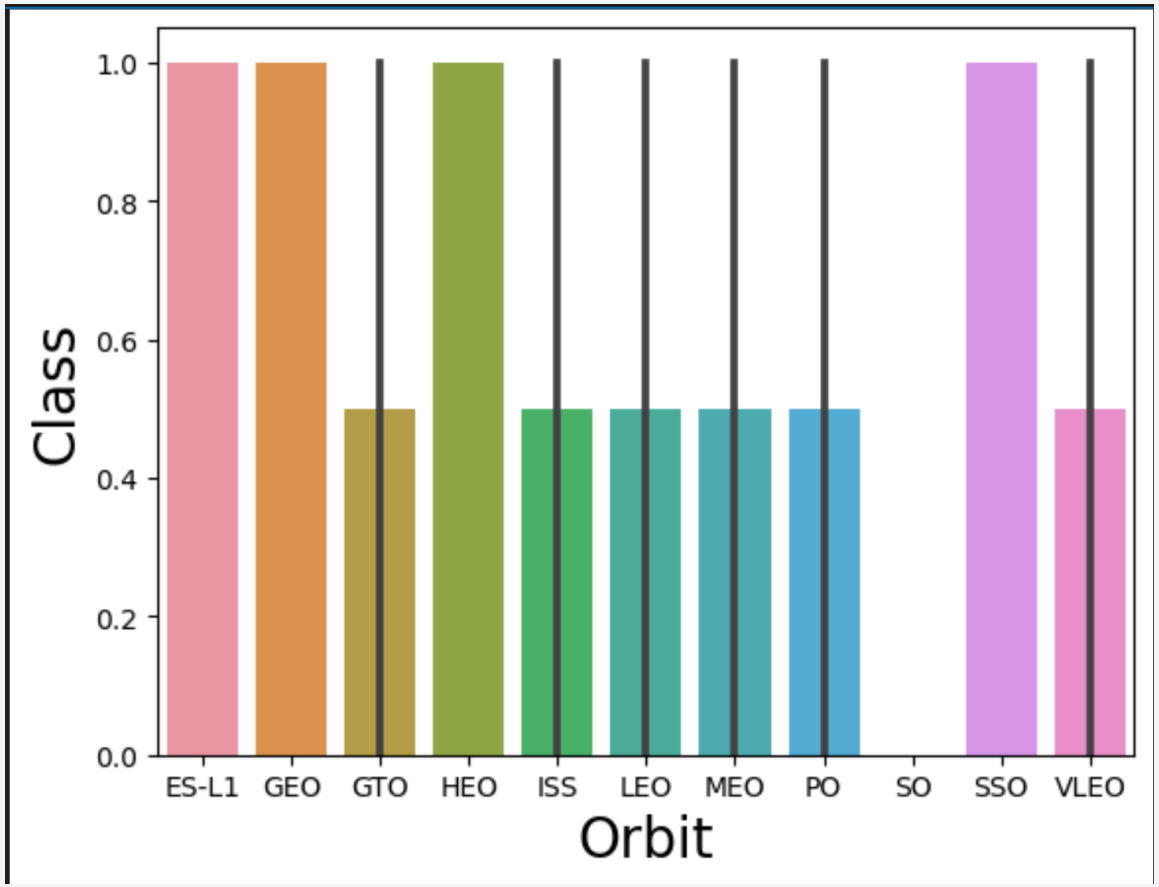


VAFB-SLC 4E launchsite :

No Rockets launched for heavy pad load.

Majority of Low Payload has been launced from CCAFS SLC 40.

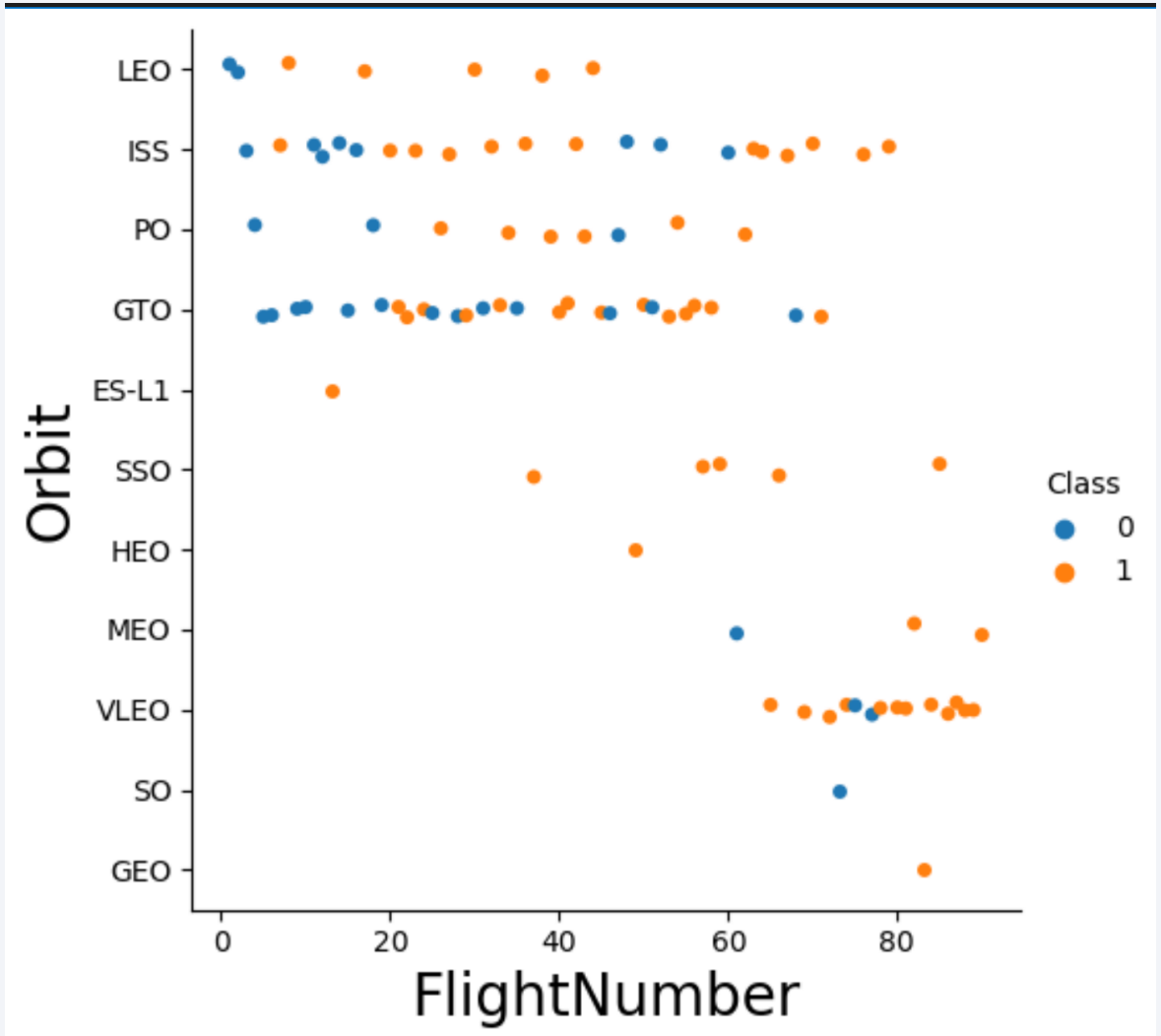
Success Rate vs. Orbit Type



Highest Success Rate:

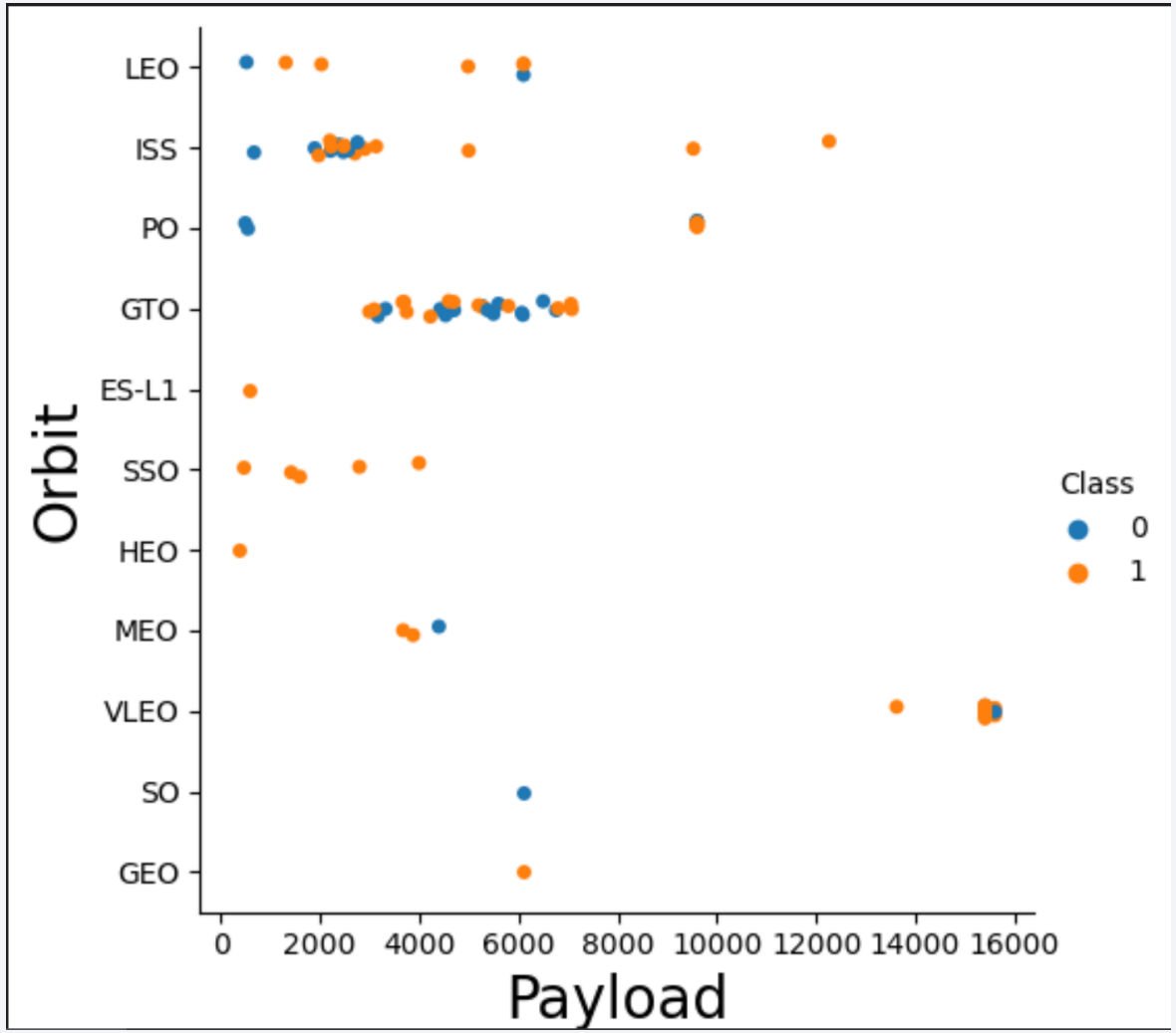
- ES-L1
- GEO
- HEO
- SSO

Flight Number vs. Orbit Type



- Significant relation in LEO orbit but there no relation in GTO
- Significant recent launches of VLEO

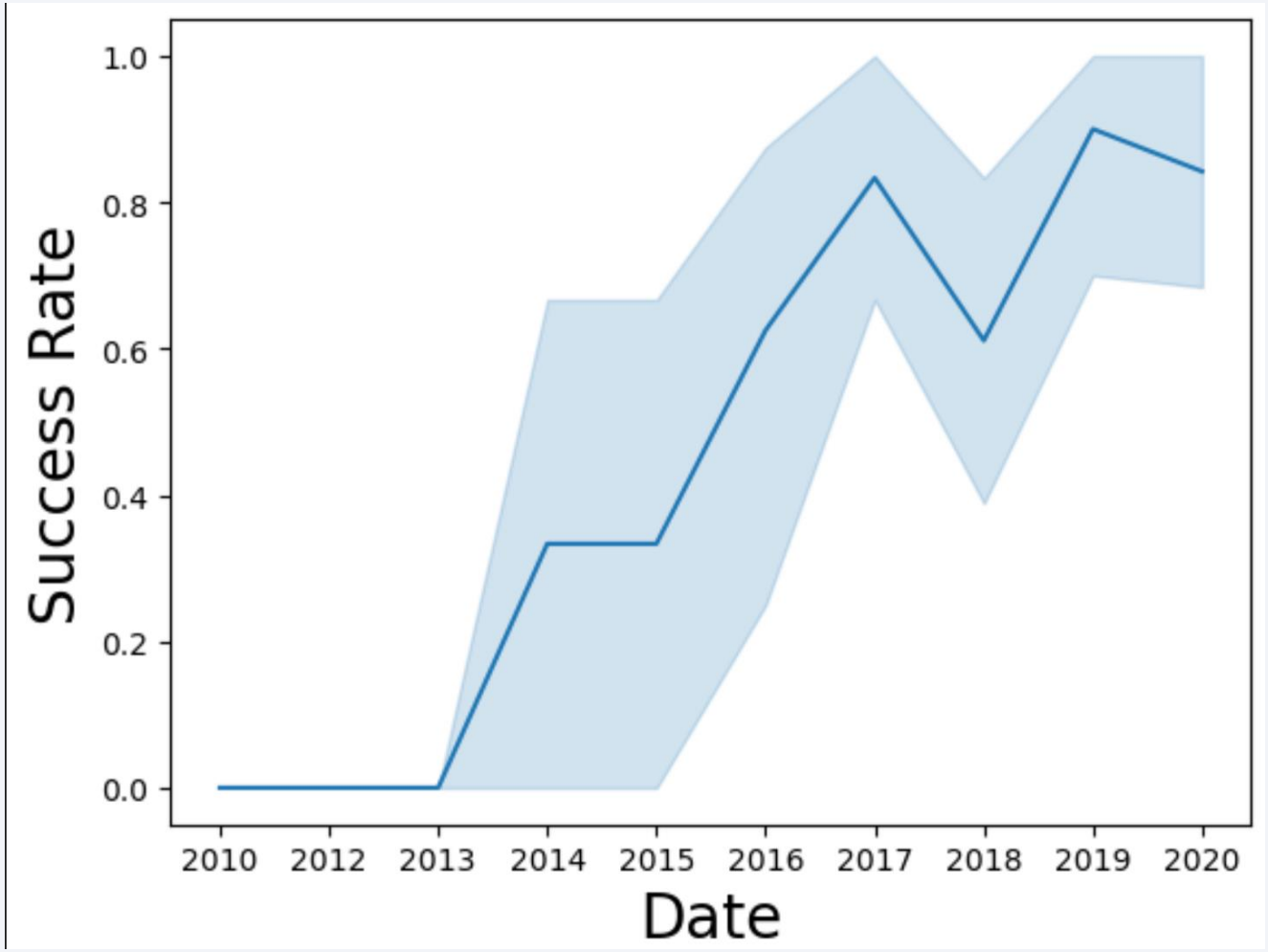
Payload vs. Orbit Type



- Strong correlation between ISS and Payload between 2000 and 4000.
- Similar correlation between GTO and Payload 4000-8000.

Launch Success Yearly Trend

- As the time passed by, the success rate of launches have significantly increased, may be due to advancement in Technology and experience gained from the previous failed launches.



All Launch Site Names

```
%sql SELECT distinct LAUNCH_SITE from SPACEXTBL;
```

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

Launch Site Names Begin with 'CCA'

In [17]:
```
%sql SELECT LAUNCH_SITE FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

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

Out[17]:

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

In [31]:
```
%sql select SUM(PAYLOAD_MASS__KG_) AS PAYLOADMASS FROM SPACEXTBL
```

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

Out[31]:

PAYLOADMASS
619967.0

Average Payload Mass by F9 v1.1

```
%sql select AVG(PAYLOAD_MASS__KG_) AS PAYLOADMASS_AVG FROM SPACEXTBL WHERE BOOSTER_VER='F9 v1.1'
```

2928.400000

First Successful Ground Landing Date

```
%sql select MIN(DATE) from SPACEXTBL where Landing_Outcome='Success(ground pad)'
```

2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome="Success (drone ship)"
and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000;
```

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

Total Number of Successful and Failure Mission Outcomes

In [57]:
```
%sql SELECT SUM(CASE WHEN MISSION_OUTCOME LIKE '%Success%' THEN 1 ELSE 0 END) AS SUCCESS,
SUM(CASE WHEN MISSION_OUTCOME LIKE '%FAIL%' THEN 1 ELSE 0 END) AS FAIL FROM SPACEXTBL;
```

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

Out[57]:

SUCCESS	FAIL
100	1

Boosters Carried Maximum Payload

```
%sql select BOOSTER_VERSION as BoosterVer
from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
```

BoosterVer
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 strftime('%m', DATE) AS Launch_Month, MISSION_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL WHERE strftime('%Y', DATE) = '2015';
```

time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Success (drone ship)
05:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Success (drone ship)
		CCAFS LC-				SKY Perfect JSAT		

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

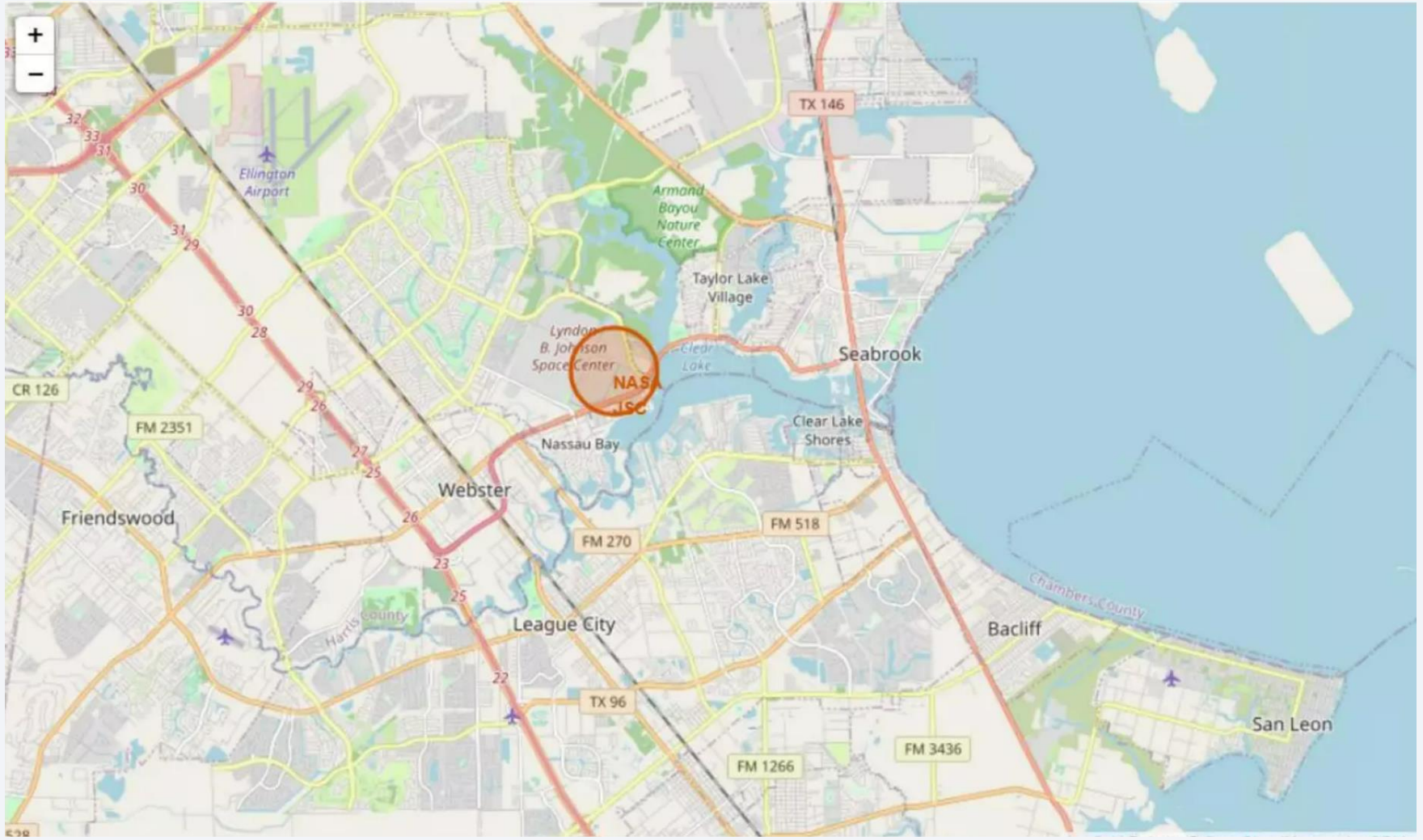
```
%sql SELECT LANDING_OUTCOME FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
ORDER BY DATE DESC;
```

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	05:21:00	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	01:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

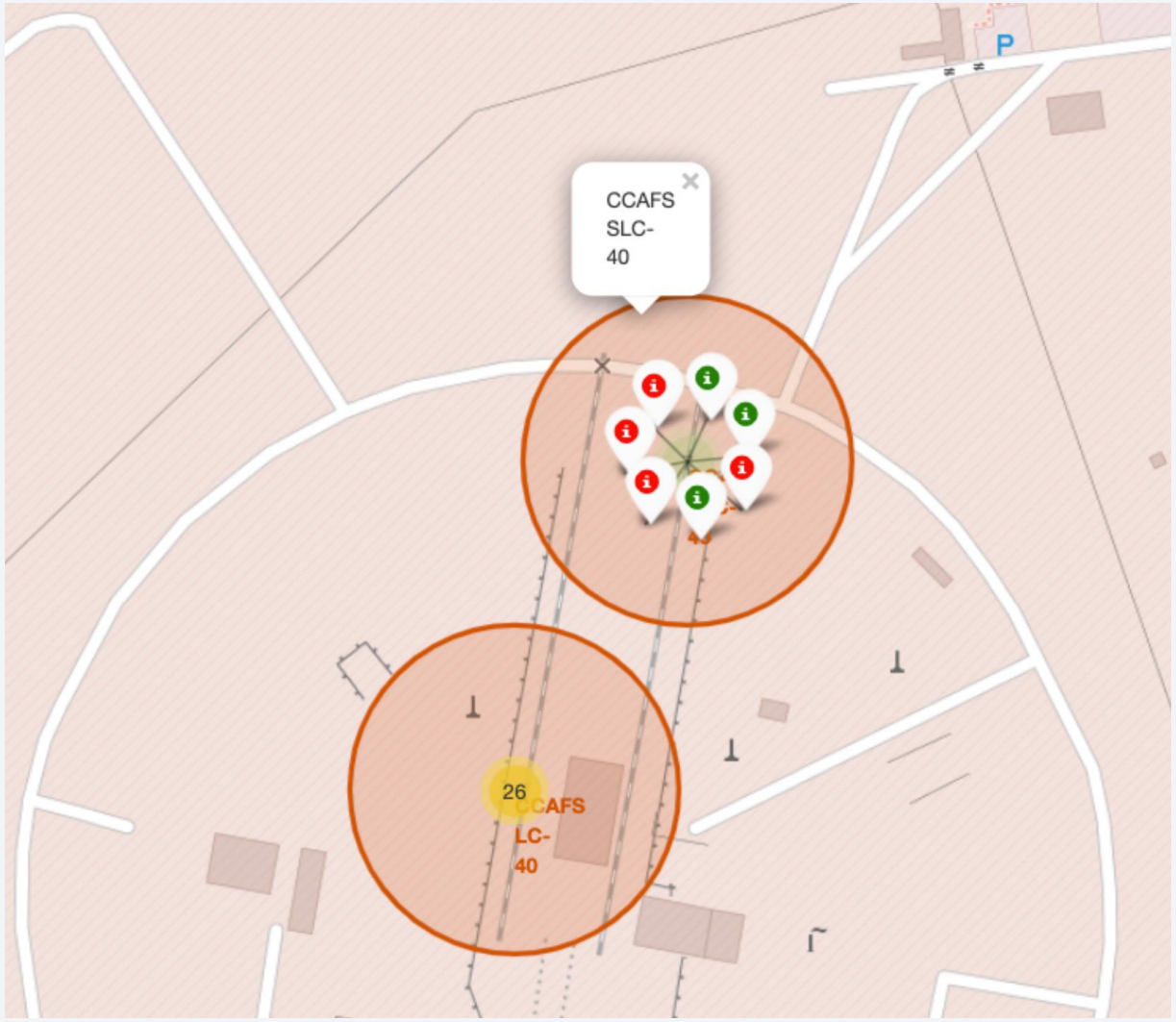
Section 3

Launch Sites Proximities Analysis

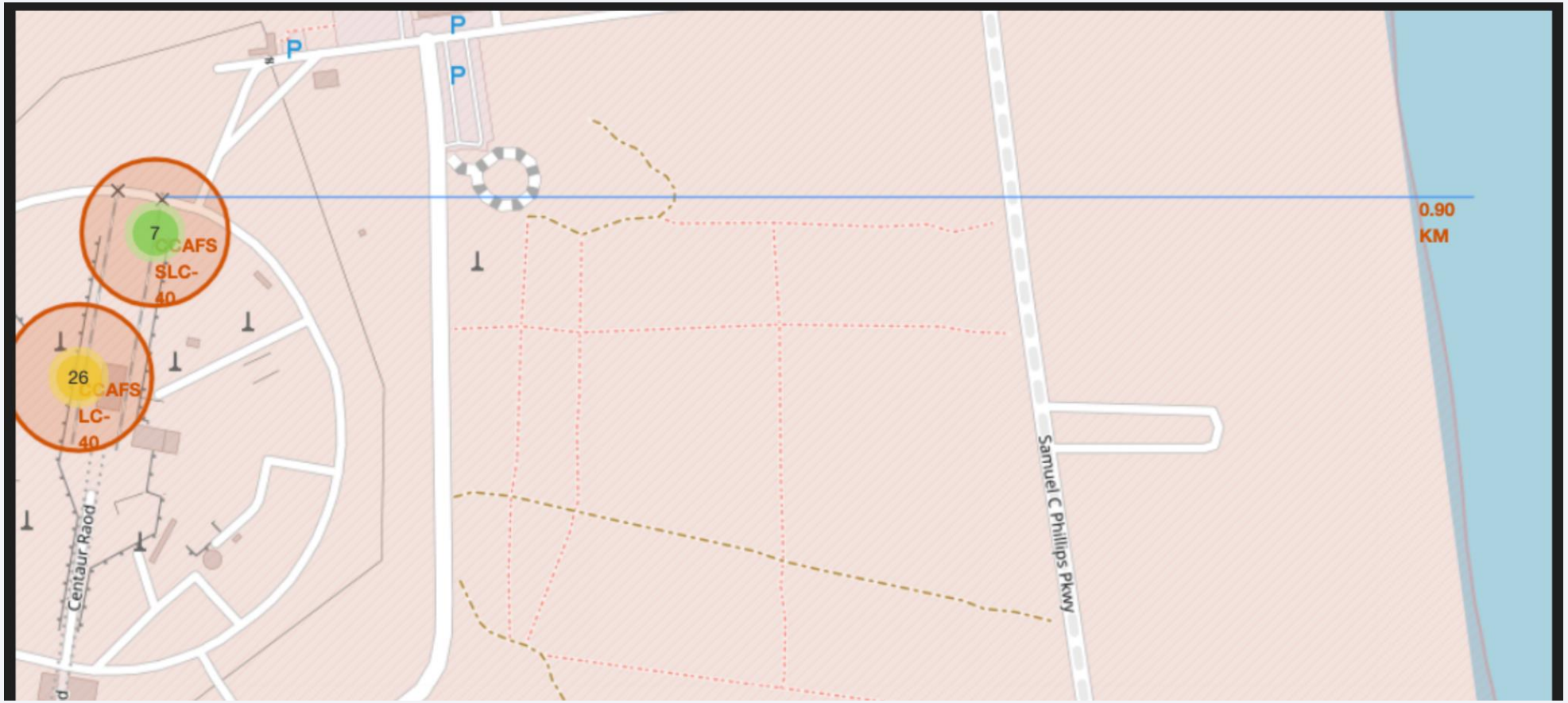
Launch Sites

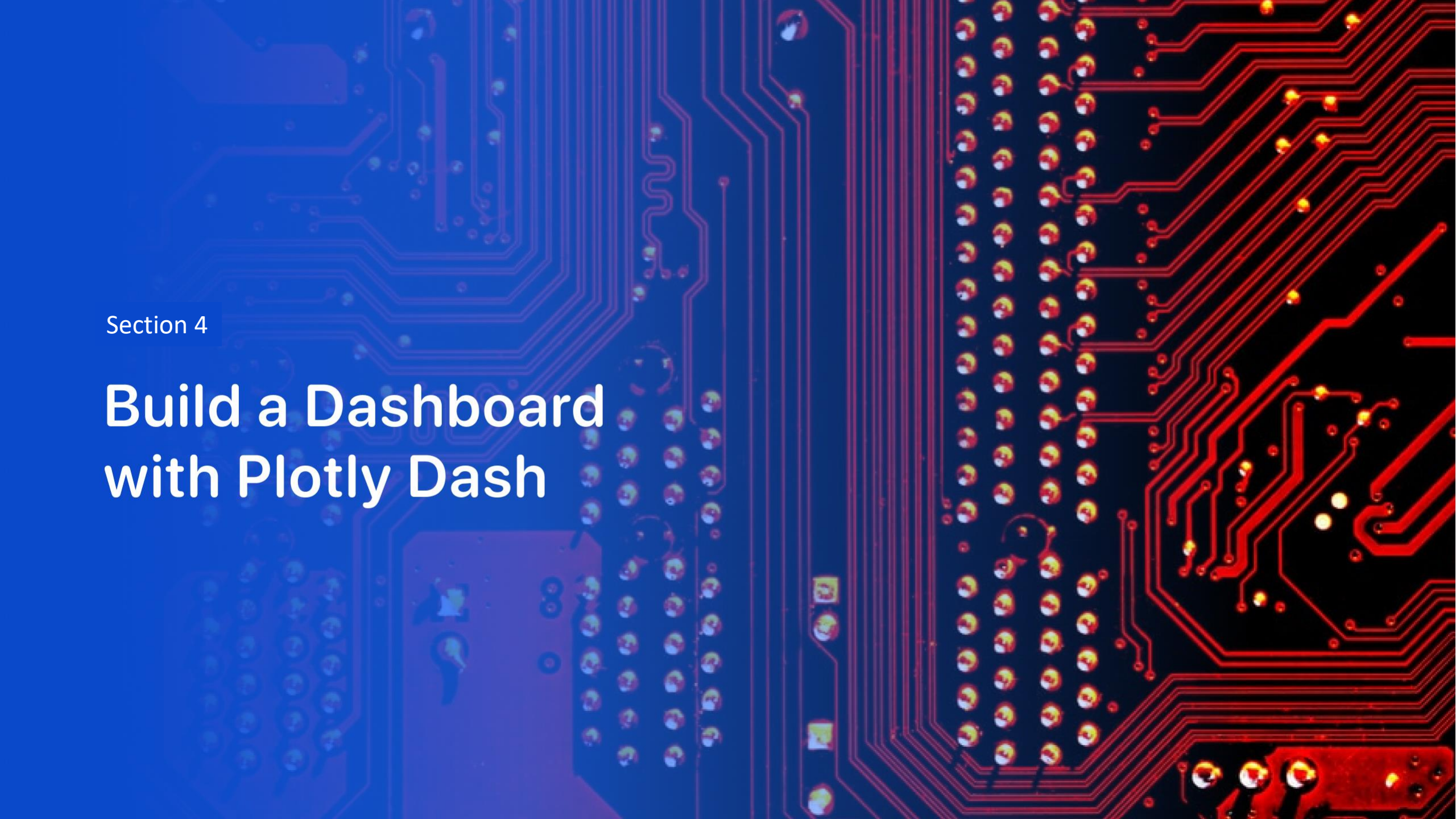


Launch Sites Close

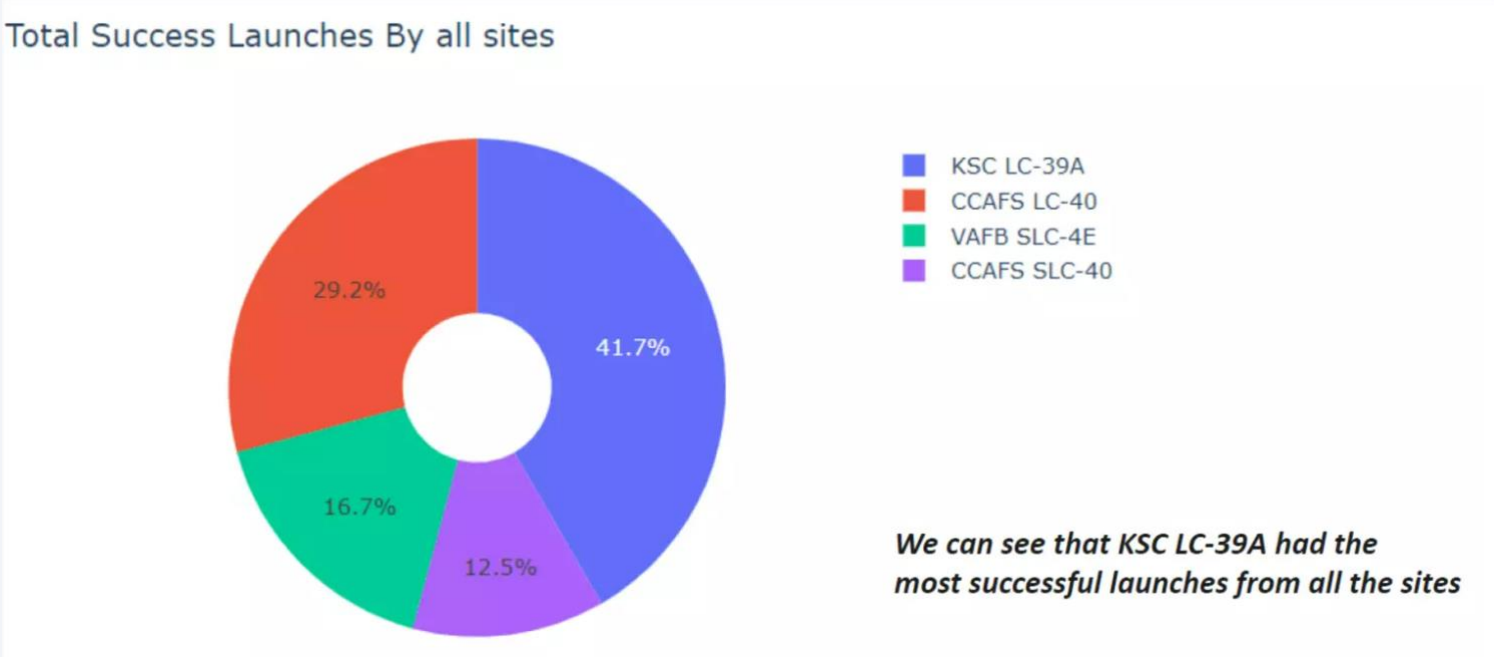


Distance at Launch Site

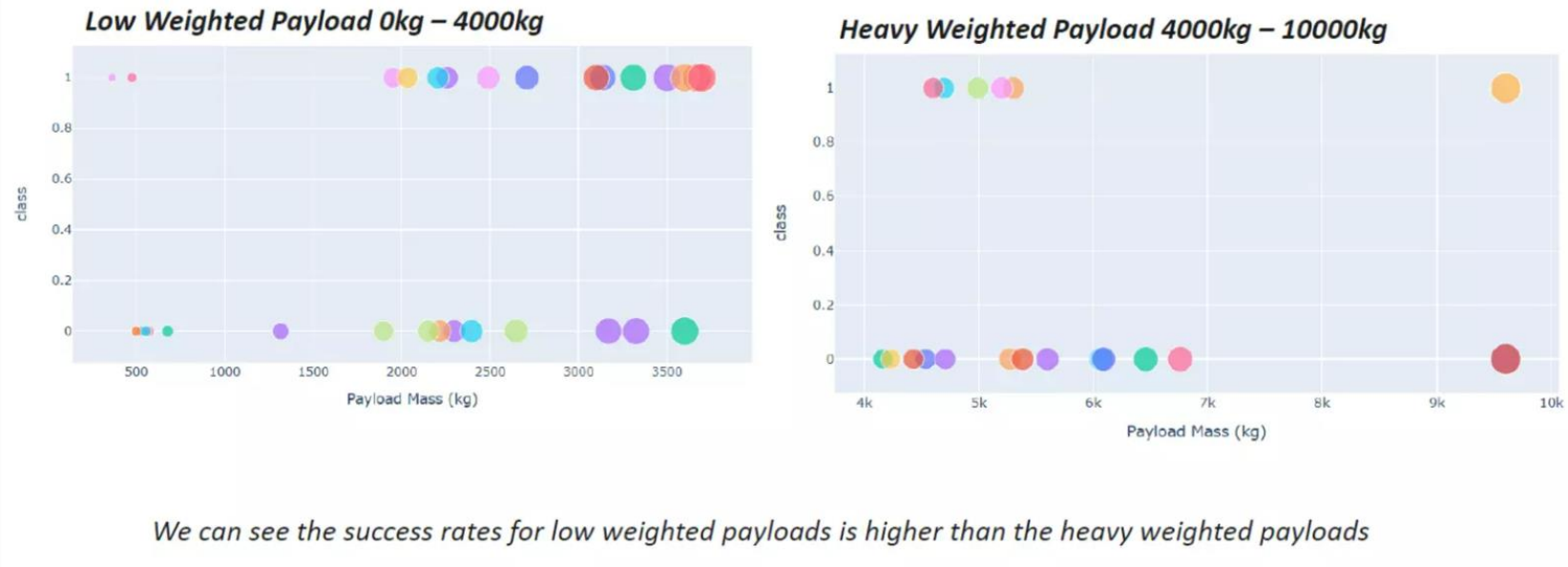



Section 4
Build a Dashboard with Plotly Dash

Pie – Successful Launches

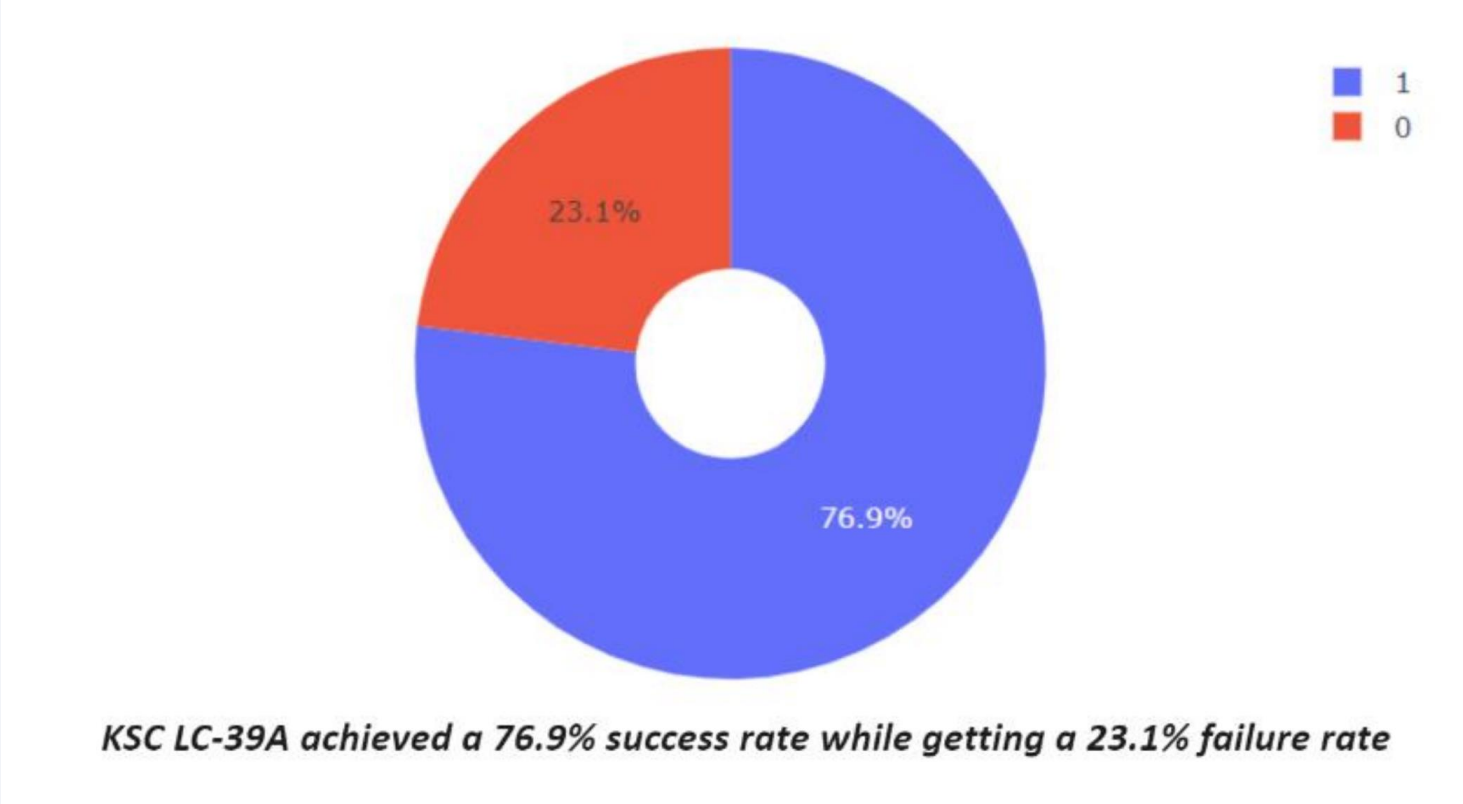


Payload vs Launch



We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Launch Site with Highest Success

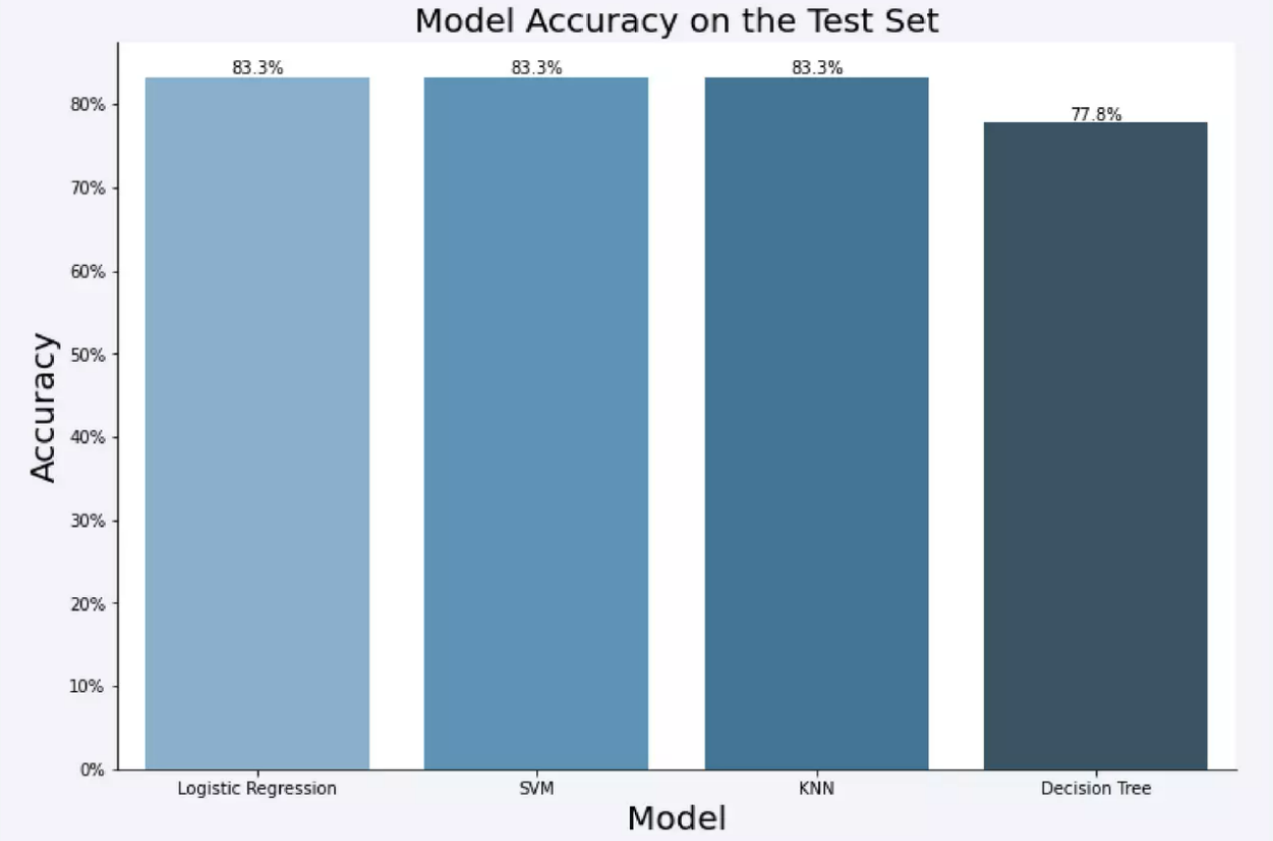


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

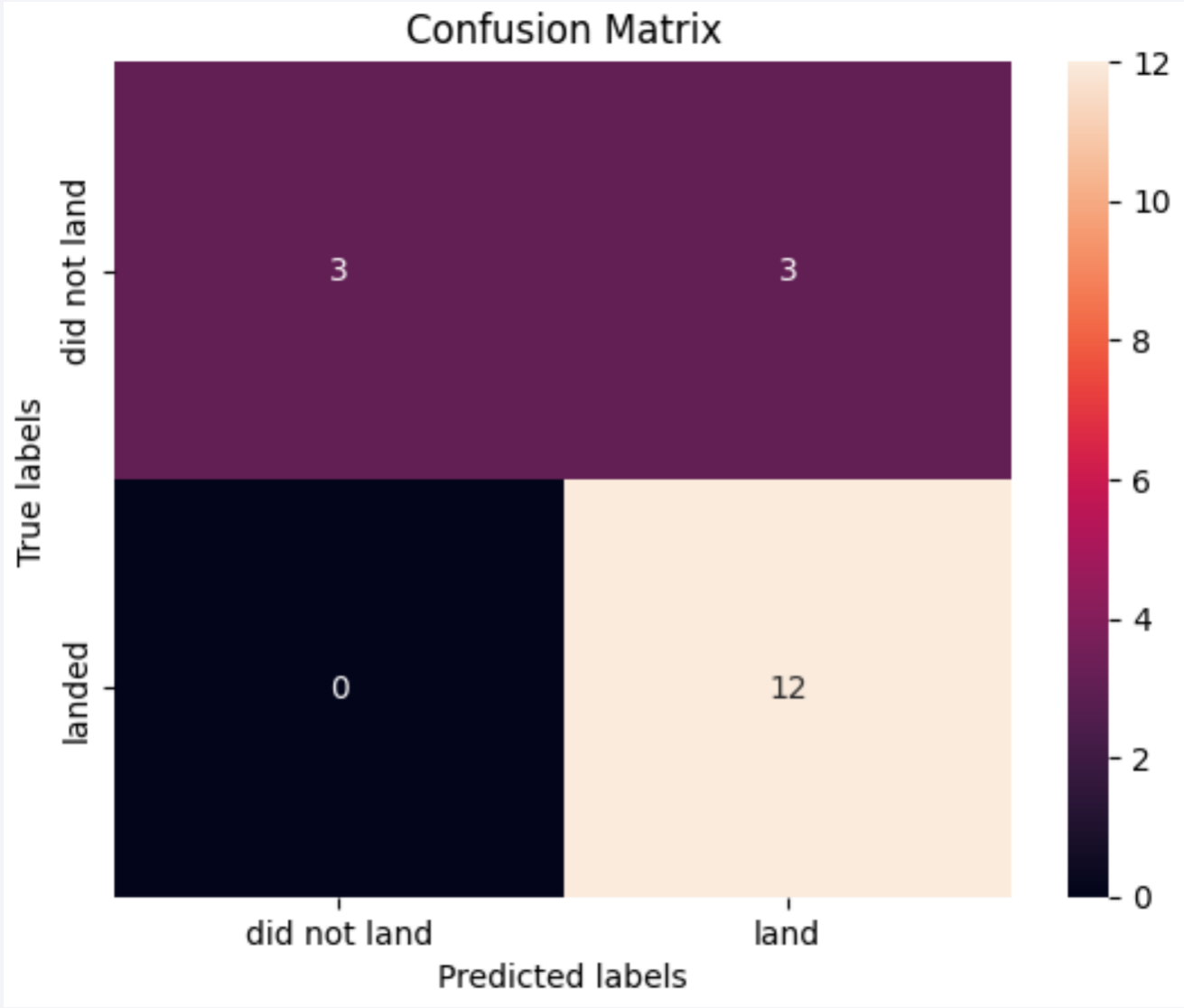
Section 5

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix



Conclusions

- Logistic Regression, KNN and SVM were the best models with a very high accuracy of 83.33% for this dataset.
- Higher payloads did not perform as good as the Low weighted payloads.
- Most Successful launch Site : KSC LC 39A
- Orbit GEO, HEO, SSO, ES L1 has the best success rate.
- Increase in success rate of launches as the time is passing by.

Thank you!