THE RACE FOR SPACE

Munasib Khan

November 2024

OUTLINE

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

EXECUTIVE SUMMARY

- Summary of methodologies
  - Data Collection via API, Web Scraping
  - Exploratory Data Analysis (EDA) with Data Visualization
  - EDA with SQL
  - Interactive Map with Folium
  - Dashboards with Plotly Dash
  - Predictive Analysis
- Summary of all results
  - Exploratory Data Analysis results
  - Interactive maps and dashboard
  - Predictive results

INTRODUCTION

- Project background and context
  - The aim of this project is to predict if the Falcon 9 first stage will successfully land. SpaceX says on its website that the Falcon 9 rocket launch cost 62 million dollars. Other providers cost upward of 165 million dollars each. The price difference is explained by the fact that SpaceX can reuse the first stage. By determining if the stage will land, we can determine the cost of a launch. This information is interesting for another company if it wants to compete with SpaceX for a rocket launch.
- Problems you want to find answers
  - What are the main characteristics of a successful or failed landing ?
  - What are the effects of each relationship of the rocket variables on the success or failure of a landing ?
  - What are the conditions which will allow SpaceX to achieve the best landing success rate ?

METHODOLOGIES

METHODOLOGY

Executive Summary

- Data collection methodology:
    - SpaceX REST API
    - Web Scrapping from Wikipedia
- Perform data wrangling
    - Dropping unnecessary columns
    - One Hot Encoding for classification models
- 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

- Datasets are collected from Rest SpaceX API and webscrapping Wikipedia
  - The information obtained by the API are rocket, launches, payload information.
    - The Space X REST API URL is api.spacexdata.com/v4/

SpaceX Rest API call → API returns JSON file → Make Dataframe from JSON → Clean Data and export it

  - The information obtained by the webscrapping of Wikipedia are launches, landing, payload information.

URL is https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Get HTML response from Wikipedia → Extract data with BeautifulSoup → Make Dataframe → Export Data

DATA COLLECTION – SPACEX API

1. Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

2. Convert Response to JSON File

```
data = response.json()
data = pd.json_normalize(data)
```

3. Transform data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
getBoosterVersion(data)
```

4. Create dictionary with data

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

5. Create dataframe

```
data = pd.DataFrame.from_dict(launch_dict)
```

6. Filter dataframe

```
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

7. Export to file

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

DATA COLLECTION – WEB SCRAPING

1. Getting Response from HTML

```
response = requests.get(static_url)
```

2. Create BeautifulSoup Object

```
soup = BeautifulSoup(response.text, "html5lib")
```

3. Find all tables

```
html_tables = soup.findAll('table')
```

4. Get column names

```
for th in first_launch_table.find_all('th'):
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0 :
        column_names.append(name)
```

5. Create dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Add data to keys

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is a
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.stri
                flag=flight_number.isdigit()
```

See notebook for the rest of code

7. Create dataframe from dictionary

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

8. Export to file

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

DATA WRANGLING

- In the dataset, there are several cases where the booster did not land successfully.
  - True Ocean, True RTLS, True ASDS means the mission has been successful.
  - False Ocean, False RTLS, False ASDS means the mission was a failure.
- We need to transform string variables into categorical variables where 1 means the mission has been successful and 0 means the mission was a failure.

1. Calculate launches number for each site

```
df['LaunchSite'].value_counts()
```

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

2. Calculate the number and occurrence of each orbit

```
df['Orbit'].value_counts()
```

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

3. Calculate number and occurrence of mission outcome per orbit type

```
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
None ASDS       2
False Ocean     2
False RTLS      1
Name: Outcome, dtype: int64
```

4. **Create landing outcome** label **from** Outcome column

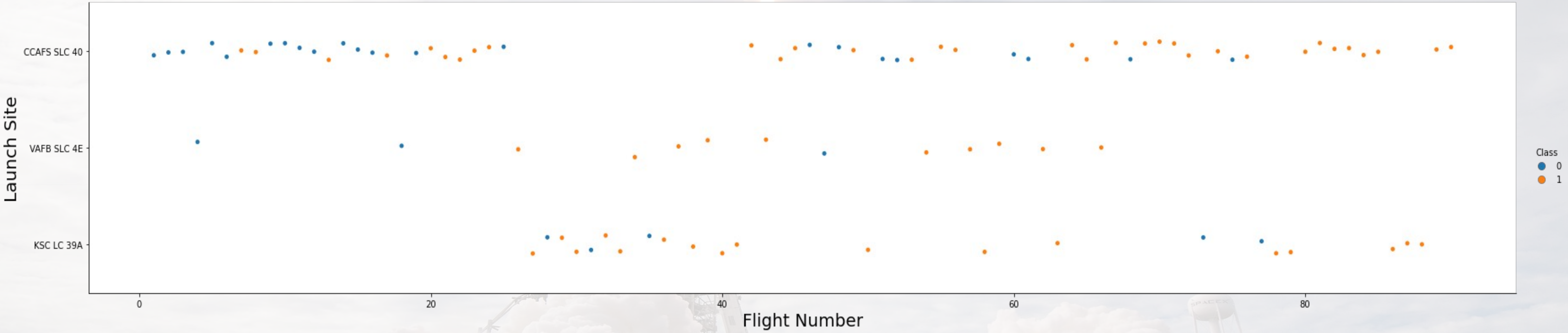
```
landing_class = []
for key,value in df["Outcome"].items():
     if value in bad_outcomes:
        landing_class.append(0)
     else:
        landing_class.append(1)
df['Class']=landing_class
```

5. Export to file

```
df.to_csv("dataset_part_2.csv", index=False)
```

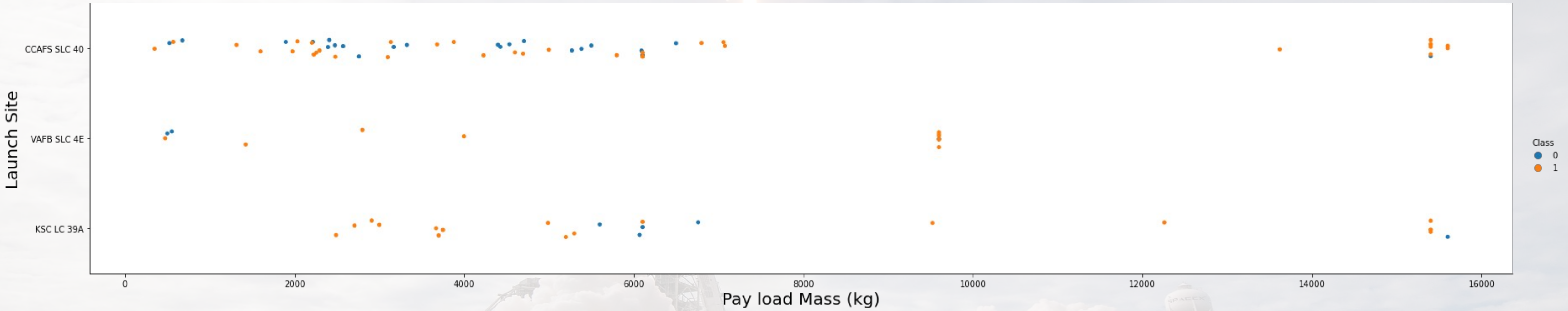
EDA WITH DATA VISUALIZATION

FLIGHT NUMBER vs LAUNCH



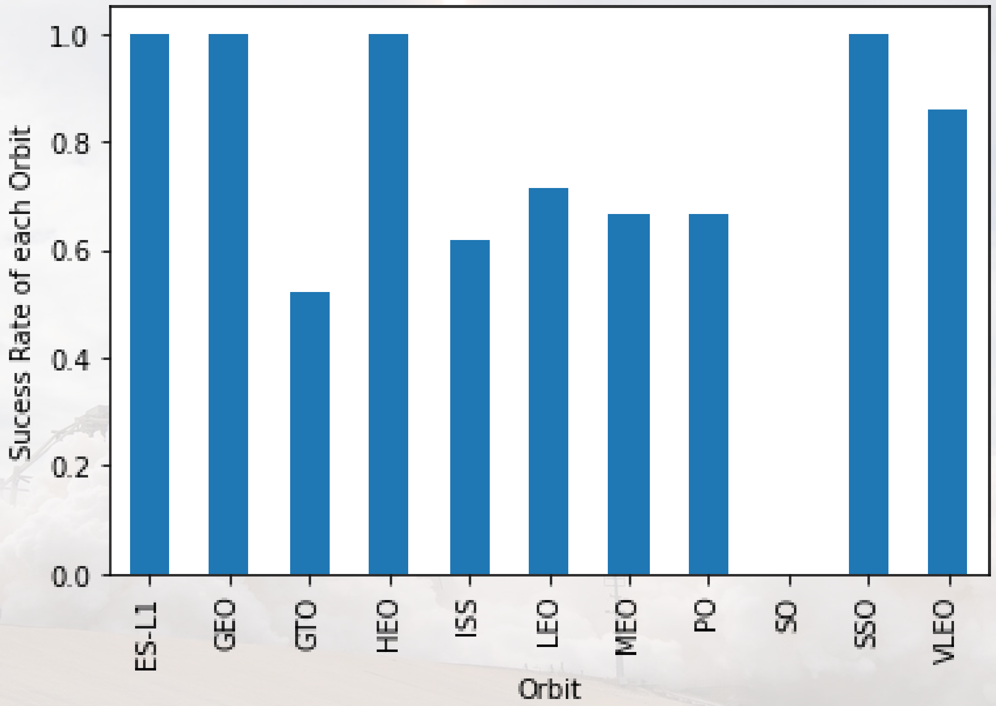
We observe that, for each site, the success rate is increasing.

PAYLOAD vs LAUNCH SITE

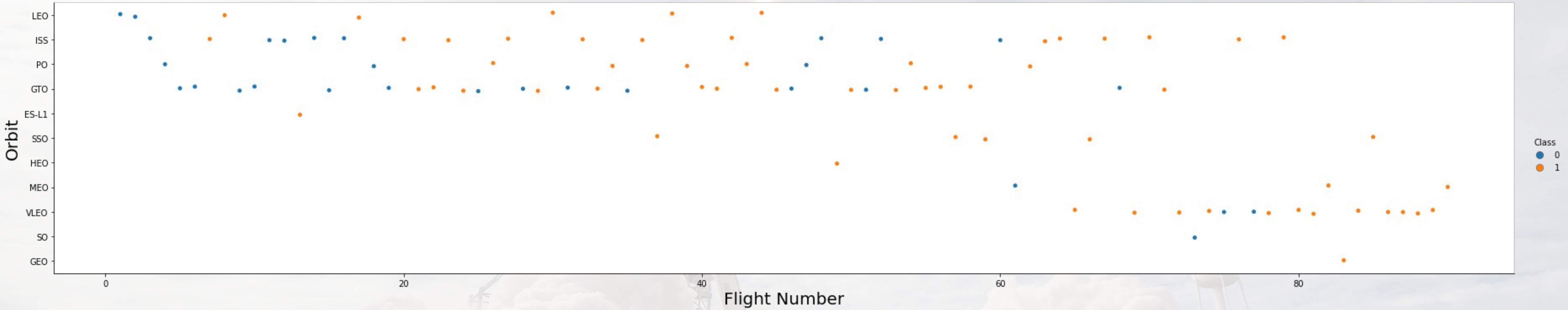


Depending on the launch site, a heavier payload may be a consideration for a successful landing. On the other hand, a too heavy payload can make a landing fail.

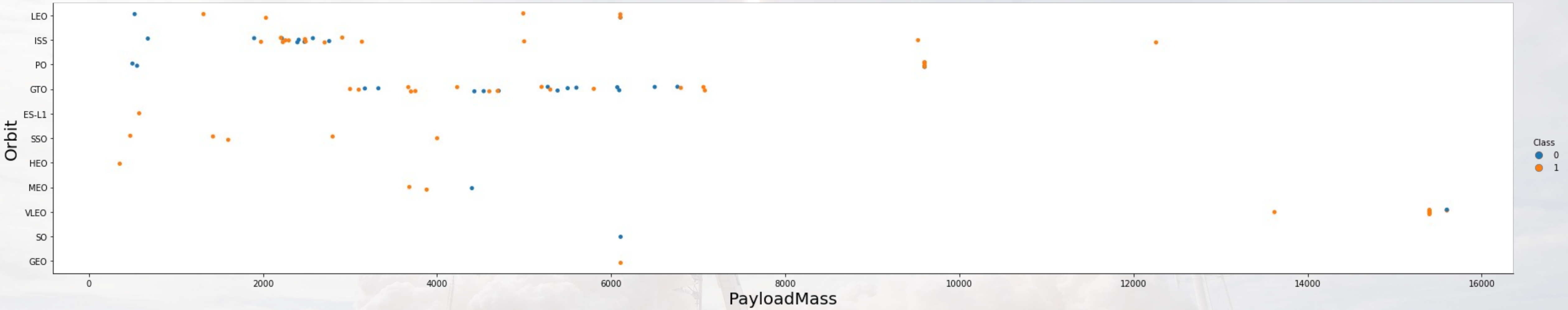
SUCCESS RATE vs ORBIT TYPE



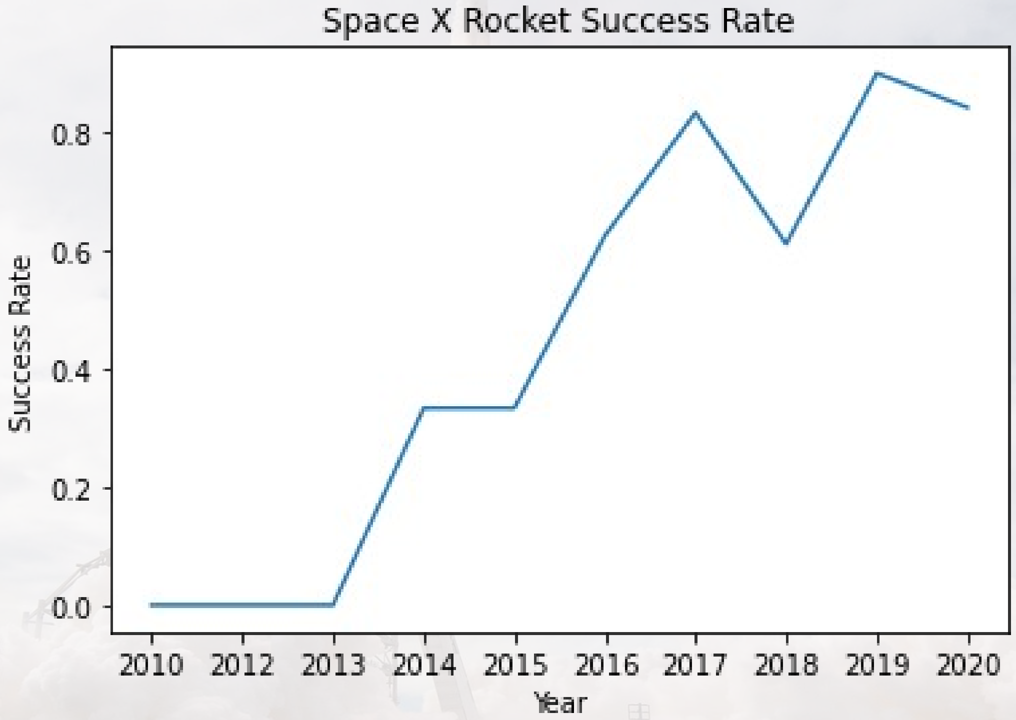
FLIGHT NUMBER vs ORBIT TYPE



PAYLOAD vs ORBIT TYPE



LAUNCH SUCCESS YEARLY TREND



Since 2013, we can see an increase in the Space X Rocket success rate.

ALL LAUNCHES SITES NAMES

SQL Query

```
SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

Results

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

LAUNCH SITE NAME BEGINS WITH 'CCA'

SQL Query

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

Results

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
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
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)

TOTAL PAYLOAD MASS

SQL Query

```
SELECT SUM("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "CUSTOMER" = 'NASA (CRS)'
```

Results

SUM("PAYLOAD_MASS__KG_")
45596

AVERAGE PAYLOAD BY MASS FOR FALCON 9

SQL Query

```
SELECT AVG("PAYLOAD_MASS__KG_") FROM SPACEXTBL WHERE "BOOSTER_VERSION" LIKE '%F9 v1.1%'
```

Results

AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

FIRST SUCCESSFUL GROUND LANDING

SQL Query

```
SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing _Outcome" LIKE '%Success%'
```

Results

MIN("DATE")
01-05-2017

SUCCESSFUL DRONE SHIP LANDING PAYLOAD BETWEEN 4000 - 6000

SQL Query

```
%sql SELECT "BOOSTER_VERSION" FROM SPACEXTBL WHERE "LANDING _OUTCOME" = 'Success (drone ship)' \
AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000;
```

Results

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

TOTAL SUCCESS AND FAILURES

SQL Query

```
%sql SELECT (SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Success%') AS SUCCESS, \
(SELECT COUNT("MISSION_OUTCOME") FROM SPACEXTBL WHERE "MISSION_OUTCOME" LIKE '%Failure%') AS FAILURE
```

Results

SUCCESS	FAILURE
100	1

BOOSTER WITH MAXIMUM PAYLOAD

SQL Query

```
%sql SELECT DISTINCT "BOOSTER_VERSION" FROM SPACEXTBL \
WHERE "PAYLOAD_MASS__KG_" = (SELECT max("PAYLOAD_MASS__KG_") FROM SPACEXTBL)
```

Results

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

LAUNCH RECORDS IN 2015

SQL Query

```
%sql SELECT substr("DATE", 4, 2) AS MONTH, "BOOSTER_VERSION", "LAUNCH_SITE" FROM SPACEXTBL\
WHERE "LANDING _OUTCOME" = 'Failure (drone ship)' and substr("DATE",7,4) = '2015'
```

Results

MONTH	Booster_Version	Launch_Site
01	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

LANDING OUTCOMES
2010-06-04 - 2017-03-20

SQL Query

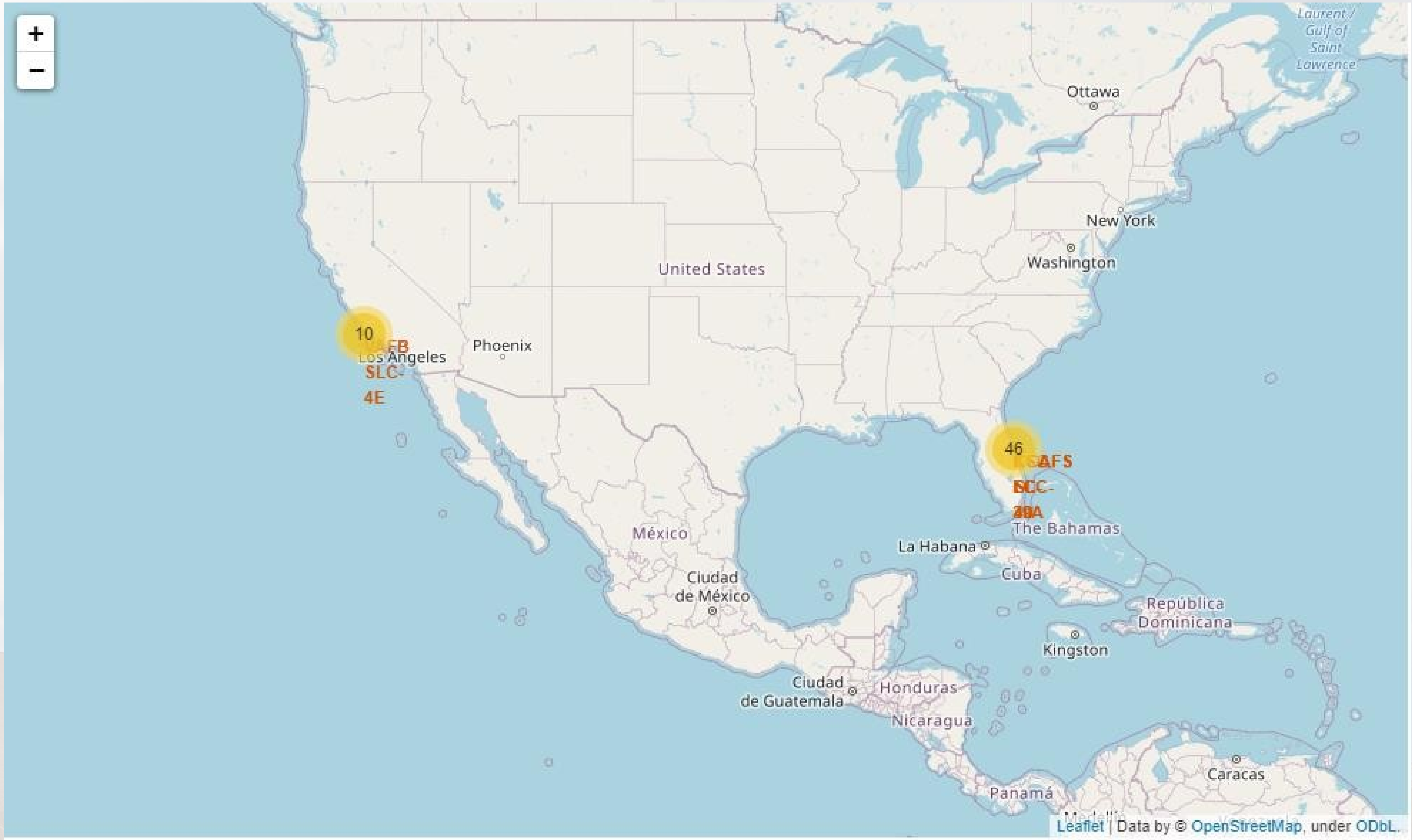
```
%sql SELECT "LANDING _OUTCOME", COUNT("LANDING _OUTCOME") FROM SPACEXTBL\
WHERE "DATE" >= '04-06-2010' and "DATE" <= '20-03-2017' and "LANDING _OUTCOME" LIKE '%Success%'\
GROUP BY "LANDING _OUTCOME" \
ORDER BY COUNT("LANDING _OUTCOME") DESC ;
```

Results

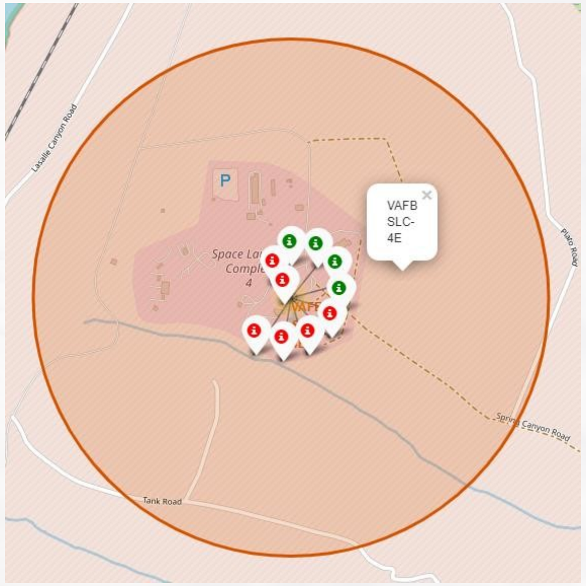
Landing _Outcome	COUNT("LANDING _OUTCOME")
Success	20
Success (drone ship)	8
Success (ground pad)	6

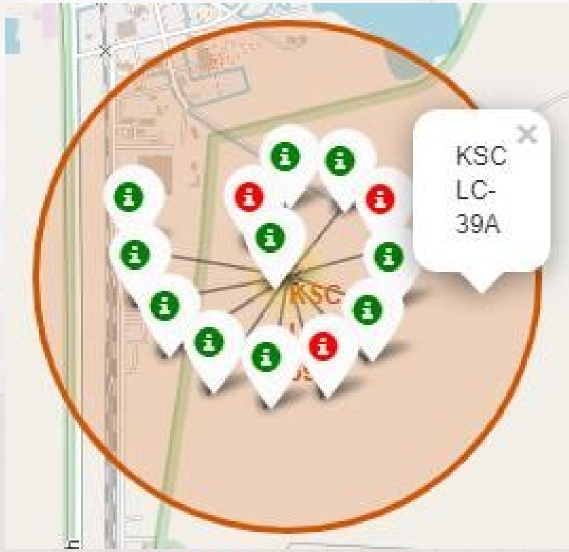
LAUNCH SITES PROXIMITIES ANALYSIS

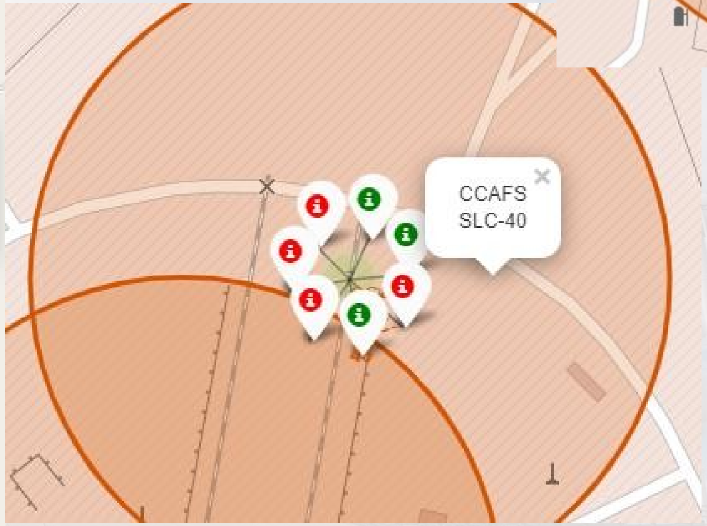
LAUNCH SITES – FOLIUM MAP

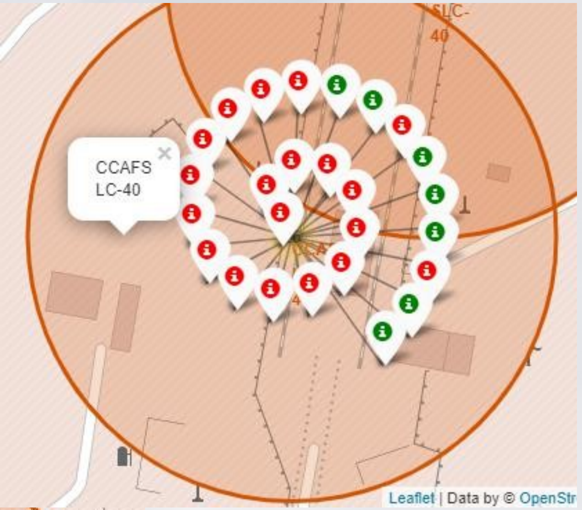


COLORED MARKERS – FOLIUM MAP



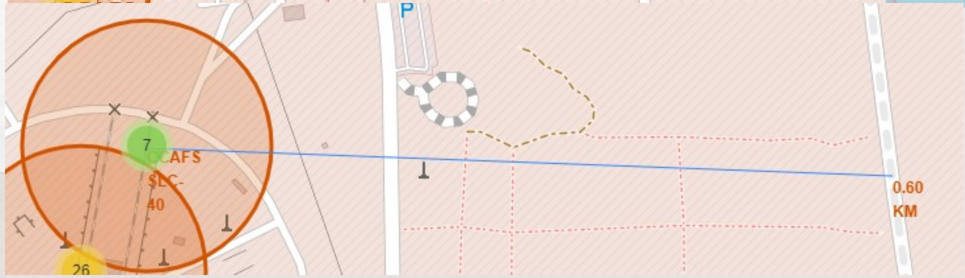






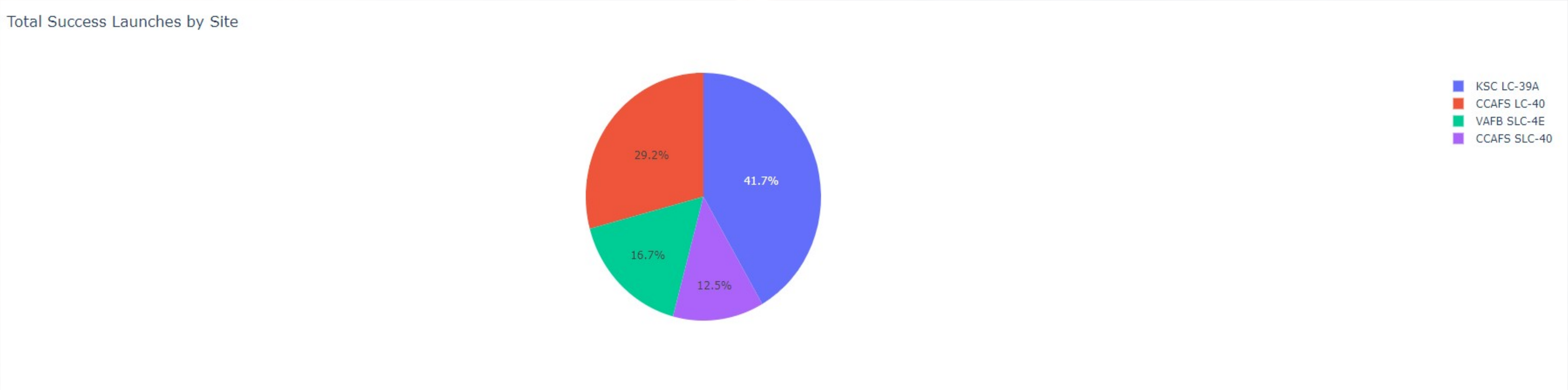
RED – UNSUCCESSFUL LAUNCHES
GREEN – SUCCESSFUL LAUNCHES

CCAFS SLC-40 DISTANCES AND PROXIMITIES

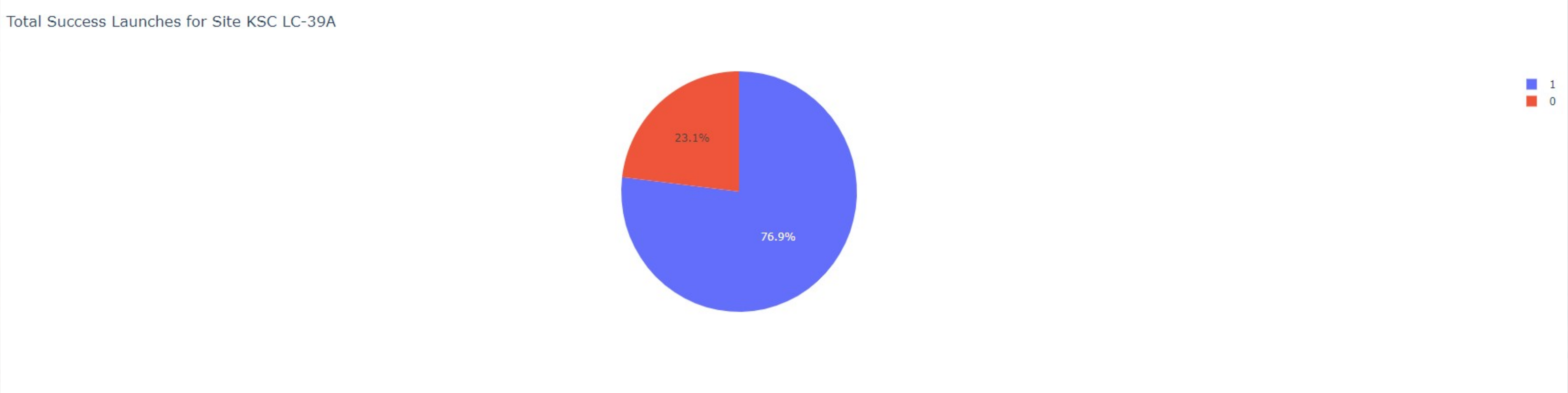


DASH BOARD PLOTLY DASH

TOTAL SUCCESS RATIO BY SITE - DASHBOARD

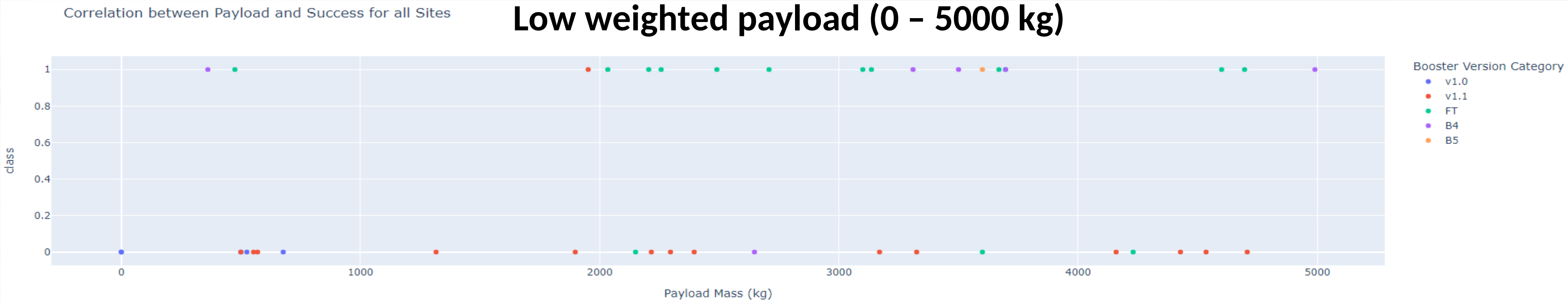


KSC LC-39A SUCCESS LAUNCHES - DASHBOARD

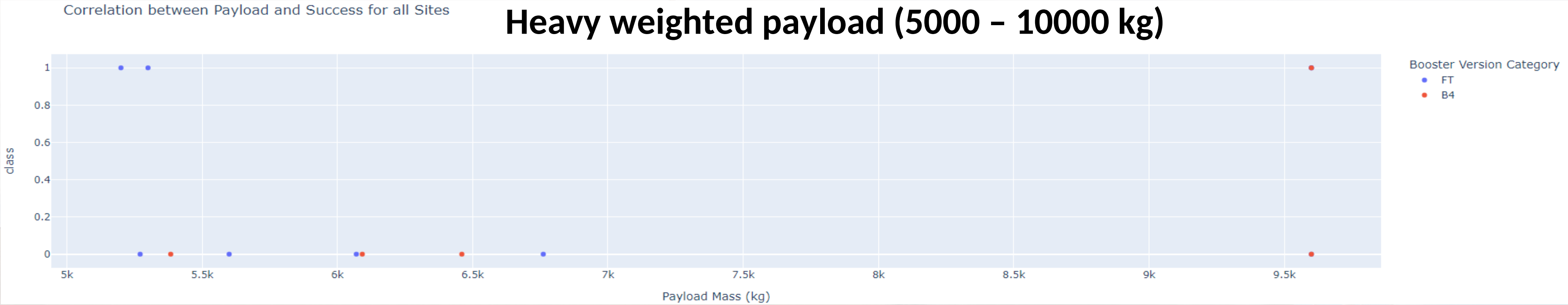


PAYLOAD MASS OUTCOMES FOR DIFFERENT INPUTS - DASHBOARD

Low weighted payload (0 – 5000 kg)



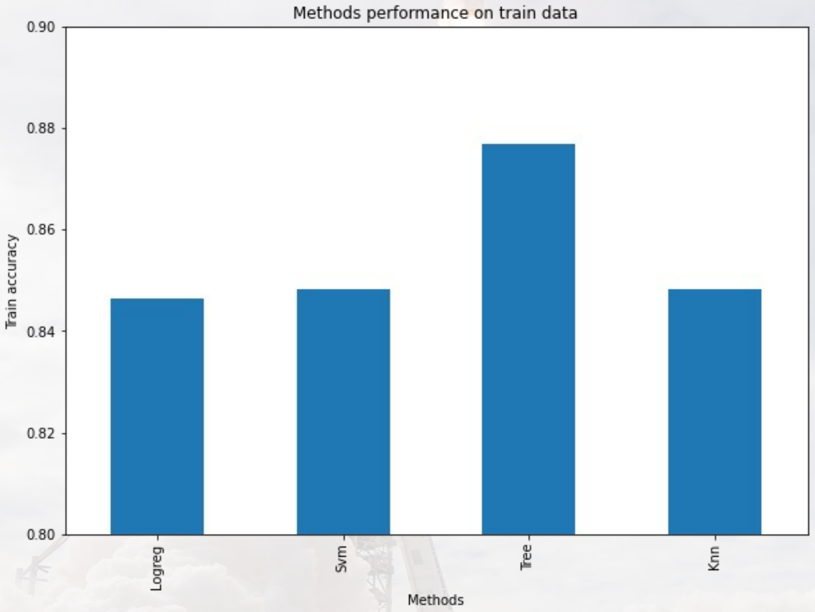
Heavy weighted payload (5000 – 10000 kg)

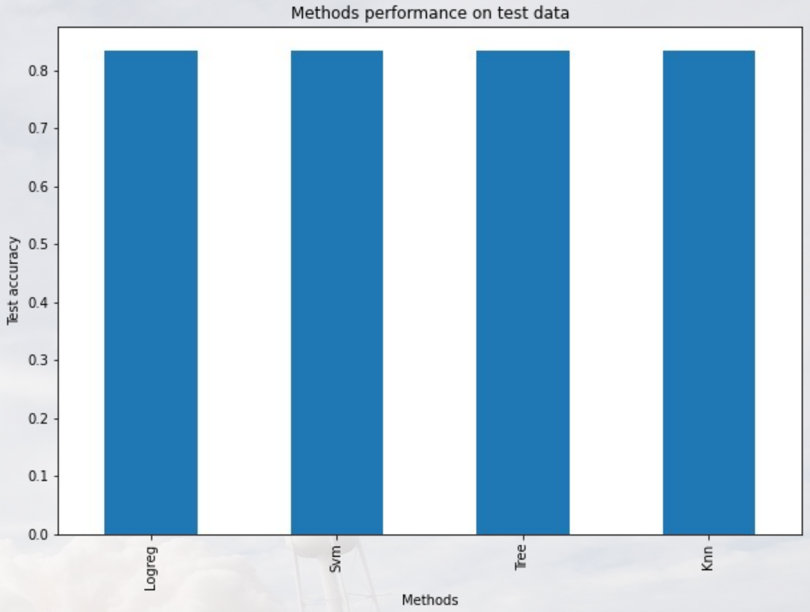


PREDICTIVE ANALYSIS CLASSIFICATION

CLASSIFICATION ACCURACY RESULTS

	Accuracy Train	Accuracy Test
Tree	0.876786	0.833333
Knn	0.848214	0.833333
Svm	0.848214	0.833333
Logreg	0.846429	0.833333





Decision tree best parameters

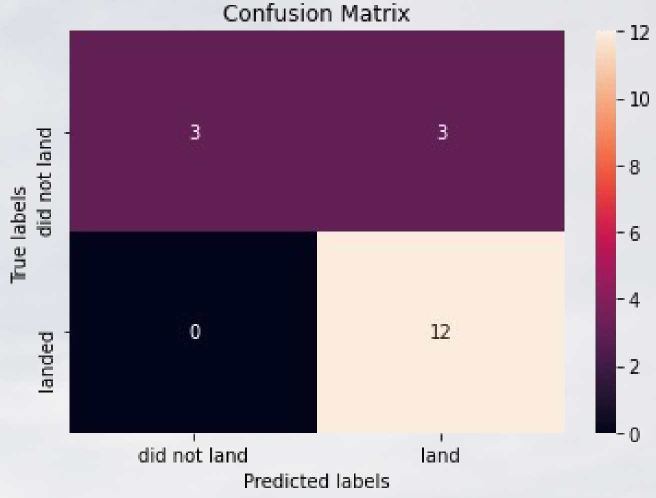
```
tuned hyperparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 12, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 2, 'splitter': 'random'}
```

CONFUSION MATRIX

Logistic regression

SVM

Decision Tree



KNN

		Actual values	
		1	0
Predicted values	1	TP	FP
	0	FN	TN

CONCLUSION

- **With many failed missions, lessons are learned and performance is better. Indeed the success depends on several factors such orbit and payload.**
- **Low weight payload missions are more successful than heavy weight payloads**
- **With the current data we can see that some launches sites are successful than the other but we don't know the why. Perhaps more data is required to for better assumption.**
- **For Model testing the Decision Tree Algorithm showed same results and proved the best.**


THANK YOU FOR BEING WITH ME