Winning Space Race with Data Science

Bhuvanesh Sharma
Dated: July 13, 2023

Outline

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

Executive Summary

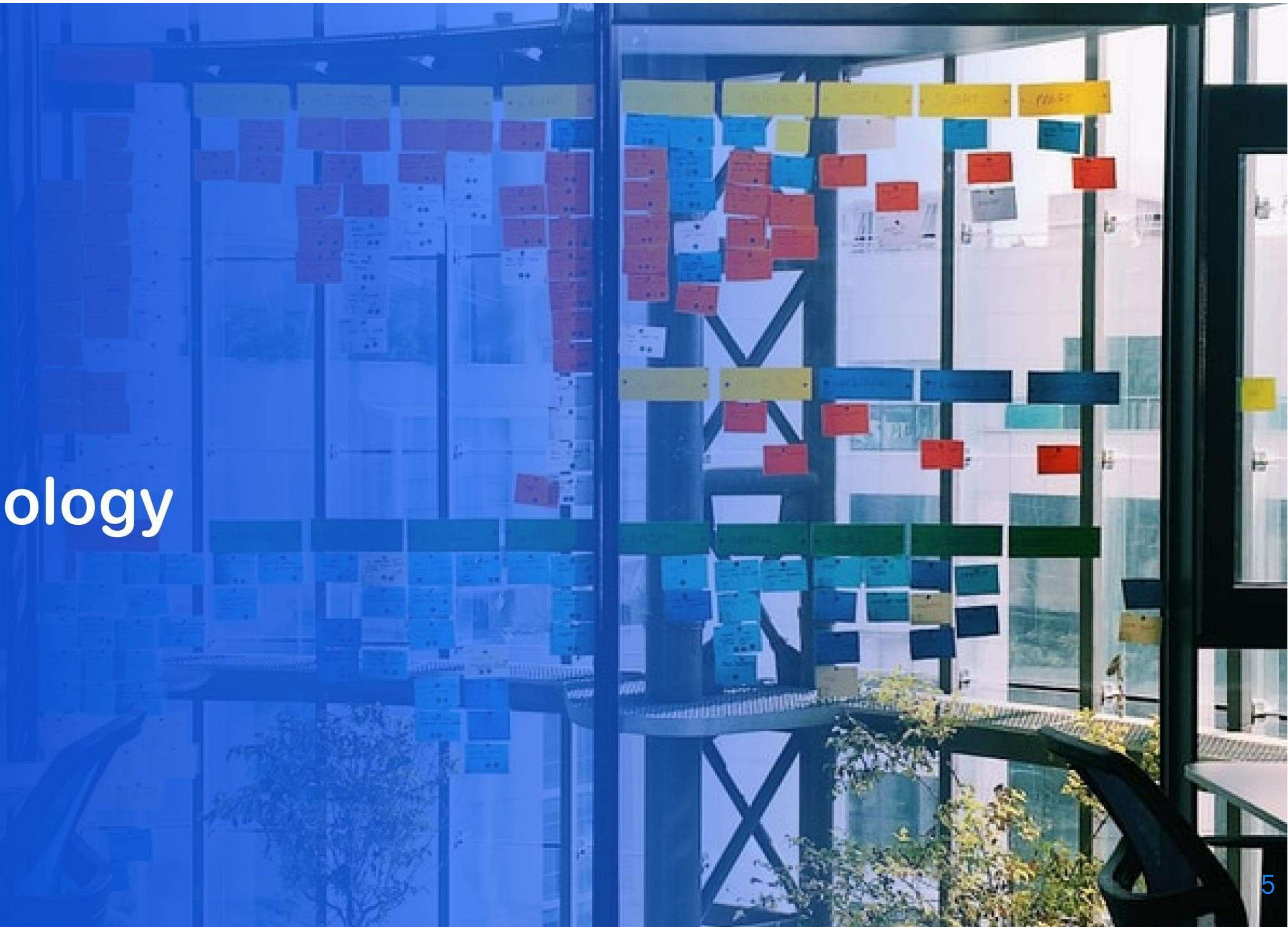
- Summary of methodologies
  - Data Collection via API, Web Scraping
  - Exploratory Data Analysis (EDA) with Data Visualization
  - EDA with SQL
  - Interactive Map with Folium
  - Dashboards with Plotly Dash
  - Predictive Analysis
- Summary of all results
  - Exploratory Data Analysis results
  - 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 claim on its website that the Falcon 9 rocket launch cost 62 million US dollars. While other providers cost upward of 165 million US dollars each. This variation in price 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 can help another company to know 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?
  - How do the rocket variables affect the success or failure of a landing?
  - What conditions allow SpaceX to achieve the best landing success rate?

Section 1

Methodology



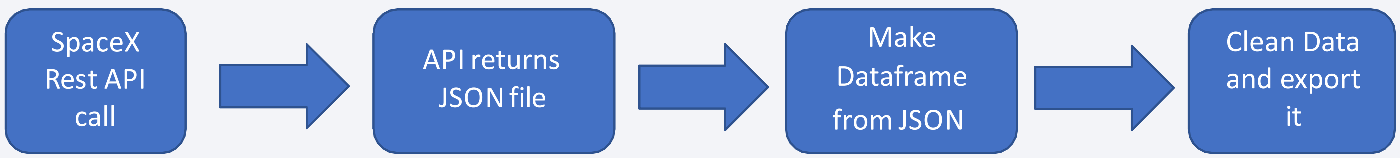
Methodology

Executive Summary

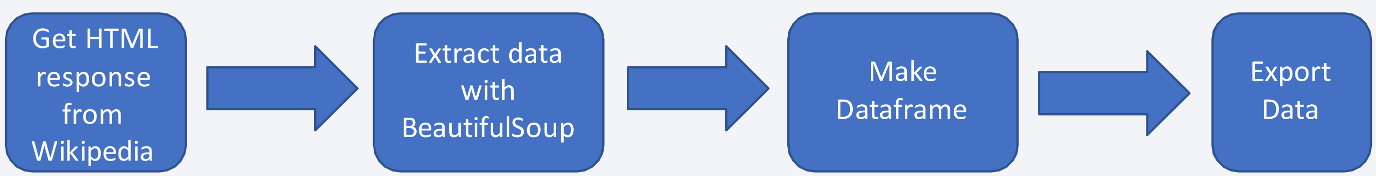
- Data collection methodology:
    - SpaceX REST API
    - Web Scrapping from Wikipedia
- Perform data wrangling
    - Dropping unnecessary columns
    - One Hot Encoding for classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics 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/



  - 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



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 - 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, and True ASDS mean 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 implies that the mission has been successful and 0 implies that the mission was a failure.

The below approach is used to wrangle the data.

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 occurence 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

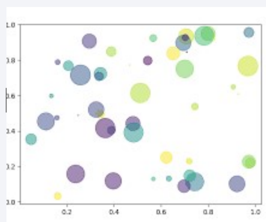
```
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
df['Class']=landing_class
```

5. Export to file

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

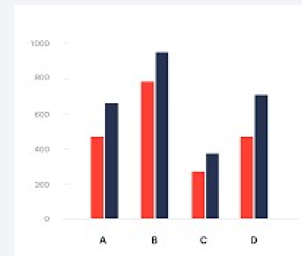
EDA with Data Visualization

- Scatter Graphs
  - Flight Number vs. Launch Site
  - Orbit vs. Flight Number
  - Payload vs. Orbit Type
  - Flight Number vs. Payload
  - Orbit vs. Payload Mass
  - Payload vs. Launch Site



Scatter plots show the relationship between variables. This relationship is called the correlation.

- Bar Graph
  - Success rate vs. Orbit



Bar graphs show the relationship between numeric and categoric variables.

- Line Graph
  - Success rate vs. Year

Line graphs show data variables and their trends. Line graphs can help to show global behavior and make prediction for unseen data.

EDA with SQL

- We performed SQL queries to gather and understand data from the dataset:
  - Displaying 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 the ground pad was achieved.
  - List the names of the boosters which have success in drone ships and have payload mass greater than 4000 but less than 6000.
  - List the total number of successful and failed mission outcomes.
  - List the names of the booster_versions which have carried the maximum payload mass.
  - List the records which will display the month names, failure landing_ouutcomes in drone ship, booster versions, launch_site for the months in year 2015.
  - Rank the count of successful landiing_outcomes between the dates 04-06-2010 and 20-03-2017 in descending order.

Build an Interactive Map with Folium

- The folium map object is a map centered on NASA Johnson Space Center in Houston, Texas
  - The red circle at NASA Johnson Space Center's coordinates with a label showing its name *(folium.Circle, folium.map.Marker).*
  - Red circles at each launch site coordinate with label showing the launch site name *(folium.Circle, folium.map.Marker, folium.features.DivIcon).*
  - The grouping of points in a cluster to display multiple and different information for the same coordinates *(folium.plugins.MarkerCluster).*
  - Markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing. *(folium.map.Marker, folium.Icon).*
  - Markers to show the distance between the launch site to key locations (railway, highway, coastway, city) and plot a line between them. *(folium.map.Marker, folium.PolyLine, folium.features.DivIcon)*
- These objects are created in order to understand the problem and the data better. We can show easily all launch sites, their surroundings, and the number of successful and unsuccessful landings.

Build a Dashboard with Plotly Dash

- Dashboard has dropdown, pie chart, rangeslider, and scatter plot components.
  - The dropdown allows a user to choose the launch site or all launch sites. (dash_core_components.Dropdown).
  - Pie chart shows the total success and the total failure for the launch site chosen with the dropdown component (plotly.express.pie).
  - Rangeslider allows a user to select a payload mass in a fixed range (dash_core_components.RangeSlider).
  - The scatter chart shows the relationship between the two variables. In our case, Success vs Payload Mass (plotly.express.scatter).

Predictive Analysis (Classification)

Data preparation

- Load dataset
- Normalize data
- Split data into training and test sets.

Model preparation

- Selection of machine learning algorithms
- Set parameters for each algorithm to GridSearchCV
- Training GridSearchModel models with the training dataset

Model evaluation

- Get the best hyperparameters for each type of model
- Compute the accuracy for each model with the test dataset
- Plot Confusion Matrix

Model comparison

- Comparison of models according to their accuracy
- The model with the best accuracy will be chosen (see Notebook for the result)

Results

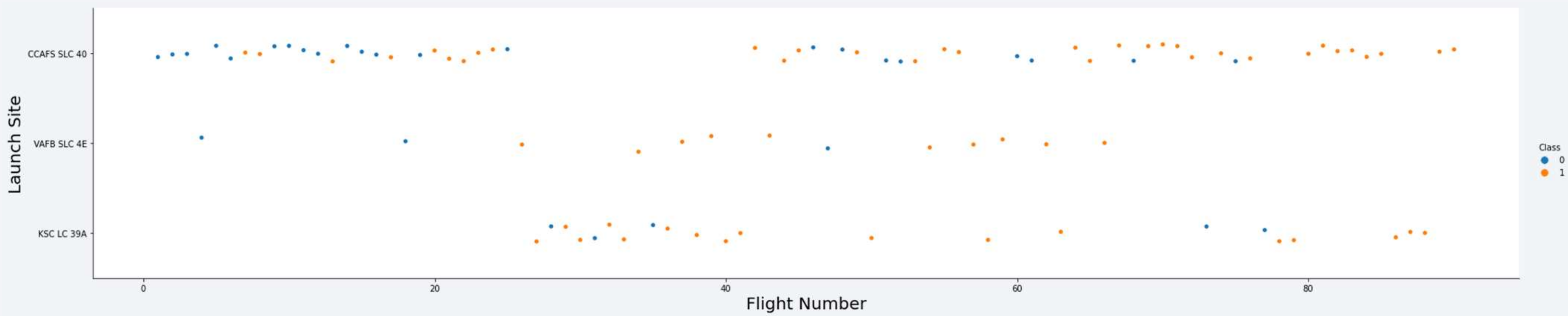
- Exploratory data analysis results
- Interactive analytics demo in snaps
- Predictive analysis results

Section 2

Insights drawn from EDA

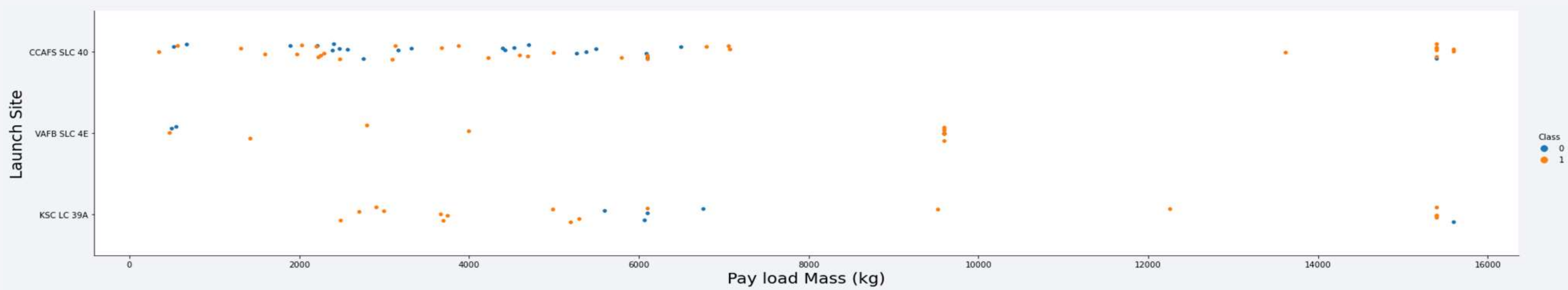
Flight Number vs. Launch Site

For each site, we see an increasing trend of success rate.



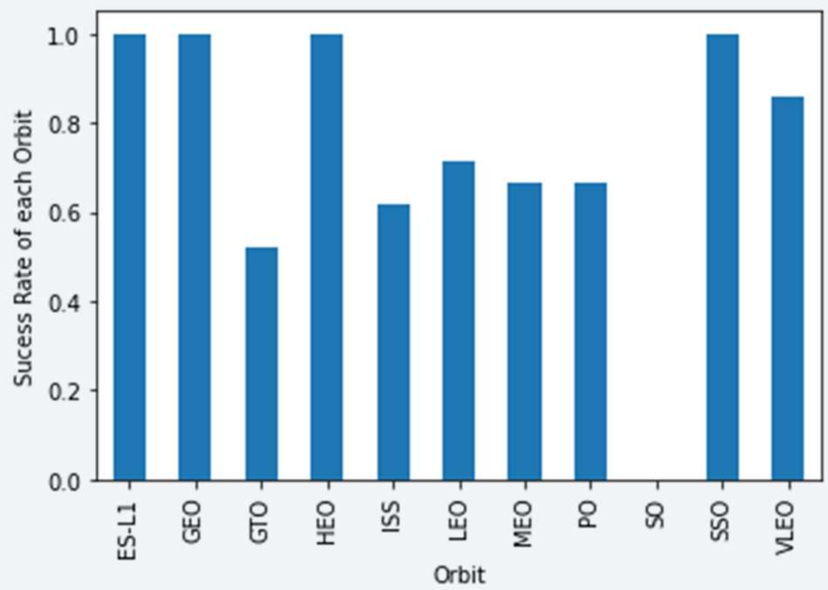
Payload vs. Launch Site

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



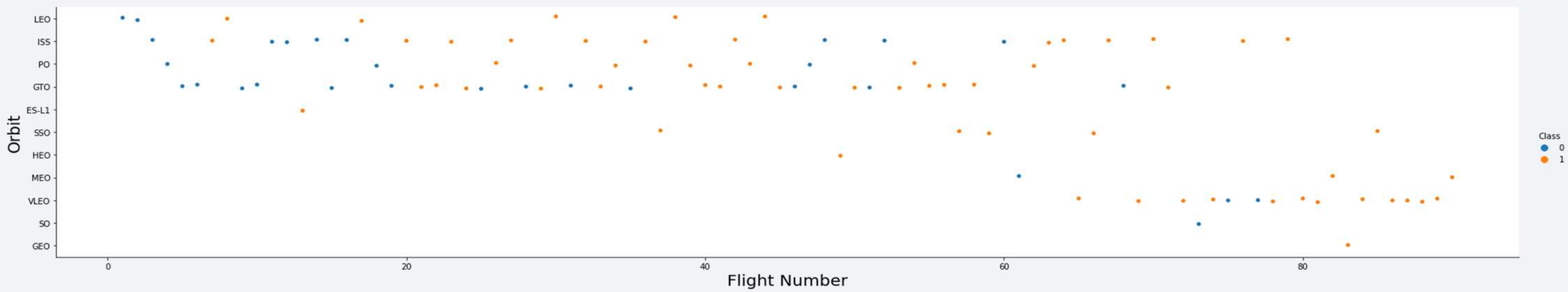
Success Rate vs. Orbit Type

With this plot, we can see the success rate for different orbit types. We observed that ES-L1, GEO, HEO, and SSO have the best success rate.



Flight Number vs. Orbit Type

We observed that the success rate increases with the number of flights for the LEO orbit. For some orbits like GTO, there is no relation between the success rate and the number of flights.
But we can also observe that the high success rate of some orbits like SSO or HEO is due to the knowledge learned during former launches for other orbits.

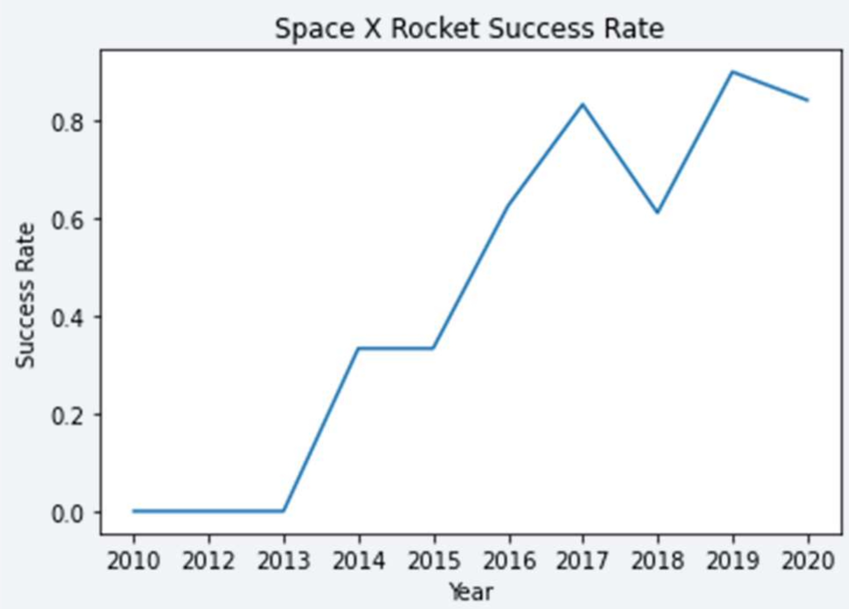


Payload vs. Orbit Type

The weight of the payloads can have a great influence on the success rate of the launches in certain orbits. For example, heavier payloads improve the success rate for the LEO orbit. We have also observed that decreasing the payload weight for a GTO orbit improves the success of a launch.

Launch Success Yearly Trend

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



All Launch Site Names

SQL Query:

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

Results :

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

Explanation:

The DISTINCT keyword in the query allows to remove duplicate LAUNCH_SITE.

Launch Site Names Begin with 'CCA'

SQL Query :

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

Explanation:

The WHERE clause followed by the LIKE clause filters launch sites that contain the substring CCA. LIMIT 5 shows 5 records from the filter.

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

Explanation :

This query returns the sum of all payload masses where the customer is NASA (CRS).

Average Payload Mass by F9 v1.1

SQL Query:

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

Results:

AVG("PAYLOAD_MASS__KG_")
2534.6666666666665

Explanation :

This query returns the average of all payload masses where the booster version contains the substring F9 v1.1.

First Successful Ground Landing Date

SQL Query:

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

Results:

MIN("DATE")
01-05-2017

Explanation :

With this query, we select the oldest successful landing.
The WHERE clause filters the dataset in order to keep only records where the landing was successful. With the MIN function, we select the record with the oldest date.

Successful Drone Ship Landing with Payload between 4000 and 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

Explanation :

This query returns the booster version where the landing was successful and the payload mass is between 4000 and 6000 kg. The WHERE and AND clauses filter the dataset.

Total Number of Successful and Failure Mission Outcomes

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

Explanation :

With the first SELECT, we show the subqueries that return results. The first subquery counts the successful mission. The second subquery counts the unsuccessful mission. The WHERE clause followed by the LIKE clause filters the mission outcome. The COUNT function counts records filtered.

Boosters Carried Maximum Payload

SQL Query:

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

Explanation :

We used a subquery to filter data by returning only the heaviest payload mass with the MAX function. The main query uses subquery results and returns a unique booster version (SELECT DISTINCT) with the heaviest payload mass.

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

2015 Launch Records

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

Explanation :

This query returns month, booster version, and launch site where the landing was unsuccessful and the landing date took place in 2015. Substr function process date in order to take month or year. Substr(DATE, 4, 2) shows month. Substr(DATE,7, 4) shows year.

Rank Landing Outcomes Between 2010-06-04 and 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

Explanation :

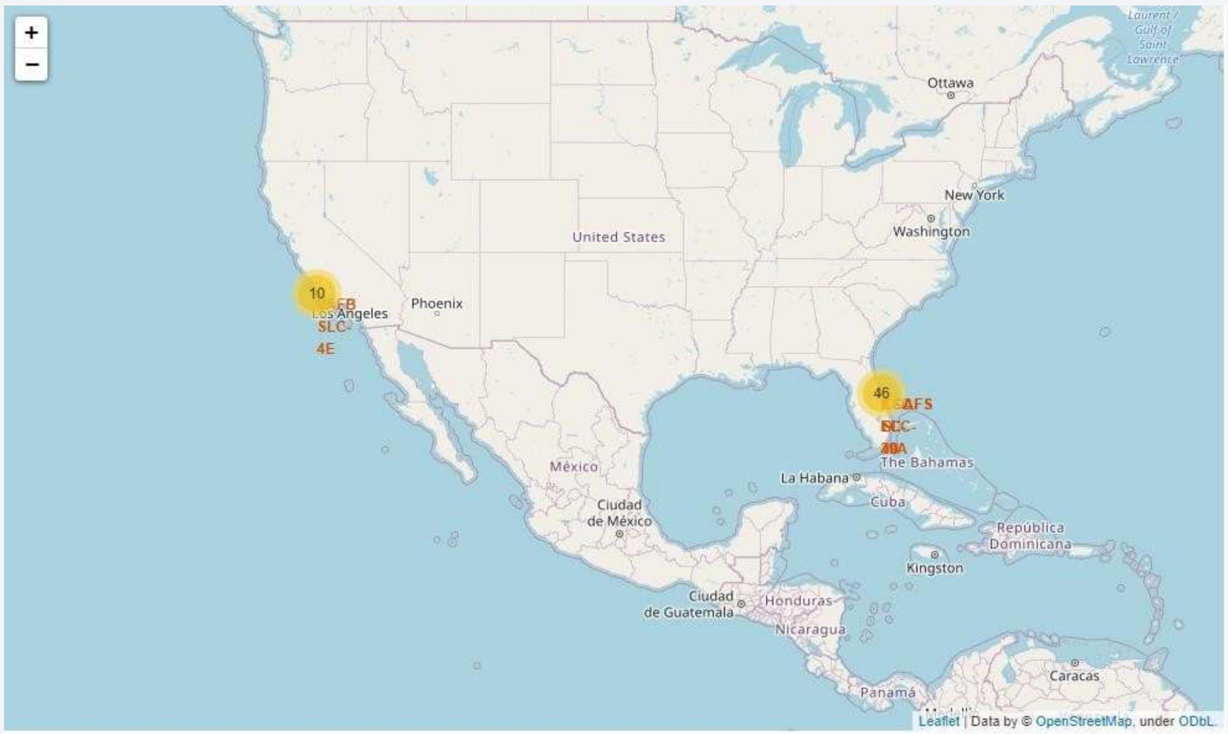
This query returns landing outcomes and their count where the mission was successful and the date is between 04/06/2010 and 20/03/2017. The GROUP BY clause groups results by landing outcome and the ORDER BY COUNT DESC shows results in decreasing order.

Section 4

Launch Sites Proximities Analysis

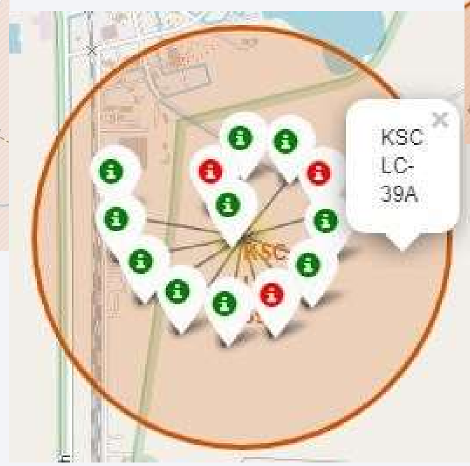
Folium map - Ground stations

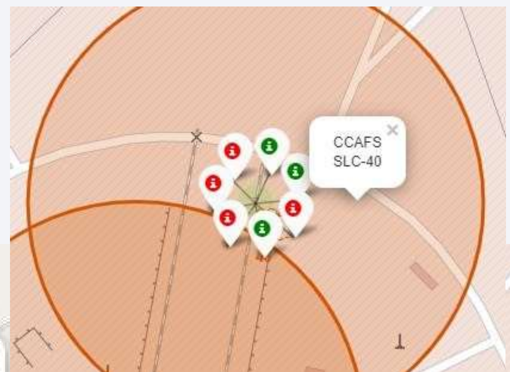
We observe that Space X launch sites are located on the coast of the United States

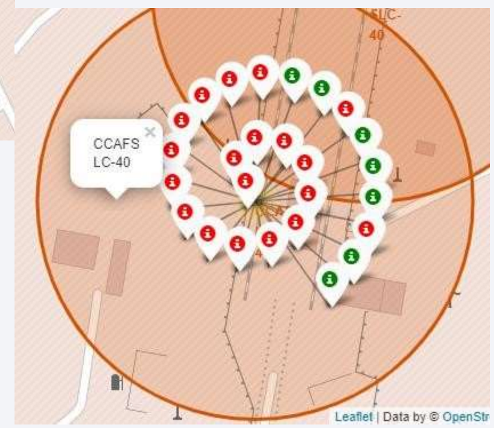


Folium map - Color Labeled Markers

The green marker represents successful launches. The red marker represents unsuccessful launches. We note that KSCLC-39A has a higher launch success rate.

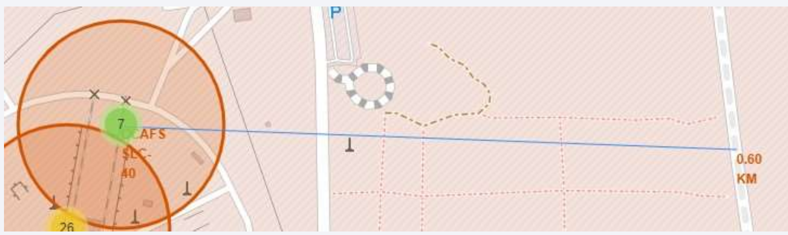


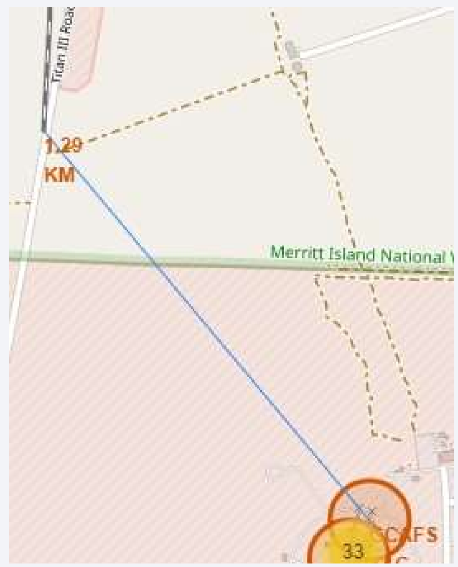




Folium Map - Distances between CCAFS SLC-40 and its proximities

Is CCAFS SLC-40 in close proximity to railways? Yes
Is CCAFS SLC-40 in close proximity to highways? Yes
Is CCAFS SLC-40 in close proximity to the coastline? Yes
Does CCAFS SLC-40 keep a certain distance away from cities? No



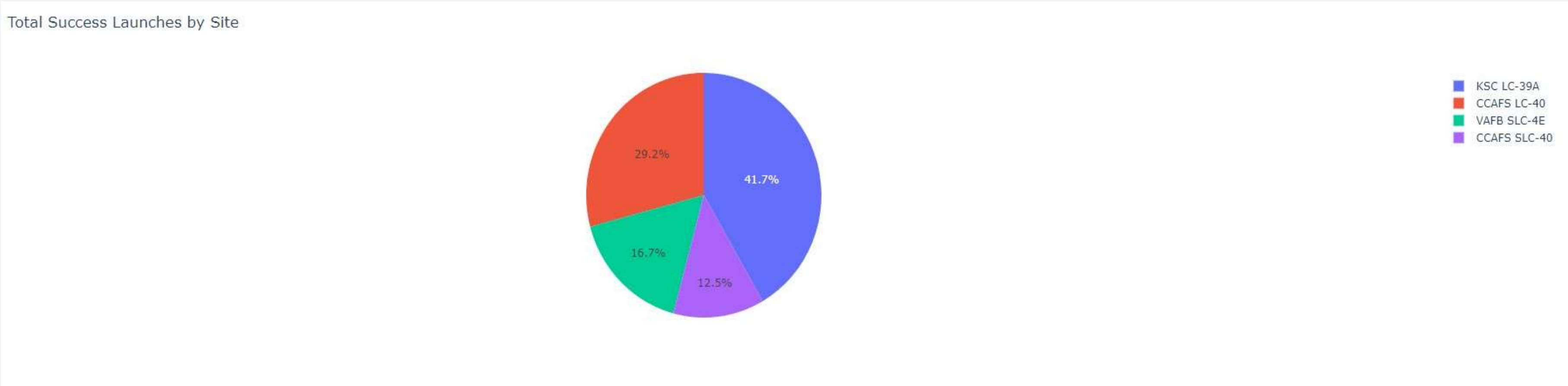


Section 5

Build a Dashboard with Plotly Dash

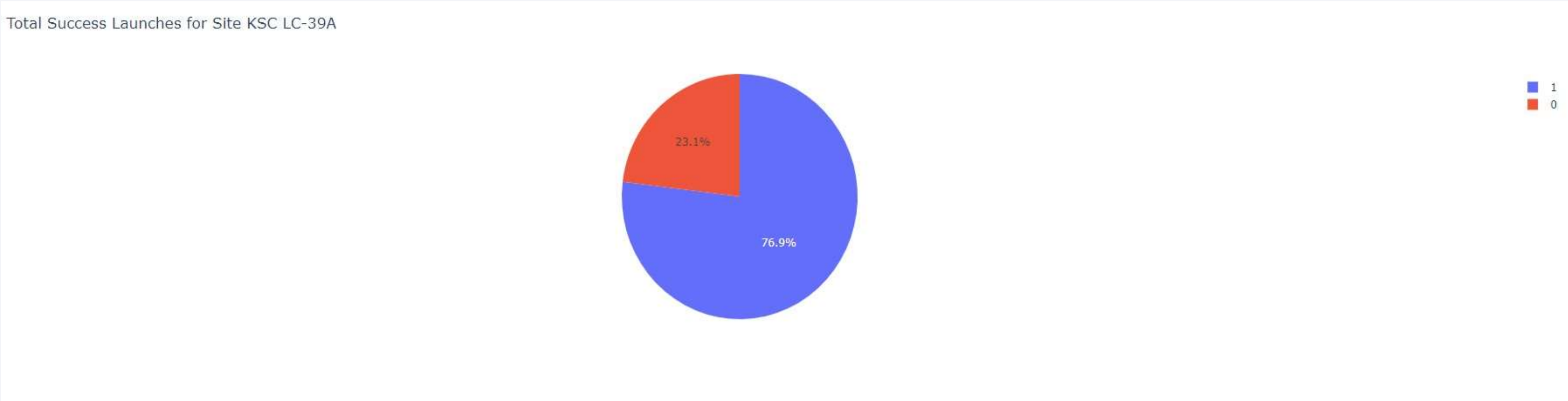
Dashboard - Total success by Site

We observe that KSC LC-39A has the best success rate of launches.



Dashboard - Total success launches for Site KSC LC-39A

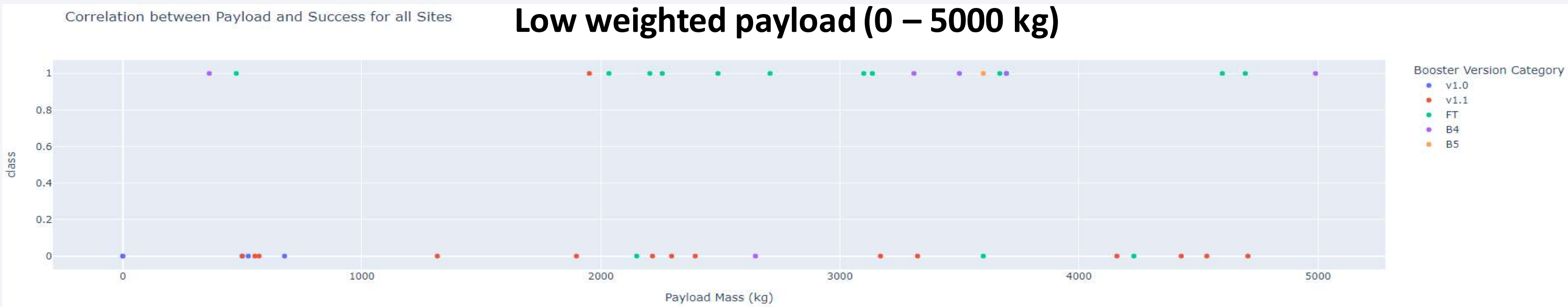
We observe that KSC LC-39A has achieved a 76.9% success rate while getting a 23.1% failure rate.



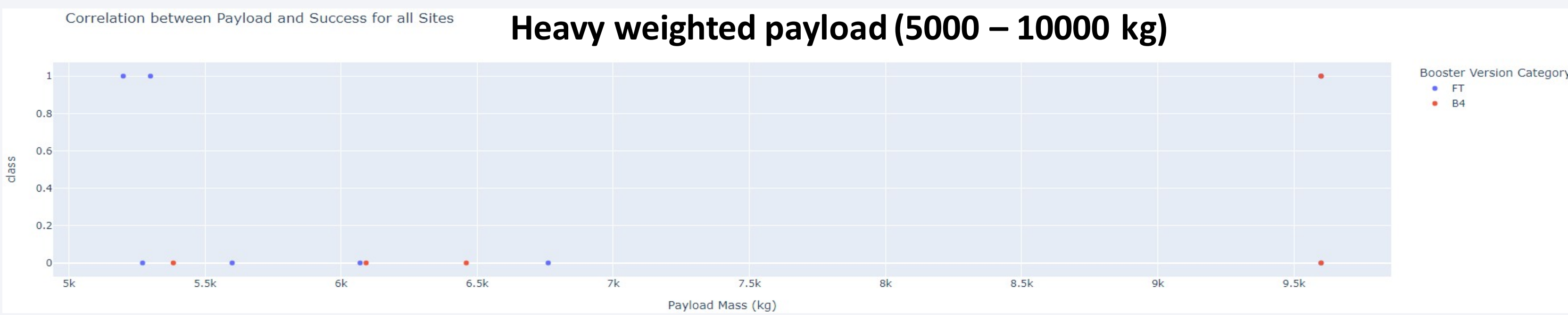
Dashboard - Payload mass vs Outcome for all sites with diff. payload mass selected

Low weighted payloads have a better success rate than the heavy weighted payloads.

Low weighted payload (0 – 5000 kg)



Heavy weighted payload (5000 – 10000 kg)



Section 6

Predictive Analysis (Classification)

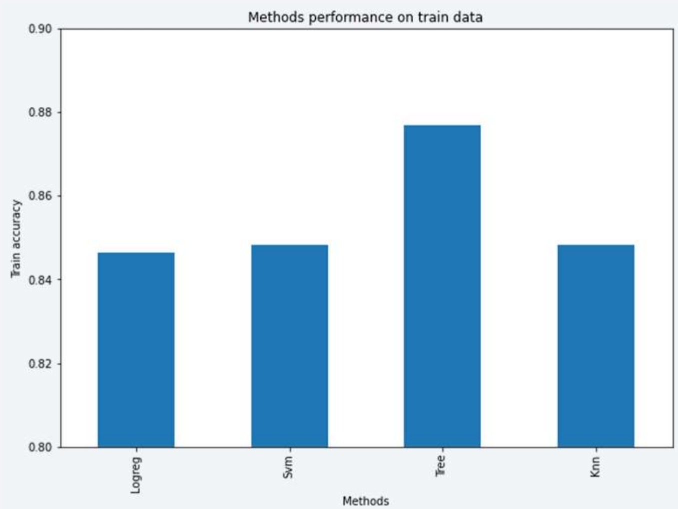
Classification Accuracy

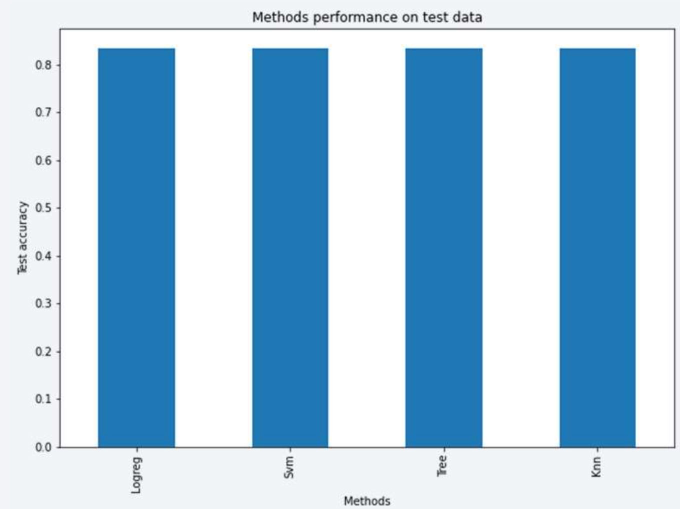
For accuracy test, all methods performed similar. We could get more test data to decide between them. But if we really need to choose one right now, we would take the decision tree.

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'}
```

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

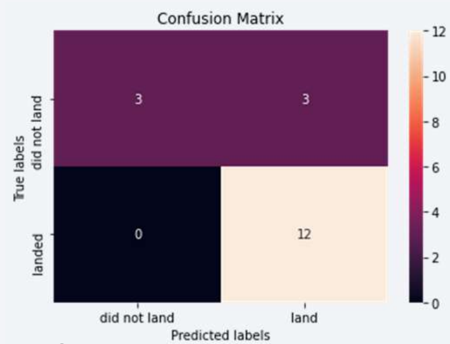




Confusion Matrix

Logistic regression

Decision Tree



kNN

SVM

We observe that the test accuracies are all equal, and the confusion matrices are also identical. The main problem of these models is false positives.

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

Conclusions

- The success of a mission can be explained by several factors such as the launch site, the orbit and especially the number of previous launches. Moreover, we can assume that there has been a gain in knowledge between launches that allowed us to go from a launch failure to a success.
- Depending on the orbits, the payload mass can be a criterion to take into account for the success of a mission. Some orbits require a light or heavy payload mass. But generally low weighted payloads perform better than heavy-weighted payloads.
- The orbits with the best success rates are GEO, HEO, SSO, and ES-L1.
- With the current data, we cannot get an insight into why some launch sites are better than others (KSC LC-39A is the best launch site). To get an answer to this problem, we could obtain atmospheric or other relevant data and perform analysis over the same to understand and implement the observation to other launch sites to increase the number of successful launches.
- For the current data, we choose the Decision Tree Algorithm as the best model even if the test accuracy between all the models used is identical. We choose Decision Tree Algorithm because it has better training accuracy.

Thank you!