IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Malgorzata H>
<20.03.2023>

Outline

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

Executive Summary

- Summary of methodologies
  - Data collecting with API and WebScraping
  - Data Wrangling
  - Exploratory Data Analysis (EDA) and Data Visualisation
  - Interactive Visual Analytics
  - Predictive Analysis

- Summary of all results
  - Data Analysis results
  - Interactive maps and dashboard
  - Predictive model

Introduction

The project helped to predict if the Falcon 9 first stage will land successfully.

SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

Therefore **if we can determine if the first stage will land, we can determine the cost of a launch.**

This information can be used to bid against SpaceX for a rocket launch.

Problem to solve:

What are the main features that influence on succesfull/fail landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
    - via RestAPi
    - via WebScraping
- Perform data wrangling
    - Describe how data was processed
- 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

REST API

Data has been gathered from an API, specifically the SpaceX REST API. This API gave us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.

WEB SCRAPING

Data has been collected also via HTTP protocol. Popular data source for obtaining Falcon 9 Launch data is web scraping related Wiki pages. Using the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records.

Web scraping Falcon 9 and Falcon Heavy Launches Records from Wikipedia

Data Collection – SpaceX API

Requesting rocket launch data from SpaceX API

Using json_normalize method to convert the json result into a dataframe

Using the API again to get information about the launches using the IDs given for each launch. Specifically by using columns `rocket`, `payloads`, `launchpad`, and `cores`.

```
In [62]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [63]: response = requests.get(spacex_url)
```

```
In [67]: # Use json_normalize meethod to convert the json result into a dataframe
         data = pd.json_normalize(response.json())
```

```
In [69]: # Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
         data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

         # We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have multip
         data = data[data['cores'].map(len)==1]
         data = data[data['payloads'].map(len)==1]

         # Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature.
         data['cores'] = data['cores'].map(lambda x : x[0])
         data['payloads'] = data['payloads'].map(lambda x : x[0])

         # We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
         data['date'] = pd.to_datetime(data['date_utc']).dt.date

         # Using the date we will restrict the dates of the launches
         data = data[data['date'] <= datetime.date(2020, 11, 13)]
```


Repository

Data Collection - Scraping

HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

↓

Creating a BeautifulSoup object from the HTML response

↓

Extracting all column/variable names from the HTML table header

```
In [7]: # use requests.get() method with the provided static_url
        # assign the response to a object
        response = requests.get(static_url)
```

```
In [12]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
         soup = BeautifulSoup(response.text, 'html.parser')
```

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plai
nrowheaders collapsible")):
    # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding t
o launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
```

[Repository](#)

Data Wrangling

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

Calculating the number of launches on each site

Calculating the number of launches on each site

Calculate the number and occurence of mission outcome per orbit type

Creating a landing outcome label from Outcome column

```
In [6]: # Apply value_counts() on column LaunchSite
        df['LaunchSite'].value_counts()
```

```
In [7]: # Apply value_counts on Orbit column
        df['Orbit'].value_counts()
```

```
In [9]: # landing_outcomes = values on Outcome column

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

We create a set of outcomes where the second stage did not land successfully:

```
In [11]: bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
         bad_outcomes
```

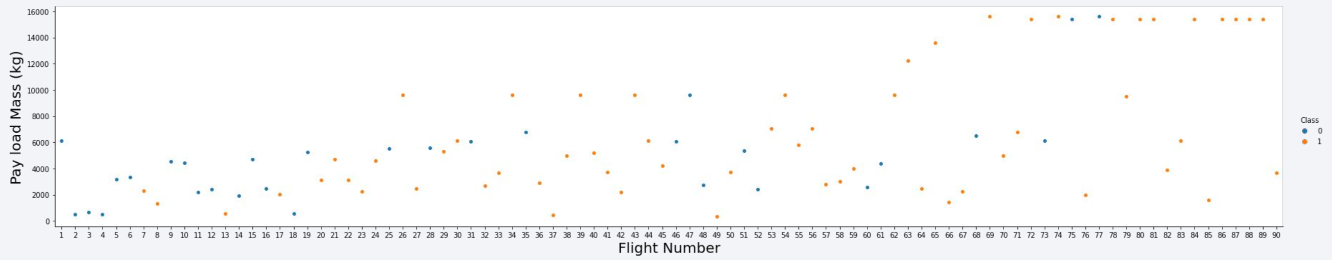
```
In [14]: # landing_class = 0 if bad_outcome
         landing_class = []
         for key,value in df["Outcome"].items():
             if value in bad_outcomes:
                 landing_class.append(0)
             else:
                 landing_class.append(1)
         # landing_class = 1 otherwise
```


Repository

EDA with Data Visualization

SCATTER GRAPHS

- Flight Number vs Payload Mass
- Fligh Number vs Launch Site
- Payload vs Launch Site
- Orbit vs Flight Number
- Payload vs Orbit Type
- Orbit vs Payload Mass



To show relation between features - correlation.

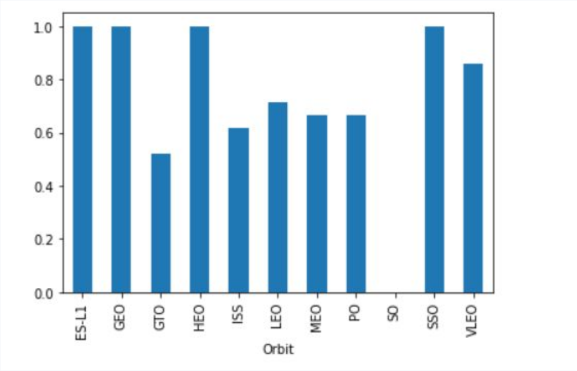
Repository

EDA with Data Visualization

BAR GRAPH

- Success rate vs Orbit

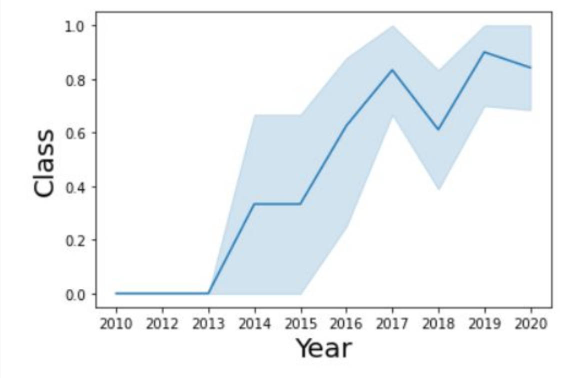
To find which orbits have high success rate.



LINE GRAPH

- Success rate vs Orbit

To observe that the sucess rate since 2013 kept increasing till 2020.



[Repository](#)

EDA with SQL

- Displaying the names of the launch sites.
- Displaying 5 records where launch sites begin with the string 'CCA'.
- Displaying the total payload mass carried by booster launched by NASA (CRS).
- Displaying the average payload mass carried by booster version F9 v1.1.
- Listing the date when the first successful landing outcome in ground pad was achieved.
- Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
- Listing the total number of successful and failure mission outcomes.
- Listing the names of the booster_versions which have carried the maximum payload mass.
- Listing the failed landing_outcomes in drone ship, their booster versions, and launch sites names 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.

Repository

Build an Interactive Map with Folium

Folium map object is a interactive map to visualise the launch data. By using latitude and longtude coordinates at each launch site and added a circle marker with a label of the name of the launch site:

- Red circle for each launch site
- Markers for all launch records. If a launch was successful `(class=1)`, then we use a green marker and if a launch was failed, we use a red marker `(class=0)`
- Marker clusters to simplify a map containing many markers having the same coordinate.

Those objects have been created to show each launch site, surrounding and the number of succesful and failure landing.

Repository

Build a Dashboard with Plotly Dash

- Dashboard has dropdown, pie chart, rangeslider and scatter plot components
  - 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)*.
  - Scatter chart shows the relationship between two variables, in particular Success vs Payload Mass *(plotly.express.scatter)*.

Predictive Analysis (Classification)

Building the Model

- Load the dataset into NumPy and Pandas
- Transform the data and then split into training and test datasets
- Decide which type of ML to use
- set the parameters and algorithms to GridSearchCV and fit it to dataset.

Evaluating the Model

- Check the accuracy for each model
- Get tuned hyperparameters for each type of algorithms.
- plot the confusion matrix.

Improving the Model

- Use Feature Engineering and Algorithm Tuning

Find the Best Model

- The model with the best accuracy score will be the best performing model.

From:
https://github.com/farishelmi17/SpaceX/blob/main/spacex_dash_app.py

16

[Repository]

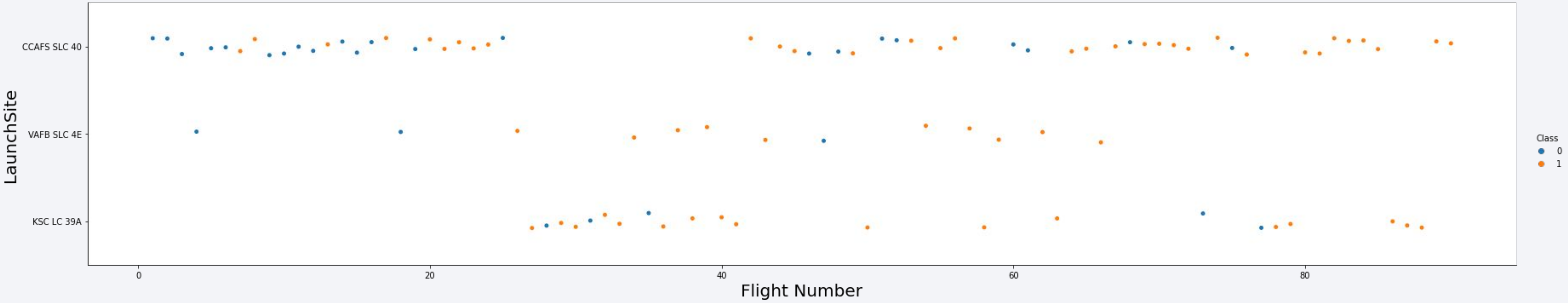
Results

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

Section 2

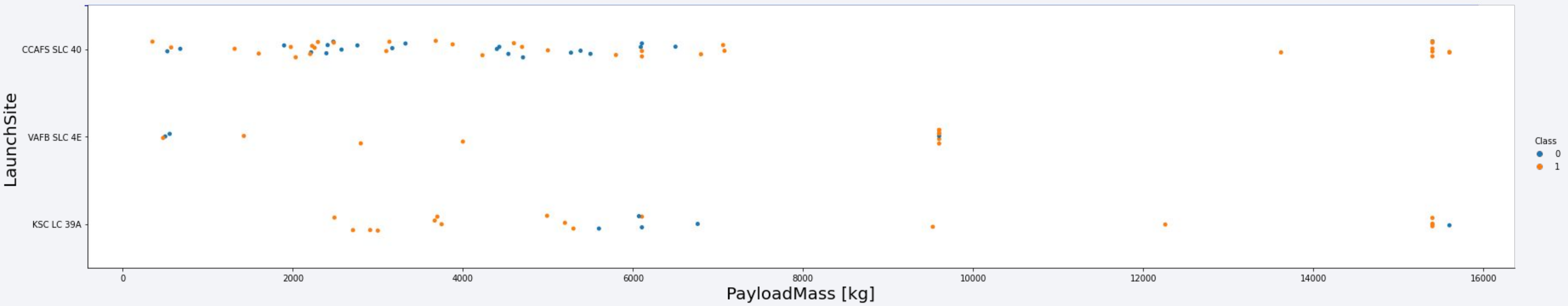
Insights drawn from EDA

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



Now we observe Payload Vs. Launch Site scatter point chart - for the VAFB-SLC launchsite there are no rockets launched for heavy payload mass(greater than 10000).

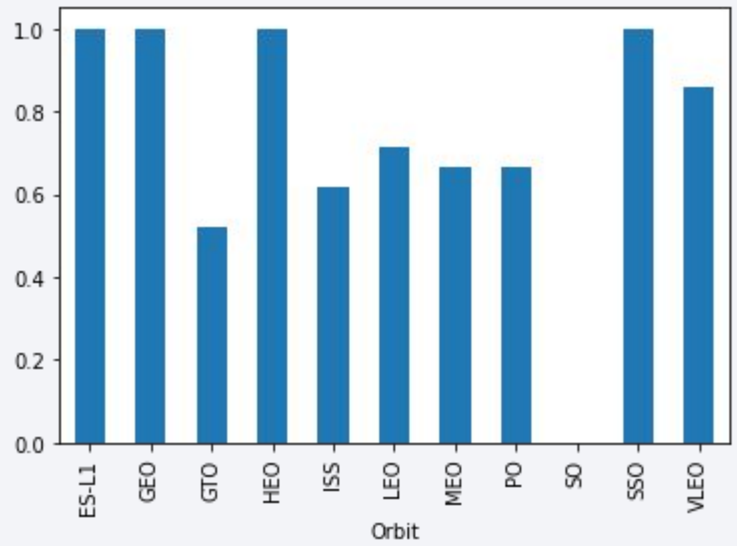
Success Rate vs. Orbit Type

That plot shows success rate for different Orbit.

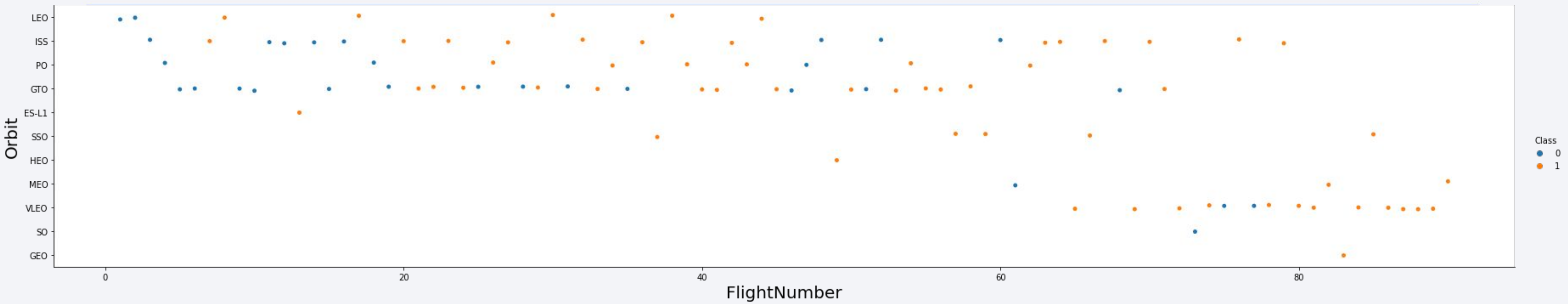
ESL-L1, GEO, HEO and SSO

has 100 % success in landing.

SO is the only orbit that has alway failure landing.

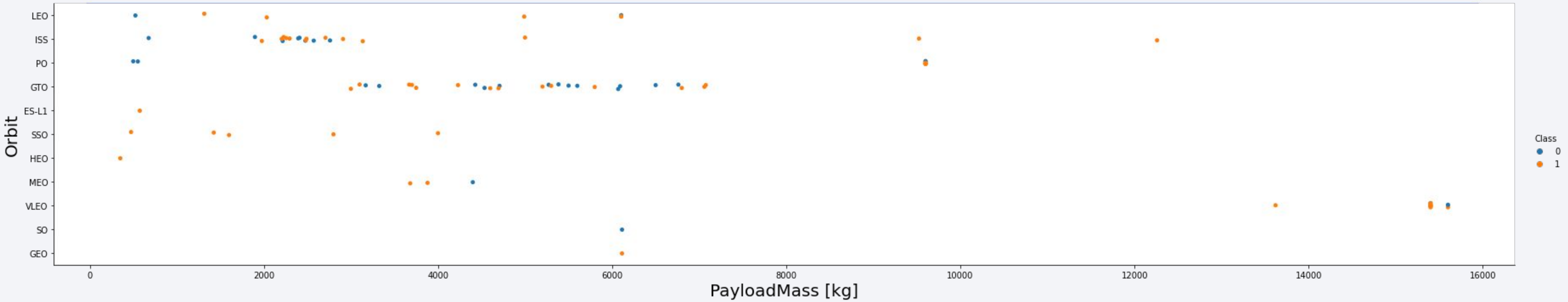


Flight Number vs. Orbit Type



The LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type

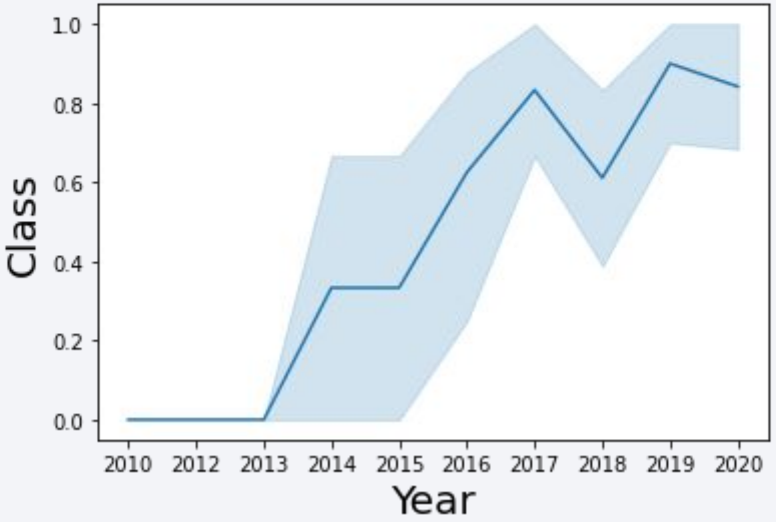


With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend

The sucess rate since 2013 kept increasing till 2020



All Launch Site Names

We use **DISTNICT** to show only unique values of launch Site Names, it allows to remove duplicates.

```
[8]: %sql select distinct (Launch_Site) from spacextbl

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

[8]:

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

Launch Site Names Begin with 'CCA'

We use the query to display 5 records where launch sites begin with the string 'CCA'.

```
[11]: %sql select * from spacextbl where Launch_Site like 'CCA%' limit 5

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

[11]:

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

Total Payload Mass

- We used query to display the total payload mass carried by boosters launched by NASA (CRS)

```
[14]: %sql select sum(PAYLOAD_MASS__KG_) from spacextbl where customer like '%CRS%'

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

[14]:

sum(PAYLOAD_MASS__KG_)
48213

Average Payload Mass by F9 v1.1

We used the query to display average payload mass carried by booster version F9 v1.1

```
[17]: %sql select avg(PAYLOAD_MASS__KG_) from spacextbl where booster_version like '%F9%1.1'

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

[17]:

avg(PAYLOAD_MASS__KG_)
4658.111111111111

First Successful Ground Landing Date

We used the query to list the date when the first succesful landing outcome in ground pad was acheived.

```
[19]: %sql select min(date) from spacextbl limit 5;

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

[19]:

min(date)
01-03-2013

Successful Drone Ship Landing with Payload between 4000 and 6000

We used the query to list the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
[28]: %sql select distinct(Booster_Version) from spacextbl where mission_outcome like '%success%' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ <6000;
```

[28]:

Booster_Version
F9 v1.1
F9 v1.1 B1011
F9 v1.1 B1014
F9 v1.1 B1016
F9 FT B1020
F9 FT B1022
F9 FT B1026
F9 FT B1030
F9 FT B1021.2
F9 FT B1032.1
F9 B4 B1040.1

F9 FT B1031.2
F9 B4 B1043.1
F9 FT B1032.2
F9 B4 B1040.2
F9 B5 B1046.2
F9 B5 B1047.2
F9 B5B1054
F9 B5 B1048.3
F9 B5 B1051.2
F9 B5B1060.1
F9 B5 B1058.2

Total Number of Successful and Failure Mission Outcomes

We used the query to list the number of success landing

```
[23]: %sql select count(*)from spacextbl where mission_outcome like '%success%';

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

[23]:

count(*)
100

We used the query to list the number of fail landing

```
[24]: %sql select count(*)from spacextbl where mission_outcome like '%fail%';

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

[24]:

count(*)
1

Boosters Carried Maximum Payload

- We used the query to list the names of the booster which have carried the maximum payload mass

```
[17]: %sql select "Booster_version" from spacextbl where PAYLOAD_MASS__KG_ IN (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

2015 Launch Records

We used the query to 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.

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

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

[59]:

month	Booster_Version	Landing _Outcome
01	F9 v1.1 B1012	Failure (drone ship)
04	F9 v1.1 B1015	Failure (drone ship)

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

We used the query to rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
[81]: %%sql
select "Landing _Outcome", count ("Landing _Outcome")
from SPACEXTBL
where date > '04-06-2010'
and date > '20-03-2017'
group by "Landing _Outcome"
order by count("Landing _Outcome") desc
```

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

[81]:

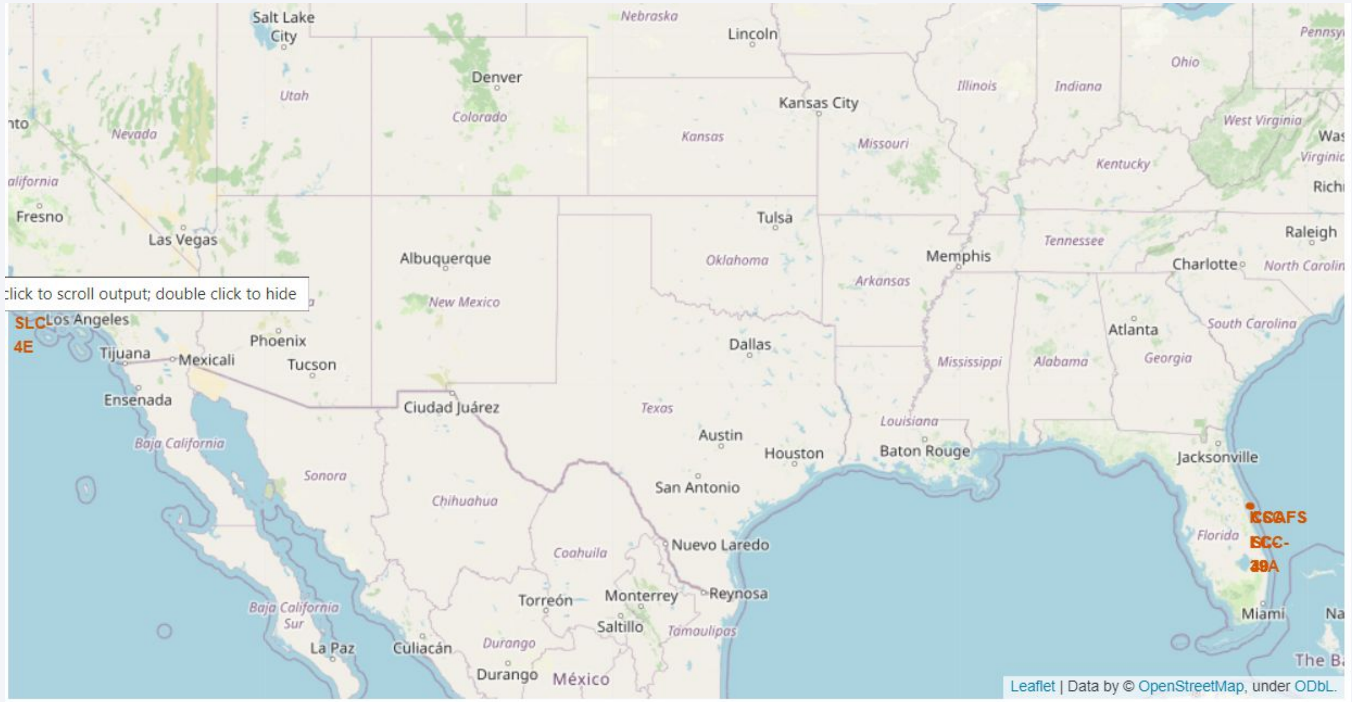
Landing _Outcome	count ("Landing _Outcome")
Success	14
No attempt	7
Success (drone ship)	6
Uncontrolled (ocean)	2
Controlled (ocean)	2
Success (ground pad)	1
Precluded (drone ship)	1

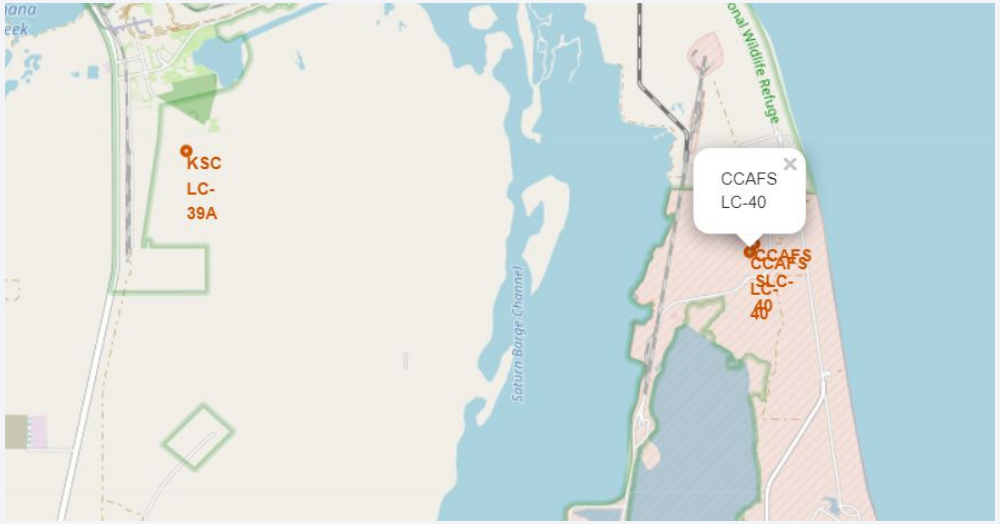
Section 3

Launch Sites Proximities Analysis

Map with all launch sites

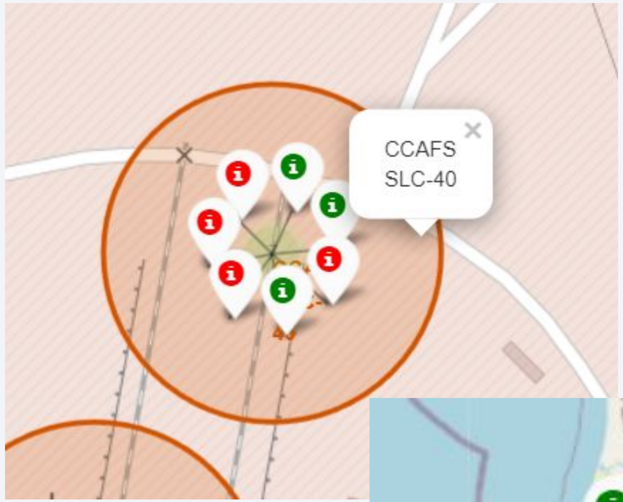
It’s site’s location map created by using site's latitude and longitude coordinates. Every highlighted circle area has a text label on a specific coordinate.

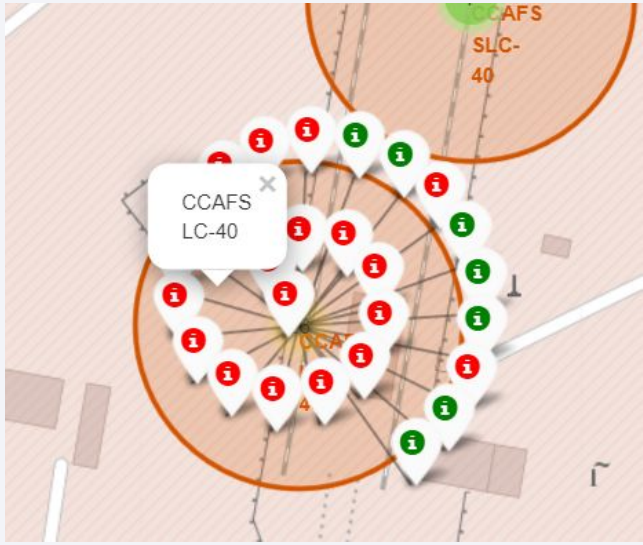


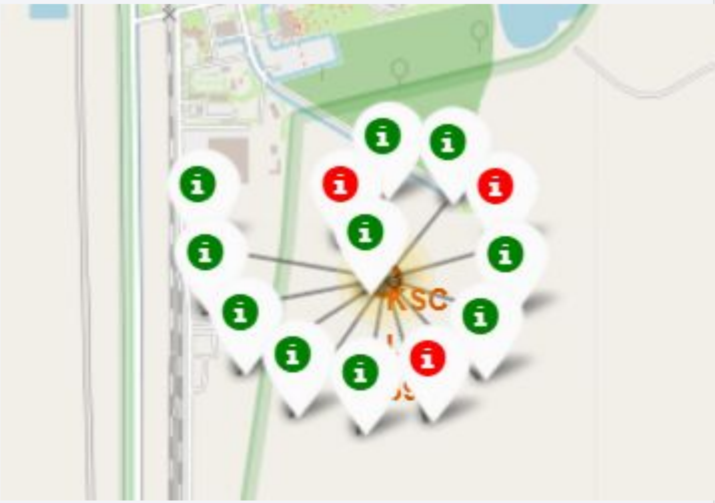


Map with successful/failed launches

Launch outcomes for each site had been added to the map. That helps to see which sites have high success rates. Green- successful, red- failed. KSC LC 39-A has the highest value of successes.

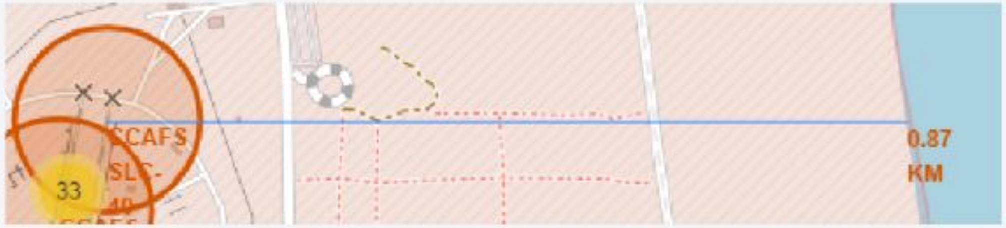


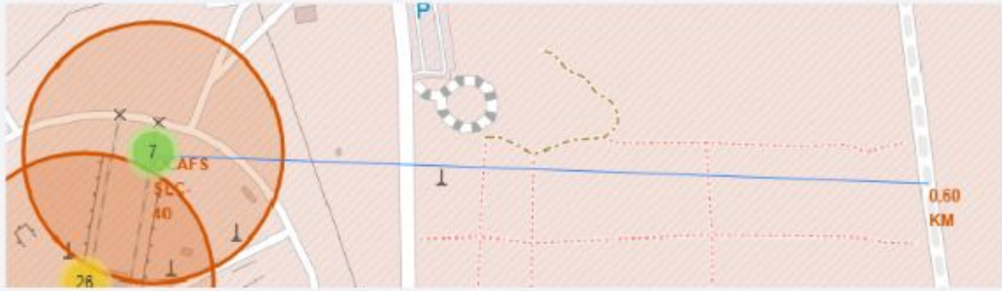




Map with calculation the distance

Map helps us to get coordinate for a mouse over a point on the map. By exploring the map, we can easily find the coordinates of any points of interests (such as railway).



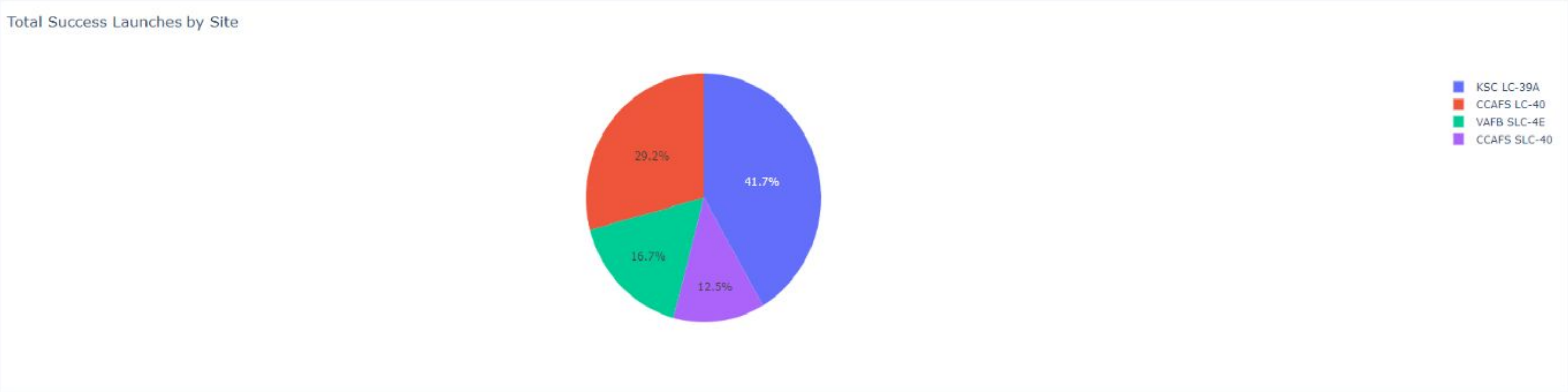


Section 4

Build a Dashboard with Plotly Dash

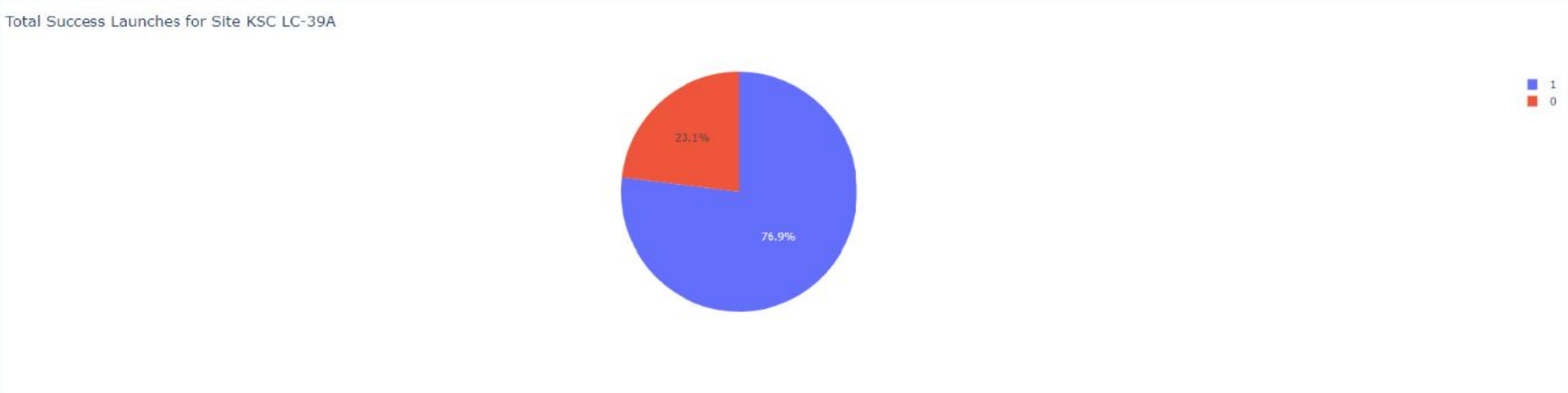
Total success by site

- KSC LC-39A has the highest rate - 41,7 %



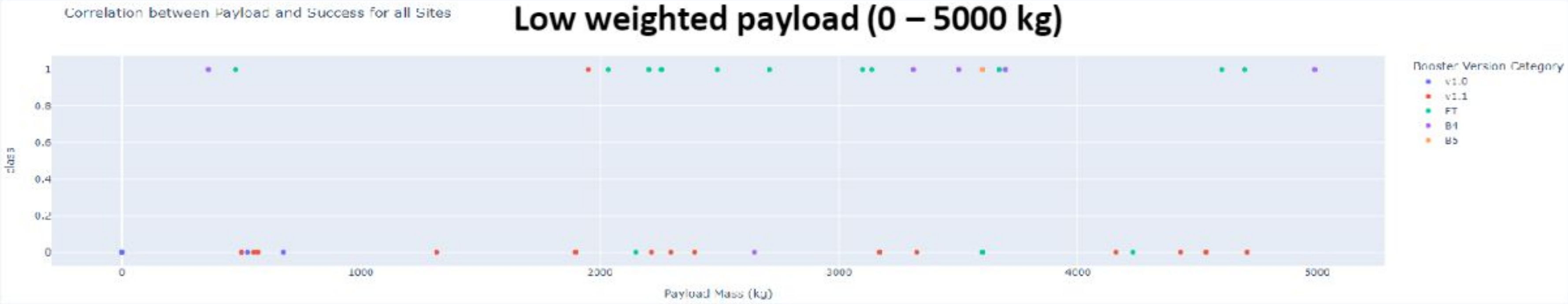
KSC LC-39A rates

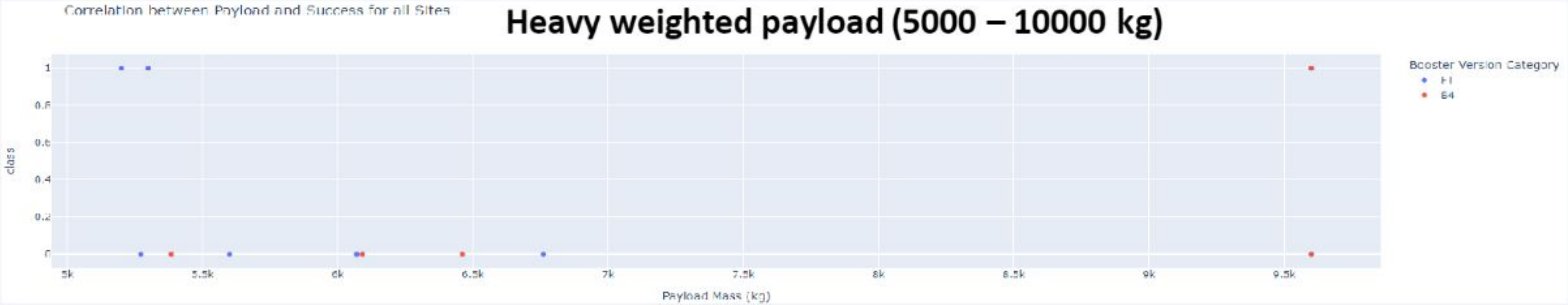
Total Success for Site KSC LC-39A is 75,9 %. The fail is 23,1 %



Payload vs Launch Outcome Scatter Plot

We can see that all the success rate for low weighted payload is higher than heavy weighted ones.



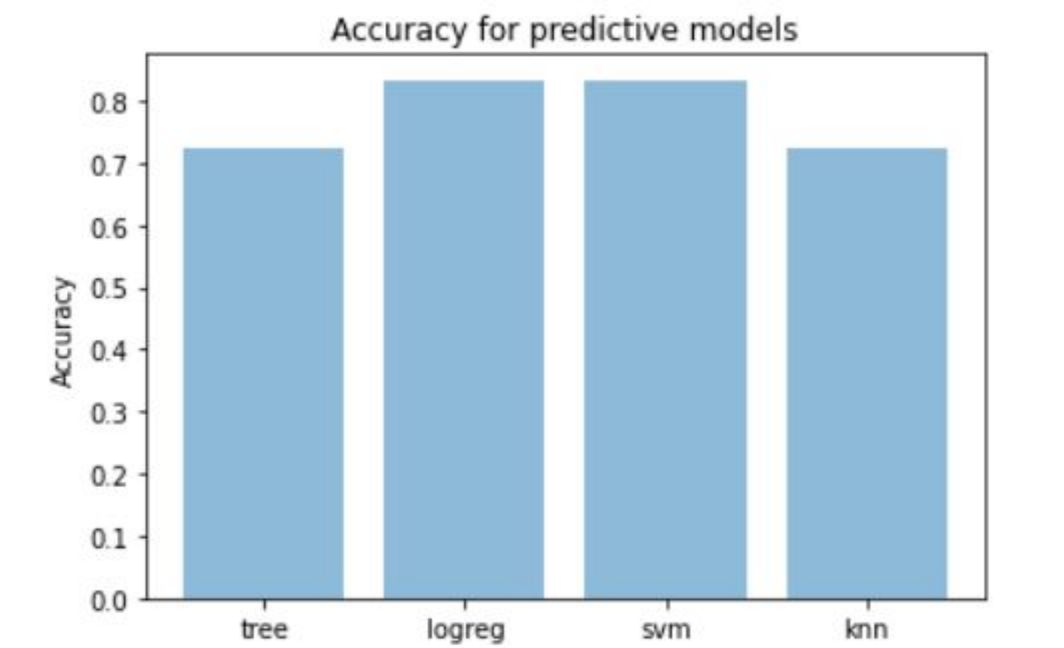


Section 5

Predictive Analysis (Classification)

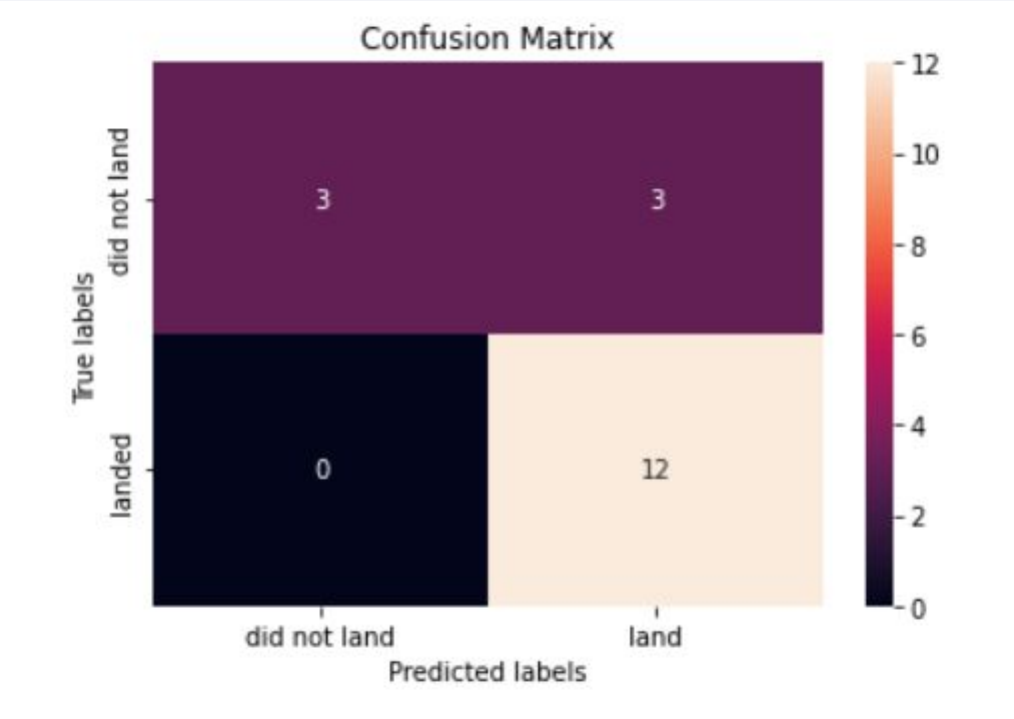
Classification Accuracy

	algorithm	score
0	tree	[0.7222222222222222]
1	logreg	[0.8333333333333334]
2	svm	[0.8333333333333334]
3	knn	[0.7222222222222222]

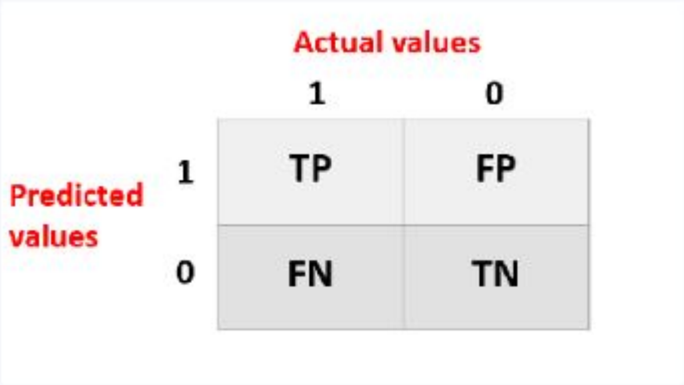


Probably all of the models should has the same accuracy (score) for all the models. But they do not.

Confusion Matrix



Confusion matrix for the SVM classifier shows that the classifier can distinguish between different classes. The major problem is the false positives.
For example unsuccessful landing marked as success.



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. Indeed, we can assume that there has been a gain in knowledge between launches that allowed to go from a launch failure to a success.
- The orbits with the best success rates are GEO, HEO, SSO, ES-L1.
- 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 the heavy weighted payloads.
- With the current data, we cannot explain 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.
- For this dataset, 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 a better train accuracy.

Thank you!