IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Yarra Prasad
24-July-2023

Outline

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

Executive Summary

Methodologies:

- Data collection from API and Web Scrapping
- Data Wrangling
- Exploratory data analysis (EDA) using SQL, Pandas, Matplotlib and Seaborn
- Interactive Visual Analytics and Dashboard with Folium and Plotly Dash
- Predictive Analysis (Classification)

Executive Summary

Results:

- The best hyperparameters for Logistic regression, SVM, Decision Tree and KNN Classifiers
- The method that perform best using Test Data

Introduction

Here we will 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. Sharing launch success metrics empowers other companies and minimizes risks, fostering collective growth.

Problems that we want Find Answers:

- How can SpaceX further improve the reusability of Falcon 9's first stage?
- How can they enhance the reliability of reused components to ensure safe and successful launches?
- What innovative approaches can be implemented to reduce the cost of launching payloads into space using Falcon 9?
- How can cost efficiencies be achieved without compromising safety and performance?
- How can SpaceX increase the payload capacity of Falcon 9 to accommodate larger and heavier payloads, expanding its potential applications?
- How can SpaceX minimize the turnaround time between Falcon 9 launches?
- How can they streamline processes to increase launch frequency?

Section 1

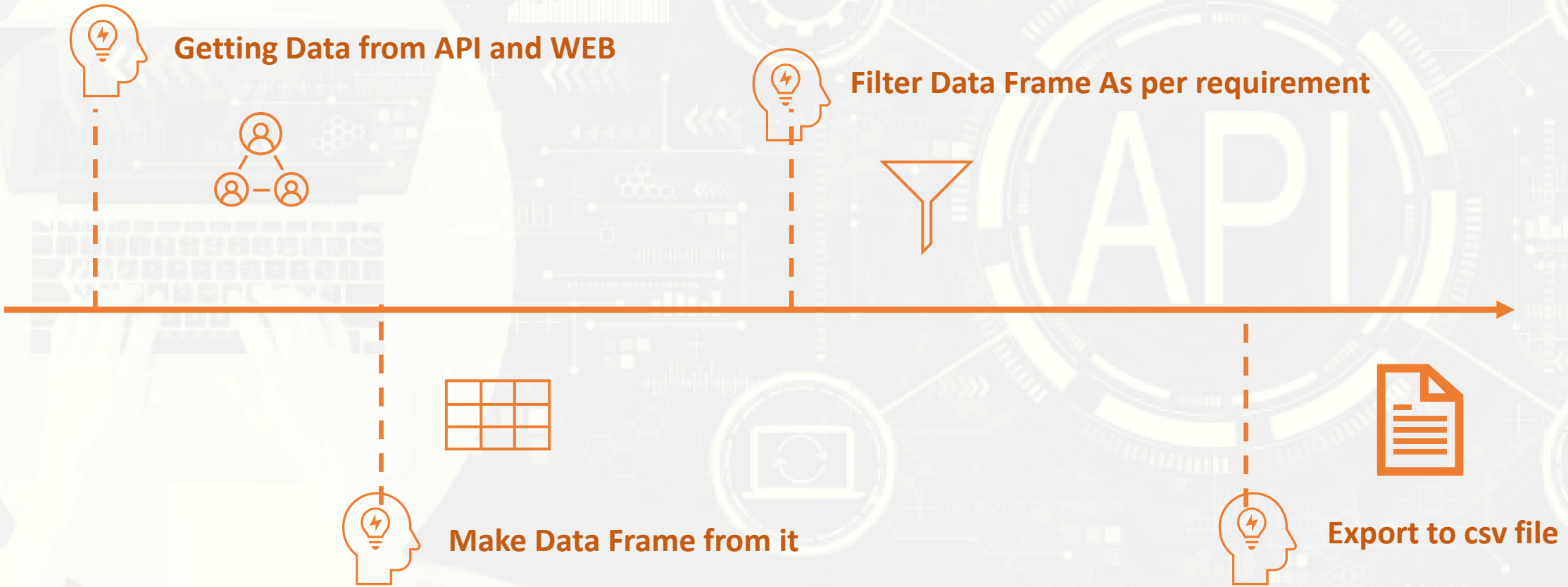
Methodology

Methodology

Data collection methodology:	• Leveraging SpaceX API's and Wikipedia Web scraping for Comprehensive Insights.
Perform data wrangling	• Conducting Exploratory Data Analysis (EDA) to Unearth Patterns and Define Supervised Model Labels.
Perform exploratory data analysis (EDA) using visualization and SQL	• Uncovering Insights through Visualization and SQL Techniques.
Perform interactive visual analytics using Folium and Plotly Dash	• Leveraging Folium and Plotly Dash for Dynamic Data Exploration.
Perform predictive analysis using classification models	• Employing Classification Models for Effective Predictions and Decision-Making.

Data Collection

Github link



Data Collection – SpaceX API

[Github link]

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

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

Getting Response From API

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomai
```

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

Converting response to .Json file

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
```

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

Create Data Frame

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

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

Filter and Export to csv

Data Collection - Scraping

Getting response from HTML

Creating Beautiful Soup objects

Create Data Frame with HTML table

Creating csv file

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_a
response = requests.get(static_url)
```

```
soup = BeautifulSoup(response.content,"html.parser")
html_tables = soup.find_all("table")
```

```
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'] = []
```

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

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	Dece
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt\n	22
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0B0006.1	No attempt	8 Oc

Data Wrangling

The process of cleaning, transforming, and preparing raw data for analysis.

Calculate number of launches at each site

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

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

Calculate number and occurrence of each orbit

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

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

Calculate landing outcome

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

```
df.head(5)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFi
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	Fal
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	Fal

Export to csv

```
df["Class"].mean()
```

```
0.6666666666666666
```

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

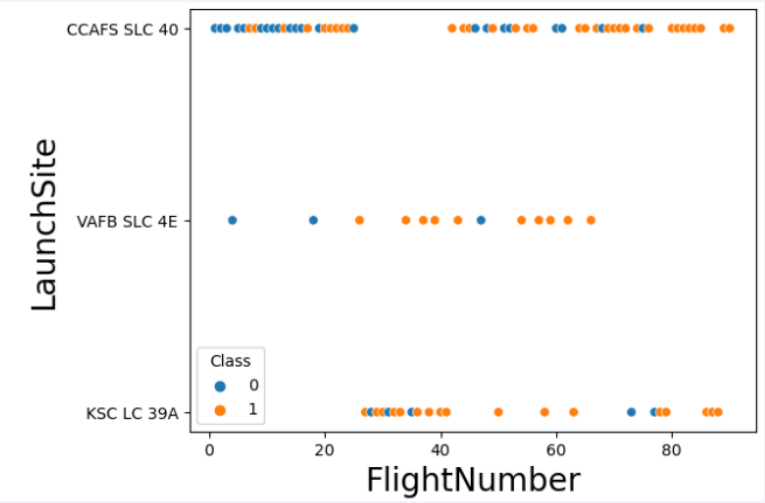
EDA with Data Visualization

Github link

Scatter Plots

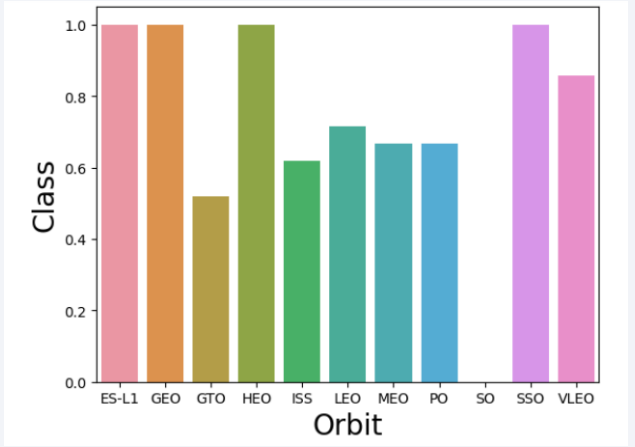
- Payload Vs Flight number
- Flight number Vs Launch Site
- Payload Vs Launch Site
- Payload Vs Orbit type
- Flight number Vs Orbit type

Scatter plots show dependency of attributes on each other

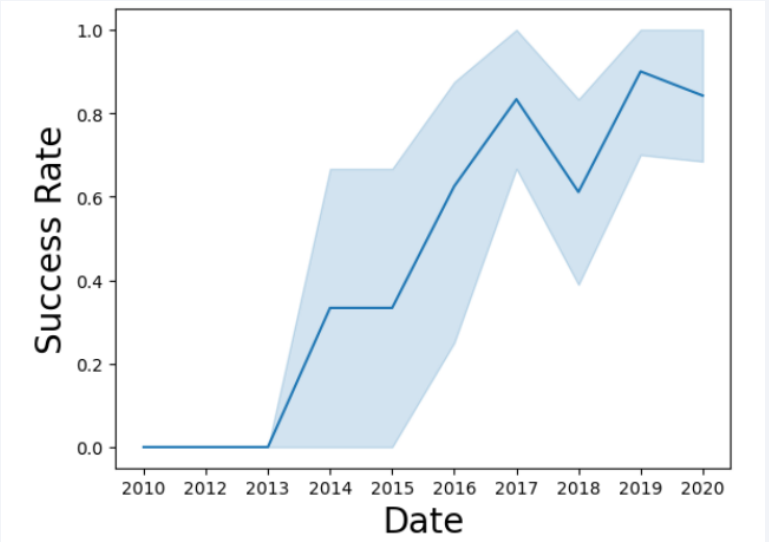


Bar graphs

Using these graphs, We can easily determine which orbit has high probability of success

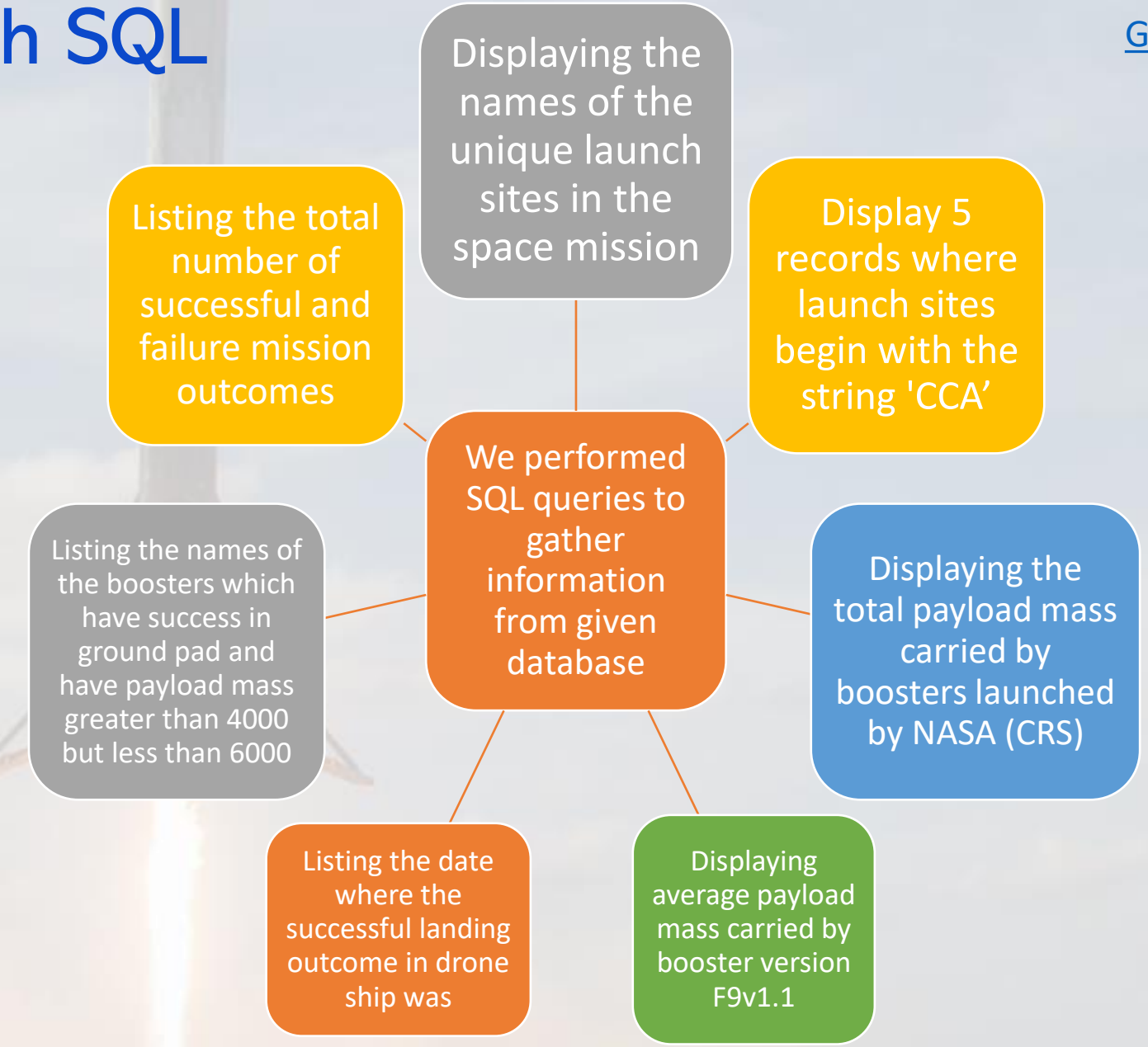


Line graphs



EDA with SQL

Github link



Build an Interactive Map with Folium

Github link

Folium makes it easy to visualize data that's been manipulated in Python on an interactive leaflet map. We use the latitude and longitude coordinates for each launch site and added a Circle Marker around each launch site with a label of the name of the launch site. It is also easy to visualize the number of success and failure for each launch site with Green and Red markers on the map.

Map Object	Code	Result
Map Marker	Folium.marker()	Map object to mark on map
Circle Maker	Folium.circle()	Create circle where marker being placed
Icon Maker	Folium.icon()	Create icon on map
Polyline	Folium.PolyLine()	Create a line between points
Marker Cluster Object	MarkerCluster()	This is good way to simplify map with many markers

Build a Dashboard with Plotly Dash

Live plotly dashboard

Github link

Pie Chart showing the total success for all sites or by certain launch site

- Percentage of success in relation to launch site

Scatter Graph showing the correlation between Payload and Success for all sites or by certain launch site

- It shows the relationship between Success rate and Booster Version Category.

Object	Code	Result
Dash and its components	import dash import dash_html_components as html import dash_core_components as dcc from dash.dependencies import Input, Output	The Dash Core Component library contains a set of higher-level components like sliders, graphs, dropdowns, tables, and more. Dash provides all the available HTML tags as user-friendly Python classes.
Pandas	Import pandas as pd	Fetching data from csv
Plotly	Import plotly.express as px	Interactive plotly library
DropDown	dcc.Dropdown()	Creates dropdown
Ring Slider	dcc.Ringslider()	Creates Ring slider
Pie Chart	px.pie()	Creates pie chart
Scatter Chart	Px.scatter()	Creates scatter plot

Predictive Analysis (Classification)

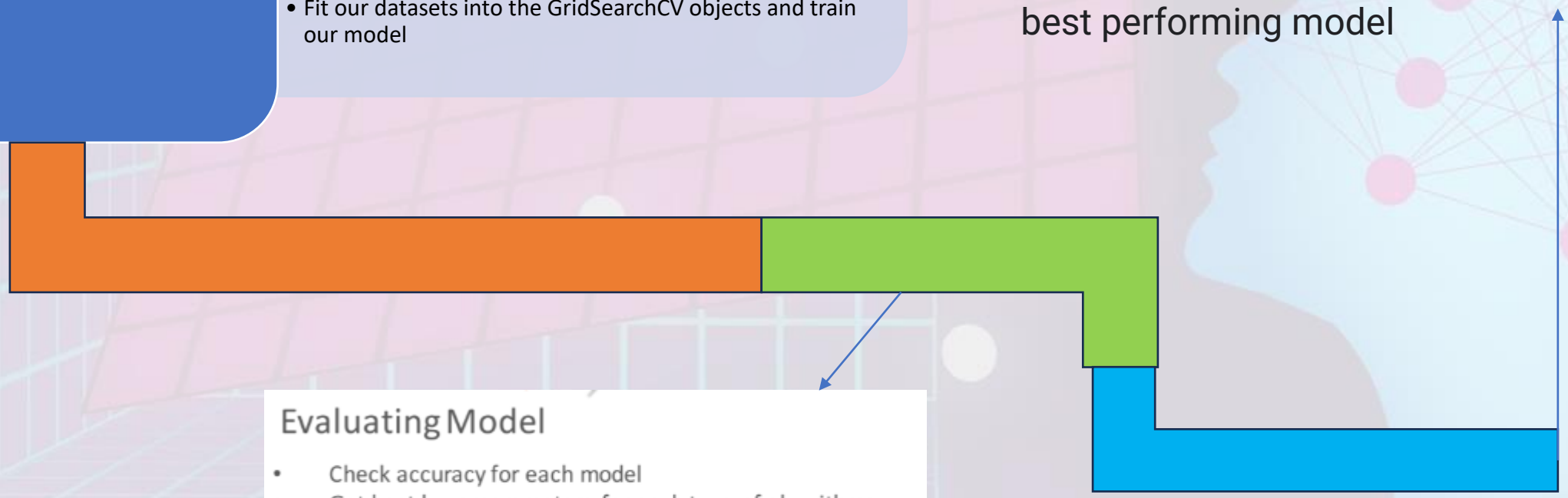
Github link

Building Model

- Load our feature engineered data into dataframe
- Transform it into NumPy arrays
- Standardize and transform data
- Split data into training and test data sets
- Check how many test samples has been created
- List down machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our model

Finding Best Performing Classification Model

- The model with best accuracy score wins the best performing model



Evaluating Model

- Check accuracy for each model
- Get best hyperparameters for each type of algorithms
- Plot Confusion Matrix

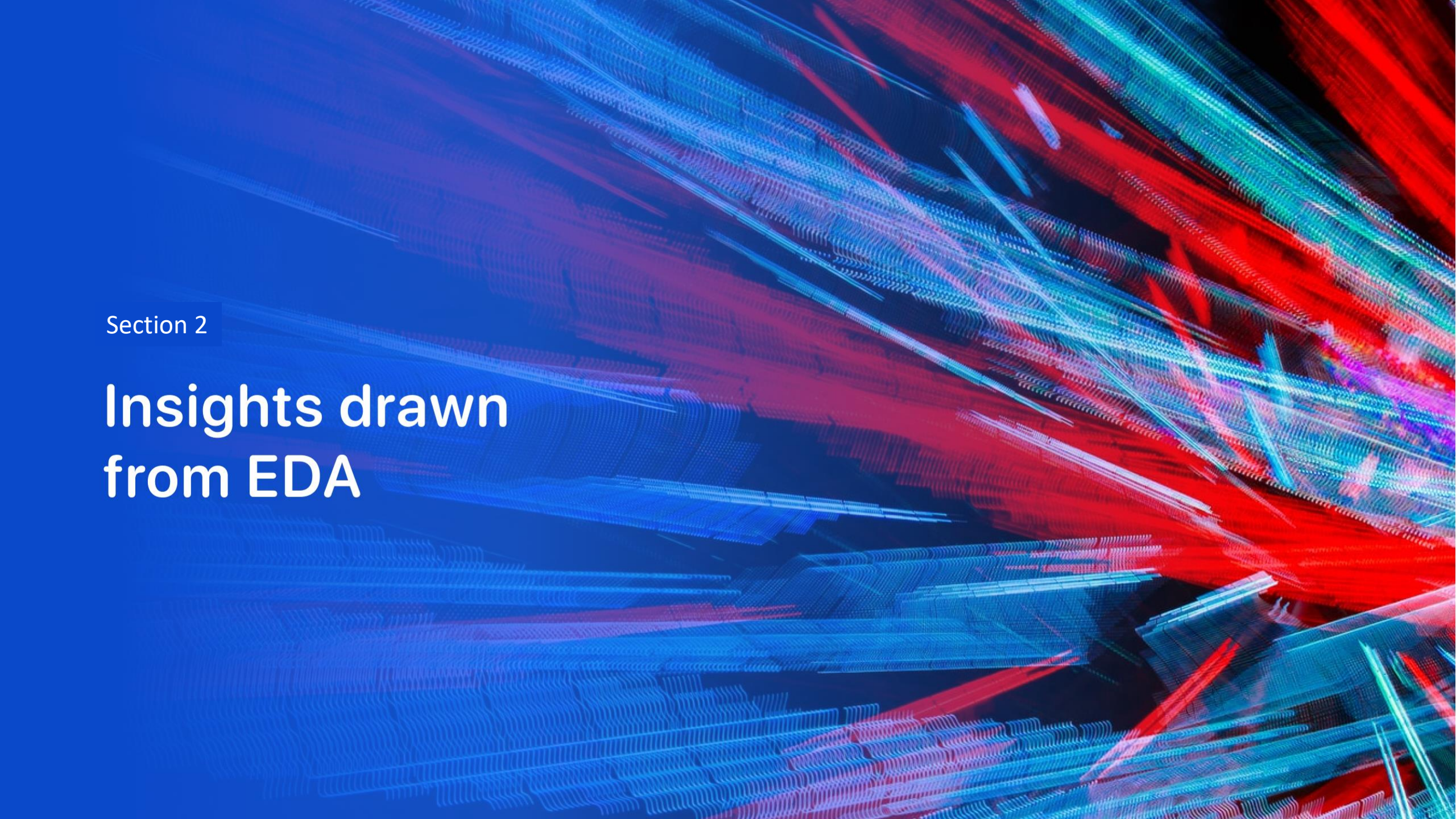
Results

Github link

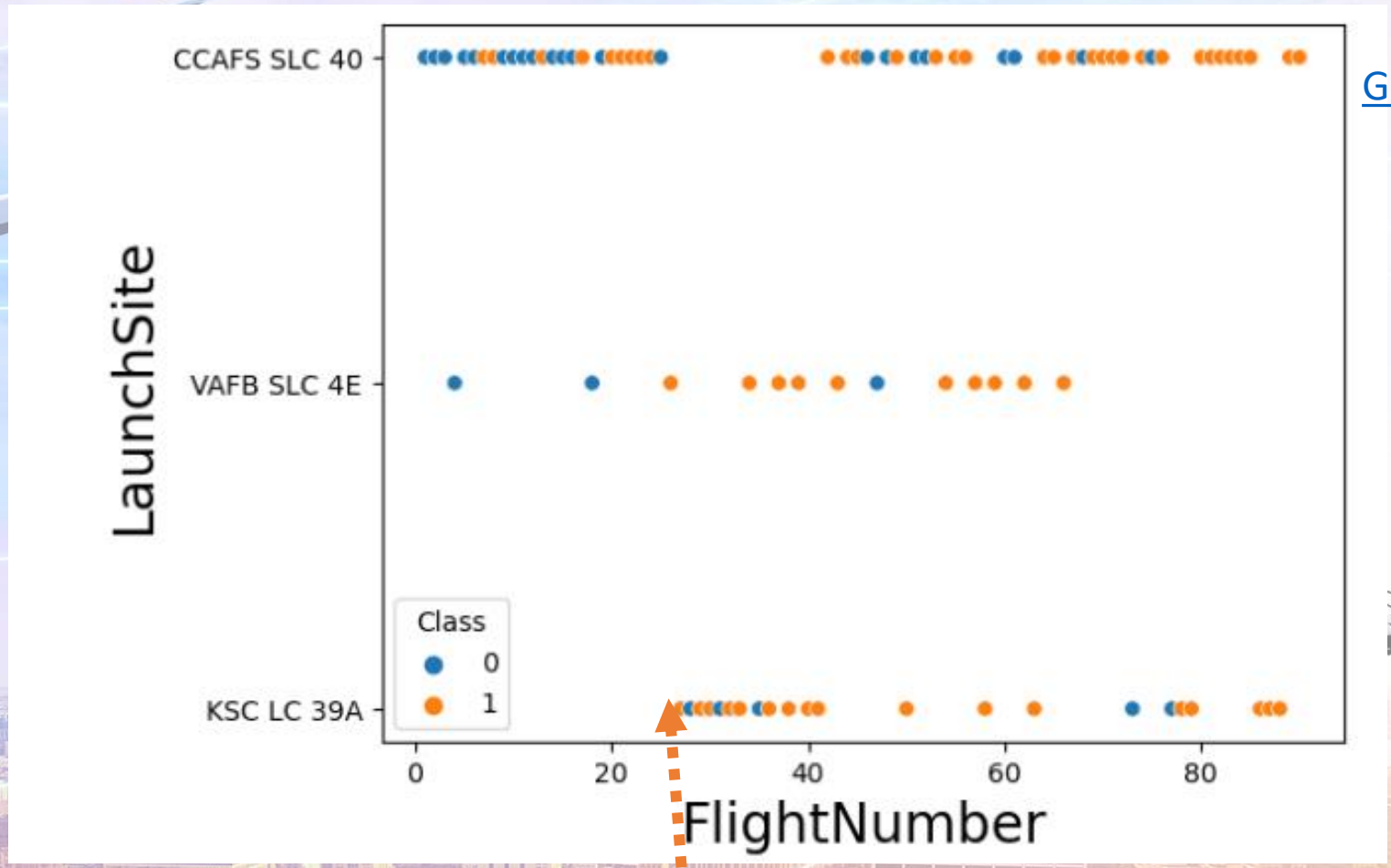
Exploratory data analysis results

Interactive analytics demo in screenshots

Predictive analysis results


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

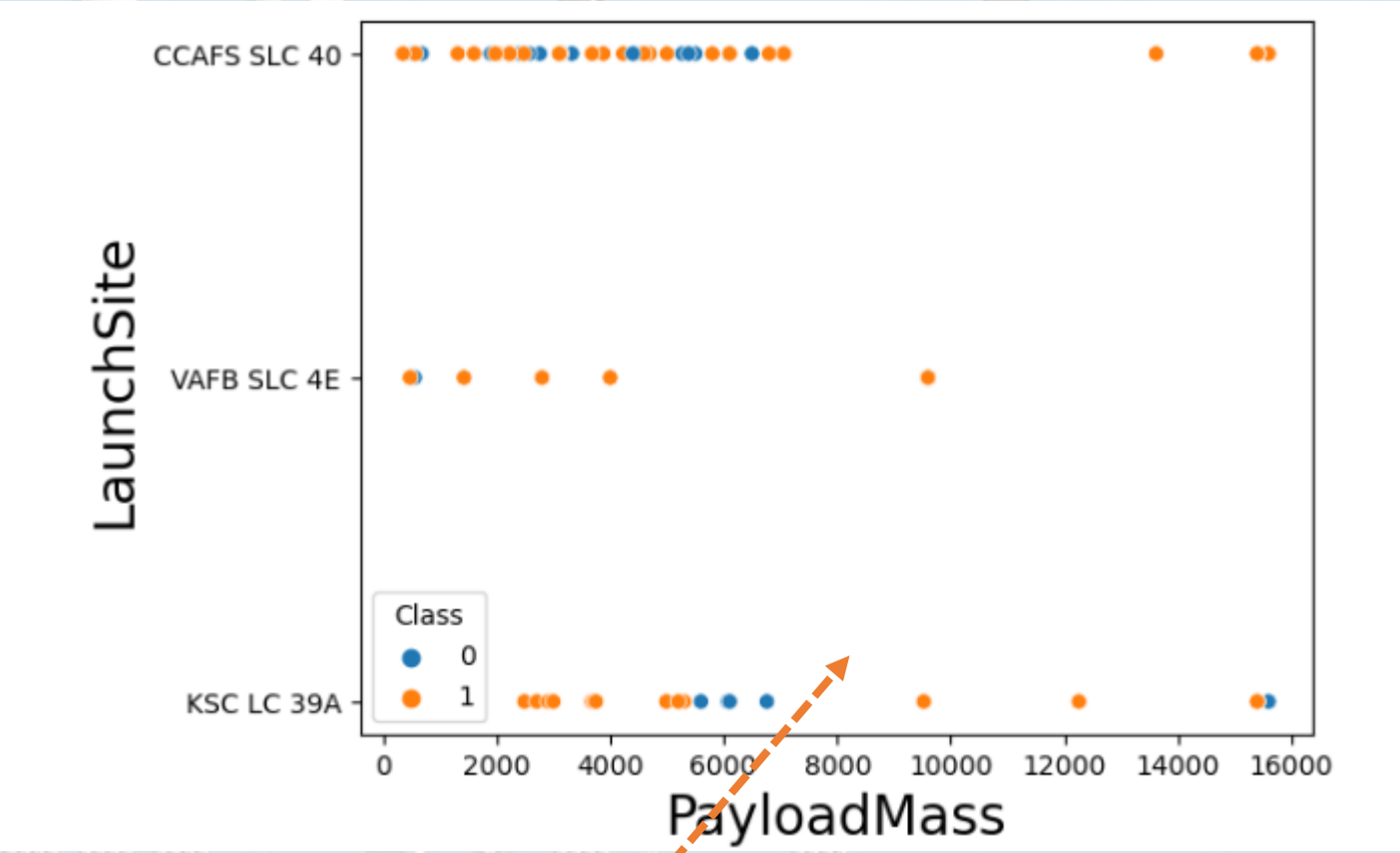


Github link

- **With higher flight numbers (greater than 30) the success rate for the Rocket is increasing.**

Payload vs. Launch Site

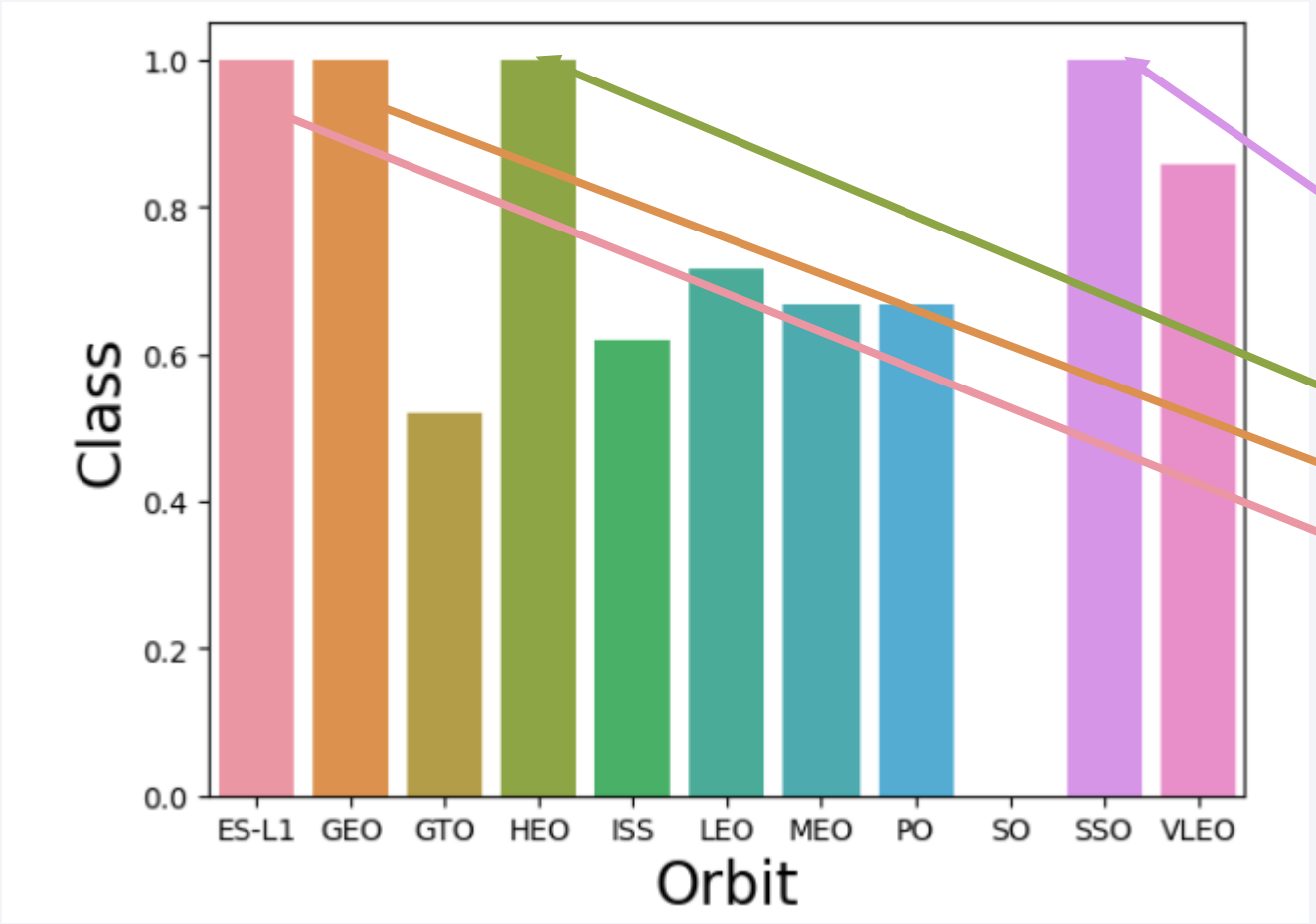
Github link



The greater the payload mass (greater than 7000 Kg) higher the success rate for the Rocket. But there are no clear pattern to take a decision, if the launch site is dependent on Pay Load Mass for a success launch.

Success Rate vs. Orbit Type

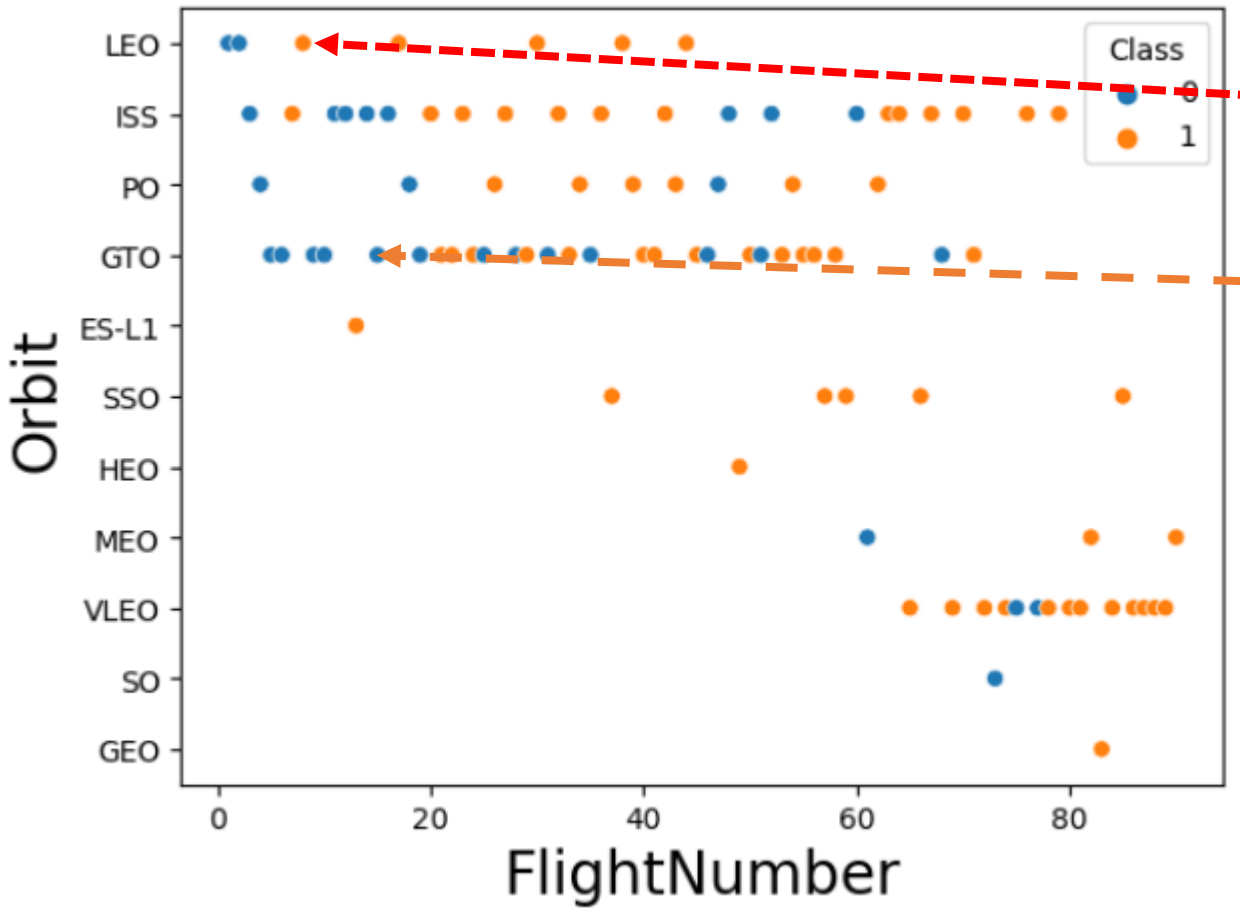
Github link



ES-L1, GEO, HEO, SSO has highest Success rates.

Flight Number vs. Orbit Type

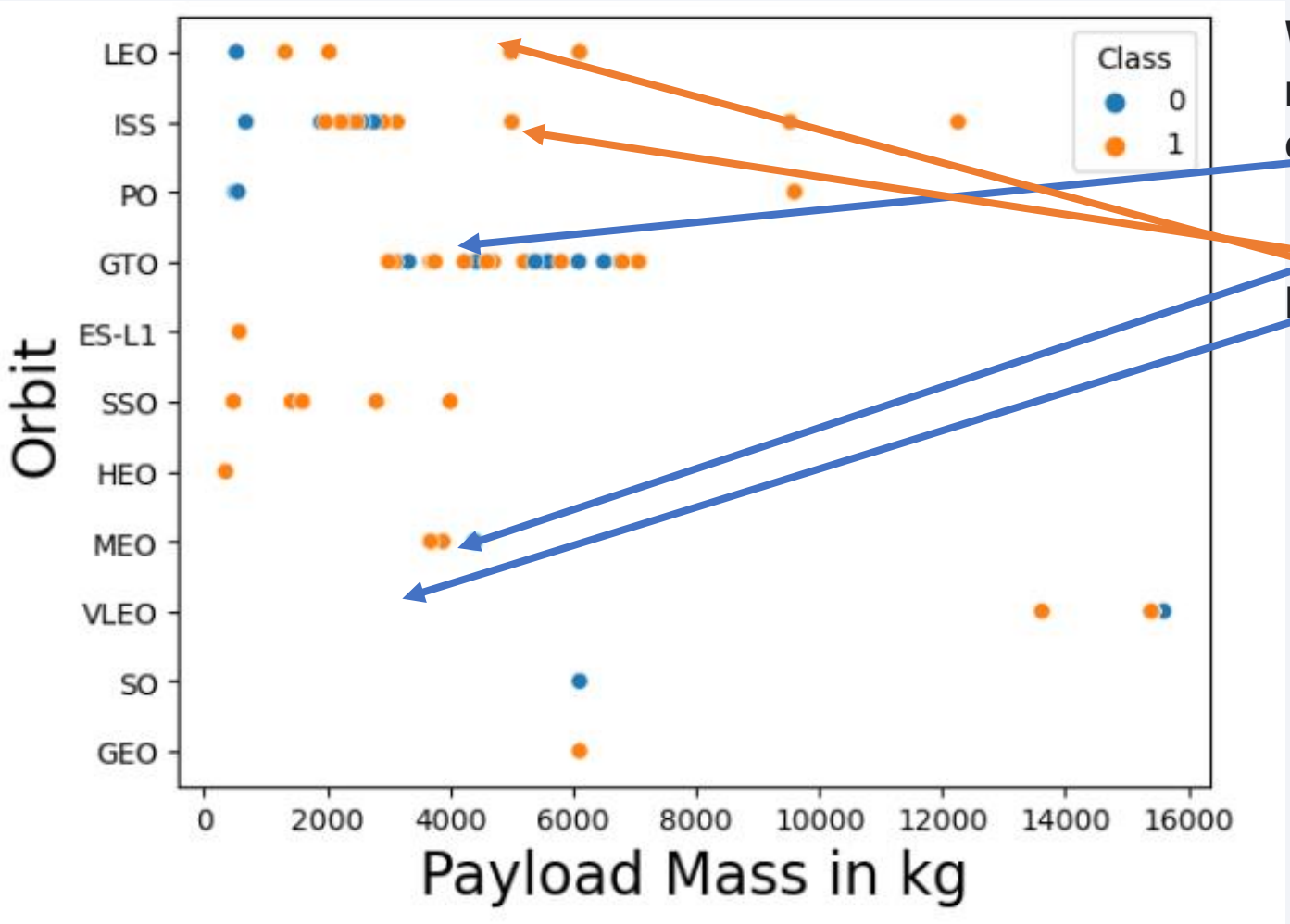
Github link



We see that for **LEO** orbit the success increases with the number of flights
On the other hand, there seems to be no relationship between flight number and the **GTO** orbit.

Payload vs. Orbit Type

Github link

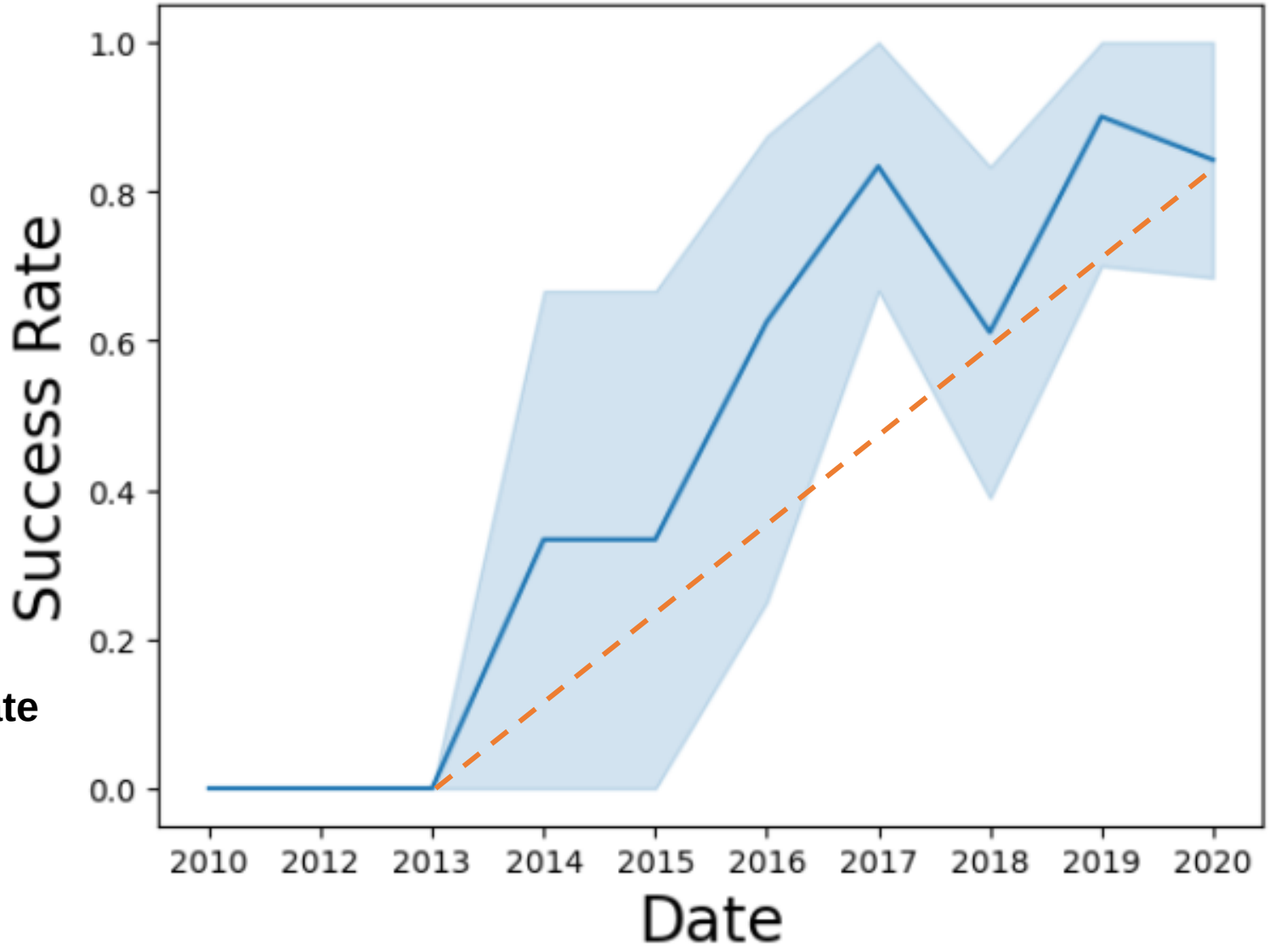


We observe that heavy payloads have a negative influence on MEO, GTO, VLEO orbits

Positive on LEO, ISS orbits

Launch Success Yearly Trend

Github link



We can observe that the success rate since 2013 kept increasing till 2020

All Launch Site Names

Github link

```
%%sql
SELECT DISTINCT("Launch_Site") AS [Unique Launch sites]
FROM SPACEXTBL;
```

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

Unique Launch sites
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Launch Site Names Begin with 'CCA'

Github link

```
%%sql
SELECT *
FROM SPACEXTBL
WHERE Launch_Site LIKE "CCA%"
LIMIT 5;
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (parachute)
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No attempt
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No attempt
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

Github link

```
1  %%sql
2  SELECT SUM(PAYLOAD_MASS__KG_) AS [Total Payload Mass]
3  FROM SPACEXTBL
4  WHERE Customer LIKE "NASA (CRS)";
```

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

Total Payload Mass
45596.0

Average Payload Mass by F9 v1.1

Github link

```
1  %%sql
2  SELECT AVG(PAYLOAD_MASS__KG_) AS [Average Payload Mass]
3  FROM SPACEXTBL
4  WHERE Booster_Version LIKE "F9 v1.1";
```

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

Average Payload Mass
2928.4

First Successful Ground Landing Date

Github link

```
%%sql
SELECT MIN(Date) AS [The first succesful landing outcome with ground pad]
FROM SPACEXTBL
WHERE Landing_Outcome LIKE "Success (ground pad)";
```

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

The first succesful landing outcome with ground pad
01/08/2018

Successful Drone Ship Landing with Payload between 4000 and 6000

Github link

```
%%sql
SELECT Booster_Version,Landing_Outcome,PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_>4000 AND PAYLOAD_MASS__KG_<6000 AND Landing_Outcome="Success (drone ship)";
```

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

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696.0
F9 FT B1026	Success (drone ship)	4600.0
F9 FT B1021.2	Success (drone ship)	5300.0
F9 FT B1031.2	Success (drone ship)	5200.0

Total Number of Successful and Failure Mission Outcomes

Github link

```
%%sql
SELECT COUNT(Mission_Outcome) AS [Total number of successful mission outcomes]
FROM SPACEXTBL
WHERE Mission_Outcome LIKE "Success%";
```

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

Total number of successful mission outcomes
100

```
%%sql
SELECT COUNT(Mission_Outcome) AS [Total number of failure mission outcomes]
FROM SPACEXTBL
WHERE Mission_Outcome = "Failure (in flight)";
```

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

Total number of failure mission outcomes
1

Boosters Carried Maximum Payload

Github link

```
%%sql
SELECT Booster_Version AS [the boosterversions which have carried the maximum payload mass],PAYLOAD_MASS__KG_
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_=(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

the boosterversions which have carried the maximum payload mass	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600.0
F9 B5 B1049.4	15600.0
F9 B5 B1051.3	15600.0
F9 B5 B1056.4	15600.0
F9 B5 B1048.5	15600.0
F9 B5 B1051.4	15600.0
F9 B5 B1049.5	15600.0
F9 B5 B1060.2	15600.0
F9 B5 B1058.3	15600.0
F9 B5 B1051.6	15600.0
F9 B5 B1060.3	15600.0
F9 B5 B1049.7	15600.0

2015 Launch Records

Github link

```
%%sql
SELECT
    CASE substr(date, 4, 2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END AS month,Date,
    Landing_Outcome,
    Booster_Version,
    Launch_Site
FROM SPACEXTBL
WHERE substr(date, 7, 4) = '2015' AND Landing_Outcome = 'Failure (drone ship)';
```

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

month	Date	Landing_Outcome	Booster_Version	Launch_Site
October	01/10/2015	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	14/04/2015	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

Github link

```
%%sql
SELECT
    COUNT(Landing_Outcome) AS Rank,Date
FROM SPACEXTBL
WHERE
    substr(date, 7, 4) || '-' || substr(date, 4, 2) || '-' || substr(date, 1, 2)
    BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY Rank DESC;
```

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

Rank	Date
10	22/05/2012
5	22/12/2015
5	04/08/2016
5	01/10/2015
3	18/04/2014
2	29/09/2013
1	28/06/2015
1	12/08/2010

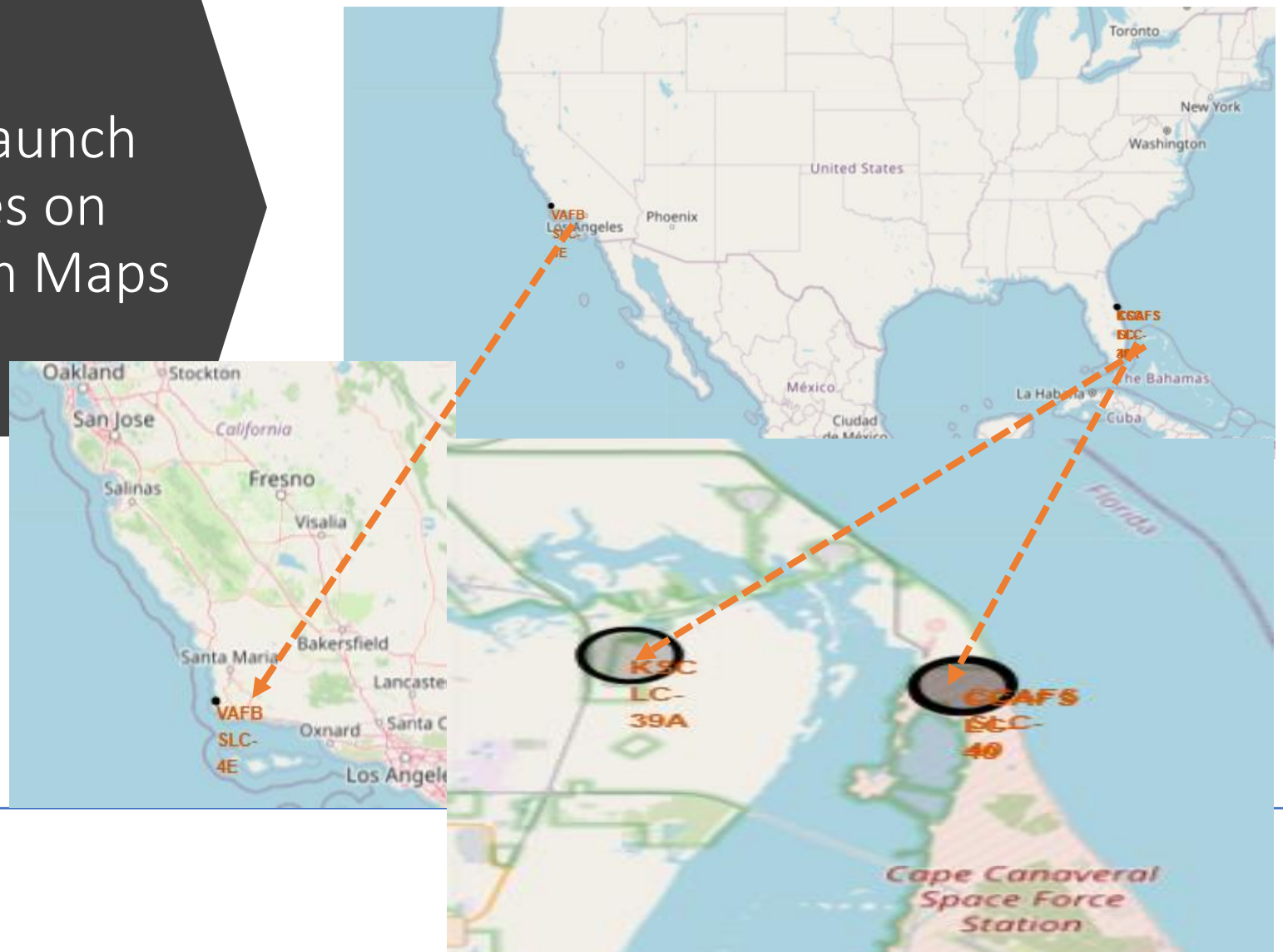
Section 3

Launch Sites Proximities Analysis

Github link

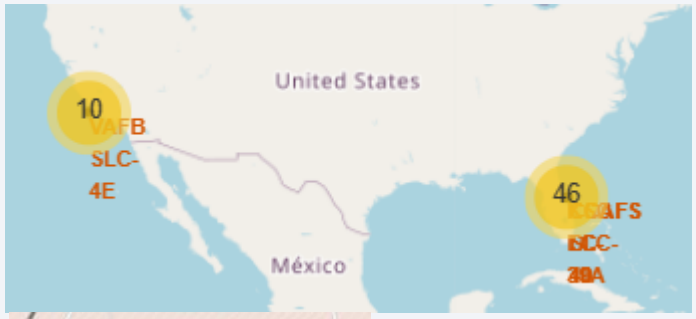
We can see that the SpaceX launch sites are near to the United States of America coasts i.e., Florida and California Regions.

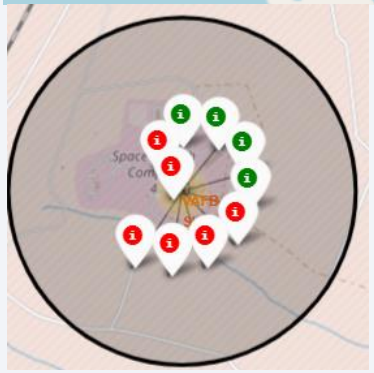
All Launch Sites on Folium Maps



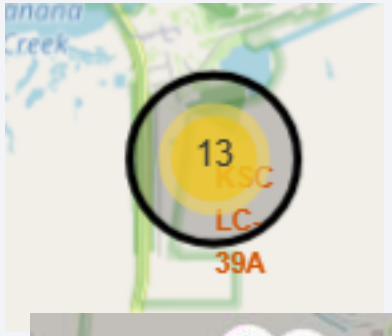
Color Labeled Launch Records

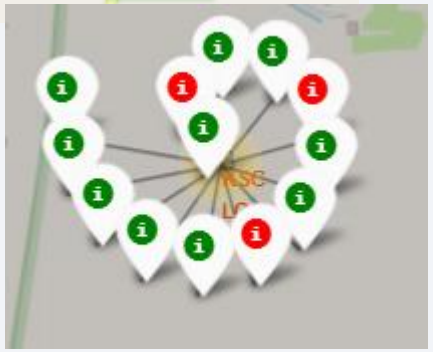
Github link



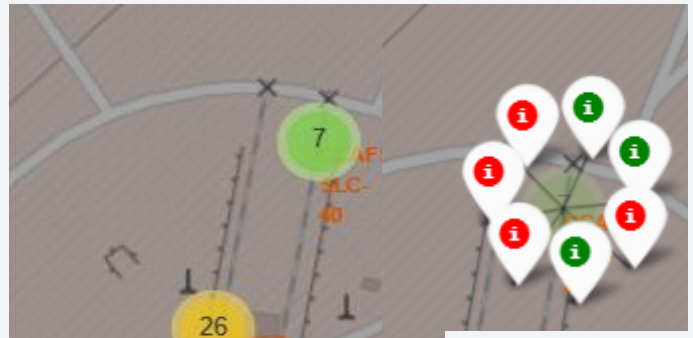


VAFB SLC-4E





KSC LC-39A



CCAFS SLC-40

CCAFS LC-40

Green Marker shows successful launches

Red Marker shows Failures

It has maximum success rate

Distance From Railroad, highway, city

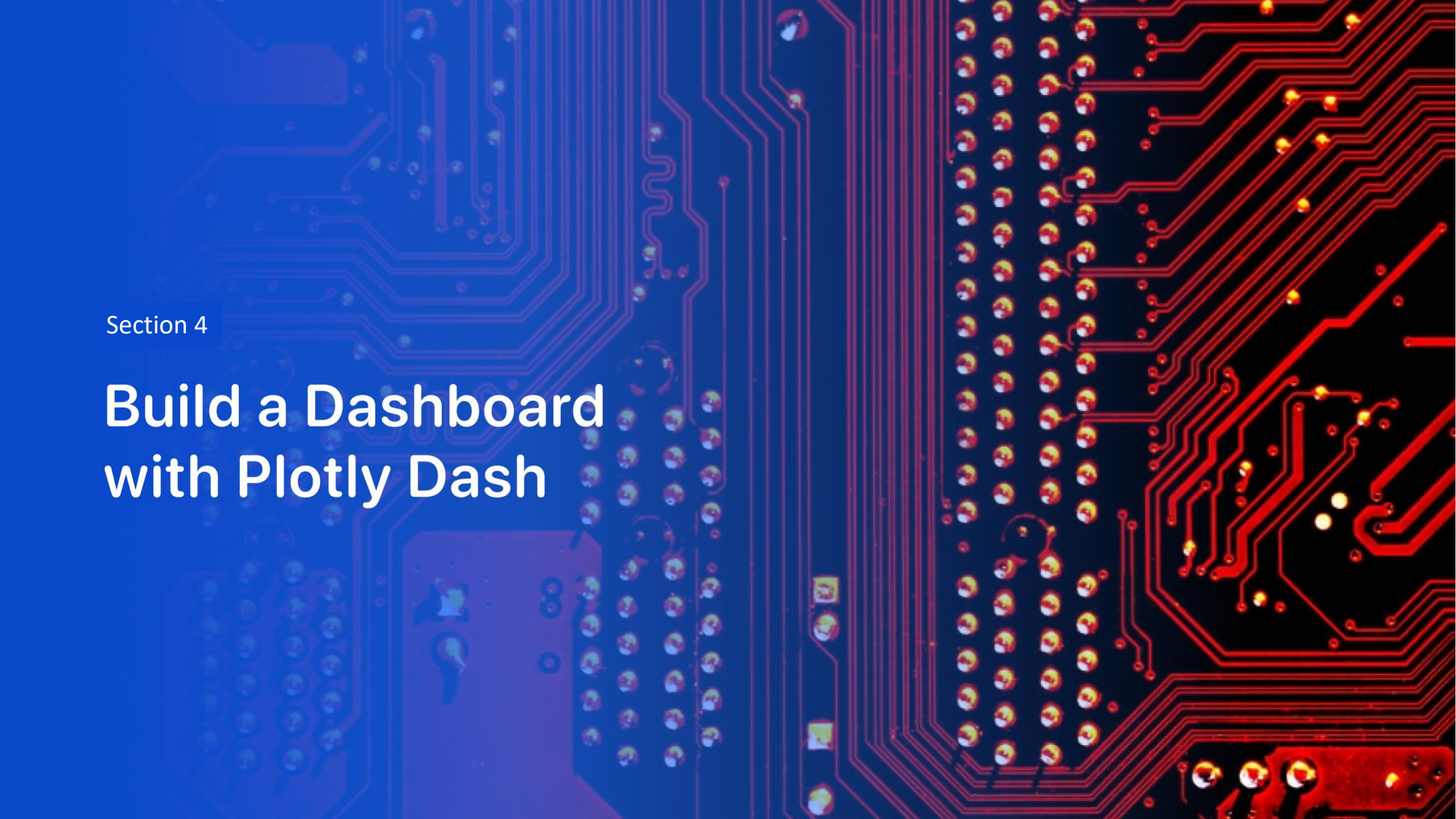
Github link

Distance for all launch sites from railway tracks are greater than 1 km for all sites. So, launch sites are not so far away from railway tracks.

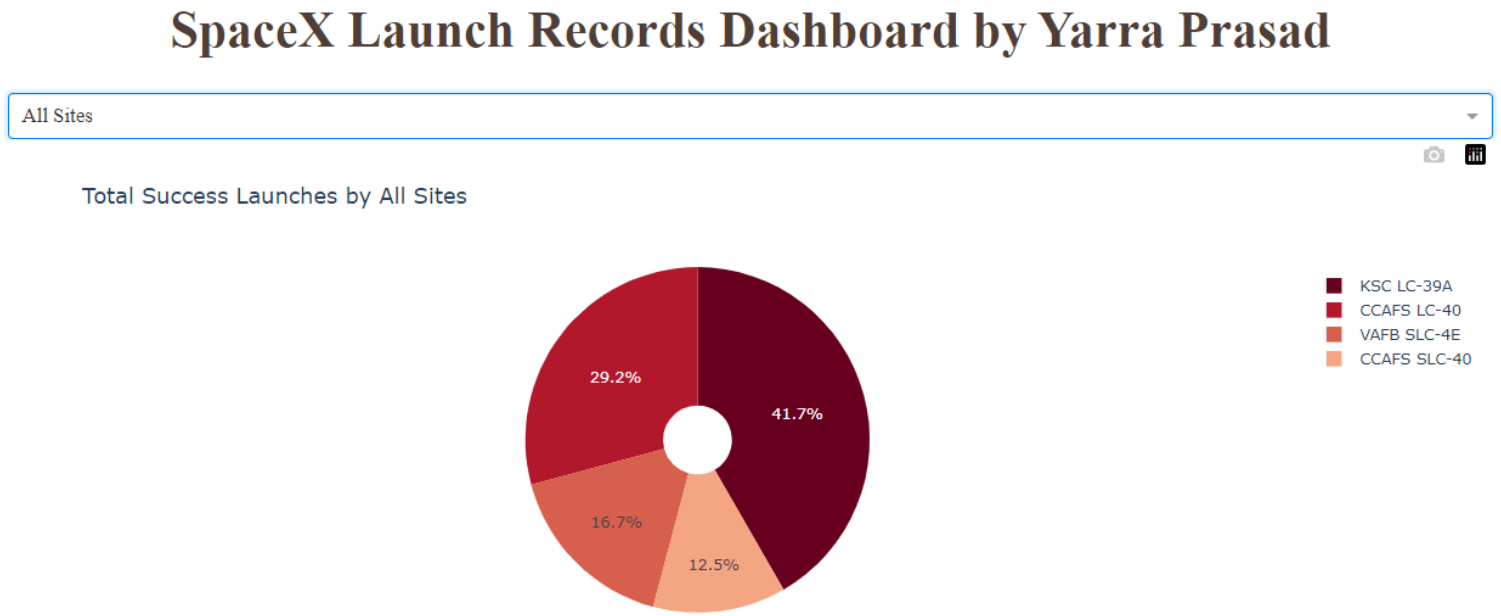
Distance for all launch sites from cities is greater than 50 km for all sites. So, launch sites are far away from cities.

```
distance_highway = calculate_distance(launch_site_lat, launch_site_lon, closest_highway[0], closest_highway[1])
print('distance_highway =',distance_highway, ' km')
distance_railroad = calculate_distance(launch_site_lat, launch_site_lon, closest_railroad[0], closest_railroad[1])
print('distance_railroad =',distance_railroad, ' km')
distance_city = calculate_distance(launch_site_lat, launch_site_lon, closest_city[0], closest_city[1])
print('distance_city =',distance_city, ' km')
```

```
distance_highway = 0.5834695366934144  km
distance_railroad = 1.2845344718142522  km
distance_city = 51.43416999517233  km
```


Section 4
Build a Dashboard with Plotly Dash

Launch Success Count for All sites

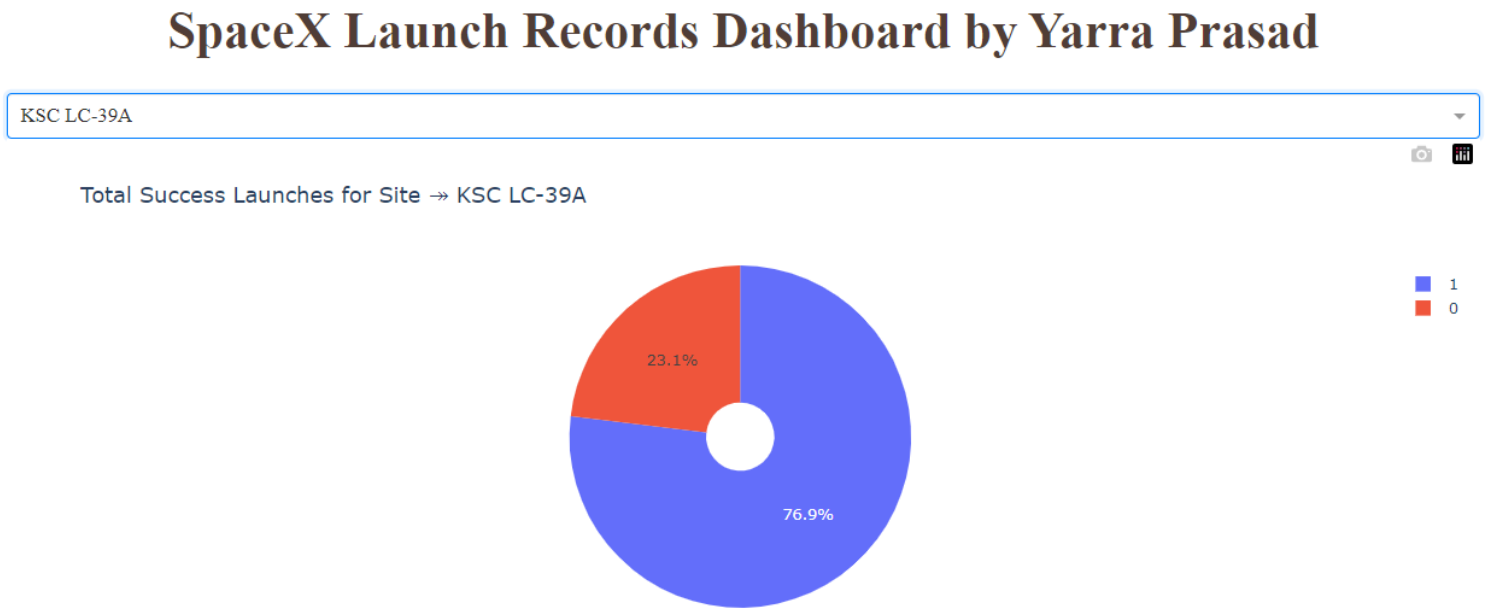


We can see that KSC LC-39A has more success rate

[Live plotly dashboard]

[Github link]

KSC LC-39A SUCCESS RATE



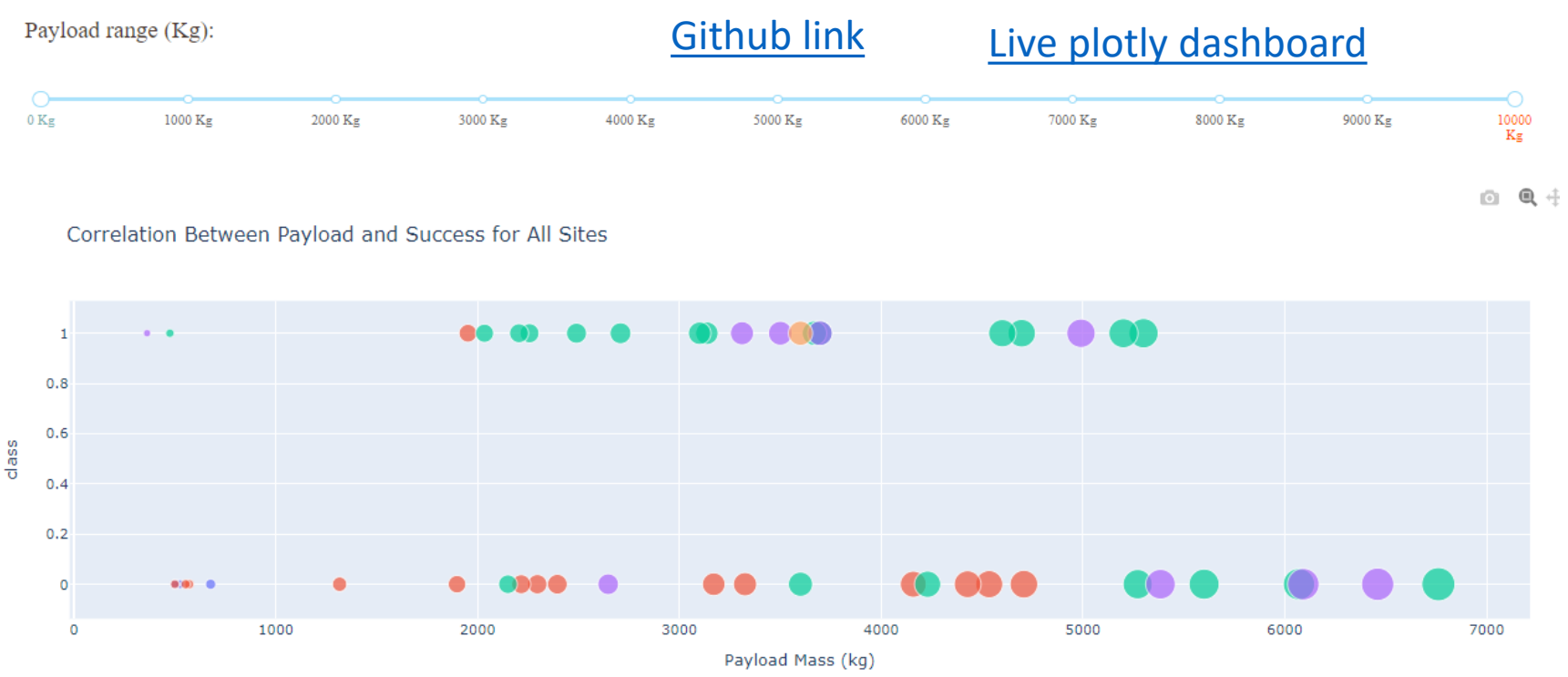
[Live plotly dashboard]

[Github link]

Correlation Between Payload and Success for all sites

Github link

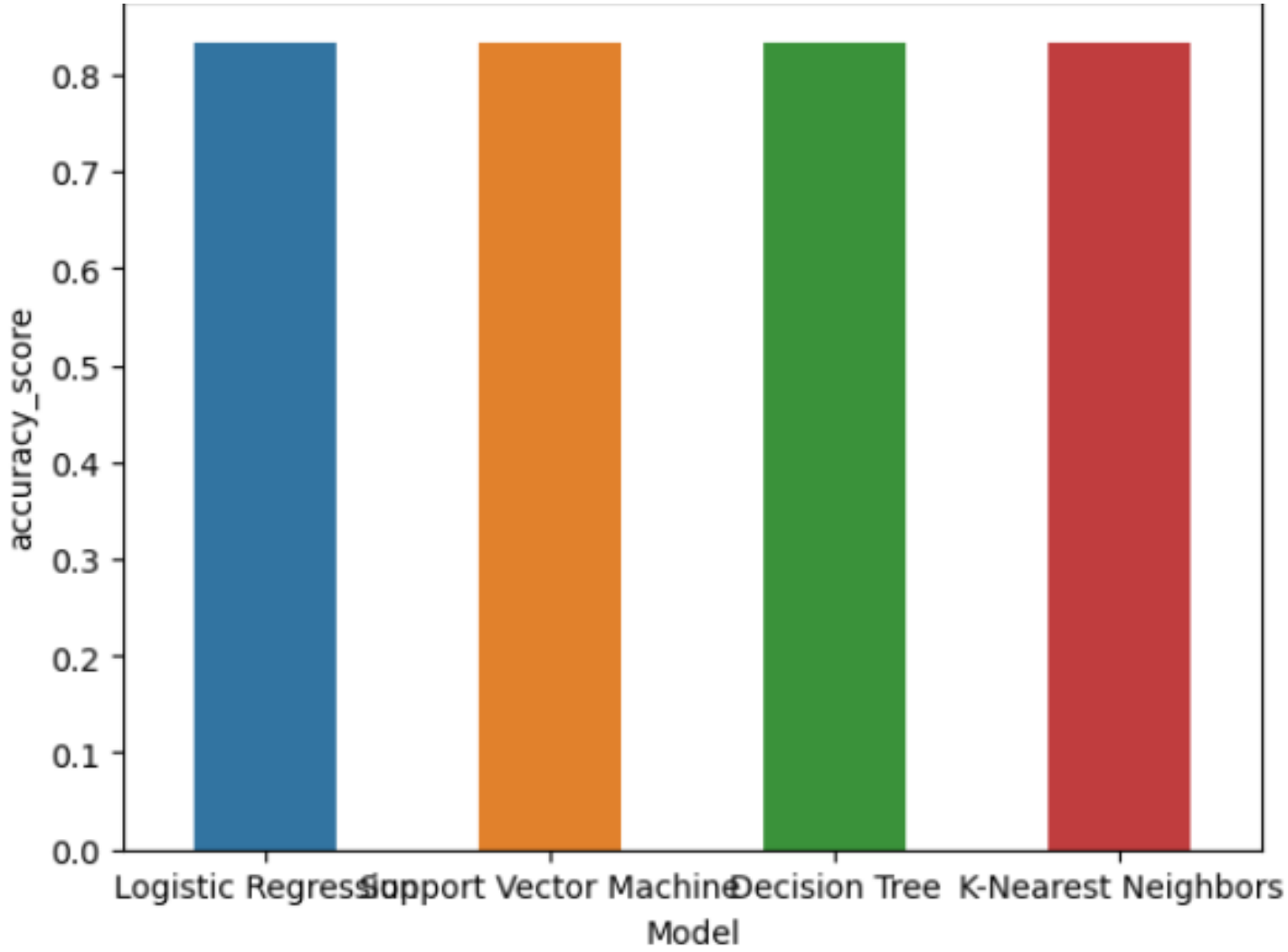
Live plotly dashboard



Section 5

Predictive Analysis (Classification)

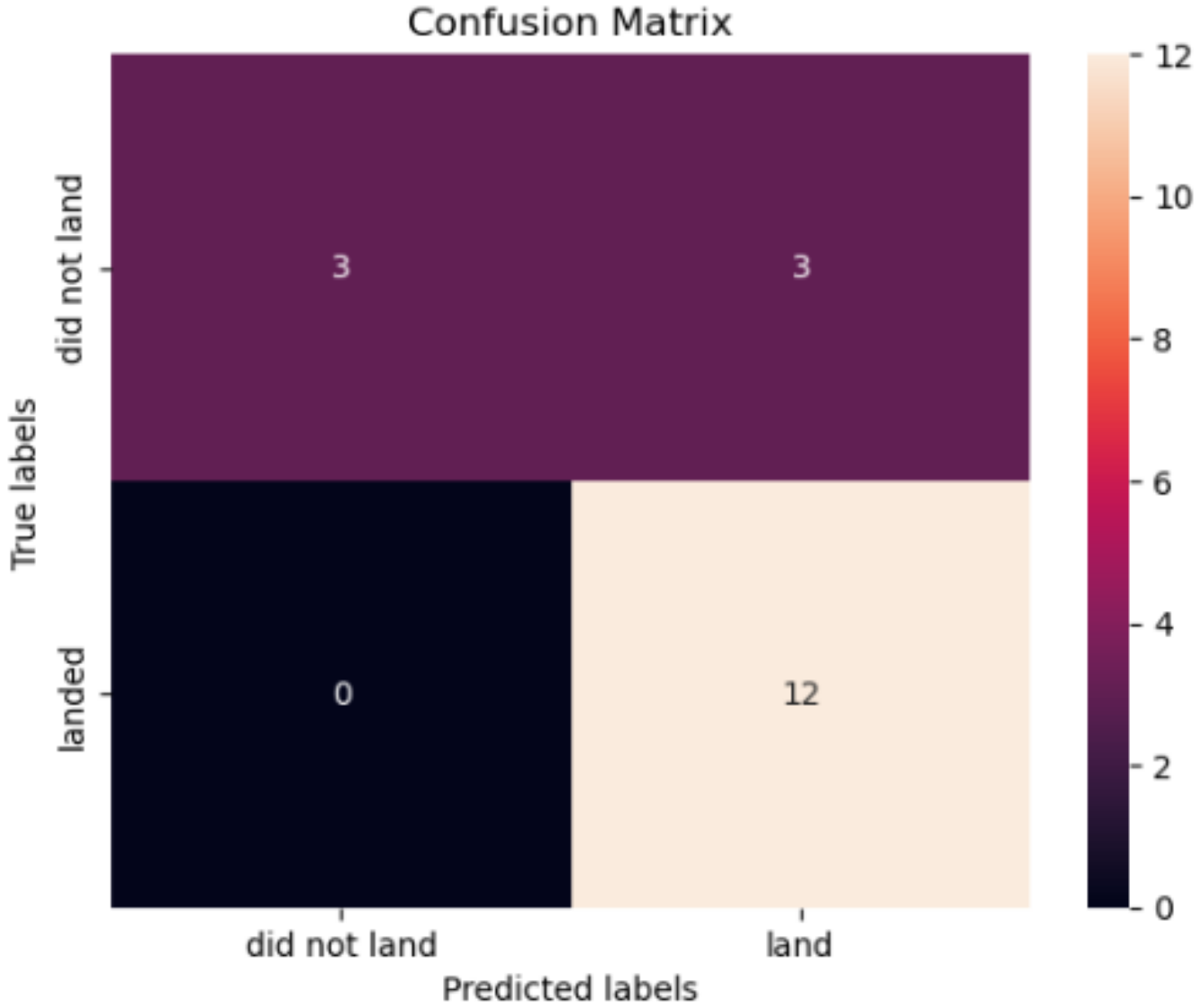
Classification Accuracy



All the Models are having same Accuracy

Github link

Confusion Matrix



For all the Models the confusion matrix is same

Github link

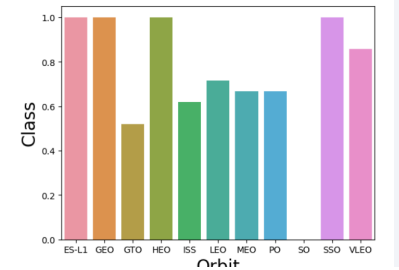
Accuracy on training and test data

Model	Accuracy on training data	Accuracy on test data	Tuning parameters
Logistic Regression	0.8464285714285713	0.8333333333333334	{'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
SVM	0.8482142857142856	0.8333333333333334	{'C': 1.0, 'gamma': 0.031622776601683793, 'kernel': 'sigmoid'}
KNN	0.8482142857142858	0.8333333333333334	{'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
Decision Tree	0.875	0.8333333333333334	{'criterion': 'gini', 'max_depth': 8, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 10, 'splitter': 'random'}

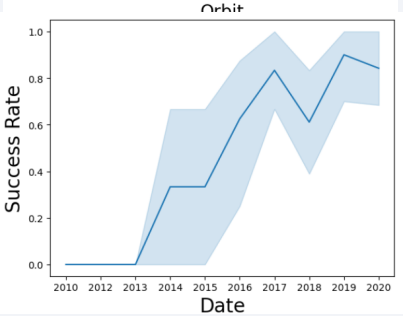
Github link

Conclusions

- Orbits ES-L1, GEO, HEO, SSO has highest Success rates

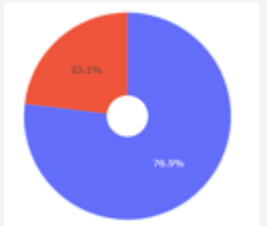


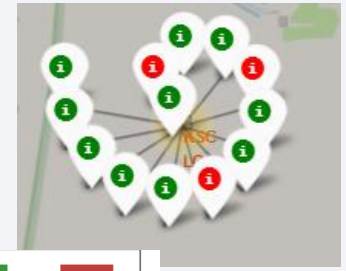
- Success rates for SpaceX launches has been increasing relatively with time and it looks like soon they will reach the required target



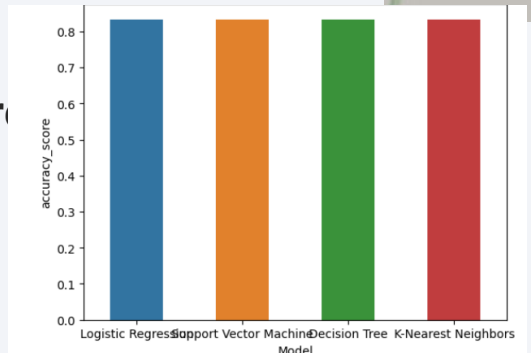
Live plotly dashboard

- KSC LC-39A had the most successful launches but increasing payload mass seems to have negative impact on success





- All Logistic Regression, KNN,SVM and Decision Tree ar performing same on test data.



Github link

Appendix

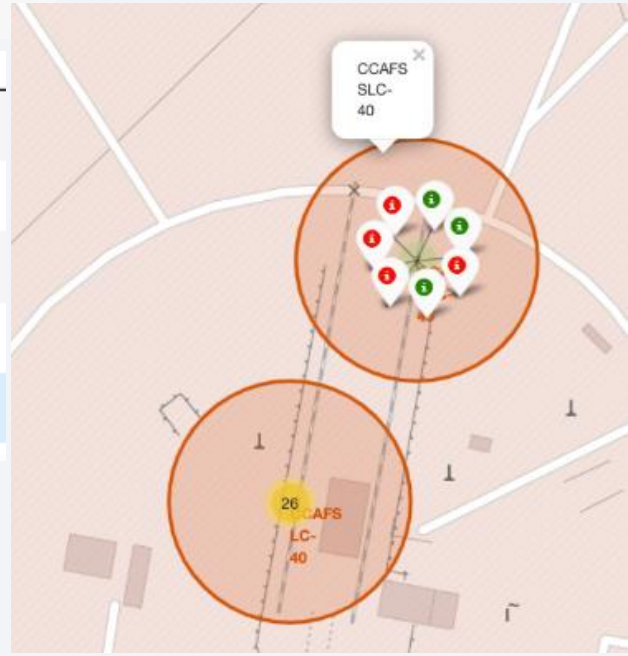
```python
def plot_confusion_matrix(y,y_predict):
    "this function plots the confusion matrix"
    from sklearn.metrics import confusion_matrix

    cm = confusion_matrix(y, y_predict)
    ax= plt.subplot()
    sns.heatmap(cm, annot=True, ax = ax); #annot=True to annotate cells
    ax.set_xlabel('Predicted labels')
    ax.set_ylabel('True labels')
    ax.set_title('Confusion Matrix');
    ax.xaxis.set_ticklabels(['did not land', 'land']); ax.yaxis.set_ticklabels(['did not land', 'landed'])
    plt.show()
```

```sql
%%sql
SELECT COUNT(Mission_Outcome) AS [Total number of successful,
                                  Failure mission outcomes]
FROM SPACEXTBL;
```

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

Total number of successful, Failure mission outcomes
101

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0



[Github link]

Thank you!