IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

JULIO ANDRÉS
CHAPARRO
18/01/2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection.
  - Data Wrangling.
  - EDA with data visualization.
  - EDA with SQL.
  - Interactive map with folium.
  - Dashboard with Plotly Dash.
  - Classification of Predictive analysis
- Summary of all results
  - Exploratory data analysis results.
  - Interactive analytics demo in screenshots.
  - Predictive Analysis with results.

Introduction

- Project Background.
- This project is made for predicting if the Falcon 9 first stage will land successfully. SpaceX advertises that Falcon 9 rocket launches on its website, with a cost of 62 millon of dollars; Other providers has a cost of 165 millons of dollars each one. This is cheaper because SpaceX can reuse the first stage. This information can be used if an alternative company wants to bid against SpaceX for a rocket launch.
- Problems for solving.
  - The influent when the rocket will land successfully.
  - Each relationship with certain rocket variables will impact in the success of landing.
  - The conditions that SpaceX has to archieve to get the best results.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - This data set contains the required information.
    - SpaceX API REST.
    - Web Scrapping
- Perform data wrangling
  - In the data set, there are several different cases where the booster didn’t land successfully. The data was performed according to the list.

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL

We are performing exploratory data analysis based on the data set of the SpaceX.

- Perform interactive visual analytics using Folium and Plotly Dash
- Enable direct data exploration and analytics and identify patterns faster.

Perform predictive analysis using classification models

With the data set, I can build interactive graphics and mark the phases in the map for the launch.

Methodology

I work with SpaceX launch data that is gathered form the SpaceX API Rest. This API will give me about the launches, including information about the rocket used, launch specifications, landing specifications and landing outcome.

The main goal is use this data for predicting whether SpaceX will atempt to land a rocket or not.

Data Collection - SpaceX

1) Getting Response from API.

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

2) Getting Response from API.

```
response= requests.get(static_json_url).json()
data= pd.json_normalize(response)
```

3) Apply custom functions to clean data.

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

4) Assign list to dictionary then dataframe.

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

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

5) Filter dataframe and export.

```
data_falcon= df.loc[df[‘BoosterVersion’]!=”Falcon 1”]
data_falcon.to_csv('spacex_web_scraped.csv',
index=False)
```

Data Collection – Web Scraping

1) Getting Response from HTML.

```
# assign the response to a object
page = requests.get(static_url)
```

2) Creating BeautifulSoup Object

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(page.text, 'html.parser')
```

3) Finding Tables.

```
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

4) Getting column names.

```
#Extract Column Names
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

Data Collection – Web Scraping

5) Creation of Dictionary.

```
# Creation of Dictionary
launch_dict= dict.fromkeys(column_names)

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


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'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6) Appending data to keys

```
#Extract each table
for table_number,table in
enumerate(soup.find_all('table',"wikitable
plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as
number corresponding to launch a number
```

7) Converting dictionary to dataframe.

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

8) Export Dataframe to .csv.

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

Link to GitHub

11

Data Wrangling

In the dataset, there are several classes where the booster didn’t land successfully. Sometimes a launch wasn’t successfully due to an accident. According to the rules, the mission outcome was successfully landed to a ground pad false RTLS means the mission was unsuccessfully landed.

Meanwhile true ASDS means the mission was outcome was successfully landed on a drone ship. I search mainly convert those outcomes into a Training Labels according to the data where 1 means successfully landed and 0 means unsuccessfully landed.

Data Wrangling Process

Perform Exploratory Data Analysis EDA On dataset

Calculate the number of launches At each site.

Calculate the number and Occurrence Of each orbit.

Calculate the number And occurrence of the Mission outcome per Orbit type.

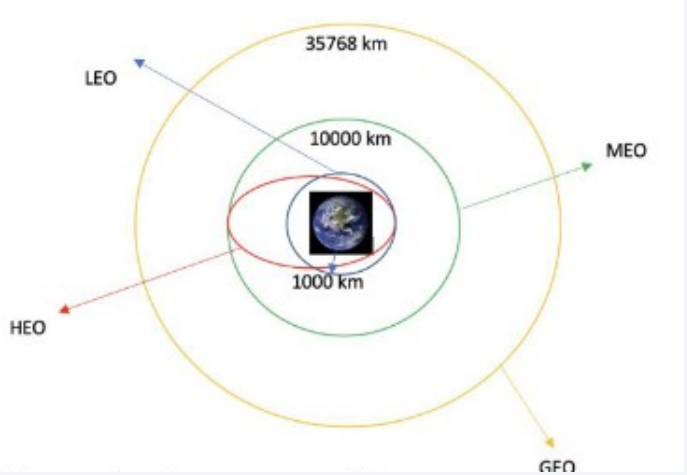
Export Dataset as .CSV.

Create a Landing outcome label form outcome column.

Work out success rate for every landing in Dataset.

Link to GitHub

Each Launch aims to an dedicated Orbit, and here there are some Common orbits types.



EDA with Data Visualization

Scatter Graphs Being Draws:

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

Bar Graph being drawn:

- Mean vs Orbit: It is used to compare
- sets of data between several groups
- at a glance. The graphs represents
- categories on one axis and a discrete
- values. The main goal is compare
- the relationship between the 2 axes.

Link to GitHub

EDA with SQL

Performed SQL Queries, to gather information about dataset, which I used for getting the answers in the dataset:

- Displaying the names of the unique launch sites in the space mission.
- Displaying the total payload mass carried by boosters launched by NASA (CRS).
- Displaying the average payload mass carried by booster version F9.
- Listing the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000.
- Listing the total number of successful an failure mission outcomes.

EDA with SQL

- Listing the records which will display the month names, successful landing_outcomes in ground pad, booster version, launch_site for the months in year 2017.
- Ranking the count of successful landing_outcomes between, the date 2010-06-04 and 2017-03-20 in descending order.

Link to GitHub

Build an Interactive Map with Folium

To visualize the launch Data into an interactive map, I look the latitude and longitude coordinates, at each launch site and added a Circle Market around each launch site with a label of name of the launch site.

We assigned the dataframe launch_outcomes to classes 0 and 1.

Using Haversine’s formula, I look the distance from the launch site to the several landmarks to find trends about what is around the Launch Site for measuring patterns.

Link to GitHub

Build a Dashboard with Plotly Dash

I used python for playing around with the data and view those data. The dashboard is built with Flask and Dash web framework.

- Pie chart showing the total launches by a certain site.
- Display a relative part of multiple classes of data.
- Size of the circle can be proportional to the total quaranty.

- Scatter graph showing the relationship with outcome and Payload mass for the different Booster Version.
- It shows the relationship between two variables.
- It’s the best method to show a non-linear pattern.

Predictive Analysis (Classification)

Build Model:

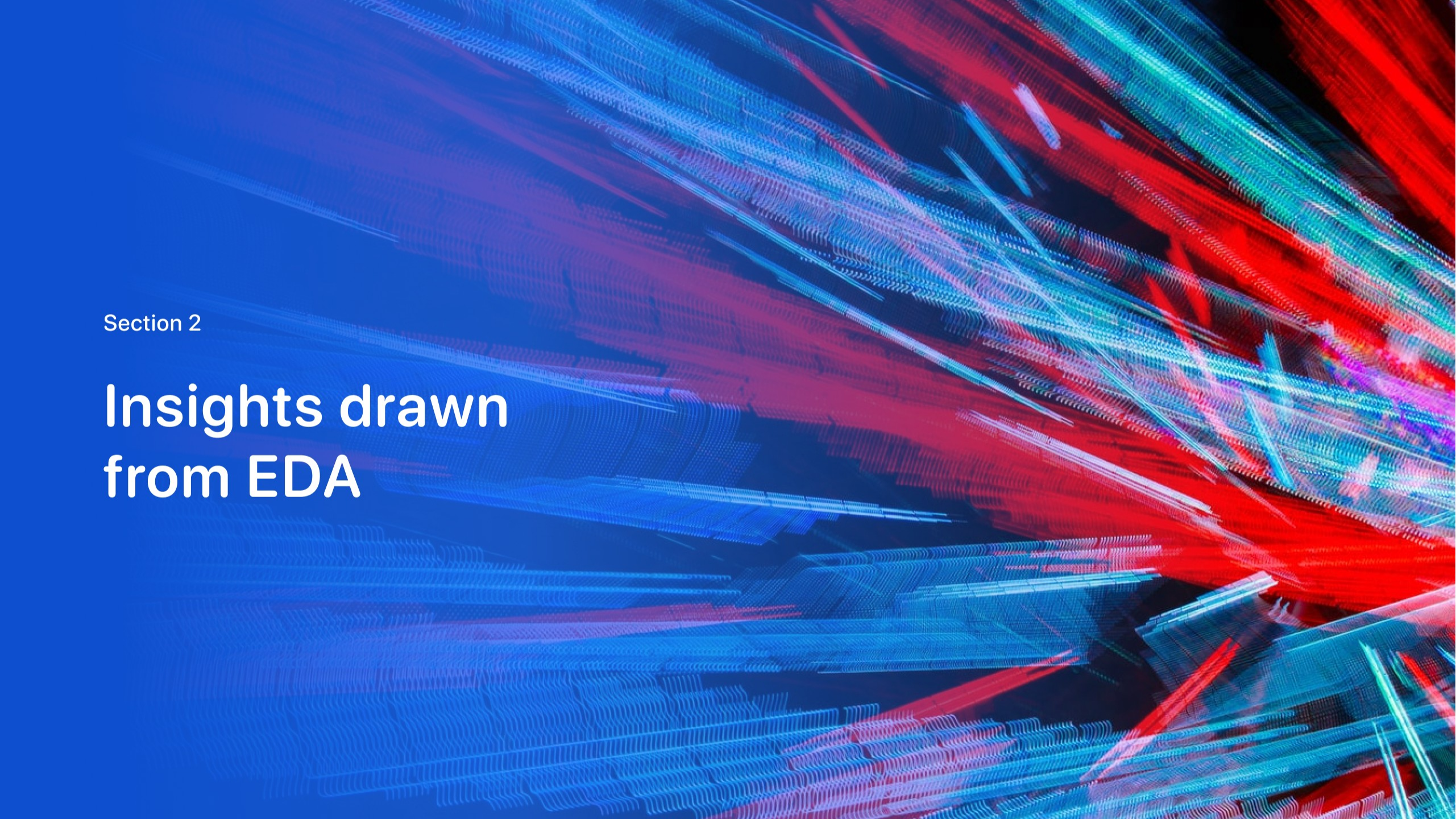
- Load the dataset into NumPy and Pandas.
- Transform Data.
- Split the data into training and test the data.
- Which type of machine learning algorithms is the best for using.
- Fit the databases into the GridSearchCV objects and train.

- Evaluate Model:
  - Plot Confusion Matrix.
  - Check accuracy for the models.
- Improve Model:
  - Algorithms Tuning.
  - Feature Engineering.
  - Find the best performance.
  - The model with the best accuracy score wins the best performance
  - There is a dictionary of algorithms with scores at the bottom of the notebook.

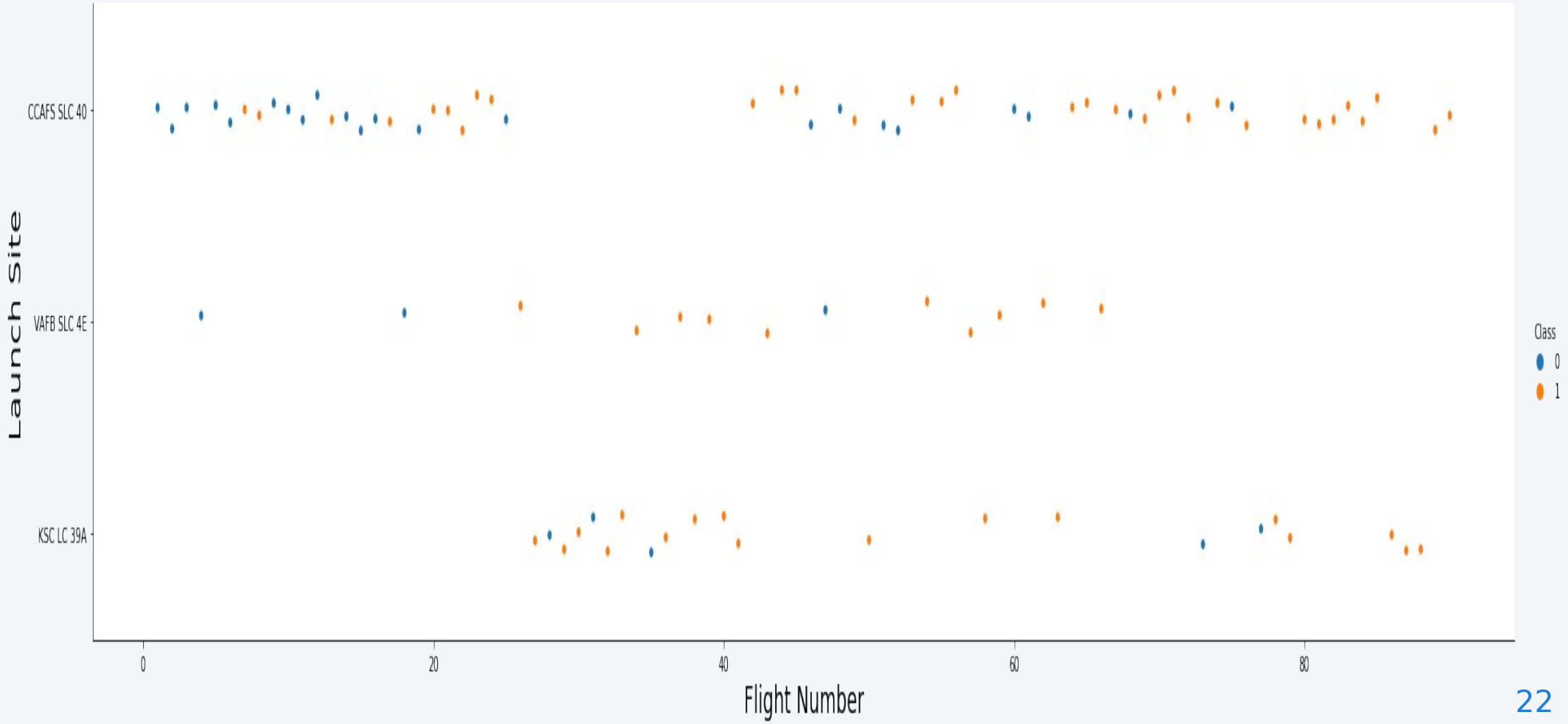
Link to GitHub

Results

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

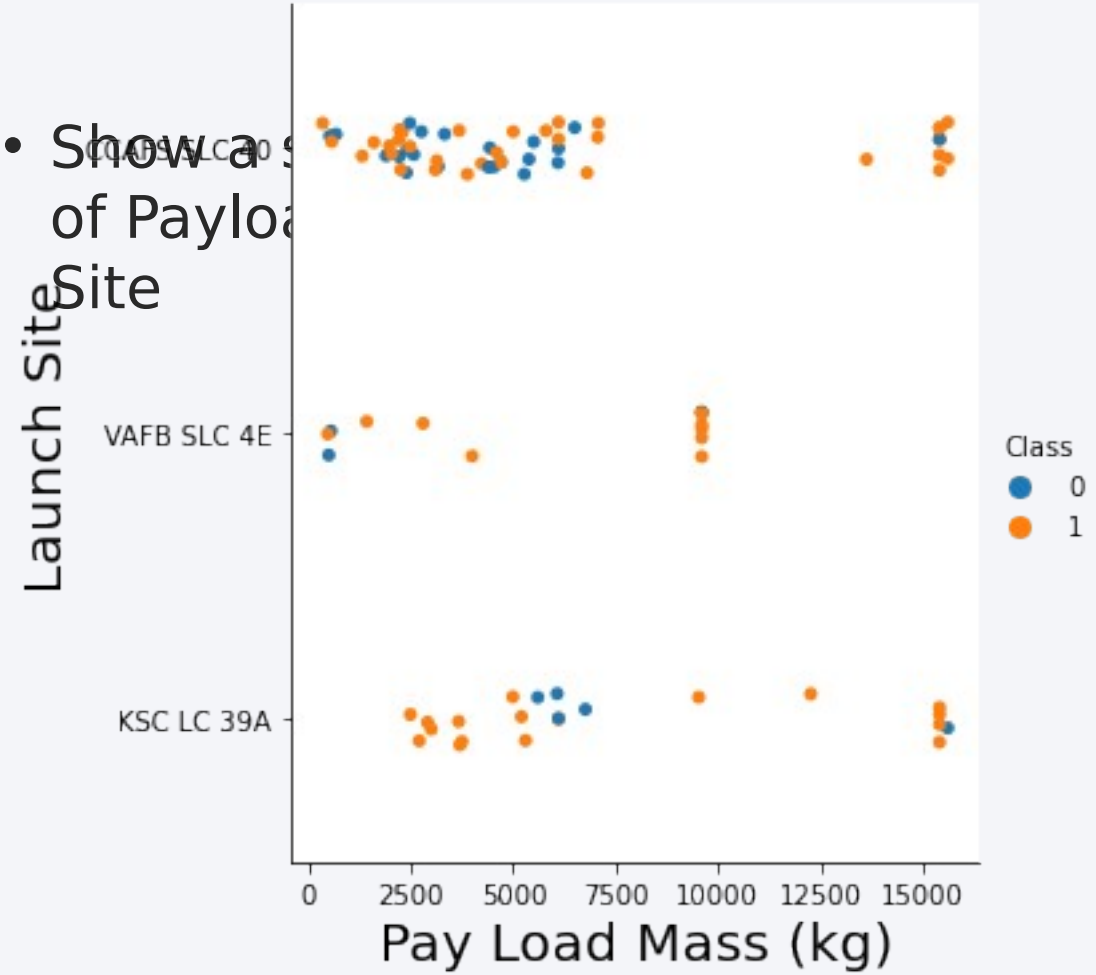

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



Payload vs. Launch Site

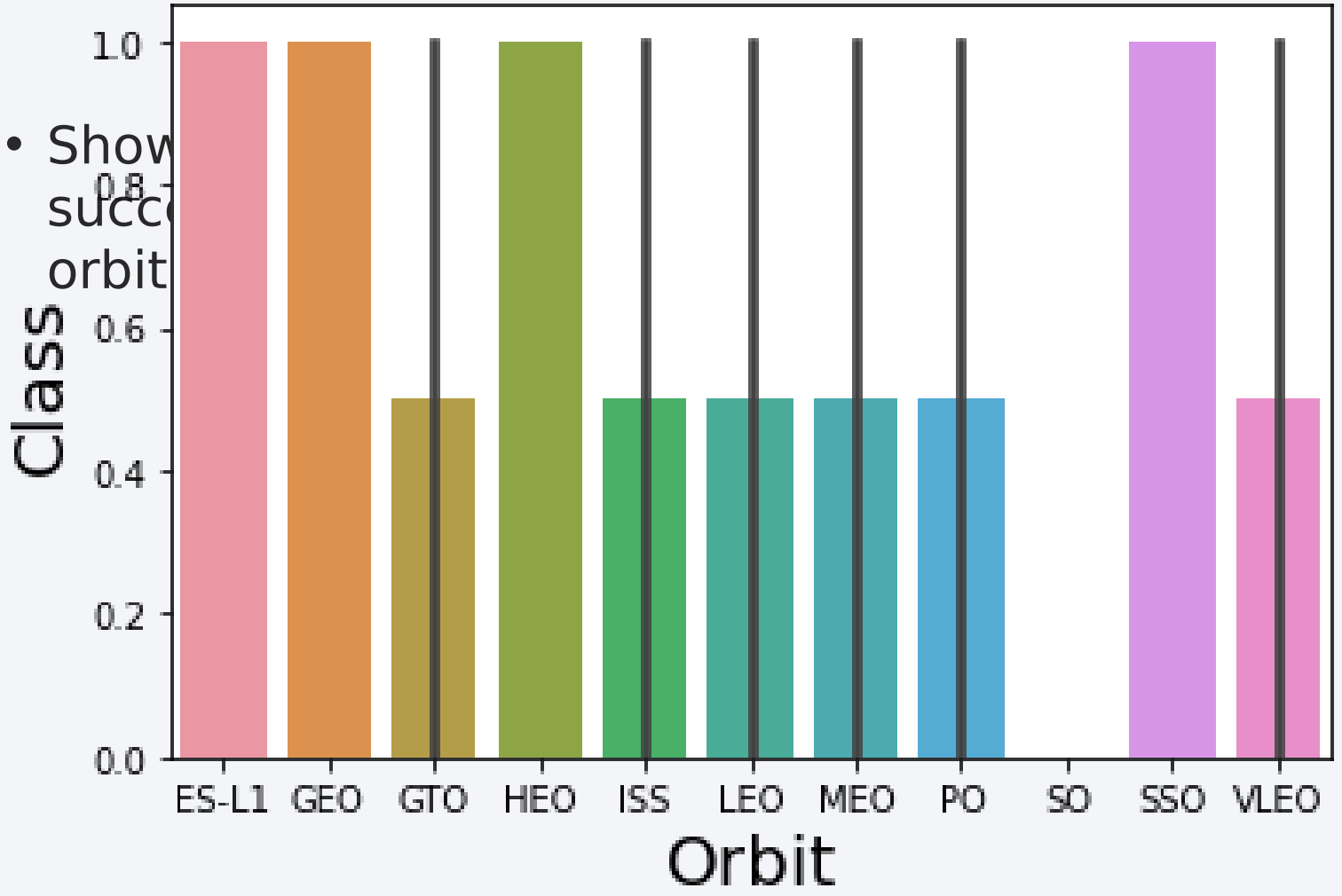
- Show a
  of Payloa
  Site



The greater the payload mass for launch site CCAFS SLC 40 the higher the sucess rate for The rocket.

Success Rate vs. Orbit Type

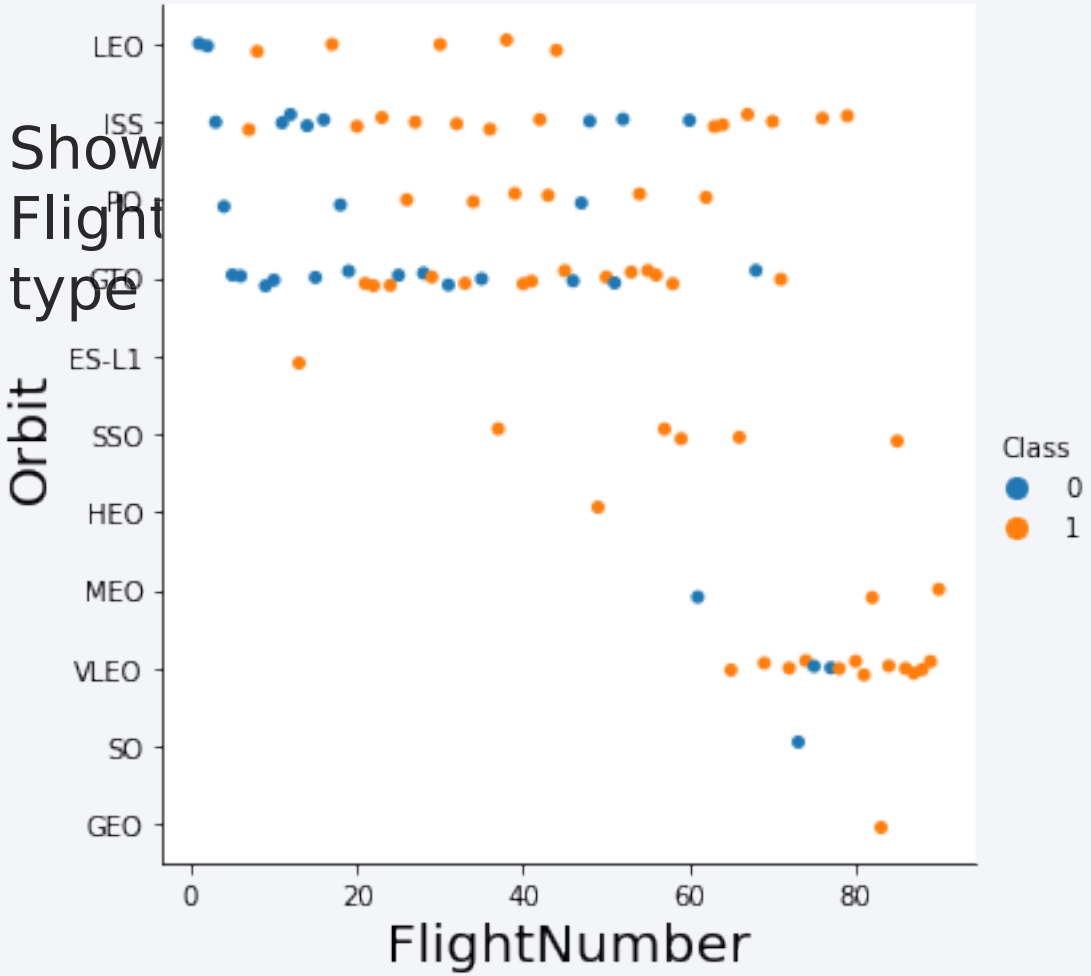
- Show
  succ
  orbit



Orbit SSO, ES-L1, GEO and HEO
Has the best sucess rate.

Flight Number vs. Orbit Type

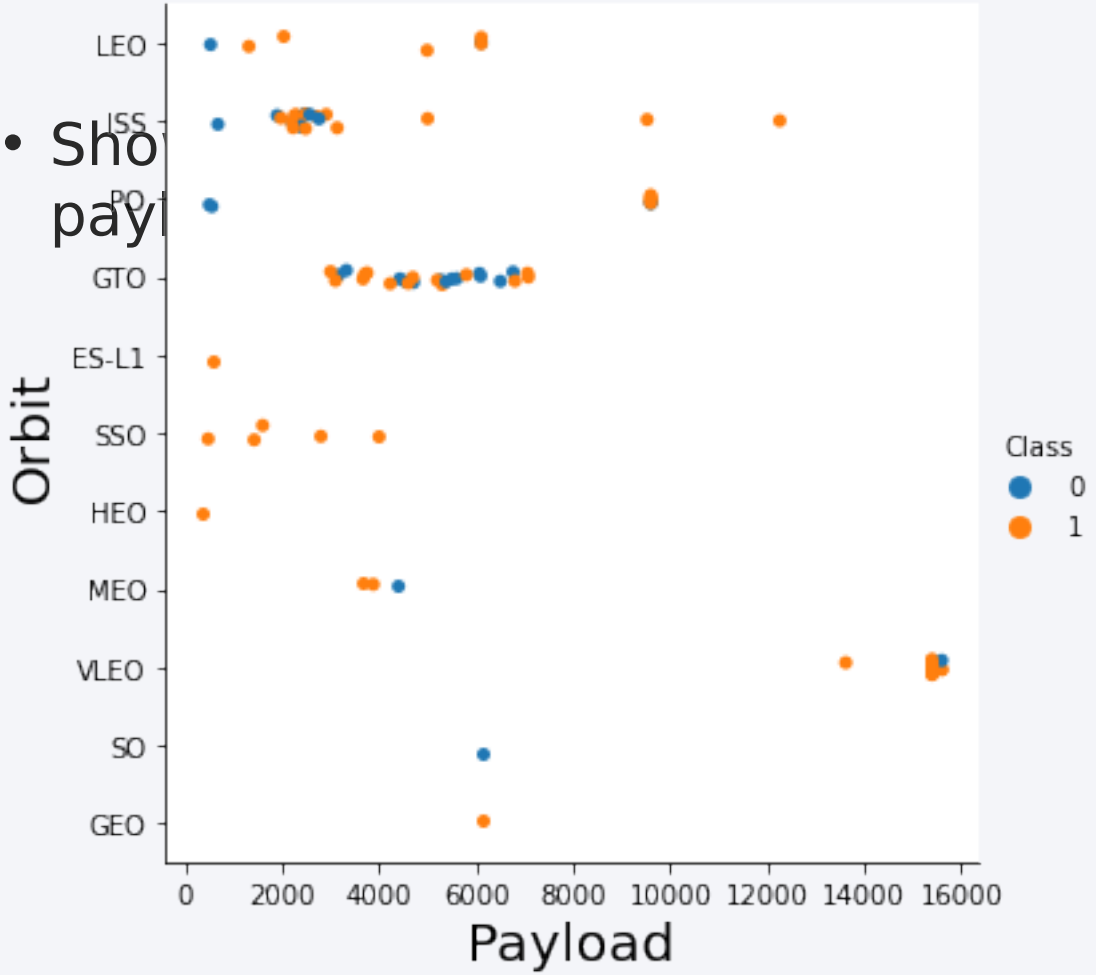
- Show
  Flight
  type



You should see that in 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

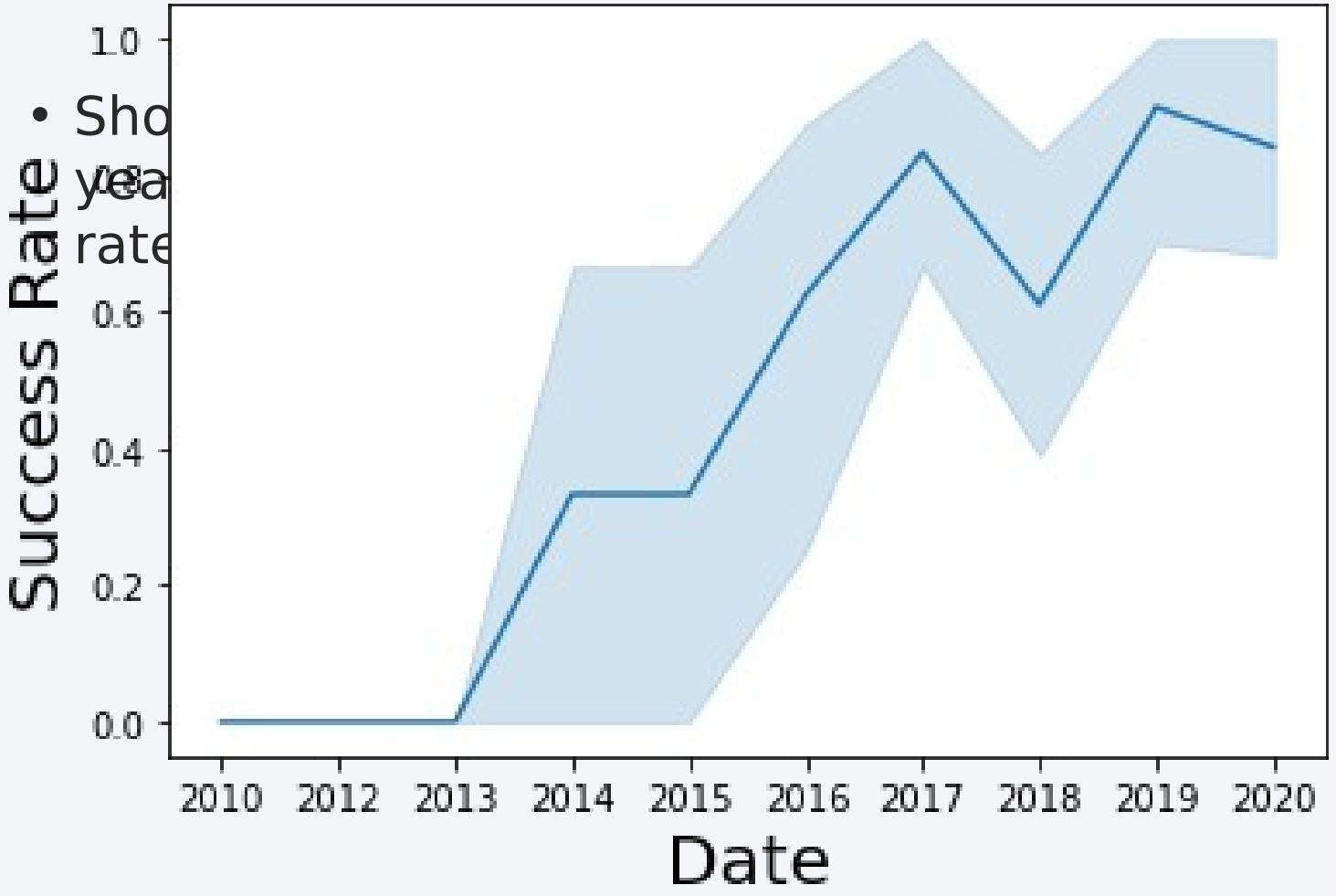
- Sho
  payl



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

- Sho
yea
rate



Observe that the sucess rate Since 2013 kept increasing till 2020.

All Launch Site Names

Select Distinct launch_site from tblSpaceX.

Using the Distinct it means that it will be only Showned unique values in the launch_site Column from tblSpaceX.

Unique Launch Sites
CCAFS SLC 40
CCAFS SLC 40
CCAFS SLC 40
VAFB SLC 4E
CCAFS SLC 40

Launch Site Names Begin with 'CCA'

	Flight Number	Date	Time (UTC)	Booster Version	Launch Site	Payload	Payload Mass (kg)	Orbit	Customer	Mission Outcome	Landing Outcome	class	Lat	Long
1	30	19/02/17	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490.00	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)	1	28.573255	-80.646895
2	31	16/03/17	06:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600.00	GTO	EchoStar	Success	No attempt	0	28.573255	-80.646895
3	32	30/03/17	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300.00	GTO	SES	Success	Success (drone ship)	1	28.573255	-80.646895
4	33	01/05/17	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	3696.65	LEO	NRO	Success	Success (ground pad)	1	28.573255	-80.646895
5	34	15/05/17	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070.00	GTO	Inmarsat	Success	No attempt	0	28.573255	-80.646895

Select top 5 * from tblSpaceX where Launch_Site LIKE ‘KSC %’

Total Payload Mass

Calculate the total payload carried by boosters from NASA

Total Payload Mass	
0	45596

```
Select
SUM(PAYLOAD_MASS_KG_)TotalPayloadMass
From tblSpaceX where Customer =’NASA (CRS)’,
TotalPayloadMass
```

Using the function SUM, it summates the total in the column Payload_Mass_KG_, where the filter In the dataset to only perform calculations on Customer NASA(CRS)

Average Payload Mass by F9 v1.1

Calculate the average payload mass carried by booster version F9 v1.1

Average Payload Mass
2928

Select
AVG(PAYLOAD_MASS_KG_)AveragePayloadMass
From tblSpaceX where Booster_Version = ‘F9 v1.1’

Using the function AVG works out the average in the column PAYLOAD_MASS_KG_ .

First Successful Ground Landing Date

Date which first successful landing outcome in drone ship was archeived
06-05-2016

Select MIN(Date) SLO from tblSpaceX where Landing_Outcome =”Success(drone ship)”

Using the function MIN works out the minimun date In the column Date, and using the filter “Success(drone ship)”.

Successful Drone Ship Landing with Payload between 4000 and 6000

Date which first successful landing outcome in drone ship was archeived	
0	F9 FT B1032.1
0	F9 B4 B1040.1
0	F9 B4 B1043.1

Select Booster_Version from tblSpaceX where Landing_outcome = ‘Success (ground pad)’ AND Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000

Select only Booster_Version, the Where clause filters The dataset to Landing_Outcome = Success_(drone ship).

The condicions Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000 is the filter.

Total Number of Successful and Failure Mission Outcomes

Successful_Mission_Outcomes	Failure_Mission_Outcomes
100	1

SELECT(SELECT Count(Mission_Outcome)
From tblSpaceX where Mission_Outcome LIKE
‘%Success%’) as Successsful_Mission_Outcomes,
(SELECT Count(Mission_Outcome) from tblSpaceX
Where Mission_Outcome LIKE ‘%Failure%’) as
Failure_Mission_Coutcomes

It is used subqueries here to produce the results. The like wildcard shows that in the record the foo phrase is
In nay part of the string in the record.

Boosters Carried Maximum Payload

```
SELECT DISTINCT Booster_Version, MAX(Payload_Mass_KG)
AS [Maximum Payload Mass] FROM tblSpaceX GROUP BY
Booster_Version ORDER BY [Maximum Payload Mass] DESC
```

There are 96 rows in this query. Using the word DISTINCT in the query means that it will only show Uniques values in the Booster_Version column from TblSpaceX.

	Booster_Version	Maximum Payload Mass
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
...		...
92	F9 v1.1 B1003	500
93	F9 FT B1038.1	475
94	F9 B4 B1045.1	362
95	F9 v1.0 B0003	0
96	F9 v1.0 B0004	0

2015 Launch Records

MONTH	booster_version	launch_site	landing__outcome
2015-10-01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
2015-10-01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)

SELECT DATE AS Month, Booster_Version, Launch_Site, Landing__Outcome from SPACEXTBL WHERE Landing__Outcome LIKE '%Failure%' and DATE > '2015-01-01' order by DATE

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

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2017-03-06	21:07:00	F9 FT B1035.1	KSC LC-39A	SpaceX CRS-11	2708	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-03-06	21:07:00	F9 FT B1035.1	KSC LC-39A	SpaceX CRS-11	2708	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-01-05	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
2017-01-05	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
2016-08-04	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (drone ship)
2016-08-04	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (drone ship)
2016-06-05	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-06-05	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)

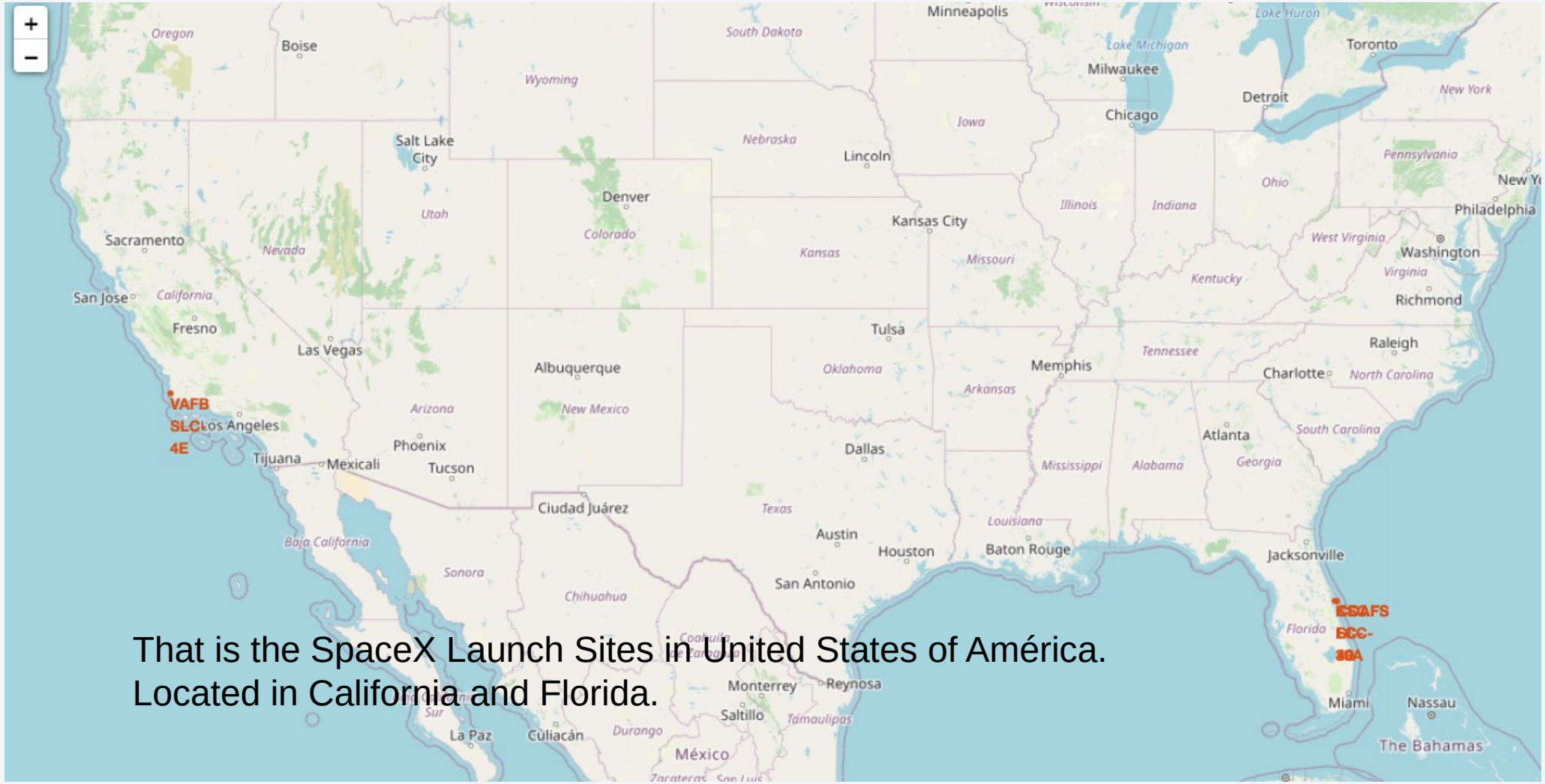
```
select * from SPACEXTBL where Landing__Outcome like 'Success%' and
(DATE between '2010-06-04' and '2017-03-20') order by date desc
```

In this Query, it is used the filter like for the Success and the Date is used between 2010-06-04 and 2017-03-2020.

Section 4

Launch Sites Proximities Analysis

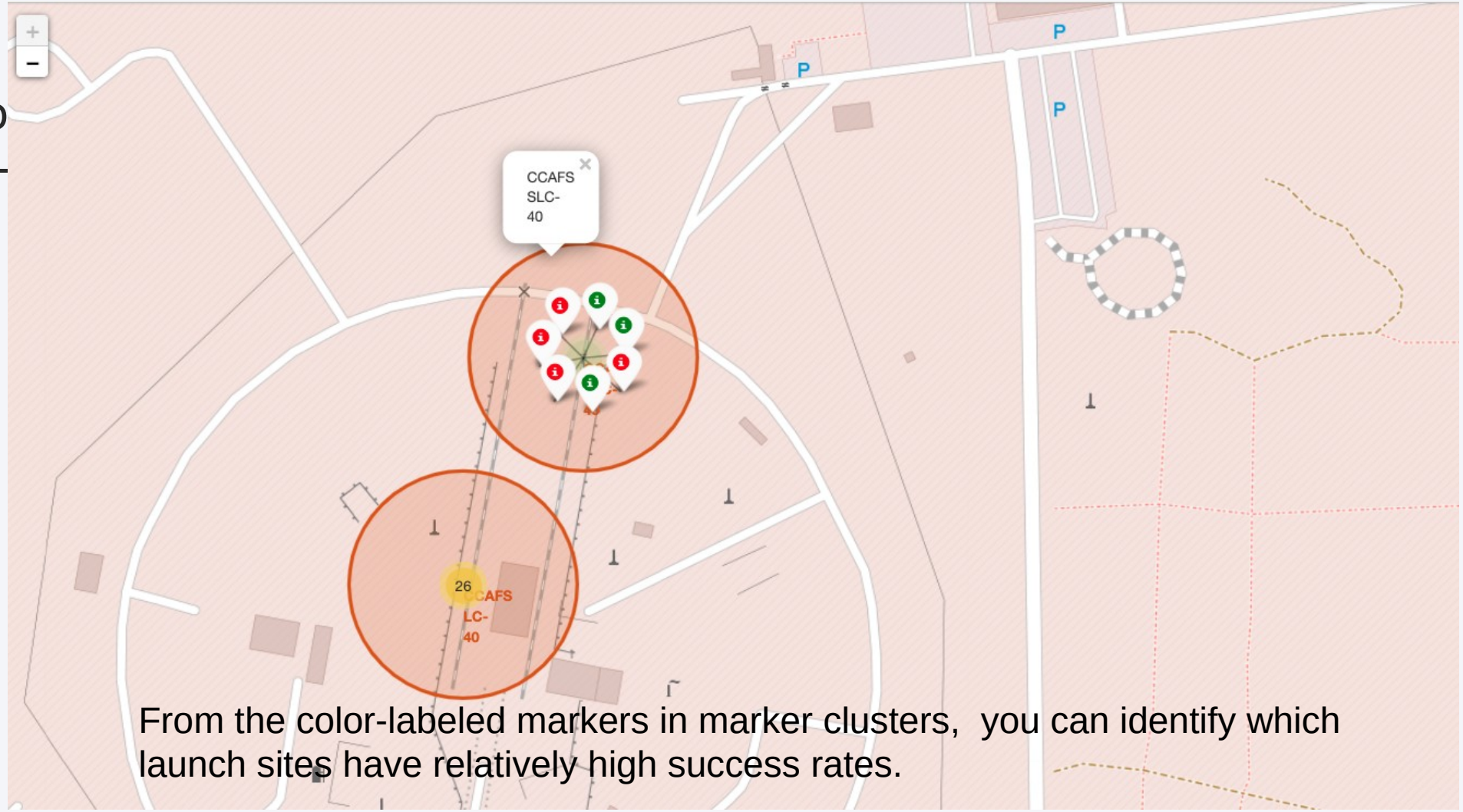
All Launch Sites Global Maps Markers.



That is the SpaceX Launch Sites in United States of América.
Located in California and Florida.

Markets.

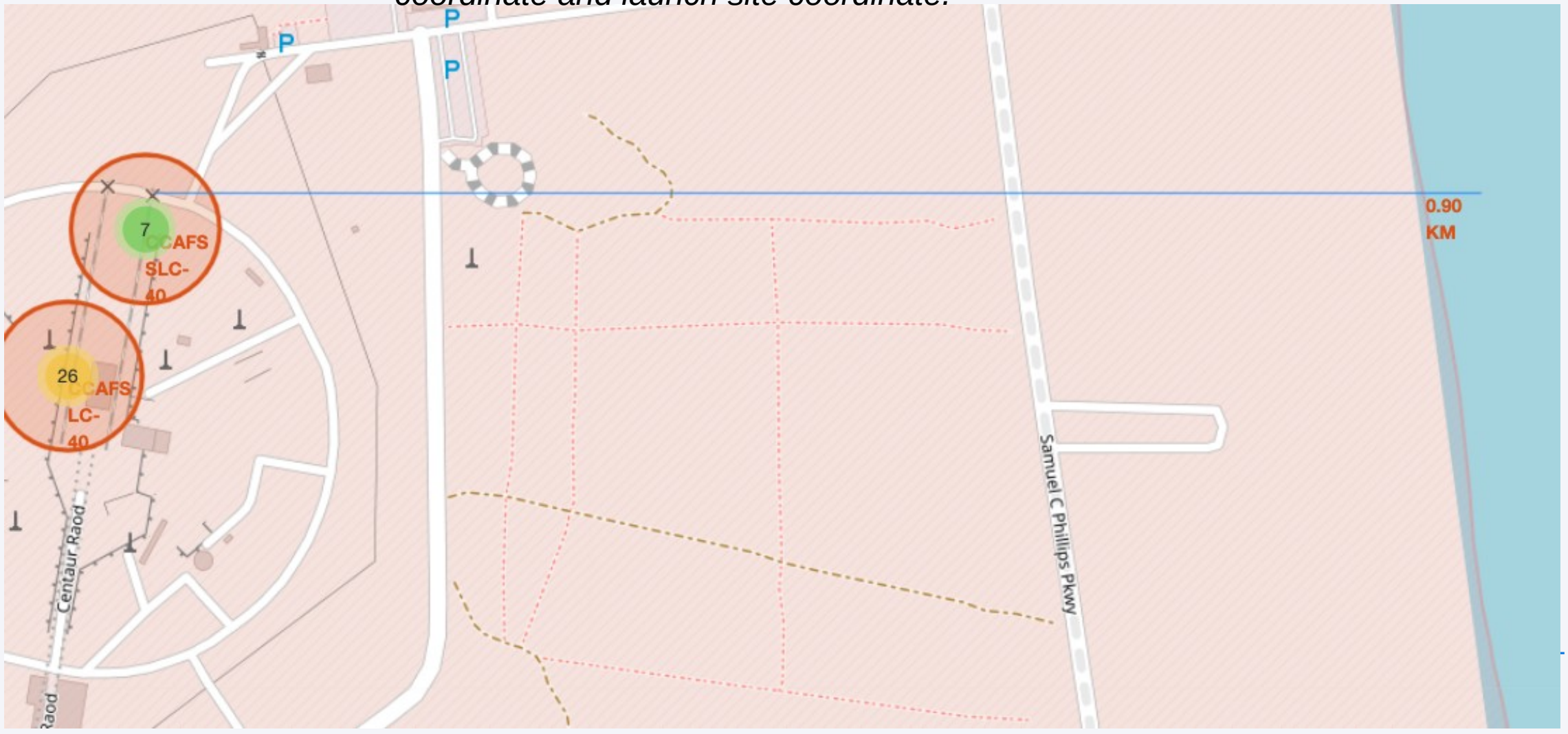
Explo
color-

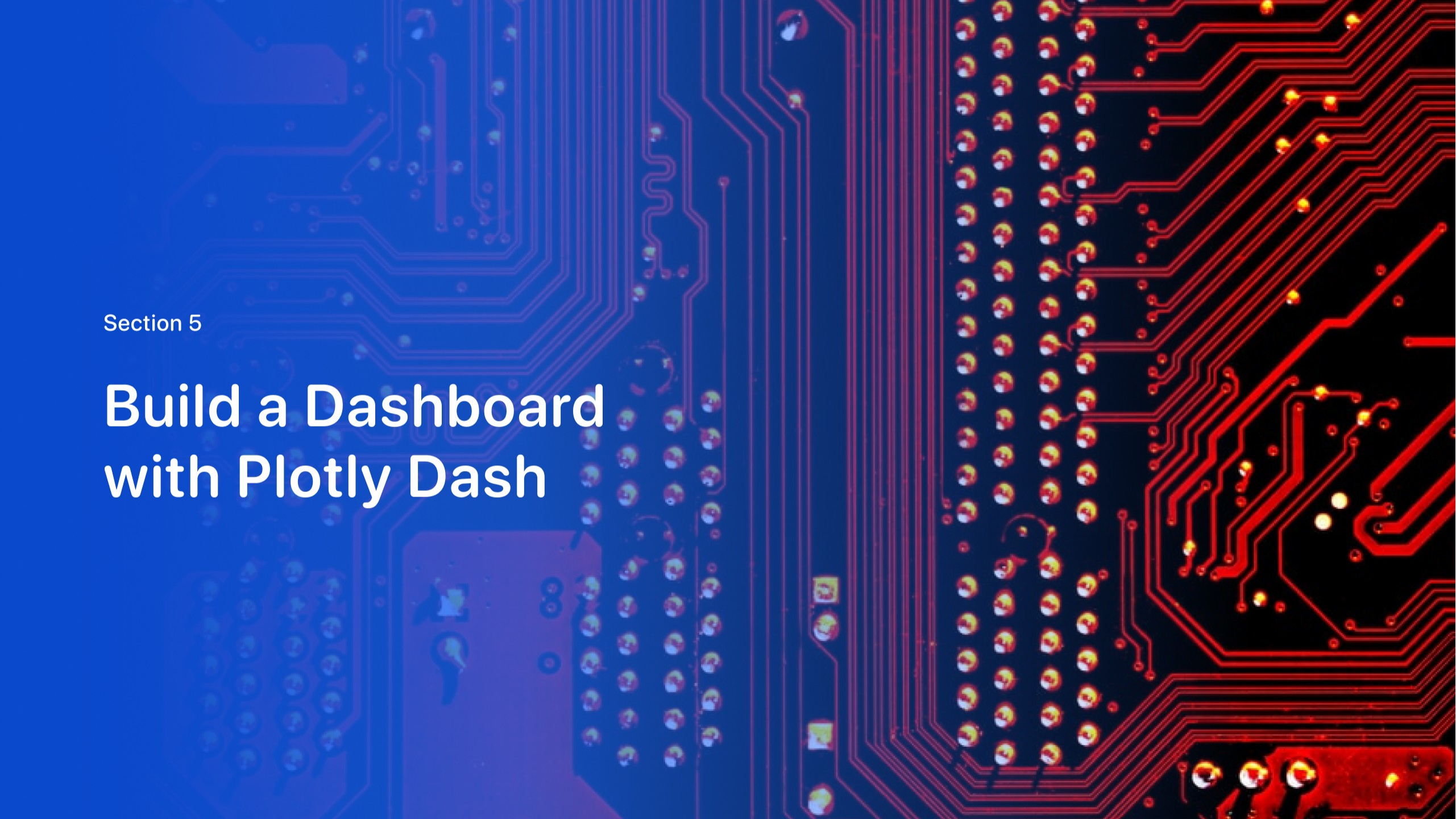


From the color-labeled markers in marker clusters, you can identify which launch sites have relatively high success rates.

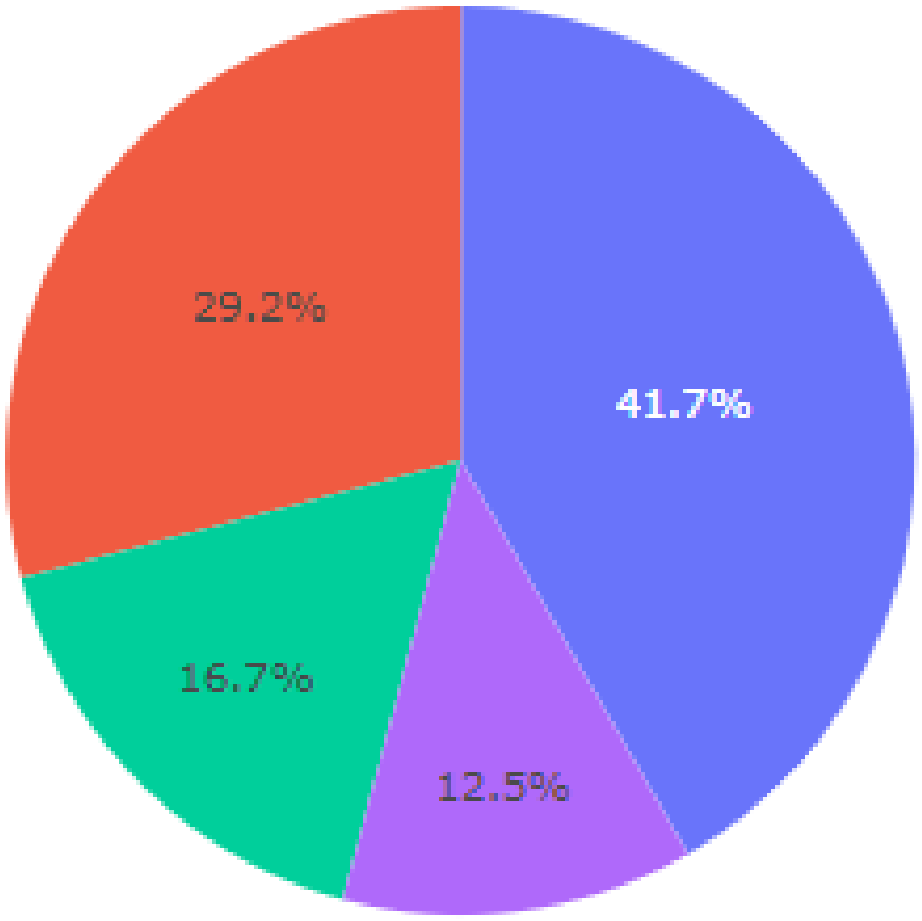
Railway Point.

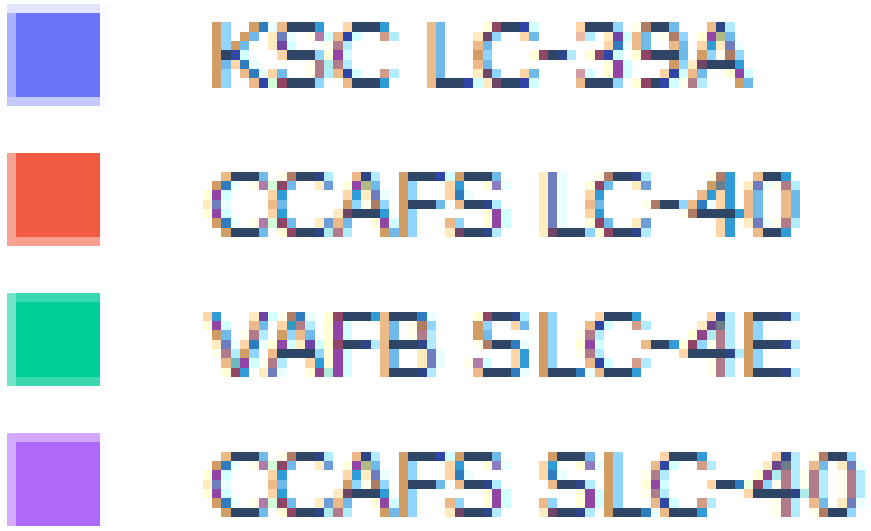
In this image '*folium.PolyLine' object using the railway point coordinate and launch site coordinate.*




Section 5
Build a Dashboard with Plotly Dash

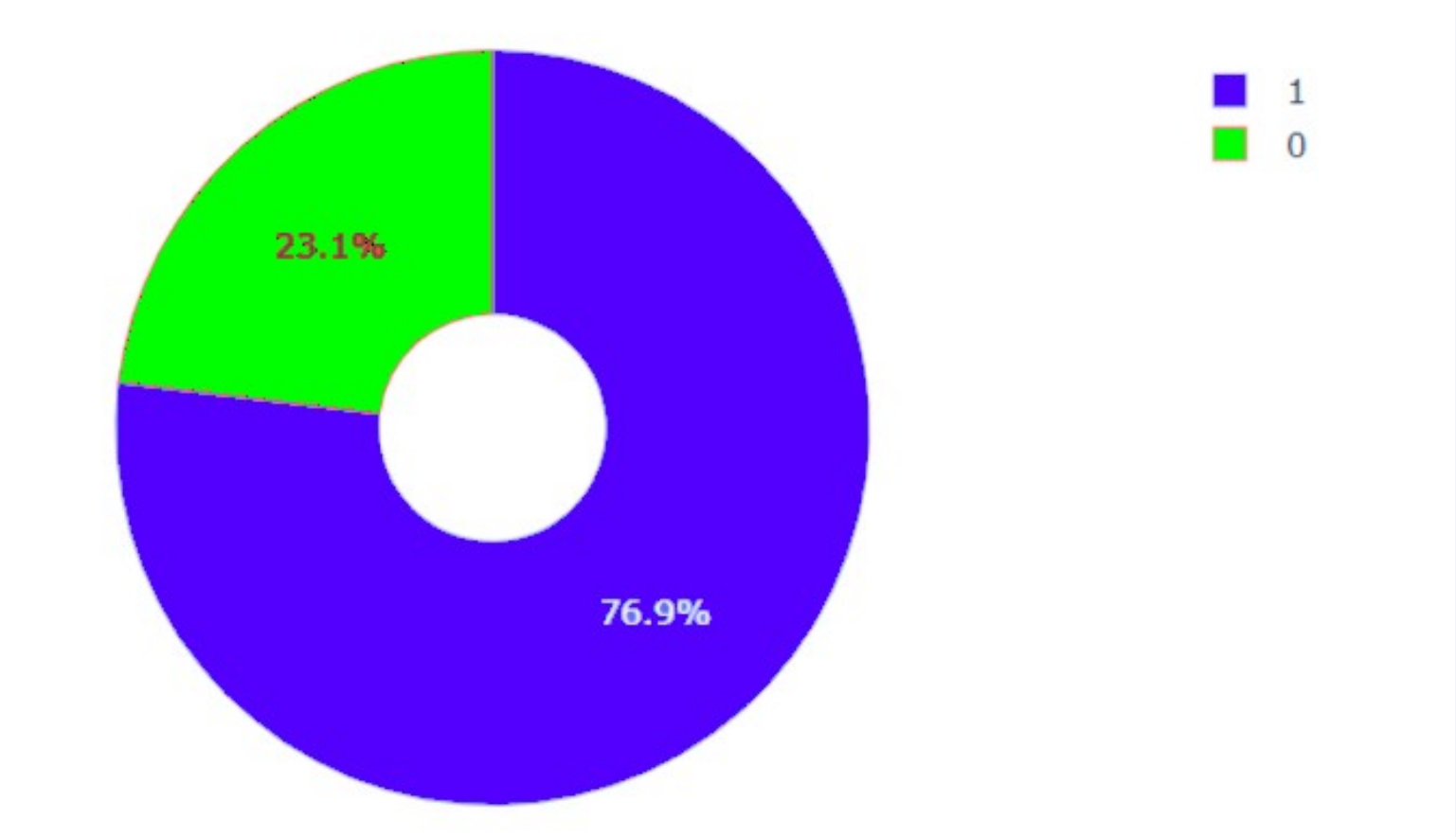
Total Success Launches By Site.





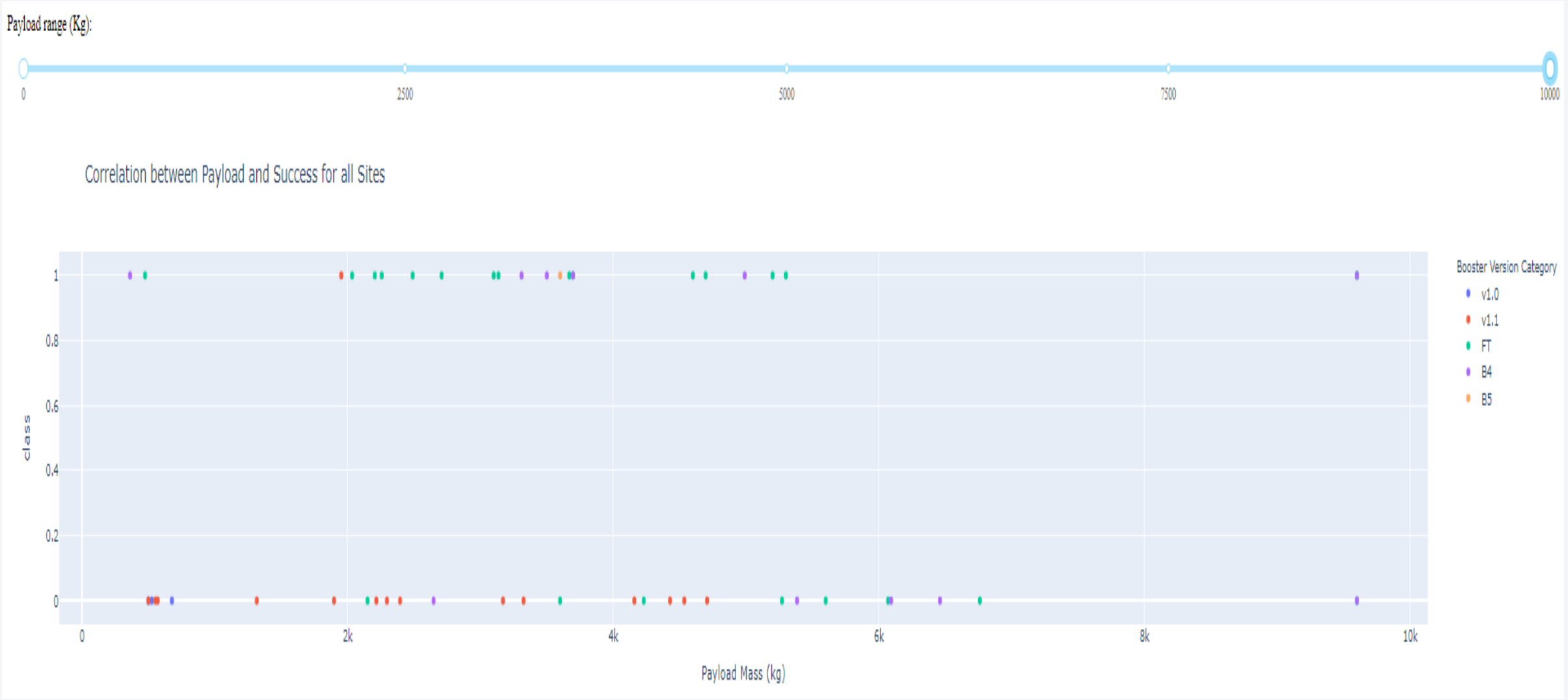
In this graphic, the most successful launches from All the sites is KSC LC-39A

Highest Launch Success Ratio.



KSC LC-39A has a success rate of 76,9% meanwhile getting a 23.1% failture rate.

Payload and Success for all Sites.



Section 6

Predictive Analysis (Classification)

Classification Accuracy

The accuracy is extremely close but i do have
A winner for decimal places, using the function:

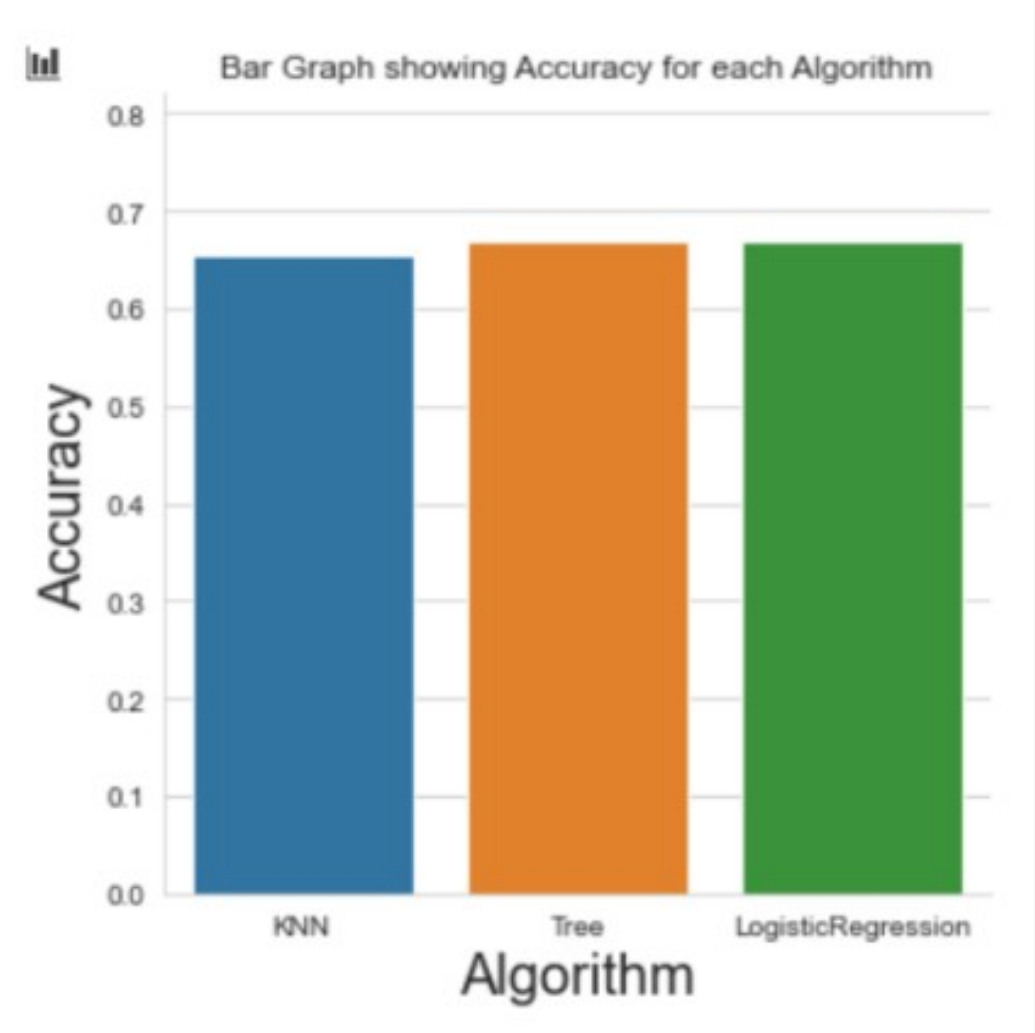
Best algorithm = max(algorithms, key=algorithms.get)

	Accuracy	Algorithm
0	0,653571	KNN
1	0,667857	Tree
2	0,667857	LogisticRegression

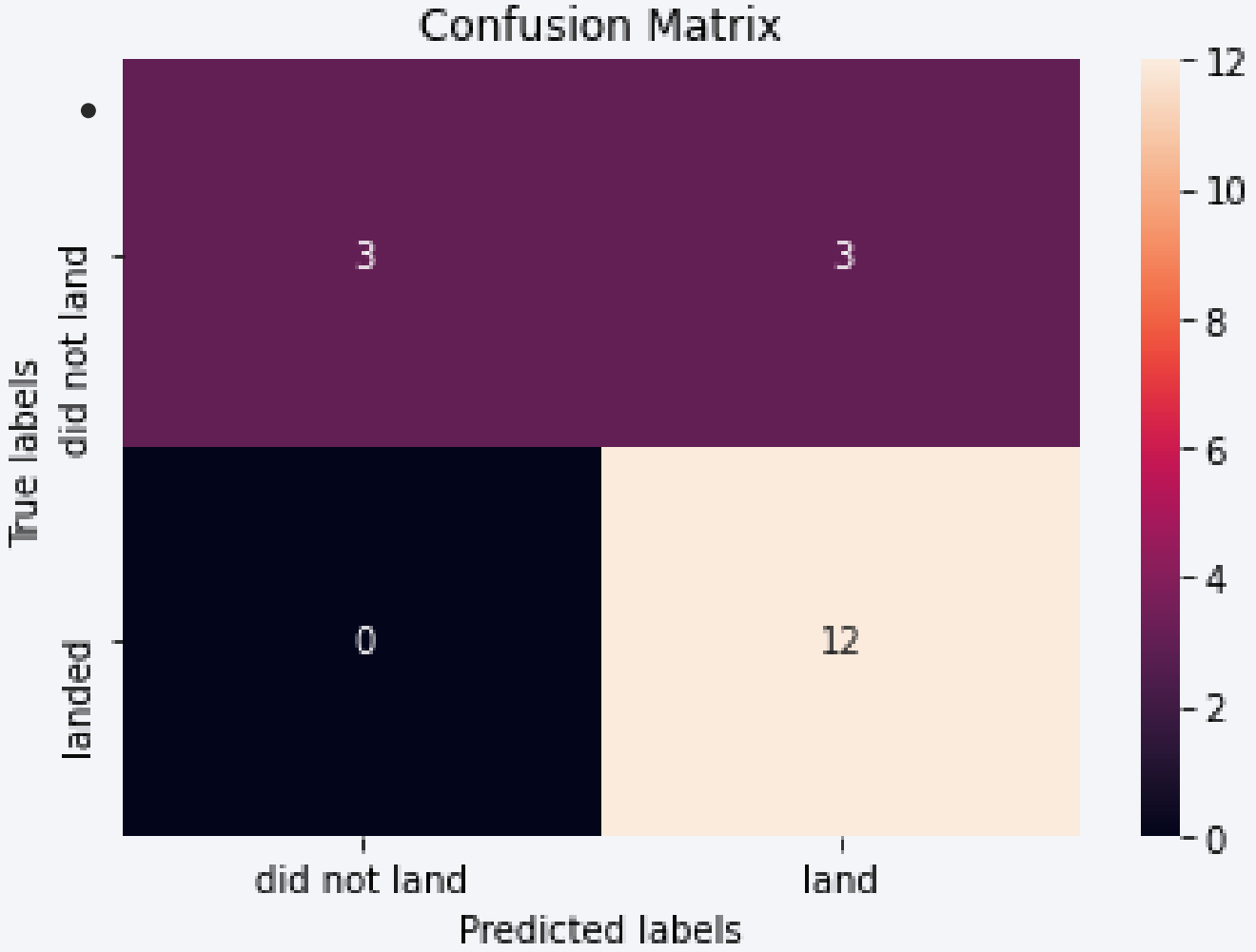
The best algorithm is the tree with a score of

0,6678571428571429

After selecting the best hyperparameters for the Decision tree classifier using the validation data, the Accuracy on the test data is 83.33%.



Confusion Matrix



Examining the confusion matrix,
It is possible that logistic regression
can distinguish between
the different classes.

I can see that the major problem
is false positives.

Conclusions

- The tree classifier algorithm is the best option for Machine Learning in this dataset.
- Low weighted payloads perform is better than the heavier payloads in the dataset.
- KSC LC-39A had the most successful launches from all the sites.
- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches.

Appendix

- Python for Everybody.
- Haversine Formula.

Python for Everybody

- Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

- Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

- http://www.py4e.com

Haversine Formula

- The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. Important in navigation, it is a special case of a more general formula in spherical trigonometry, the law of haversines, that relates the sides and angles of spherical triangles.

$$\operatorname{hav}(\theta) = \sin^2\left(\frac{\theta}{2}\right) = \frac{1-\cos(\theta)}{2}$$

Thank you!