IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Thiago Viana
18/04/2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection
  - Data Wrangling
  - EDA with Data Visualization and SQL
  - Interactive Maps with Folium
  - Dashboard with Plotly
  - Predictive analysis for Classification Models

- Summary of all results
- Data Analysis results
- Predictive Analysis Models and Results

Introduction

- Project background and context
  - 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. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.
- Problems you want to find answers
  - Determine if the first stage will land successfully;
  - The effect of each relationship of rocket variables on the success of landing;
  - The conditions which will give SpaceX the best results.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Via SpaceX Rest API;
  - Web Scrapping from Wikipedia.
- Perform data wrangling
  - One hot encoding data fields for Machine Learning and dropping irrelevant columns.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Plotting Scatter Graphs and Bar Graphs to show patterns between data.
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection – SpaceX API

1. Getting Response from API

```
In [20]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [21]: response = requests.get(spacex_url)
```

2. Converting Response to a .json file

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

3. Apply custom functions to clean data

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

4. Assign list to dictionary the create dataframe

```
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}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
# Create a data from launch_dict
df = pd.DataFrame(launch_dict)
```

5. Filter dataframe and export to flat file

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

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

data_falcon9.head()
```

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

Data Collection - Scraping

Getting response from HTML

```
data  = requests.get(static_url).text
```

Creating Beautiful Object

```
soup = BeautifulSoup(data, 'html5lib')
```

Finding tables

```
html_tables=soup.find_all("table")
html_tables
```

Getting column names

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

Creation of dictionary and appending data to keys

```
launch_dict= dict.fromkeys(column_names)
```

Converting dictionay to dataframe

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

Dataframe to .CSV

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

Data Wrangling

This is the process of cleaning and unifying messy and complex dataset for a cleaning one, permitting an easy access and analysis.

Calculate the number of launches at each site

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

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
```

Calculate the number and occurrence of each orbit

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

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
```

Calculate the 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
False Ocean     2
None ASDS       2
False RTLS      1
```

Create landing outcome label from Outcome column

```
landing_class = []
for i in df['Outcome']:
    if i in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

```
df['Class']=landing_class
df[['Class']].head(8)
```

Export dataset as .CSV

```
df.to_csv("dataset_part\_2.csv", index=False)
```

EDA with Data Visualization

This is an approach to analysing datasets and summarizing their mains characteristics

- Scatter Graphs have been Drawn:
  - Flight number x Launch Site
  - Payload x Launch Site
  - Flight number x Orbit type
  - Payload x Orbit Type

Scatter plots were used to show how much one variable is affected by other.

- Bar Graph
  - Success rate x Orbit type

Bar Graphs are easy to interpret and understand the relationship between different attributes.

- Line Graph
  - Launch success x Yearly trend

Line graph show trends clearly and can help to make predictions about the data not yet recorded.

EDA with SQL

SQL queries were performed to gather information from the given dataset:

- Display 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 ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

Build an Interactive Map with Folium

Folium is a Python library used for visualizing geospatial data. It is easy to use and yet a powerful library. Folium is a Python wrapper for Leaflet.

- Map objects used:
    - Map marker: To make a mark on map
    - Icon marker: To create an icon on map
    - Circle marker: To create a circle where the Marker is placed
    - Polyline: To create a line between points
    - Marker Cluster Object: Is a good way to simplify a map containing markers that have the same coordinate
    - AntPath: To create an animated line between points

Build a Dashboard with Plotly Dash

Pie chart was used to show the total launches including all sites or certain sites

- Displays the percentage of success in relation to launch site
- Displays relative proportions of multiples classes of data

Scatter Graph was used to show the correlation between payload and success for all sites or by certain sites

- Shows the relationship between success rate and booster version category
- Shows the range of data (maximum and minimum)
- It is a good way to display a non-linear pattern

Predictive Analysis (Classification)

- Building a Classification Model
  - Load the dataset
  - Transform the data into NumPy arrays
  - Standardize
  - Split the data into training and test
  - Check how many test samples has been created
  - Choose how machine learning algorithms will be used
  - Set the parameters to GridSearchCV
  - Fit the datasets into the GridSearchCV objects and trains the dataset
- Evaluating model
  - Check the accuracy for each model
  - Get the best hyperparameters for each type of algorithms
  - Plot Confusion Matrix
- Improving model
  - Feature engineering
- Finding the best classification model
  - The best accuracy scores will show the best performing model

Results

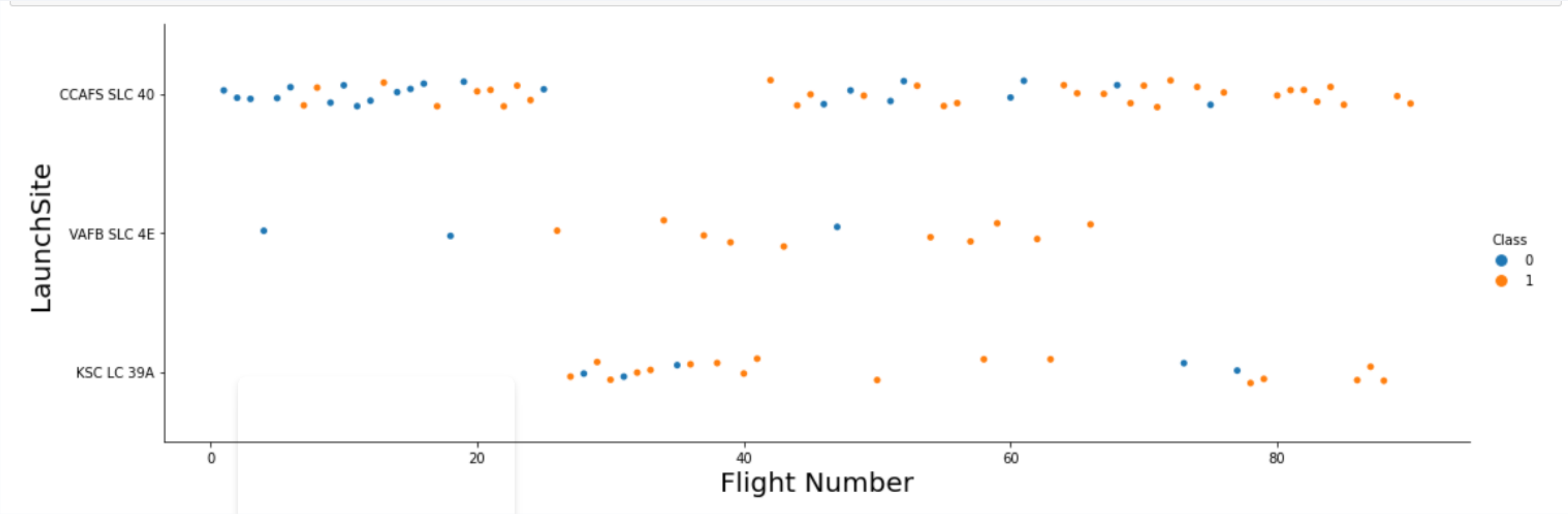
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

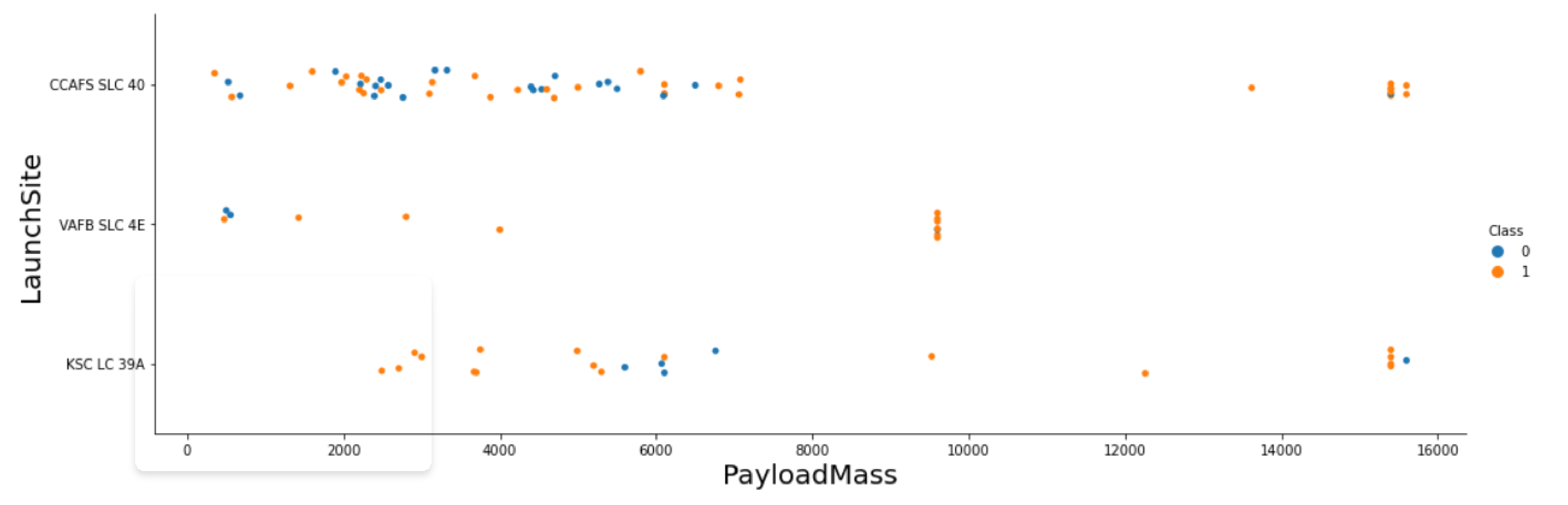
Flight Number vs. Launch Site

- Scatter plot of Flight Number vs. Launch Site
  - The higher the number of flights, the higher the success rate at a launch site



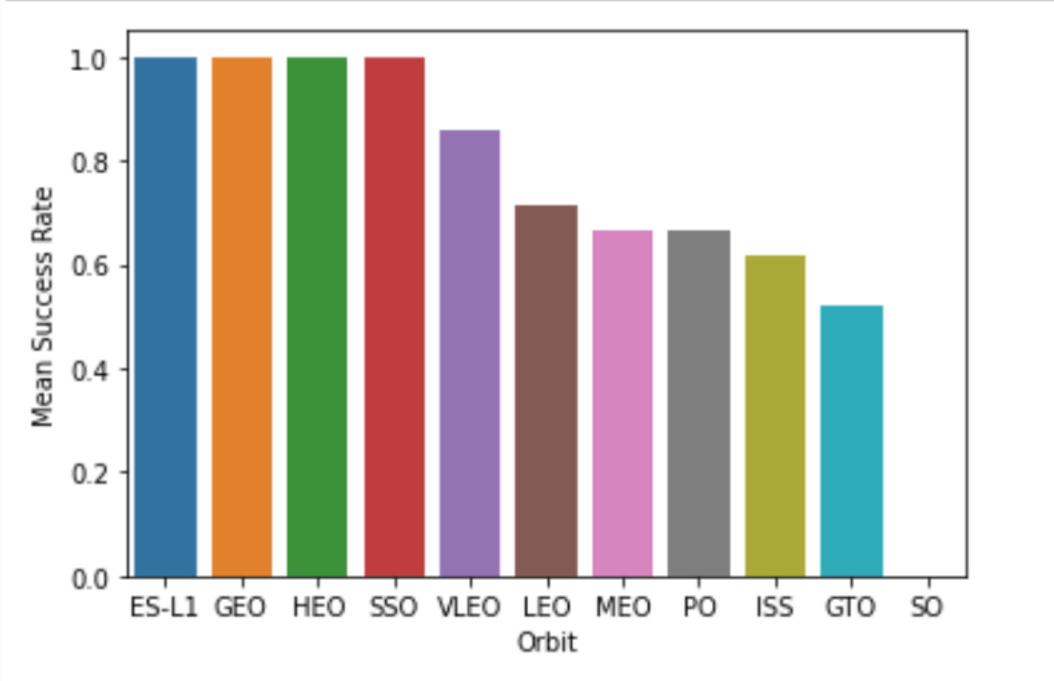
Payload vs. Launch Site

- Scatter plot of Payload vs. Launch Site
  - The higher the payload mass, the higher the success rate for the Rocket. There is no quite clear pattern to be found, using this visualization, to make a decision if the Launch Site is dependant on Payload mass for a success launch.



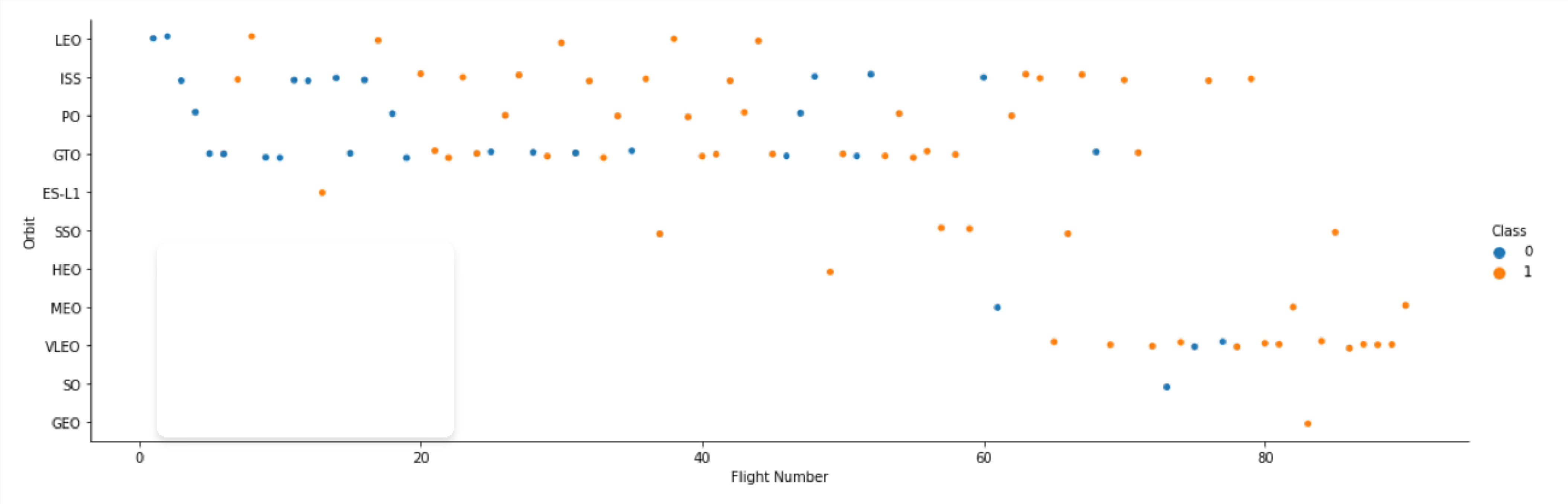
Success Rate vs. Orbit Type

- Bar chart for the success rate of each orbit type
  - ES-L1, GEO, HEO and SSO has the highest success rates.



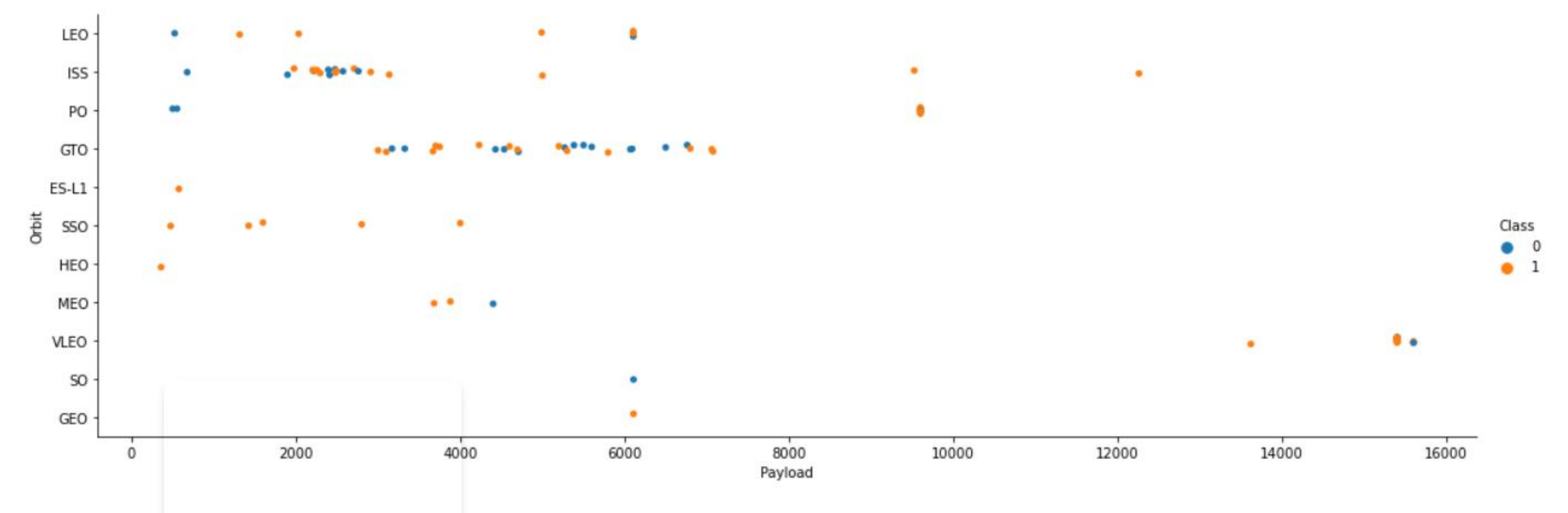
Flight Number vs. Orbit Type

- Scatter point of Flight number vs. Orbit type
  - Leo orbit success is related with the number of flights and, there is no relationship between flight number when in GTO orbit



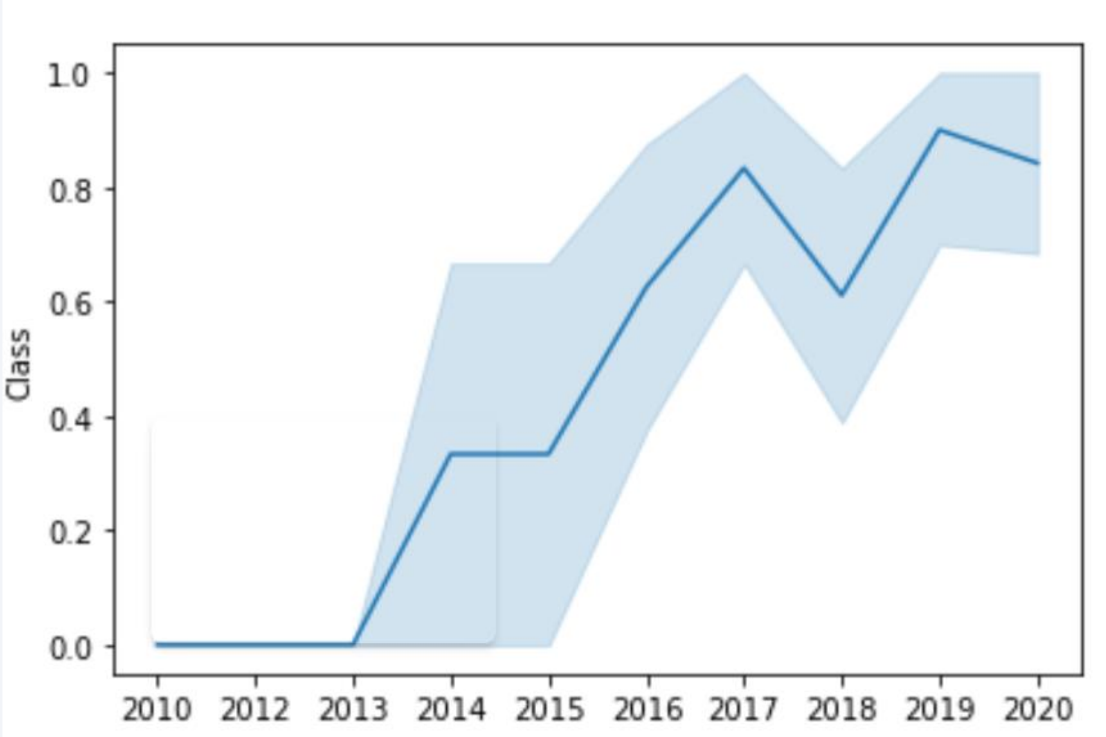
Payload vs. Orbit Type

- Scatter point of 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

- Line chart of yearly average success rate
  - The success rate since 2013 kept increasing till 2020



EDA with SQL

All Launch Site Names

- Names of the unique launch sites
  - Using DISTINCT:

```
%sql SELECT DISTINCT launch_site as "Launch_Sites" FROM SPACEXTBL;
```

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

Launch Site Names Begin with 'CCA'

- Finding 5 records where launch sites begin with `CCA`

```
%sql SELECT * from SPACEXTBL WHERE launch_site like 'CCA%' LIMIT 5;
```

DATE	Time (UTC)	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	Landing _Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	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)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Calculating the total payload carried by boosters from NASA

```
%sql SELECT sum(payload_mass__kg_) as "Total_Payload_Mass" from SPACEXTBL WHERE customer = 'NASA (CRS)';
```

Total_Payload_Mass
45596

Average Payload Mass by F9 v1.1

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

```
%sql SELECT avg(PAYLOAD_MASS__KG_) as "AVG_Payload_Mass" from SPACEXTBL WHERE booster_version like 'F9 v1.1%'
```

AVG_Payload_Mass
2534

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
%sql SELECT min(DATE) as "First_Successfull_Landing" from SPACEXTBL WHERE Landing _Outcome = "Success (ground pad)";
```

First_Successfull_Landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```
%sql SELECT booster_version from SPACEXTBL WHERE Landing _Outcome = 'Success (drone ship)'
and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

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

Total Number of Successful and Failure Mission Outcomes

- Total number of successful and failure mission outcome

```
%sql select mission_outcome, count(*) from SPACEXTBL GROUP BY mission_outcome
```

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- Names of the booster which have carried the maximum payload mass

```
%sql select booster_version, payload_mass__kg_ from SPACEXTBL
WHERE payload_mass__kg_ = (SELECT max(payload_mass__kg_) from SPACEXTBL)
```

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

- Failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
%sql SELECT landing__outcome, booster_version, launch_site from SPACEXTBL1 WHERE Date like '2015%' \
and landing__outcome like "Failure%"
```

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- Ranking landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

```
%sql SELECT landing__outcome, count(*) from (SELECT * from SPACEXTBL1 WHERE Date BETWEEN '2010-06-04'\
and '2017-03-20') GROUP BY landing__outcome ORDER BY landing__outcome desc
```

landing__outcome	2
Uncontrolled (ocean)	2
Success (ground pad)	3
Success (drone ship)	5
Precluded (drone ship)	1
No attempt	10
NASA (COTS) NRO	1
Failure (parachute)	1
Failure (drone ship)	5
Controlled (ocean)	3

Section 3

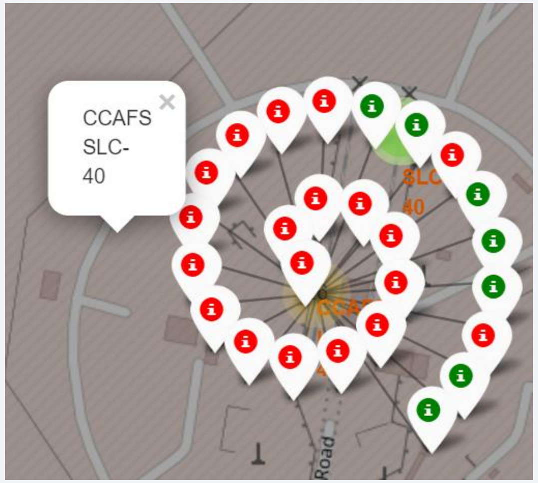
Launch Sites Proximities Analysis

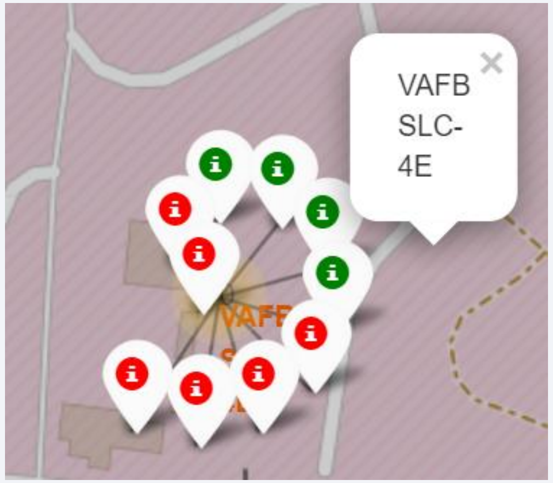
Launch sites on Folium Map

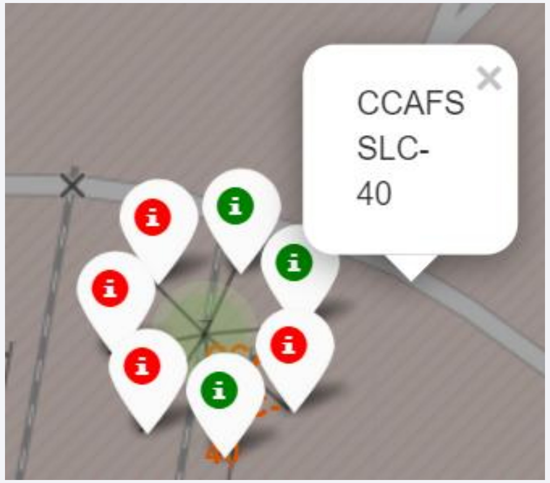
- It is possible to see that the launch sites are near to U.S. coast

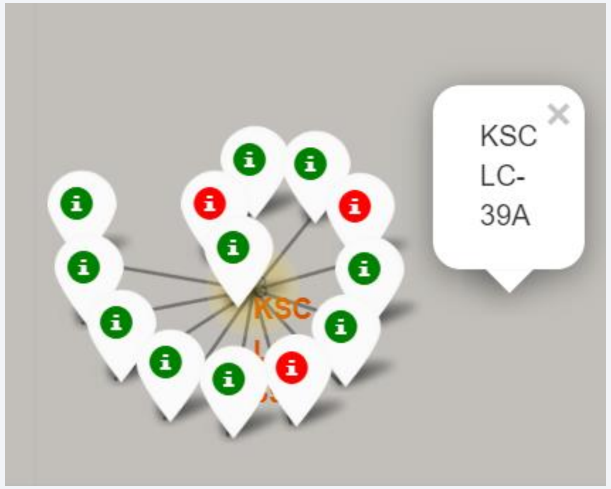
Color-labeled launch records

Green marker: Successful launches
Red marker: Failures

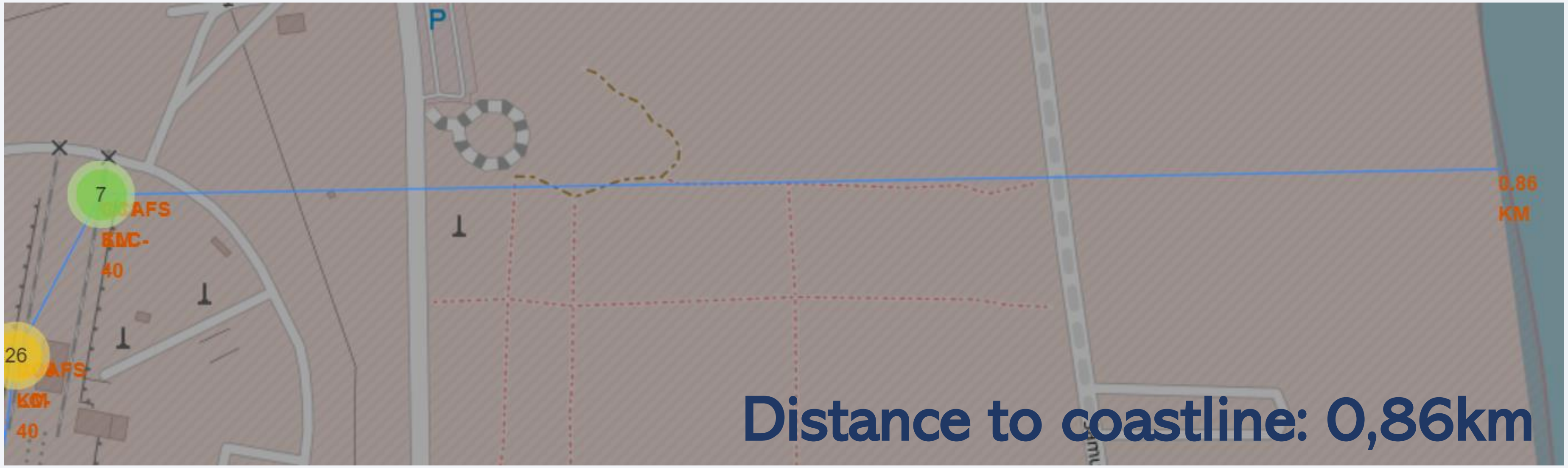


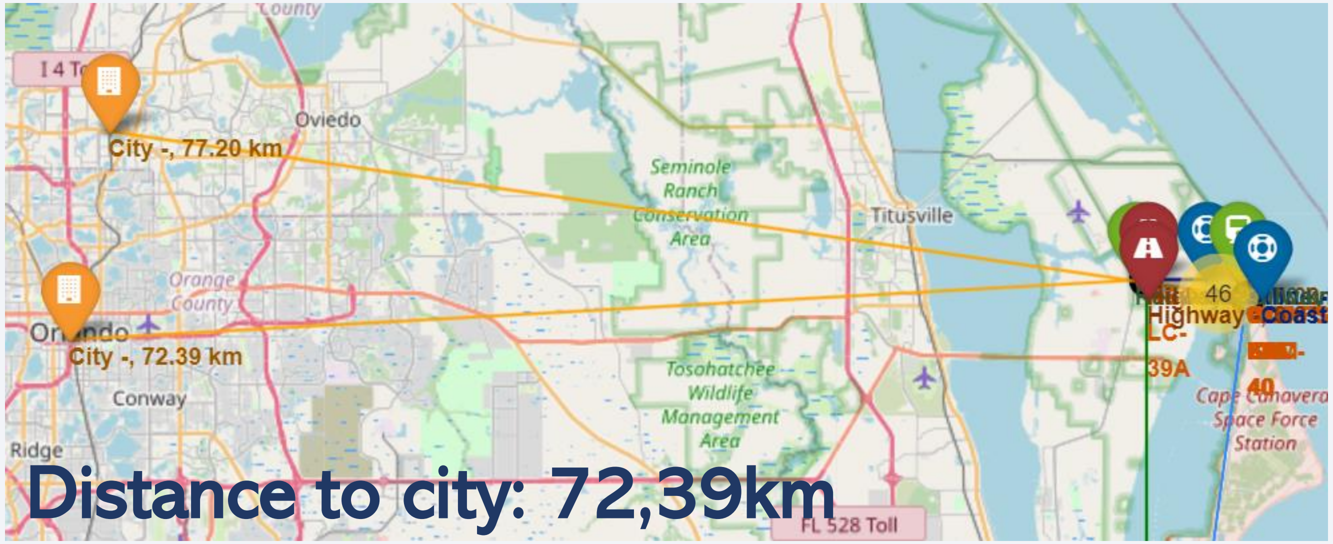




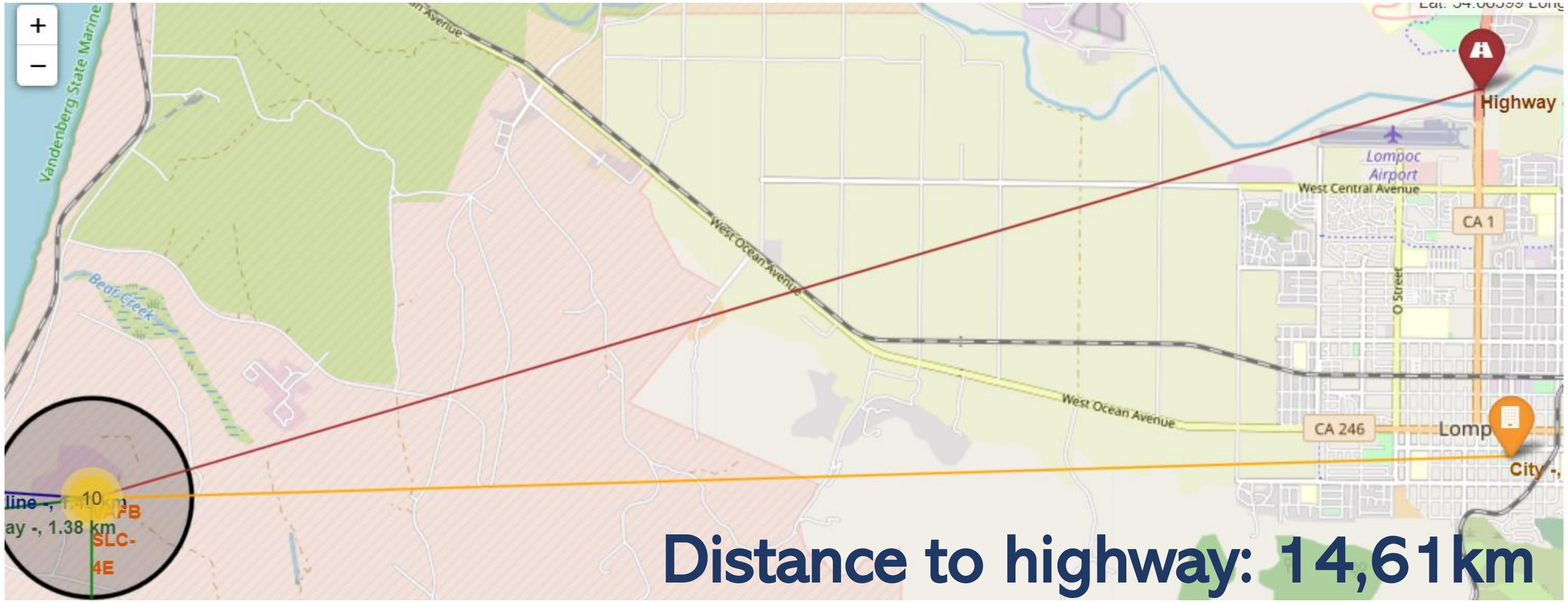


Site distances from city and, coastline





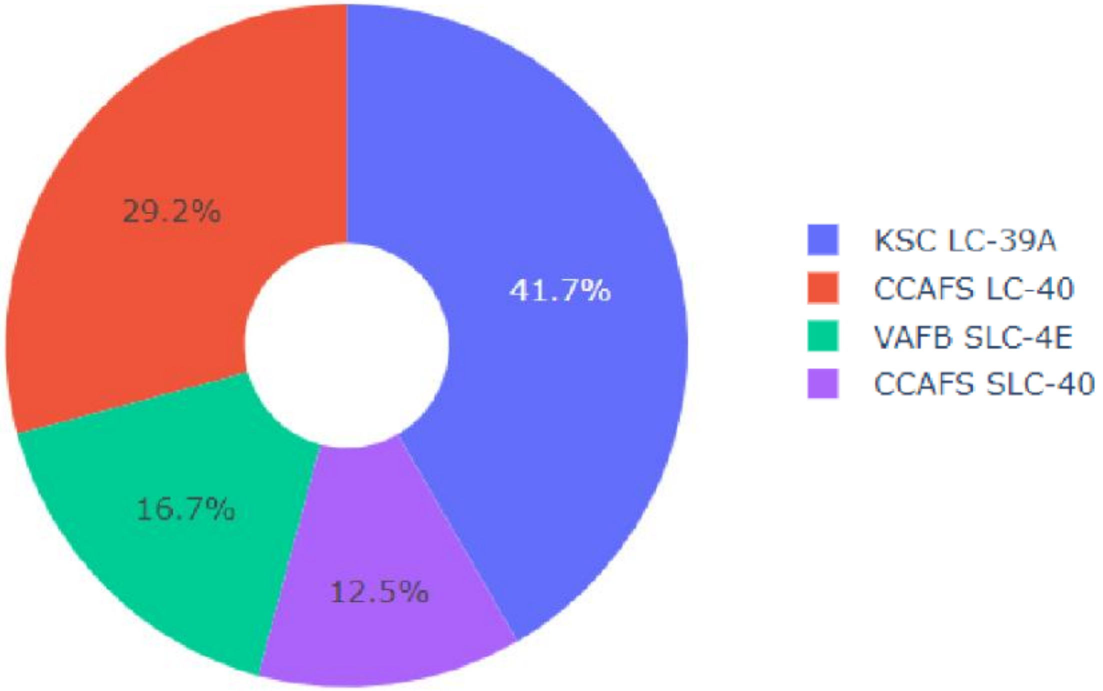
Site distances from railway and highway



Section 4

Build a Dashboard with Plotly Dash

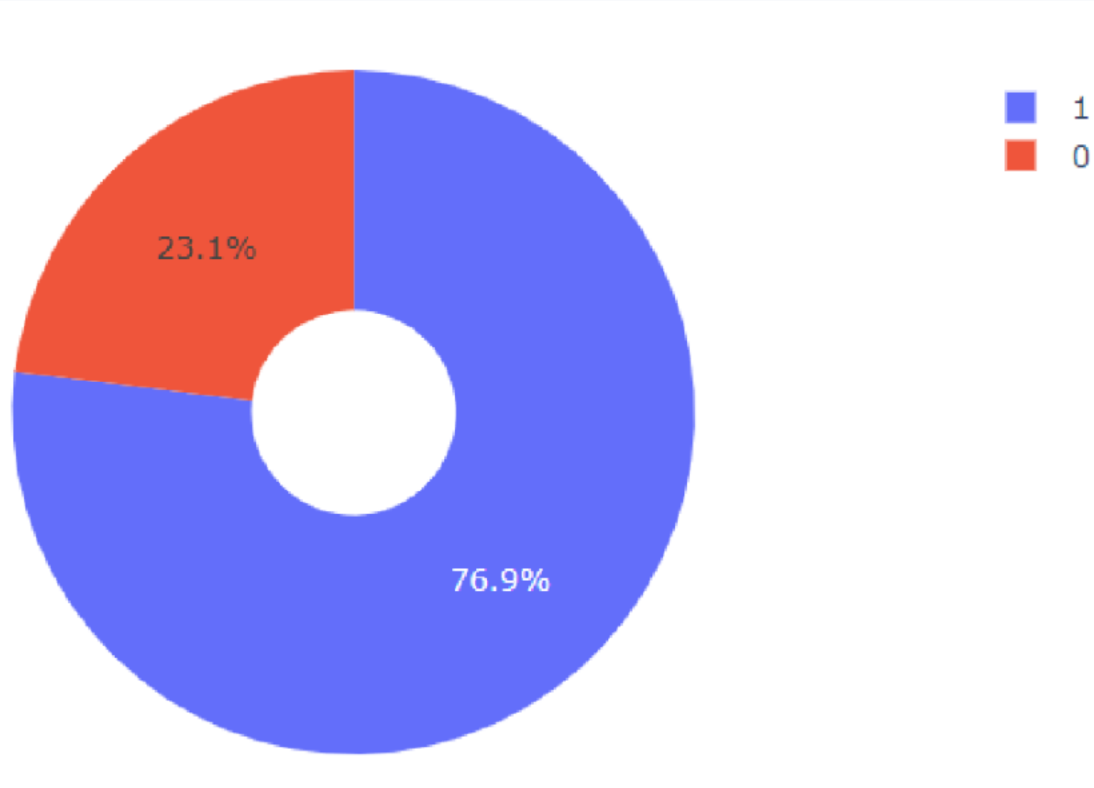
Launch success for all sites



The most successfull launches are from KSC LC – 39A

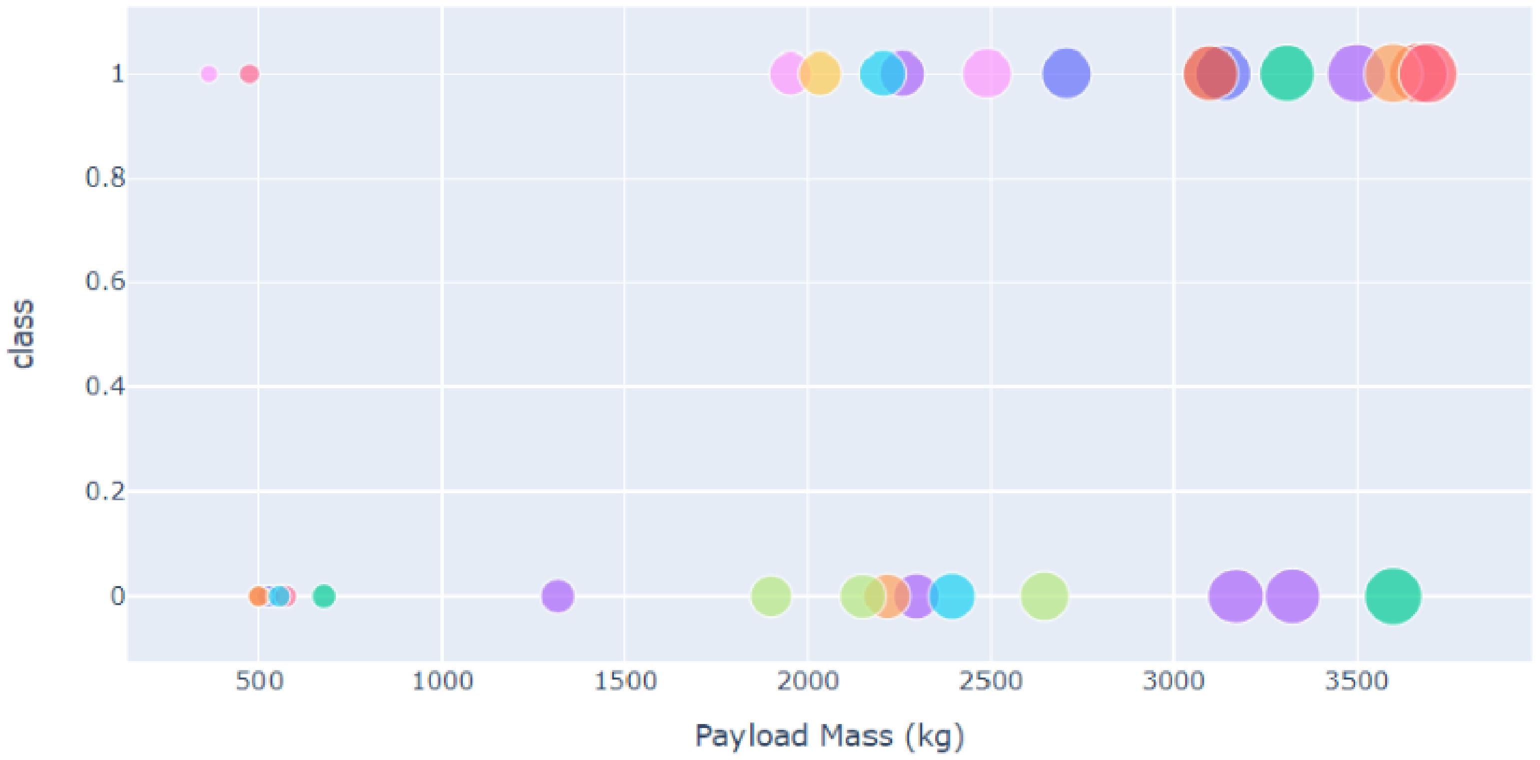
Launch site with highest launch success ratio

- With 76,9 success ratio, KS LC – 39A has the highest launch success.



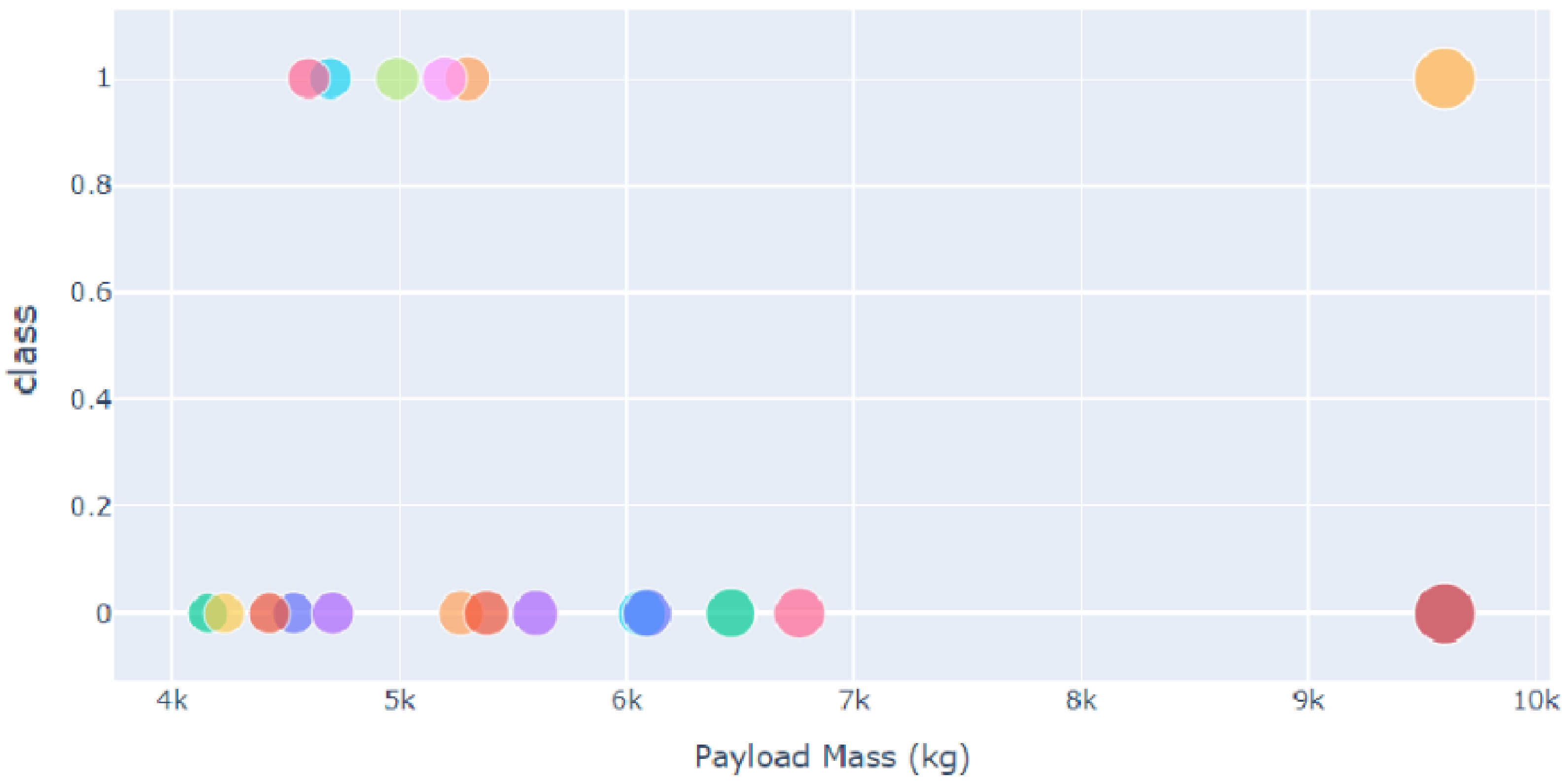
Payload vs. Launch Outcome scatter plot for all sites

Low weighted payload: 0 to 4000kg



Payload vs. Launch Outcome scatter plot for all sites

Heavy weighted payload: 4000 to 10000kg



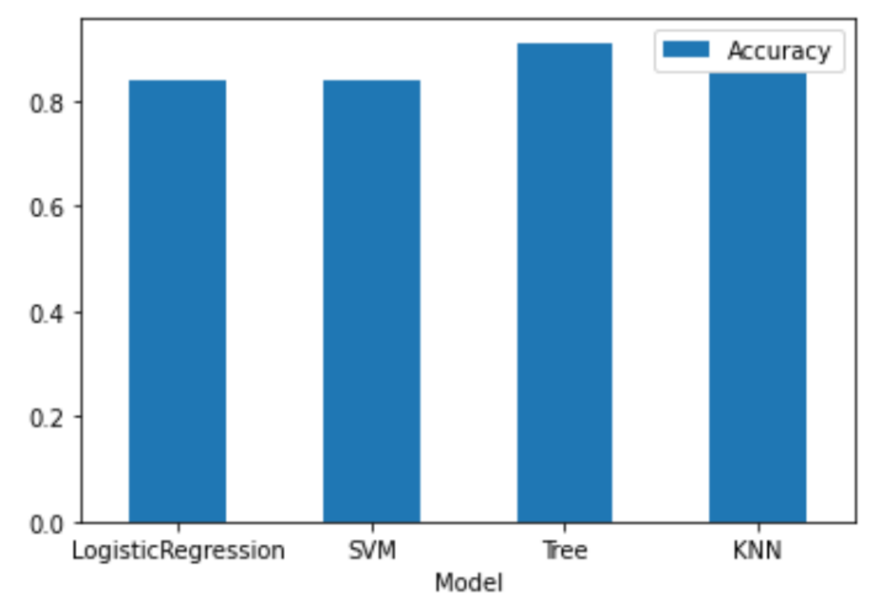
Section 5

Predictive Analysis (Classification)

Classification Accuracy

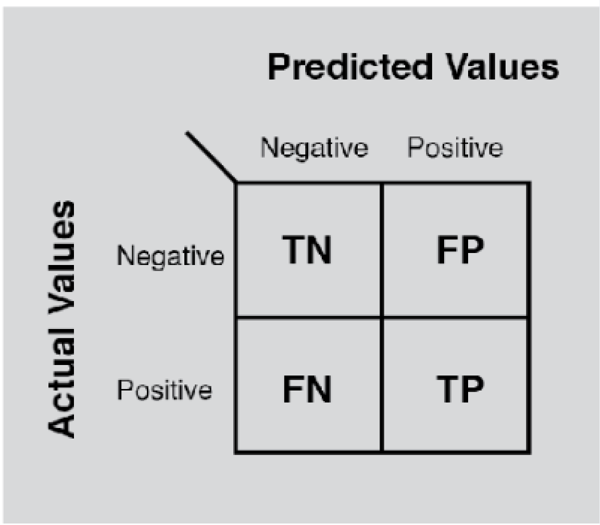
- Built model accuracy for all built classification models, in a bar chart
  - Decision Tree performs best than other models.

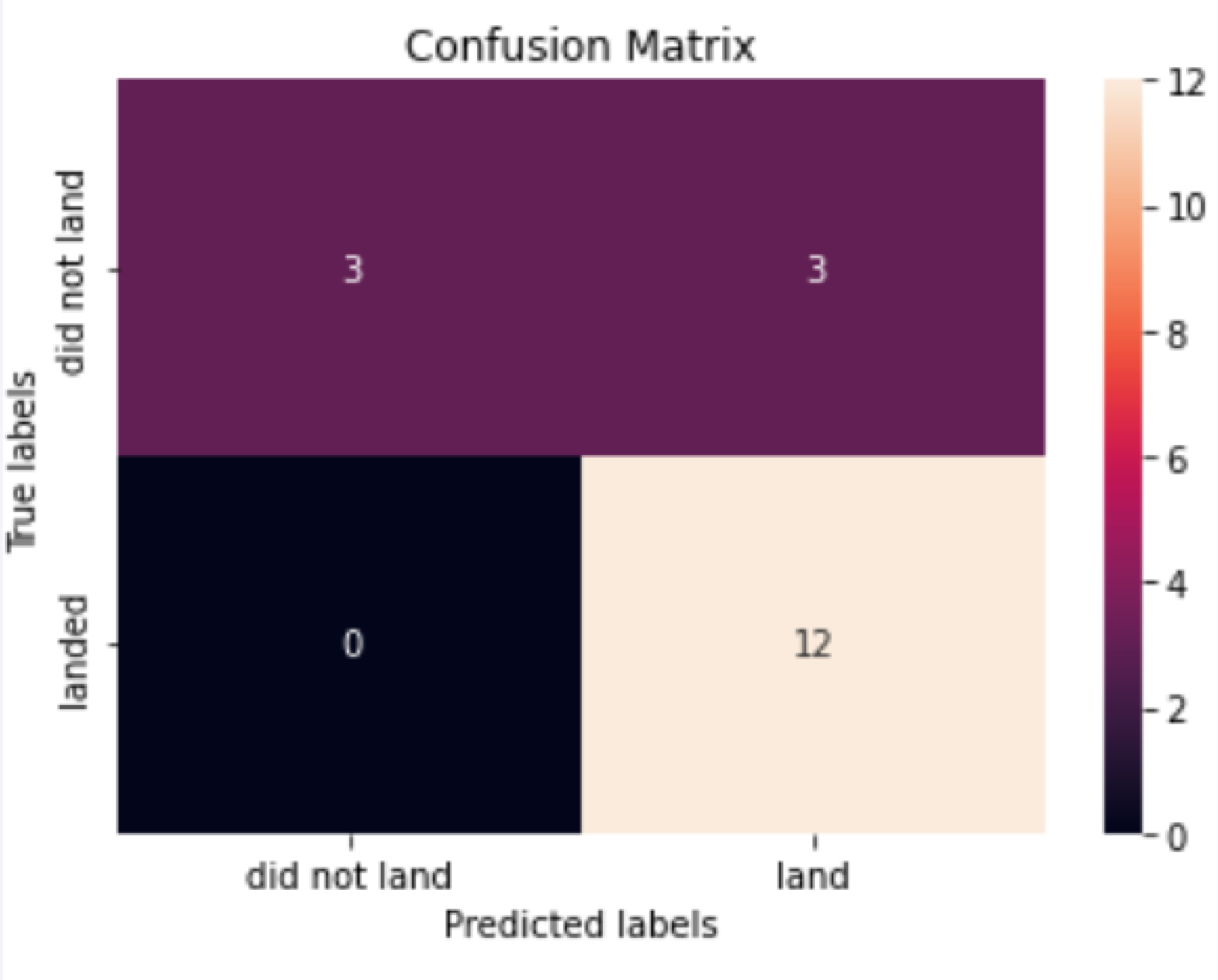
	Model	Accuracy	Prediction score
0	LogisticRegression()	0.8464285714285713	0.8333333333333334
1	LogisticRegression()	0.8464285714285713	0.8333333333333334
2	DecisionTreeClassifier()	0.9035714285714287	0.7222222222222222
3	KNeighborsClassifier()	0.8482142857142858	0.8333333333333334



Confusion Matrix

- The confusion matrix can distinguish between different classes. It differentiates between positives, negatives, false positives, and false negatives.





Conclusions

- For this dataset, Tree Classifier is the best algorithm for machine learning;
- Low weighted payloads perform better than heavy payloads;
- KSC LC – 39A had the most successful launches among all sites;
- Orbit GEO, HEO, SSO, ESS-L1 has the best success rate.

Appendix

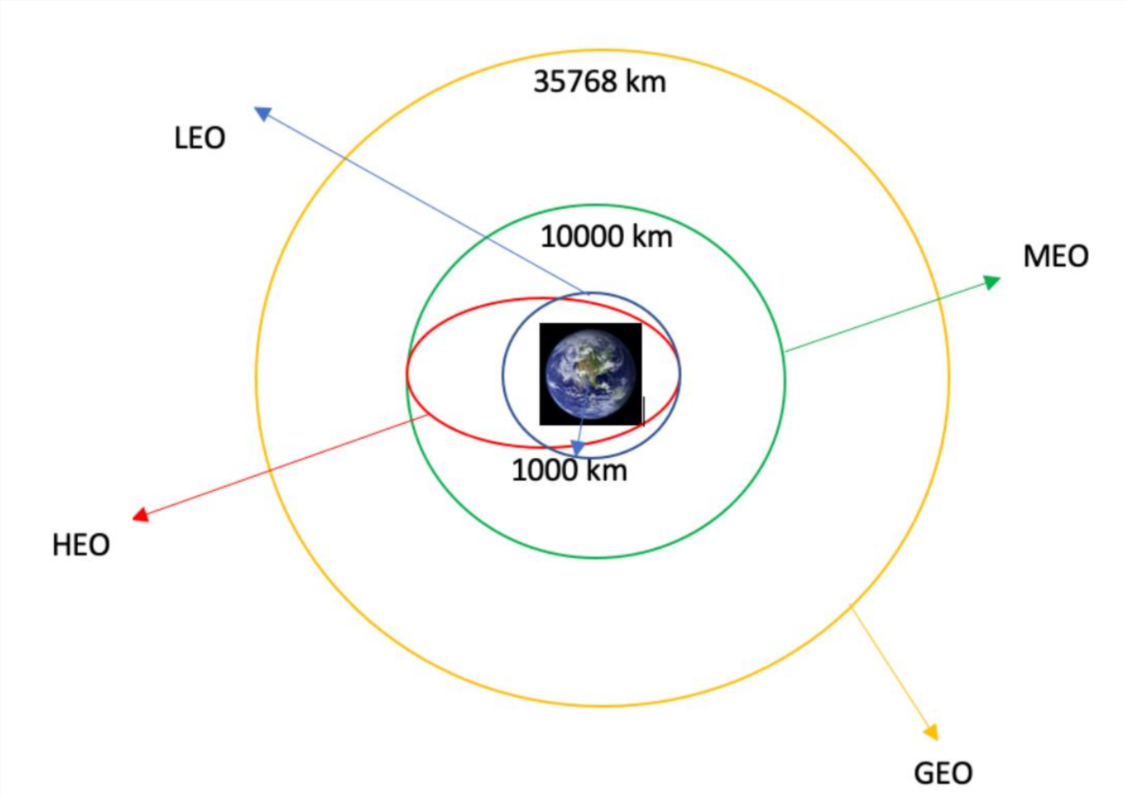
Each launch aims to an dedicated orbit, and here are some common orbit types:

•**LEO**: Low Earth orbit (LEO)is an Earth-centred orbit with an altitude of 2,000 km (1,200 mi) or less (approximately one-third of the radius of Earth),[1] or with at least 11.25 periods per day (an orbital period of 128 minutes or less) and an eccentricity less than 0.25.[2] Most of the manmade objects in outer space are in LEO.

•**HEO** A highly elliptical orbit, is an elliptic orbit with high eccentricity, usually referring to one around Earth.

•**MEO** Geocentric orbits ranging in altitude from 2,000 km (1,200 mi) to just below geosynchronous orbit at 35,786 kilometers (22,236 mi). Also known as an intermediate circular orbit. These are "most commonly at 20,200 kilometers (12,600 mi), or 20,650 kilometers (12,830 mi), with an orbital period of 12 hours.

•**GEO** It is a circular geosynchronous orbit 35,786 kilometres (22,236 miles) above Earth's equator and following the direction of Earth's rotation.



Folium MeasureControl

It is a good way to show distances using only code

Thank you!