IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Prakash Soundarrajan
04-Feb-2022

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- **Project background and context**

  To save the launching cost of the Space Y, we are going to bench mark how SpaceX is able to launch the rockets with a cost of 62 million dollars; whereas other providers cost upward of 165 million dollars each. With an initial assessment, we were able to find out that much of the savings is because SpaceX can reuse the first stage. Analyzing the SpaceX data on the successful launches, we will be able to find out the inputs required to increase the probability of our launch to be successful.

- **Problems you want to find answers**

  1. What are all the influences which makes the rocket to land successfully?
  2. What are all conditions the SpaceX maintained to get the best results and ensure the best rocket success landing rate.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Irrelevant columns were dropped
  - Missing data were filled with mean
  - One hot Encoding were done
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Scatter Graphs, Bar Graphs to show relationships between variables to show patterns of data
- Perform interactive visual analytics using Folium and Plotly Dash
  - Used Folium to understand geographical patterns about launch sites & Plotly Dash for Web page
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
  - Using SpaceX REST API we gathered the SpaceX Launch Data.
  - With API, data about launches, rocket used, payload delivered, launch, landing, and landing outcome were received
  - And by Web Scrapping we obtained data about Falcon 9 Launch data from Wikipedia using BeautifulSoup.
- You need to present your data collection process use key phrases and flowcharts

GIT URL

Data Collection – SpaceX API

Imported Necessary Libraries

```
In [1]: import requests
        import pandas as pd
        import numpy as np
        import datetime
```

Receive data using request & SpaceX API url data

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

static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_
static_response = requests.get(static_json_url).json()
```

Convert the data to json file

Using custom functions clean the data

```
def getboosterversion(data):
    for x in data['rocket']:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        boosterversion.append(response['name'])

def getLaunchSite(data):
    for x in data['launchpad']:
        response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
        Longitude.append(response['longitude'])
        Latitude.append(response['latitude'])
        LaunchSite.append(response['name'])

def getPayloadData(data):
    for load in data['payloads']:
        response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
        PayloadMass.append(response['mass_kg'])
        Orbit.append(response['orbit'])
```

```
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}

space_data = pd.DataFrame.from_dict(launch_dict)

data_falcon9 = space_data[space_data['BoosterVersion'] == 'Falcon 9']

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

Filter out unnecessary data and save the file to CSV

GIT URL

Data Collection - Scraping

Receive data using request from static url

```
page = requests.get(static_url)
```

Create Beautifulsoup Object

```
soup = BeautifulSoup(page.text, 'html.parser')
```

Find the tables and get the column names

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

```
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
```

Create a dictionary to use it to get required informations

```
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']=[]
```

Append data to the keys

```
In [12]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(
            # get table row
             for rows in table.find_all("tr")
                 #check to see if first table
```

Convert dictionary → DataFrame → CSV file

```
df = pd.DataFrame.from_dict(launch_dict)
df.to_csv('spacex_web_scraped.csv', index=False)
```

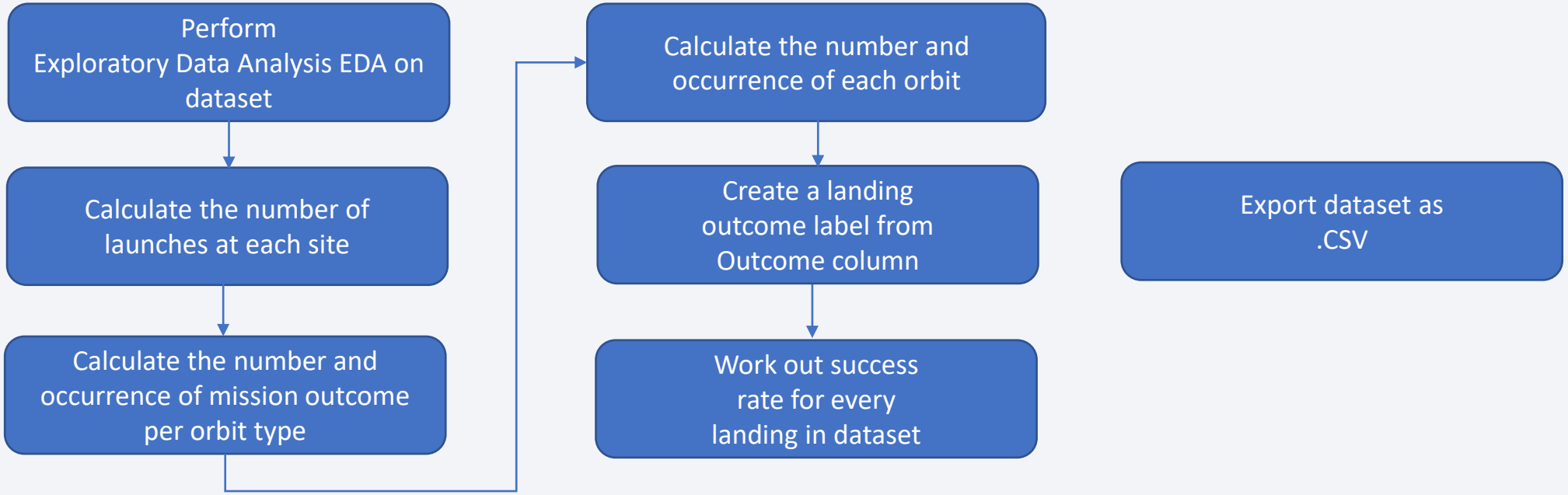
GIT URL

Data Wrangling

- Describe how data were processed

  In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad. True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.

- You need to present your data wrangling process using key phrases and flowcharts



GIT URL

EDA with Data Visualization

Scatter Graphs being drawn:

- 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

Line Graph being drawn:

- Success Rate VS. Year

GIT URL

EDA with SQL

- Displaying the names of the unique launch sites in the space mission
- Displaying 5 records where launch sites begin with the string 'KSC'
- Displaying the total payload mass carried by boosters launched by NASA (CRS)
- Displaying average payload mass carried by booster version F9 v1.1
- Listing the date where the successful landing outcome in drone ship was achieved.
- 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 and failure mission outcomes
- Listing the names of the booster_versions which have carried the maximum payload mass
- Listing the records which will display the month names, successful landing_outcomes in ground pad ,booster versions, 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.

GIT URL

Build an Interactive Map with Folium

To visualize the Launch Data into an interactive map.

- We took the Latitude and Longitude Coordinates at each launch site and added a Circle Marker around each launch site with a label of the name of the launch site.
- We assigned the data frame launch_outcomes(failures, successes) to classes 0 and 1 with Green and Red markers on the map in a MarkerCluster()
- Using Haversine's formula we calculated the distance from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns. Lines are drawn on the map to measure distance to landmarks

GIT URL

Build a Dashboard with Plotly Dash

The dashboard is built with Flask and Dash web framework.

- Graphs
- Pie Chart showing the total launches by a certain site/all sites
- display relative proportions of multiple classes of data.
- size of the circle can be made proportional to the total quantity it represents.

Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions

- It shows the relationship between two variables.
- It is the best method to show you a non linear pattern.
- The range of data flow, i.e. maximum and minimum value, can be determined.
- Observation and reading are straightforward.

GIT URL

Predictive Analysis (Classification)

BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

EVALUATING MODEL

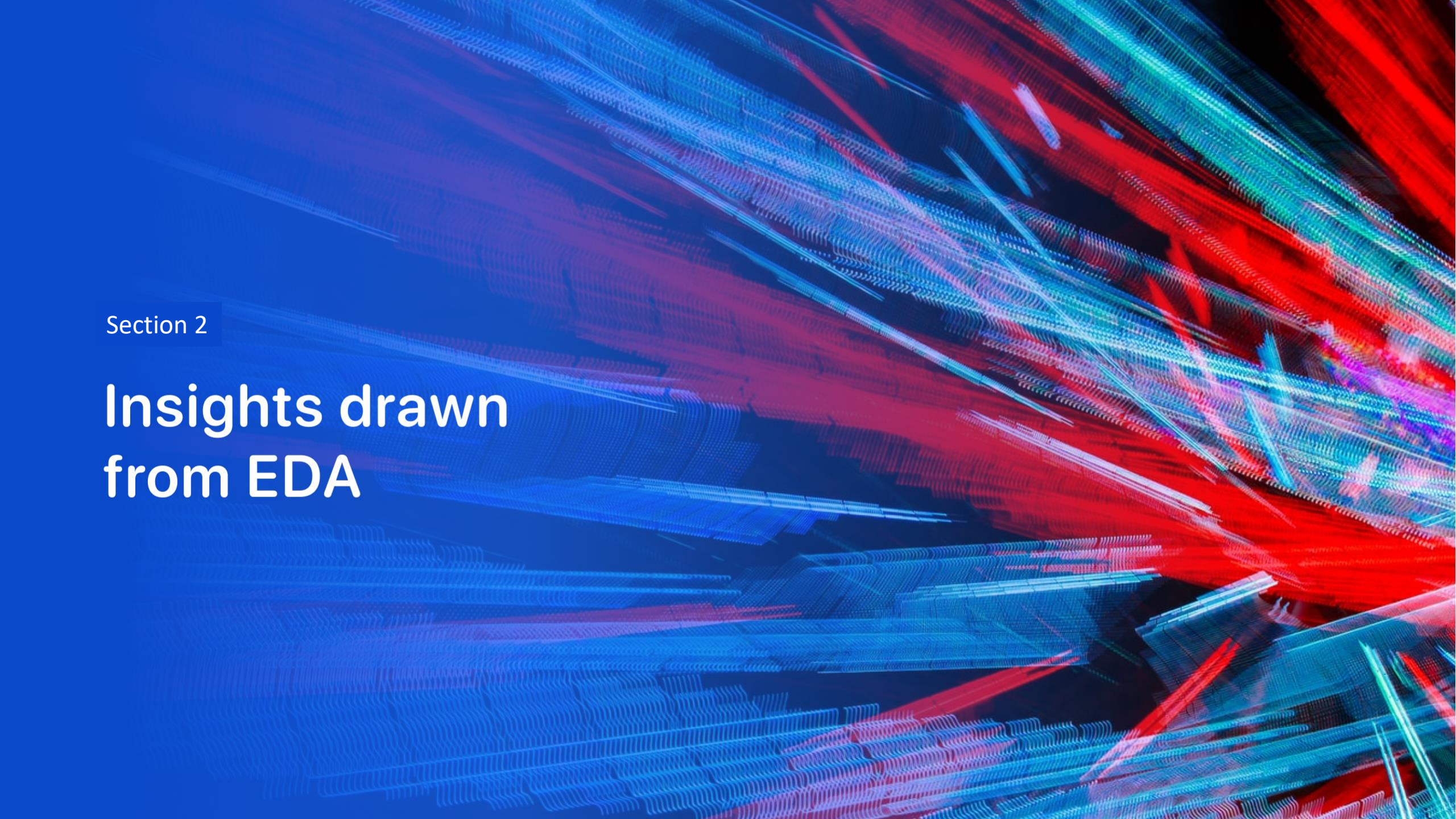
- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

IMPROVING MODEL

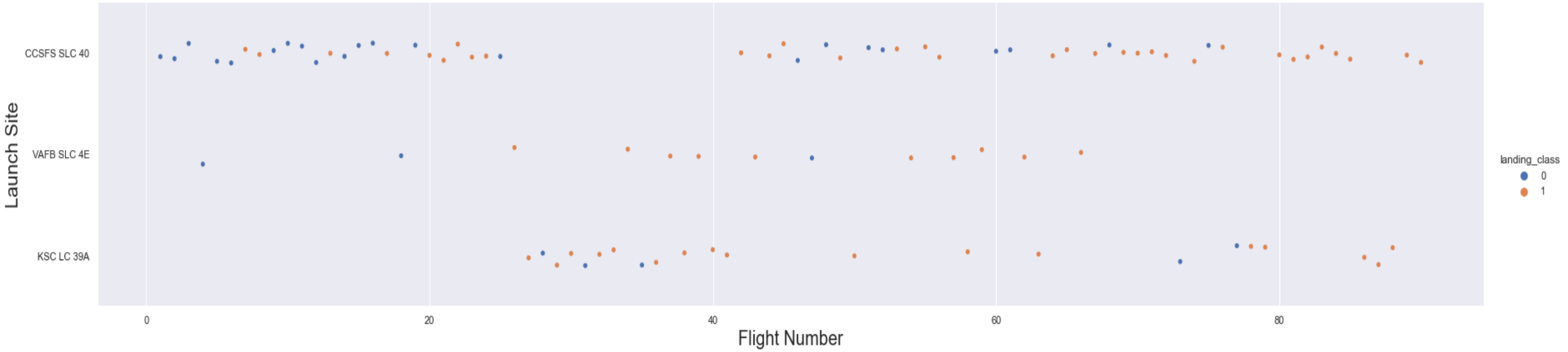
- Feature Engineering
- Algorithm Tuning
- FINDING THE BEST PERFORMING CLASSIFICATION MODEL
- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

Results

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

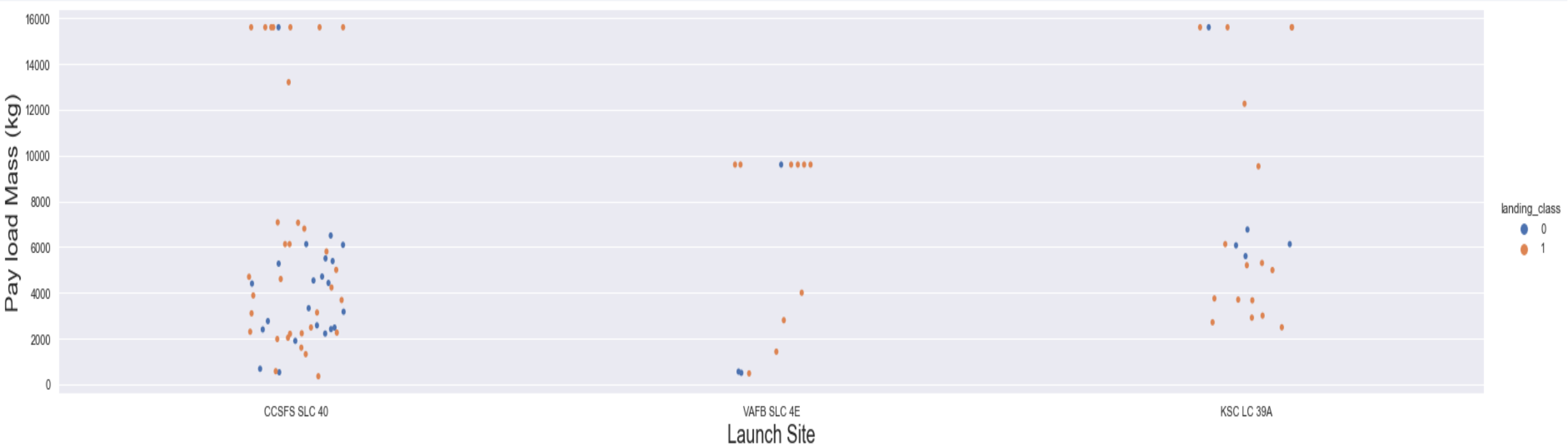

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



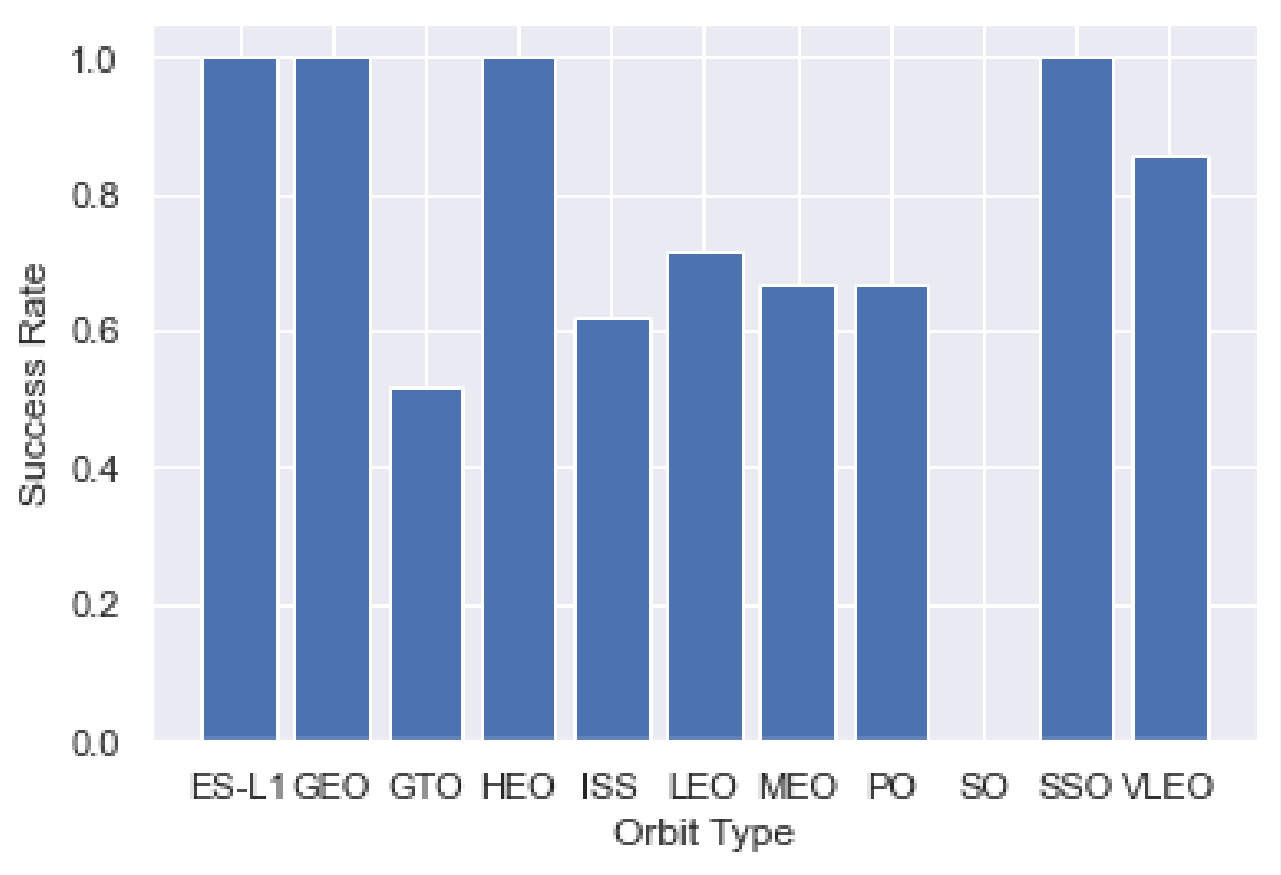
1. The more amount of flights at a launch site the greater the success rate at a launch site.

Payload vs. Launch Site



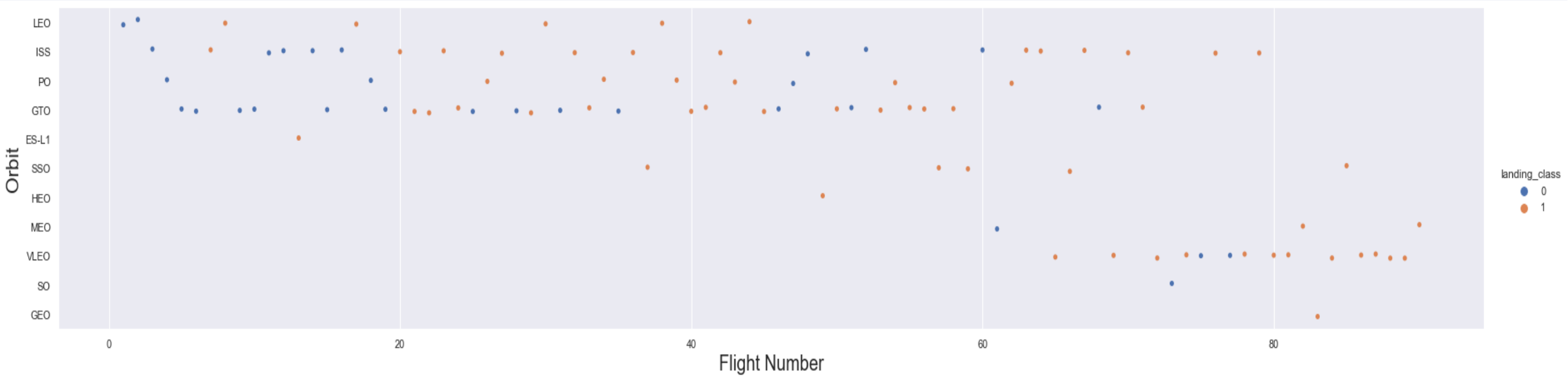
1. Irrespective of Launch site, if the Pay load is more 8000 kg, the probability of successful landing is more than 90%

Success Rate vs. Orbit Type



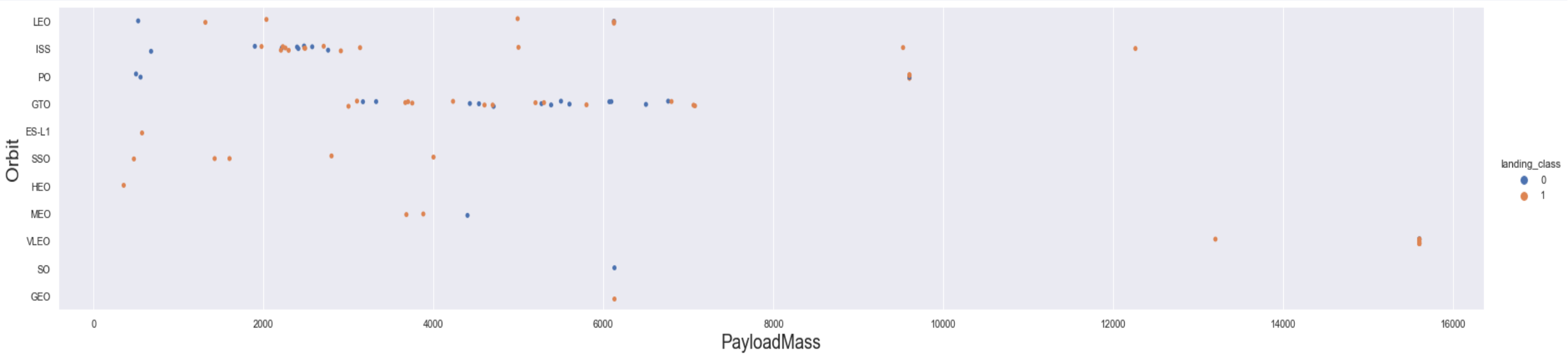
1. Success Ratio at ES-L1, GEO, HEO, SSO is 1
2. It is better to plan launches to this orbit for high probability of success

Flight Number vs. Orbit Type



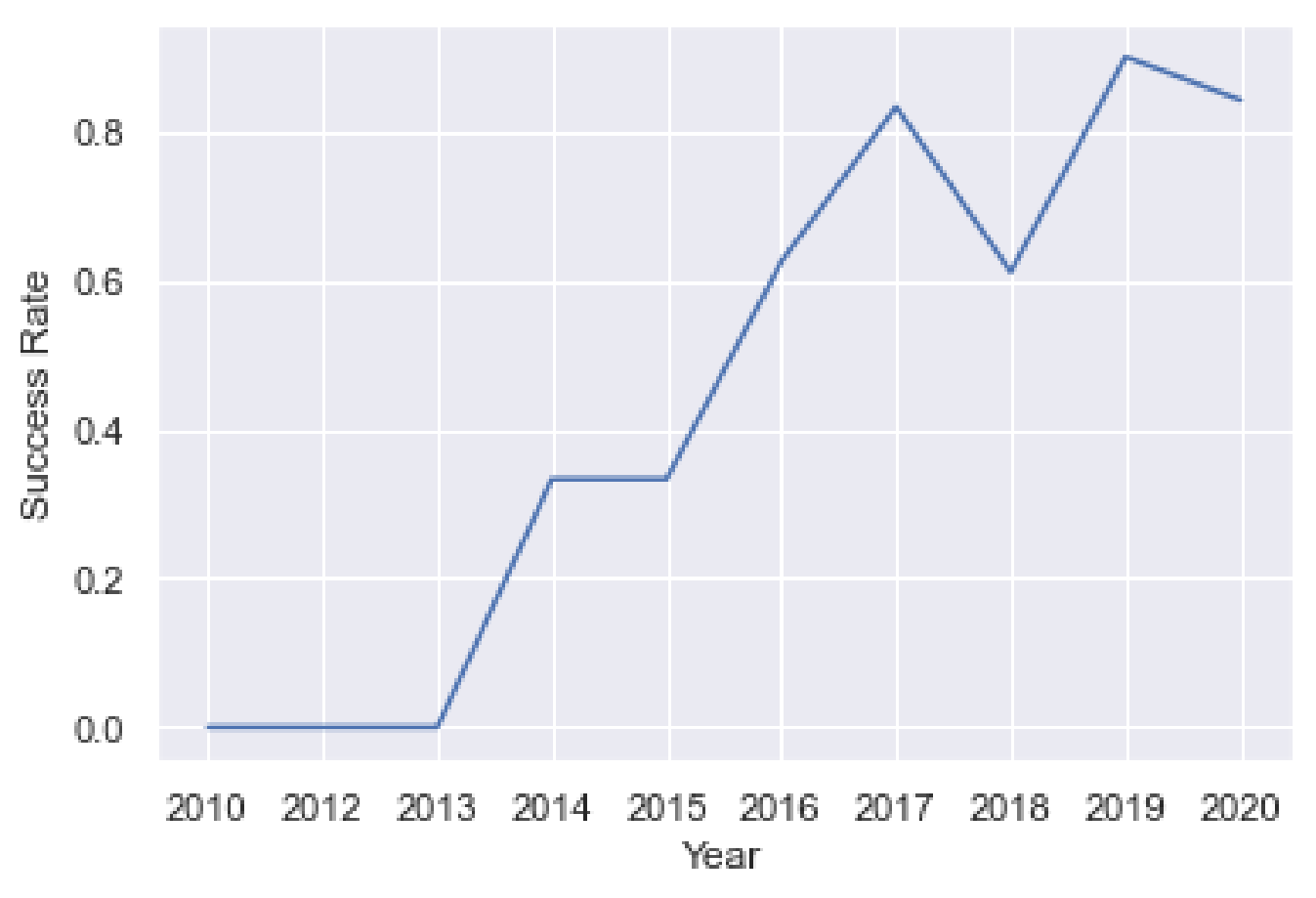
1. LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



1. With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
2. 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 success rate since 2013 kept increasing till 2020

All Launch Site Names

Unique Launch Sites

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

QUERY EXPLAINATION

Using the word ***DISTINCT*** in the query means that it will only show Unique values in the ***Launch_Site*** column from ***tblSpaceX***

Launch Site Names Begin with 'CCA'

Select TOP5 * from tblSpaceX WHERE Launch_Site LIKE 'KSC%'

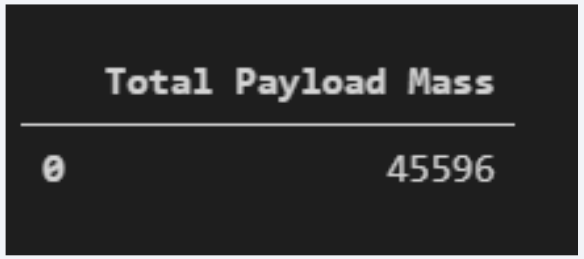
	Date	Time_UTC	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS_KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	19-02-2017	2021-07-02 14:39:00.0000000	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
1	16-03-2017	2021-07-02 06:00:00.0000000	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	No attempt
2	30-03-2017	2021-07-02 22:27:00.0000000	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
3	01-05-2017	2021-07-02 11:15:00.0000000	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
4	15-05-2017	2021-07-02 23:21:00.0000000	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	No attempt

QUERY EXPLAINATION

Using the word ***TOP 5*** in the query means that it will only show 5 records from ***tblSpaceX*** and ***LIKE*** keyword has a wild card
with the words ***'KSC%'*** the percentage in the end suggests that the Launch_Site name must start with KSC.

Total Payload Mass

Select SUM(PAYLOAD_MASS_KG_) TotalPayloadMass from tblSpaceX where Customer='NASA(CRS)'", 'TotalPayloadMass



	Total Payload Mass
0	45596

QUERY EXPLAINATION

Using the function ***SUM*** summates the total in the column ***PAYLOAD_MASS_KG_***

The ***WHERE*** clause filters the dataset to only perform calculations on ***Customer NASA (CRS)***

Average Payload Mass by F9 v1.1

Select AVG(PAYLOAD_MASS_KG_) AveragePayloadMass from tblSpaceX where Booster_Version='F9v1.1'

QUERY EXPLAINATION

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

The ***WHERE*** clause filters the dataset to only perform calculations on ***Booster_version F9 v1.1***

	Average Payload Mass
0	2928

First Successful Ground Landing Date

Select MIN(Date) SLO from tblSpaceX where Landing_Outcome="Success(droneship)"

Date which first Successful landing outcome in drone ship was acheived.
06-05-2016

QUERY EXPLAINATION

Using the function ***MIN*** works out the minimum date in the column ***Date***
The ***WHERE*** clause filters the dataset to only perform calculations on ***Landing_Outcome***
Success (drone ship)

Successful Drone Ship Landing with Payload between 4000 and 6000

Select Booster_Version from tblSpaceX where Landing_Outcome='Success(groundpad)' AND Payload_MASS_KG_>4000 AND Payload_MASS_KG_<6000

	Date which first Successful landing outcome in drone ship was acheived.
0	F9 FT B1032.1
1	F9 B4 B1040.1
2	F9 B4 B1043.1

QUERY EXPLAINATION

Selecting only ***Booster_Version***

The ***WHERE*** clause filters the dataset to ***Landing_Outcome = Success (drone ship)***

The ***AND*** clause specifies additional filter conditions

Payload_MASS_KG_>4000 AND ***Payload_MASS_KG_<6000***

Total Number of Successful and Failure Mission Outcomes

```
SELECT (SELECT Count (Mission_Outcome from tblSpaceX where Mission_Outcome
LIKE '%Success%') as Successful_Mission_Outcomes
(SELECT Count(Mission_Outcome from tblSpaceX where Mission_Outcome
LIKE ''%Failure%') as Failure_Mission _Coutcomes
```

	Successful_Mission_Outcomes	Failure_Mission_Outcomes
0	100	1

QUERY EXPLAINATION

PHRASE "(Drone Ship was a Success)"

LIKE '%Success%'

Word 'Success' is in the phrase the filter will include it in the dataset

Boosters Carried Maximum Payload

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

	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

97 rows × 2 columns

QUERY EXPLAINATION

Using the word ***DISTINCT*** in the query means that it will only show Unique values in the ***Booster_Version*** column from ***tblSpaceX***

GROUP BY puts the list in order set to a certain condition.

DESC means its arranging the dataset into descending order

2015 Launch Records

```
SELECT DATENAME(month, DATEADD(month, MONTH(CONVERT(date, Date, 105)), 0) - 1) AS
Month, Booster_Version, Launch_Site, Landing_Outcome
FROM tblSpaceX WHERE (Landing_Outcome LIKE N'%Success %') AND
(YEAR(CONVERT(date, Date, 105)) = '2017')
```

Month	Booster_Version	Launch_Site	Landing_Outcome
January	F9 FT B1029.1	VAFB SLC-4E	Success (drone ship)
February	F9 FT B1031.1	KSC LC-39A	Success (ground pad)
March	F9 FT B1021.2	KSC LC-39A	Success (drone ship)
May	F9 FT B1032.1	KSC LC-39A	Success (ground pad)
June	F9 FT B1035.1	KSC LC-39A	Success (ground pad)
June	F9 FT B1029.2	KSC LC-39A	Success (drone ship)
June	F9 FT B1036.1	VAFB SLC-4E	Success (drone ship)
August	F9 B4 B1039.1	KSC LC-39A	Success (ground pad)
August	F9 FT B1038.1	VAFB SLC-4E	Success (drone ship)
September	F9 B4 B1040.1	KSC LC-39A	Success (ground pad)
October	F9 B4 B1041.1	VAFB SLC-4E	Success (drone ship)
October	F9 FT B1031.2	KSC LC-39A	Success (drone ship)
October	F9 B4 B1042.1	KSC LC-39A	Success (drone ship)
December	F9 FT B1035.2	CCAFS SLC-40	Success (ground pad)

QUERY EXPLAINATION

The Date fields are stored as ***NVARCHAR*** the ***MONTH*** function returns name month. The function ***CONVERT*** converts ***NVARCHAR*** to ***Date***. ***WHERE*** clause filters ***Year*** to be 2017

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

```
SELECT COUNT(Landing_Outcome) FROM tblSpaceX
WHERE (Landing_Outcome LIKE '%Success%') AND (Date > '04-06-2010')
AND (Date < '20-03-2017')
```

	Successful Landing Outcomes Between 2010-06-04 and 2017-03-20
0	34

QUERY EXPLAINATION

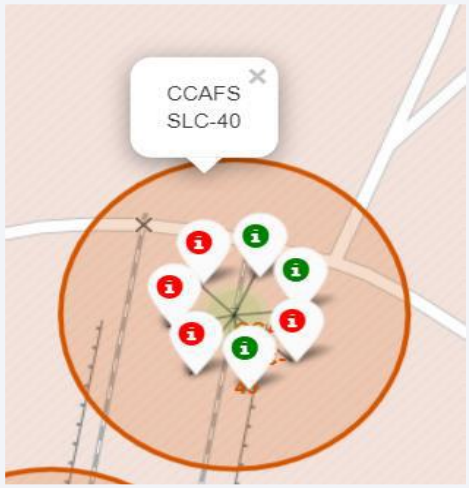
Function ***COUNT*** counts records in column ***WHERE*** filters data ***LIKE (wildcard)***
AND (conditions)AND (conditions)

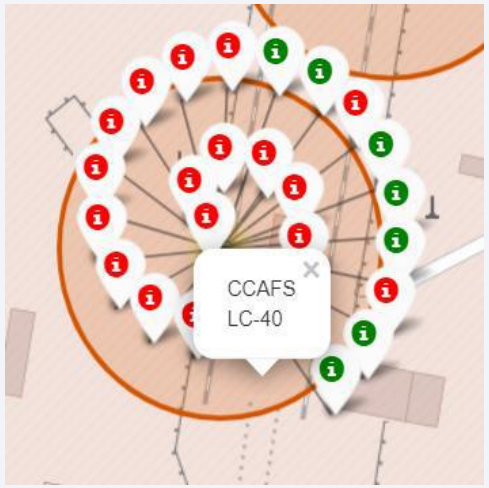
Section 3

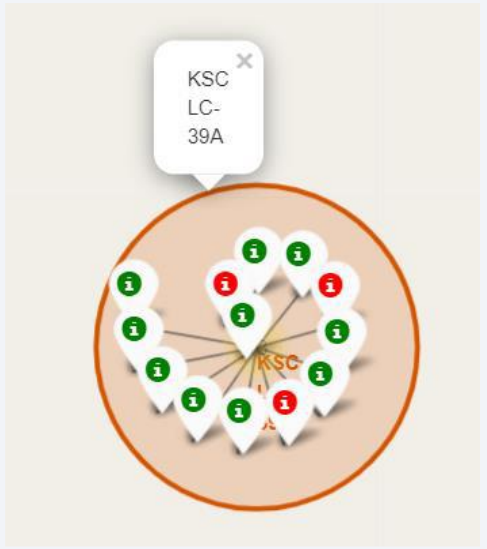
Launch Sites Proximities Analysis

All launch sites global map markers

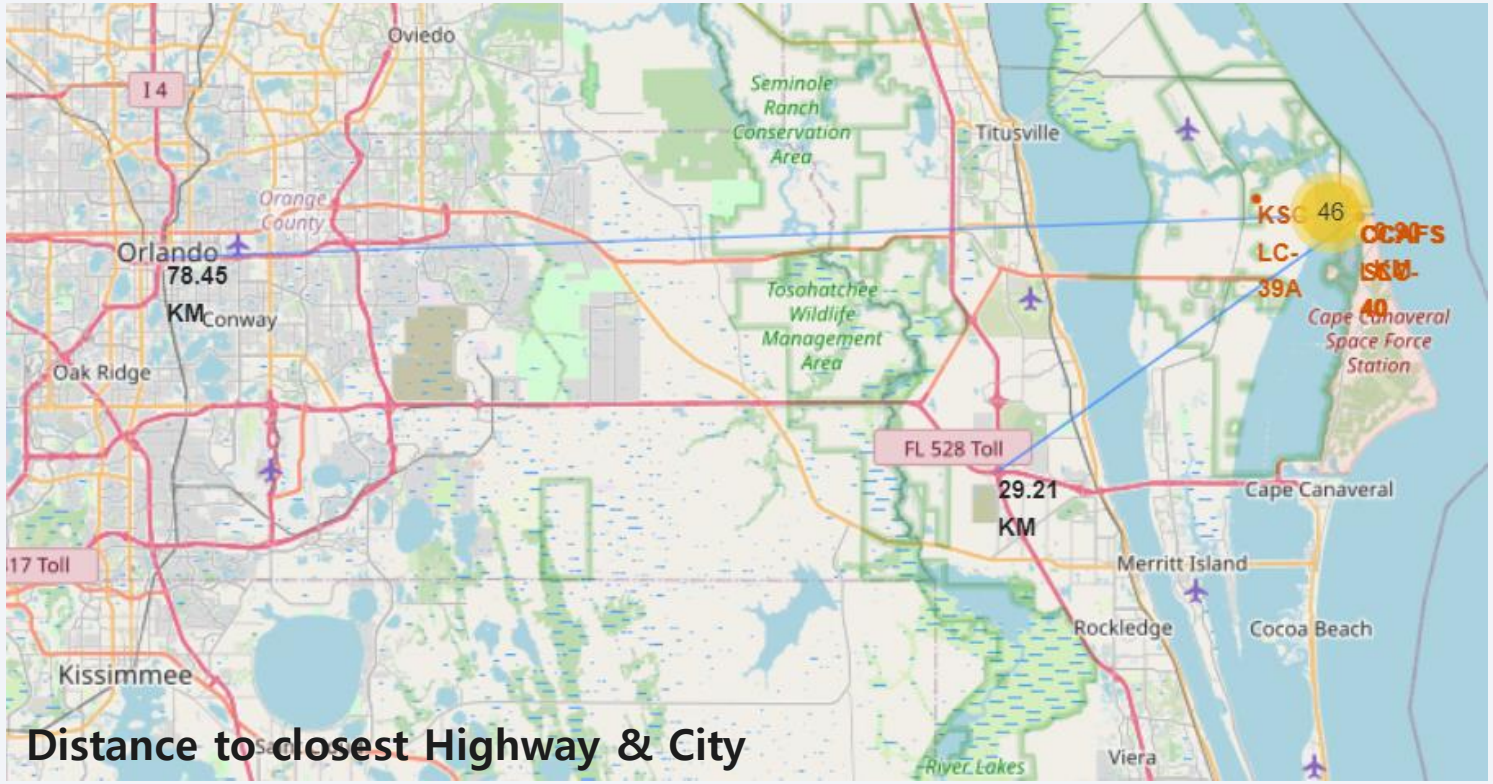
Color Labelled Markers

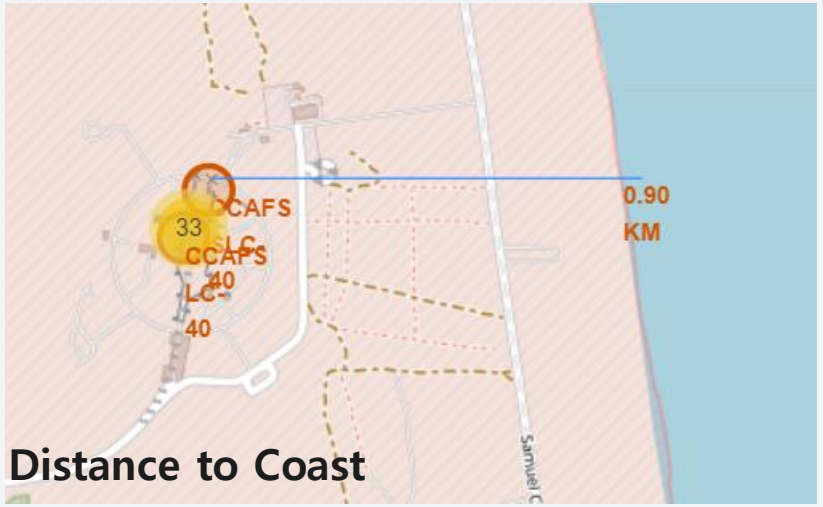


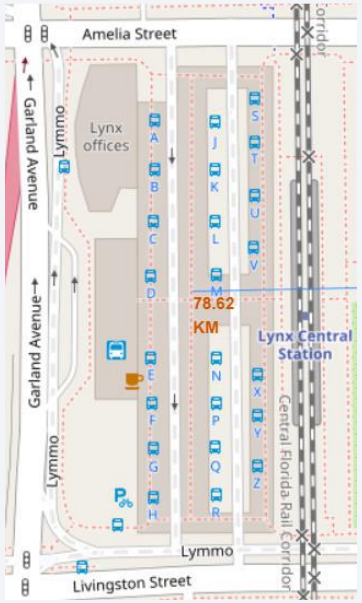




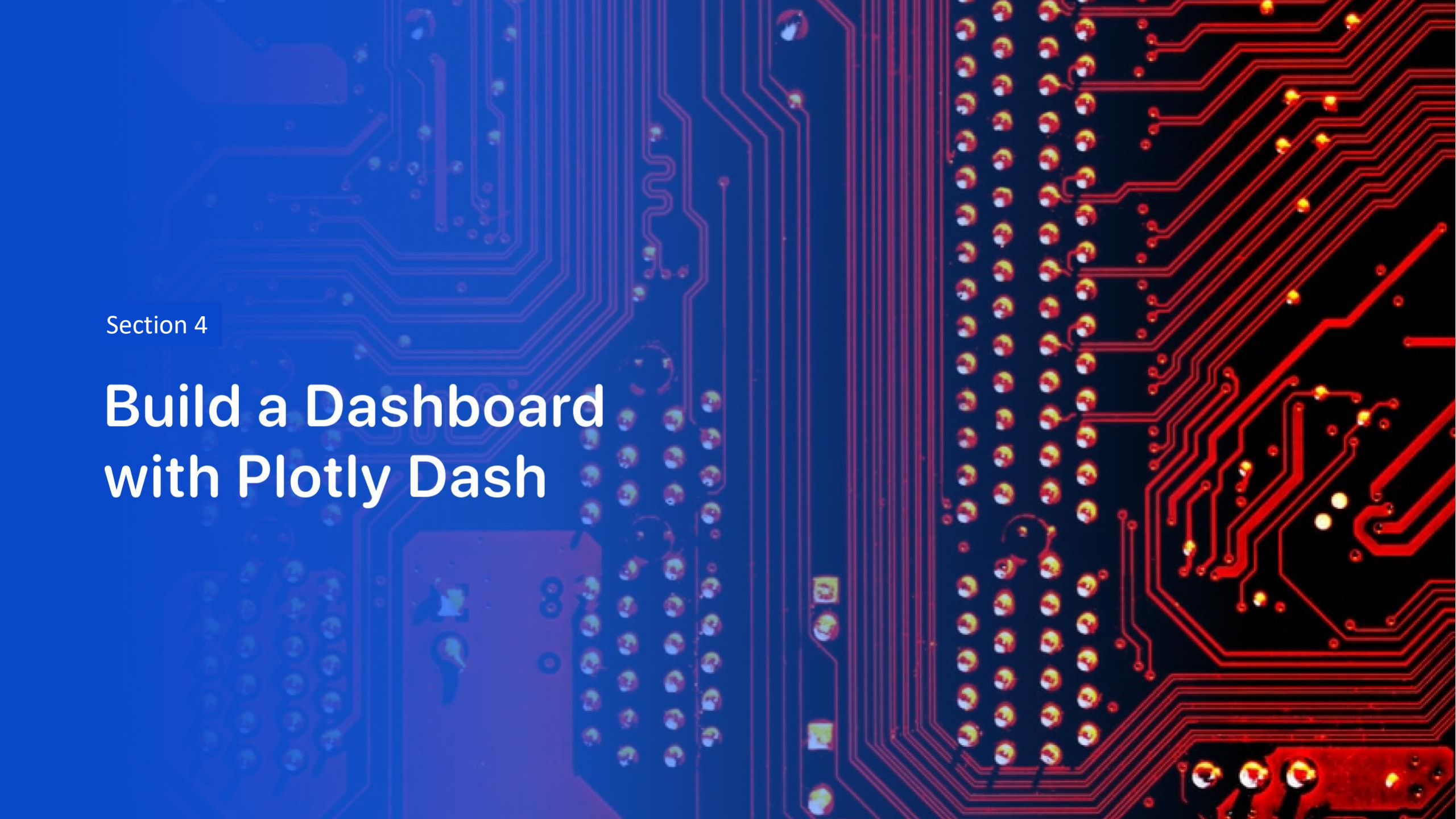
Working out Launch Sites distance to landmarks to find trends



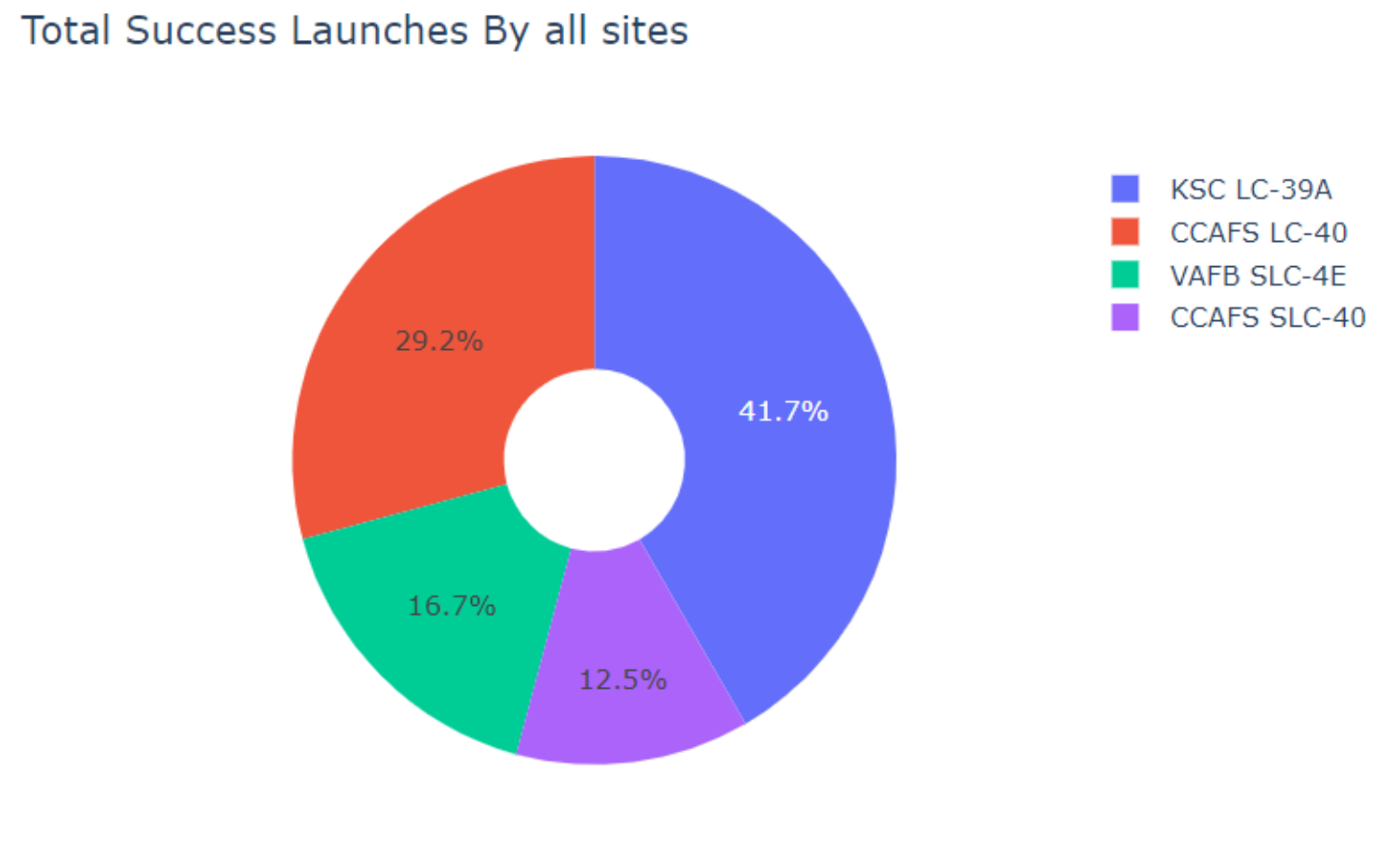




- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

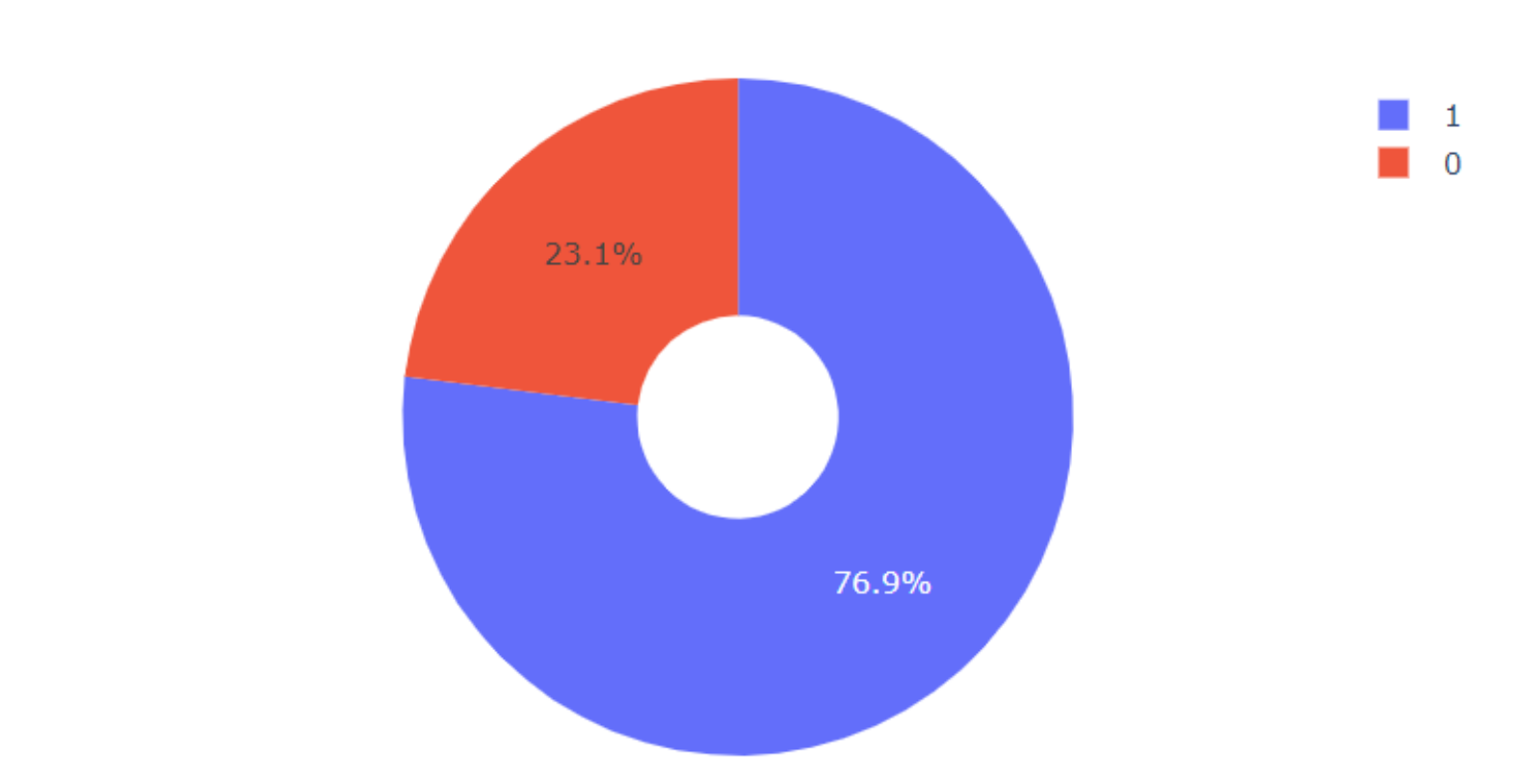

Section 4
Build a Dashboard with Plotly Dash

Pie chart Showing Success percentage



We can see that KSC LC-39A had the most successful launches from all the sites

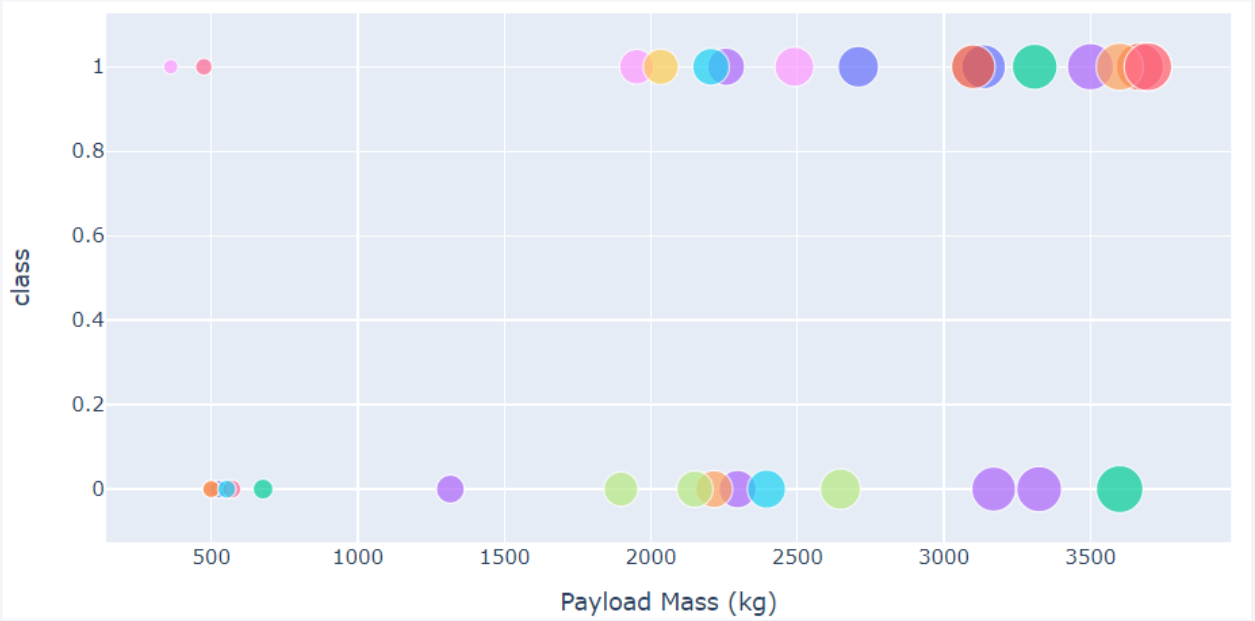
Launch Site with High Success Ratio



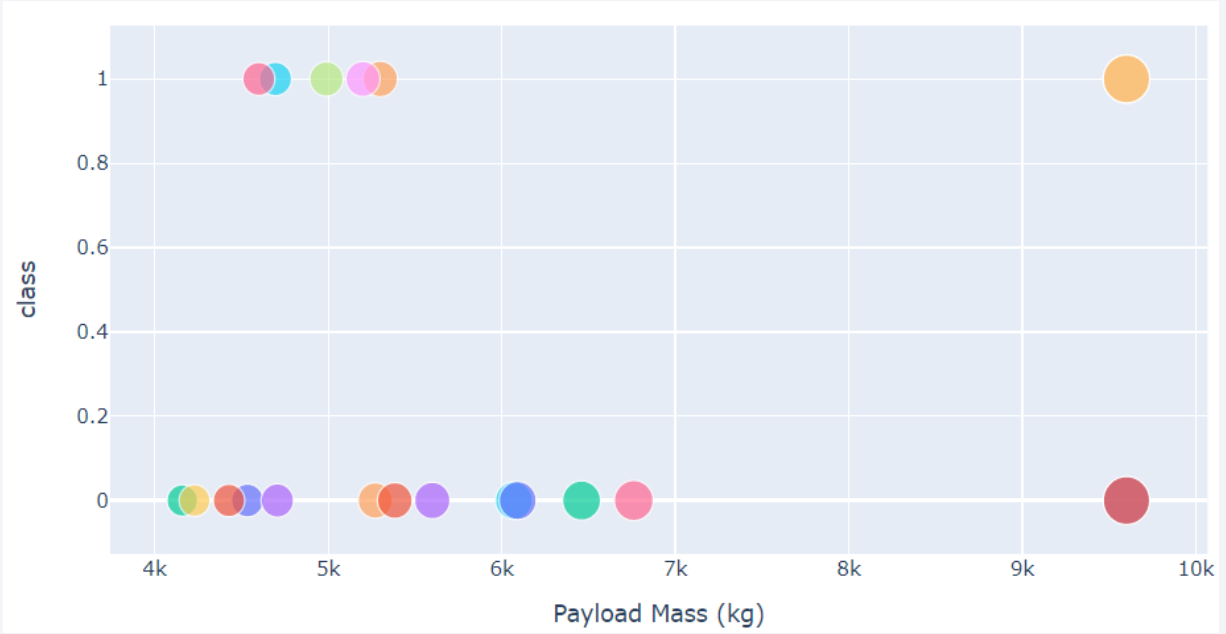
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs. Launch Outcome

Low Weighted Payload 0kg –4000kg



Heavy Weighted Payload 4000kg –10000kg



We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

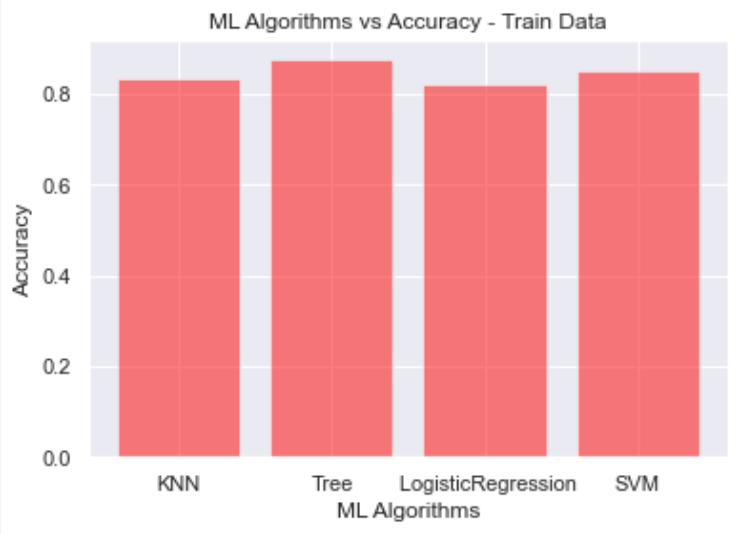
GIT URL

Classification Accuracy

The Decision Tree is the Best Method for this project

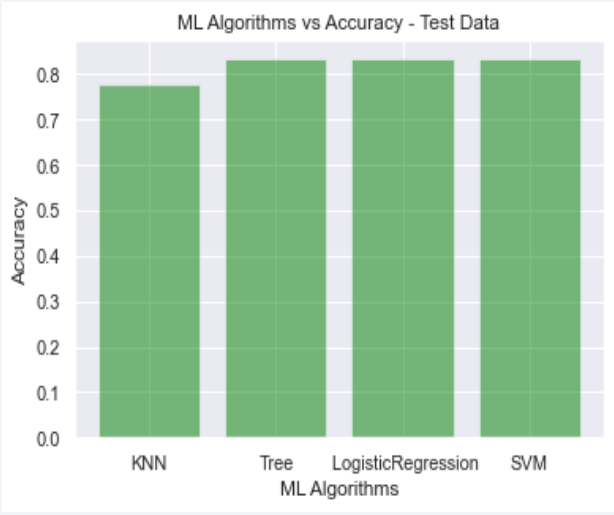
	ML_Algorithm	Accuracy
0	KNN	0.833929
1	Tree	0.873214
2	LogisticRegression	0.821429
3	SVM	0.848214

ACCURACY WITH TRAIN DATA



	ML_Algrithm	Accuracy
0	KNN	0.777778
1	Tree	0.833333
2	LogisticRegression	0.833333
3	SVM	0.833333

ACCURACY WITH TEST DATA

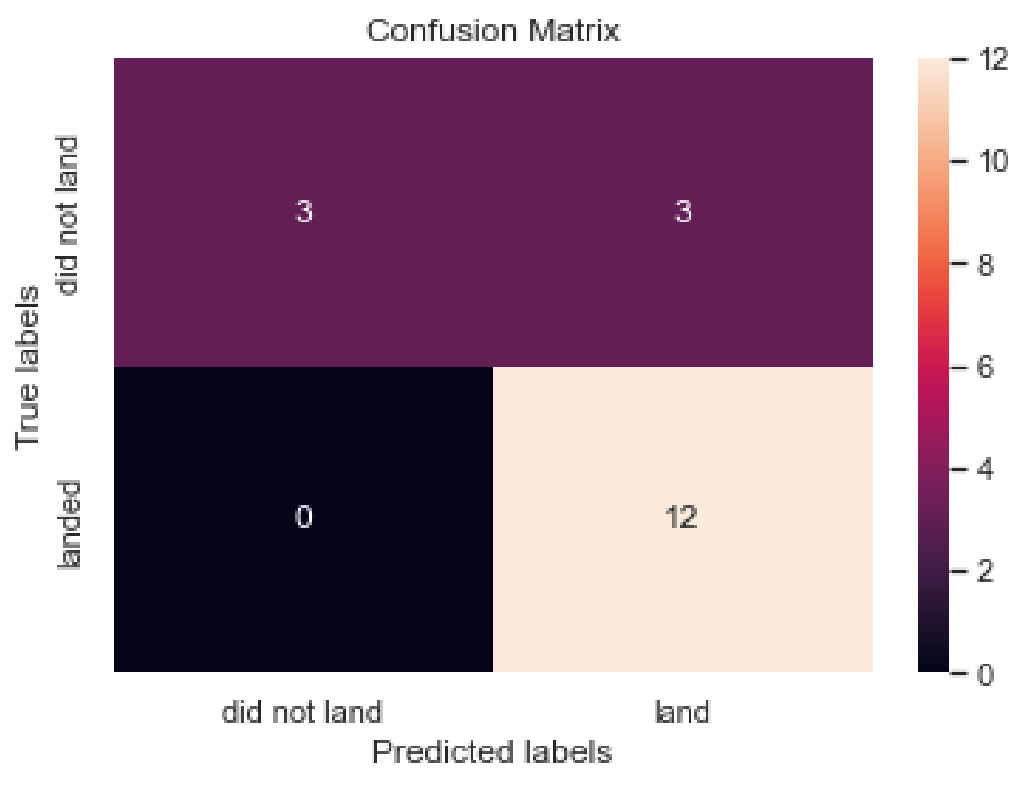



```
Best Algorithm is Tree with a score of 0.8732142857142857
Best Params is : {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 1
0, 'splitter': 'random'}
```

```
Best Algorithm is Tree_Test with a score of 0.8333333333333334
```

After selecting the best hyperparameters for the decision tree classifier using the validation data, we achieved 83.33% accuracy on the test data.

Confusion Matrix



Examining the confusion matrix, we see that Decision Tree Algorithm can distinguish between the different classes.

Only 3 cases, Decision Tree couldn't predict the right class

Conclusions

- The Decision Tree Algorithm is the best ML method to used for this project. It has high Accuracy Rate
- Launching at Orbits like ES-L1, GEO, HEO, SSO gives the 100% Probability of Success
- Pay load should be more 8000 kg, to have 90% probability of successful landing
- Both KSC LC-39A and VAFB SLC-4E Launch sites has 77% successful Landing.

Thank you!