Winning Space Race with Data Science

Abduljalil Abdulmumin Abiri
3rd October, 2021

Outline

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

NOTE: EDA - Exploratory Data Analysis

Introduction

- Project background and context:

SpaceX is the only company in the world that can reuse the first stage of rocket launch. This means that they are able to save a lot on expenses needed to launch a rocket and thus can provide rocket launches to organizations such as NASA at a much lower cost than rival companies. So as a Data Scientist, we can predict if the first stage of rocket launch will land successfully using data from SpaceX past launches and thus determine the cost of a launch. This information could prove immensely useful to SpaceX rivals.

We want to find answers to the questions below:

- The price of each launch.
- Which Parameters determine if the first stage can be reused.
- What influences a successful rocket landing.

Section 1

Methodology

Methodology

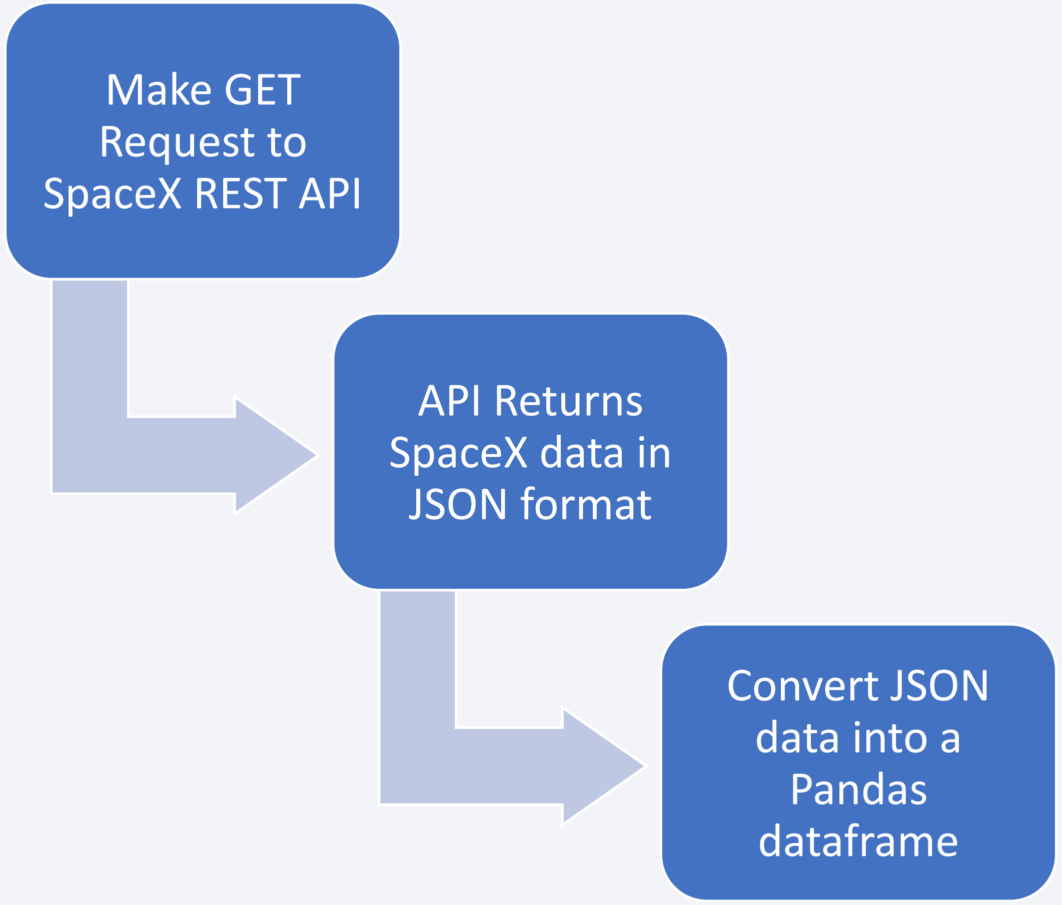
Executive Summary

- Data collection methodology:
  - Data on past Falcon 9 launches was collected by scrapping data from the Wikipedia page containing information about every single Falcon 9 launches. We used BeautifulSoup library for web scraping.
  - Also collected data stored in SpaceX Rest API.
- Perform data wrangling
  - Performed feature engineering on the data and removed irrelevant columns.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

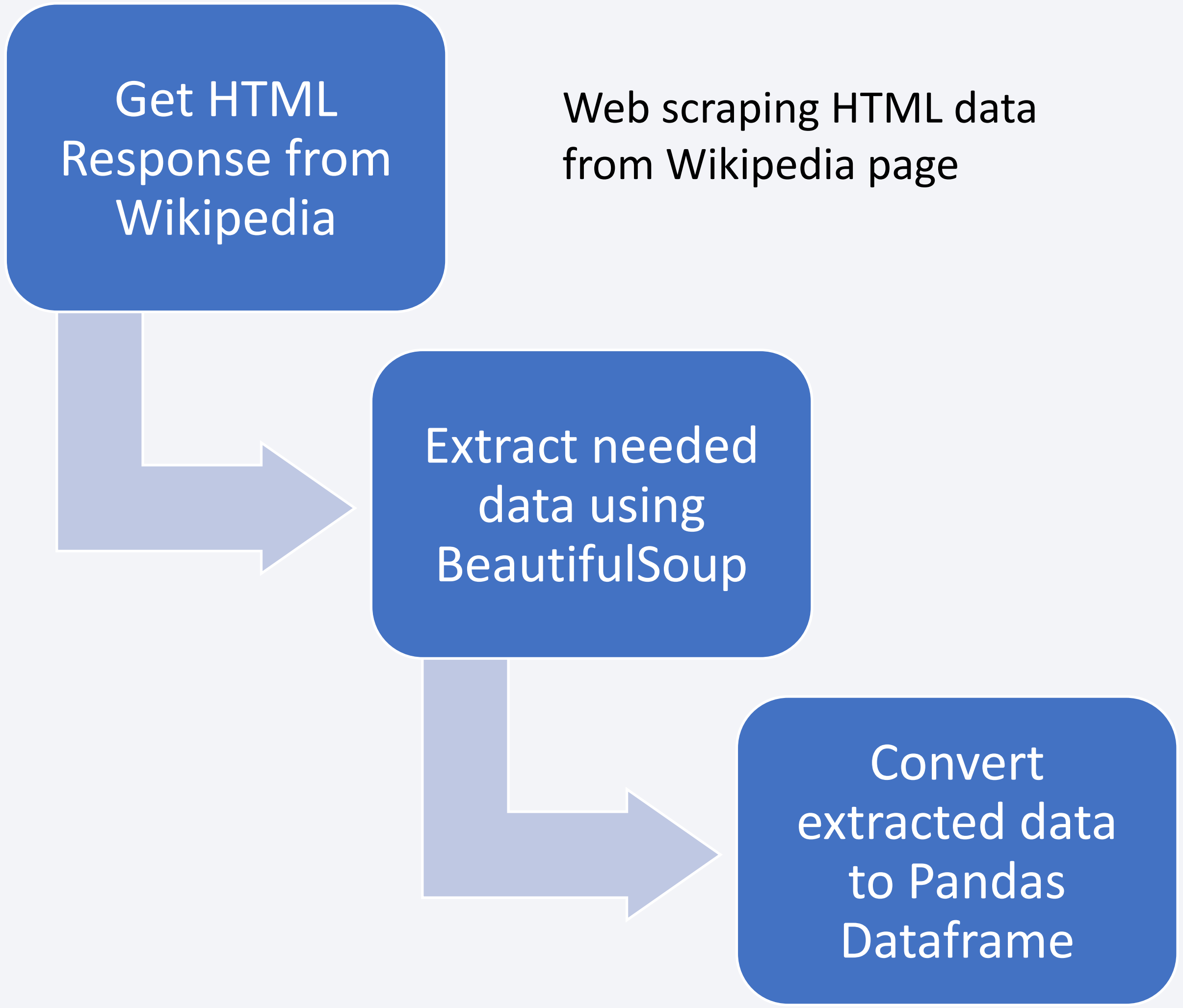
Data Collection

The flowcharts below describe how data was collected for the analysis and prediction.

SpaceX REST API


Make GET Request to SpaceX REST API
API Returns SpaceX data in JSON format
Convert JSON data into a Pandas dataframe

Web scraping HTML data from Wikipedia page


Get HTML Response from Wikipedia
Extract needed data using BeautifulSoup
Convert extracted data to Pandas Dataframe

Data Collection – SpaceX API

Request rocket launch data from SpaceX API.

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

Decode data as JSON file then turn into Pandas DataFrame

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

Get needed column data from each API Endpoint

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

getcolumnName(data)
```

Where:

- columnName includes: LaunchSite, payloadData, coreData etc.
- Endpoint includes: launchpads, cores, launch_site, etc.

Construct dataset using data obtained.

```
launch_df=pd.DataFrame.from_dict(launch_dict)
data_falcon9=launch_df.loc[launch_df['BoosterVersion']!='Falcon 1']
meanPayload=data_falcon9['PayloadMass'].mean()
data_falcon9['PayloadMass'].replace(np.NaN, meanPayload,inplace=True)
```

GITHUB LINK TO NOTEBOOK

- LINK

Data Collection - Scraping

GITHUB LINK TO COMPLETE CODE:
[LINK]

Request Falcon9 Launch Wikpedia page

```
static_url =
"https://en.wikipedia.org/w/index.php
?title=List_of_Falcon_9_and_Falcon_H
eavy_launches&oldid=1027686922"
response = requests.get(static_url)
BE4U = BeautifulSoup(response.text)
```

Extract all tables and column names from HTML Table using BeautifulSoup Library

```
html_tables = BE4U.find_all('table')

for th in daTH:
•name =extract_column_from_header(th)
•duh_names.append(name)
•if (name!=None) and (len(name) >0):
•column_names.append(name)
```

Create DataFrame from extracted HTML Tables.

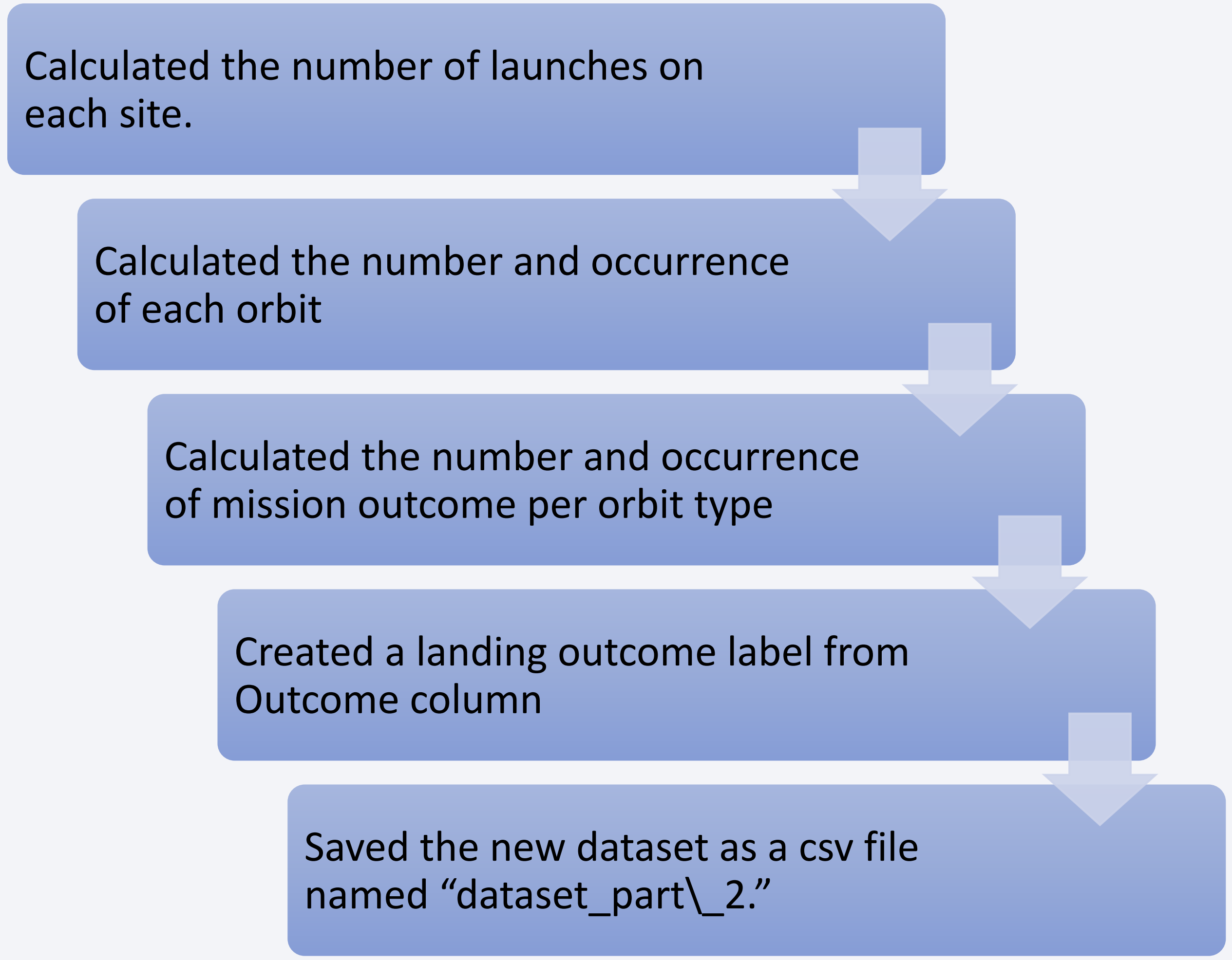
Save as CSV file.

```
launch_dict= dict.fromkeys(column_names)
launch_dict[‘columnName.'] = []    #Where
columnName are ‘Flight No’, ‘Launch Site’,
‘Payload’, ‘Payload Mass’ etc.
extracted_row = 0

for table_number,table in
enumerate(BE4U.find_all('table',"wikitable
plainrowheaders collapsible")):
•for rows in table.find_all("tr"):
  •#Append the columnNames into the strings..
   See the complete code in the Github URL
   provided.

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

Data Wrangling

Performed Exploratory Data Analysis on the dataset obtained.
The flowchart describes the process taken for the Data Analysis

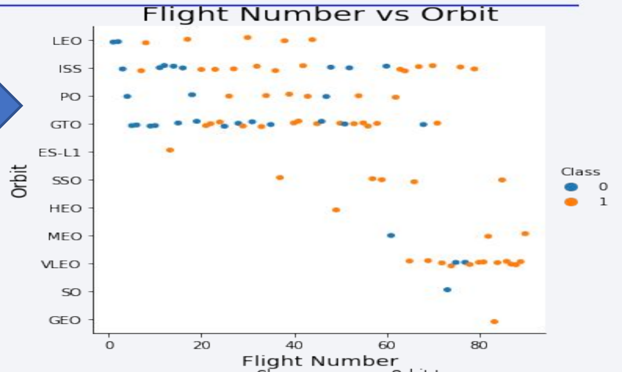
GITHUB LINK TO COMPLETE CODE:
[LINK]

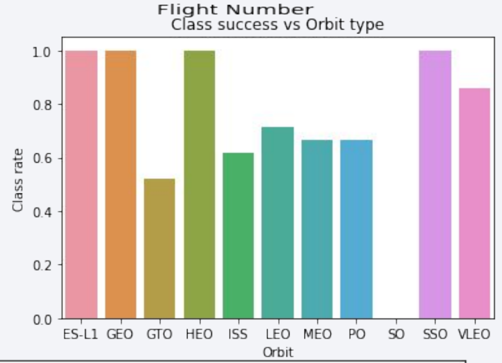


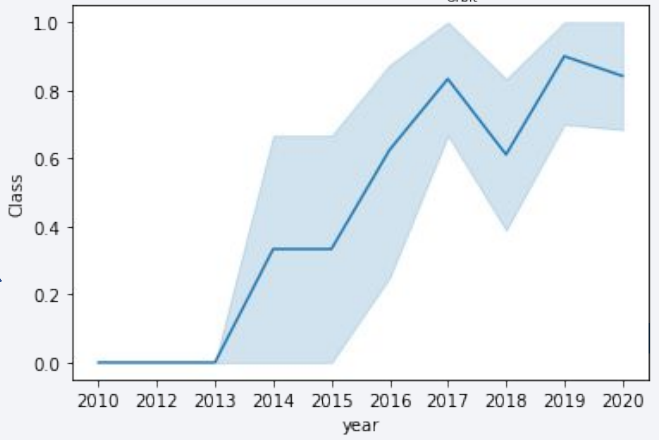
1. Calculated the number of launches on each site.
2. Calculated the number and occurrence of each orbit
3. Calculated the number and occurrence of mission outcome per orbit type
4. Created a landing outcome label from Outcome column
5. Saved the new dataset as a csv file named “dataset_part_2.”

EDA with Data Visualization

- The graphs drawn in the analysis are:
  - Scatter plot graphs: Scatter plot graphs are great at showing the correlation between two continuous variables. Graphs drawn with scatter graph include: Flight Number vs Payload Mass, Orbit vs Flight Number etc.
  - Bar Chart graph: A bar chart is great at making comparison of metric values across different subgroups of data. The success rate vs Orbit type used the bar chart to show the success rate of each orbit.
  - Line Graph: A line graph is great at emphasizing changes in values for one variable (plotted on the vertical axis) for continuous values of the second variable (Horizontal axis)







GITHUB LINK TO COMPLETE CODE:
LINK

EDA with SQL

- The SQL Queries performed are:
  - Displaying the names of the unique launch sites in the space mission
  - Displaying 5 records where launch sites begin with the string 'CCA'
  - Displaying average payload mass carried by booster version F9 v1.1
  - Listed the date when the first successful landing outcome in ground pad was achieved.
  - Listed the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - Listed the total number of successful and failure mission outcomes
  - Listed the names of the booster_versions which have carried the maximum payload mass. Use a subquery.
  - Listed 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

GITHUB LINK TO COMPLETE CODE: LINK

Build an Interactive Map with Folium

- We added markers for each sites location, we added circles to each site area, We added lines between a launch site to its proximities, We added a markerCluster to group each attempted launch
- We added the markers to mark each site location on the map, then we added the circles to highlight a site’s area for easy identification on the map. We added lines to visualize the distances between a launch site to its proximities, finally we added markerCluster to group each attempted launch according to launch site sp as to avoid too much clutter on the map.

GITHUB LINK: LINK

Build a Dashboard with Plotly Dash

- The scatter plot interactive graph was added to show the relationship with Outcome and Payload Mass (Kg) for the different booster versions.
  - Scatter plot was used because it is the best visualization method to show a non-linear pattern.
- The Pie chart interactive graph was added to show the total launches by a certain site or all sites together.
  - Pie chart was selected because it can visualize the relative proportions of different classes of data.

GITHUB LINK TO COMPLETE CODE: LINK

Predictive Analysis (Classification)

Building Model

- Load csv data into pandas DataFrame, Transform data, split data into training and test data sets
- Decide which Machine Learning Algorithms to use and set our parameters and algorithms to GridSearchCV
- Fit the dataset into the GridSearchCV object and train the model.

Evaluating Model

- Check accuracy for each model using P-value and R squared
- Get tuned hyper parameters for each type of Machine Learning algorithms.
- Plot confusion Matrix

Improving Model

- Apply feature Engineering to improve model performance

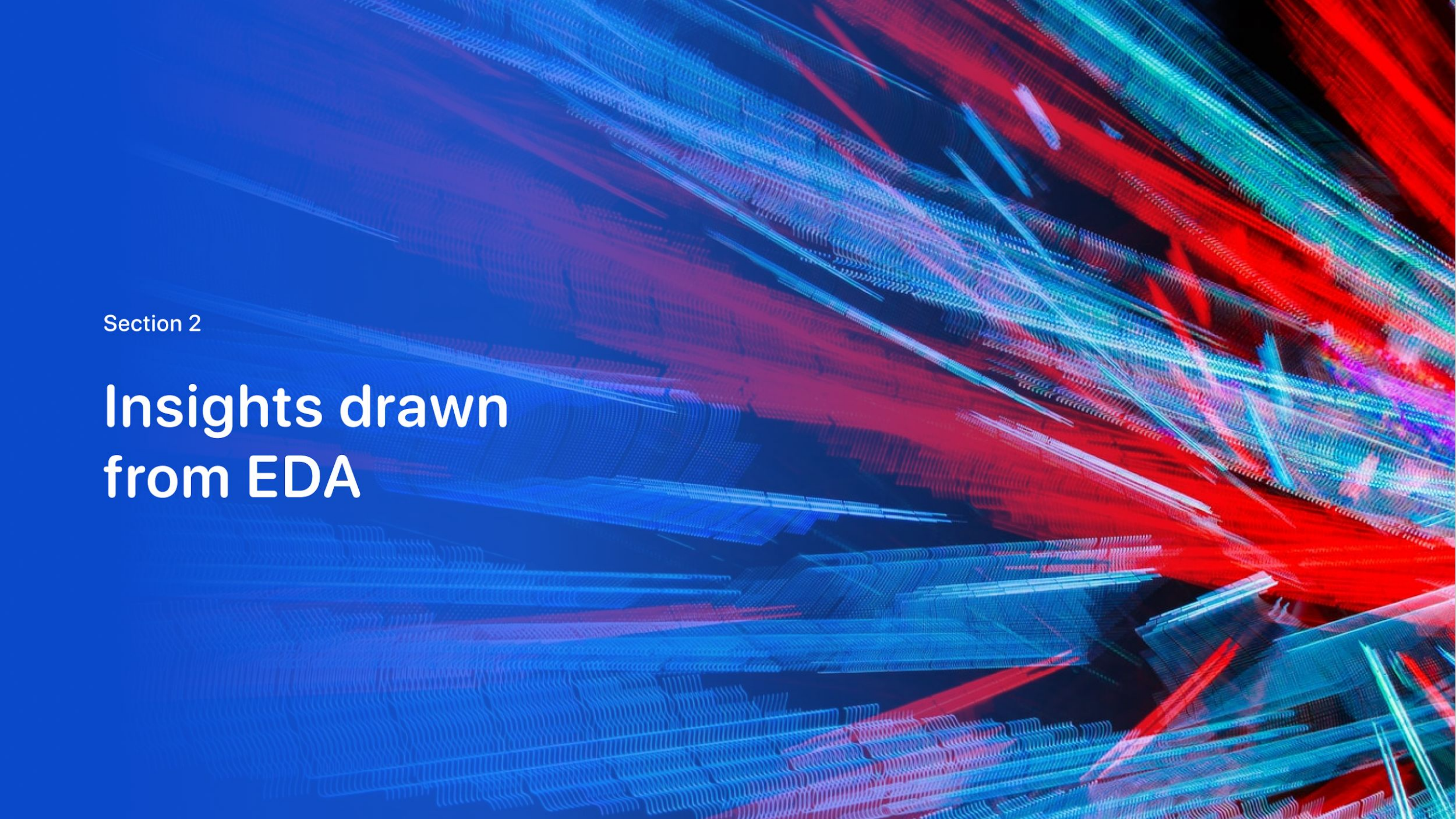
Determining the best Classification model

- The model with the best accuracy score on the test dataset is chosen as best performing model.

GITHUB LINK: LINK

Results

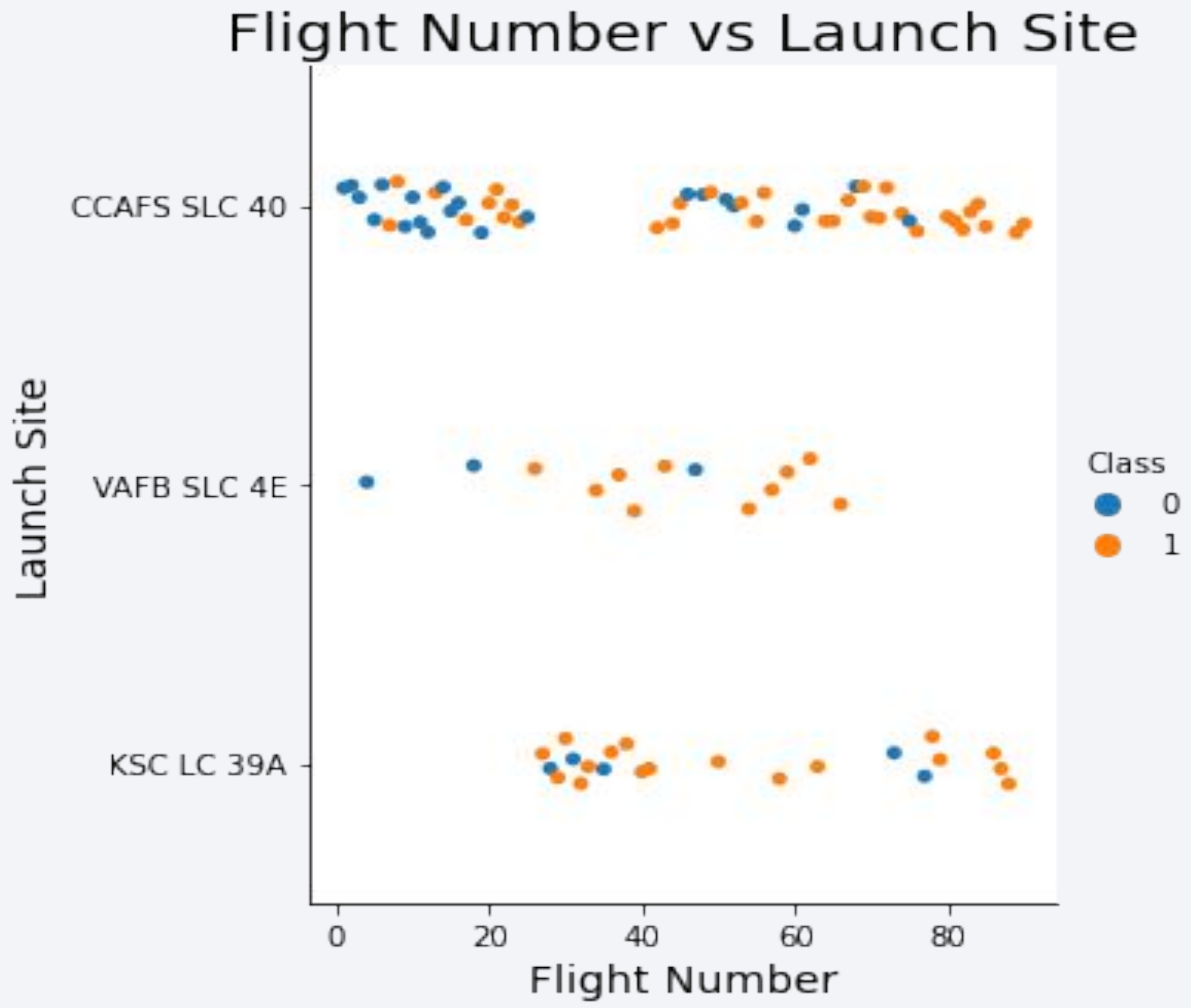
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



Section 2

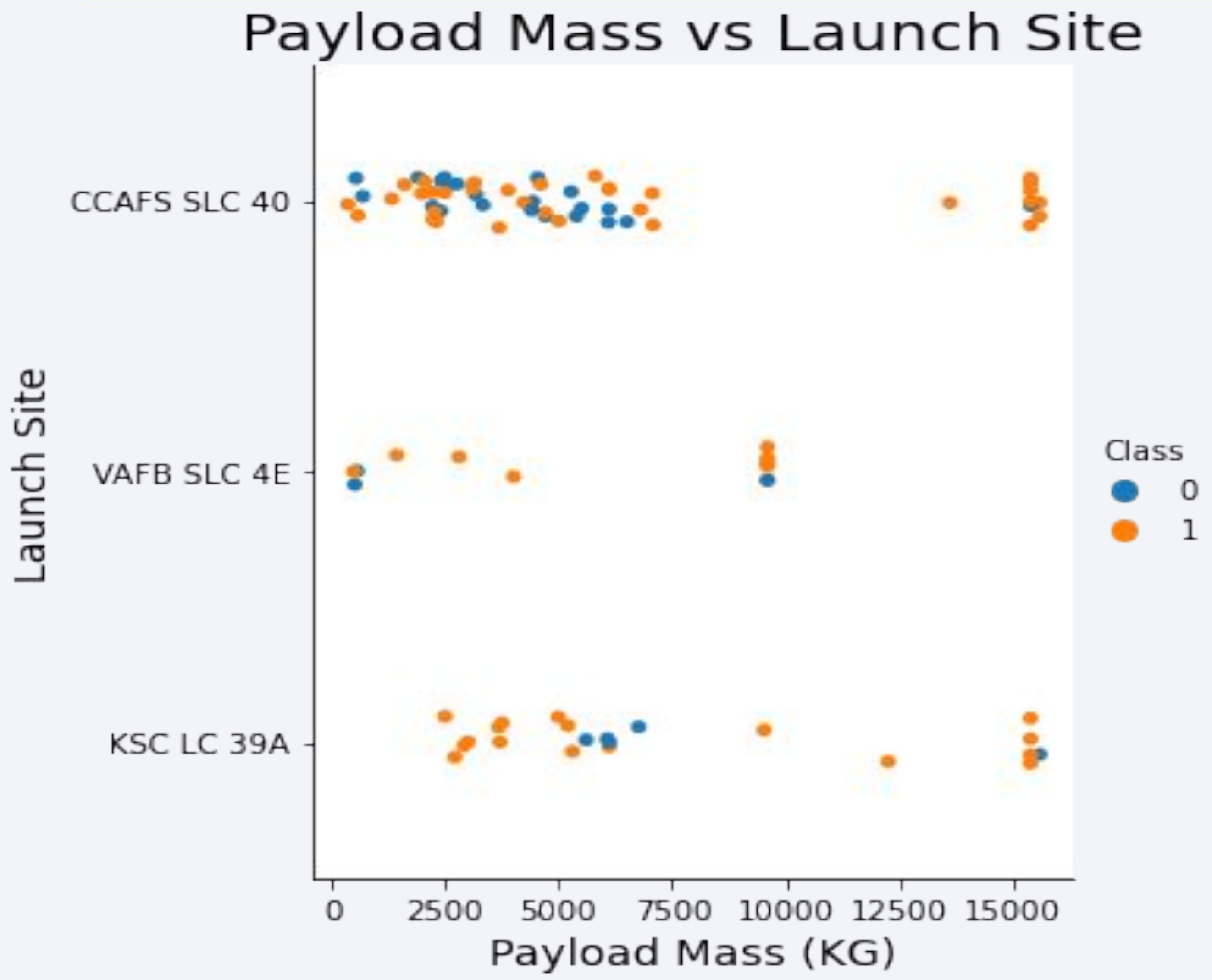
Insights drawn from EDA

Flight Number vs. Launch Site



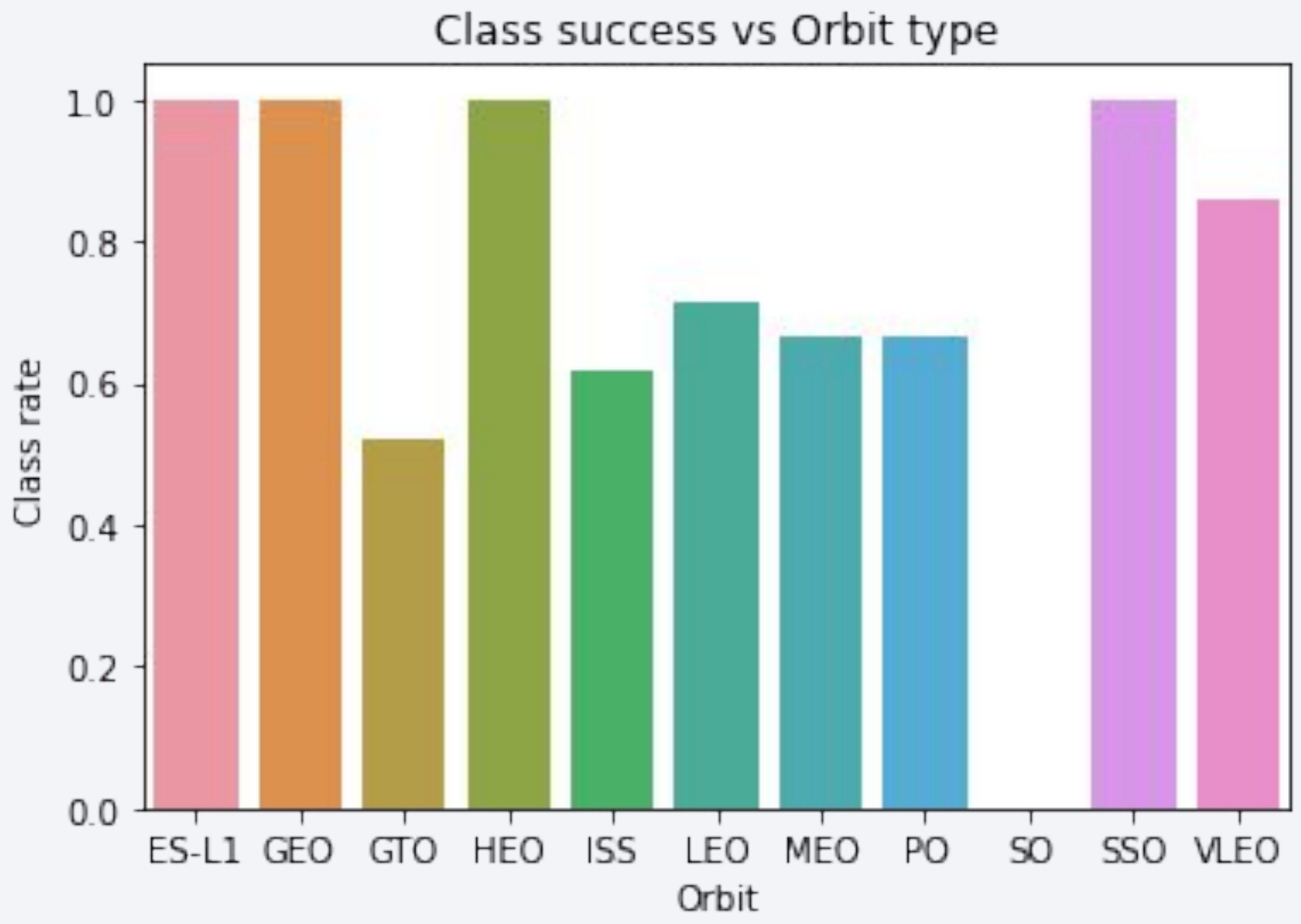
- From the scatter plot, we see that in general the more flights taken the higher the success rate

Payload vs. Launch Site



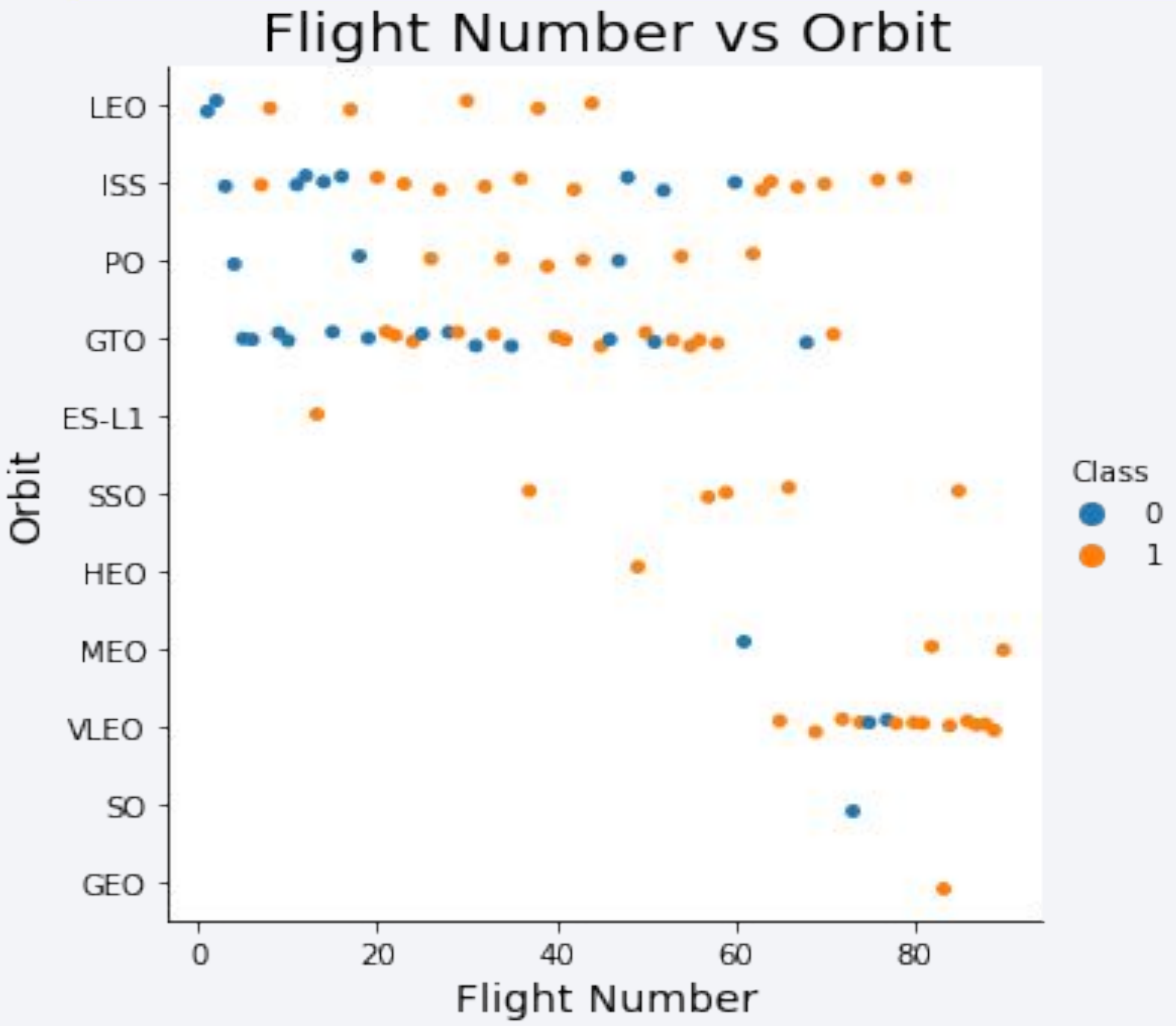
- From the scatter plot above, there is no clear relationship between the Mass of the payload and the different Launch Sites.

Success Rate vs. Orbit Type



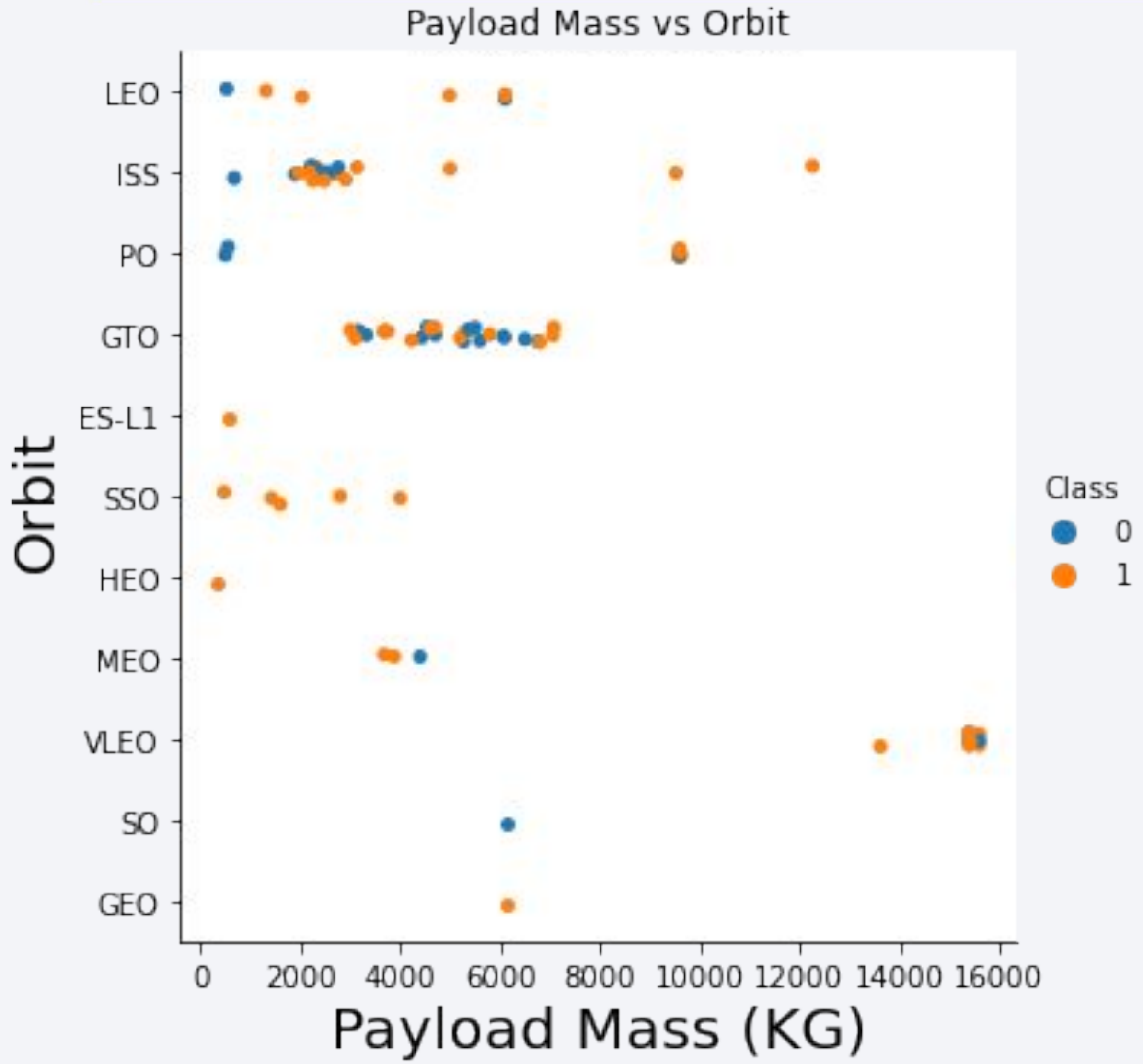
- From the bar chart, we observe that Orbit types ES-LI, GEO, HEO AND SSO have high rates of success.

Flight Number vs. Orbit Type



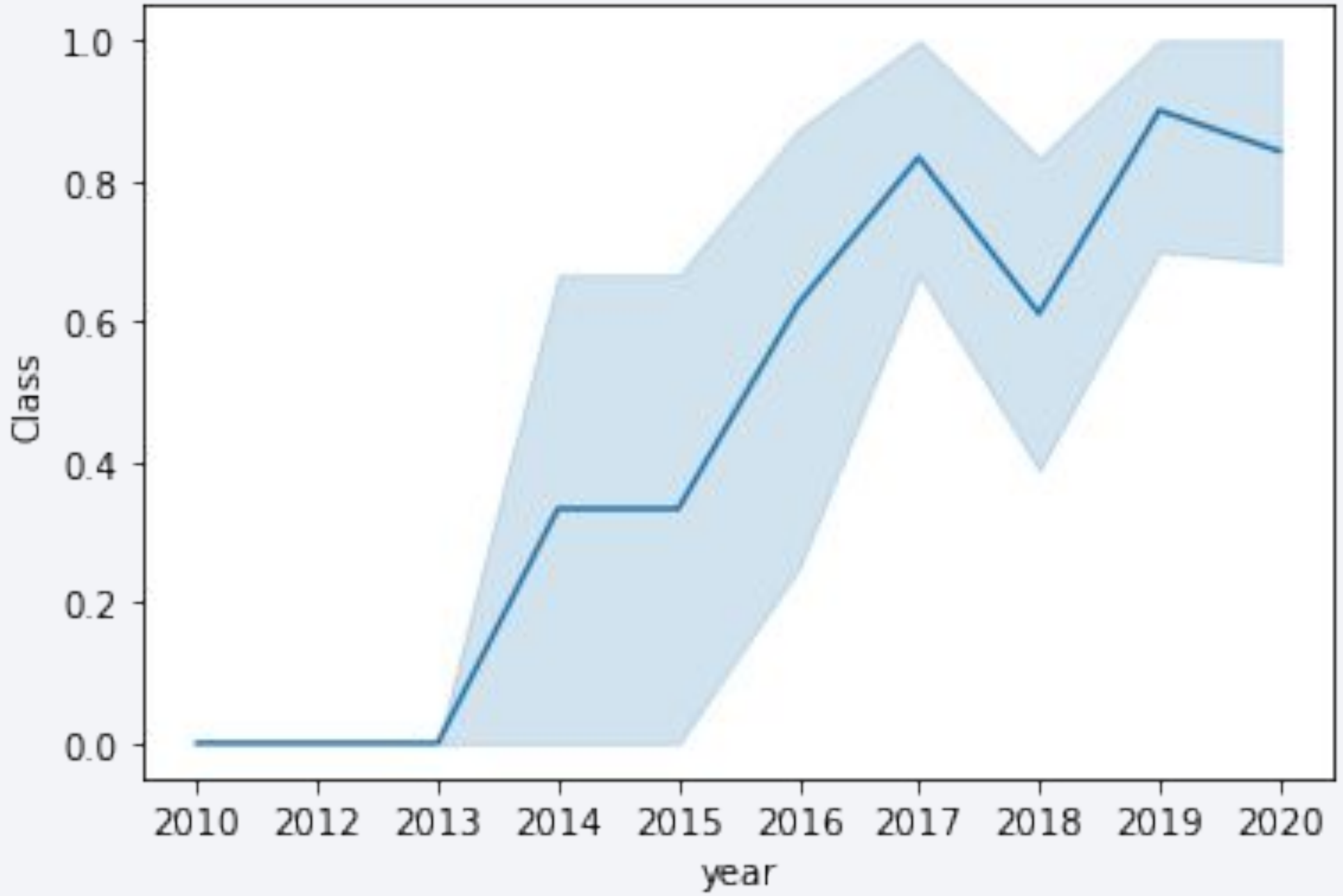
- From the scatterplot, we observe that in the LEO orbit the success seems to be related to the number of flights. There is no clear relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



- We observe that the Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

Launch Success Yearly Trend



From the Line Chart we can observe that the success rate since 2013 was steadily increasing till 2020.

All Launch Site Names

SQL QUERY - %sql SELECT distinct (Launch_Site) as DLAUNCH from SPACEX2

RESULT -

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

- This shows us all the launch sites where launch attempts were carried out.

Launch Site Names Begin with 'CCA'

SQL Query - %sql SELECT * FROM SPACEX2 WHERE LAUNCH_SITE LIKE 'CCA%' limit 5.

RESULT -

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

This query displays the first 5 records whose launch sites names begin with 'CCA'

Total Payload Mass

SQL Query - %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEX2 WHERE CUSTOMER = 'NASA (CRS)‘

RESULT -

SUM
45596

This query calculates and displays the total sum of payload mass for only rows with “NASA(CRS)” as Customers.

Average Payload Mass by F9 v1.1

SQL Query - %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEX2 WHERE BOOSTER_VERSION = 'F9 v1.1'

RESULT –

AVG
2928.4

This query calculates and displays the average payload mass of all rows whose booster version is “F9 v1.1”

First Successful Ground Landing Date

SQL Query - %sql SELECT MIN(DATE) FROM SPACEX2 WHERE Landing__Outcome = 'Success (ground pad)'

RESULT –

MIN DATE
2015-12-22

This query shows the date of the first successful landing in ground pad.

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL Query - %sql SELECT BOOSTER_VERSION FROM SPACEX2 WHERE Landing__Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000

RESULT –

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

This Query displays the names of boosters which have success in drone ship and a payload mass between 4000 and 6000.

Total Number of Successful and Failure Mission Outcomes

SQL Query - %sql SELECT COUNT(MISSION_OUTCOME) FROM SPACEX2 WHERE MISSION_OUTCOME = 'Success' OR MISSION_OUTCOME = 'Failure (in flight)‘

RESULT -

Total
100

This query shows the total number of successful and failure mission outcomes.

Boosters Carried Maximum Payload

SQL Query - %sql SELECT BOOSTER_VERSION FROM SPACEX2 WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX2)

RESULT -

This Query displays the names of booster versions which carried the maximum payload mass

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

2015 Launch Records

SQL Query - %sql select landing__outcome, booster_version, launch_site from Spacex2 where YEAR(DATE)='2015' AND Landing__outcome LIKE 'Fail%‘

RESULT –

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

This query displays the failed landing outcome, booster version and launch site of records for the year 2015.

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

SQL Query - %sql select * FROM SPACEX2 WHERE Landing__Outcome LIKE 'Success%' AND (DATE between '2010-06-04' and '2017-03-20') ORDER BY date DESC

This Query displays records between dates (2010-06-04 to 2017-03-20) in descending order whose landing outcome begins with “Success”.

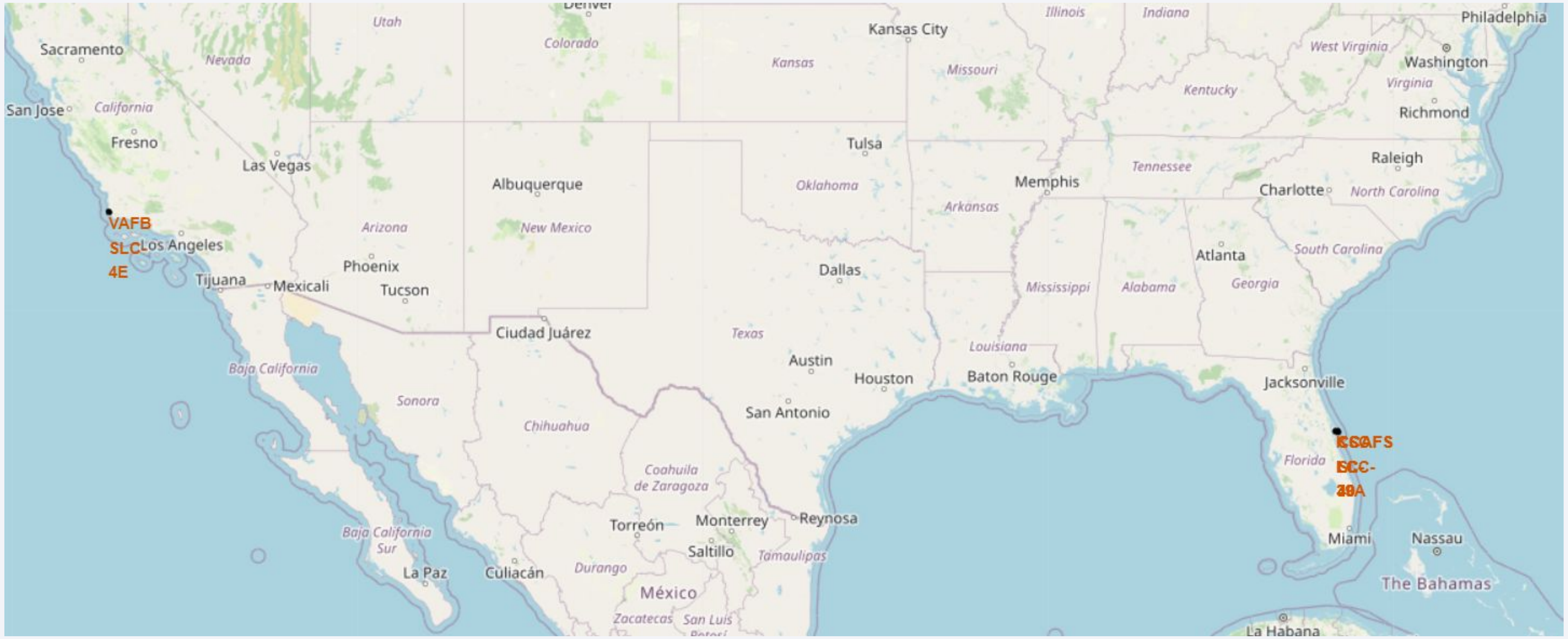
RESULT -

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Success (drone ship)
2016-08-14	05:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-07-18	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Success (drone ship)
2016-05-06	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-04-08	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (drone ship)
2015-12-22	01:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

Section 4

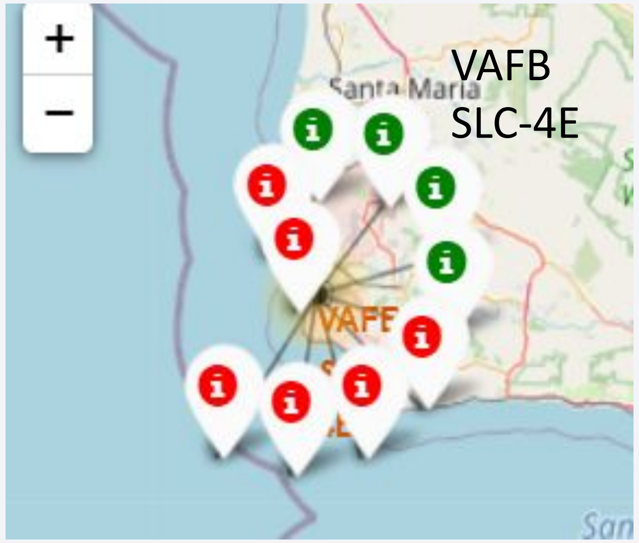
Launch Sites Proximities Analysis

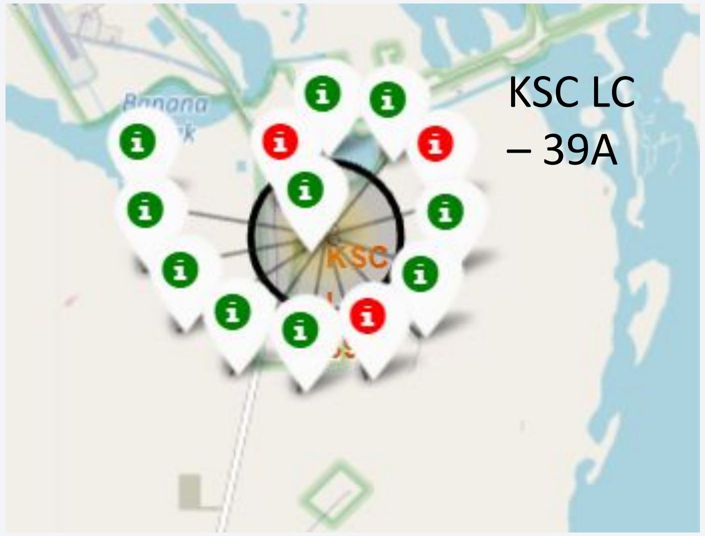
All Launch sites on the world map

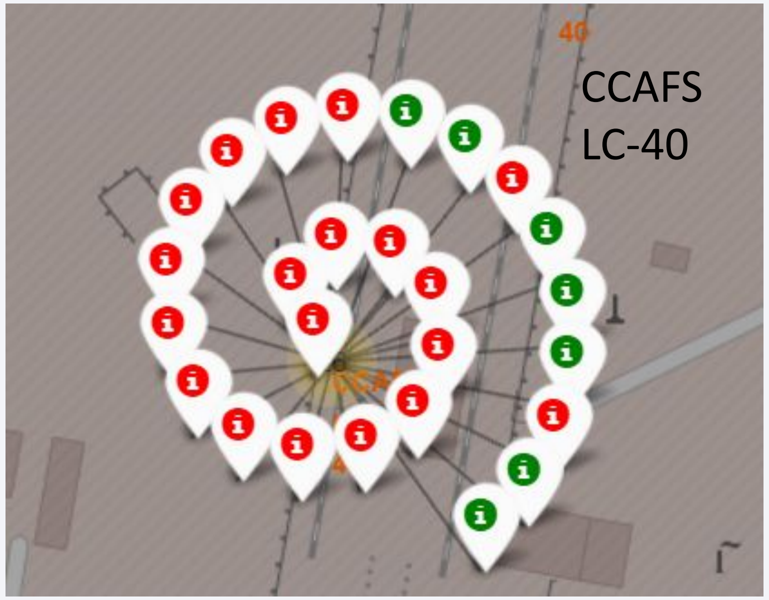


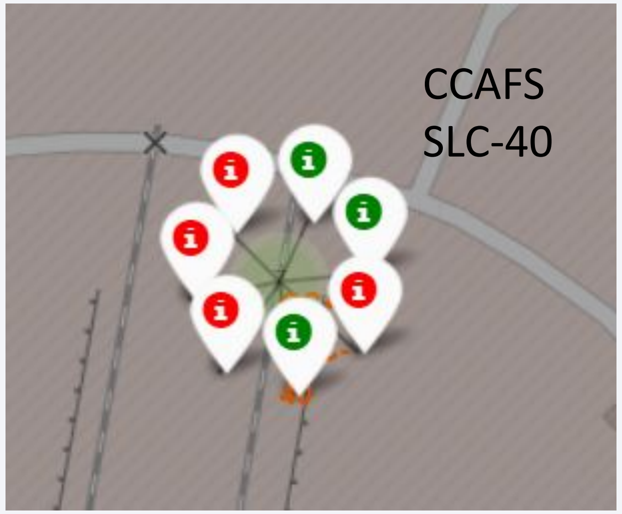
The map above shows where all the unique launch sites are. Three are located in Florida while one is located in California.

Marked Success/failed launches for each launch site



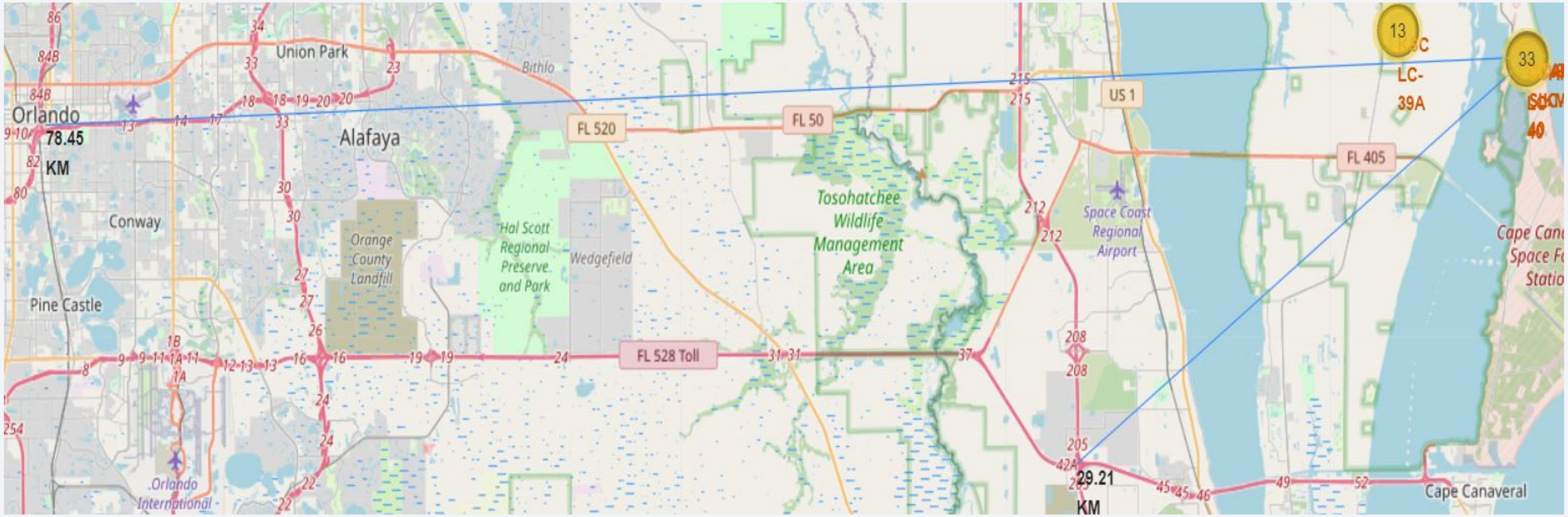






The four maps show the successful/ failed landing outcomes for each launch site. Green Marker indicates successful launch while Red Marker indicated a failed launch.

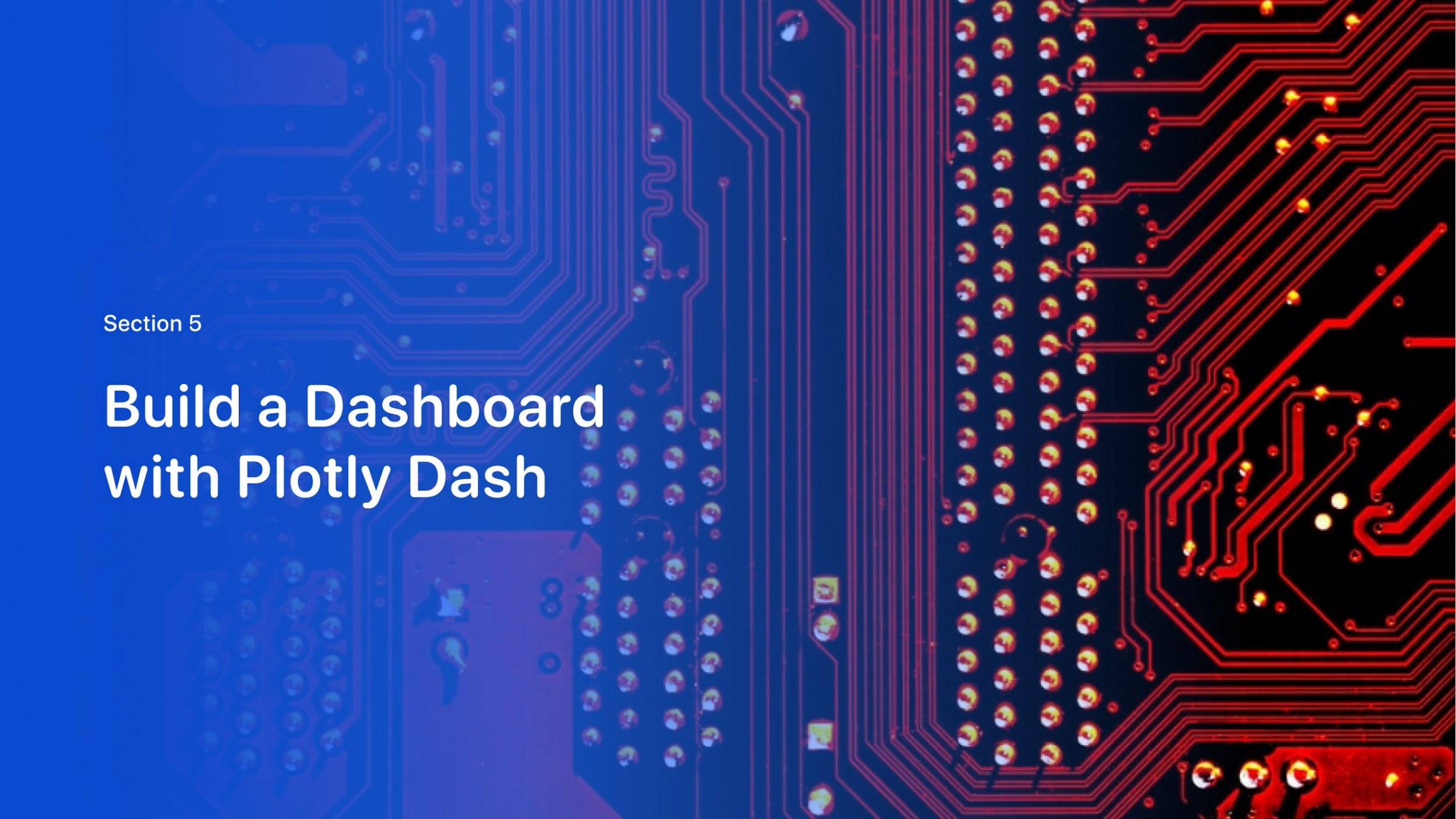
Distances between each launch site to its proximities



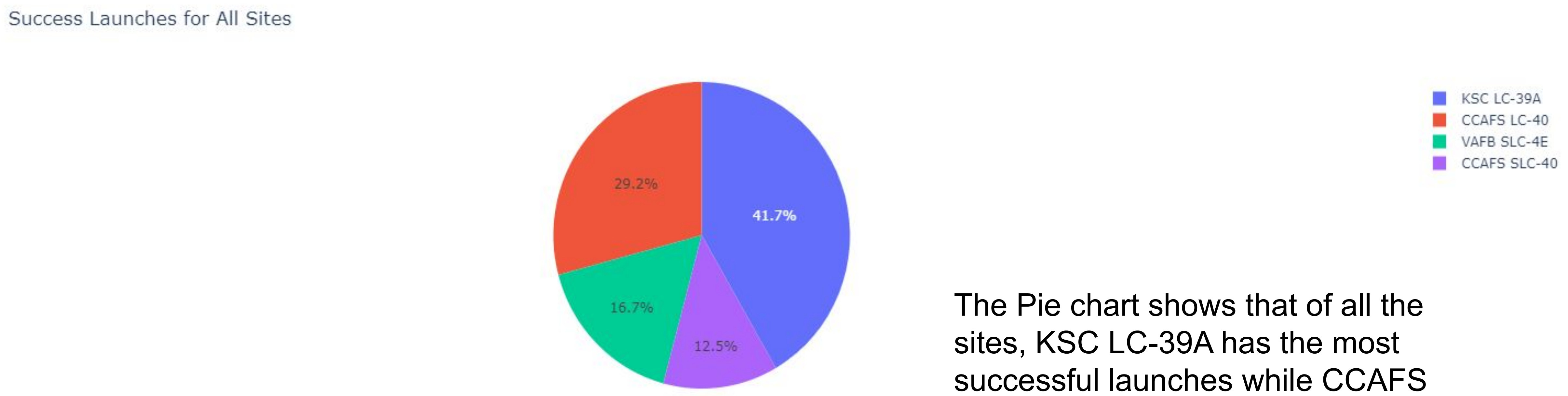
- The map above shows the distance between launch sites which begin with 'CCAFS' to various major highways and their distances are indicated.

Section 5

Build a Dashboard with Plotly Dash

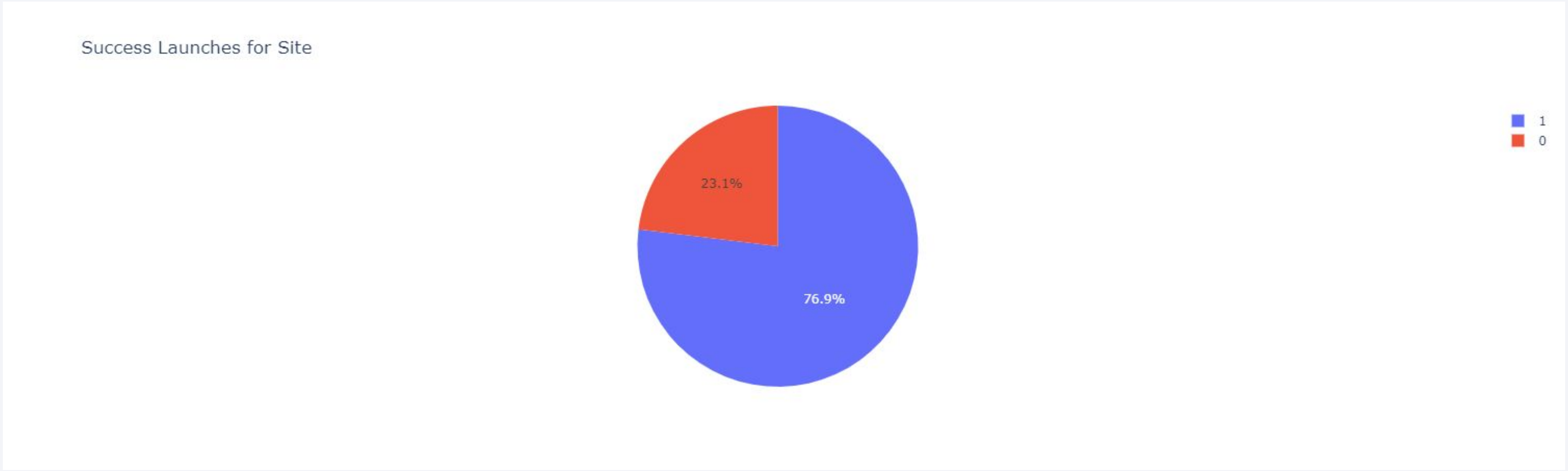


Launch success count for all sites



The Pie chart shows that of all the sites, KSC LC-39A has the most successful launches while CCAFS SLC-40 had the least successful launches.

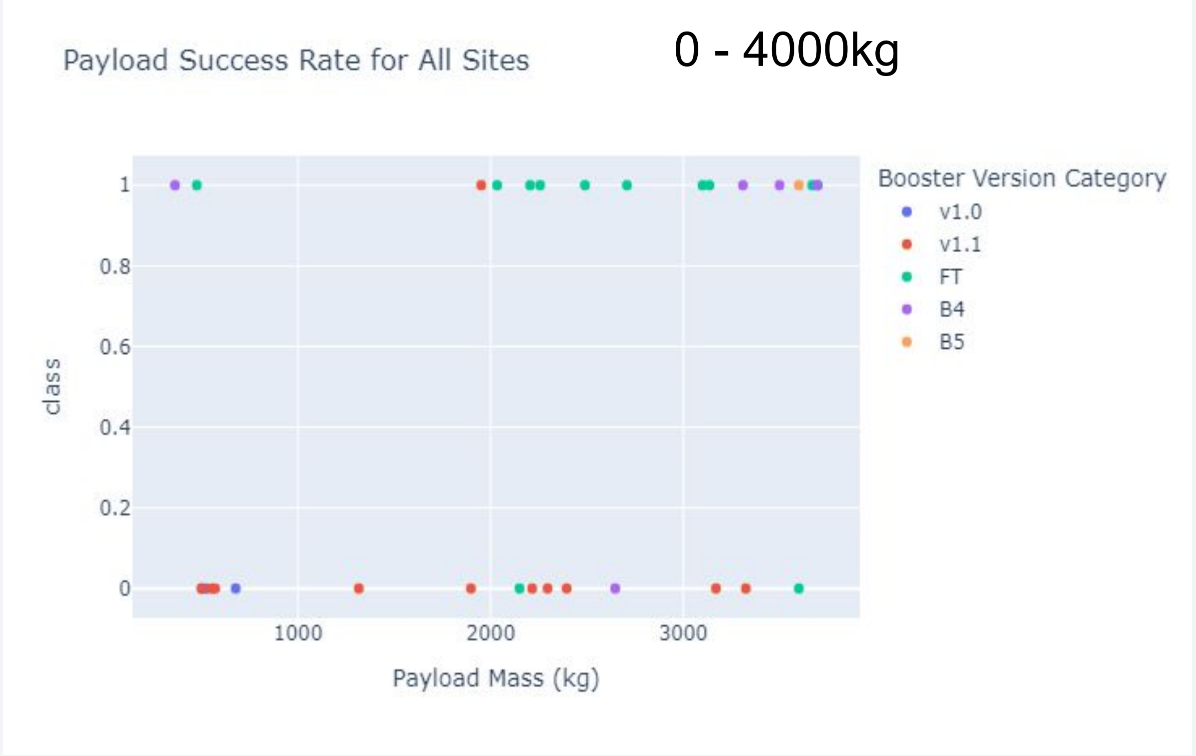
Pie chart for the launch site with highest launch success ratio



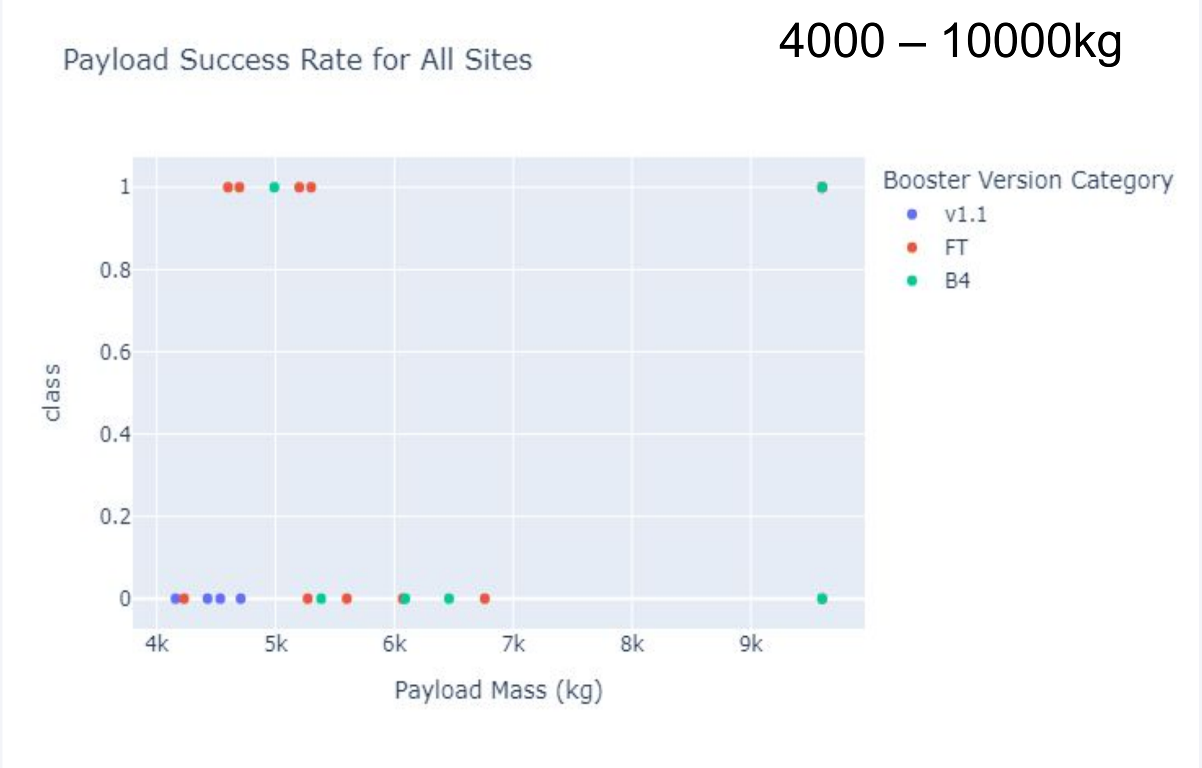
The Pie chart above shows that KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate. Thus making it the launch site with the highest launch success ratio

Payload vs. Launch Outcome scatter plot for all Sites at different payload range

0 - 4000kg



4000 – 10000kg

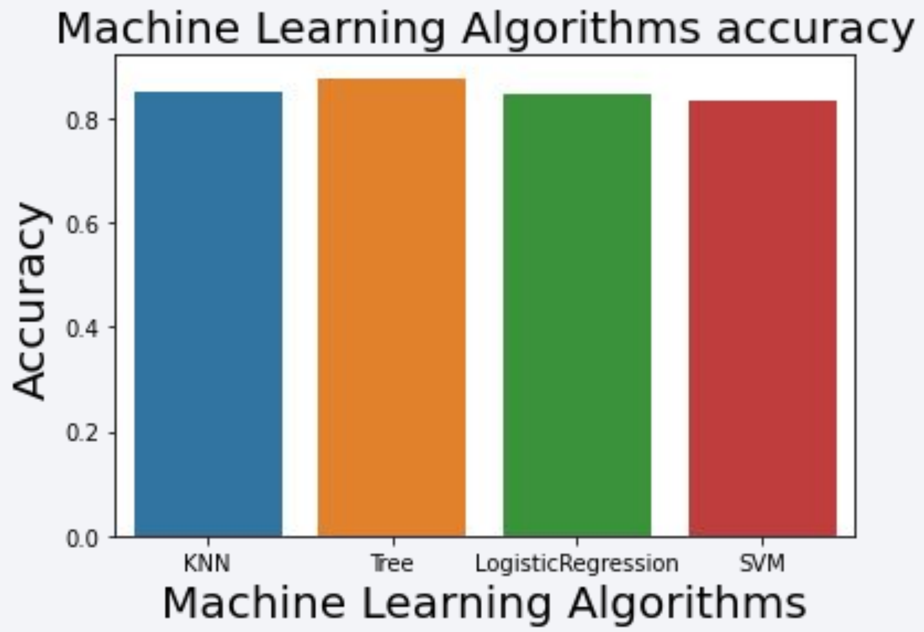


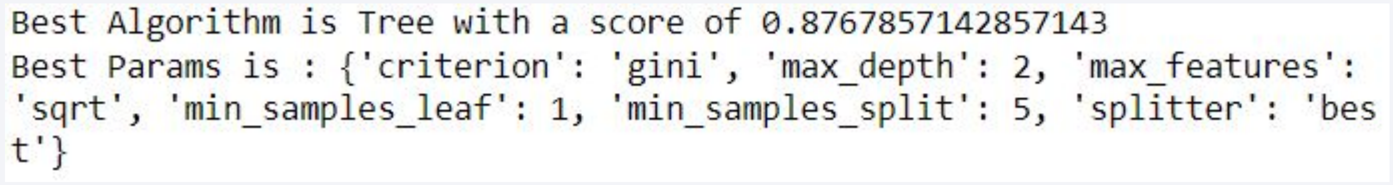
From the scatter plot graphs above, we see that low weight payload mass have more successful launches than heavy weighted payload mass

Section 6

Predictive Analysis (Classification)

Classification Accuracy

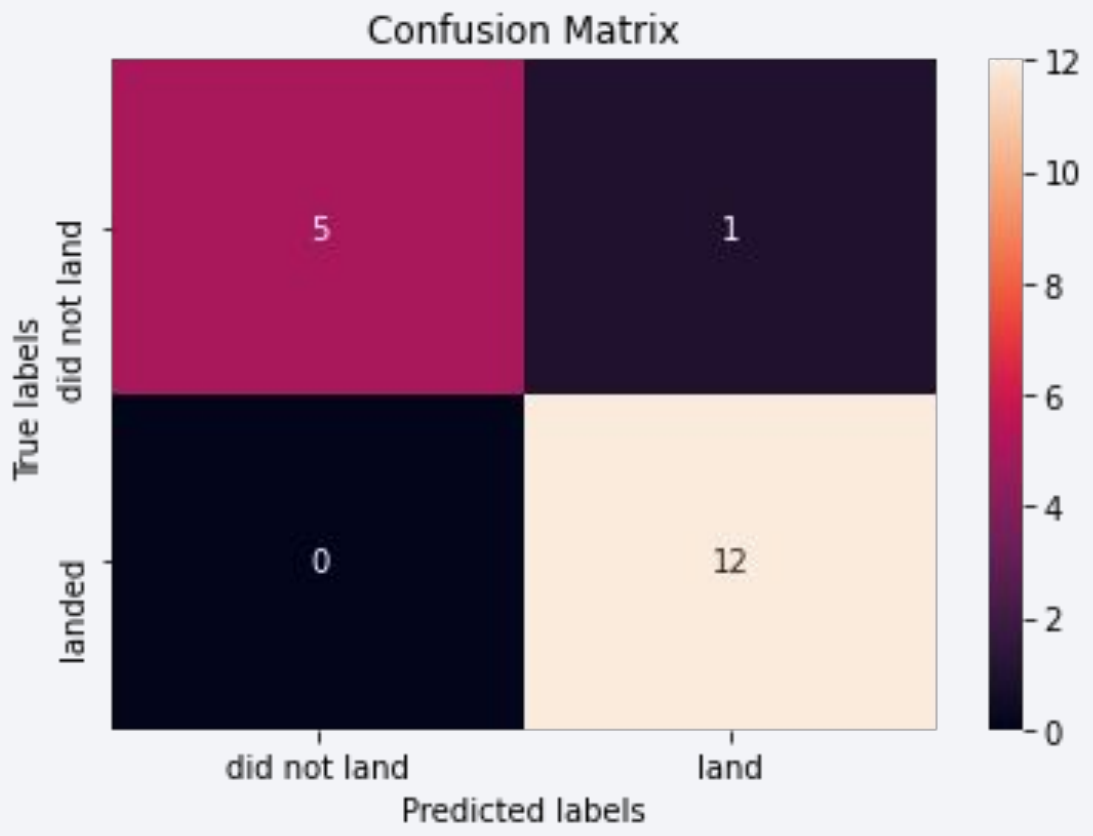




```
Best Algorithm is Tree with a score of 0.8767857142857143
Best Params is : {'criterion': 'gini', 'max_depth': 2, 'max_features':
'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitter': 'bes
t'}
```

- From the Bar chart above, we see that Decision Tree Classifier model was the most accurate model with a score of 87.68%

Confusion Matrix



This is the confusion matrix for the Decision Tree Classifier model. Upon close examination, we see that 17 records were predicted correctly while only one record was incorrectly misclassified as landed. This is a good result and thus makes our Model suitable for predicting future landing outcomes.

Conclusions

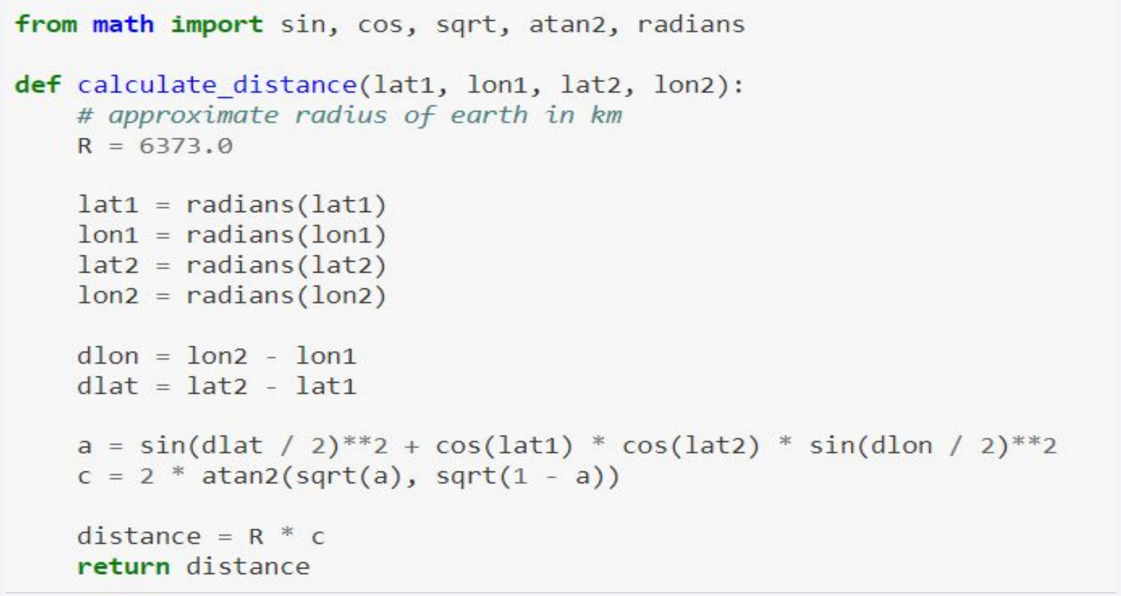
- From our detailed analysis, we can conclude that Low weighted payloads perform better than heavier payloads. This information can prove useful to SpaceX rivals wanting to minimize risk to their rocket launches.
- We see that KSC LC-39A had the most successful launches from all the sites.
- We observed that Orbit GEO, HEO, SSO, ES-L1 has the best success rate.
- The Decision Tree Classifier Algorithm was the most suitable ML algorithm for predicting landing outcome for the dataset.
- The more flights taken the better the chance of a successful flight.

Appendix

Haversine Formula - The Haversine formula calculates the shortest distance between two points on a sphere using their latitudes and longitudes measured along the surface.

The Haversine is: $d = 2r sin^{-1}\left(\sqrt{sin^2\left(\frac{\Phi_2 - \Phi_1}{2}\right) + cos(\Phi_1)cos(\Phi_2)sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$

- We applied the Haversine formula when calculating the proximity between each launch site to major highways, railways, etc. This is shown in the code snippet below:



```
from math import sin, cos, sqrt, atan2, radians

def calculate_distance(lat1, lon1, lat2, lon2):
    # approximate radius of earth in km
    R = 6373.0

    lat1 = radians(lat1)
    lon1 = radians(lon1)
    lat2 = radians(lat2)
    lon2 = radians(lon2)

    dlon = lon2 - lon1
    dlat = lat2 - lat1

    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    distance = R * c
    return distance
```

Thank you!