IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Anele Mucavele
11 August 2024

Outline

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

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 e analytics demo in screenshots
- Predictive analysis results

Introduction

- **Project background and context**
- We predicted 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**
- What influences if the rocket will land successfully?
- The effect each relationship with certain rocket variables will impact in determining the success rate of a successful landing.
- What conditions does SpaceX have to achieve to get the best results and ensure the best rocket success landing rate.

Section 1

Methodology

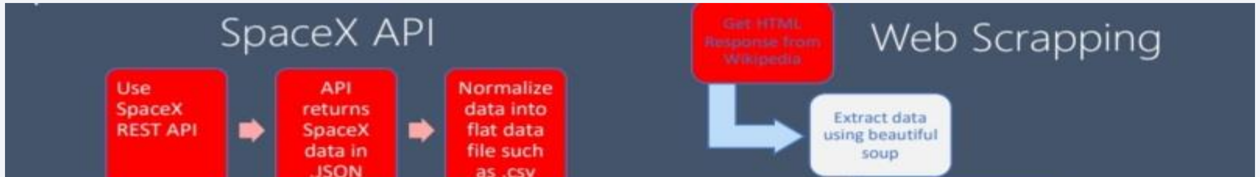
Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API
  - (Web Scrapping) from Wikipedia
- Perform data wrangling
  - One Hot Encoding data fields for Machine Learning and dropping irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Plotting : Scatter Graphs, Bar Graphs to show relationships between variables to show patterns of data.
  - Performed interactive visual analytics using Folium
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

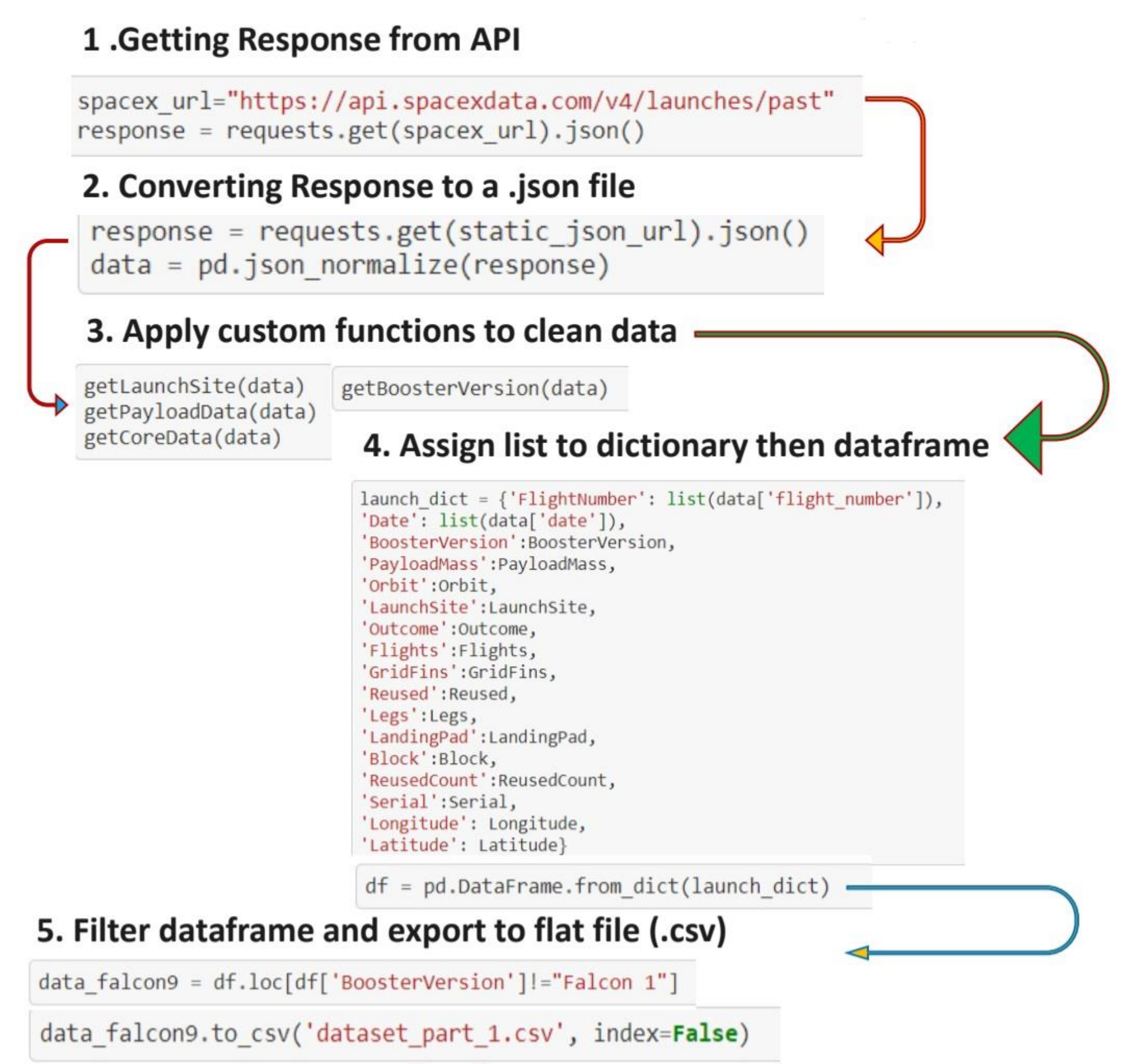
- **Describe how data sets were collected.**
- **You need to present your data collection process use key phrases and flowcharts**
- *The following datasets was collected:*
- We worked with SpaceX launch data that is gathered from the SpaceX REST API.
- This API will give us data about launches, including information about the rocket used,
- payload delivered, launch specifications, landing specifications, and landing outcome.
- Our goal is to use this data to predict whether SpaceX will attempt to land a rocket or not.
- The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/.
- Another popular data source for obtaining Falcon 9 Launch data is web scraping Wikipedia using BeautifulSoup.


SpaceX API
Use SpaceX REST API
API returns SpaceX data in JSON
Normalize data into flat data file such as .csv
Get HTML Response from Wikipedia
Web Scrapping
Extract data using beautiful soup

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose



1 .Getting Response from API

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

2. Converting Response to a .json file

```
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,
'GridFins':GridFins,
'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 to flat file (.csv)

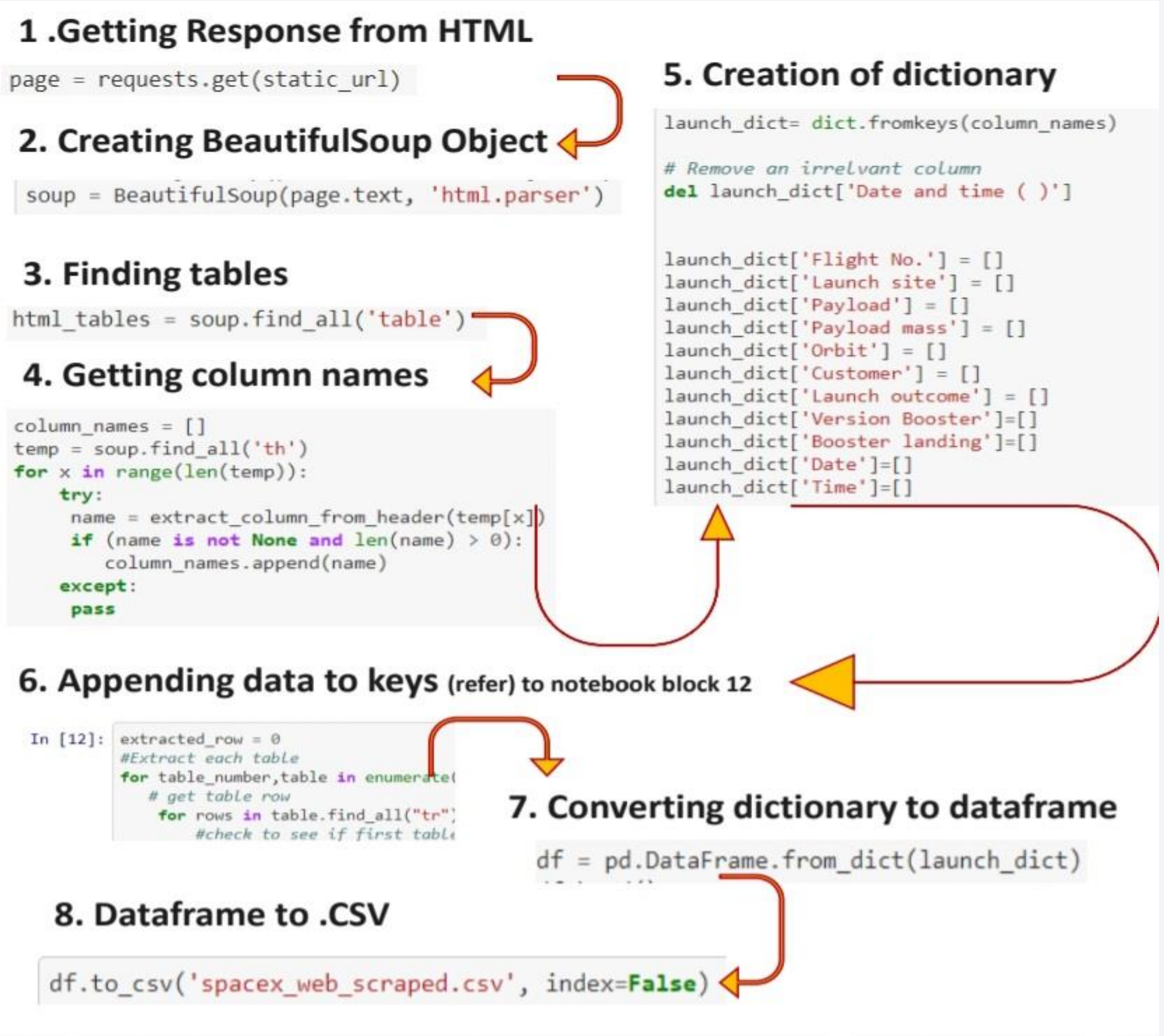
```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

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

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose



1 .Getting Response from HTML

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

2. Creating BeautifulSoup Object

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

3. Finding tables

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

4. Getting 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
```

5. 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 (refer) to notebook block 12

```
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
```

7. Converting dictionary to dataframe

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

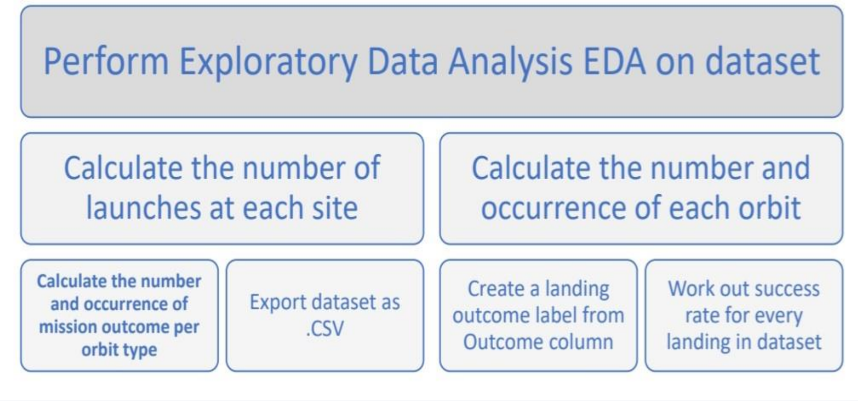
8. Dataframe to .CSV

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

Data Wrangling

- Describe how data were processed
- You need to present your data wrangling process using key phrases and flowcharts
- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose
- 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. We mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

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 mission outcome per orbit type

Export dataset as .CSV

Create a landing outcome label from Outcome column

Work out success rate for every landing in dataset

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

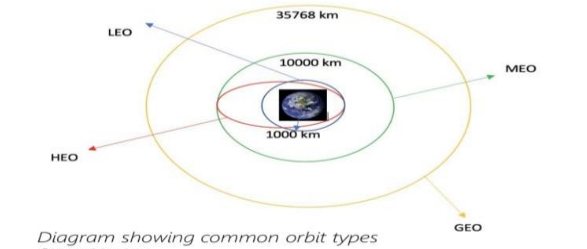


Diagram showing common orbit types SpaceX uses

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

- 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
- Line graphs are useful in that they show data variables and trends very clearly and can help to make predictions about the results of data not yet recorded
- A bar diagram makes it easy to compare sets of data between different groups at a glance.
- The graph represents categories on one axis and a discrete value in the other. The goal is to show the relationship between the two axes. Bar charts can also show big changes in data over time.
- Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation . Scatter plots usually consist of a large body of data

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

- Performed SQL queries to gather information about the dataset. For example of some questions we were asked about the data we needed information about. Which we are using SQL queries to get the answers in the dataset :
- 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.

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
- Explain why you added those objects
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

- 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 dataframe 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
- Example of some trends in which the Launch Site is situated in.
- *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*

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
- Explain why you added those plots and interactions
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose
- Used Python Anywhere to host the website live 24/7 so your can play around with the data and view the data
- - 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.

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
- You need present your model development process using key phrases and flowchart
- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose
- **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

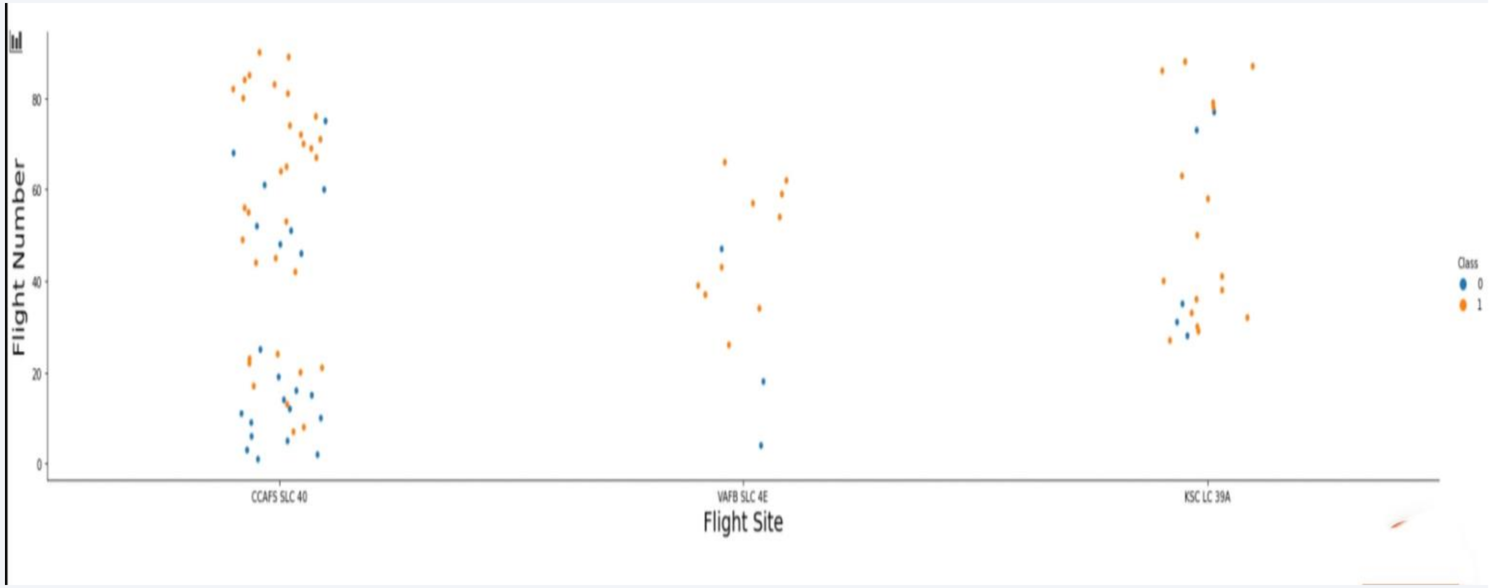
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

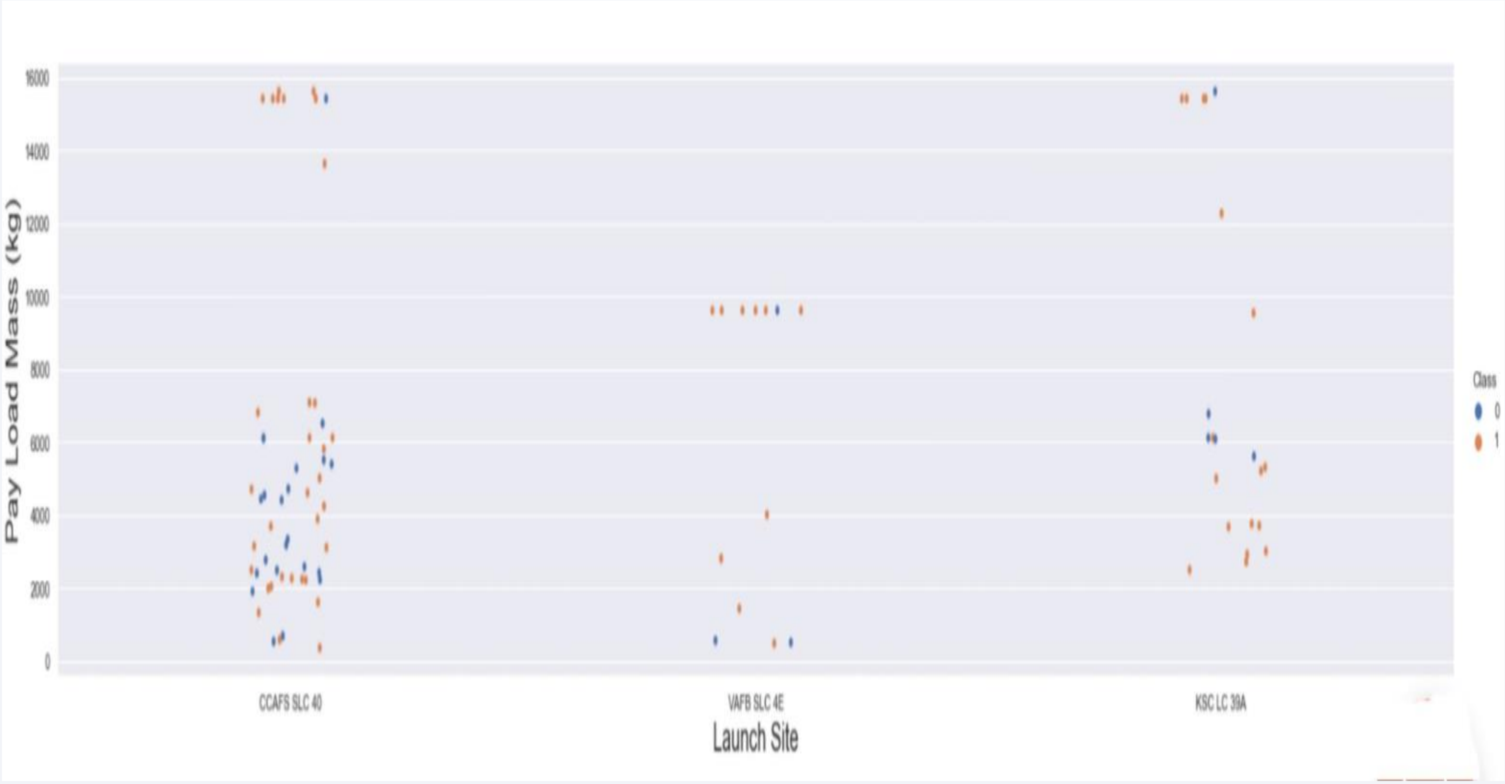
Flight Number vs. Launch Site

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



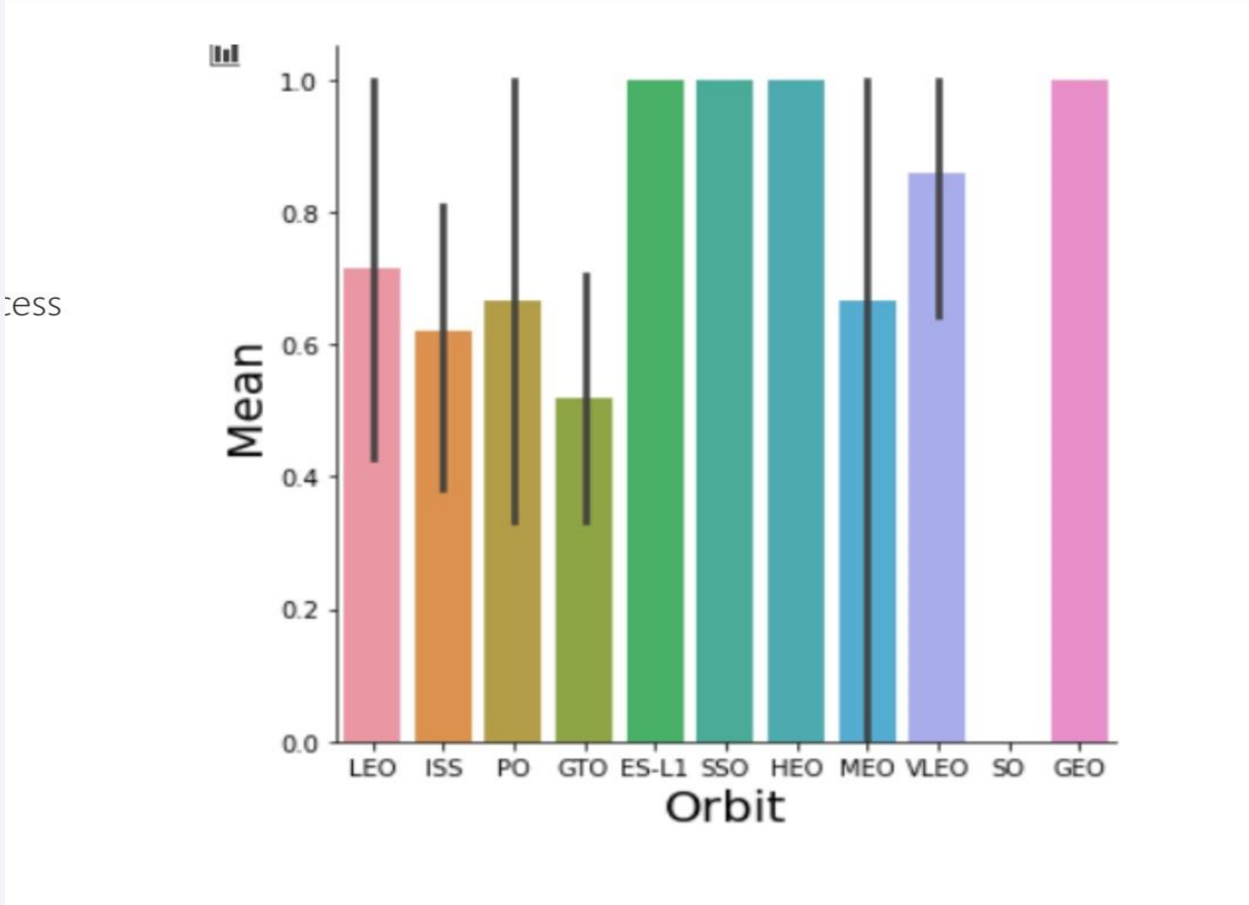
Payload vs. Launch Site

- *The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket. There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependent on Pay Load Mass for a success launch.*



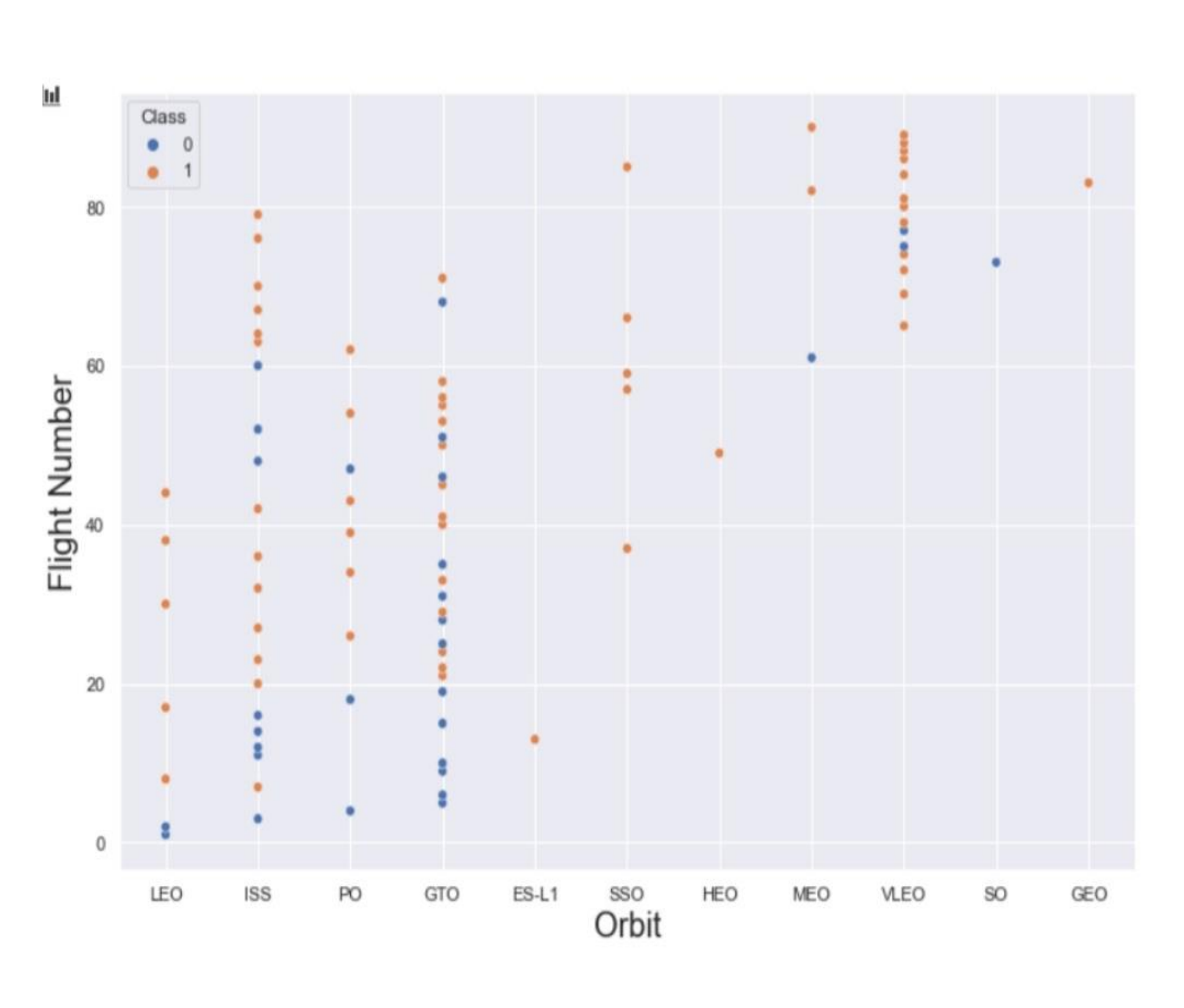
Success Rate vs. Orbit Type

- *Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate*



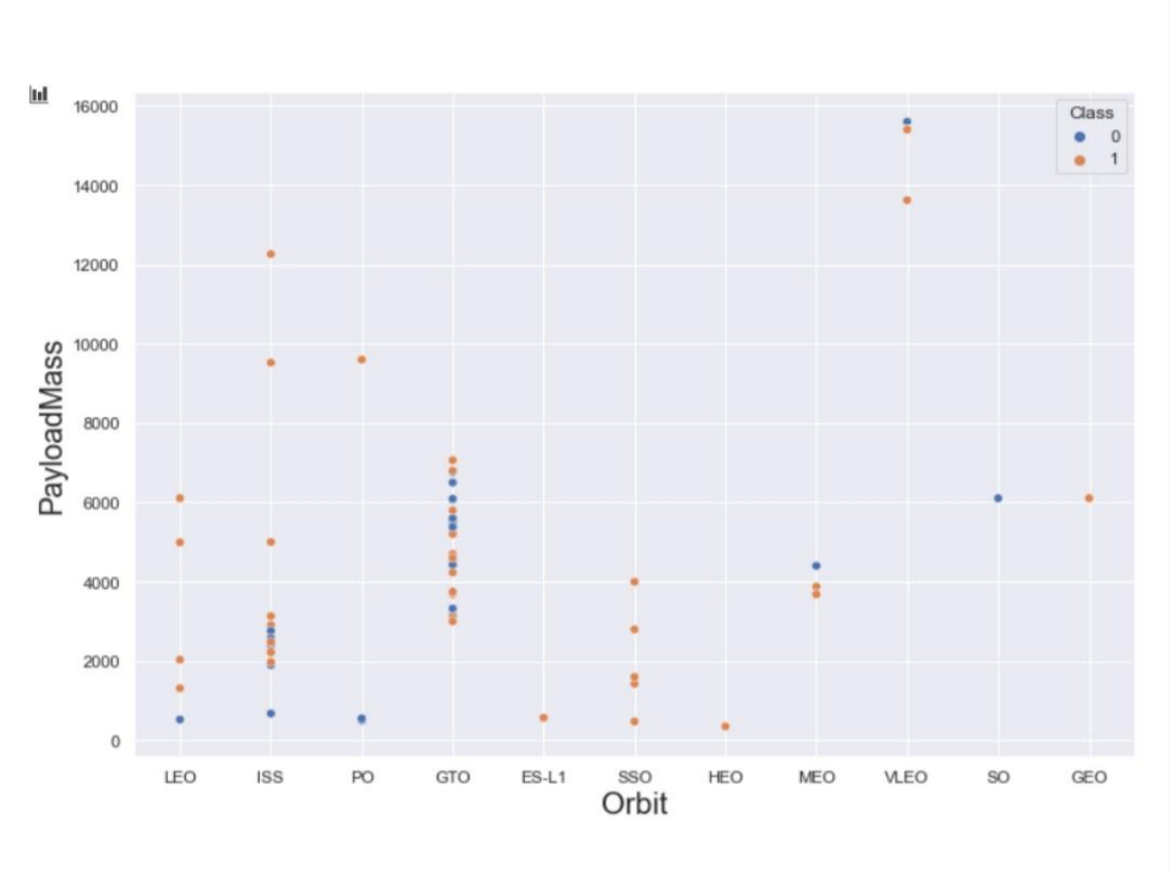
Flight Number vs. Orbit Type

- *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.*



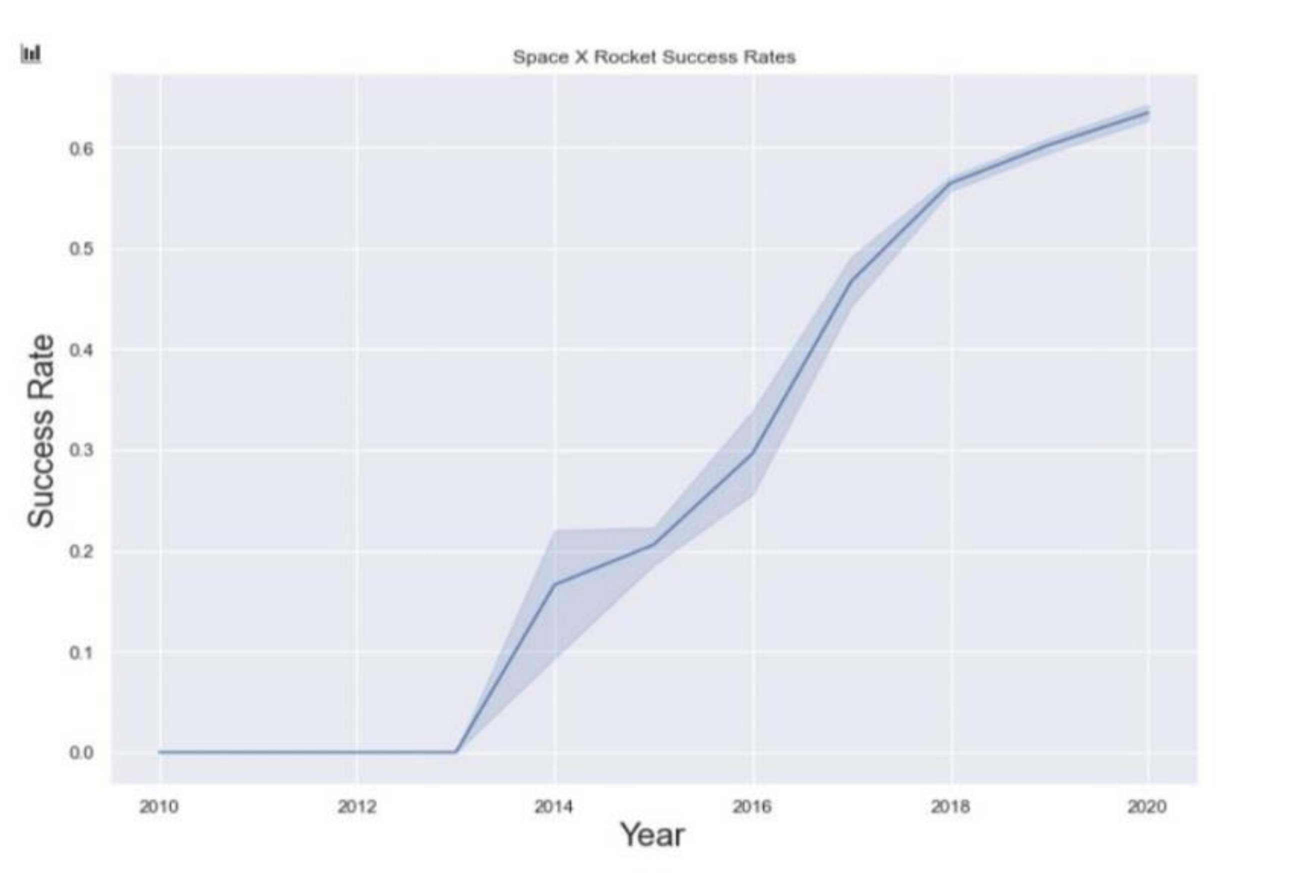
Payload vs. Orbit Type

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



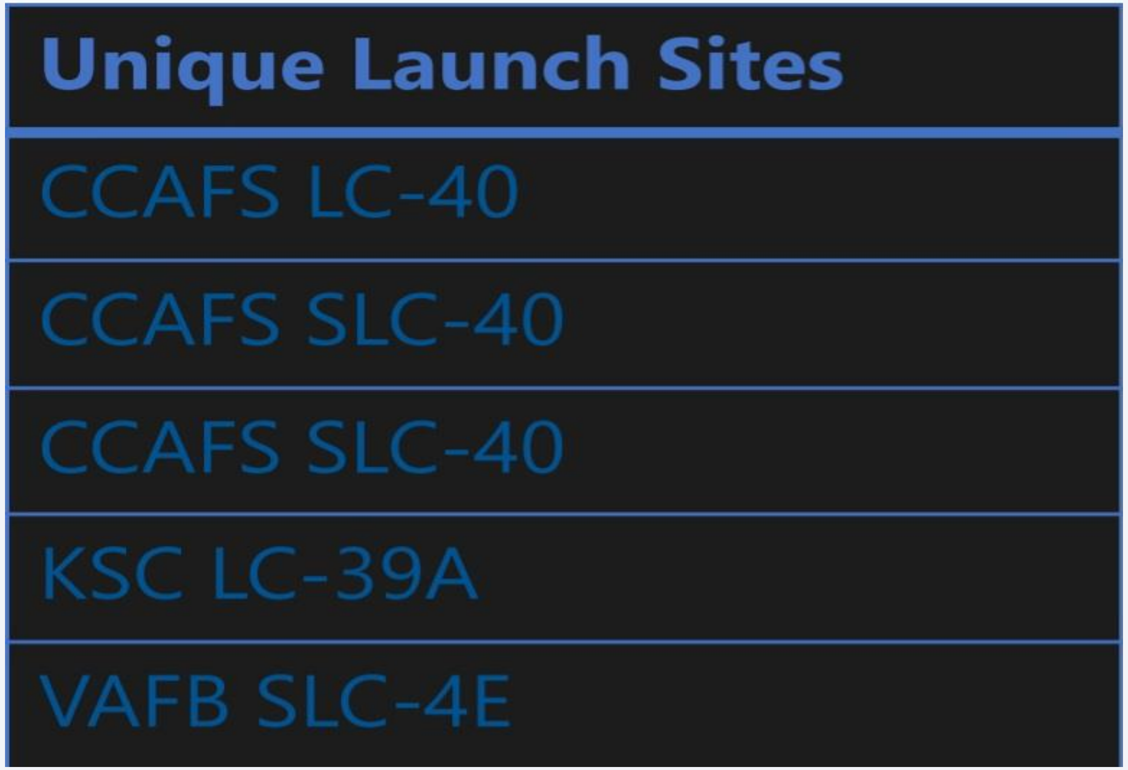
Launch Success Yearly Trend

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



All Launch Site Names

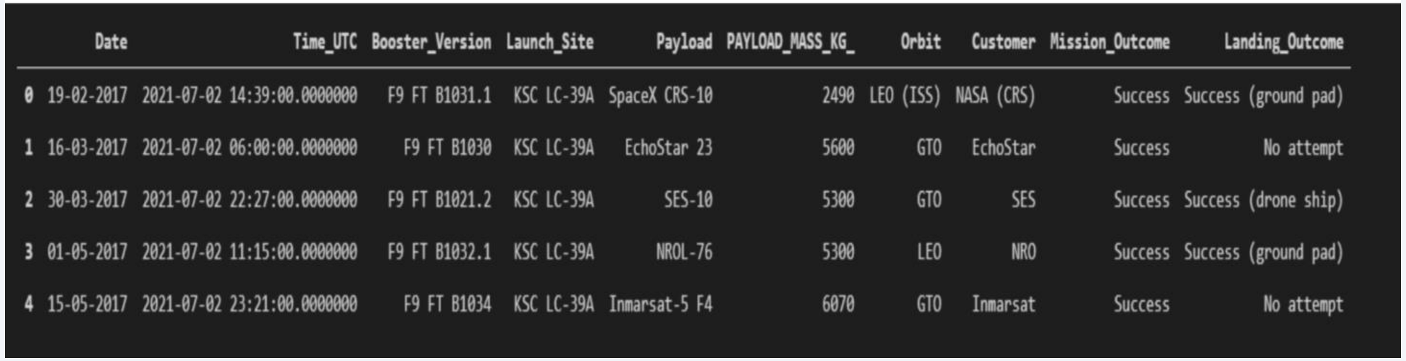
- **SQL QUERY**
- select DISTINCT Launch_Site
- from tblSpaceX
- **QUERY EXPLAINATION**
- Using the word DISTINCT in the query means that it will only
- show Unique values in the Launch_Site column from tblSpaceX



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

Launch Site Names Begin with 'CCA'

- SQL QUERY
- Select TOP 5 * from tblSpaceX
- WHERE Launch_Site LIKE ‘KSC%’

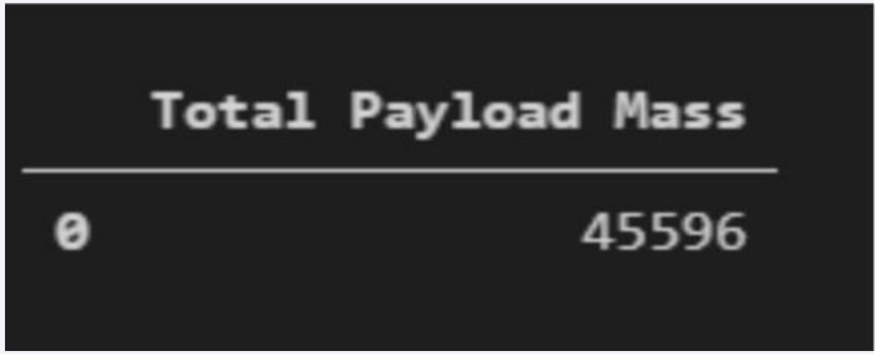


	Date	Time_UTC	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS_KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
0	19-02-2017	2021-07-02 14:39:00.0000000	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
1	16-03-2017	2021-07-02 06:00:00.0000000	F9 FT B1030	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

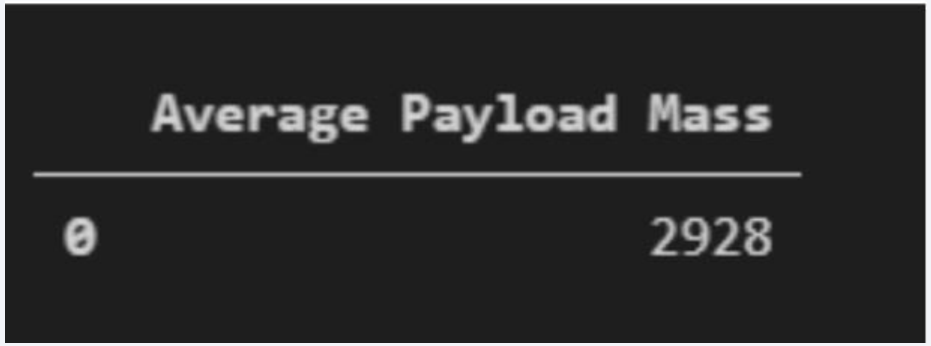
- SQL QUERY
- *Select SUM(PAYLOAD_MASS_KG_) TotalPayloadMass from tblSpaceX where Customer = ‘NASA (CRS)’”,’TotalPayloadMass*
- QUERY EXPLAINATION
- Using the function SUM summates the total in the column PAYLOAD_MASS_KG_
- The WHERE clause filters the dataset to only perform calculations on Customer NASA (CRS)



```
   Total Payload Mass
0               45596
```

Average Payload Mass by F9 v1.1

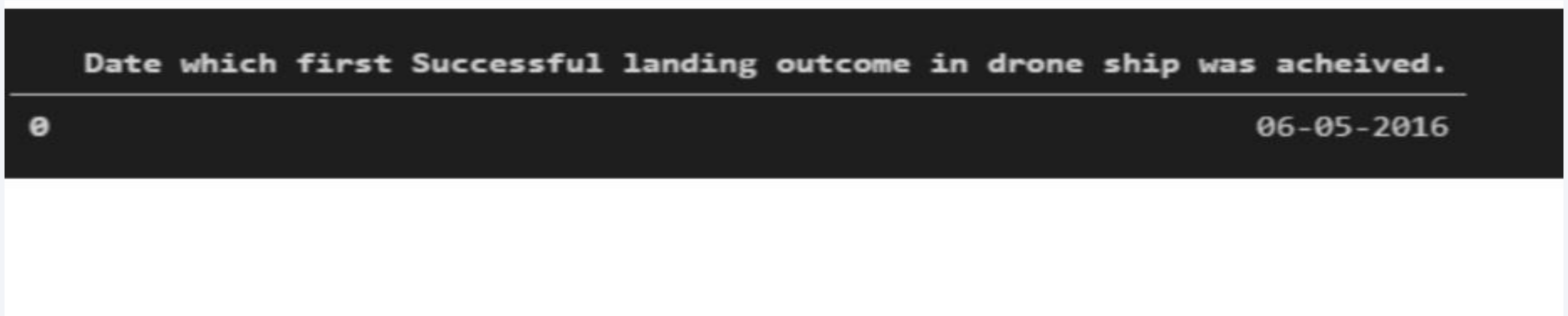
- SQL QUERY
- *Select AVG(PAYLOAD_MASS_KG_) AveragePayloadMass from tblSpaceX where Booster_Version = ‘F9 v1.1’*
- QUERY EXPLAINATION
- Using the function AVG works out the average in the column
- PAYLOAD_MASS_KG_
- The WHERE clause filters the dataset to only perform calculations on Booster_version F9 v1.1



```
     Average Payload Mass
0                    2928
```

First Successful Ground Landing Date

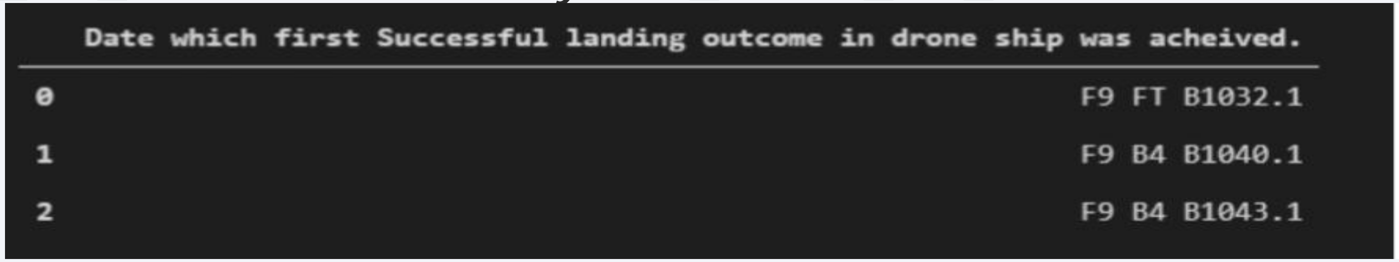
- SQL QUERY
- *Select MIN(Date) SLO from tblSpaceX where Landing_Outcome = ”Success (drone ship)”*
- QUERY EXPLAINATION
- Using the function MIN works out the minimum date in the column Date
- The WHERE clause filters the dataset to only perform calculations on Landing_Outcome Success (drone ship)



```
   Date which first Successful landing outcome in drone ship was acheived.
0                                                               06-05-2016
```

Successful Drone Ship Landing with Payload between 4000 and 6000

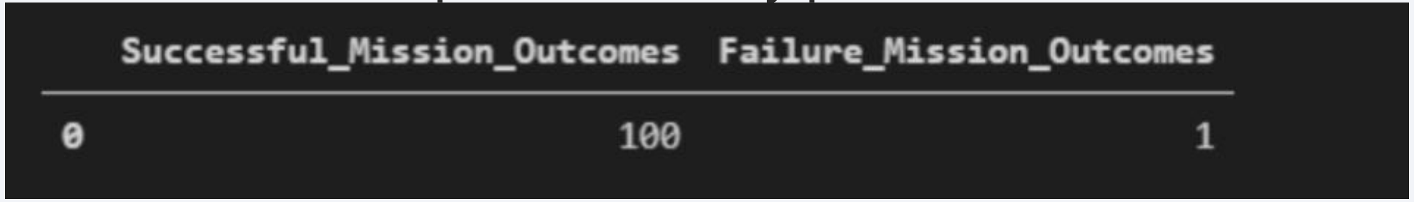
- SQL QUERY
- *Select Booster_Version from tblSpaceX where Landing_Outcome = ‘Success (ground pad)’AND Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000*
- QUERY EXPLAINATION
- Selecting only Booster_Version
- The WHERE clause filters the dataset to Landing_Outcome = Success (drone ship)
- The AND clause specifies additional filter conditions
- Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000



```
  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
```

Total Number of Successful and Failure Mission Outcomes

- SQL QUERY
- *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*
- QUERY EXPLAINATION
- a much harder query I must say, we used subqueries here to produce the results. The LIKE ‘%foo%’ wildcard shows that in the record the foo phrase is in any part of the string in the records for example.



```
   Successful_Mission_Outcomes  Failure_Mission_Outcomes
0                          100                         1
```

PHRASE “(Drone Ship was a Success)”

LIKE ‘%Success%’

Word ‘Success’ is in the phrase the filter will include it in the dataset

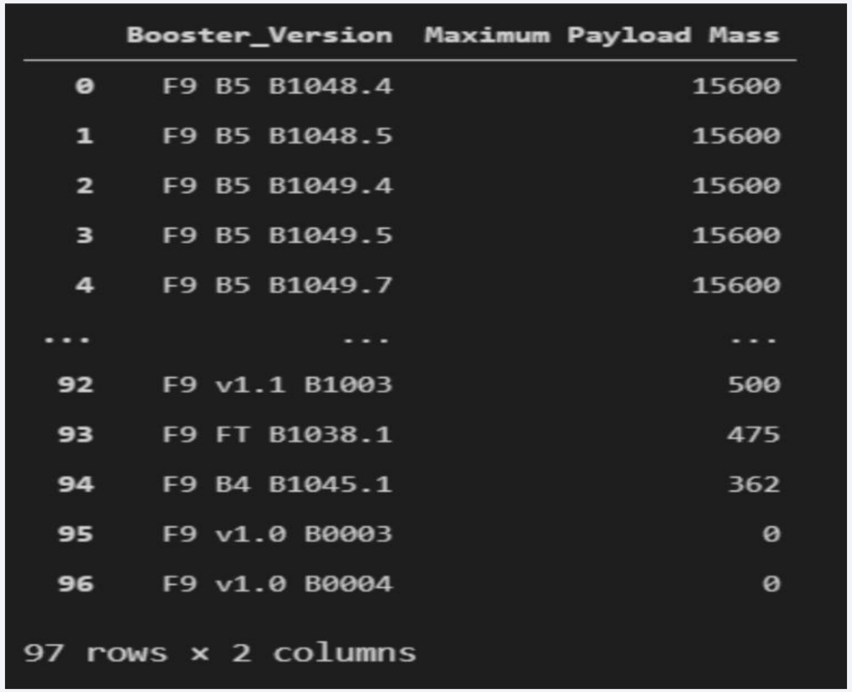
Boosters Carried Maximum Payload

- SQL QUERY
- *SELECT DISTINCT Booster_Version, MAX(PAYLOAD_MASS*

_KG_) 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 x 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

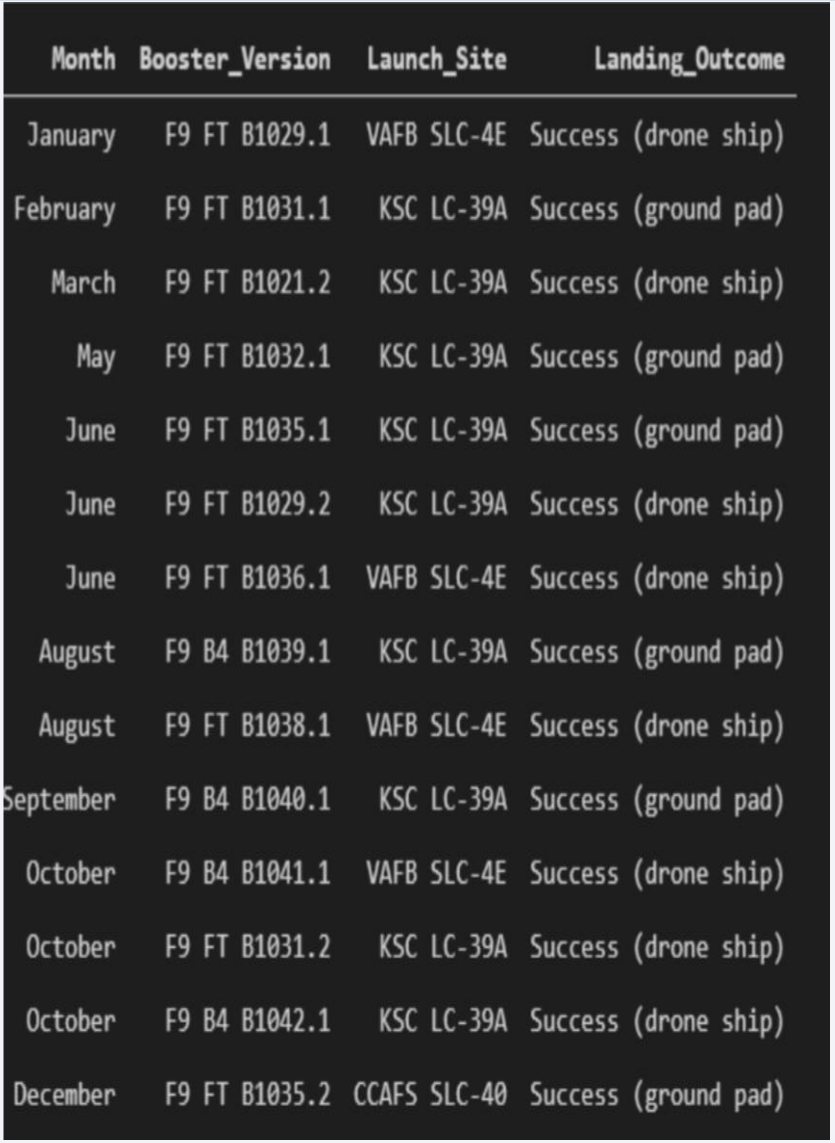
GROUP BY puts the list in order set to a certain condition.

DESC means its arranging the dataset into descending order

2015 Launch Records

- SQL QUERY
- 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)) = '2015')
QUERY EXPLAINATION
a much more complex query as I had my Date fields in
SQL Server stored as NVARCHAR the MONTH function returns name month
The function CONVERT converts NVARCHAR to Date.
WHERE clause filters Year to be 2015



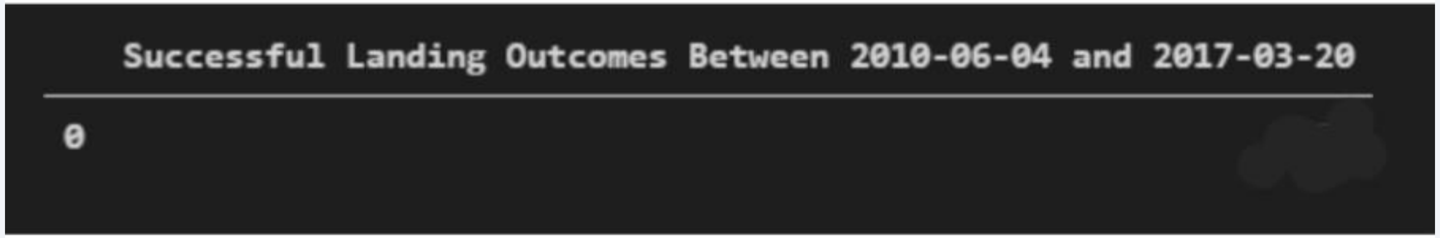
Month	Booster_Version	Launch_Site	Landing_Outcome
January	F9 FT B1029.1	VAFB SLC-4E	Success (drone ship)
February	F9 FT B1031.1	KSC LC-39A	Success (ground pad)
March	F9 FT B1021.2	KSC LC-39A	Success (drone ship)
May	F9 FT B1032.1	KSC LC-39A	Success (ground pad)
June	F9 FT B1035.1	KSC LC-39A	Success (ground 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)

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

- SQL QUERY
- 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
```

- QUERY EXPLAINATION
- Function COUNT counts records in column WHERE filters data

LIKE (wildcard)

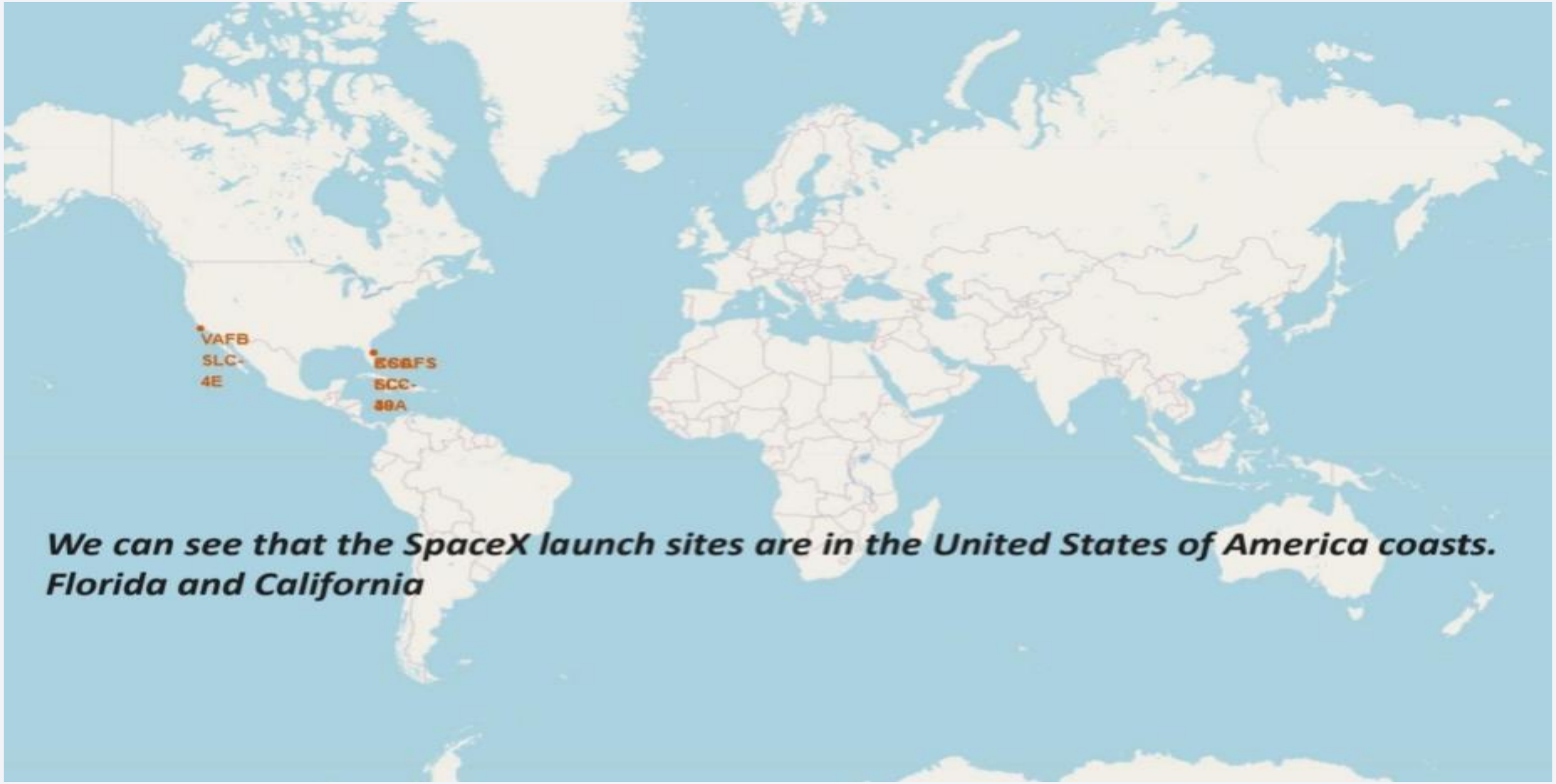
AND (conditions)

AND (conditions)

Section 3

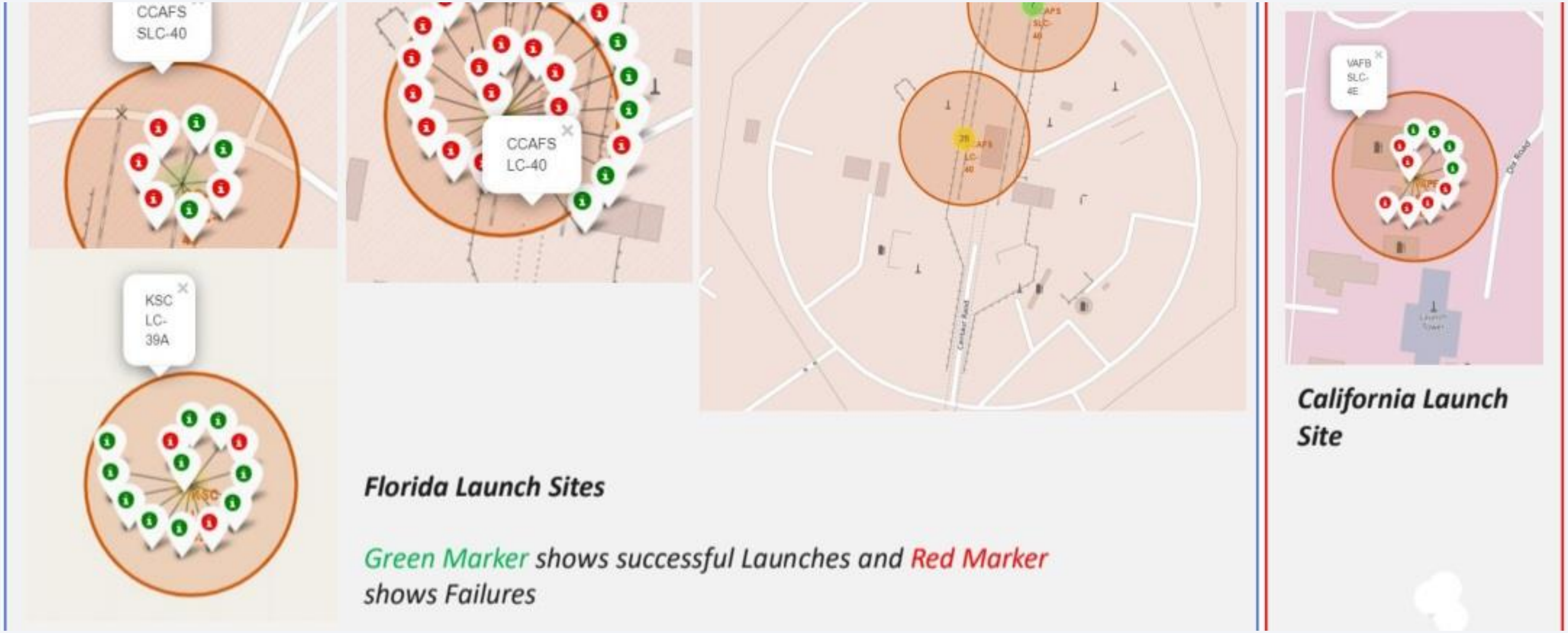
Launch Sites Proximities Analysis

ALL LAUNCH SITES GLOBAL MAP MARKERS



We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

COLOR LABELLED MARKERS

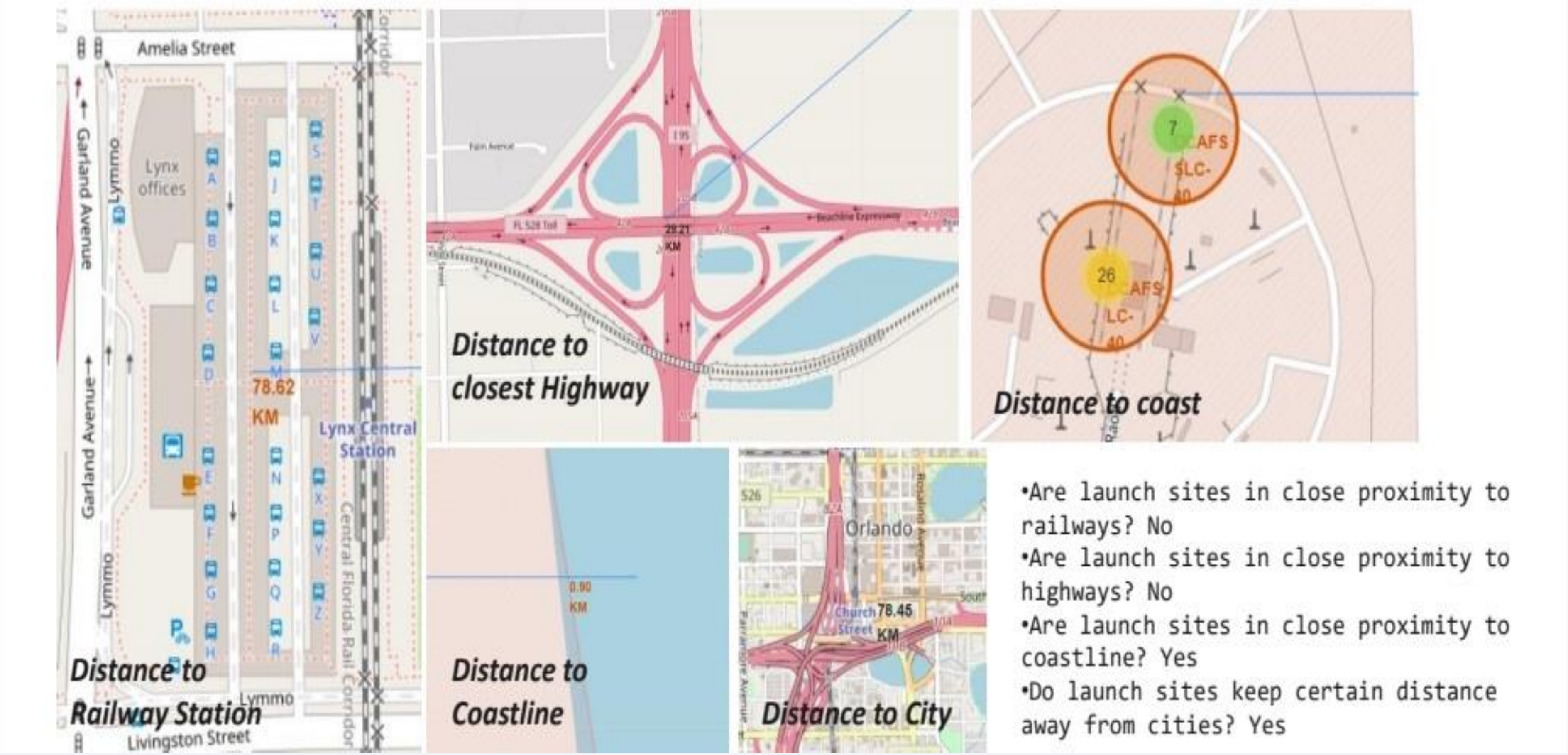


Florida Launch Sites

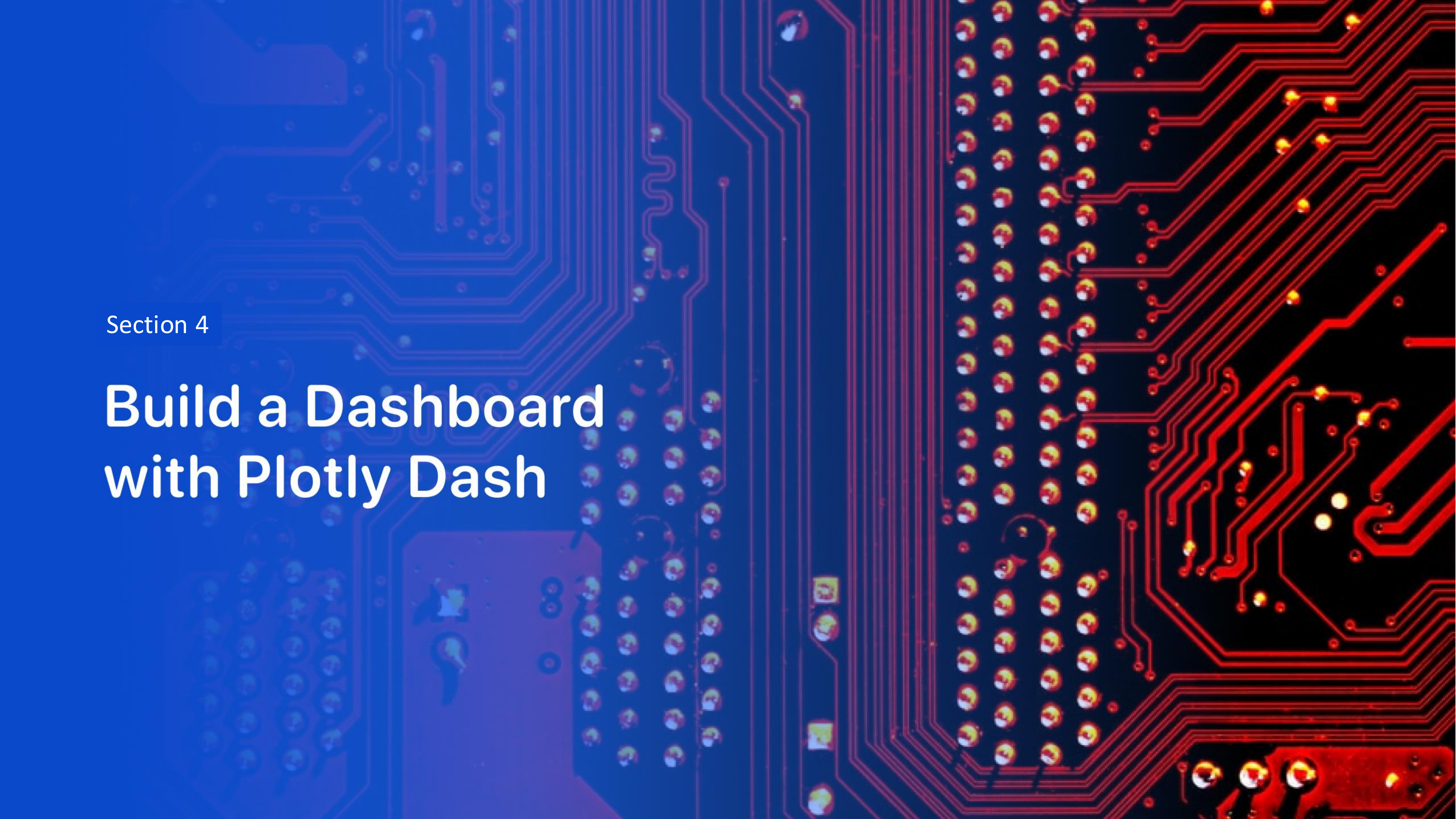
Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

Working out Launch Sites distance to landmarks to find trends with Haversine formula using CCAFS-SLC-40 as a reference



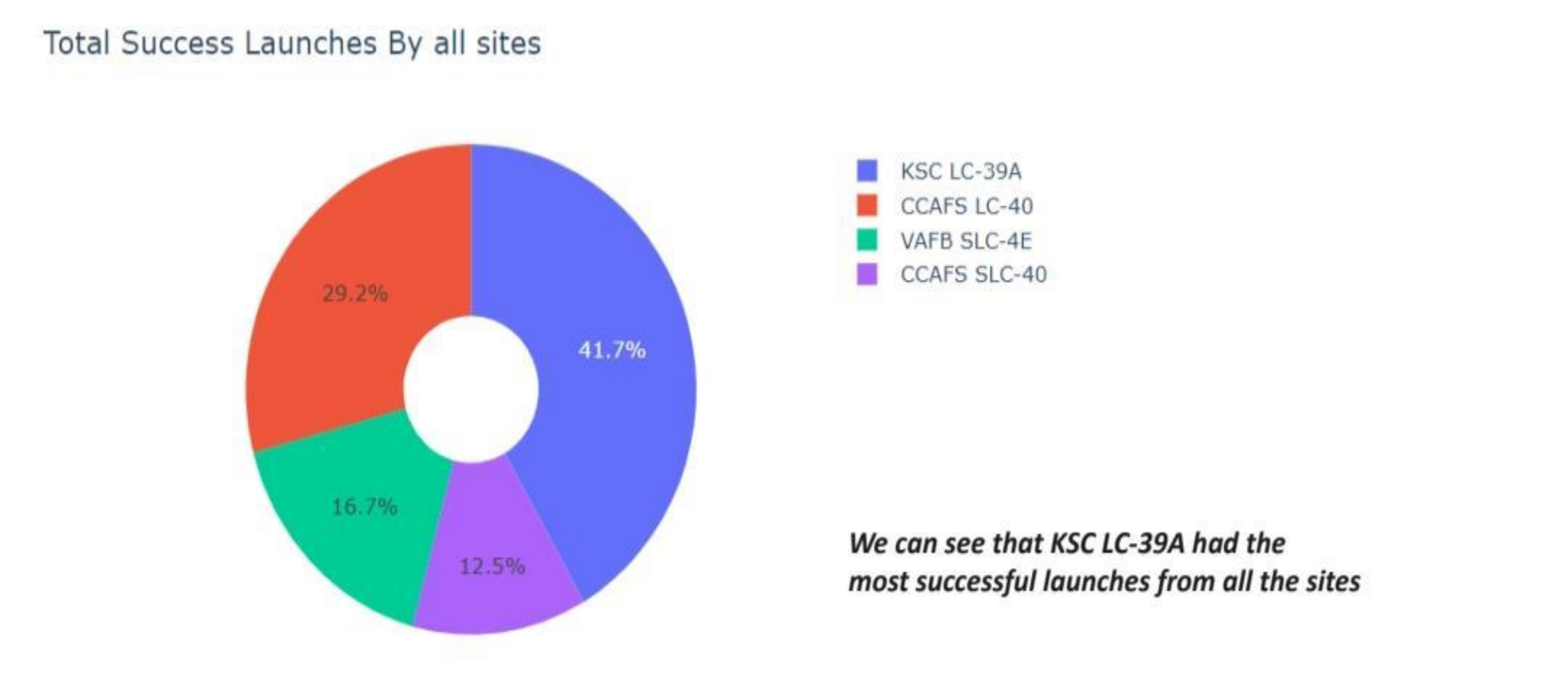
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 4

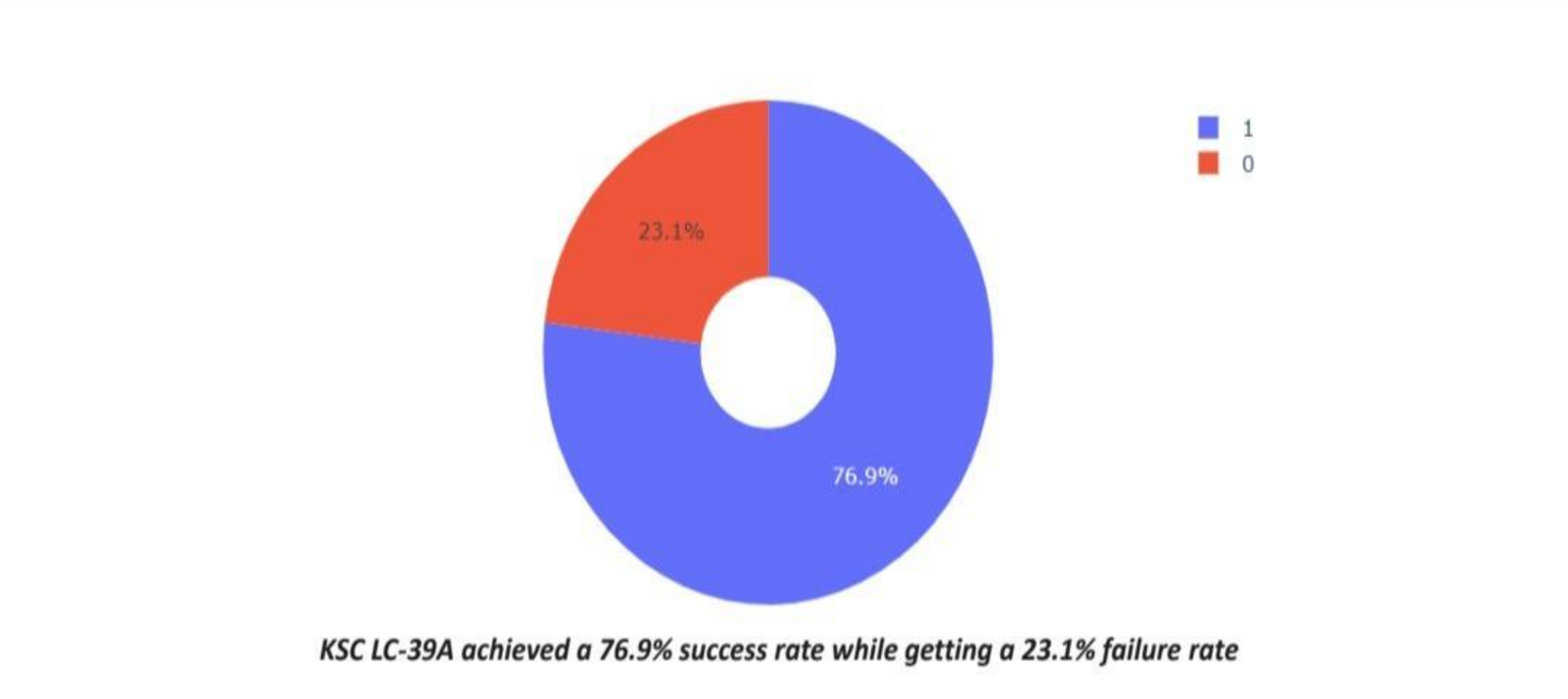
Build a Dashboard with Plotly Dash

DASHBOARD – Pie chart showing the success percentage achieved by each launch site



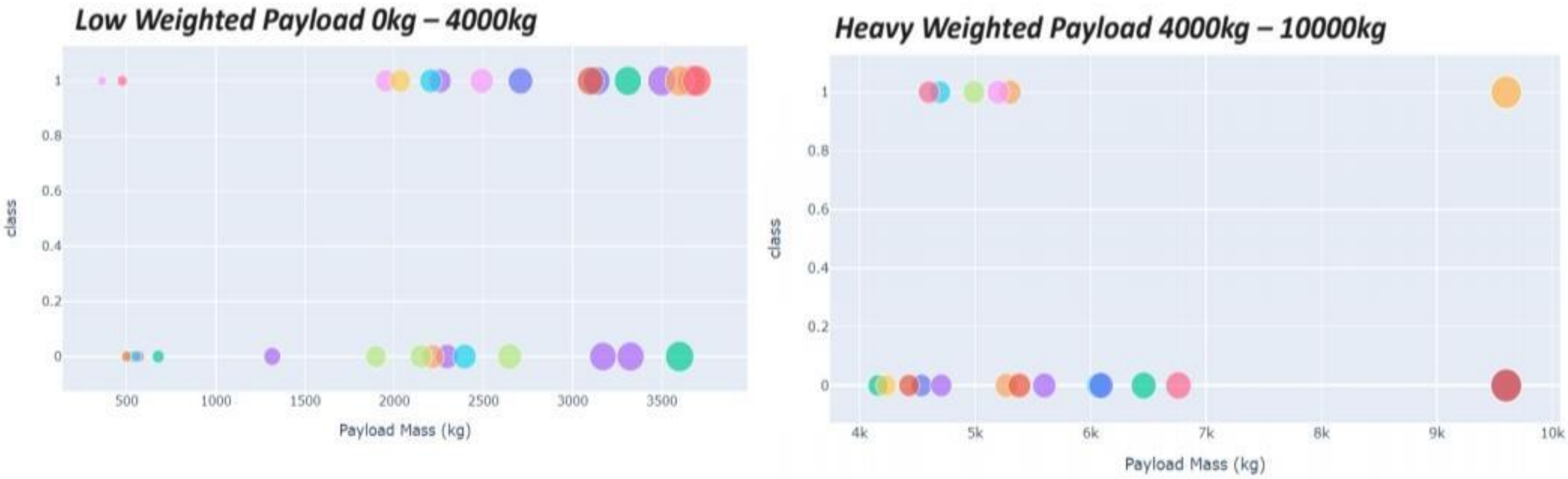
We can see that KSC LC-39A had the most successful launches from all the sites

DASHBOARD – Pie chart for the launch site with highest launch success ratio



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

DASHBOARD – Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider



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

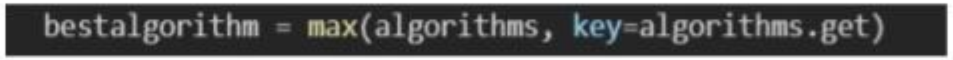
Section 5

Predictive Analysis (Classification)

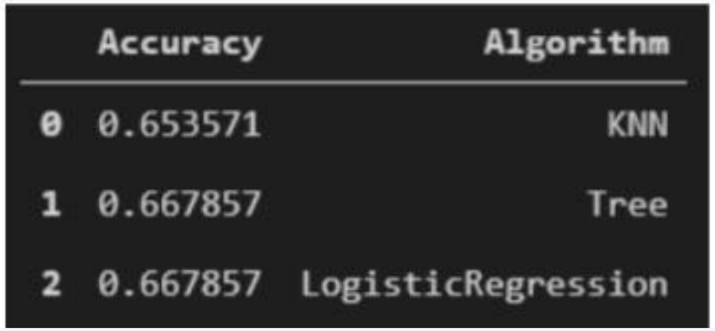
Classification Accuracy

- Classification Accuracy using training data

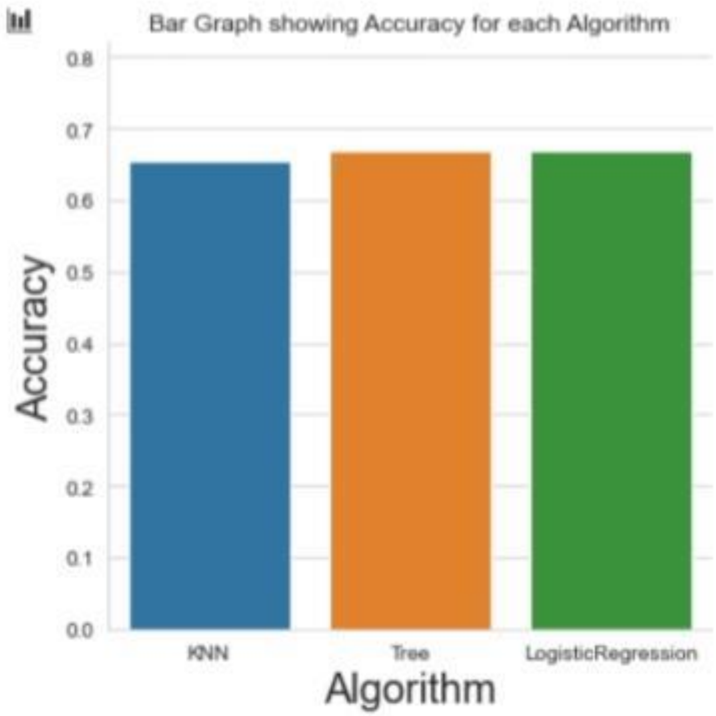
As you can see our accuracy is extremely close but we do have a winner its down to decimal places!



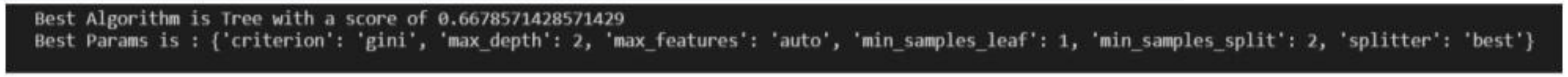
```
bestalgorithm = max(algorithms, key=algorithms.get)
```



```
   Accuracy            Algorithm
0  0.653571                  KNN
1  0.667857                 Tree
2  0.667857   LogisticRegression
```



The tree algorithm wins!!

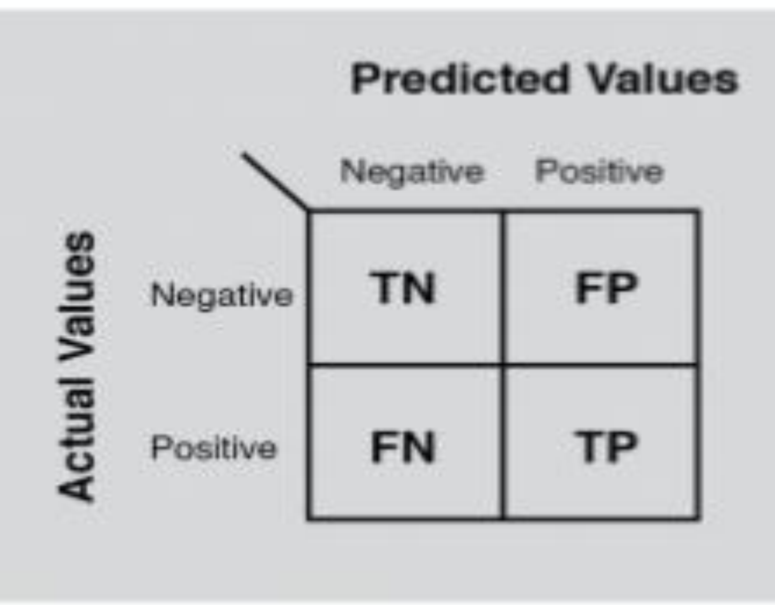


```
Best Algorithm is Tree with a score of 0.6678571428571429
Best Params is : {'criterion': 'gini', 'max_depth': 2, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
```

Confusion Matrix

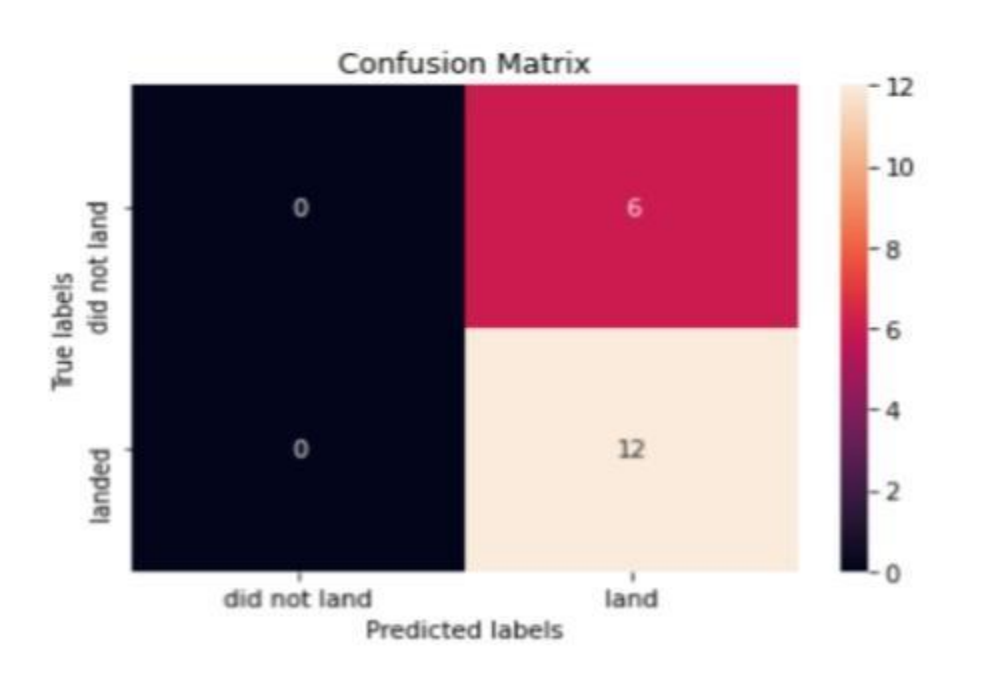
- Confusion Matrix for the Tree

Examining the confusion matrix, we see that Tree can distinguish between the different classes. We see that the major problem is false positives.



Predicted Values

Actual Values	Negative	Positive
Negative	TN	FP
Positive	FN	TP



Conclusions

1. The Tree Classifier Algorithm is the best for Machine Learning for this dataset
2. Low weighted payloads perform better than the heavier payloads
3. The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
4. We can see that KSC LC-39A had the most successful launches from all the sites
5. Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Appendix

- Haversine formula
- ADGGoogleMaps Module (not used but created)
- Module sqlserver (ADGSQLSERVER)
- PythonAnywhere 24/7 dashboard

Thank you!