IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ermelindo Varela
October 2021

Outline

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

Executive Summary

- Summary of methodologies
  - ✓ Data collection
  - ✓ Data wrangling
  - ✓ Exploratory data analysis (EDA) using visualization and SQL
  - ✓ Interactive visual analytics using Folium and Plotly Dash
  - ✓ Predicative analysis using classification models
- Summary of all results
  - ✓ Exploratory data analysis results
  - ✓ Interactive analytics demo in screenshots
  - ✓ Predictive analysis 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
  - What are the variables causing the rockets to 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

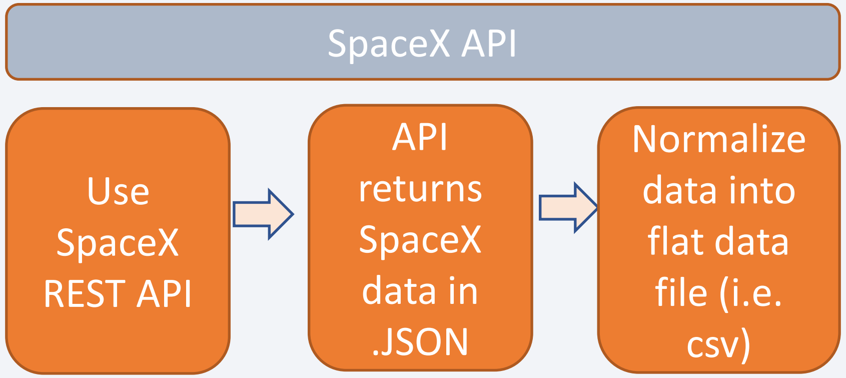
Methodology

Executive Summary

- Data collection:
    - SpaceX Rest API
    - (Web Scrapping) from Wikipedia
- Data wrangling
    - One Hot Encoding data fields for Machine Learning
- Exploratory data analysis (EDA) using visualization and SQL
    - Plotting: Scatter Graphs, Bar Graphs to show relationships between variables and patterns of data
- Interactive visual analytics using Folium and Plotly Dash
- Predictive analysis using classification models
    - How to build, tune, evaluate classification models

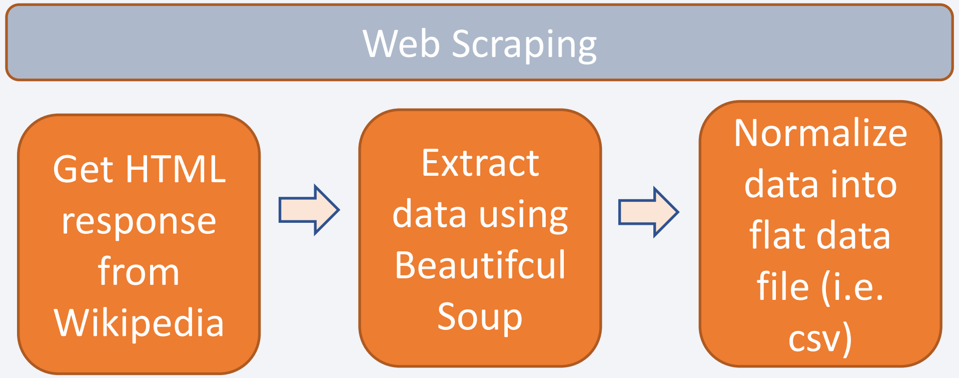
Data Collection

- Describe how data sets were collected.
  - Using SpaceX REST API, we obtained the SpaceX launch data
  - This API provides information regarding launches, namely: rocket used, payload delivered, landing outcome and specifications
  - The goal is using 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 source used was webscraping Wikipedia using BeautifulSoup to obtain data regarding Falcon 9



SpaceX API

Use SpaceX REST API → API returns SpaceX data in .JSON → Normalize data into flat data file (i.e. csv)

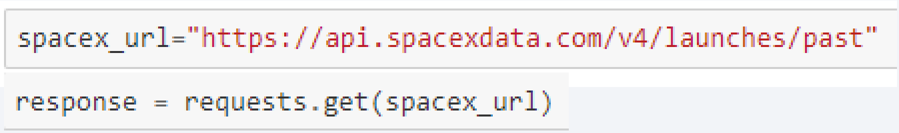


Web Scraping

Get HTML response from Wikipedia → Extract data using Beautifcul Soup → Normalize data into flat data file (i.e. csv)

Data Collection – SpaceX API

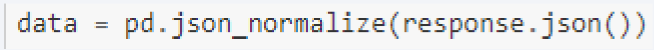
1. Getting response from API



```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

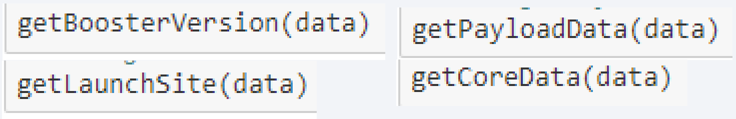
```
response = requests.get(spacex_url)
```

2. Converting response to JSON



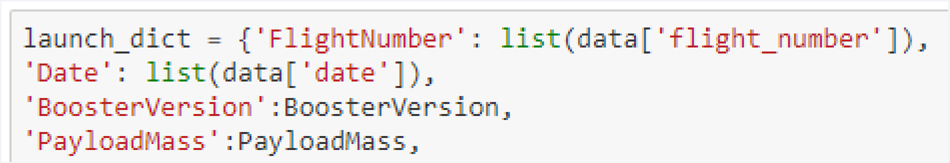
```
data = pd.json_normalize(response.json())
```

3. Cleaning data



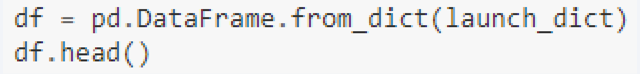
```
getBoosterVersion(data)
getPayloadData(data)
getLaunchSite(data)
getCoreData(data)
```

4. Assigning list to dictionary then dataframe

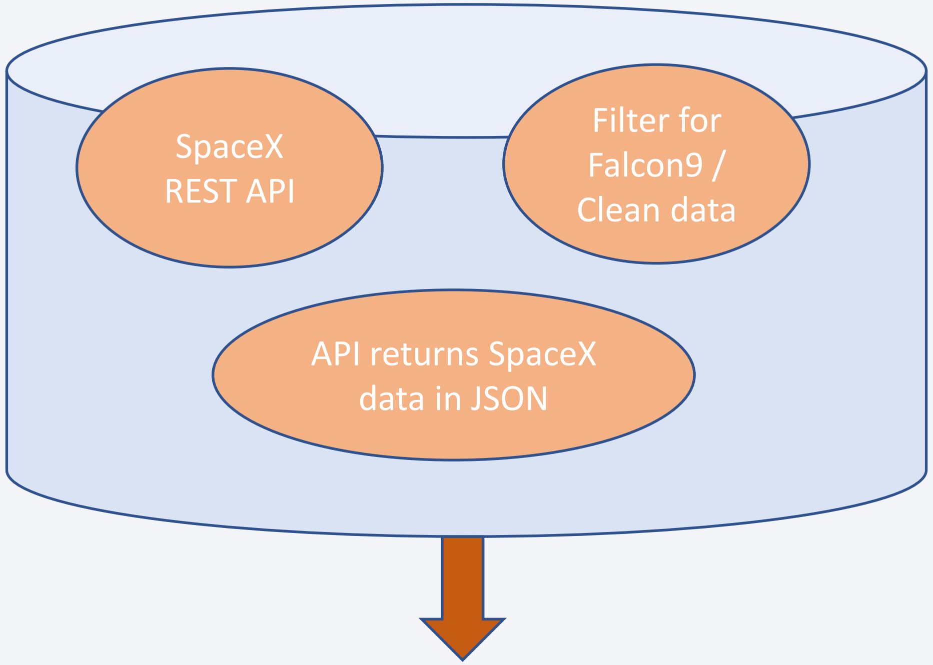


```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
```

5. Filter dataframe and export to a csv file



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

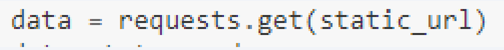


Normalize data in csv format

GitHub URL

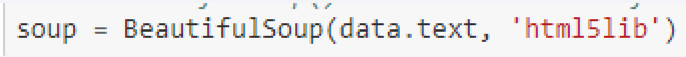
Data Collection – Webscraping

1. Getting response from HTML



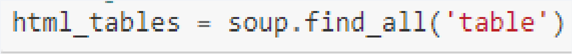
```
data = requests.get(static_url)
```

2. Creating BeautifulSoup Object



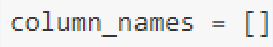
```
soup = BeautifulSoup(data.text, 'html5lib')
```

3. Finding tables

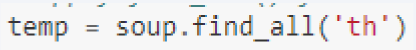


```
html_tables = soup.find_all('table')
```

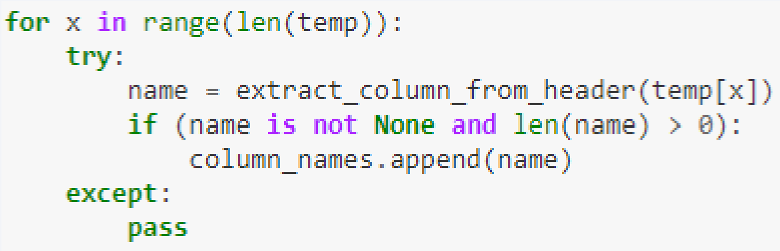
4. Getting column names



```
column_names = []
```

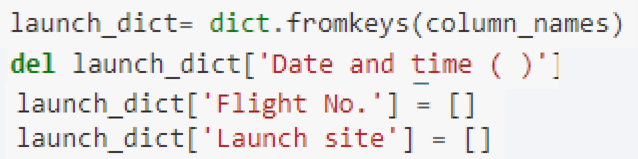


```
temp = soup.find_all('th')
```



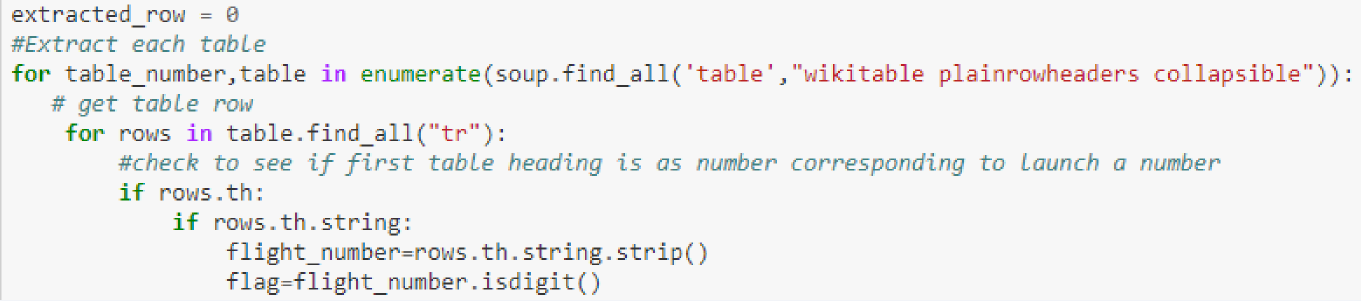
```
for x in range(len(temp)):
    try:
        name = extract_column_from_header(temp[x])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

5. Creating dictionaries



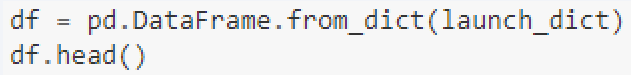
```
launch_dict= dict.fromkeys(column_names)
del launch_dict['Date and time ( )']
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
```

6. Appending data to keys

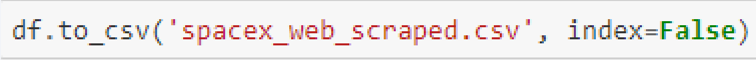


```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
```

7. Converting dictionary to dataframe and to csv

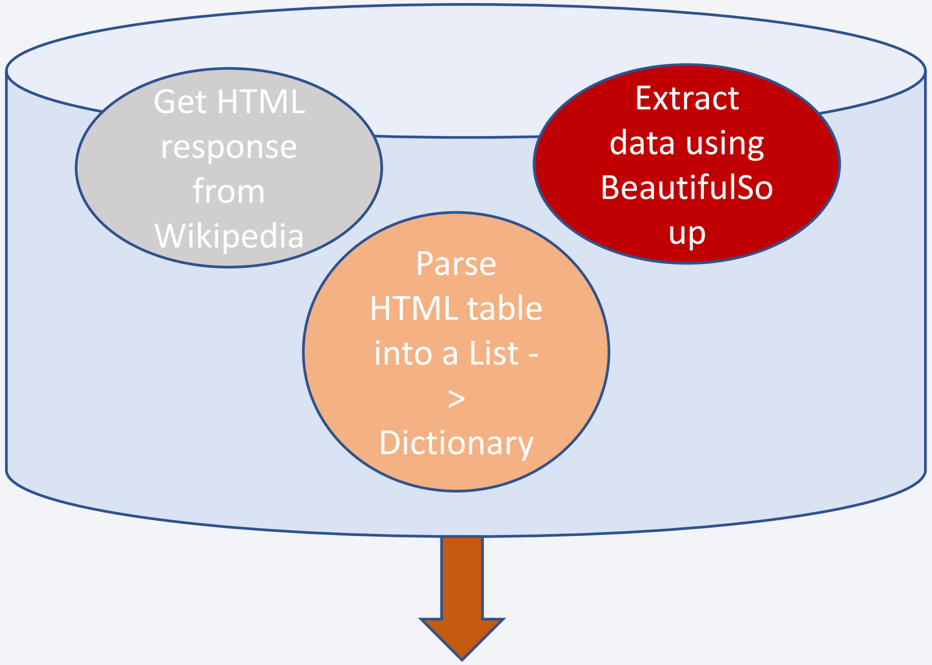


```
df = pd.DataFrame.from_dict(launch_dict)
df.head()
```



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

Data Collection – Webscraping



Get HTML response from Wikipedia

Parse HTML table into a List -> Dictionary

Extract data using BeautifulSoup

Normalize data in csv format

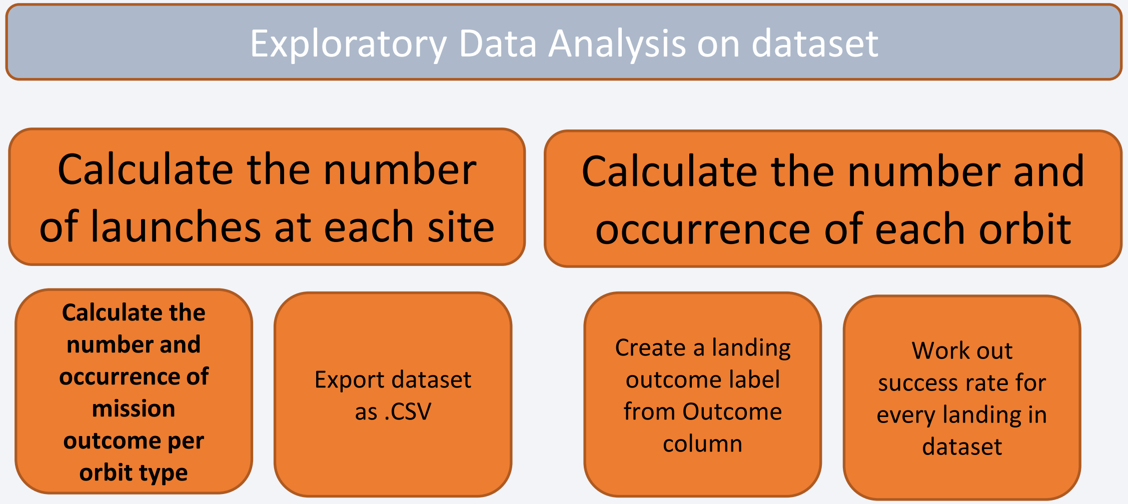
GitHub URL to Notebook

Data Wrangling

Introduction

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.
In this lab we will mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

Process



Exploratory Data Analysis 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

GitHub URL to Notebook

EDA with Data Visualization

Scatter Graphs

- 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

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.

Bar Graph:

- Mean VS. Orbit

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.

Line Graph:

- 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

GitHub URL to Notebook

EDA with SQL

Using SQL queries we summarized the following information 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.

GitHub URL to Notebook

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 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

GitHub URL to Notebook

Build a Dashboard with Plotly Dash

The dashboard is built with Flask and Dash web framework.

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.

GitHub URL to Notebook

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

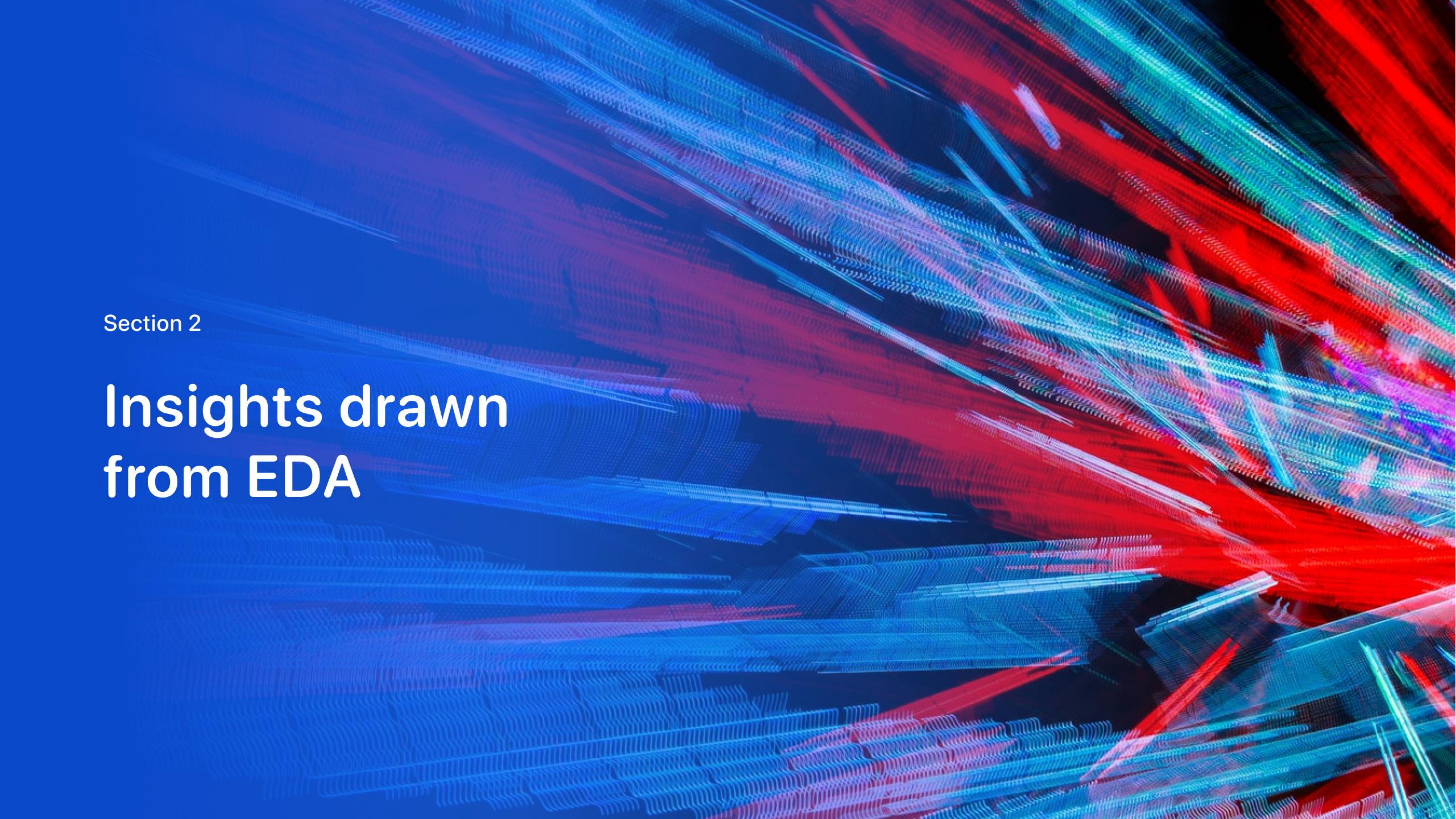
FINDING THE BEST PERFORMING CLASSIFICATION MODEL

- 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

GitHub URL to Notebook

Results

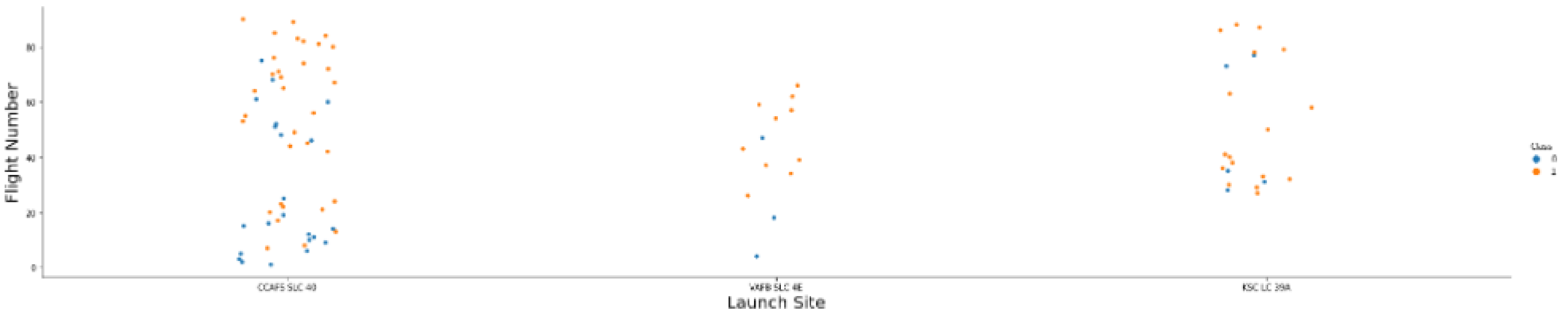
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



Section 2

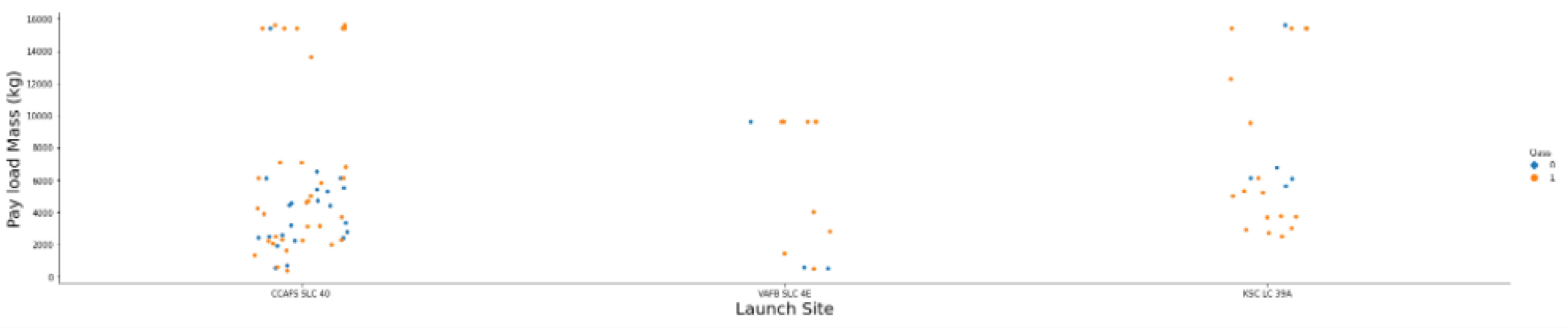
Insights drawn from EDA

Flight Number vs. Launch Site



- The success rate increases with the number of flights, which makes sense because there is a learning curve.

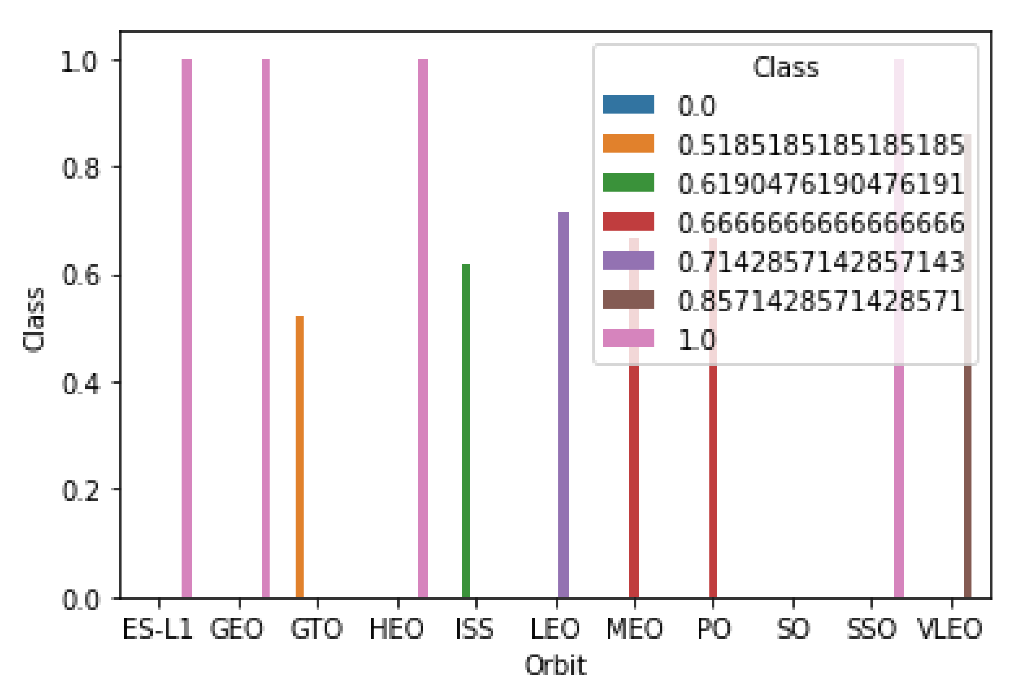
Payload vs. Launch Site



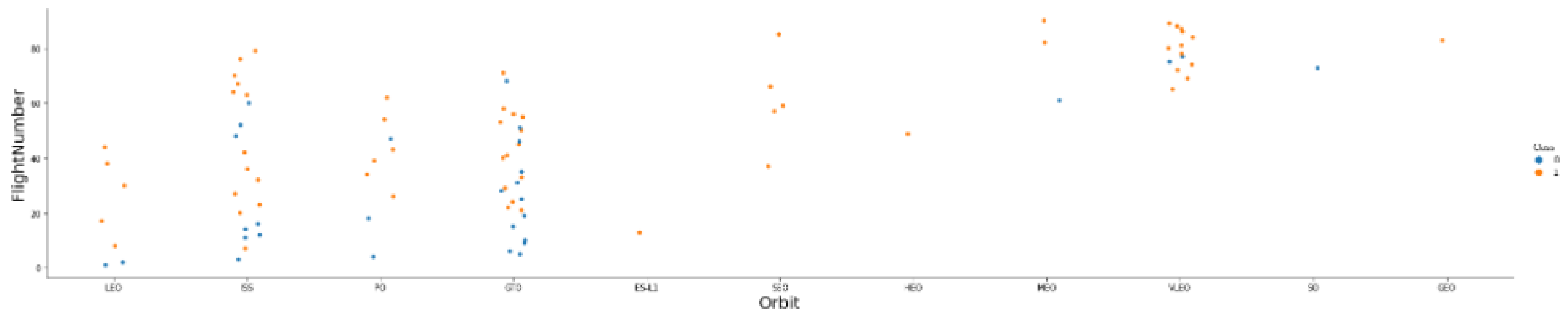
- There is no clear pattern from the graph above in order to determine whether a success launch increases or decreases depending on a certain combination of Pay Load Mass and Launch Site.

Success Rate vs. Orbit Type

- The highest success rate are in the following orbits: ES-L1, GEO, HEO, SSO

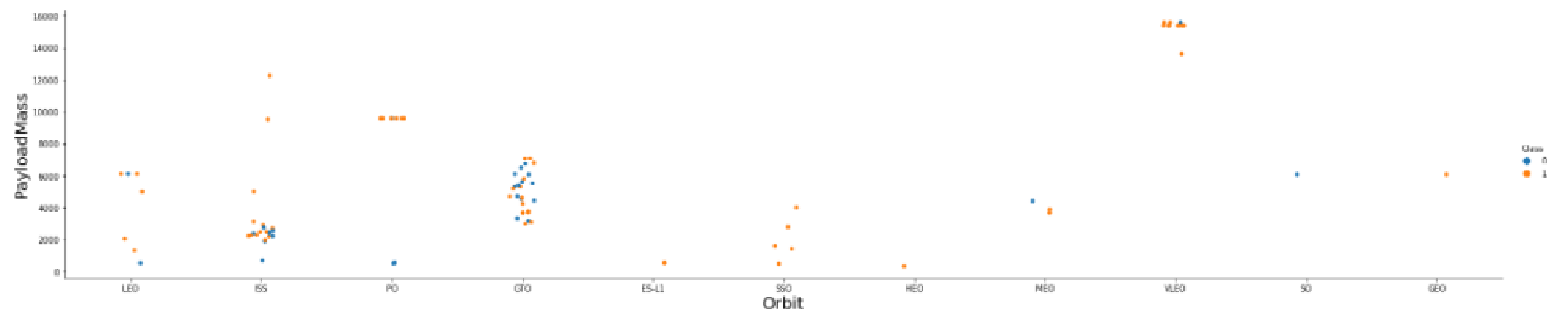


Flight Number vs. Orbit Type



- 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.

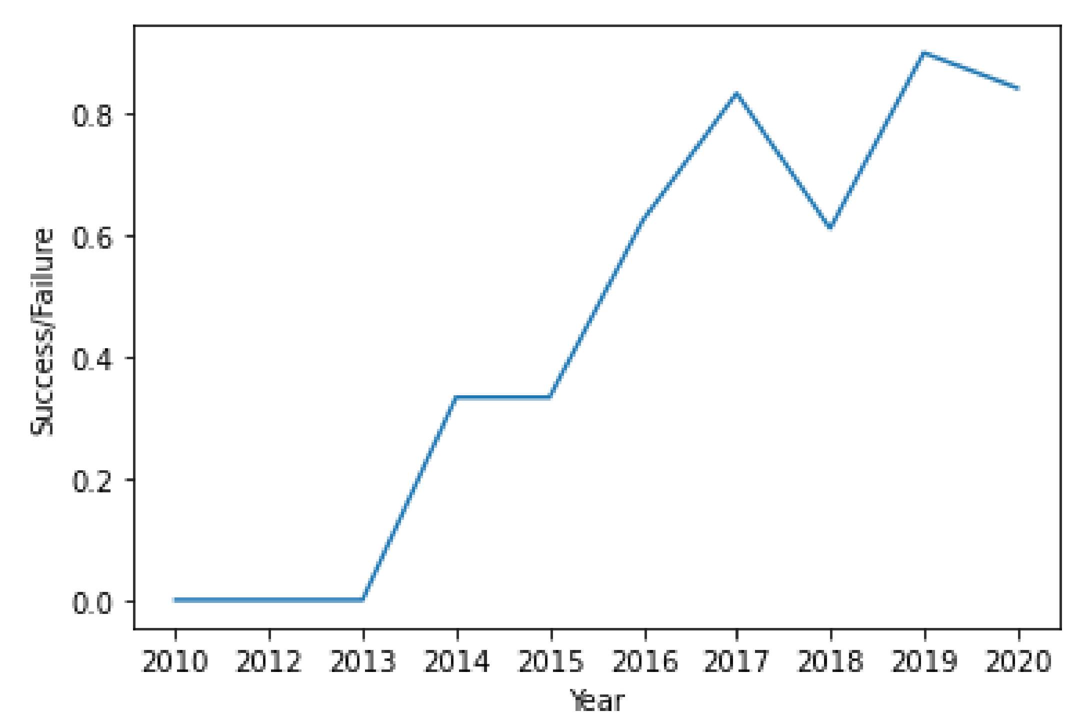
Payload vs. Orbit Type



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

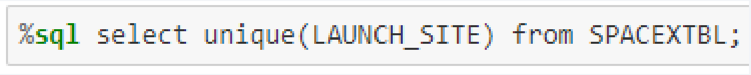
Launch Success Yearly Trend

- The success rate has increased consistently from 2013 till 2020

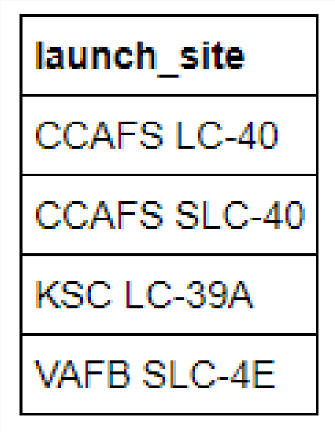


Unique Launch Site Names

- Find the names of the unique launch sites



```
%sql select unique(LAUNCH_SITE) from SPACEXTBL;
```



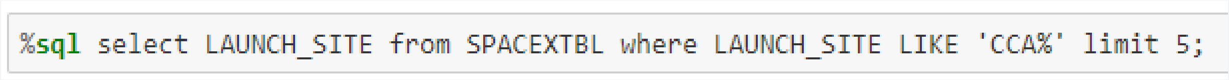
launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

- Query Explanation

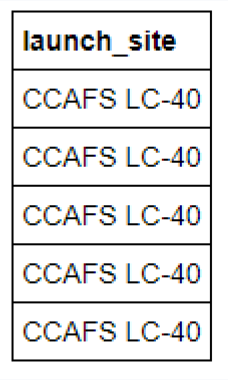
Using the word ***Unique*** in the query means that it will only show Unique values in the ***Launch_Site*** column from ***tblSpaceX***

Launch Site Names Begin with 'CCA'

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



```
%sql select LAUNCH_SITE from SPACEXTBL where LAUNCH_SITE LIKE 'CCA%' limit 5;
```



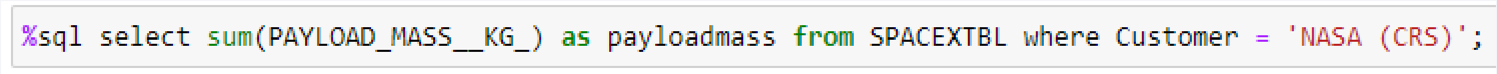
launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

- Query Explanation

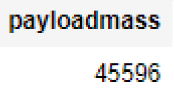
Using the word ***TOP 5*** in the query means that it will only show 5 records from ***tblSpaceX*** and ***LIKE*** keyword has a wild card with the words ***‘CCA%’*** the percentage in the end suggests that the Launch_Site name must start with CCA.

Total Payload Mass

- Calculate the total payload carried by boosters from NASA



```
%sql select sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL where Customer = 'NASA (CRS)';
```



payloadmass
45596

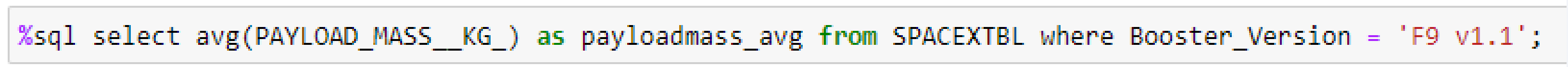
- Query Explanation

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

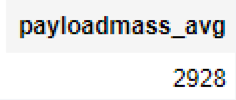
The ***WHERE*** clause filters the dataset to only perform calculations on ***Customer NASA (CRS)***

Average Payload Mass by F9 v1.1

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



```
%sql select avg(PAYLOAD_MASS__KG_) as payloadmass_avg from SPACEXTBL where Booster_Version = 'F9 v1.1';
```



payloadmass_avg
2928

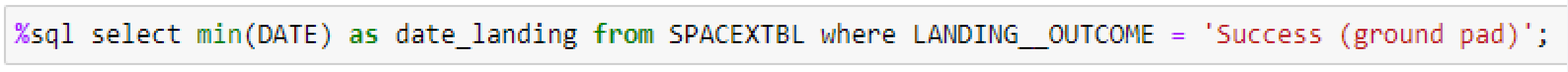
- Query Explanation

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

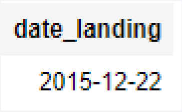
The ***WHERE***clause filters the dataset to only perform calculations on ***Booster_version F9 v1.1***

First Successful Ground Landing Date

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



```
%sql select min(DATE) as date_landing from SPACEXTBL where LANDING__OUTCOME = 'Success (ground pad)';
```



date_landing
2015-12-22

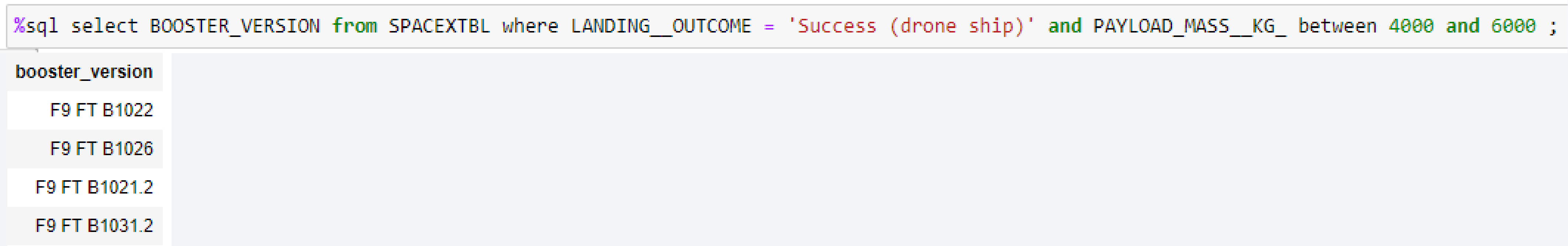
- Present your query result with a short explanation here

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 (ground pad)***

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the 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_ between 4000 and 6000 ;
```

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

- Query Explanation

Selecting only ***Booster_Version***

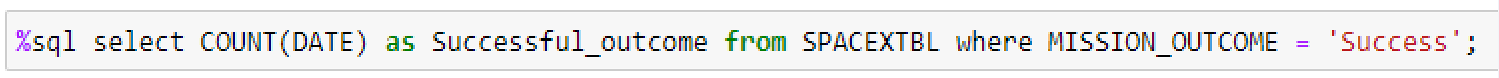
The ***WHERE***clause filters the dataset to ***Landing_Outcome = Success (drone ship)***

The ***BETWEEN*** clause specifies additional filter conditions

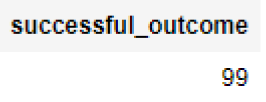
Payload_MASS_KG_>4000 and ***Payload_MASS_KG_<6000***

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful mission outcomes

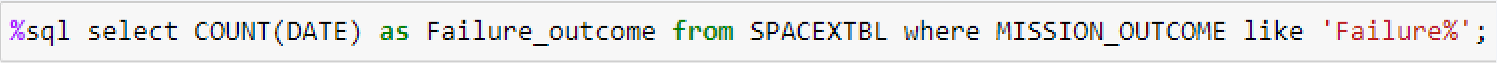


```
%sql select COUNT(DATE) as Successful_outcome from SPACEXTBL where MISSION_OUTCOME = 'Success';
```

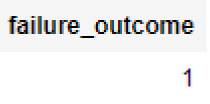


successful_outcome
99

- Calculate the total number of failure mission outcomes



```
%sql select COUNT(DATE) as Failure_outcome from SPACEXTBL where MISSION_OUTCOME like 'Failure%';
```



failure_outcome
1

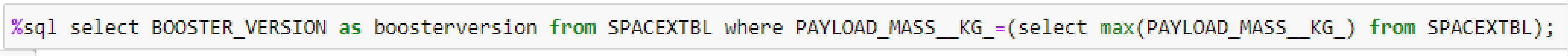
- Query Explanation

Broken down in two queries

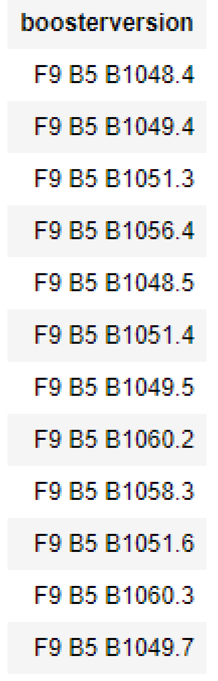
The ***WHERE*** clause filters the dataset for MISSION_OUTCOME ***= Success*** *and* MISSION_OUTCOME ***= Failure***

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass



```
%sql select BOOSTER_VERSION as boosterversion from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
```



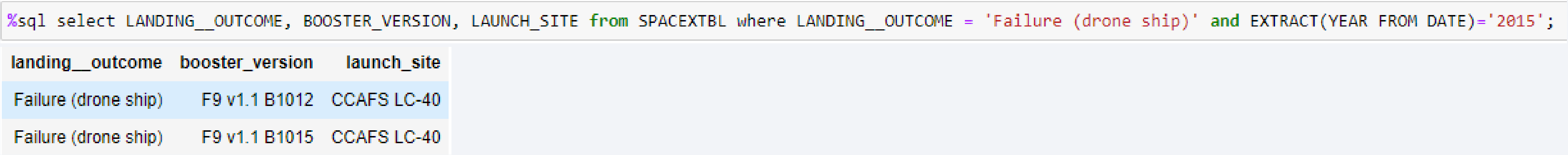
boosterversion
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

- Query Explanation

Using a subquery together with **MAX** function to select the boosters with Maximum load

2015 Launch Records

- List the 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 SPACEXTBL where LANDING__OUTCOME = 'Failure (drone ship)' and EXTRACT(YEAR FROM DATE)='2015';
```

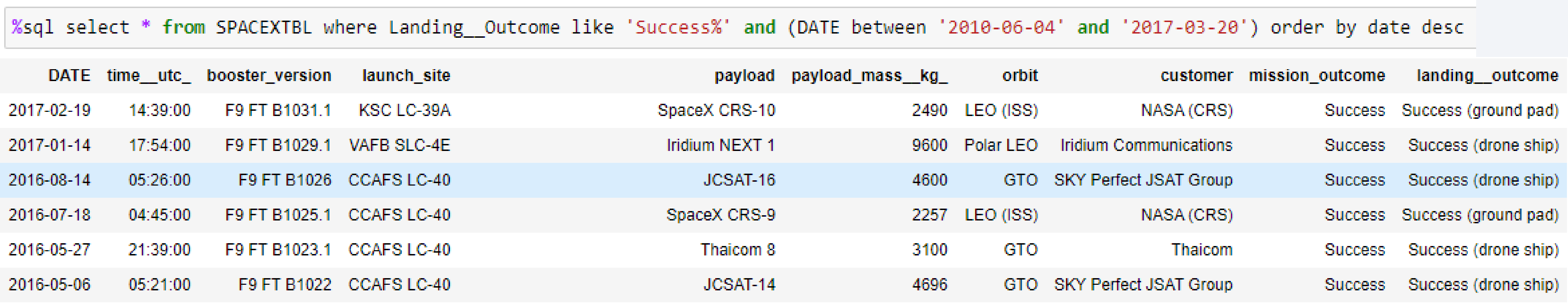
landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- Query Explanation

Using WHERE clause to filter ***Failure (drone ship)*** and **EXTRACT** to obtain 2015

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

- 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



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

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)

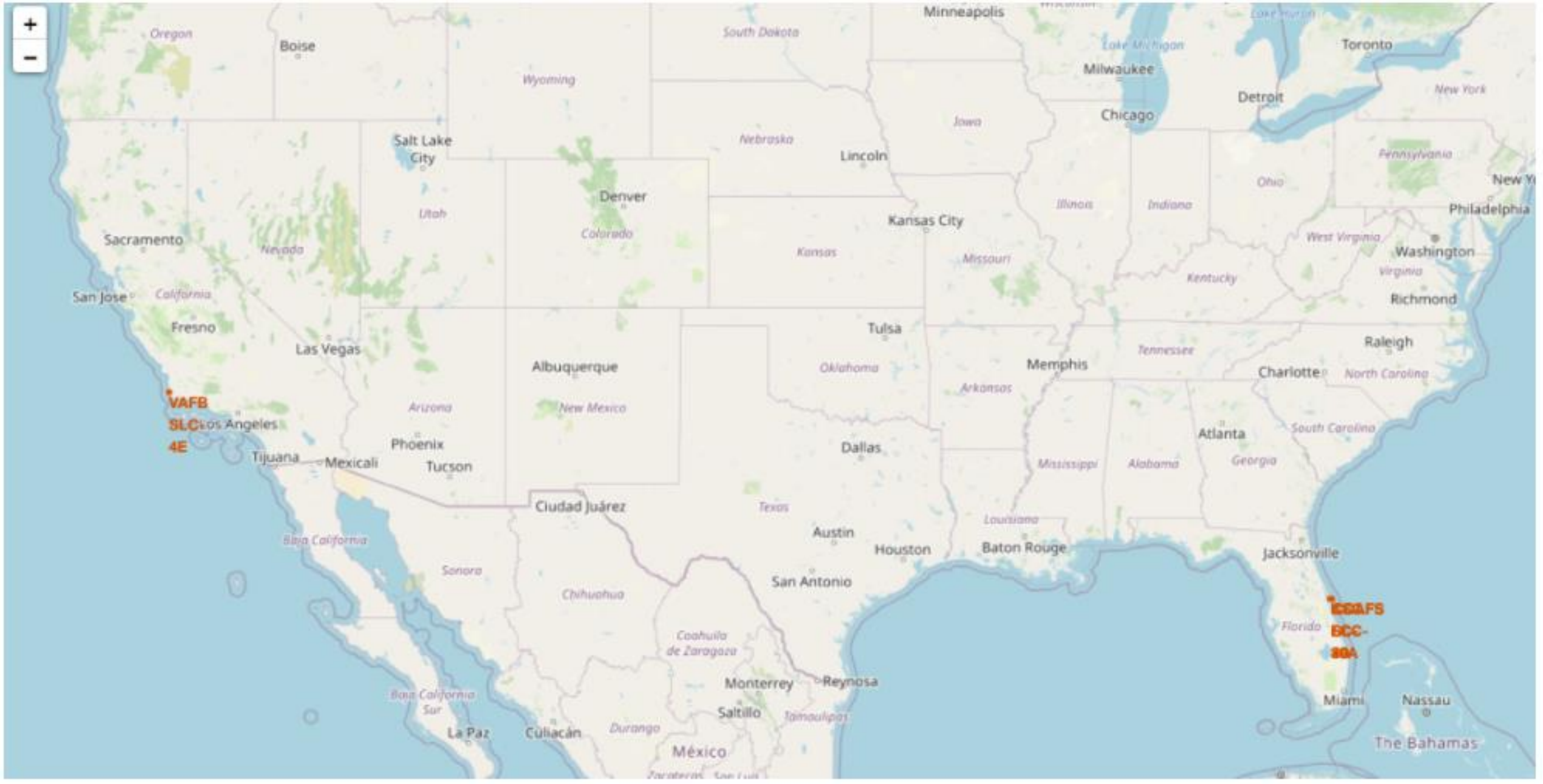
- Query Explanation

Using WHERE clause to filter the relevant dates

Section 4

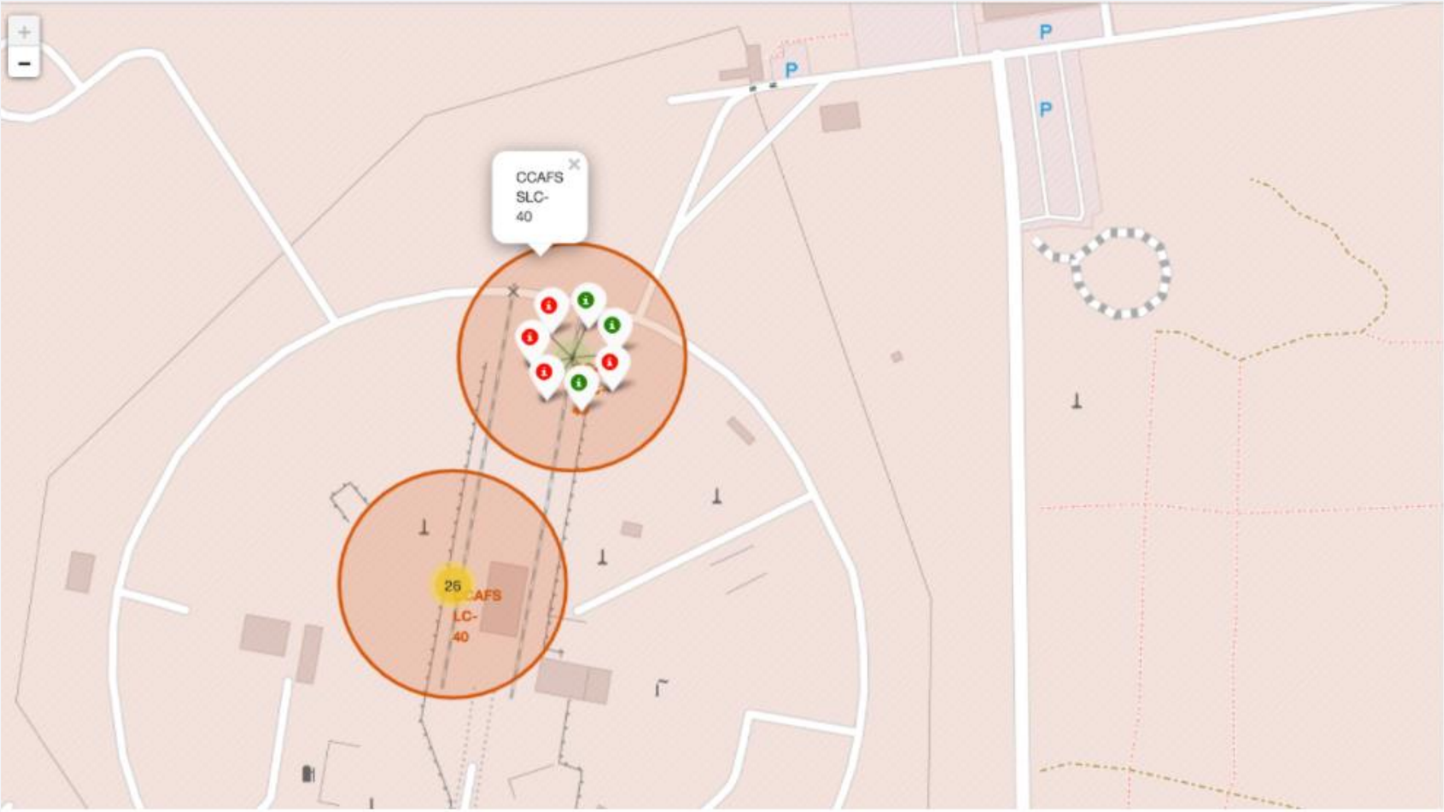
Launch Sites Proximities Analysis

All Launch Sites



SpaceX launch sites are in coastal areas (Florida and California)

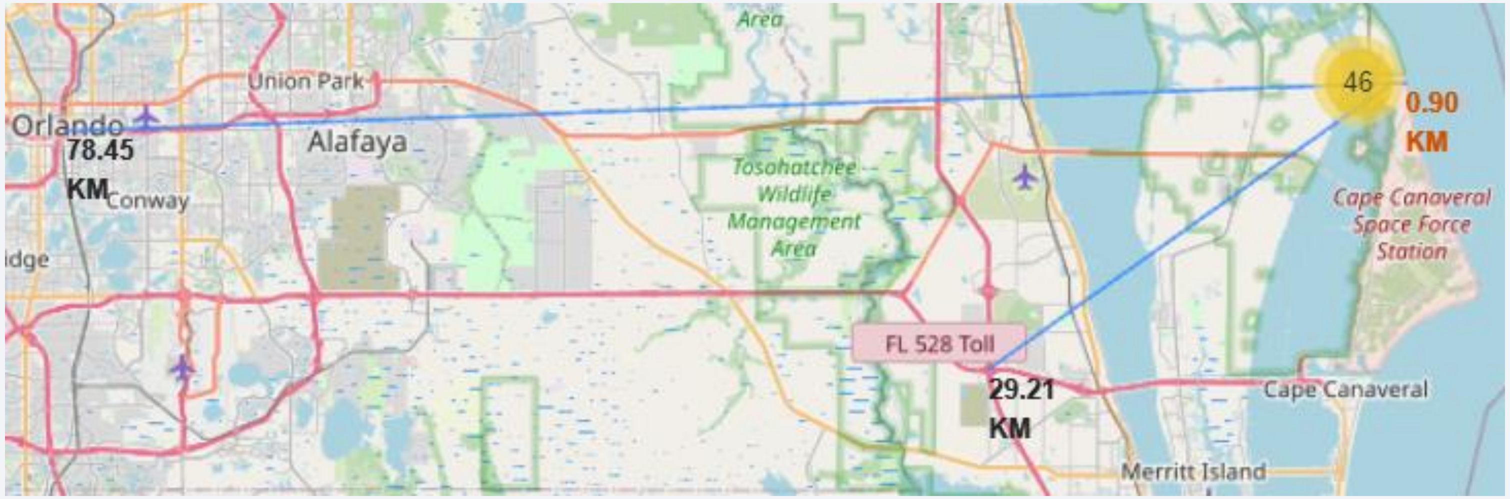
Color-labeled Markers



Green marker denotes successful launches and Red Marker failures.

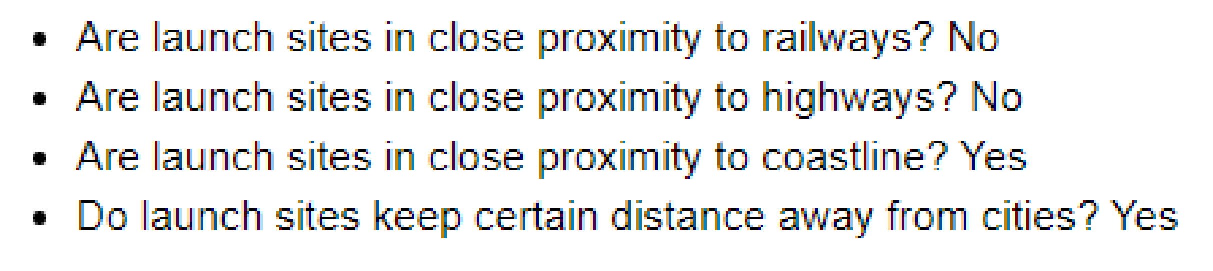
Calculating distance to certain landmarks using CCAFS-SLC-40 as reference

Distance to city

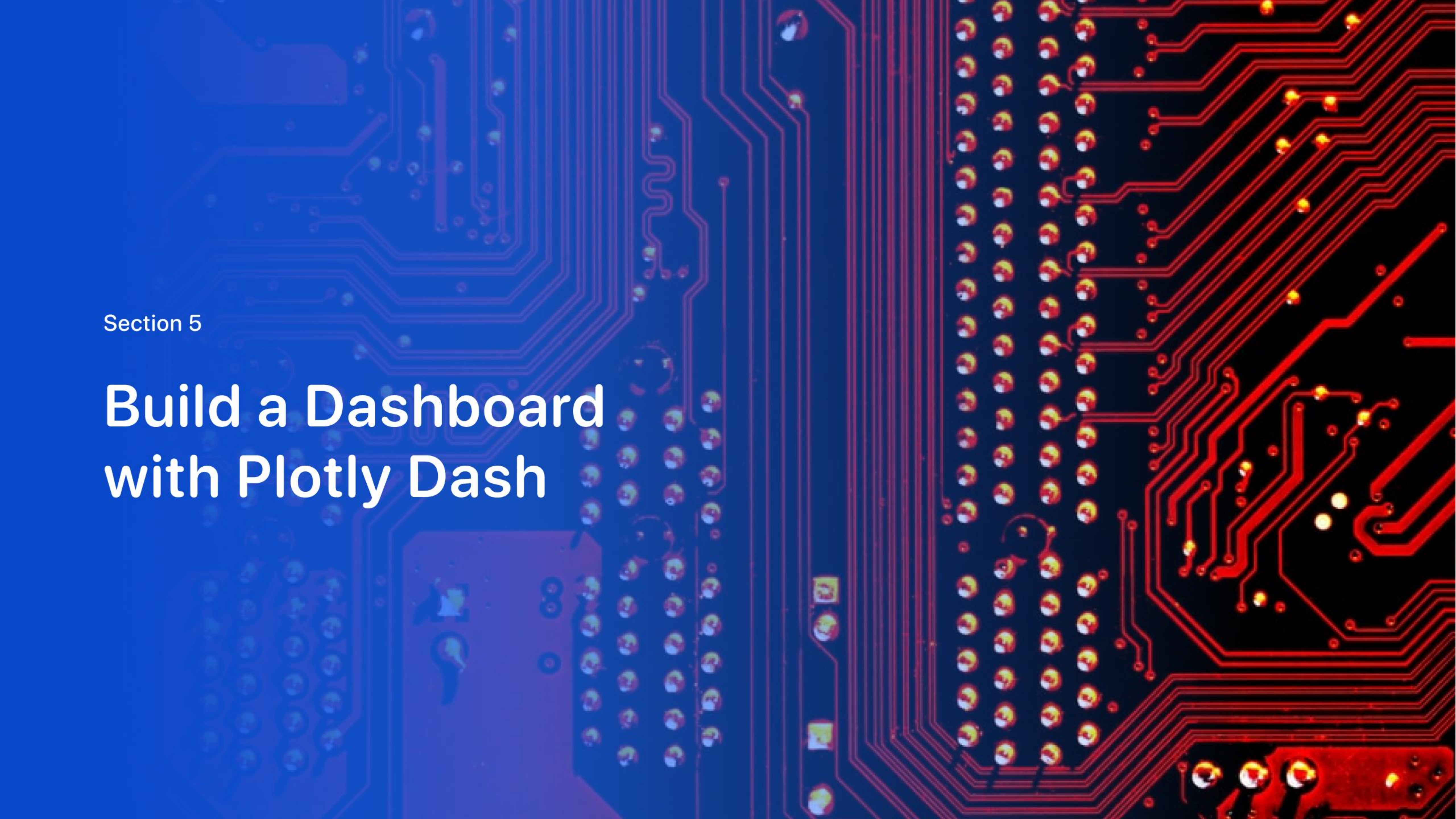


Distance to coastline

Distance to highway



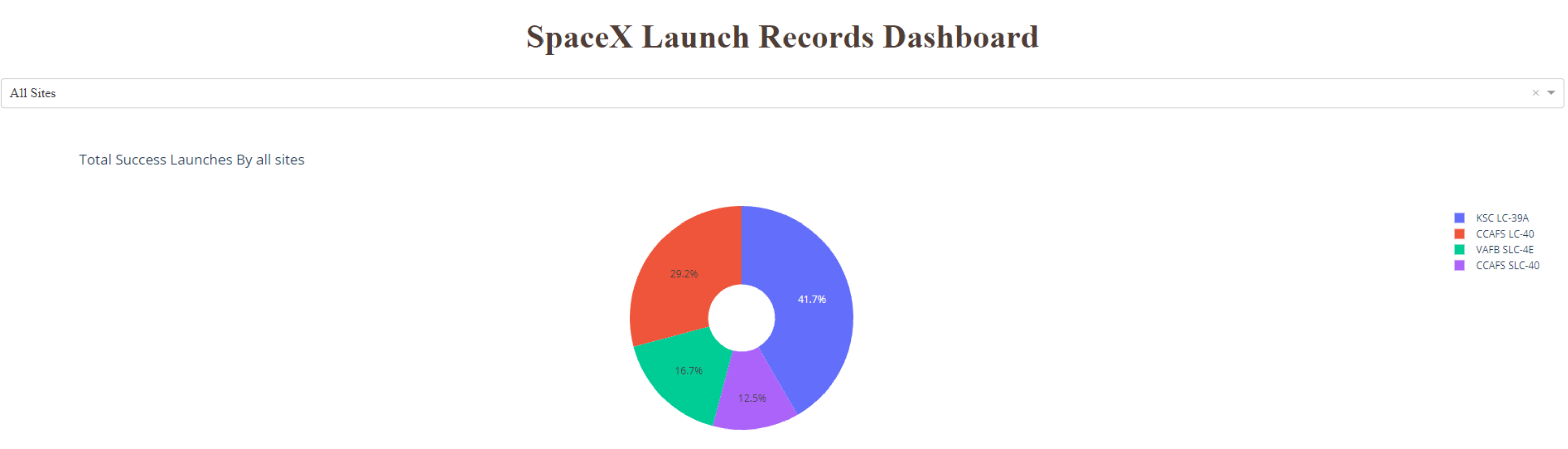
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 5

Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



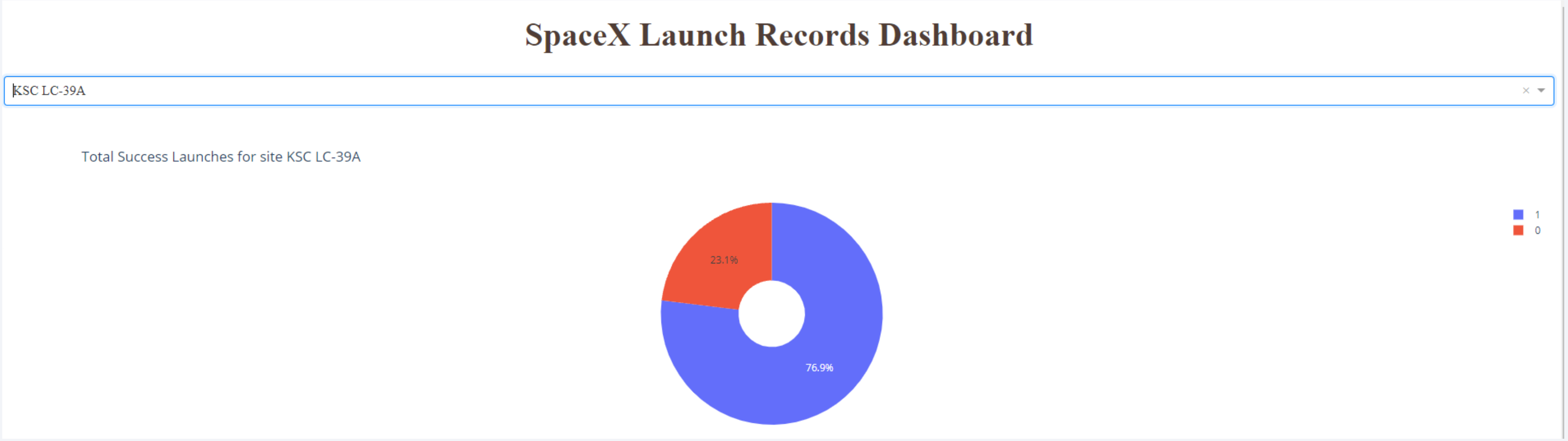
SpaceX Launch Records Dashboard

All Sites

Total Success Launches By all sites

KSC LC-39A is the most successful of all the sites

Pie chart for the launch site with highest launch success ratio



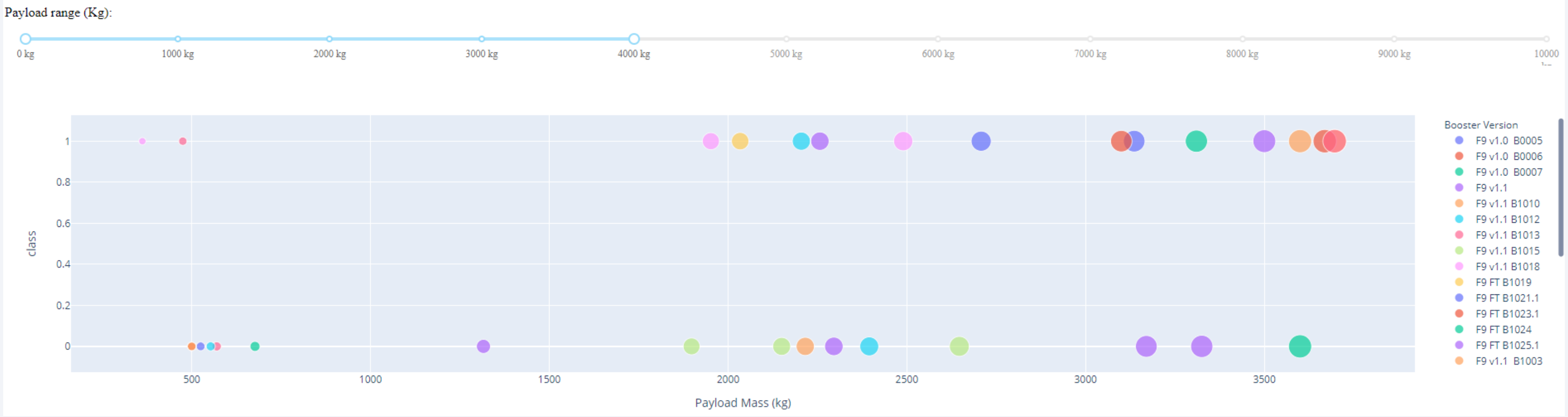
SpaceX Launch Records Dashboard

KSC LC-39A

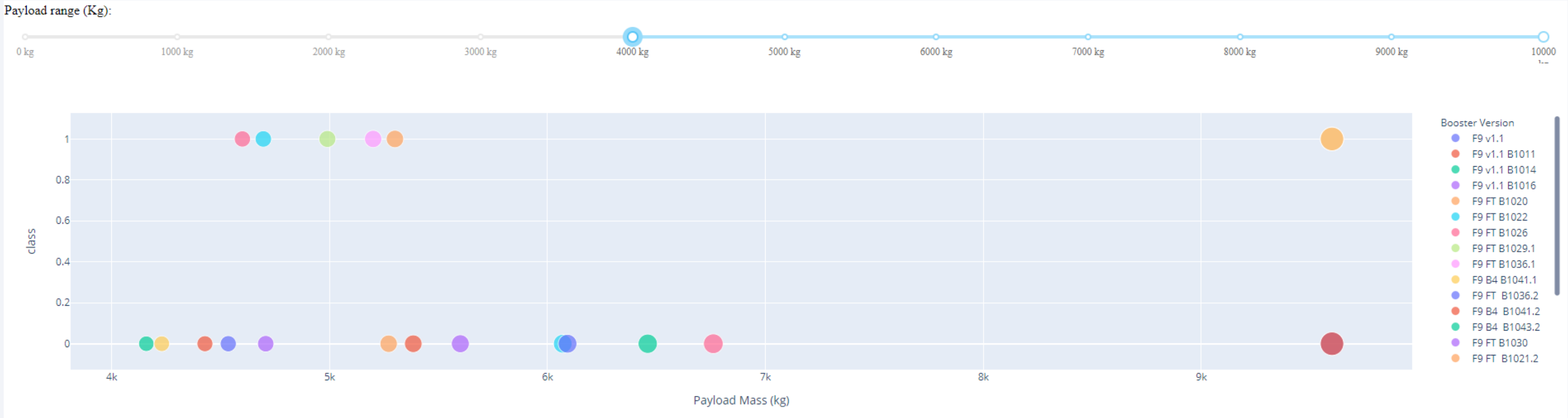
Total Success Launches for site KSC LC-39A

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

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

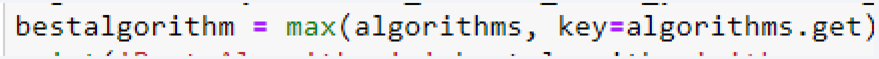
We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 6

Predictive Analysis (Classification)

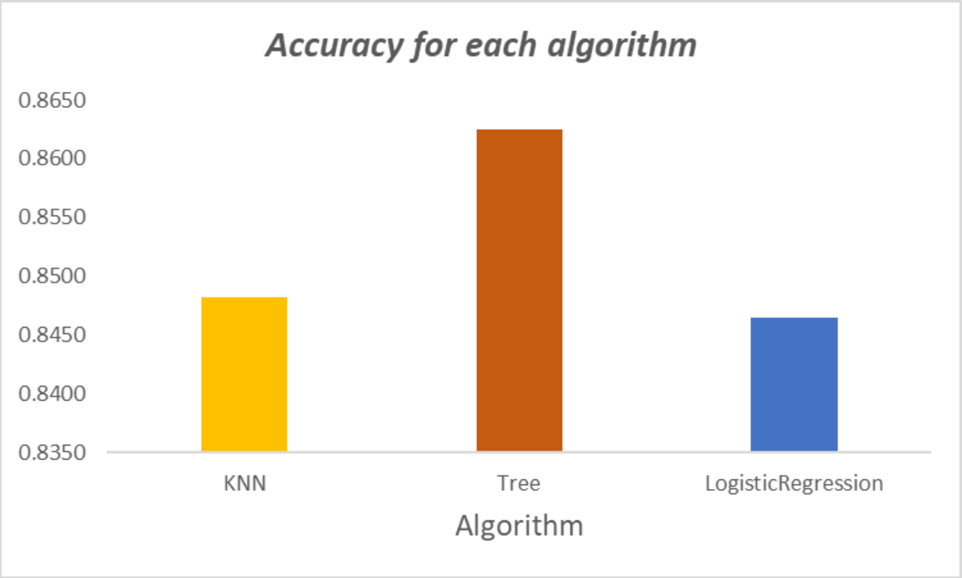
Classification Accuracy

As can be seen in the table below, the 3 models present similar results, with a slight advantage for the tree algorithm.

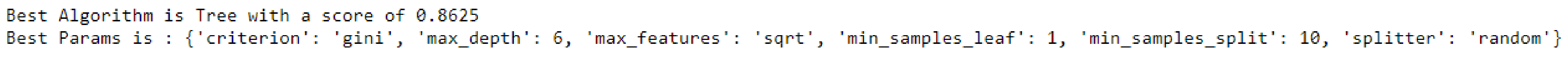


```
bestalgorithm = max(algorithms, key=algorithms.get)
```

Algorithm	Accuracy
KNN	0.8482
Tree	0.8625
LogisticRegression	0.8464



After selecting the best hyperparameters for the decision tree classifier using the validation data, 86.25% of accuracy is obtained.



```
Best Algorithm is Tree with a score of 0.8625
Best Params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 10, 'splitter': 'random'}
```

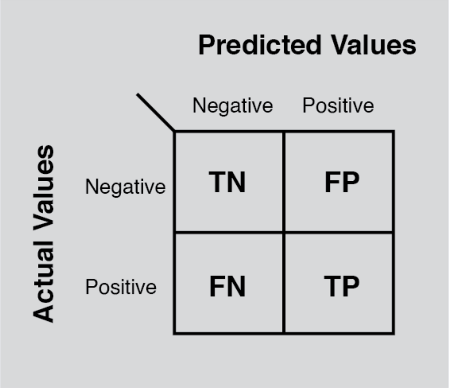
Confusion Matrix – Tree algorithm

Examining the confusion matrix,

we see that Tree can distinguish

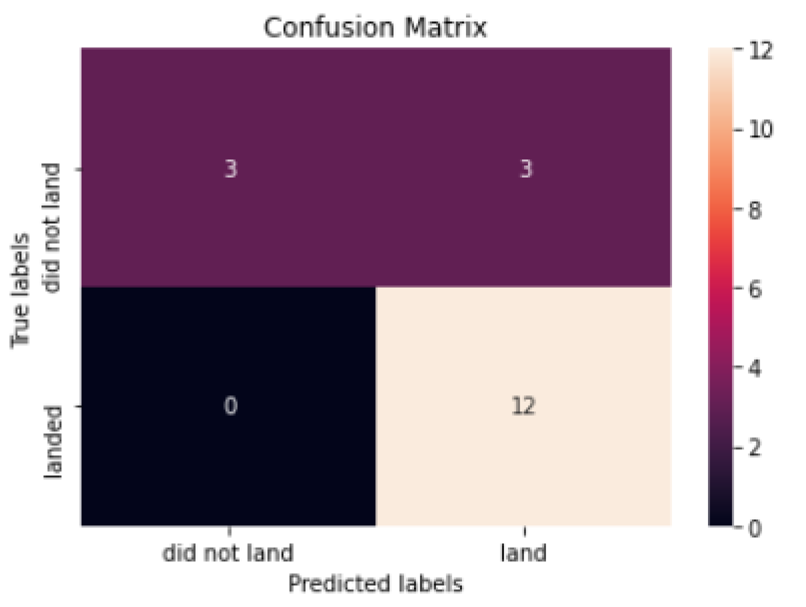
between the different classes.

We see that the major problem is false positives.



Predicted Values

Actual Values		Negative	Positive
	Negative	TN	FP
	Positive	FN	TP



Conclusions

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

Thank you!