IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Dongwon Yoon
11-10-2021

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
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

- Space X 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 **Space X can reuse the first stage**. Therefore if we can determine if the first stage will land, we can determine the cost of a launch.

- Our job is to **determine the price of each launch**. We will do this by gathering information about Space X and creating dashboards. We will also **determine if SpaceX will reuse the first stage**. We will train a machine learning model and use public information to predict if SpaceX will reuse the first stage.

Section 1

Methodology

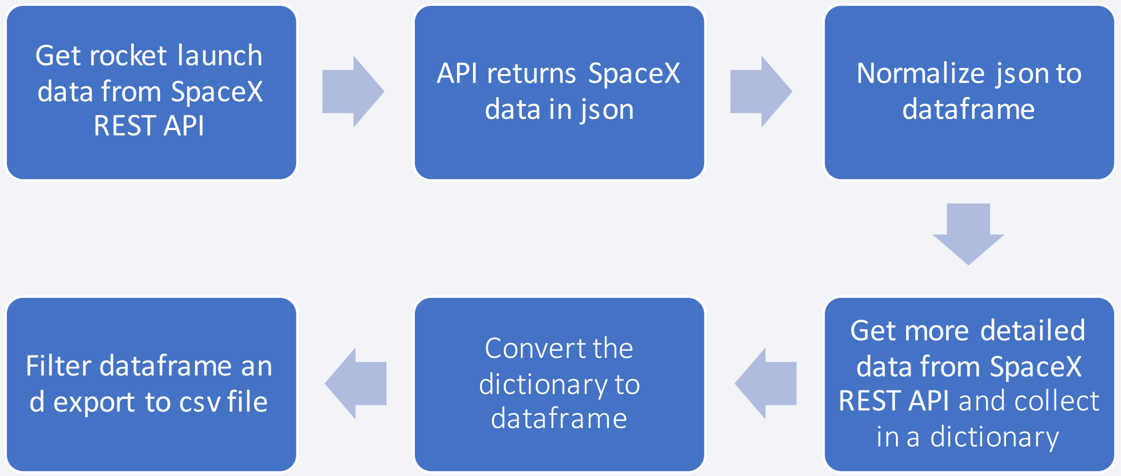
Methodology

- **Data collection** methodology:
    - Collected Data by "Data Collection API" and "Webscraping"
- Perform **data wrangling**
    - Converted the "Outcome" column into training labels with 1 means the booster successfully landed and 0 means it was unsuccessful.
- Perform **exploratory data analysis (EDA)** using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform **predictive analysis** using classification models
    - Used classification models. (Logistic Regression, Support Vector machines, Decision Tree Classifier and K-nearest neighbors)
    - Preprocess and train_test_split
    - Train the model and perform Grid Search to find the best hyperparameters.
    - Evaluated the classification models with the best accuracy using the test data.
    - We draw the confusion matrix to check in which our classification model
      is confused when it makes predictions

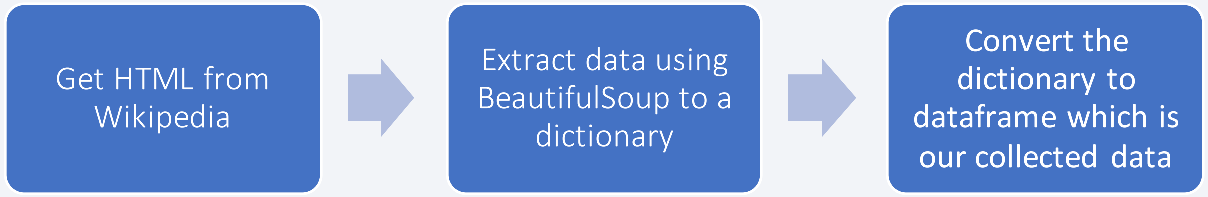
Data Collection

- We collected Data using "SpaceX REST API" and "Webscraping"

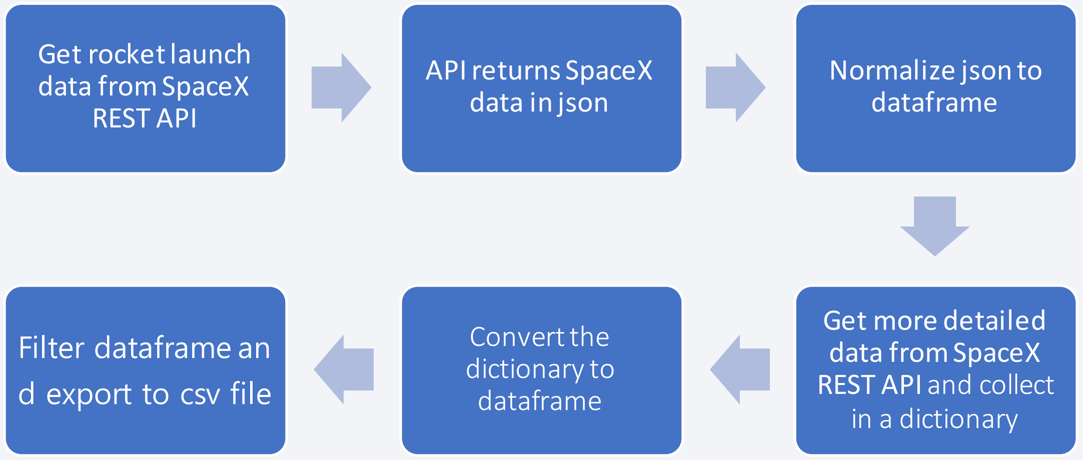
SpaceX REST API


Get rocket launch data from SpaceX REST API
API returns SpaceX data in json
Normalize json to dataframe
Get more detailed data from SpaceX REST API and collect in a dictionary
Convert the dictionary to dataframe
Filter dataframe an d export to csv file

Web Scraping


Get HTML from Wikipedia
Extract data using BeautifulSoup to a dictionary
Convert the dictionary to dataframe which is our collected data

Data Collection – SpaceX API


Get rocket launch data from SpaceX REST API
API returns SpaceX data in json
Normalize json to dataframe
Get more detailed data from SpaceX REST API and collect in a dictionary
Convert the dictionary to dataframe
Filter dataframe and export to csv file

- GitHub URL of the completed SpaceX API calls notebook
  - Lab 1-1: Data Collection API Lab.ipynb

```
# Get rocket launch data from SpaceX REST API
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

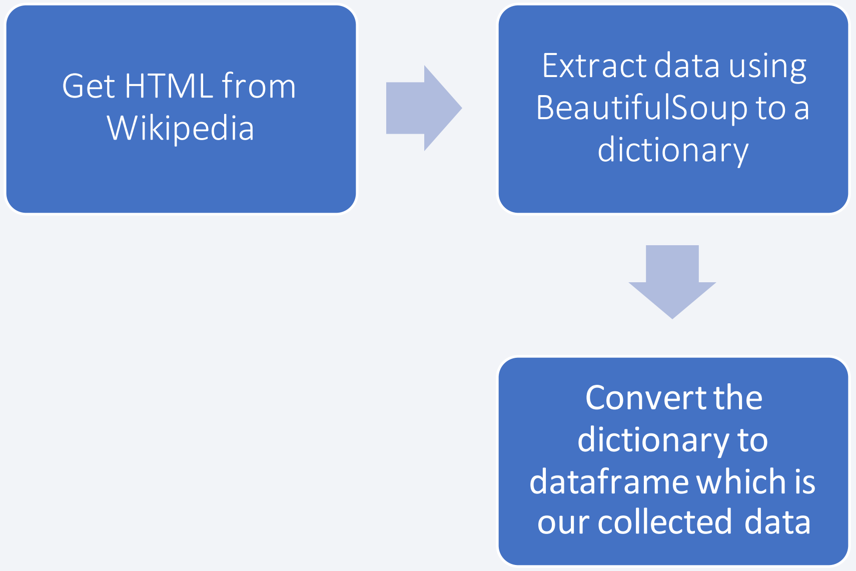
```
# API returns SpaceX data in json, Normalize json to dataframe
j = response.json()
data = pd.json_normalize(j)
```

```
# Get more detailed data from SpaceX REST API and collect in a dictionary
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}
```

```
# Convert the dictionary to dataframe
df = pd.DataFrame.from_dict(launch_dict)
```

```
# Filter dataframe and export to csv file
data_falcon9 = df[df['BoosterVersion']!='Falcon 1']
...
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping


Get HTML from Wikipedia
Extract data using BeautifulSoup to a dictionary
Convert the dictionary to dataframe which is our collected data

- GitHub URL of the completed web scraping notebook
  - Lab 1-2: Data Collection with Web Scraping lab.ipynb

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

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

```
html_tables = []
rows = soup.find_all('tr')
for row in rows:
    html_tables.append(row)
```

```
ths = first_launch_table.find_all('th')

for th in ths:
    #print(th)
    column_name = extract_column_from_header(th)
    #print(column_name)
    if column_name is not None and len(column_name) > 0:
        column_names.append(column_name)
```

```
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty
list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
...
```

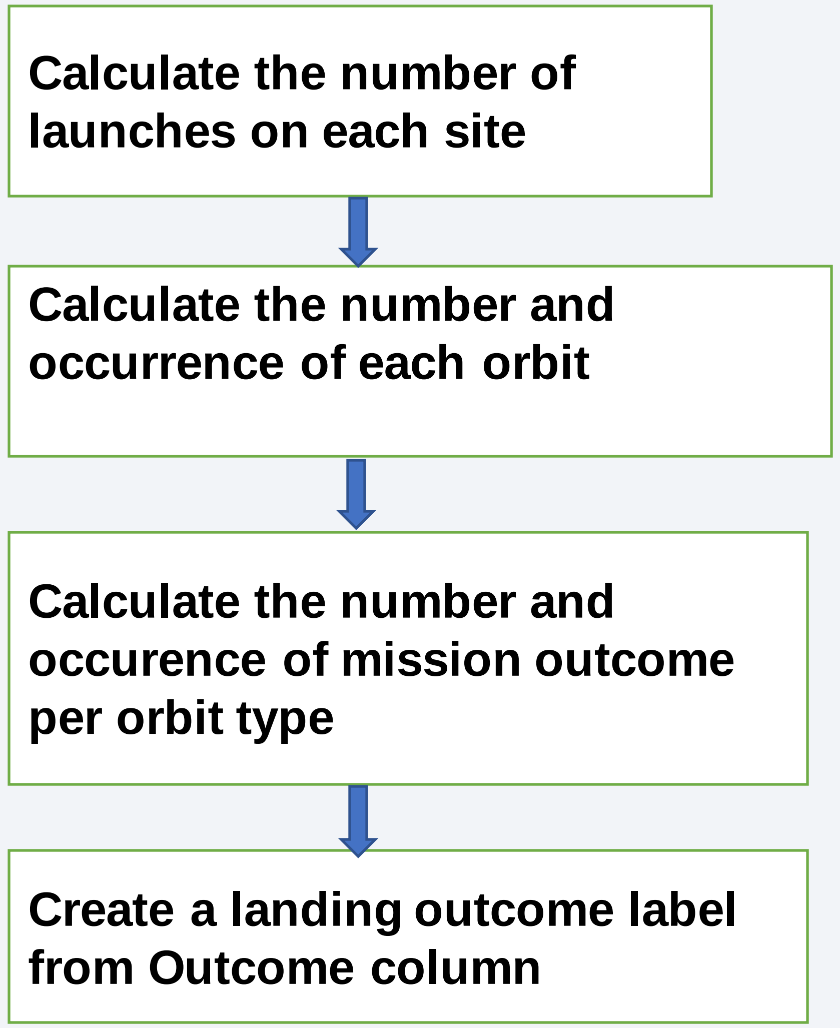
```
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
    for rows in table.find_all("tr"):
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False

        row=rows.find_all('td')

        if flag:
            #print(row)

            extracted_row += 1
            launch_dict['Flight No.'].append(flight_number)
            datatimelist=date_time(row[0])
```

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

Data Wrangling

We converted 8 kinds of landing outcomes to classes(variable Y) either 0 or 1. 0 is a bad outcome, that is, the booster did not land. 1 is a good outcome, that is, the booster did land. The variable Y will represent the classification variable that represents the outcome of each launch.

- GitHub URL of the completed data wrangling related notebooks
  - Lab 2: Data Wrangling

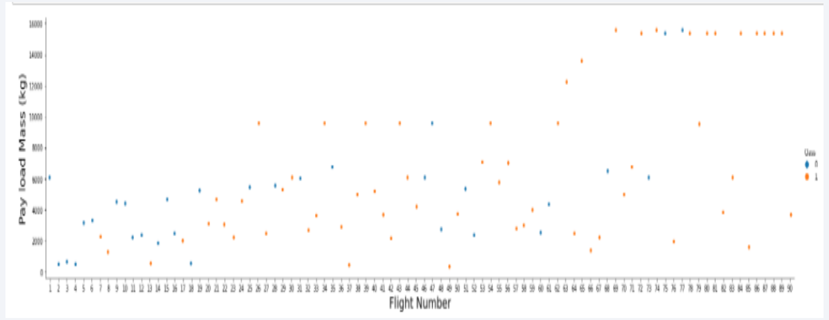

Calculate the number of launches on each site
Calculate the number and occurrence of each orbit
Calculate the number and occurence of mission outcome per orbit type
Create a landing outcome label from Outcome column

EDA with Data Visualization

Scatter Chart

Flight Number and Payload Mass
Flight Number and Launch Site
Payload and Launch Site
Flight Number and Orbit
Payload and Orbit

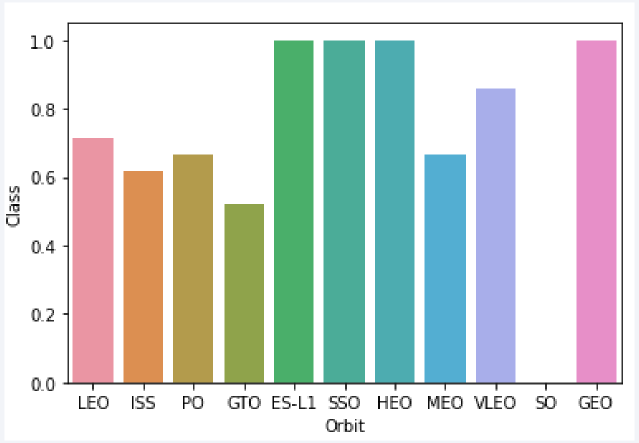
We used scatter plots to see the correlation between two variables.



[GitHub URL of EDA with data visualization notebook]

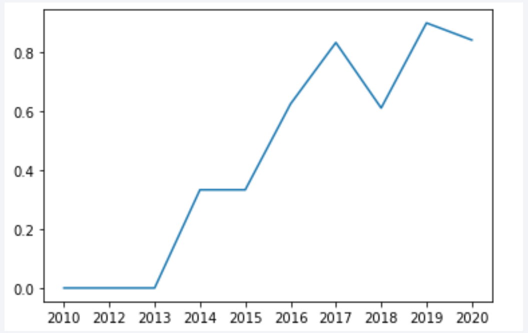
Bar Chart

This shows relationships between success rate and orbit type



Line Chart

x axis is Year and y axis is average success rate, this shows the average launch success trend



EDA with SQL(1)

1. Display the names of the unique launch sites in the space mission
   - select distinct LAUNCH_SITE from spacextbl;
2. Display 5 records where launch sites begin with the string 'CCA'
   - select * from spacextbl where LAUNCH_SITE like 'CCA%' limit 5;
3. Display the total payload mass carried by boosters launched by NASA (CRS)
   - select sum(payload_mass__kg_) from spacextbl where customer='NASA (CRS)';
4. Display average payload mass carried by booster version F9 v1.1
   - select avg(payload_mass__kg_) from spacextbl where booster_version='F9 v1.1'
5. List the date when the first successful landing outcome in ground pad was acheived.
   - select min(DATE) from spacextbl where landing__outcome like 'Success%';

[GitHub URL of your completed EDA with SQL notebook]()

EDA with SQL(2)

6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

   - select booster_version from spacextbl where landing__outcome='Success (drone ship)' and payload_mass__kg_ > 4000 and payload_mass__kg_ < 6000;

7. List the total number of successful and failure mission outcomes

   - select count(*) from spacextbl where MISSION_OUTCOME like 'Failure%';

8. List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

   - select booster_version from spacextbl where payload_mass__kg_=(select max(payload_mass__kg_) from spacextbl);

9. List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

   - select landing__outcome, booster_version, launch_site from spacextbl where landing__outcome like '%(drone ship)%' and year(DATE)=2015;

10. 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

    - select count(*) as cnt, landing__outcome from spacextbl where date > '2010-06-04' and date < '2017-03-20' group by landing__outcome order by cnt desc;

GitHub URL of your completed EDA with SQL notebook

Build an Interactive Map with Folium

Added map objects are

- Latitude and Longitude Coordinates
- Circle Marker for each launch site with a label to show the name of launch site.
- Green and Red markers to indicate success and failure for launch outcomes.
- Marker Cluster to draw many locations
- Lines are drawn on the map to measure distance to landmarks

- GitHub URL of your completed interactive map with Folium map

Build a Dashboard with Plotly Dash

Dashboard was built with Flask and Dash.
Graphs
- Pie Chart shows 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 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 of Plotly Dash lab

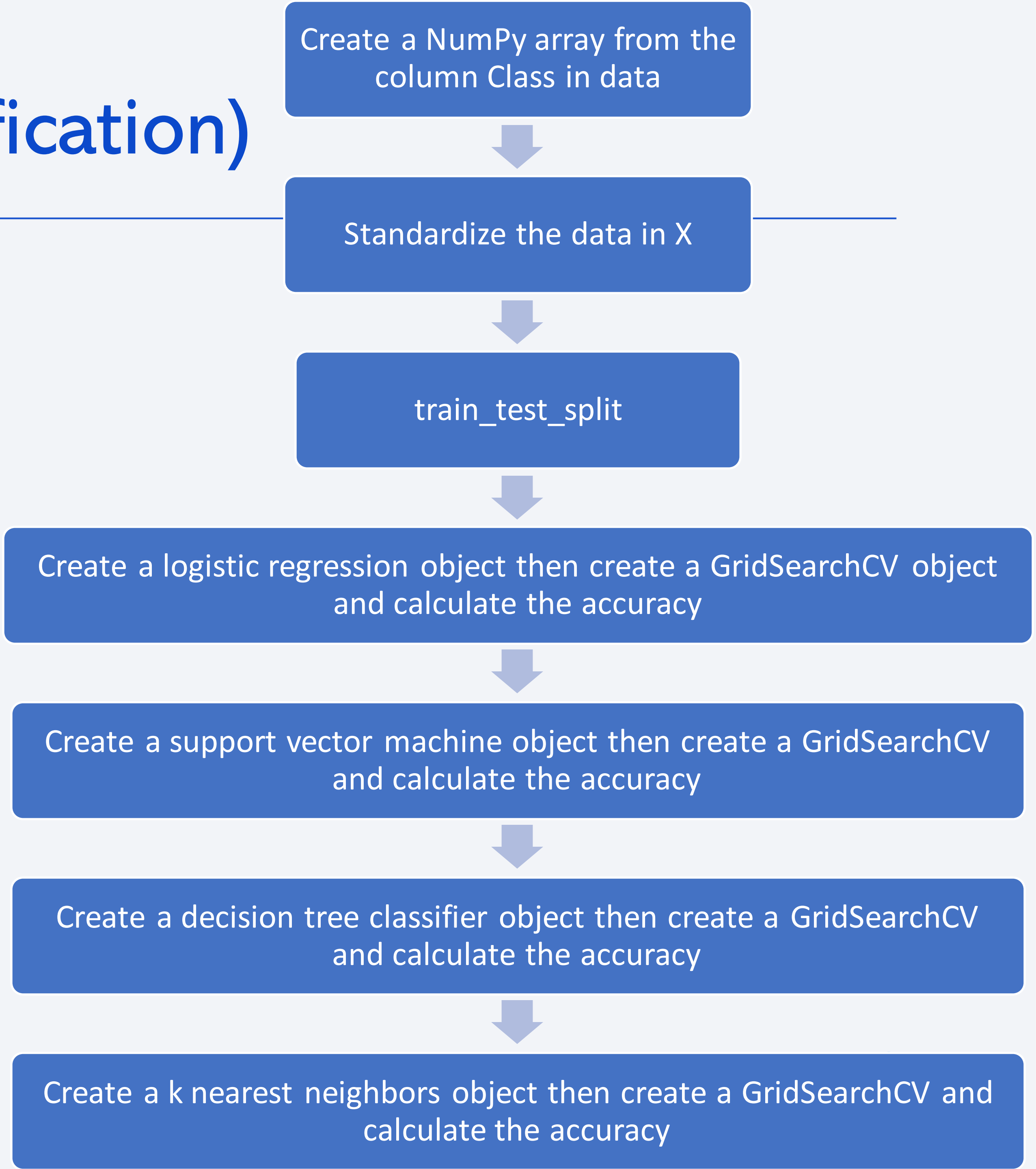
Predictive Analysis (Classification)

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 algorithm
- Plot Confusion Matrix

Improving Model
- Feature Engineering
- Algorithm Tuning

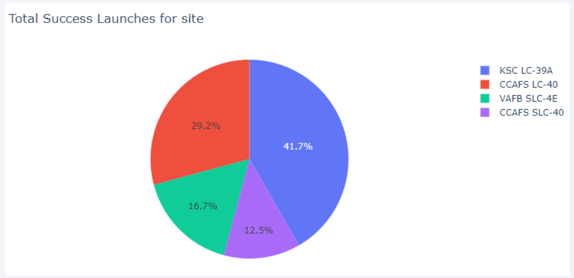
Finding the best classification model
- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithm with scores at the bottom of the notebook

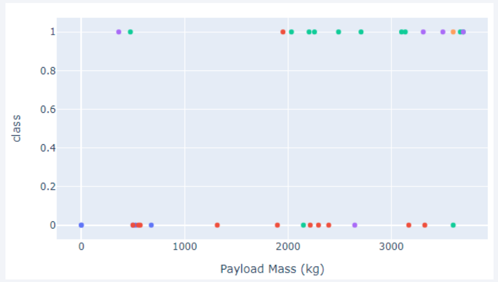

Create a NumPy array from the column Class in data
Standardize the data in X
train_test_split
Create a logistic regression object then create a GridSearchCV object and calculate the accuracy
Create a support vector machine object then create a GridSearchCV and calculate the accuracy
Create a decision tree classifier object then create a GridSearchCV and calculate the accuracy
Create a k nearest neighbors object then create a GridSearchCV and calculate the accuracy

GitHub URL of completed predictive analysis lab

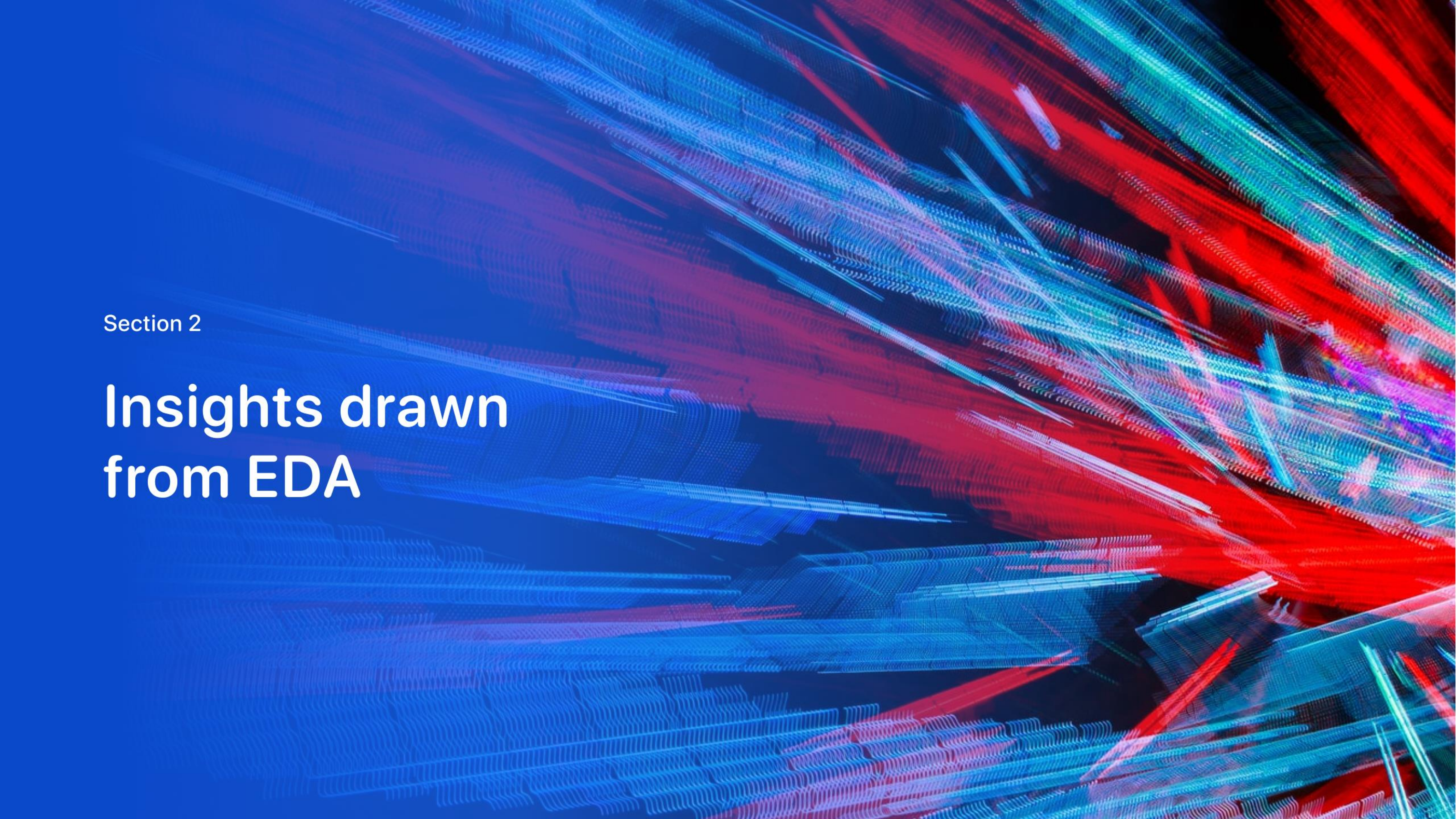
Results

- Exploratory data analysis results
  - Low weighted payloads perform better than the heaver payloads
  - The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
  - We can see that KCC LC-39A had the most successful launches from all the sites
  - Orbit GEO, HEO, SSO, ES-L1 has the best success rate.
- Interactive analytics demo in screenshots





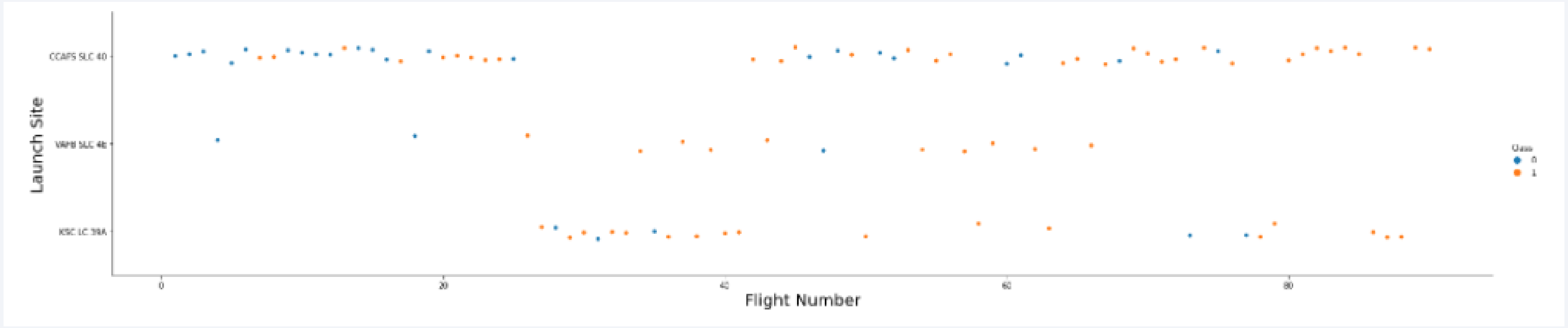
- Predictive analysis results
  - **Decision tree classifier** is the most accurate model.



Section 2

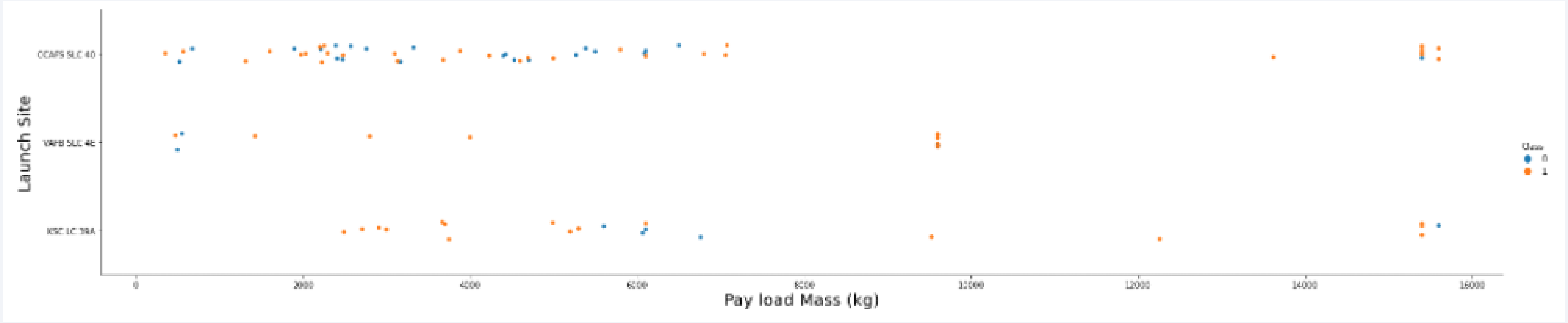
Insights drawn from EDA

Flight Number vs. Launch Site



- Overall, as Flight Number increases, the success rate seems to increase for all 3 Launch Site as well. And recently, most of the launches have been successful.

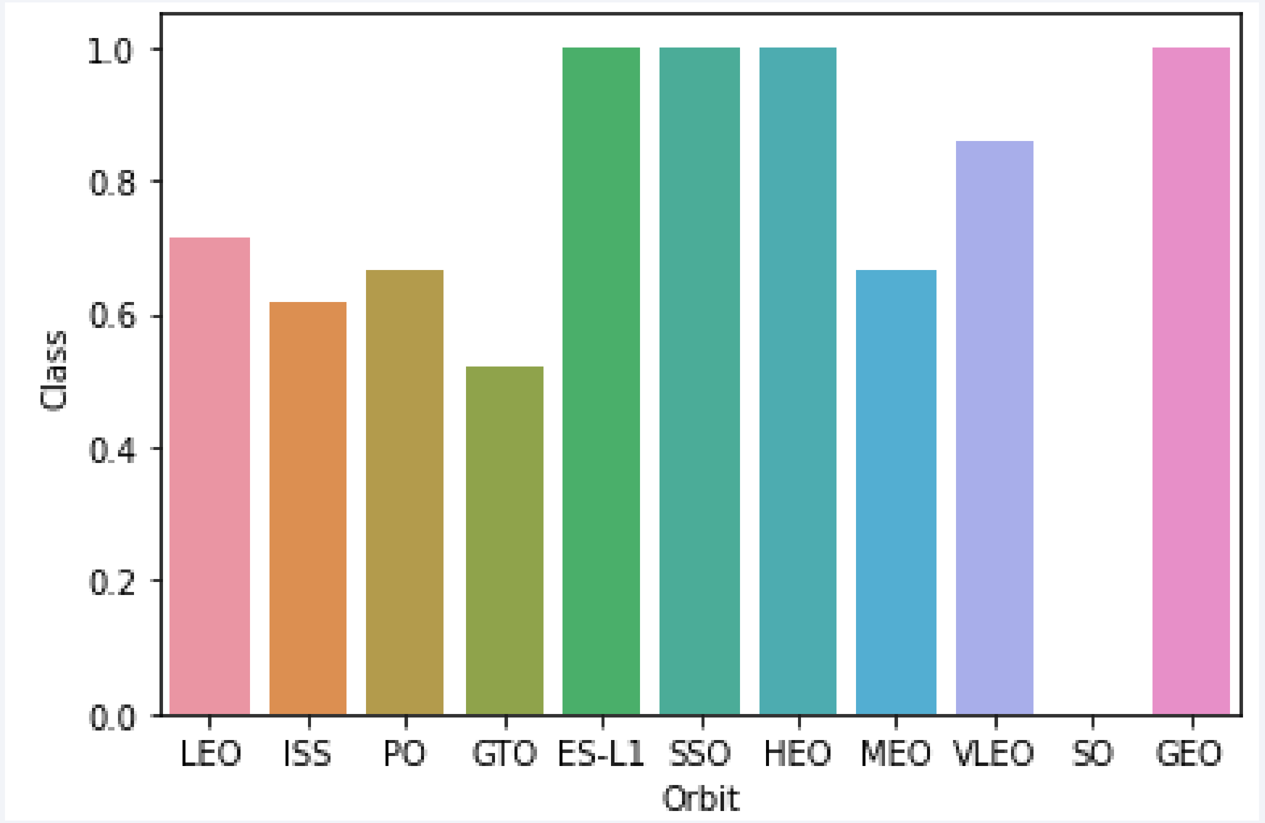
Payload vs. Launch Site



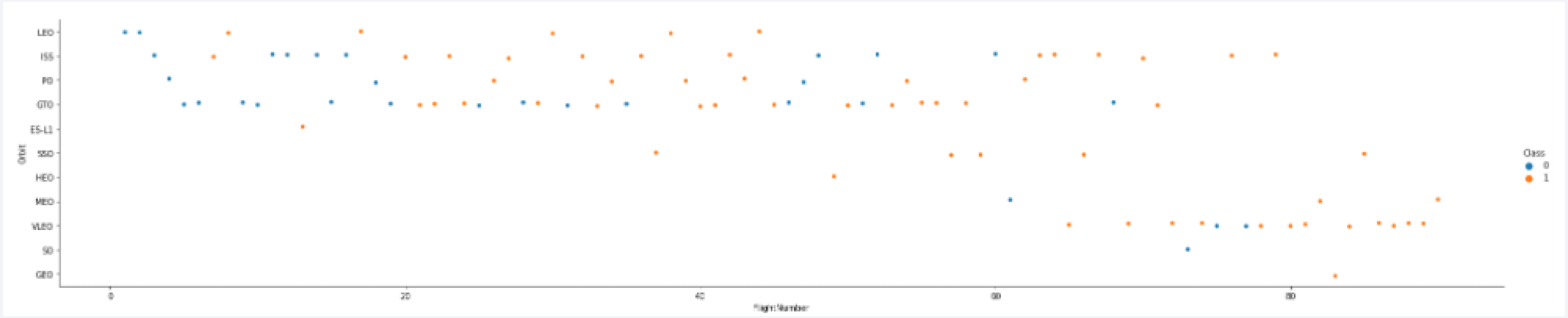
- The greater the payload mass, the higher the success rate for the Rocket. There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependent on Pay Load Mess for a success launch.

Success Rate vs. Orbit Type

- Orbit GEO, HEO, SSO and ES-L1 have the best success rate.

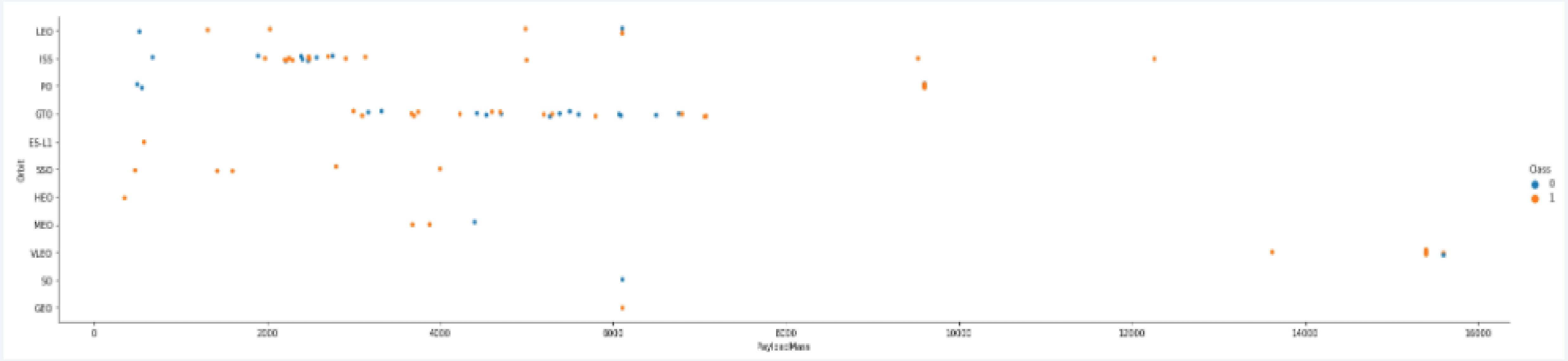


Flight Number vs. Orbit Type



- For 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.

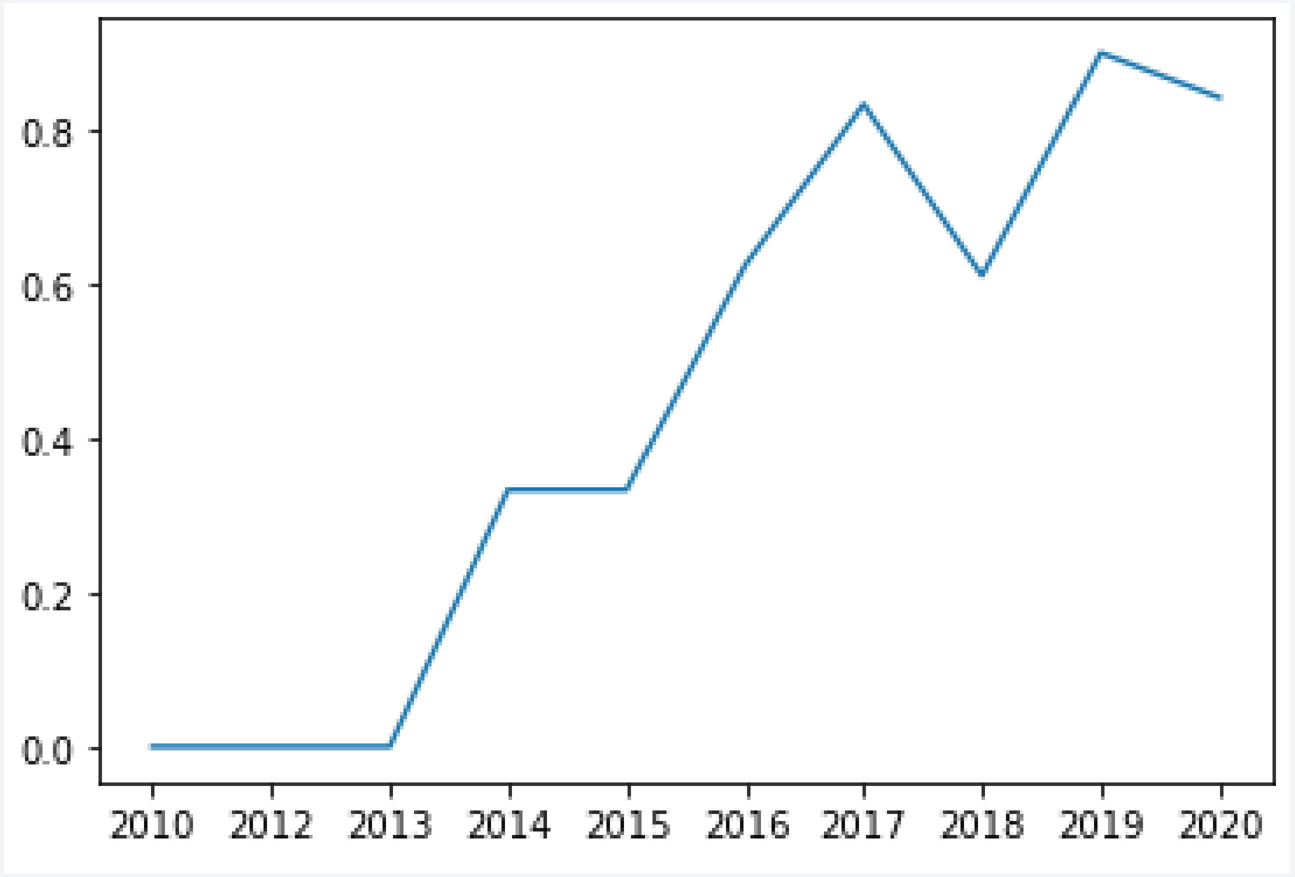
Payload vs. Orbit Type



- Heavy payloads tends to have a negative influence on GTO orbit and positive on ISS orbit.

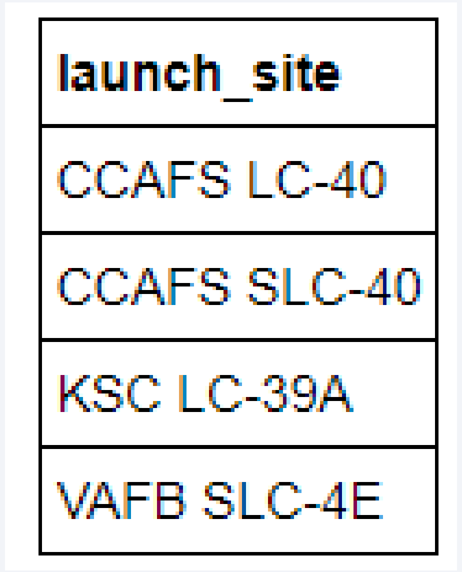
Launch Success Yearly Trend

- The success rate kept increasing since 2013 till 2017.



All Launch Site Names

```
select distinct LAUNCH_SITE from spacextbl;
```



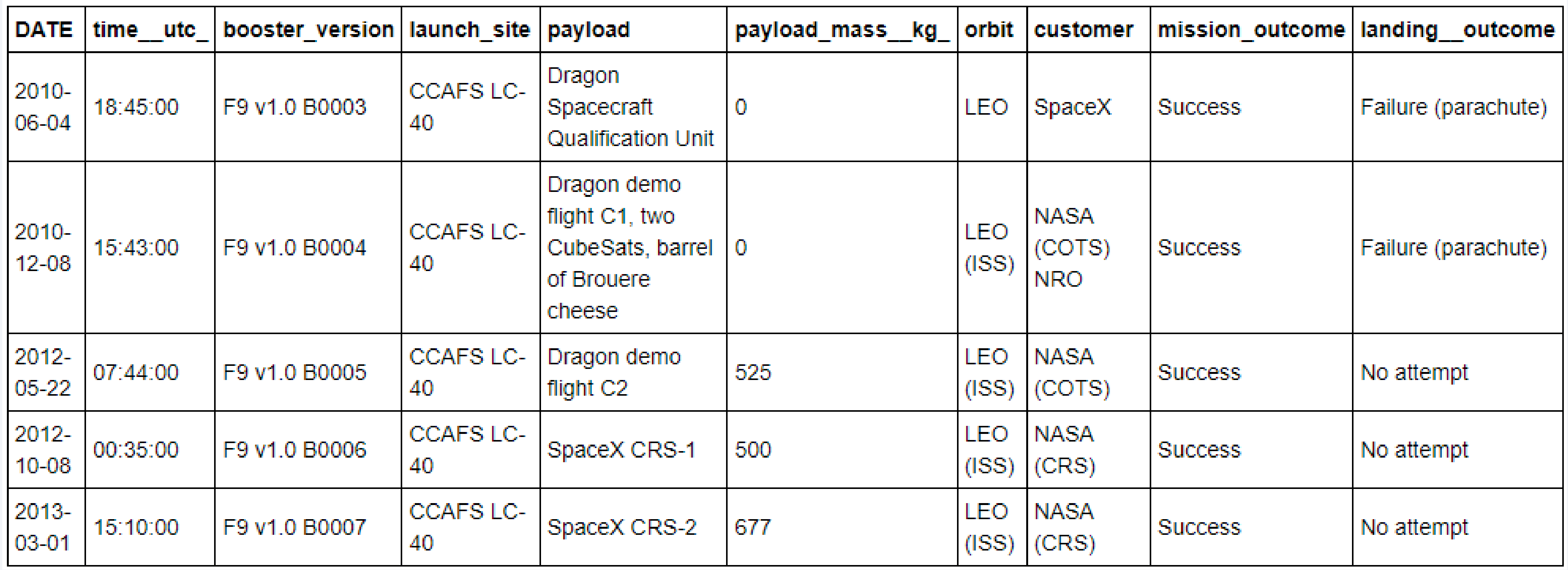
launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Using the word ***distinct*** *in the query means that it will only show unique values in the Launch Site column from spacextbl*

Launch Site Names Begin with 'CCA'

select * from spacextbl where LAUNCH_SITE like 'CCA%' limit 5;

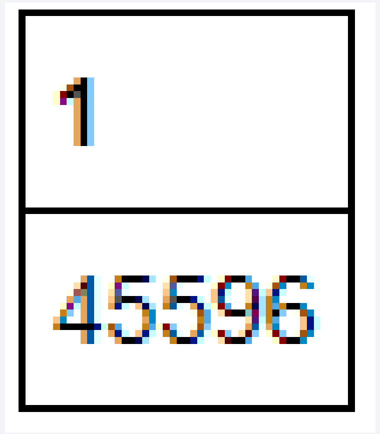
Filtered LAUNCH_SITE begin with ‘CCA’ using like keyword and used limit keyword to display only 5 records.



DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
select sum(payload_mass__kg_) from spacextbl where customer='NASA (CRS)';
```



1
45596

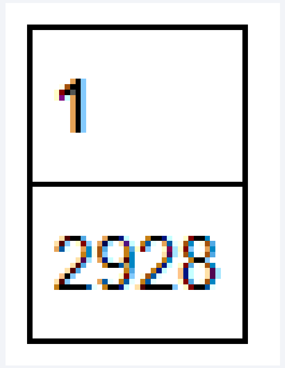
Using the function sum(payload_mass__kg) we can get the total in the column payload_mass__kg

The where clause filters the dataset to only perform calculation on NASA (CRS)

Average Payload Mass by F9 v1.1

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

select avg(payload_mass__kg_) from spacextbl where booster_version='F9 v1.1'



1
2928

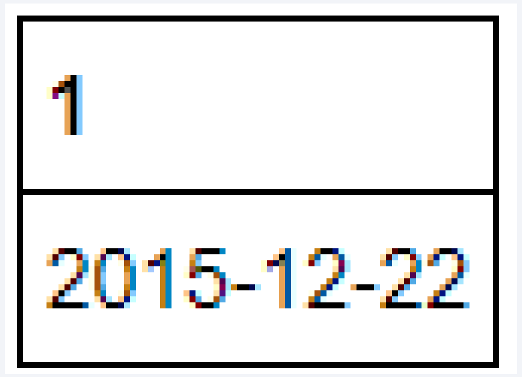
Using the function avg() works out the average in the column payload_mass__kg

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

select min(DATE) from spacextbl where landing__outcome like 'Success%';



1
2015-12-22

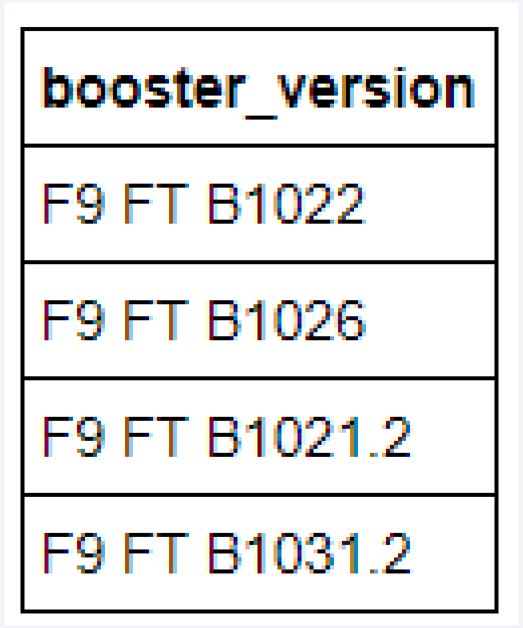
Using the function min() works out the minimum date in the column DATE.

The where clause filters the dataset to only perform calculations when landing__outcome is successful.

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

```
select booster_version from spacextbl where landing__outcome='Success (drone ship)' and payload_mass__kg_ > 4000 and payload_mass__kg_ < 6000;
```



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

Selecting only booster_version.

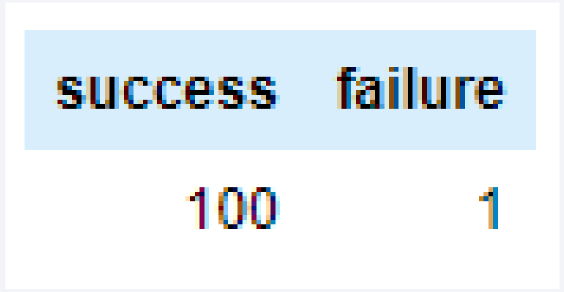
The where clause filters the dataset to landing__outcome=Success (drone ship).

The and clause specifies additional filter conditions of payload_mass__kg_'s range.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
select \
(select count(*) from spacextbl where MISSION_OUTCOME  like 'Success%')  as Success,  \
(select count(*) from spacextbl where MISSION_OUTCOME  like 'Failure%') as Failure \
from SYSIBM.SYSDUMMY1;
```



success	failure
100	1

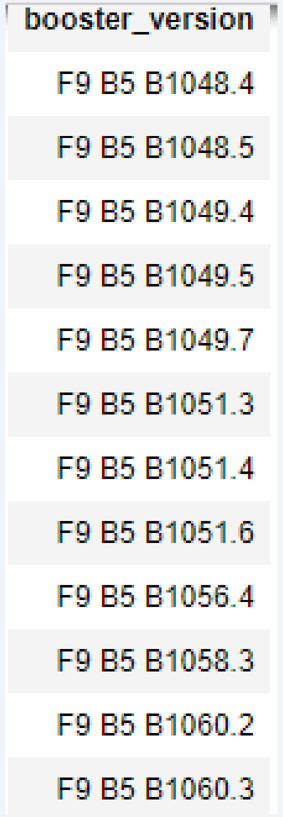
Used count() function which calculates the total number of result records.

The where clause filters the dataset to MISSION_OUTCOME including Success or Failure

Boosters Carried Maximum Payload

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

select distinct booster_version from spacextbl where
payload_mass__kg_=(select max(payload_mass__kg_) from spacextbl);



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

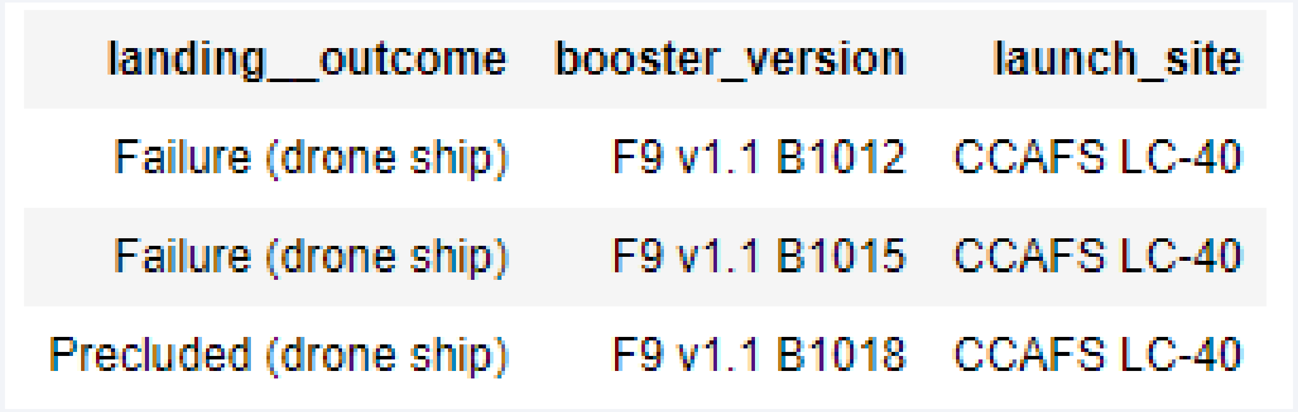
Using distinct in the query means that it will only show unique values in the booster_version column from spacextbl

Used subquery to get the maximum payload mass from spacextbl

2015 Launch Records

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

select landing__outcome, booster_version, launch_site from spacextbl where landing__outcome like '%(drone ship)%' and year(DATE)=2015;

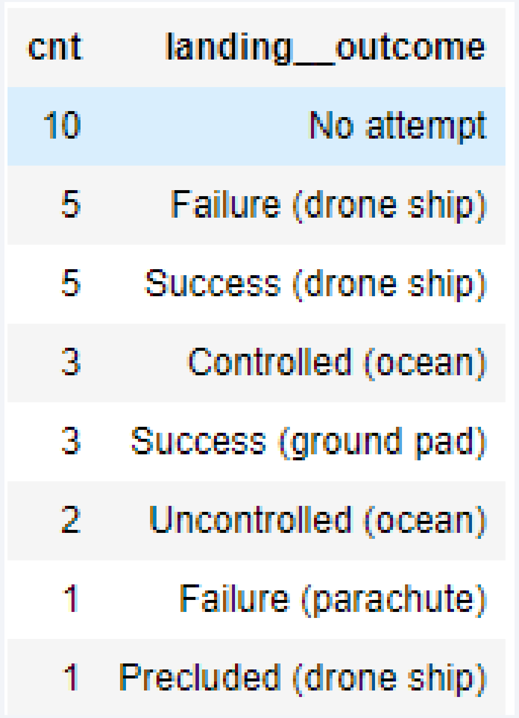


landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40
Precluded (drone ship)	F9 v1.1 B1018	CCAFS LC-40

The where clause filters landing__outcomes in drone ship and DATE with year 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
- select count(*) as cnt, landing__outcome from spacextbl where date > '2010-06-04' and date < '2017-03-20' group by landing__outcome order by cnt desc;



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

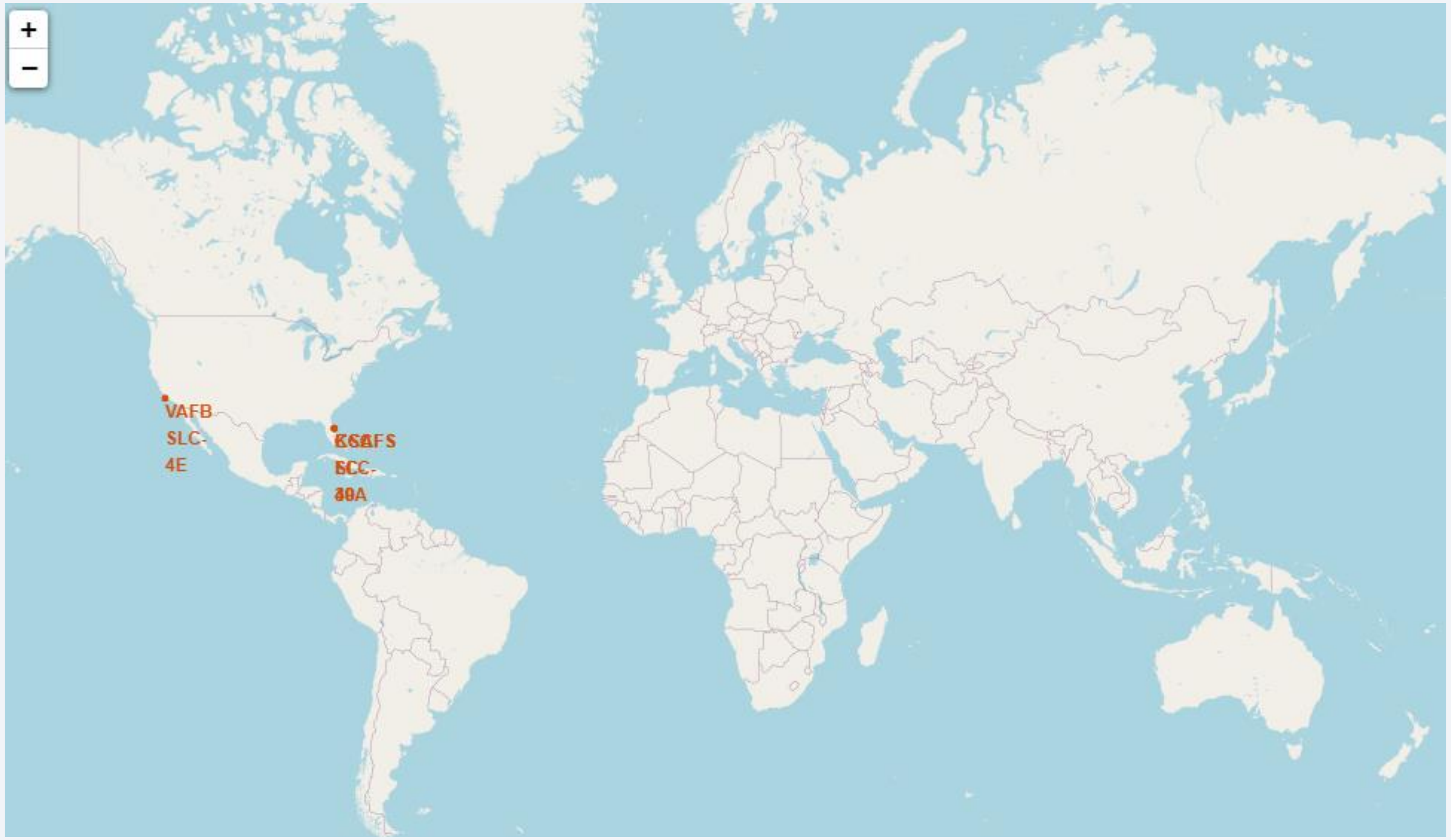
To get count of landing outcomes, we filtered date and grouped by landing outcome.

To show rank by count, we ordered the result table by descending order

Section 4

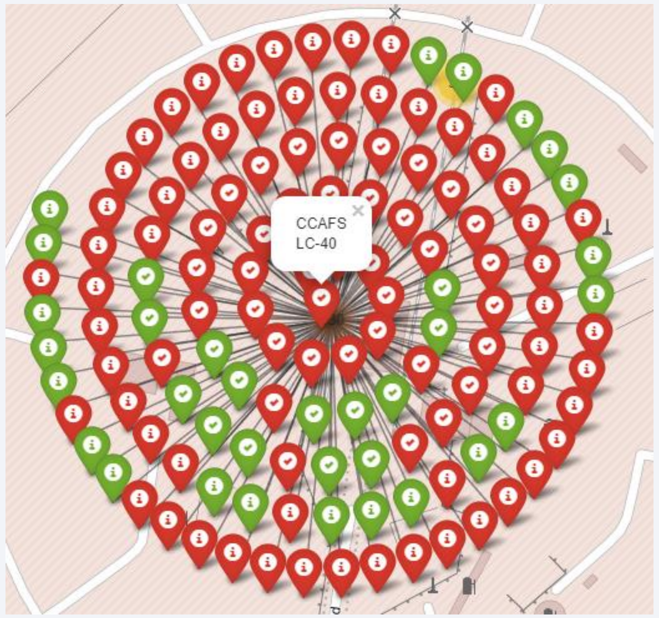
Launch Sites Proximities Analysis

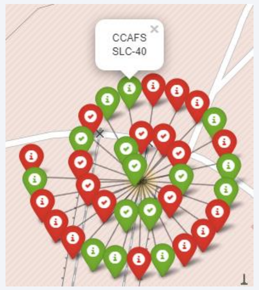
Mark all launch sites on a map

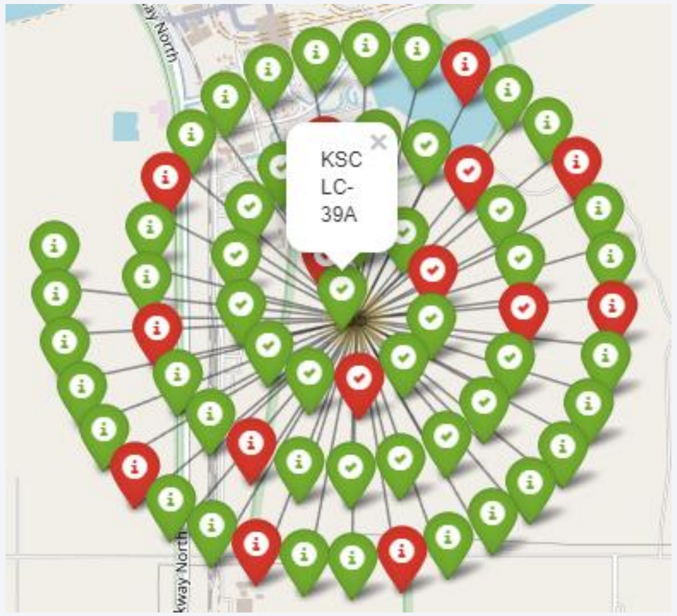


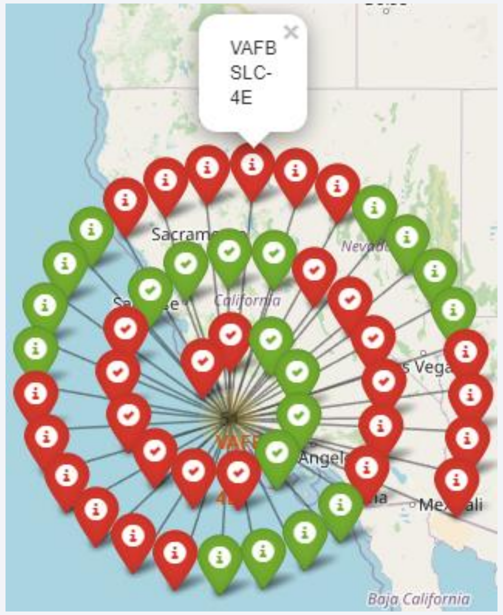
We can see that the SpaceX launch sites are in the USA coasts, Florida and California.

<Folium Map Screenshot 2>







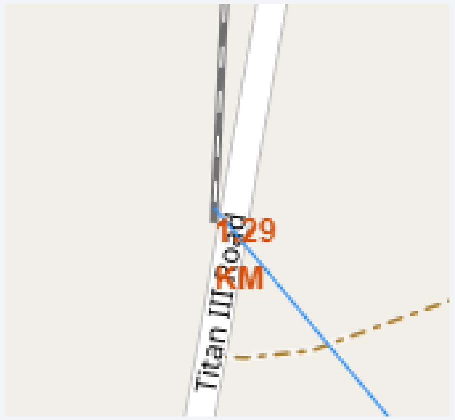


3 Florida launch sites

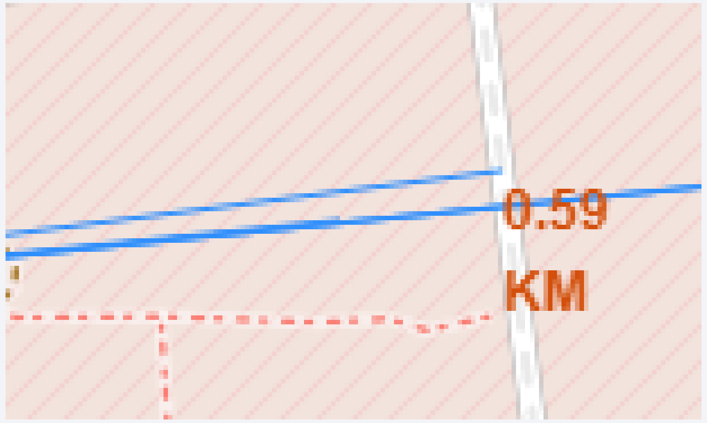
1 California launch sites

Green Marker shows successful launches and Red Marker shows failures.
We can see that KSC LC-39A site was most successful.

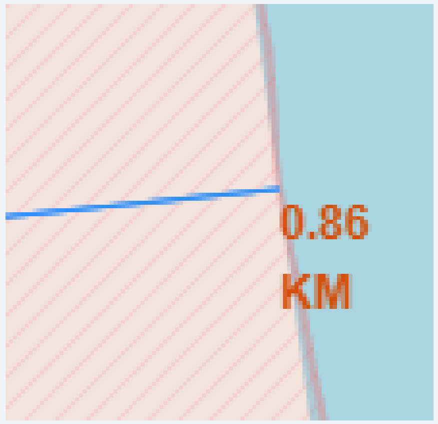
Calculate the distances between a launch site to its proximities



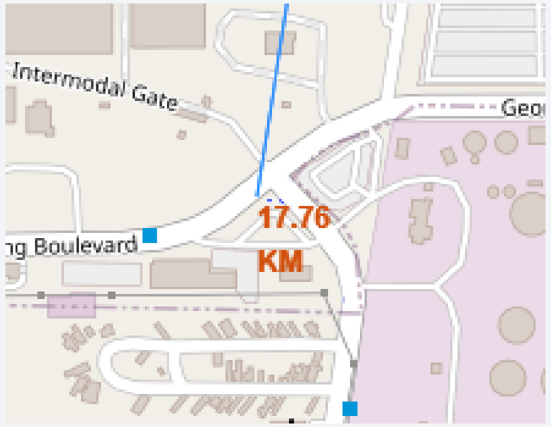
Railway



Highway

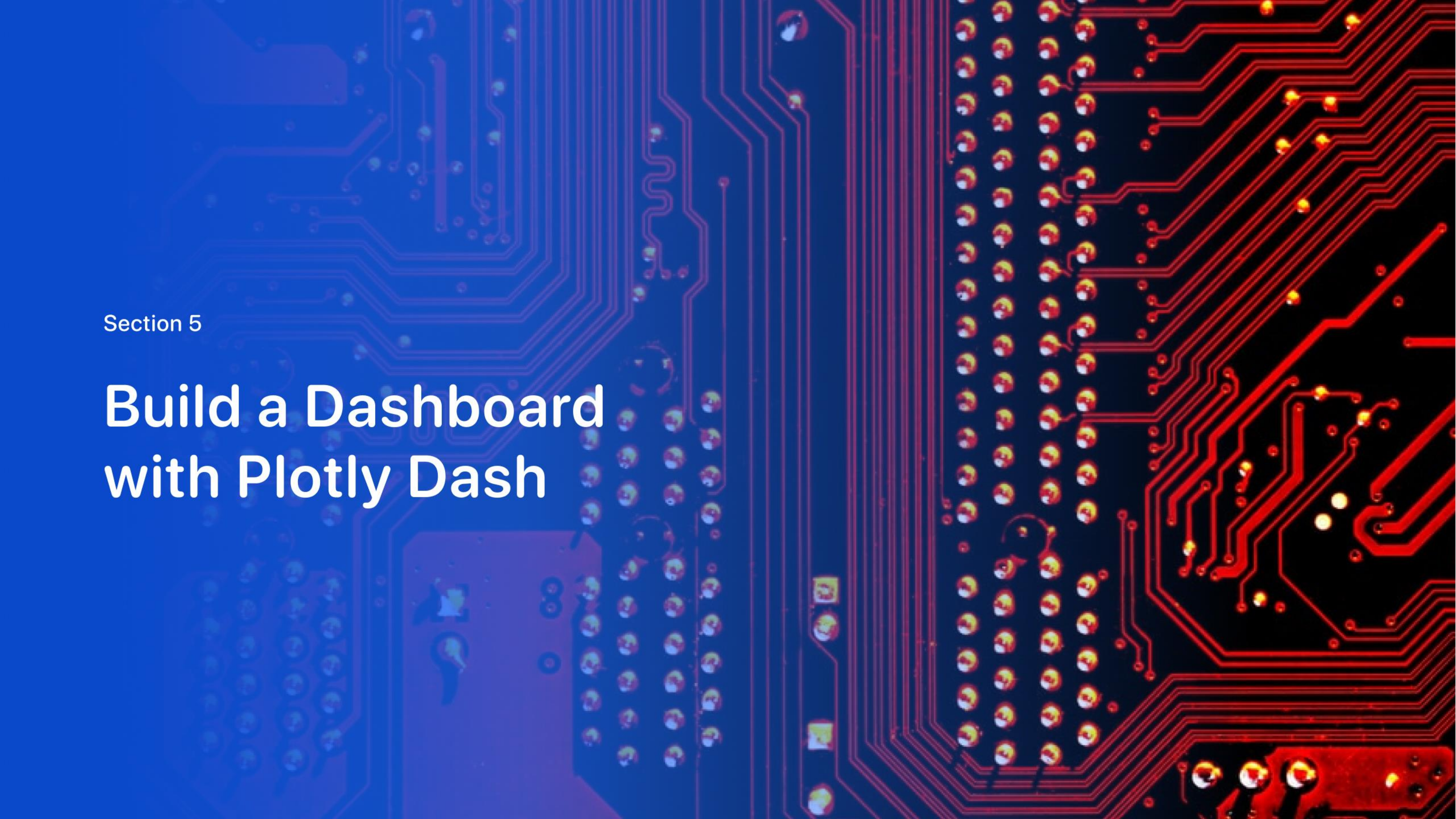


Coastline



City

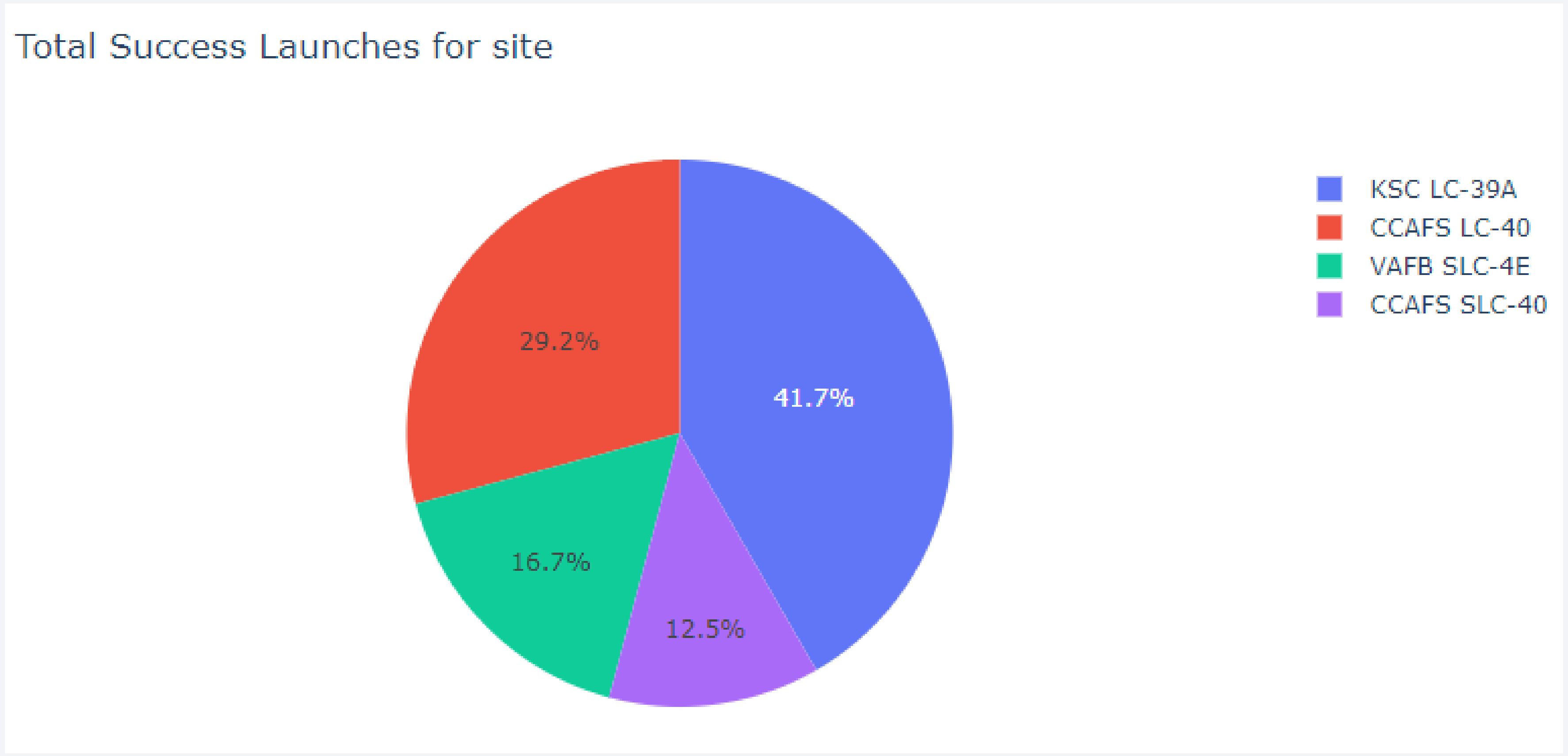
- Are launch sites in close proximity to railways? No, the distance is longer than 1 km.
- Are launch sites in close proximity to highways? Yes, the distance is less than 1 km.
- Are launch sites in close proximity to coastline? Yes, the distance is less than 1 km
- Do launch sites keep certain distance away from cities? Yes, this distance is very long around 17 km



Section 5

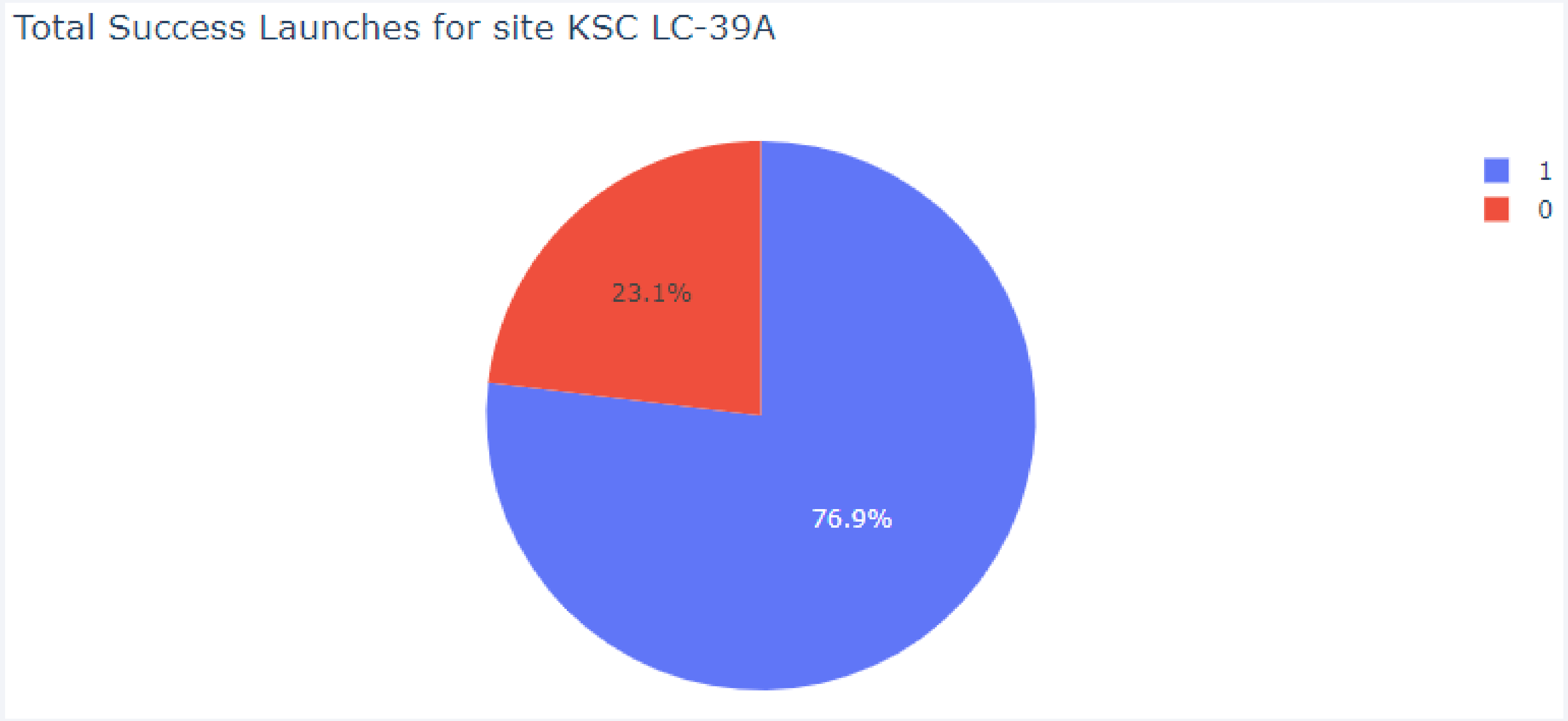
Build a Dashboard with Plotly Dash

Launch success for all sites



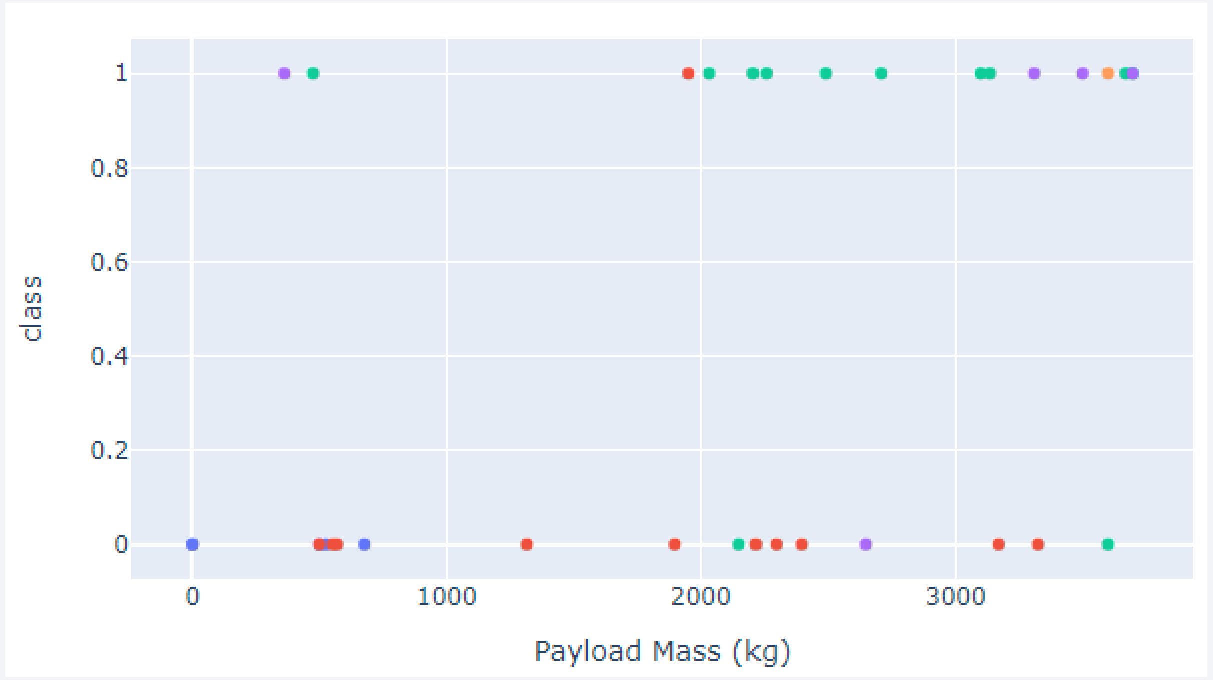
- We can see that KSC LC-39A had the most successful launches from all the sites

Pie chart for the launch site KSC LC-39A which is the highest launch success ratio

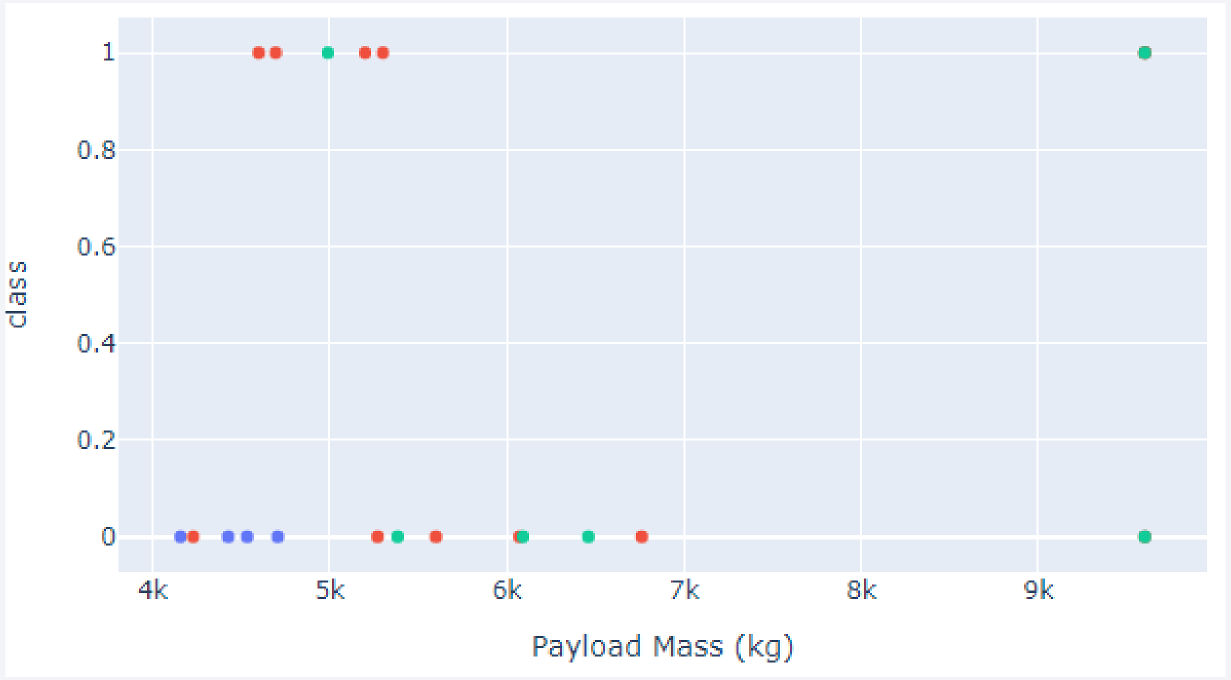


- KSC LC-39A achieved a 76.9% success rate getting a 23.1% failure rate.

Payload and Launch Outcome scatter plot for all sites, With different payload selected in the range slider



Low weighted Payload 0 kg ~ 4000 kg



Heavy weighted Payload 4000 kg ~ 10000 kg

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

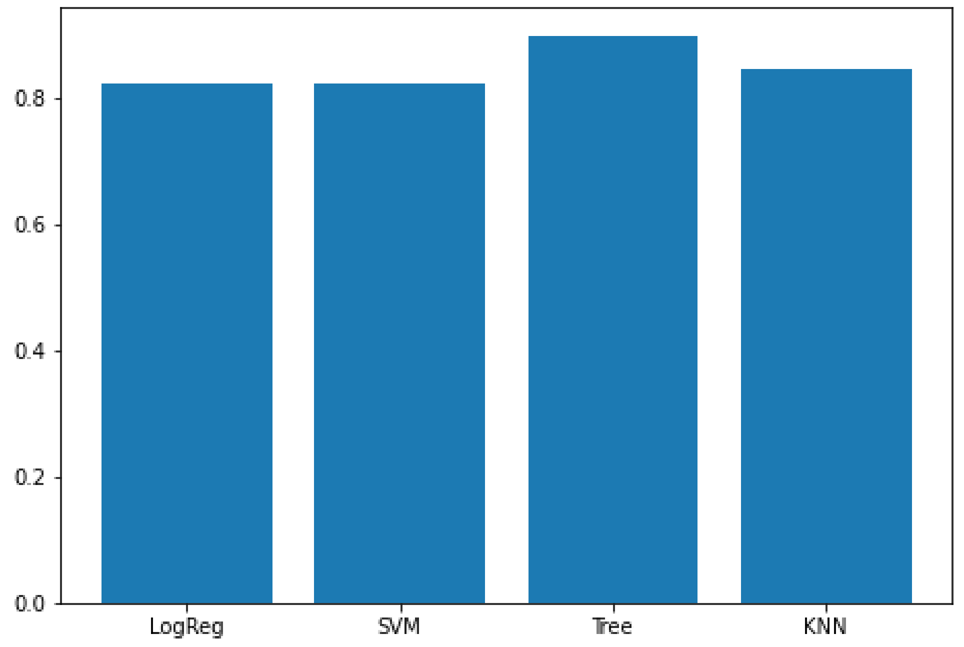
Section 6

Predictive Analysis (Classification)

Classification Accuracy

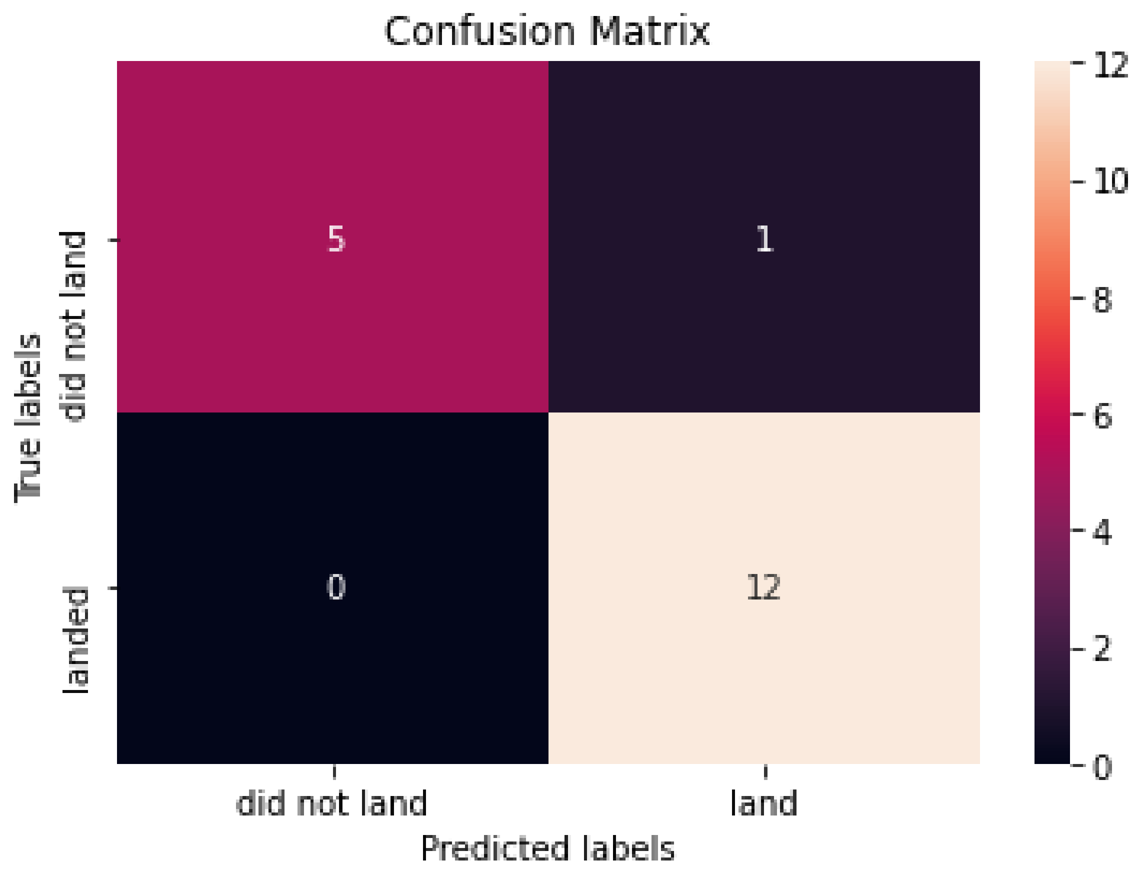
- We can see that decision tree classifier is the most accurate model.

```
logistic regression train set accuracy : 0.8222222222222222
svm train set accuracy : 0.8222222222222223
tree train set accuracy : 0.9
knn train set accuracy : 0.8444444444444444
```



Confusion Matrix for decision tree

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



Conclusions

- The decision tree classifier algorithm is the best for this dataset
- Low weighted payloads perform better than the heaver payloads
- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
- We can see that KCC LC-39A had the most successful launches from all the sites
- Orbit GEO, HEO, SSO, ES-L1 has the best success rate.
- **Insight: Recently, the success rate is almost 100%, so it would be okay to expect SpaceX will always reuse the first stage.**

Appendix – Data Source

- SpaceX URL: https://api.spacexdata.com/v4/launches/past

- Wiki - List of Falcon 9 and Falcon Heavy launches: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Appendix - Haversine formula

- The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. Important in navigation, it is a special case of a more general formula in spherical trigonometry, the law of haversines, that relates the sides and angles of spherical triangles.
- https://en.wikipedia.org/wiki/Haversine_formula

$$\begin{aligned} d &= 2r \arcsin\left(\sqrt{\operatorname{hav}(\varphi_2 - \varphi_1) + \cos(\varphi_1)\cos(\varphi_2)\operatorname{hav}(\lambda_2 - \lambda_1)}\right) \\ &= 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + \cos(\varphi_1)\cos(\varphi_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right) \end{aligned}$$

Thank you!