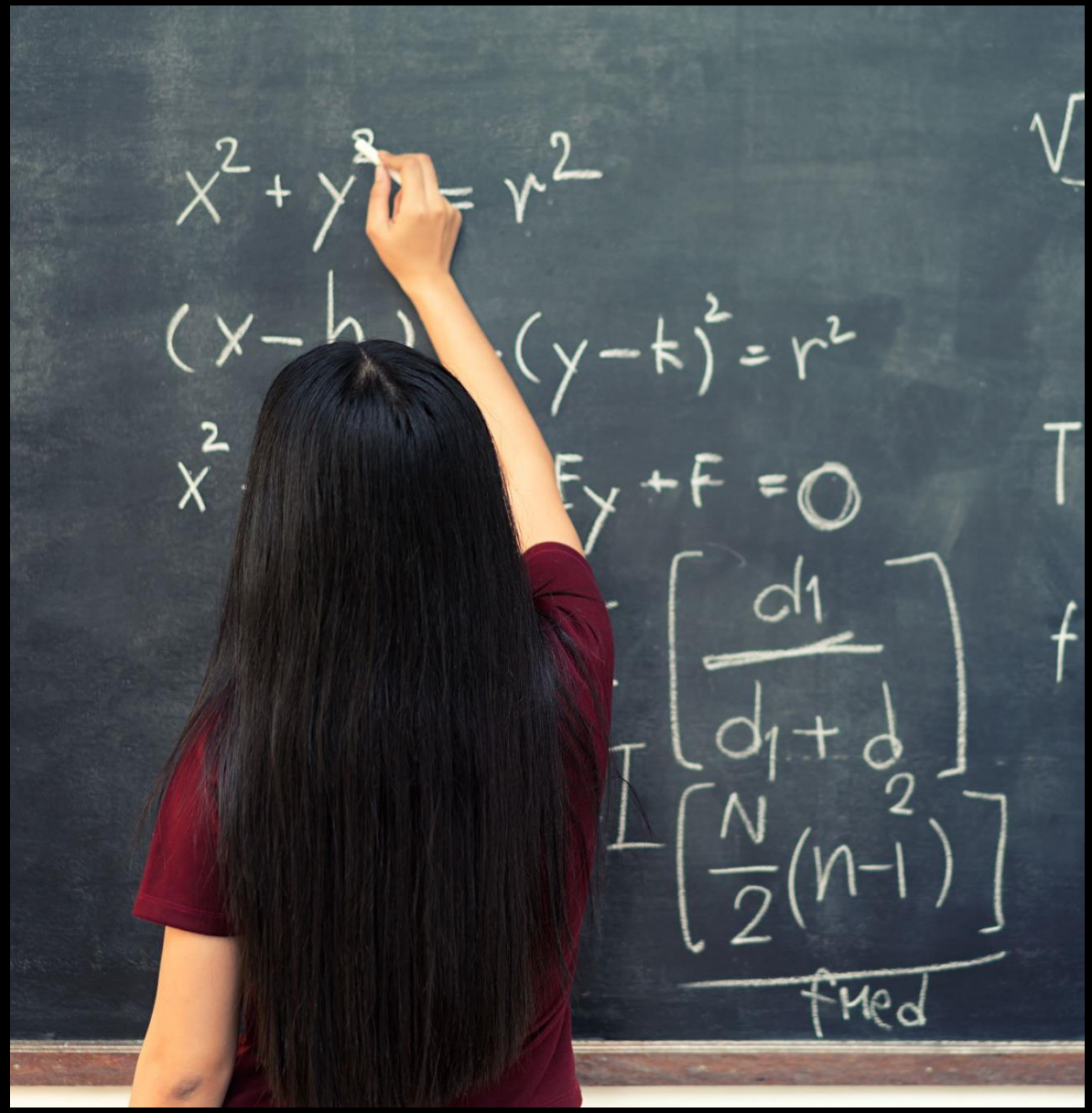
IBM Data Science

Project – Space Y

Matteo Ciccarese

Outline

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



Executive Summary

- **Summary of methodologies**
  - > Data collection
  - > Data wranling
  - > EDA: - data visualization
    - SQL
  - > Interactive map (Folium)
  - > Dashboard (Piotly Dash)
  - > Predictive analysis
- **Summary of all results**
  - > Exploratory data analysis results
  - > Interactive analytics (screenshots)
  - > Results

Introduction

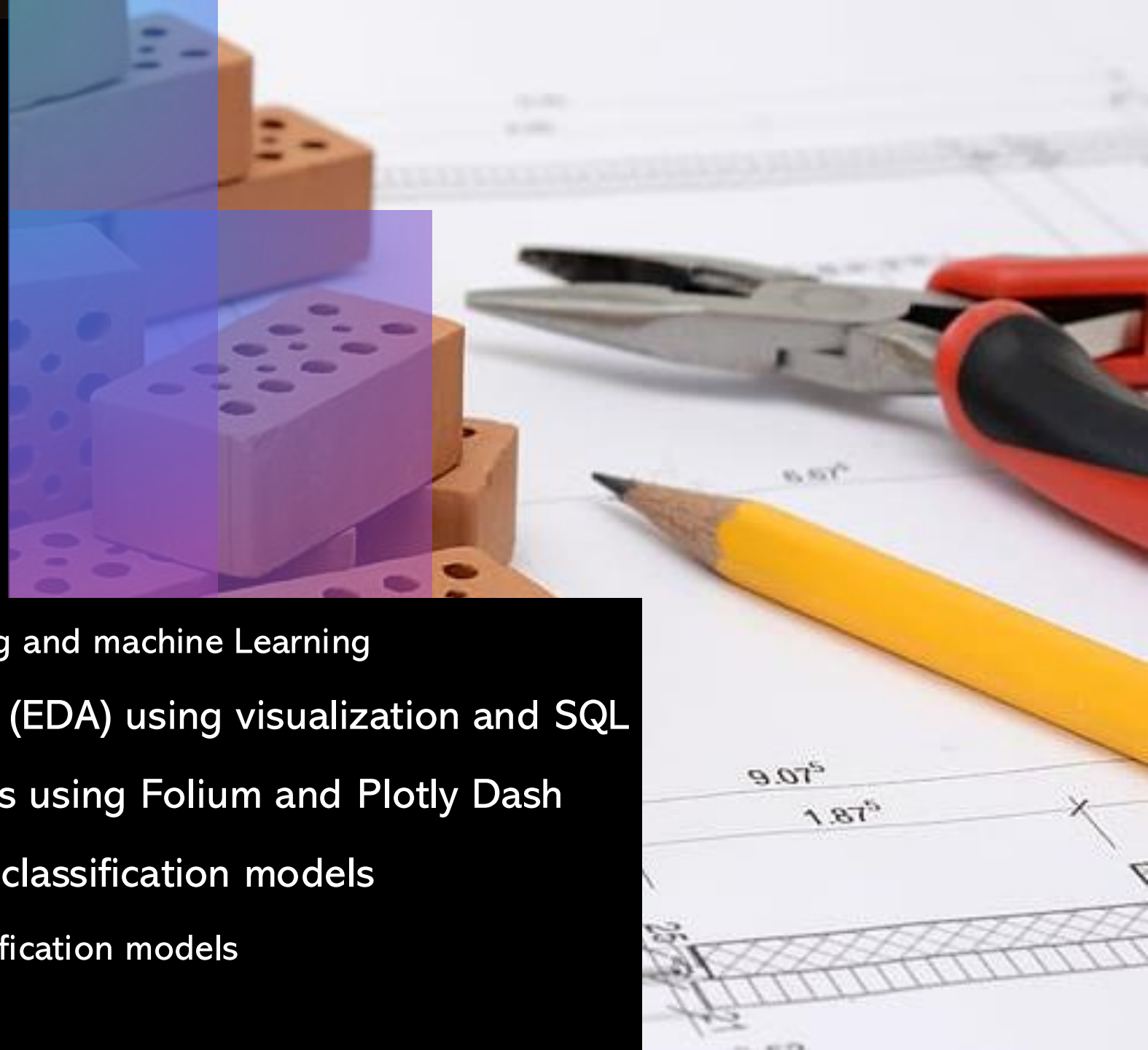
In this project, we will predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. 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.

- **What determines the success or failure of a rocket launch/landing?**
- **What interactions between variables impact the success or failure of the launch/landing?**
- **What conditions must Spase X ensure to achieve the best results in its missions?**

Methodology

Executive Summary

- Data collection methodology:
  - Space X Rest API
  - Web Scrapping (Wikipedia)
- Perform data wrangling
  - Data processed: One Hot Encoding and machine Learning
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models



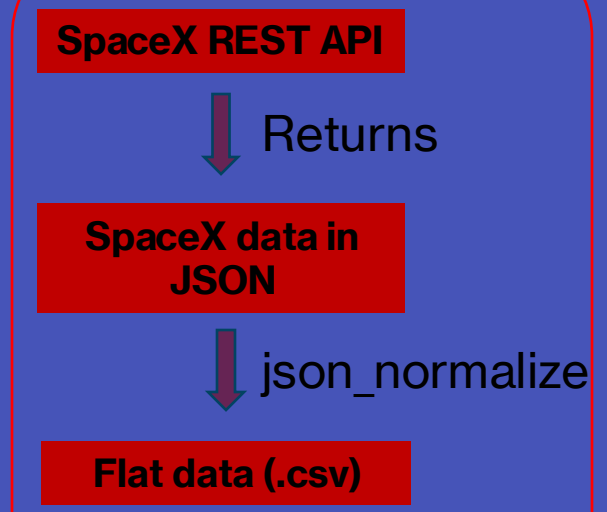
Data Collection

- **Data collection:**

- **We worked with SpaceX launch data collected by an API, specifically the SpaceX REST API. This API provided us with launch data, including:**
  - the rocket used,
  - the payload delivered,
  - the launch specifications,
  - the landing specifications,
  - the landing outcome.

- **Web scraping related Wiki pages:**
  - We used the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records.

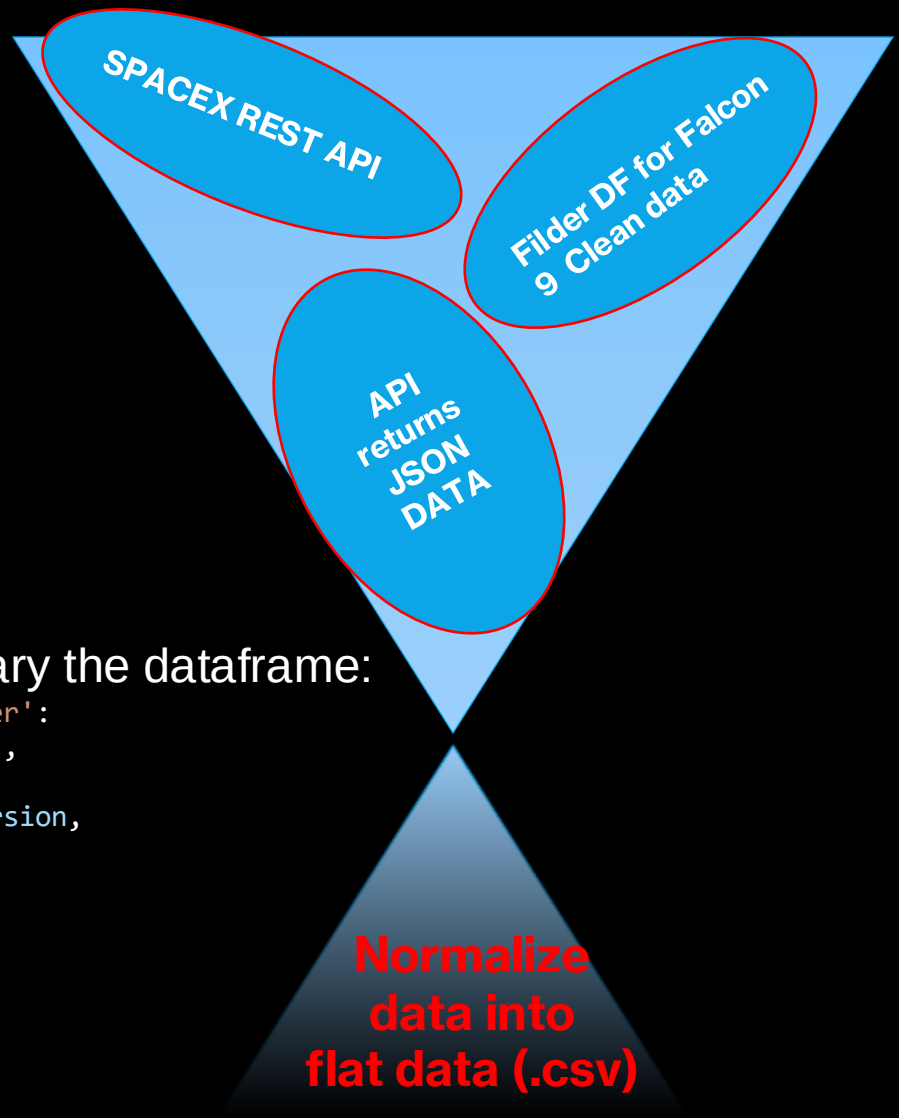
SPACE X API



- **Goal:**
  - **Use this data to predict whether SpaceX will attempt to land a rocket or not.**

WEB SCRAPPING

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 file:

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Cleaning data applying custom function:

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

4. Assign list to dictionary the 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 and export df in flat file:

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
data_falcon9.to_csv('dataset_part_1.csv', index=False
```

GitHub url

Data Collection - Scraping

1. Getting Response from HTML:

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

2. Creatibg BeautifulSoup Object:

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

3. Finding tables:

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

4. Getting columns names:

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

5. Creation 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. Append data to keys:

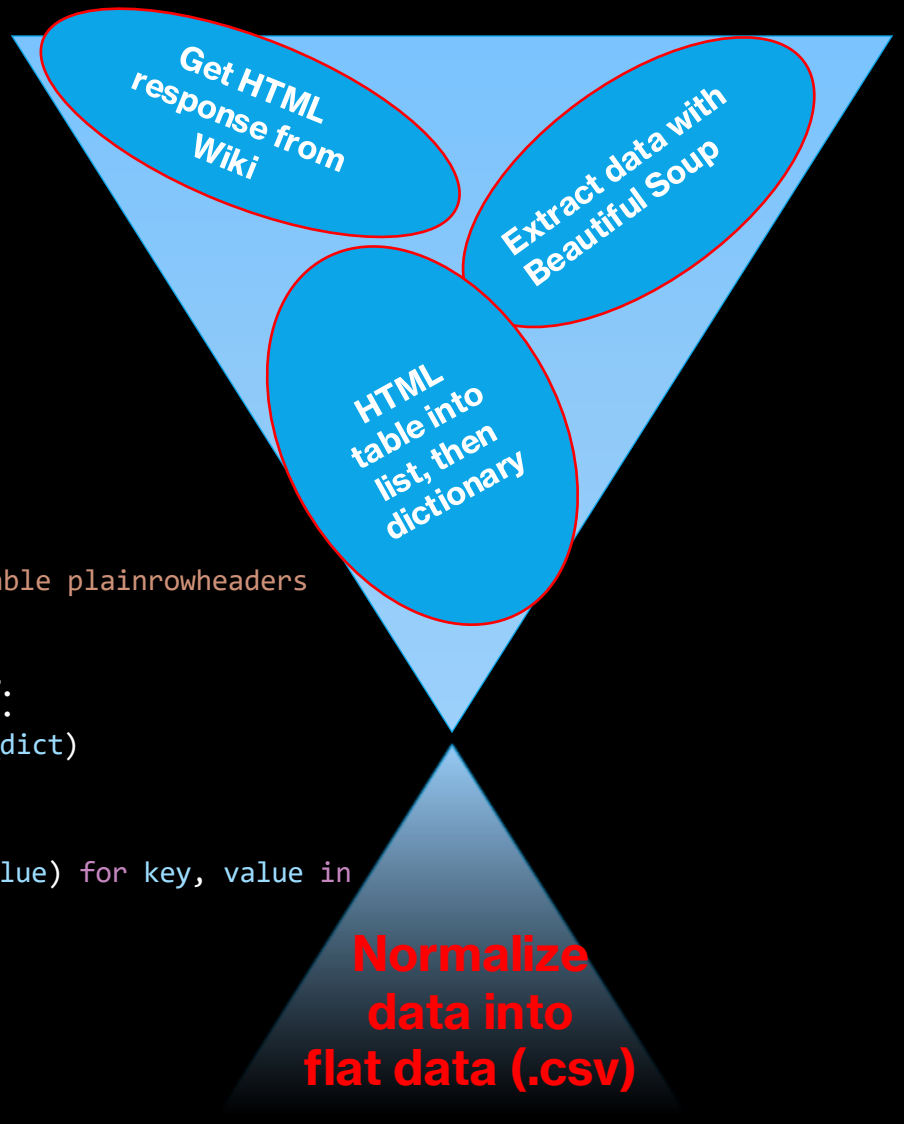
```
extracted_row = 0
#Extract each table
for table_number,table in
enumerate(soup.find_all('table',"wikitable plainrowheaders
collapsible")):…..
```

7. Converting dictionary to df:

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

8. Df to .csv:

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in
launch_dict.items() })
```



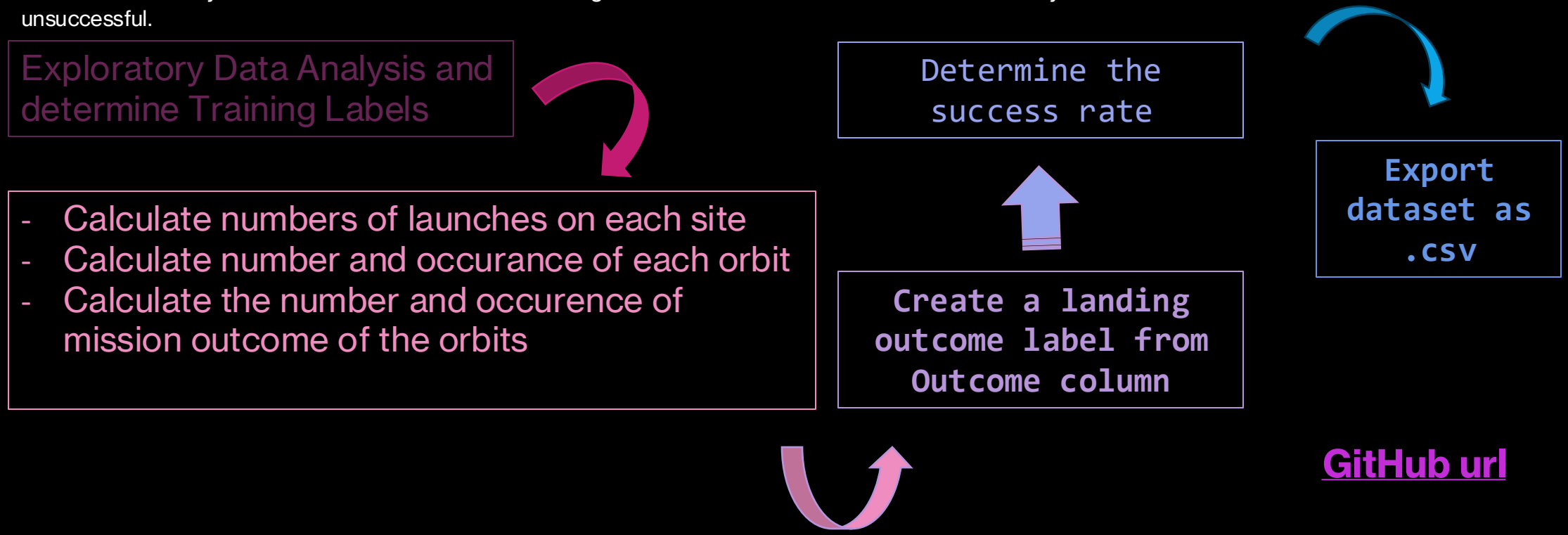
GitHub url

Data Wrangling

In this lab, we performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.

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 mainly converted those outcomes into Training Labels with `1` means the booster successfully landed `0` means it was unsuccessful.



GitHub url

EDA with Data Visualization

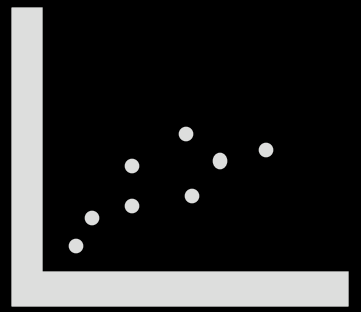
We predicted if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is due to the fact that SpaceX can reuse the first stage.
We performed Exploratory Data Analysis and Feature Engineering using `Pandas` and `Matplotlib`.

We plotted:
- Scatter Plot
- Bar Chart
- Line Plot

Scatter Plot:

A scatter plot uses dots to represent values for two different numeric variables. The position of each dot on the horizontal and vertical axis indicates values for an individual data point. Scatter plots are used to observe relationships between variables.

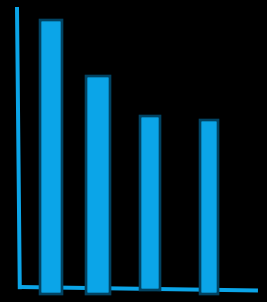
- FlightNumber vs PayloadMass
- FlightNumber vs LaunchSite
- Payload vs LaunchSite
- Orbit vs FlightNumber
- Payload vs OrbitType
- Orbit vs PayloadMass



Bar Chart:

A bar chart (aka bar graph, column chart) plots numeric values for levels of a categorical feature as bars. Levels are plotted on one chart axis, and values are plotted on the other axis. Each categorical value claims one bar, and the length of each bar corresponds to the bar's value.

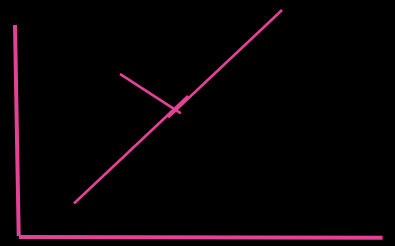
- Mean vs Orbit



Line graph:

Line graph is useful to show data variability and trends. It can help to make predictions.

- SuccesRate vs Year



GitHub url

EDA with SQL

Performing SQL queries

We ran SQL queries to answer some homework questions. Here are some homework questions that were run using SQL.

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

GitHub url

Build an Interactive Map with Folium

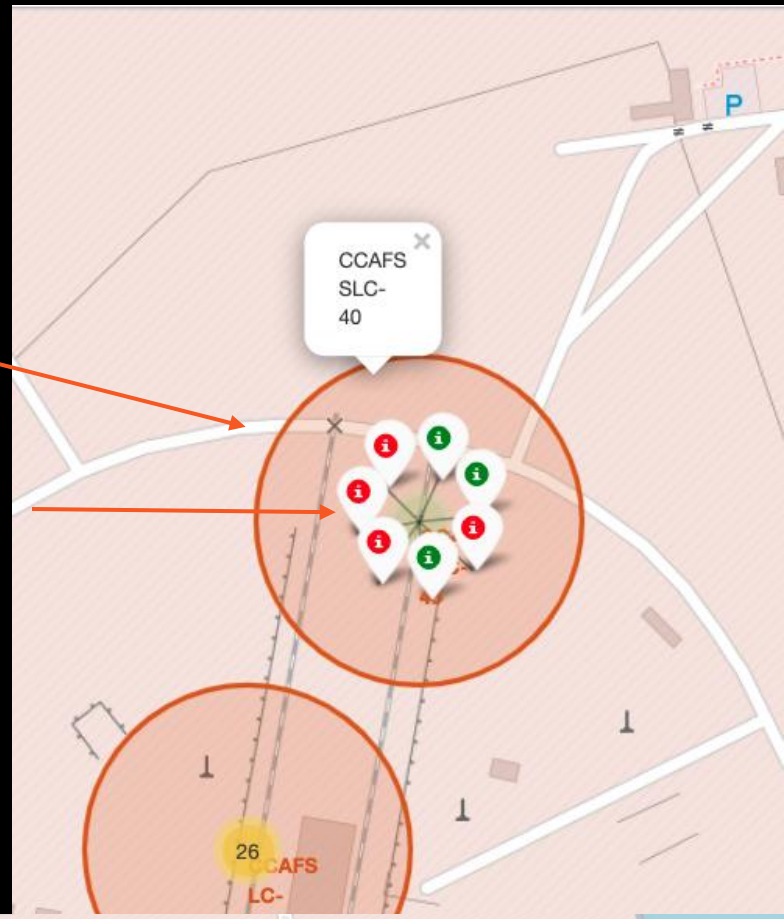
Visualizing the Launch Data into a iteractive map.

We augmented our study with a geographic visualization of the Space X project launch sites.

By entering the latitude and longitude of each launch site we marked it with an orange circle.

We then assigned the launch_outcomes(failures, successes) df to classes 0 and 1 with green and red markers on the map in MarkerCluster()

In order tu calculate the distance from LounchSite to various land marks (city, hoghways, railways, coast line) we used Haversine's formula.
A line wes designed to visualize this distance



After we plot distance lines to the proximities, we can answer the following questions easily:

- 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? no

Build a Dashboard with Plotly Dash

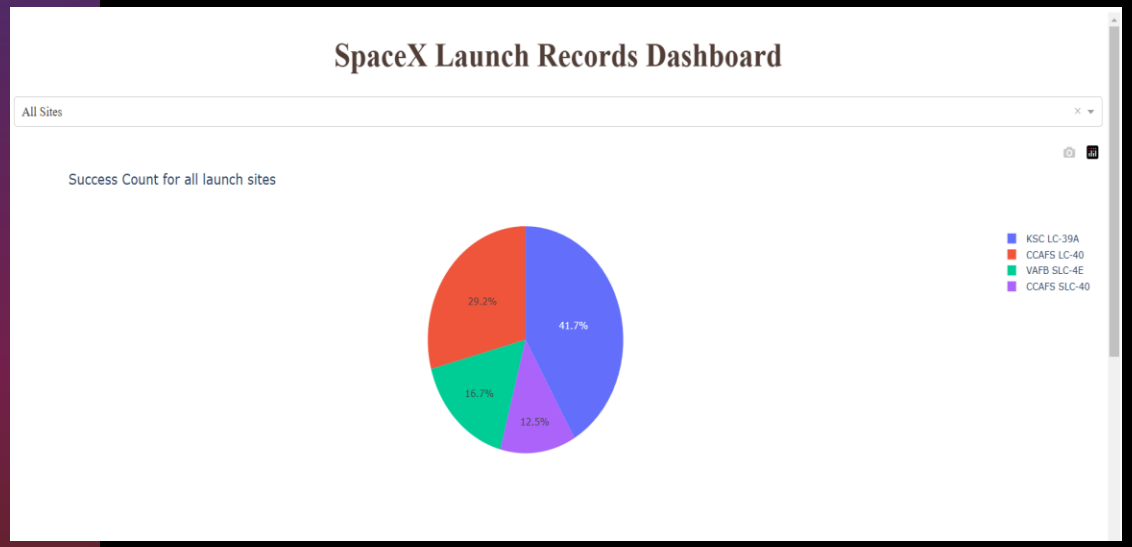
GitHub url

To ensure an iterative visualization of Space X graphs, we created an interactive dashboard.

- The dashboard was built using Flask and Dash web framework.

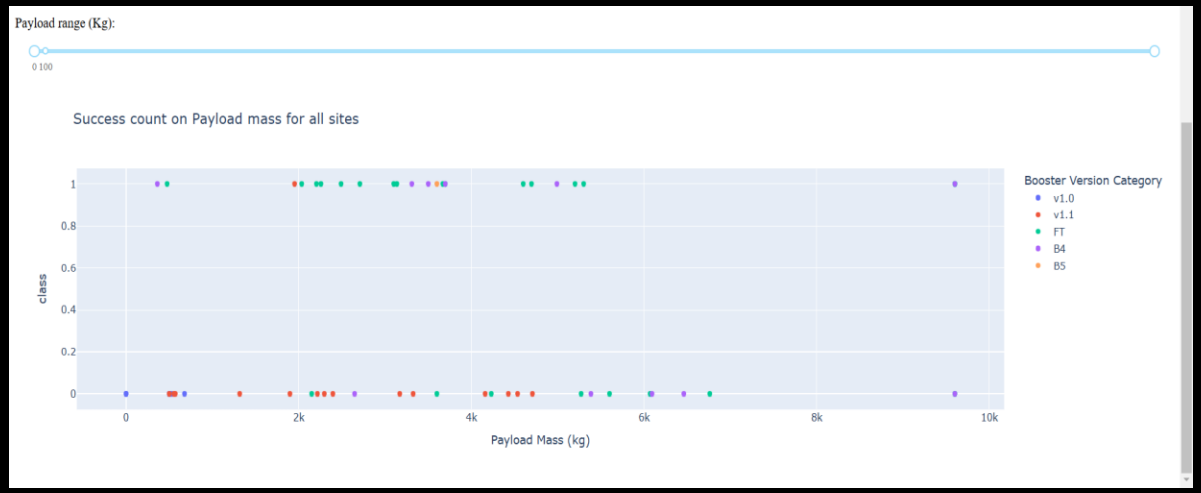
Pie chart: show the total launches by a certain site or all sites

- Was used to display relative proportions of multiple classes of data;
- Size of the circle can be made proportional to the total quantity it raresents.



Scatter Plot: show the relationship with Putcome and PayloadMass (Kg) for the different Booster Versions

- It is the best method to show a non – linear pattern
- Allows you to determine the range of data flow (max / min)



Predictive Analysis (Classification)

GitHub url

Build, evaluate, improve and find the best performing classification model

Build:

- ✓ Load dataset into NumPy and Pandas
- ✓ Trasform data
- ✓ Split data (training – test sets)
- ✓ Check test samples quantity
- ✓ Chose machine learning type
- ✓ Set parameters and algorithms to GridSeachCV
- ✓ Fit dataset in GridSearchCV object
- ✓ Train dataset

Improve:

- ✓ **Feature Engineering**
- ✓ **Algorithm Turning**

Evaluate:

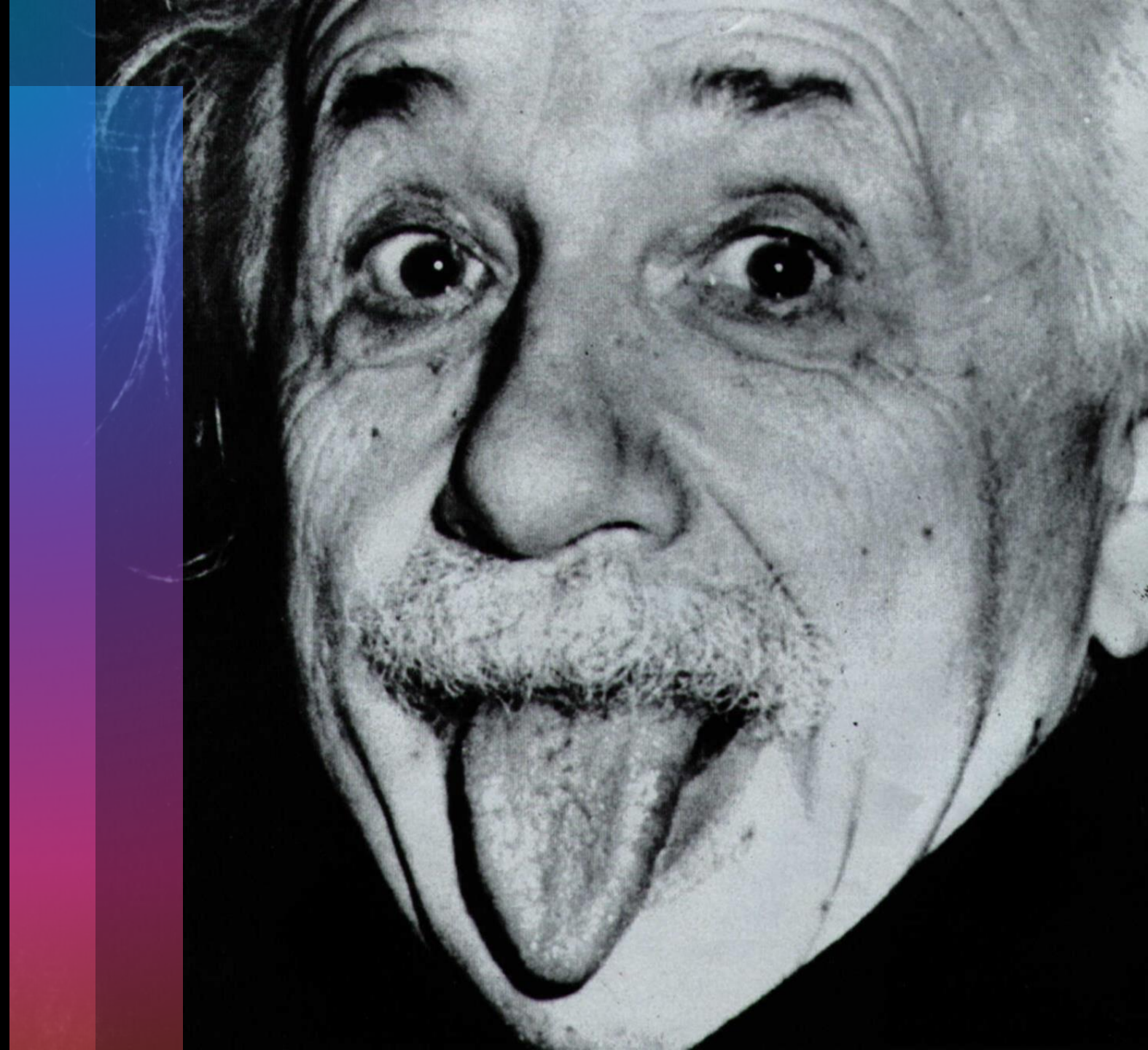
- ✓ Find accuracy for each model
- ✓ Turnhyperparameters for each type of algorithm
- ✓ Plot Confusion Matrix

Find the best:

- ✓ The model with the best accuracy score is the best performing model

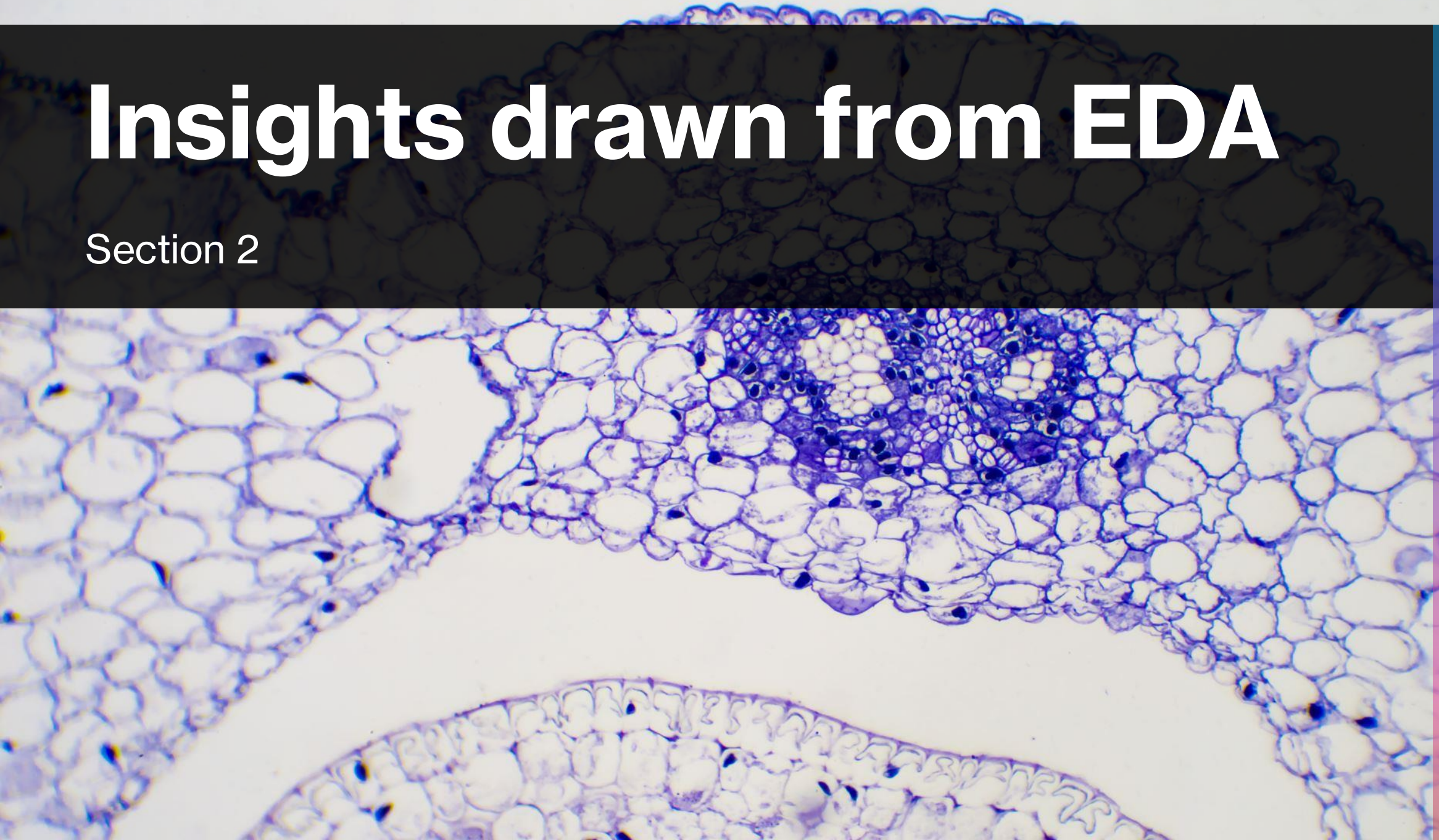
Results

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



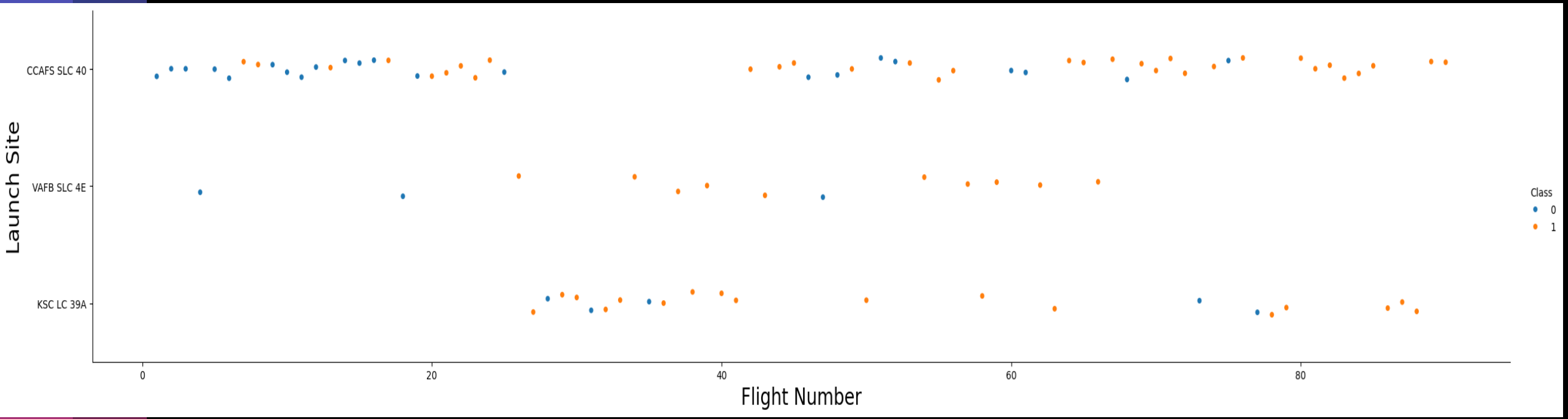
Insights drawn from EDA

Section 2



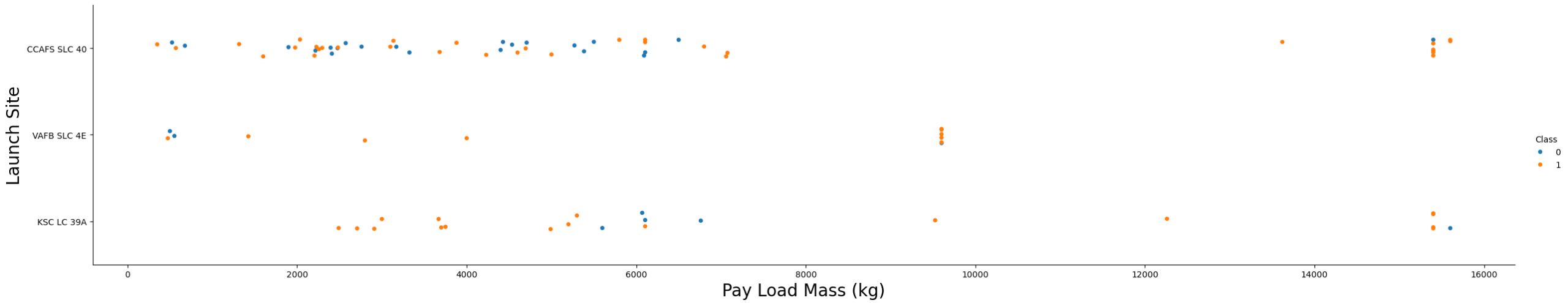
Flight Number vs. Launch Site

The graph represents the Number of Flights (x-axis) and the Launch Sites (y-axis). The greater the number of flights at the launch site, the greater the success of the launch site



Payload vs. Launch Site

The graph represents the Payload Mass (x-axis) and the Launch Sites (y-axis). The higher the payload mass for the launch site, the higher the success rate of the rocket.

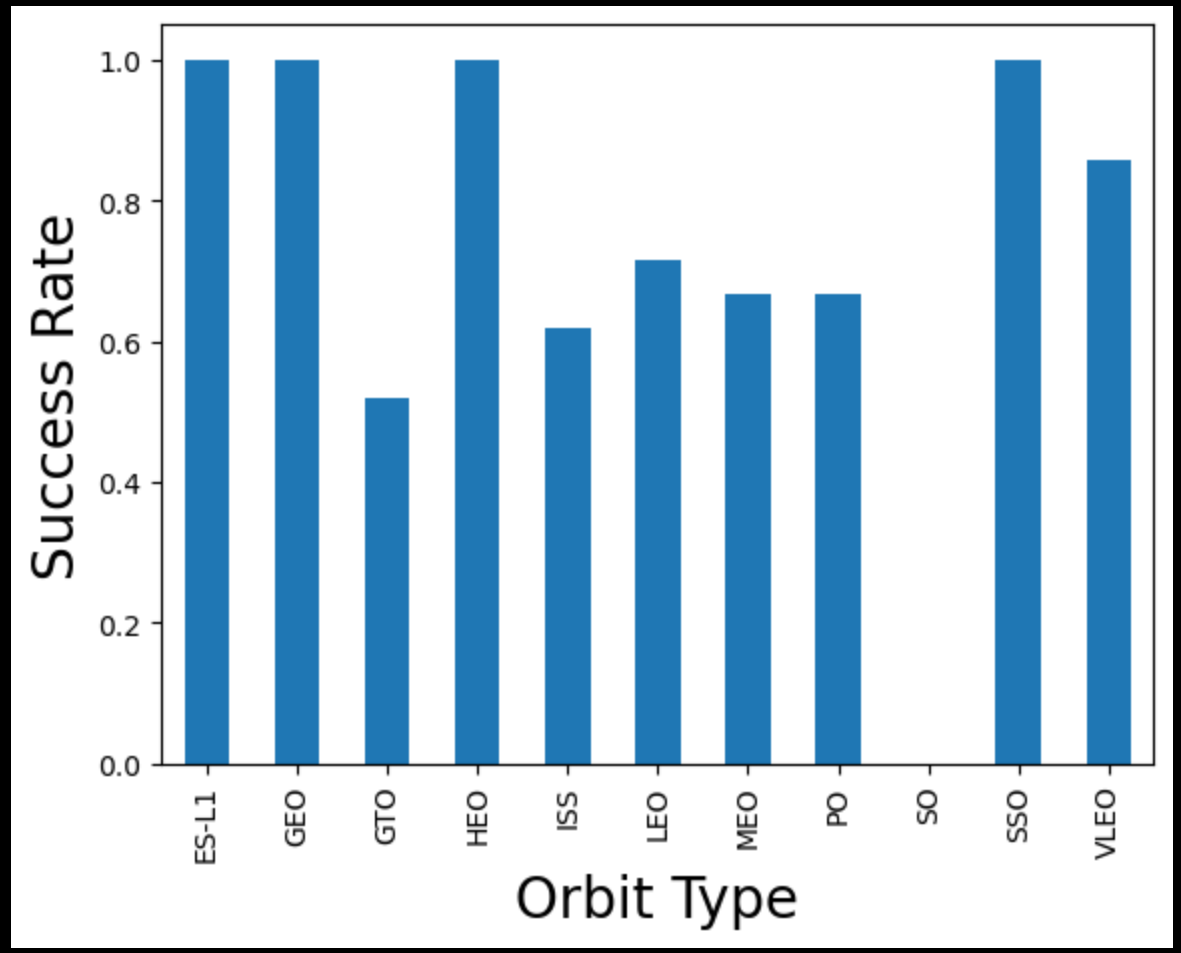


However, this visualization does not provide a clear pattern on the dependence of the variables

Success Rate vs. Orbit Type

The graph represents the Orbit Type (x-axis) and the Success Rate (y-axis).
The graph shows a clear, higher degree of success in the orbits:

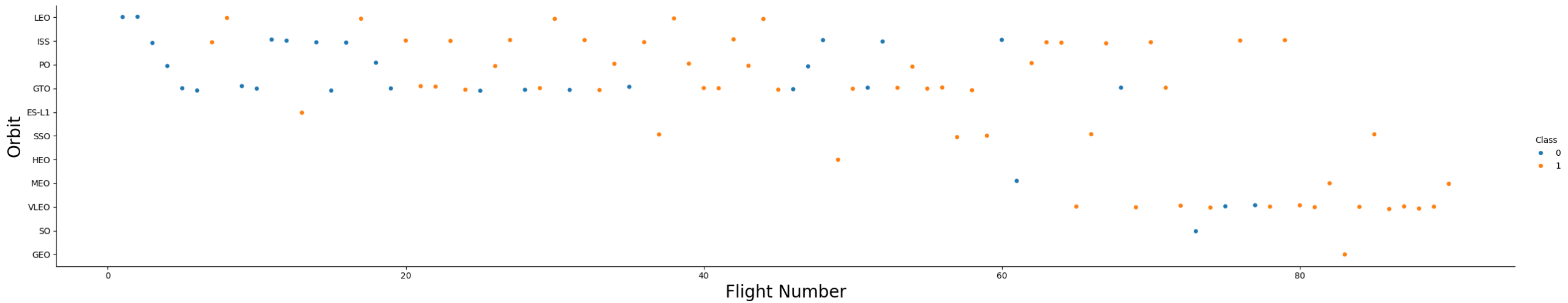
- ES-L1
- GEO
- HEO
- SSO



Flight Number vs. Orbit Type

The graph represents the Flight Number (x-axis) and the Orbit Type (y-axis).

The graph shows how in the LEO orbit after the first failures all the launches were successful, this implies a relationship between the Number of Launches and this orbit.

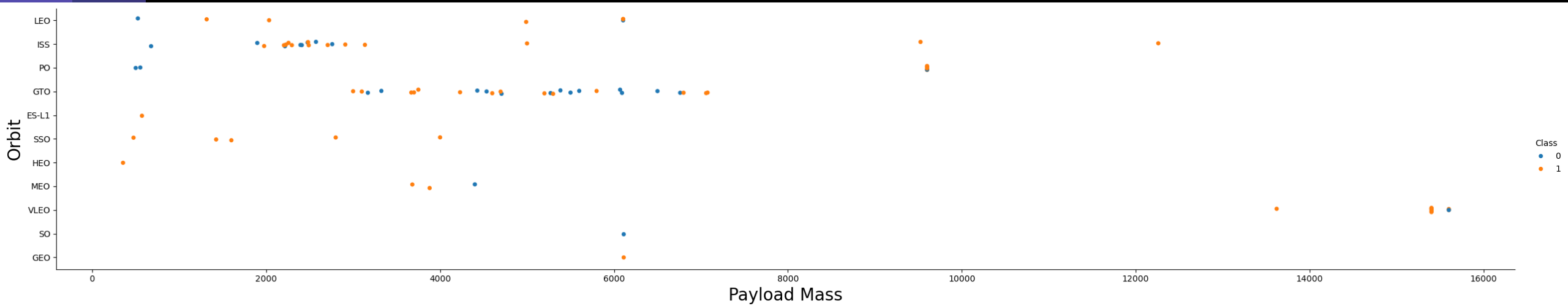


However this relationship is weak or absent in the other launch sites.

Payload vs. Orbit Type

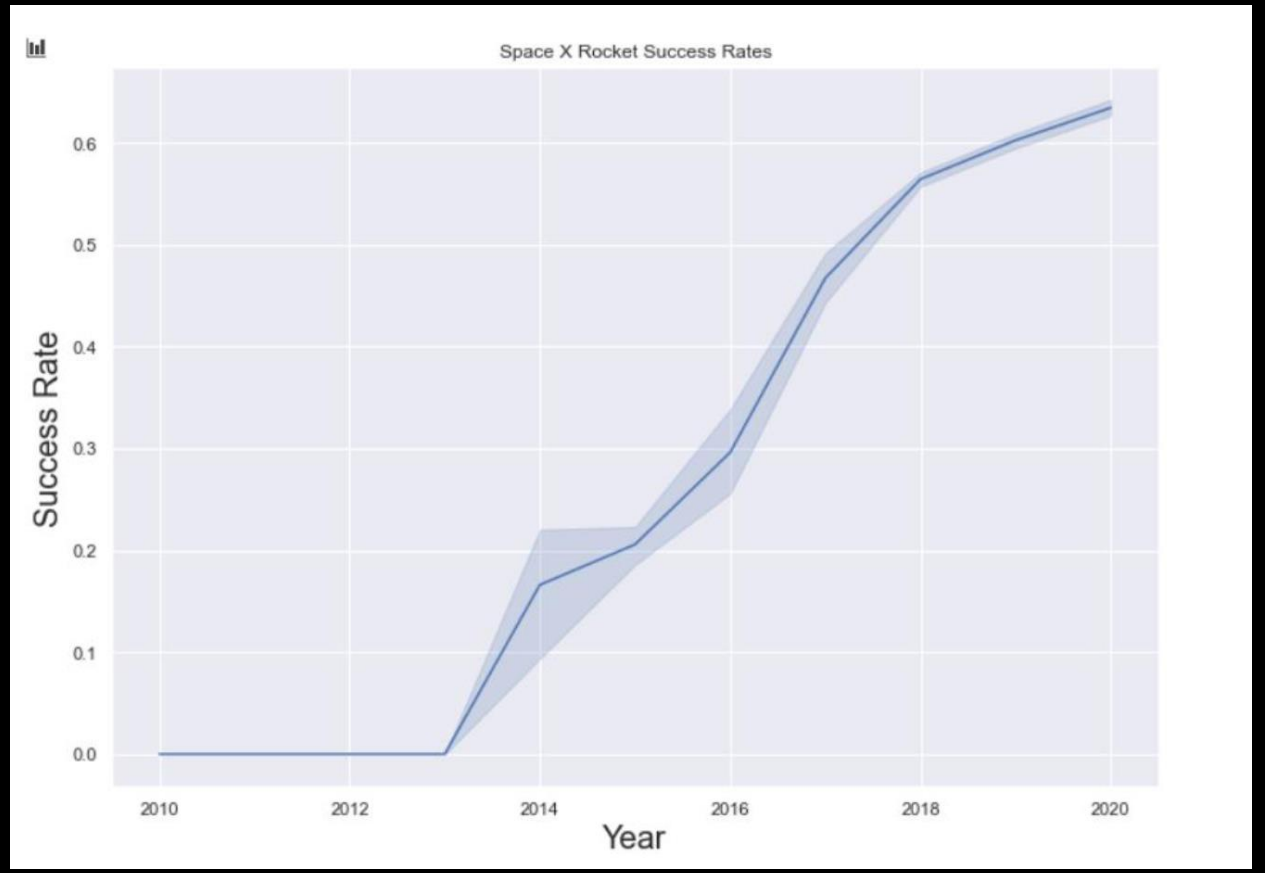
The graph represents the Payload Mass (x-axis) and the Orbit Type (y-axis).

The graph shows a negative influence of Payload on Orbyts VLEO, MEO and vice versa it seems to positively influence the orbits LEO



Launch Success Yearly Trend

The graph shows the success yearly trand.
Since 2013 it kept increasing till 2020.



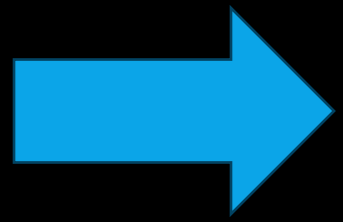
EDA with .SQL

All Launch Site Names

Use ' DISTINCT' in the query to show ***Unique*** values in the ***Lounch_Site*** column from SPACEXTBL table

SQL query

```
SELECT DISTINCT Launch_Site
FROM SPACEXTBL;
```



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

Launch Site Names Begin with 'CCA'

TASK: Find 5 records where launch sites begin with `CCA'`

SQL query:

SELECT * FROM SPACEXTBL **where** Launch_site **like** 'CCA%' **LIMIT** 5;

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

Use LIKE keyword has a wild card with the words ' the percentage in the end suggests that the Launch_Site name must start with CCA.
Use the LIMIT 5 to show only 5 records from SPACIALXTBL table

Total Payload Mass

TASK: Display the total payload mass carried by boosters launched by NASA (CRS)

SQL query:

```
SELECT SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTBL WHERE Customer
LIKE 'NASA (CRS)';
```

SUM(PAYLOAD_MASS__KG_)
45596

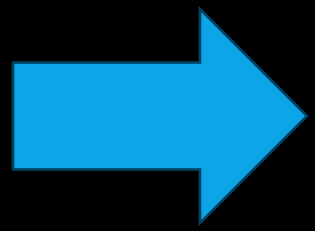
Use the function SUM summates the total in the column PAYLOAD_MASS_KG_
The WHERE clause filters the dataset to only perform calculations on Customer NASA (CRS)

Average Payload Mass by F9 v1.1

TASK: Display average payload mass carried by booster version F9 v1.1

SQL query:

```
SELECT AVG(PAYLOAD_MASS__KG_)
FROM SPACEXTBL WHERE
Booster_Version LIKE 'F9
v1.1' ;
```



AVG(PAYLOAD_MASS__KG_)
2928.4

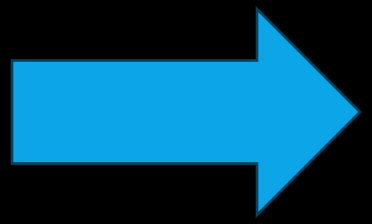
Use the function AVG works out the average in the column PAYLOAD_MASS_KG_
The WHERE clause filters the dataset to only perform calculations on Booster_version F9 v1.1

First Successful Ground Landing Date

TASK: List the date when the first succesful landing outcome in ground pad was acheived.

SQL query:

```
SELECT min(Date) from
SPACEXTBL where
"Landing_Outcome" = 'Success
(ground pad)';
```



min(Date)
2015-12-22

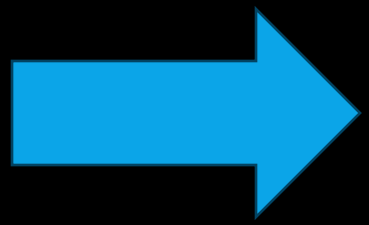
Use the function MIN works out the minimum date in thecolumn Date
The WHERE clause filters the dataset to only perform calculations on Landing_Outcome Success (ground pad)

Successful Drone Ship Landing with Payload between 4000 and 6000

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

SQL query:

```
SELECT DISTINCT BOOSTER_VERSION
FROM SPACEXTBL WHERE
PAYLOAD_MASS__KG_ BETWEEN 4000
AND 6000 AND LANDING_OUTCOME =
'Success (drone ship)';
```



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

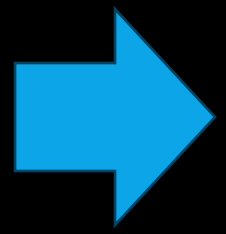
Use ' DISTINCT' in the query to show ***Unique*** values in the Booster_version
The WHERE clause filters the dataset toPayloadMass > 4000 AND Payload_MASS_KG_ < 6000
More we add the clause Success

Total Number of Successful and Failure Mission Outcomes

TASK: List the total number of successful and failure mission outcomes

SQL query:

```
select Mission_Outcome,
COUNT(*) from SPACEXTBL
GROUP BY
Mission_Outcome;
```



Mission_Outcome	COUNT(*)
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Use 'GROUP BY' clause groups rows based on the values in a specified column

Boosters Carried Maximum Payload

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

SQL query:

```
SELECT DISTINCT BOOSTER_VERSION
FROM SPACEXTBL WHERE
PAYLOAD_MASS__KG_ = (SELECT
MAX(PAYLOAD_MASS__KG_) FROM
SPACEXTBL) ORDER BY
BOOSTER_VERSION;
```

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

Use 'DISTINCT' in the query means that it will only show Unique values in the Booster_Version column from SPACEXTBL table.
The 'MAX' command returns the maximum (highest) value in a specified column.
The 'ORDER BY' clause is used to sort the result set in ascending or descending order based on a specified column.

2015 Launch Records

TASK: List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Landing_Outcome	substr(Date,1,4)	substr(Date,6,2)	Booster_Version	Launch_Site
Failure (drone ship)	2015	01	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	2015	04	F9 v1.1 B1015	CCAFS LC-40

SQL query:

```
select "Landing_Outcome",
substr(Date,1,4),
substr(Date,6,2),
"Booster_Version", "Launch_Site"
from SPACEXTABLE where
"Landing_Outcome" = "Failure
(drone ship)" and
substr(Date,1,4)="2015";
```

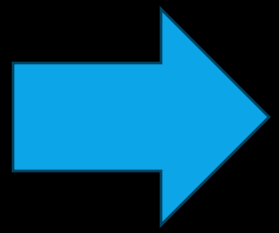
This query is long but simple, I selected several variables from the table and inserted a 'WHERE' condition

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

TASK: 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 query:

```
SELECT "Landing_Outcome",
COUNT("Landing_Outcome") AS
COUNTS FROM SPACEXTBL WHERE DATE
BETWEEN '2010-06-04' AND '2017-
03-20' AND "Landing_Outcome"
LIKE 'Success%' GROUP BY
"Landing_Outcome"
```



Landing_Outcome	COUNTS
Success (drone ship)	5
Success (ground pad)	3

The COUNT command counts the number of rows or non-null values in a specified column.

Launch Sites Proximites Analysis

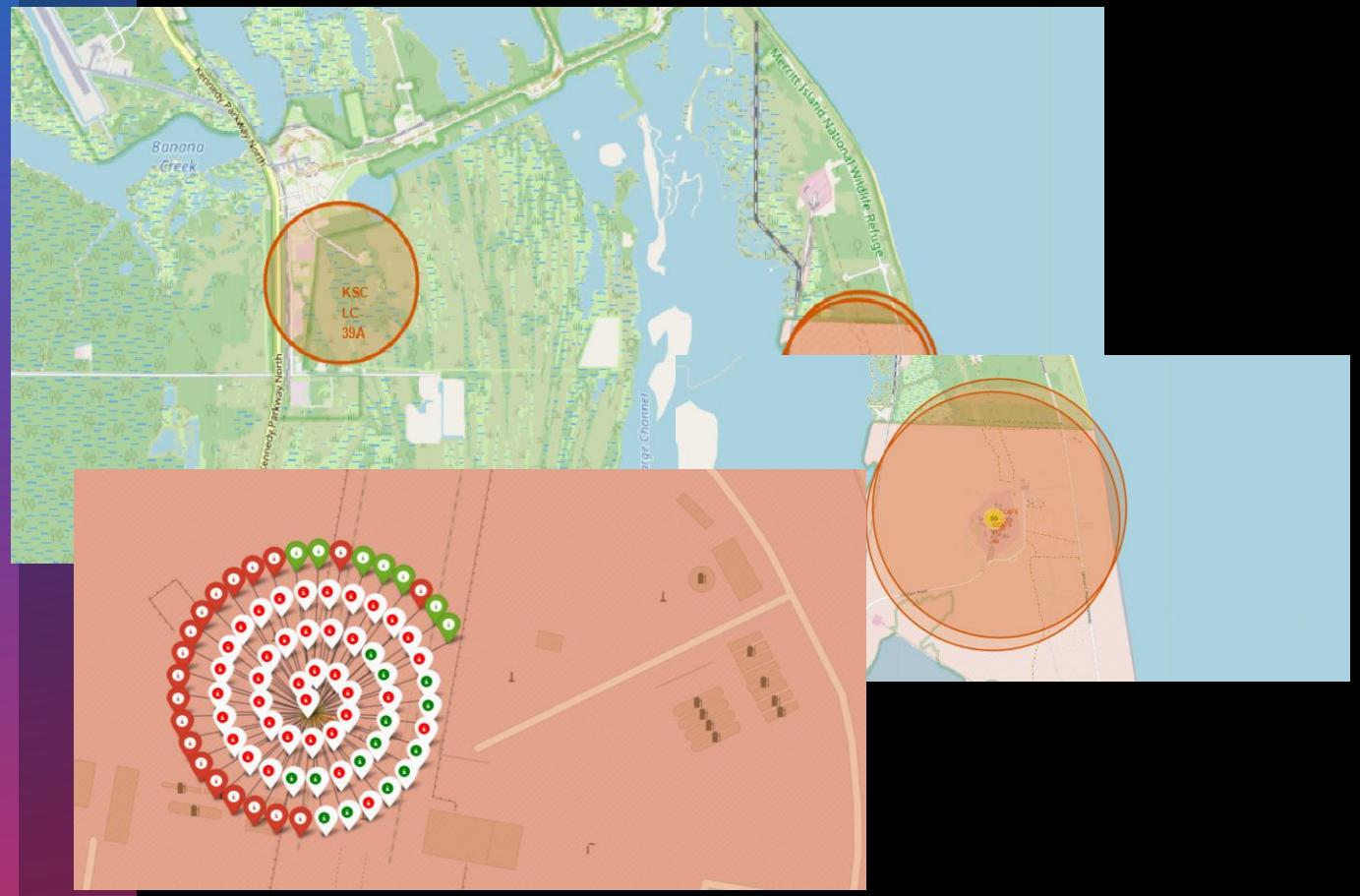
Section 3

All launch sites global map markers

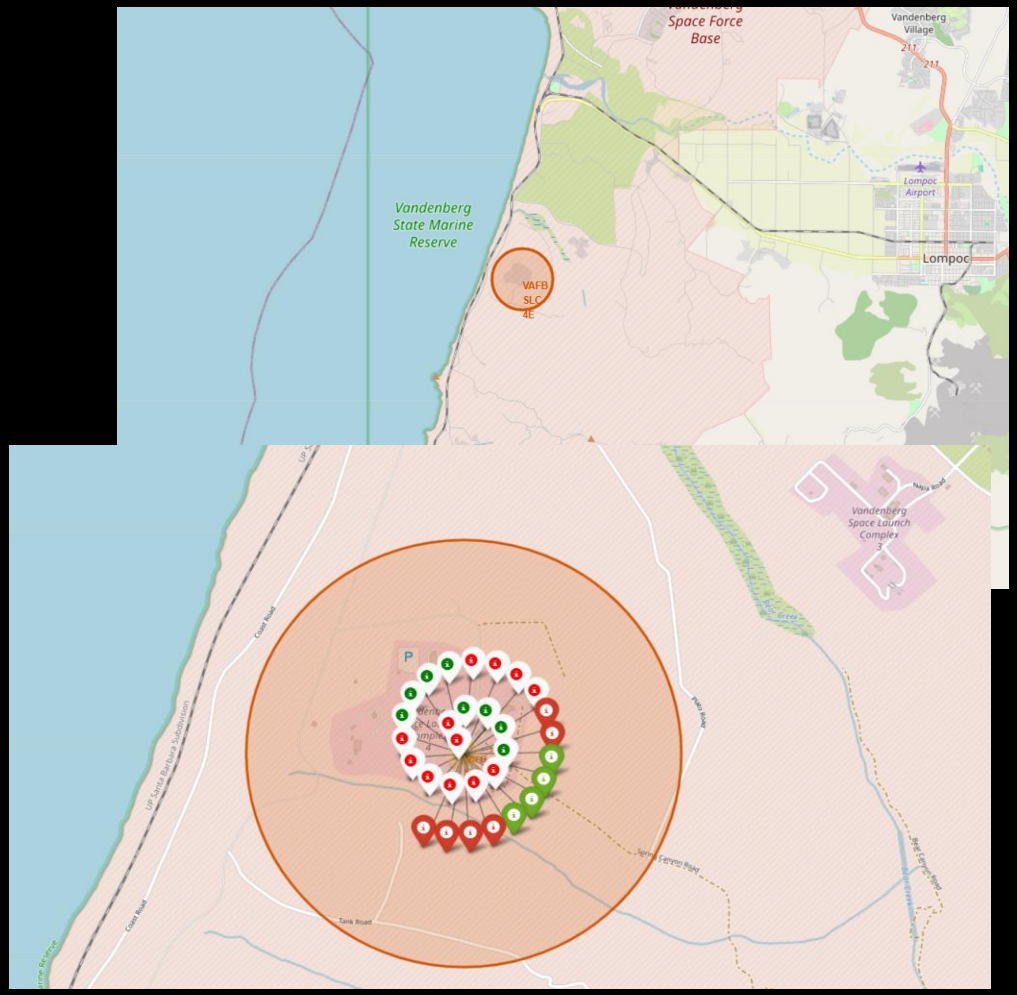
We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

Launch sites successful / Failures

Florida Launch Sites



California Launch Sites

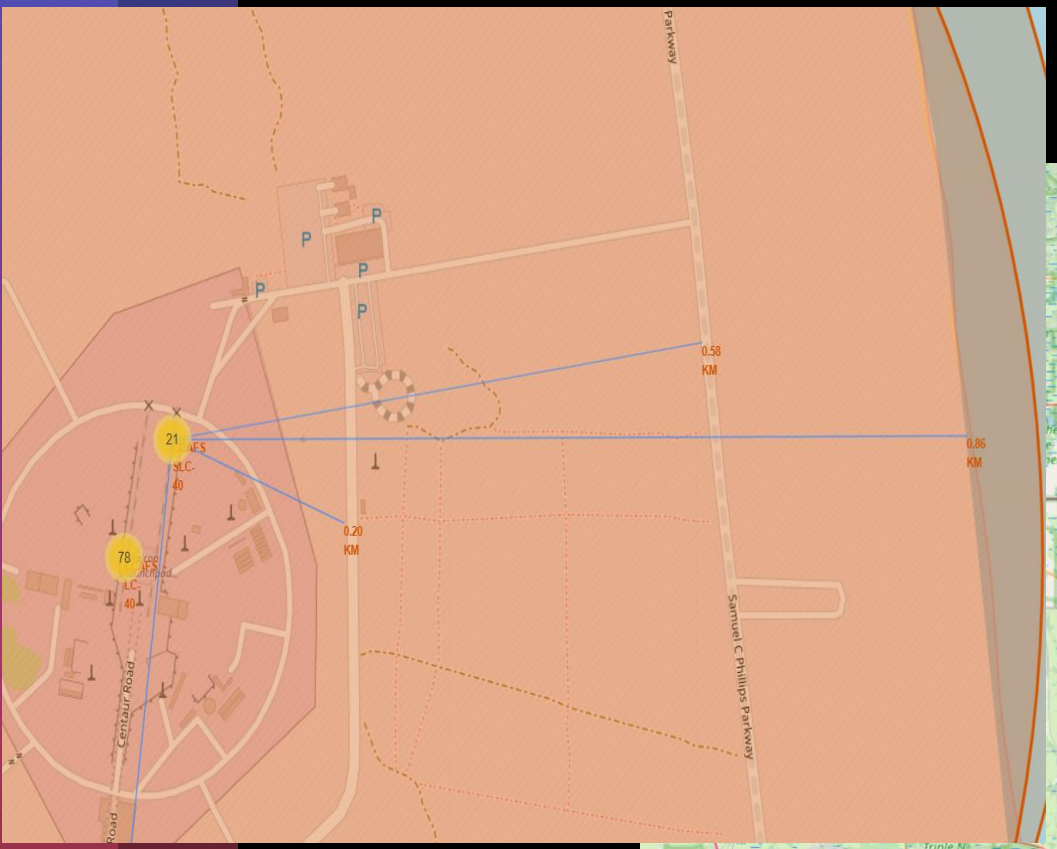


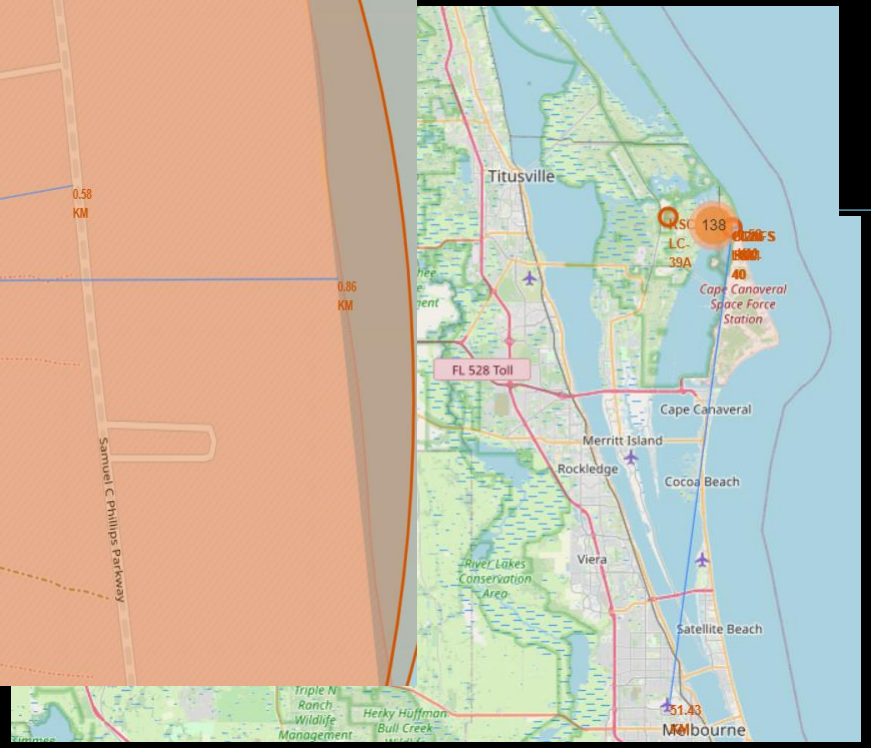
Green Marker shows successful Launches and Red Marker shows Failures

Distance from launch site to railway, highway, coast and nearest city

We calculated the distances between the launch site and the airport, the highway, the coast and the nearest city. The coordinates of the points were identified using the visualization and the distances calculated using the Python function reported.

Blue lines indicate the distances on the map.

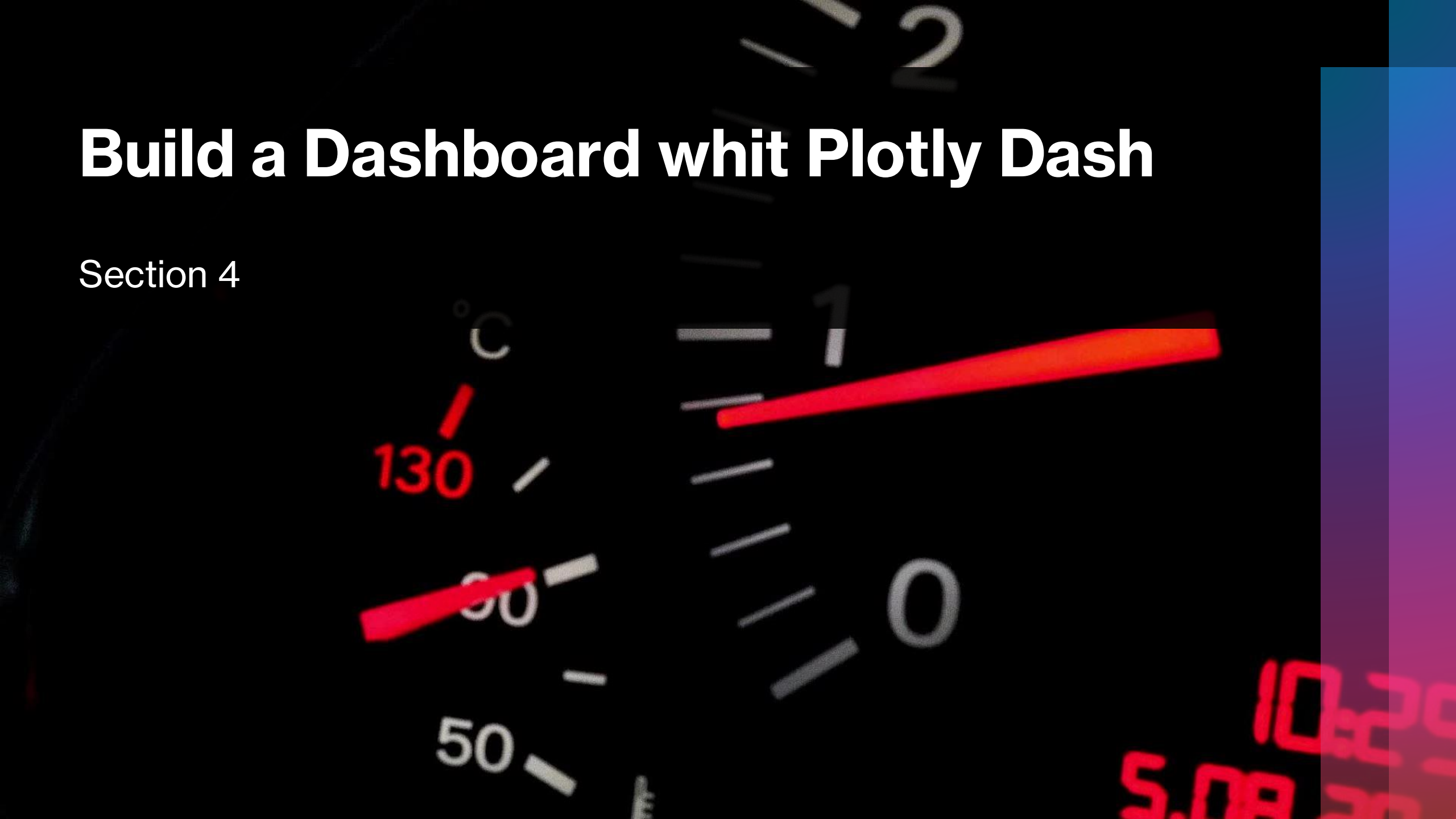





```python
from math import sin, cos, sqrt, atan2, radians

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

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

    dlon = lon2 - lon1
    dlat = lat2 - lat1

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

    distance = R * c
    return distance
```

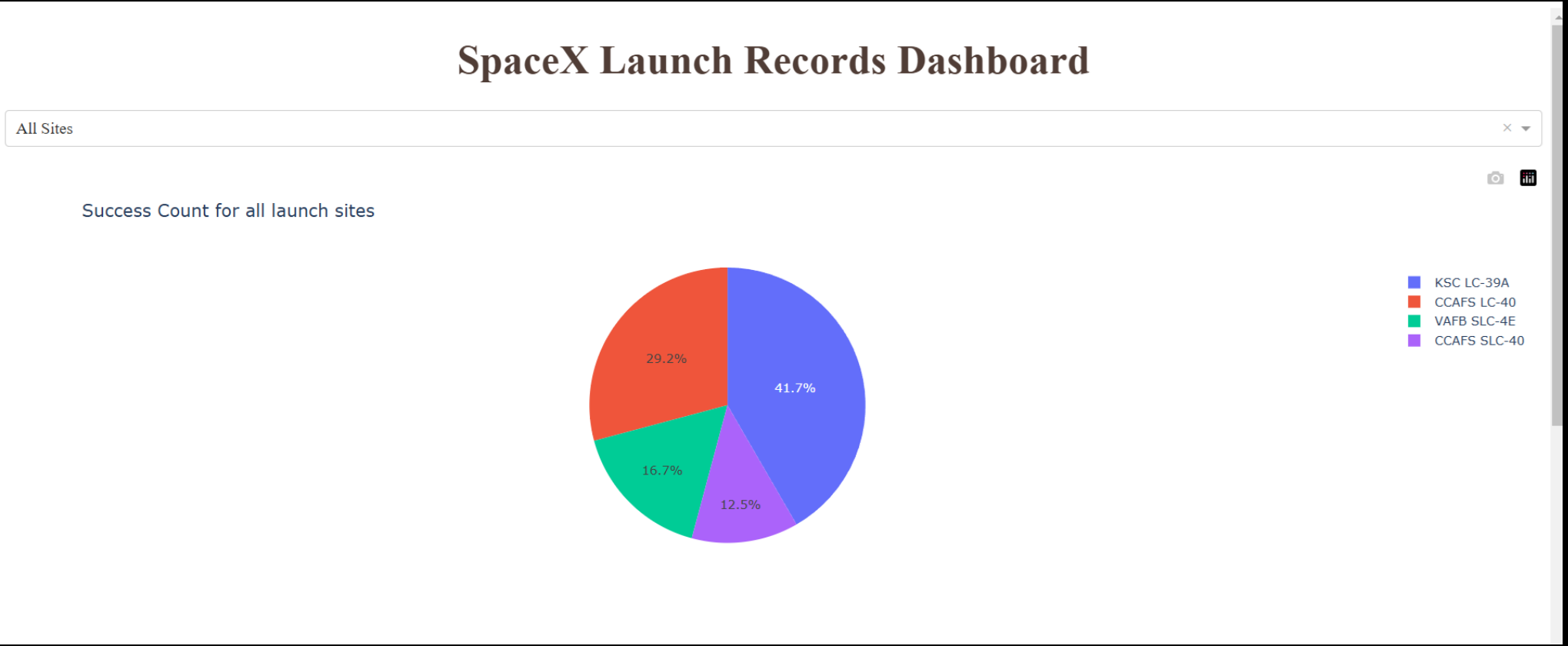
Build a Dashboard whit Plotly Dash

Section 4



Pie chart: launch success count for all sites

The graph represents the percentage of launch success for each launch site. We can see that the most successful launch site is KSC LC 39A

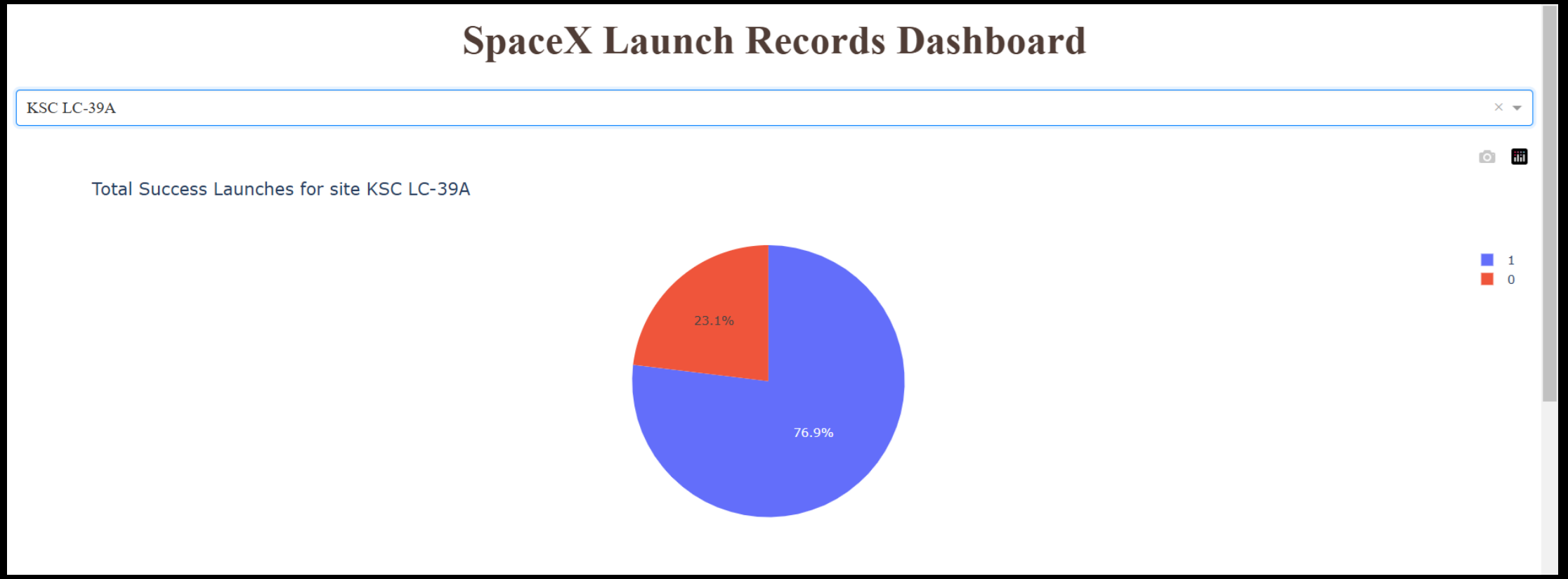


Piechart for the launch site with highest launch success ratio

The graph shows the success and failure rates of the KSC LC 39A launch site in detail.

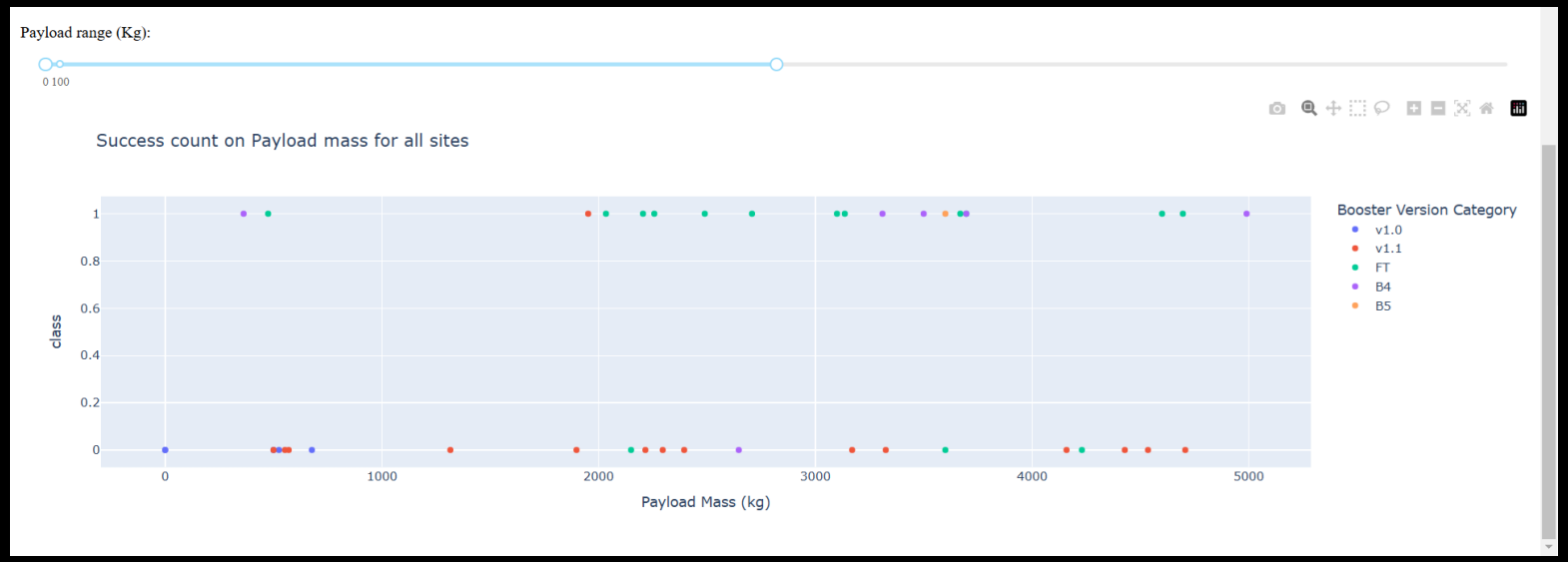
1 = success

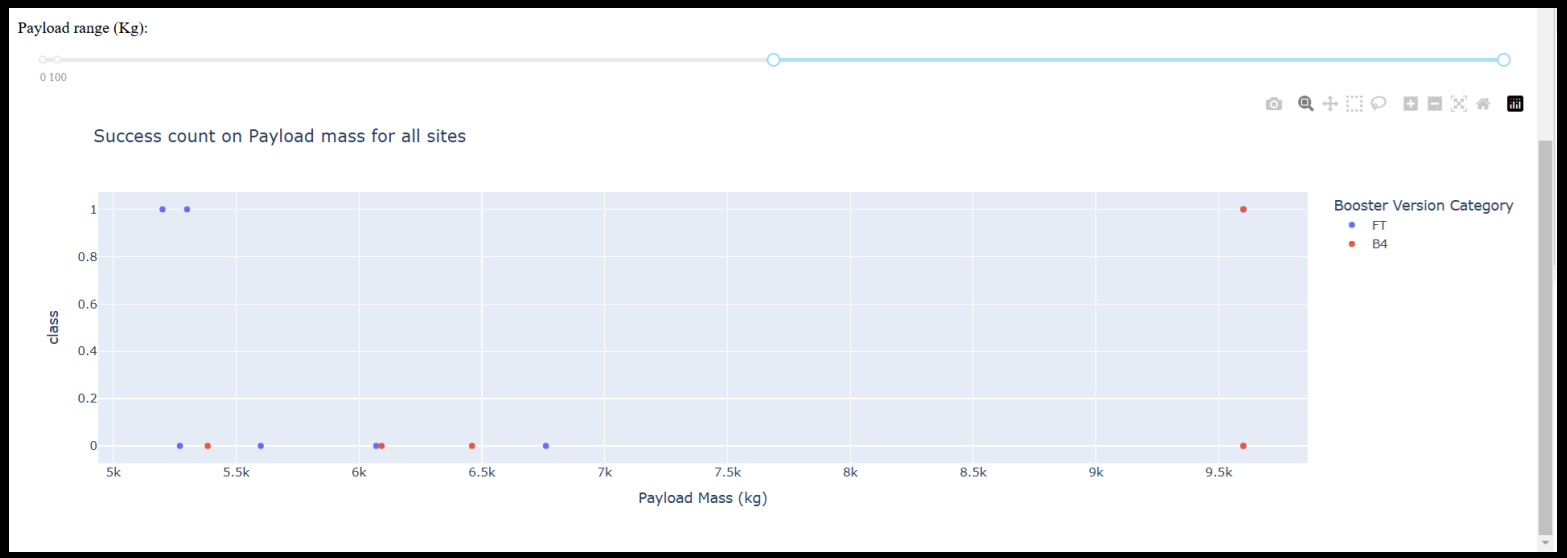
0 = Fail



Scatter plot Payload vs. Launch Outcome

The graph shows payload versus launch result for all sites, with different payloads selected in the 0 - 4000 and 5000 - 10000 range slider.



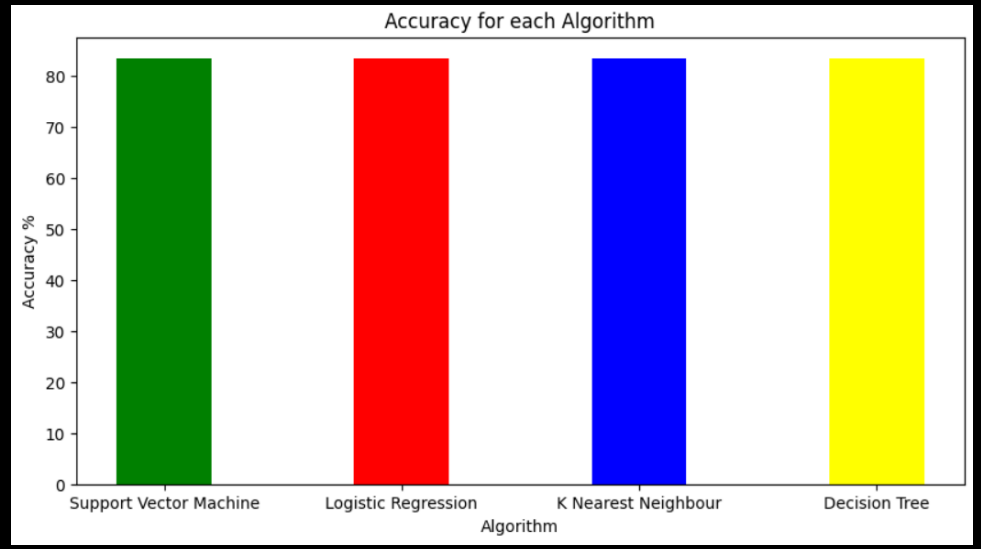


Predictive Analysis (Classification)

Section 5

Classification Accuracy

The accuracy was calculated for each model, and represented in the graph. In the table we can see how each model has an excellent level of accuracy

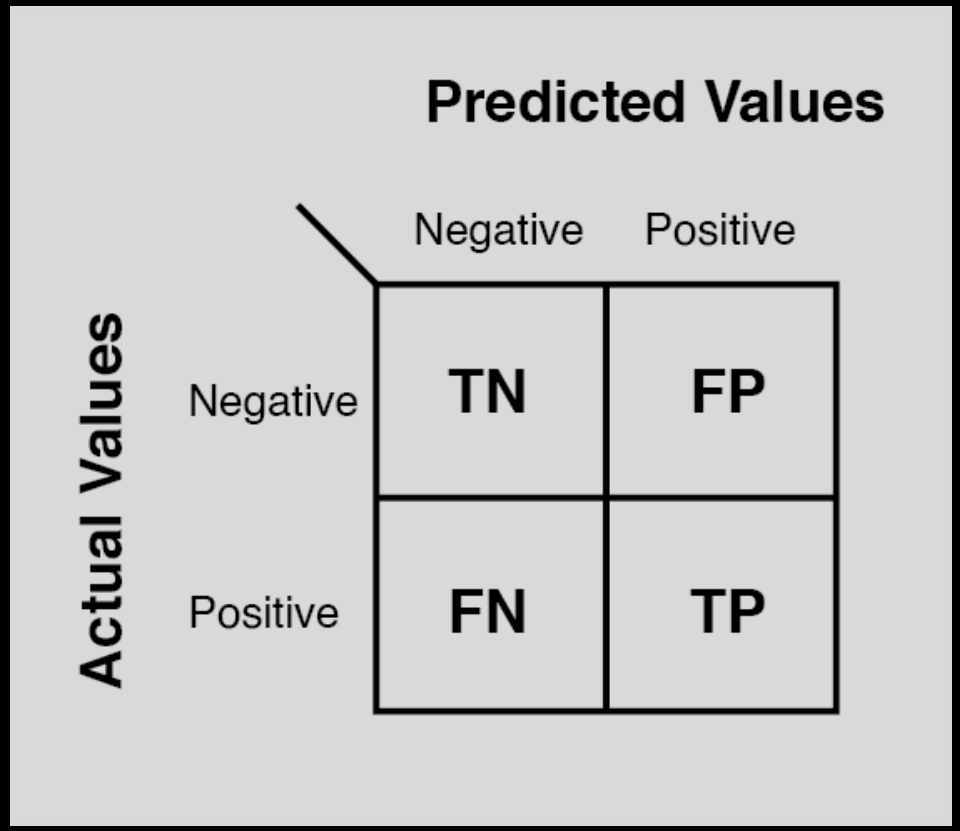


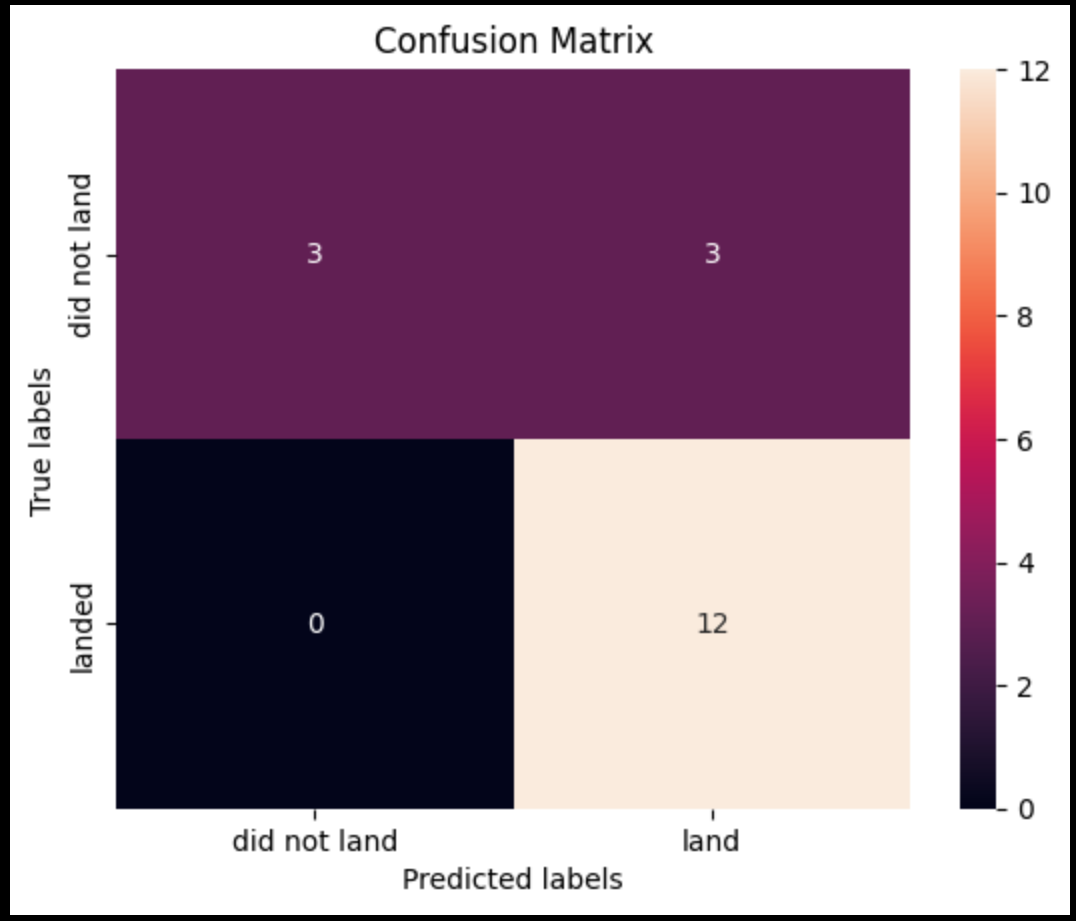

```
accuracy = [svm_cv_score, logreg_score, knn_cv_score, tree_cv_score]
accuracy = [i * 100 for i in accuracy]

method = ['Support Vector Machine', 'Logistic Regression', 'K Nearest
Neighbour', 'Decision Tree']
models = {'ML Method':method, 'Accuracy Score (%)':accuracy}

ML_df = pd.DataFrame(models)
ML_df
```

ML Method	Accuracy Score (%)
Support Vector Machine	83.333333
Logistic Regression	83.333333
K Nearest Neighbour	83.333333
Decision Tree	83.333333

Confusion Matrix

Here is an analysis of the confusion matrix with an explanatory image





Conclusions

1. The results of our analysis underline how the main variables in the success or failure of a launch are represented by the Orbit, Booster Version, Launch Site;

2. Low weighted payloads perform better than the heavier payloads;

3. KSC LC 39A is the launch site with the most success;

4. Orbit ES-L1 , GEO , HEO , SSO has the best Success Rate;

5. The success rates of SpaceX launches are directly proportional to time, so every year the successes increase;

Thank you