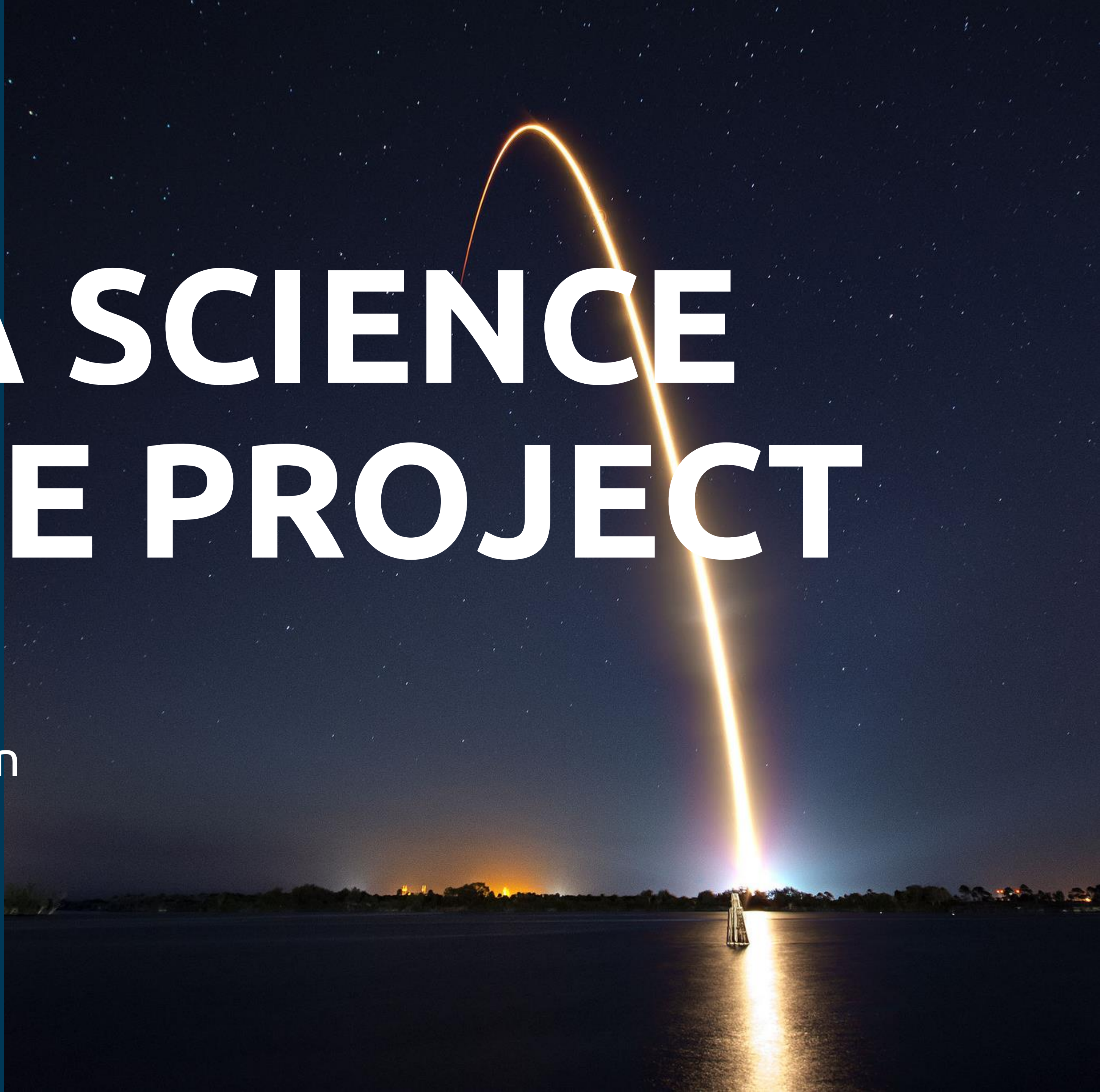
IBM DATA SCIENCE CAPSTONE PROJECT

Space X Falcon 9 Landing Prediction

William Larsson Krighom



OUTLINE

1) Executive Summary
2) Introduction
3) Methodology
4) Results
5) Conclusions
6) Appendix

EXECUTIVE SUMMARY

Summary of Methodologies:

These are the steps this report will go thru:

1. Data Collection
2. Data Wrangling
3. Exploratory Data Analysis
4. Interactive Visual Analytics
5. Predictive Analysis (Classification)

Summary of Results:

This project will produce sets of outputs and visualizations:

1. Exploratory Data Analysis (EDA) results
2. Geospatial analytics
3. Interactive dashboard
4. Predictive analysis of classification models

INTRODUCTION

- SpaceX launches Falcon 9 rockets at a cost that is approximately $100m cheaper than other providers. The reason that SpaceX saves so much money is that they reuse the first stage of the rocket, but it is not always retrieved intact.

- If we can predict whether the first stage lands successfully, we can figure out the cost of the launch, this information can be used to assess whether or not an alternate company should bid and SpaceX for a launch.

METHODOLOGY SUMMARY

1. Data Collection
   - Making GET requests to the SpaceX REST API
   - Web Scraping
2. Data Wrangling
   - `.fillna()` method is used to remove NaN values
   - With the `.value_counts()` method we can:
     - Se the number of launches for every site
     - Se the number and occurrence for every orbit
     - Se the number and occurrence of mission outcome for each orbit type
   - Making a landing outcome label that is:
     - 1 if the booster landed successfully
     - 0 if the booster was not landed successfully
3. Exploratory Data Analysis
   - With SQL queries were used on the dataset to manipulate and evaluate it
   - Pandas and Matplotlib were used to visualize the relationships between variables, and determine patterns
4. Interactive Visual Analytics
   - Folium was used to make geospatial analytics
   - A dashboard using Plotly was created for interactive visualisation
5. Data Modelling and Evaluation
   - Scikit-Learn was used to:
     - Pre-process /standardize data
     - Using the `train_test_split` method to split the data into training and testing data
     - Train a few different classification models
     - Using `GridSearchCV` to find hyperparameters
   - For every classification model Plot there corresponding confusion matrices
   - Assessing each classification models accuracy

DATA COLLECTION – SPACE X REST API

With the SpaceX API we retrieve the data about launches, information on the rocket, the payload delivered, specifications of the launch, specifications of the landing, and landing outcome.

1
- Accessing the SpaceX REST API with a GET response
- Then convert the response from .json file to a Pandas DataFrame

2
- Make lists that are used to store data
- Construct the DataFram by making a dictionary of the lists

3
- Use the dictionary to create a Pandas DataFrame

4
- Remove all the non Falcon 9 launches from the DataFrame
- Replace the old FlightNumber in its column with a new index
- Using the mean PayloadMass value to replace missing values

[GitHub]

1
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
data = pd.json_normalize(response.json())
```

2
```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

```
# Call getBoosterVersion
getBoosterVersion(data)

# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)
```

```
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}
```

3
```
# Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

4
```
data_falcon9 = df[df['BoosterVersion']!='Falcon 1']
```

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
```

```
# Calculate the mean value of PayloadMass column
data_falcon9 = data_falcon9.fillna(value={'PayloadMass': data_falcon9['PayloadMass'].mean()})
```

DATA COLLECTION – WEB SCRAPING

Web scraping the Wikipedia page titled List of Falcon 9 and Falcon Heavy launches to collect historical launch records.

1
- Use the static URL to request the HTML page
- Make a object to assign the response

2
- Make a BeautifulSoup object with the HTML response
- Find the tables in the HTML page

3
- From the HTML page tables get all column header names

4
- Make a dictionary with the column names as keys

5
- Make a export ready Pandas DataFrame by converting the dictionary

GitHub

1
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```
```
# use requests.get() method with the provided static_url
response = requests.get(static_url)
# assign the response to a object
data = response.text
```

2
```
soup = BeautifulSoup(data, 'lxml')
html_tables = soup.find_all('table')
```

3
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if(name != None and len(name) > 0):
        column_names.append(name)
```

4
```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
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'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

5
```
df=pd.DataFrame(launch_dict)
```

DATA MANIPULATION/WRANGLING – PANDAS

- There is a column called `LaunchSite` the SpaceX dataset that contains several Space X launch facilities.
- There exist several dedicated orbits that each launch aims to

Initial Data Exploration:

- With the method `.value_counts()` we can determine the following:

1. How many launches there was from each site
2. How many times each orbit was targeted
3. For every orbit type what there number of outcomes was

GitHub

1

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

2

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

3

```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

DATA MANIPULATION/WRANGLING – PANDAS

- The landing outcomes can be found in the `Outcome` column:
  - `True Ocean` It landed in the Ocean
  - `False Ocean` It failed to land in the Ocean
  - `True RTLS` It landed on a ground pad
  - `False RTLS` It failed to land on a ground pad.
  - `True ASDS` It landed on a drone ship
  - `False ASDS` It failed to land on a drone ship.
  - `None ASDS` and `None None` Maens that it did not land.

Data Wrangling:

- To make it easier to predict if there will be a successful landing we converted the outcomes to binary values i.e 1 for success and 0 for failed. This is how it's done:

1. Creating a set where all the unsuccessful outcomes are stored, `bad_outcome`
2. Making a list that we fil with 0 if the current row belongs to `bad_outcome` and 1 otherwise, called `landing_class`
3. Use `landing_class` to Create the `Class` column
4. Lastly export the data to a .csv file.

GitHub

1

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

3

```
landing_class = []
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

3

```
df['Class']=landing_class
```

4

```
df.to_csv("dataset_part\_2.csv", index=False)
```

EXPLORATORY DATA ANALYSIS – VISUALIZATION

SCATTER CHARTS

When visualizing the relationships/correlations between two variables scatter plots are one of the best ways to do it. In this presentation scatter plots were used to visualize the correlations between Flight Number and Launch Site, Orbit Type and Flight Number, Payload and Launch Site and Payload and Orbit Type

BAR CHART

Bar charts are useful when comparing categorical and numerical variables. In this presentation a bar chart was used to visualize the Success Rate and Orbit Type

LINE CHARTS

Line charts are useful when there is a numerical value that changes over time, in this presentation it is used to visualize the relationships between Success Rate and Year

GitHub

EXPLORATORY DATA ANALYSIS (EDA) – SQL

These are the SQL queries performed on the data(what they did):

1. View all the names of launch sites
2. View 5 records where launch sites names start with 'CCA'
3. Summarize all payload mass carried by boosters launched by NASA (CRS)
4. Get the average payload mass that boosters of version F9 v1.1 carried
5. Get the date of the first successful landing on a ground pad
6. Get the boosters that landed successfully with payload mass between 4000 and 6000 kg on a drone ship
7. Get the total number of successful and failed mission outcomes
8. Get what booster versions that have carried the maximum payload mass
9. Get the landing outcomes that failed to land on drone ships, what booster version they had and launch site names for 2015
10. Between the date 2010-06-04 and 2017-03-20 get the landing outcome and list them on descending order(by date)

GitHub

GEOSPATIAL ANALYSIS – FOLIUM

Put all the launch sites on the map

- Started with making the Folium Map
- Mark the launch sites on the map with `folium.Circle` and `folium.Marker`

Put the success/failed launches on each site

- Sence many launches were on the same site they can be clustered this was done with the `MarkerCluster()` object
- The markers were colour coded based on outcome.

GitHub

INTERACTIVE DASHBOARD – PLOTLY DASH

1. `px.pie()` was used to make a Pie chart, the pie chart could be filtered with a dropdawn menu that was added with `dcc.Dropdown()`

2. `px.scatter()` was used to make a Scatter graph, a range slider was added with `RangeSlider()`

GitHub

PREDICTIVE ANALYSIS - CLASSIFICATION

Model Development

- To prepare the dataset for model development:
  - Load dataset
  - Clean data
  - Split data using `train_test_split()`
  - Decide which type of algorithms are most appropriate
- For each chosen algorithm:
  - Use `GridSearchCV`
  - Fit to the parameters
  - Train the model

Model Evaluation

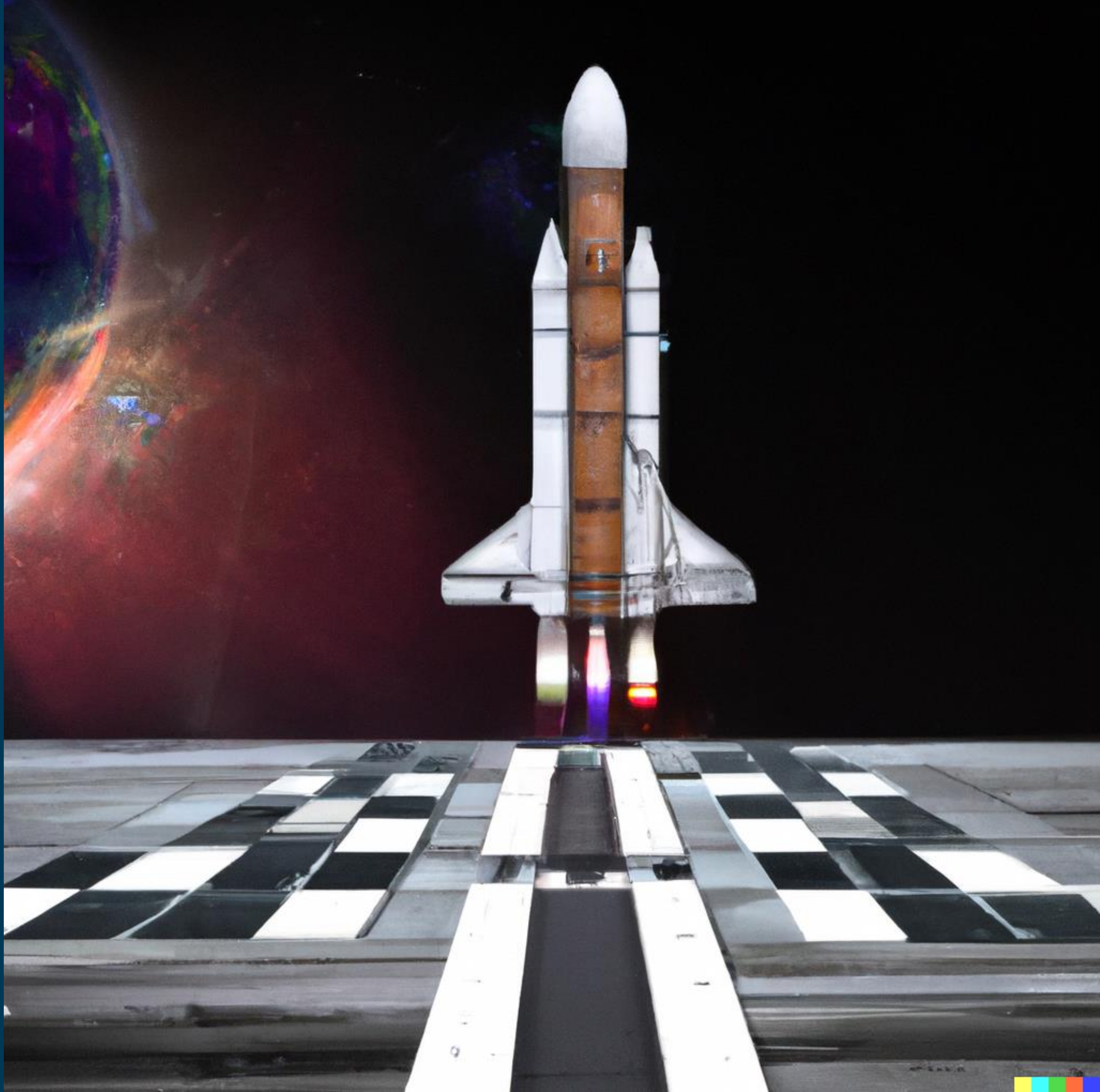
- For every chosen algorithm:
  - Using the output `GridSearchCV` object:
    - Check the tuned hyperparameters
    - Check the accuracy
  - Plot Confusion Matrix

Finding the Best Classification Model

- Review the accuracy scores
- Chose the best model

GitHub

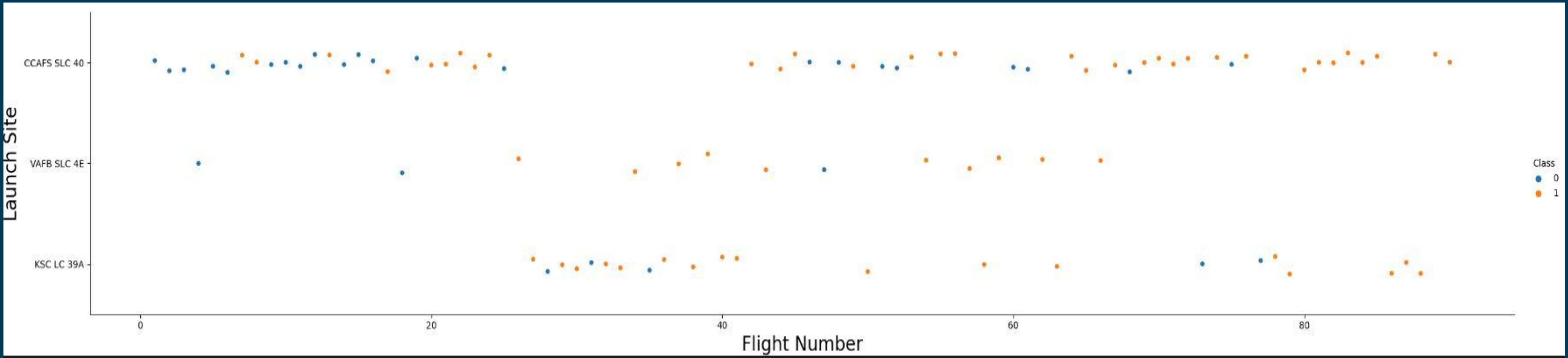
RESULTS



EDA - WITH VISUALIZATION

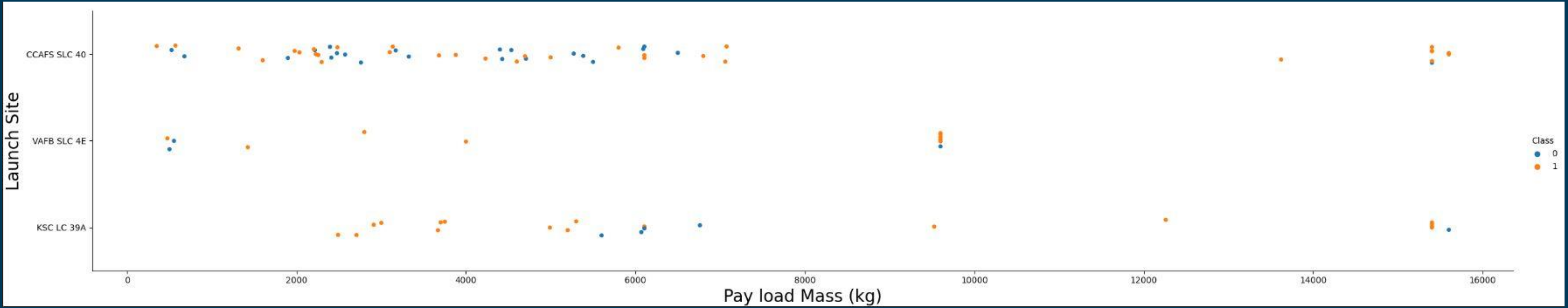
LAUNCH SITE VS. FLIGHT NUMBER

The y axis is the launch site and the x axis is the Flight number.
The coulure of the dot represents if it was a success or not.



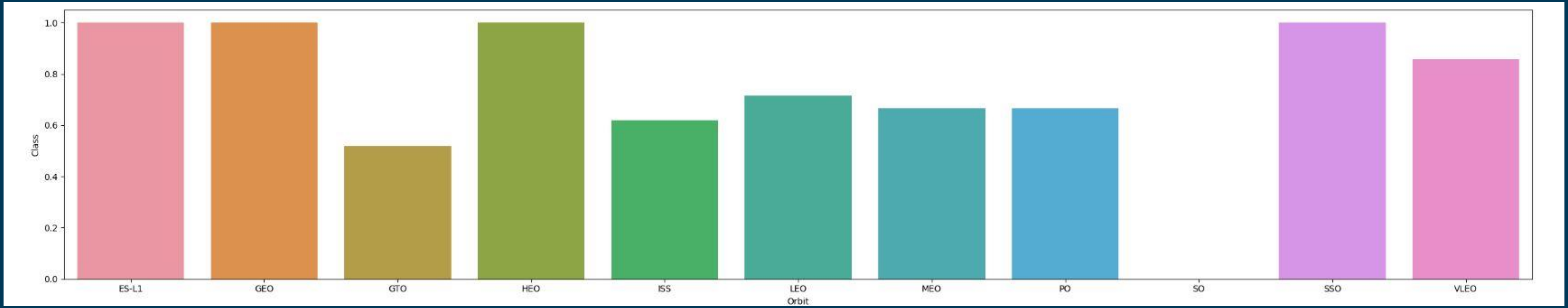
LAUNCH SITE VS. PAYLOAD MASS

In this plot the y axis is the launch site and the x axis is the payload mass.
The dots represents if it was a successful launch.



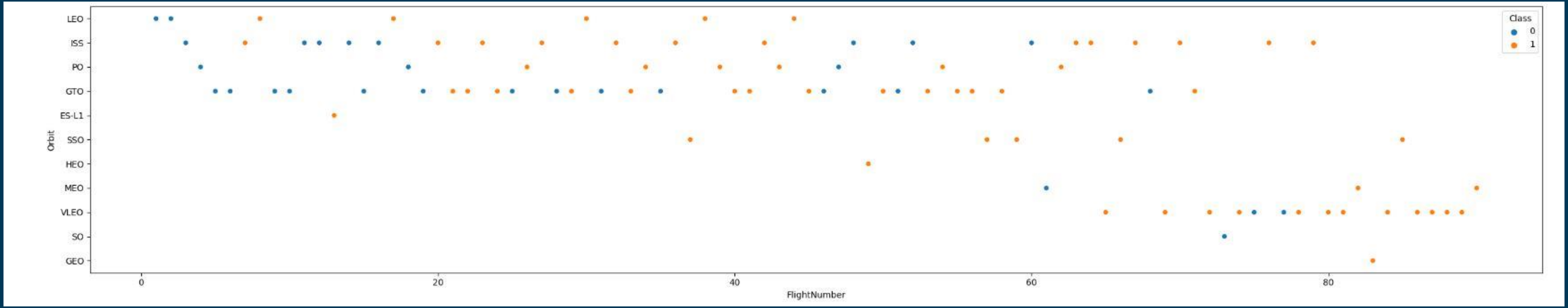
SUCCESS RATE VS. ORBIT TYPE

The x axis is the launch site and the y axis is the success rate.



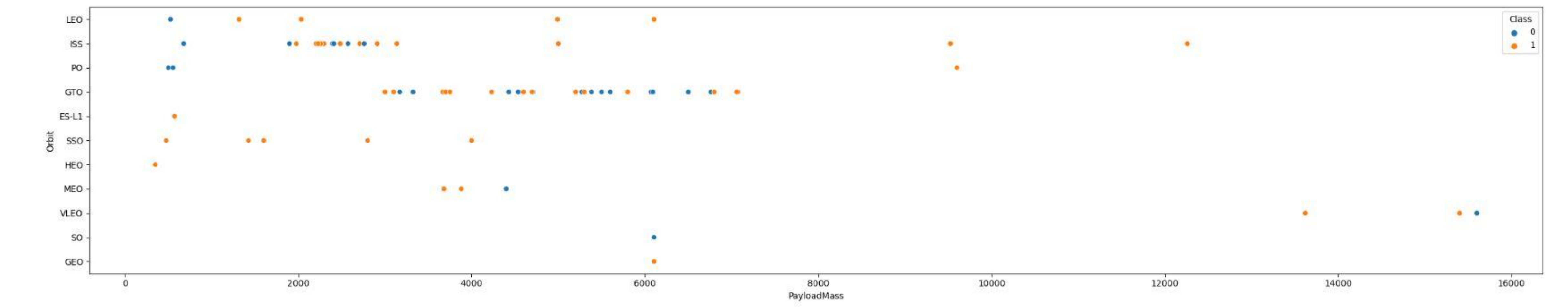
ORBIT TYPE VS. FLIGHT NUMBER

The x axis is the flight number and the y axis is the orbit type.
The coulure of the dots represents if it was a success.



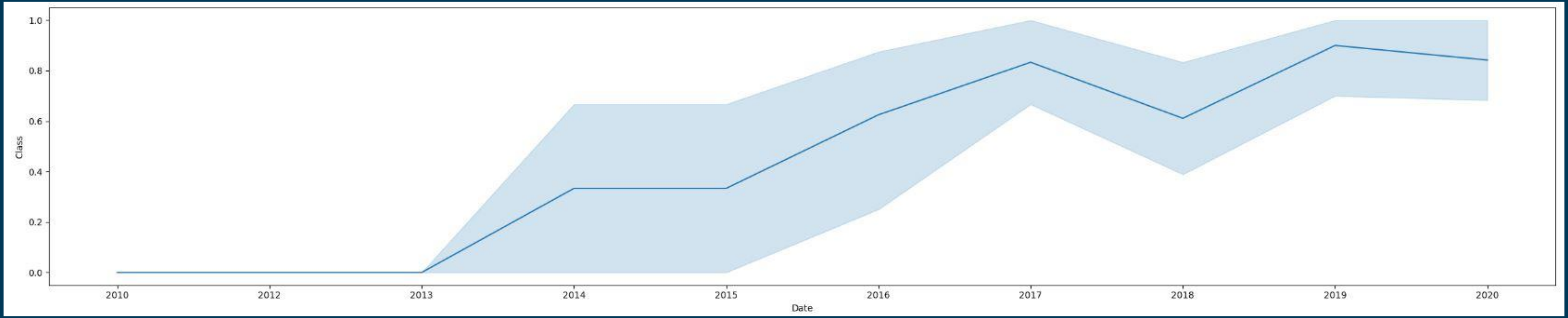
ORBIT TYPE VS. PAYLOAD MASS

The x axis is the payload mass and the y axis is the orbit type.
The coulure of the dots represents if it was a success.



LAUNCH SUCCESS YEARLY TREND

The x axis is the date and the y axis is the success rate.



EDA - WITH SQL

ALL LAUNCH SITE NAMES

Display the names of the unique launch sites in the space mission

```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL;
```

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

`DISTINCT` returns all unique values in the `LAUNCH_SITE` column from `SPACEXTBL` table.

LAUNCH SITE NAMES BEGIN WITH 'CCA'

Display 5 records where launch sites begin with the string 'CCA'

```
%sql 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
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

`LIMIT 5` gives only 5 rows, and `LIKE ‘CCA%’` is used to only give string values beginning with ‘CCA’.

TOTAL PAYLOAD MASS

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';
```

TOTAL_PAYLOAD_MASS
45596

SUM is used to summarize the numeric values of the LAUNCH column, and the WHERE filters the results to only boosters from NASA (CRS).

AVERAGE PAYLOAD MASS BY F9 V1.1

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVERAGE_PAYLOAD_MASS FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';
```

AVERAGE_PAYLOAD_MASS
2928.4

AVG gives the average of the PAYLOAD_MASS__KG_ column, WHERE only selects results of the F9 v1.1 booster version.

FIRST SUCCESSFUL GROUND LANDING DATE

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

```
%sql SELECT MIN(DATE) AS FIRST_SUCCESSFUL_GROUND_LANDING FROM SPACEXTBL WHERE `Landing _Outcome` = 'Success (ground pad)';
```

FIRST_SUCCESSFUL_GROUND_LANDING
01-05-2017

MIN gives the earliest date from the DATE column and the WHERE filters the results to only the successful ground pad landings.

SUCCESSFUL DRONE SHIP LANDING WITH PAYLOAD BETWEEN 4000 AND 6000

List the names of the boosters which have success in drone ship and have 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

WHERE filters the to the data we want the AND is used so multiple restrictions can be used and BETWEEN narrows the search between 4000 < x < 6000.

TOTAL NUMBER OF SUCCESSFUL AND FAILURE MISSION OUTCOMES

List the total number of successful and failure mission outcomes

```
%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS TOTAL_NUMBER FROM SPACEXTBL GROUP BY MISSION_OUTCOME;
```

Mission_Outcome	TOTAL_NUMBER
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

COUNT is used to summarize the total number of each mission outcomes, GROUPBY groups the results by type of mission outcome.

BOOSTERS CARRIED MAXIMUM PAYLOAD

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

```
%sql 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 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

SELECT statement in brackets is a subquery that finds the maximum payload, it is then used in the WHERE condition. DISTINCT then gives only distinct booster versions.

2015 LAUNCH RECORDS

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.

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE, substr(Date,4,2) AS Months FROM SPACEXTBL WHERE (`Landing _Outcome` = 'Failure (drone ship)') AND substr(Date,7,4)='2015' ORDER BY Months;
```

Booster_Version	Launch_Site	Months
F9 v1.1 B1012	CCAFS LC-40	01
F9 v1.1 B1015	CCAFS LC-40	04

WHERE is used to filter the results for only failed landing outcomes, AND only for the year of 2015.

RANK LANDING OUTCOMES BETWEEN 2010-06-04 AND 2017-03-20

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

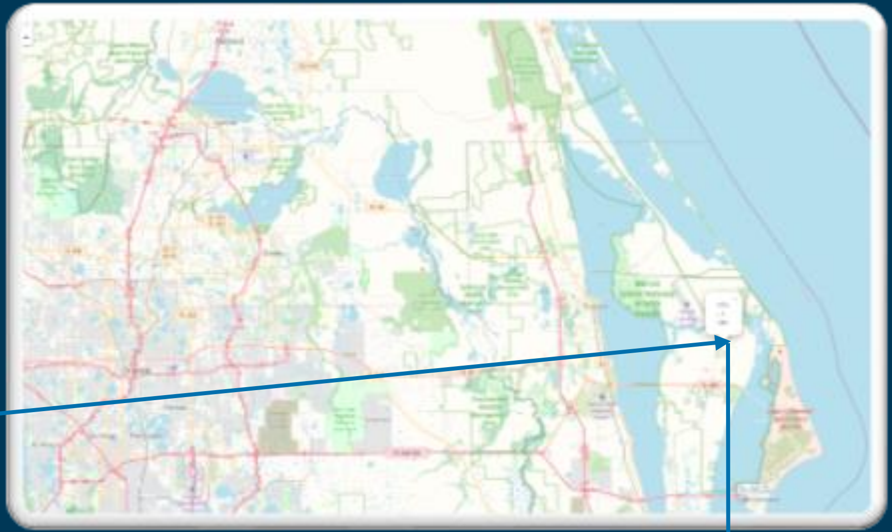
```
%sql SELECT `Landing _Outcome`, COUNT(`Landing _Outcome`) AS TOTAL_NUMBER FROM SPACEXTBL WHERE `Landing _Outcome` LIKE 'Success%' AND DATE BETWEEN '04-06-2010' AND '20-03-2017' GROUP BY `Landing _Outcome` ORDER BY TOTAL_NUMBER DESC;
```

Landing _Outcome	TOTAL_NUMBER
Success	20
Success (drone ship)	8
Success (ground pad)	6

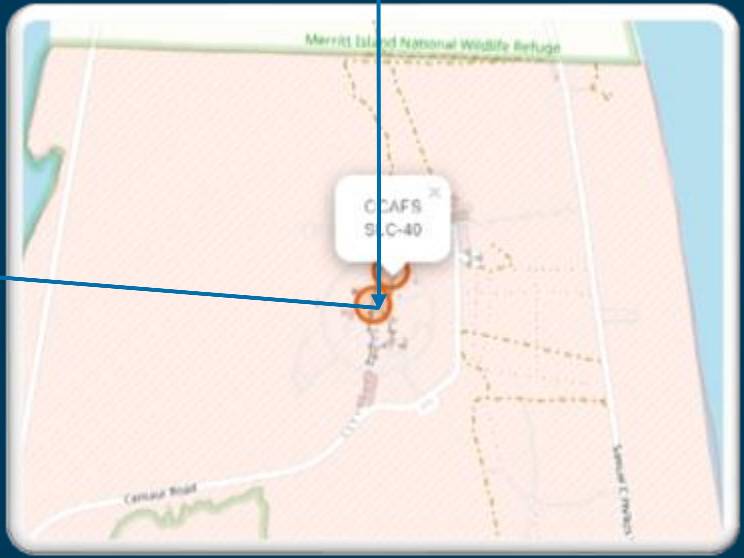
`BETWEEN` filters so only between the dates are selected which is used with `LIKE` so only successful landings are chosen. `GROUP BY` is used to group the findings and `ORDER BY` orders them `DESC` is then used to specify the descending order.

LAUNCH SITES PROXIMITY ANALYSIS – FOLIUM INTERACTIVE MAP

ALL LAUNCH SITES ON A MAP



The SpaceX launch sites are all on coasts of USA(Florida and California).

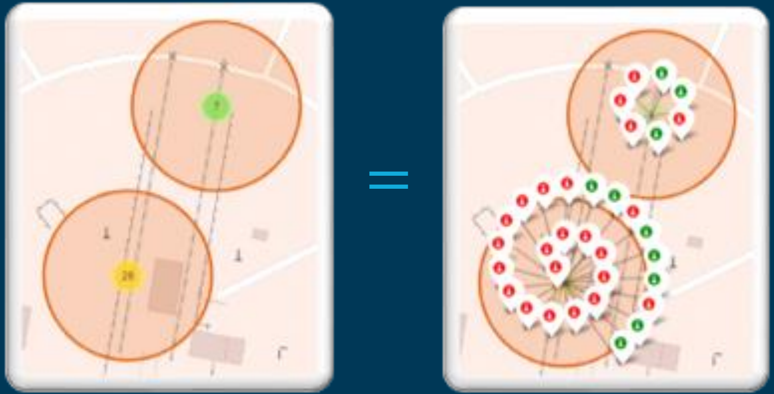


SUCCESS/FAILED LAUNCHES FOR EACH SITE

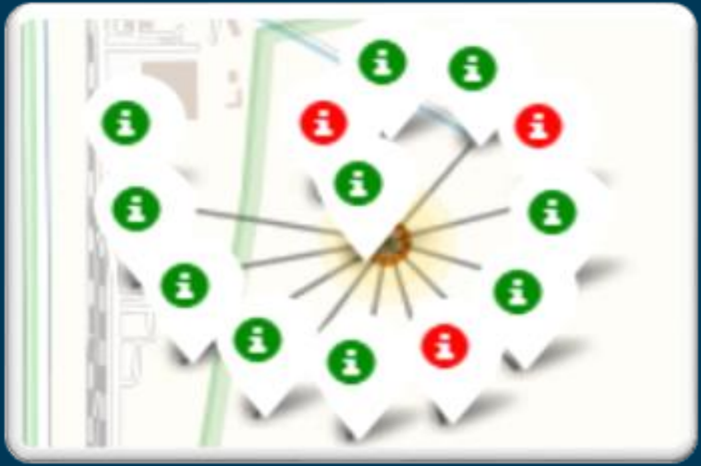
For each site the launches was grouped into clusters, were successful launches was markt with green icons and red icons if it failed

VAFB SLC-4E

CCAFS SLC-40 and CCAFS LC-40



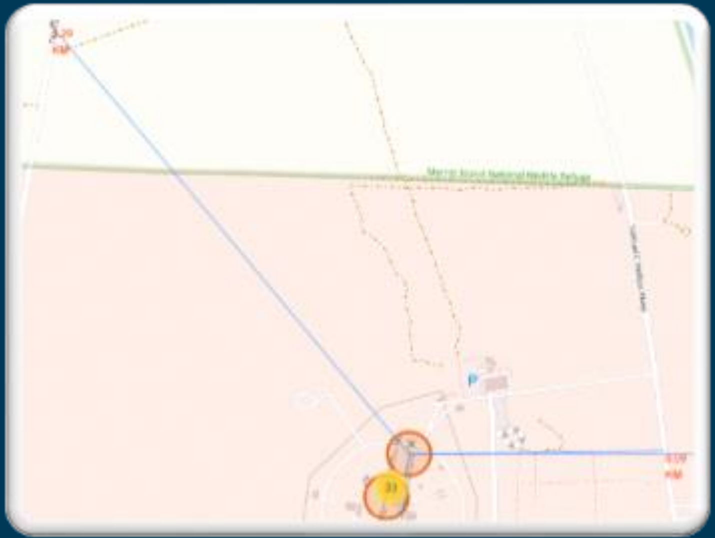
KSC LC-39A

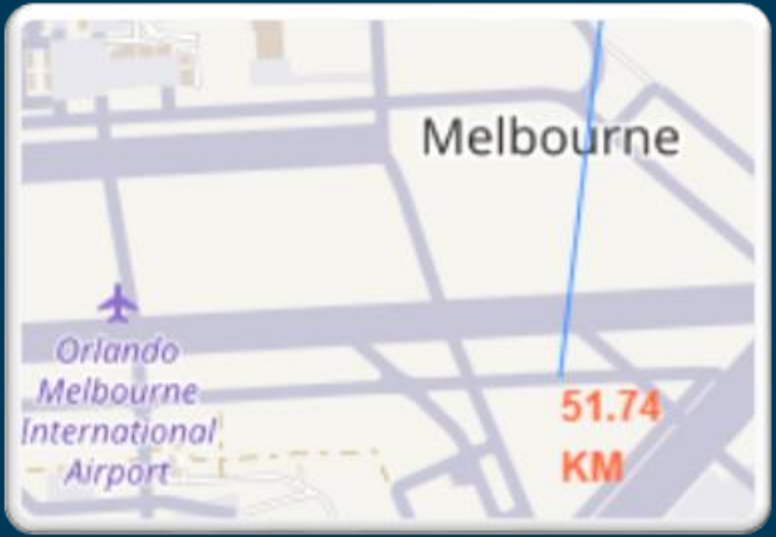


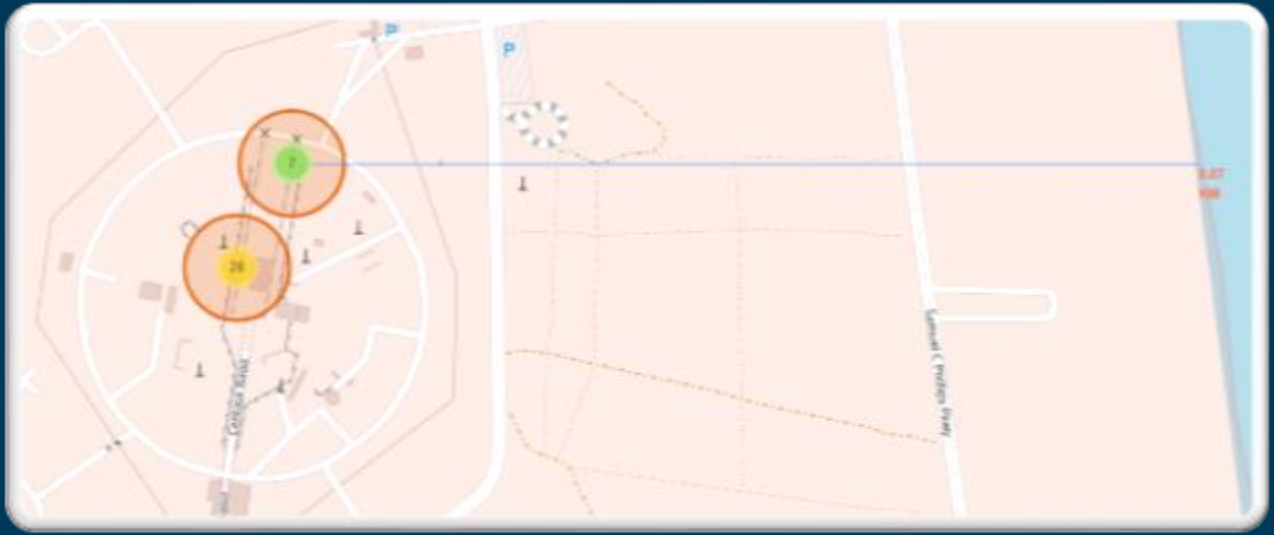
PROXIMITY OF LAUNCH SITES TO OTHER POINTS OF INTEREST

If we take the launch site CCAFS SLC-40 for example we can see the following when it comes to distance to points of interest.

- The launch site is less then 1km away from the coastline and highway, the closets railway is only 1.29km away.
- The launch site is also kept a distance from city's, the nearest city is 52km away.



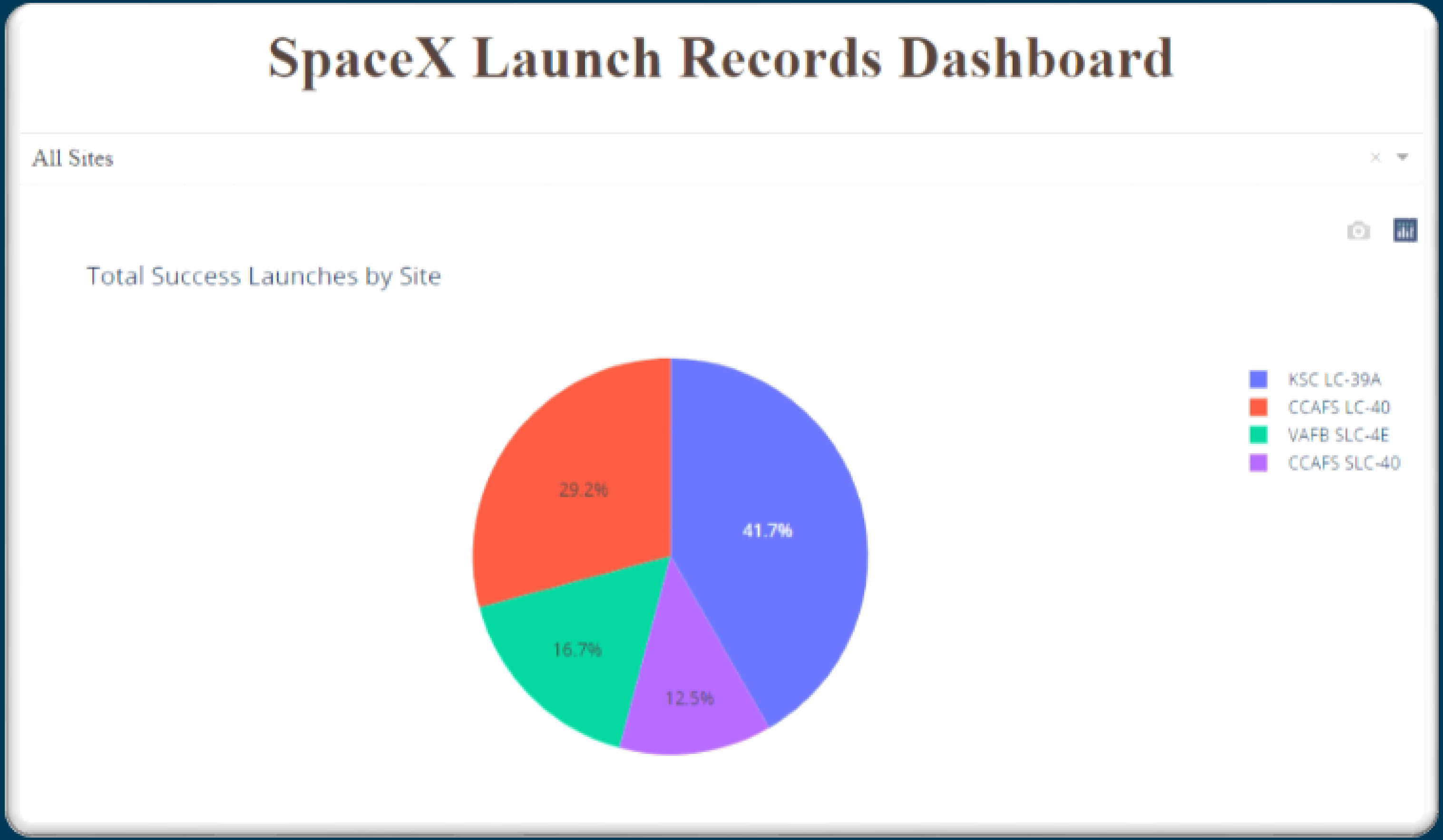




INTERACTIVE DASHBOARD - PLOTLY DASH

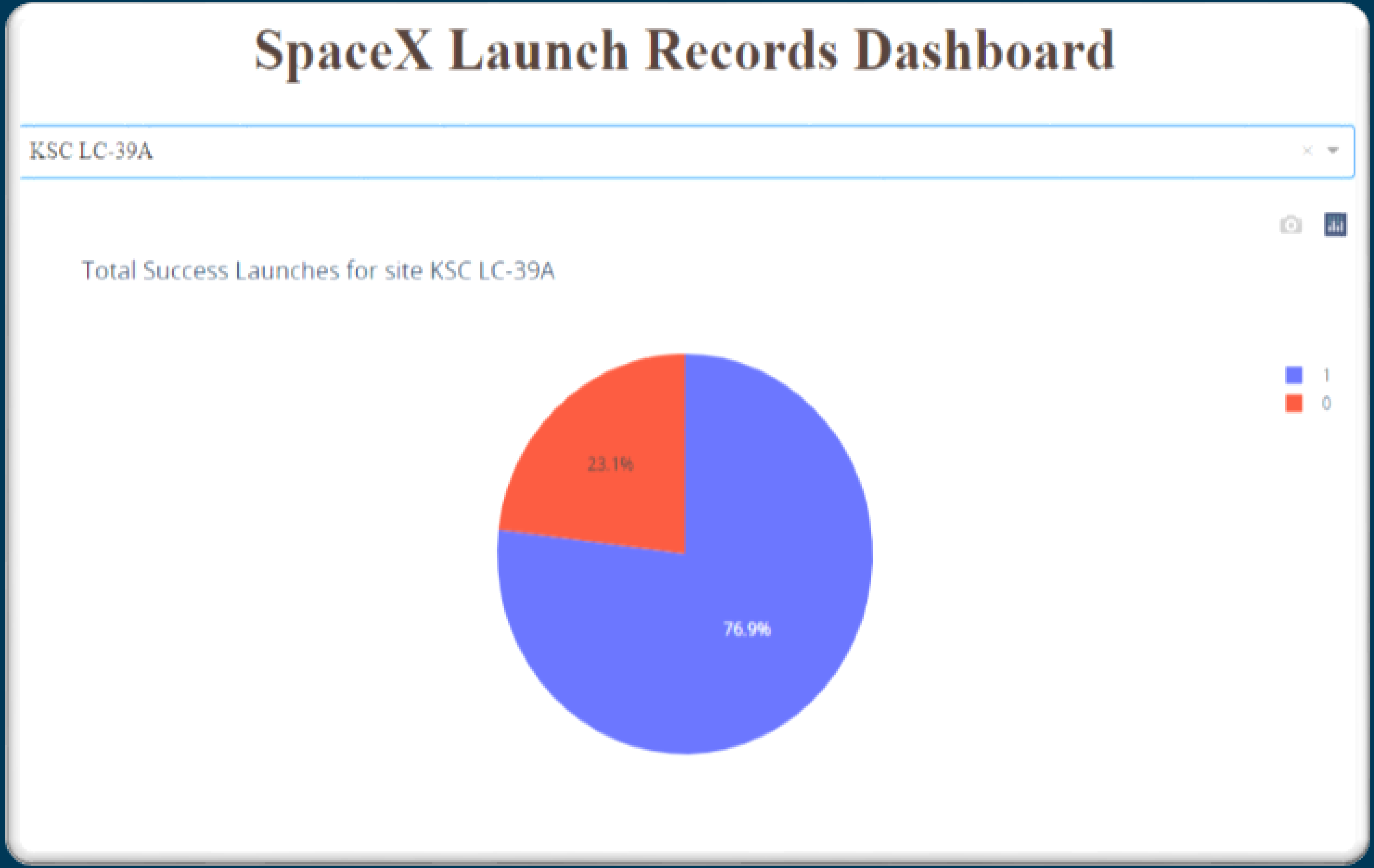
LAUNCH SUCCESS COUNT FOR ALL SITES

The Launch site with the most successful launches was KSC LC-39A and the one with the least was CCAFS SLC-40



PIE CHART FOR THE LAUNCH SITE WITH HIGHEST LAUNCH SUCCESS RATIO

KSC LC-39A was the launch site with the highest success rate.



Note: $class \begin{cases} 0, Failure \\ 1, Success \end{cases}$

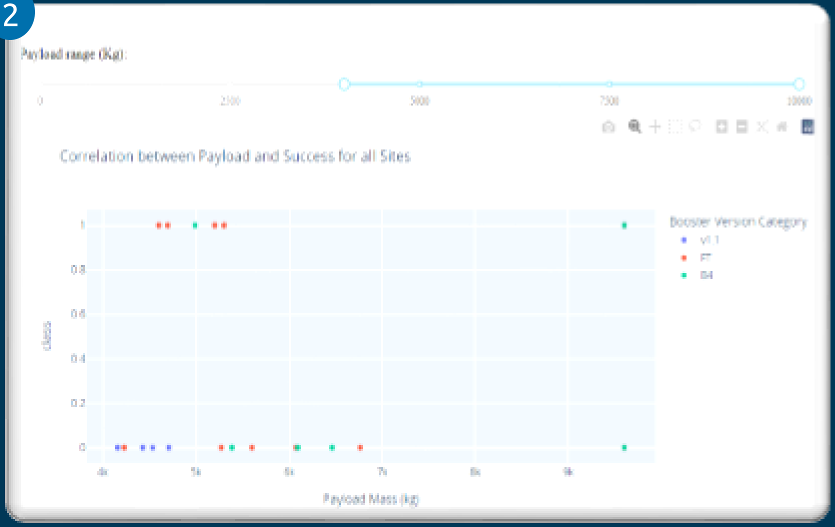
LAUNCH OUTCOME VS. PAYLOAD SCATTER PLOT FOR ALL SITES

- The plot bellow display all the launches but the plot 1 and 2 are narrowed down to low payloads(0-4000kg) and high payloads(4000-10 000 kg) from these plot we can see that the launches with low payload have more successful launches.

Note: $class\begin{cases}0, Failure\\1, Success\end{cases}$

1

2

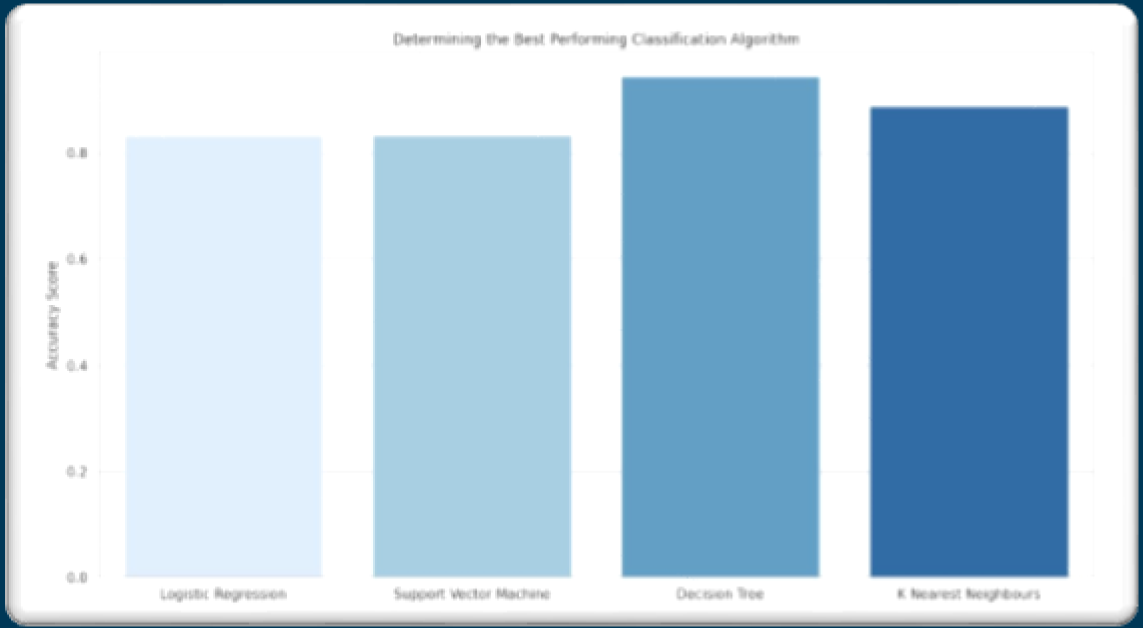


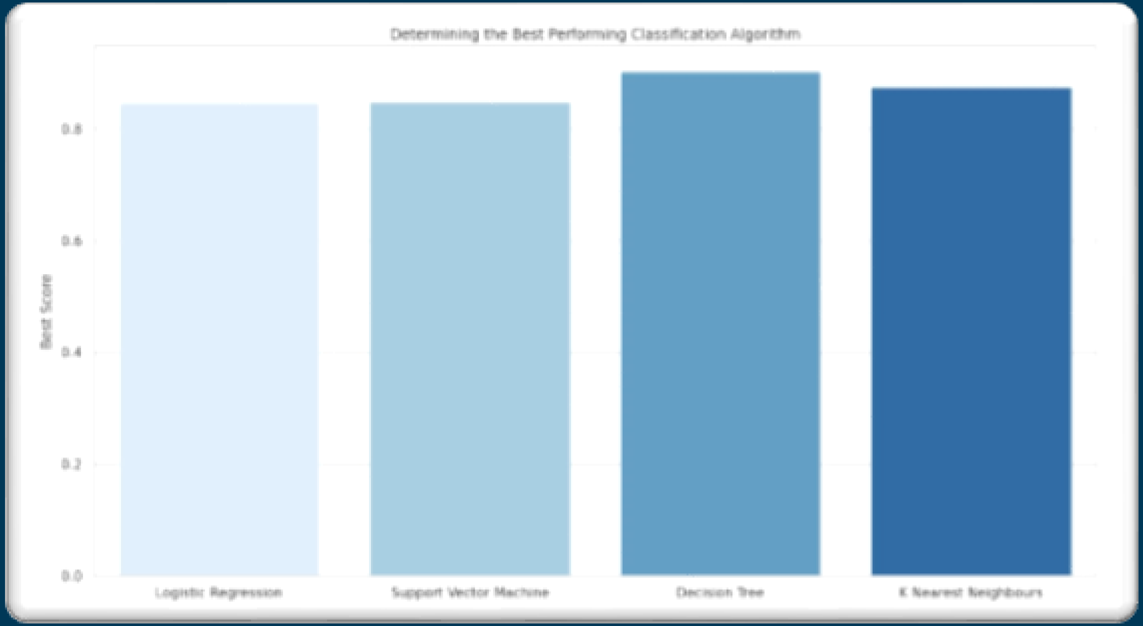
PREDICTIVE ANALYSIS - CLASSIFICATION

CLASSIFICATION ACCURACY

	Algorithm	Accuracy Score	Best Score
0	Logistic Regression	0.833333	0.846429
1	Support Vector Machine	0.833333	0.848214
2	Decision Tree	0.944444	0.901786
3	K Nearest Neighbours	0.888889	0.875000

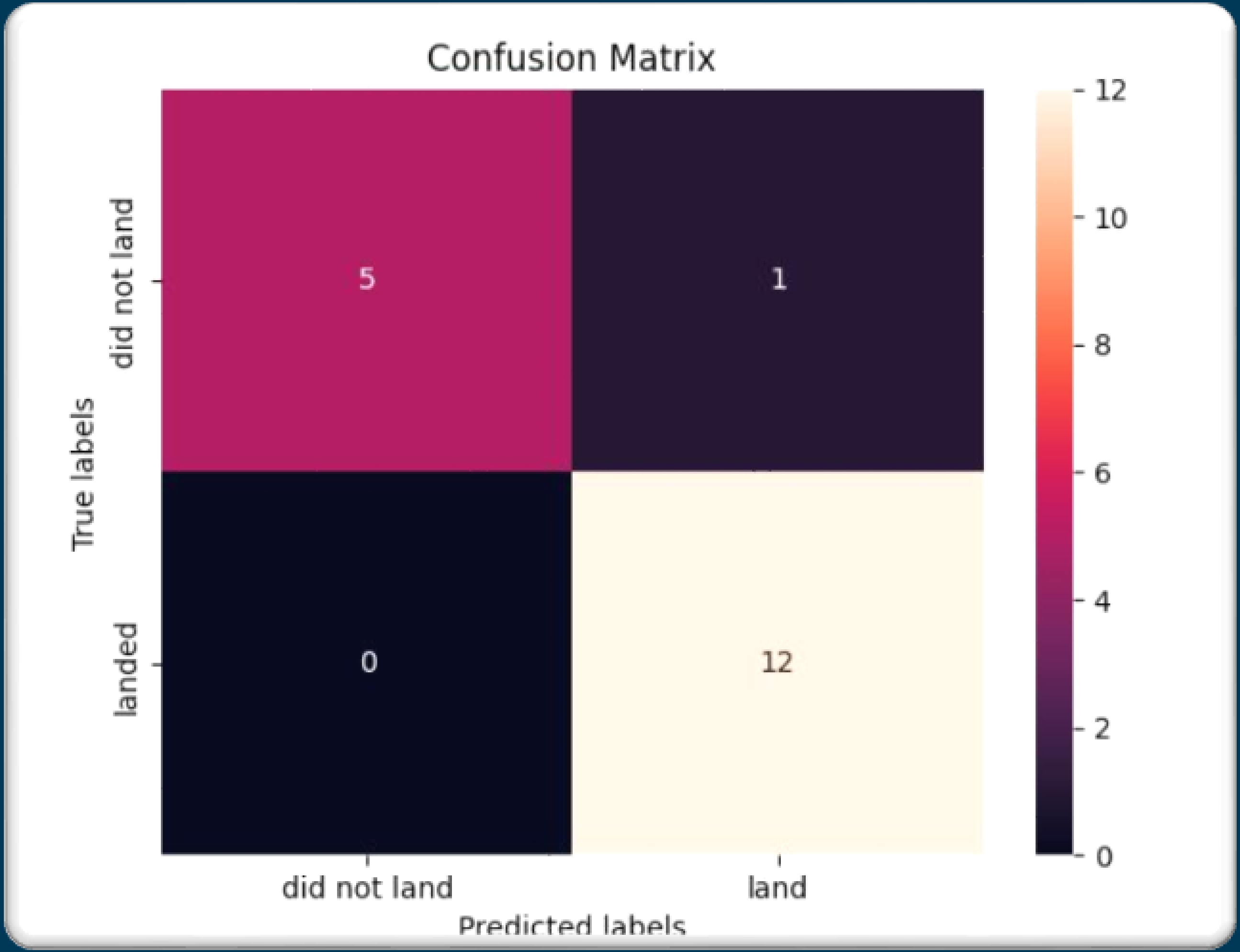
We can see from the data and the bar charts that the Decision Tree model hade the best results.





CONFUSION MATRIX

This is the confusion matrix for the Decision Tree model. In it we can see that it predicted a total of 17 landing outcomes correctly were 12 landed successfully and 5 did not land successfully. There was only one prediction that was wrong and it was a false posetiv.



CONCLUSIONS

CONCLUSIONS

- As they have launched more flights over the years we can see that there success rate has gone up meaning they are getting more consistent results.

- We can see that some orbit type have better success rate than others.

- Launches with low payload(under 4000kg) had better success rate then the heavy launches.

- The Decision Tree model was the best performing classification model and hade an accuracy of 94.44%.

APPENDIX

All code can be found in my github