



IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Chaiya Saengchan
31/8/2023



Outline

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

Executive Summary

- Summary of methodologies
 - Data Collection via API, Web Scraping
 - Exploratory Data Analysis (EDA) with Data Visualization
 - EDA with SQL
 - Interactive Map with Folium
 - Dashboards with Plotly Dash
 - Predictive Analysis
- Summary of all results
 - Exploratory Data Analysis results
 - Interactive maps and dashboard
 - Predictive results

Introduction

- Project background and context
 - The aim of this project is to predict if the Falcon 9 first stage will successfully land. SpaceX says on its website that the Falcon 9 rocket launch cost 62 million dollars. Other providers cost upward of 165 million dollars each. The price difference is explained by the fact that SpaceX can reuse the first stage. By determining if the stage will land, we can determine the cost of a launch. This information is interesting for another company if it wants to compete with SpaceX for a rocket launch.
- Problems you want to find answers
 - What are the main characteristics of a successful or failed landing ?
 - What are the effects of each relationship of the rocket variables on the success or failure of a landing ?
 - What are the conditions which will allow SpaceX to achieve the best landing success rate ?

Section 1

Methodology

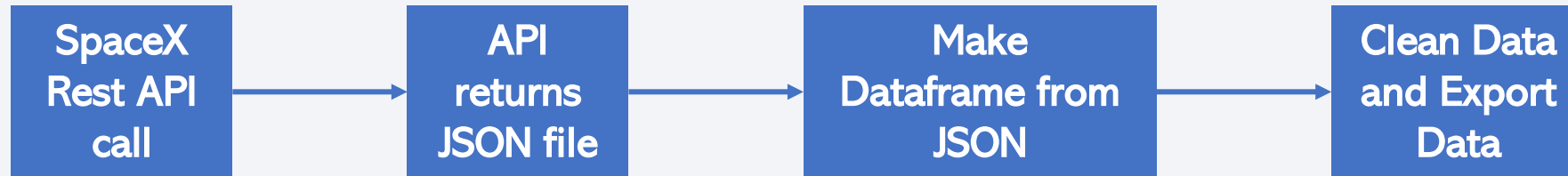
Methodology

Executive Summary

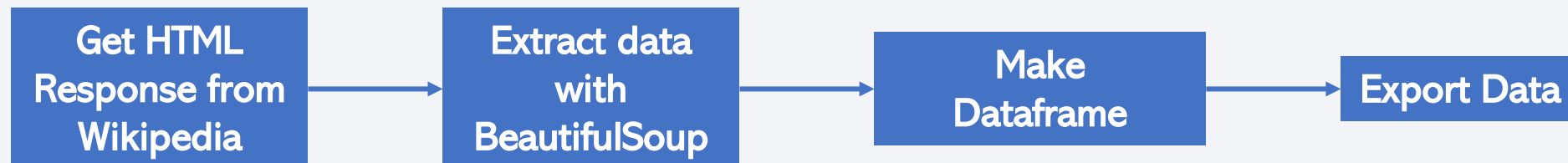
- Data collection methodology:
 - SpaceX REST API
 - Web Scrapping from Wikipedia
- Perform data wrangling
 - Dropping unnecessary columns
 - One Hot Encoding for classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
 - How to build, tune, evaluate classification models

Data Collection

- Datasets are collected from Rest SpaceX API and webscrapping Wikipedia
 - The information obtained by the API are rocket, launches, payload information.
 - The Space X REST API URL is api.spacexdata.com/v4/



- The information obtained by the webscrapping of Wikipedia are launches, landing, payload information.
 - URL is <https://en.wikipedia.org/w/index.php?title=List of Falcon 9 and Falcon Heavy launches&oldid=1027686922>



Data Collection – SpaceX API

[Link to code](#)

1. Getting Response from API

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

2. Convert Response to JSON File

```
response = requests.get(static_json_url)
response.json()

# Get the head of the dataframe
data = pd.json_normalize(response.json())
data.head()
```

3. Transform data

```
# Call getBoosterVersion
getBoosterVersion(data)

# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)
```

4. Create dictionary with 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}
```

5. Create dataframe

```
# Create a data from launch_dict
data_launch = pd.DataFrame(launch_dict)
```

6. Filter dataframe

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data_launch[(data_launch['BoosterVersion']!='Falcon 1')]
data_falcon9.head()
```

7. Export to file

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


Data Collection - Scraping

[Link to code](#)

1. Getting Response from HTML

```
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
# assign the response to a object
Falcon9_data = requests.get(url=static_url)
```

2. Create BeautifulSoup Object

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(Falcon9_data.content, 'html.parser')
# soup
```

3. Find all tables

```
# Use the find_all function in the BeautifulSoup object, with element type 'table'.
# Assign the result to a list called 'html_tables'
html_tables = soup.find_all('table')
# html_tables
```

4. Get column names

```
column_names = []

# Apply find_all() function with 'th' element
# Iterate each th element and apply the provided function
# Append the Non-empty column name (if name is not None)

for th in first_launch_table.find_all('th'):
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

5. Create dictionary

```
launch_dict = dict.fromkeys(column_names)

# Remove an irrelevant 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'] = []

launch_dict['Customer']
```

6. Add 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()
```

[See notebook for the rest of code](#)

7. Create dataframe from dictionary

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

8. Export to file

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

Data Wrangling

[Link to code](#)

- In the dataset, there are several cases where the booster did not land successfully.
 - True Ocean, True RTLS, True ASDS means the mission has been successful.
 - False Ocean, False RTLS, False ASDS means the mission was a failure.
- We need to transform string variables into categorical variables where 1 means the mission has been successful and 0 means the mission was a failure

1. Calculate launches number for each site

```
# 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. Calculate the number and occurrence of each orbit

```
# 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. Calculate number and occurrence of mission outcome per orbit type

```
# 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
```

4. Create landing outcome label from Outcome column

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
count = 0
landing_class = list()
for i in df['Outcome']:
    if i in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

5. Export to file

```
df.to_csv("dataset_part\2.csv", index=False)
```

EDA with Data Visualization

[Link to code](#)

- 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 relationship between variables.
This relationship is called the correlation.*

- Bar Graph

- Success rate vs. Orbit

Bar graphs show the relationship between numeric and categoric variables.

- Line Graph

- Success rate vs. Year

Line graphs show data variables and their trends. Line graphs can help to show global behavior and make prediction for unseen data.

EDA with SQL

[Link to code](#)

- We performed SQL queries to gather and understand data from dataset:
 - Displaying 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 successful landing outcome in ground pad was achieved.
 - 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.
 - 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 successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order

Build an Interactive Map with Folium

[Link to code](#)

- Folium map object is a map centered on NASA Johnson Space Center at Houston, Texas
 - Red circle at NASA Johnson Space Center's coordinate with label showing its name (folium.Circle, folium.map.Marker).
 - Red circles at each launch site coordinates with label showing launch site name (folium.Circle, folium.map.Marker, folium.features.DivIcon).
 - The grouping of points in a cluster to display multiple and different information for the same coordinates (folium.plugins.MarkerCluster).
 - Markers to show successful and unsuccessful landings. Green for successful landing and Red for unsuccessful landing. (folium.map.Marker, folium.Icon).
 - Markers to show distance between launch site to key locations (railway, highway, coastway, city) and plot a line between them. (folium.map.Marker, folium.PolyLine, folium.features.DivIcon)
- These objects are created in order to understand better the problem and the data. We can show easily all launch sites, their surroundings and the number of successful and unsuccessful landings.

Build a Dashboard with Plotly Dash

[Link to code](#)

- Dashboard has dropdown, pie chart, rangeslider and scatter plot components
 - Dropdown allows a user to choose the launch site or all launch sites (`dash_core_components.Dropdown`).
 - Pie chart shows the total success and the total failure for the launch site chosen with the dropdown component (`plotly.express.pie`).
 - Rangeslider allows a user to select a payload mass in a fixed range (`dash_core_components.RangeSlider`).
 - Scatter chart shows the relationship between two variables, in particular Success vs Payload Mass (`plotly.express.scatter`).

Predictive Analysis (Classification)

[Link to code](#)

- **Data preparation**
 - Load dataset
 - Normalize data
 - Split data into training and test sets.
- **Model preparation**
 - Selection of machine learning algorithms
 - Set parameters for each algorithm to GridSearchCV
 - Training GridSearchModel models with training dataset
- **Model evaluation**
 - Get best hyperparameters for each type of model
 - Compute accuracy for each model with test dataset
 - Plot Confusion Matrix
- **Model comparison**
 - Comparison of models according to their accuracy
 - The model with the best accuracy will be chosen (see Notebook for result)

Results

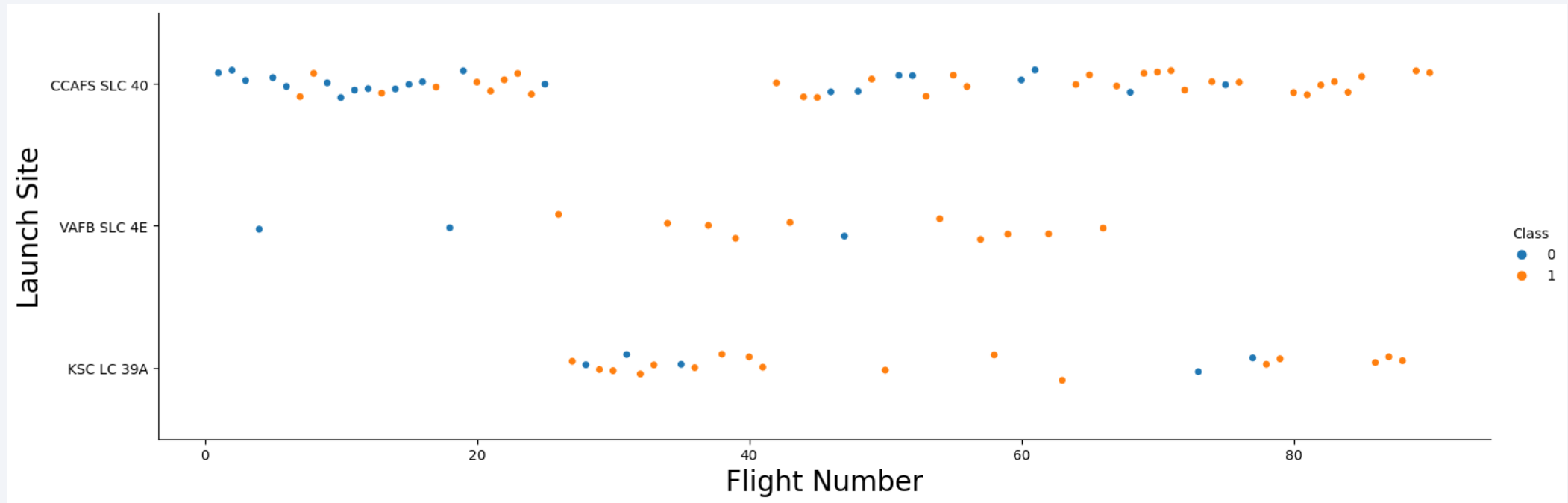
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



Section 2

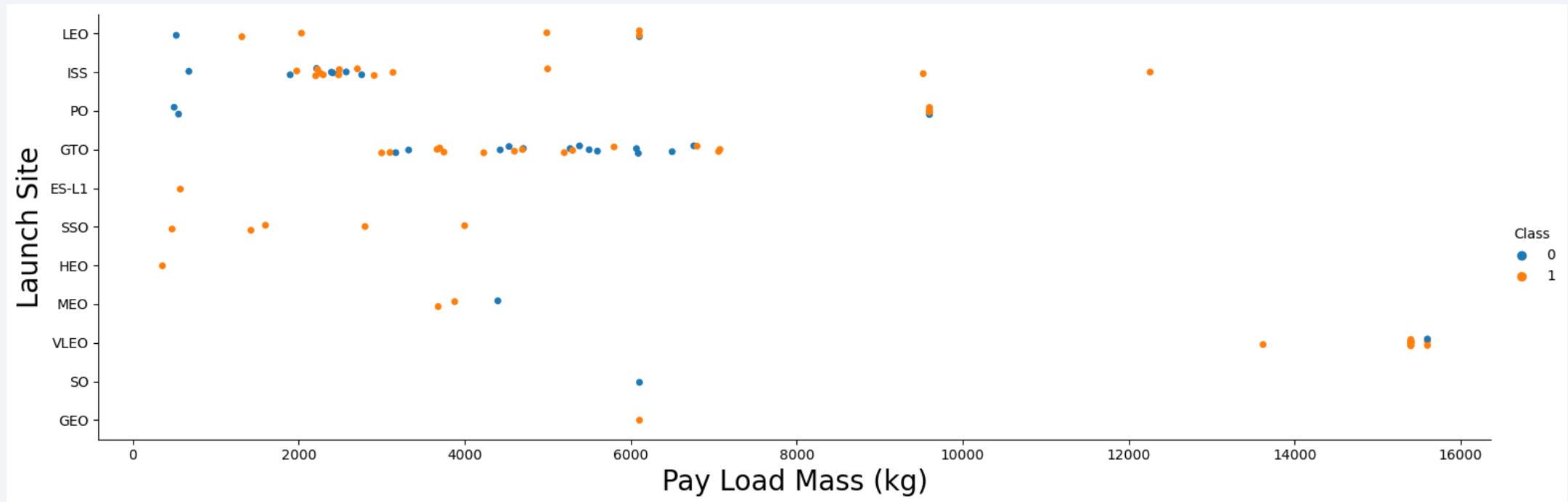
Insights drawn from EDA

Flight Number vs. Launch Site



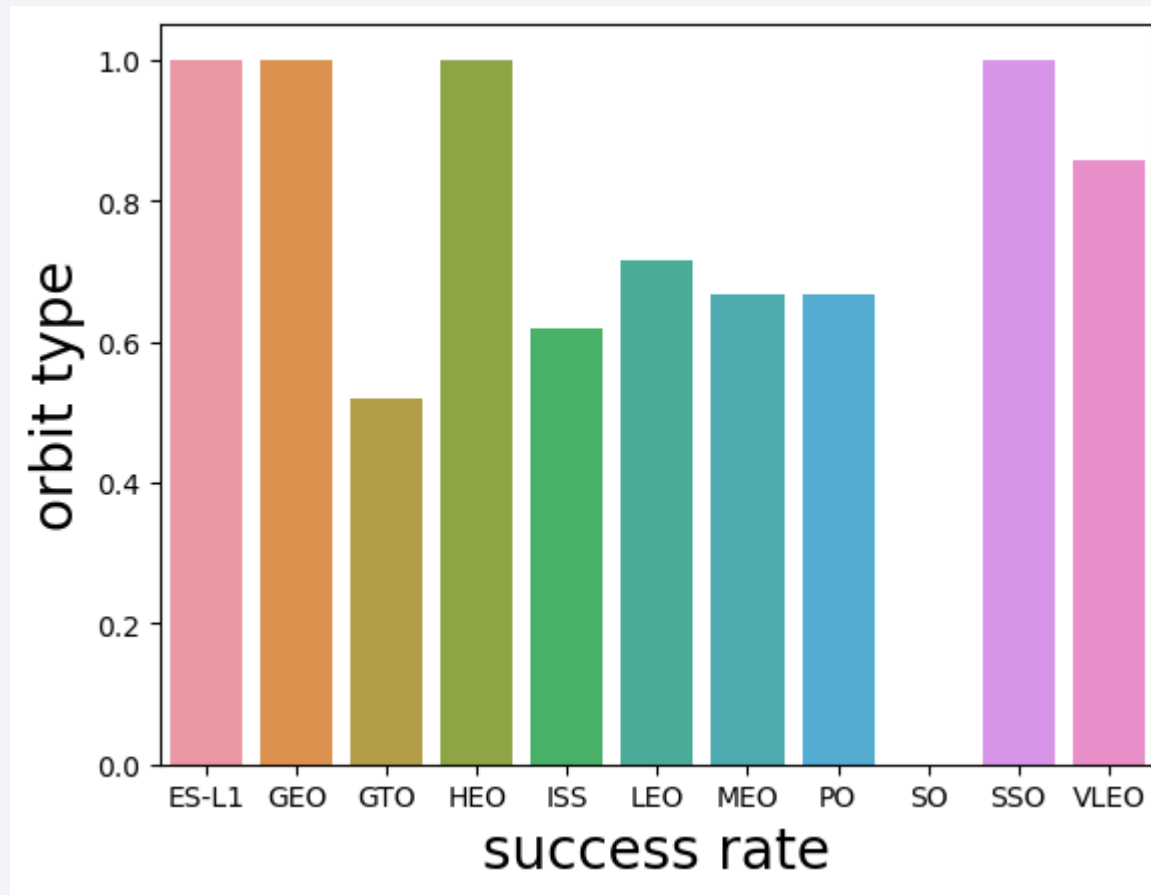
We observe that, for each site, the success rate is increasing

Payload vs. Launch Site



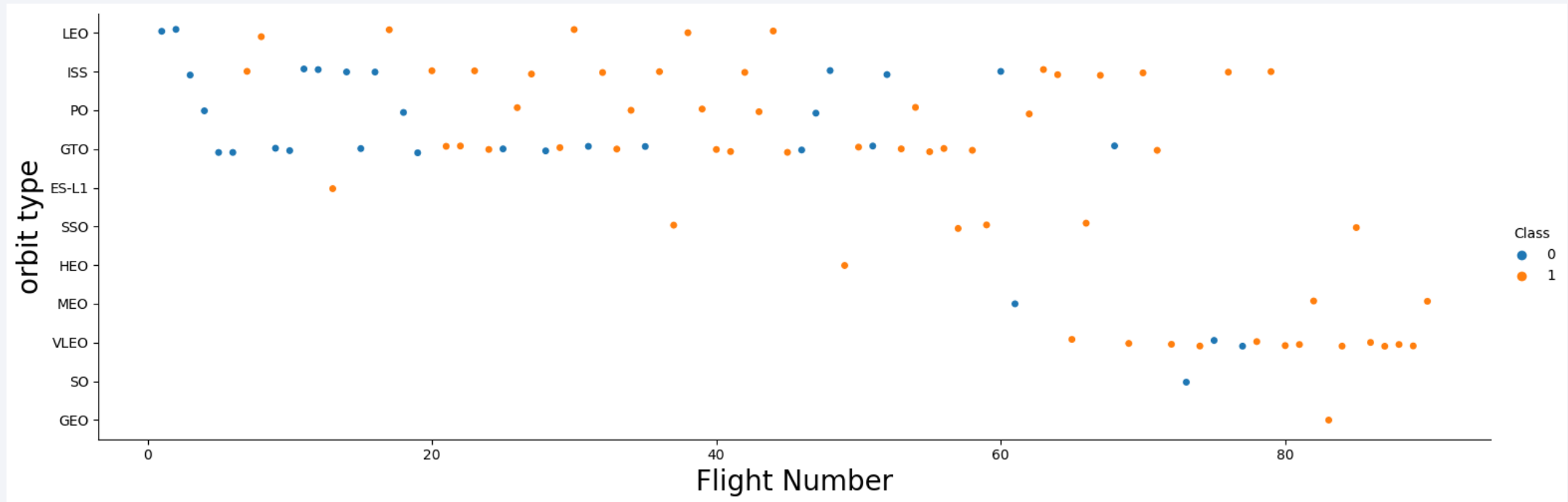
Depending on the launch site, a heavier payload may be a consideration for a successful landing. On the other hand, a too heavy payload can make a landing fail.

Success Rate vs. Orbit Type



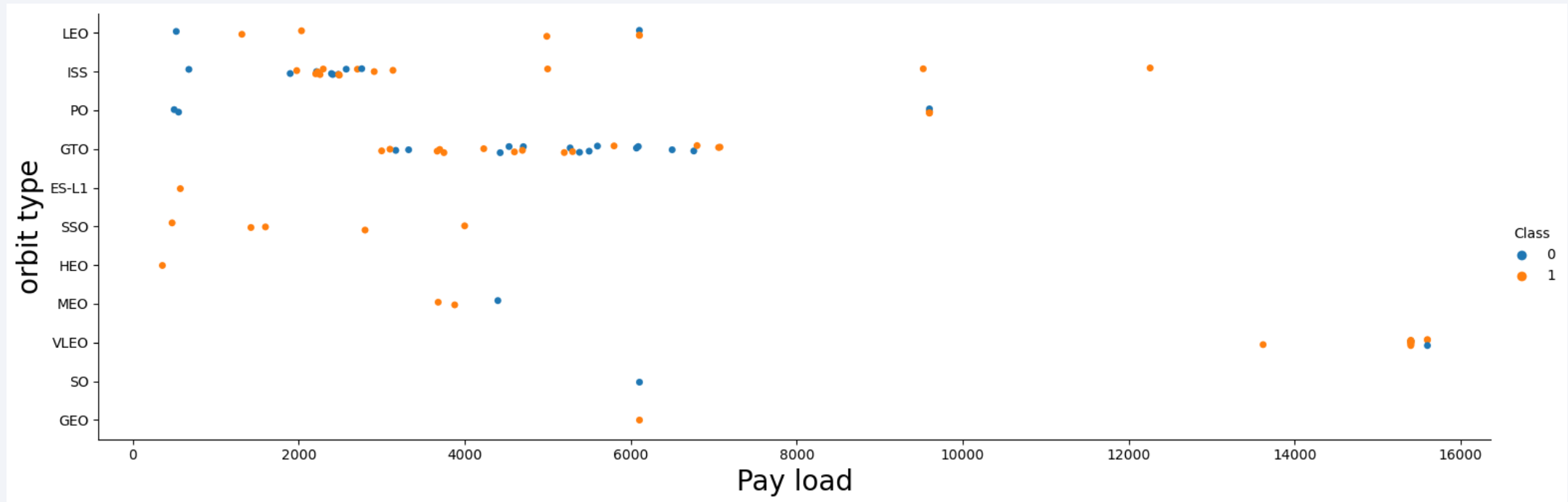
With this plot, we can see success rate for different orbit types. We note that ES-L1, GEO, HEO, SSO have the best success rate.

Flight Number vs. Orbit Type



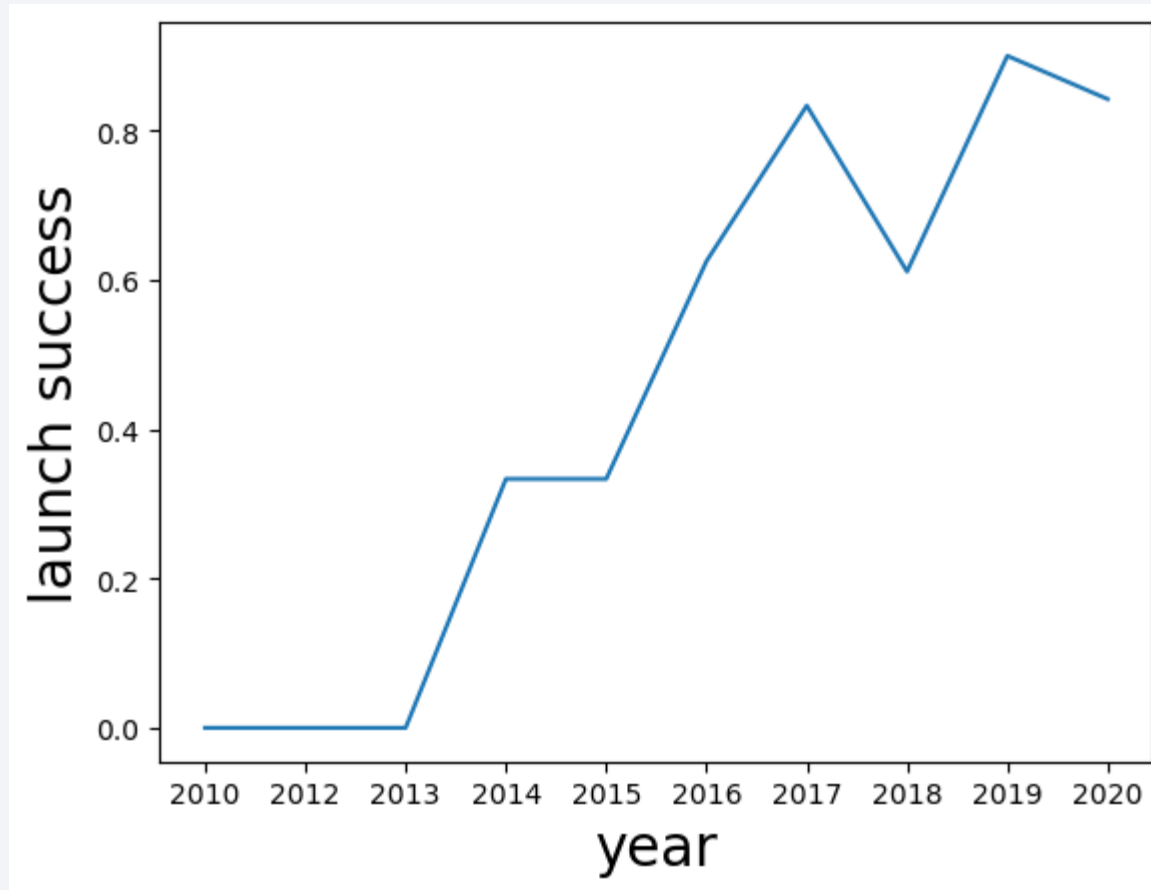
We notice that the success rate increases with the number of flights for the LEO orbit. For some orbits like GTO, there is no relation between the success rate and the number of flights. But we can suppose that the high success rate of some orbits like SSO or HEO is due to the knowledge learned during former launches for other orbits.

Payload vs. Orbit Type



The weight of the payloads can have a great influence on the success rate of the launches in certain orbits. For example, heavier payloads improve the success rate for the LEO orbit. Another finding is that decreasing the payload weight for a GTO orbit improves the success of a launch.

Launch Success Yearly Trend



- Since 2013, we can see an increase in the Space X Rocket success rate.

All Launch Site Names

SQL Query

```
[13]: %sql select distinct Launch_Site from SPACEXTABLE
```

Results

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

Explanation

The use of DISTINCT in the query allows to remove duplicate LAUNCH_SITE.

Launch Site Names Begin with 'CCA'

SQL Query

```
%sql select * from SPACEXTABLE where Launch_Site like "CCA%" limit 5
```

Explanation

The WHERE clause followed by LIKE clause filters launch sites that contain the substring CCA. LIMIT 5 shows 5 records from filtering.

Results

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS_KG	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

SQL Query

```
%sql select sum(PAYLOAD_MASS_KG_) as 'total_payload_mass_of_NASA_CRS'
```

```
from SPACEXTABLE where Customer = 'NASA (CRS)'
```

Results

total_payload_mass_of_NASA_CRS
45596

Explanation

This query returns the sum of all payload masses where the customer is NASA (CRS).

Average Payload Mass by F9 v1.1

SQL Query

```
%sql select avg(PAYLOAD_MASS_KG_) as  
'average_payload_mass_of_booster_version_F9_v1.1' from SPACETABLE  
where Booster_Version = 'F9 v1.1'
```

Results

average_payload_mass_of_booster_version_F9_v1.1
2928.4

Explanation

This query returns the average of all payload masses where the booster version contains the substring F9 v1.1.

First Successful Ground Landing Date

SQL Query

```
%sql select min(Date) as first_successful_landing_Date,  
Booster_Version, Payload, Customer from SPACEXTABLE  
where Landing_Outcome like "Success (ground pad)"
```

Results

first_successful_landing_Date	Booster_Version	Payload	Customer
2015-12-22	F9 FT B1019	OG2 Mission 2 11 Orbcomm-OG2 satellites	Orbcomm

Explanation

With this query, we select the oldest successful landing. The WHERE clause filters dataset in order to keep only records where landing was successful. With the MIN function, we select the record with the oldest date.

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL Query

```
%sql select Booster_Version, PAYLOAD_MASS_KG_, Landing_Outcome from SPACEXTABLE
where (Landing_Outcome like "Success (ground pad)")
and (PAYLOAD_MASS_KG_ > 4000 and PAYLOAD_MASS_KG_ < 6000)
```

Results

Booster_Version	PAYLOAD_MASS_KG_	Landing_Outcome
F9 FT B1032.1	5300	Success (ground pad)
F9 B4 B1040.1	4990	Success (ground pad)
F9 B4 B1043.1	5000	Success (ground pad)

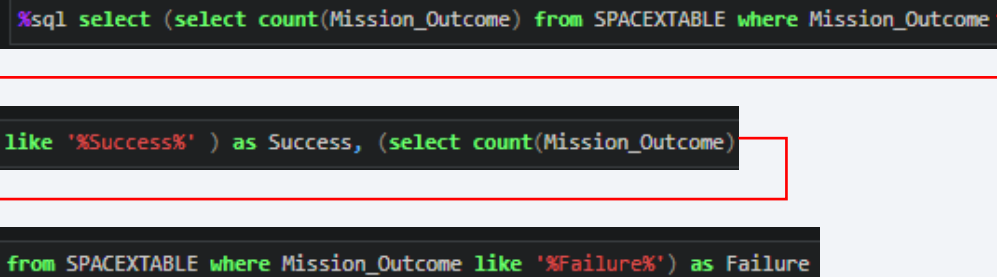
Explanation

This query returns the booster version where landing was successful and payload mass is between 4000 and 6000 kg. The WHERE and clauses filter the dataset.

Total Number of Successful and Failure Mission Outcomes

SQL Query

```
%sql select (select count(Mission_Outcome) from SPACEXTABLE where Mission_Outcome  
like '%Success%' ) as Success, (select count(Mission_Outcome)  
from SPACEXTABLE where Mission_Outcome like '%Failure%') as Failure
```



Results

Success	Failure
100	1

Explanation

With the first SELECT, we show the subqueries that return results. The first subquery counts the successful mission. The second subquery counts the unsuccessful mission. The WHERE clause followed by LIKE clause filters mission outcome. The COUNT function counts records filtered.

Boosters Carried Maximum Payload

SQL Query

```
%sql select Booster_Version, PAYLOAD_MASS_KG_ from SPACESTATION
where PAYLOAD_MASS_KG_ = (select max(PAYLOAD_MASS_KG_) from SPACESTATION)
```

Explanation

We used a subquery to filter data by returning only the heaviest payload mass with MAX function. The main query uses subquery results and returns unique booster version (SELECT DISTINCT) with the heaviest payload mass.

Results

Booster_Version	PAYLOAD_MASS_KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

SQL Query

```
%sql select substr(Date,6,2) AS month_in_2015, Booster_Version, Launch_Site,  
Landing_Outcome from SPACEXTABLE  
where Landing_Outcome like 'Failure (drone ship)'
```

Explanation

This query returns month, booster version, launch site where landing was unsuccessful and landing date took place in 2015. Substr function process date in order to take month or year. Substr(DATE, 4, 2) shows month. Substr(DATE,7, 4) shows year.

Results

Done.

month_in_2015	Booster_Version	Launch_Site	Landing_Outcome
10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)
01	F9 v1.1 B1017	VAFB SLC-4E	Failure (drone ship)
04	F9 FT B1020	CCAFS LC-40	Failure (drone ship)
06	F9 FT B1024	CCAFS LC-40	Failure (drone ship)

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

SQL Query

```
%sql select Landing_Outcome, count(Landing_Outcome) as 'Count' from SPACESTATION
where (Date > '2010-06-04' and Date < '2017-03-20') group by Landing_Outcome
order by count(Landing_Outcome) desc
```

Explanation

This query returns landing outcomes and their count where mission was successful and date is between 04/06/2010 and 20/03/2017. The GROUP BY clause groups results by landing outcome and ORDER BY COUNT DESC shows results in decreasing order.

Results

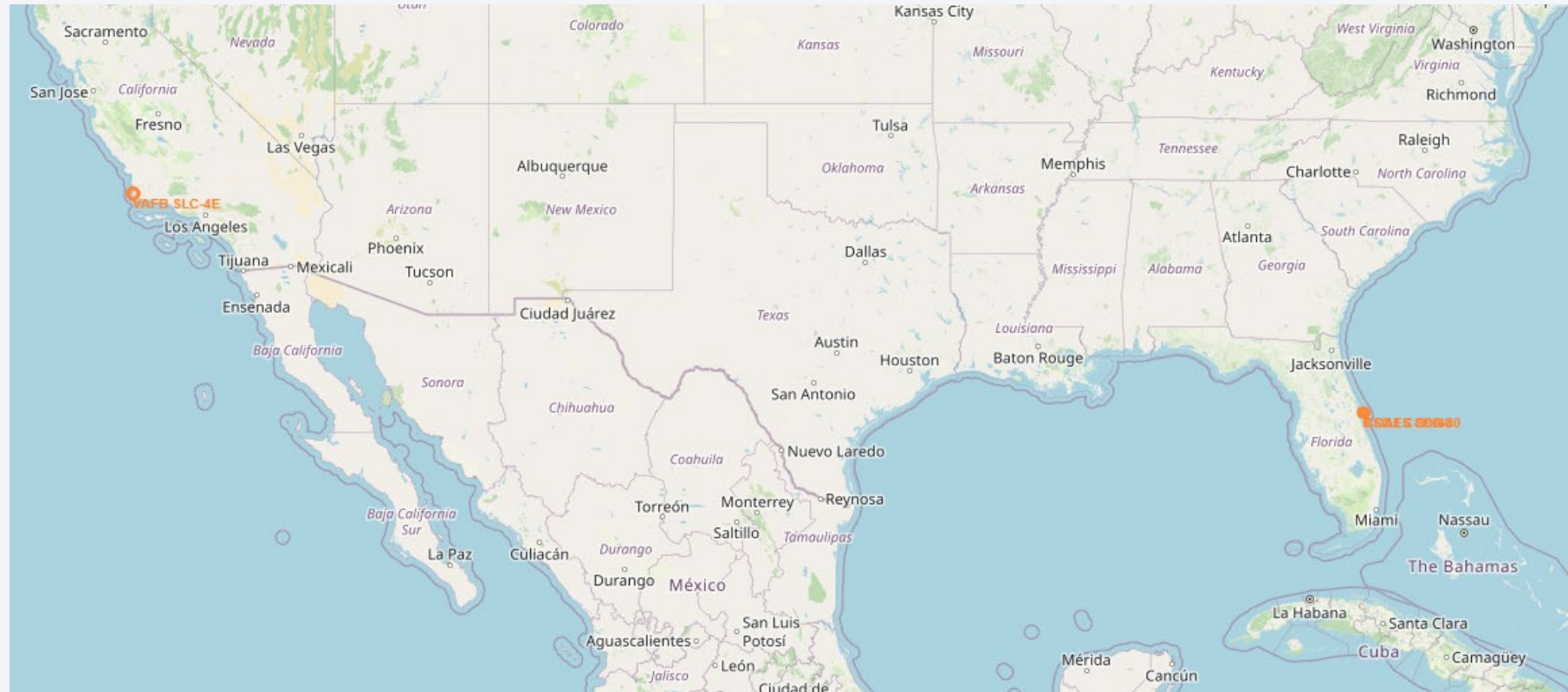
Landing_Outcome	Count
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

Section 3

Launch Sites Proximities Analysis

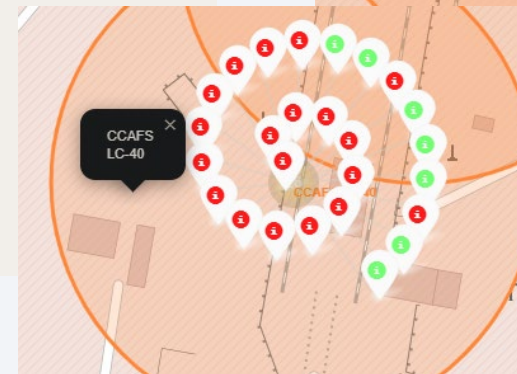
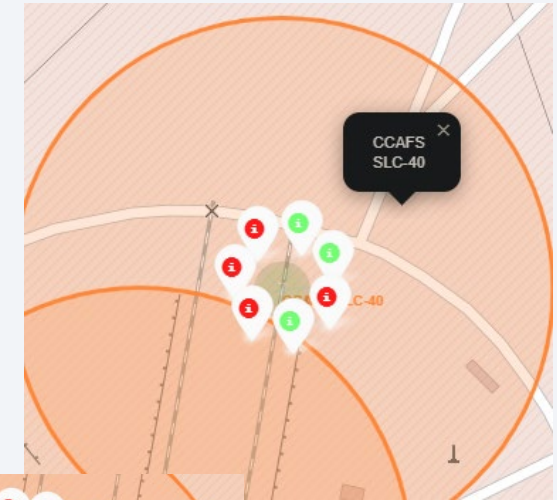
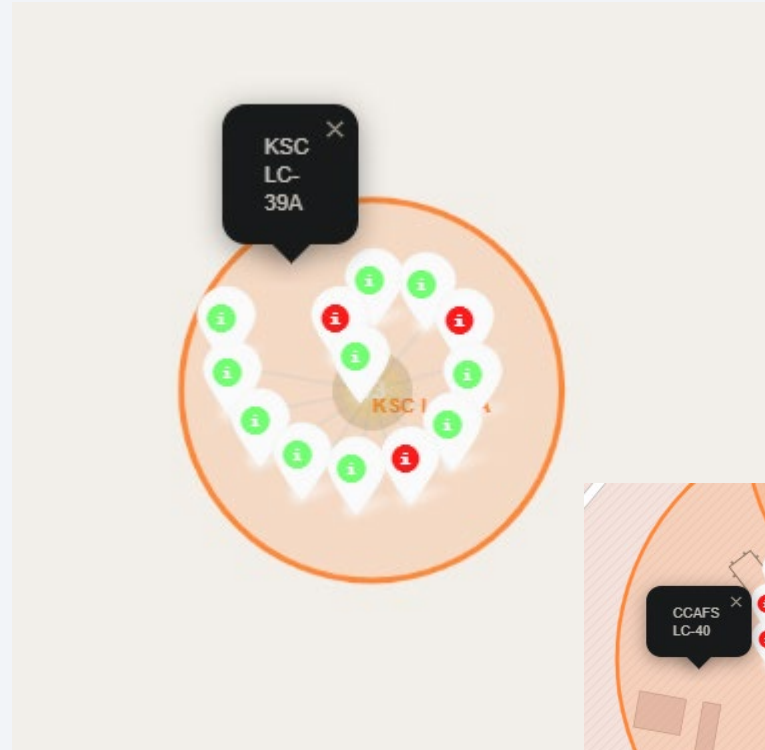


Folium map – Ground stations



We see that SpaceX launch site is located on the coast of the United States, at Vandenberg State Marine Reserve and coasts of Florida.

Folium map – Color Labeled Markers

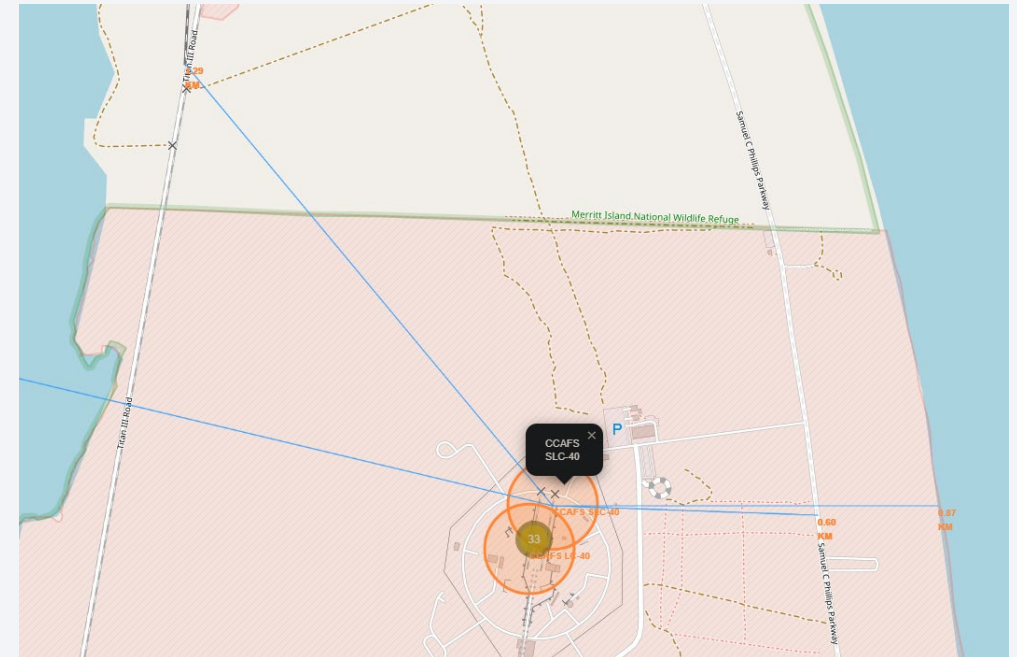
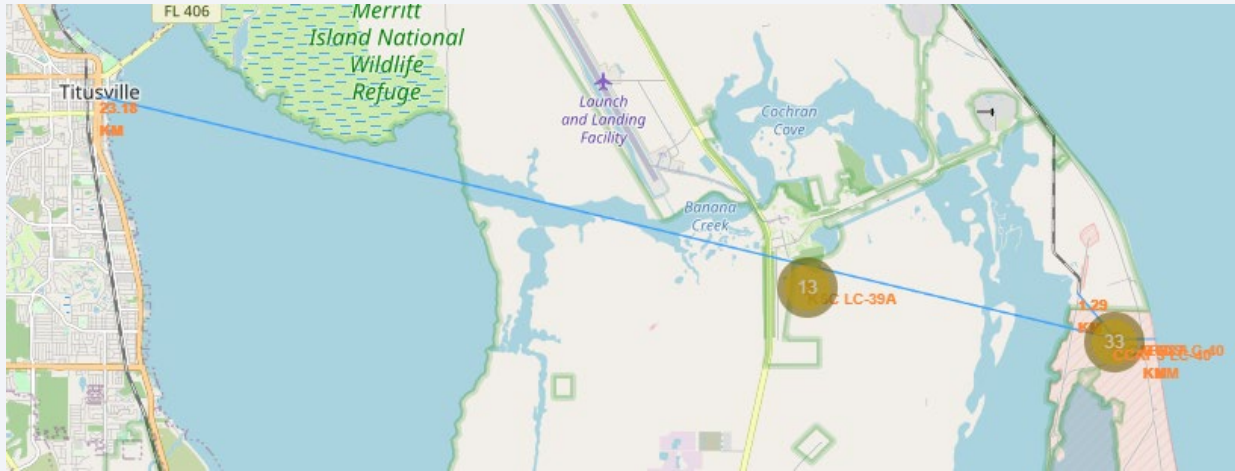


Green marker represents successful launches.

Red marker represents unsuccessful launches.

We note that KSC LC-39A has a higher launch success rate.

Folium Map – Distances between CCAFS SLC-40 and its proximities



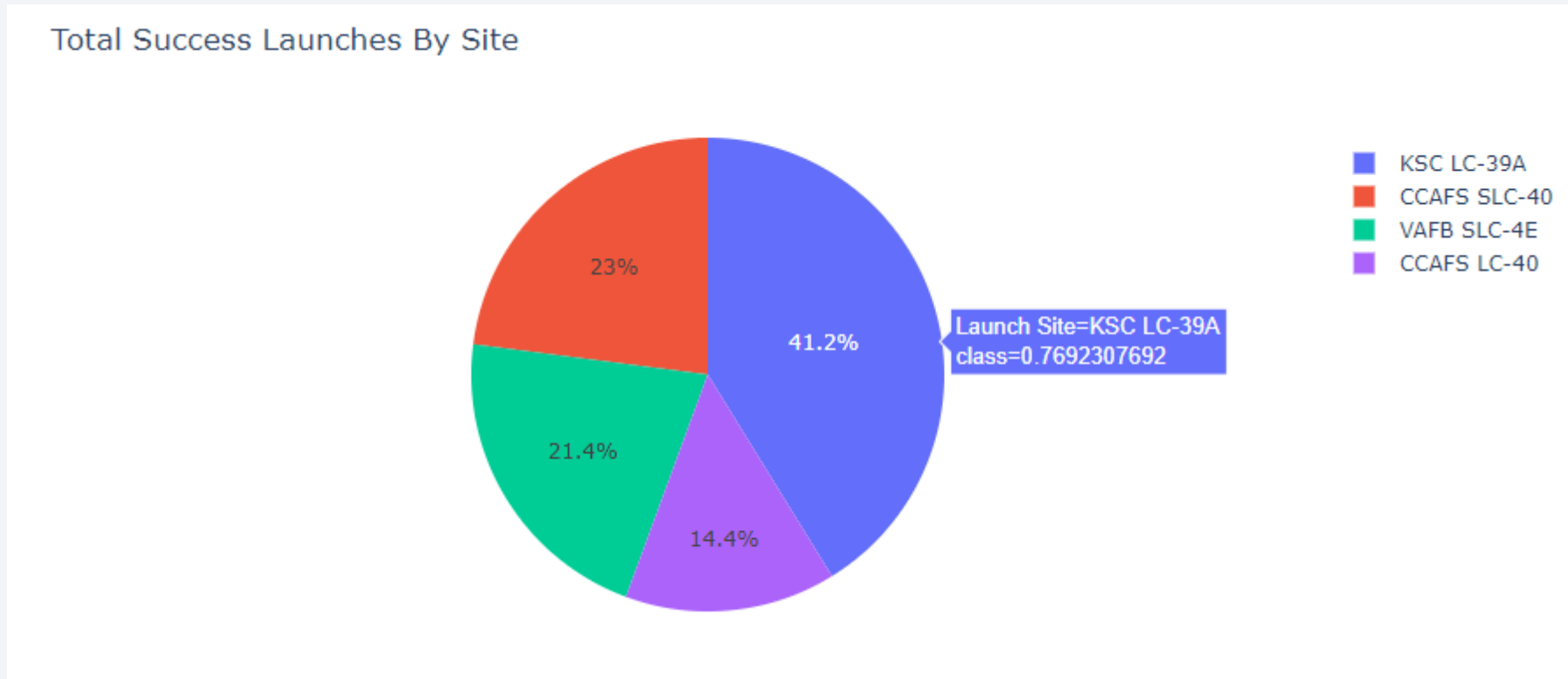
- Is CCAFS SLC-40 close to the railways ? Yes
- Is CCAFS SLC-40 close to the highways ? Yes
- Is CCAFS SLC-40 close to the coastline? Yes
- Do CCAFS SLC-40 keeps certain distance away from cities ? No



Section 4

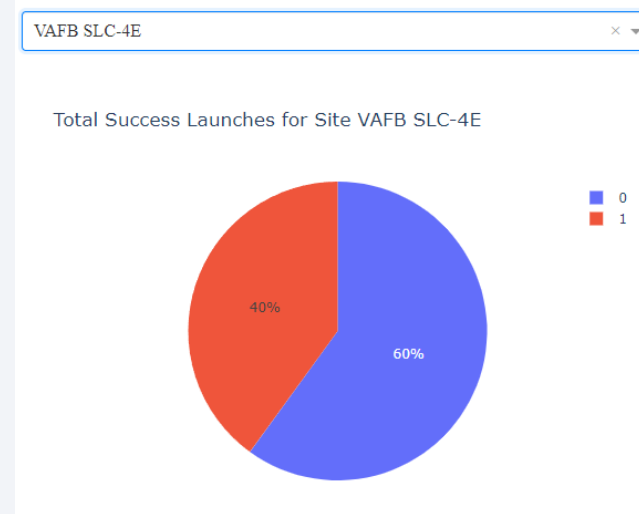
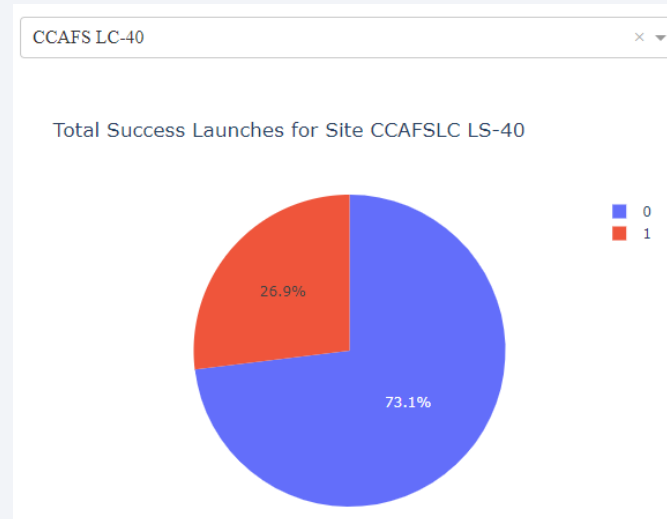
Build a Dashboard with Plotly Dash

Dashboard – Total success launches by Site

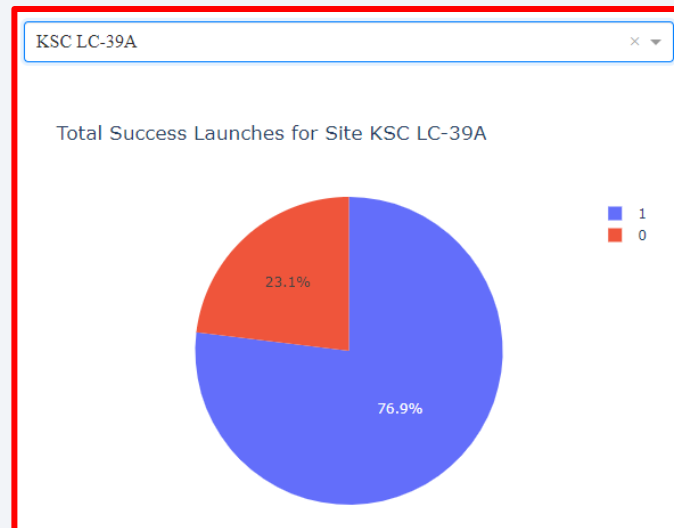


We see that KSC LC-39A has the best success rate of launches.

Dashboard – Total success launches on each Site



We see that KSC LC-39A has achieved a 76.9% success rate while getting a 23.1% failure rate



Dashboard – Payload mass vs Outcome for all sites with different payload mass selected



- We see a range of 2000 to 6000 kg of payload has the highest launch success rate.
- We see a range of over 6000 kg of payload has the lowest launch success rate.
- We see F9 Booster version “FT” has the highest launch success rate.

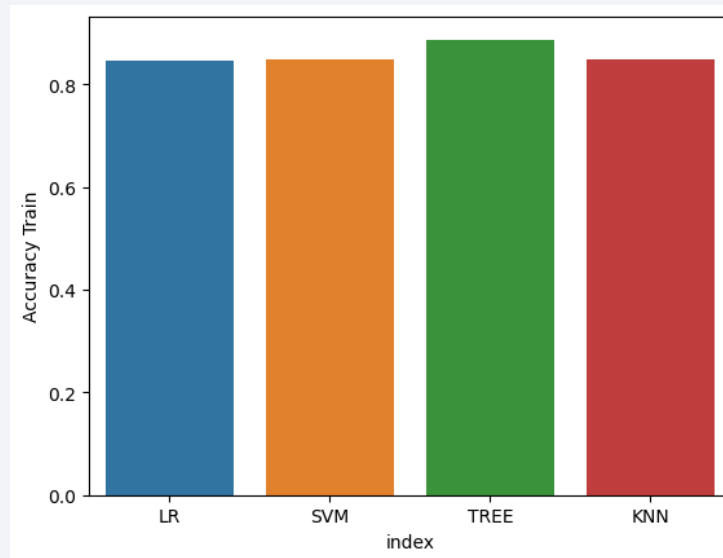
Section 5

Predictive Analysis (Classification)

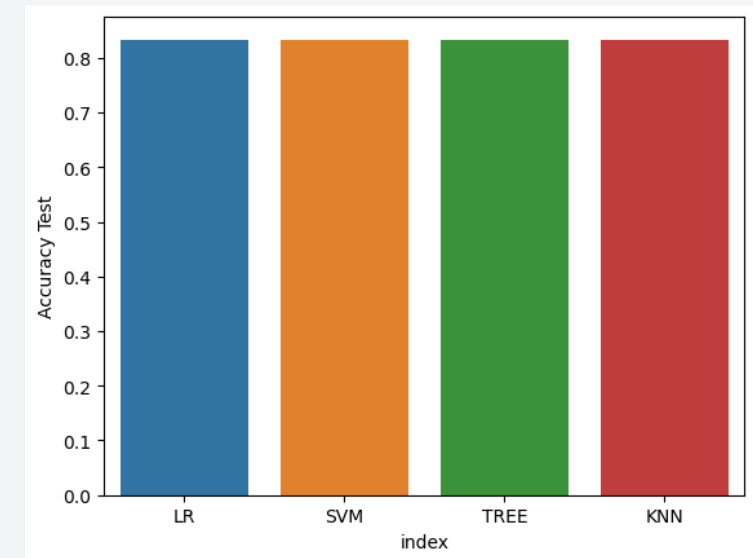
Classification Accuracy

	index	Accuracy Train	Accuracy Test
0	LR	0.846429	0.833333
1	SVM	0.848214	0.833333
2	TREE	0.887500	0.833333
3	KNN	0.848214	0.833333

Accuracy Train over each methods



Accuracy Test over each methods

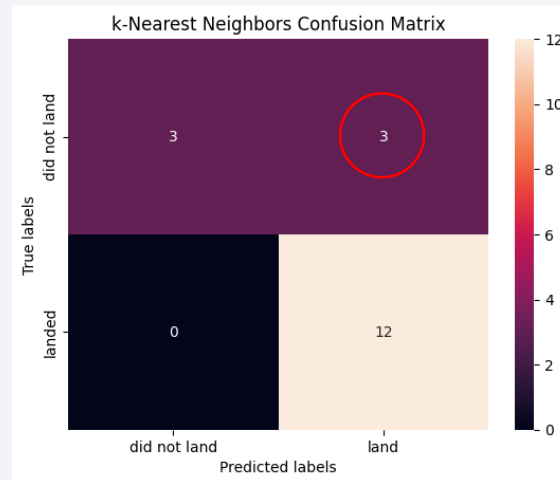
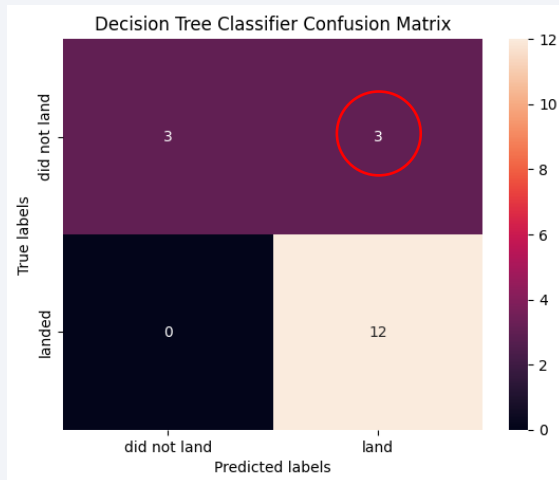
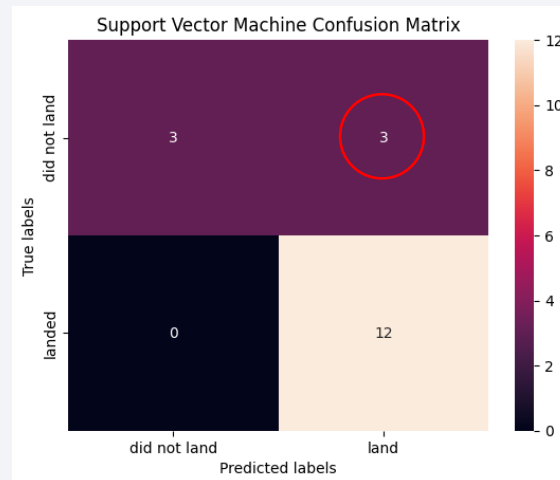
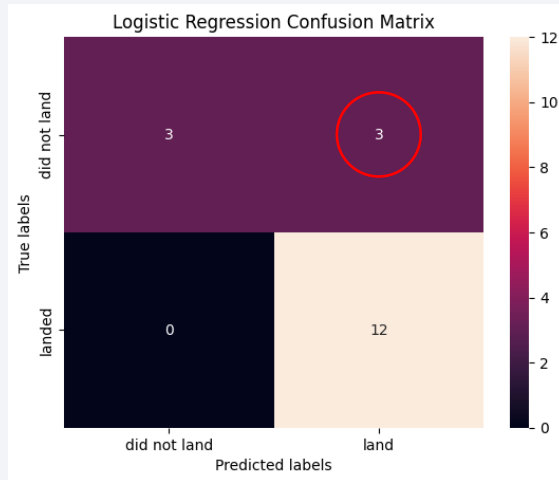


For accuracy test, all methods performed similar. We could get more test data to decide between them. But if we really need to choose one right now, we would like to take the decision tree.

Decision tree best parameters

```
tuned hyperparameters : (best parameters) {'criterion': 'gini', 'max_depth': 4,
'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitte
r': 'best'}
accuracy : 0.8875
```

Confusion Matrix



The objective is to minimize the number of False Negatives (FN) or False Positives (FP)

As the test accuracy score are all equal, the confusion matrices are also identical. Therefore, the best model could not be found.

However, the main problem of these models are **false positives**.

True	0	TN	FP
	1	FN	TP
		0	1
		Predict	

Conclusions

- The success of a mission can be explained by several factors such as the launch site, the orbit and especially the number of previous launches. Indeed, we can assume that there has been a gain in knowledge between launches that allowed to go from a launch failure to a success.
- The orbits with the best success rates are GEO, HEO, SSO, ES-L1.
- Depending on the orbits, the payload mass can be a criterion to take into account for the success of a mission. Some orbits require a light or heavy payload mass. But generally low weighted payloads perform better than the heavy weighted payloads.
- With the current data, we cannot explain why some launch sites are better than others (KSC LC-39A is the best launch site). To get an answer to this problem, we could obtain atmospheric or other relevant data.
- For this dataset, we choose the Decision Tree Algorithm as the best model even if the test accuracy between all the models used is identical. We choose Decision Tree Algorithm because it has a better train accuracy.

Thank you!

