IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Thiago Miranda Lima
17/12/2022

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- Project background and context
- Problems you want to find answers

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API
- Perform data wrangling
  - Data processing and cleaning
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Decision Tree, KNN, Linear Regression, SVM

Data Collection

- Spacex launch dataset from SpaceX REST API using GET Request
- Next, decode response using .json and turn into Pandas Dataframe using .json_normalize()
- we selected relevant columns and performed data treatment such as removing unnecessary records and adjusting data type and format
- we filter the dataframe to only include falcon 9 launch
- finally we treat the missing data

Data Collection – SpaceX API

- Spacex launch dataset from SpaceX REST API using GET Request
- The link to notebook is: [Data Collection]

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

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

Check the content of the response

```
print(response.content)
```

```
b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":f
l,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"o
rocket-lost-launch.html","wikipedia":"https://en.wikipedia.org/wiki/D
es":[{"time":33,"altitude":null,"reason":"merlin engine failure"}],"d
ight_number":1,"name":"FalconSat","date_utc":"2006-03-24T22:30:00.000
lse,"legs":false,"reused":false,"landing_attempt":false,"landing_succ
tempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":
l},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"htt
n.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":null,"static_fi
remature engine shutdown"}],"details":"Successful first stage burn an
psules":[],"payloads":["5eb0e4b6b6c3bb0006eeb1e2"],"launchpad":"5e9e4
n":"hour","upcoming":false,"cores":[{"core":"5e9e289ef35918416a3b2624
unch_library_id":null,"id":"5eb87cdaffd86e000604b32b"},{"fairings":{"
x.com/4a/80/k1oAkY0k_o.png"},"reddit":{"campaign":null,"launch":null,
e":"http://www.spacex.com/news/2013/02/11/falcon-1-flight-3-mission-s
da69955f709d1eb","success":false,"failures":[{"time":140,"altitude":3
```

Data Collection - Scraping

- first we use the GET method to get the page content and we use Beautiful Soup to do the parser.
- Then we extract the column names.
- Link of the Notebook: [jupyter-labs-webscraping]

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

In [4]:
```
site = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

In [6]:
```
soup = BeautifulSoup(site.text,'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

TASK 2: Extract all column/variable names from the HTML table header

In [8]:
```
html_tables = soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

In [ ]:
```
first_launch_table = html_tables[2]
```

Next, we just need to iterate through the `<th>` elements and apply the provided `extract_column_from_header()` to extract column name one by one

In [11]:
```
column_names = []

for col in first_launch_table.find_all('th'):
    if (col.text != '') and (len(col.text)>0):
        column_names.append(extract_column_from_header(col))
```

Data Wrangling

We calculate:

- the number of launches on each site
- the occurrence number of each orbit
- the occurrence number of mission outcome per orbit type
- and finally we create the landing outcome from Outcome columnAdd

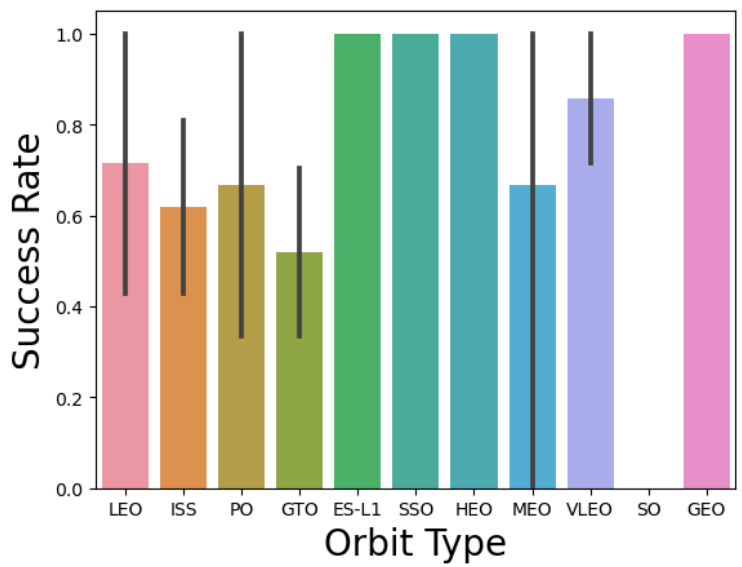
Link to notebook: Data Wrangling

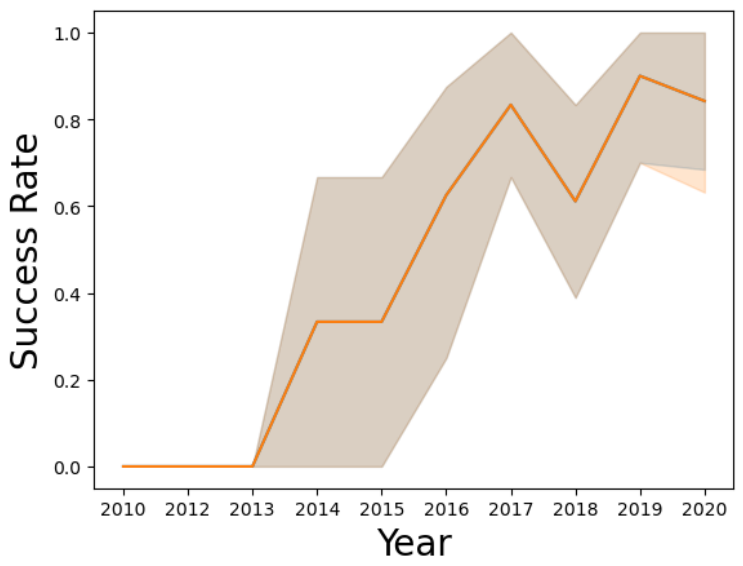
EDA with Data Visualization

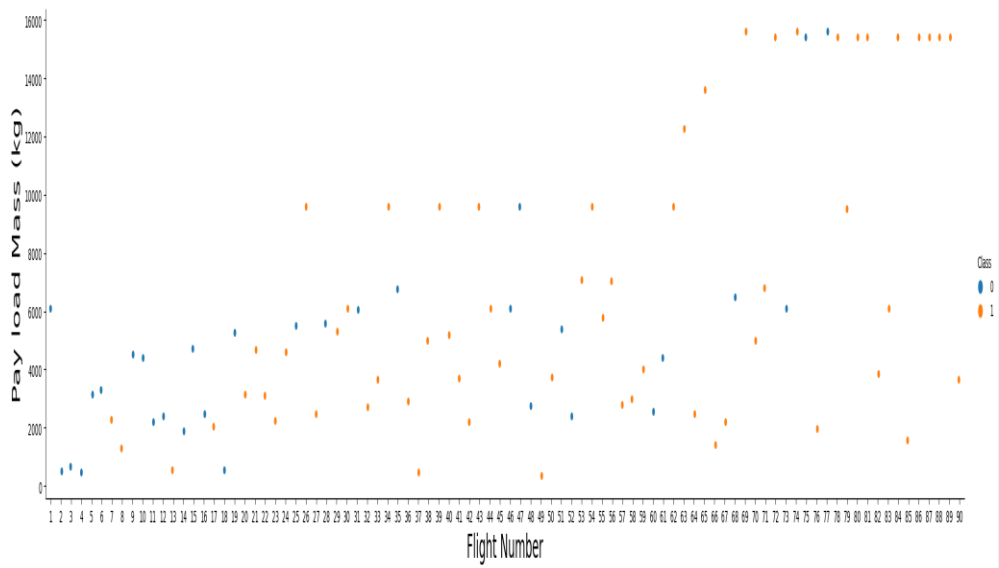
We visualize the following relationships:

- Flight Number and Launch Site, Payload Mass and Launch Site, Orbit Type and Success Rate, Flight Number and Orbit Type, Payload and Orbit Type, Launch Success yearly trend and finally we create dummy variables to categorical columns

The link to the notebook with all the graphics is here: DataViz







EDA with SQL

We loaded the SpaceX dataset into sqlite3 into jupyter notebook and applied EDA with sql to get insight from the data. We view information such as:

- The names of unique launch sites in the space mission.
- The total payload mass carried by boosters launched by NASA (CRS)
- The average payload mass carried by booster version F9 v1.1
- The total number of successful and failure mission outcomes
- The failed landing outcomes in drone ship, their booster version and launch site names.

Link to notebook here: Labs EDA SQL

Build an Interactive Map with Folium

We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.

Assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.

Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.

We calculated the distances between a launch site to its proximities. We answered some question for instance:

- Are launch sites near railways, highways and coastlines.
- Do launch sites keep certain distance away from cities.

Link: Generating Maps in Python

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- Link to notebook: DashBoard

Predictive Analysis (Classification)

We Create a Numpy array from the column Class applying the method to_numpy() and assign to Y variable.

Standardize the X and divide the sample in train and test using tain_test_split function pandas

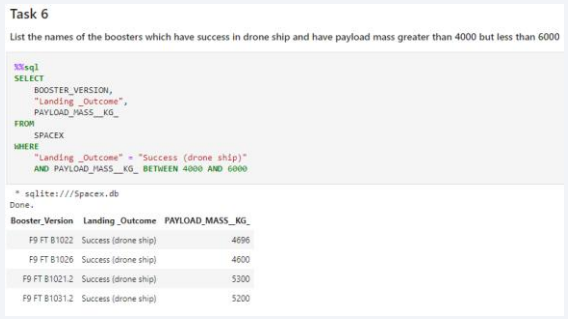
We analyzed the data, adjusted the hyperparameters and created the confusion matrix and calculate accuracy for the following learning models:

- Linear Regression
- Decision Tree
- SVM
- KNN

Link here: Machine Learning

Results

- Exploratory data analysis results



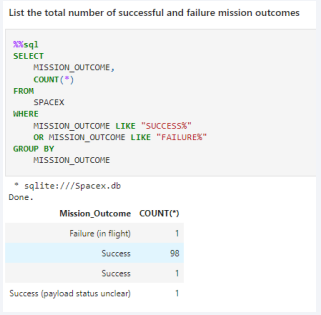
Task 6

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,
    "Landing _Outcome",
    PAYLOAD_MASS__KG_
FROM
    SPACEX
WHERE
    "Landing _Outcome" = "Success (drone ship)"
    AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

```
 * sqlite:///Spacex.db
Done.
```

Booster_Version	Landing _Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200



List the total number of successful and failure mission outcomes

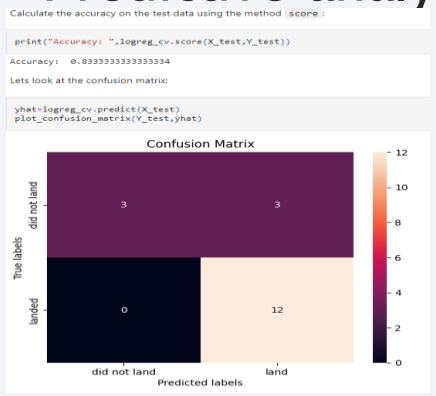
```
%%sql
SELECT
    MISSION_OUTCOME,
    COUNT(*)
FROM
    SPACEX
WHERE
    MISSION_OUTCOME LIKE "SUCCESS%"
    OR MISSION_OUTCOME LIKE "FAILURE%"
GROUP BY
    MISSION_OUTCOME
```

```
 * sqlite:///Spacex.db
Done.
```

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

- Interactive analytics demo in screenshots

- Predictive analysis results



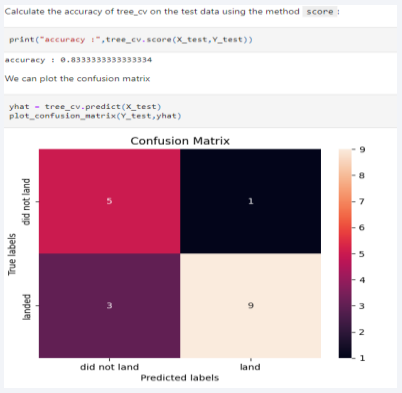
Calculate the accuracy on the test data using the method `score`:

```
print("Accuracy: ",logreg_cv.score(X_test,Y_test))
```

```
Accuracy:  0.8333333333333334
```

Lets look at the confusion matrix:

```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

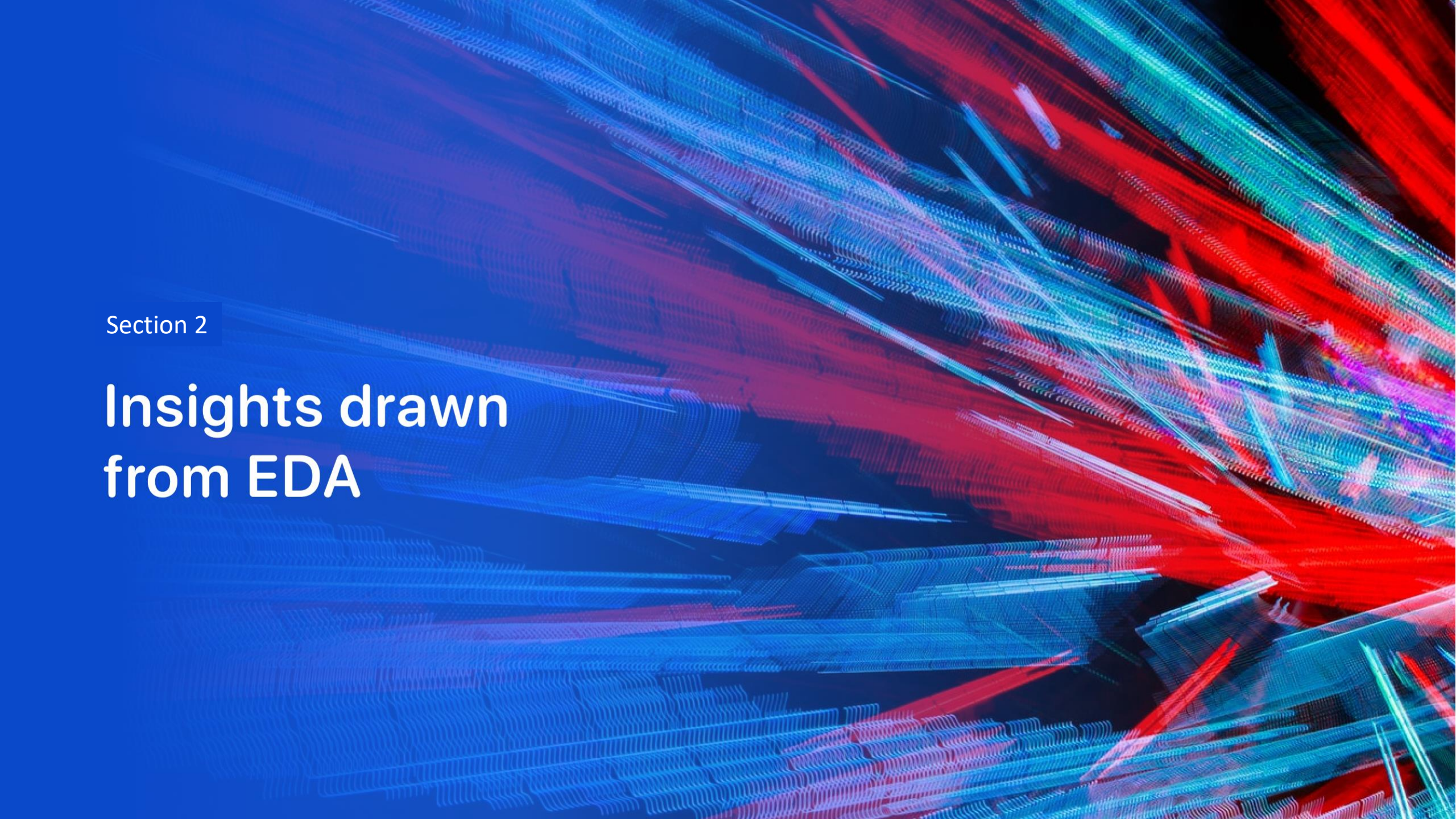
Calculate the accuracy of tree_cv on the test data using the method `score`:

```
print("accuracy :",tree_cv.score(X_test,Y_test))
```

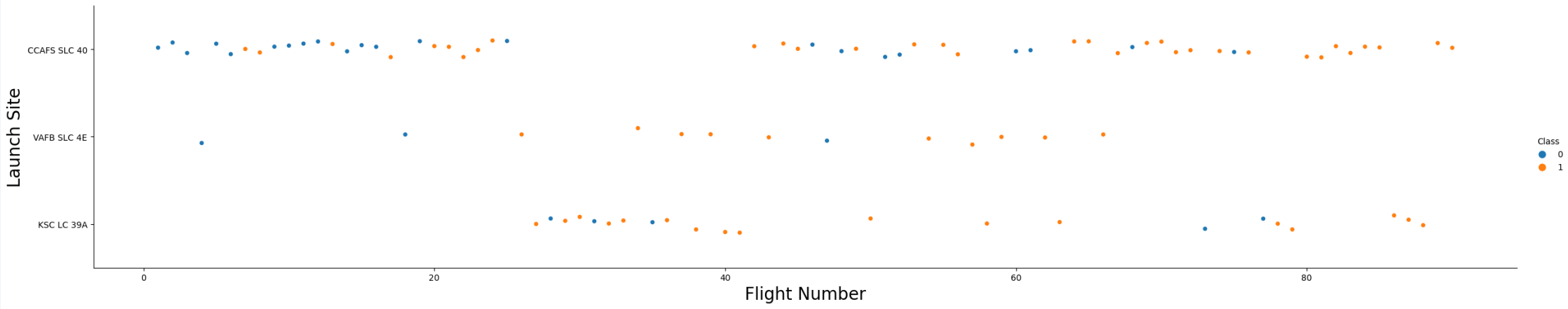
```
accuracy : 0.8333333333333334
```

We can plot the confusion matrix

```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

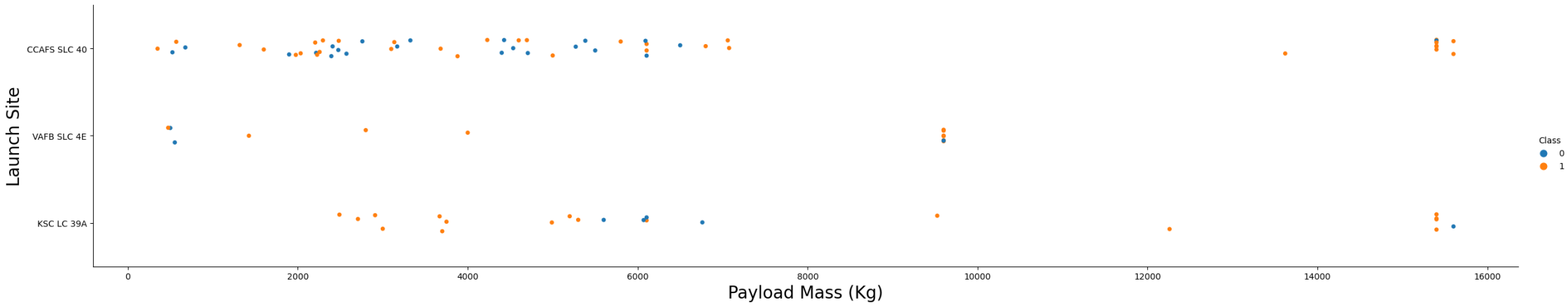

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



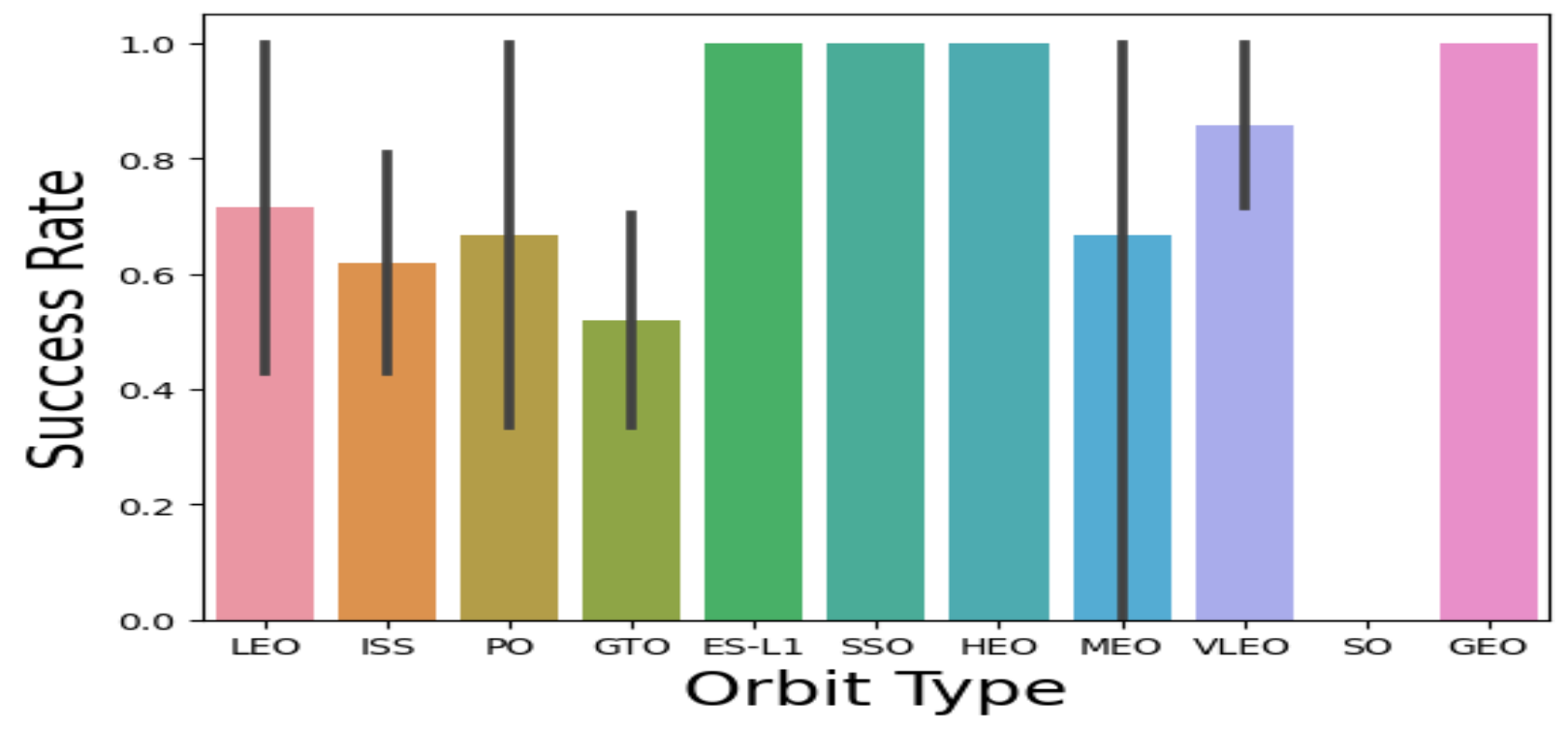
- with the increase in the number of flights the success rate increases

Payload vs. Launch Site



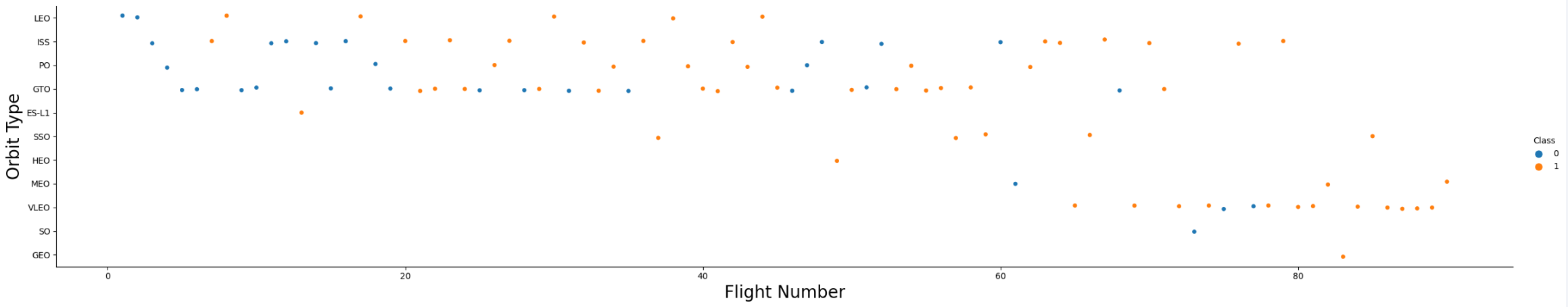
- VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000)

Success Rate vs. Orbit Type



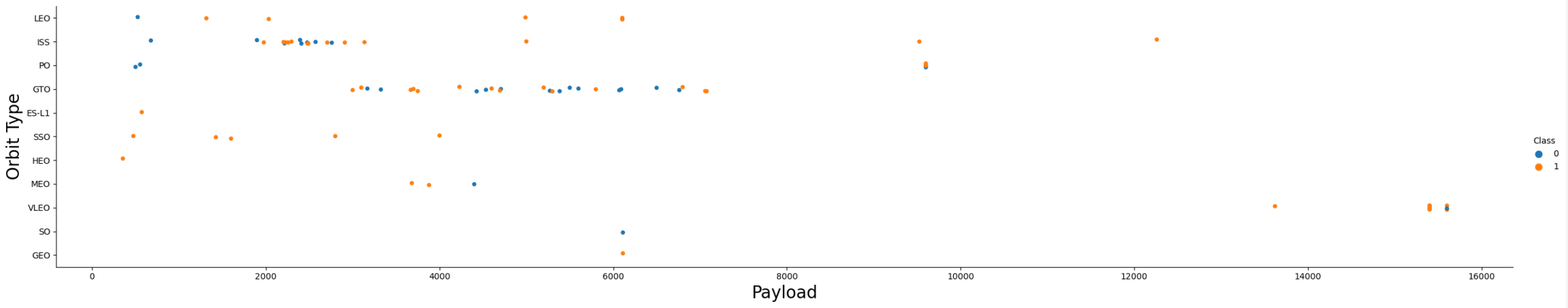
Some Orbit Type are more successful than others

Flight Number vs. Orbit Type



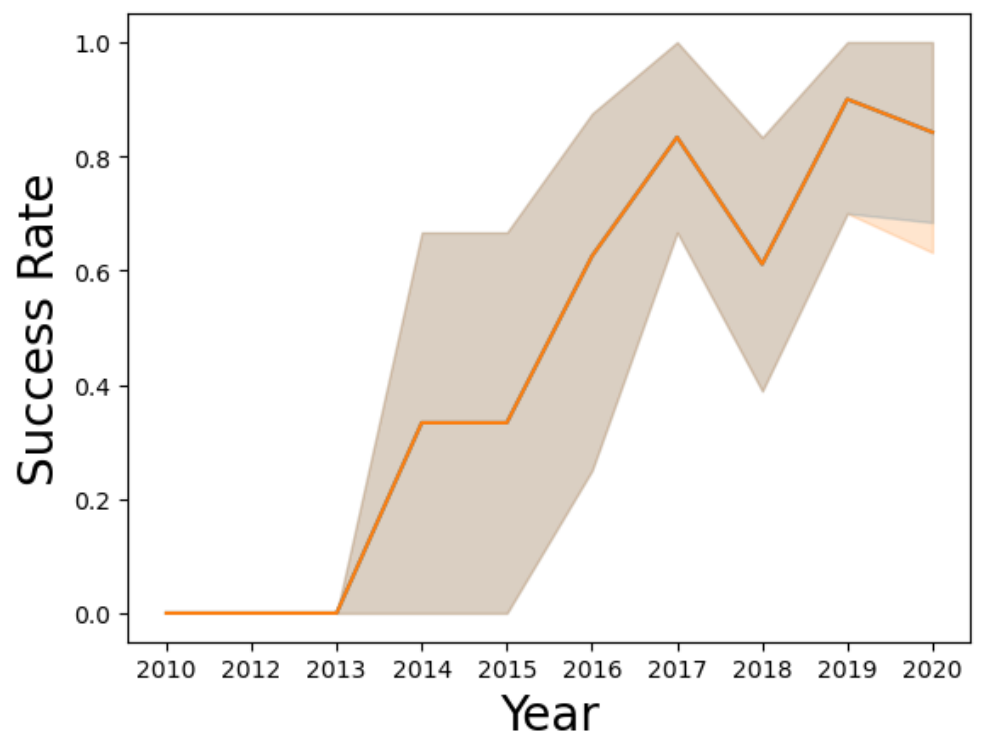
- in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit

Payload vs. Orbit Type



- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here

Launch Success Yearly Trend



The success rate since 2013 kept increasing till 2020

All Launch Site Names

```
%%sql
SELECT
    DISTINCT(LAUNCH_SITE)
FROM
    SPACEX
```

```
 * sqlite:///Spacex.db
Done.
```

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

We select the distinct values of the Launch Site column

Launch Site Names Begin with 'CCA'

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

```
%%sql
SELECT
    LAUNCH_SITE
FROM
    SPACEX
WHERE
    LAUNCH_SITE LIKE "CCA%"
LIMIT 5
```

```
 * sqlite:///Spacex.db
Done.
```

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

We select the rows that start with CCA and limit the results to 5

Total Payload Mass

```
%%sql
SELECT
    CUSTOMER,
    SUM(PAYLOAD_MASS__KG_) TOTAL
FROM
    SPACEX
WHERE
    CUSTOMER = "NASA (CRS)"
GROUP BY
    CUSTOMER
```

```
 * sqlite:///Spacex.db
Done.
```

Customer	TOTAL
NASA (CRS)	45596

We selected the customer and aggregate by sum of payload mass where customer equals Nasa (CRS) for this, grouped by customer

Average Payload Mass by F9 v1.1

```
%%sql
SELECT
    BOOSTER_VERSION,
    AVG(PAYLOAD_MASS__KG_) MEAN
FROM
    SPACEX
WHERE
    BOOSTER_VERSION = "F9 v1.1"
GROUP BY
    BOOSTER_VERSION
```

```
 * sqlite:///Spacex.db
Done.
```

Booster_Version	MEAN
F9 v1.1	2928.4

Selected Booster Version and average payload mass where Booster version is F9 V1.1 and group by Booster version

First Successful Ground Landing Date

```
%%sql
SELECT
    "Landing _Outcome",
    MIN(DATE) FIRST
FROM
    SPACEX
WHERE
    "Landing _Outcome" LIKE "SUCCESS%"
```

```
 * sqlite:///Spacex.db
Done.
```

Landing _Outcome	FIRST
Success (ground pad)	01-05-2017

Select Landing Outcome and first date where landing outcome is like Success

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql
SELECT
    BOOSTER_VERSION,
    "Landing _Outcome",
    PAYLOAD_MASS__KG_
FROM
    SPACEX
WHERE
    "Landing _Outcome" = "Success (drone ship)"
    AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

```
 * sqlite:///Spacex.db
Done.
```

Booster_Version	Landing _Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

Selected Booster Version, Landing Outcome and Payload Mass where Landing Outcome is Success (Drone Ship) and payload mas is between 4000 and 6000 kg

Total Number of Successful and Failure Mission Outcomes

```
%%sql
SELECT
    MISSION_OUTCOME,
    COUNT(*)
FROM
    SPACEX
WHERE
    MISSION_OUTCOME LIKE "SUCCESS%"
    OR MISSION_OUTCOME LIKE "FAILURE%"
GROUP BY
    MISSION_OUTCOME
```

```
 * sqlite:///Spacex.db
Done.
```

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

Selected Mission Outcome and count of rows where Mission outcome like Success or Mission Outcome like Failure, group by mission outcom

Boosters Carried Maximum Payload

```
%%sql
SELECT
    DISTINCT(BOOSTER_VERSION),
    PAYLOAD_MASS__KG_
FROM
    SPACEX
WHERE
    PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX)
```

```
 * sqlite:///Spacex.db
Done.
```

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

- Selected the distinct Booster version where the mass equal the maximum mass of sample

2015 Launch Records

```
%%sql
SELECT
    DATE,
    "Landing _Outcome" ,
    BOOSTER_VERSION,
    LAUNCH_SITE
FROM
    SPACEX
WHERE
    "Landing _Outcome" = "Failure (drone ship)"
    AND YEAR(DATE) = '2015'
```

- Selected Date, Landing Outcome, Booster Version and Launch Site where landing outcome = Failure (Drone Ship) and Year equals 2015

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

```
%%sql
SELECT
    DATE,
    "Landing _Outcome",
    COUNT("Landing _Outcome") QTD
FROM
    SPACEX
WHERE
    ("Landing _Outcome" = "Failure (drone ship)" OR "Landing _Outcome" = "Success (ground pad)")
    AND DATE BETWEEN "04/06/2010" AND "20/03/2017"
GROUP BY
    DATE,"Landing _Outcome"
ORDER BY DATE DESC
```

- We selected the columns and to do conditions where Landing Outcome = Failure or success in the range date, group by Date and Landing outcome and order by date Descending

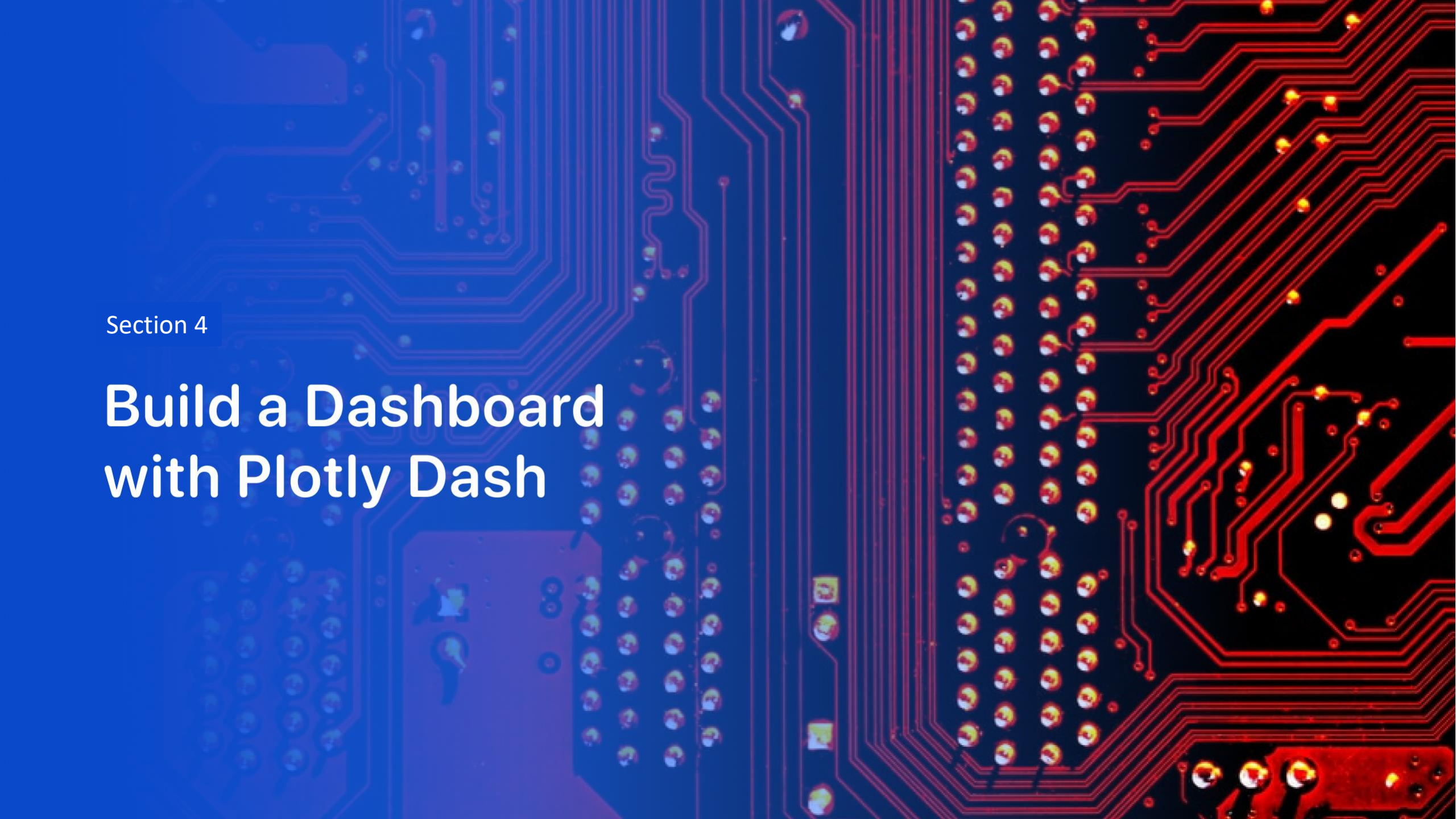
Section 3

Launch Sites Proximities Analysis

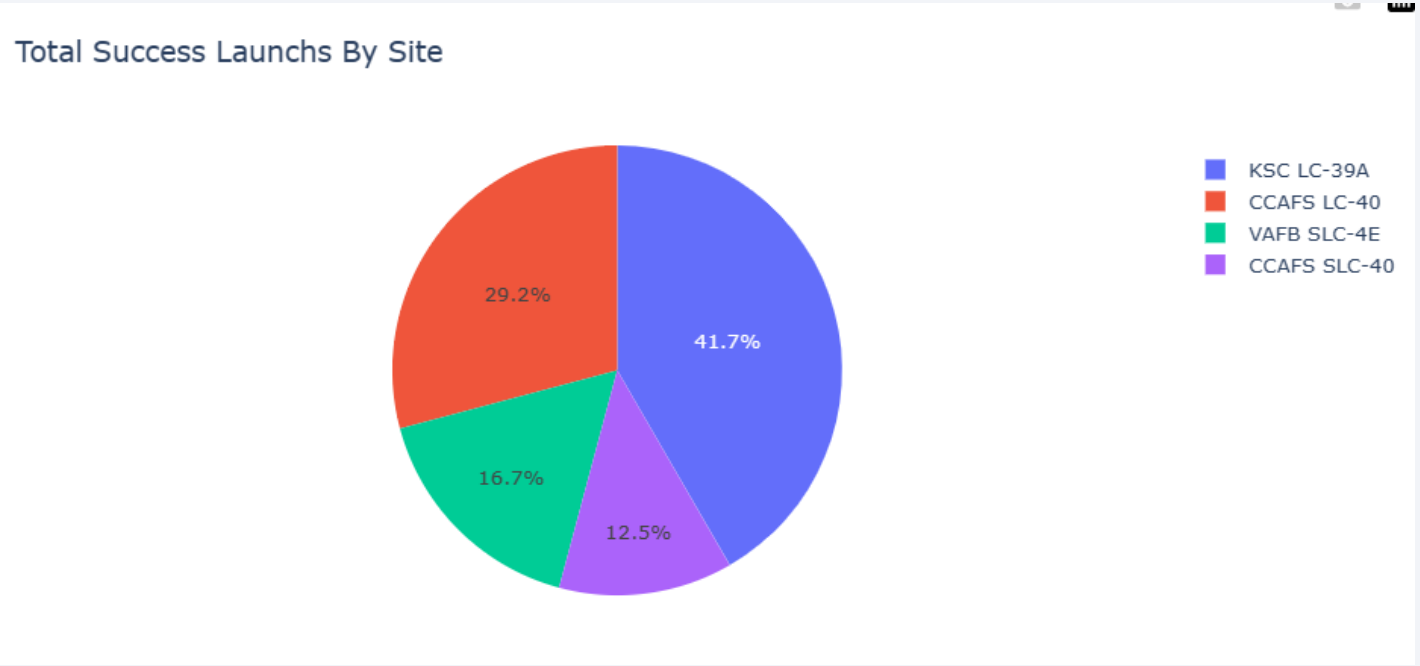
- The Launch Site is located in USA Florida and California

Markers with colors

<Folium Map Screenshot 3>

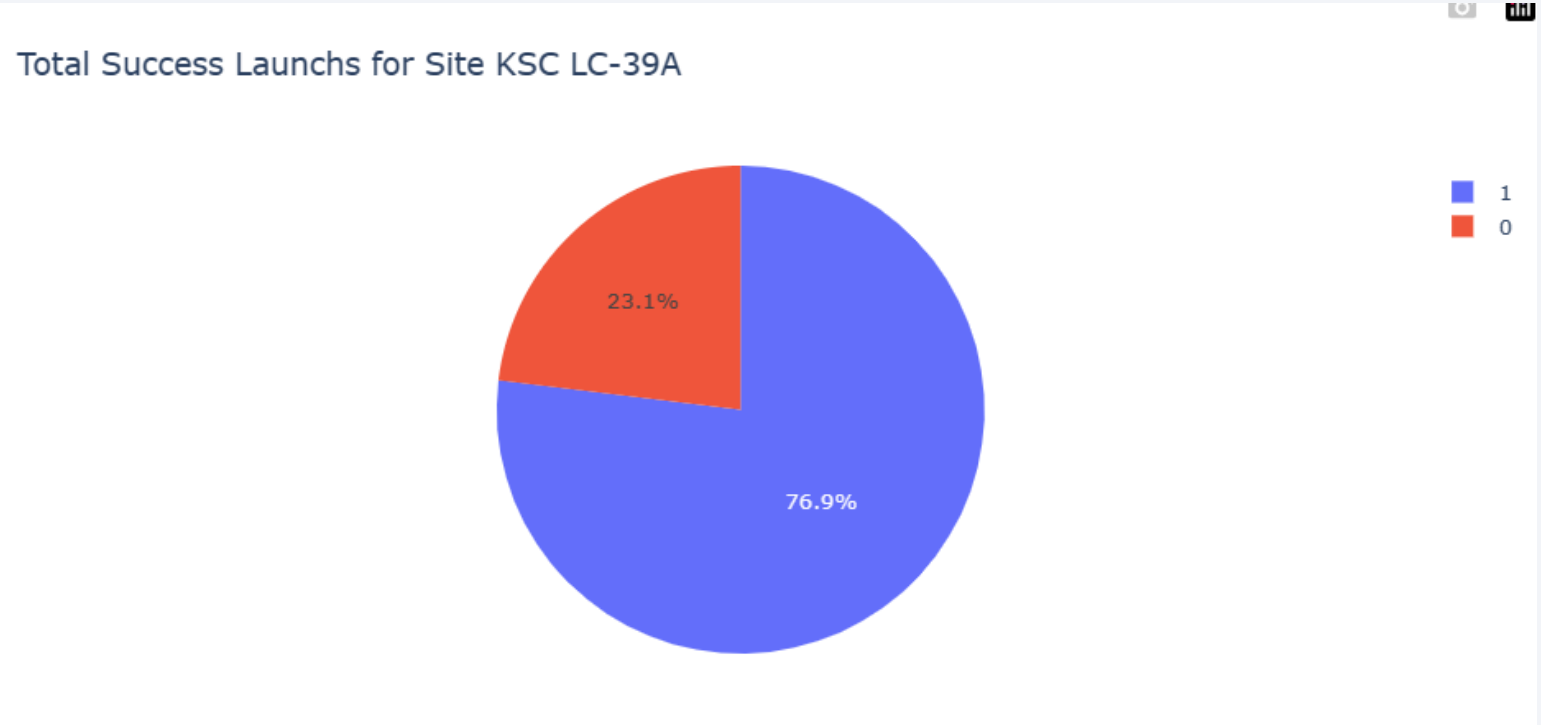

Section 4
Build a Dashboard with Plotly Dash

Total Success Launchs By Site



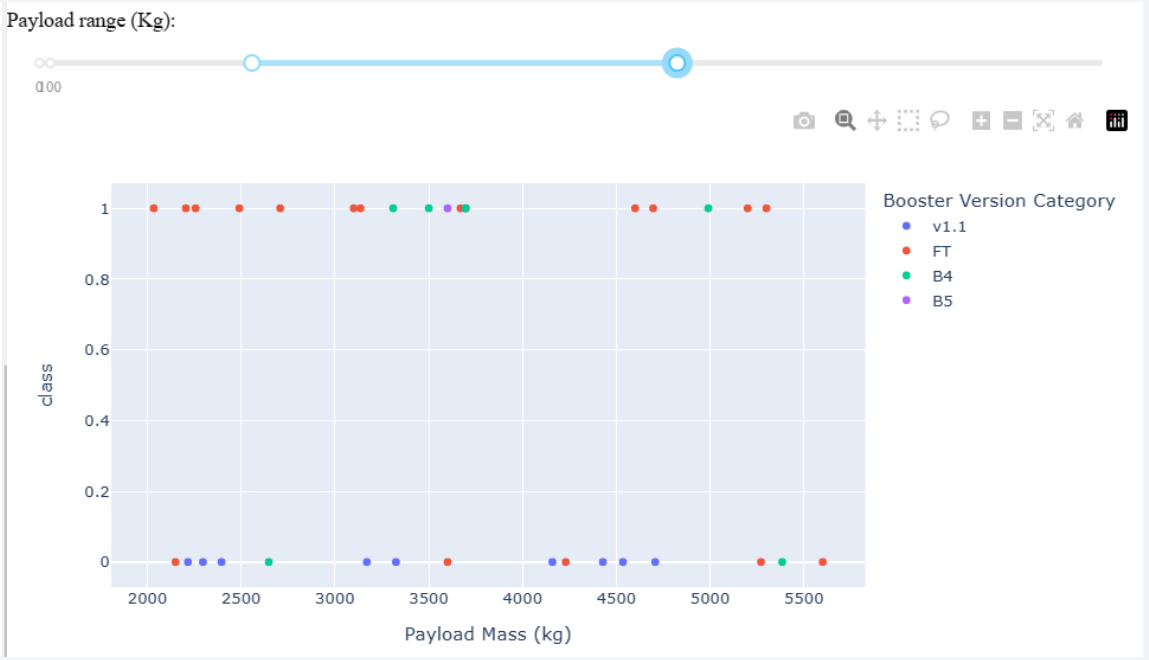
KSC LC-39A is the Site with highest launch success

Total Success Launches for KSC LC-39A



The site has success in 76,9% of launches

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

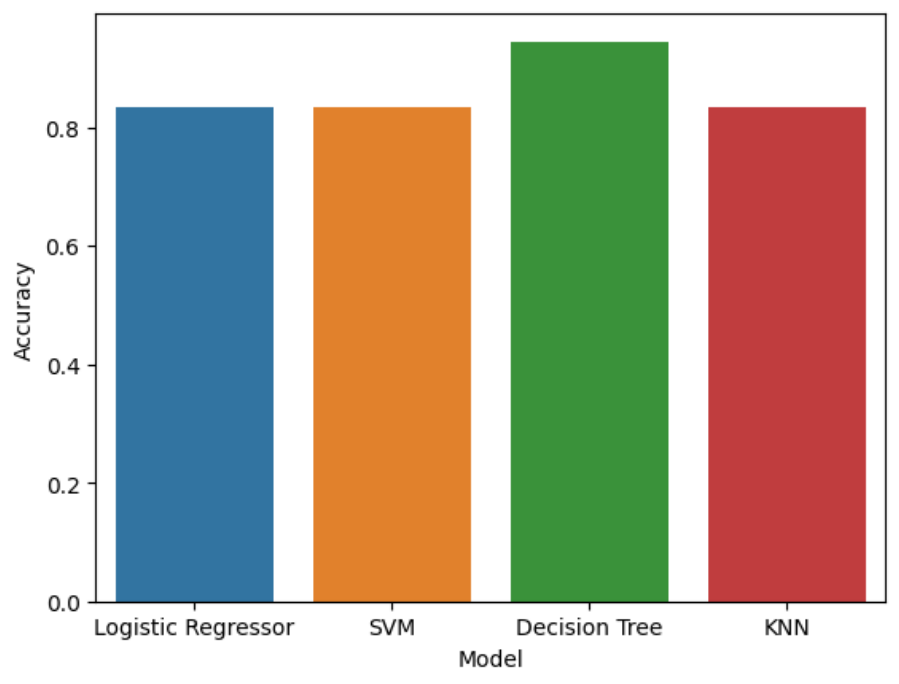


FT has more success with low weight

Section 5

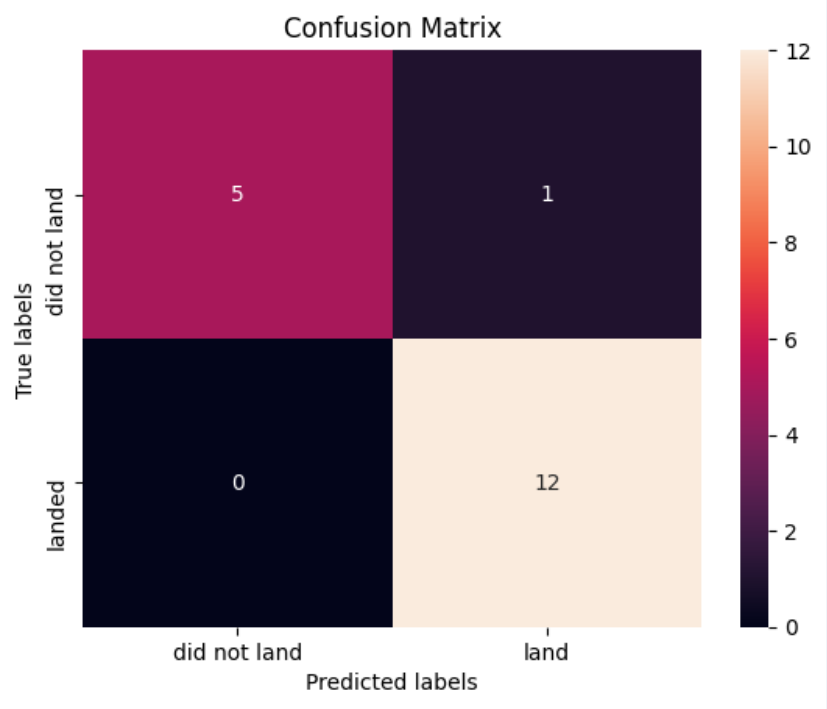
Predictive Analysis (Classification)

Classification Accuracy



The highest classification accuracy is Decision Tree

Confusion Matrix



- The model presents the lower False Positive

Conclusions

- Point 1
- Point 2
- Point 3
- Point 4
- …

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!