IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ting Singley
November 2, 2023

Outline

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

Executive Summary

Summary of methodologies

- Data collection
- Perform data wrangling
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Summary of all results

Decision Tree indicates best accuracy on the training dataset. However, all models give a 83% accuracy on the test dataset.

	best_score_train	score_test
LR	0.846429	0.833333
SVM	0.848214	0.833333
TREE	0.916071	0.833333
KNN	0.848214	0.833333

Introduction

The new company Space Y aims to make space travel affordable for everyone. One big barrier for Space Y to enter the market is the high costs estimated166% more comparing to its competitor Space X each launch.

The object of this project is to gather information about SpaceX. With the historical information of SpaceX available to public, we can utilize Python machine learning models to predict if future first stage will land successfully and if it can reuse the first stage and therefore we can determine the price of each launch

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Python requests library allows us to make HTTP requests which extract the information from various website
- Perform data wrangling
  - After information was extracted from the internet, data was read into a Pandas DataFrame. Various methods such as describe(), dtypes, isnull(), etc were used to get data ready for analysis.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Besides Pandas DataFrame, SQL were also used to narrow down what kind of information need to be extracted to perform analysis. Visualization tools such as pie chart, scatterplot, barplot, etc. were also used to gain more insight of the data
- Perform interactive visual analytics using Folium and Plotly Dash
  - Folium is another visualization tool which plots the launch sites on the map and Plotly Dash publishes an interactive visualization for the audients.
- Perform predictive analysis using classification models
  - Skilean, a machine learning library offers various models to predict future unknow outcome. Historical data are used to train the models. Historical data can be spitted in between training dataset (train the models) and testing dataset (to test how close the model can predict the result accurately). The accuracy of each models can be estimated. Based on the result, we find a model that performs the best

Data Collection

The data were collected from the below websites

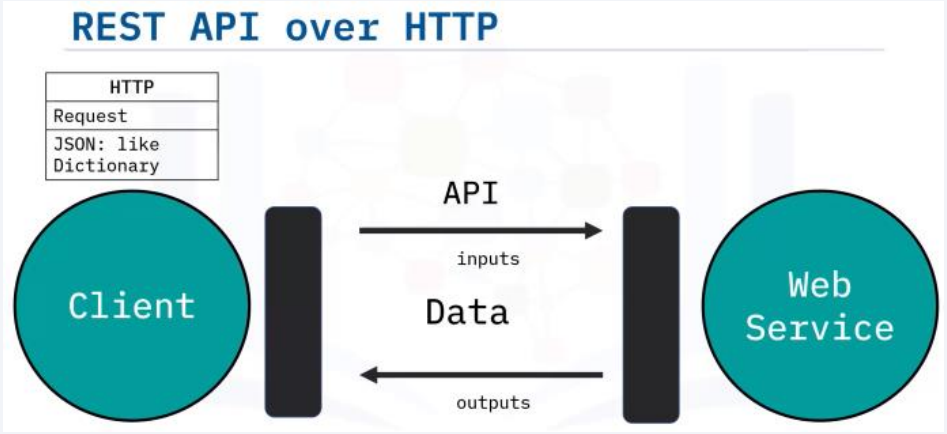
https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches_(2010%E2%80%932019)

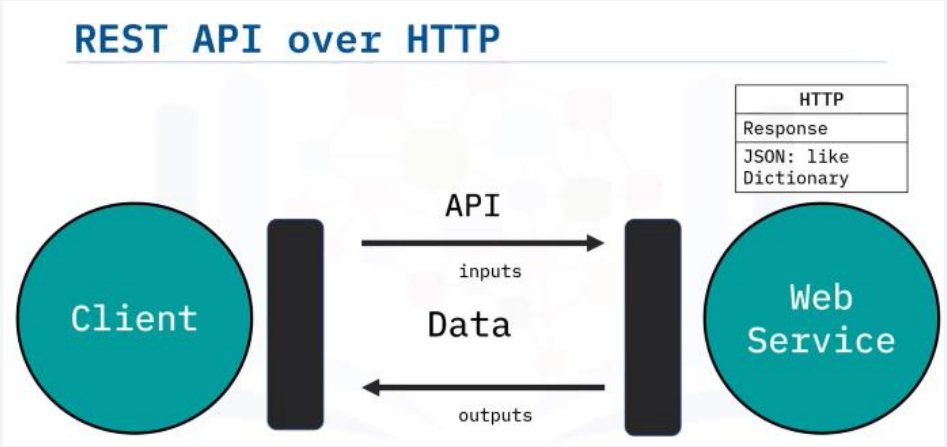
https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

An assessment such as descriptive statistics and visualization was applied to the dataset to assess the content, quality and initial insights about the data.

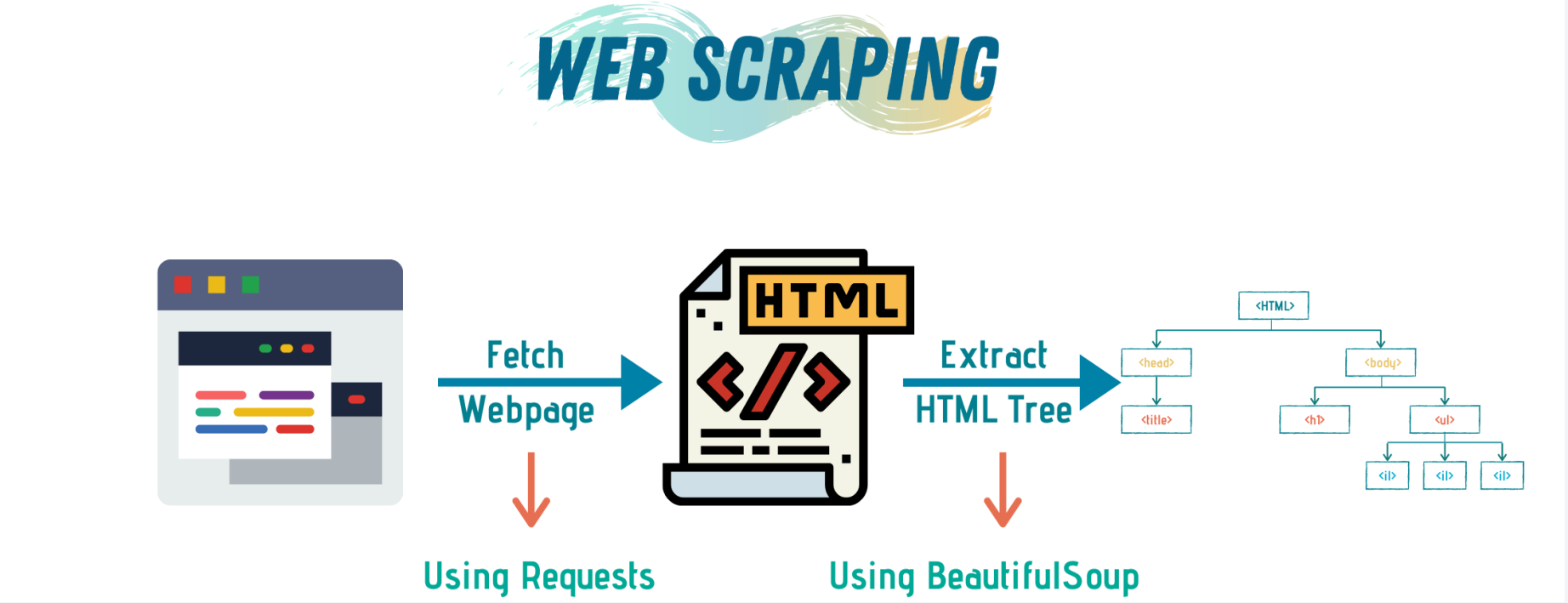
Data Collection – SpaceX API

- Python requests library make HTTP requests to get data from an API
- Takes the dataset and uses the rocket column to call the API and append the data to the list
- Takes the dataset and uses the launchpad column to call the API and append the data to the list
- Takes the dataset and uses the payloads column to call the API and append the data to the lists
- Takes the dataset and uses the cores column to call the API and append the data to the lists
- Requests rocket launch data from SpaceX API
- https://github.com/tingsingley/Data-Science-Capstone-Project/blob/main/1%20Collecting%20the%20data.ipynb





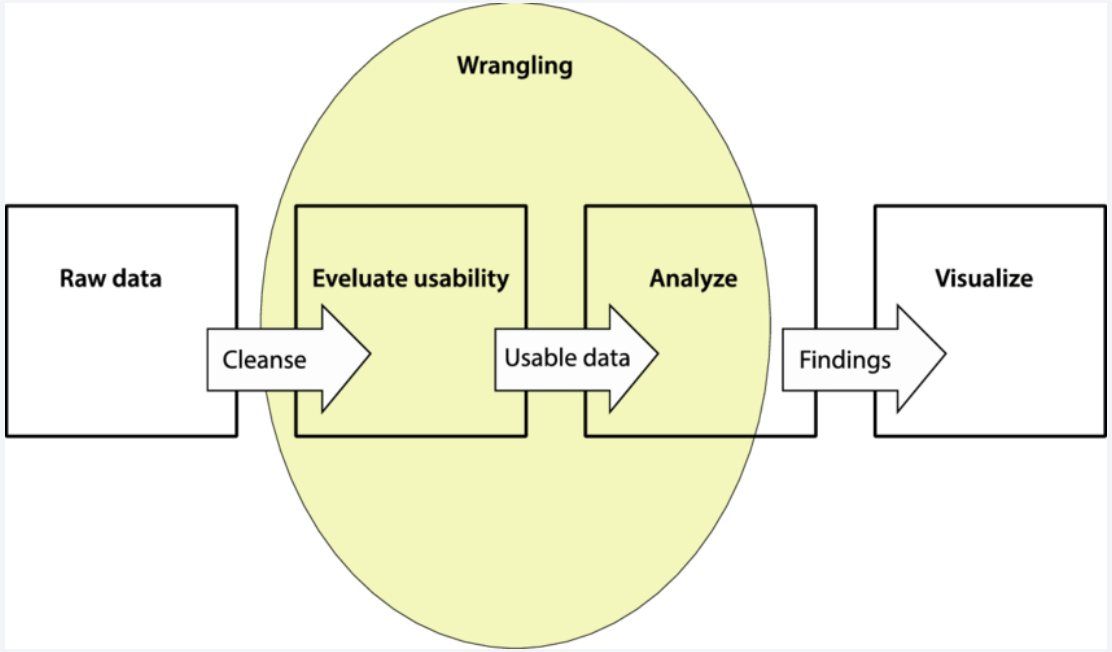
Data Collection - Scraping



- https://github.com/tingsingley/Data-Science-Capstone-Project/blob/main/2%20Web%20scraping%20Falcon%209%20and%20Falcon%20Heavy%20Launches%20Records%20from%20Wikipedia.ipynb

Data Wrangling

- Dataset was read into a Pandas DataFrame
- Various methods offer by Pandas to evaluate usability of the data or obtain more insight of the data (e.g. isnull(), dtypes, describe())
  - Identify if there is any missing values in the dataset
  - Identify data types of each columns
  - Provide statistical insight
- https://github.com/tingsingley/Data-Science-Capstone-Project/blob/main/3%20Data%20wrangling.ipynb



EDA with Data Visualization

- In order to see and understand the trends, outliers and patterns in the data, the following data visualization techniques were utilized. For example
  - Scatterplt: examine if there is a relationship between flight number and payload mass or flight number and launch site or payload mass and lunch site
  - Barplot: examine if there is any relationship between the success rate and orbit type and the magnitude
  - Lineplot: examine frequency of the successful rate in each year
- https://github.com/tingsingley/Data-Science-Capstone-Project/blob/main/5%20Exploring%20and%20Preparing%20Data.ipynb

EDA with 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***
- https://github.com/tingsingley/Data-Science-Capstone-Project/blob/main/4%20SQL%20Notebook.ipynb

Build an Interactive Map with Folium

Various map objects and purpose of each is summarized below:

- Map: to initiate a map
- Marker: create an icon as a text label
- Circle: add highlighted circle area on locations specified by the latitude and longitude
- Popup: a popup label showing name of the location when hoop over
- Polyline: to add a line by connecting the locations specified in the Marker object
- https://github.com/tingsingley/Data-Science-Capstone-Project/blob/main/6%20Launch%20Sites%20Locations%20Analysis%20with%20Folium.ipynb

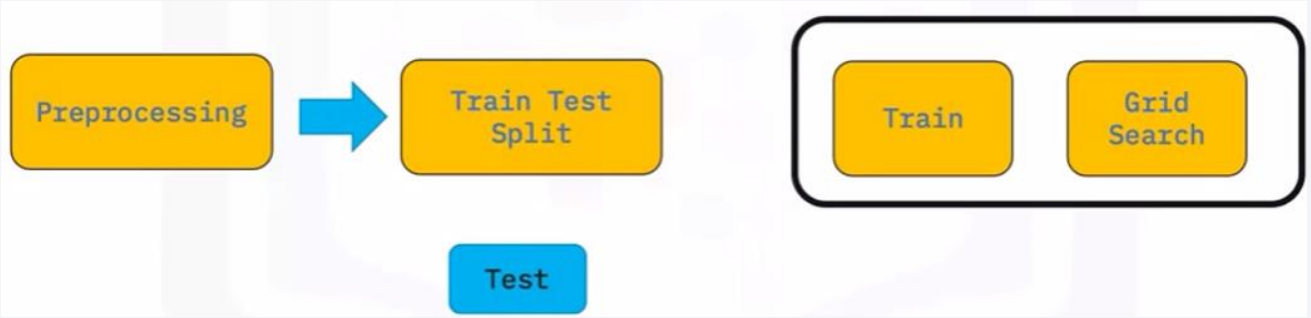
Build a Dashboard with Plotly Dash

Choices of selecting either all launch sites or an individual site is available to display an interactive pie chart, a slider and a scatter chart

- Pie chart for all sites tells the successful rate between those launches sites
- Pie chart for each site displays the successful/failure rate of that specific site
- Scatter plot shows the relationship between the count of payload mass and successful/failure rate
- With different range of the payload mass, scatterplot displays the changes accordingly
- https://github.com/tingsingley/Data-Science-Capstone-Project/blob/main/7%20spacex_dash_app.py

Predictive Analysis (Classification)

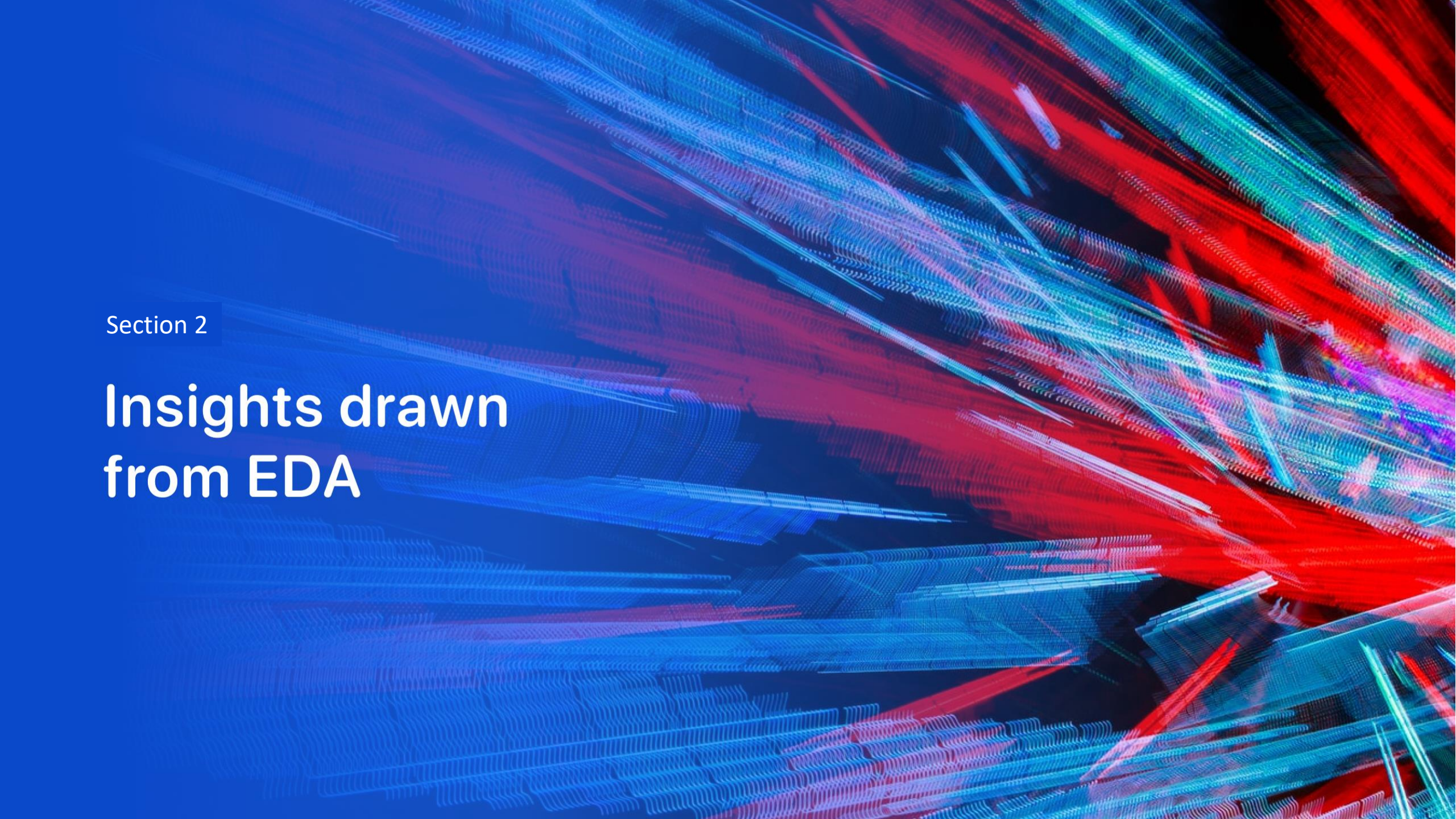
- The procedures of the predictive analysis included preprocessing the data and then split the data into training and testing dataset. We use training dataset to train the model and perform Grid Search on different models to find the best hyperparameter values. . Models used in this project included Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbors. Finally, apply the model on the test dataset and use score method to calculate the accuracy and visualize the result on Confusion Matrix



- https://github.com/tingsingley/Data-Science-Capstone-Project/blob/main/8%20Machine%20Learning%20Prediction.ipynb

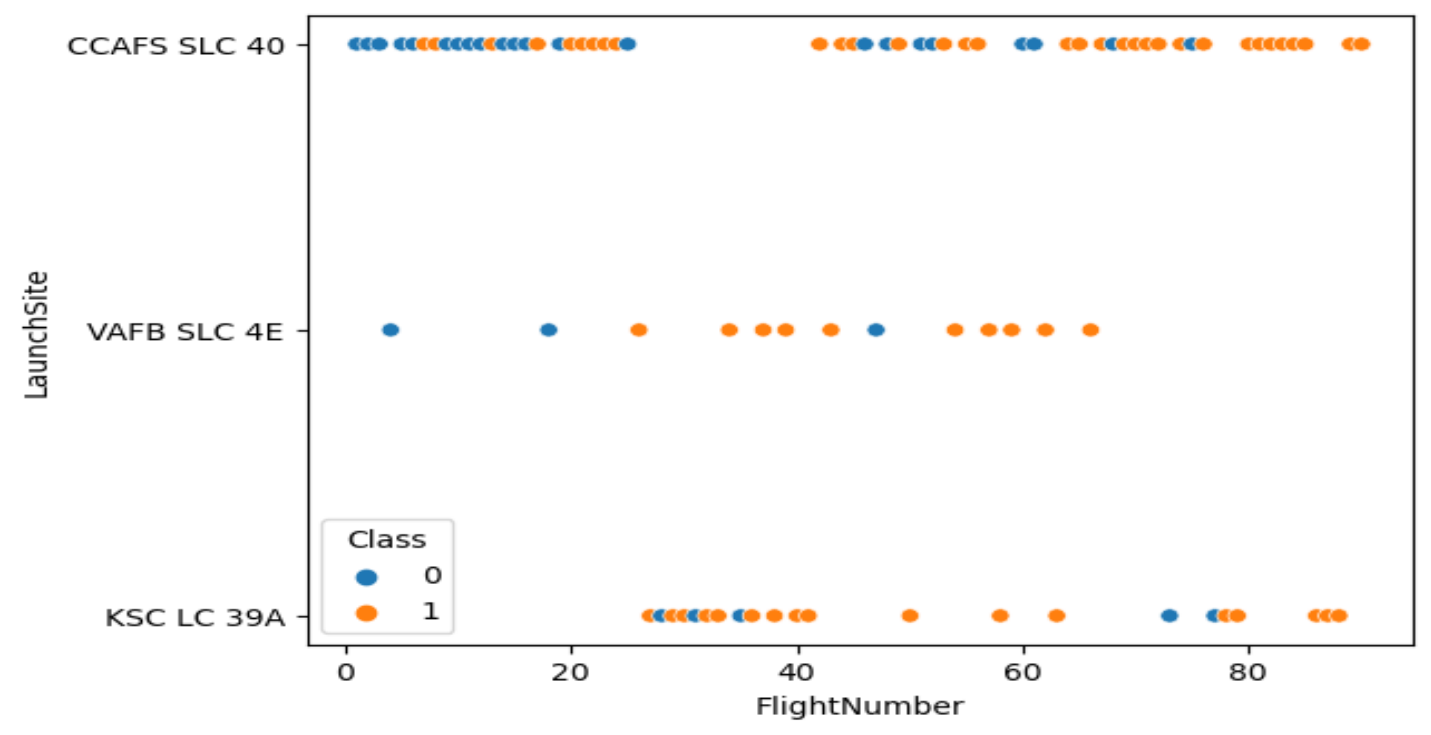
Results

- Exploratory data analysis results
  - For the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000)
  - SpaceX have better successful rates when in ES-L1, GEO, HEO, and SSO. Orbit GTO has the worst successful rate
  - Seems there is no relationship between flight number and Orbit when in GTO orbit. When in EL-L1, SSO and HEO, the successful rates are 100%
  - The successful rate started increasing in 2013 and kept it up until 2020
- Interactive analytics demo in screenshots
  - Three out of four launch sites are located at East coast and one launch site is located at the West coast
  - KSC LC launch site has the highest successful rate
  - CCAFS SLC launch site has the highest unsuccessful rate
- Predictive analysis results
  - All the models applied on the test dataset return the same accuracy score


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

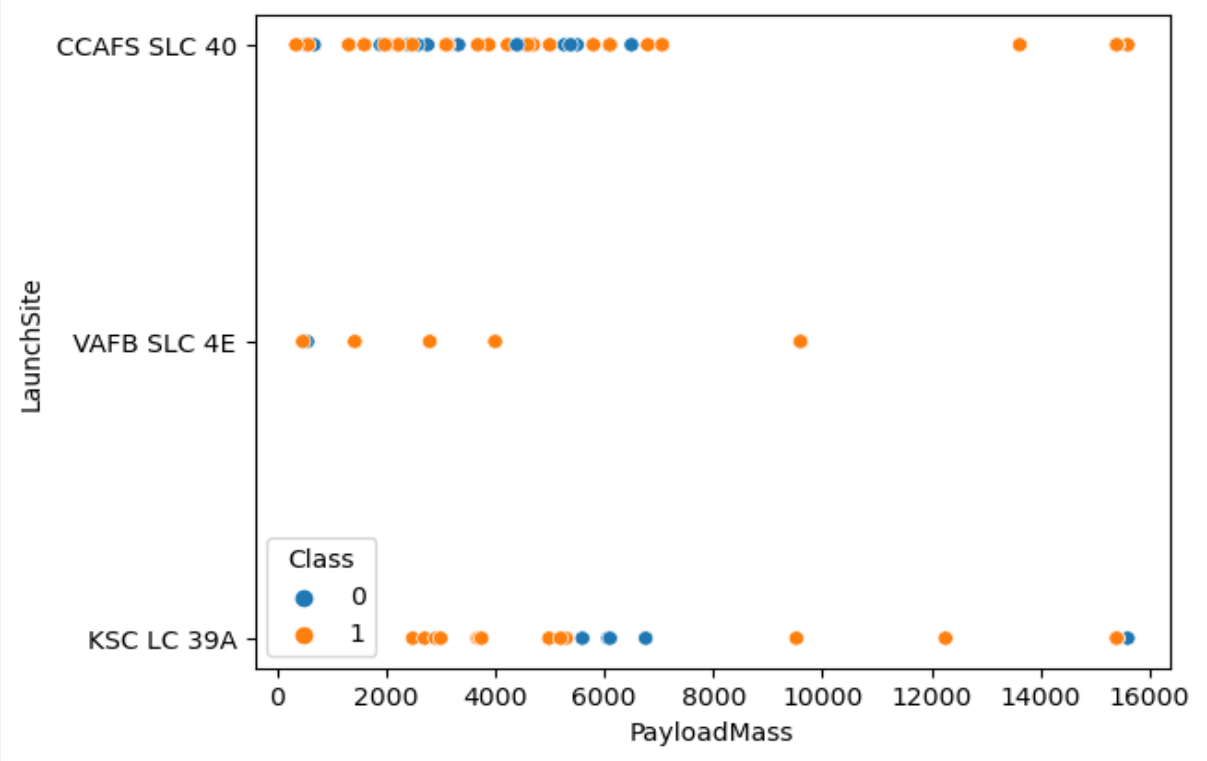
- Show a scatter plot of Flight Number vs. Launch Site



Flight numbers are corelated to the date. The higher the flight number, the later the date. The above chart shows that SpaceX stopped using VAFB SLC launch site after flight number 66. Both CCAFS SLC and KSC LC launch sites had higher successful rate since flight number 80, showing the successful rate has been improved through out the time.

Payload vs. Launch Site

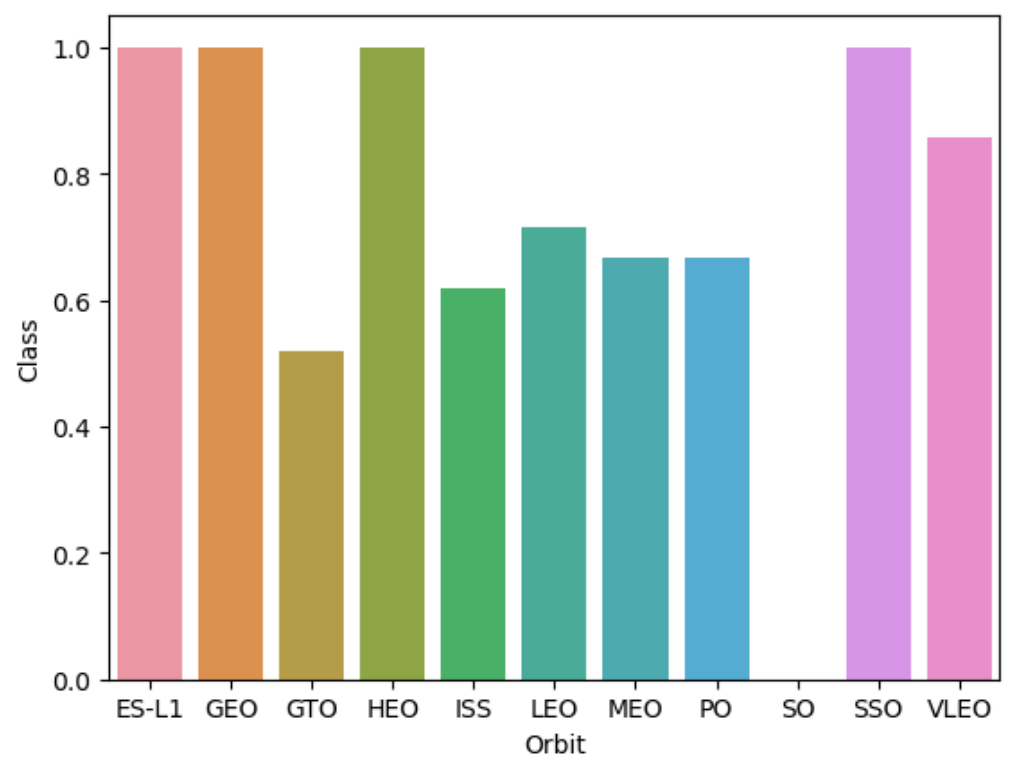
- Show a scatter plot of Payload vs. Launch Site



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

Success Rate vs. Orbit Type

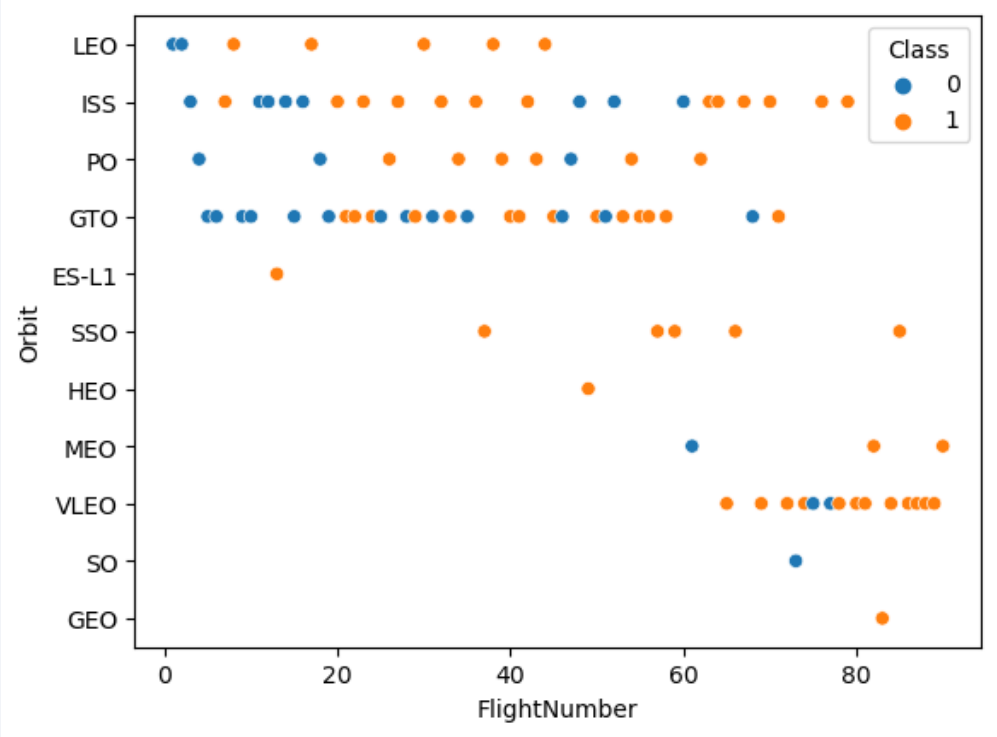
- Show a bar chart for the success rate of each orbit type



SpaceX have better successful rates when in ES-L1, GEO, HEO, and SSO.
Orbit GTO has the worst successful rate

Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type

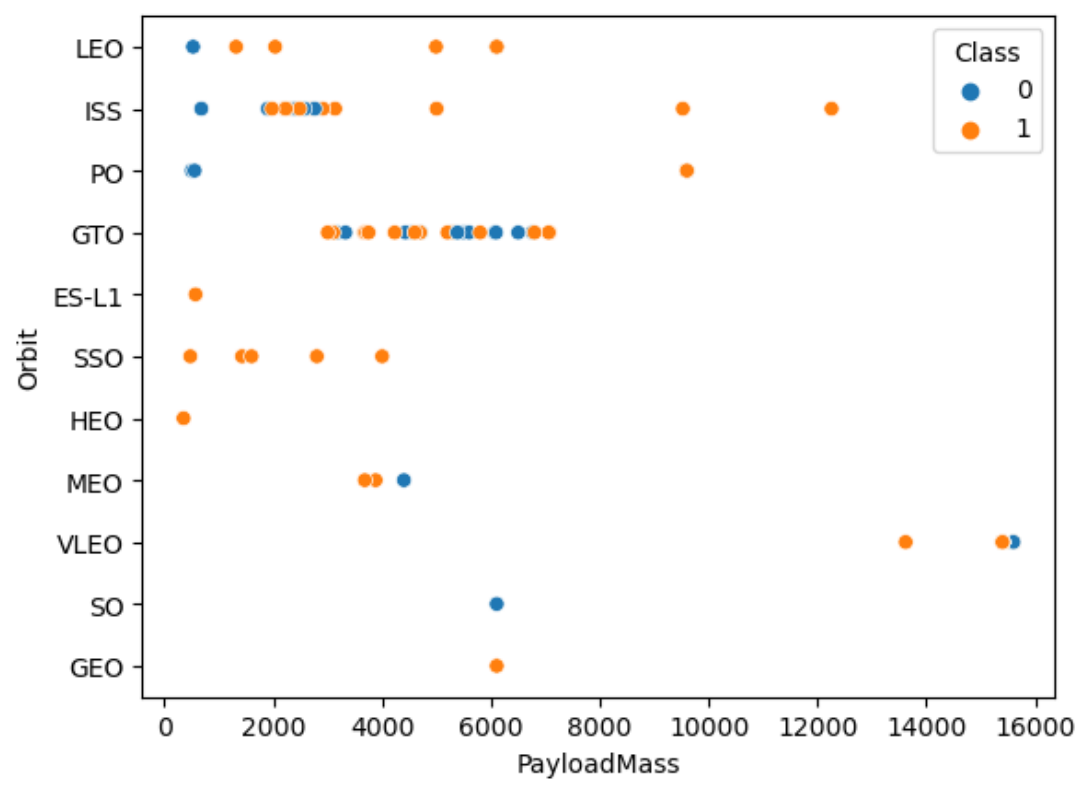


Seems there is no relationship between flight number and Orbit when in GTO orbit

When in EL-L1, SSO and HEO, the successful rates are 100%

Payload vs. Orbit Type

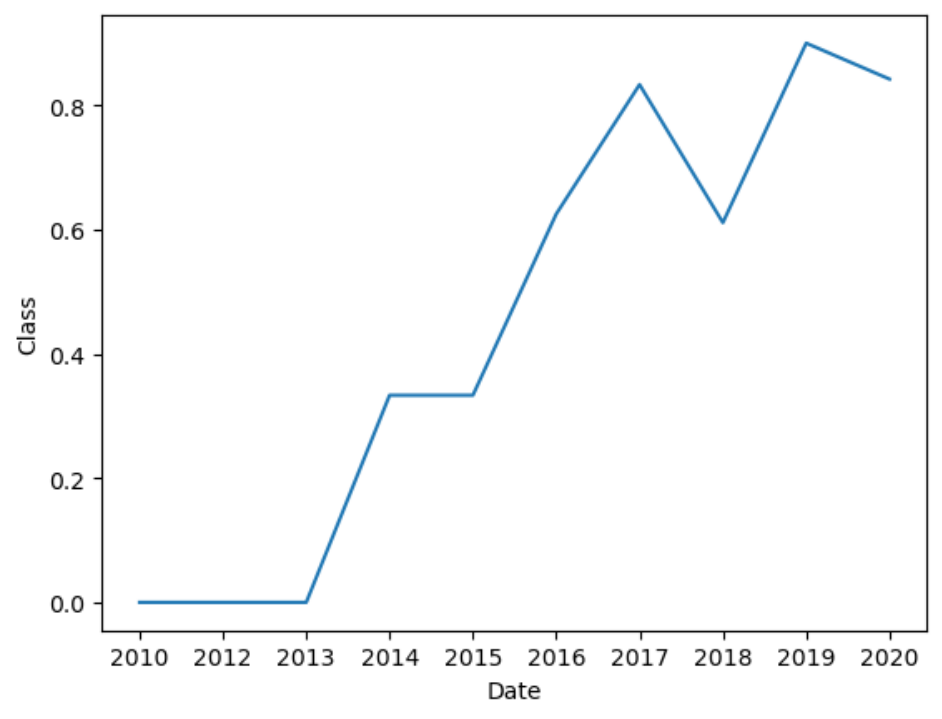
- Show a scatter point of payload vs. orbit type



- With heavy payloads the successful landing are more for Polar, LEO and ISS.
- Cannot distinguish the relationship for GTO since there are mixed positive landing rate and negative landing
- Only low payload for ES-L1, SSO, and HEO and the very high successful rate

Launch Success Yearly Trend

- Show a line chart of yearly average success rate



The successful rate started increasing in 2013 and kept it up until 2020

All Launch Site Names

- Find the names of the unique launch sites

```
%%sql

select distinct Launch_Site
from SPACEXTABLE
```

Running query in 'sqlite:///my_data1.db'

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

Using SELECT DISTINCT to extract the data and the results shows there are 4 unique launch sites

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

```
%%sql

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

Running query in 'sqlite:///my_data1.db'

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

Use wildcard (%) to extract lunch sites that start with the letter ‘CCA’ and use LIMIT to get a specific number of records

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
%%sql

select Booster_Version, customer, sum(PAYLOAD_MASS__KG_) as total_payload_mass
from SPACEXTABLE
where customer like '%NASA%CRS%'
group by Booster_Version
```

Running query in 'sqlite:///my_data1.db'

Booster_Version	Customer	total_payload_mass
F9 B4 B1039.2	NASA (CRS)	2647
F9 B4 B1039.1	NASA (CRS)	3310
F9 B4 B1045.2	NASA (CRS)	2697
F9 B5 B1056.2	NASA (CRS)	2268
F9 B5 B1058.4	NASA (CRS)	2972
F9 B5 B1059.2	NASA (CRS)	1977
F9 B5B1050	NASA (CRS)	2500
F9 B5B1056.1	NASA (CRS)	2495
F9 B5B1059.1	NASA (CRS), Kacific 1	2617
F9 FT B1035.2	NASA (CRS)	2205

Truncated to displaylimit of 10.

- Use GROUP BY to pivot the type of the Booster
- Use function sum() to add total payload mass for each Booster
- Use WHERE clause to set condition that only display Booster from NASA

Average Payload Mass by F9 v1.1

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

```
%%sql

select Booster_Version, avg(PAYLOAD_MASS__KG_) as average_payload_mass
from SPACEXTABLE
where Booster_Version like 'F9 v1.1%'
group by Booster_Version
```

Running query in 'sqlite:///my_data1.db'

Booster_Version	average_payload_mass
F9 v1.1	2928.4
F9 v1.1 B1003	500.0
F9 v1.1 B1010	2216.0
F9 v1.1 B1011	4428.0
F9 v1.1 B1012	2395.0
F9 v1.1 B1013	570.0
F9 v1.1 B1014	4159.0
F9 v1.1 B1015	1898.0
F9 v1.1 B1016	4707.0
F9 v1.1 B1017	553.0

Truncated to displaylimit of 10.

- Use GROUP BY to pivot the type of the Booster
- Use WHERE clause to limit what booster version to be extracted, in this case F9 v1.1
- Use function avg() to calculate the average of payload mass for each F9 v1.1 booster version

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
%%sql

select *, min('Date')
from SPACEXTABLE
where Landing_Outcome == 'Success'
```

Running query in 'sqlite:///my_data1.db'

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome	min('Date')
2018-07-22	05:50:00	F9 B5B1047.1	CCAFS SLC-40	Telstar 19V	7075	GTO	Telesat	Success	Success	Date

Use function min() to find the earliest date in the Date column and use WHERE clause to result (i.e. the landing outcome to be success)

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```
%%sql

select Booster_Version, PAYLOAD_MASS__KG_, landing_outcome
from SPACEXTABLE
where (PAYLOAD_MASS__KG_ > 4000) and (PAYLOAD_MASS__KG_ < 6000)
     and landing_outcome like '%success%drone%'
```

Running query in 'sqlite:///my_data1.db'

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

Use WHERE clause to set the condition of what data to be extracted

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
%%sql

select *
from (select count(distinct(landing_outcome)) as successful_outcomes
      from SPACEXTABLE
      where landing_outcome like '%success%'),

      (select count(distinct(landing_outcome)) as unsuccessful_outcomes
      from SPACEXTABLE
      where landing_outcome like '%failure%')
```

Running query in 'sqlite:///my_data1.db'

successful_outcomes	unsuccessful_outcomes
3	3

Use two subqueries in FROM clause. One to get the number of successful outcomes and the other one is to get the number of unsuccessful outcomes.

Boosters Carried Maximum Payload

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

```
%%sql

select *
from SPACEXTABLE
where PAYLOAD_MASS__KG_ = (
    select max(PAYLOAD_MASS__KG_)
    from SPACEXTABLE)
```

Running query in 'sqlite:///my_data1.db'

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2019-11-11	14:56:00	F9 B5 B1048.4	CCAFS SLC-40	Starlink 1 v1.0, SpaceX CRS-19	15600	LEO	SpaceX	Success	Success
2020-07-01	02:33:00	F9 B5 B1049.4	CCAFS SLC-40	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600	LEO	SpaceX	Success	Success
2020-01-29	14:07:00	F9 B5 B1051.3	CCAFS SLC-40	Starlink 3 v1.0, Starlink 4 v1.0	15600	LEO	SpaceX	Success	Success
2020-02-17	15:05:00	F9 B5 B1056.4	CCAFS SLC-40	Starlink 4 v1.0, SpaceX CRS-20	15600	LEO	SpaceX	Success	Failure

- Use a subquery to extract a scalar number which is used in the WHERE clause as a condition

2015 Launch Records

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

```
%%sql

select substr(Date, 6,2) as Month, Date, Landing_Outcome, Booster_Version, Launch_Site
from (select *, substr(Date, 1,4) as Year
      from SPACEXTABLE)
where Landing_Outcome like '%failure%drone%'
      and Year='2015'
```

Running query in 'sqlite:///my_data1.db'

Month	Date	Landing_Outcome	Booster_Version	Launch_Site
10	2015-10-01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Original dataset has no Year column. We use a subquery to add additional column called Year and the result of the subquery became the new dataset and is placed in the FROM clause then we can filter the data in year 2015.

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

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

```
%%sql

select Date, Landing_Outcome, count(*) as count_landing_outcomes
from SPACEXTABLE
where Date between '2010-06-04' and '2017-03-20'
group by Landing_Outcome
order by count_landing_outcomes desc
```

Running query in 'sqlite:///my_data1.db'

Date	Landing_Outcome	count_landing_outcomes
2012-05-22	No attempt	10
2015-12-22	Success (ground pad)	5
2016-08-04	Success (drone ship)	5
2015-10-01	Failure (drone ship)	5
2014-04-18	Controlled (ocean)	3
2013-09-29	Uncontrolled (ocean)	2
2015-06-28	Precluded (drone ship)	1
2010-08-12	Failure (parachute)	1

- Use GROUP BY to get the kind of landing outcomes and use function count() to get the number of outcomes for each type of landing outcomes
- Use ORDER BY DESC to show the result in a descending order

Section 3

Launch Sites Proximities Analysis

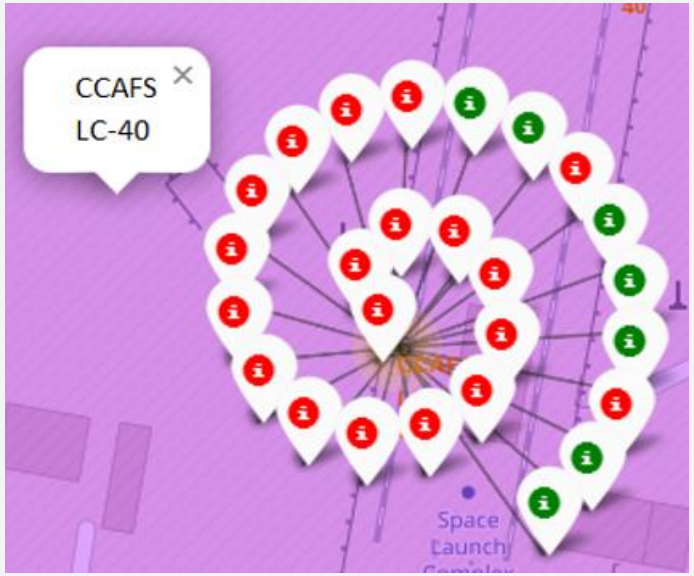
Create a Folium Map that includes all the launch sites

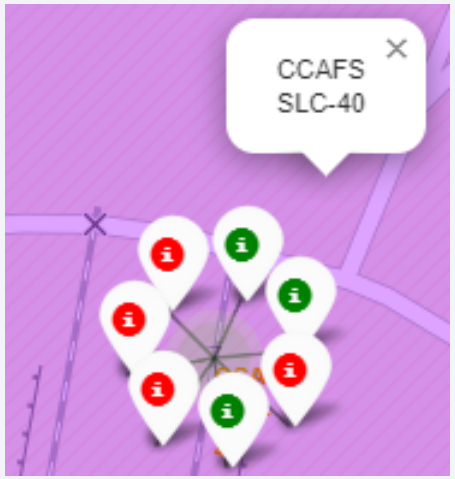
- Site map to show all the launch sites

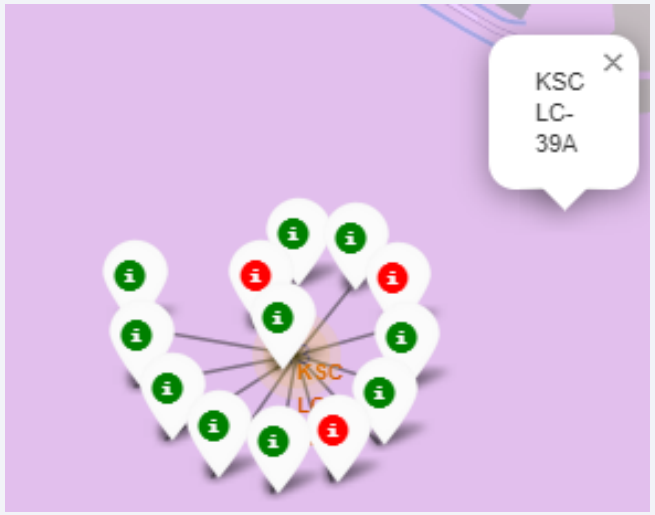
Three out of four launch sites are located at East coast and one launch site is located at the West coast

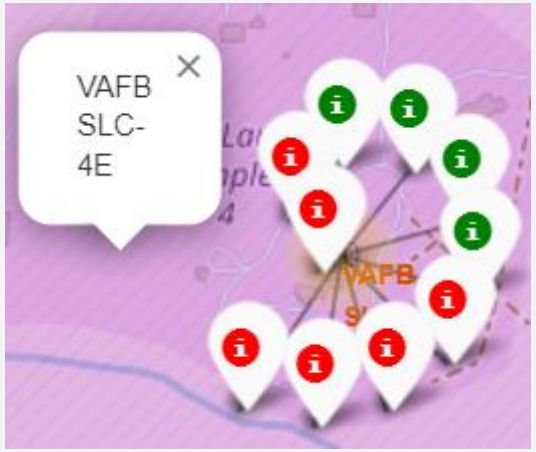
Add launch results of each launch site to the map

Launch results of each launch site: Green represent success and Red represents unsuccessful





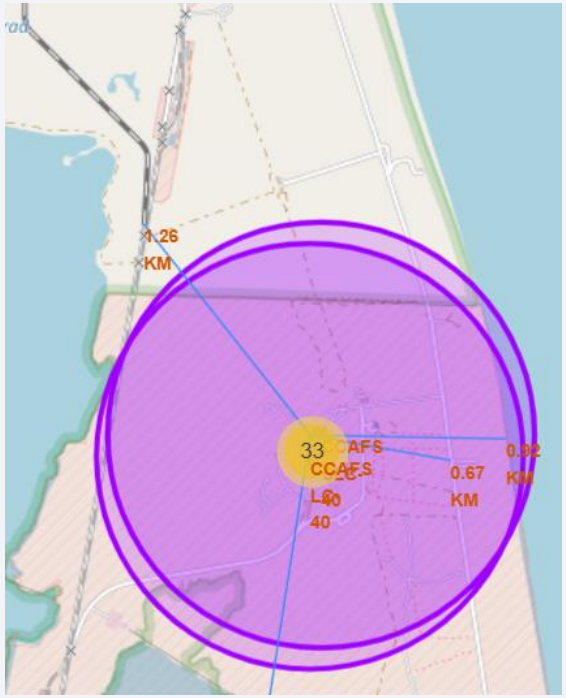




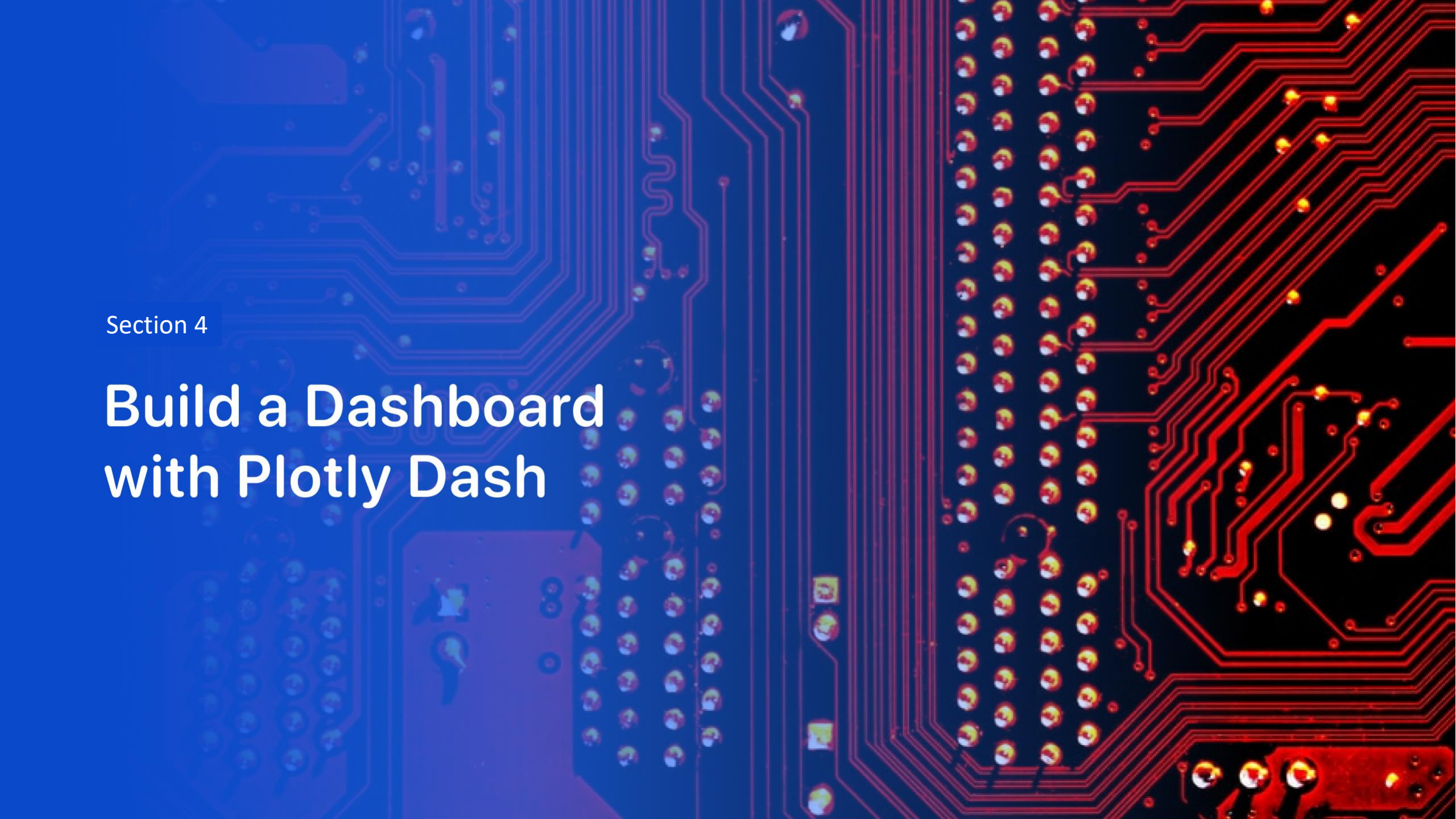
- KSC LC launch site has the highest successful rate
- CCAFS SLC launch site has the highest unsuccessful rate

Distances between a launch site to its proximities

- Distance to a closest city, railway, highway and coastline

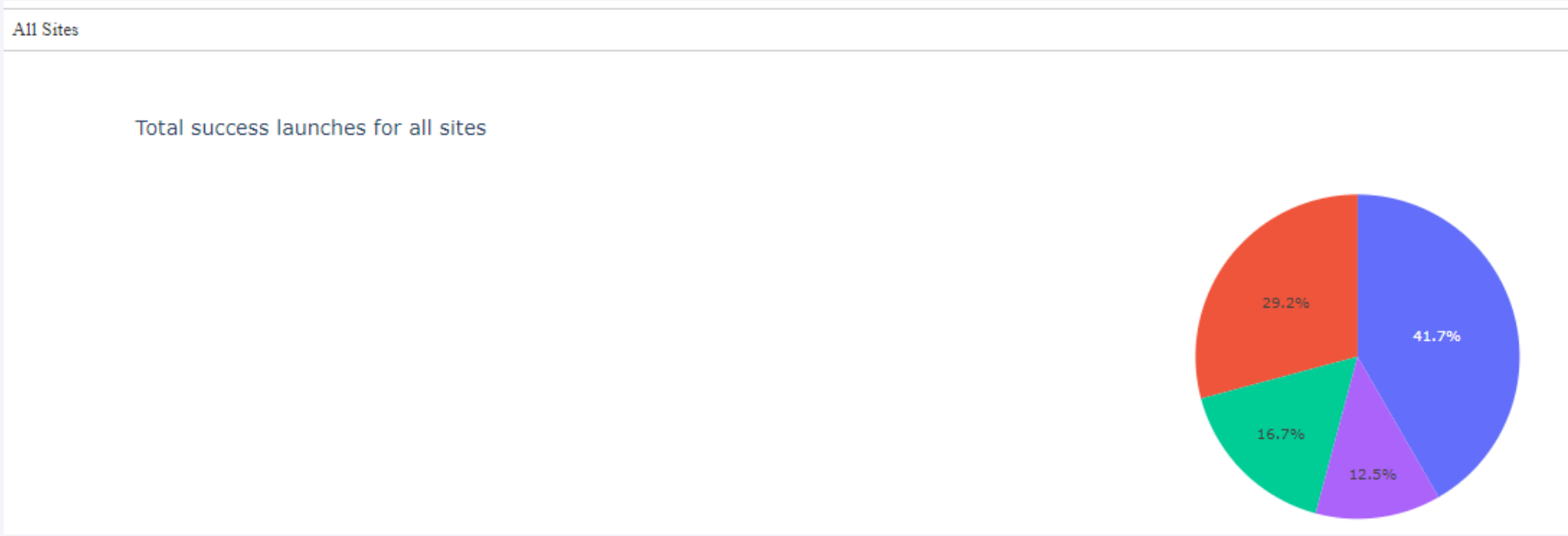


- CCAFS SLC and CCAFS LC launch sites are very close to the highway and the coastline


Section 4
Build a Dashboard with Plotly Dash

Total success launches for all sites

- Successful launches for all sites



Within the four launch site, the site has the highest successful rate is 41.7%

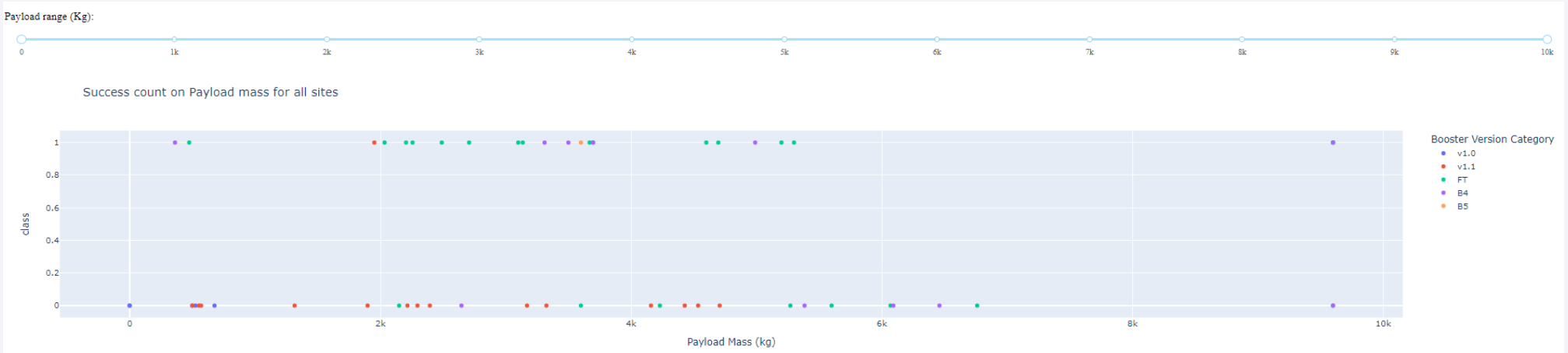
The site has the highest successful rate

- KSC LC launch site has the highest successful rate

76.9% of its outcome returns successful

Scatter plot for all sites with different payload selection

- Compare scatter results with different slider range



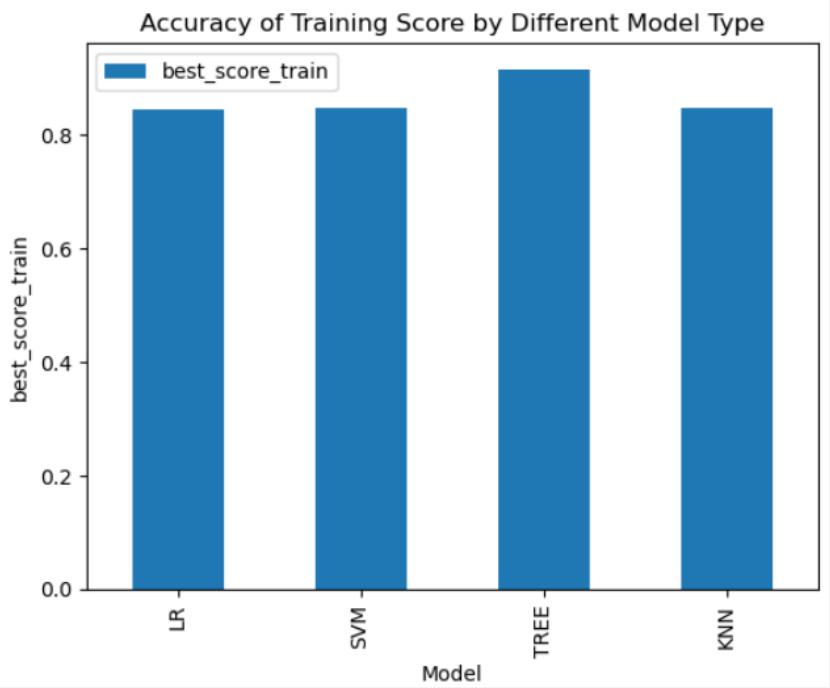
The plot is interactive. With different selection, plot results are updated simultaneously

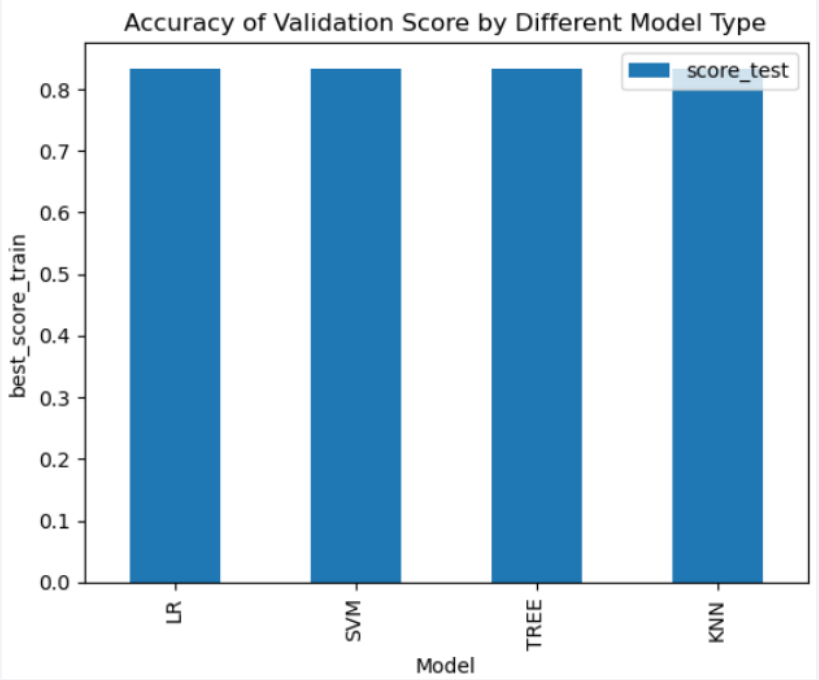
Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Visualize the built model accuracy for all built classification models, in a bar chart



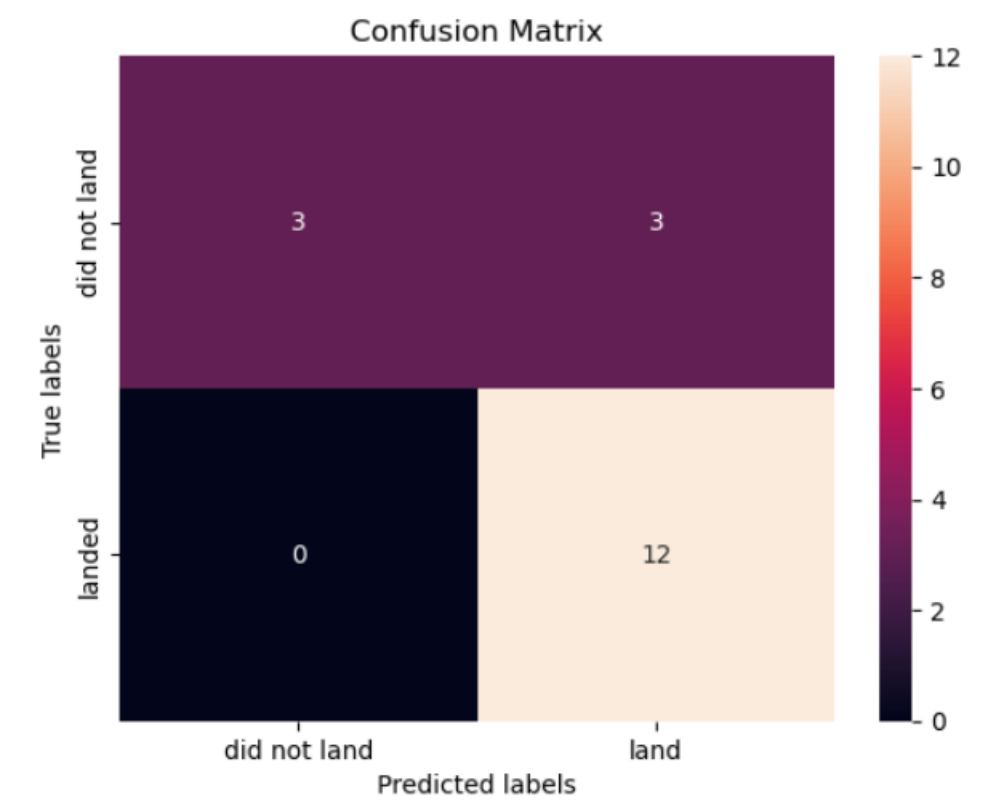


Decision Tree output the best accuracy score when use the training dataset.

All of the models return the same validation scores when use the test dataset

Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



- The major problem is false positive.

Conclusions

- Point 1

The different models were all trained on the same training dataset and the same test dataset and therefore, we are getting the same results from all the models

- Point 2

We should split the train vs. test dataset randomly and repeat the process to revaluate the results returned by different models

Thank you!