Winning Space Race with Data Science

Raihan Insan Habibi

Wednesday, January 5, 2022 - Friday, January 14,2022

Outline

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

Executive Summary

- In this presentation, our goal is to find the best location for the space X to launch and land. We will also find other features that affects the launch and land outcome.
- We will use the Space X API to extract the information
- We will use SQL to filter out the data and draw out conclusions
- We will use matplotlib to plot beautiful visualizations such as bar charts and scatter plots, we will use these modules find out the pattern of the investigation.
- We will create a machine learning module that predicts the data
- We also need to find how accurate the module is

Introduction

In December 2015, Elon Musk made the first-ever orbital rocket called the ‘Falcon 9 heavy’. It was the first ever rocket that made a vertical landing back to Earth and it was a successful return . This Mission started back in 2010, where after several failed attempts, the engineering was a masterpiece. The failed attempts were caused by environmental features and the launch and landing sites, the location was not too perfect for the rocket. As a result, the rocket was destroyed for every attempt, losing a lot of money and resources in the budget. As a Data Scientist, I will show you in this presentation how I was able to find a perfect location for the rocket to land and launch using Python, SQL and the Space X API to analyze the data and give the best result.

Section 1

Methodology

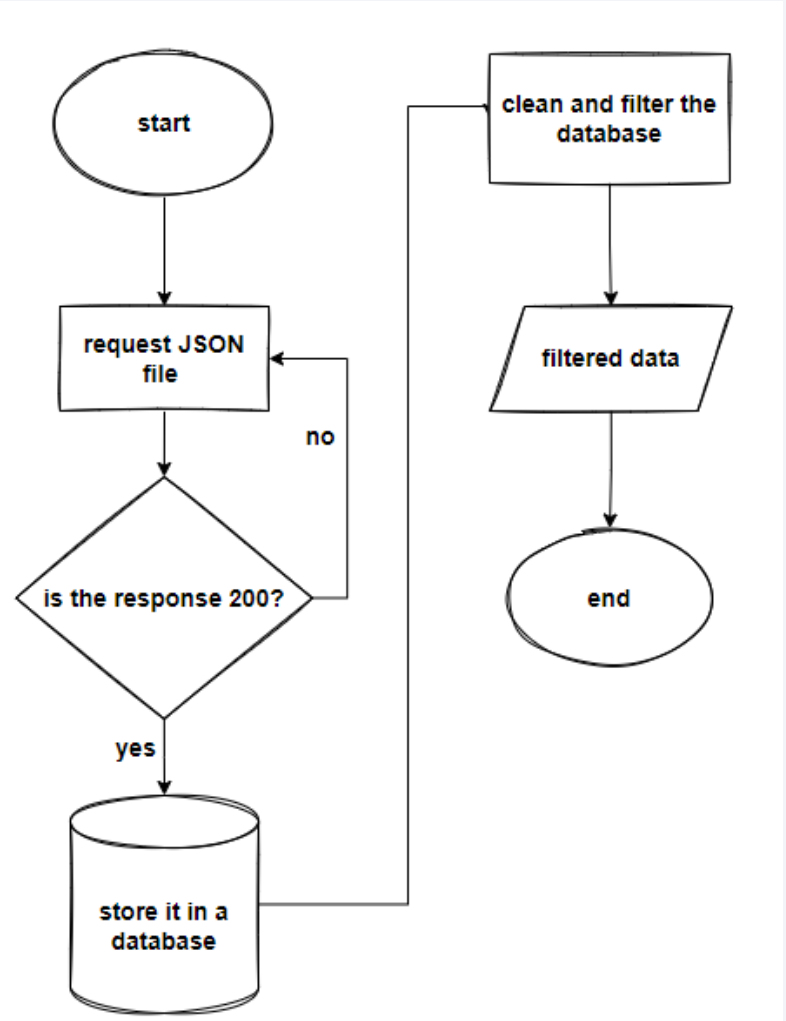
Methodology

Executive Summary

- Data collection methodology:
  - Using the space x API, extract the information in the form of a JSON file.
  - Convert the JSON file into a dataset (for cleaning and filtering).
- Perform data wrangling
  - Beautiful Soup Module to extract the information we need from the HTML .
  - First we used the title dot text method to extract the title of the website.
  - Then using a 'for loop' to automatically append the information to our new filtered dataset.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models:
  - Using the dataset, we need to train and test the dataset using the train_test_split method
  - We then create a module to predict the future outcome
  - Finally we state how accurate the module is.

Data Collection

- The URL is first input into a string.
- we take the response by requesting it using the Request Module.
- If the response is a success we can collect the data in the form of a JSON file then filter out the data we need.
- We can then form this data into a database.

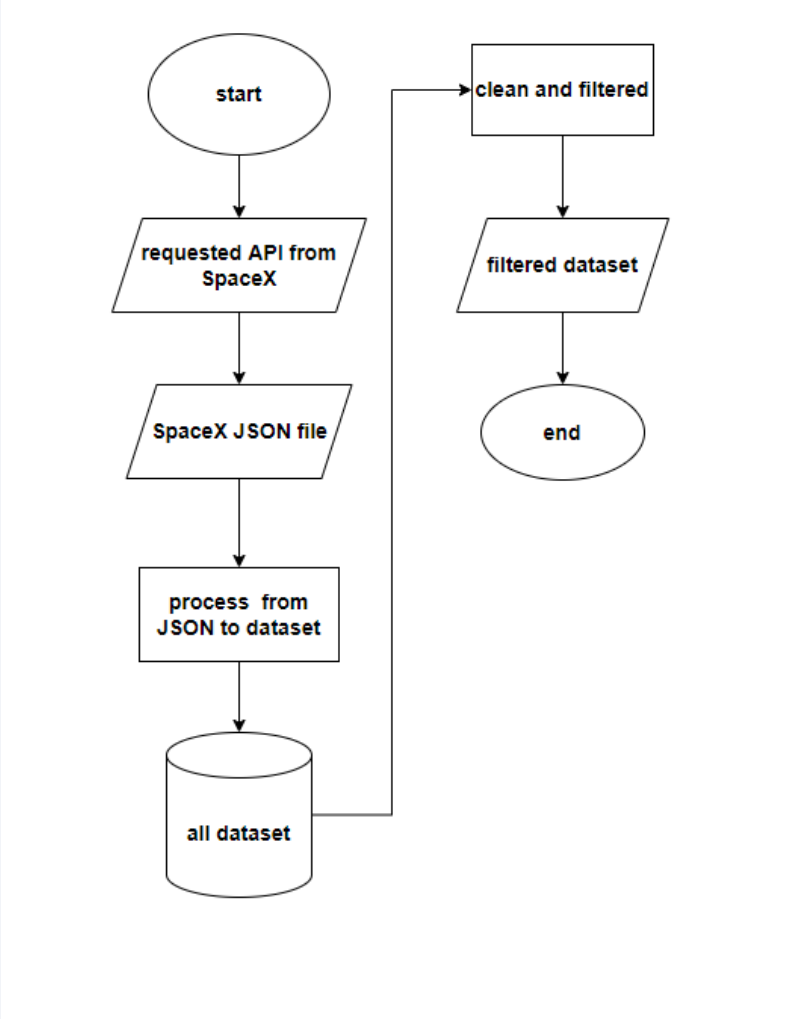


Data Collection – SpaceX API

reference:

https://github.com/RaihanTeachesCodes/IBM_course_10-/blob/master/data%20collection.ipynb

- When we requested the SpaceX API using the request module we have to get a status code of 200, to have a successful request and response.
- We then normalize the JSON file to turn it in to a dataset.



Data Collection - Scraping

Reference:

https://github.com/RaihanTeachesCodes/IBM_course_10-/blob/master/data%20collection.ipynb

- Using the data we collected we filter it out the column using the:

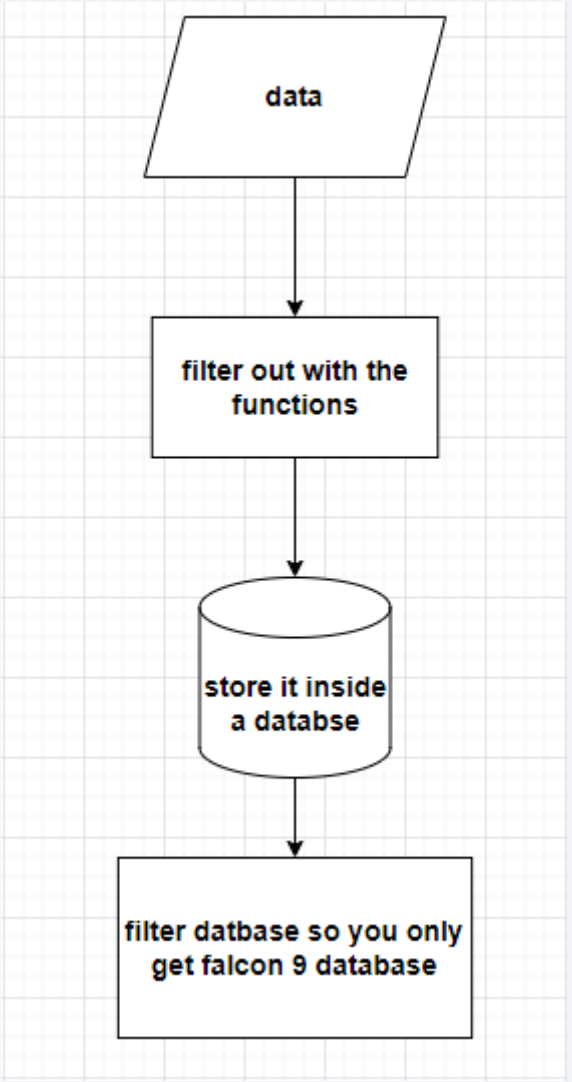
  -getboosterversion( )

  -getlaunchsite( )

  -getpayloaddata( )

  -getcoredata( ).

- I then append the output of the functions to a list that I created.

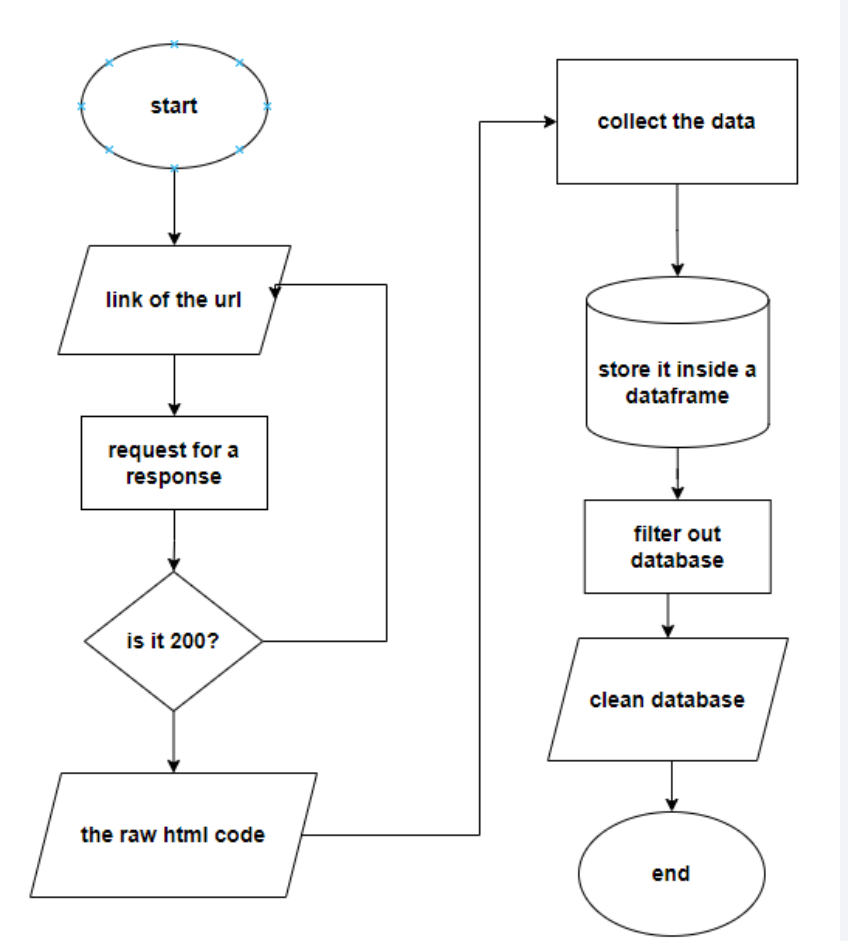


Data Wrangling

Reference:

https://github.com/RaihanTeachesCodes/IBM_course_10-final-project/blob/master/data%20wrangling.ipynb

- I first extract all HTML from the Wikipedia website.
- I used the title dot text method from the soup module to extract the title.
- The data then was processed by using a “for loop” to automatically scrape the html.
- I append these data to an empty dataset that I created.
- The output is a filtered dataset.



EDA with Data Visualization

Reference:

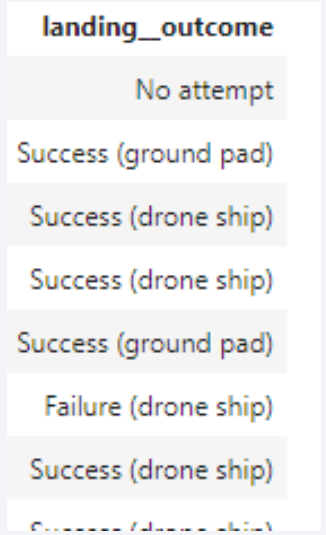
https://github.com/RaihanTeachesCodes/IBM_course_10-final-project/blob/master/EDA%20with%20Data%20Visualization.ipynb

- To visualize the difference of relationship between the 2 data I used a cat plot because it was a module that can make it much easier to tell the difference.
- For separating the class I used a bar chart because it was easier to understand the difference of each class.
- To visualize the launch success yearly trend I used a line chart because I can see the trend much easier to draw out conclusions based from the gradient .

EDA with SQL

Reference:

https://github.com/RaihanTeachesCodes/IBM_course_10-final-project/blob/master/eda%20sql.ipynb



- We take out the database by first connecting to my credentials.
- I Filtered the dataset so at the end result I would get the Rank of 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

Build an Interactive Map with Folium

Reference:

https://github.com/RaihanTeachesCodes/IBM_course_10-final-project/blob/master/interactive%20visualization%20with%20folium.ipynb

- I have created circles around the mark because I want to show the accessible landing site in a specific radius.
- The markers are used to point out the location of the landing site. And the line is used to represent the distance between a launch site to its proximities.
- These object creates a beautiful visualization to the folium map with clear understanding about the data.

Build a Dashboard with Plotly Dash

Reference:

https://github.com/RaihanTeachesCodes/IBM_course_10-/blob/master/spacex_dash_app.py

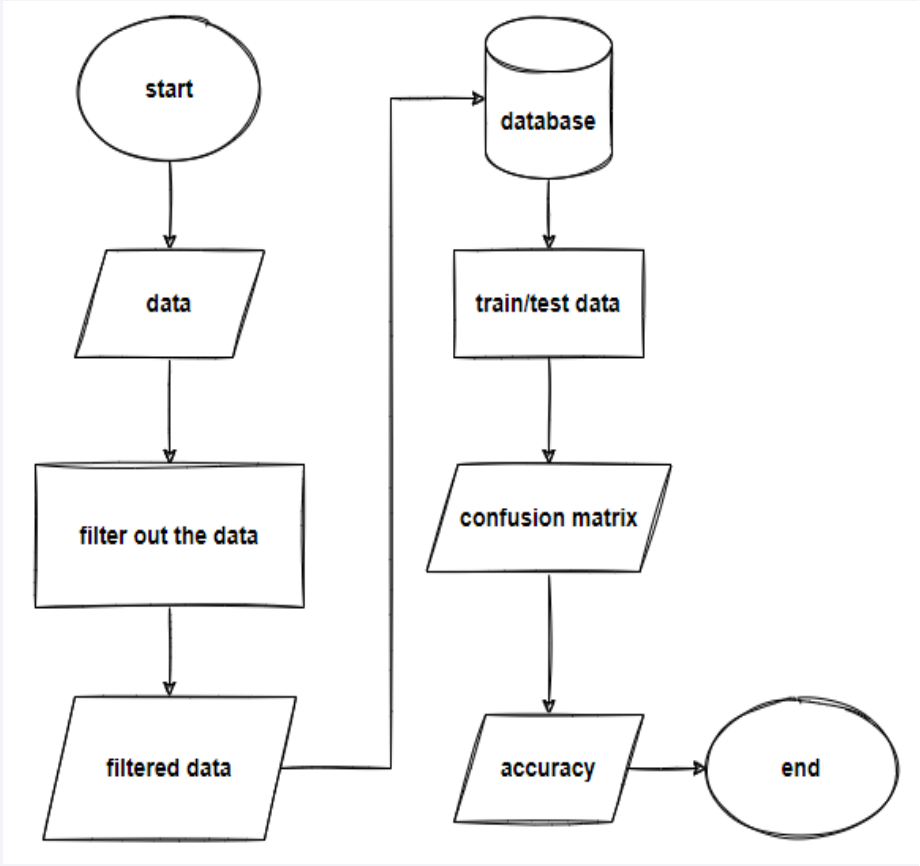
- I have added the pie charts because its easier to classify each variable values in different class.
- the Dropdown options is there to give different data about the launching site.
- the slider changes the results of the payload mass, giving different ranges
- And lastly the scatter chart shows the relationship between two variables.

Predictive Analysis (Classification)

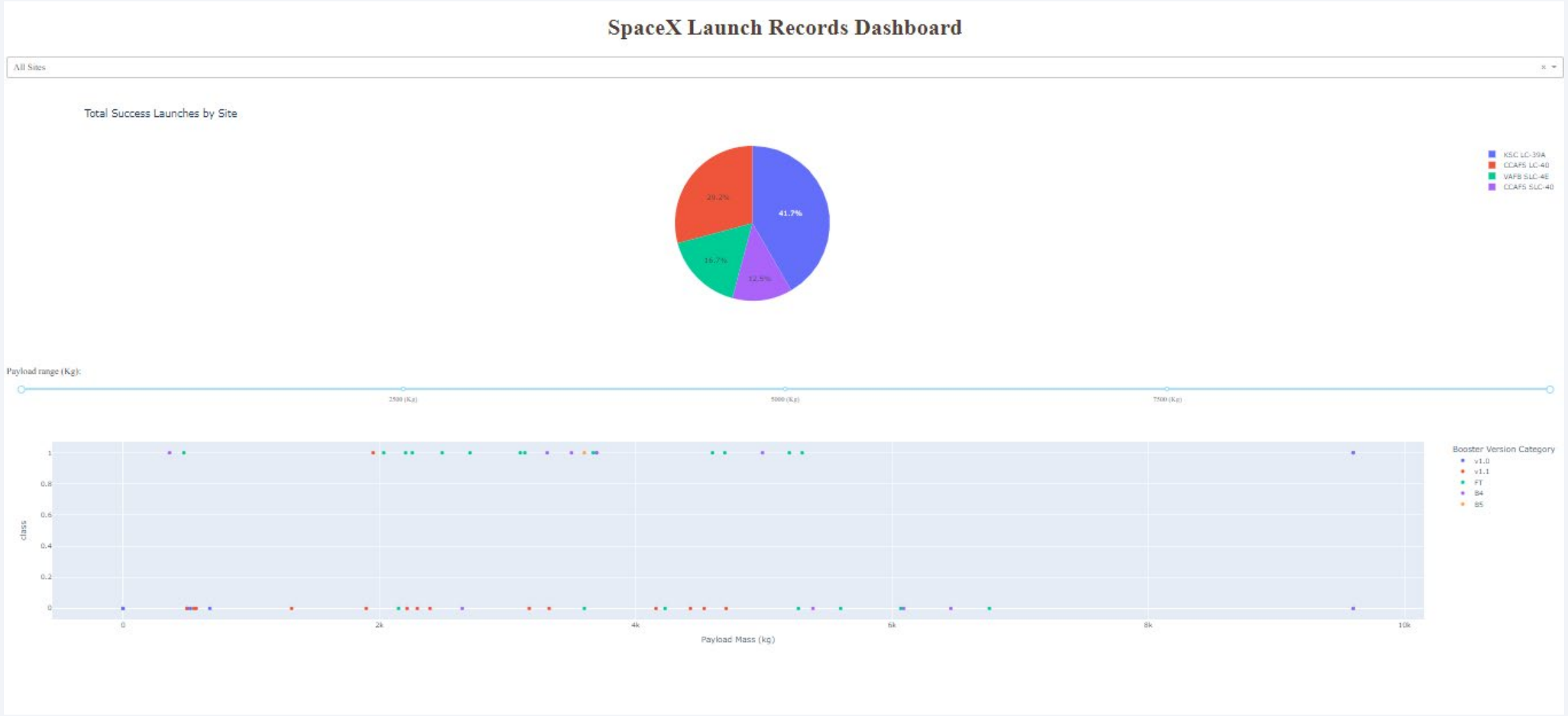
Reference:

https://github.com/RaihanTeachesCodes/IBM_course_10-/blob/master/Machine%20Learning%20Prediction.ipynb

- After I took the database, I transformed the dataset so it is trainable.
- Then I train and tested the data using the train_test_split method to create a module.
- I then output all types of accuracy score and find out which module give out the highest accuracy.
- I made a confusion matrix to evaluate the module



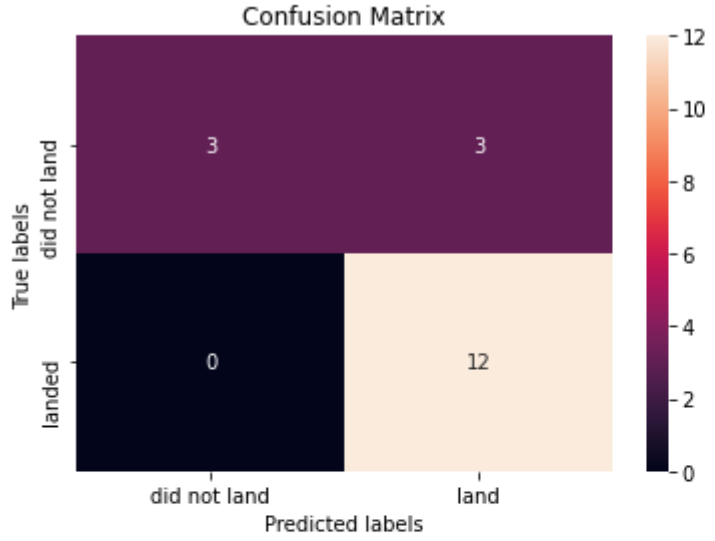
Results


SpaceX Launch Records Dashboard
All Sites
Total Success Launches by Site
41.7%
29.2%
16.7%
12.5%
KSC LC-39A
CCAFS LC-40
VAFB SLC-4E
CCAFS SLC-40
Payload range (Kg):
2500 (Kg)
5000 (Kg)
7500 (Kg)
class
Payload Mass (kg)
Booster Version Category
v1.0
v1.1
FT
B4
B5

Results

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitud
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.5618
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.5618
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.5618
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.6320
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.5618

Results

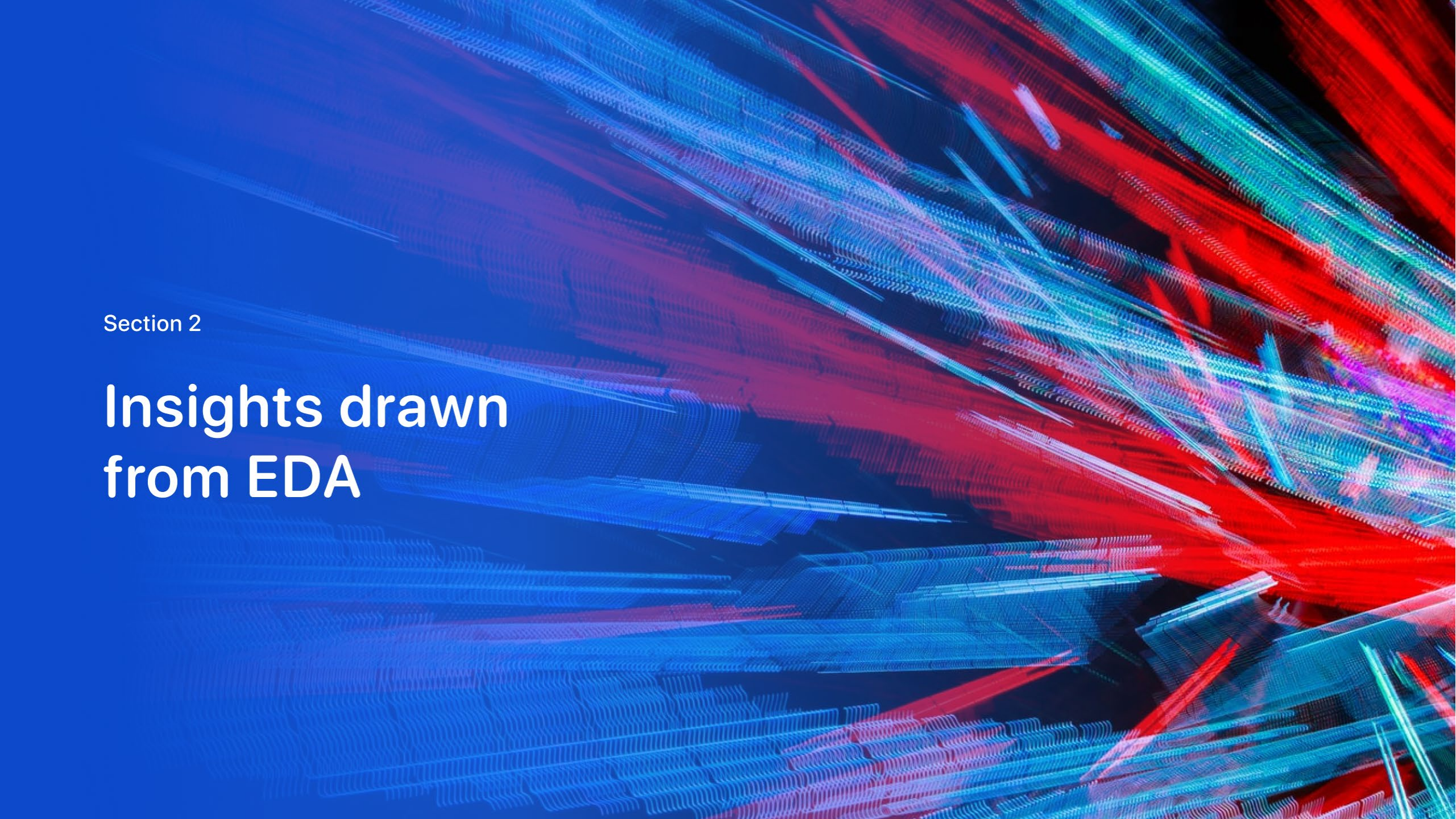


TASK 12

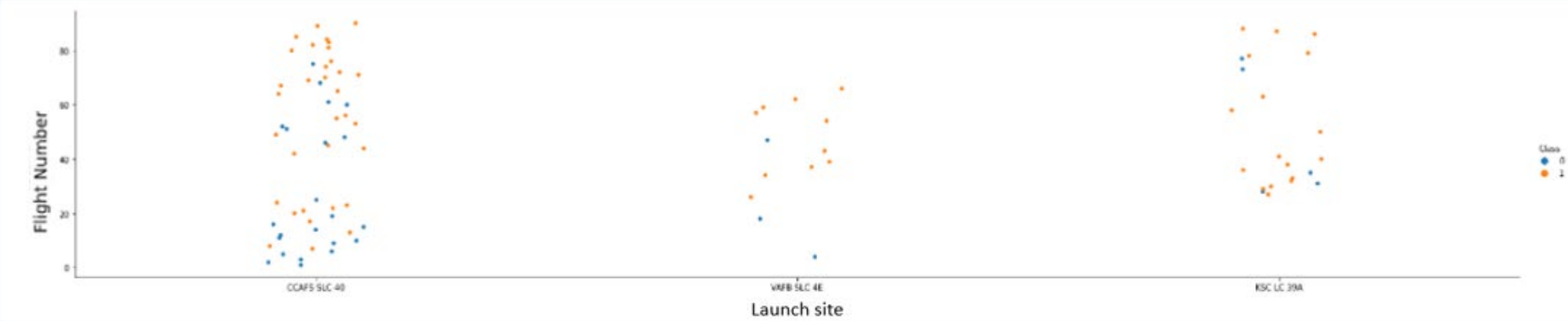
Find the method performs best:

In [41]:
```
scores = [lr_score,svm_score,tree_score,knn_score]
print(scores)
print(scores.index(max(scores)))
```

```
[0.8333333333333334, 0.8333333333333334, 0.7777777777777778, 0.8333333333333334]
```

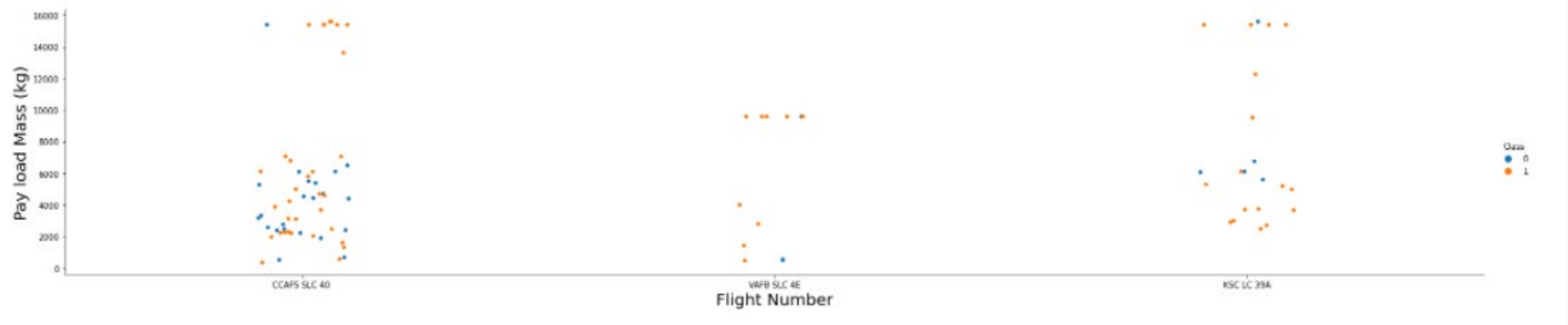

Section 2
Insights drawn
from EDA

Flight Number vs. Launch Site



- As you can see from this scatter plot, most launch site is in ‘CCAFS LC-40’ ,
- The least number is in launch site ‘VAFB SLC 4E’.

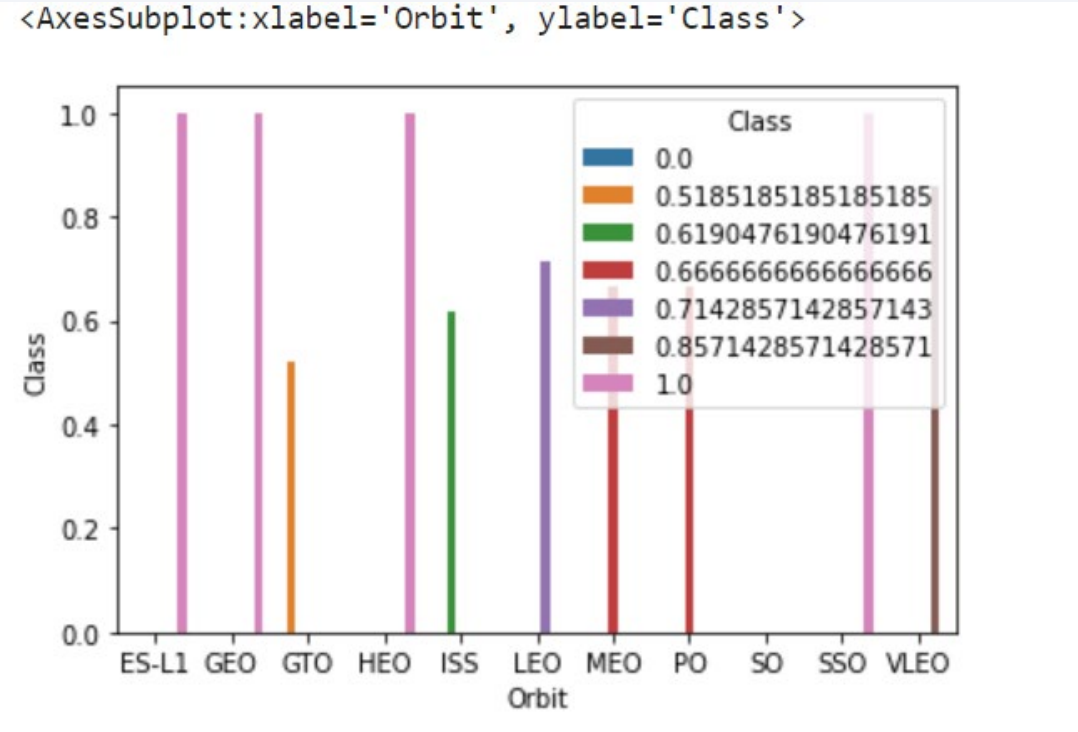
Payload vs. Launch Site



- The highest pay load mass will be in flight number '**CCAFS LC-40**'.

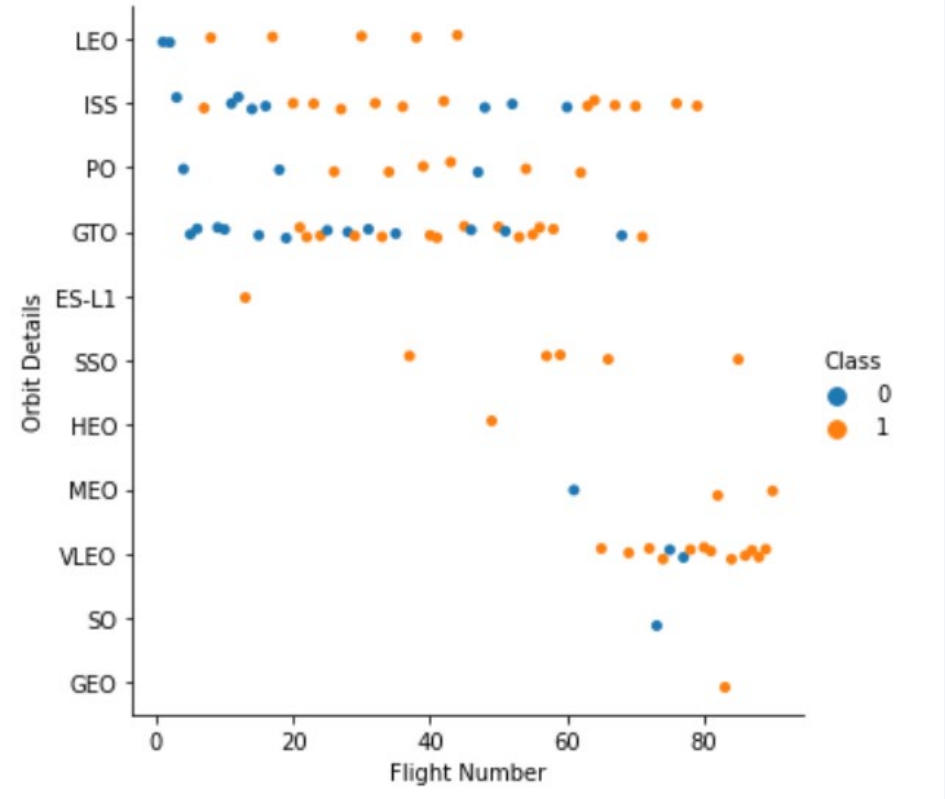
- The smallest pay load mass will be between **CCAFS LC-40** or **VAFB SLC 4E**

Success Rate vs. Orbit Type



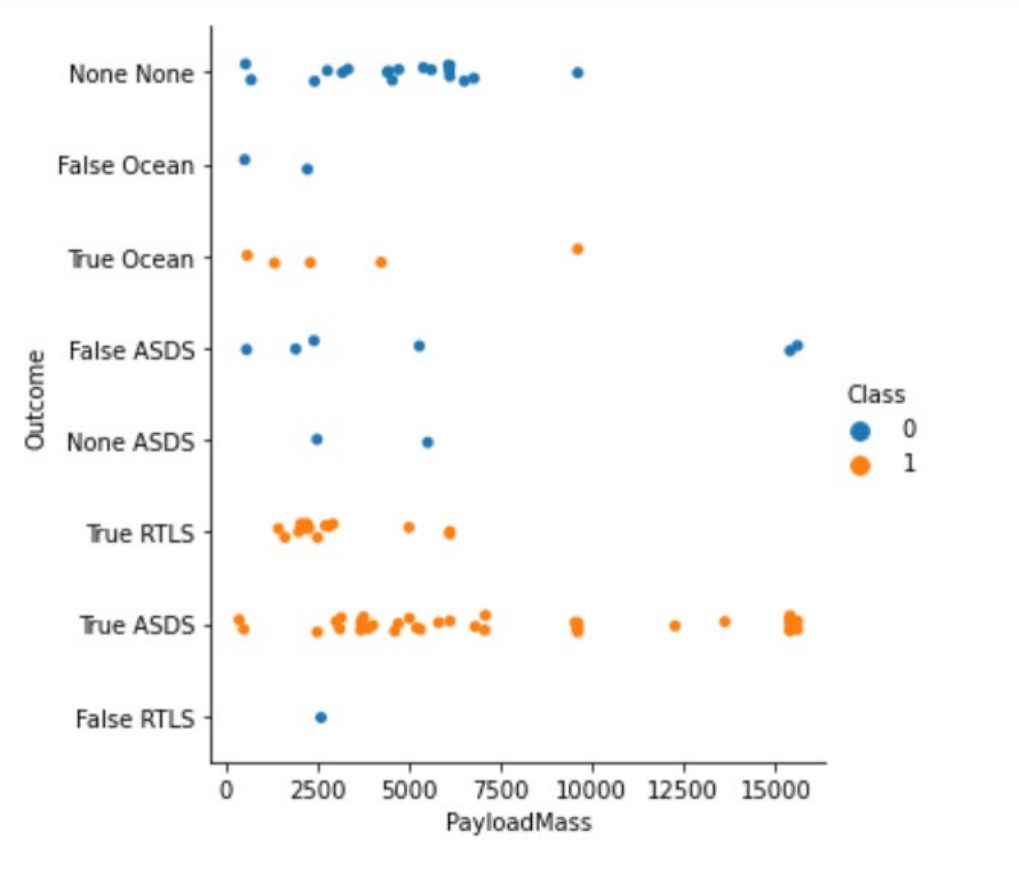
- The bar chart shows the relationship of the success rate and orbit type.
- It shows that the class with the pink color give a 100% success rate.
- While the class with the blue color gives 0% success rate.

Flight Number vs. Orbit Type



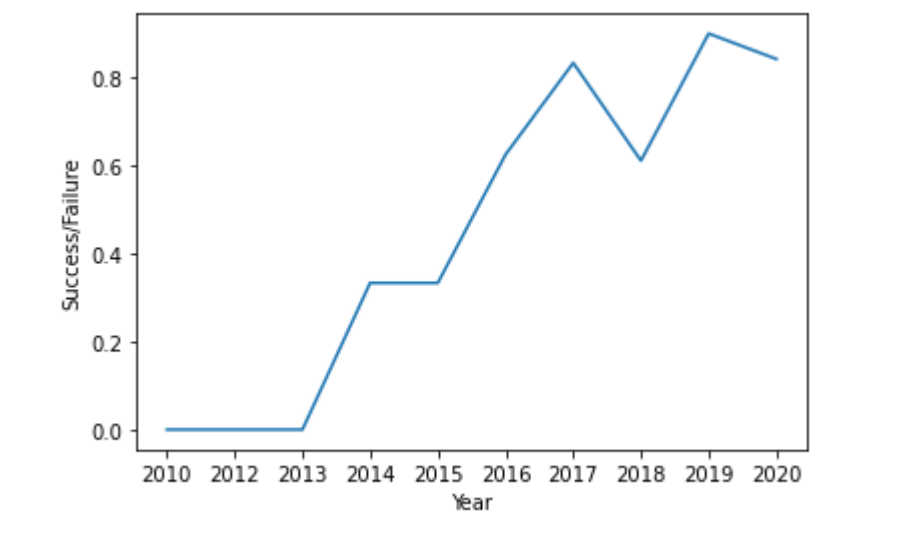
- The graph shows the relationship between flight number and orbit details.
- This shows that GEO has small relationship with flight number.
- While GTO has the highest relationship with flight number.

Payload vs. Orbit Type



- The scatter plot shows the relationship between Pay load mass and orbit outcome.
- The true ASDS give the highest payloadmass

Launch Success Yearly Trend



- The line graph shows the success rate / year of every launch .
- The gradient expresses the trend increases.
- There is a fluctuation between 2017 – 2019.

All Launch Site Names

```
%sql select DISTINCT(LAUNCH_SITE) from spacex
```

```
 * ibm_db_sa://gmz49121:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

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

- The result gives out 3 unique names of the launch names
- I used the distinct function to give out the unique values.

Launch Site Names Begin with 'CCA'

```
%sql select * from SPACEX where LAUNCH_SITE like 'CCA%' limit 5
```

```
 * ibm_db_sa://gmz49121:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

DATE	time_utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing_outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- These launch names are 5 records where launch sites begin with `CCA`
- There are actually more data but I only give only limit 5 for display purposes.

Total Payload Mass

```
%sql select sum(PAYLOAD_MASS__KG_) from spacex where CUSTOMER='NASA (CRS)'
```

```
 * ibm_db_sa://gmz49121:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

1
45596

- Using the sum function I was able to find out the sum of the payloadmass.
- I only took the data where the customer column is ‘NASA (CRS)’

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION='F9 v1.1'
```

```
 * sqlite:///my_data1.db
Done.
```

avg(PAYLOAD_MASS__KG_)
2928.4

- The output is the average payload mass carried by booster version F9 v1.1
- Using the avg function I was able to find out the total average of the payloadmass.

First Successful Ground Landing Date

```
%sql select DATE from spacex where LANDING__OUTCOME = 'Success (ground pad)' order by DATE
```

```
 * ibm_db_sa://gmz49121:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

DATE
2015-12-22
2016-07-18
2017-02-19
2017-05-01
2017-06-03
2017-08-14
2017-09-07
2017-12-15
2018-01-08

- I filtered the column where the landing column is a success.
- I output a date column.
- Then I ordered the date in increasing order

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select BOOSTER_VERSION from SPACEX where LANDING__OUTCOME='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000;
```

```
 * ibm_db_sa://gmz49121:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

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

- Using the ‘between’ method I’m able to range the values between the 2 numbers
- While I output the booster version column

Total Number of Successful and Failure Mission Outcomes

```
%sql select count(MISSION_OUTCOME) as missionoutcomes from SPACEX GROUP BY MISSION_OUTCOME;
```

```
 * ibm_db_sa://gmz49121:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

missionoutcomes
1
99
1

- I used a count function to find the number of mission outcomes.
- I also group the variables so it gives the same data

Boosters Carried Maximum Payload

```
%sql select BOOSTER_VERSION from SPACEX where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEX);
```

```
 * ibm_db_sa://gmz49121:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

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

- Dropdown the booster version column.
- The column have the maximum payloadmass

2015 Launch Records

```
%sql select MONTHNAME(DATE) as Month, landing__outcome, booster_version, launch_site
from SPACEXDATASET where DATE like '2015%' AND landing__outcome like 'Failure (drone ship)'
```

```
 * ibm_db_sa://nxs27972:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/BLUDB
Done.
```

MONTH	landing__outcome	booster_version	launch_site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- I outcome the records on 2015. where the landing outcome is a failure.
- This shows that the launch site was a failure at CCAFS LC- 40.

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

```
%sql SELECT LANDING__OUTCOME FROM SPACEX WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY DATE DESC;
```

```
 * ibm_db_sa://gmz49121:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

landing__outcome
No attempt
Success (ground pad)
Success (drone ship)
Success (drone ship)
Success (ground pad)
Failure (drone ship)
Success (drone ship)
Success (drone ship)
Success (drone ship)
Failure (drone ship)
Failure (drone ship)
Success (ground pad)
Precluded (drone ship)

No attempt
Failure (drone ship)
No attempt
Controlled (ocean)
Failure (drone ship)
Uncontrolled (ocean)
No attempt
No attempt
Controlled (ocean)
Controlled (ocean)
No attempt
No attempt
Uncontrolled (ocean)
No attempt
No attempt
No attempt
Failure (parachute)
Failure (parachute)

- In this code I outcome the landing outcome of the dataset where I used a between method to take the range values in the date column.
- I ordered it by date in descending order

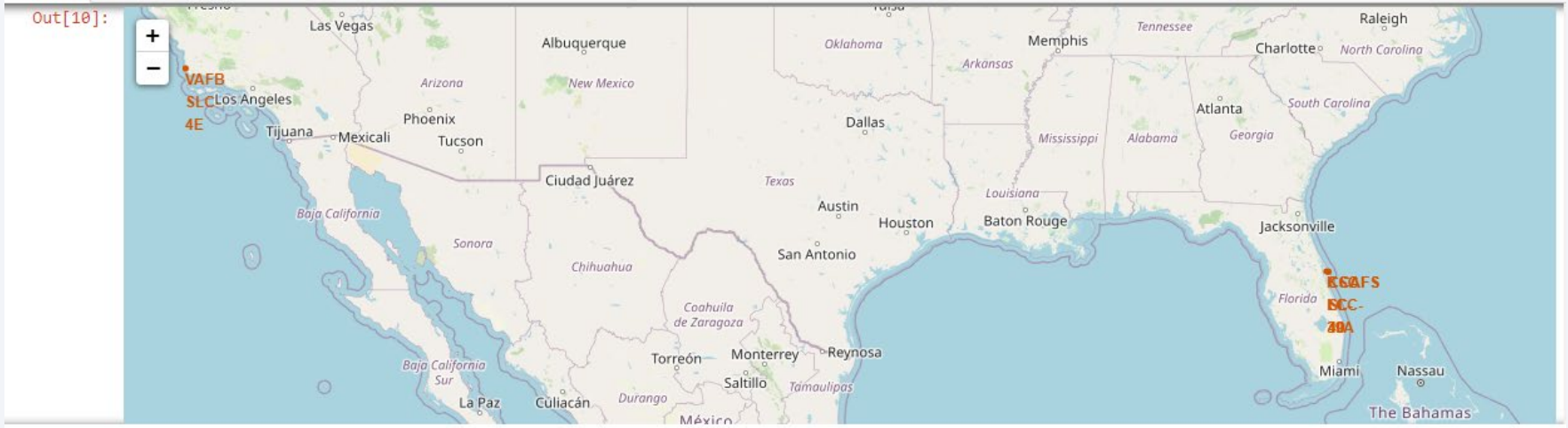
Section 4

Launch Sites Proximities Analysis

all launch sites on a map

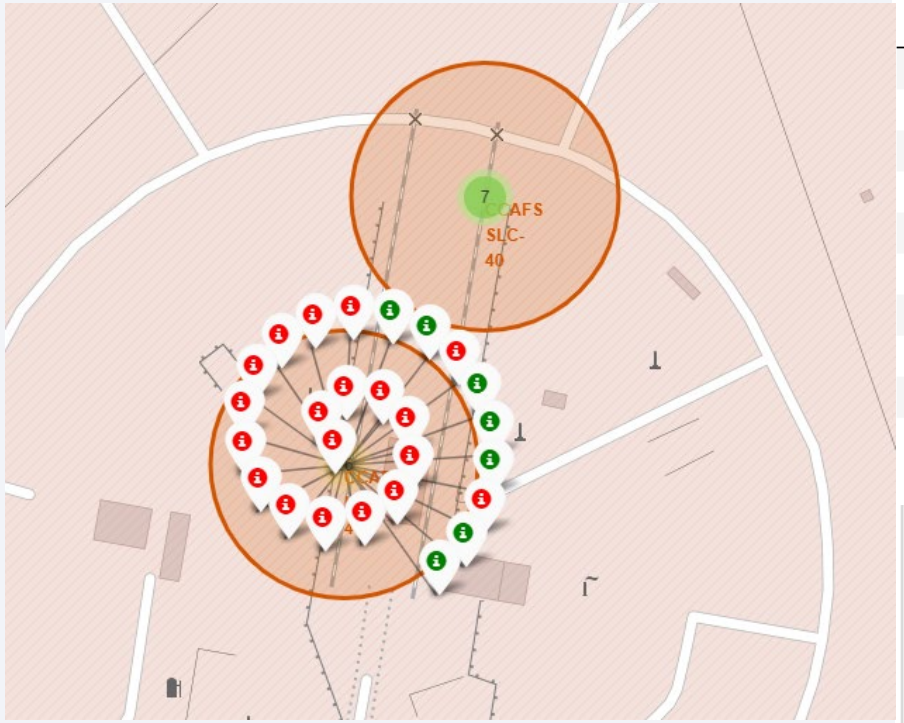
```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch site name as a popup label
for i in range (len(launch_sites_df.index)):
    coordinate = [launch_sites_df["Lat"][i], launch_sites_df["Long"][i]]
    circle = folium.Circle(coordinate, radius=100, color='#d35400', fill=True).add_child(folium.Popup(launch_sites_df["Launch Site"][i]))
    marker = folium.map.Marker(
        coordinate,
        icon=DivIcon(
            icon_size=(20,20),
            icon_anchor=(0,0),
            html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % launch_sites_df["Launch Site"][i],
            )
        )
    site_map.add_child(circle)
    site_map.add_child(marker)
```

Out[10]:



- There are 3 points in the map that represent the launching site for the space X rocket

Marking the success/failed launches for each site on the map



	Launch Site	Lat	Long	class	marker_color
46	KSC LC-39A	28.573255	-80.646895	1	green
47	KSC LC-39A	28.573255	-80.646895	1	green
48	KSC LC-39A	28.573255	-80.646895	1	green
49	CCAFS SLC-40	28.563197	-80.576820	1	green
50	CCAFS SLC-40	28.563197	-80.576820	1	green
51	CCAFS SLC-40	28.563197	-80.576820	0	red
52	CCAFS SLC-40	28.563197	-80.576820	0	red
53	CCAFS SLC-40	28.563197	-80.576820	0	red
54	CCAFS SLC-40	28.563197	-80.576820	1	green
55	CCAFS SLC-40	28.563197	-80.576820	0	red

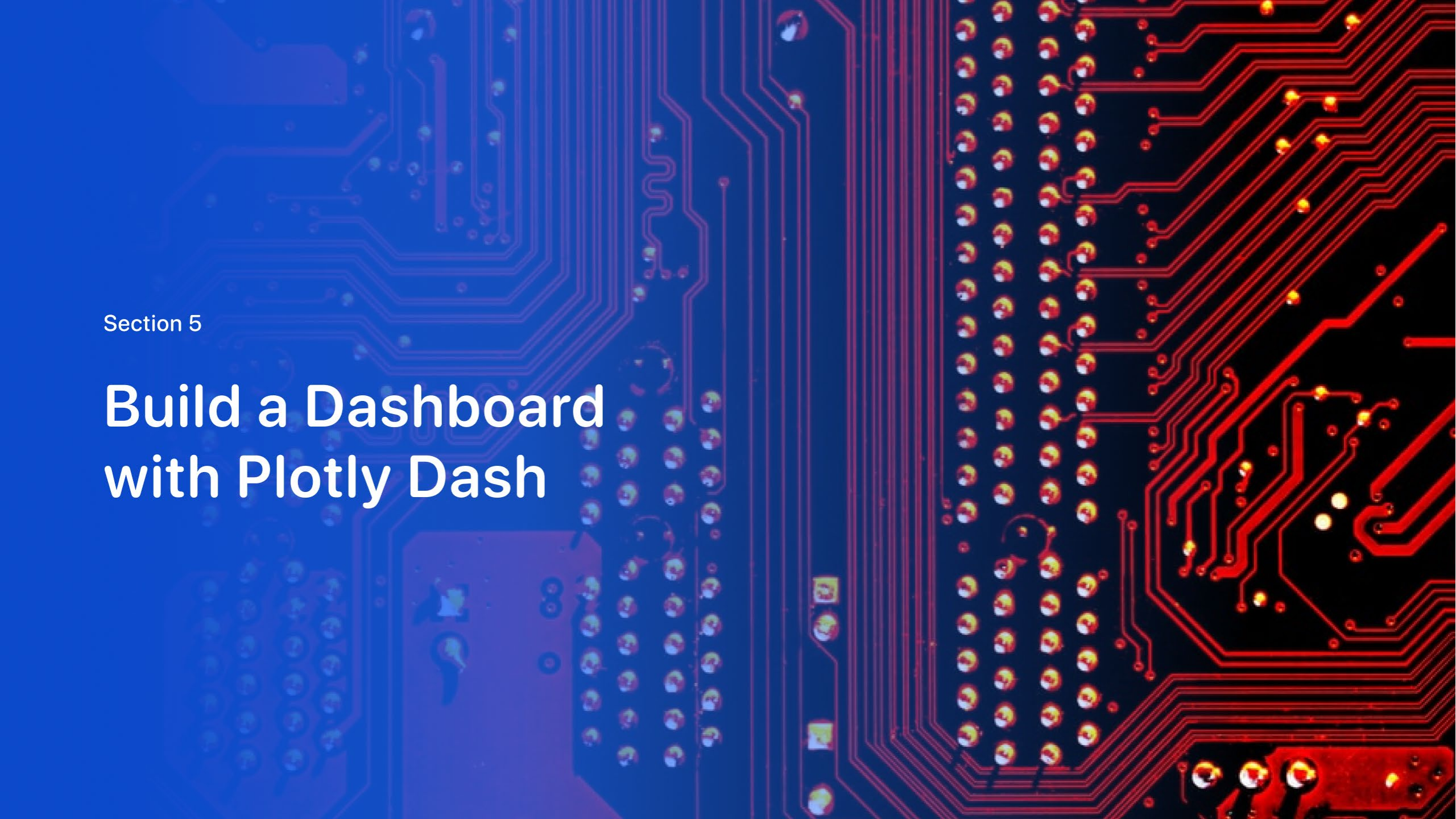
```
site_map.add_child(marker_cluster)

for index, record in spacex_df.iterrows():
    marker = folium.Marker([record['Lat'], record['Long']],
                   icon=folium.Icon(color='white', icon_color=record['marker_color']))
    marker_cluster.add_child(marker)
site_map
```

- The markers represent the successful and failed launches, in this map I only represent one of the location in the map as an example.
- The green represent successful, while the red represent fail mark

Calculating the distances between a launch site to its proximities

- The line in the map represents the distance to the nearest shoreline
- We will use some mathematics to figure out the length of the line.
- In this example its only 9 KM

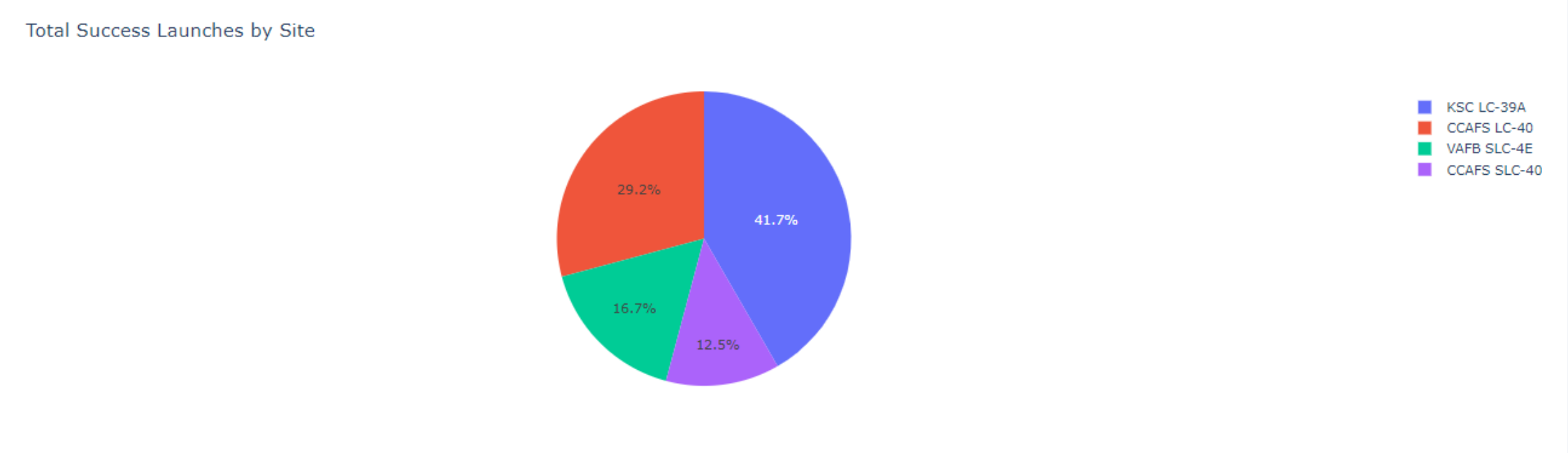
```
4]: #Work out distance to coastline
coordinates = [
    [28.56342, -80.57674],
    [28.56342, -80.56756]]

lines=folium.PolyLine(locations=coordinates, weight=1)
site_map.add_child(lines)
distance = calculate_distance(coordinates[0][0], coordinates[0][1], coordinates[1][0], coordinates[1][1])
distance_circle = folium.Marker(
    [28.56342, -80.56794],
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % "{:10.2f} KM".format(distance),
        )
    )
site_map.add_child(distance_circle)
site_map
```



Section 5

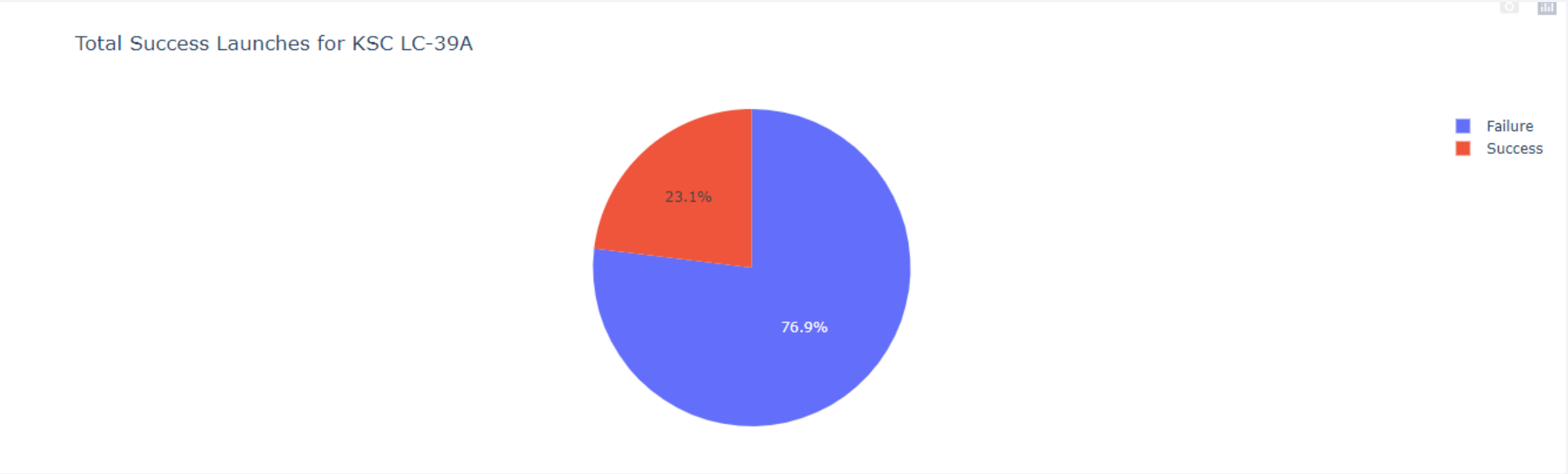
Build a Dashboard with Plotly Dash

screenshot of launch success count for all sites, in a piechart



- This pie chart represent the launch success count for each sites in the location.
- This shows the KSC LC-39A gives the highest success count of 41.7%
- And the CCAFS SLC-40 is the lowest success launch with 12.55.

pie chart for the launch site with highest launch success ratio



- Here is the relationship of the success and fail launch site of KSC LC 39A
- The Ratio is 76.9/23.1 = 3.3 success ratio

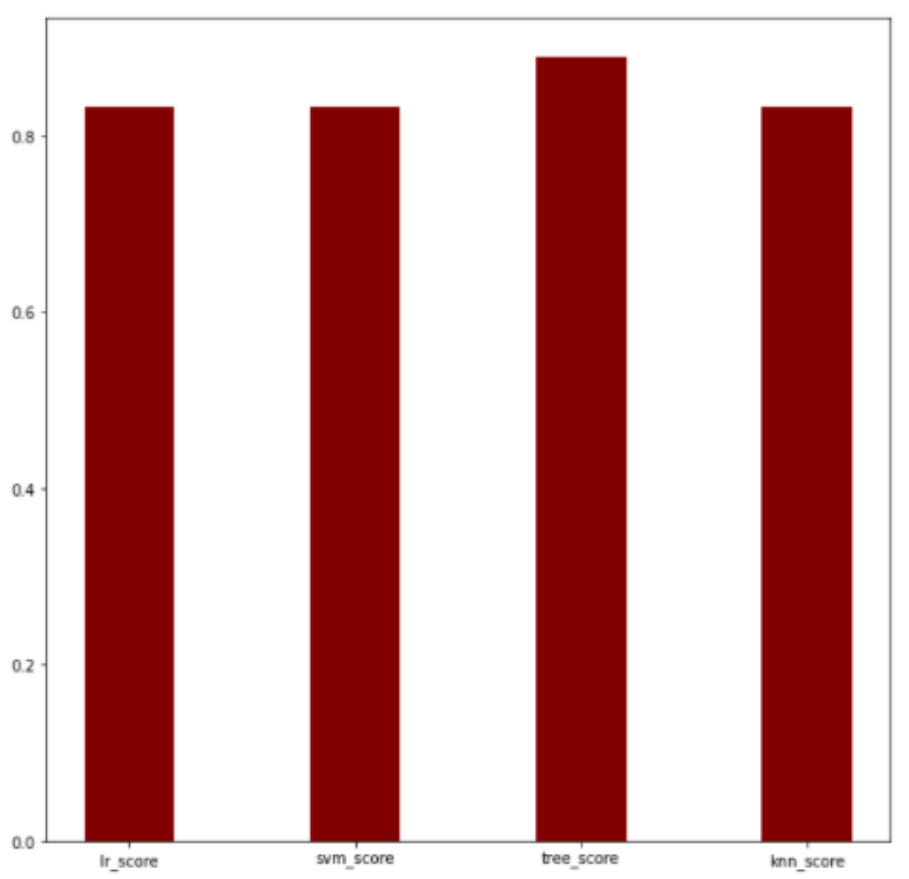
Payload vs. Launch Outcome scatter plot for all sites

- The screenshots shows the relationship of the payload mass and the launch site class.
- The slider represent different ranges of pay load mass values
- The top one range is the full range of the data, a high payload mass, this shows that it has a strong relationship compared to the bottom screenshot which I put to a low range of payload mass. In the bottom screen shot, there is less relationship with the launch class.

Section 6

Predictive Analysis (Classification)

Classification Accuracy




```
scores_x = ["lr_score","svm_score","tree_score","knn_score"]
scores = [lr_score,svm_score,tree_score,knn_score]
print(scores)
print(scores.index(max(scores)))
```

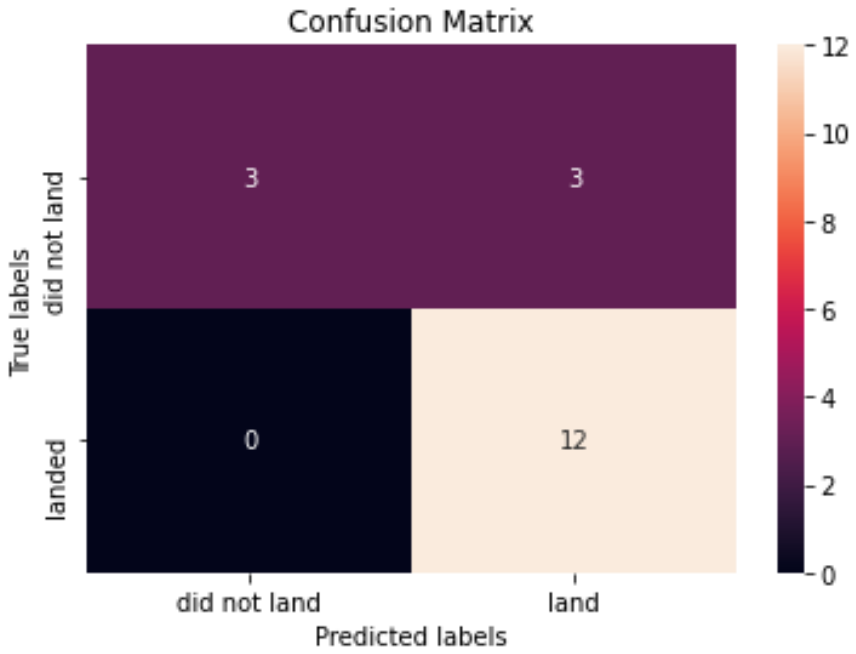
```
[0.8333333333333334, 0.8333333333333334, 0.8888888888888888, 0.8333333333333334]
2
```

```
fig = plt.figure(figsize = (10, 10))
plt.bar( scores_x, scores, color ='maroon',
        width = 0.4)
```

```
<BarContainer object of 4 artists>
```

- After Analyzing through each module, the highest accuracy was the tree score module. With an accuracy of 0.89

Confusion Matrix




```
tree_score = tree_cv.score(X_test,Y_test)
tree_score
```

```
3]: 0.8888888888888888
```

We can plot the confusion matrix

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

- Using the plot_confusion_matrix function I was able to create the confusion Matrix module.
- This will evaluate the tree score module.

Conclusions

- The best place for the launch site based on the Space X dataset, is at the location KSC LC 39A because of its environmental features based from its data.
- After training and testing the data sheet I was able to build a machine learning module on four different types and the highest accuracy was the tree score module.
- EDA Insights data tells us that the payload mass (Kg) is directly proportional to the successful launch rate (%).

Appendix

- Information about the history of the falcon 9: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
- Data collection: https://github.com/RaihanTeachesCodes/IBM_course_10-/blob/master/data%20collection.ipynb
- Data wrangling: https://github.com/RaihanTeachesCodes/IBM_course_10-final-project/blob/master/data%20wrangling.ipynb
- Data visualization: https://github.com/RaihanTeachesCodes/IBM_course_10-final-project/blob/master/EDA%20with%20Data%20Visualization.ipynb
- EDA SQL: https://github.com/RaihanTeachesCodes/IBM_course_10-final-project/blob/master/eda%20sql.ipynb
- Data visualization with folium: https://github.com/RaihanTeachesCodes/IBM_course_10-final-project/blob/master/interactive%20visualization%20with%20folium.ipynb
- Space dash APP: https://github.com/RaihanTeachesCodes/IBM_course_10-/blob/master/spacex_dash_app.py
- Machine Learning modules: https://github.com/RaihanTeachesCodes/IBM_course_10-/blob/master/Machine%20Learning%20Prediction.ipynb
- My repositories: https://github.com/RaihanTeachesCodes/IBM_course_10-final-project

Thank you!