



IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Hariharasudhan Rajaguru
23-03-2022



Outline

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

Outline

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

Executive Summary

- Summary of methodologies
 - Data collection
 - Data wrangling
 - EDA with data visualization
 - EDA with SQL
 - Creating Folium map to visualize successful landing rates
 - Creating Dash to visualize the successful landing factors
 - Predictive analysis
- Summary of all results
 - Exploratory data analysis results
 - Interactive Folium map screenshots
 - Predictive analysis results

Introduction

- Project background and context

We predicted if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

- Problems you want to find answers
 1. What are the factors playing major role to safely launch the first stage successfully
 2. Finding relationships with each variable for a successful landing
 3. What are the variable measures has to taken for a successful landing



Section 1

Methodology

Methodology

Executive Summary

Data collection methodology:

- SpaceX rest API
- Web scrapping from Wikipedia
- Perform data wrangling
 - For Machine learning used one hot encoding and dropped irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL
- Plotting Scatter plots, Cat plots and Bar graphs to understand the relationships between the independent and the dependent variables
- Performing interactive visual analytics using Folium and Plotly Dash to find the successful landing sites and the key factors
- Perform predictive analysis using classification models
 - How to build, tune, evaluate classification models

Data Collection

- We worked with SpaceX launch data that is gathered from the SpaceX REST API.
- This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
- The SpaceX REST API endpoints, or URL, starts with `https://api.spacexdata.com/v4/rockets/`
- Another popular data source for obtaining Falcon 9 Launch data is web scraping Wikipedia using BeautifulSoup.

Data Collection – SpaceX API

- Collecting data from Space X API as a .Json file

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

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

- Converting the .json file into Pandas dataframe using .json_normalize method

```
#Use json_normalize meethod to convert the json result into a dataframe  
data=pd.json_normalize(response.json())
```

- Filtering datas of falcon 9

```
data_falcon9=launch_d[launch_d.BoosterVersion!='Falcon 1']
```

- Replacing payloadmass column npNaN values with payloadmass column mean value

```
# Calculate the mean value of PayloadMass column  
mean_1=data_falcon9['PayloadMass'].mean()  
#mean_1  
# Replace the np.nan values with its mean value  
data_falcon9['PayloadMass']=data_falcon9['PayloadMass'].fillna(mean_1)  
data_falcon9
```

<https://github.com/HariharasudhanRajaguru-DS/Capstone-Project/blob/main/jupyter-labs-spacex-data-collection-api.ipynb>

Using SpaceX API to collect the data



SpaceX data collected in .JSON file



Normalizing the data using
pd.json_normalize()



Filtering data fields for Falcon9

Data Collection - Scraping

- Scraping web data from the URL:

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

- Using HTTP request.get method to collect the data from the web page

```
# use requests.get() method with the provided static_url  
# assign the response to a object  
response=requests.get(static_url).text
```

- Creating a BeautifulSoup object from a response text content

```
soup=BeautifulSoup(response,"html.parser")
```

- Extracting all columns/variable name form the html table header

```
html_tables=soup.find_all('table')
```

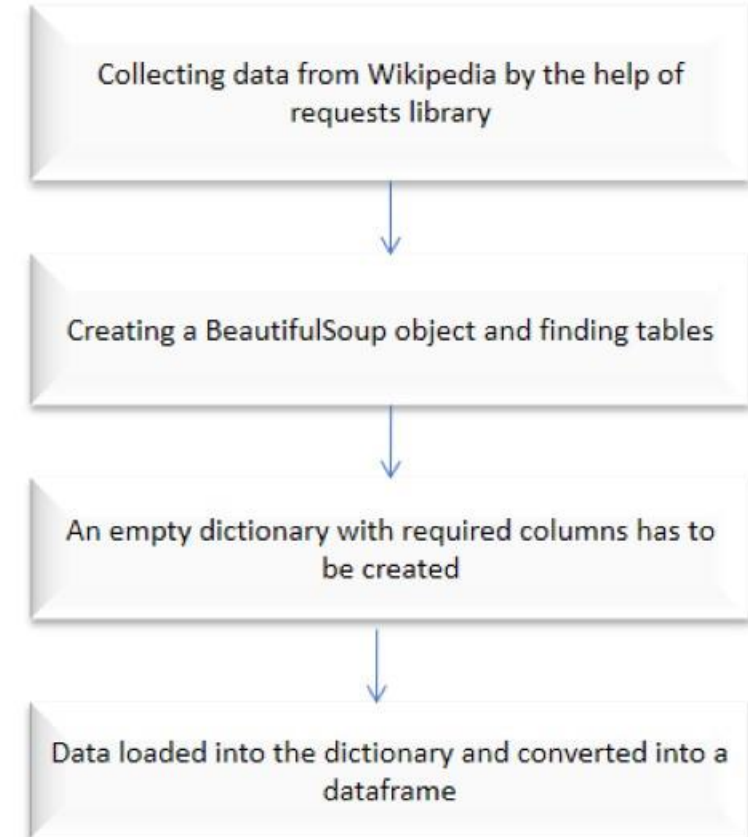
- Creating an empty dictionary to load the data

```
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']=[]
```

- Converting the dictionary into a pandas dataframe

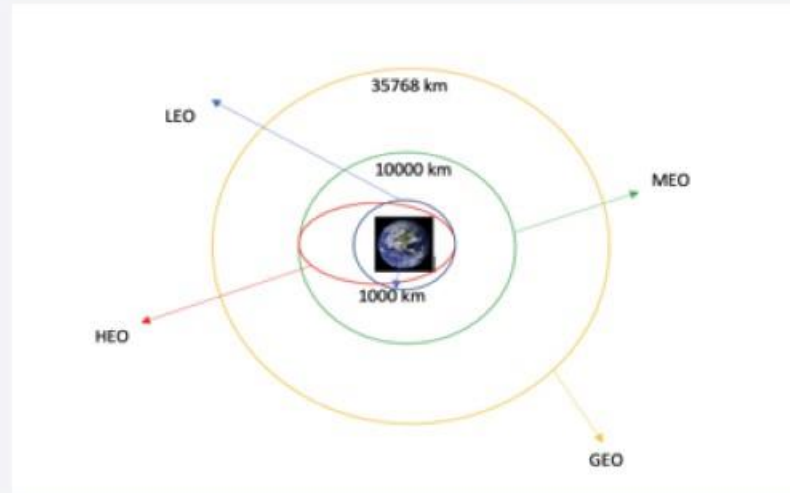
```
df=pd.DataFrame(launch_dict)
```

https://github.com/HariharasudhanRajaguru-DS/Capstone-Project/blob/main/CapstoneProject_Web%20Scraping.ipynb

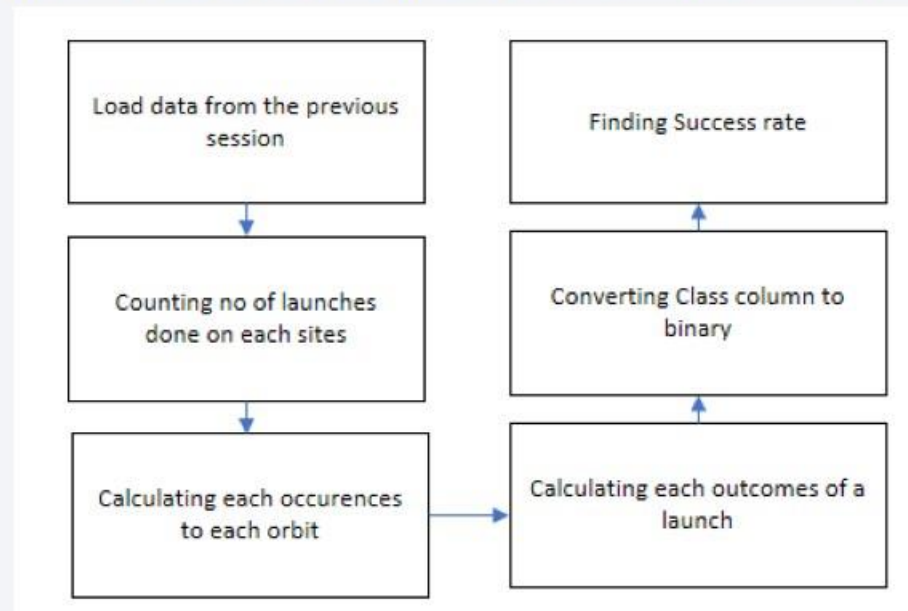


Data Wrangling

- There are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad. True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.



Orbit types and used by SpaceX



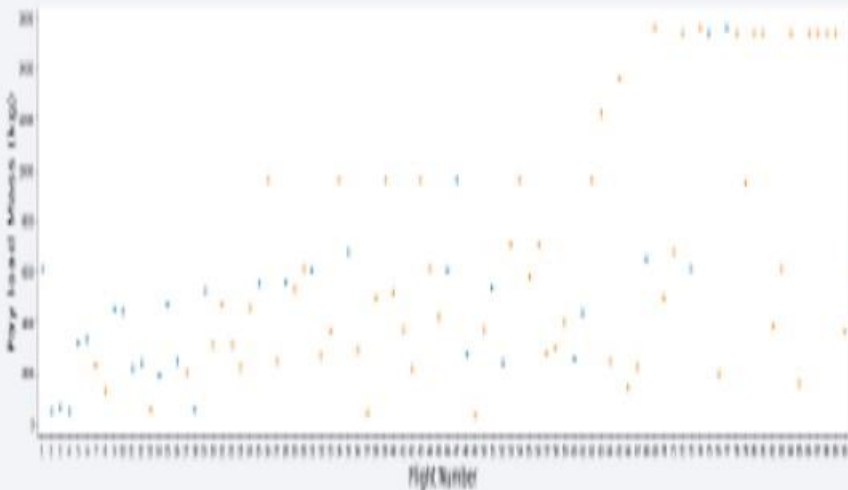
https://github.com/HariharasudhanRajaguru-DS/Capstone-Project/blob/main/Capstone_project-week1_Data%20Wrangling.ipynb

EDA with Data Visualization

Scatter point

- 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

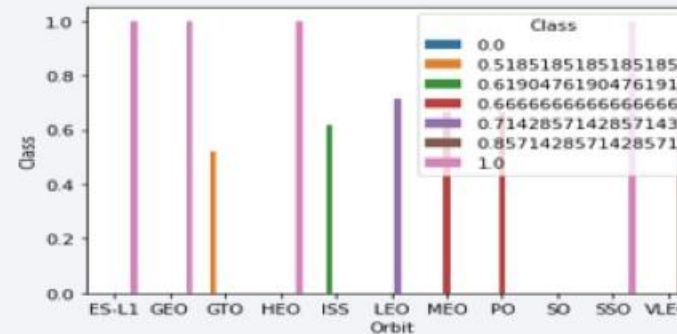
Here we can find the variables and how It's contributing to a successful landing.



Bar Graph

- Class column Mean vs Orbit

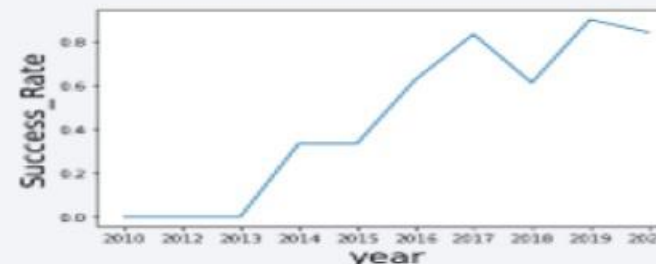
Helps to find which orbit has a huge success rate.



Line Graph

- Success rate VS Year

Using a line graph we can observe that the success rate since 2013 kept increasing till 2020.



https://github.com/HariharasudhanRajaguru-DS/Capstone-Project/blob/master/Capstone_project-week2-DataAnalysis_EDA_matplotlib.ipynb

EDA with SQL

Data has been loaded in IBM DB2 using load data, Made DB connection, Queries has been executed in Jupiter notebook using magic SQL

- Querying the names of the unique launch sites
- Querying 5 records where launch sites begin with the string 'KSC'
- Querying the total payload mass carried by boosters launched by NASA (CRS)
- Querying average payload mass carried by booster version F9 v1.1
- Querying the date where the successful landing outcome in the drone ship was achieved.
- Querying the names of the boosters which have success in the ground pad and have payload mass greater than 4000 but less than 6000
- Querying the total number of successful and failure mission outcomes
- Querying the names of the booster versions which have carried the maximum payload mass.
- Querying the records which will display the month names, successful landing_outcomes in ground pad, booster versions, launch_site for the months in the year 2017
- Querying the count of successful landing_outcomes between the dates 2010-06-04 and 2017-03-20 in descending order.

https://github.com/HariharasudhanRajaguru-DS/Capstone-Project/blob/master/Capstone_project-week2-Data_Analysis.ipynb

Build an Interactive Map with Folium

- Using Latitude and Longitude Coordinates at each launch site and added a Circle Marker around each launch site with a label of the name of the launch site.
- We assigned the data frame `launch_outcomes(failures, successes)` to classes 0 and 1 with Green and Red markers on the map using `MarkerCluster()`
- We calculated the distance with the help of Haversine's formula from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns. We drew the lines on the map to measure the distance.
- Analysing the following scenarios.
- Are launch sites in close proximity to railways?
- Are launch sites in close proximity to highways?
- Are launch sites in close proximity to the coastline?
- Do launch sites keep a certain distance away from cities?

https://github.com/HariharasudhanRajaguru-DS/Capstone-Project/blob/master/CapstoneProject_week3-FoliumMap.ipynb

Build a Dashboard with Plotly Dash

- The dashboard is built with Flask and Dash web framework.
- Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions
- shows the relationship between two variables.
- the best method to show you a non-linear pattern.
- The range of data flow, i.e. maximum and minimum value, can be determined.
- Observation and reading are straightforward.

<https://github.com/HariharasudhanRajaguru-DS/Capstone-Project/blob/main/Ploty%20Dash%20capstone%20project.txt>

Predictive Analysis (Classification)

BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

EVALUATING MODEL

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithm
- Plot Confusion Matrix

IMPROVING MODEL

- Feature Engineering
- Algorithm Tuning

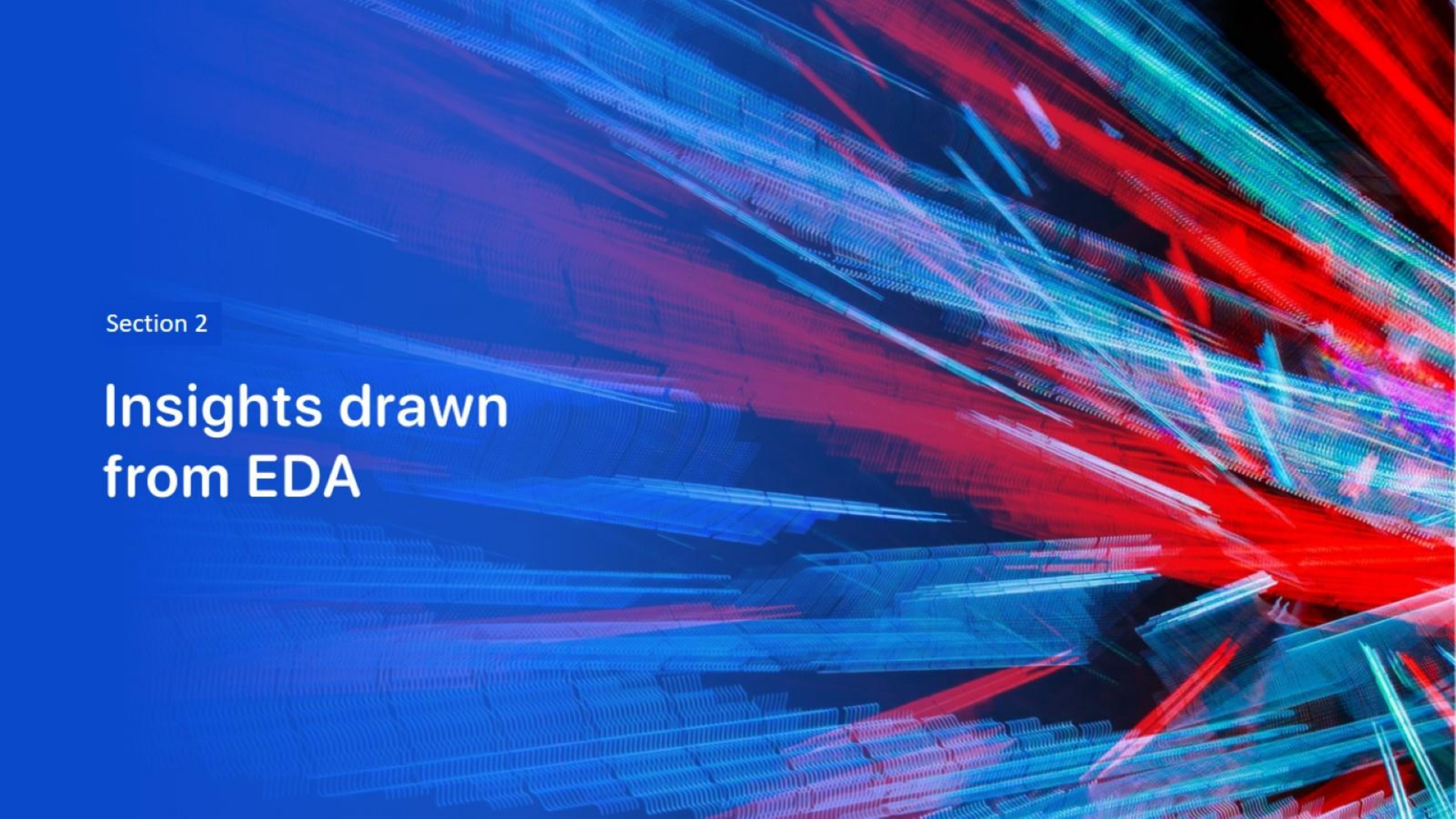
FINDING THE BEST PERFORMING CLASSIFICATION MODEL

- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

https://github.com/HariharasudhanRajaguru-DS/Capstone-Project/blob/master/capstone_project-week4.ipynb

Results

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

The background of the slide is an abstract composition. It features a solid blue area on the left side, which transitions into a dynamic pattern of diagonal streaks in shades of blue, red, and cyan on the right. These streaks have a textured, almost woven appearance, suggesting a digital or data-driven theme. The overall effect is one of movement and complexity.

Section 2

Insights drawn from EDA

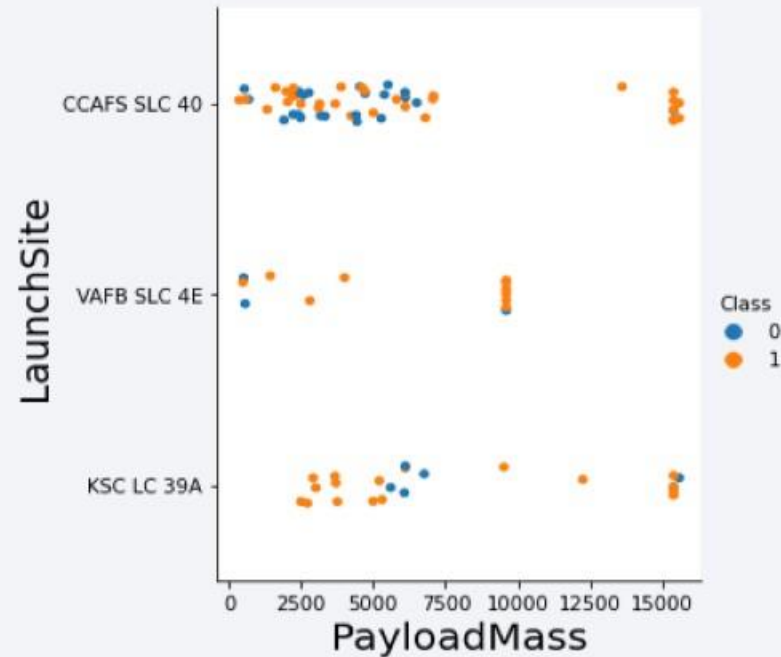
Flight Number vs. Launch Site



scatter plot of Flight Number vs. Launch Site

- From the above diagram we can clearly find CCAFS SLC 40 launching site has the highest no of launches, and followed by KSC LC 39A, and followed by VAFBSLC 4E.

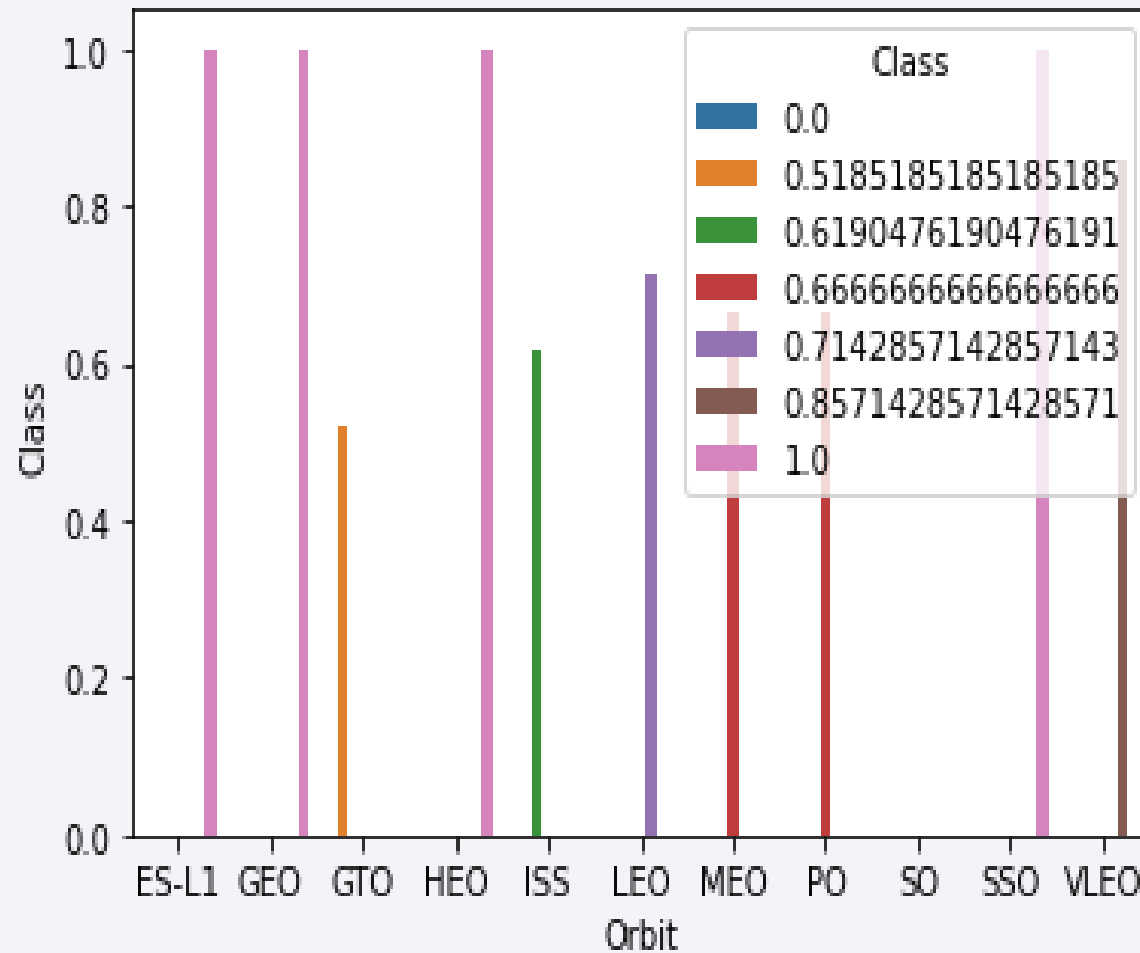
Payload vs. Launch Site



scatter plot of Payload vs. Launch Site

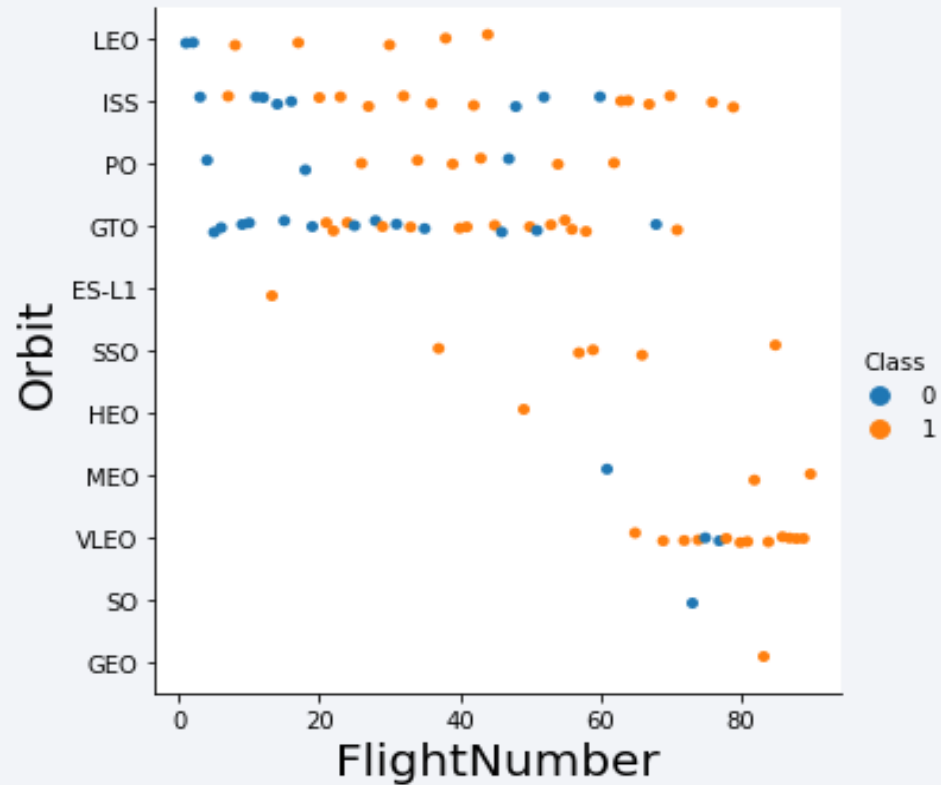
From the above plot, we can find that from CCAFS SLC 40 launching site the flight's payload mass has the highest of 15000.

Success Rate vs. Orbit Type



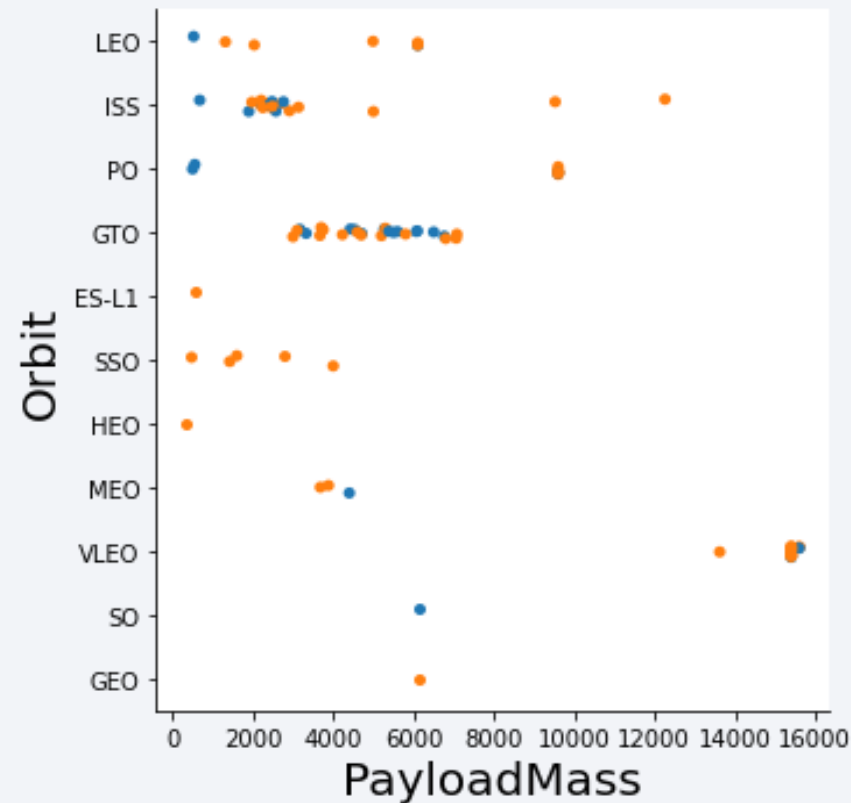
From the bar chart, we can find orbits GEO, HEO, SSO, ES-L1 has the Best Success Rate

Flight Number vs. Orbit Type



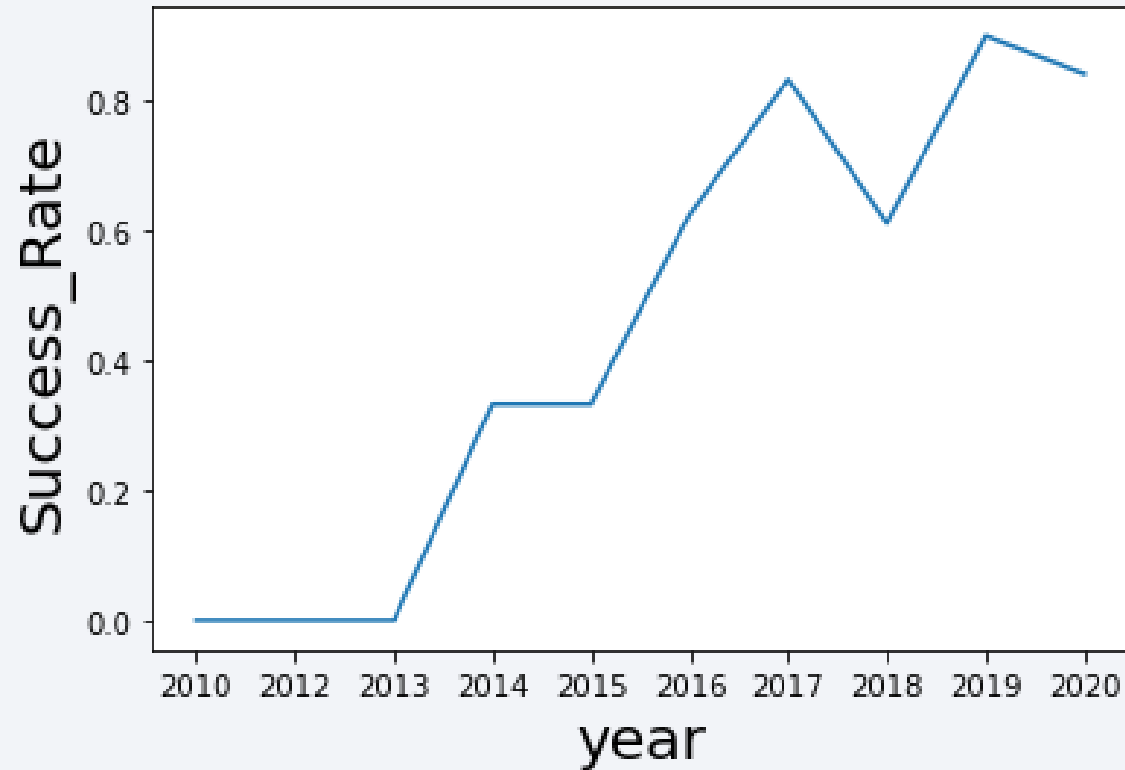
As you see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight numbers when in GTO orbit.

Payload vs. Orbit Type



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

Launch Success Yearly Trend



We can observe that the success rate since 2013 kept increasing till 2020

All Launch Site Names

```
In [14]: %sql select distinct(launch_site) as uni_launch_site from spacex;
```

```
* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb  
Done.
```

```
Out[14]:
```

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

Using a distinct function on the launch_site column we can get the unique launching site names without duplicates.

Launch Site Names Begin with 'CCA'

In [17]: `%sql select * from spacex where launch_site like 'CCA%' limit 5;`

`* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb`
Done.

Out[17]:

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	None	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	None	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	None	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	None	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	None	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Here we have used the like function to get the launch_site column names starting with 'CCA%'. Here 'CCA%' means it can be anything after these letters. And also we are using the Limit function to limit the output as 5 records to display.

Total Payload Mass

```
In [25]: %sql Select sum(PAYLOAD_MASS__KG_) from spacex where customer = 'NASA (CRS)';
```

```
* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb  
Done.
```

```
Out[25]:
```

1
45596

By using the sum function on the payload_mass__kg column we can sum up the total payload. And also while using, where customer= 'NASA (CRS)' condition we can clearly calculate the total payload mass.

Average Payload Mass by F9 v1.1

```
In [26]: %sql select avg(PAYLOAD_MASS_KG_) from spacex where booster_version='F9 v1.1';
* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8l1cg.databases.appdomain.cloud:32536/bludb
Done.
```

```
Out[26]:
```

1
2928

With the help of the avg function, we can calculate the average value of a particular column. Here we are about to calculate the avg payload_mass_kg_ for the booster version of F9v1.1. Filtering using the where condition on the booster_version column we can easily calculate the avg payload mass.

First Successful Ground Landing Date

```
In [40]: %sql select min(Date) from spacex where landing__outcome='Success (ground pad)';
```

```
* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb  
Done.
```

```
Out[40]:
```

1
2015-12-22

Using the min function on the date column we can get the latest date entry from the column. Here we need to find the outcome 'Success (ground pad)'. So we are using this condition in where clause to filter the result.

Successful Drone Ship Landing with Payload between 4000 and 6000

```
In [41]: %sql select booster_version from spacex where landing__outcome='Success (drone ship)' and payload_mass__kg_>4000 and payload_mas  
s__kg_<6000;
```

```
* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb  
Done.
```

```
Out[41]:
```

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

Here our result has to meet 3 conditions. Those are landing_outcome must be success drone ship and payload mass must be greater than 4000 and lesser than 6000.

Total Number of Successful and Failure Mission Outcomes

```
In [44]: %sql select mission_outcome,count(*) as cnt from spacex group by mission_outcome;
```

```
* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb  
Done.
```

```
Out[44]:
```

mission_outcome	cnt
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

We have to find the no the success and failure outcomes. So I have used count and group by functions on the mission_outcome column so that we can clearly get the result. Here count function counts the outcome by grouping each to a group.

Boosters Carried Maximum Payload

```
In [56]: %%sql
select booster_version, payload_mass_kg_
from spacex where payload_mass_kg_=(select max(payload_mass_kg_) from spacex)
order by 1 asc;

* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8l1cg.databases.appdomain.cloud:32536/bludb
Done.
```

Out[56]:

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

Here I'm using a subquery to find the maximum payload, that matches that payload mass to the booster version which was carried that amount of payload.

2015 Launch Records

```
In [58]: %%sql
select landing__outcome,booster_version,launch_site
from spacex
where year(date)=2015
and landing__outcome like 'Failure (drone ship)';

* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

```
Out[58]:
```

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

We have to find the failure drone ship landing outcome in the year 2015.

By using the year function on the date column we can filter the year 2015 and by using landing outcome as failure drone ship fetched the data we have required.

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

```
In [61]: %%sql
select landing__outcome,count(*) cnt
from spacex
where date between '2010-06-04' and '2017-03-20'
group by landing__outcome
order by 2 desc;

* ibm_db_sa://ycy02072:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/bludb
Done.
```

```
Out[61]:
```

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

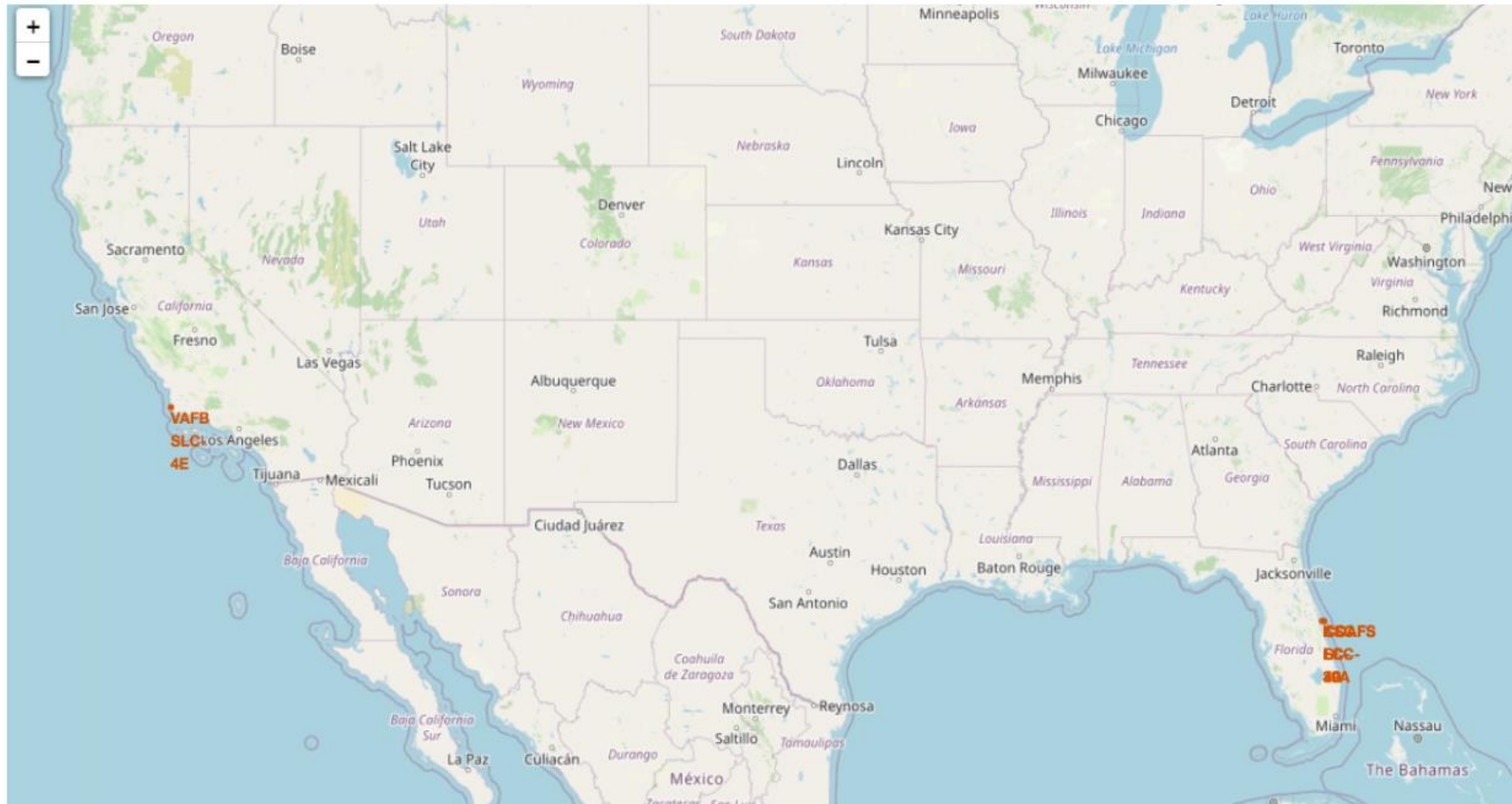
Here I've grouped landing outcomes using group by function and using count function on the landing outcome column then used DESC function we can order in descending order ranking.

A satellite view of Earth from space, showing the curvature of the planet and the glowing city lights of the Eastern United States and parts of Canada at night. The background is a deep blue gradient.

Section 3

Launch Sites Proximities Analysis

Marking launch sites and analyzing



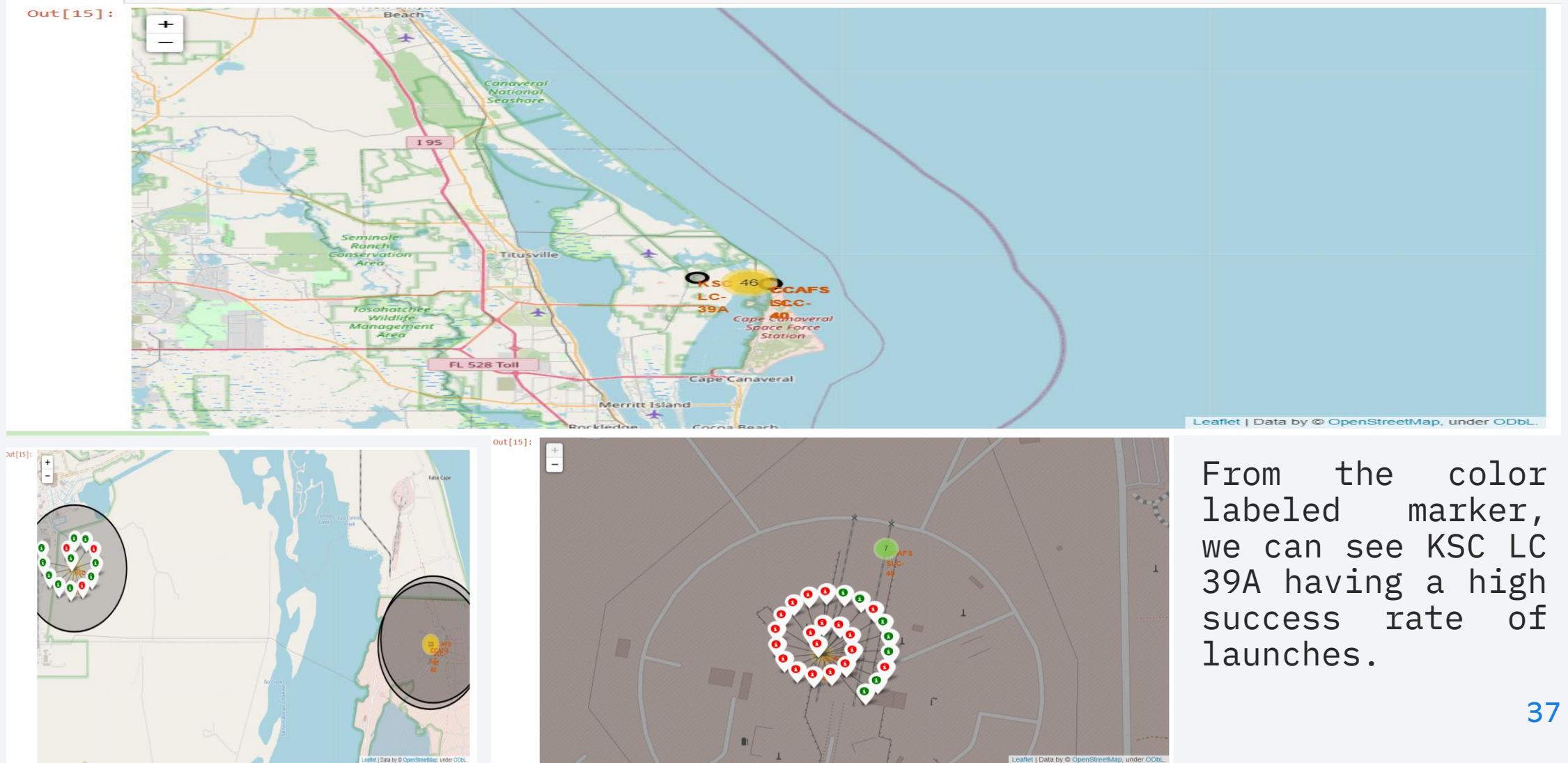
- Are all launch sites in proximity to the Equator line?

Yes. Because it is very helpful to reach the orbit since the distance to orbit from the Equator is less comparatively from other lines.

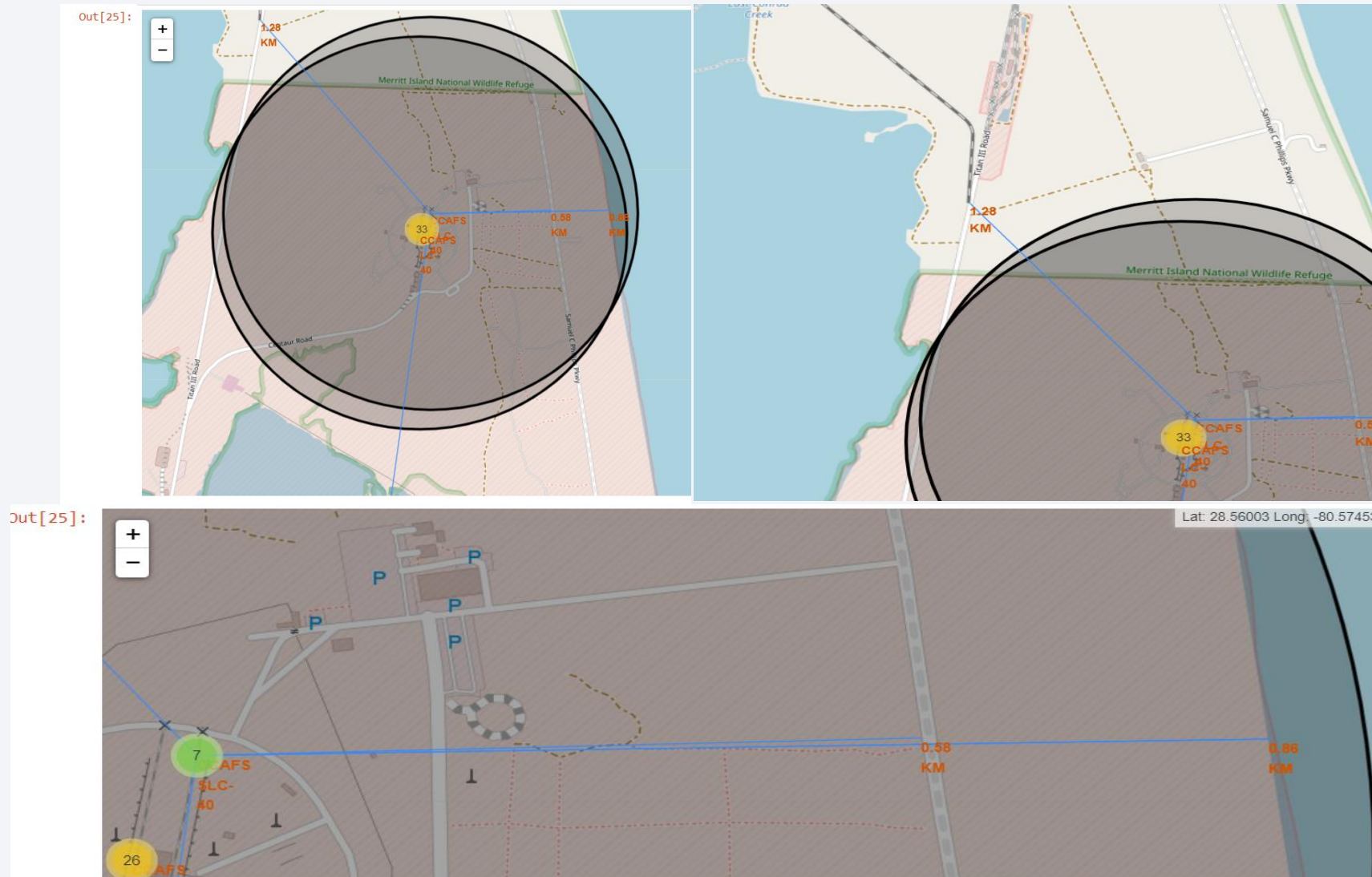
- Are all launch sites in very close proximity to the coast?

Yes. All launch sites are in very close proximity to the coast. So that the stages of the flight will not fall off the land.

Successful and failure launches of each sites



Folium Map launch site to its proximities



1. Launch sites are in close proximity to the equator to minimize fuel consumption by using Earth's $\sim 30\text{km/sec}$ eastward spin to help spaceships get into orbit.
2. Launch sites are in close proximity to the coastline so they can fly over the ocean during launch, for at least two safety reasons-- (1) crew has the option to abort launch and attempt water landing (2) minimize people and property at risk from falling debris.
3. Launch sites are in close proximity to highways, which allows for easy transport required people and property.
4. Launch sites are in close proximity to railways, which allow transport for heavy cargo.
5. Launch sites are not in close proximity to cities, which minimizes danger to population-dense areas.



Section 4

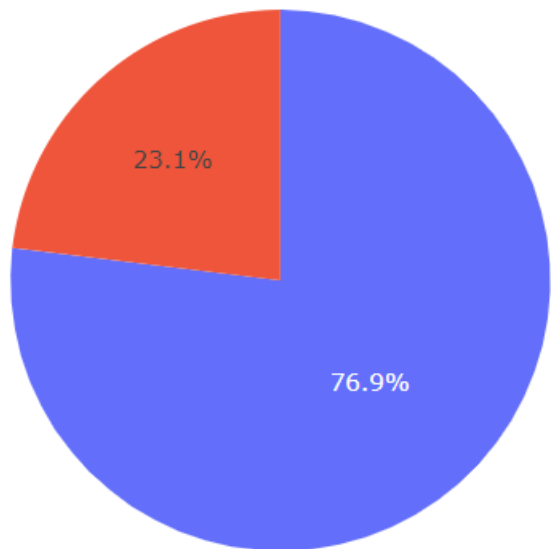
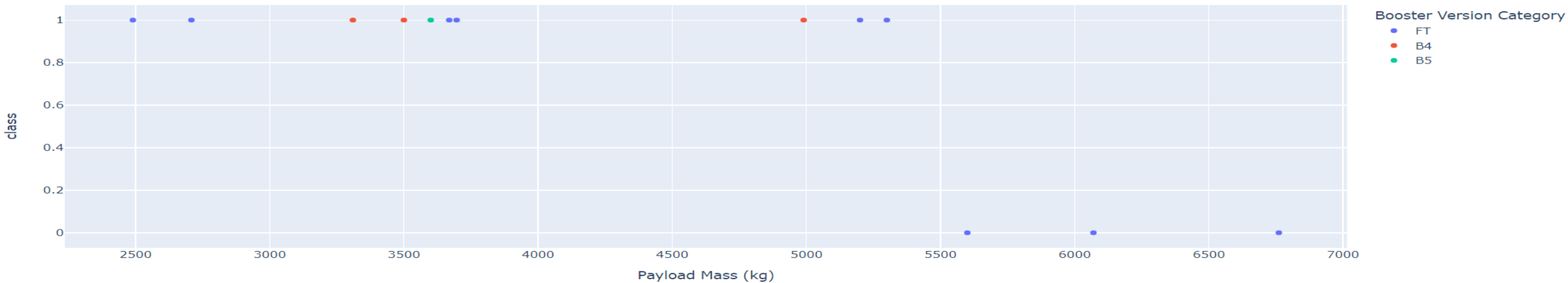
Build a Dashboard with Plotly Dash

Finding success count from all sites



By this pie chart, we can jump to the conclusion of KSC LC-39A has the highest success rate compared to other launch sites.

KSC LC-39A Launch site analysis

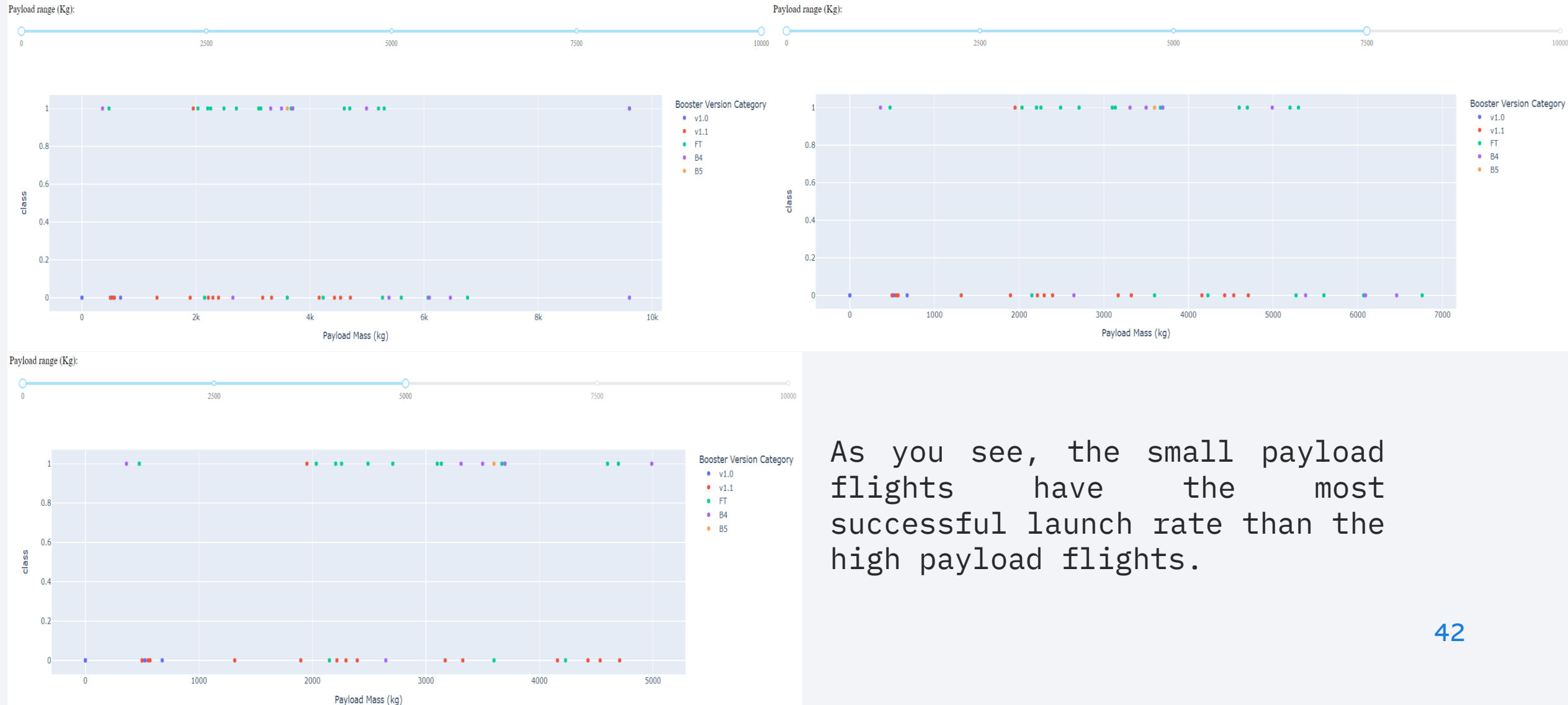


From the charts, we came to know that the launch site has a very good success rate.

Different types of booster versions has been launched from this site.

But the payload never exceeded 10000 kg.

Payload vs. Launch Outcome scatter plot for all sites

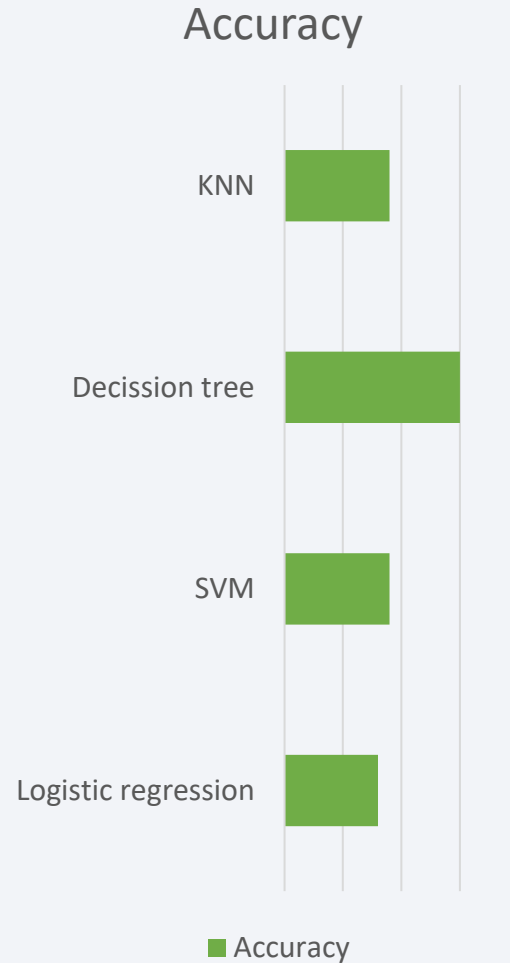


As you see, the small payload flights have the most successful launch rate than the high payload flights.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

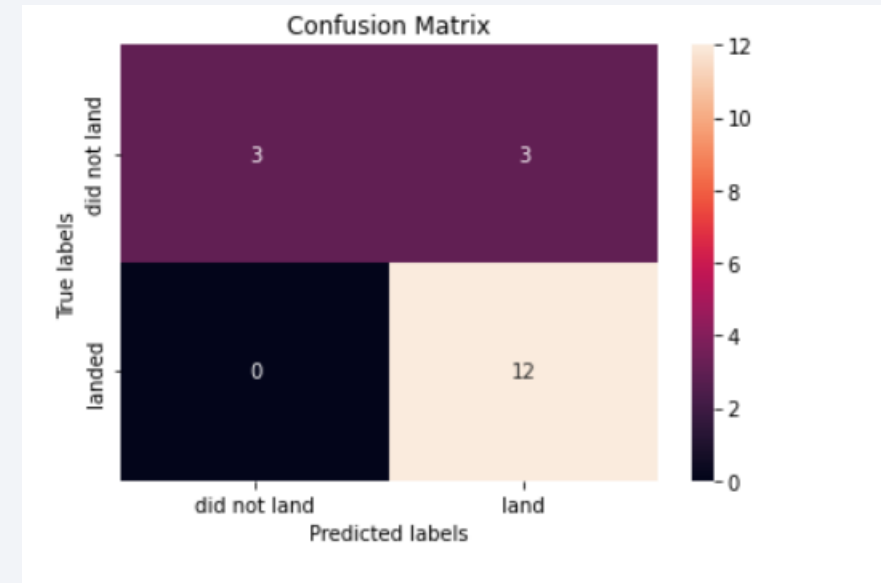


This chart, clearly defines that all the models are performing nearly the same.

Compared to other models decision tree has a high accuracy level.

Confusion Matrix

We can see that Tree can distinguish between the different classes. We see that the major problem is false positives.



Conclusions

- i. The Tree Classifier Algorithm is the best for Machine Learning for this dataset
- ii. Low weighted payloads perform better than the heavier payloads
- iii. The success rates for SpaceX launches is directly proportional to time in years they will eventually perfect the launches
- iv. We can see that KSC LC-39A had the most successful launches from all the sites
- v. Orbit GEO, HEO, SSO, ES-L1 has the Best Success Rate

Appendix

References went through

<https://python.swaroopch.com/>

<https://dash.plotly.com/>

<https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761>

<https://scikit-learn.org/stable/modules/tree.html>

Thank you!

