IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Bernardo Melo
02/20/2025

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- **Summary of all results**
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context
    - 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.
- Problems you want to find answers
    - Can we predict whether the first stage will land successfully?
    - Can we use this info to define a launch cost?
    - What factors influence the successful landing of the Falcon 9 first stage?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected through SpaceX API and Wikipedia scrapping.
- Perform data wrangling
  - One hot encode applied on categorical features.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

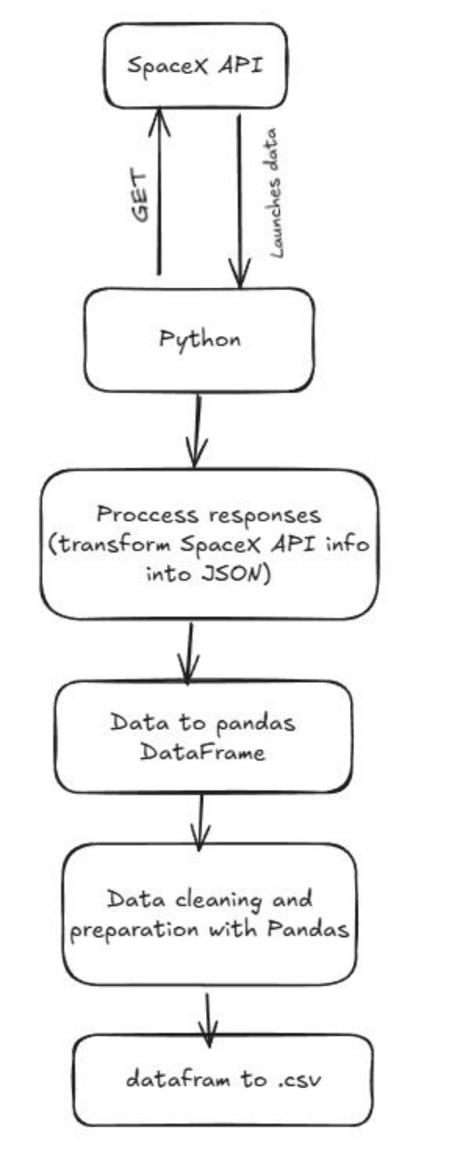
Data Collection

- Data was collected via SpaceX open API and Wikipedia scrapping;
- Data was decoded to a JSON object to facilitate handling thru pandas;
- Converted data to a DataFrame of pandas;
- All the necessary cleaning and wrangling;

Data Collection – SpaceX API

- Used python requests to make GET request and get data from SpaceX API, prepared and cleaned, as well as saving it to a .csv with pandas.

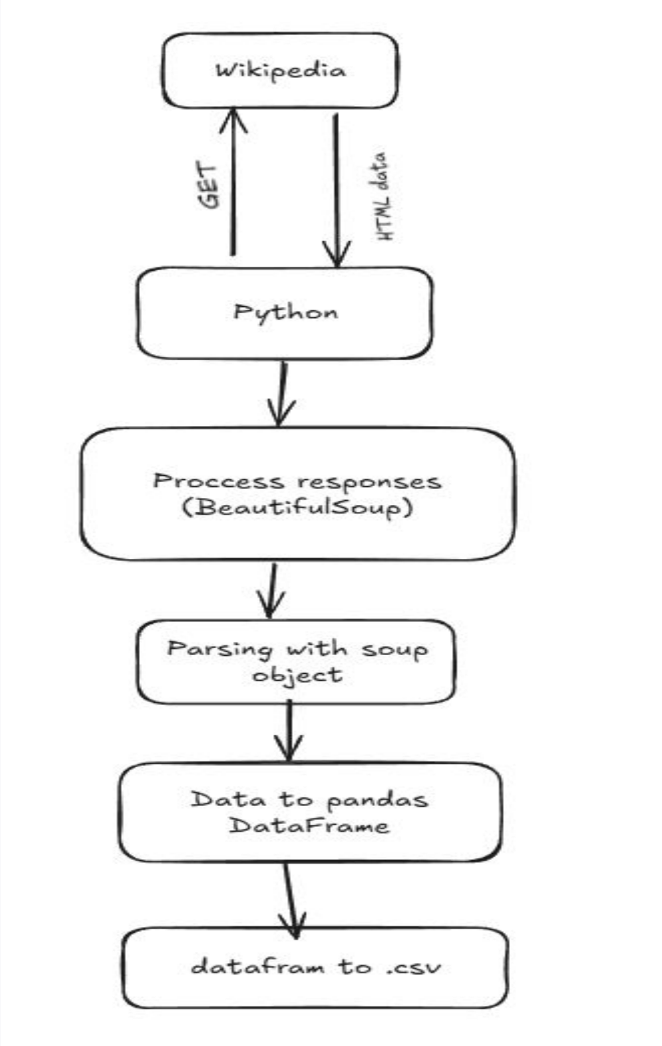
- https://github.com/bernardomelo/DataScienceCapstone-IBM/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- Used requests to make a get request and extract HTML from the response;
- BeautifulSoup to parse HTML;
- Pandas to convert into df and then .csv.

- https://github.com/bernardomelo/DataScienceCapstone-IBM/blob/main/jupyter-labs-webscraping.ipynb



Data Wrangling

- Performed exploratory data analysis and determined the training labels;
- Calculated the number of launches at each site, and the number and occurrence of each orbits;
- Created landing outcome label from outcome column and exported the results to csv.

https://github.com/bernardomelo/DataScienceCapstone-IBM/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

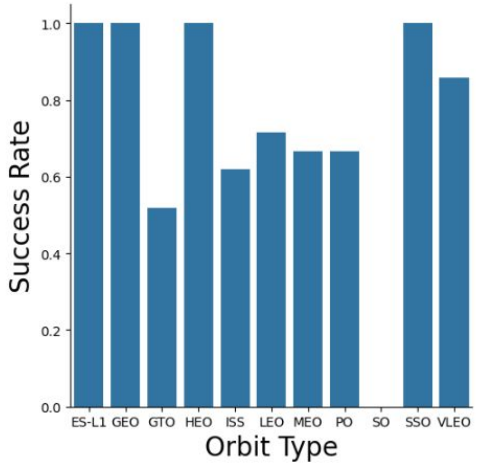
- Mainly Scatter Plot’s used to determine relationship between features. Bar chart was also used.
- https://github.com/bernardomelo/DataScienceCapstone-IBM/blob/main/eda dataviz.ipynb

TASK 3: Visualize the relationship between success rate of each orbit type

Next, we want to visually check if there are any relationship between success rate and orbit type.

Let's create a `bar chart` for the sucess rate of each orbit

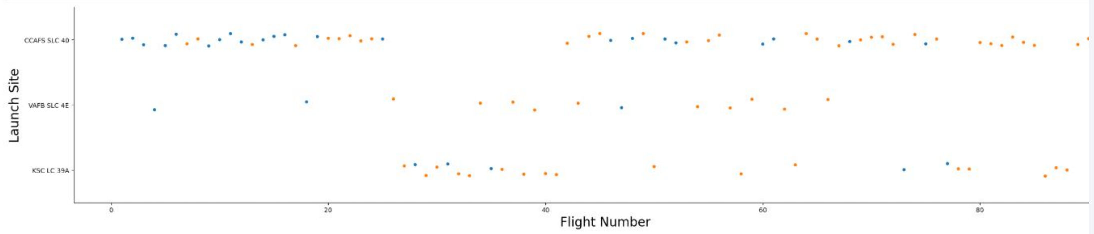
```
# HINT use groupby method on Orbit column and get the mean of Class column
sns.catplot(x= 'Orbit', y = 'Class', data = df.groupby('Orbit')['Class'].mean().reset_index(), kind = 'bar')
plt.xlabel('Orbit Type',fontsize=20)
plt.ylabel('Success Rate',fontsize=20)
plt.show()
```



TASK 1: Visualize the relationship between Flight Number and Launch Site

Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(x='FlightNumber', y='LaunchSite', hue='Class', data=df, aspect=5)
plt.xlabel('Flight Number', fontsize=20)
plt.ylabel('Launch Site', fontsize=20)
plt.show()
```



EDA with SQL

- **We applied EDA with SQL to get insight from the data. These were the queries wrote:**
  - Names of unique launch sites in the space mission;
  - Total payload mass carried by boosters launched by NASA (CRS);
  - Average payload mass carried by booster version F9 v1.1;
  - Total number of successful and failure mission outcomes;
  - Failed landing outcomes in drone ship, their booster version and launch site names;

- https://github.com/bernardomelo/DataScienceCapstone-IBM/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Marked all launch sites, added map objects such as markers, circles, lines to mark the success or failure of launches for each site;
- Assigned the feature launch outcomes o class 0 and 1 i.e., 0 for failure, and 1 for success;
- Using the color-labeled marker clusters, identified which launch sites have relatively high success rate;
- Calculated the distances between a launch site to its proximities. Some answered questions:
    - Are launch sites near railways, highways and coastlines?
    - Do launch sites keep certain distance away from cities:

- https://github.com/bernardomelo/DataScienceCapstone-IBM/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Plotted pie charts showing total launches on certain launch sites;
- Scatter graph with relationship between Outcome and Payload for different booster versions;

- https://github.com/bernardomelo/DataScienceCapstone-IBM/blob/main/spacex_app.py

Predictive Analysis (Classification)

- Loaded data using pandas and numpy, transformed it and split train/test;
- Built some models (logreg, SVM, decision tree, KNN);
- Used GridSearch to tune different hyperparameters;
- Used Jaccard Score, F1 and accuracy as metrics to define the better model
- Found best model to use.
- https://github.com/bernardomelo/DataScienceCapstone-IBM/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

Results

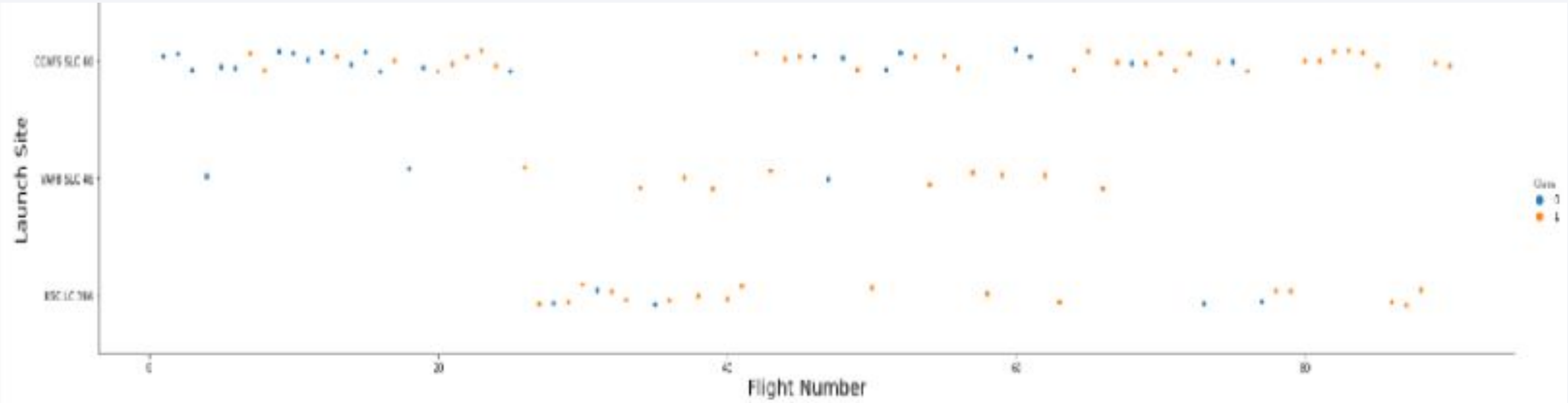
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

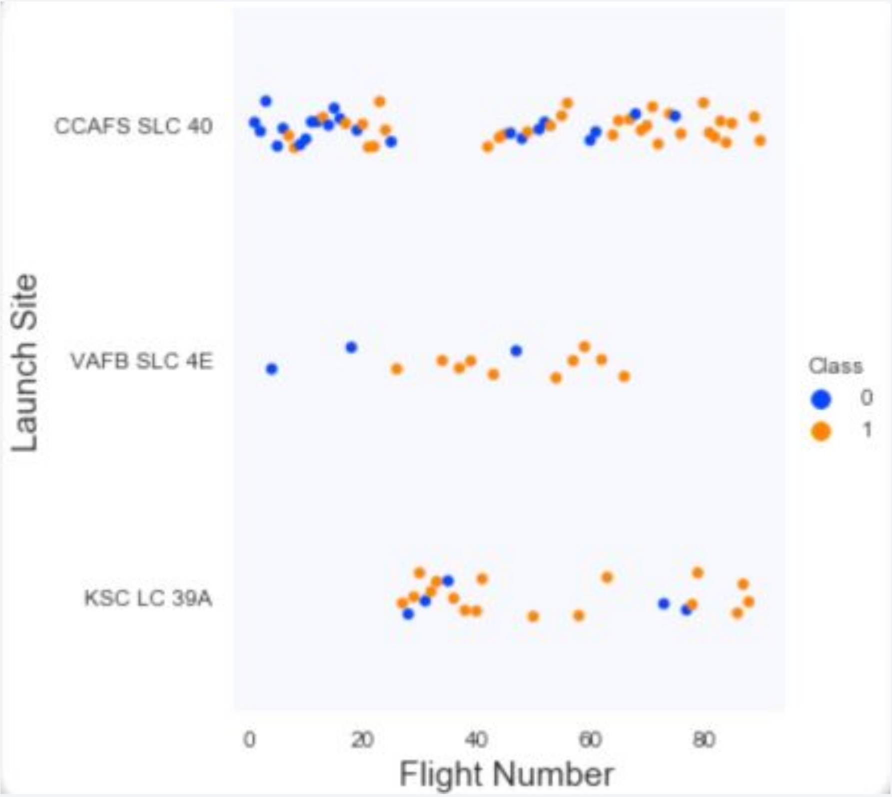
- Plot shows that the larger the flight amount at certain site, the greater the success rate there.



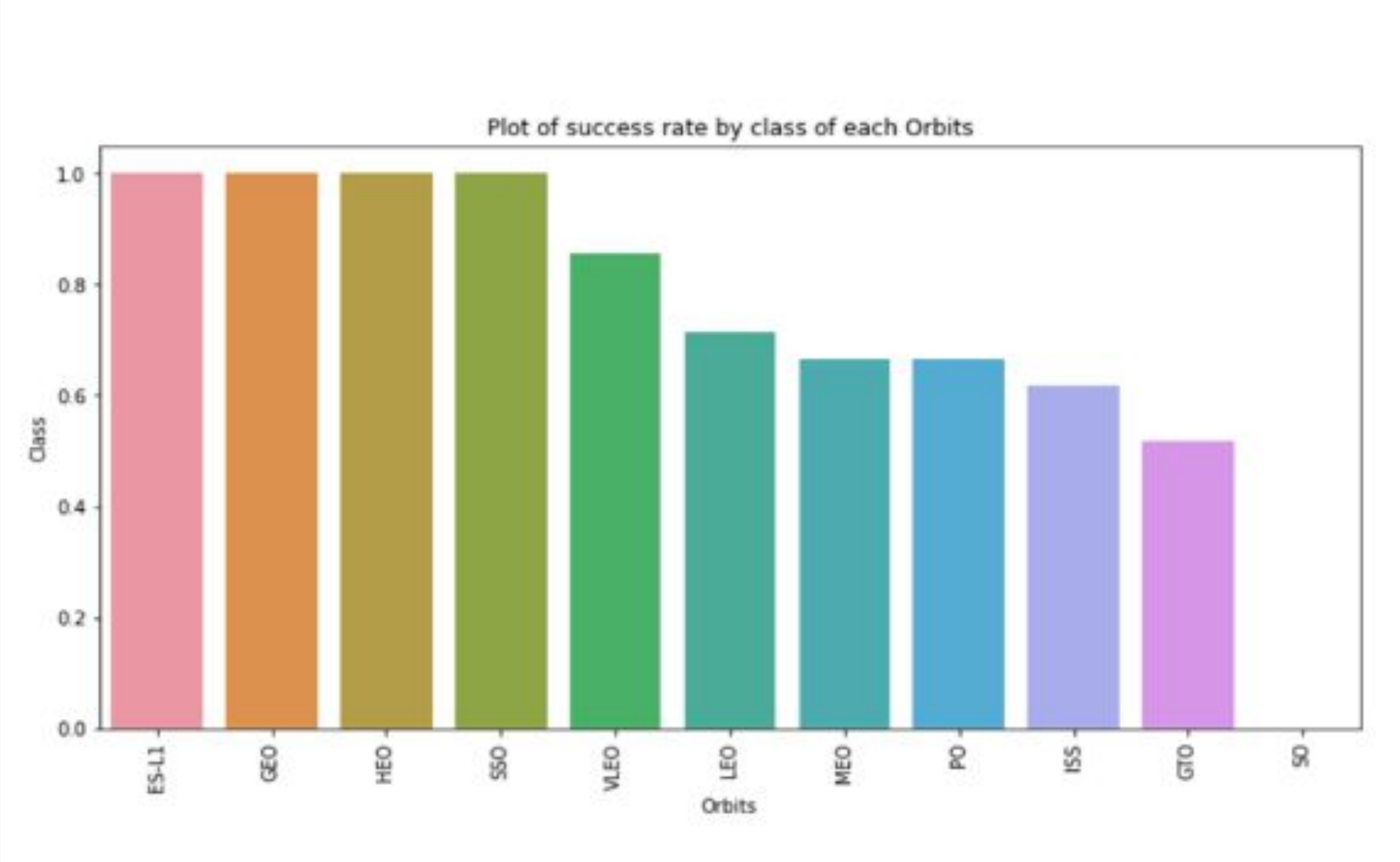
Payload vs. Launch Site

The scatter plot of Launch Site vs. Flight Number shows that:

- Number of flights increases, the rate of success at a launch site increases;
- No early flights launched from KSC LC 39A, explains why launches from this site are more successful;
- Above a flight number of around 30, there are significantly more successful landings (Class = 1).

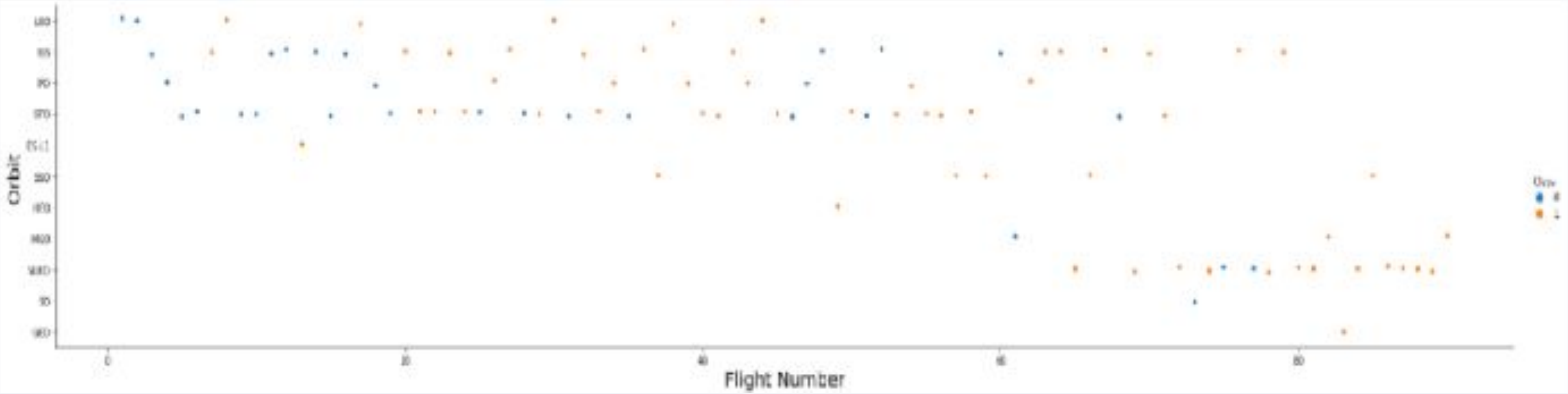


Success Rate vs. Orbit Type



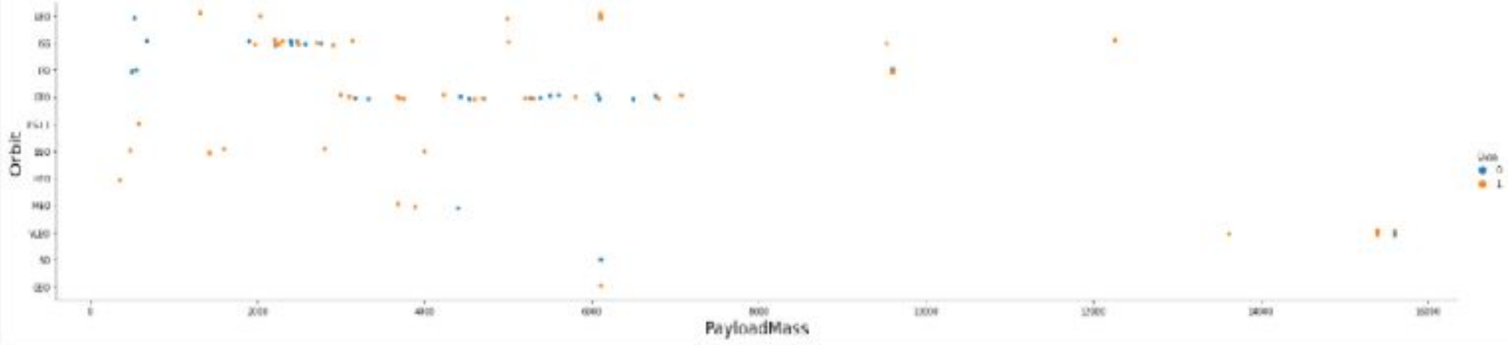
Flight Number vs. Orbit Type

- Plot shows that in the LEO orbit, success is related to the number of flights, but in GTO orbit, there aren't any relationships between flight number and the orbit.



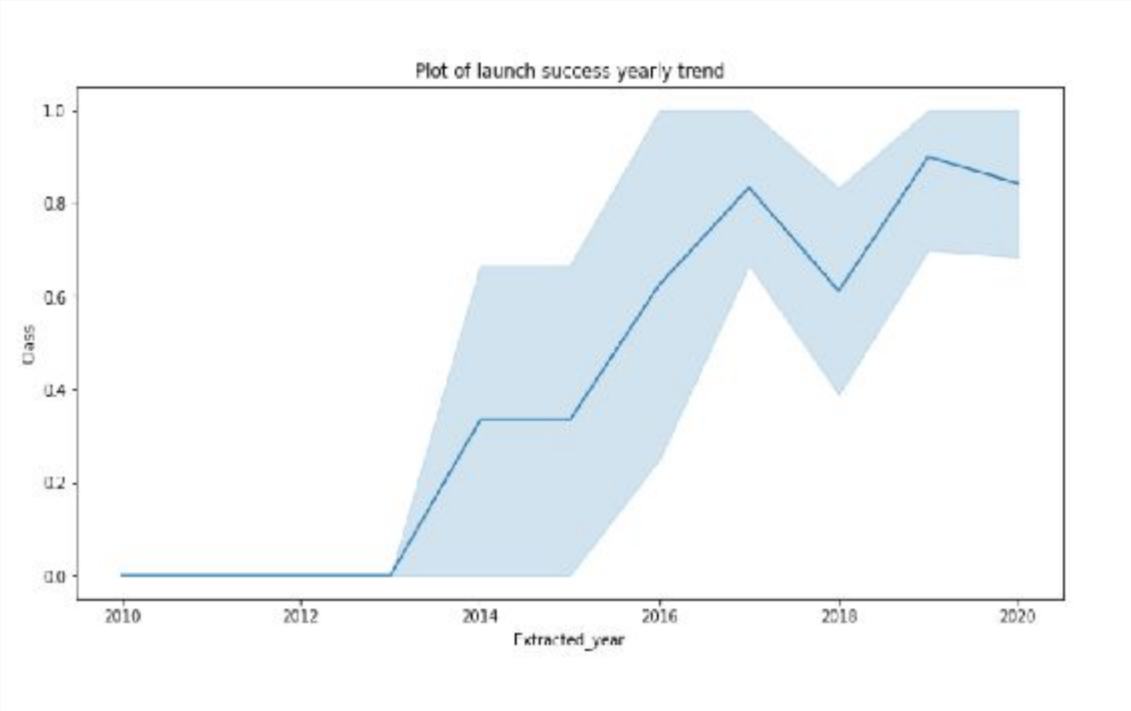
Payload vs. Orbit Type

- Plot show us that PO, LEO and ISS are more successful with heavy payloads.



Launch Success Yearly Trend

- Plot shows us a steady increase rate from 2013 to 2020.



All Launch Site Names

- Simple select query with DISTINCT to show unique launch site names.

```
In [11]: %sql select distinct launch_site from SPACEXTBL;

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

Out[11]:

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

Launch Site Names Begin with 'CCA'

- Simple select query with WHERE clause in launch_site column with LIKE clause searching for ‘CAA%’ string beginnings, with LIMIT to 5, to only print the first 5 rows.

```
In [12]: %sql select * from SPACEXTBL where launch_site like 'CCA%' limit 5;

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

Out[12]:

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

Total Payload Mass

- Calculates the total payload mass for NASA (CRS) missions, adds up all payload masses and filters records to include only launches for NASA (CRS).

```
In [13]: %sql select sum(payload_mass__kg_) as total_payload_mass from SPACEXTBL where customer = 'NASA (CRS)';

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

Out[13]:

total_payload_mass
45596

Average Payload Mass by F9 v1.1

- Calculates the total payload mass for NASA (CRS) missions, adds up all payload masses and filters records with WHERE clause to look for F9 v1.1.

```
In [14]: %sql select avg(payload_mass__kg_) as average_payload_mass from SPACEXTBL where booster_version like '%F9 v1.1%';

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

Out[14]:

average_payload_mass
2534.6666666666665

First Successful Ground Landing Date

- Selects the minimal date on first_succesful_landing column with clause WHERE in landing_outcome being “Success (ground pad)”.

In [16]:
```
%sql select min(date) as first_successful_landing from SPACEXTBL where landing_outcome = 'Success (ground pad)';
```

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

Out[16]:

first_successful_landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Selects the booster_version column with clause WHERE in landing_outcome being “Success (ground pad)” and payload_mass_kg in the asked range.

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
In [17]: %sql select booster_version from SPACEXTBL where landing_outcome = 'Success (drone ship)' and payload_mass__kg_ between 400(
```

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

Out[17]:

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

Total Number of Successful and Failure Mission Outcomes

- Counts the number of missions for each outcome category.

In [18]:
```
%sql select mission_outcome, count(*) as total_number from SPACEXTBL group by mission_outcome;
```

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

Out[18]:

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

Boosters Carried Maximum Payload

- Determined the booster that have carried the maximum payload using a subquery in the WHERE clause and the MAX() function.

```
In [19]: %sql select booster_version from SPACEXTBL where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXTBL);

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

Out[19]:

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

2015 Launch Records

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

- Present your query result with a short explanation here

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

- We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2010-03-20. Applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order.

In [24]:

```
%%sql select landing_outcome, count(*) as count_outcomes from SPACEXTBL
      where date between '2010-06-04' and '2017-03-20'
      group by landing_outcome
      order by count_outcomes desc;
```

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

Out[24]:

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

Section 3

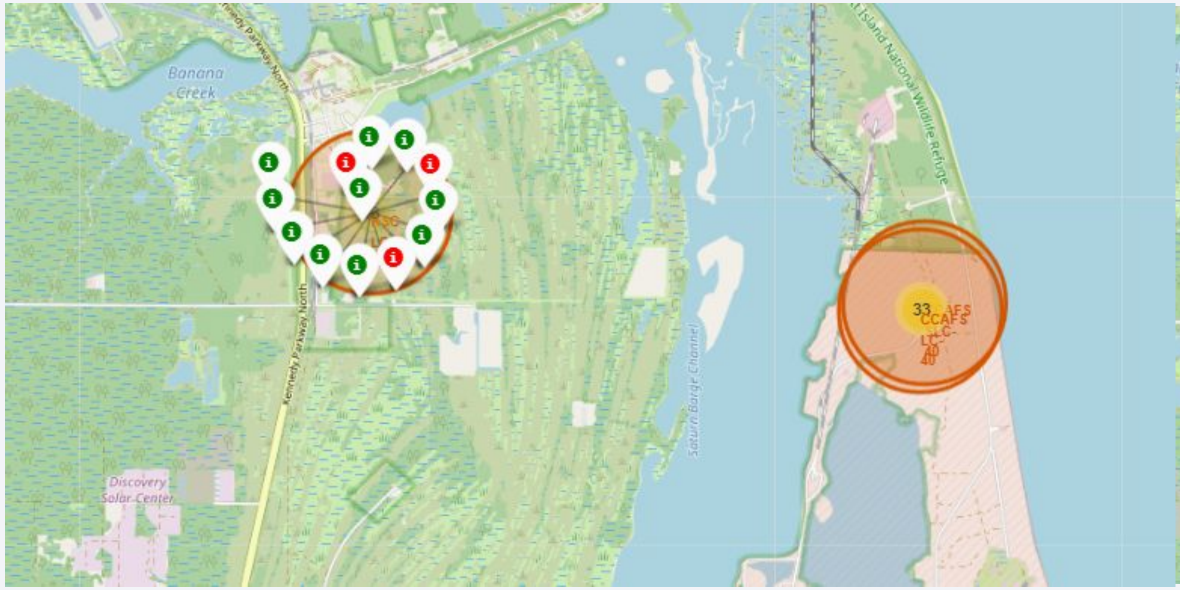
Launch Sites Proximities Analysis

All launch sites global map marker

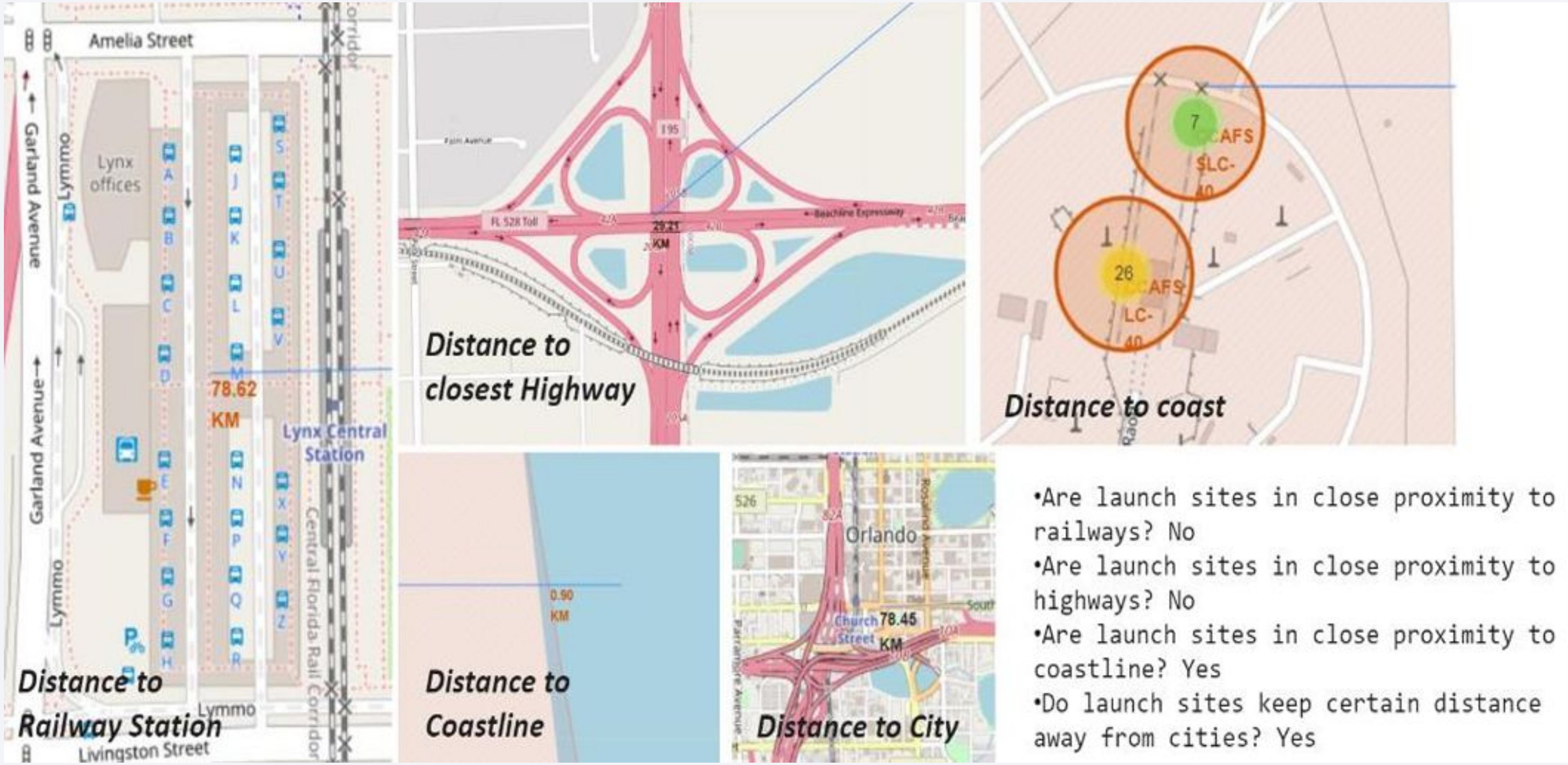
- From the picture, we can see that basically all launch sites are in US coastal places.

Markers showing launch sites with color labels

- Green markers are successes, red failures;
- More successes in general than fails.



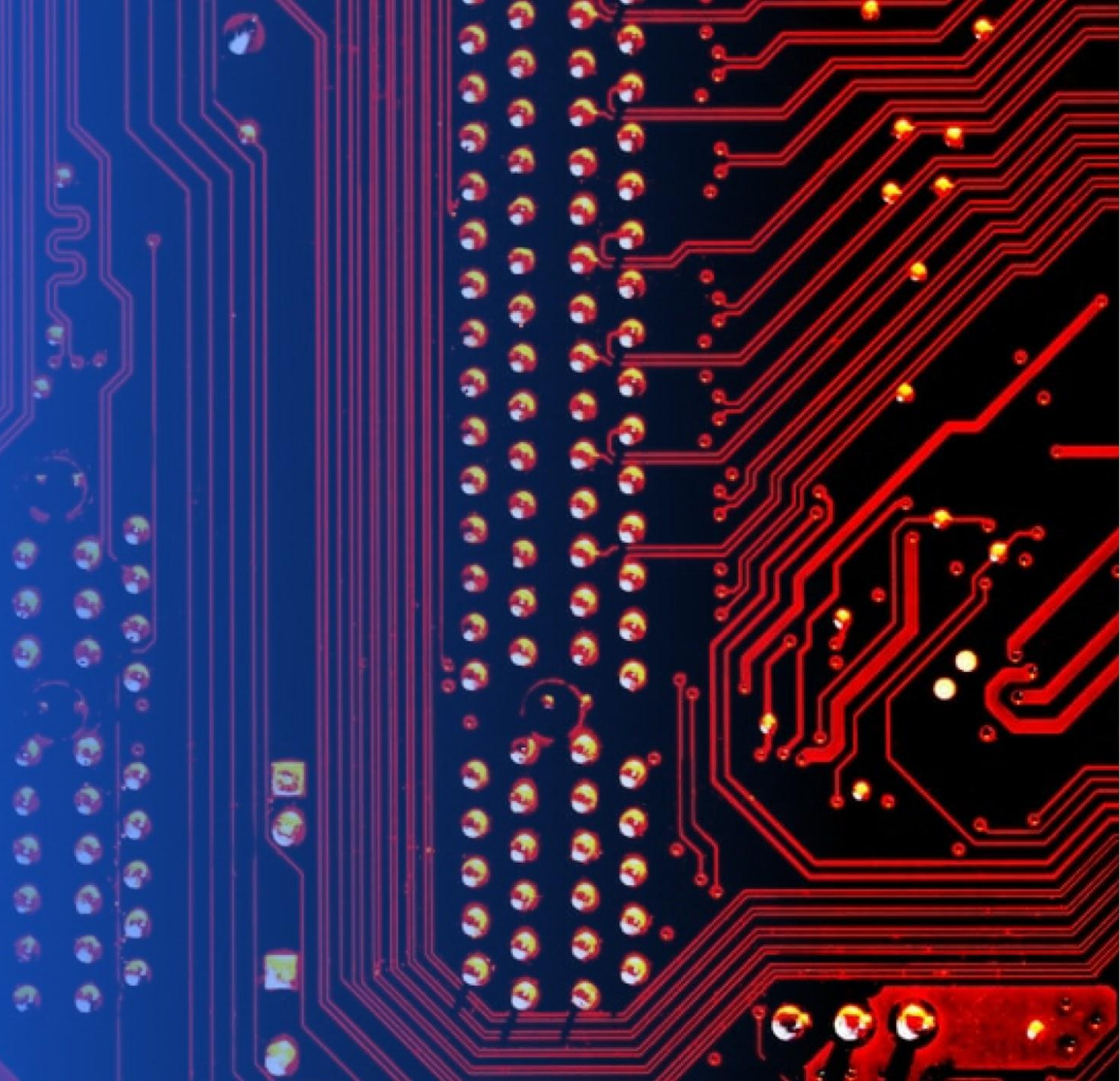
Distance from launch sites to landmarks



- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

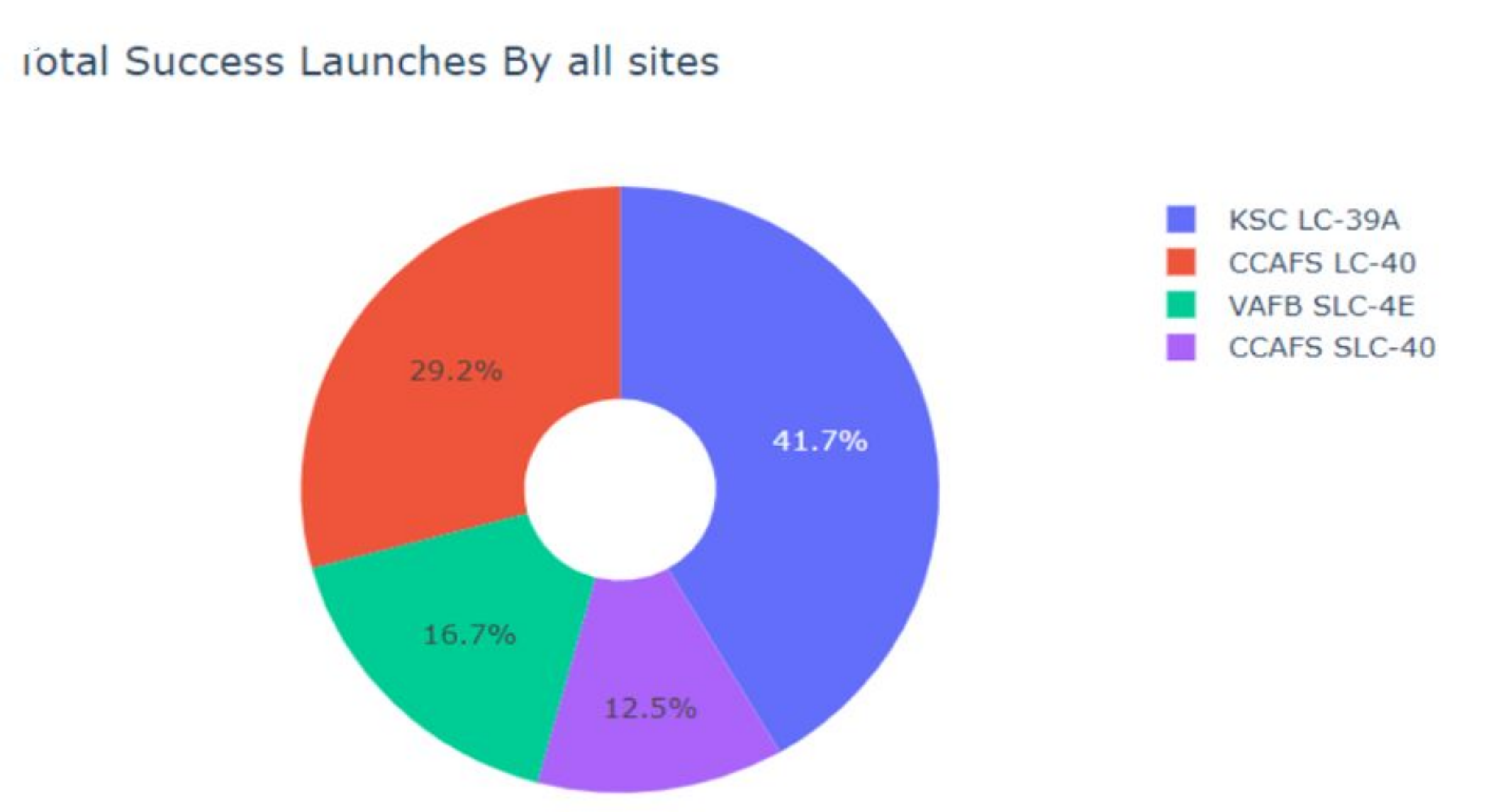
Section 4

Build a Dashboard with Plotly Dash



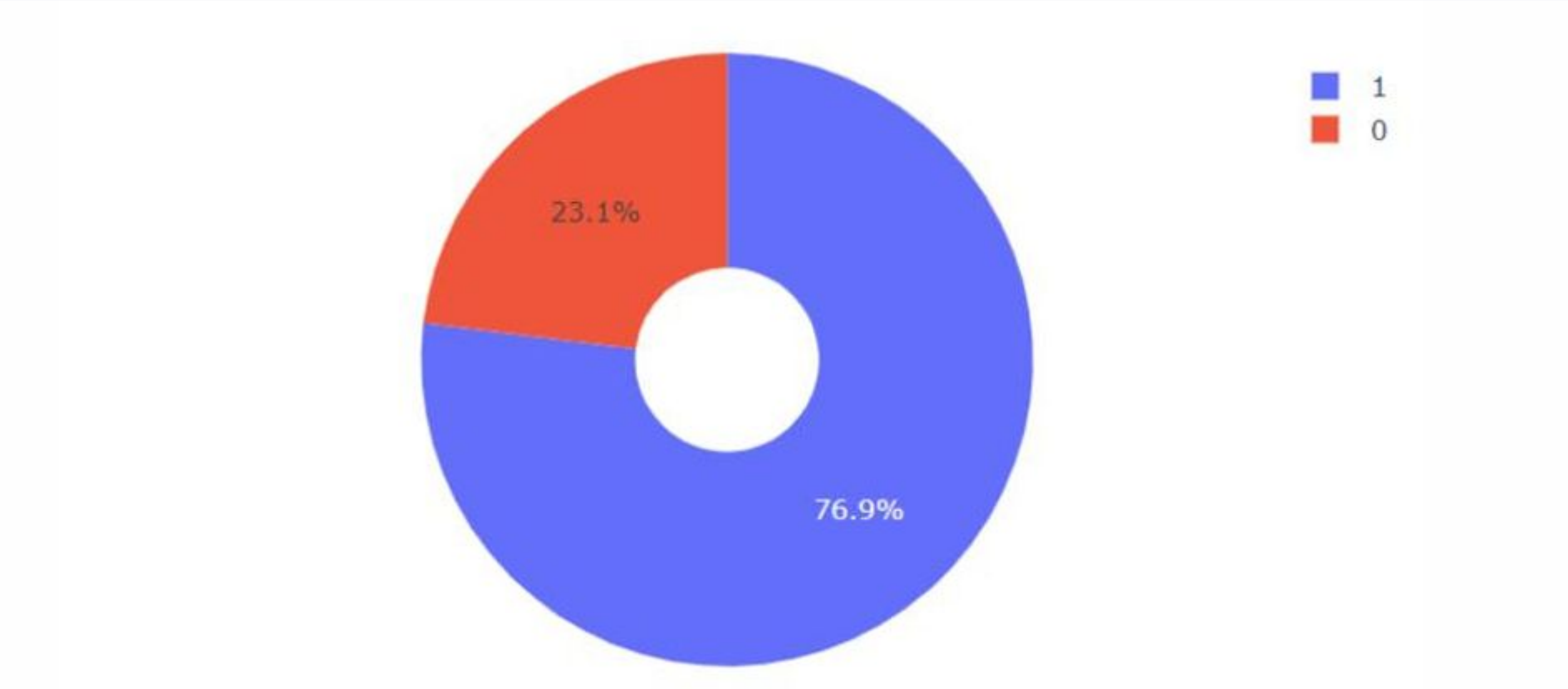
Success Launches by sites

- KSC LC-39A seem to have the most successes.



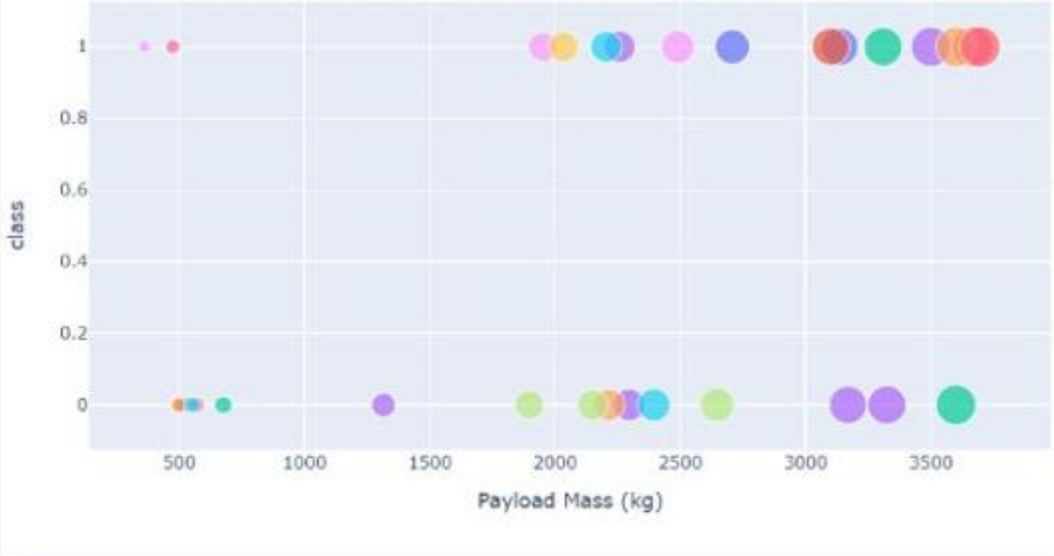
Success/Failure ratio

- 76.9% Success rate
- 23.1% Failure rate

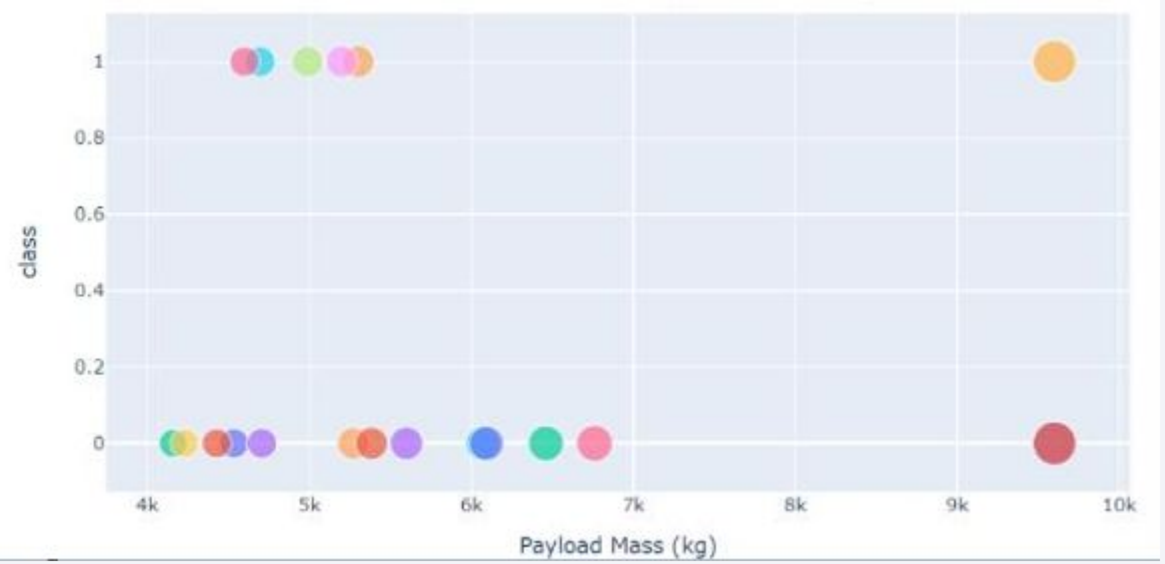


Scatter plot of Payload vs Launch Outcome for all sites

Low weighted payload



Heavy weighted payload



Section 5

Predictive Analysis (Classification)

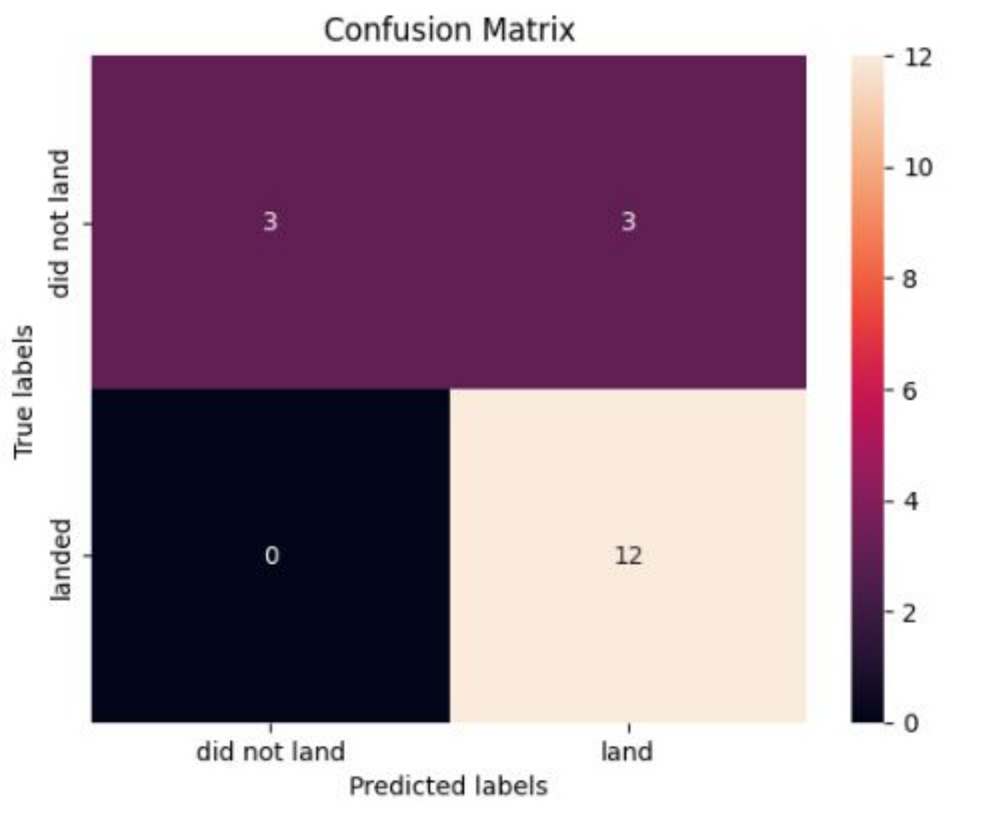
Classification Accuracy

[46]:

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.833333	0.845070	0.882353	0.819444
F1_Score	0.909091	0.916031	0.937500	0.900763
Accuracy	0.866667	0.877778	0.911111	0.855556

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task

Thank you!