IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Kamil Karim
10.10.2022

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

SpaceX is a revolutionary company that revolutionized the space industry by offering a dedicated Falcon 9 rocket launch for just $62 million. The other suppliers cost more than $165 million each. Much of the savings is due to SpaceX's amazing idea of reusing the first stage of a launch by re-landing the rocket for the next mission. If you repeat the process, the price will drop even more. As a data scientist at a startup competing with SpaceX, the goal of this project is to build a machine learning pipeline to predict the outcome of future first-stage landings. The project is critical to determining the right price for SpaceX's rocket launch.

The problems include:

- Identify all factors that affect landing results.
- The relationship between each variable and how it affects the results.
- Conditions required to increase the probability of a successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX REST API and web scrapping from Wikipedia
- Perform data wrangling
  - Data was processed using one-hot encoding for 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 collection is the process of collecting and measuring information about a target variable in an established system, which can then answer relevant questions and evaluate the results.

- Data collection was done using get request to the SpaceX API.
- Then we decoded the response content as a Json using .json() function and transformed it into a pandas dataframe using json_normalize() .
- We then cleaned the data, checked for missing values and populated it with whatever was needed.
- Web scrapping was then done from Wikipedia for Falcon 9 launch records with BeautifulSoup.
- Then parse the table and transform it into a pandas dataframe.

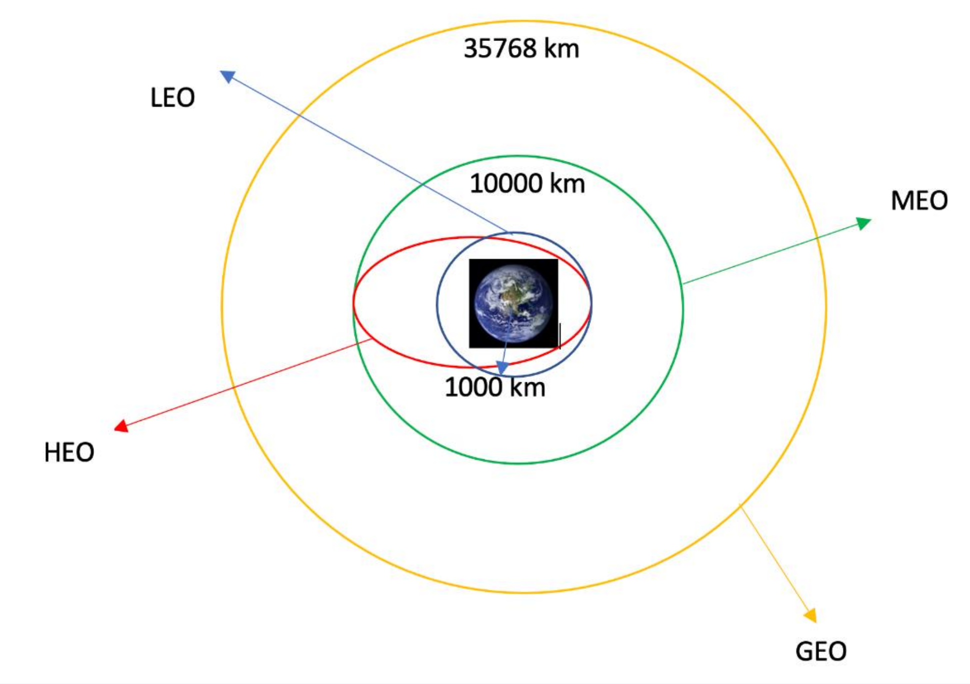
Data Collection – SpaceX API

- Using the get request to the SpaceX API to collect the data, clean it.
- https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Spacex%20Data%20Collection%20Api.ipynb

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]: response = requests.get(spacex_url)

In [13]: # Use json_normalize meethod to convert the json result into a
         data = pd.json_normalize(response.json())

In [ ]: # Lets take a subset of our dataframe keeping only the features we want and the flight numb
        data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

        # We will remove rows with multiple cores because those are falcon rockets with 2 extra roc
        data = data[data['cores'].map(len)==1]
        data = data[data['payloads'].map(len)==1]

        # Since payloads and cores are lists of size 1 we will also extract the single value in the
        data['cores'] = data['cores'].map(lambda x : x[0])
        data['payloads'] = data['payloads'].map(lambda x : x[0])

        # We also want to convert the date_utc to a datetime datatype and then extracting the date
        data['date'] = pd.to_datetime(data['date_utc']).dt.date

        # Using the date we will restrict the dates of the launches
        data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Collection - Scraping

- We performed an exploratory data analysis and determined the training labels.
- We calculated the number of launches at each location and the number and occurrence of each orbital.
- We created a landing result label from the result column and exported the results to CSV.
- https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Webscrapping.ipynb



Data Wrangling

- Data Wrangling is the process of cleaning and unifying messy and complex data sets for easy access and Exploratory Data Analysis (EDA).

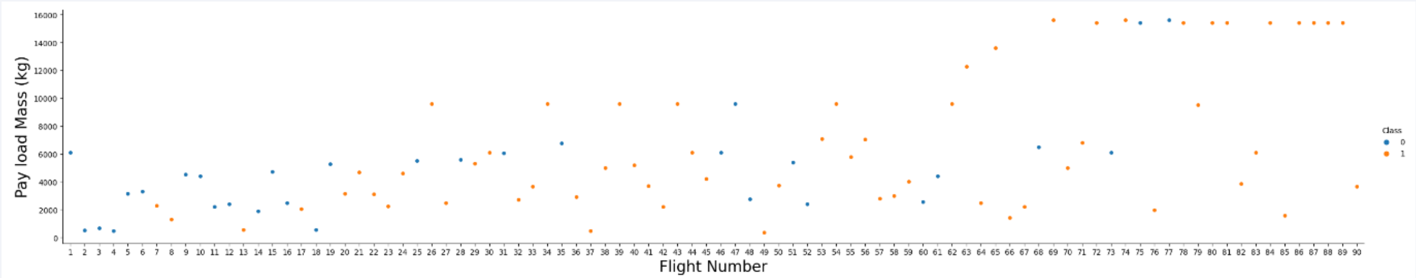
- We will first sum the number of launches on each site, then calculate the number and occurrence of mission outcome per orbit type.

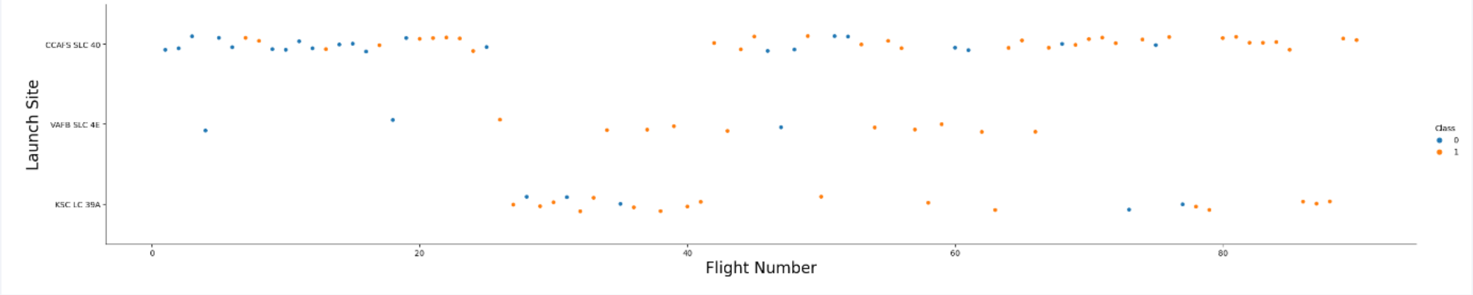
- We then create a landing outcome label from the outcome column. This will make it easier for further analysis, visualization, and ML. Lastly, we will export the result to a CSV.

- https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Spacex%20Data%20Wrangling.ipynb

EDA with Data Visualization

We first started by using scatter graph to find the relationship between the attributes such as between:





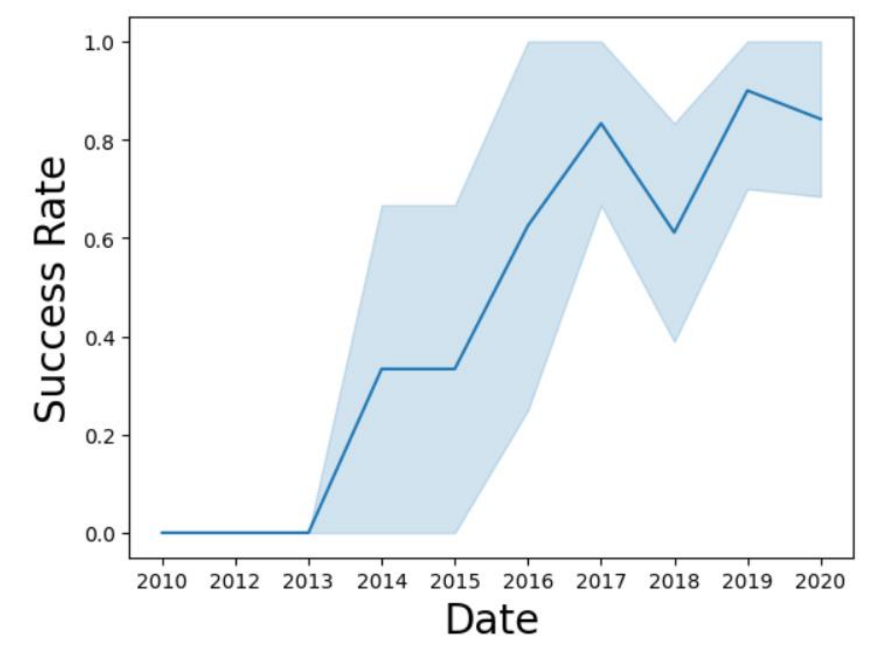
- Payload and Flight Number
- Flight Number and Launch Site
- Payload and Launch Site
- Flight Number and Orbit Type
- Payload and Orbit Type

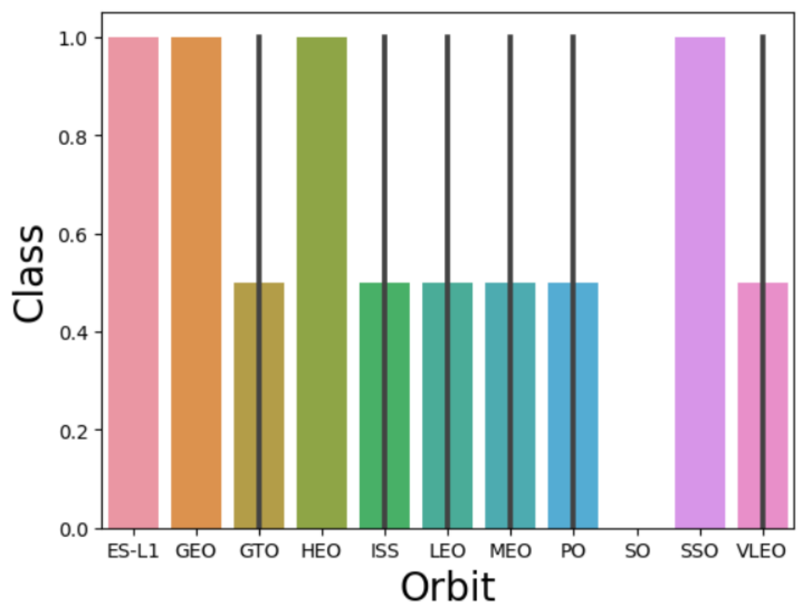
Scatter plots show dependency of attributes on each other, once a pattern is determined from the graphs it becomes easy to see which factors affecting the most to the success of the landing outcomes.

- https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Eda%20Dataviz.ipynb

EDA with Data Visualization

- The Bar graphs here is used to determine which orbits have the highest probability rate of success.
- We use the line graph to show a trends or pattern of the attribute over time which is used to see the launch success yearly trend.
- https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Eda%20Dataviz.ipynb





EDA with SQL

We then loaded the SpaceX dataset into a PostgreSQL database

Then we wrote queries to better understand the dataset. We wrote wrote queries to finf out:

- 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
- https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Eda%20Sql.ipynb

Build an Interactive Map with Folium

- In order to interact with the map, we visualized the launched data and took the 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 feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- We summed the distances between a launch site to its proximities. We answered questions like:
- Are launch sites near railways, highways and coastlines?
- Do launch sites keep certain distance away from cities?

https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- We built the interactive dashboard with plotly dash
- We then plotted pie charts showing the total launches by certain sites
- We then plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version
- https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Spacex%20Dashboard%20App.ipynb

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- https://github.com/kamillearn/IBM-Applied-Data-Science-Capstone/blob/main/Machine%20Learning%20Prediction%20Part%205.ipynb

Results

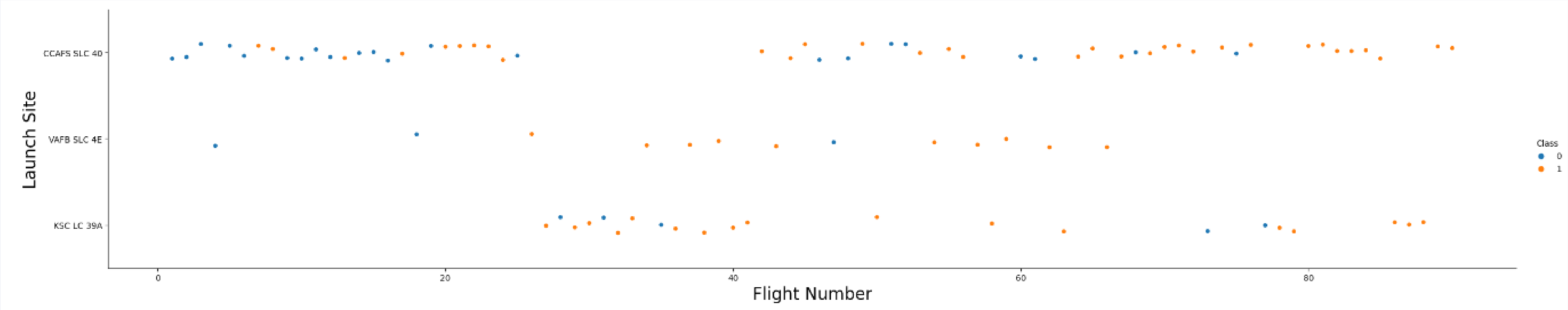
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

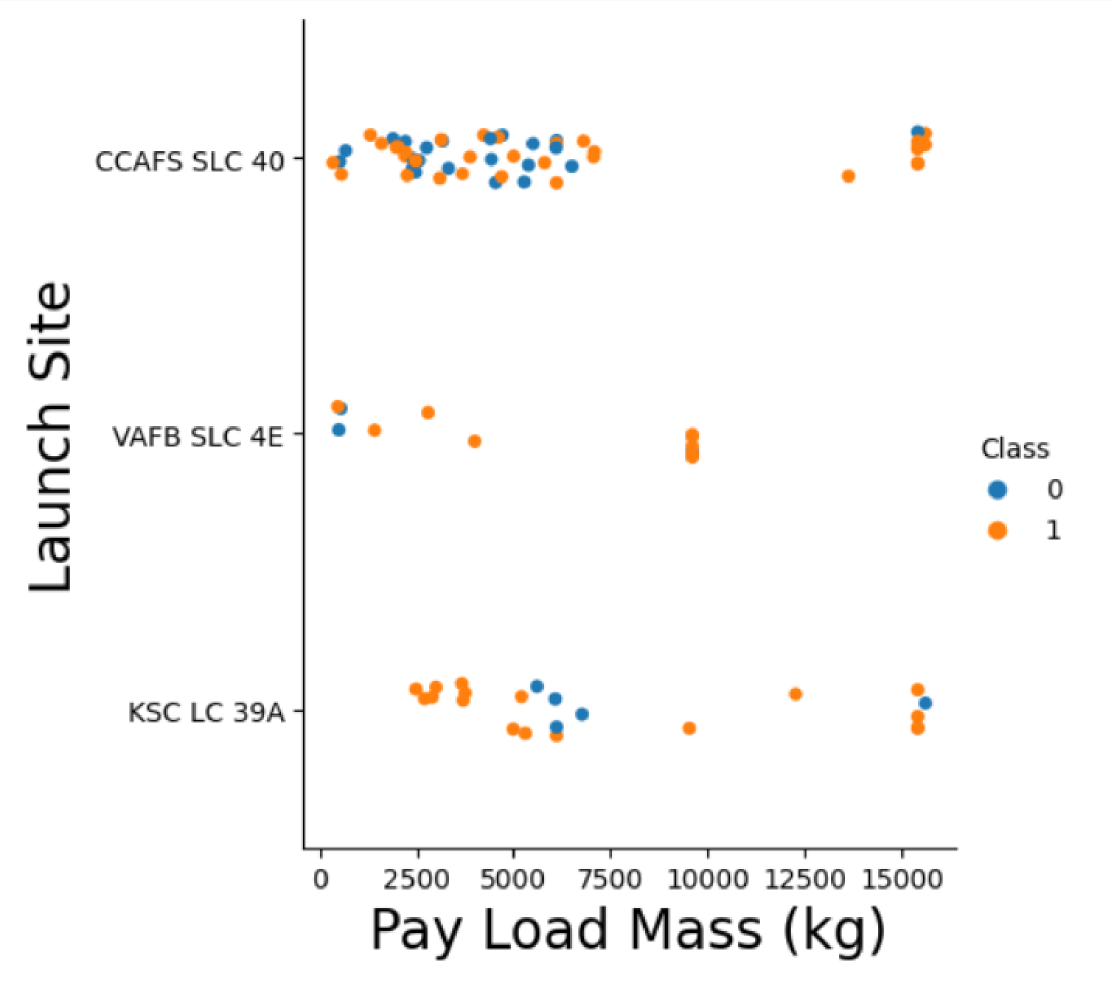
Flight Number vs. Launch Site

- We can observe from the plot that the larger the flight amount at a launch site, the greater the success rate at a launch site



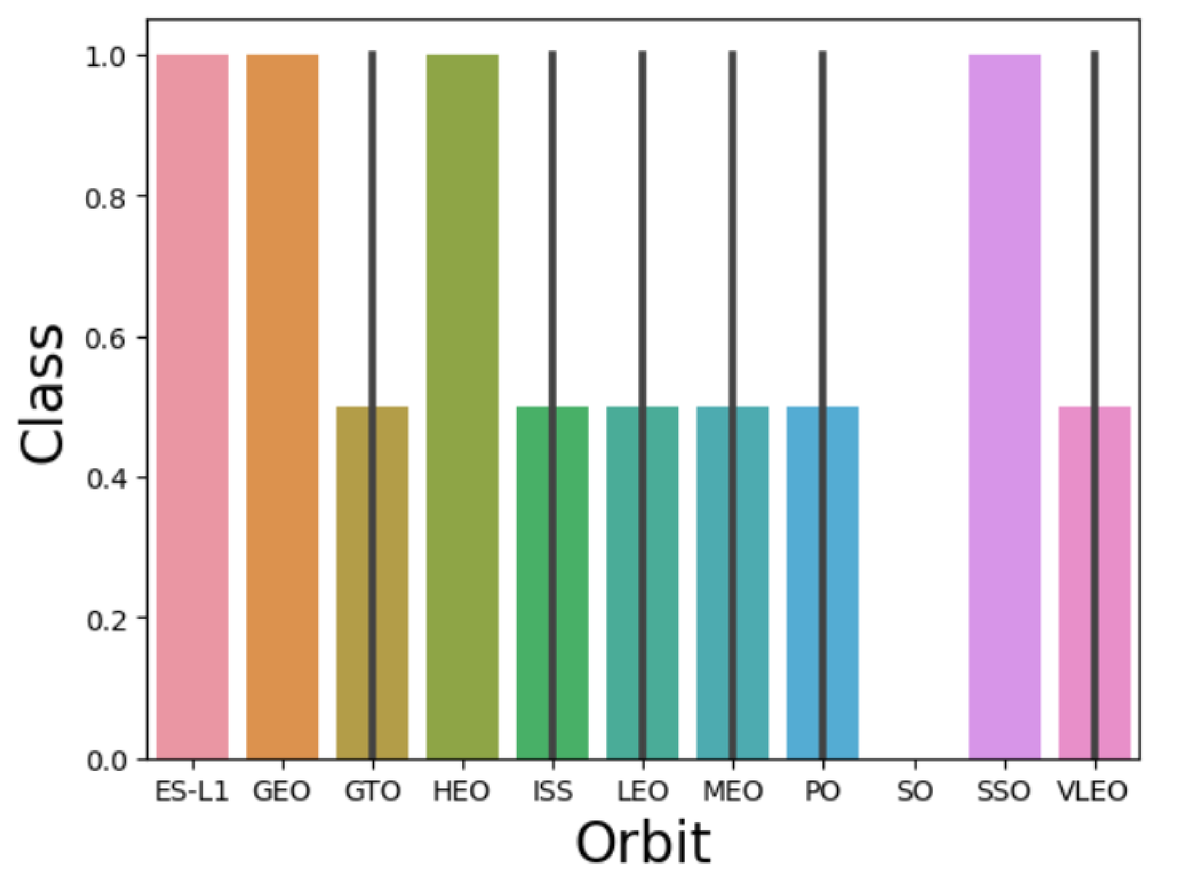
Payload vs. Launch Site

- This scatter plot shows once the pay load mass is greater than 7000kg the probability of the success rate will be highly increased.



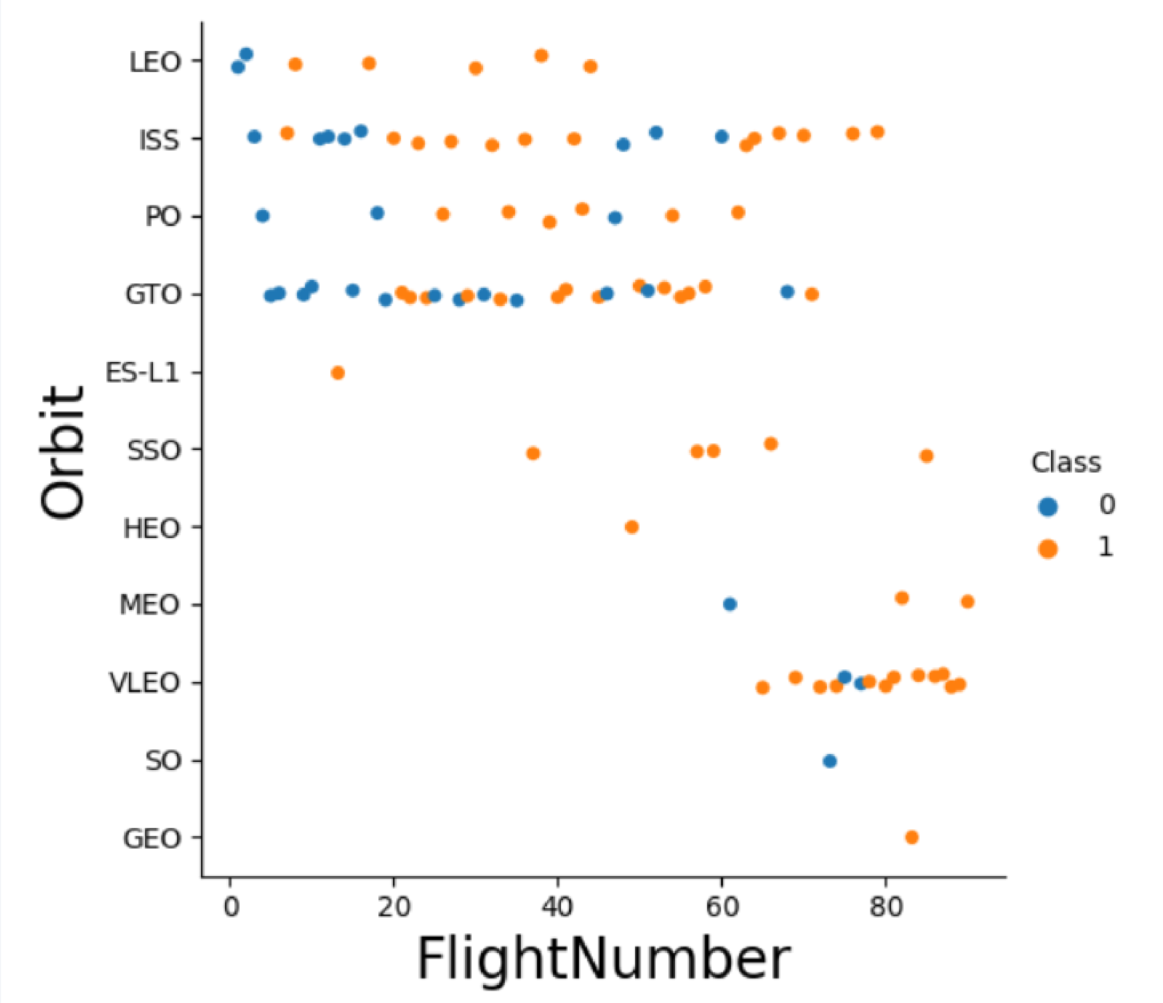
Success Rate vs. Orbit Type

- We can see from this bar that ES-L1, GEO, HEO & SSO have the highest success rate while SO orbit produced 0% rate of success



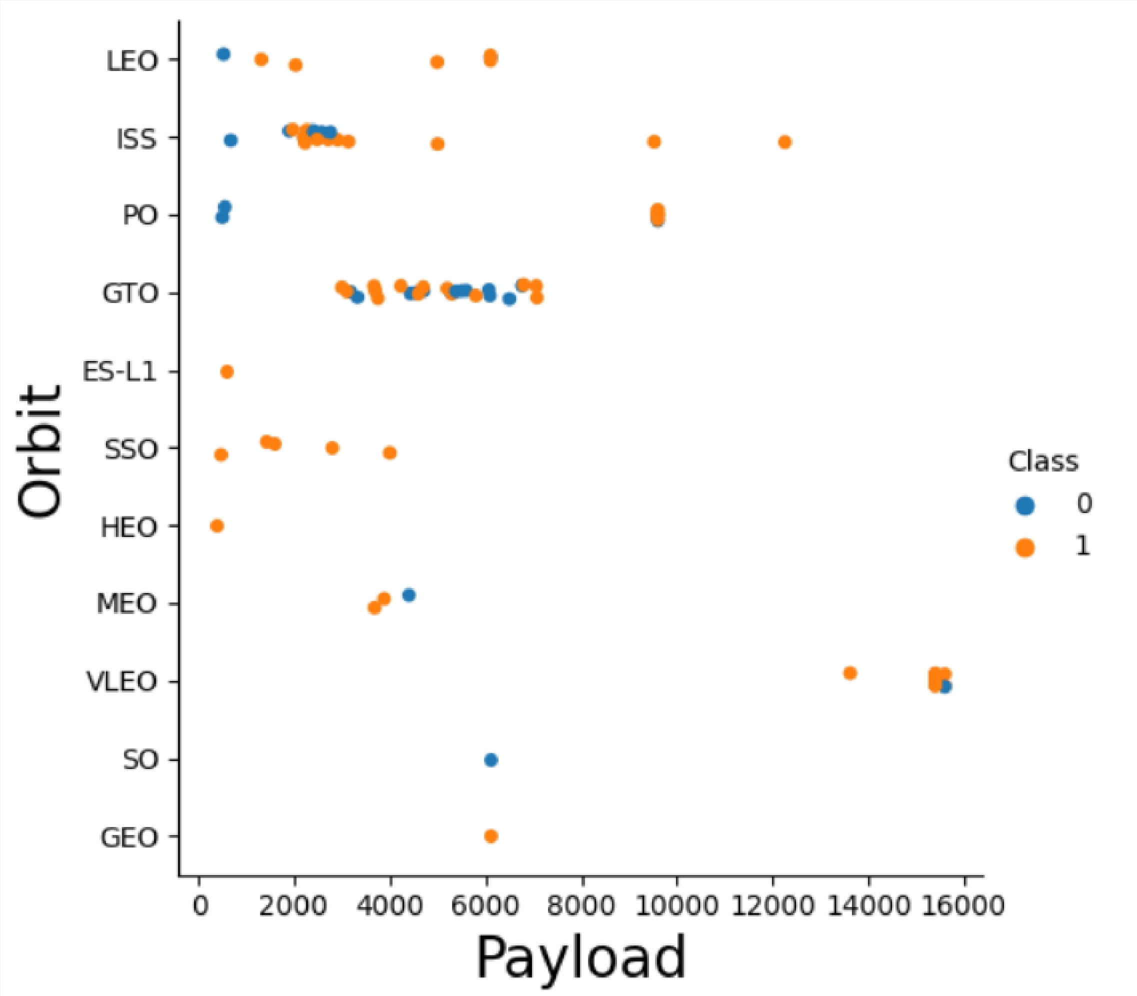
Flight Number vs. Orbit Type

- We can observe that the larger the flight number the greater on each orbits the greater the success rate, except for GTO which shows no relationship between flight number and the orbit



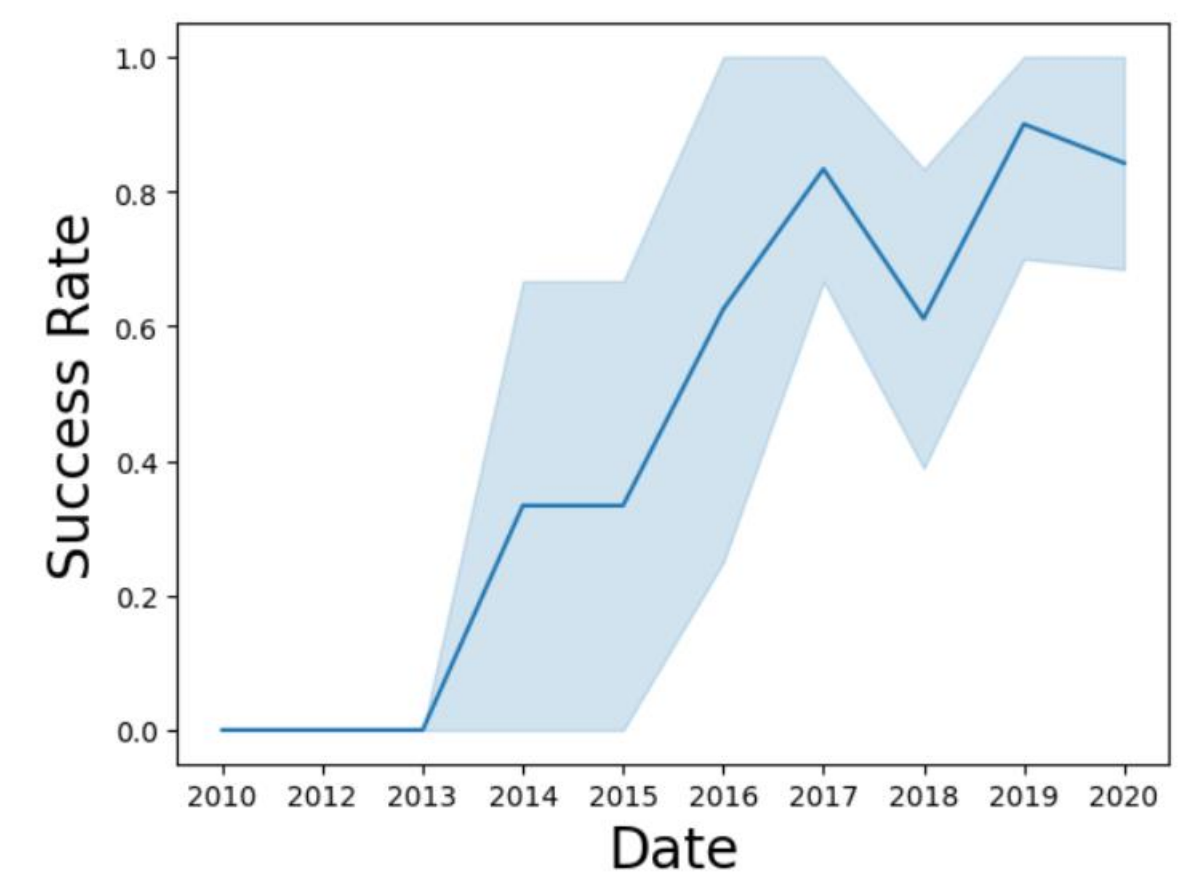
Payload vs. Orbit Type

- Here we can see that PO, LEO and ISS orbits have more successful landing while MEO and VLEO orbits dont, as for GTO, there is not a relationship between orbit and payload



Launch Success Yearly Trend

- From the plot, we can observe that the success rate kept increasing on since 2013 till 2020.



All Launch Site Names

- We used the key word DISTINCT to show only unique launch sites from the SpaceX data.

```
[32]: %sql select distinct(LAUNCH_SITE) from SPACEXTBL

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

[32]:

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

Launch Site Names Begin with 'CCA'

- We used the query below to display 5 records where launch sites begin with `CCA`

```
[33]: %sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

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

[33]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
[34]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'

 * sqlite:///my_data1.db
Done.

[34]: sum(PAYLOAD_MASS__KG_)
                       45596
```

- We calculated the total payload carried by boosters from NASA as 45596 using the query above

Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928.4

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

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

[35]:

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- We used min() function to find the date of the first successful landing outcome on ground pad which shows 22nd December 2015
- The code worked in DB2 cloud however it didnt work in JP NB

```
1  SELECT MIN(date) AS "First Successful Landing" FROM SPACEXTBL where LANDINGOUTCOME = 'Success (ground pad)'
```

History | **Results**

Result set 1 | Details

Filter table   Total:1

First Successful Landing
2015-12-22

```
%sql SELECT MIN(date) AS 'First Successful Landing' FROM SPACEXTBL where LANDINGOUTCOME = 'Success (ground pad)'

 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: LANDINGOUTCOME
[SQL: SELECT MIN(date) AS 'First Successful Landing' FROM SPACEXTBL where LANDINGOUTCOME = 'Success (ground pad)']
(Background on this error at: http://sqlalche.me/e/e3q8)
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- We use the where clause to name of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```sql
select BOOSTER_VERSION from SPACEXTBL
where LANDINGOUTCOME = 'Success (drone ship)'
and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

History | Results

Result set 1 | Details

Filter table

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

```
%sql select BOOSTER_VERSION from SPACEXTBL where LANDINGOUTCOME = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000

 * sqlite:///my_data1.db
(sqlite3.OperationalError) no such column: LANDINGOUTCOME
[SQL: select BOOSTER_VERSION from SPACEXTBL where LANDINGOUTCOME = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000]
(Background on this error at: http://sqlalche.me/e/e3q8)
```

Total Number of Successful and Failure Mission Outcomes

- We used wildcard like '%' to filter for WHERE MissionOutcome was a success or a failure.

```
[173]: %sql SELECT COUNT(MISSION_OUTCOME) FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Success%'

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

[173]:

COUNT(MISSION_OUTCOME)
100

```
[174]: %sql SELECT COUNT(MISSION_OUTCOME) FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Fail%'

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

[174]:

COUNT(MISSION_OUTCOME)
1

Boosters Carried Maximum Payload

- We used where and max clause to find the booster versions which have carried the maximum payload mass

```
[114]: %sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)

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

[114]:

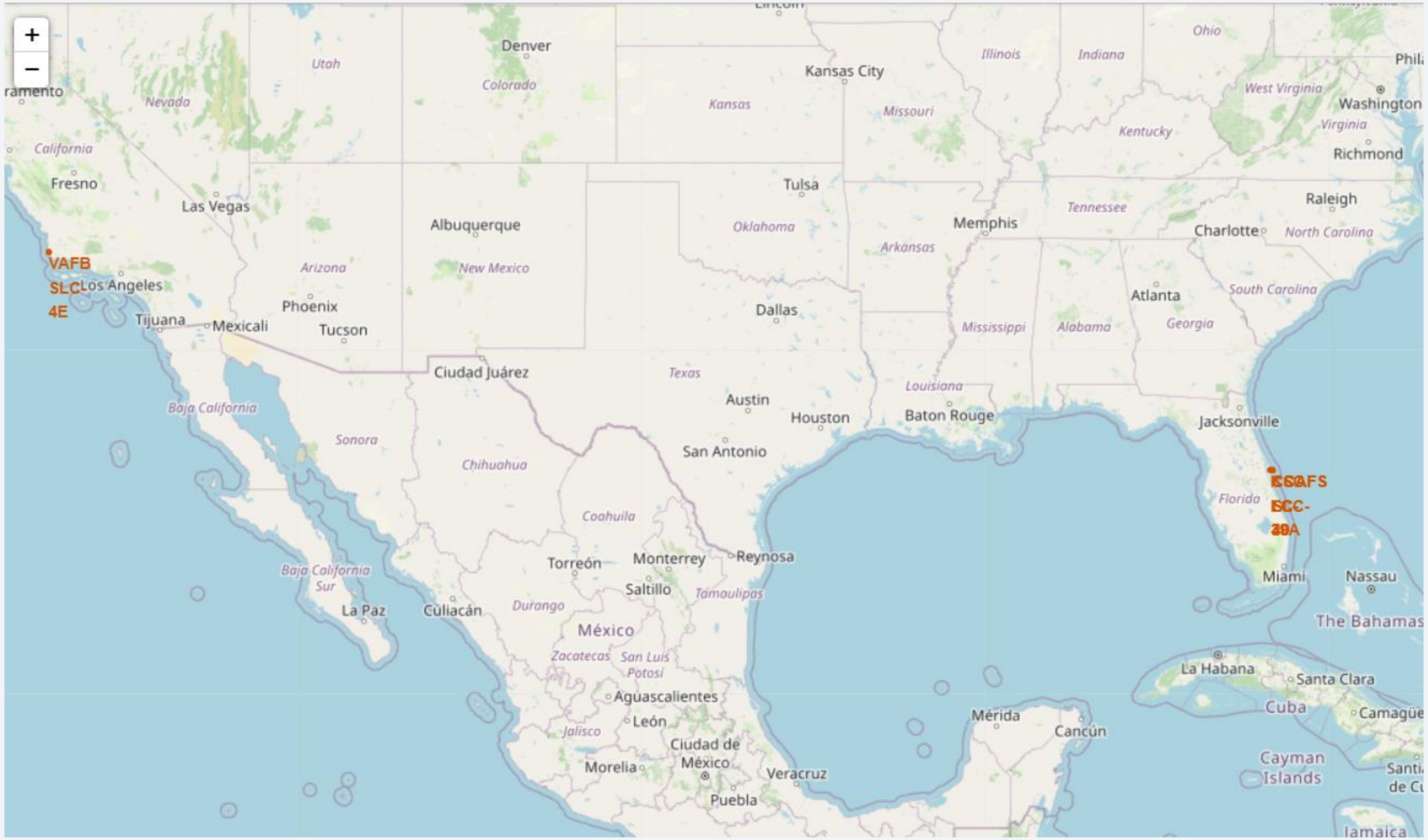
Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

Section 3

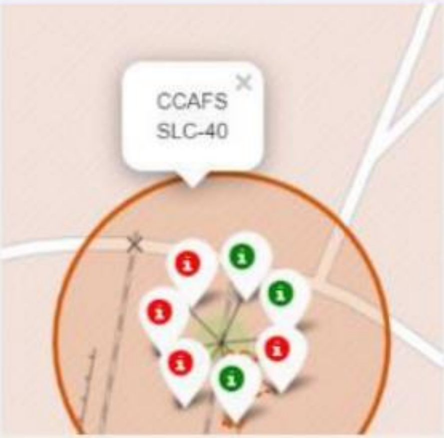
Launch Sites Proximities Analysis

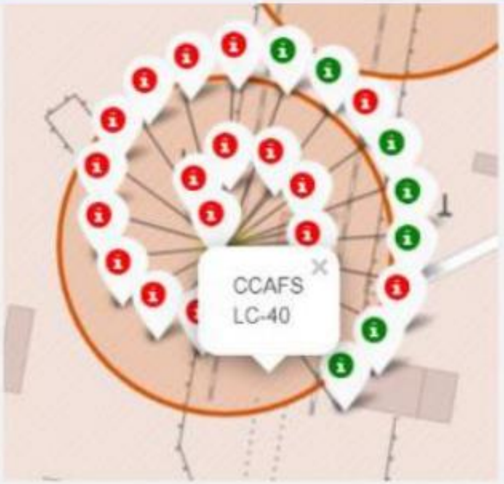
All launch sites global map markers

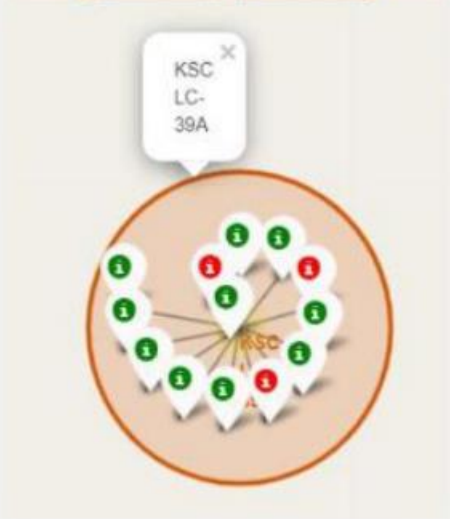
- All the SpaceX launch sites are located inside the United States



Markers showing launch sites with color labels







Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

Launch Site distance to landmarks

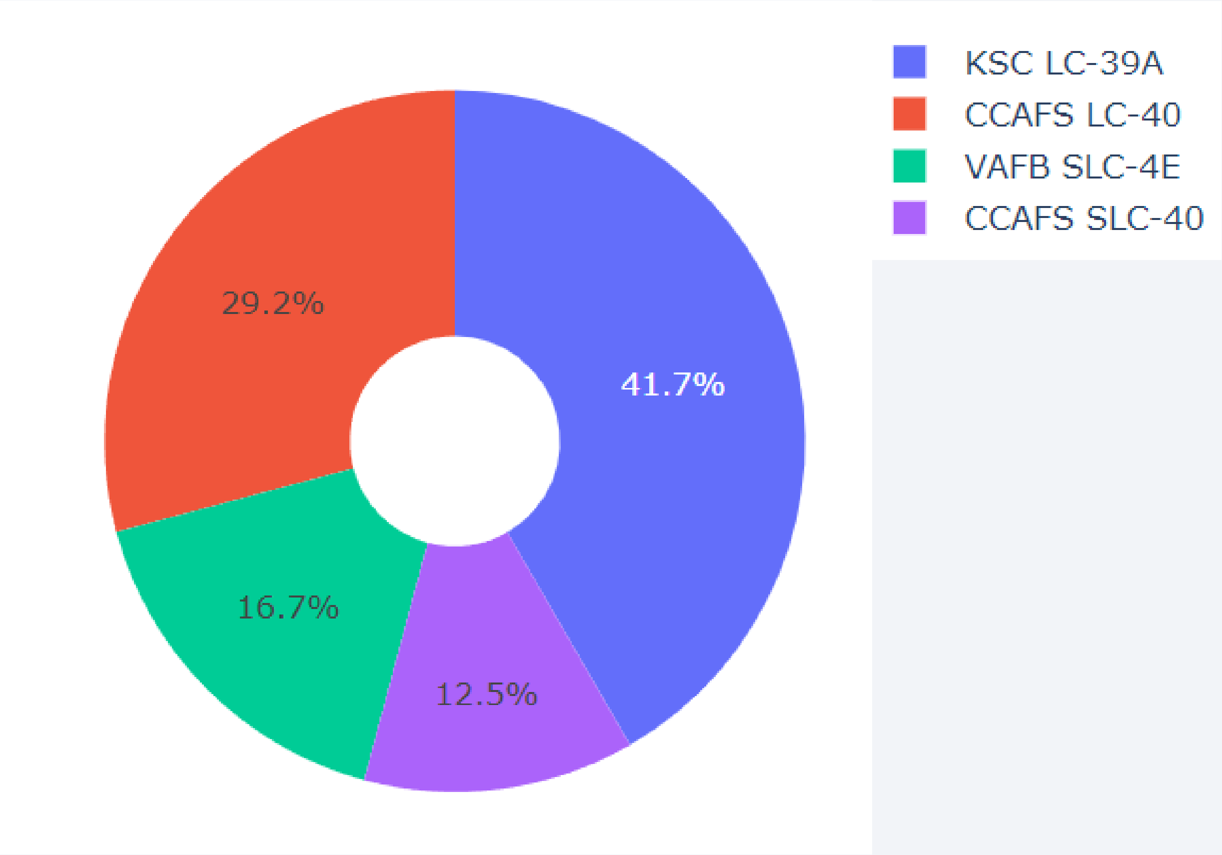
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

Build a Dashboard with Plotly Dash

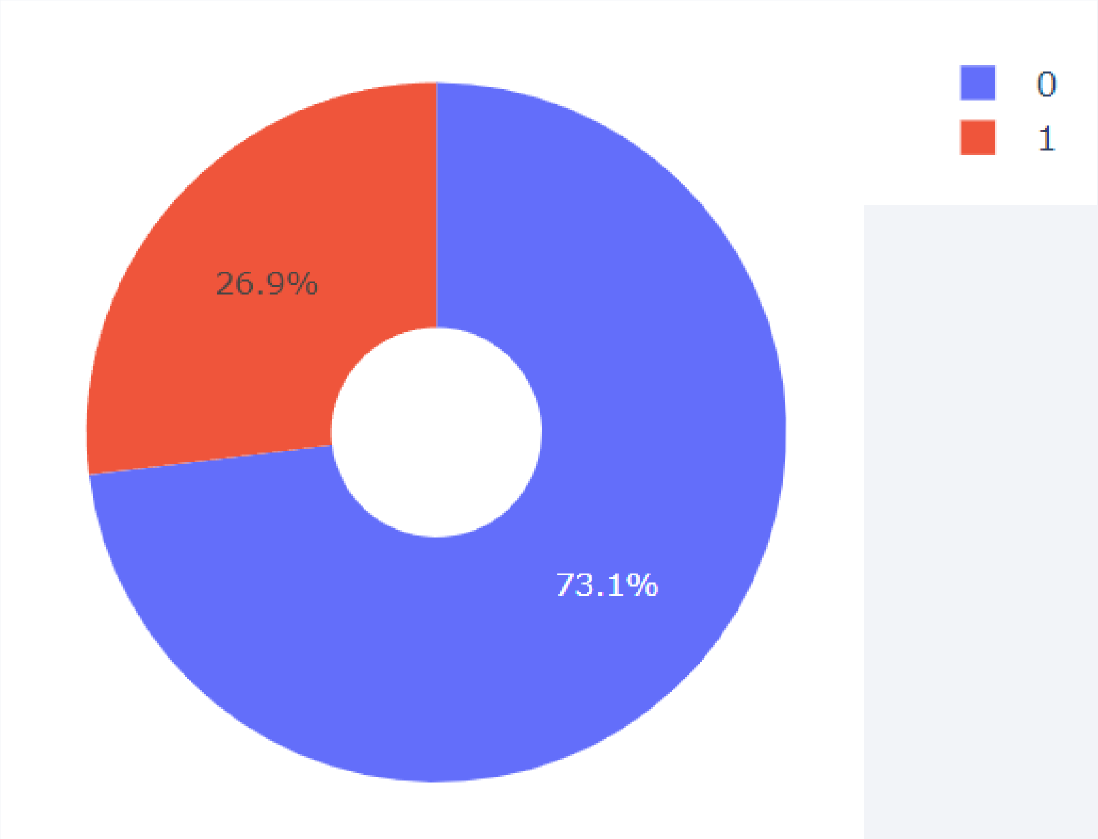
The success percentage by each sites.

- We can that KSC LC-394 had the most successful launches from all the sites



The highest launch-success ratio: KSC LC-39A

- KSC LC-394 achieved a 76.9% success rate while getting a 23.1% failure rate



Payload vs Launch Outcome Scatter Plot

- We can see that all the success rate for low weighted payload is higher than heavy weighted payload

Section 5

Predictive Analysis (Classification)

Classification Accuracy

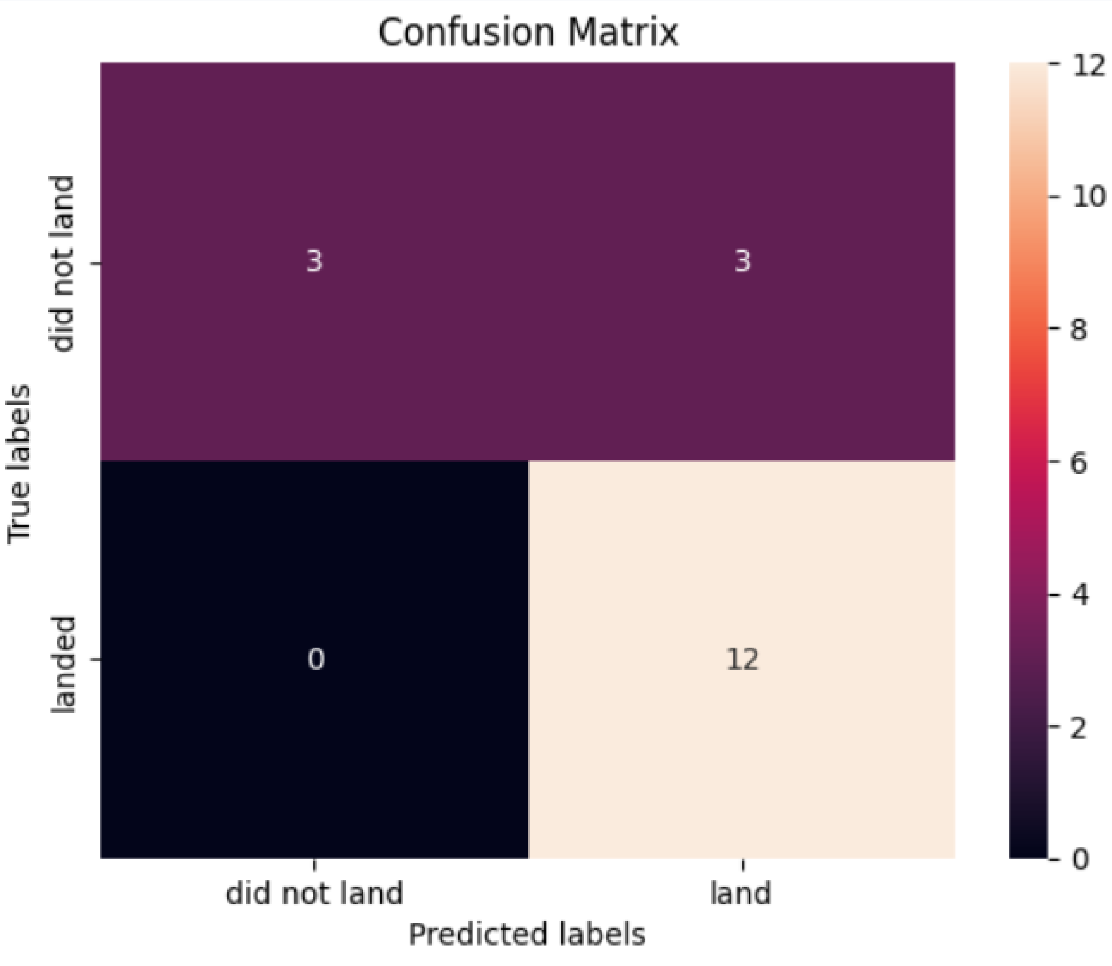
- The decision tree classifier is the model with the highest classification accuracy

```
[34]: algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.8875
Best Params is : {'criterion': 'gini', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix

- The confusion matrix of a decision tree classifier shows that the classifier can distinguish between different classes. The main problem is false positives, such as an unsuccessful landing, which the classifier marks as a successful landing.



Conclusions

- Can we draw the following conclusion:
- • The tree classifier algorithm is the best machine learning method for this dataset.
- • Lighter payloads (defined as 4000kg and below) perform better
- than heavier payloads.
- • Beginning in 2013, SpaceX's launch success rate will increase,
- The time to 2020 is proportional and will eventually refine future releases.
- • KSC LC-39A was the most successful launch of all sites; 76.9%
- • SSO Orbit has the highest success rate; 100% and more than 1 time.

Thank you!