Data Science in Winning the Space Race

Chen Qi
28 October 2022

TABLE OF CONTENTS

Executive Summary

- 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
- Results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result from Machine Learning Lab

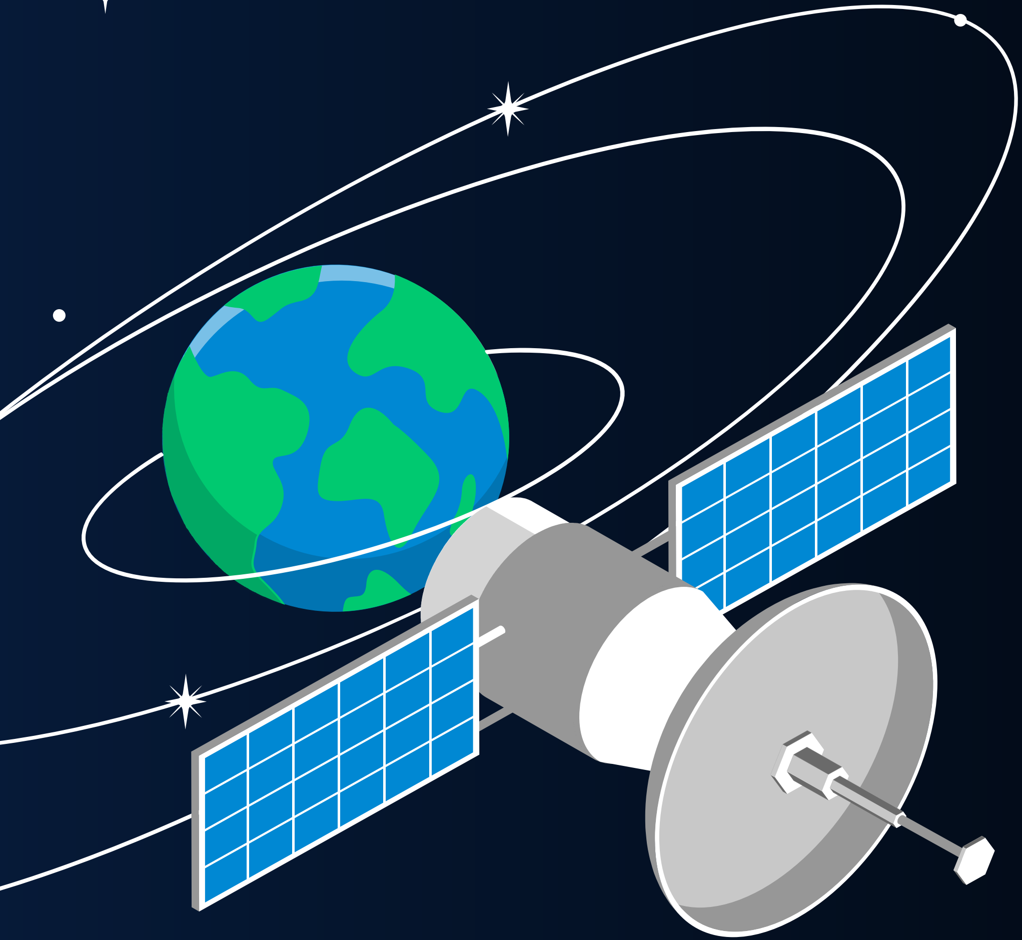
Introduction

The rises of SpaceX seems to abridge our distance between us and the Space. More specifically, SpaceX is revolutionary company because it is offering individual rocket launch (i.e. the Falcon 9) at as low as 62 million dollars, whereas other providers spends to 165 millions to do the same things. The decision of SpaceX to reuse the first stage of the rocket launch is the main reason for such a dramatic decrease in cost. Additionally, if they repeat such process, their cost is likely to further decrease. Positioning myself as a data scientist in the rivalry company of SpaceX, in this project, I will create a machine learning pipeline to predict the landing outcome of the first stage of SpaceX. Thus, identifying the right price to bid against SpaceX for a rocket launch.

Introduction

Problems to be solved:

- Identifying factors that are significantly affecting the landing outcome.
- The relationship between variables and how such correlations affect the landing.
- The optimal conditions for successful landings



01

Methodology

Methodology

Executive Summary:

- Data Collection
- Data Wrangling
- Exploratory Data Analysis (EDA) with Visualization and SQL
- Interactive Visual Analysis with Folium and Plotly Dash
- Predictive Analysis with Classification Model

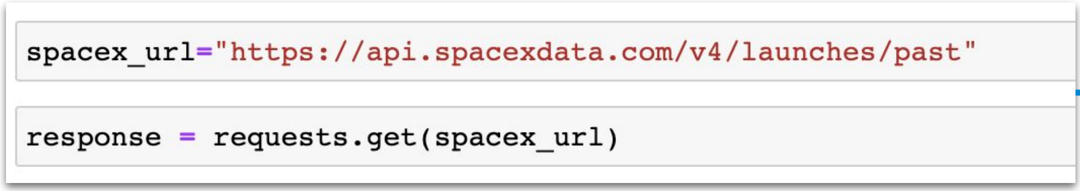
Methodology - Data Collection

- Data Collection is the process of gathering and examining the data that are valuable in finding the answers to our questions. In this project, I applied two methods in collecting related data: From REST APIs and from Web Scraping.

- Rest APIs:
  - It started with the get request. Then, I read and decode the response in JSON form and convert it into a pandas dataframe using json_normalize(). Last but not least, we do a basic cleaning for the data: checking for missing values and fill with according data.

- Web Scraping
  - I use the BeautifulSoup to extract the launch records as HTML tabl and then parse and convert the table into a pandas dataframe for basic cleaning and further operations.

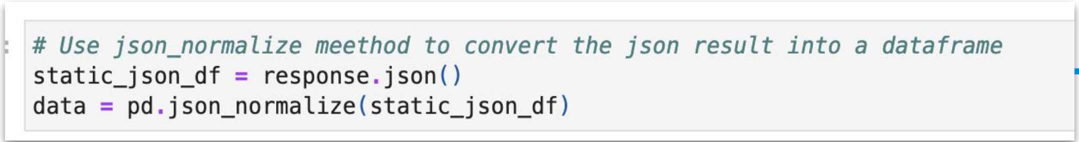
Data Collection - SpaceX API



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

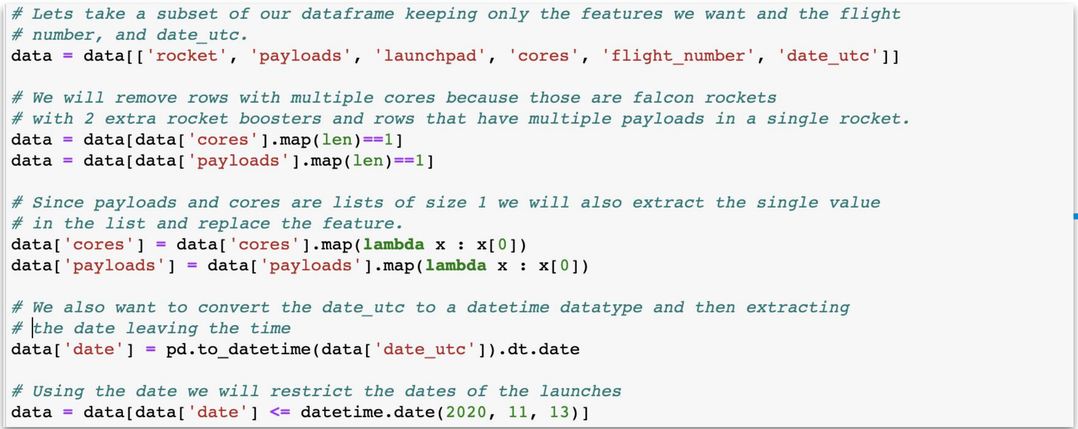
```
response = requests.get(spacex_url)
```

Identify the data resources and use get() to import it.



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

Apply json_normalize for converting data into pandas dataframe.

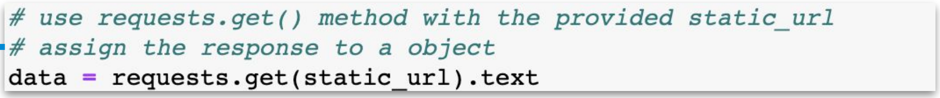


```
# Lets take a subset of our dataframe keeping only the features we want and the flight
# number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets
# with 2 extra rocket boosters and rows that have multiple payloads in a single rocket.
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 list and replace the feature.
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 leaving the time
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)]
```

Perform basic data cleaning and filling the missing values,

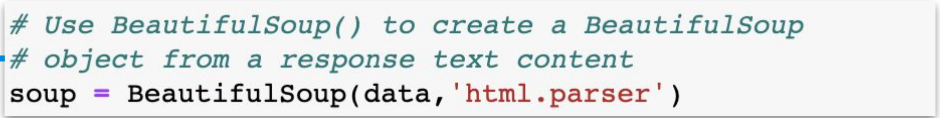
Data Collection - Web Scraping

Obtain the data from our resources



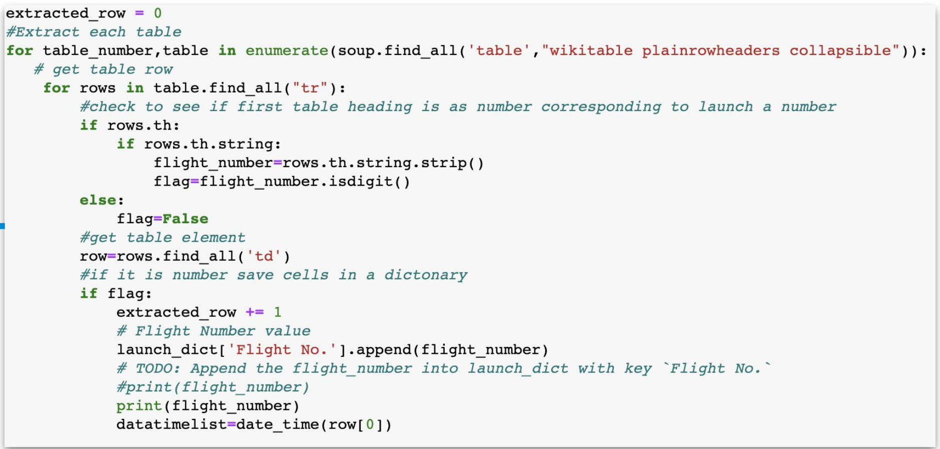
```
# use requests.get() method with the provided static_url
# assign the response to a object
data = requests.get(static_url).text
```

Create a BeautifulSoup object from previous response



```
# Use BeautifulSoup() to create a BeautifulSoup
# object from a response text content
soup = BeautifulSoup(data,'html.parser')
```

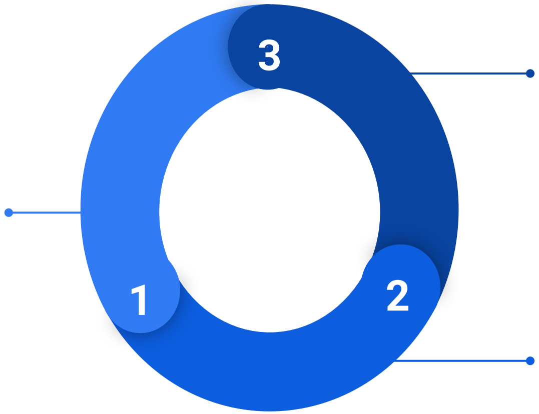
Extract every rows and columns we need from previous BeautifulSoup Object



```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            launch_dict['Flight No.'].append(flight_number)
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            #print(flight_number)
            print(flight_number)
            datatimelist=date_time(row[0])
```

Methodology - Data Wrangling

Concept:
As part of the preprocessing of data, data wrangling will clean and convert complex and messy data into a simpler and more valuable one. As the wrangling is finished, Exploratory Data Analysis can be carried out and the big pictures and general insights of the data can be learnt.



Data Calculation

Calculate the number of launches on each site and the number of mission outcome per orbit type.

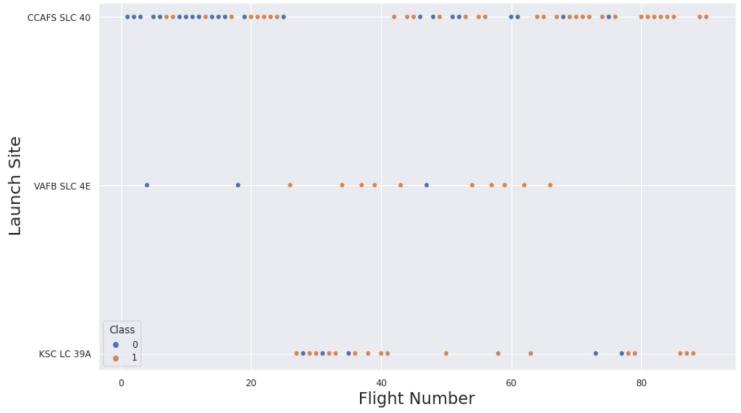
Exportation

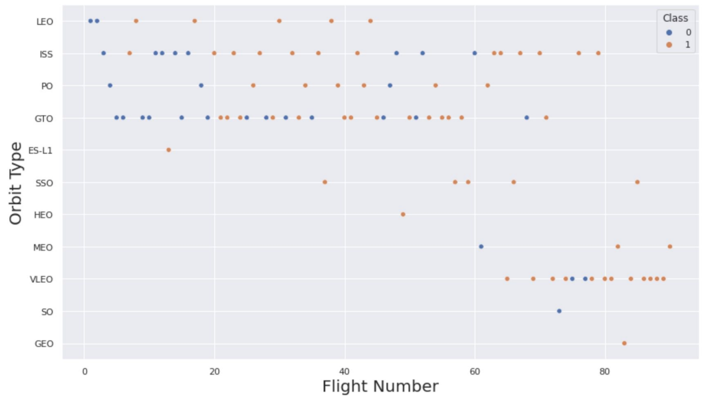
Export the outcome to a csv for further operations.

Labeling

Create outcome label from the outcome column, make it easier for further analysis.

Methodology - EDA with Visualization



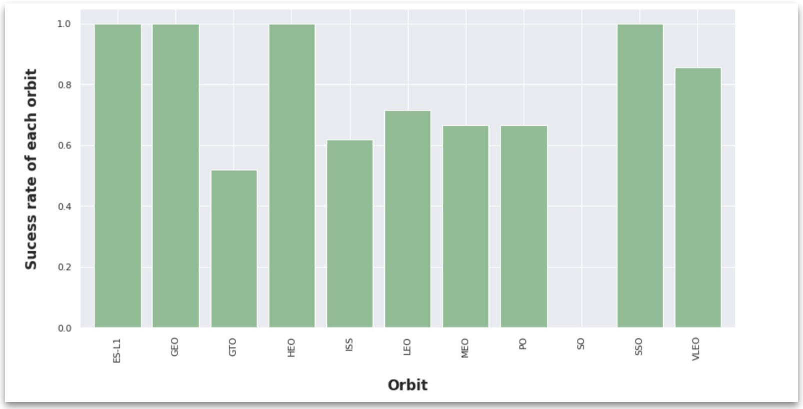


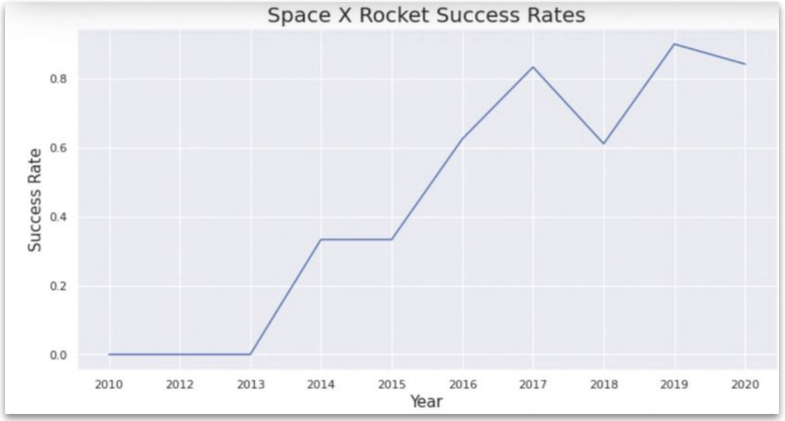
First, I would like to use scatter graph to depict relationships between the attributes. Based on the scatter plots we then move on to determine what further visualization is needed. For example, the relationships between these variables are scatter plotted:

- Payload and Flight No.
- Payload and Orbit Type
- Payload and Launch Site
- Flight Number and Orbit Type
- Flight Number and Launch Site

From the relationship depicted, it can be easily found which factors are affecting the most to the successful landings.

Methodology - EDA with Visualization





From previous results from scatter plot, i move on to further examine the relationship of orbit and successful rate with bar plot.

Then, to generalize a trend of success rate, i move on to draw an line graph, showing how the success rate vary based on time.

Methodology - EDA with SQL

SQL commands provided us with another way of understanding and formatting the data. In this project, i decided to follow the instruction to play around with all the SQL commands and gain more insights from them. However, at this point, i will rely more on the visualization technique for this project. Examples of the sql commands are:

- Listing the date when the first successful landing outcome in ground pad was achieved.
- Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
- Listing the total number of successful and failure mission outcomes.
- Listing the names of the booster_versions which have carried the maximum payload mass.
- Listing the failed landing_outcomes in drone ship, their booster versions, and launch sites names for in year 2015.
- Rank the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20, in descending order.

Methodology - Advanced Visualization

Dashboard with Plotly Dash

To allow users to better interact and understand the study, i built an interactive dashboard with Plotly Dash. Pie Charts are plotted showing the total launches in different sites. And scatter plots are drawn to show the relationship with outcome and payload mass for different booster version.

Interactive Map with Folium

To visualize the launch data into a interactive map, I took the coordinates at each launch site and add a marker around them with introduction on the label.

I then assigned the dataframe launch_outcomes to class 0 (failure) and 1 (success) with Red and Green markers.

Then, the answers to question like " How close the launch sites with railways, highways and coastlines?" can be easily answered by the map.

Methodology - ML Analysis (Classification)

1 **Building the Model**

While building the models, we need to: split the training and evaluating set; Decides what ML models to use; and set parameters and algorithms to GridSearchCV and fi the data

2 **Model Evaluation**

Plot the confusion matrix; Get tuned hyperparameters for each type of algorithms; examine the overall

3 **Model Improvement**

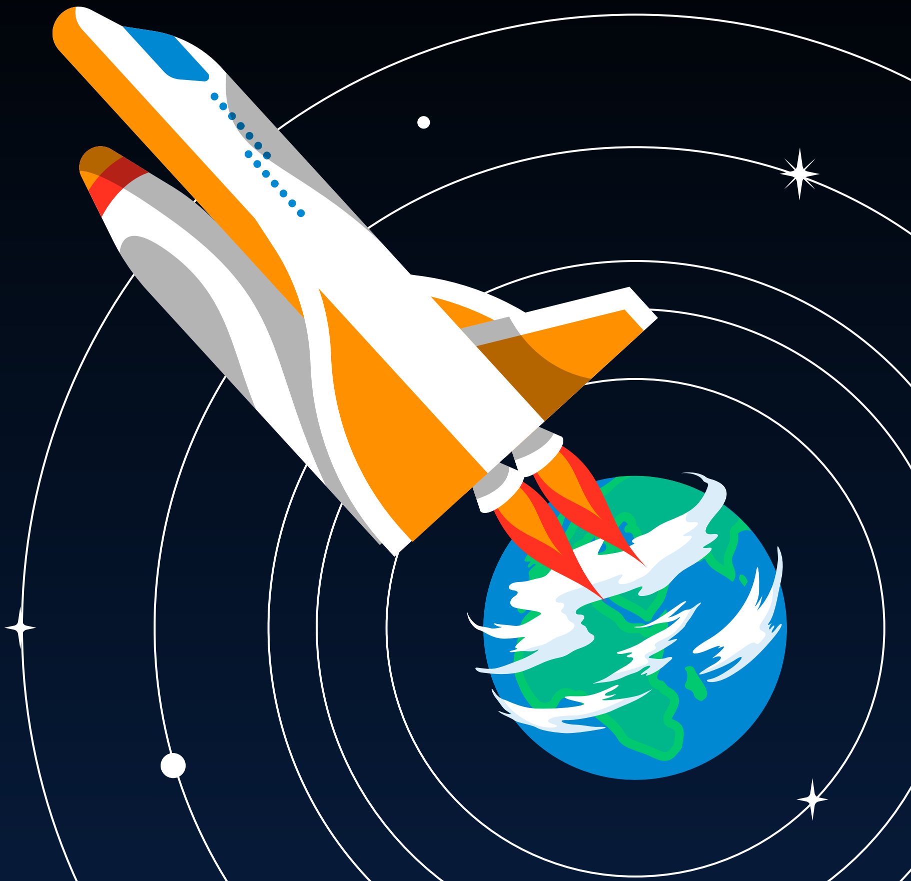
Apply Feature Engineering and Algorithm Tuning.

4 **Find the Best Model**

Select the model with best accuracy as the final model.

02

Results

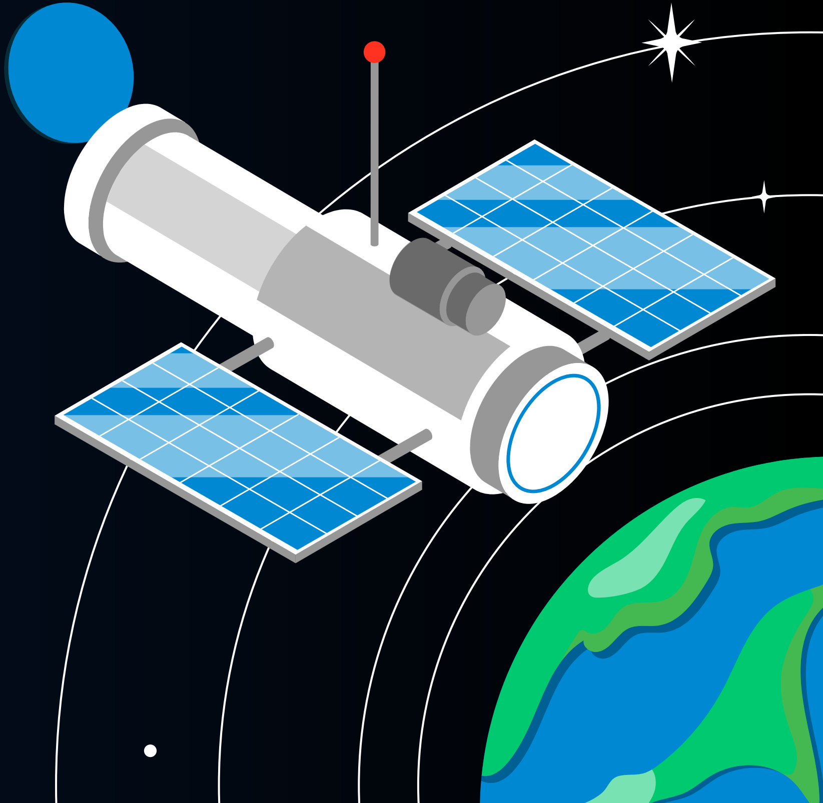


Results

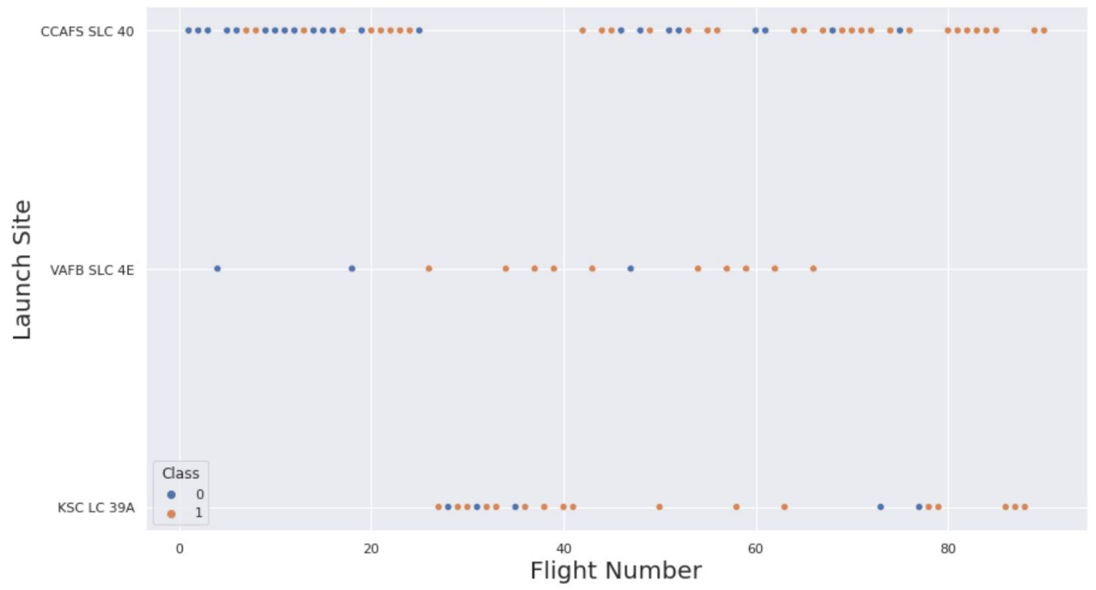
In this section of the report, I will demonstrate the results mainly from **3 different perspectives:**

- Exploratory Data Analysis (EDA) results
  - Carried out three most outstanding correlations I found while doing EDA with visualization.
- Interactive Analytics demo in screenshots
- Predictive analysis results

EDA ANALYSIS



Results - Insights from EDA



Flight Number vs. Launch Site

The scatter plot between Flight number and Launch Site shows that sites with larger flight number ten to have a greater chance of successfully launching.

However, site CCAFS SLC40 shows the least pattern of this general trend.

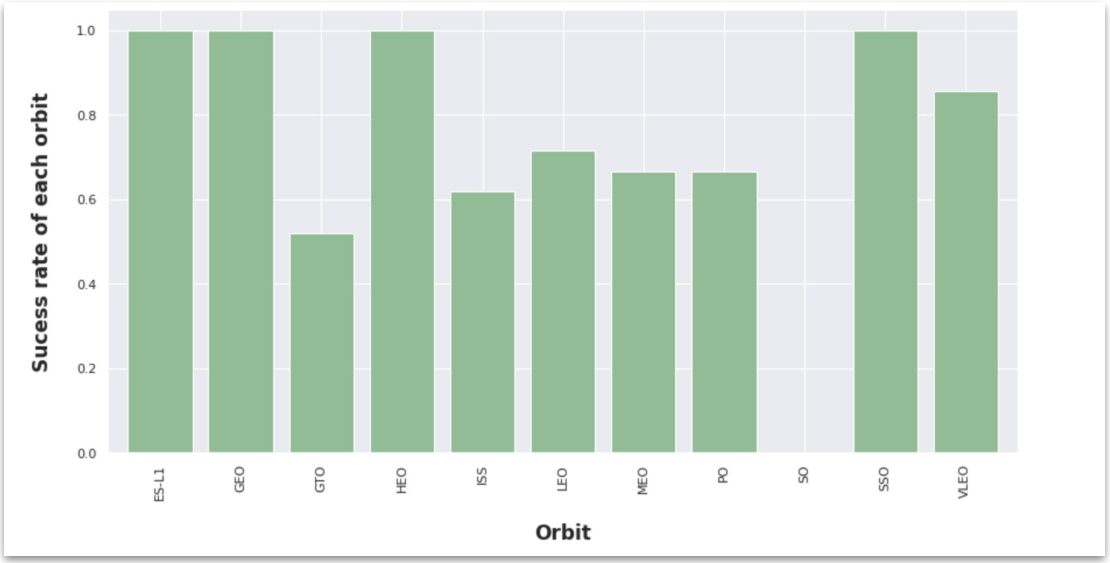
Results - Insights from EDA

Success Rate vs. Orbit Type

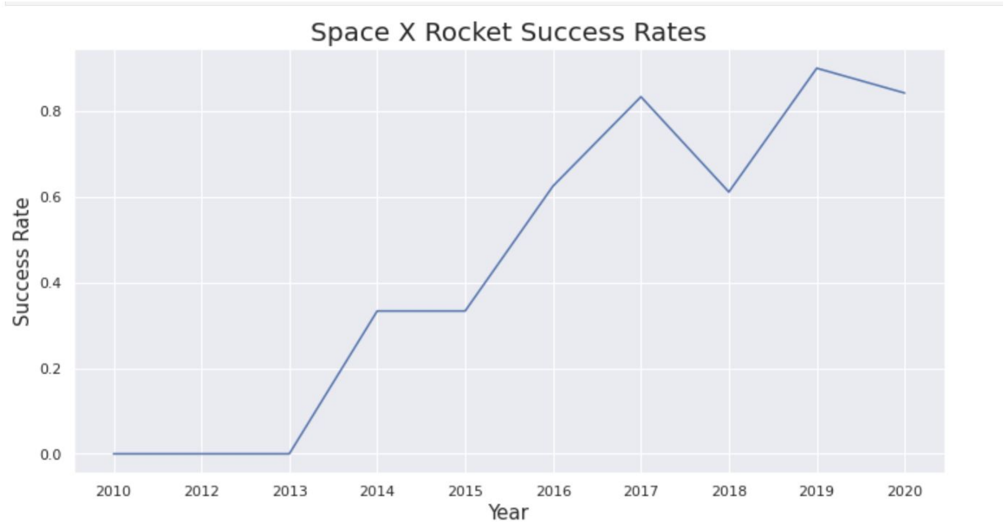
From the bar graph, the fluctuation of successful landing rate fluctuates at different orbits.

The ES-L1, GEO, HEO, and SSO orbits seem to be the safest for landing.

While the GTO orbit is showing a very low possibility of successful landing.



Results - Insights from EDA

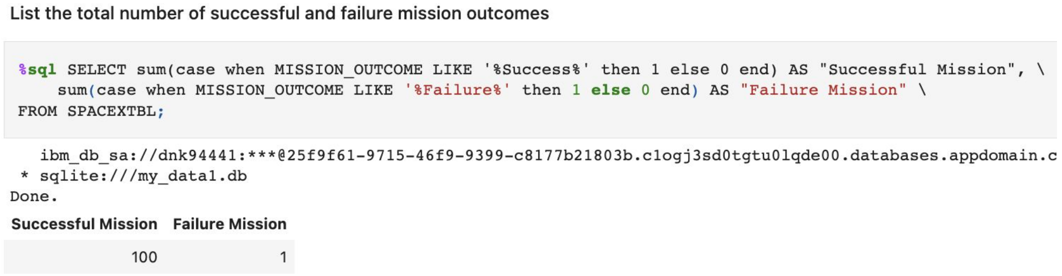


Success Rate vs. Orbit Type

From the line graph, it is not hard to find that the successful rate of landing is generally increasing as time passes by from 2010-2020.

However, what is worth noticing is that there is a decrease in success rate at 2018. When conducting further analysis, we need to keep this special time in mind.

Results - SQL exploration



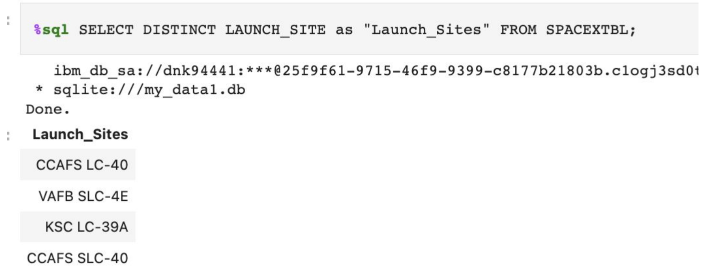
List the total number of successful and failure mission outcomes

```
%sql SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "Successful Mission", \
    sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "Failure Mission" \
FROM SPACEXTBL;
```

```
   ibm_db_sa://dnk94441:***@25f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.c
 * sqlite:///my_data1.db
Done.
```

Successful Mission	Failure Mission
100	1

I used wildcard like % to filter WHERE MissionOutcome was a success or a failure



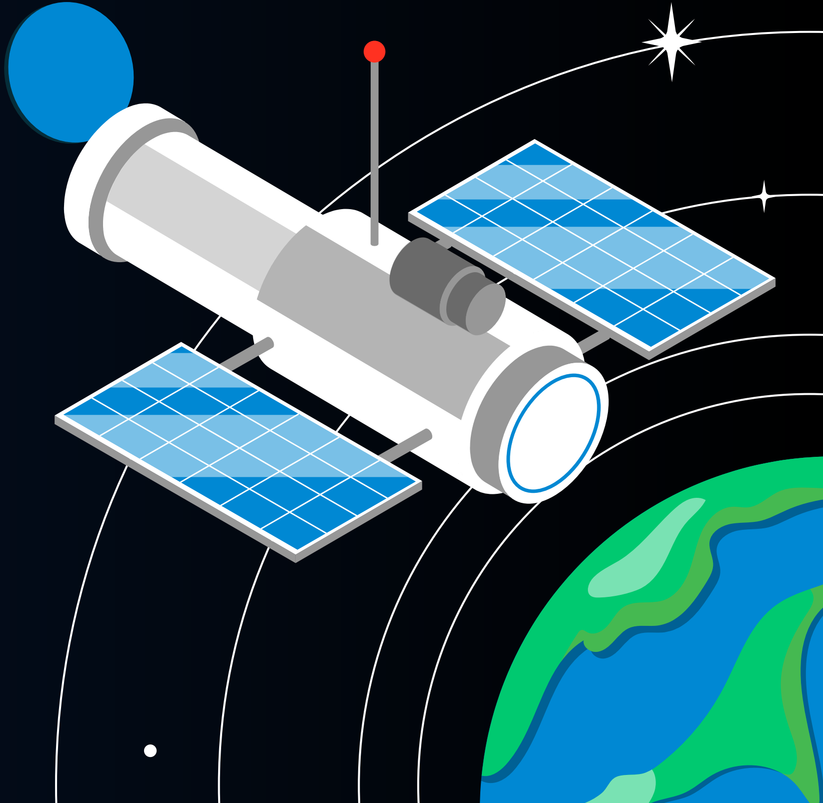
```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

```
   ibm_db_sa://dnk94441:***@25f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0t
 * sqlite:///my_data1.db
Done.
```

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

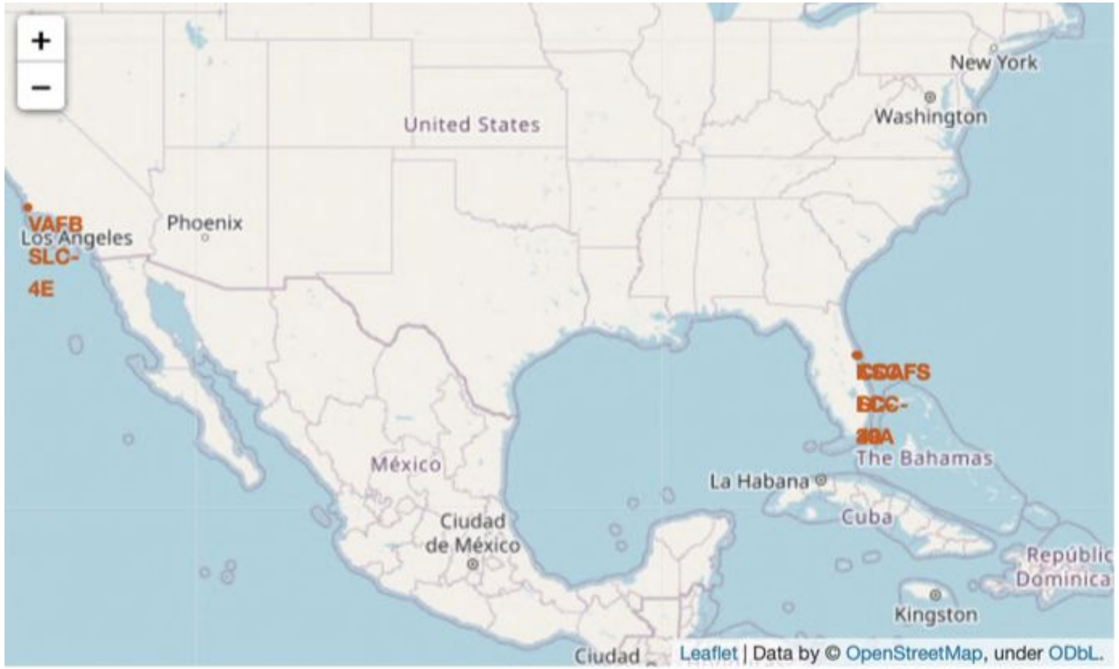
Used DISTINCT function to show only unique launch sites from SpaceX data.

Launch Sites Proximities Analysis

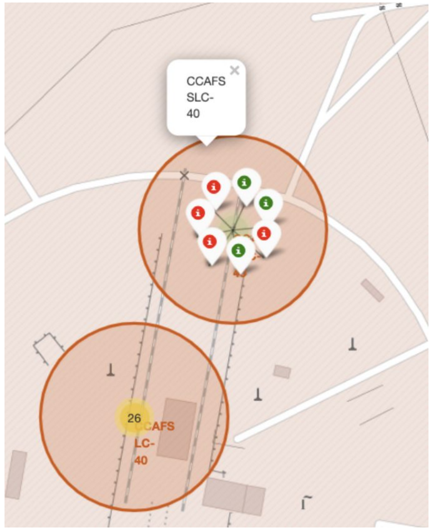


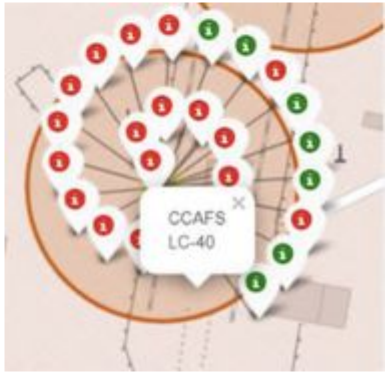
Results - Map

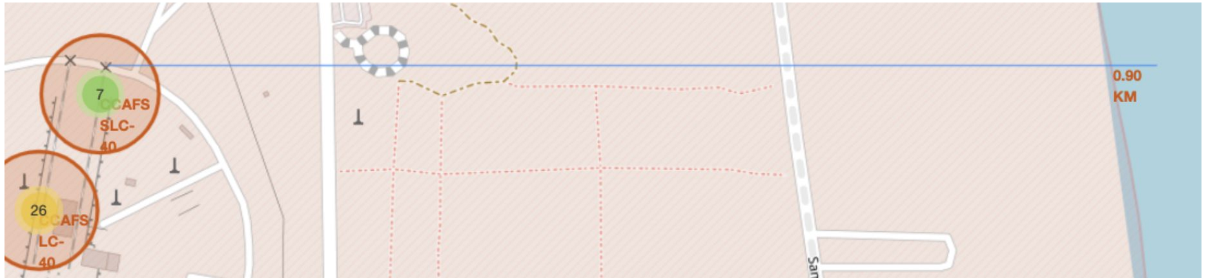
From this map covering the space of the United States, we can see where each Launch sites are located



Results - Proximities Analysis

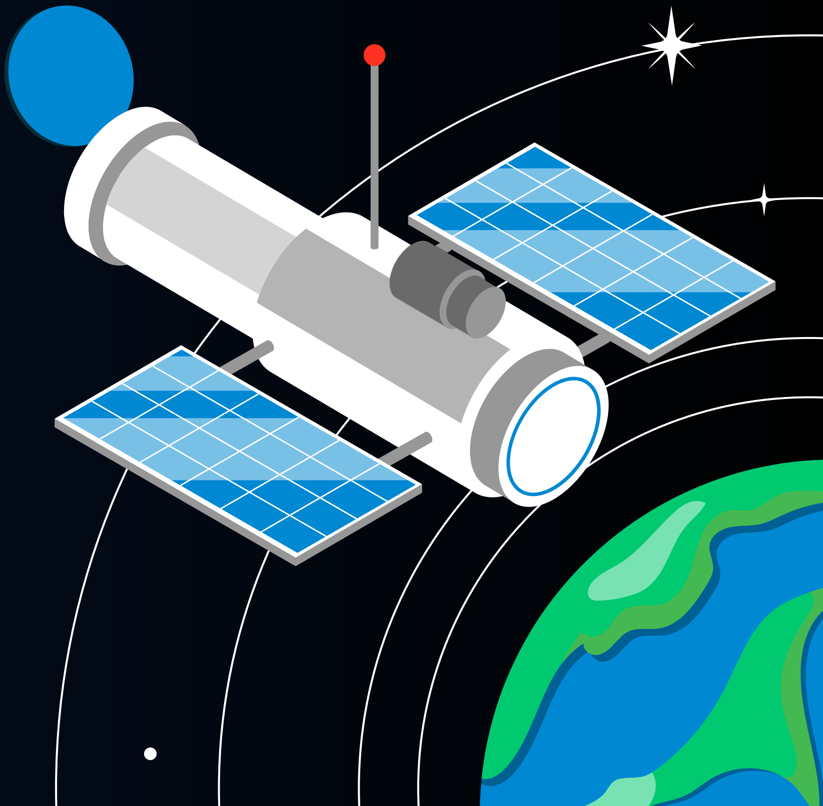




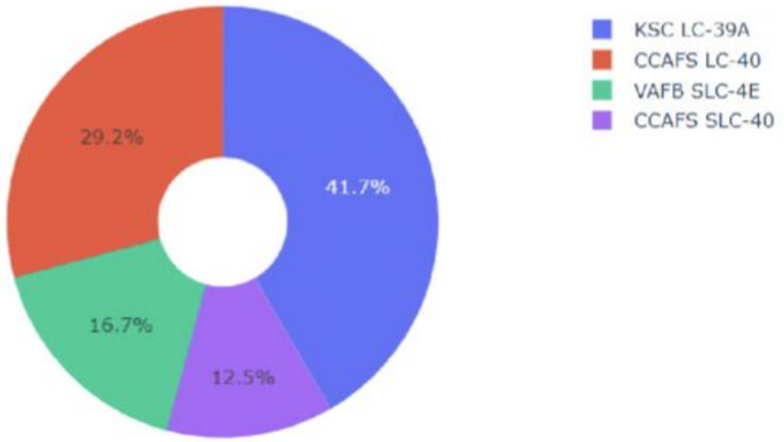


From the maps and the labels, a concise structure of these locations can be built and vividly shown to the users. Not only can we see the success rate at each site, wen can also observe that how different proximity to city structure influence the success rate.

Dashboard With Plotly Dash

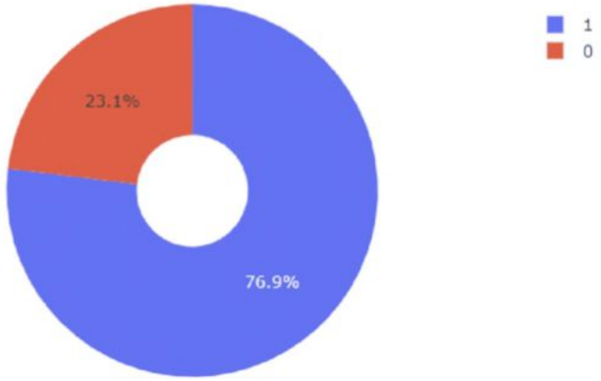


Results - Dashboard

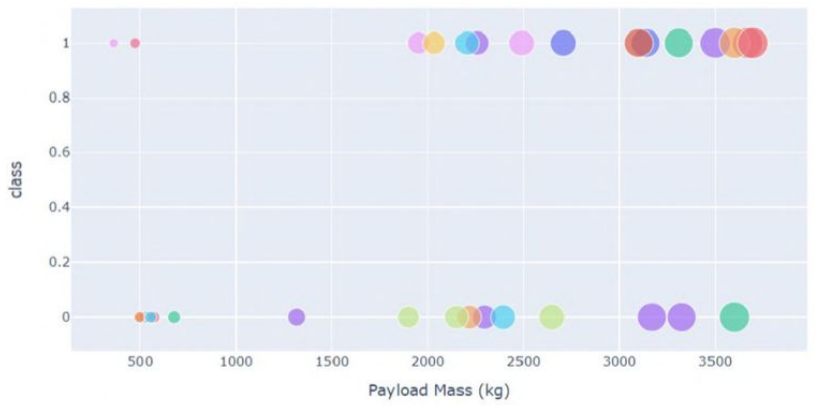


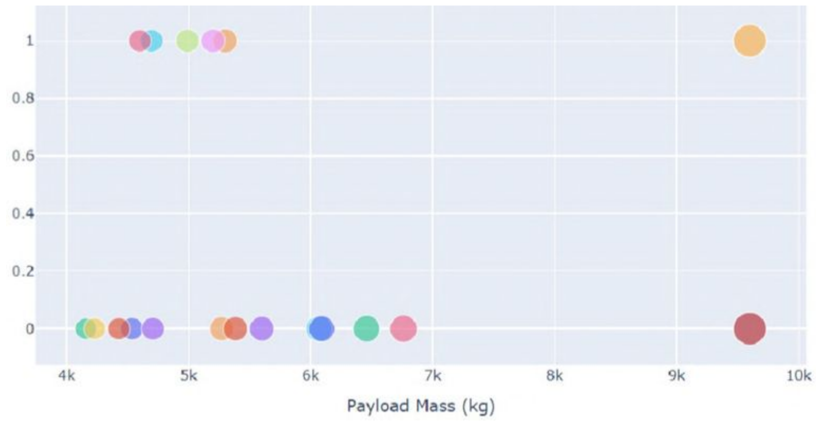
From the dashboard, not only can we see how the successful launches are distributed among different sites.

We can also zoom in to see how the success rate is for each site. For example, the pie chart to the right shows that KSC LC-39A achieved a 76.9% success rate.



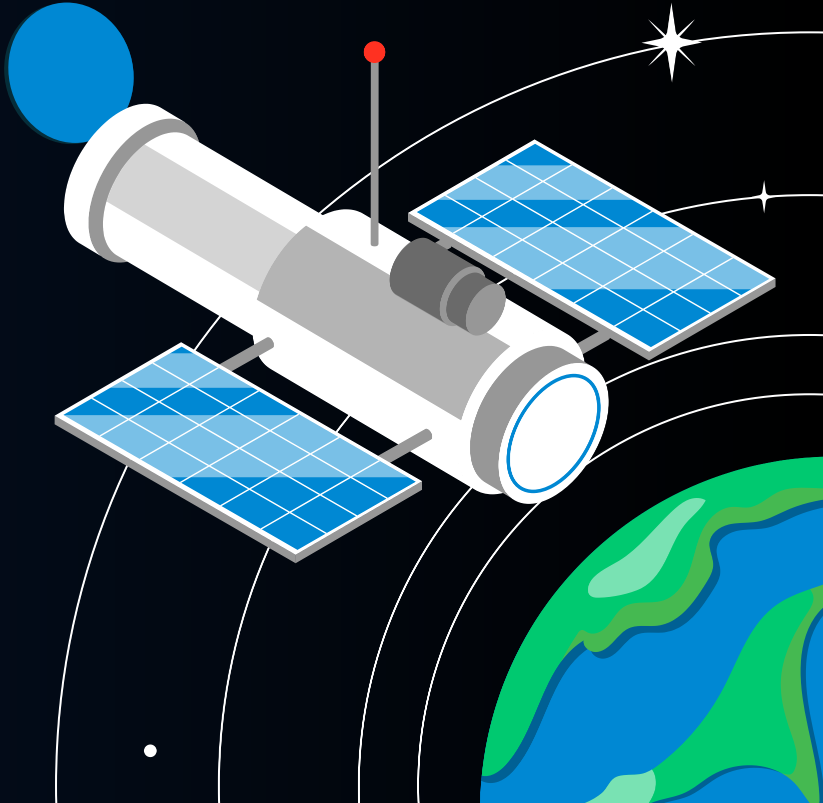
Results - Dashboard





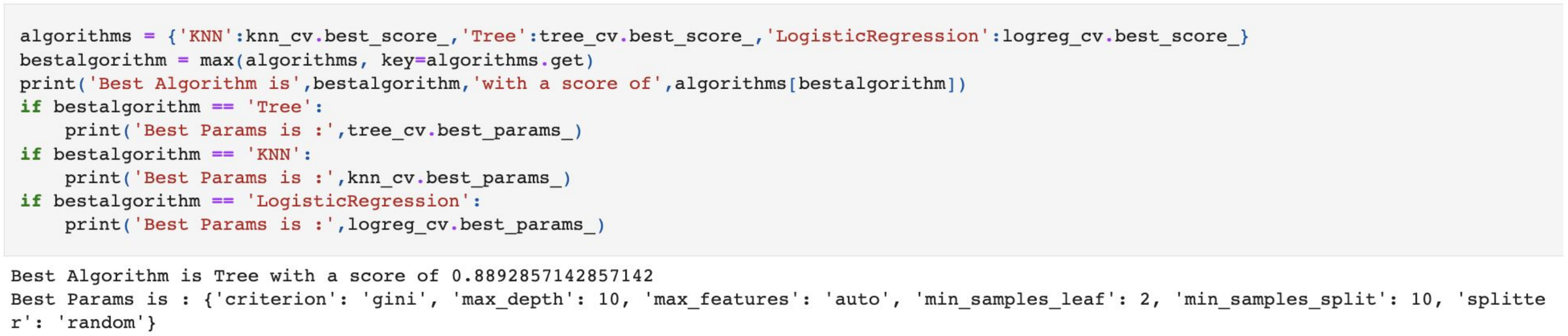
Through Dashboard, we can also view how the success rate vary from low-weight payload mass to high-weight payload mass. We can see that the success rate for low-weight payload mass is seriously higher than that of the high-weight payload mass.

Predictive Algorithms Classification Models



Results - Algorithm Accuracy

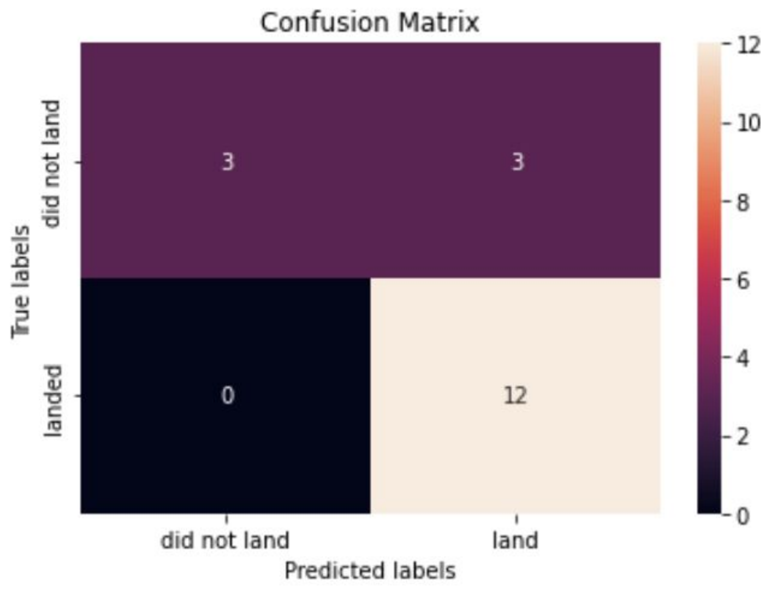
As we can see from the output from the code below, we could identity the best algorithm for this project to be the **Tree Algorithm** due to its highest classification accuracy.



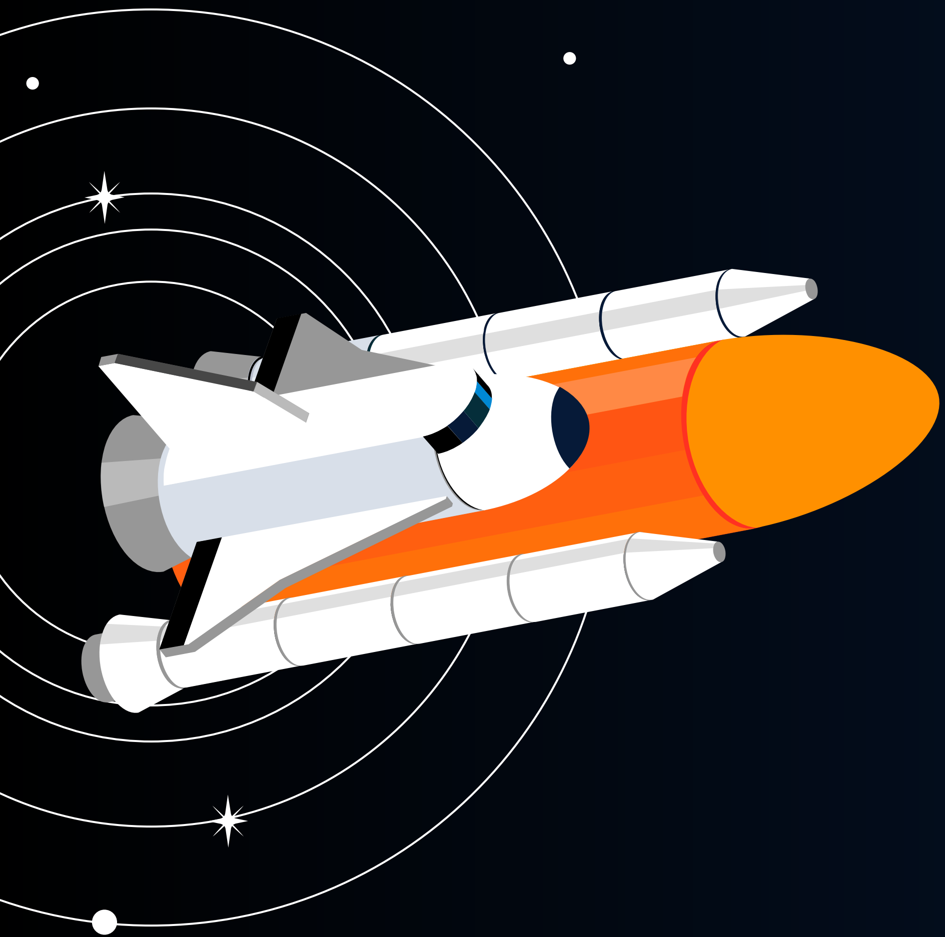
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params 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 0.8892857142857142
Best Params is : {'criterion': 'gini', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitte
r': 'random'}
```

Results - Confusion Matrix



The confusion matrix for the decision the classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives. That is unsuccessful landing marked as successful landing



03

Conclusion

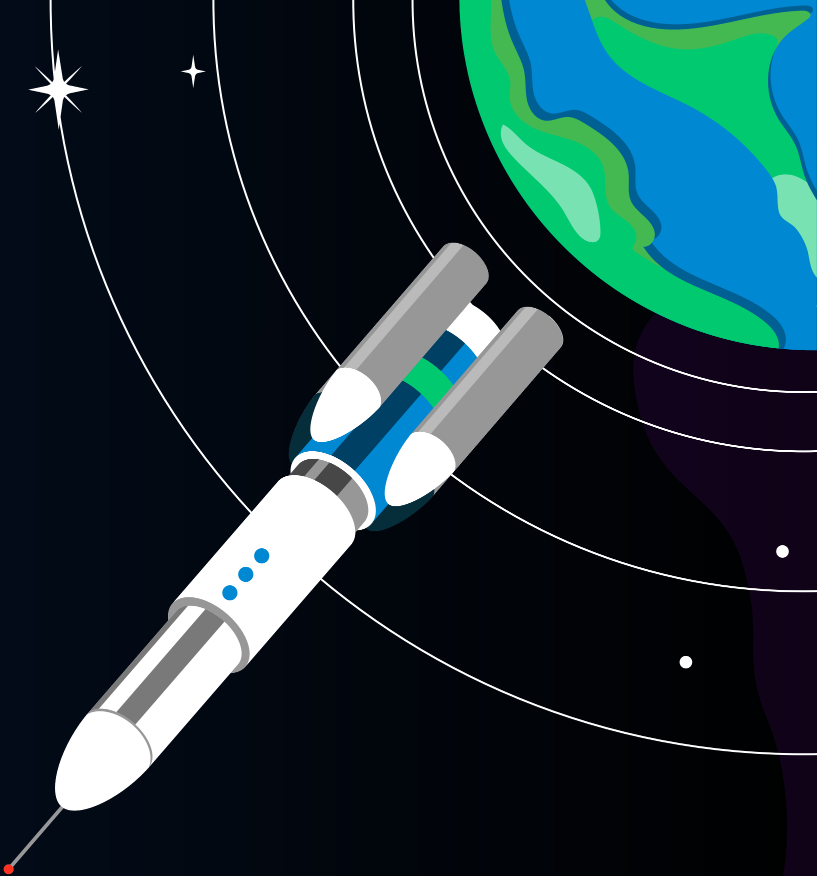
Results - Classification Accuracy

Here are the final conclusions:

- ❏ The Tree Classifier Algorithm is the Machine Learning that best fits the need of our project.
- ❏ The weight of payloads can be a significant influence to whether a a landing is successful. More specifically low-weight payloads (<4000kg) perform much better.
- ❏ KSC LC-39A has the most successful launches of any sites at 76.9%
- ❏ Starting from the year 2013, the success rate for SpaceX launches is increased, directly proportional time in years to 2020, which it will eventually perfect the launches in the future.

03

Appendix



Appendix

Resources of Codes:

- ❏ Data Collection
  - ❏ From APIs: https://github.com/ChenQi000320/SpaceY-Project—-IBM-DS/blob/master/jupyter-labs-spacex-data-collection-api.ip
  - ❏ WebScraping: https://github.com/ChenQi000320/SpaceY-Project—-IBM-DS/blob/master/jupyter-labs-webscraping.ipynb
- ❏ Data Wrangling: https://github.com/ChenQi000320/SpaceY-Project—-IBM-DS/blob/master/labs-jupyter-spacex-Data%20wrangling.ipynb

Appendix

Resources of Codes:

- ❑ EDA with Data Visualization:https://github.com/ChenQi000320/SpaceY-Project—-IBM-DS/blob/master/jupyter-labs-eda-dataviz.ipynb
- ❑ EDA with SQL:https://github.com/ChenQi000320/SpaceY-Project—-IBM-DS/blob/master/jupyter-labs-eda-sql-coursera_sqllite.ipynb
- ❑ Data Mapping:https://github.com/ChenQi000320/SpaceY-Project—-IBM-DS/blob/master/lab_jupyter_launch_site_location.ipynb
- ❑ Dashboard:https://github.com/ChenQi000320/SpaceY-Project—-IBM-DS/blob/master/spacex_dash_app.py
- ❑ Prediction (Classification Model):https://github.com/ChenQi000320/SpaceY-Project—-IBM-DS/blob/master/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb