IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Timmy Li
19th April 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection using web Scraping and API
  - Exploratory Data Analysis (EDA) with data visualization, data wrangling, and SQL
  - Interactive maps with Folium
  - Dashboards using Plotly Dash
  - Machine Learning predictive analysis
- Summary of all results
  - Exploratory Data Analysis results
  - Interactive analytics using maps and dashboard
  - Predictive Analytics result

Introduction

- Project background and context

  Space X promotes the launches of its Falcon 9 rockets on its website, charging $62 million, while other companies charge at least $165 million per launch. One of the main reasons for this cost difference is that Space X can reuse the initial stage of the rocket. Hence, by determining the success of the first stage's landing, it becomes possible to estimate the cost of a launch. Such information can be useful for other companies who want to compete with Space X in the rocket launch market. The objective of this project is to develop a machine learning pipeline that can predict the likelihood of a successful landing of the first stage..

- Problems you want to find answers
  - What are the variables that play a role in determining the success of a rocket's landing?
  - Is there a correlation between different factors that contribute to the success rate of a rocket's landing?
  - What are the specific conditions that are necessary for a landing program to be successful?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX API
  - Web scraping from Wikipedia.
- Perform data wrangling
  - Sorting and cleaning data
  - One-hot encoding to arrange different classification models
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Developing, fine-tuning, and evaluating classification models, including construction, optimization, and accuracy assessment.

Data Collection

- The data was collected using various methods

  - Data collection involves the systematic gathering and measuring of information on specific variables within an established system to enable the evaluation of outcomes and the answering of relevant questions. The dataset in question was obtained through both REST API and web scraping methods on Wikipedia.

  - With REST API, the data was extracted using a GET request, and the resulting response content was decoded in JSON format before being converted into a pandas data frame with the help of json_normalize().

  - Afterward, the data was cleaned, checked for missing values, and any gaps were filled as needed.

  - For web scraping, the launch records were extracted as an HTML table using BeautifulSoup, parsed, and then converted into a pandas data frame for further analysis.

Data Collection – SpaceX API

- Getting response from API
- Using json_normalize to convert response to JSON file in data frame
- Filtering and cleaning data whilst finding missing values

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

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

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

```
# 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 h
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 featur
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)]
```

The link to the notebook is
https://github.com/timmyli045/SpaceX-Data-Science-Capstone-/blob/main/Data%20Collection%20API.ipynb

Data Collection - Scraping

- Requesting Falcon9 Wiki page response from url
- Creating BeautifulSoup object from html response
- Extract all variable names from html header and add data to keys

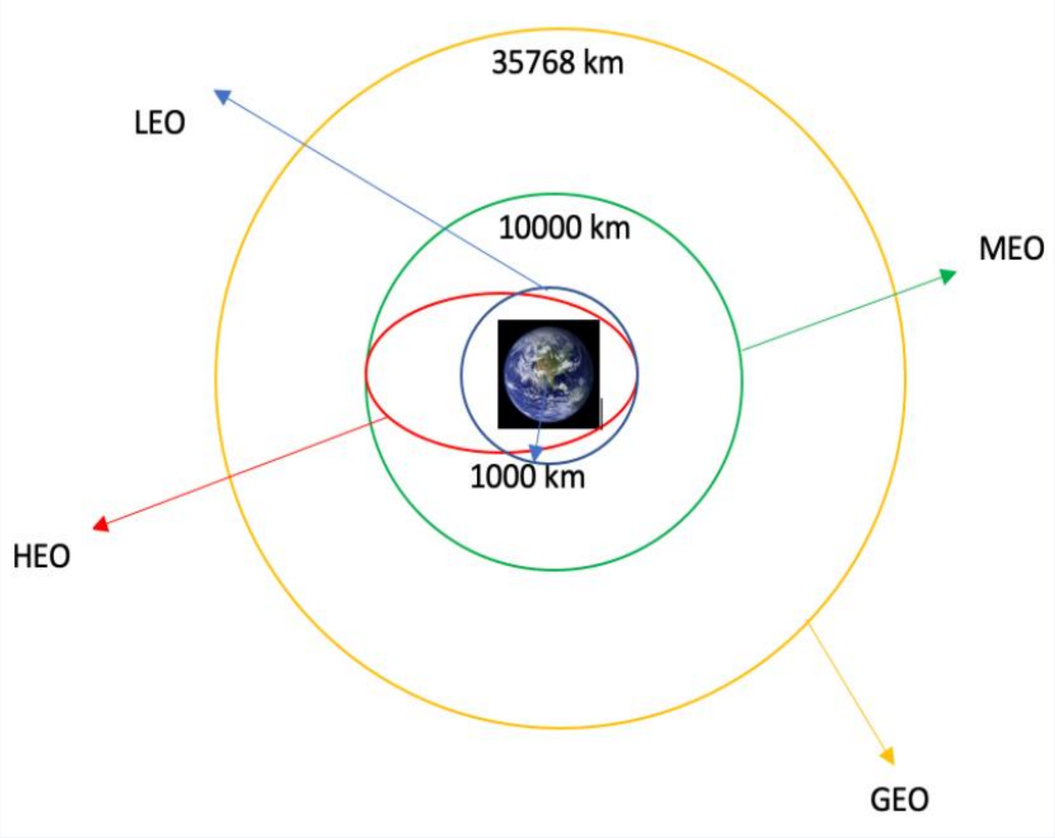
The link to the notebook is https://github.com/timmyli045/SpaceX-Data-Science-Capstone-/blob/main/Data%20Collection%20with%20Web%20Scraping.ipynb

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

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

```
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')
```

Data Wrangling

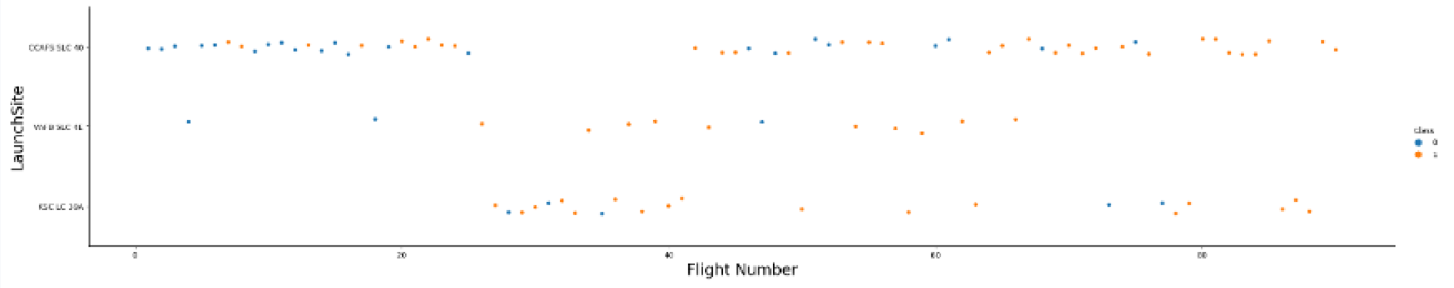


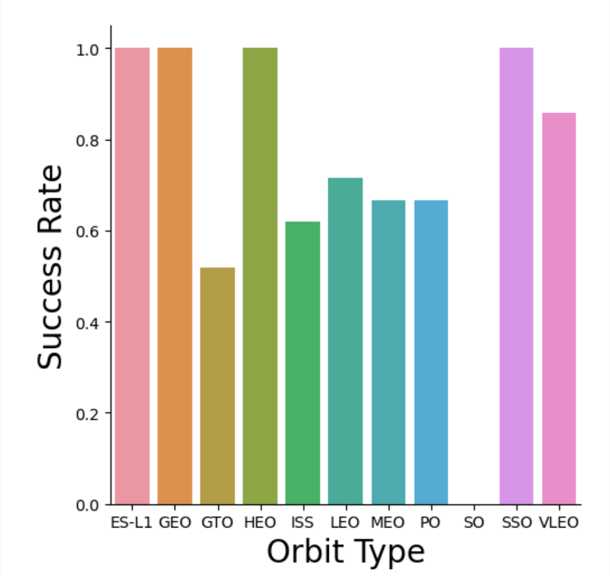
- Data Wrangling refers to the process of cleaning and consolidating complex and unorganized data sets for the purpose of facilitating easy access and analysis.
- To achieve this, the first step involves calculating the number of launches that occur at each site, followed by an assessment of the frequency and count of mission outcomes for each orbit type.
- Next, a landing outcome label is created based on the outcome column to make future analysis, visualization, and machine learning more accessible. Finally, the results are exported to a CSV file.

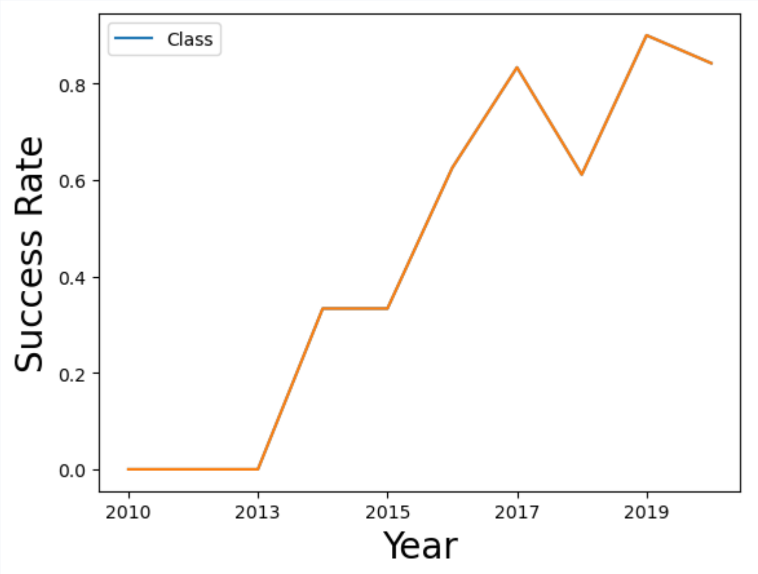
The link to the notebook is https://github.com/timmyli045/SpaceX-Data-Science-Capstone-/blob/main/Data%20Wrangling.ipynb

EDA with Data Visualization

- Scatter plots display the relationship between different attributes and facilitate the identification of the most influential factors that contribute to landing success.
- The relationship between attributes can be easily interpreted using bar graphs, which is particularly useful for determining the most successful orbits.
- Meanwhile, line graphs can show the annual trend of attribute patterns, such as launch success rate over time.







The link to the notebook is https://github.com/timmyli045/SpaceX-Data-Science-Capstone-/blob/main/EDA%20with%20Data%20Visualization.ipynb

EDA with SQL

Implementing SQL to gather and interpt data from dataset:

- Show launch site names
- Show 5 records of launch sites starting with 'CCA'
- Display total payload mass for NASA (CRS) launched boosters
- Show average payload mass for booster version F9 v1.1
- List the date of the first successful landing on a ground pad
- List booster names with successful drone ship landings and a payload mass between 4000 and 6000
- List the total number of successful and failed mission outcomes
- List booster versions that have carried the highest payload mass
- List failed landing outcomes on drone ships, including booster versions and launch site names, for the year 2015
- Rank the landing outcomes or success count between June 4, 2010 and March 20, 2017 in descending order.

The link to the notebook is https://github.com/timmyli045/SpaceX-Data-Science-Capstone-/blob/main/EDA%20Using%20SQL.ipynb

Build an Interactive Map with Folium

- To make an interactive map that presents launch data, we used latitude and longitude coordinates for each launch site.
  - We added a circular marker to the map for each launch site, which was labeled with the site name.
- We then categorized the launch outcomes dataframe into success and failure classes, using red and green markers respectively, and incorporated MarkerCluster() for better visual representation.
- We calculated the distances between a launch site to its proximities answering some question for instance:
  - How close are the launch sites with nearest cities?
  - How close are the launch sites to highways, coastlines, and railways?
  - How are successful and unsuccessful landings affected from proximities?

The link to the notebook is https://github.com/timmyli045/SpaceX-Data-Science-Capstone-/blob/main/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- The dashboard comprises several components, including dropdown, pie chart, rangeslider, and scatter plot.
  - The dropdown component (dash_core_components.Dropdown) allows the user to choose between specific launch sites or all launch sites.
  - The pie chart (plotly.express.pie) displays the total success and failure outcomes for the selected launch site.
  - The rangeslider (dash_core_components.RangeSlider) enables users to choose a payload mass within a fixed range.
  - Finally, the scatter chart (plotly.express.scatter) illustrates the correlation between two variables, specifically the relationship between success and payload mass.

Predictive Analysis (Classification)

- Data Preparation and Building
    - Load dataset and normalizing data into NumPy
    - Split data into training and test sets.
- Model preparation
    - Selection of machine learning algorithms
    - Set parameters for each algorithm to GridSearchCV whilst training GridSearchModel models with training dataset
- Model evaluation
    - Get best hyperparameters for each type of model
    - Computing accuracy for each model with test dataset with each type of algorithms
    - Plot Confusion Matrix
- Model comparison
    - Comparison of models according to their accuracy and tune accordingly
    - Choosing the best performing model

Results

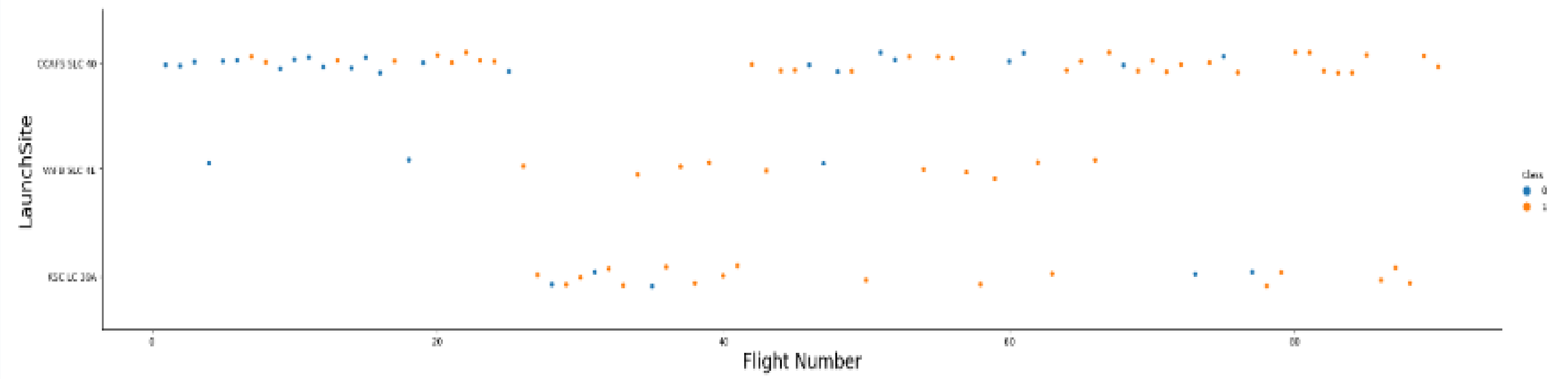
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

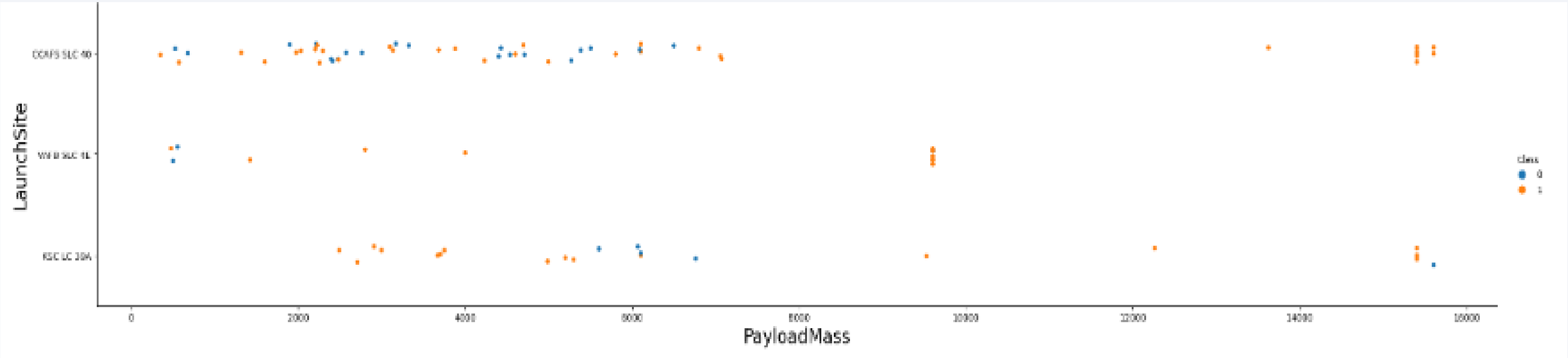
Flight Number vs. Launch Site

- From the plot, we found that the larger the flight amount at a launch site, the greater the success rate at a launch site respectively.



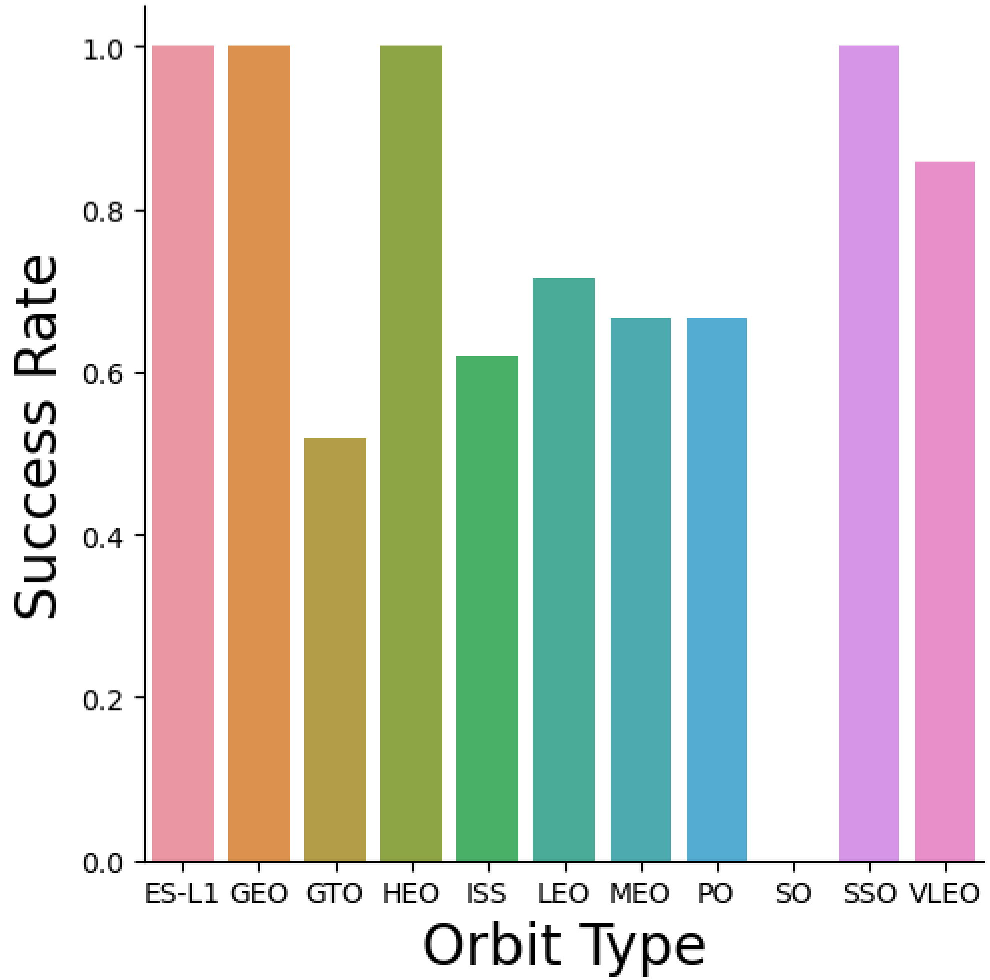
Payload vs. Launch Site

- From the plot, we found that the greater the payload (>7500kg), the probability of success will respectively be higher



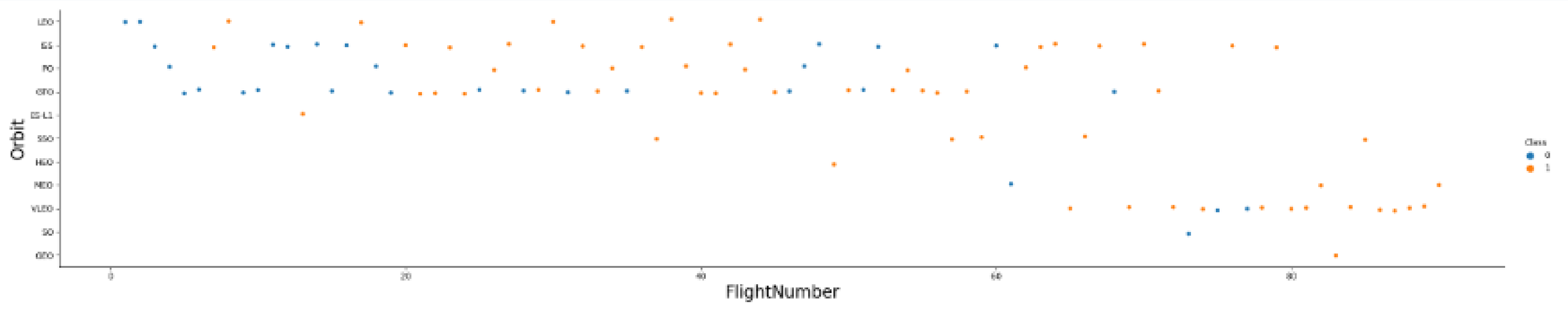
Success Rate vs. Orbit Type

- From the bar graph, we can conclude that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



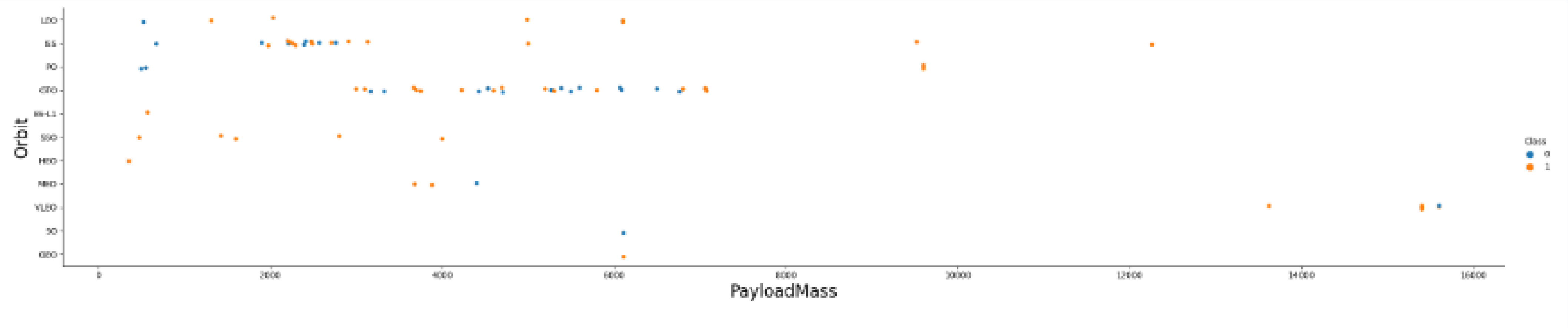
Flight Number vs. Orbit Type

- The data analysis showed that the success rate of LEO orbit increases as the number of flights increases, but for orbits like GTO, there is no apparent correlation between the success rate and the number of flights.



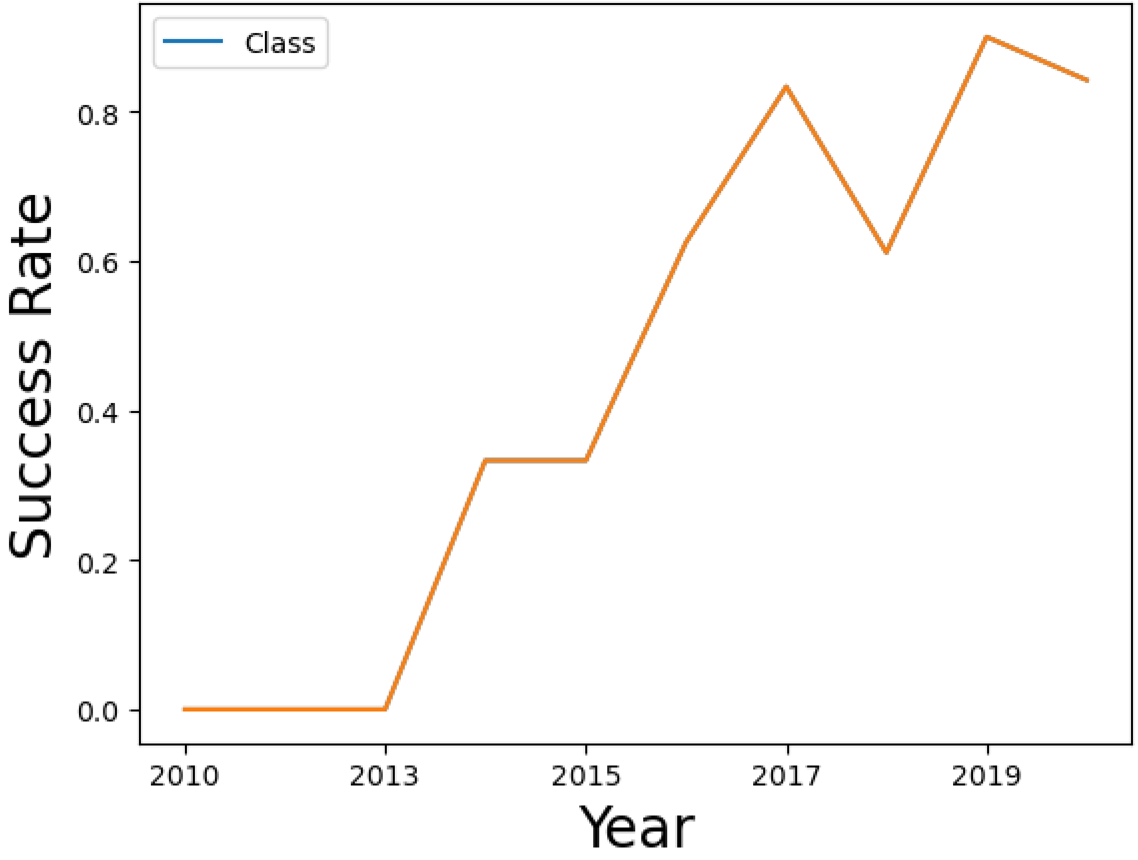
Payload vs. Orbit Type

- We can observe that with heavy payloads, the success rate landing are more for PO, LEO and ISS orbits. On the other hand, MEO and VEO orbit have a negative impact.



Launch Success Yearly Trend

- From the line graph, we can conclude that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

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

```
sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTABLE ORDER BY 1;
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b218
Done.
```

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

Launch Site Names Begin with 'CCA'

```
sql SELECT * FROM SPACEXTABLE WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30426/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

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

Total Payload Mass

- We calculated the total payload carried by boosters from NASA as 111268 using the query below

```
sql SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD FROM SPACEXTABLE WHERE PAYLOAD  LIKE '%CRS%';
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.a
Done.
```

total_payload
111268

Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1 as 2928

```
sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVG_PAYLOAD FROM SPACEXTABLE WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appd
Done.
```

avg_payload
2928

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 22nd December 2015

```
sql SELECT MIN(DATE) AS FIRST_SUCCESS_GP FROM SPACEXTABLE WHERE LANDING__OUTCOME = 'Success (ground pad)';
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.c
Done.
```

first_success_gp
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000 AND LANDING__OUTCOME = 'Succ
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30426/bludb
Done.
```

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

- Using **WHERE** and **AND** clauses to determine successful landings with a payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

```
sql SELECT MISSION_OUTCOME, COUNT(*) AS QTY FROM SPACEXTABLE GROUP BY MISSION_OUTCOME ORDER BY MISSION_OUTCOME;
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30426/bludb
Done.
```

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

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

Boosters Carried Maximum Payload

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

```
sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30426/bludb
Done.
```

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

2015 Launch Records

- We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

```
sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTABLE WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND DATE_PART('YEAR',
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30426/bludb
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

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

```
sql SELECT LANDING__OUTCOME, COUNT(*) AS QTY FROM SPACEXTABLE WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING
```

```
 * ibm_db_sa://dgl20130:***@125f9f61-9715-46f9-9399-c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30426/bludb
Done.
```

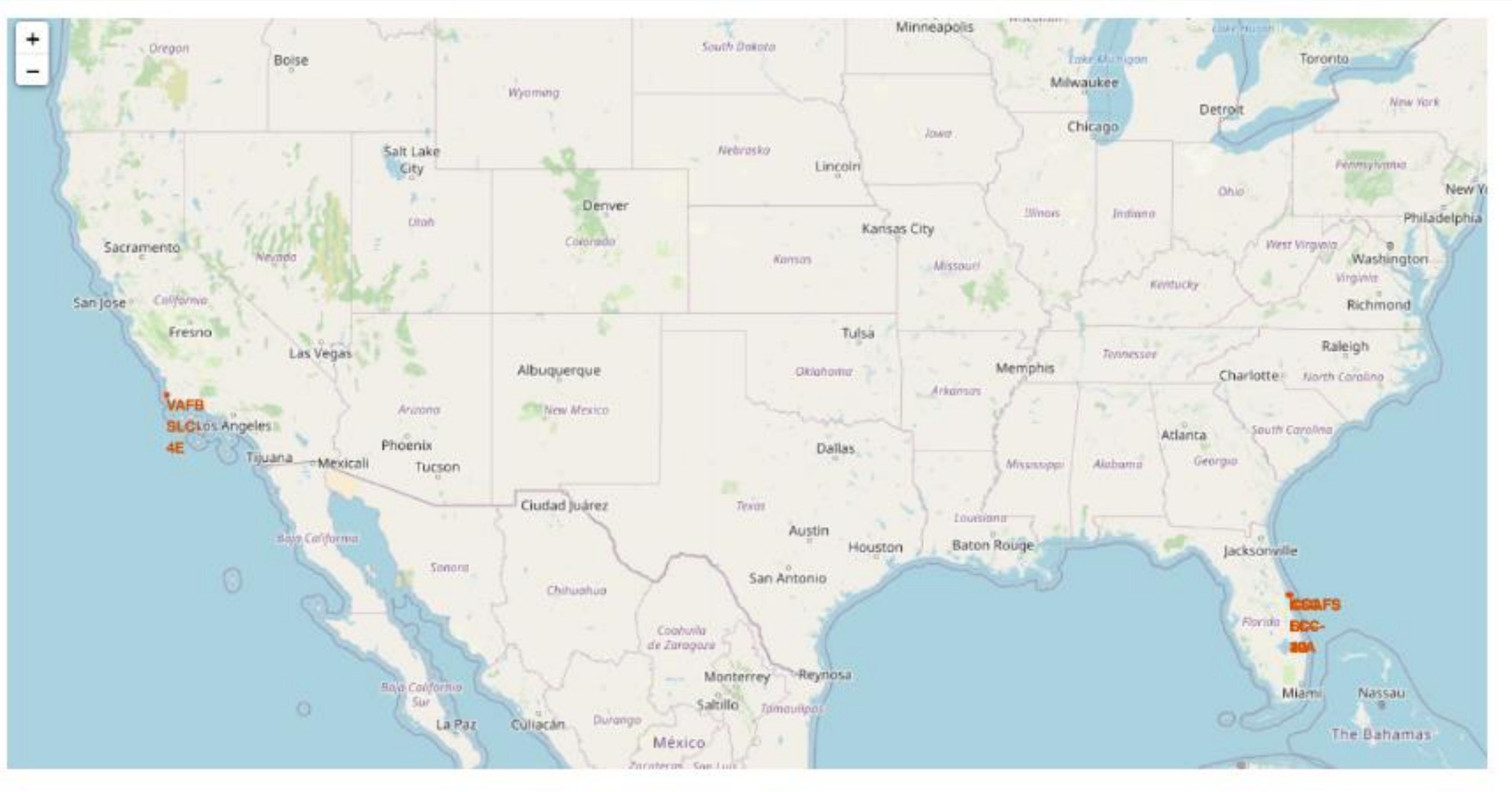
landing__outcome	qty
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

- 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.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

Section 4

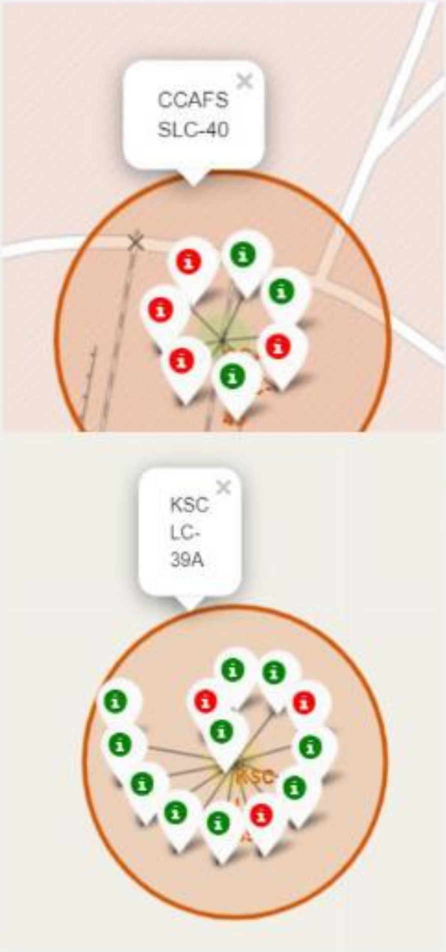
Launch Sites Proximities Analysis

All launch sites global map markers



- All SpaceX launch sites are in California and Florida, 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

37

Launch Site distance to landmarks

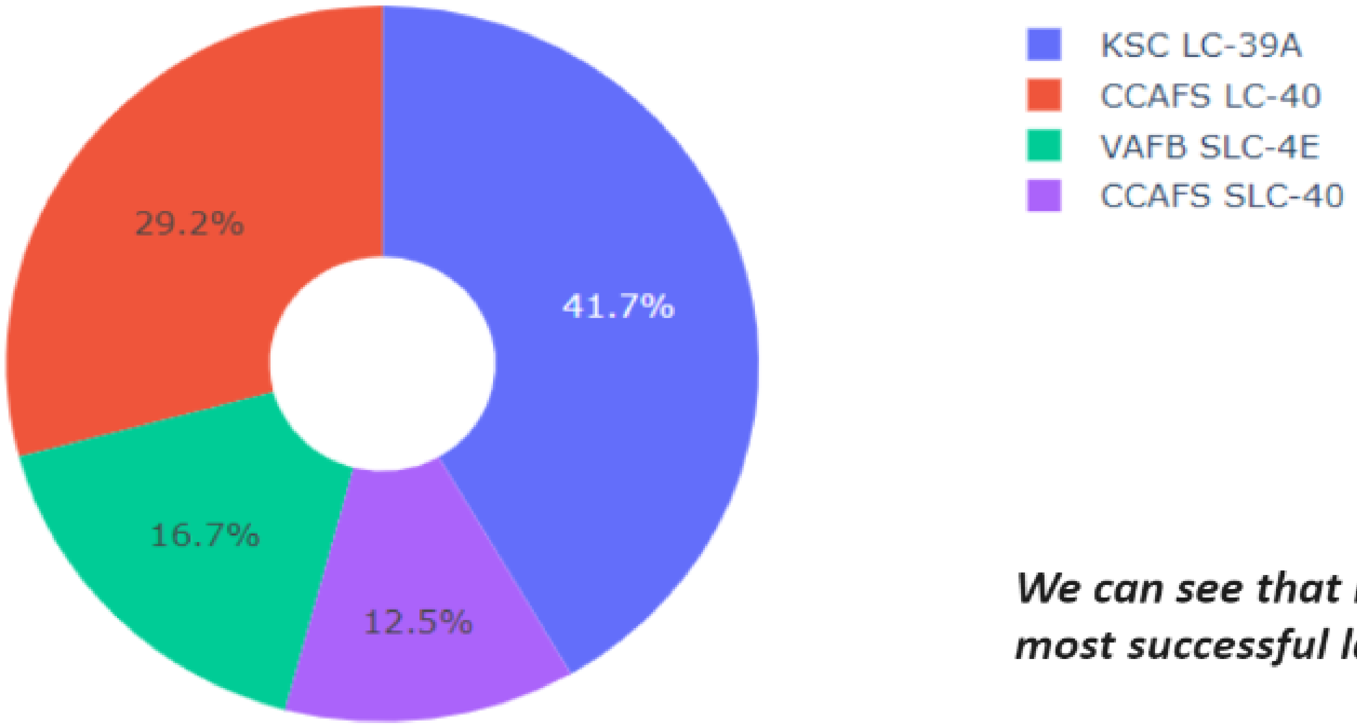
- Is CCAFS SLC-40 in close proximity to railways ? Yes
- Is CCAFS SLC-40 in close proximity to highways ? Yes
- Is CCAFS SLC-40 in close proximity to coastline ? Yes
- Do CCAFS SLC-40 keeps certain distance away from cities ? No

Section 5

Build a Dashboard with Plotly Dash

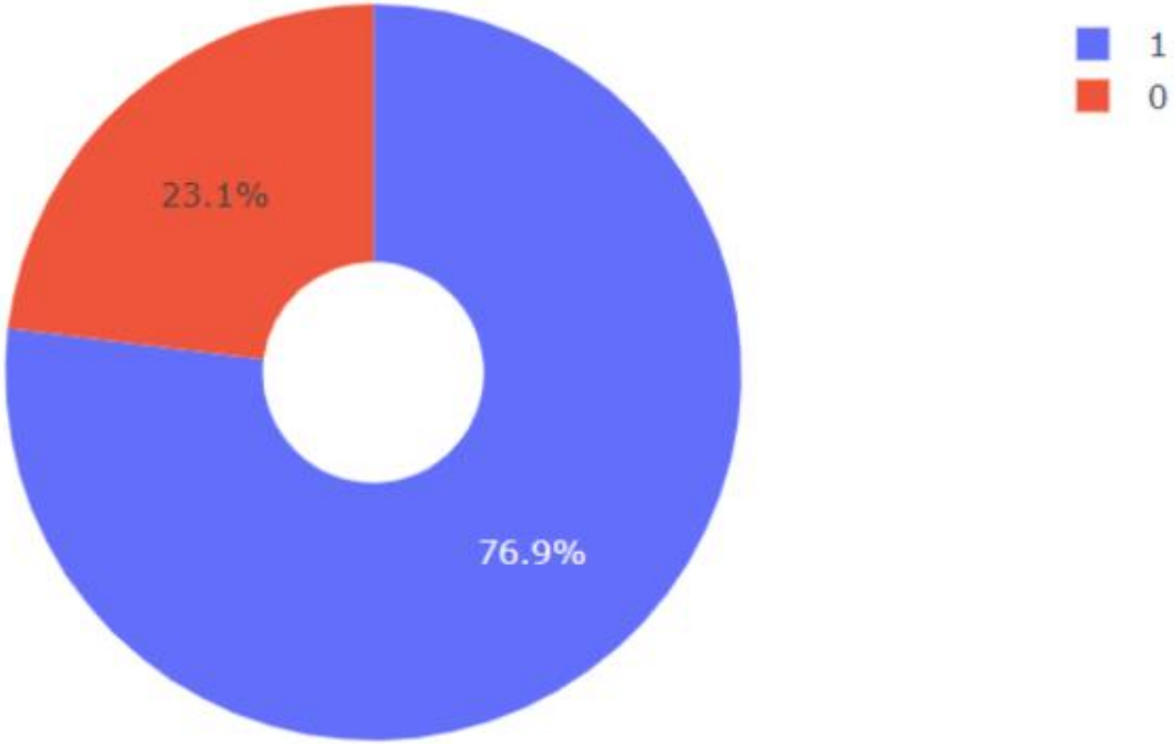
Pie chart showing the success percentage achieved by each launch site

Total Success Launches By all sites



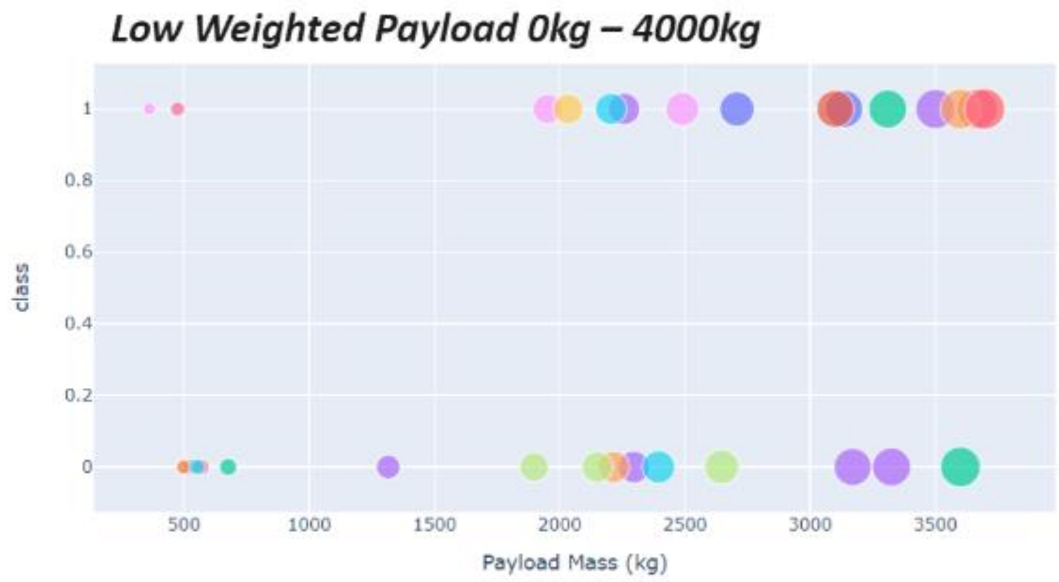
We can see that KSC LC-39A had the most successful launches from all the sites

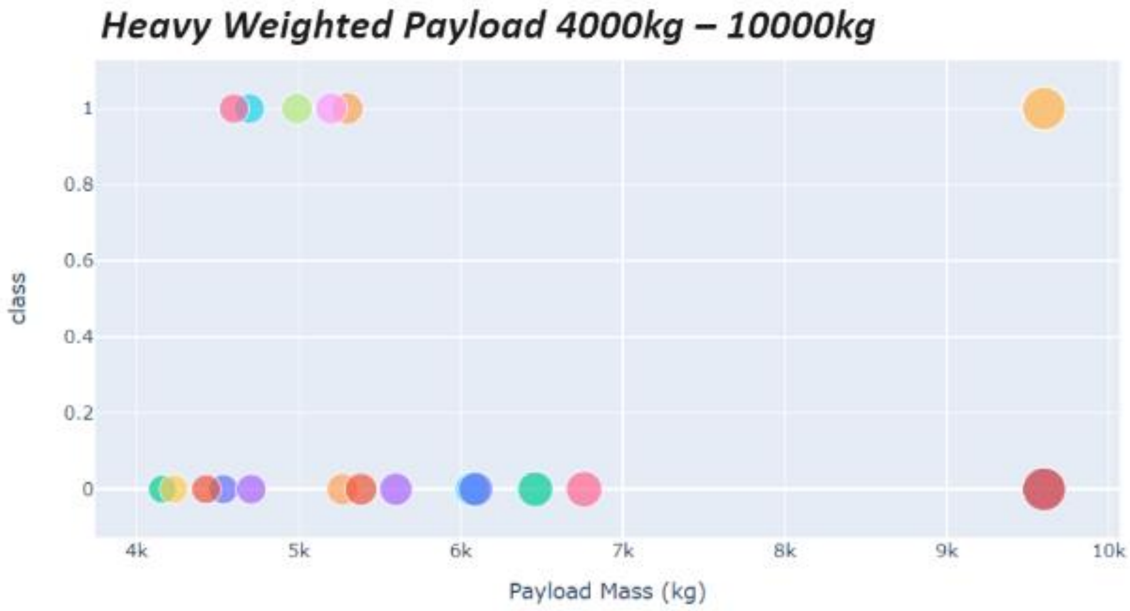
Pie chart showing the Launch site with the highest launch success ratio



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider





We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 6

Predictive Analysis (Classification)

Classification Accuracy

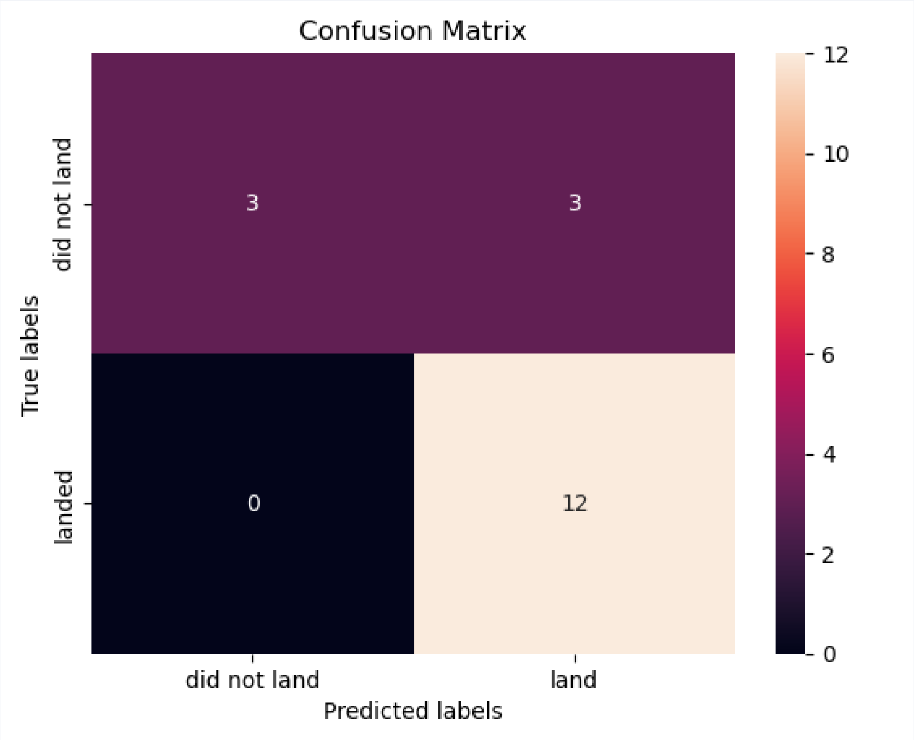
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

- By utilizing the following code, it was determined that the Tree Algorithm had the greatest classification accuracy, making it the optimal algorithm.

Confusion Matrix

- The decision tree classifier's confusion matrix indicates that it can differentiate between the various classes. However, the main issue is the occurrence of false positives, where the classifier incorrectly identifies unsuccessful landings as successful landings.



Conclusions

We can conclude that:

- Success of a mission is influenced by factors like launch site, orbit, and the number of previous launches. Gain in knowledge between launches could lead to a transition from launch failure to success.
- Orbits with the best success rates are GEO, HEO, SSO, ES-L1.
- Payload mass may be a criterion to consider for success, as some orbits require a light or heavy payload mass, though generally low-weighted payloads perform better.
- Decision Tree Algorithm is chosen as the best model for the dataset, even though test accuracy among all models used is the same. This is because Decision Tree Algorithm has better train accuracy.
- The reason for some launch sites being better than others (such as KSC LC-39A being the best) is not currently explained by the available data.

Thank you!