IBM **Developer**
SKILLS NETWORK

Winning Space Race
with Data Science

Umar Khan
Tuesday 2nd May 2023

Outline

- Executive Summary

- Introduction

- Methodology

- Results

- Conclusion

Executive Summary

- Summary of methodologies

  - API and web scraping data collecting

  - SQL and Python Libraries for Exploratory Data

  - Folium provides interactive visual analytics.

  - Machine Learning

- Summary of all results

  - Results of Exploratory Data Analysis

  - Visual pictures that are interactive

  - Predictions from Machine Learning

Introduction

Project background and context

Space X provides Falcon 9 rocket launches at a substantially cheaper cost than competitors, owing to its capacity to reuse the first stage. To compete with Space X, other businesses must know if their first stage will successfully land, and the project's purpose is to develop a machine learning pipeline that can anticipate this result.

Issues to resolve

- Identify the elements that influence whether the rocket's first stage lands successfully.

- Understand how features interact with one another.

- Determine the operational conditions that must be met in order for the landing program to be effective.

Section 1

Methodology

Methodology

- Data collection methodology:

    - Data was gathered via SpaceX APIs and Web Scraping techniques.

- Perform data wrangling

    - The data was cleaned and prepared for feature engineering analysis. This procedure covers handling null values and applying one-shot encoding to categorical variables.

- Perform exploratory data analysis (EDA) using visualization and SQL

- Perform interactive visual analytics using Folium and Plotly Dash

- Perform predictive analysis using classification models

    - The accuracy performance of Machine Learning models is adjusted, tweaked, and assessed.

Data Collection

- Datasets were gathered using API calls to the SpaceX API. In addition, the BeautifulSoup library was utilized to scrape relevant web data.

- JSON-formatted data was standardized for API calls. HTML parsing was used for BeautifulSoup.

- The data was arranged in two-dimensional form.

- Data was cleaned and became ready for analysis.

Data Collection – SpaceX API

- I also used a get request to access the SpaceX API to collect the necessary data. I conducted some basic data cleaning and formatting, as well as data wrangling, after obtaining the data to verify that it was ready for analysis.

Data Collection - Scraping

- To collect Falcon 9 launch data, I used Beautiful Soup web scraping. We parsed the table data and transformed it into a pandas data frame, which we then used for additional research.

```
In [92]:  static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
In [93]:  # use requests.get() method with the provided static_url
          response = requests.get(static_url)
          # assign the response to a object
```

Create a `BeautifulSoup` object from the HTML `response`

```
In [94]:  # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
          soup = BeautifulSoup(response.content,'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```
In [95]:  # Use soup.title attribute
          soup.title.string
```
```
Out[95]:  'List of Falcon 9 and Falcon Heavy launches - Wikipedia'
```

Data Wrangling

- The proper training labels were chosen by exploratory data analysis. The number of launches at each location, as well as the frequency and occurrence of each orbit, were estimated.
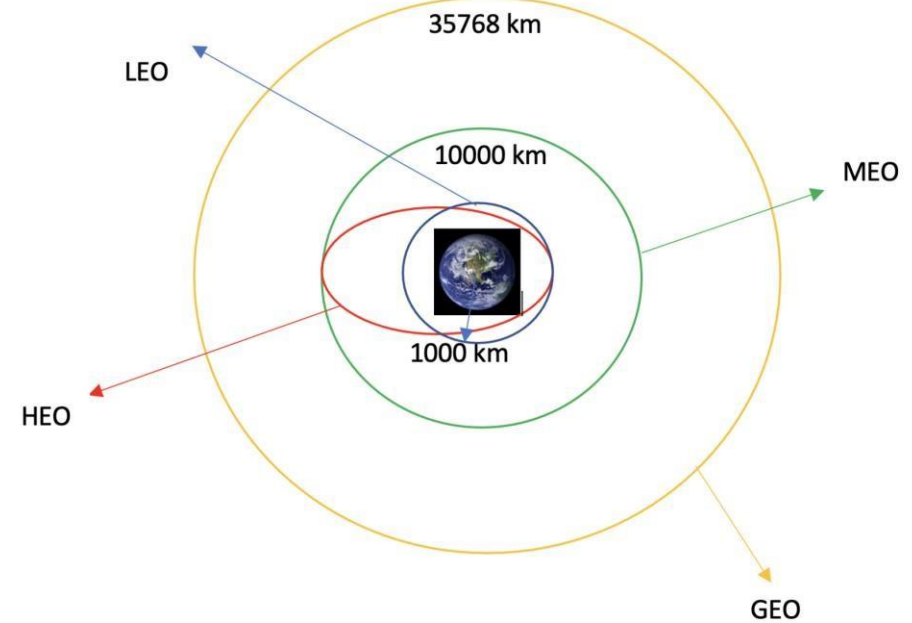
EDA with Data Visualization

- We looked at the relationship between launch sites and other elements. We wanted to discover if there was any correlation between launch success and failure.

- We also looked at the success rate by year.
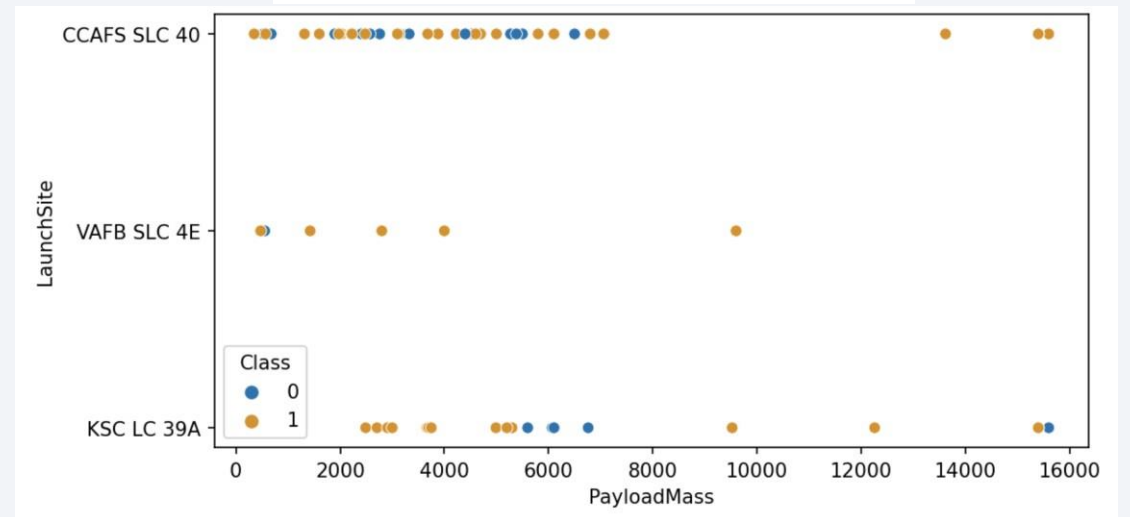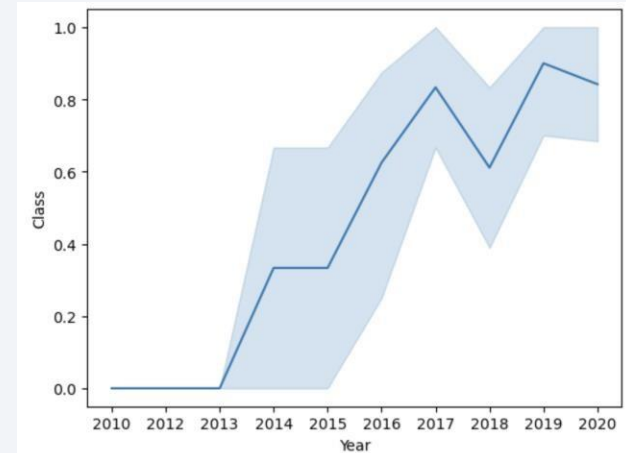
EDA with SQL

- We used SQL queries to do exploratory data analysis on the data in order to learn more about insight.

- We extracted information such as the unique names of launch sites used in space missions, the total payload mass carried by boosters launched by NASA's CRS program, the average payload mass carried by the F9 v1.1 booster version, and the total number of successful and failed mission outcomes.

Build an Interactive Map with Folium

- The Interactive now includes markers, circles, and Folium Map.

- These new capabilities enabled us to track the success rate of launches..

- The accessibility of launch sites to beaches, trains, and highways was explored.

Build a Dashboard with Plotly Dash

- Plotty

- We used a scatterplot to investigate the relationship between Outcome and Payload Mass (kg) in terms of different booster versions.

- We added pie charts to show the overall number of launches each launch location.

Predictive Analysis (Classification)

- For predictive analysis, we utilized the Scikit-Learn Library.

- We divided the dataset into train-test splits. Then we experimented with various predictive models.

- The confusion matrix was then shown, and the accuracy per model was computed.

- The most accurate model was then chosen.

Results

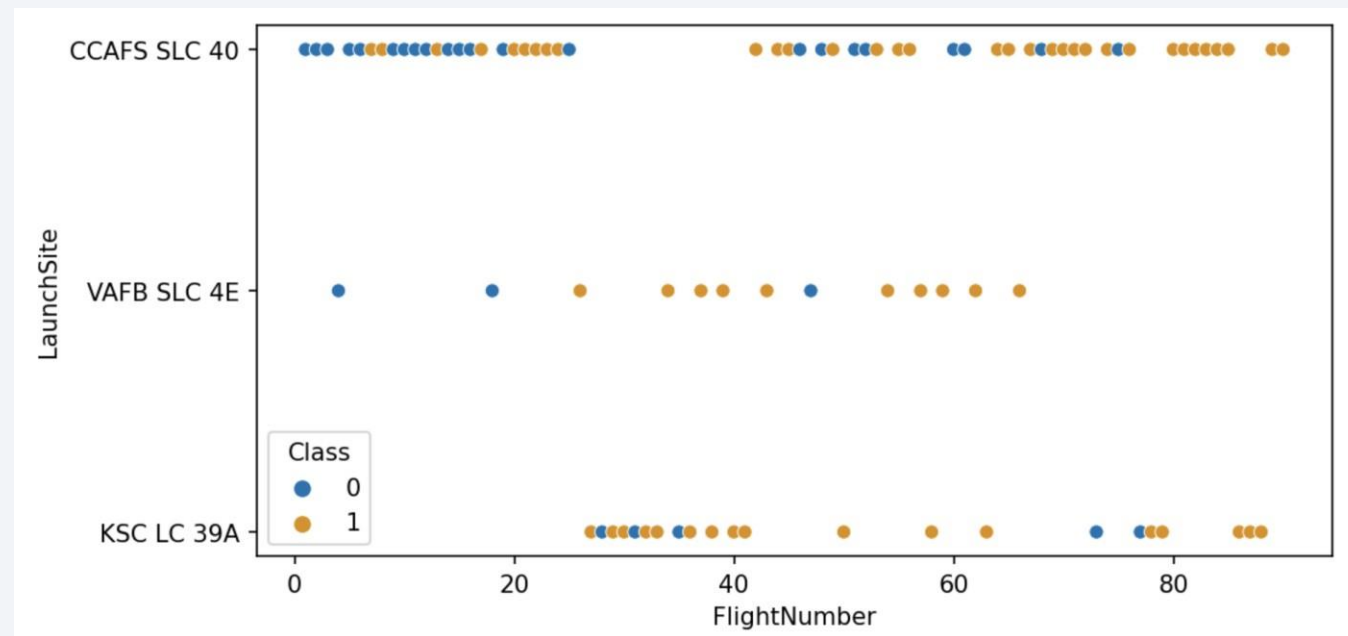- After cleaning and arranging the data, we discovered that the Decision Tree model is the best fit for our data.

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- Based on the plot, it can be observed that as the number of flights increases at a given launch site, the likelihood of success also tends to increase.

Payload vs. Launch Site

- A correlation can be observed between the success rate of the rocket at launch site CCAFS SLC 40 and the mass of its payload, where a higher payload mass is associated with a higher success rate.
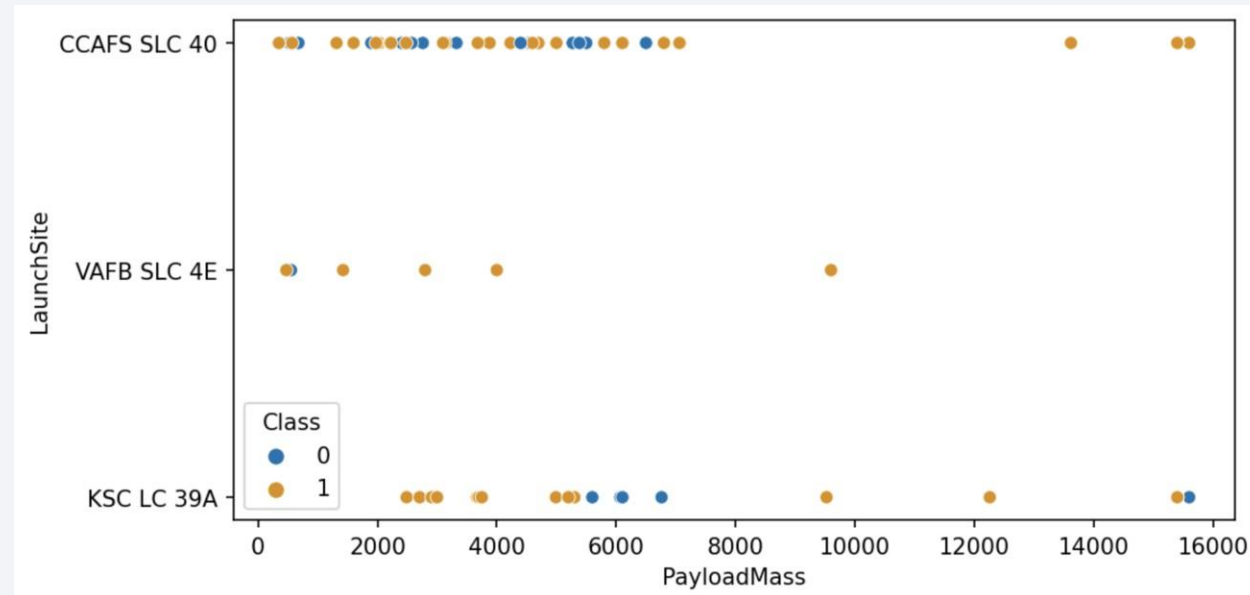
Success Rate vs. Orbit Type

- Based on the plot, it can be inferred that LEO, ISS, PO, GTO, and ES-L1 exhibited the highest success rates.
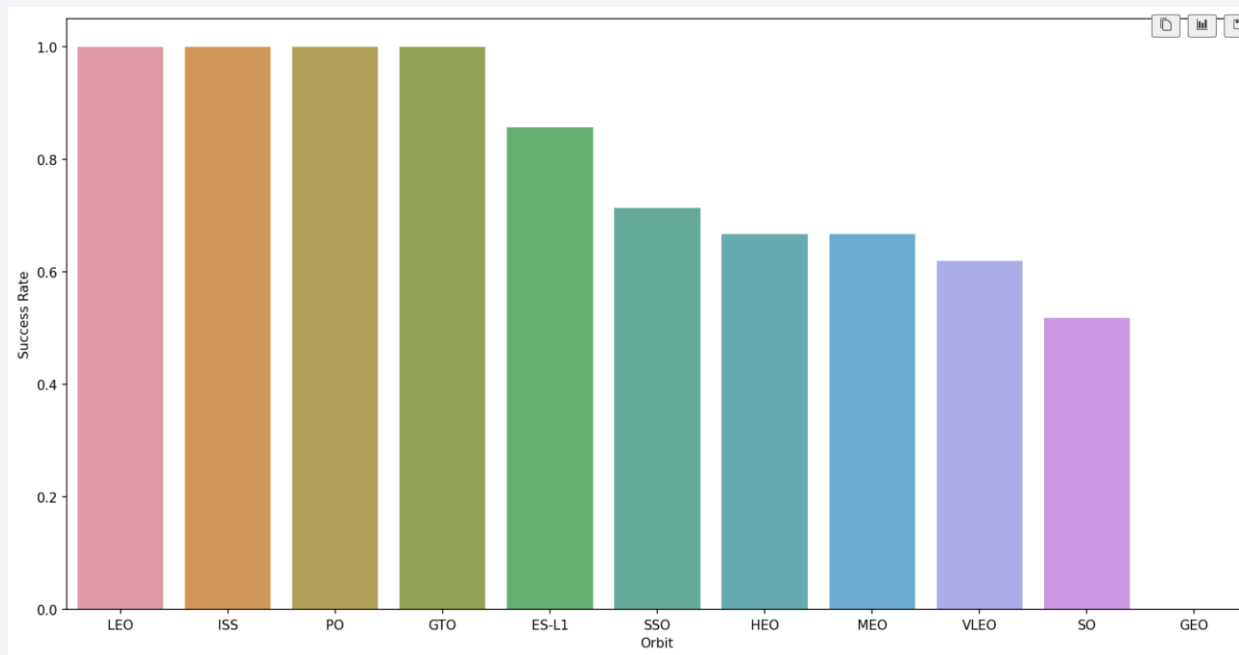
Flight Number vs. Orbit Type

- The following plot illustrates the correlation between Flight Number and Orbit type. It can be observed that in the LEO orbit, the success rate appears to be linked to the number of flights, while in the GTO orbit, there seems to be no such association between the success rate and flight number.
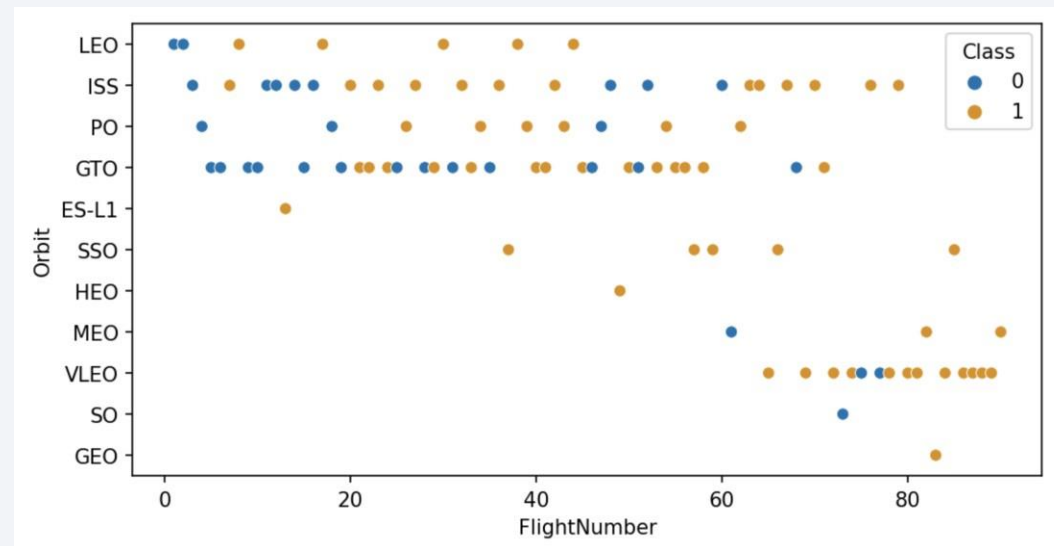
Payload vs. Orbit Type

- It can be observed that for orbits such as PO, LEO, and ISS, a higher success rate in terms of landing is associated with heavy payloads.
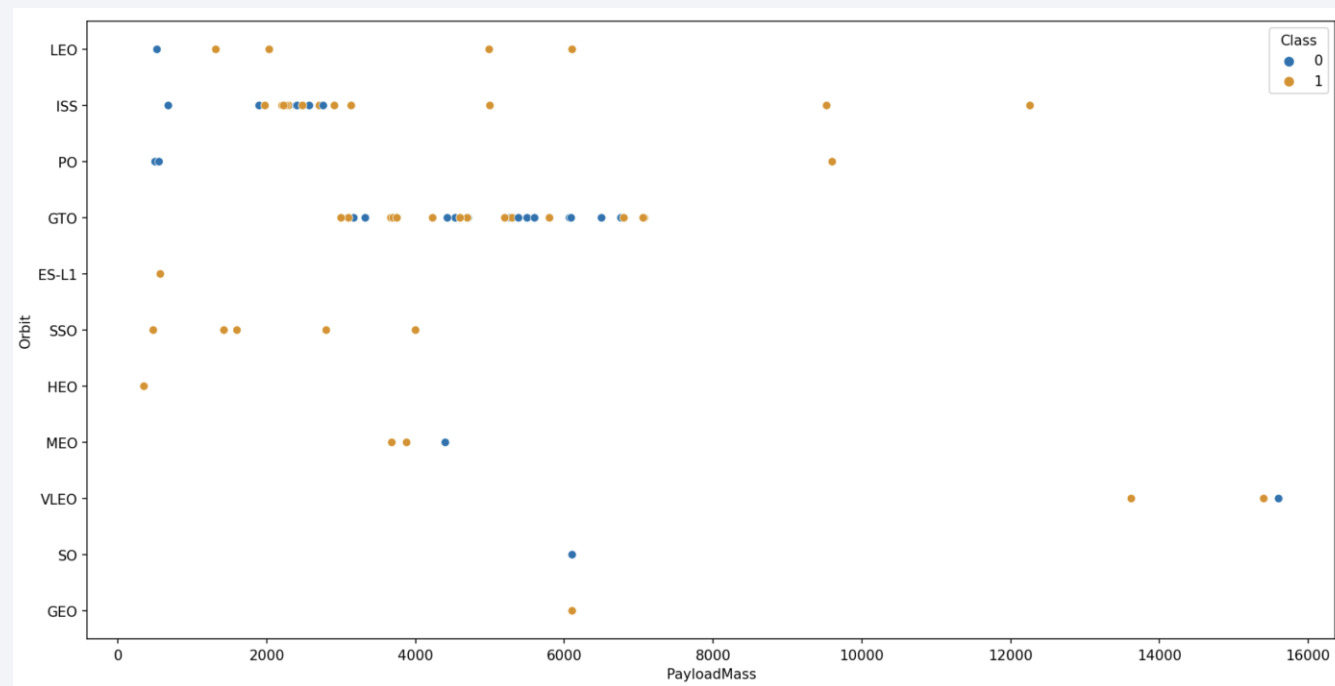
Launch Success Yearly Trend

- Based on the plot, it can be observed that the success rate has consistently increased from 2013 to 2020.
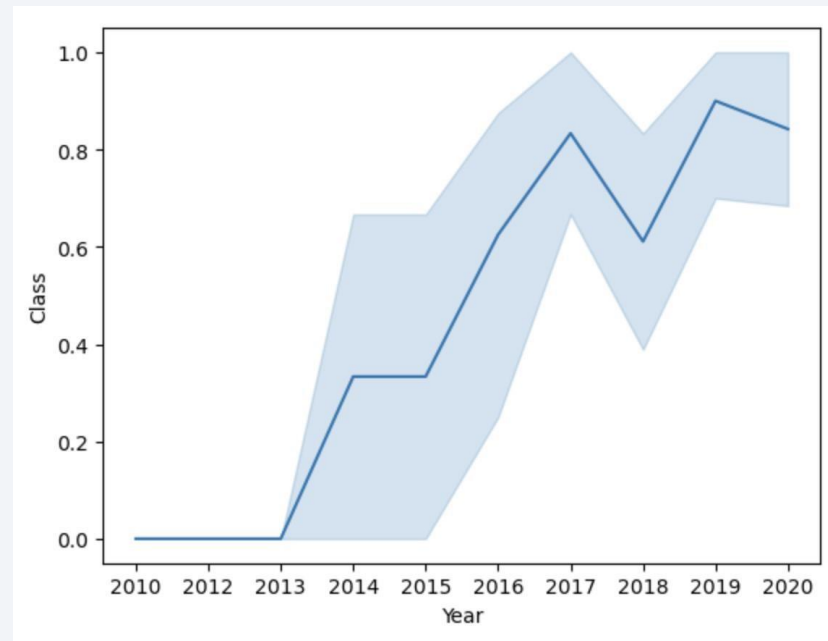
All Launch Site Names

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

```
%sql SELECT DISTINCT Launch_Site FROM SPACEXTBL;
```

 * sqlite:///my_data1.db
Done.

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

Launch Site Names Begin with 'CCA'

- The query was utilized to retrieve 5 records from a dataset where the launch sites begin with 'CCA'.

Total Payload Mass

- By executing the query provided below, we were able to calculate that the total payload carried by NASA's boosters is 45596.

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS Total_Payload_Mass FROM SPACEXTBL WHERE Customer LIKE 'NASA (CRS)';

 * sqlite:///my_data1.db
Done.

Total_Payload_Mass
              45596
```

Average Payload Mass by F9 v1.1

- By performing the following query, it was determined that the average payload mass carried by booster version F9 v1.1 is 2928.4.


```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS Average_Payload_Mass FROM SPACEXTBL WHERE Booster_Version LIKE 'F9 v1.1'
```

 * sqlite:///my_data1.db
Done.

Average_Payload_Mass
2928.4

First Successful Ground Landing Date

- Based on our observations, the date on which the first successful landing occurred on a ground pad was December 22nd, 2015.

```
%sql SELECT MIN(Date) AS Min_Date FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success (ground pad)';
✓ 0.0s

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

Min_Date
22-12-2015

Successful Drone Ship Landing with Payload between 4000 and 6000

- We deployed the WHERE clause to narrow down the dataset to boosters that had accomplished a successful landing on a drone ship. Subsequently, we applied an AND condition to further refine the results and only include boosters that had a payload mass greater than 4000 but less than 6000 at the time of the successful landing.

```
%sql SELECT * FROM SPACEXTBL WHERE Landing_Outcome LIKE '%Success (drone ship)%' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
✓ 0.0s
* sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
06-05-2016	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
14-08-2016	05:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
30-03-2017	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
11-10-2017	22:53:00	F9 FT B1031.2	KSC LC-39A	SES-11 / EchoStar 105	5200	GTO	SES EchoStar	Success	Success (drone ship)

Total Number of Successful and Failure Mission Outcomes

- UNION method is used to merge both results in the same column. It can be observed that total number of:

  - Success outcome is 100

  - Failure outcome is 1

```
%sql SELECT COUNT(*) FROM SPACEXTBL WHERE Mission_Outcome LIKE '%Success%' UNION SELECT COUNT(*) Failure FROM SPACEXTBL WHERE Mission_Outcome LIKE '%Failure%';
```
✓ 0.0s   Python

* sqlite:///my_data1.db
Done.

COUNT(*)
1
100

Boosters Carried Maximum Payload

- Using a subquery within the WHERE clause and the MAX() function, we were able to identify the booster that carried the maximum payload.

```
%sql SELECT * FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```
✓ 0.0s   Python

* sqlite:///my_data1.db
Done.

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
11-11-2019	14:56:00	F9 B5 B1048.4	CCAFS SLC-40	Starlink 1 v1.0, SpaceX CRS-19	15600	LEO	SpaceX	Success	Success
07-01-2020	02:33:00	F9 B5 B1049.4	CCAFS SLC-40	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600	LEO	SpaceX	Success	Success
29-01-2020	14:07:00	F9 B5 B1051.3	CCAFS SLC-40	Starlink 3 v1.0, Starlink 4 v1.0	15600	LEO	SpaceX	Success	Success
17-02-2020	15:05:00	F9 B5 B1056.4	CCAFS SLC-40	Starlink 4 v1.0, SpaceX CRS-20	15600	LEO	SpaceX	Success	Failure
18-03-2020	12:16:00	F9 B5 B1048.5	KSC LC-39A	Starlink 5 v1.0, Starlink 6 v1.0	15600	LEO	SpaceX	Success	Failure
22-04-2020	19:30:00	F9 B5 B1051.4	KSC LC-39A	Starlink 6 v1.0, Crew Dragon Demo-2	15600	LEO	SpaceX	Success	Success
04-06-2020	01:25:00	F9 B5 B1049.5	CCAFS SLC-40	Starlink 7 v1.0, Starlink 8 v1.0	15600	LEO	SpaceX, Planet Labs	Success	Success
03-09-2020	12:46:14	F9 B5 B1060.2	KSC LC-39A	Starlink 11 v1.0, Starlink 12 v1.0	15600	LEO	SpaceX	Success	Success
06-10-2020	11:29:34	F9 B5 B1058.3	KSC LC-39A	Starlink 12 v1.0, Starlink 13 v1.0	15600	LEO	SpaceX	Success	Success
18-10-2020	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600	LEO	SpaceX	Success	Success
24-10-2020	15:31:34	F9 B5 B1060.3	CCAFS SLC-40	Starlink 14 v1.0, GPS III-04	15600	LEO	SpaceX	Success	Success
25-11-2020	02:13:00	F9 B5 B1049.7	CCAFS SLC-40	Starlink 15 v1.0, SpaceX CRS-21	15600	LEO	SpaceX	Success	Success

31

2015 Launch Records

- To filter the dataset for failed landing outcomes on a drone ship, along with their corresponding booster versions and launch site names for the year 2015, we utilized a combination of the WHERE clause, LIKE operator, AND condition.

```
%sql SELECT substr(Date,4,2) AS Month, Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTBL
WHERE Landing_Outcome LIKE '%Failure (drone ship)%' AND substr(Date,7,4) = '2015';
✓ 0.0s

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

Month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- We selected Landing outcomes and the COUNT of landing outcomes. We then used the WHERE clause to filter the data for landing outcomes that fell between the dates 2010-06-04 to 2010-03-20.

- Next, we applied the GROUP BY clause to group the landing outcomes and used the ORDER BY clause to arrange the grouped landing outcomes in descending order.

```
%sql SELECT Landing_Outcome,COUNT(*) FROM SPACEXTBL WHERE Date between '04-06-2010' and '20-03-2017' GROUP BY Landing_Outcome ORDER BY COUNT(*) DESC;
 ✓ 0.0s
 * sqlite:///my_data1.db
Done.
```

| Landing_Outcome	COUNT(*)
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

Section 3

Launch Sites Proximities Analysis

Launch Sites of SpaceX

- As can be seen from the map below, all launch sites of SpaceX are in the United States, mostly around two locations.
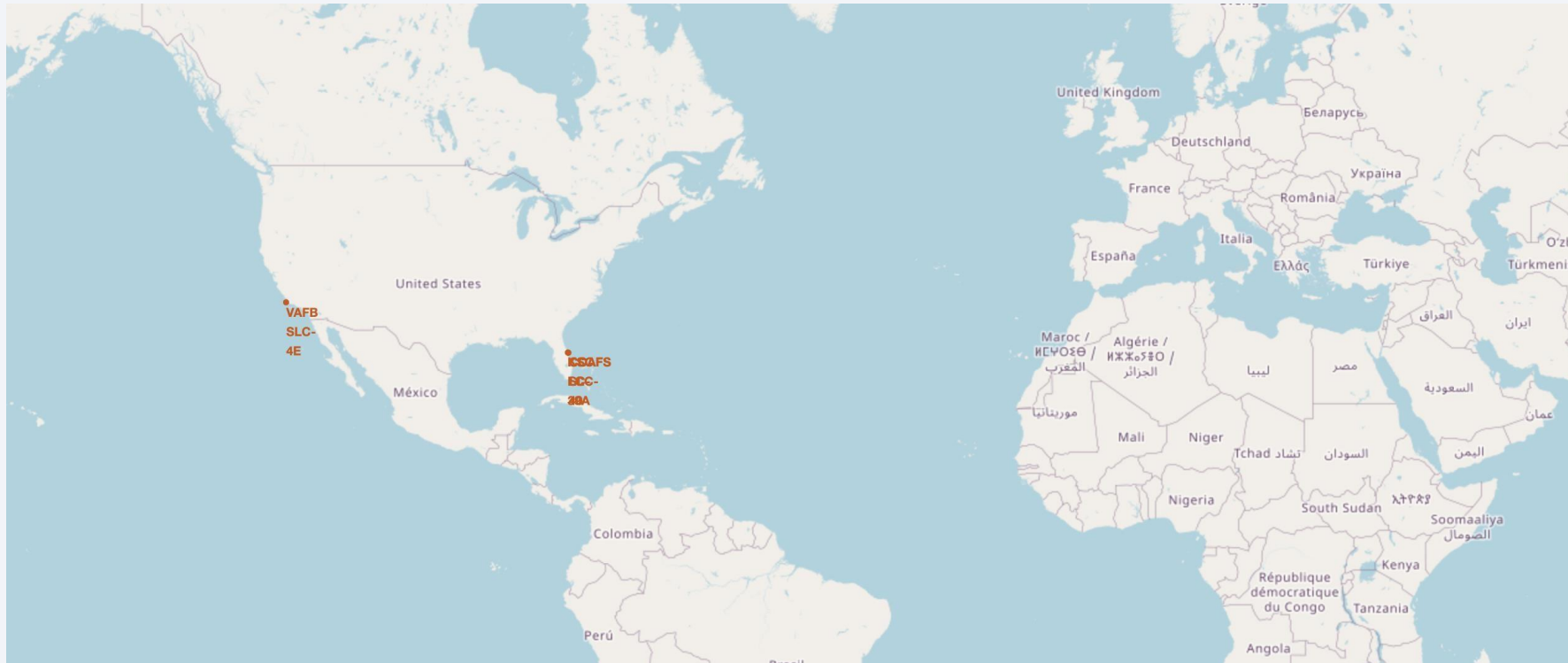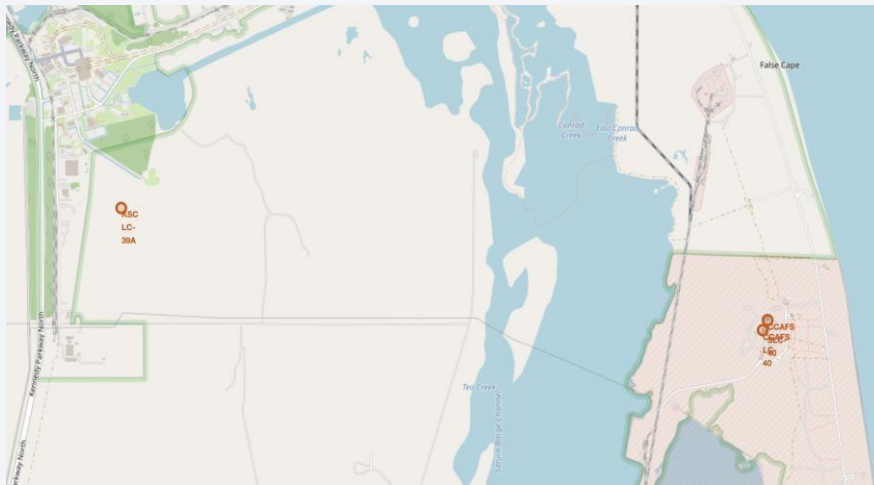
Launch Sites with Labels
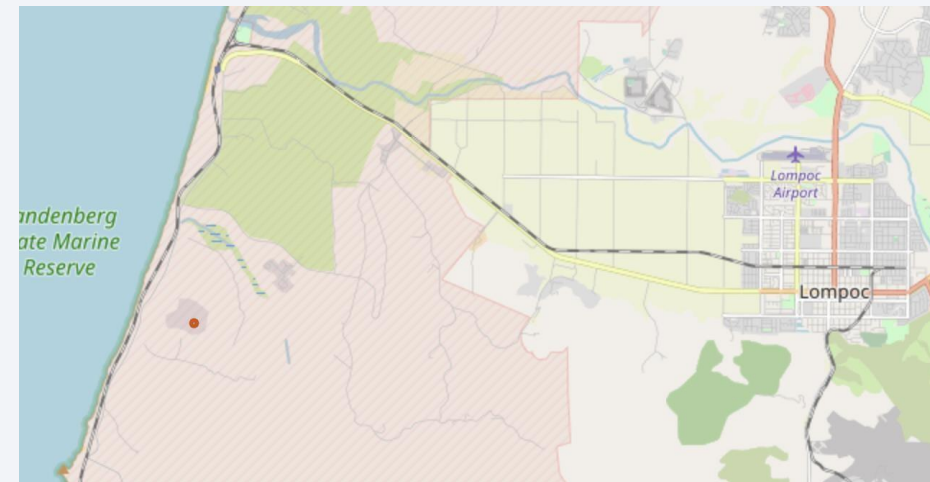
- Green is for success

- Red is for failure.
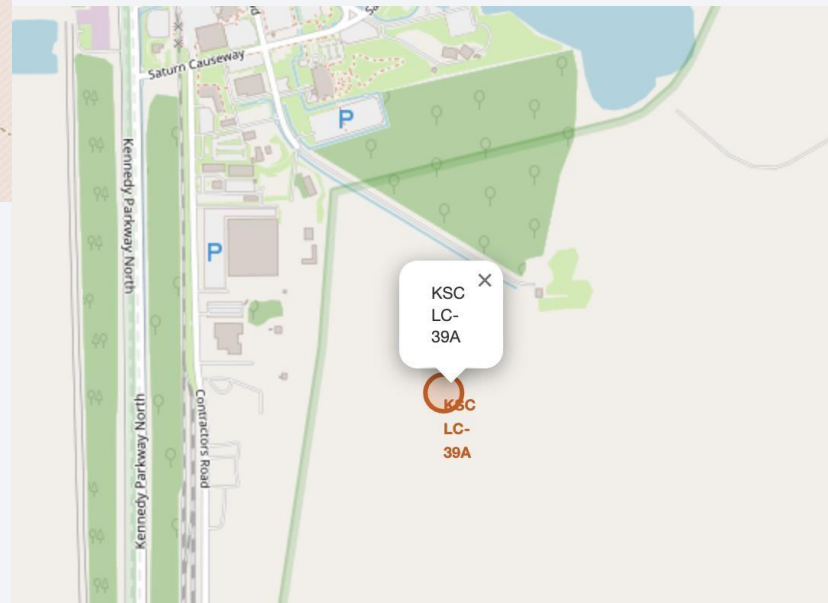
Launch Site Distance to Landmarks

•It can be observed that eastern launch sites are quite close to landmarks meanwhile western one is distant.
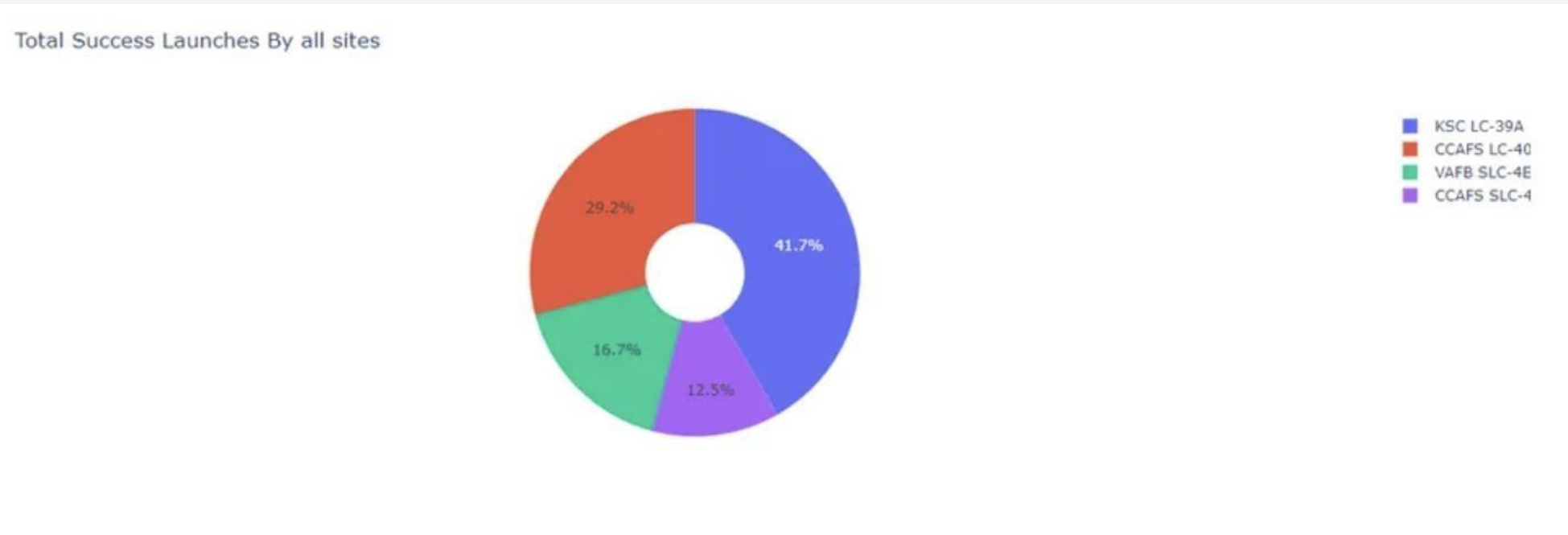
Eastern

Western

Section 4

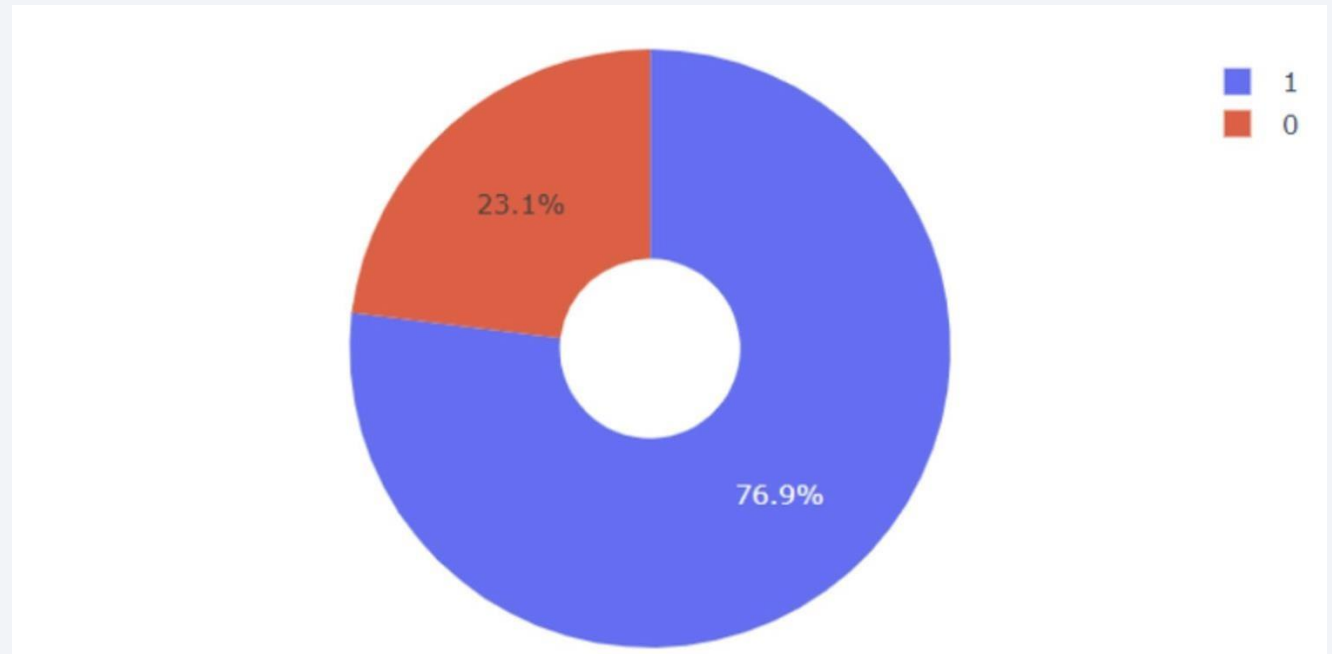**Build a Dashboard
with Plotly Dash**

Total Success Launches by All Sites

- As can be seen below, KSC LC-39A is the most successful launch of all launches.

Total Success Launches By all sites

KSC LC-39A: 41.7%
CCAFS LC-40: 29.2%
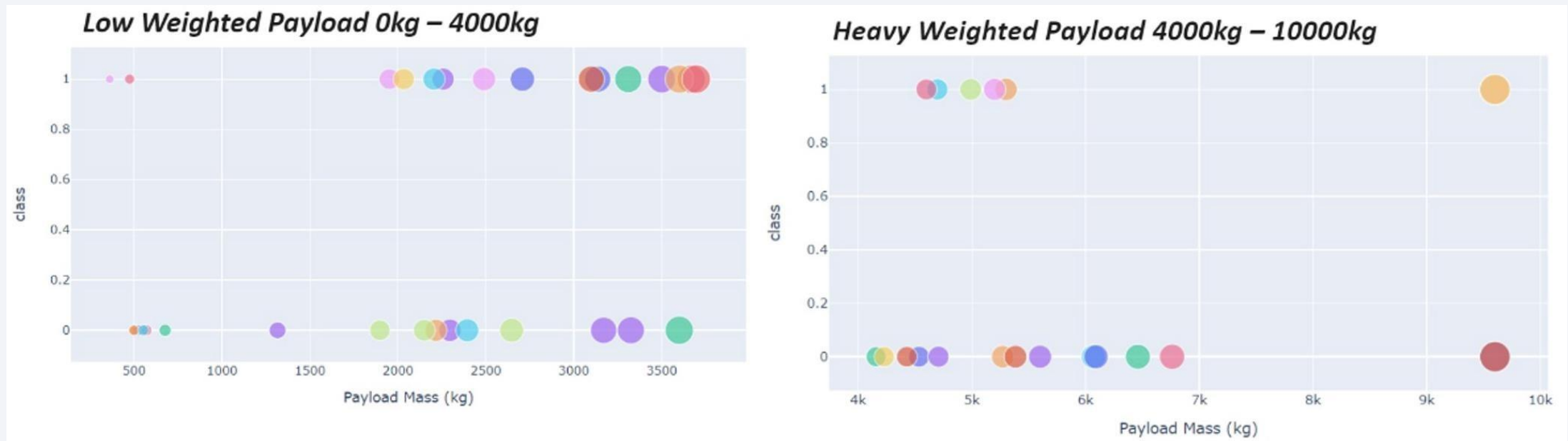VAFB SLC-4E: 16.7%
CCAFS SLC-4: 12.5%

KSC LC-39A Success Rate

- As can be seen from the image, success rate of KSC LC-39A is 76.9%.

Relationship between Payload and Launch Outcome across all launch sites

- One could observe that the success rates are higher for payloads with lower weight compared to those with heavier weight.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The decision tree classifier is considered to have the highest classification accuracy among the models used.

```python
methods = {'KNN':knn_cv.best_score_,
           'DecisionTree':tree_cv.best_score_,
           'LogisticRegression':logreg_cv.best_score_,
           'SVM':svm_cv.best_score_}


Best_Method = max(algorithms,key=algorithms.get)
Best_Method
```

```
'DecisionTree'
```

```python
tree_cv = GridSearchCV(estimator=tree, param_grid=parameters, cv=10)
tree_cv.fit(X_train,Y_train)
```

```
GridSearchCV(cv=10, estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [2, 4, 6, 8, 10, 12, 14, 16, 18],
                         'max_features': ['auto', 'sqrt'],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10],
                         'splitter': ['best', 'random']})
```

```python
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 6
accuracy : 0.9035714285714287
```

Confusion Matrix

- Based on the confusion matrix of the decision tree classifier, it can be inferred that the classifier is capable of accurately distinguishing between different classes. However, the major issue appears to be the occurrence of false positives, which implies that the classifier sometimes labels unsuccessful landings as successful ones.
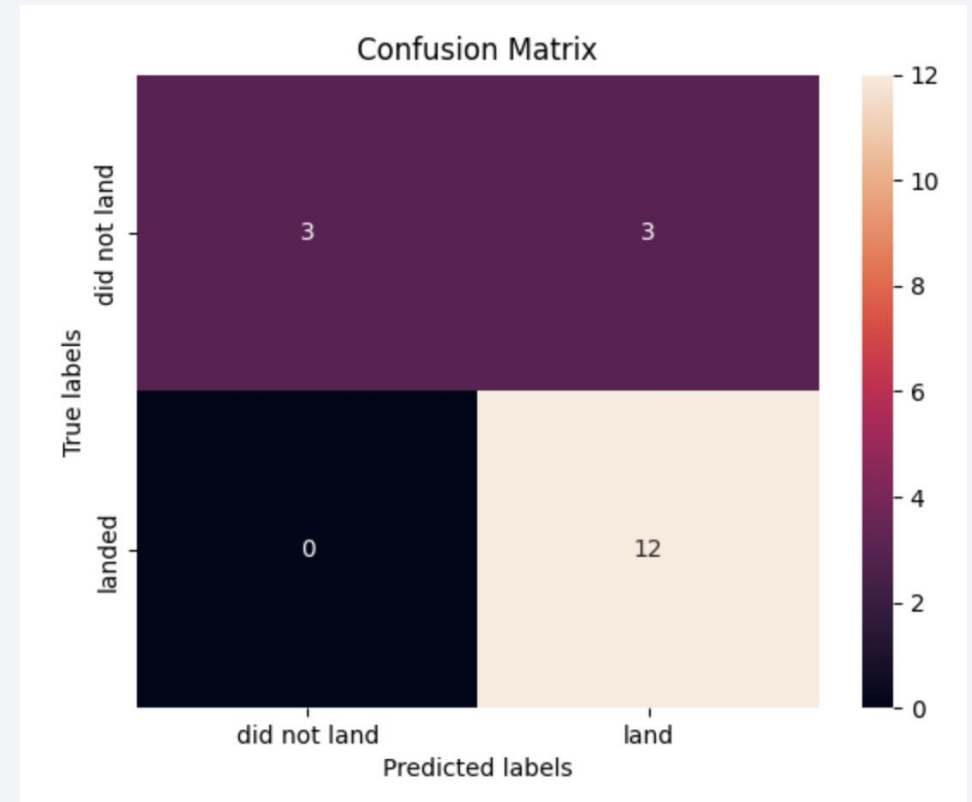
Conclusions

Based on the analysis, we can draw the following conclusions:

- There is a positive correlation between the flight amount at a launch site and the success rate at the site.

- Launch success rate has been increasing steadily from 2013 to 2020.

- KSC LC-39A is the launch site with the highest number of successful launches.

- The orbits ES-L1, GEO, HEO, SSO, VLEO have the highest success rates.

- The decision tree classifier is the most suitable machine learning algorithm for this task based on its performance.

Overall, these findings provide useful insights into the factors that contribute to launch success and can be used to improve future launches.

Thank you!