IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Seyed Javad Miraftabzadeh
28th Sep 2022

Outline

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 announces the launch of the Falcon 9 rocket on its website at 62 million dollars. Other providers charge more than $165 million per launch. The major part of SpaceX's savings is because Space X can reuse the first stage. So, if we can determine if the first stage landed correctly, we can calculate the cost of the launch. We can use this information in the rival company against SpaceX to launch rockets. The goal of this project is to create a machine learning pipeline to predict the success of the first stage.

- Problems you want to find answers
  - What factors determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - One-hot encoding was applied to 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
  - We use these models: logistic regression, support vector machine, decision tree classifier, k nearest neighbors and measure the accuracy of theme at the end

Data Collection

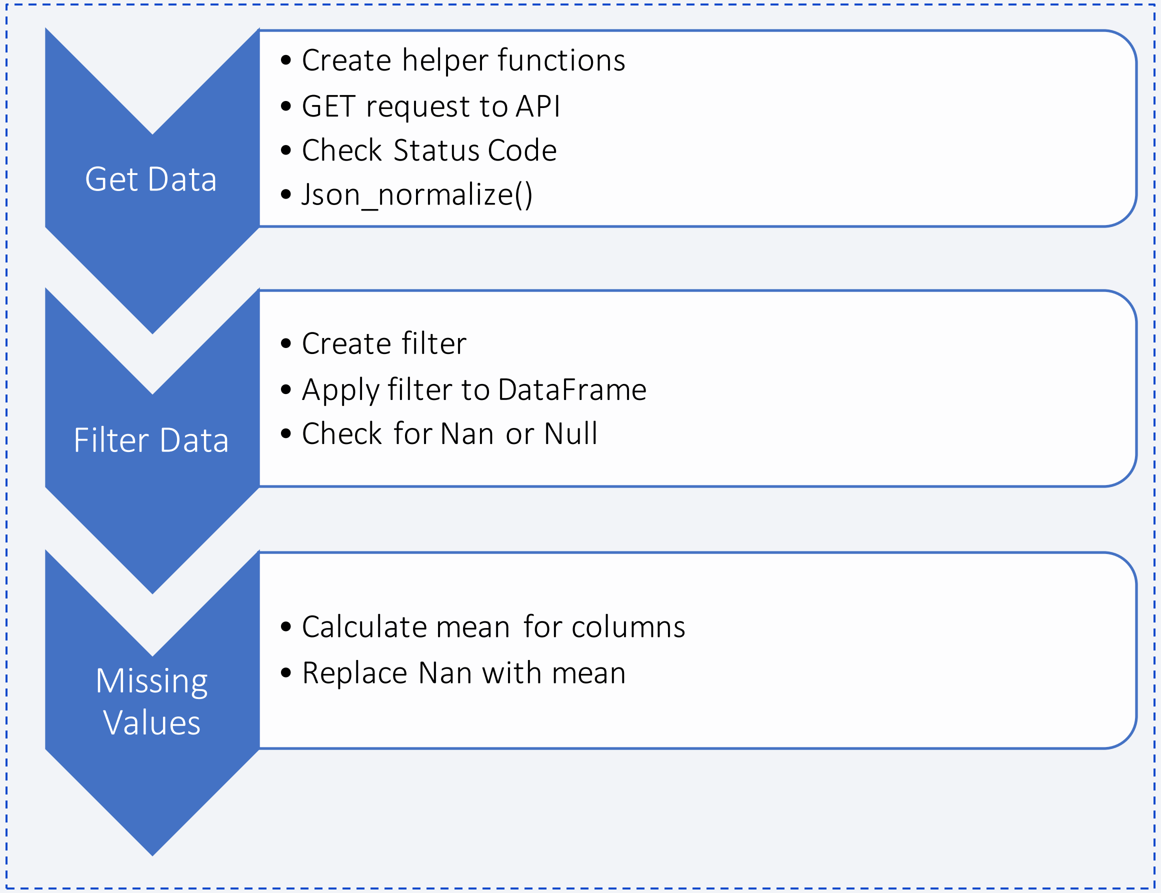
- The data was collected in various methods
  - Data collection was done using get request to the SpaceX API.
  - We decoded the response content as a Json using **`.json()`** function call and turn it into a pandas **`DataFrame`** using **`.json_normalize()`**.
  - We then cleaned the data, and filter theme to only include Falcon 9 data, and dealing for missing values and fill in missing values where necessary.
  - After these we save the data in csv format.
  - In second step, we performed web scraping from Wikipedia for Falcon 9 launch records with **`BeautifulSoup`**.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas **`DataFrame`** for future analysis.
  - After these we save the data in csv format.

Data Collection – SpaceX API

1. Request and parse the SpaceX launch data using the GET request
2. Filter the DataFrame to only include Falcon 9 launches
3. Dealing with Missing Values

- GitHub URL:

  https://github.com/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/01_jupyter-labs-spacex-data-collection-api.ipynb

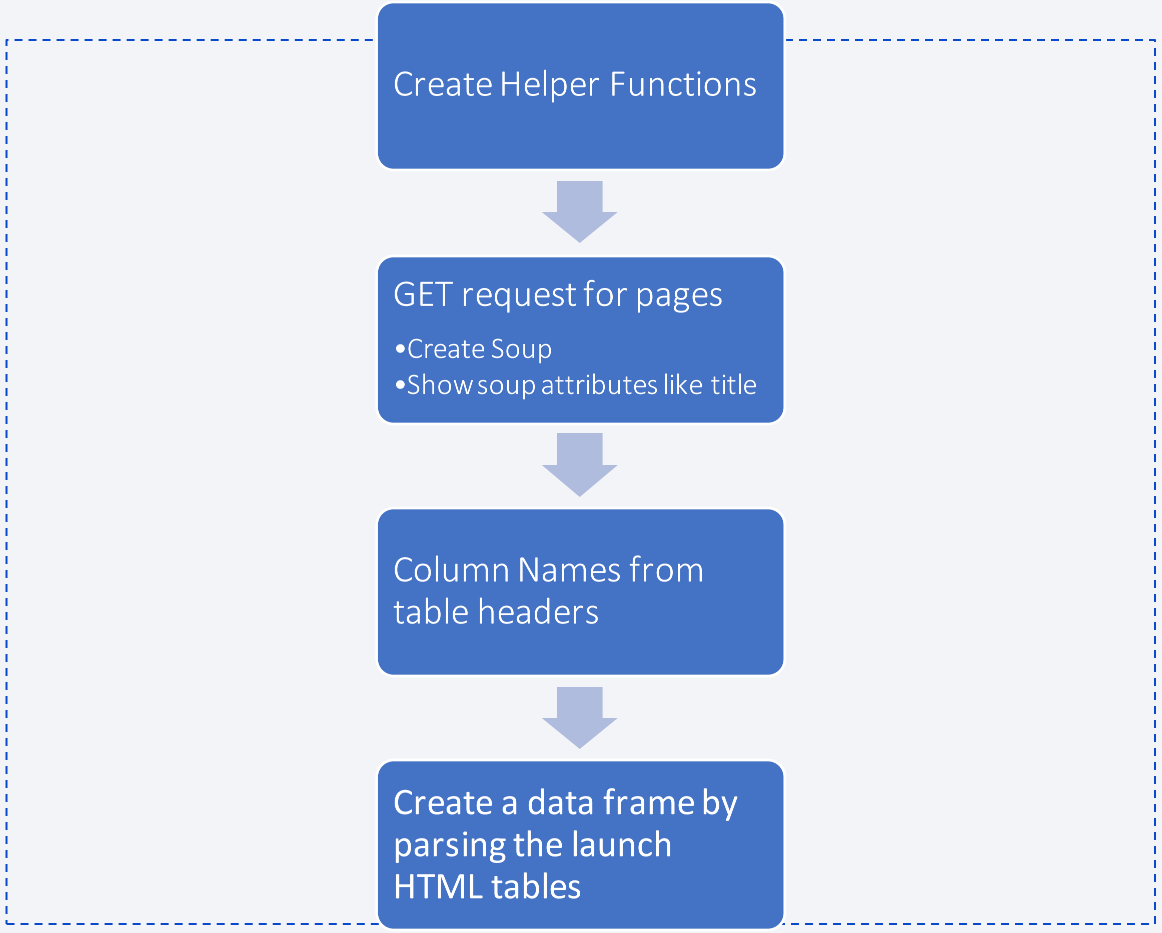


Data Collection - Scraping

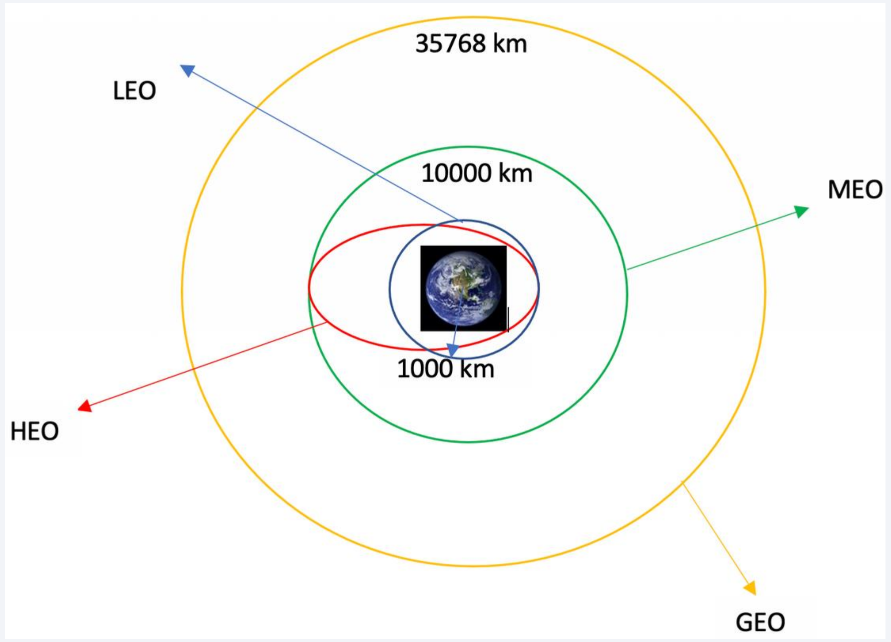
1. Request the Falcon9 Launch Wiki page
2. Create DataFrame by Parsing data with Beautifull Soup

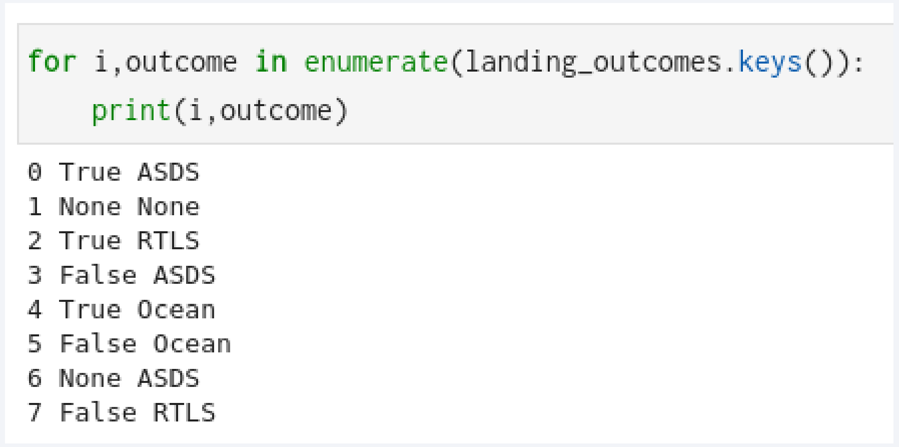
- GitHub URL:

  https://github.com/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/02_jupyter-labs-webscraping.ipynb


Create Helper Functions
GET request for pages
•Create Soup
•Show soup attributes like title
Column Names from table headers
Create a data frame by parsing the launch HTML tables

Data Wrangling





```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

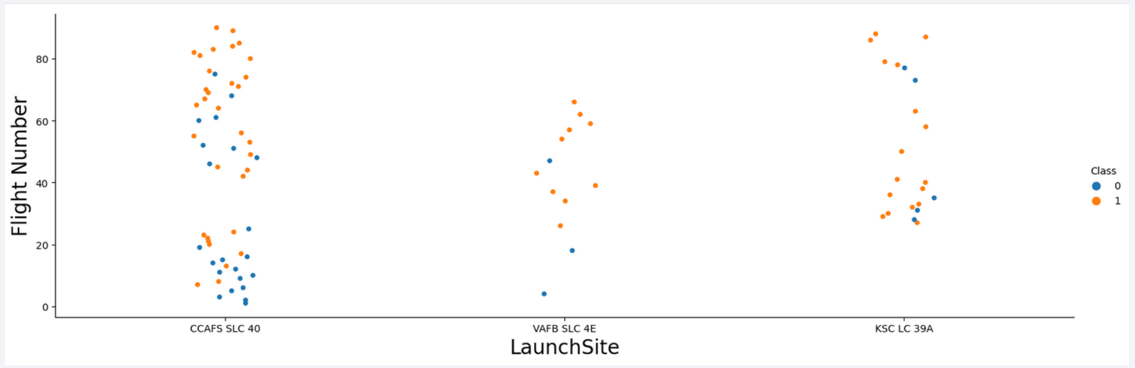
- Calculate the number of launches on each site
- Calculate the number and occurrence of each orbit
- Calculate the number and occurrence of mission outcome per orbit type
- Create a landing outcome label from Outcome column
- Save result in CSV file for later use.
- Github URL: https://github.com/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/03_labs-jupyter-spacex-Data%20wrangling.ipynb

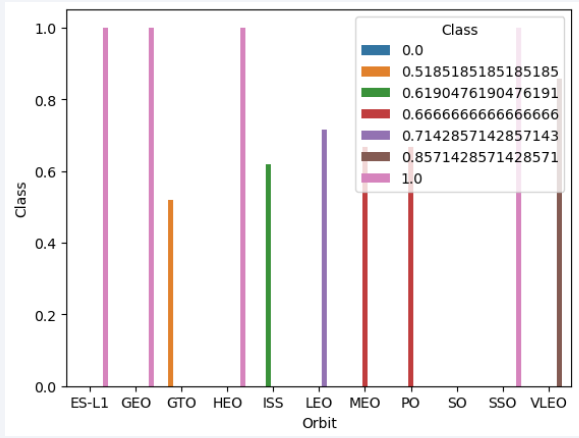
EDA with SQL

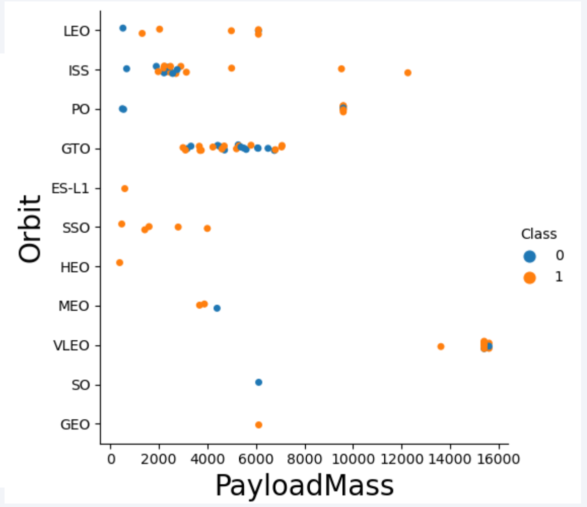
- Load data in SQL DB and execute following queries
  - Display the names of the unique launch sites
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version 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
  - List the records which will display the month names, failure landing ... for the months in year 2015.
  - Rank the count of successful landing between the date 04-06-2010 and 20-03-2017 I
- GitHub URL: https://github.com/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/04_jupyter-labs-eda-sql-coursera_sqllite.ipynb

EDA with Data Visualization

- Exploratory Data Analysis
  - Scatter plot for Flight Number vs. Payload Mass, Flight Number vs Launch Site, Payload vs Launch Site, Flight Number vs Orbit type, Payload vs Orbit type
  - Bar chart for success rate of each orbit type
  - launch success yearly trend
- GitHub URL: https://github.com/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/05_jupyter-labs-eda-dataviz.ipynb







Build an Interactive Map with Folium

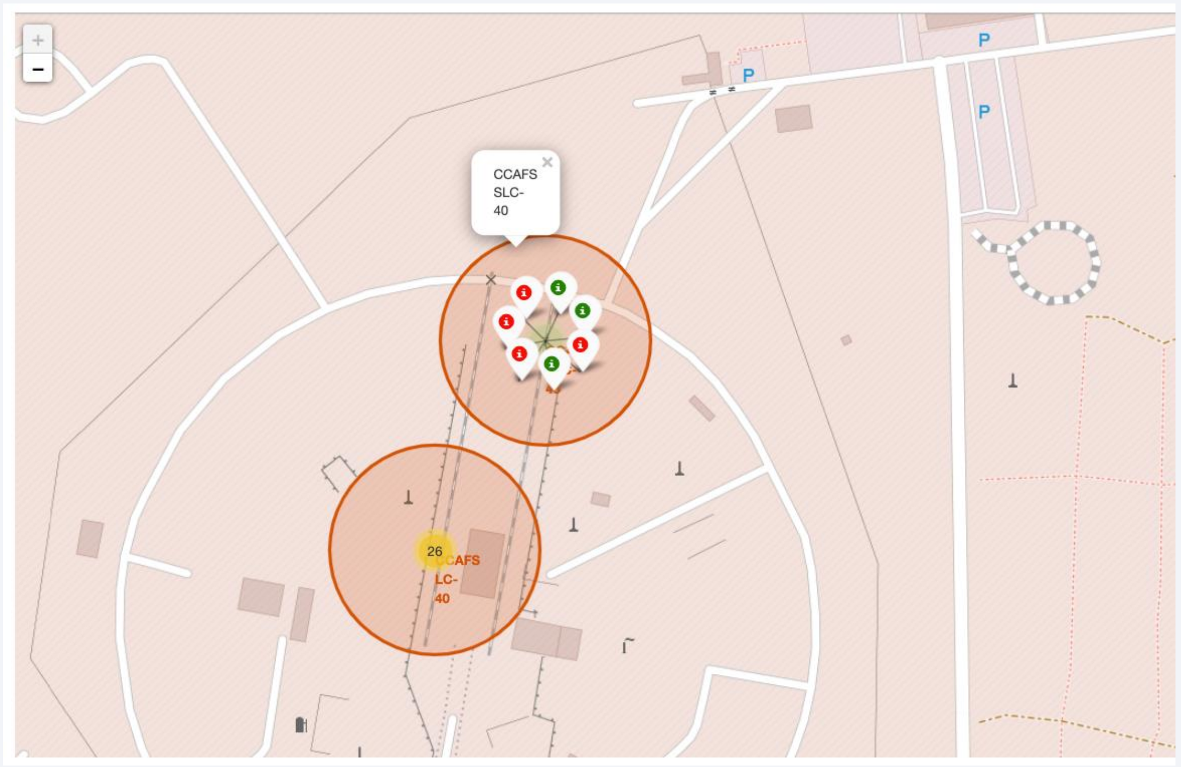
- We use Folium to create interactive maps and **mark all launch sites** on a map, **mark the success/failed launches for each site** on the map and **calculate the distances between a launch site to its proximities**

- GitHub URL:

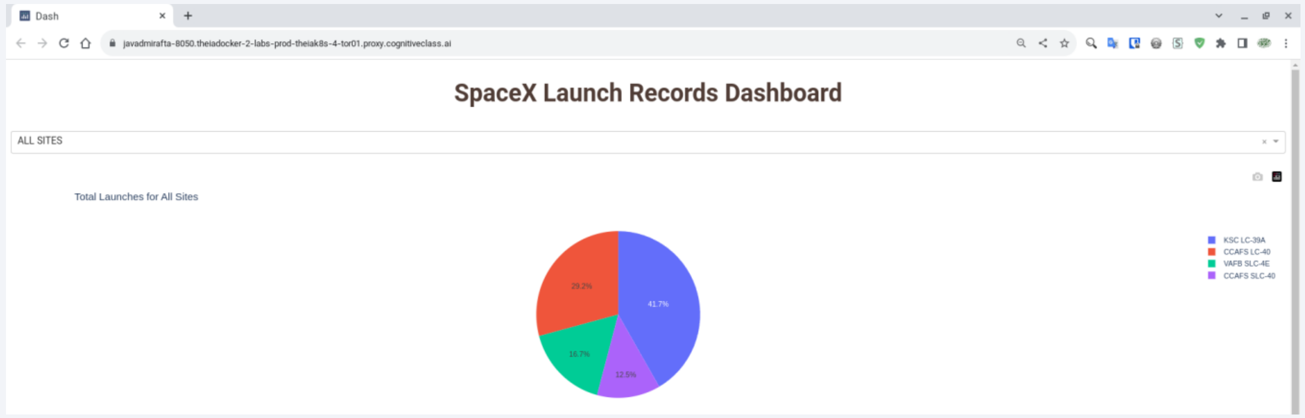
  https://github.com/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/06_lab_jupyter_launch_site_location.ipynb

  and

  https://nbviewer.org/github/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/06_lab_jupyter_launch_site_location.ipynb



Build a Dashboard with Plotly Dash


Dash
javadmirafta-8050.theiadocker-2-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai
SpaceX Launch Records Dashboard
ALL SITES
Total Launches for All Sites
41.7%
29.2%
16.7%
12.5%
KSC LC-39A
CCAFS LC-40
VAFB SLC-4E
CCAFS SLC-40

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- GitHub URL:

  https://github.com/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- We load data, create Pandas Data Frame and using NumPy to create, transform, standardize and split data sets
- Then we create different machine learning models to predict result:
  - logistic regression
  - support vector machine
  - decision tree classifier
  - k nearest neighbors
- And calculate the accuracy for each the model
- Finally we find the best performing classification tools
- GitHub URL:

  https://github.com/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/07_SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

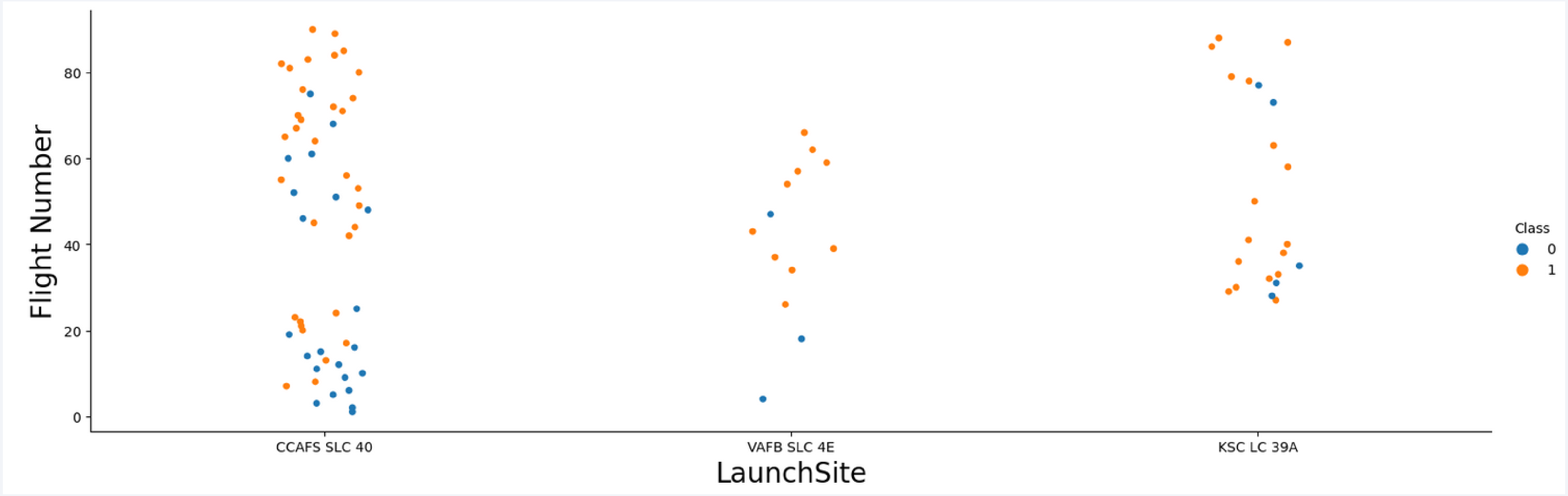
Results

- Logistic Regression
  - tuned hyperparameters :(best parameters) {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
- Support vector machine
  - tuned hyperparameters :(best parameters) {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
- Decision tree classifier
  - tuned hyperparameters :(best parameters) {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'best'}
- K nearest neighbors
  - tuned hyperparameters :(best parameters) {'algorithm': 'auto', 'n_neighbors': 9, 'p': 1}
- Accuracy for **Logistics Regression** method: 0.8333333333333334
- Accuracy for **Support Vector Machine** method: 0.8333333333333334
- Accuracy for **Decision tree method**: 0.8333333333333334
- Accuracy for **K nears neighbors** method: 0.8333333333333334

Section 2

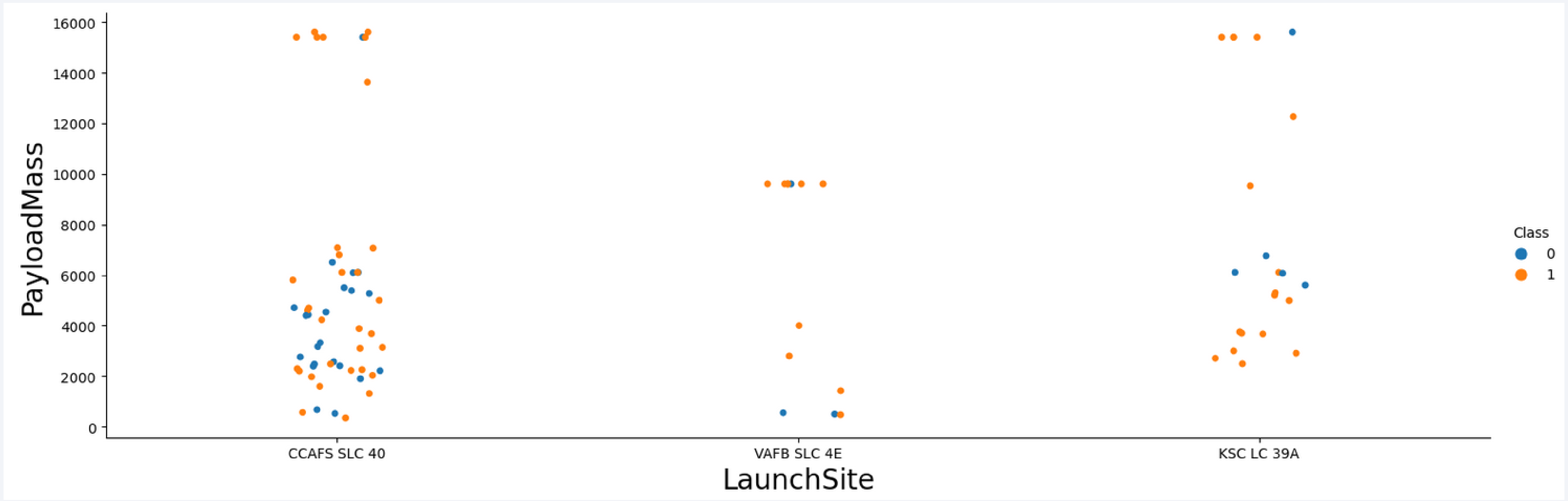
Insights drawn from EDA

Flight Number vs. Launch Site



- The plot shows that the greater flight number have a greater success chance and on the KSK LC 39A most of flight successes.

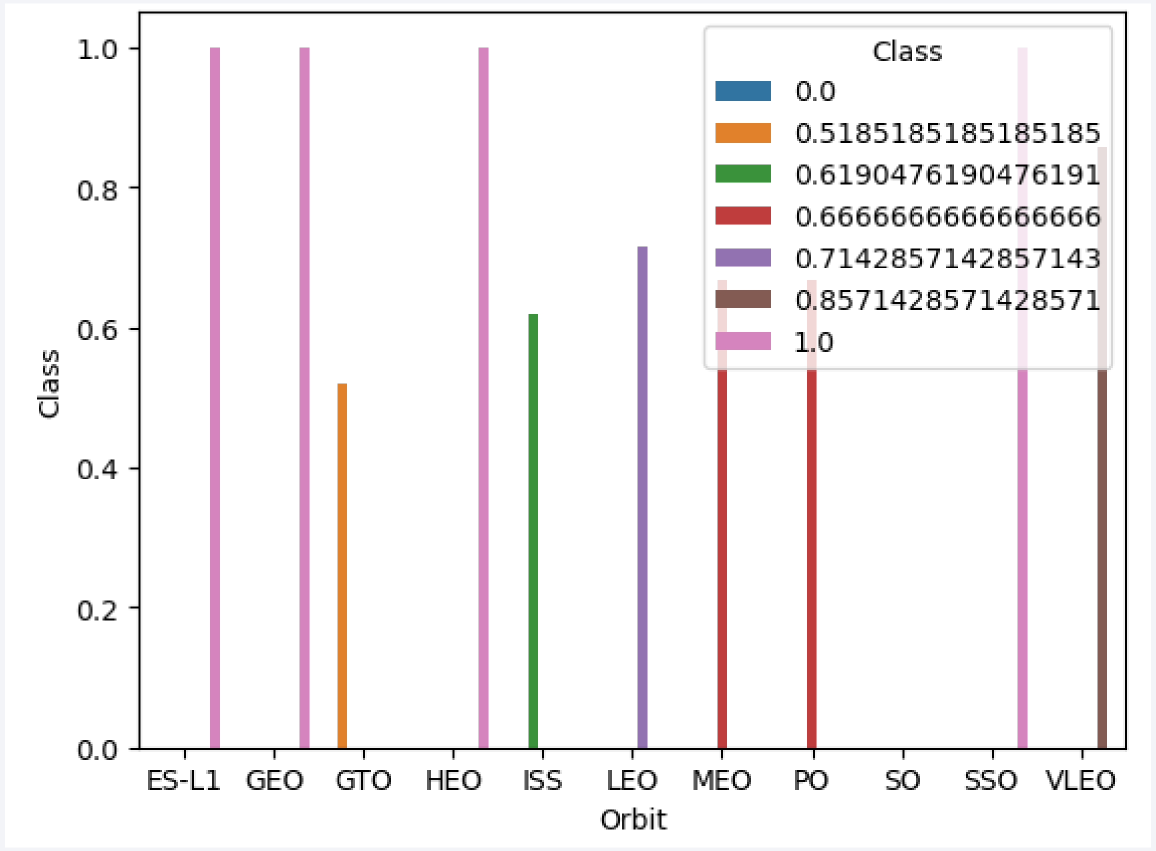
Payload vs. Launch Site



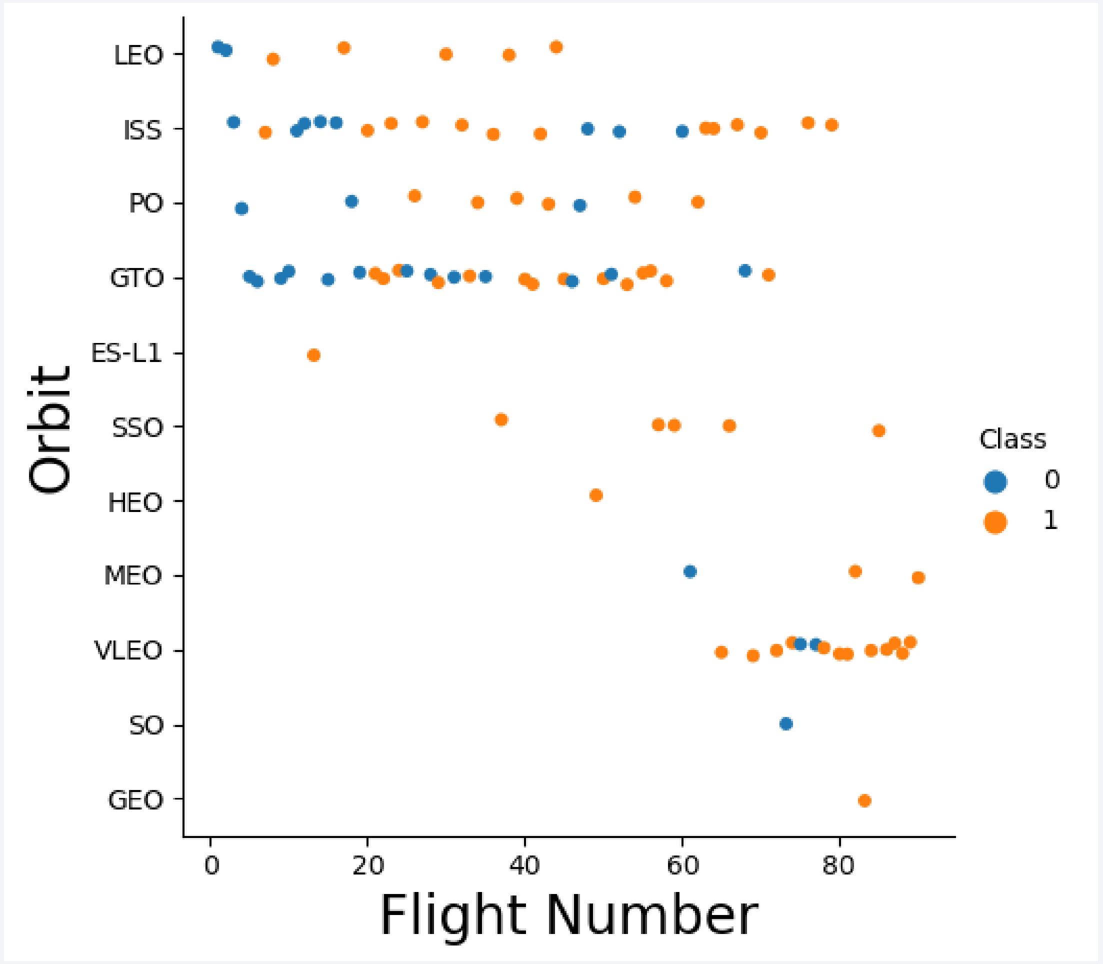
- On CCAFS SLC 40 and VAFB SLC 4E the greater Payload Mass have a better success chance
- On KSC LC 39A Payload less than 4000 ang greater than 8000 have a better success chance

Success Rate vs. Orbit Type

- From the plot the ES-L1, GEO, HEO, SSO had the most success rate.
- After them VLEO had most success rate
- Next success rate are for MEO and PO



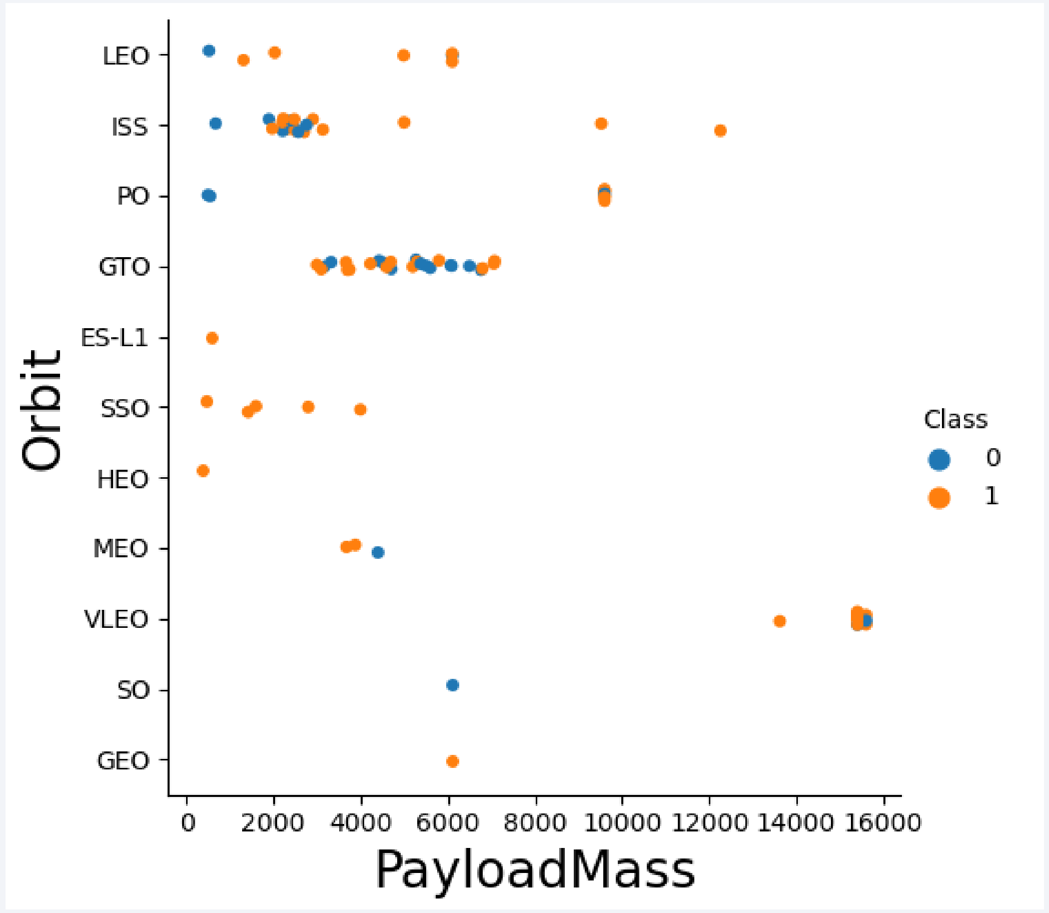
Flight Number vs. Orbit Type



- As show on the plot LEO, ISS, PO, VLEO had a better success rate when the Flight number is bigger.
- GTO shows no relation between success rate and flight number
- ES-L1, SSO, HEO and GEO has no unsuccessful flight
- MEO and SO has no successful flight

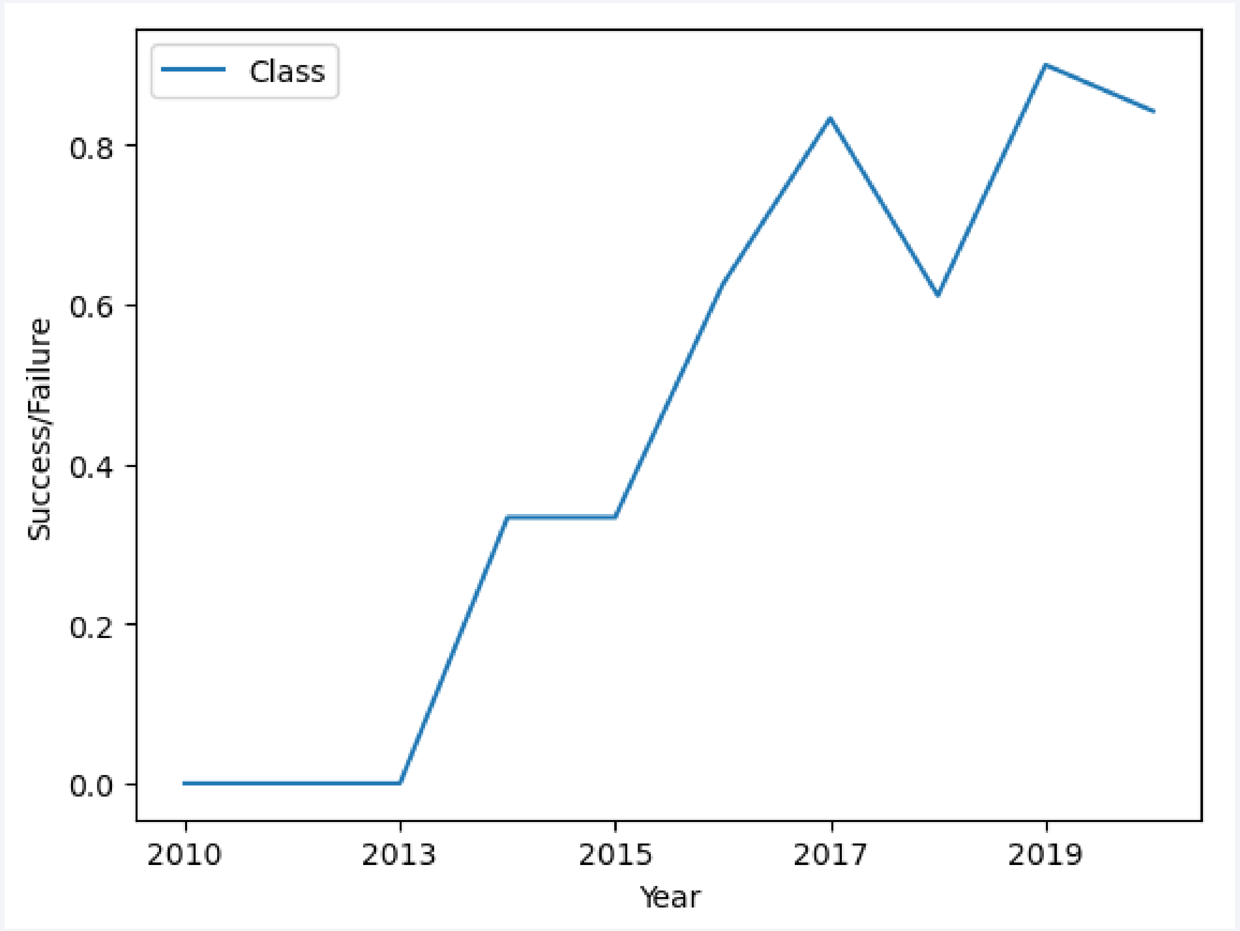
Payload vs. Orbit Type

- Most heavy payload is for SSO and HEO
- After them ISS and PO have heaviest Payloads

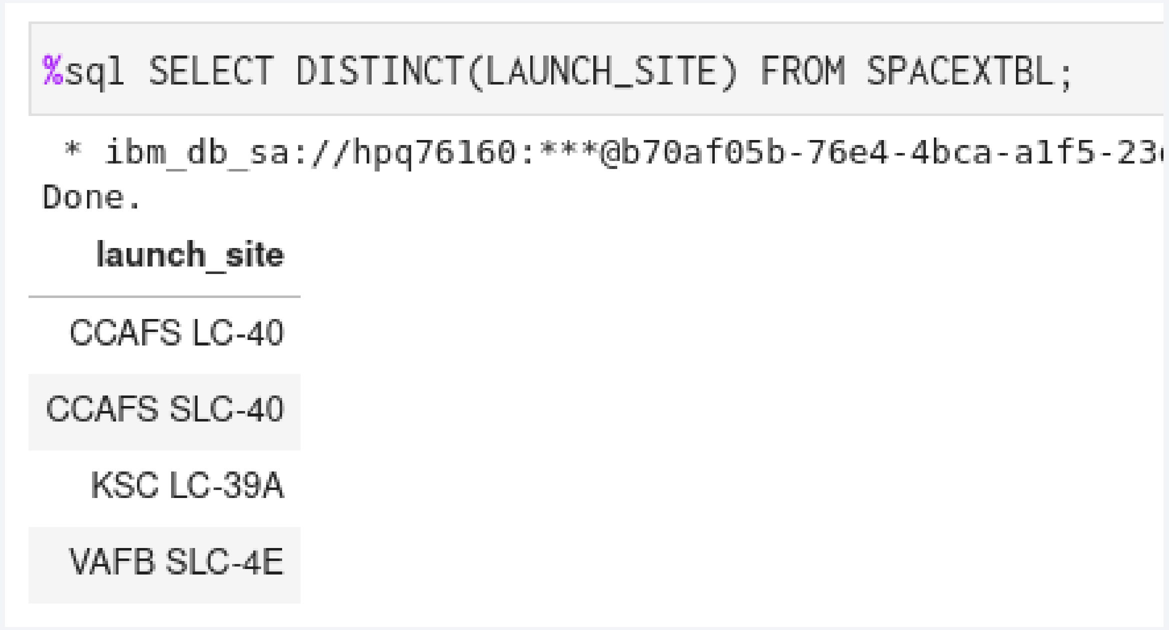


Launch Success Yearly Trend

- Average success shows and up trends from 2013 to 2020. It means every year Space-X had more successful landing
- There is a gap on 2018, but the trends is to up



All Launch Site Names



```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL;

 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23
Done.
```

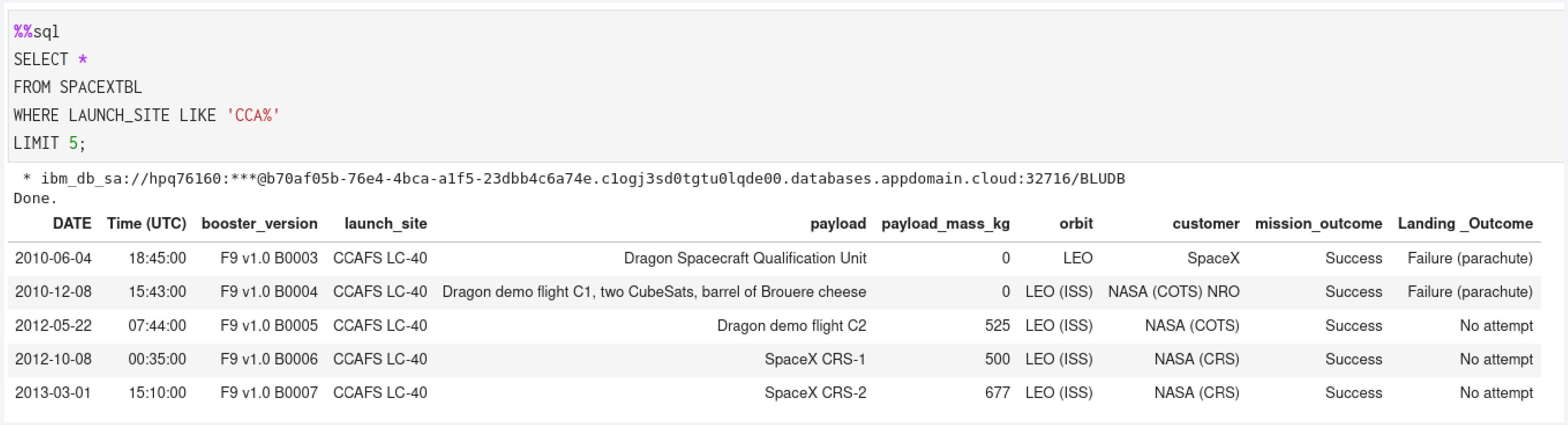
launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

- Names of the unique launch sites
  - CCAFS LC-40
  - VAFB SLC-4E
  - KSC LC-39A
  - CCAFS SLC-40
- We use DISTINCT to find the unique names in SQL query

Launch Site Names Begin with 'CCA'

- To find 5 records where launch sites begin with `CCA` must can this query:

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



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

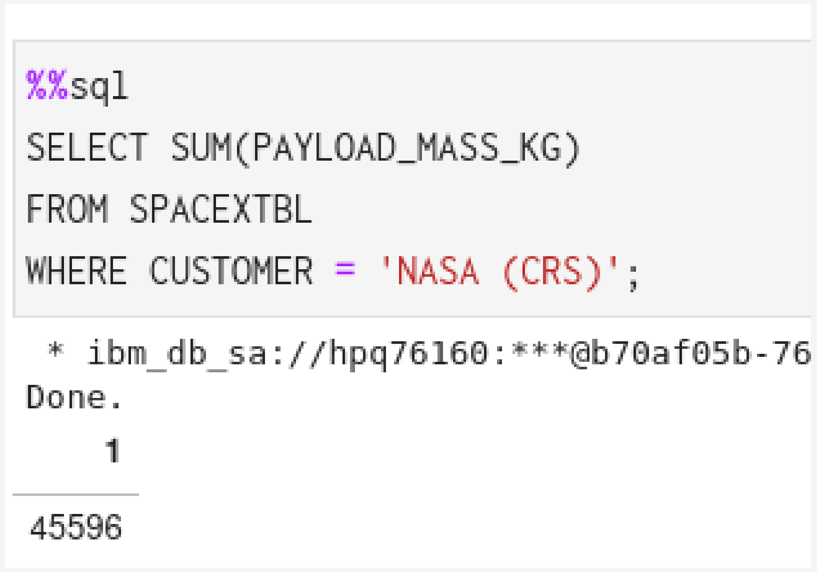
```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/BLUDB
Done.
```

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

Total Payload Mass

- To calculate the total payload carried by boosters from NASA use this query:

```
SELECT SUM(PAYLOAD_MASS_KG)
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)';
```



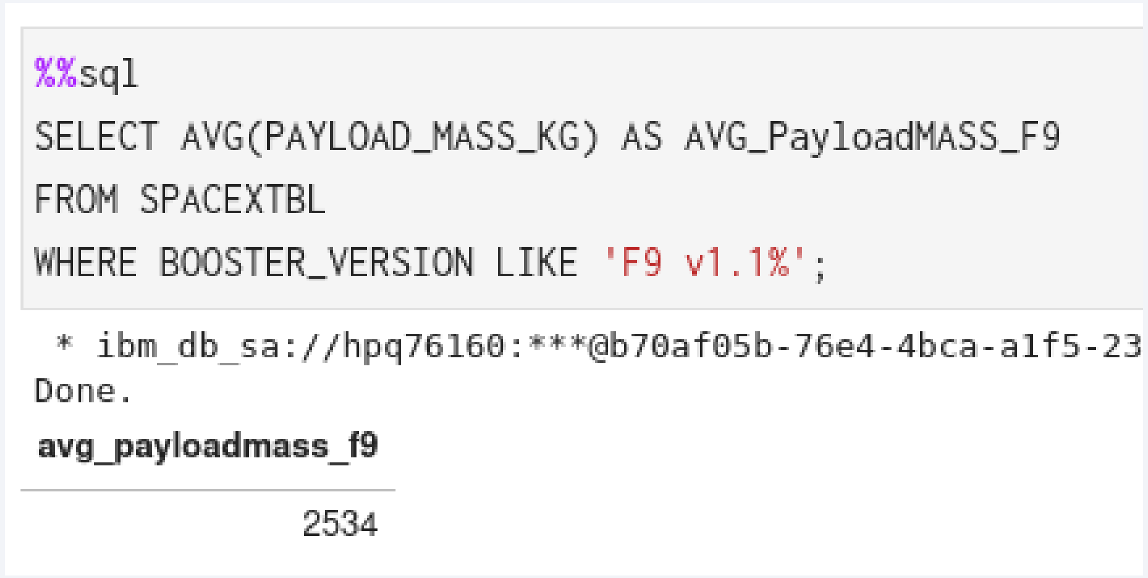
```
%%sql
SELECT SUM(PAYLOAD_MASS_KG)
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)';
```

```
 * ibm_db_sa://hpq76160:***@b70af05b-76
Done.
     1
45596
```

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

```
SELECT AVG(PAYLOAD_MASS_KG) AS AVG_PayloadMASS_F9
FROM SPACEXTBL
WHERE BOOSTER_VERSION LIKE 'F9 v1.1%';
```



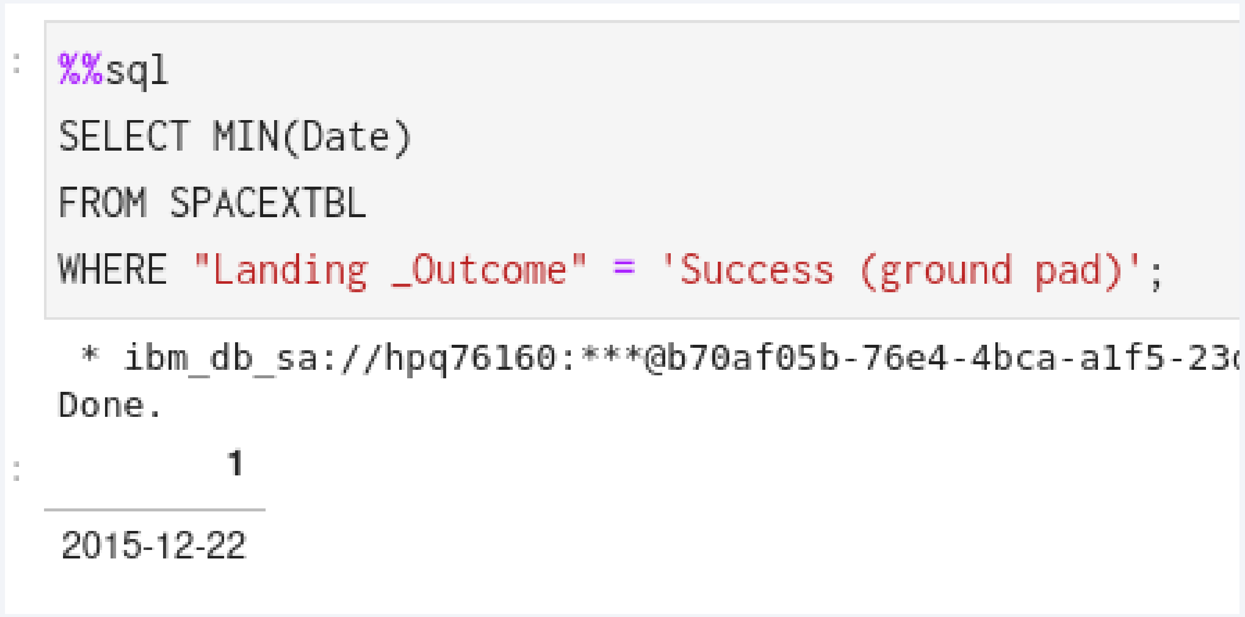
```
%%sql
SELECT AVG(PAYLOAD_MASS_KG) AS AVG_PayloadMASS_F9
FROM SPACEXTBL
WHERE BOOSTER_VERSION LIKE 'F9 v1.1%';
```

```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23
Done.
```

avg_payloadmass_f9
2534

First Successful Ground Landing Date

- First successful landing outcome on ground pad was 22nd December 2015



```
%%sql
SELECT MIN(Date)
FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Success (ground pad)';
```

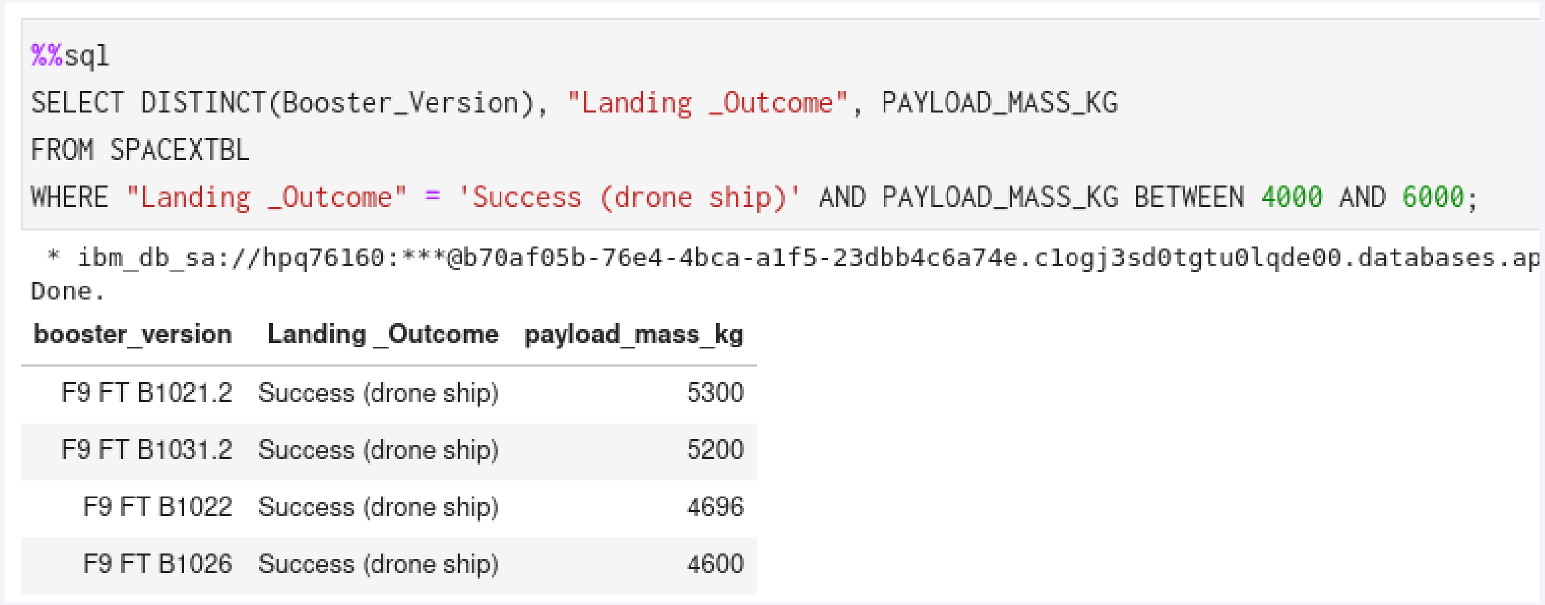
```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23
Done.
```

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We can use this query for Successful Drone Ship Landing with Payload between 4000 and 6000

```
SELECT DISTINCT(Booster_Version), "Landing _Outcome", PAYLOAD_MASS_KG
FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Success (drone ship)' AND PAYLOAD_MASS_KG BETWEEN 4000 AND 6000;
```



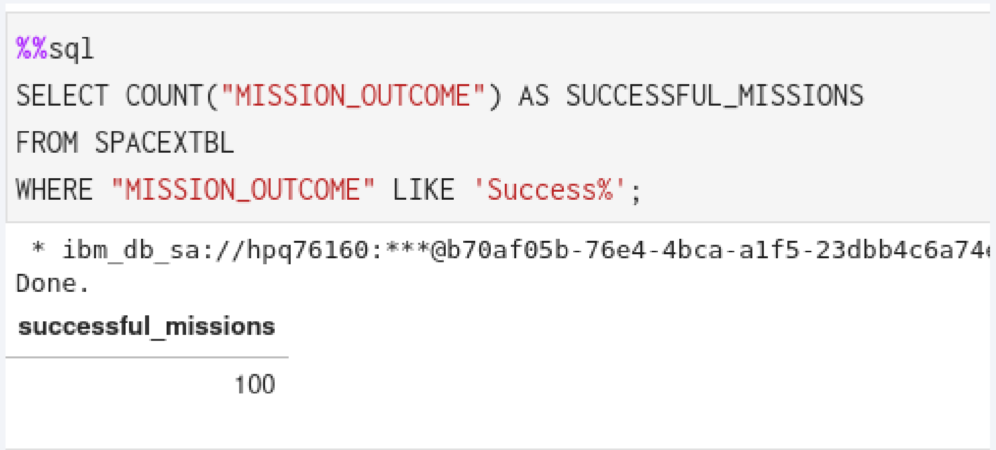
```
%%sql
SELECT DISTINCT(Booster_Version), "Landing _Outcome", PAYLOAD_MASS_KG
FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Success (drone ship)' AND PAYLOAD_MASS_KG BETWEEN 4000 AND 6000;
```

```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.ap
Done.
```

booster_version	Landing _Outcome	payload_mass_kg
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600

Total Number of Successful and Failure Mission Outcomes

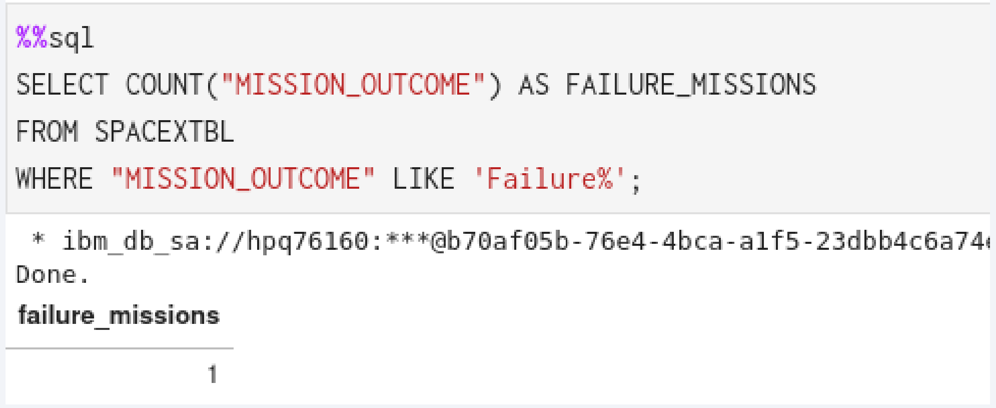
```
SELECT COUNT("MISSION_OUTCOME") AS SUCCESSFUL_MISSIONS
FROM SPACEXTBL
WHERE "MISSION_OUTCOME" LIKE 'Success%';
```



```
%%sql
SELECT COUNT("MISSION_OUTCOME") AS SUCCESSFUL_MISSIONS
FROM SPACEXTBL
WHERE "MISSION_OUTCOME" LIKE 'Success%';
```

```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74
Done.
successful_missions
                100
```

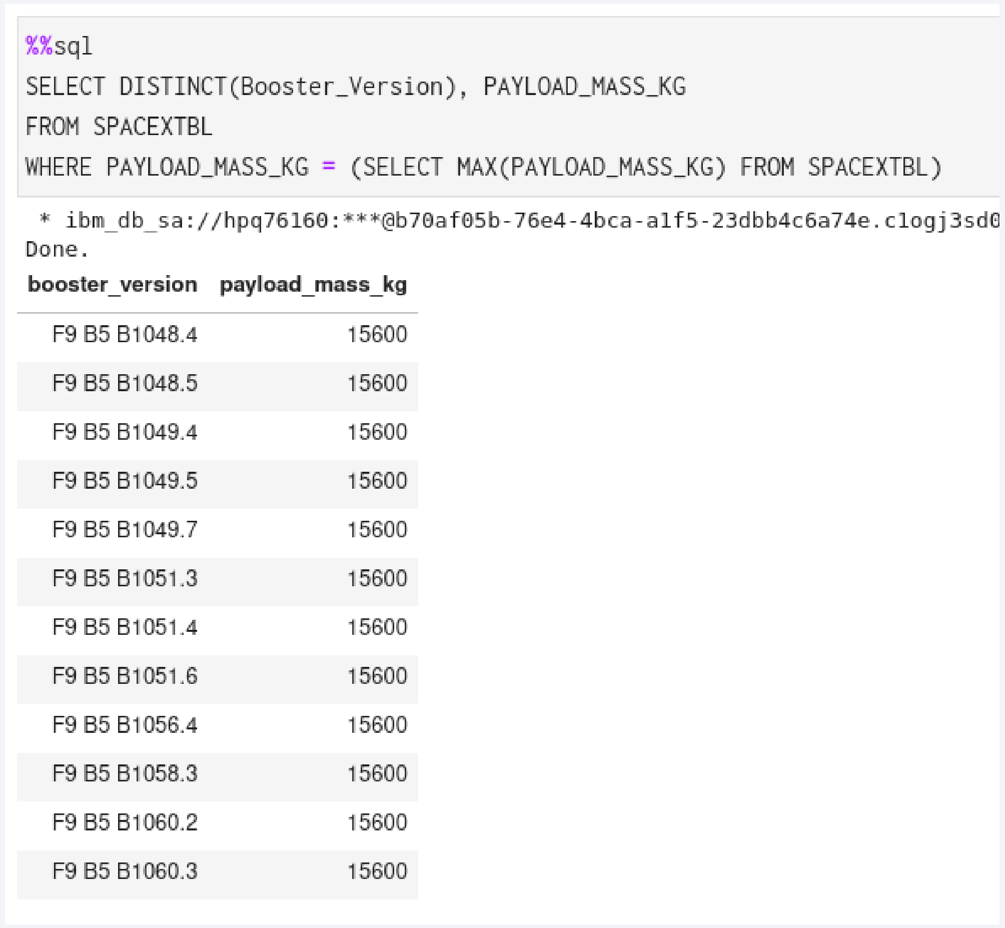
```
SELECT COUNT("MISSION_OUTCOME") AS FAILURE_MISSIONS
FROM SPACEXTBL
WHERE "MISSION_OUTCOME" LIKE 'Failure%';
```



```
%%sql
SELECT COUNT("MISSION_OUTCOME") AS FAILURE_MISSIONS
FROM SPACEXTBL
WHERE "MISSION_OUTCOME" LIKE 'Failure%';
```

```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74
Done.
failure_missions
               1
```

Boosters Carried Maximum Payload



```
%%sql
SELECT DISTINCT(Booster_Version), PAYLOAD_MASS_KG
FROM SPACEXTBL
WHERE PAYLOAD_MASS_KG = (SELECT MAX(PAYLOAD_MASS_KG) FROM SPACEXTBL)
```

```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.clogj3sd0
Done.
```

booster_version	payload_mass_kg
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600

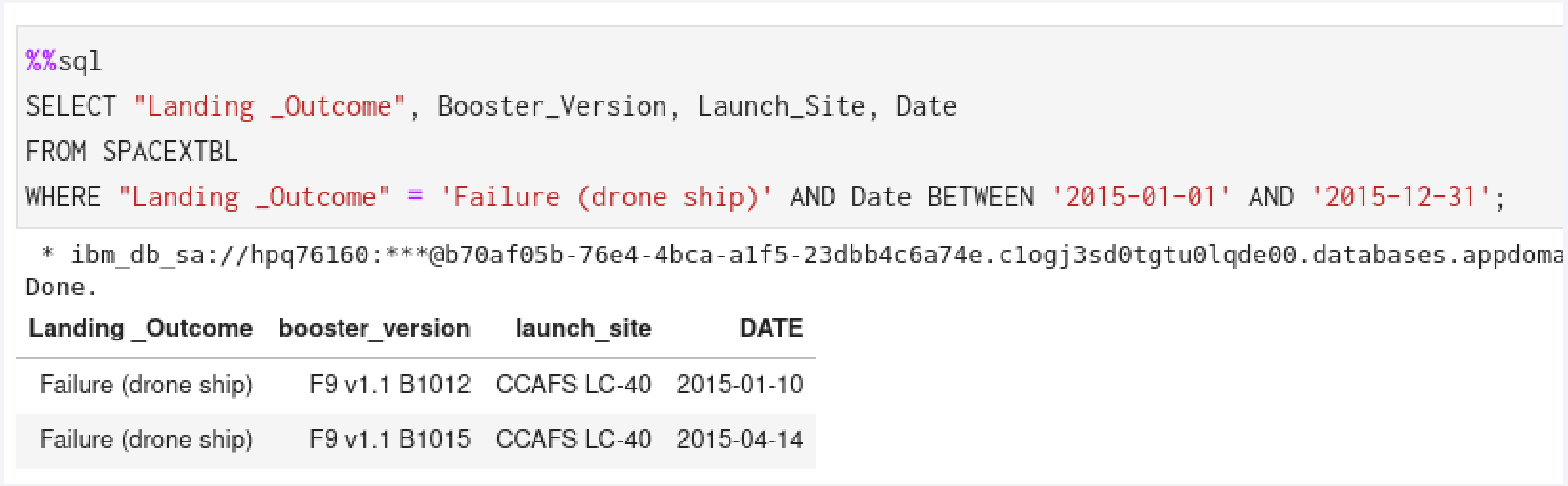
- List of the booster which have carried the maximum payload mass (kg)

F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600
F9 B5 B1051.3	15600
F9 B5 B1051.4	15600
F9 B5 B1051.6	15600
F9 B5 B1056.4	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600
F9 B5 B1060.3	15600

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - Use Where, between asd condition to get the results.

```
Landing _Outcome         booster_version    launch_site     DATE
Failure (drone ship)      F9 v1.1 B1012      CCAFS LC-40     2015-01-10
Failure (drone ship)      F9 v1.1 B1015      CCAFS LC-40     2015-04-14
```



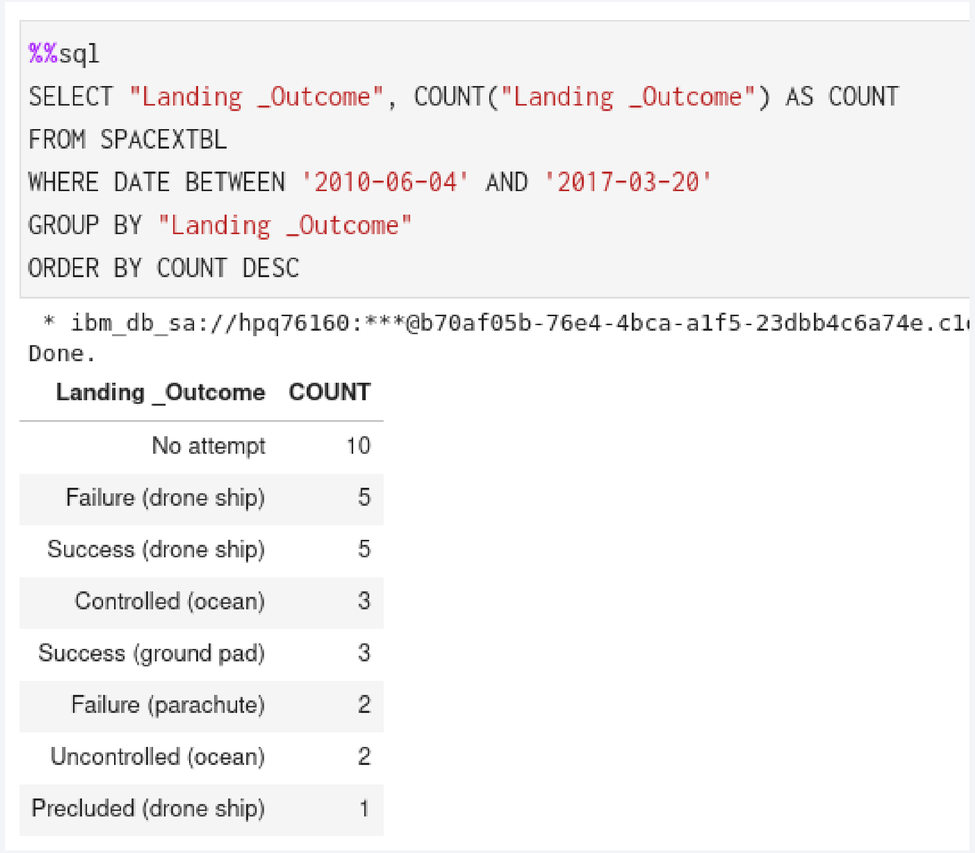
```
%%sql
SELECT "Landing _Outcome", Booster_Version, Launch_Site, Date
FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Failure (drone ship)' AND Date BETWEEN '2015-01-01' AND '2015-12-31';
```

```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdoma
Done.
```

Landing _Outcome	booster_version	launch_site	DATE
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40	2015-01-10
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40	2015-04-14

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

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



```
%%sql
SELECT "Landing _Outcome", COUNT("Landing _Outcome") AS COUNT
FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing _Outcome"
ORDER BY COUNT DESC
```

```
 * ibm_db_sa://hpq76160:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.cl
Done.
```

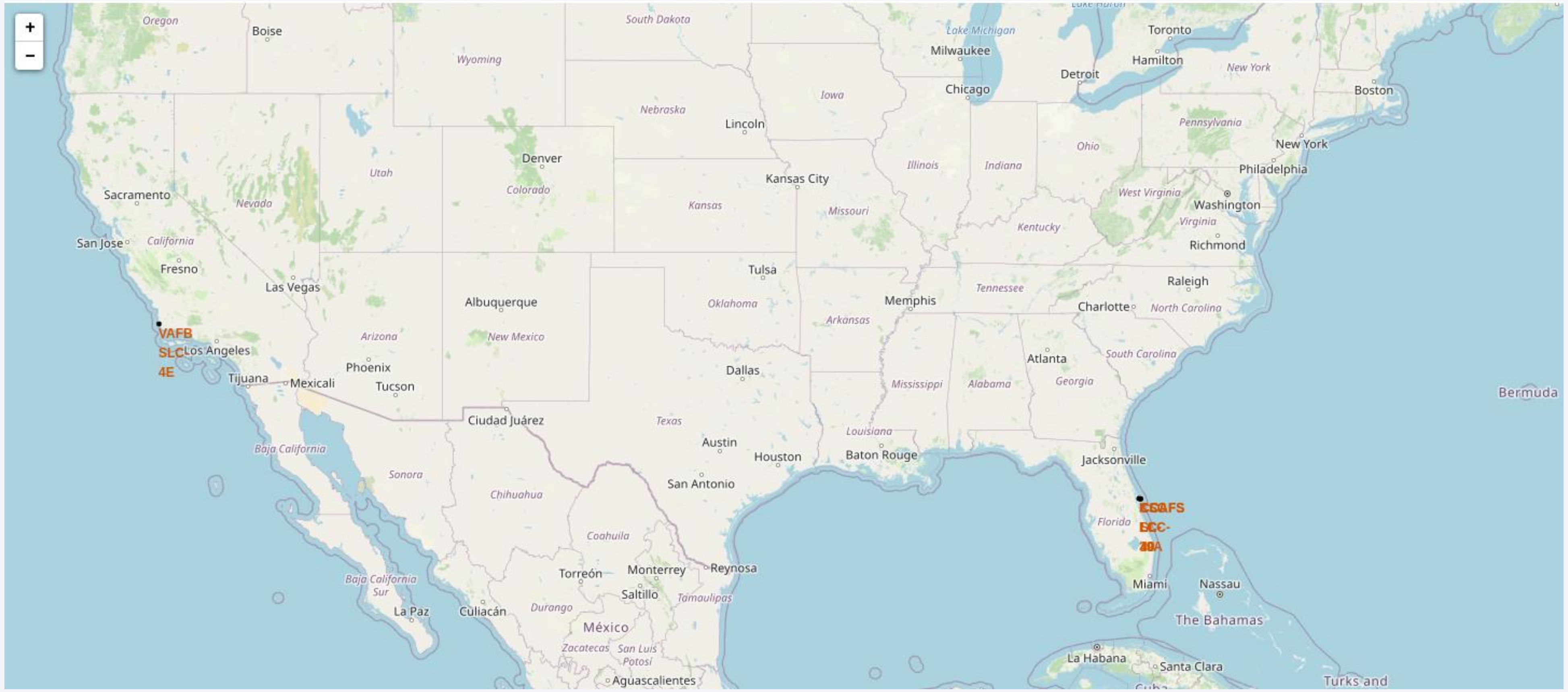
Landing _Outcome	COUNT
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

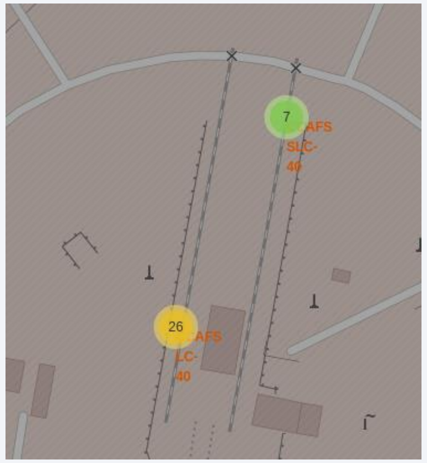
All launch sites on a map

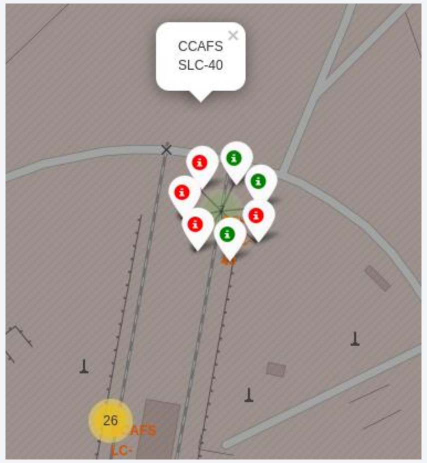
- Space-X launch sites are in USA, Florida and California.

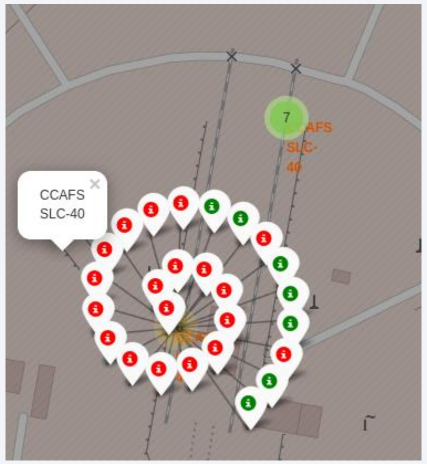


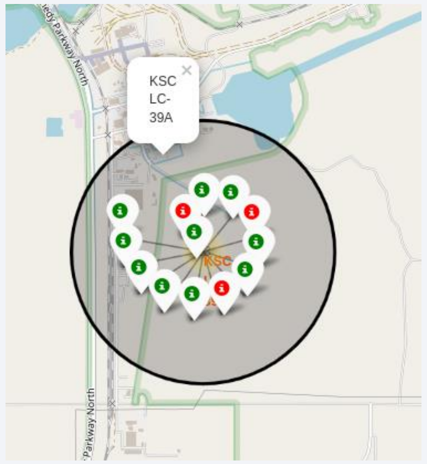
Show the color-labeled launch outcomes on the map

Green Marker shows successful launches and Red Marker shows failure



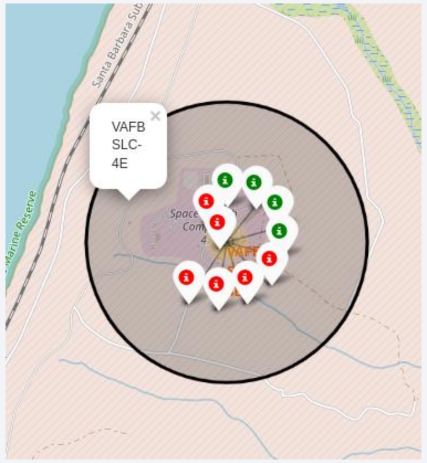






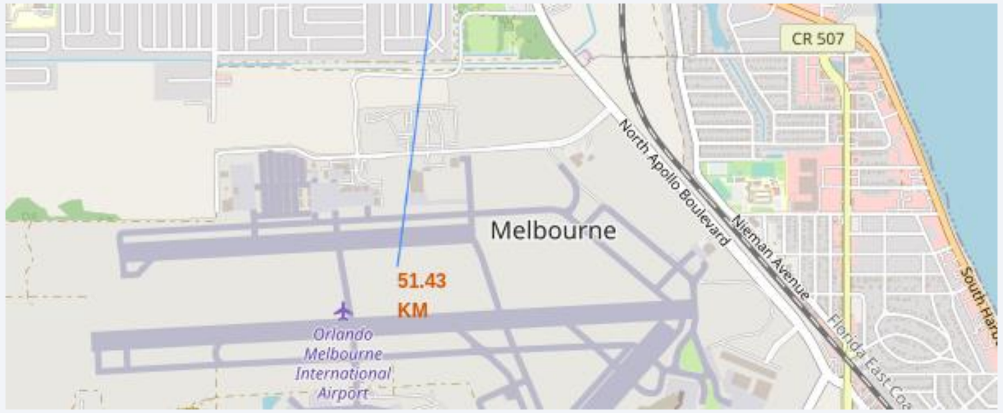
Florida Launch Sites

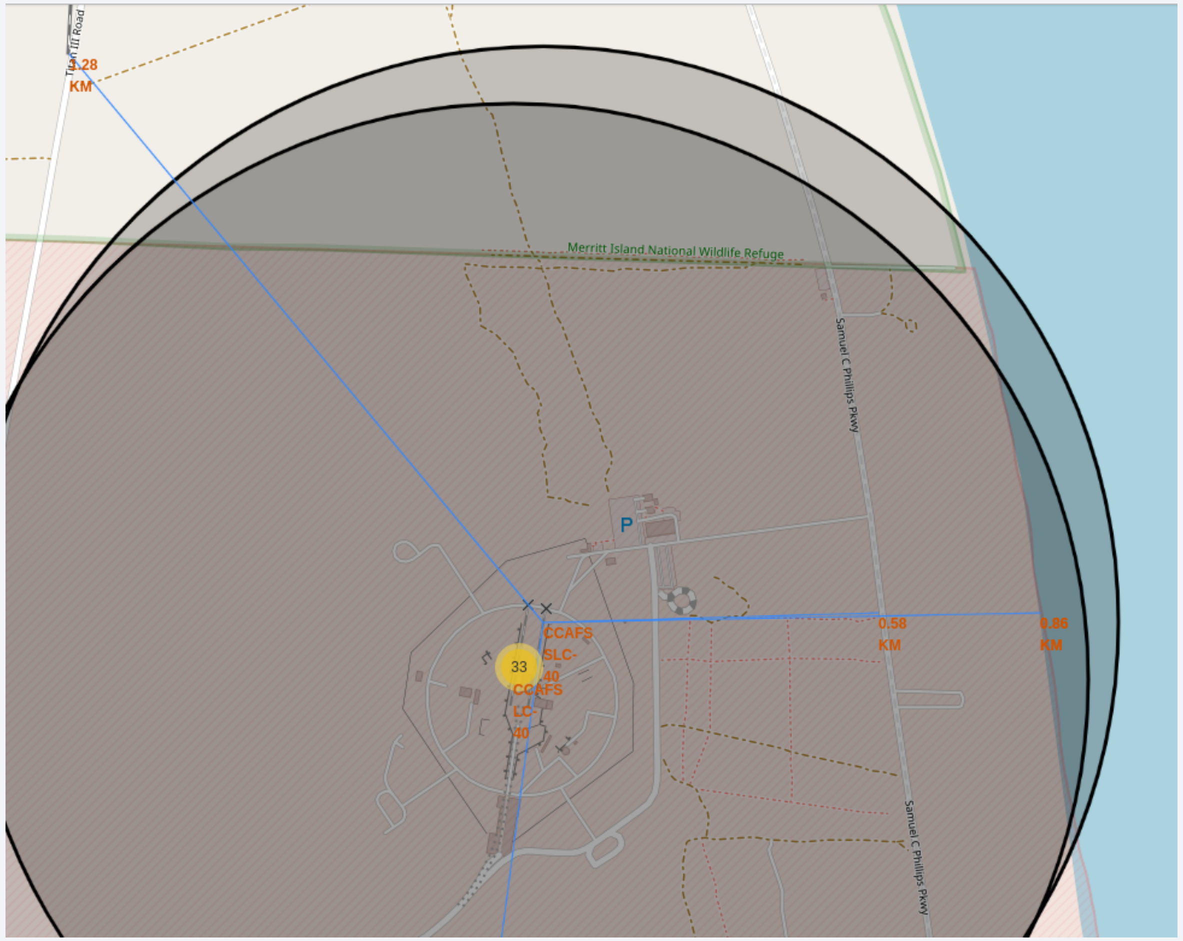
California Launch Sites

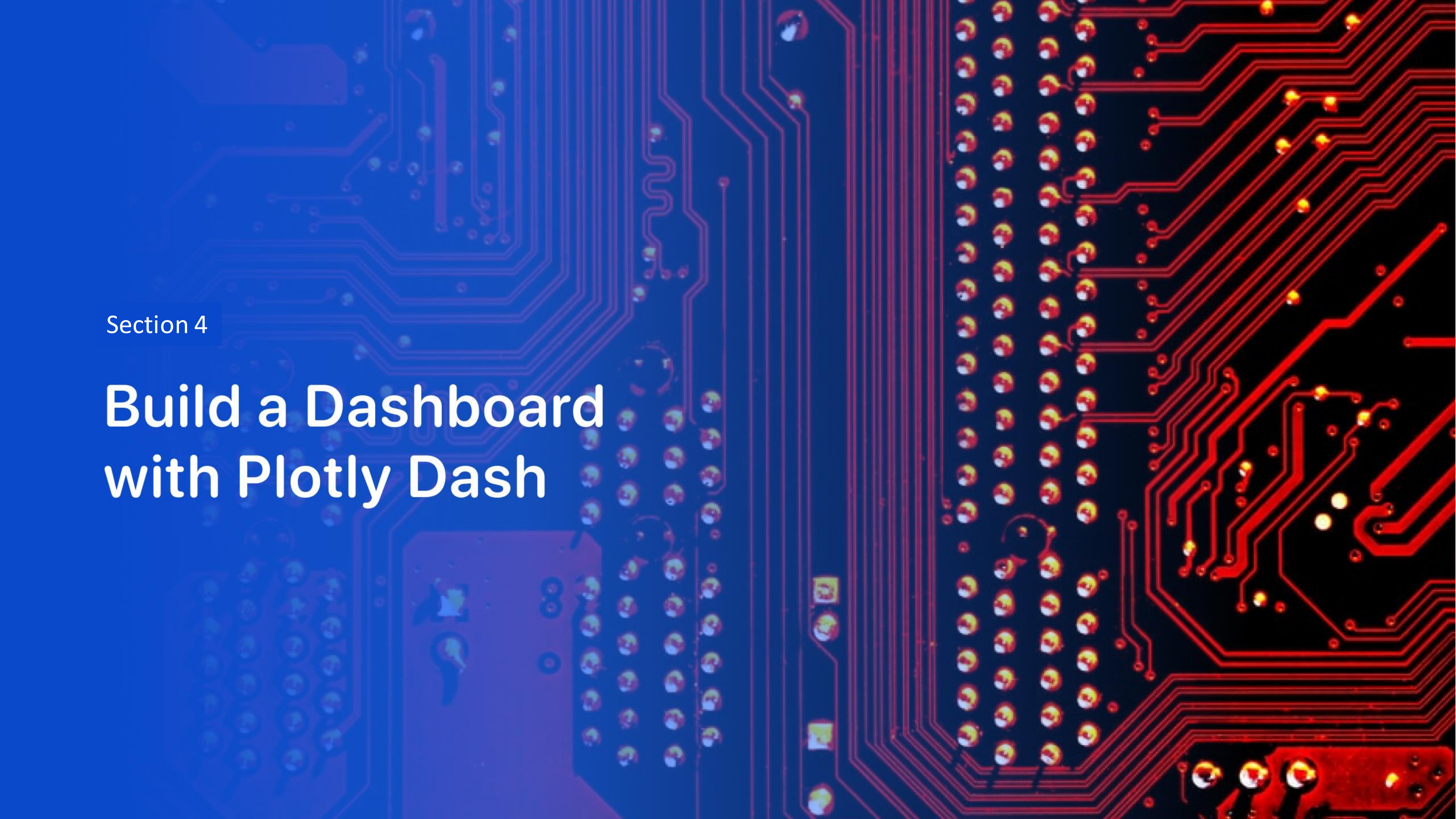


Launch Site distance to landmarks

- Distance to Orlando Melborn Internation Airport: 51.43 KM
- Distance to Nisara Railroad: 1.28 KM
- Distance to Samuel C Philips PKwy: 0.58 KM
- Distance to coastline: 0.86 KM



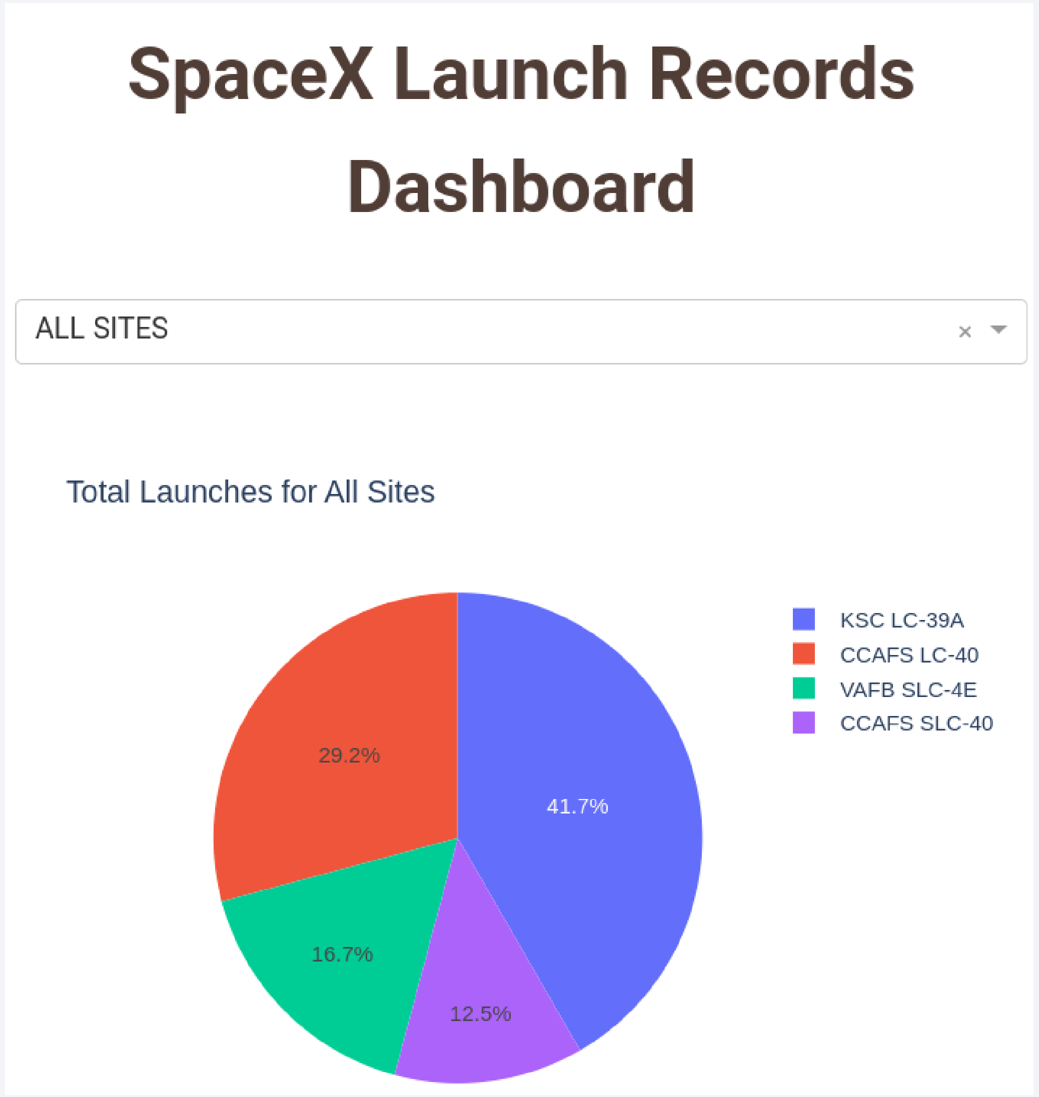




Section 4

Build a Dashboard with Plotly Dash

Pie chart to shoe all sites launch success



SpaceX Launch Records Dashboard

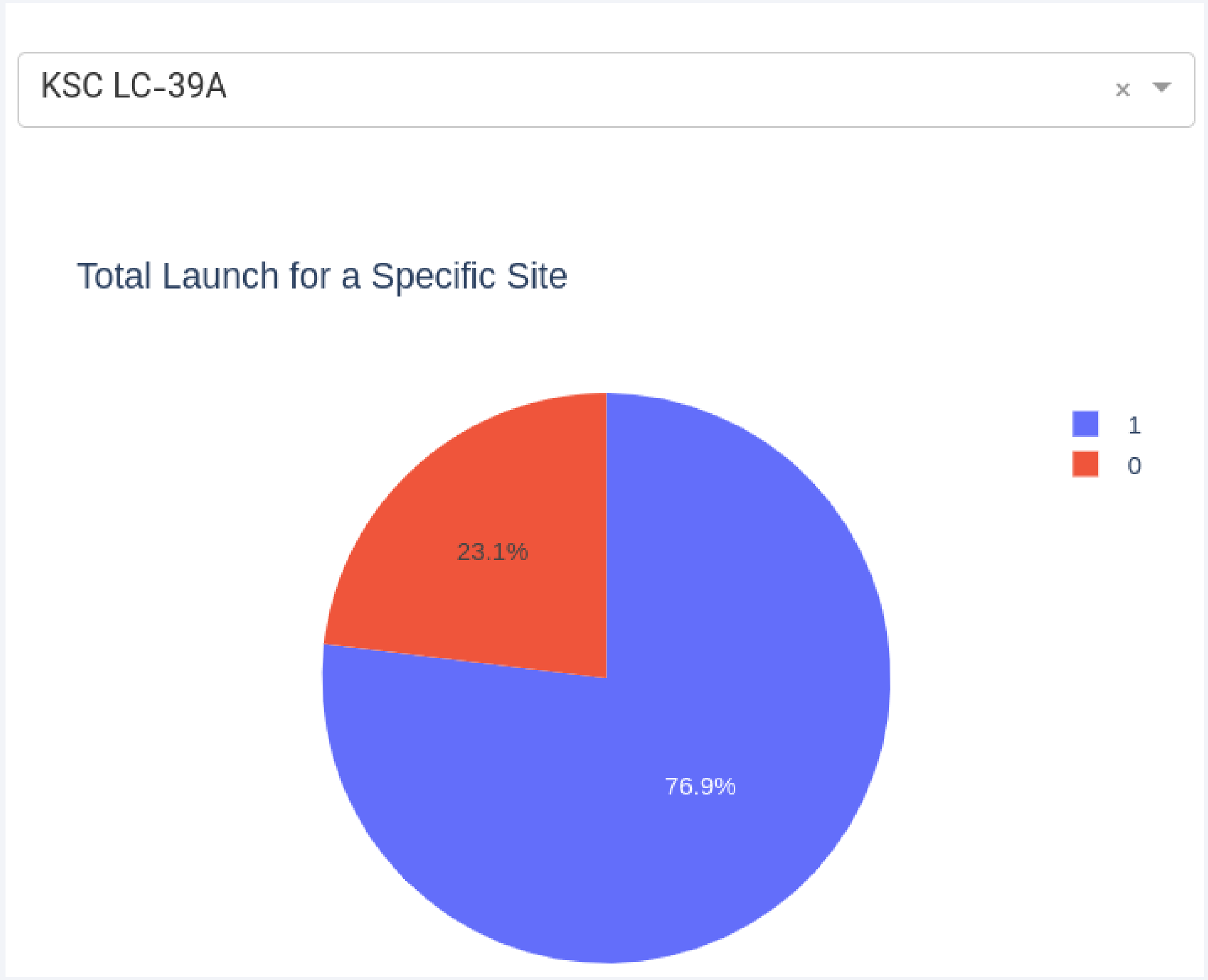
ALL SITES

Total Launches for All Sites

- We see the KSC LC-39A has the most successful launches in all sites
- Second place is for CCAFS LC-40
- Third is VAFB SLC-4E
- Forth is CCAFS SLC-40

KSC LC-39A pie chart success vs failure

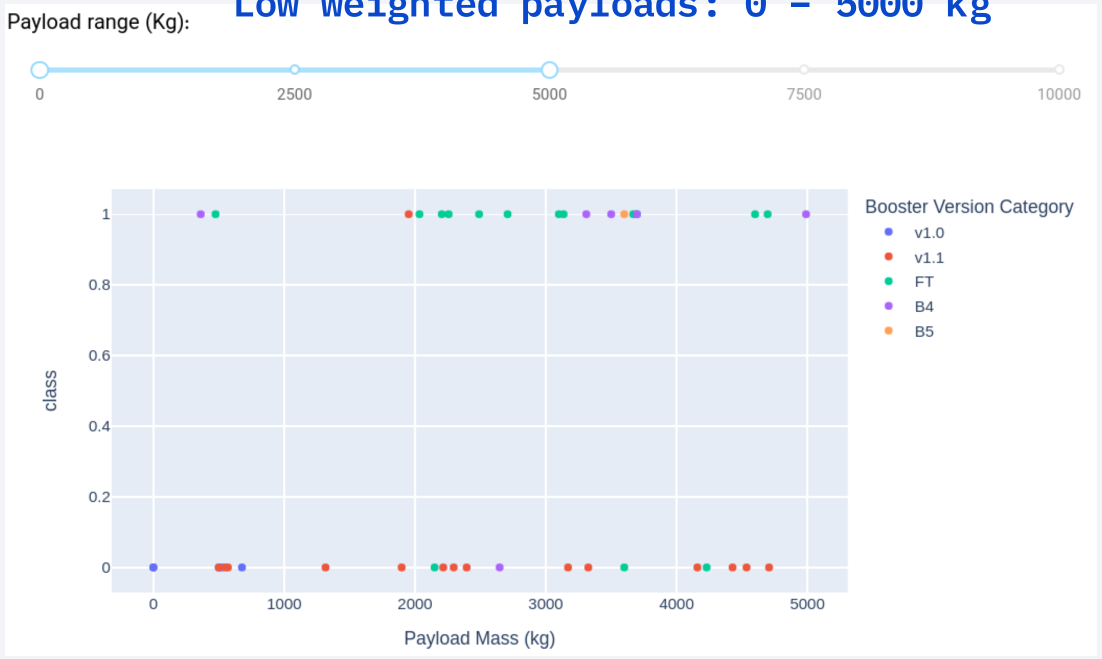
- KSC LC-39A has 76.9% success ratio



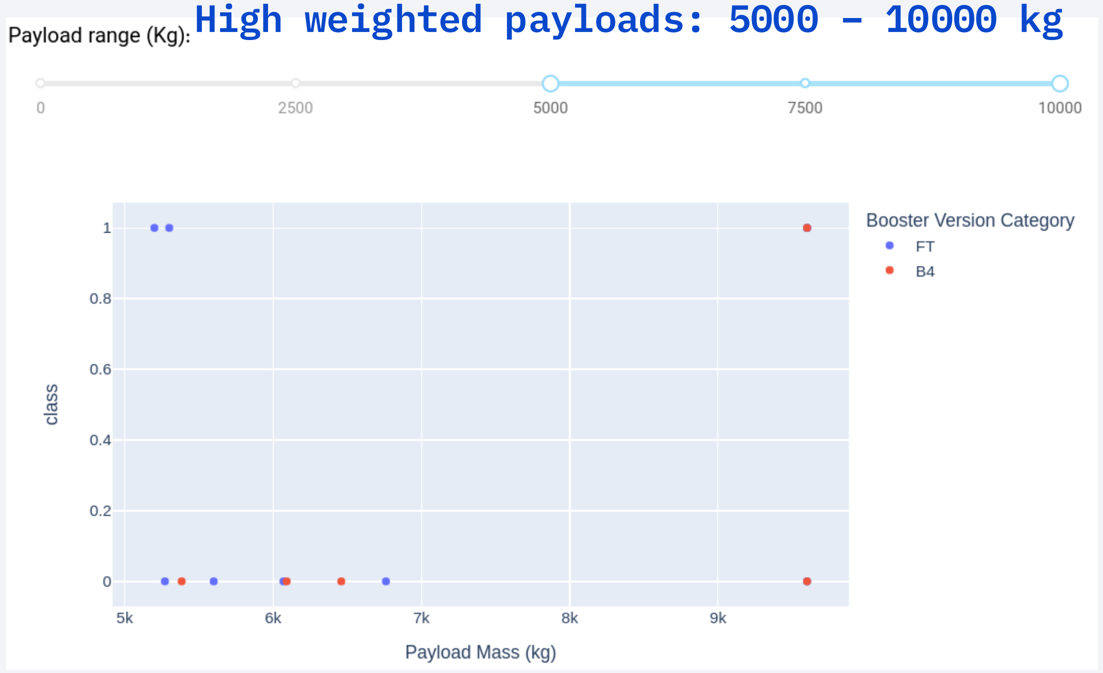
Scatter plot of Payload vs Launch Outcome for all sites

- Scatter plot of payload vs launch outcome for all sites with different payload in range slider.
- We see the success rate for low weighted payloads is higher than the heavy weighted payloads

Low weighted payloads: 0 – 5000 kg



High weighted payloads: 5000 – 10000 kg

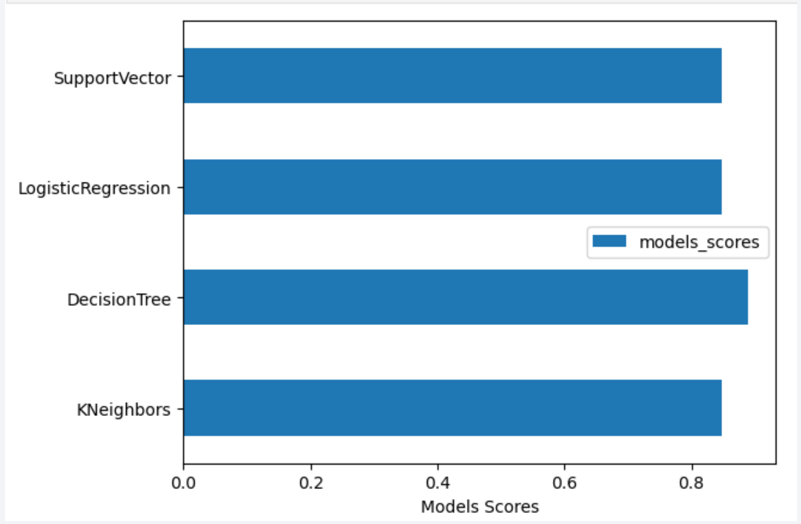


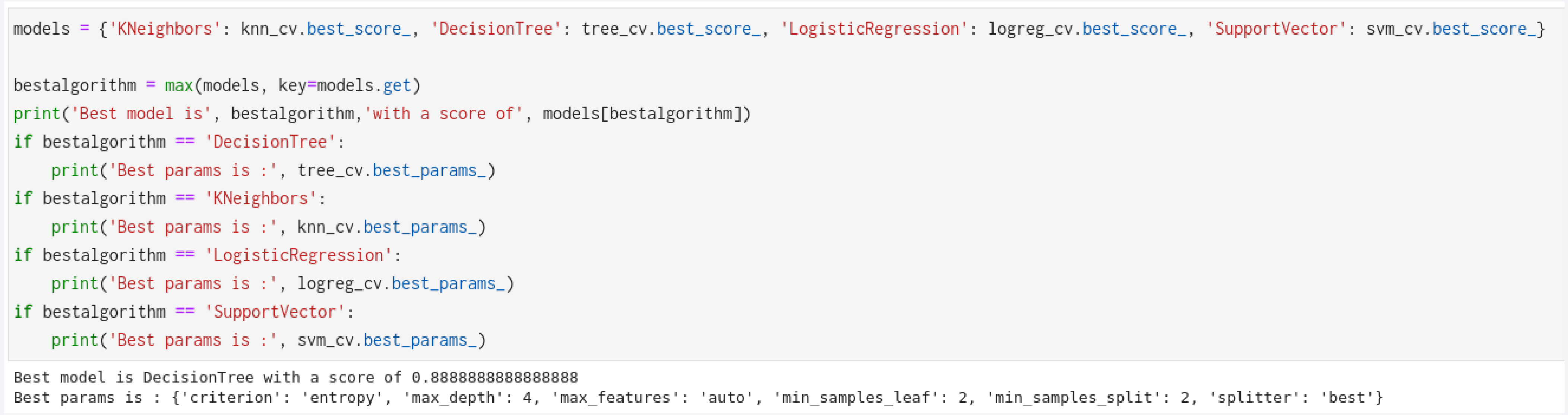
Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The Decision Tree has the best accuracy with 0.88



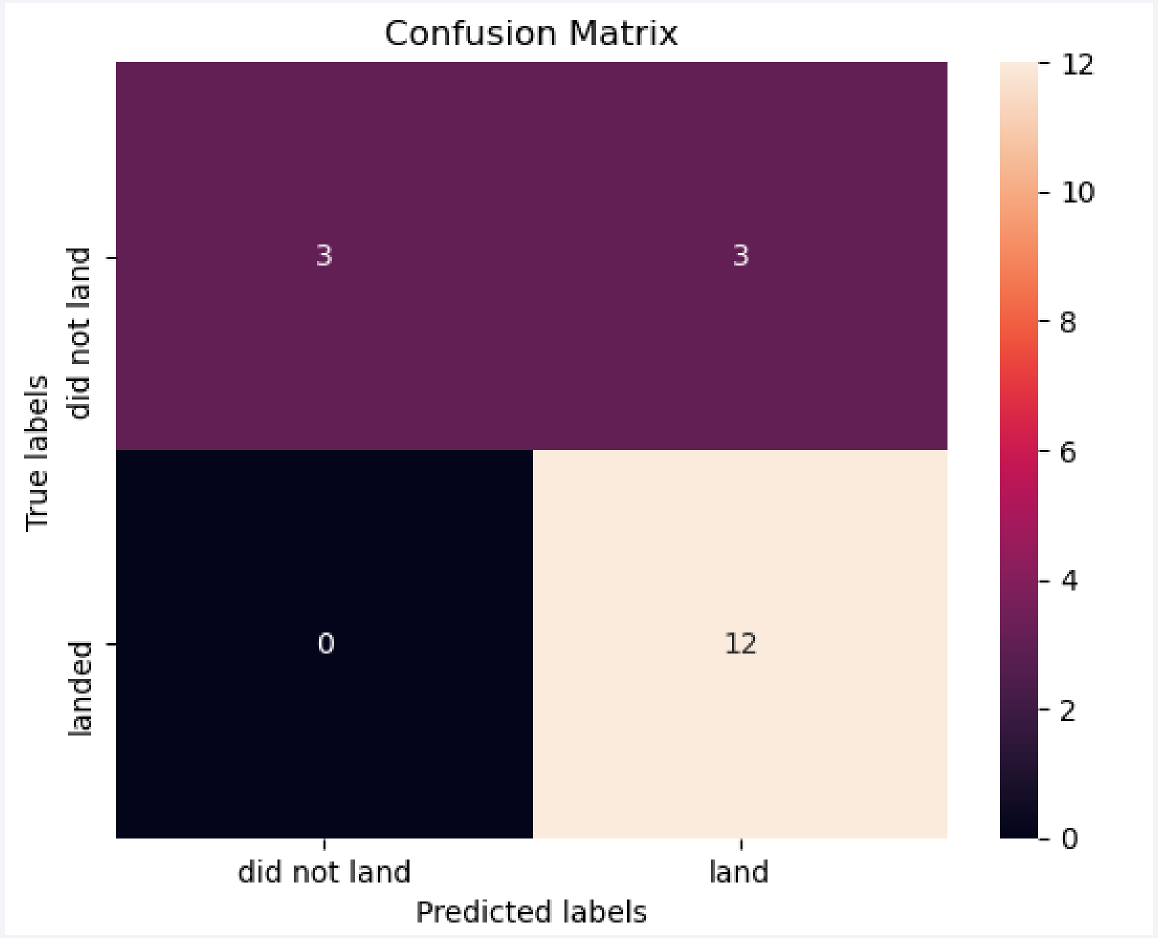


```
models = {'KNeighbors': knn_cv.best_score_, 'DecisionTree': tree_cv.best_score_, 'LogisticRegression': logreg_cv.best_score_, 'SupportVector': svm_cv.best_score_}

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 0.8888888888888888
Best params is : {'criterion': 'entropy', 'max_depth': 4, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'best'}
```

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

- Larger flight amount at launch site has the greater success rate
- from 2013 succes rates increase yearly in Space-X
- ES-L1, GEO, HEO, SSO, VLEO orbits has the most succes rates
- KSC LC-39A had the most successful launches in all sites
- The Decision tree classifier is the best machine learning algorithm for this task.

Appendix

- All codes are available in github:
  - https://github.com/miraftab/IBM_Applied_Data_Science_Capstone.git
- Interactive maps does not display on the github.com, so you can see that notrbook map here:
  - https://nbviewer.org/github/miraftab/IBM_Applied_Data_Science_Capstone/blob/main/06_lab_jupyter_launch_site_location.ipynb

Thank you!