IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Khushi Patel
16.09.2023

Outline

Executive Summary

- Summary of methodologies used:

  -Data Collection using API

  -Data Collection using WebScraping

  -Data Wrangling

  -Data Analysis with SQL

  -Data Visualization

  -Interactive visual analytics using folium and Dash

  -Machine Learning

- Summary of all results obtained:

  -Data Analysis Result

  -Interactive analytics screenshots

  -Predictive Analytics machine learning Results

Introduction

SpaceX is a company who has disrupt the space industry by offering a rocket launches specifically Falcon 9 as low as 62 million dollars; while other providers cost upward of 165 million dollar each. Most of this saving thanks to SpaceX astounding idea to reuse the first stage of the launch by re-land the rocket to be used on the next mission. Repeating this process will make the price down even further. As a data scientist of a startup SpaceY rivaling SpaceX, the goal of this project is to create the machine learning pipeline to predict the landing outcome of the first stage in the future. This project is crucial in identifying the right price to bid against SpaceX for a rocket launch.

The problems included:

- Identifying all factors that influence the landing outcome.
- The relationship between each variables and how it is affecting the outcome.
- The best condition needed to increase the probability of successful landing.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX REST API and web scraping method
- Perform data wrangling
  - Data was processed using one hot encoding
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

- The process of gathering and analyzing accurate data from various sources to find answers to research problems, trends and probabilities, etc., to evaluate possible outcomes is Known as Data Collection.
- The data was collected using REST API.Some steps involved were:
    - Get request to import data
    - Decode data using json and json_normalize
    - Clean data and replace missing values appropriately
    - Web scraping using BeautifulSoup and converting table into pandas dataframe

Data Collection – SpaceX API

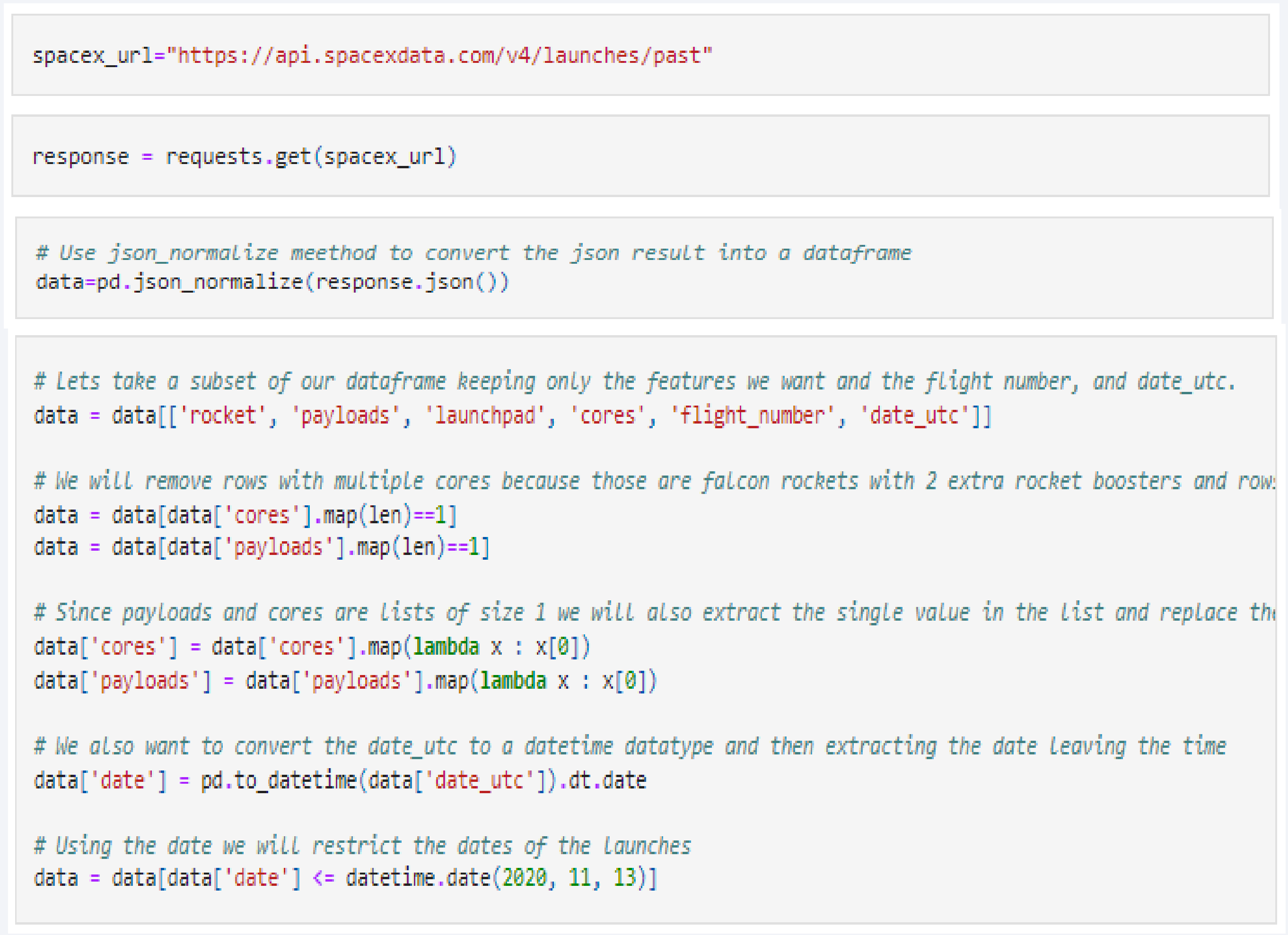
Import Data
(get request)

Convert json file to dataframe using json_normalize

Clean the data
(Replace missing values)

- GitHub URL:

https://github.com/khushi13124/DataScience_Courseera/blob/main/Capstone%20Project(Module-10)/jupyter-labs-spacex-data-collection-api.ipynb



```
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 row
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 th
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)]
```

Data Collection - Scraping

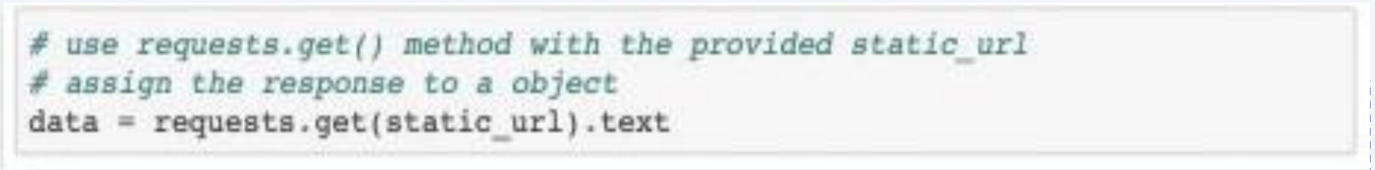
Convert data into text

Create BeautifulSoup Object

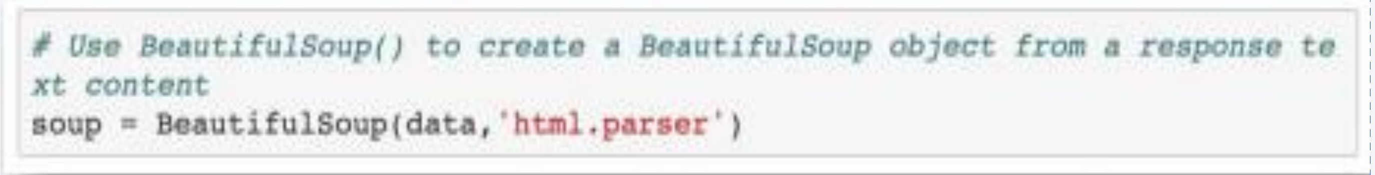
Extract column names

- GitHub URL:

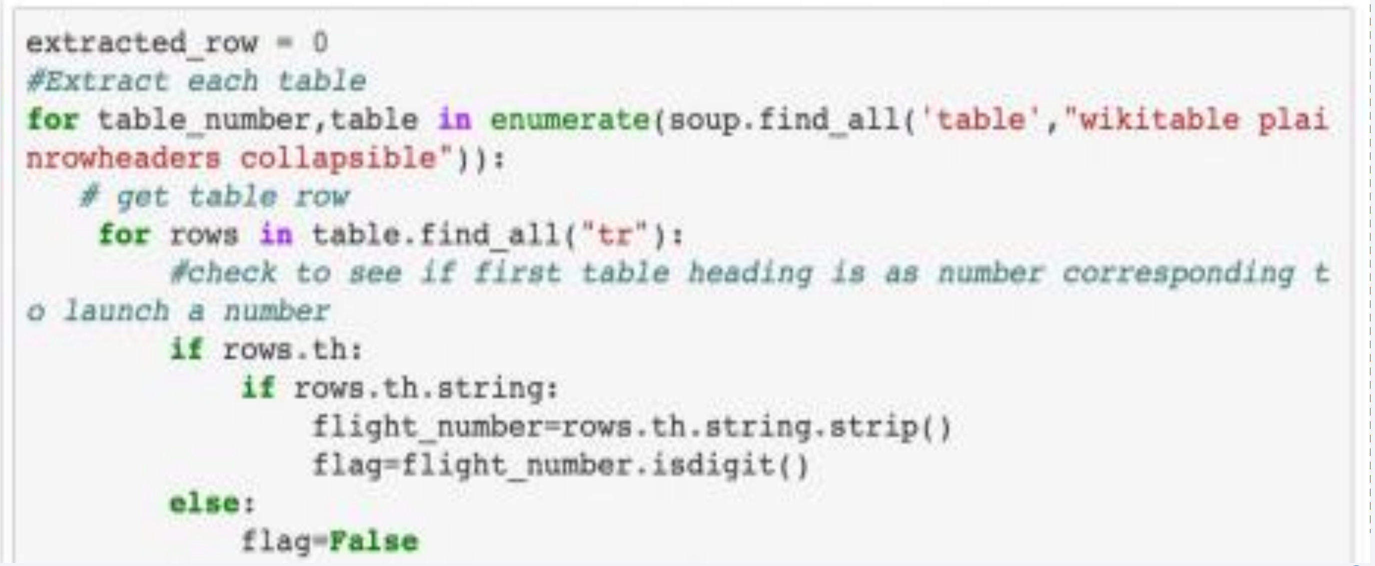
https://github.com/khushi13124/DataScience_Courseera/blob/main/Capstone%20Project(Module-10)/jupyter-labs-webscraping.ipynb



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



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



```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plai
nrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding t
o launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
```

Data Wrangling

- Data wrangling is the process of removing errors and combining complex data sets to make them more accessible and easier to analyze.

- GitHub URL:

https://github.com/khushi13124/DataScience_Courseera/blob/main/Capstone%20Project(Module-10)/IBM-DS0321EN-SkillsNetwork_labs_module_1_L3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

Number & Occurrence of each orbit

Find mission outcome per orbit type

Create landing_outcome label

EDA with Data Visualization

- Scatter plots show dependency of attributes on each other. Once a pattern is determined from the graphs. It’s very easy to see which factors affecting the most to the success of the landing outcomes.

- We used it to find relation between: Payload and Flight Number, Flight number and launch site, Payload and Launch site, Flight number and orbit type , etc.

- GitHub URL:

https://github.com/khushi13124/DataScience_Courseera/blob/main/Capstone%20Project(Module-10)/IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

Select the correct chart to plot

Label the axis

See the plot

Look for patterns that can prove helpful

EDA with SQL

Queries performed using SQL:

- Displaying the names of the launch sites.
- Displaying 5 records where launch sites begin with the string 'CCA'.
- Displaying the total payload mass carried by booster launched by NASA (CRS).
- Displaying the average payload mass carried by booster version F9 v1.1.
- 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

Github URL:
https://github.com/khushi13124/DataScience_Courseera/blob/main/Capstone%20Project(Module-10)/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Using folium we built an interactive map to visualize the launch data
- We took the latitude and longitude coordinates at each launch site and added a circle marker around each launch site with a label of the name of the launch site.
- We then assigned the dataframe launch_outcomes(failure,success) to classes 0 and 1 with Red and Green markers on the map in MarkerCluster().
- We also found answers to following questions:
    - How close the launch sites with railways, highways and coastlines?
    - How close the launch sites with nearby cities?
- Explain why you added those objects
- GitHub URL:

https://github.com/khushi13124/DataScience_Courseera/blob/main/Capstone%20Project(Module-10)/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

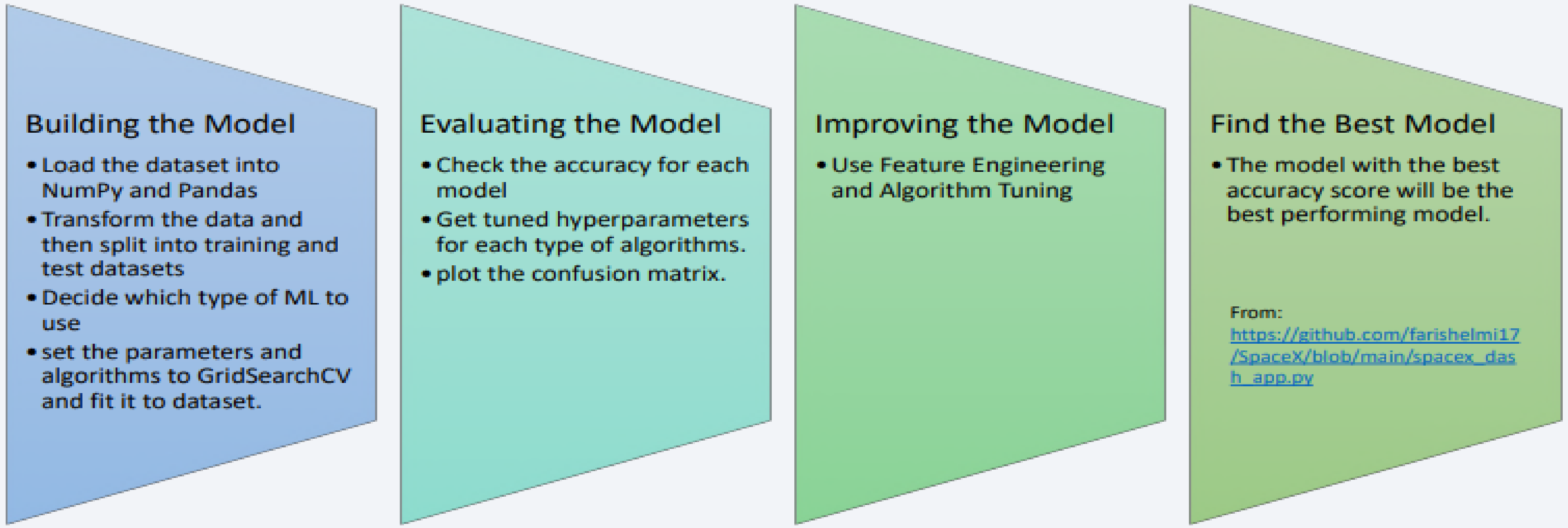
- We made a dashboard to allow to user to select the data as they want and personalize the plots as they want.
- We made

  -pie chart to show the total launches by each site.

  -scatter plot for outcome and payload for different booster version.
- GitHub URL:

https://github.com/khushi13124/DataScience_Courseera/blob/main/Capstone%20Project(Module-10)/spacex_dash_app.py

Predictive Analysis (Classification)



Building the Model

- Load the dataset into NumPy and Pandas
- Transform the data and then split into training and test datasets
- Decide which type of ML to use
- set the parameters and algorithms to GridSearchCV and fit it to dataset.

Evaluating the Model

- Check the accuracy for each model
- Get tuned hyperparameters for each type of algorithms.
- plot the confusion matrix.

Improving the Model

- Use Feature Engineering and Algorithm Tuning

Find the Best Model

- The model with the best accuracy score will be the best performing model.

From:
https://github.com/farishelmi17/SpaceX/blob/main/spacex_dash_app.py

- GitHub URL:

https://github.com/khushi13124/DataScience_Courseera/blob/main/Capstone%20Project(Module-10)/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

The results are categorized into following fields:

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

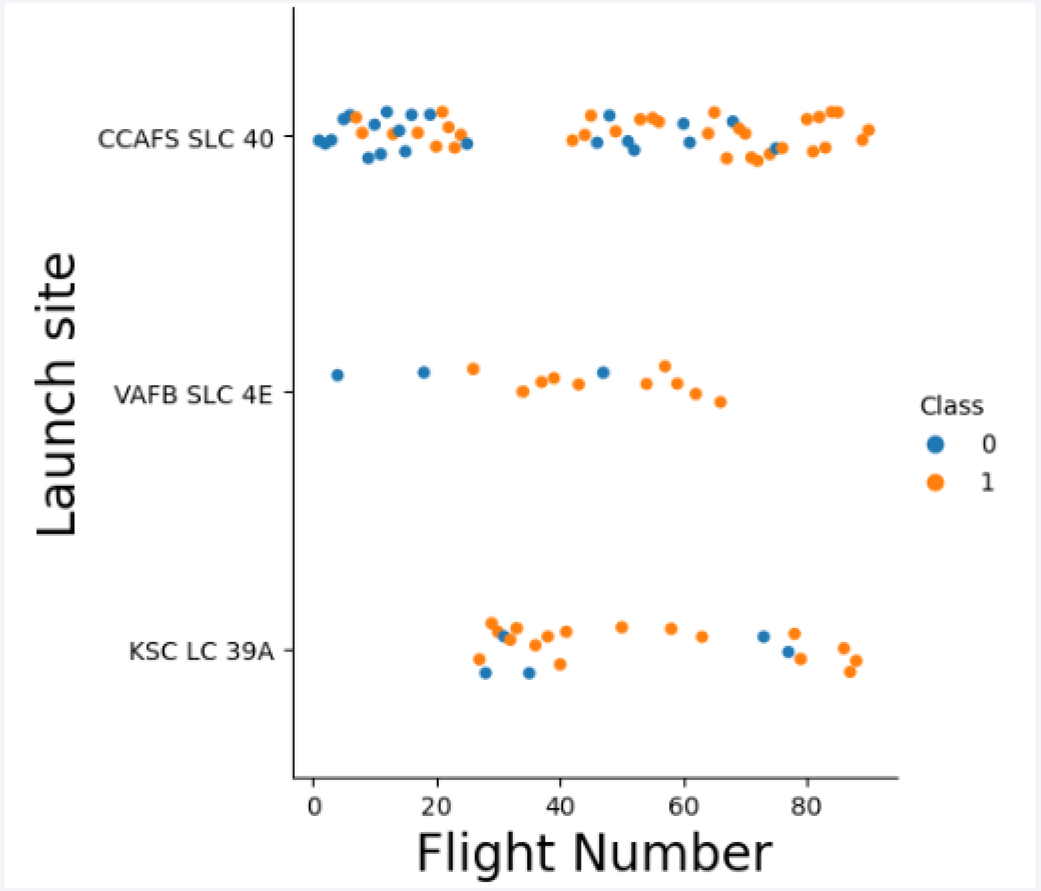
These results are shown in the slides that follow.

Section 2

Insights drawn from EDA

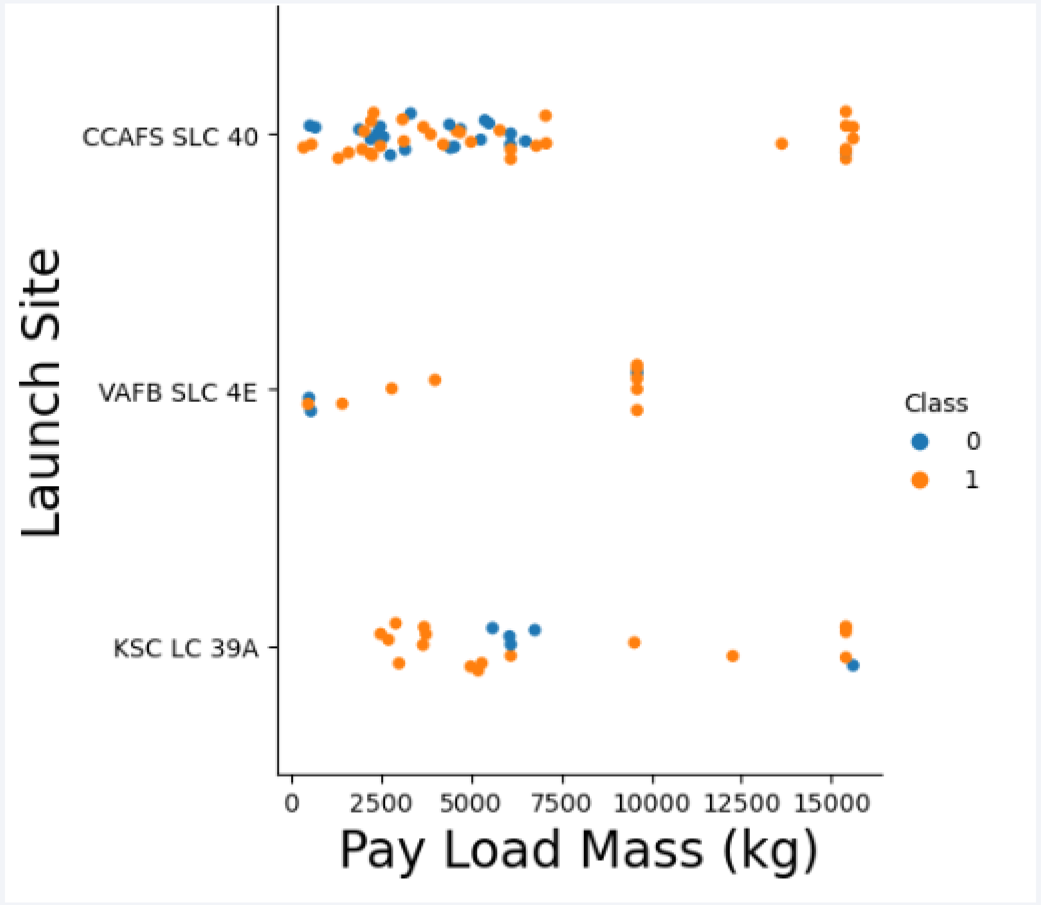
Flight Number vs. Launch Site

- As the flight number increases, the first stage is more likely to land successfully.



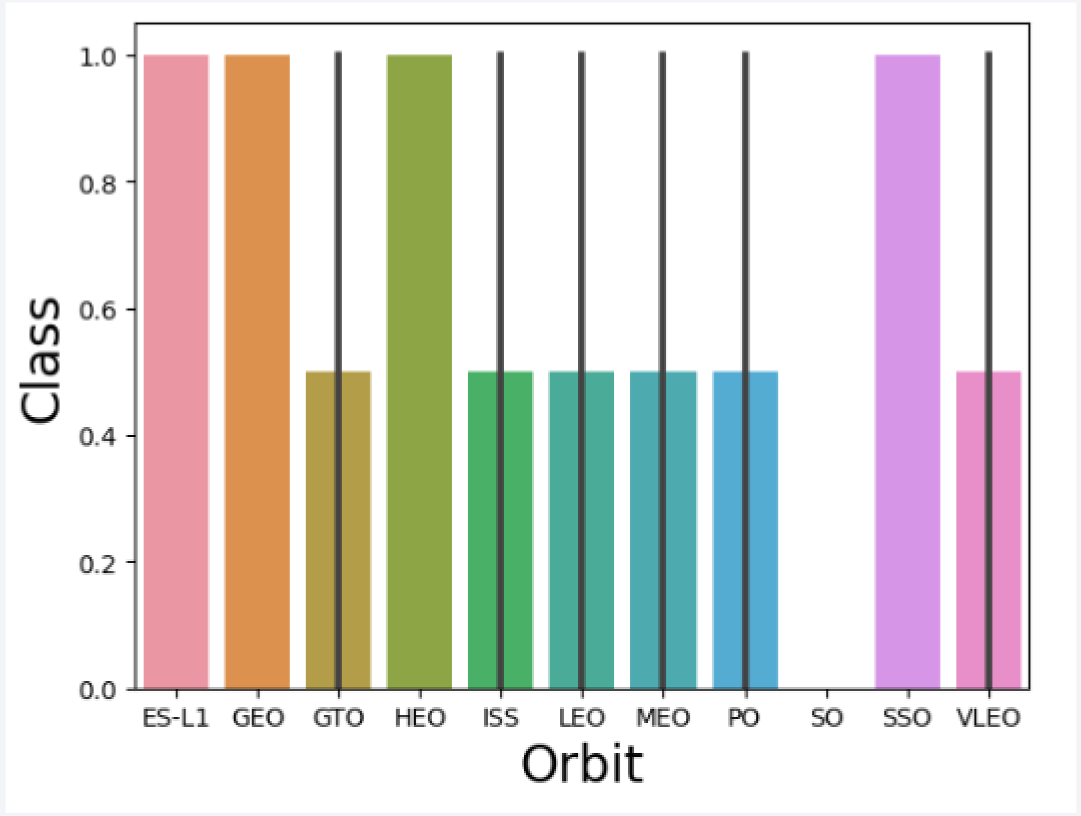
Payload vs. Launch Site

- For the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).



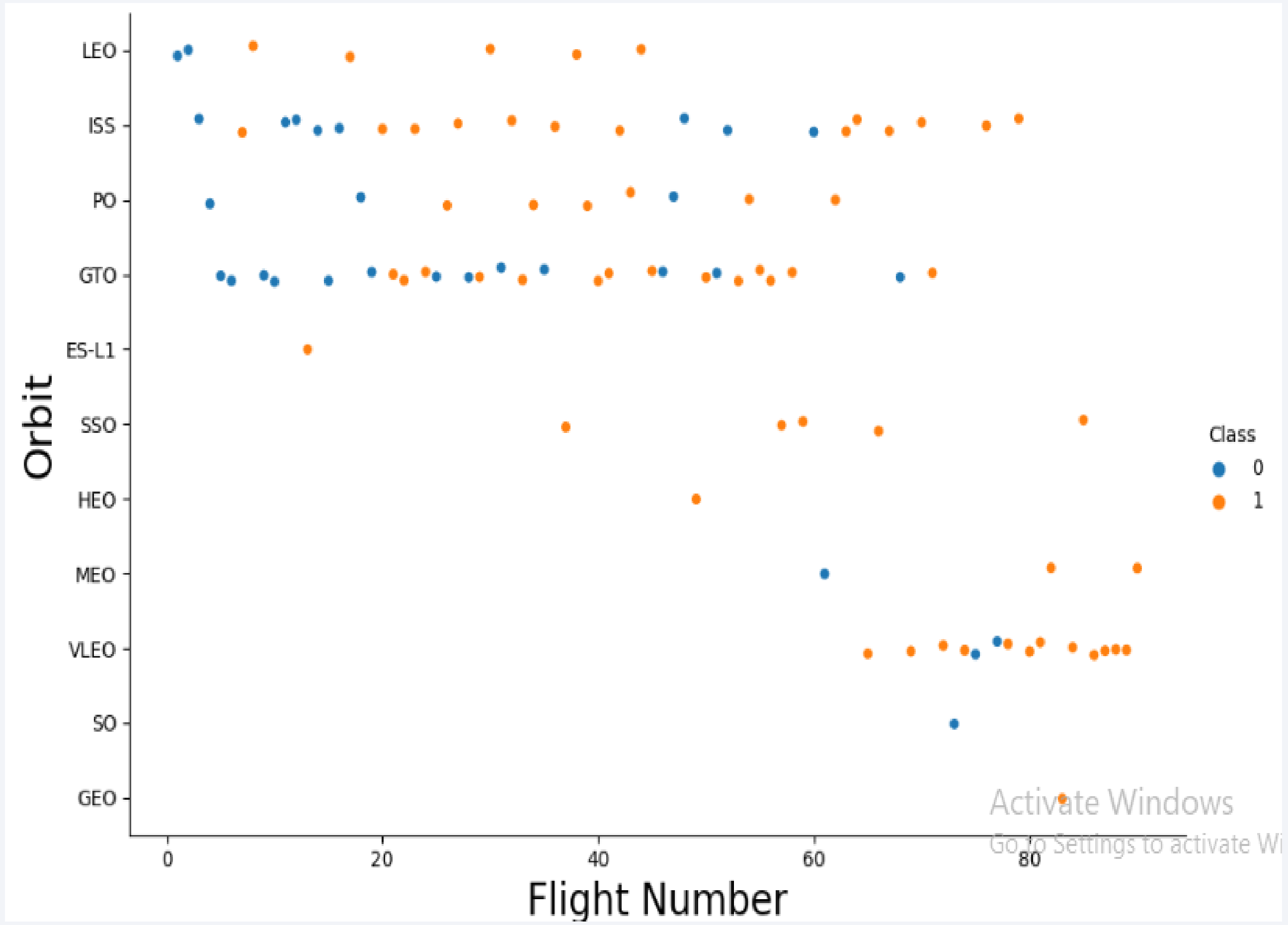
Success Rate vs. Orbit Type

- Some orbits like ES-L1, GEO, HEO, SSO have 100% success rates.



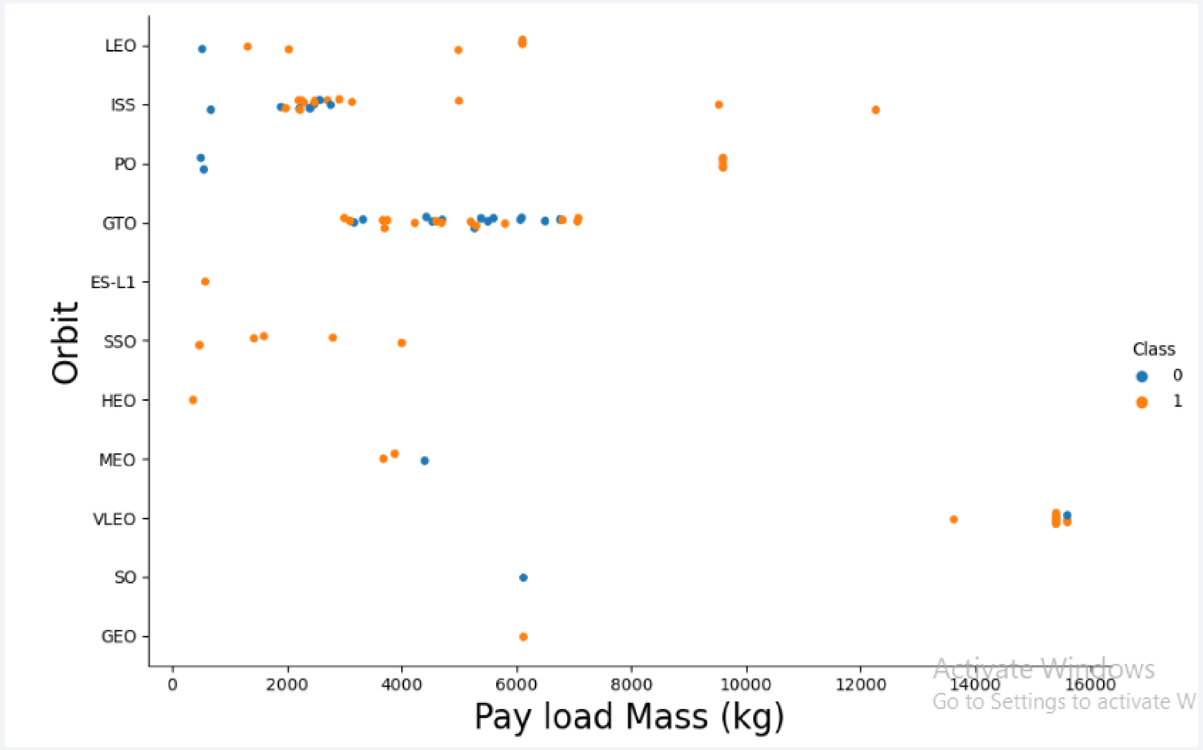
Flight Number vs. Orbit Type

- In the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



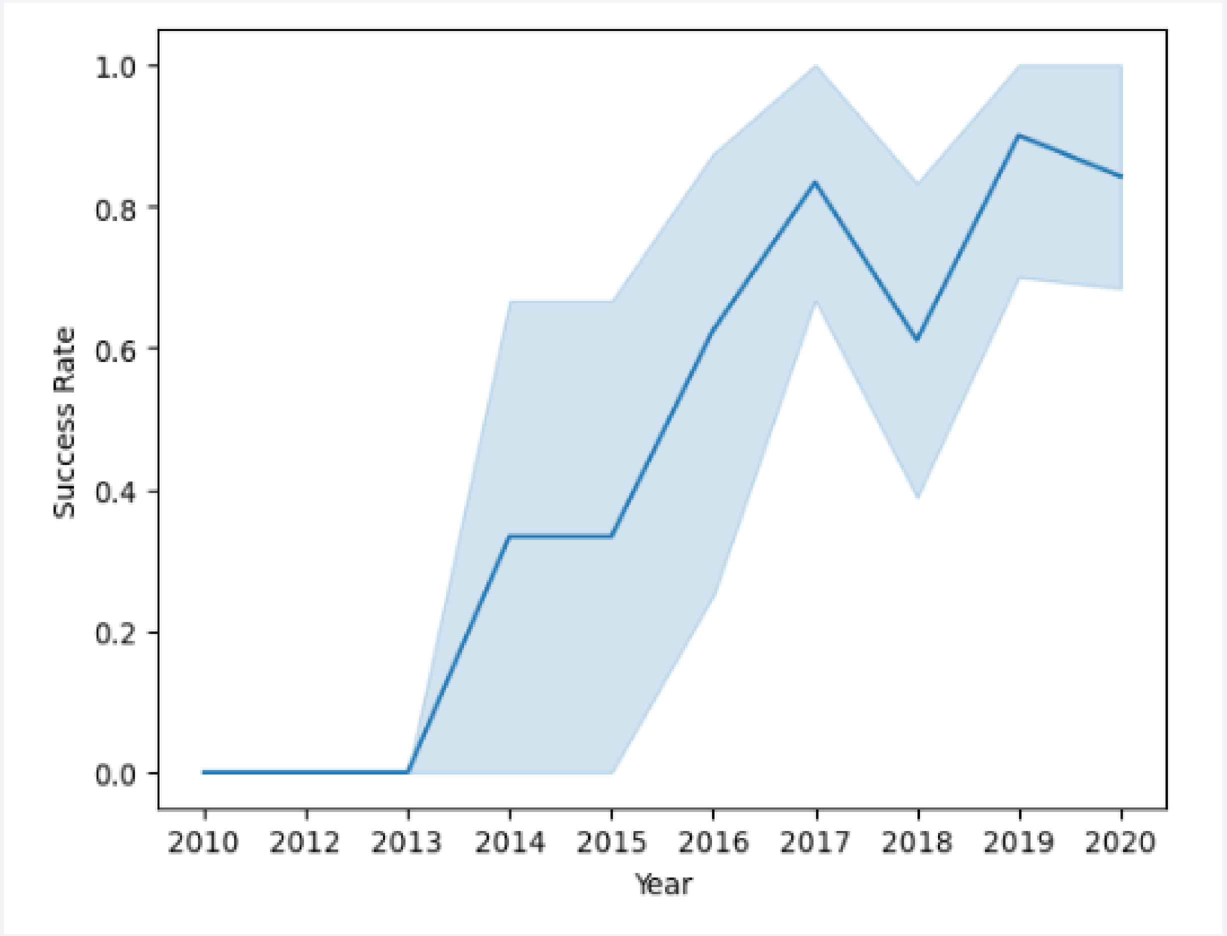
Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.



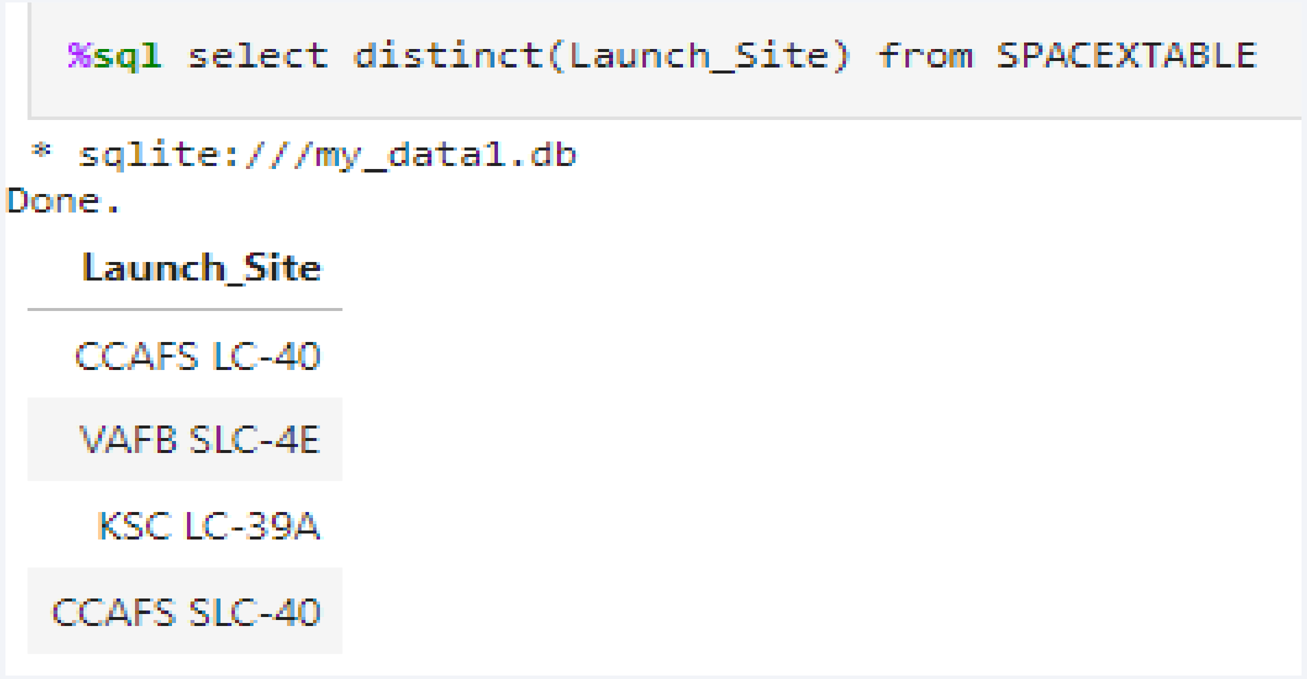
Launch Success Yearly Trend

- Launch Success rate is increasing since 2013



All Launch Site Names

- This is done using DISTINCT which selects only unique values from a mentioned attribute



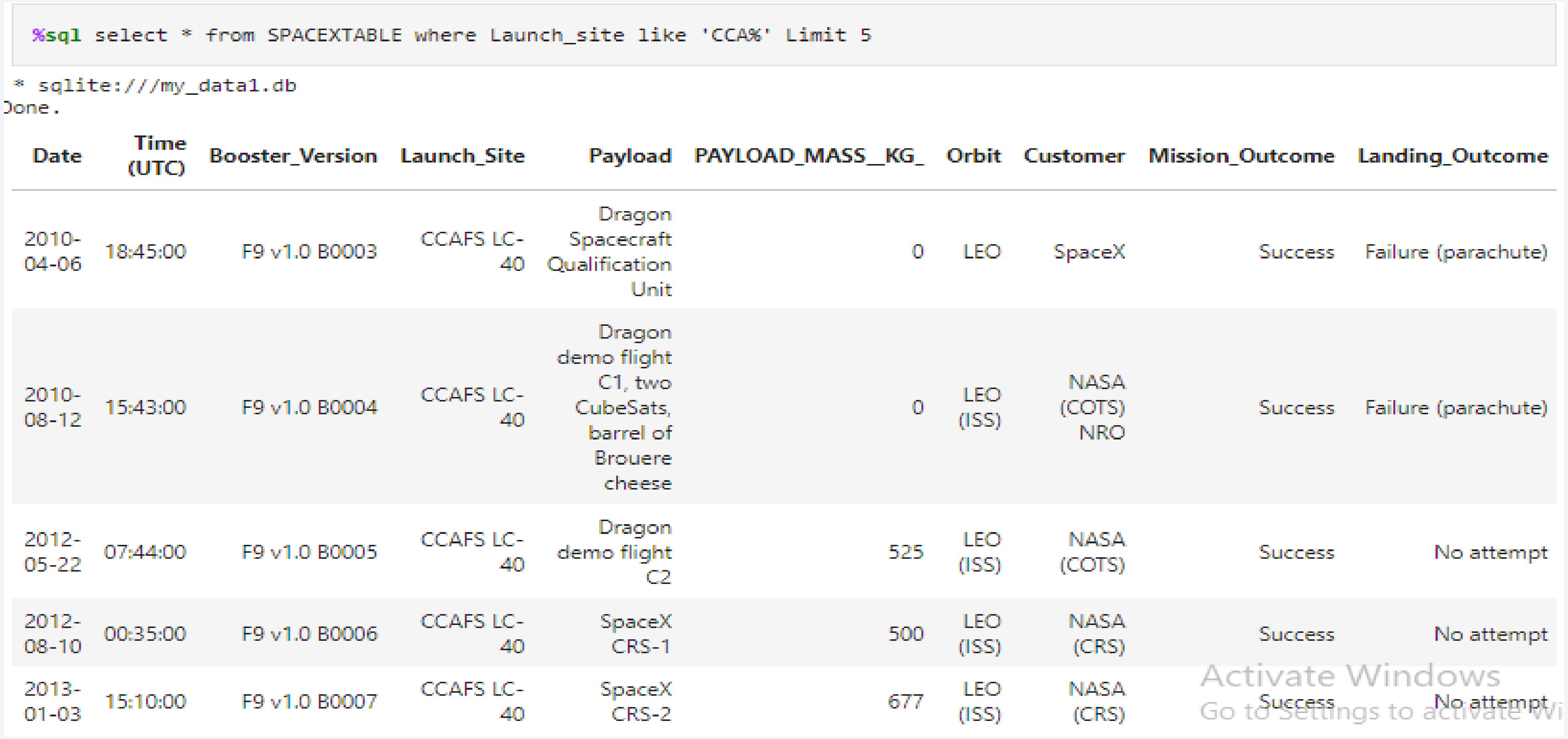
```
%sql select distinct(Launch_Site) from SPACEXTABLE
```

```
 * 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'

- This is done using wildcard character %



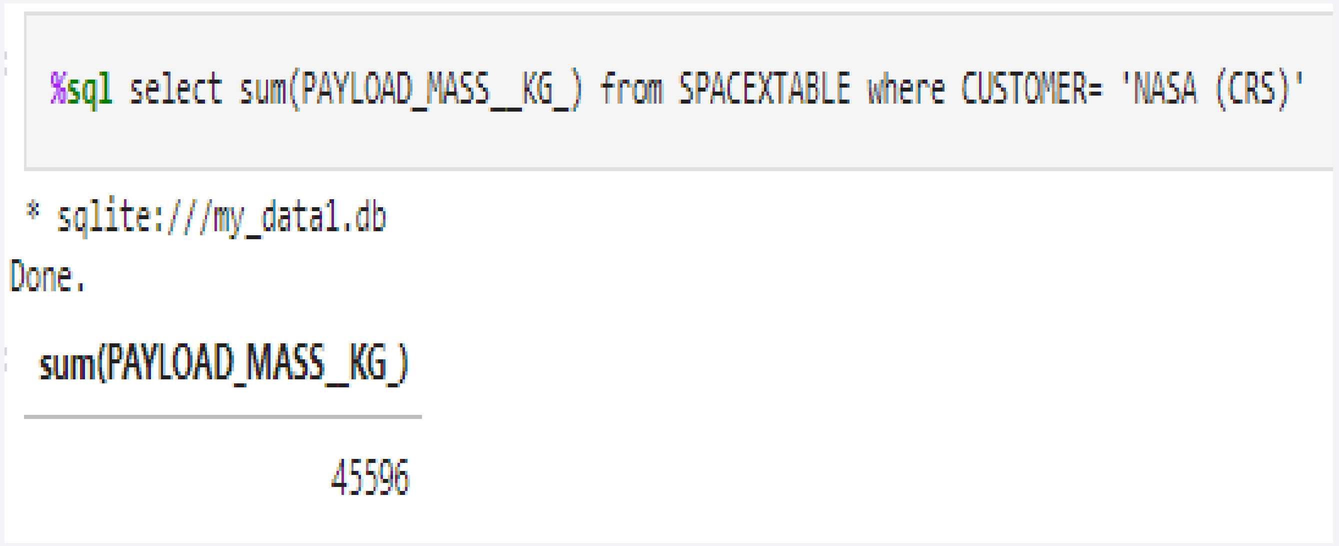
```
%sql select * from SPACEXTABLE where Launch_site like 'CCA%' Limit 5
```

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

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

Total Payload Mass

- This is done using aggregate function sum()



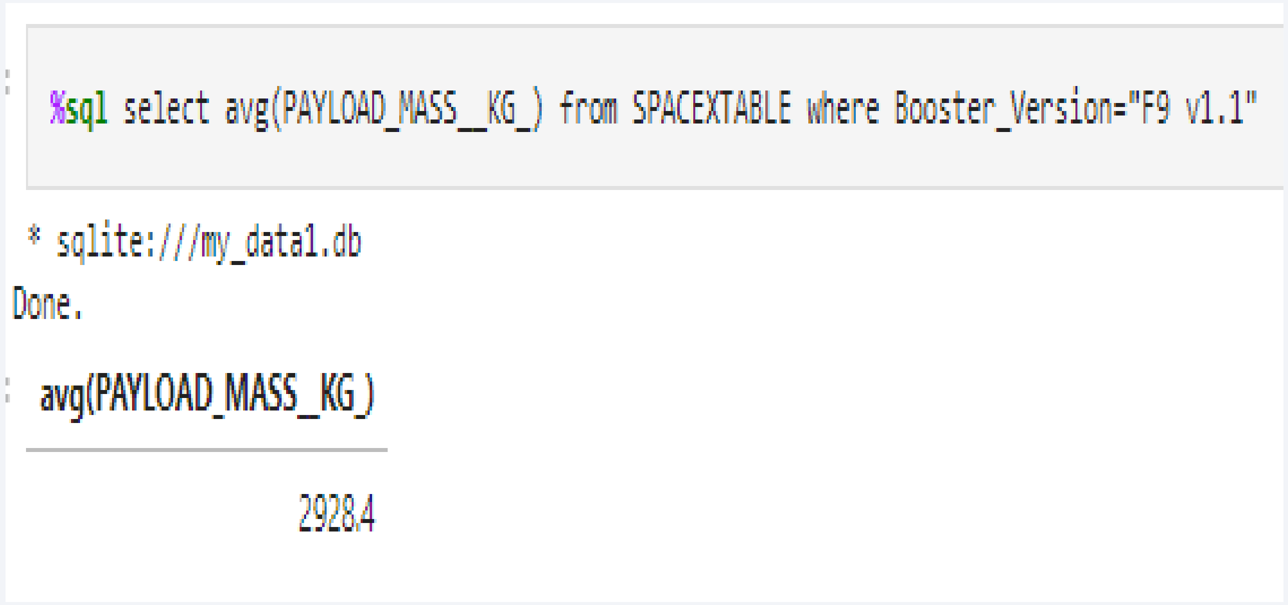
```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where CUSTOMER= 'NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- This is done using operator =



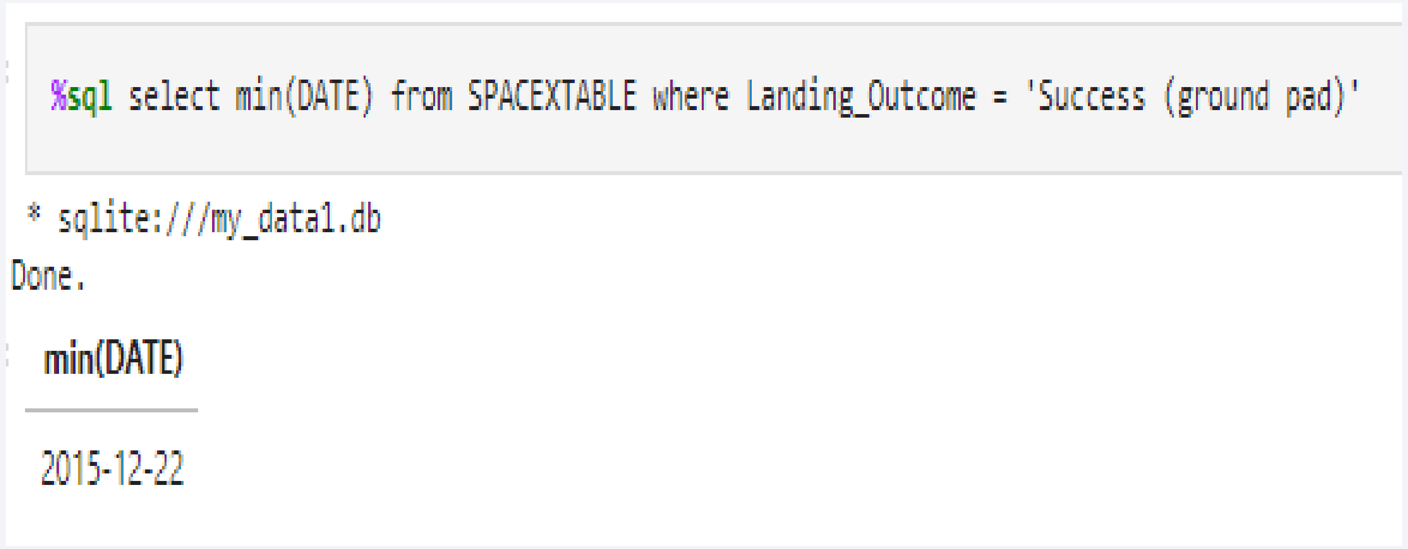
```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where Booster_Version="F9 v1.1"
```

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

```
avg(PAYLOAD_MASS__KG_)
                2928.4
```

First Successful Ground Landing Date

- This is done using min() function



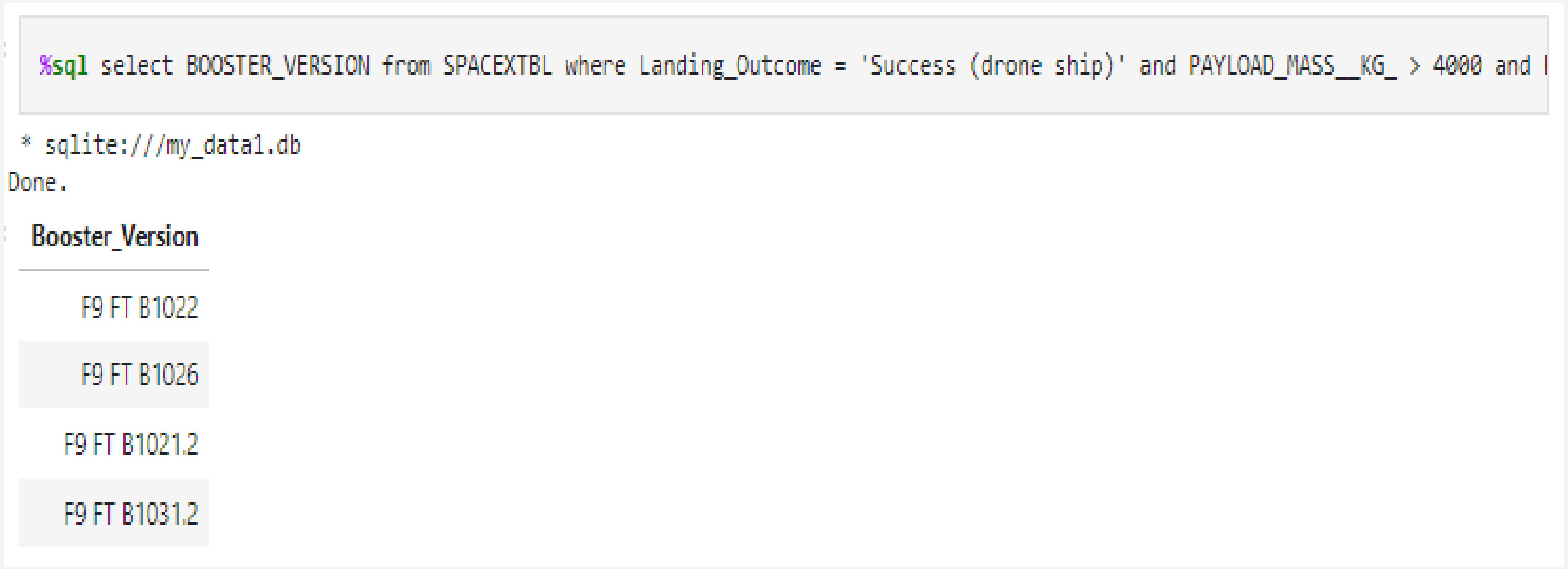
```
%sql select min(DATE) from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)'
```

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

min(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- This is done using combination of various operators



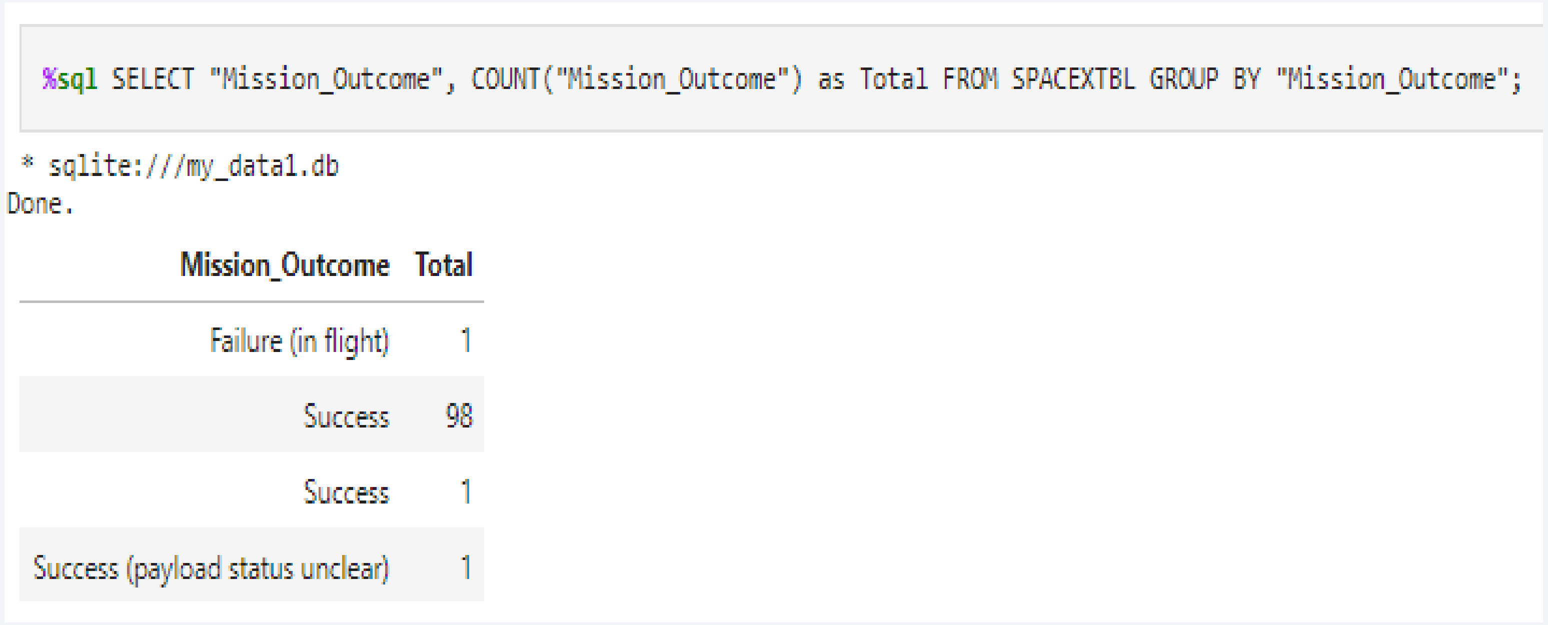
```
%sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and
```

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

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

Total Number of Successful and Failure Mission Outcomes

- Done using group by clause



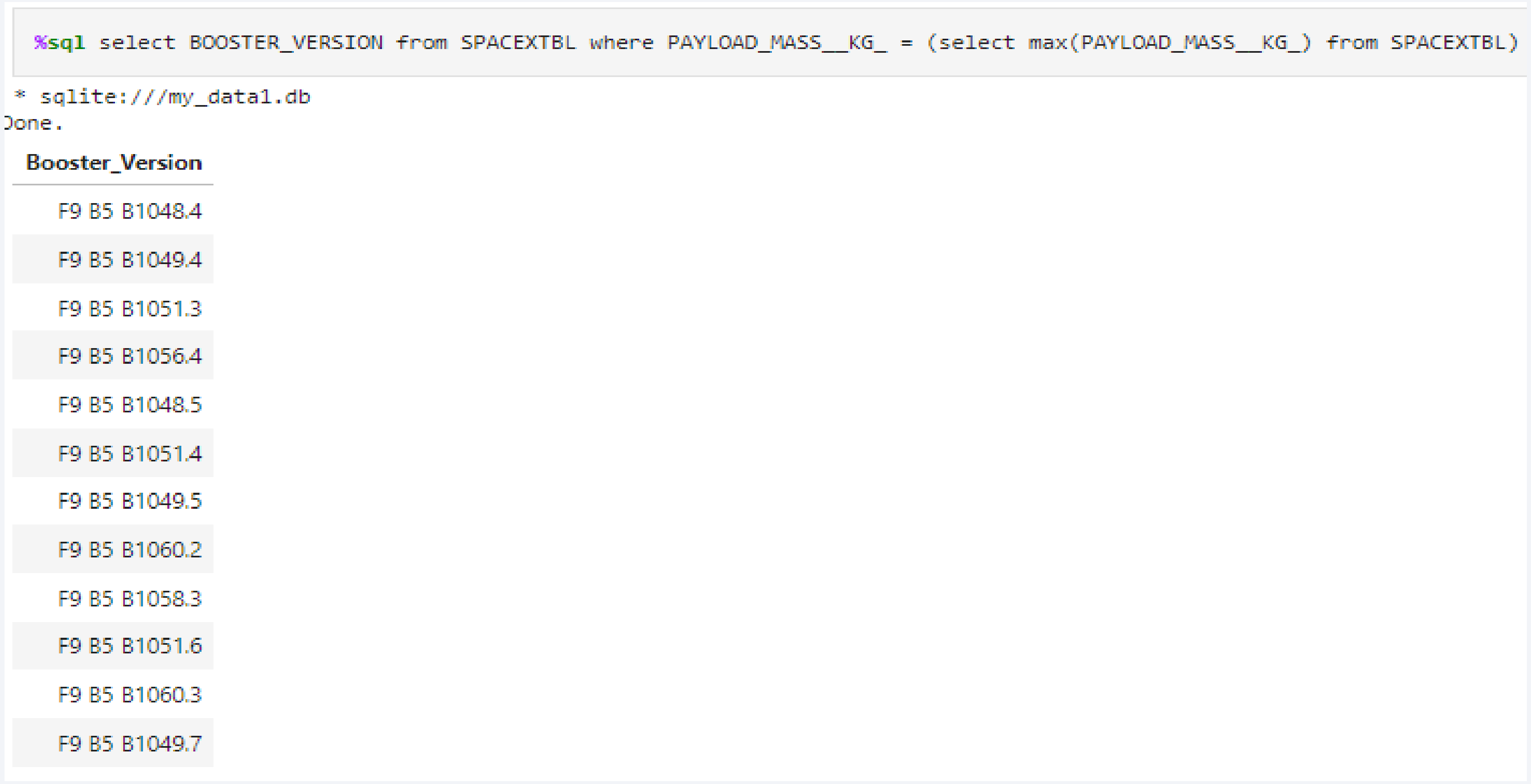
```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

Mission_Outcome	Total
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- Using nested query



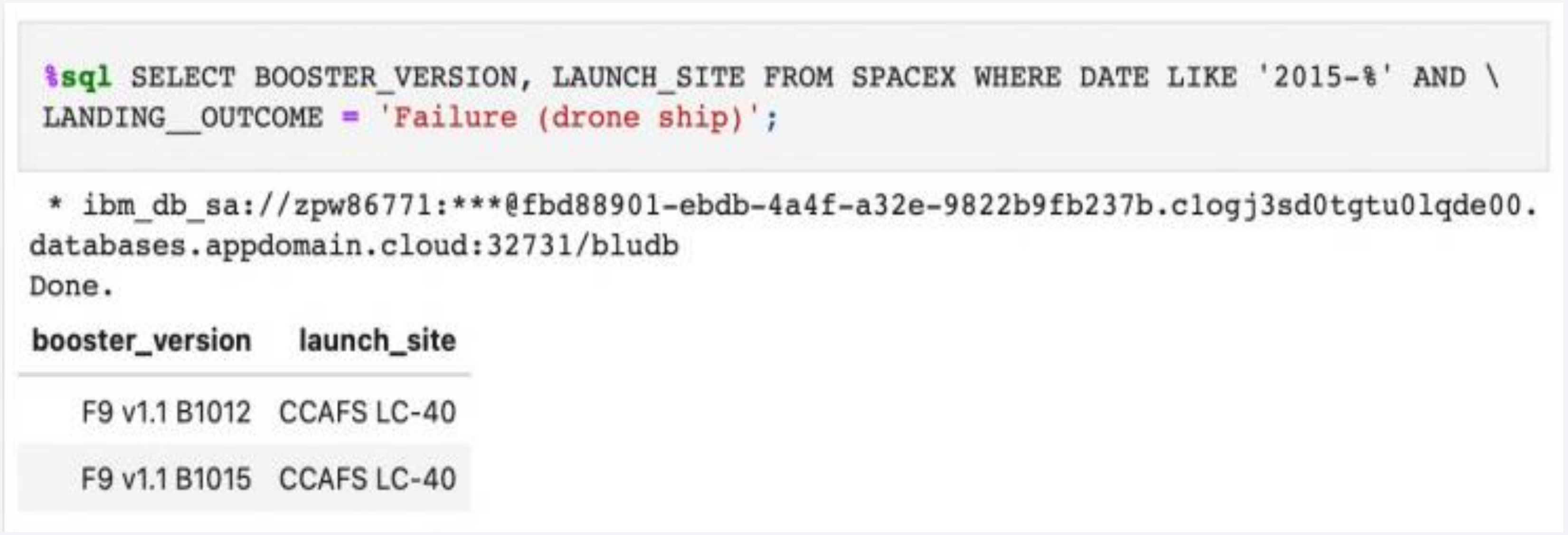
```
%sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

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

2015 Launch Records

- Using like, where, and ,etc.



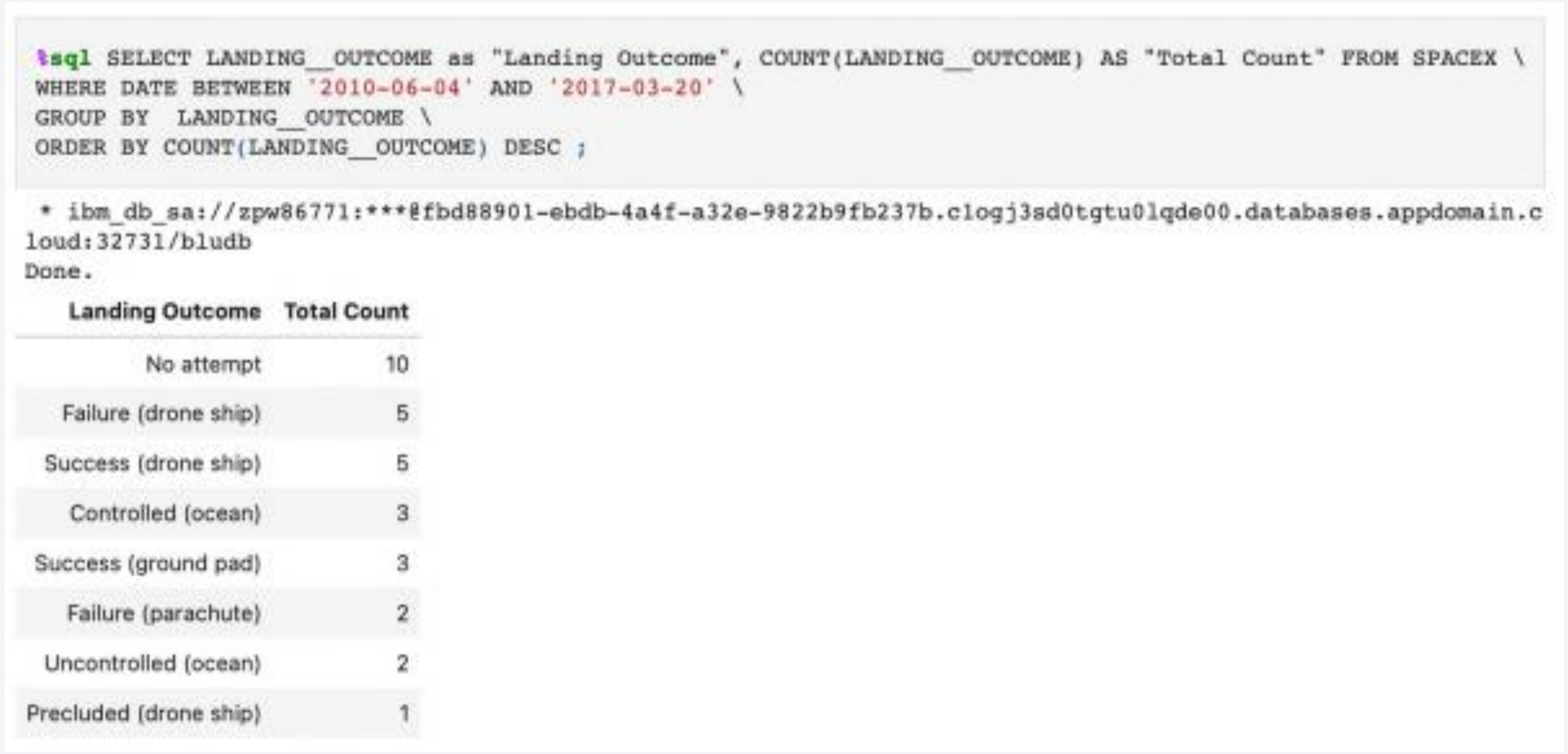
```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.
databases.appdomain.cloud:32731/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

- This is done using group by(),order by(),,and other queries together



```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEX \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING__OUTCOME \
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

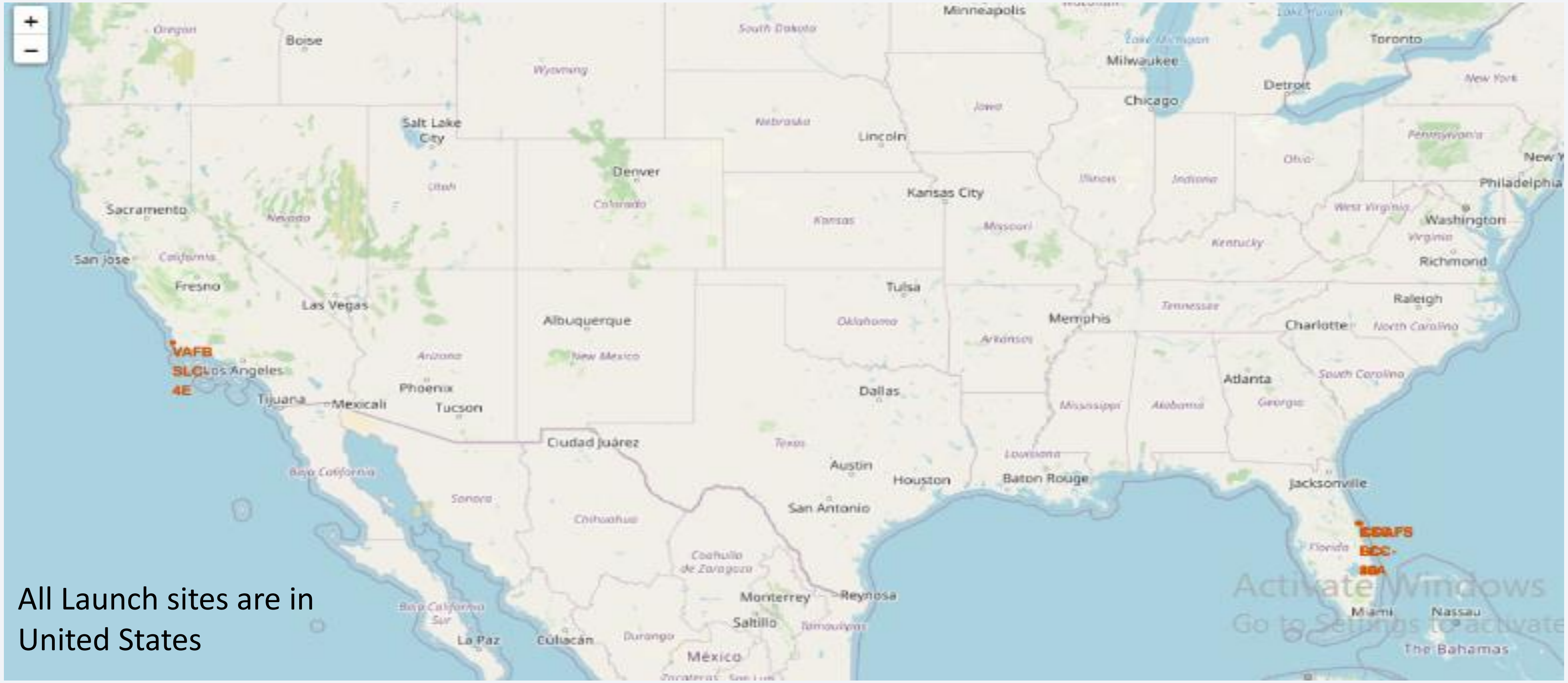
```
 * ibm_db_sa://zpw86771:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.c
loud:32731/bludb
Done.
```

Landing Outcome	Total 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

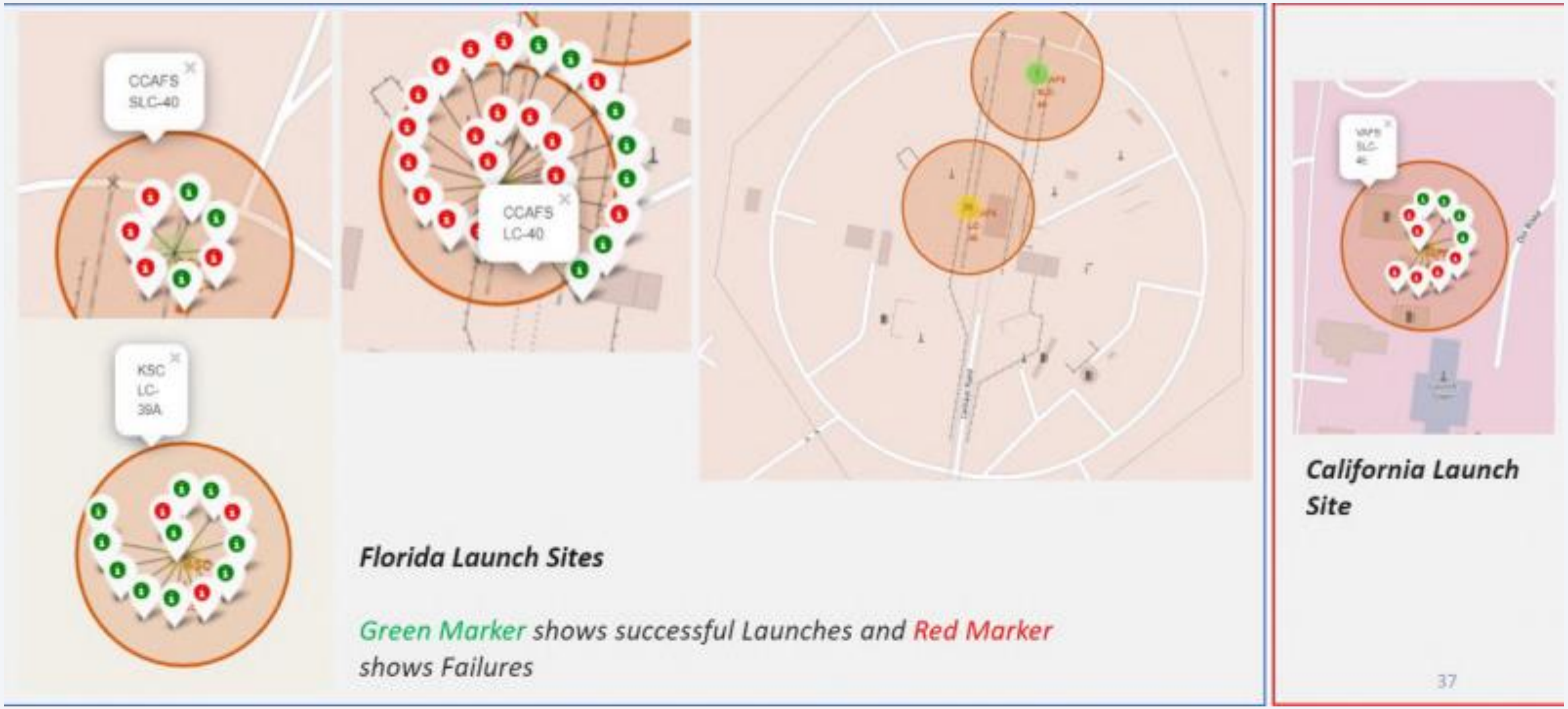
Launch Sites Proximities Analysis

Launch Sites Marker



All Launch sites are in United States

Launch Sites with Color Labels



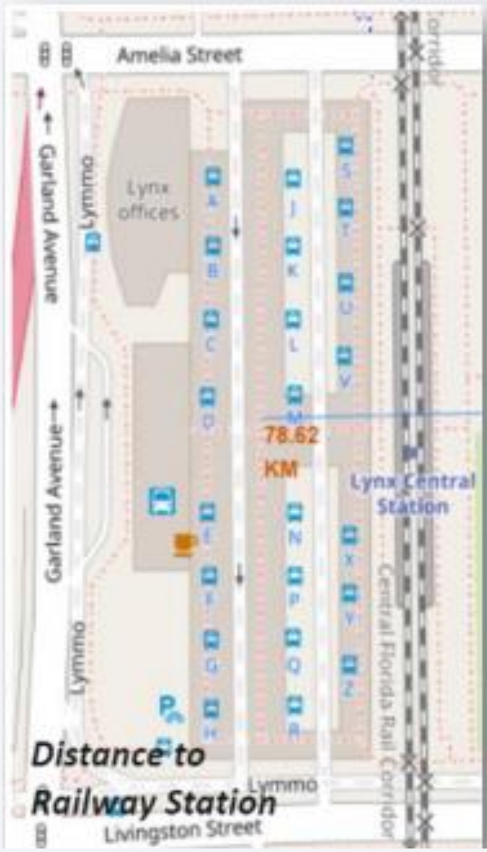
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

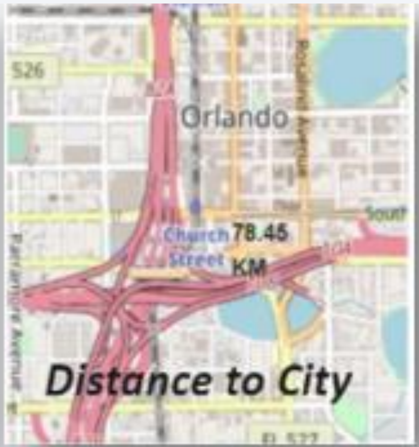
California Launch Site

37

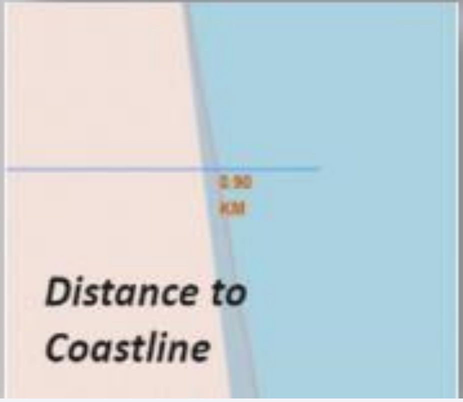
Distances from Launch sites



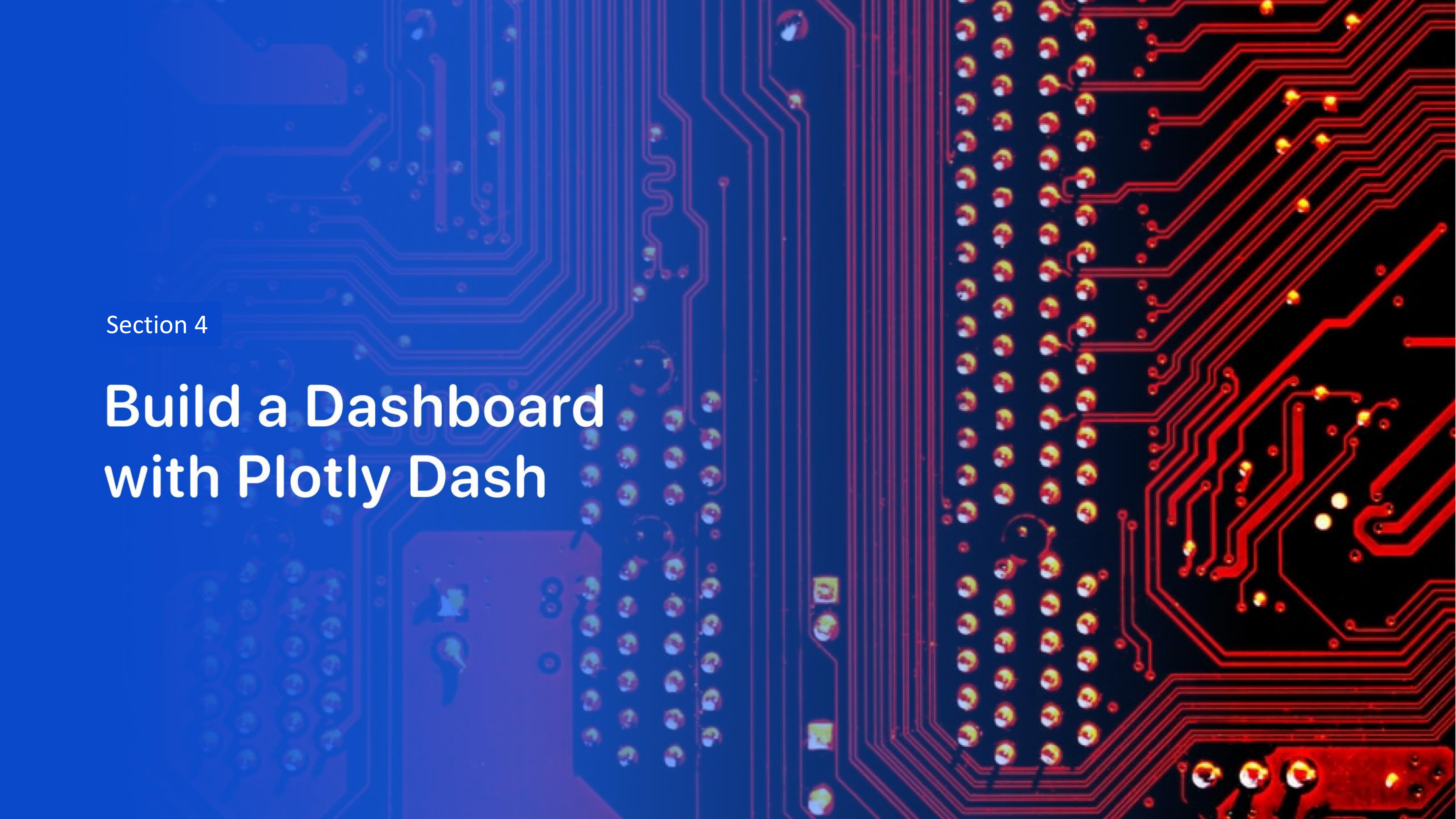
Distance to closest Highway



7
26
Distance to coast



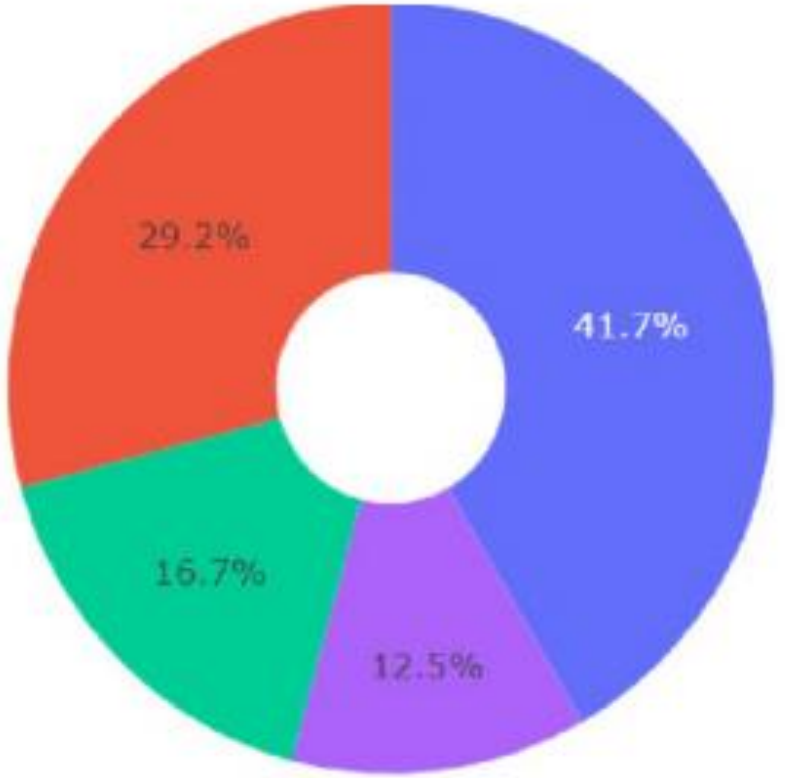
- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes



Section 4

Build a Dashboard with Plotly Dash

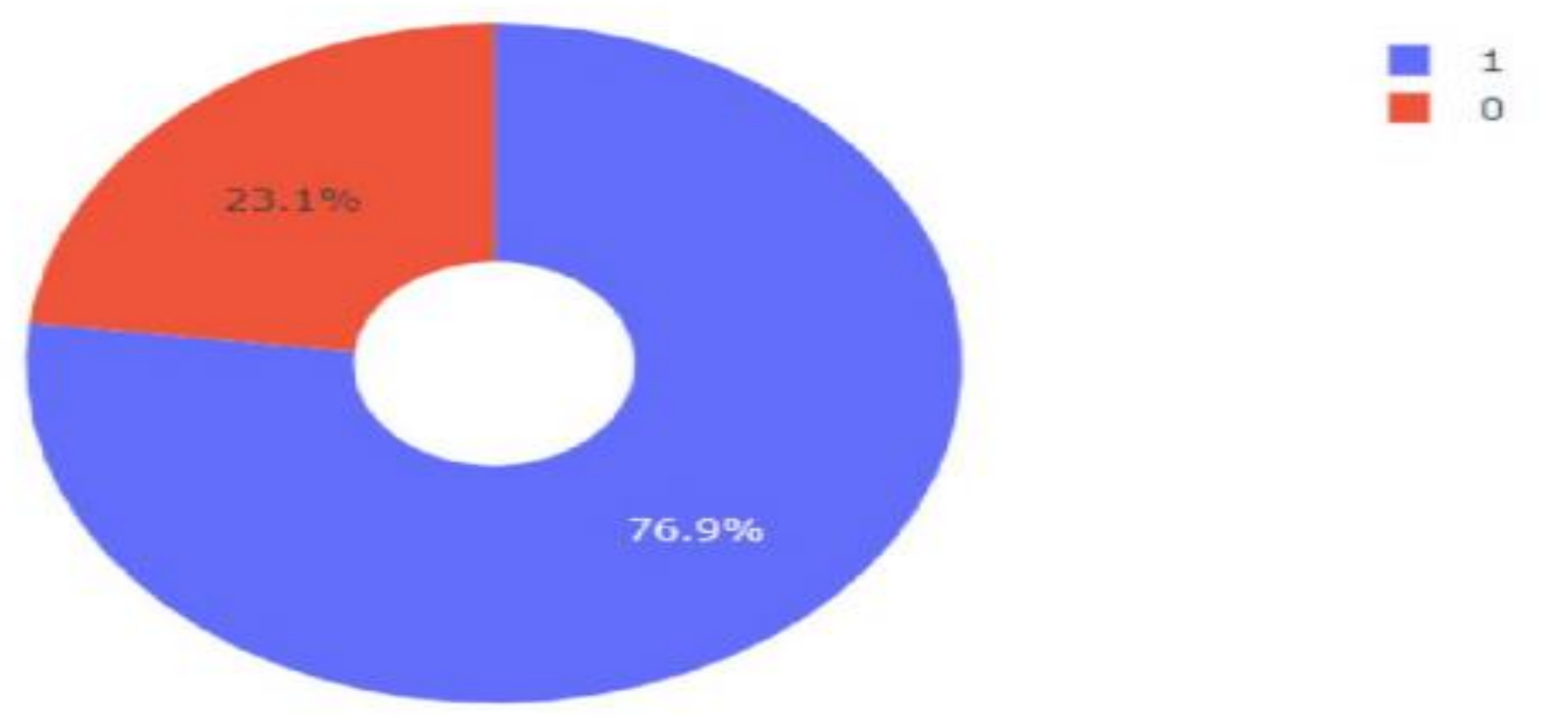
Success Percentage of Each Site



🟦 KSC LC-39A
🟥 CCAFS LC-40
🟩 VAFB SLC-4E
🟪 CCAFS SLC-40

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

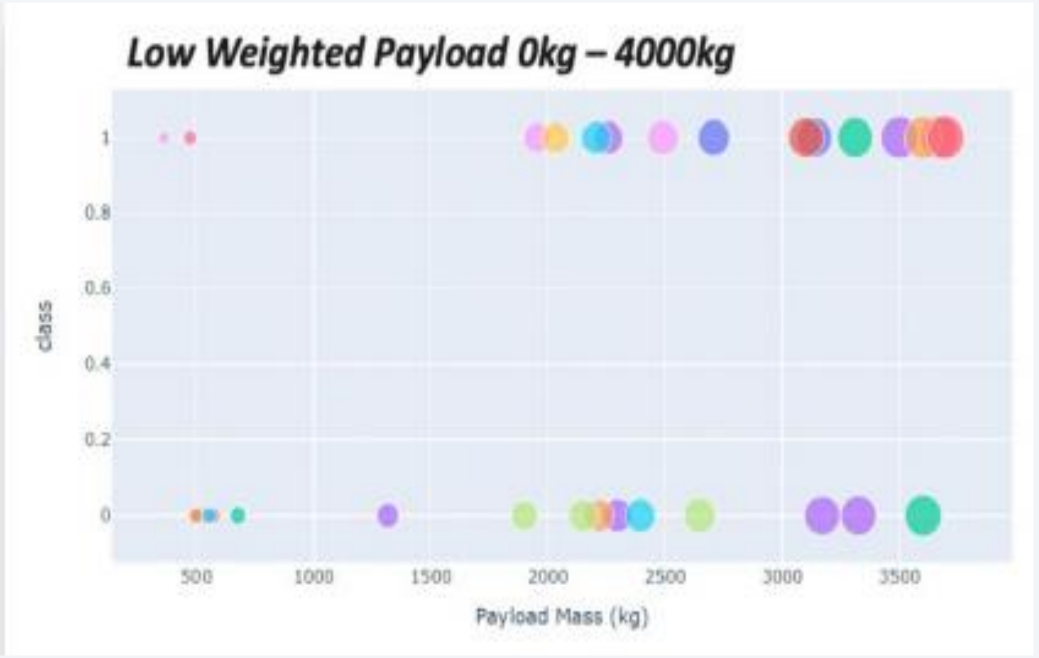
Launch Site with Highest success ratio

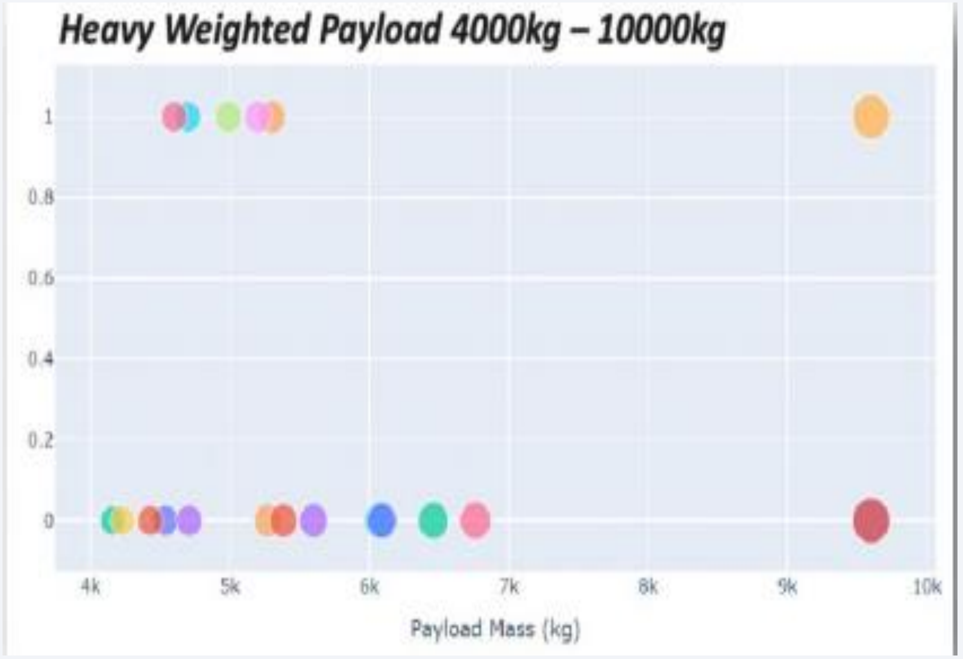


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs Launch

- Success Rate of Light weight Payloads is higher:



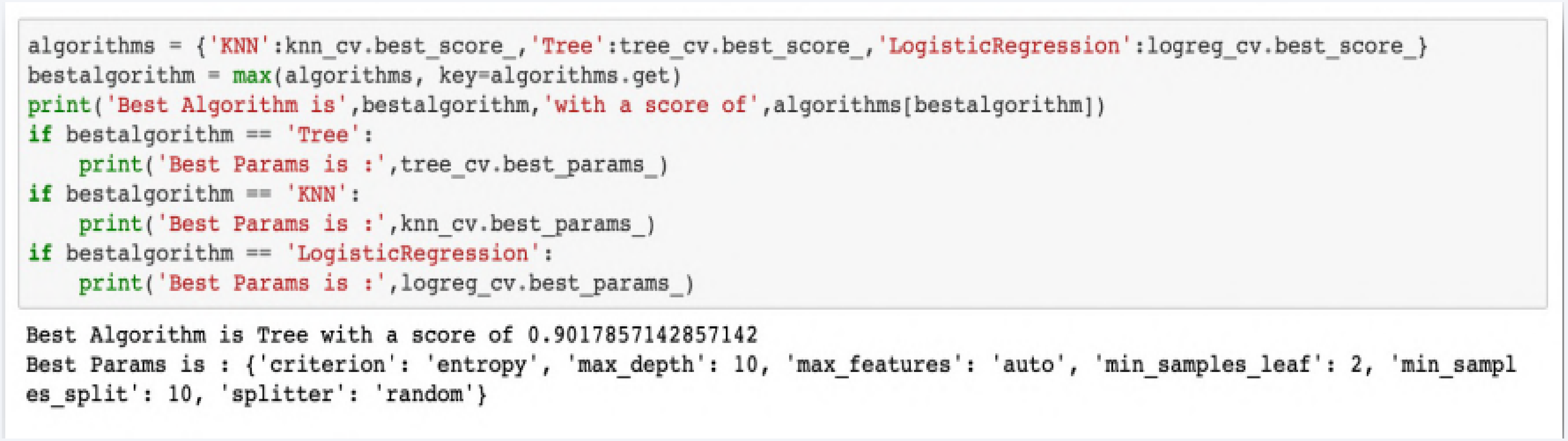


Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Using the below mentioned code, we found out that the best algorithm is Tree Algorithm.

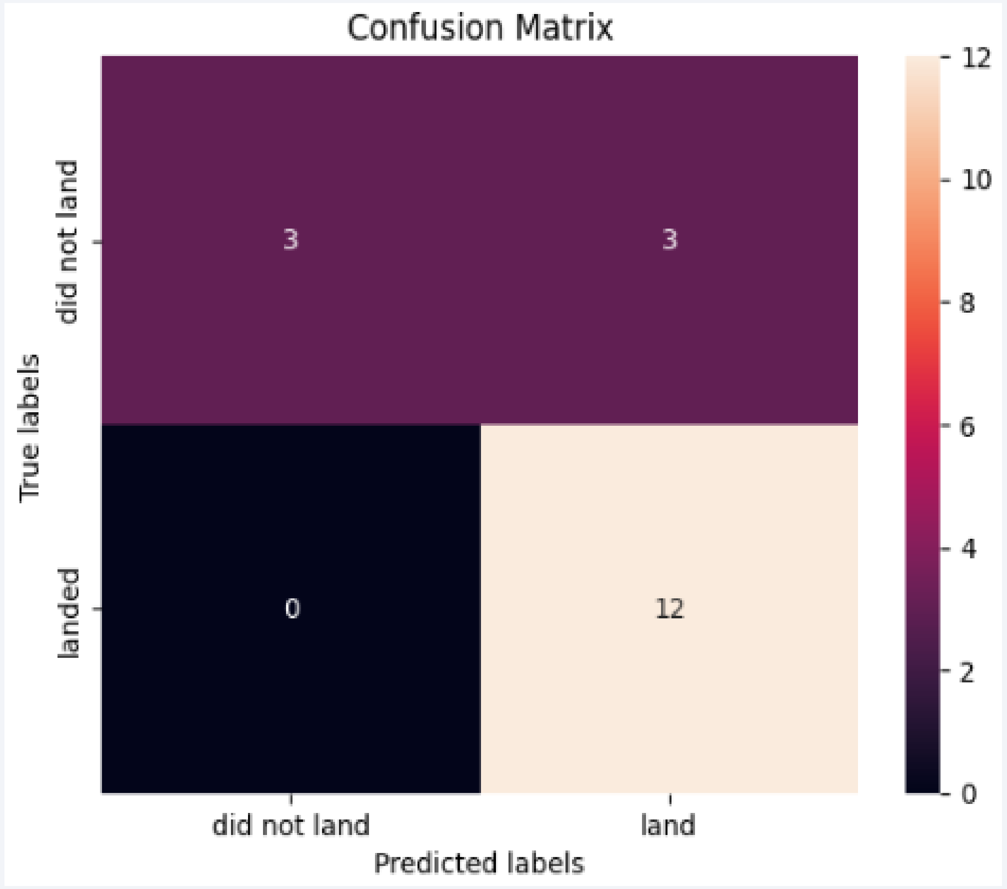


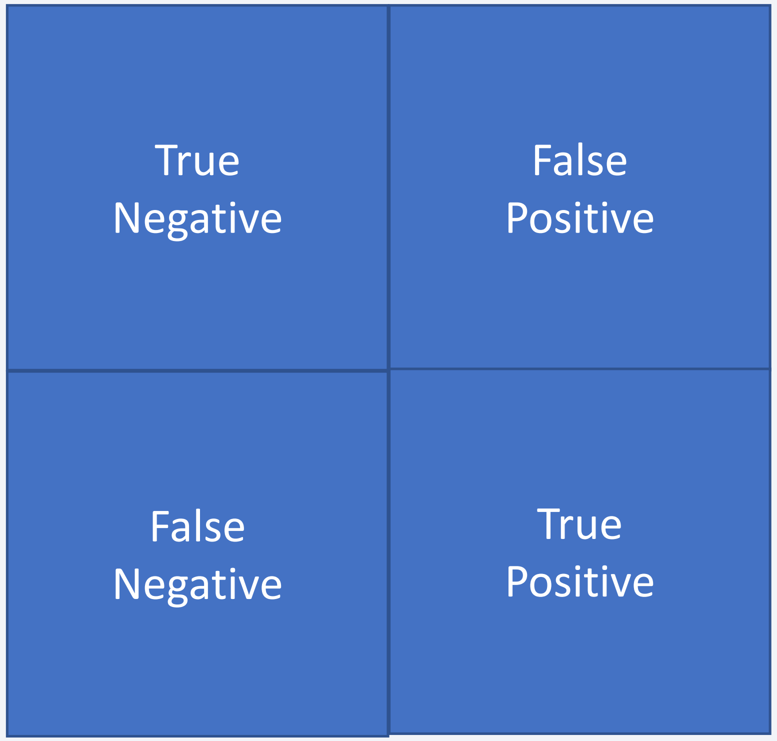
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params 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 0.9017857142857142
Best Params is : {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_sampl
es_split': 10, 'splitter': 'random'}
```

Confusion Matrix

- A confusion matrix shows the accuracy between real and predicted values.The confusion matrix of the Best model is:





Conclusions

We can conclude that:

- Low weighted payloads performed better than heavier ones!
- The success rate of SpaceX launches has increased from 2013.
- KSC LC-39A has most number of successful launches
- SSO orbits have the most success rate
- Best machine Learning approach: Classification Tree

Thank you!