IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ahad Khan
25-05-2024

Outline

Executive Summary

- Summary of methodologies
    - Data Collection through API and Web Scraping
    - Data Wrangling
    - Exploratory Data Analysis with SQL and Data Visualization
    - Interactive Visual Analysis with Folium
    - Machine Learning Predictions
- Summary of all results
    - Exploratory Data Analysis Result
    - Interactive analytics in screenshots
    - Predictive analytics result

Introduction

- Project background and context

SpaceX advertises Falcon9 rocket launches on its website with a cost of 62 ml dollars whereas other providers cost more than 165 ml dollars each, much of the saving is because SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. The goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Project background and context
  - 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:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

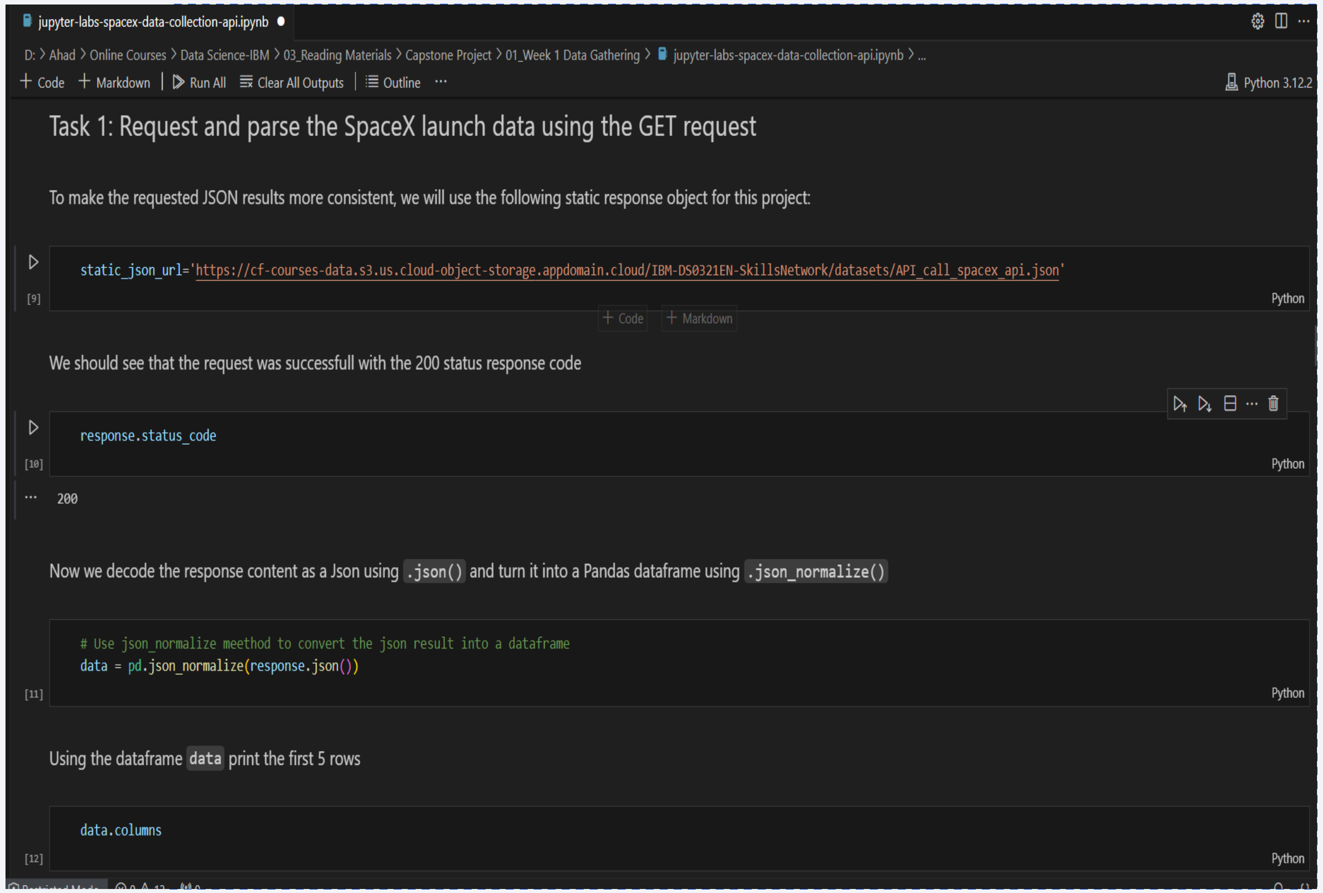
Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas data frame using .json_normalize().
  - We then cleaned the data, checked for missing values and fill in missing values where necessary.
  - In addition, 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 data Frame for future analysis.

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

- Link for the Lab: [Click here]()



jupyter-labs-spacex-data-collection-api.ipynb
D: > Ahad > Online Courses > Data Science-IBM > 03_Reading Materials > Capstone Project > 01_Week 1 Data Gathering > jupyter-labs-spacex-data-collection-api.ipynb > ...
+ Code + Markdown | Run All Clear All Outputs | Outline ...
Python 3.12.2

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
[9]
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

We should see that the request was successfull with the 200 status response code

```
[10]
response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

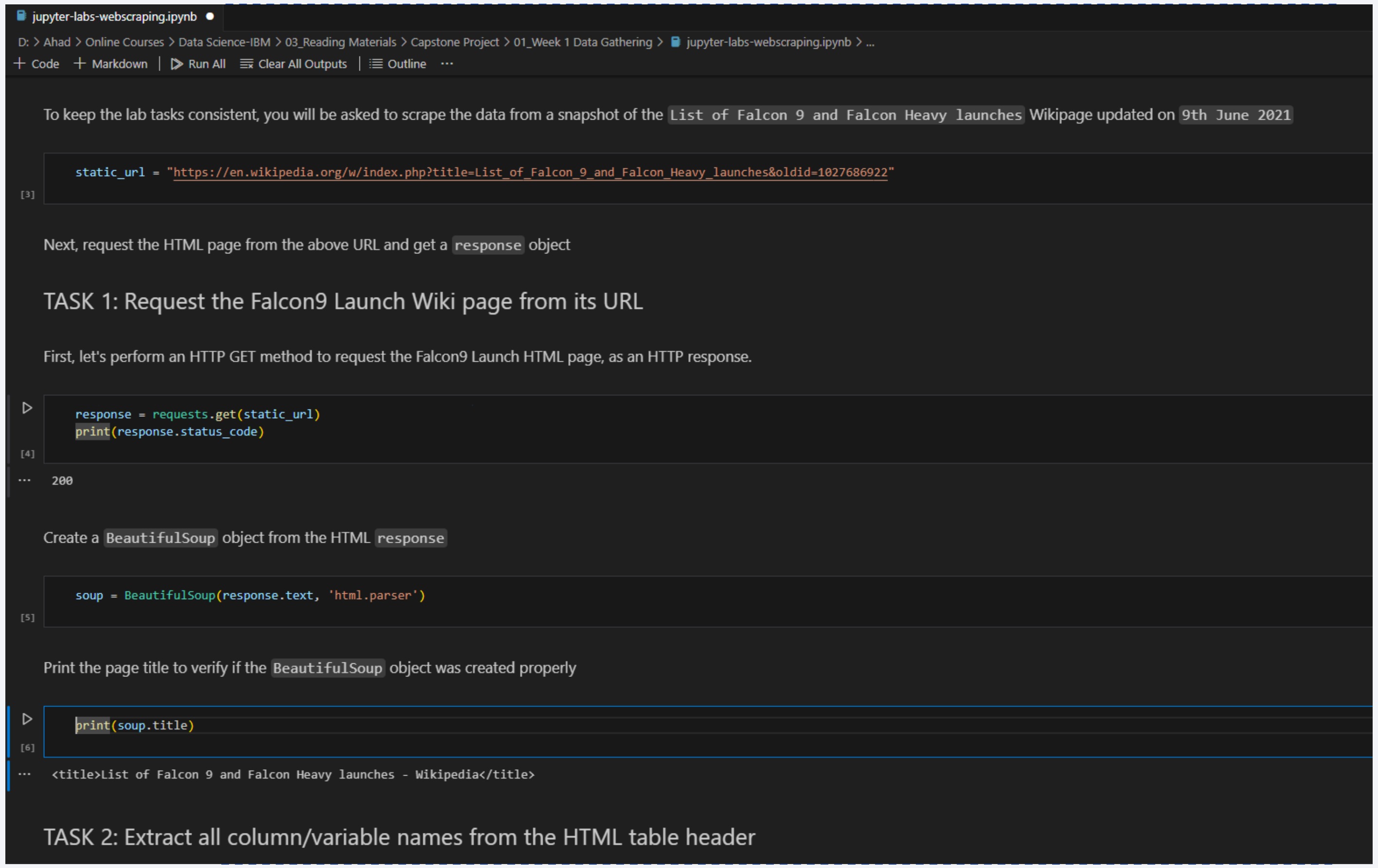
```
[11]
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
[12]
data.columns
```

Data Collection - Scraping

- We applied web scrapping to web scrape Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas data frame
- Link for the Lab



To keep the lab tasks consistent, you will be asked to scrape the data from a snapshot of the `List of Falcon 9 and Falcon Heavy launches` Wikipage updated on `9th June 2021`

```
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.

```
response = requests.get(static_url)
print(response.status_code)
```

```
200
```

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

```
soup = BeautifulSoup(response.text, 'html.parser')
```

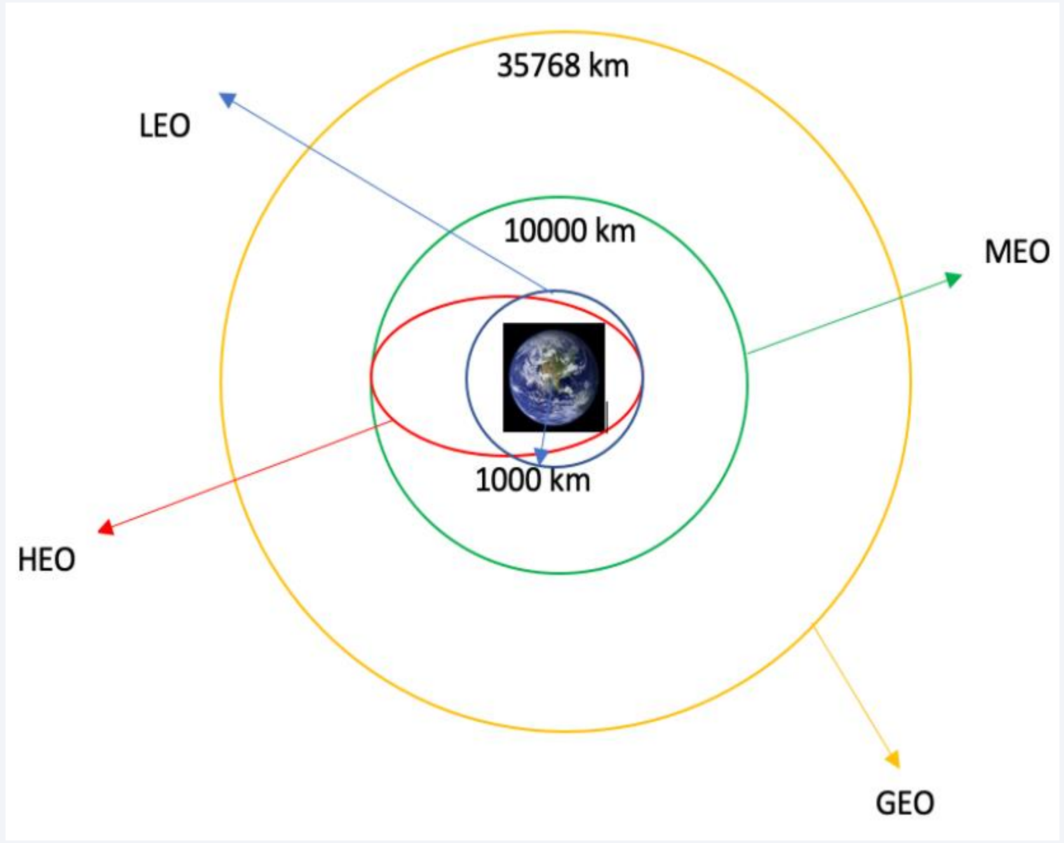
Print the page title to verify if the `BeautifulSoup` object was created properly

```
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

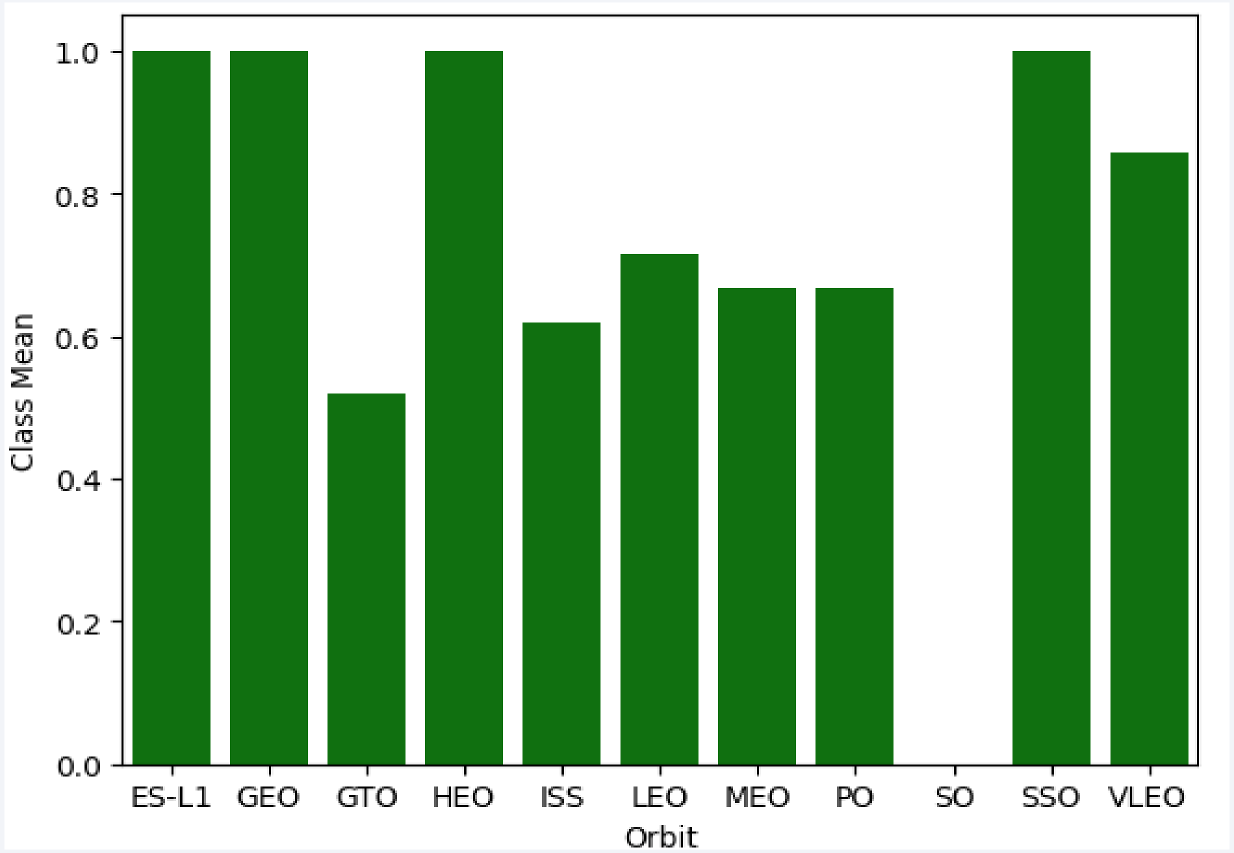
Data Wrangling



- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook

EDA with Data Visualization

- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.
- Link for the Lab



EDA with SQL

- We loaded the SpaceX dataset into a SQL Lite database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
    - The names of unique launch sites in the space mission.
    - The total payload mass carried by boosters launched by NASA (CRS)
    - The average payload mass carried by booster version F9 v1.1
    - The total number of successful and failure mission outcomes
    - The failed landing outcomes in drone ship, their booster version and launch site names.
- Link for the Lab

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
    - Are launch sites near railways, highways and coastlines.
    - Do launch sites keep certain distance away from cities.

Build a Dashboard with Plotly Dash

- 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.
- The link to the Lab

Predictive Analysis (Classification)

- We loaded the data using NumPy and Pandas, transformed the data, split our data into training and testing with Scikit-Learn.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- Link for the Lab

Results

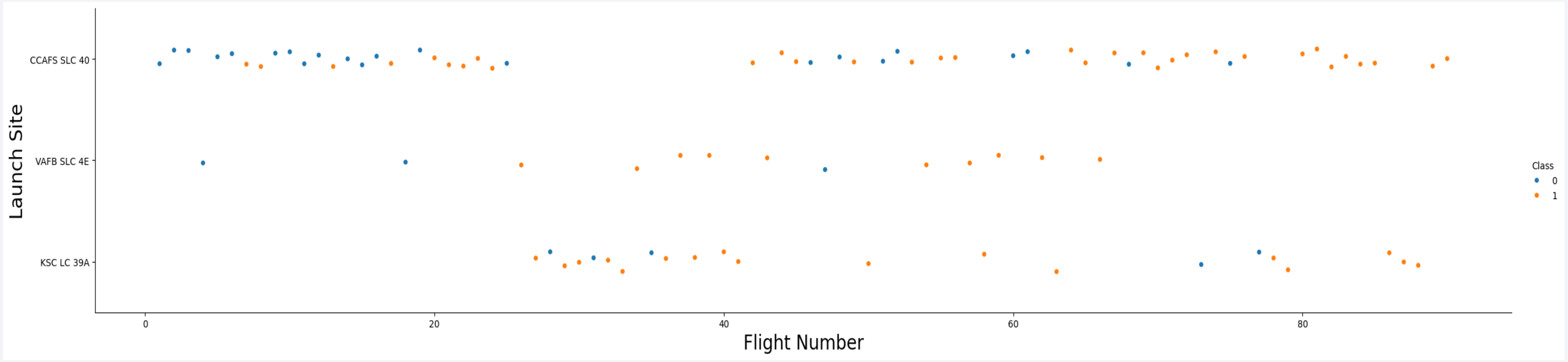
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

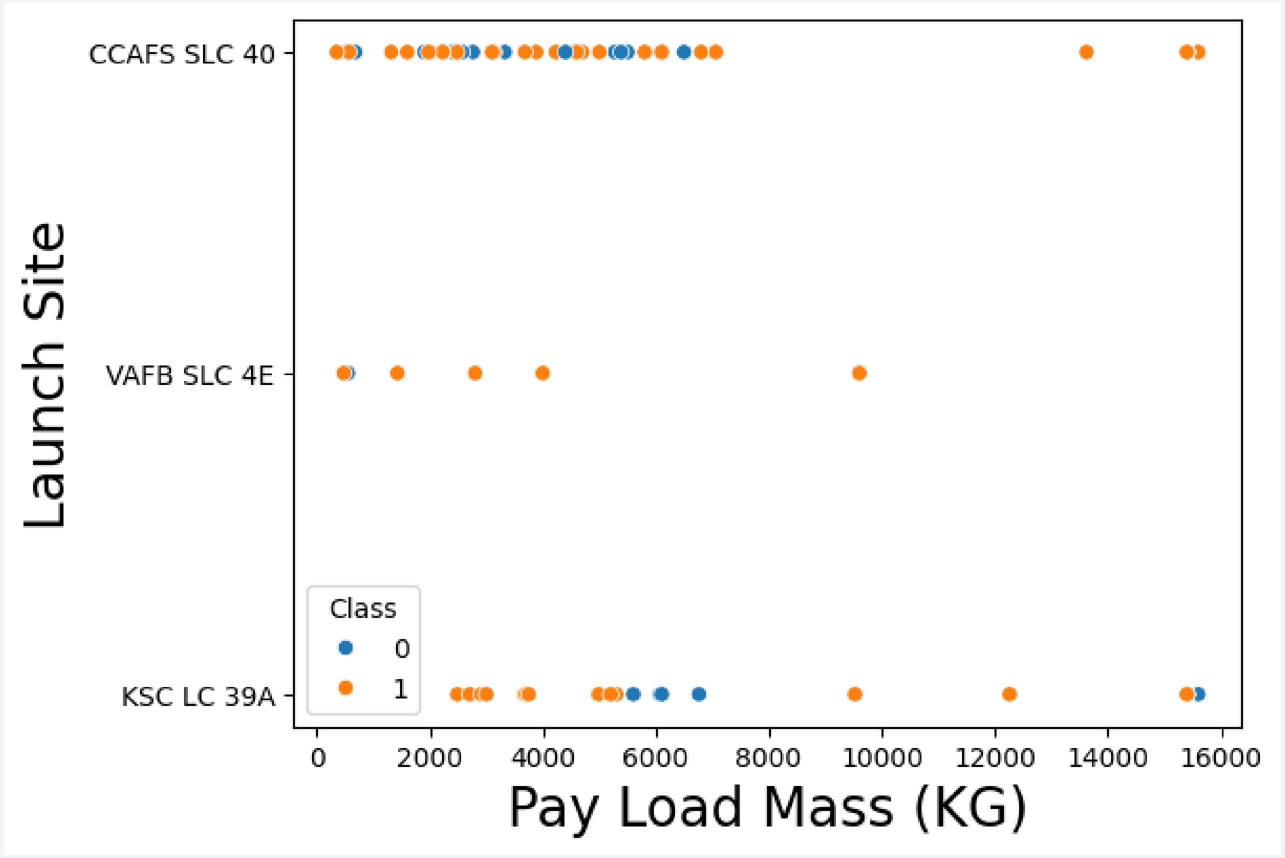
Flight Number vs. Launch Site

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



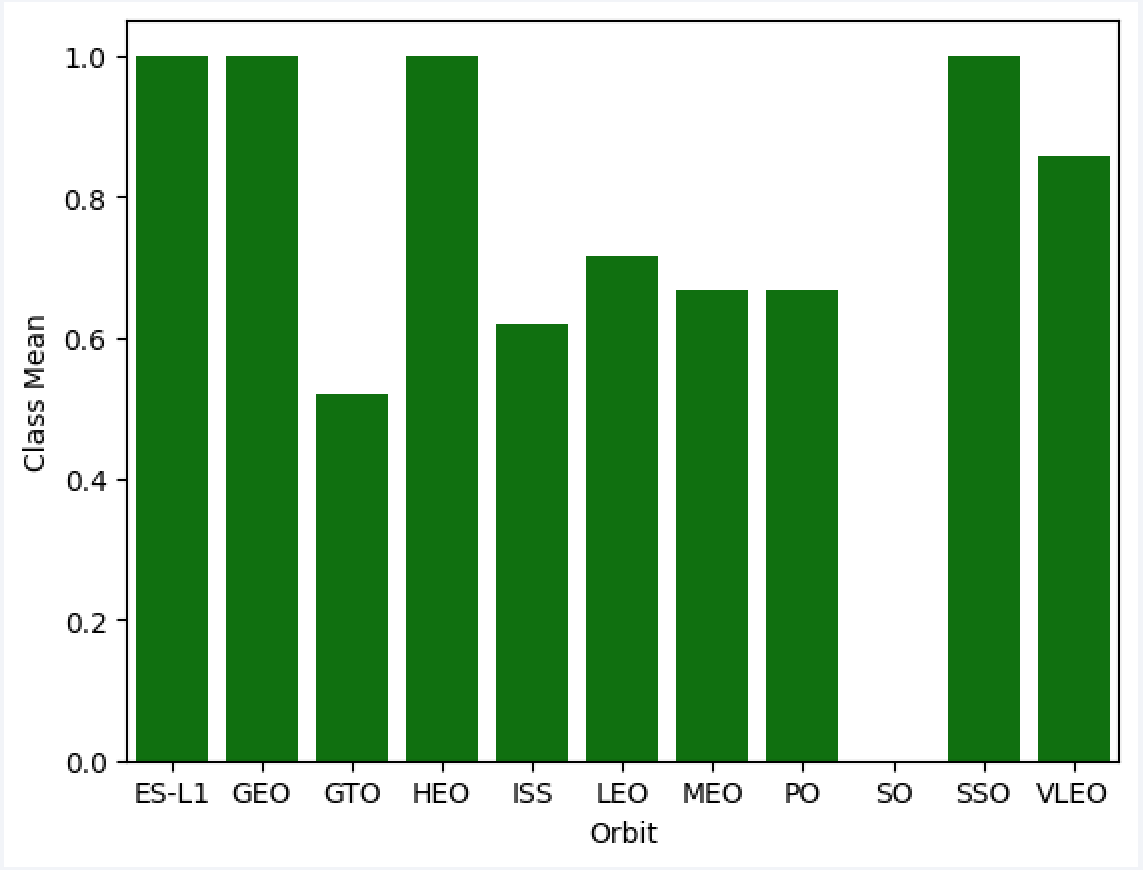
Payload vs. Launch Site

- The greater the Payload mass for the launch site, the higher the success rate of the rocket



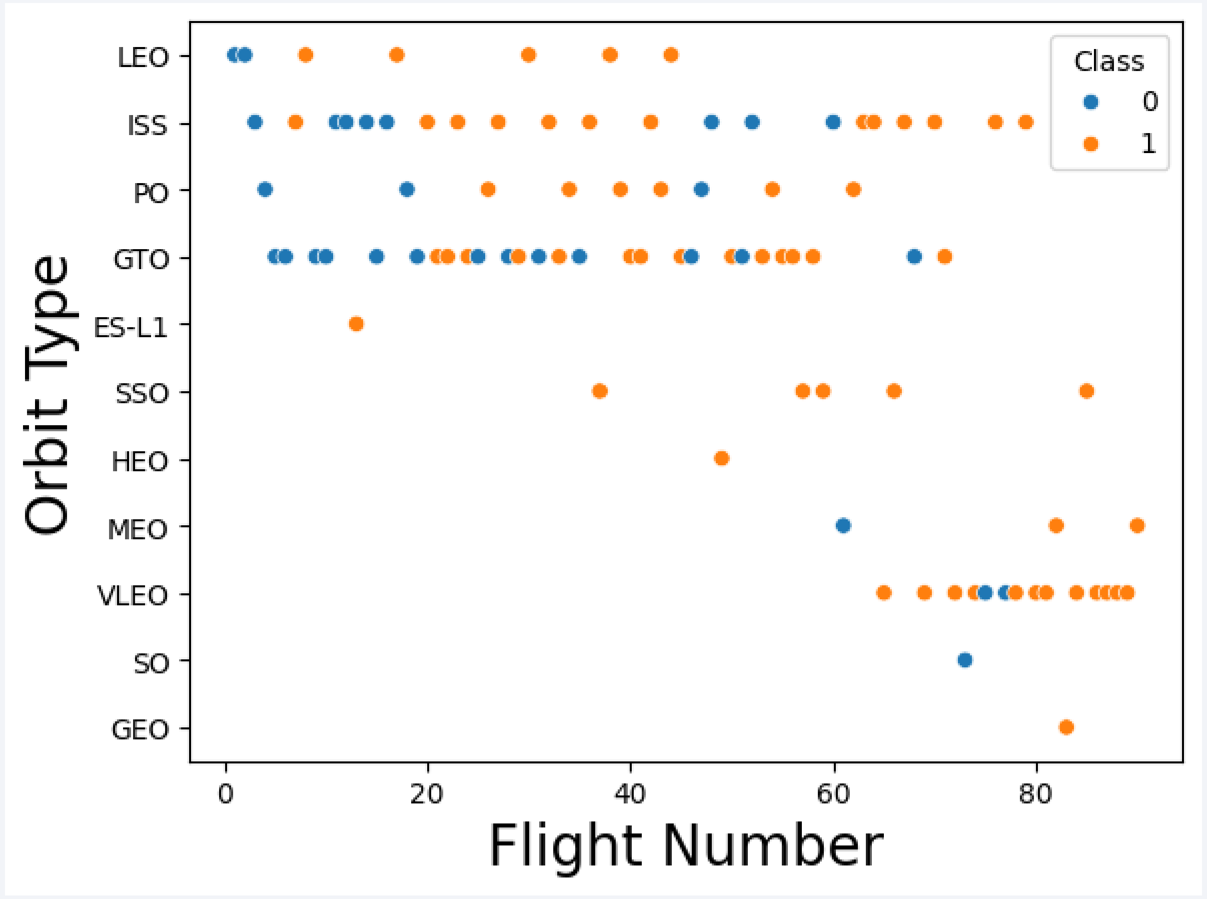
Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



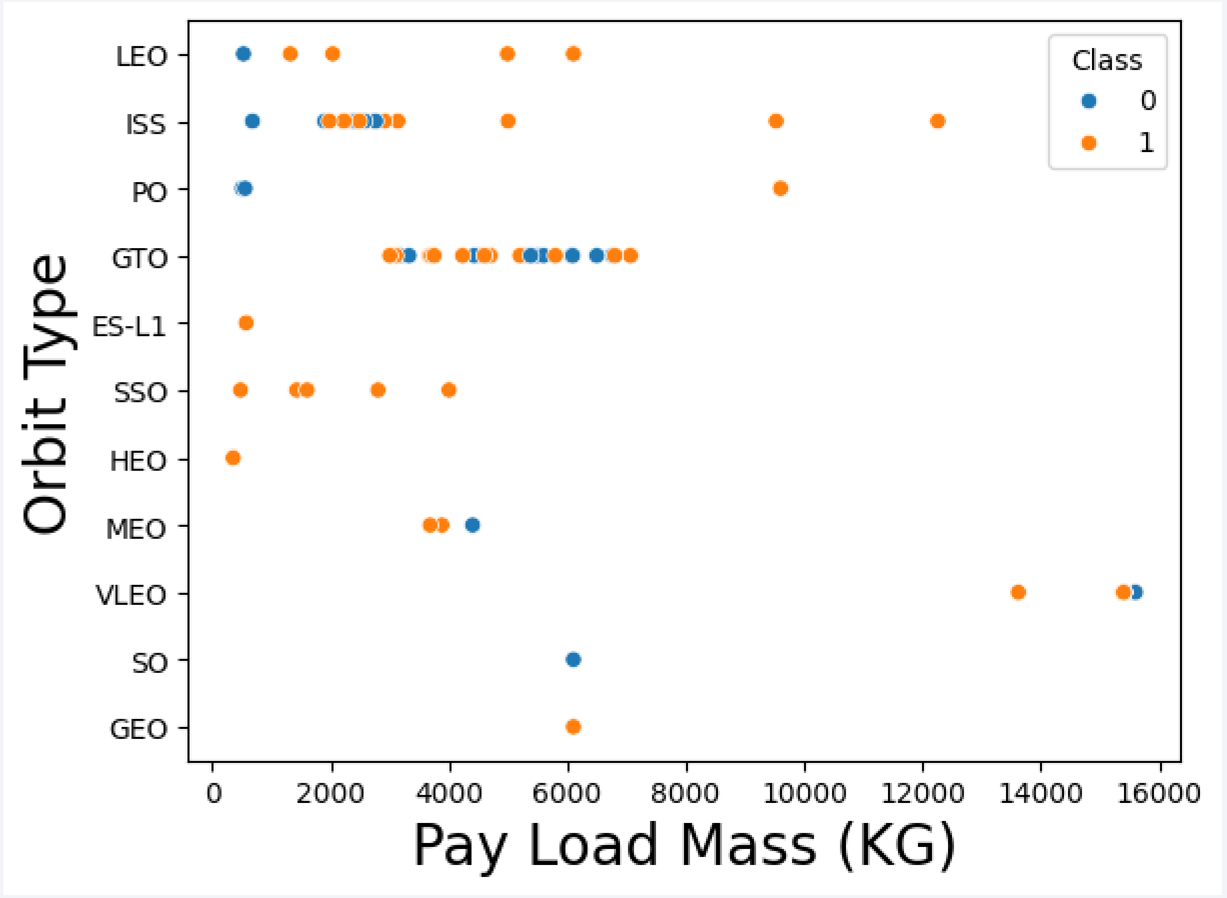
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



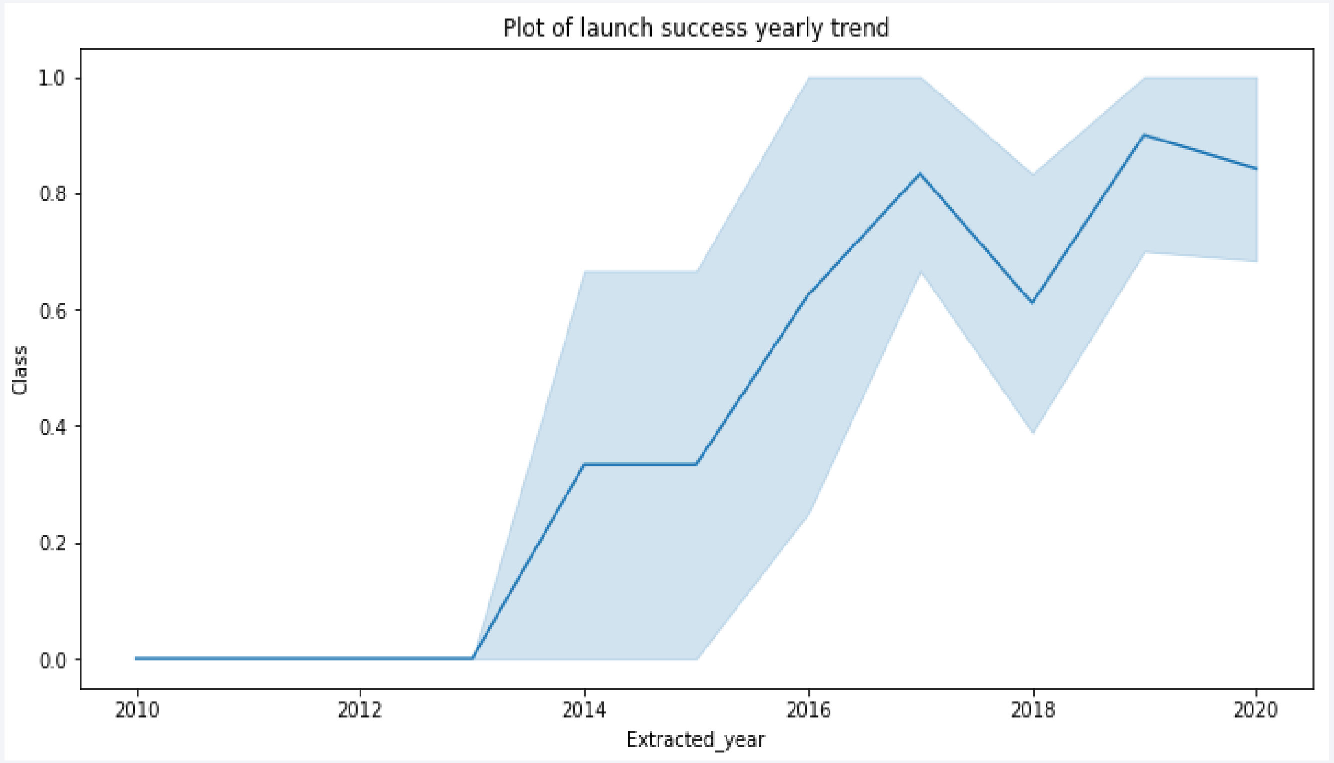
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



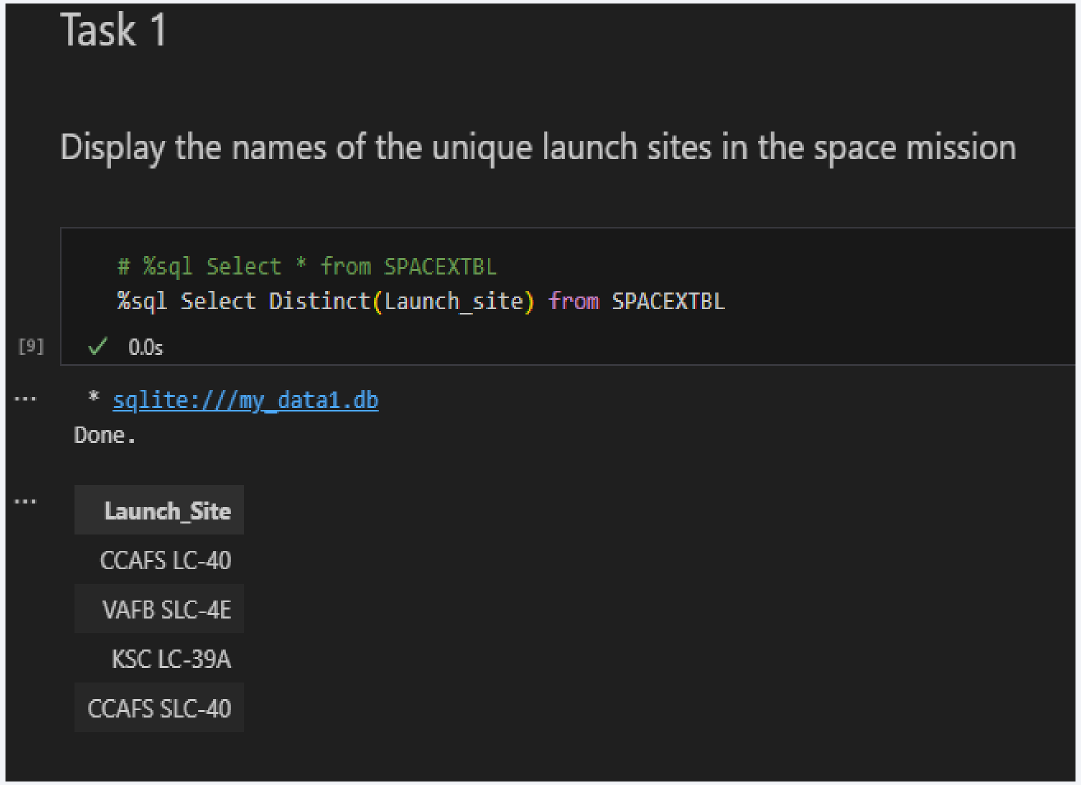
Launch Success Yearly Trend

- From the plot, we can observe 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.
- Query- `%sql Select Distinct(Launch_site) from SPACEXTBL`



Task 1

Display the names of the unique launch sites in the space mission

```
# %sql Select * from SPACEXTBL
%sql Select Distinct(Launch_site) from SPACEXTBL
```

[9] ✓ 0.0s

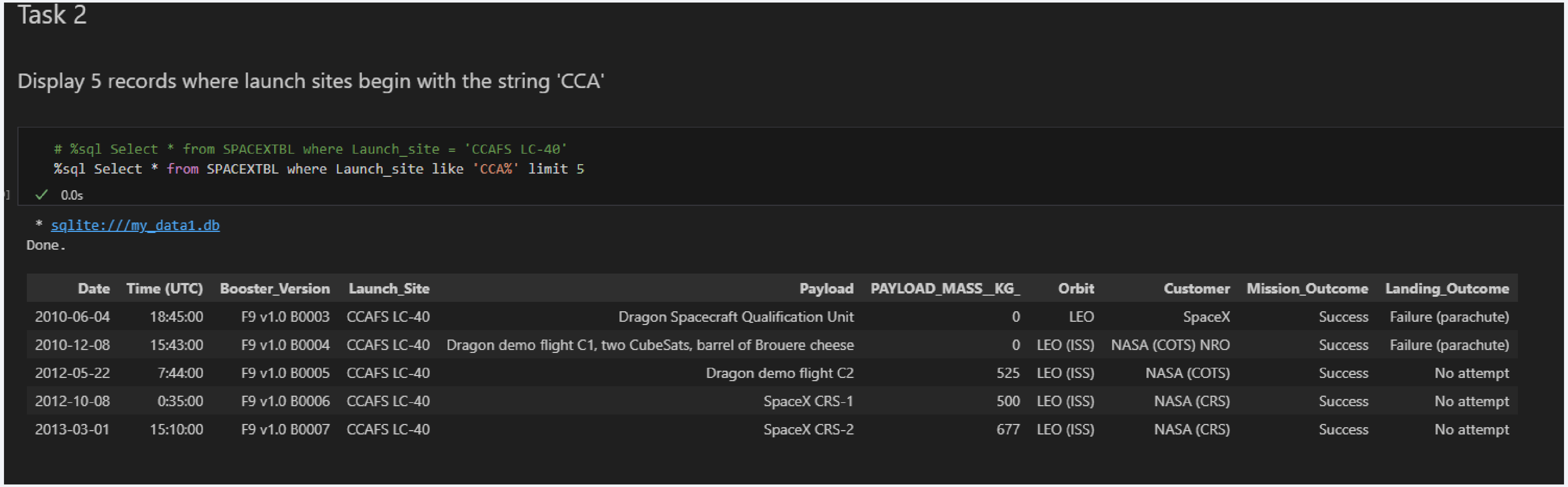
```
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

We used the Like operator with a wildcard for finding the Launch site and limited the result to 5

```
%sql Select * from SPACEXTBL where Launch_site like 'CCA%' limit 5
```



Task 2

Display 5 records where launch sites begin with the string 'CCA'

```
# %sql Select * from SPACEXTBL where Launch_site = 'CCAFS LC-40'
%sql Select * from SPACEXTBL where Launch_site like 'CCA%' limit 5
```

✓ 0.0s

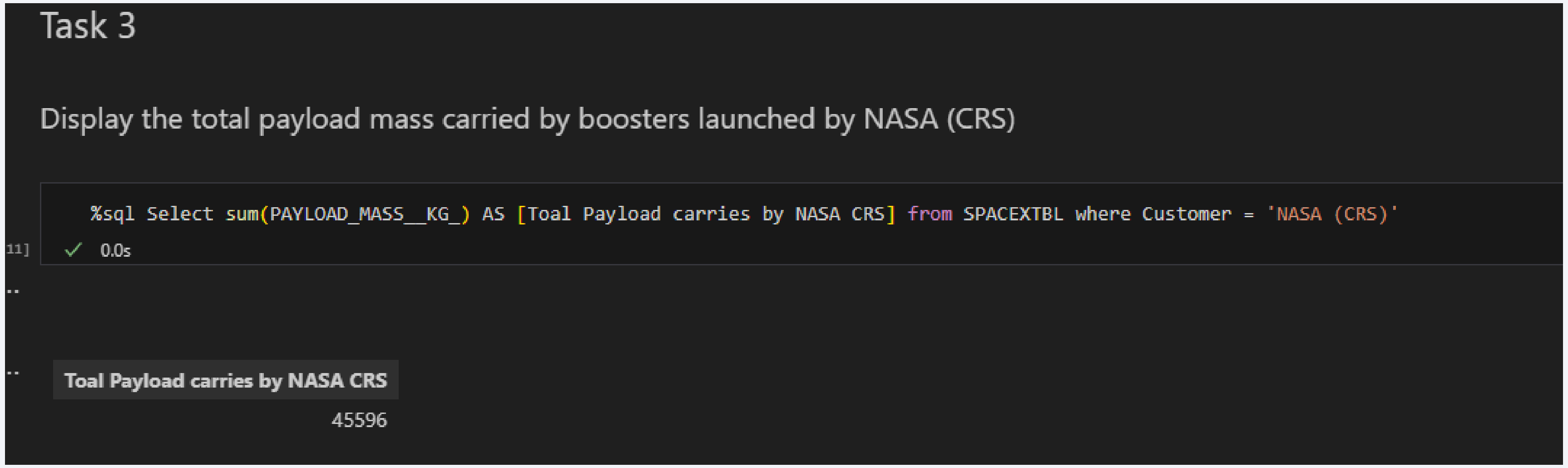
```
 * sqlite:///my_data1.db
Done.
```

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

We used the Sum function on the PAYLOAD_MASS__KG_ COLUMN and where clause for getting result only the ‘NASA (CRS)’ customer

```
%sql Select sum(PAYLOAD_MASS__KG_) AS [Total Payload carries by NASA CRS] from
SPACEXTBL where Customer = 'NASA (CRS)'
```



Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

```
%sql Select sum(PAYLOAD_MASS__KG_) AS [Toal Payload carries by NASA CRS] from SPACEXTBL where Customer = 'NASA (CRS)'
```

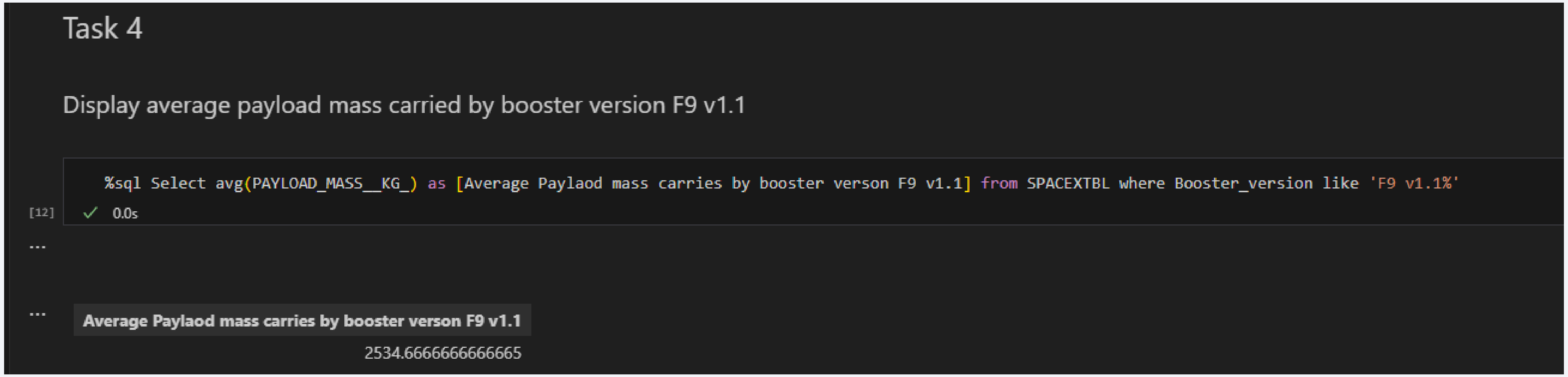
```
✓ 0.0s
```

Toal Payload carries by NASA CRS
45596

Average Payload Mass by F9 v1.1

Using the Average function on the Payload mass column and where clause for booster version column

```
%sql Select avg(PAYLOAD_MASS__KG_) as [Average Paylaod mass carries by booster verson F9 v1.1] from SPACEXTBL where
Booster_version like 'F9 v1.1%'
```



Task 4

Display average payload mass carried by booster version F9 v1.1

```
%sql Select avg(PAYLOAD_MASS__KG_) as [Average Paylaod mass carries by booster verson F9 v1.1] from SPACEXTBL where Booster_version like 'F9 v1.1%'
```

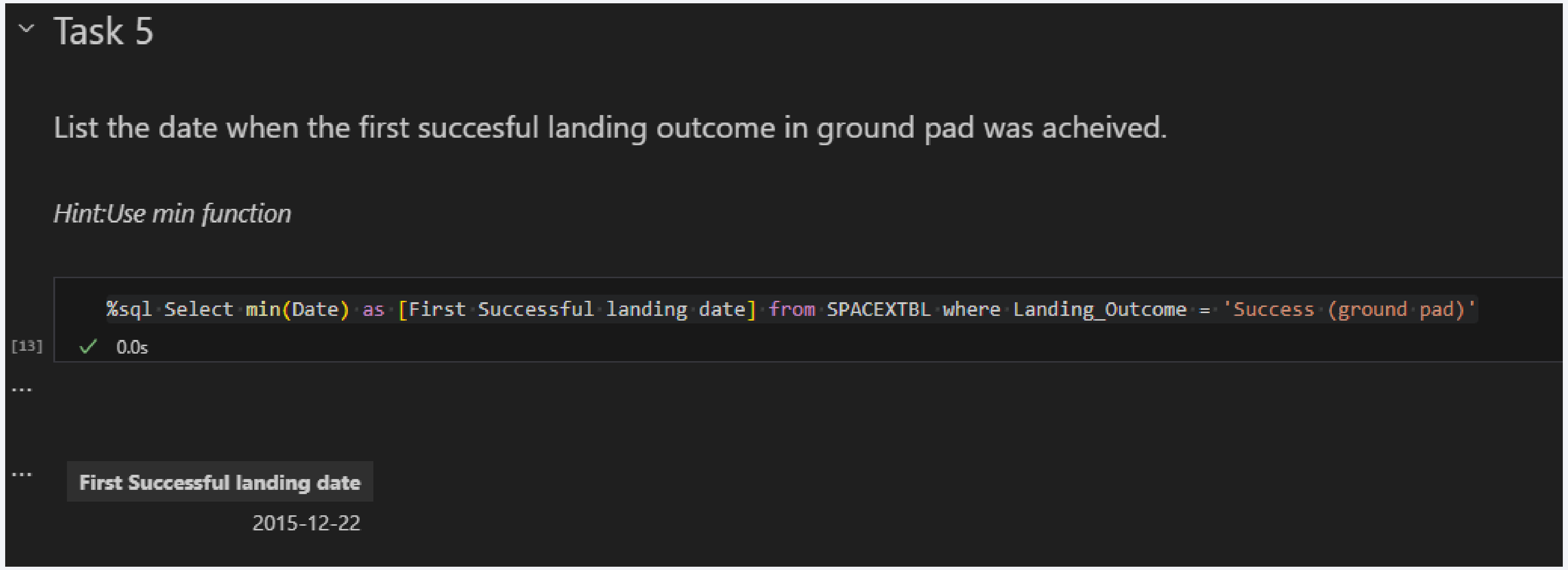
[12] ✓ 0.0s

Average Paylaod mass carries by booster verson F9 v1.1
2534.6666666666665

First Successful Ground Landing Date

Used the Minimum function and Date column and where clause for Landing Outcome

```
%sql Select min(Date) as [First Successful landing date] from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
```



Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

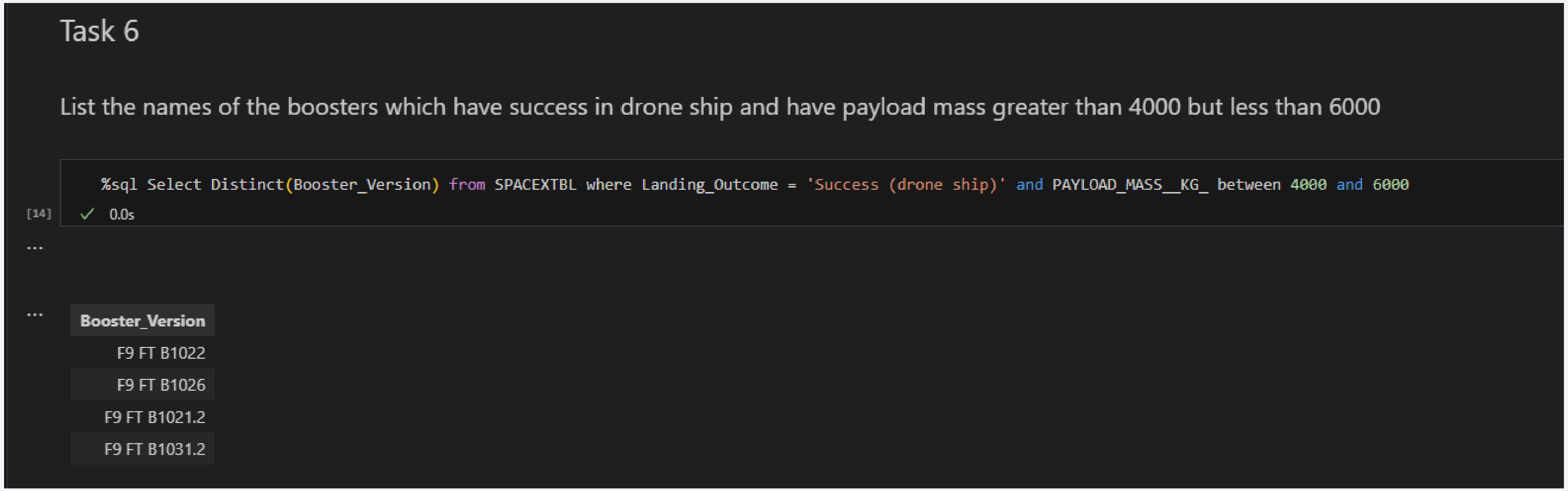
```
%sql Select min(Date) as [First Successful landing date] from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
[13] ✓ 0.0s
```

First Successful landing date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

We used the Distinct function for booster version and 2 where clauses, one for Landing Outcome and second for payload mass using BETWEEN and AND operators

```
%sql Select Distinct(Booster_Version) from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000
```



Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql Select Distinct(Booster_Version) from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000
```

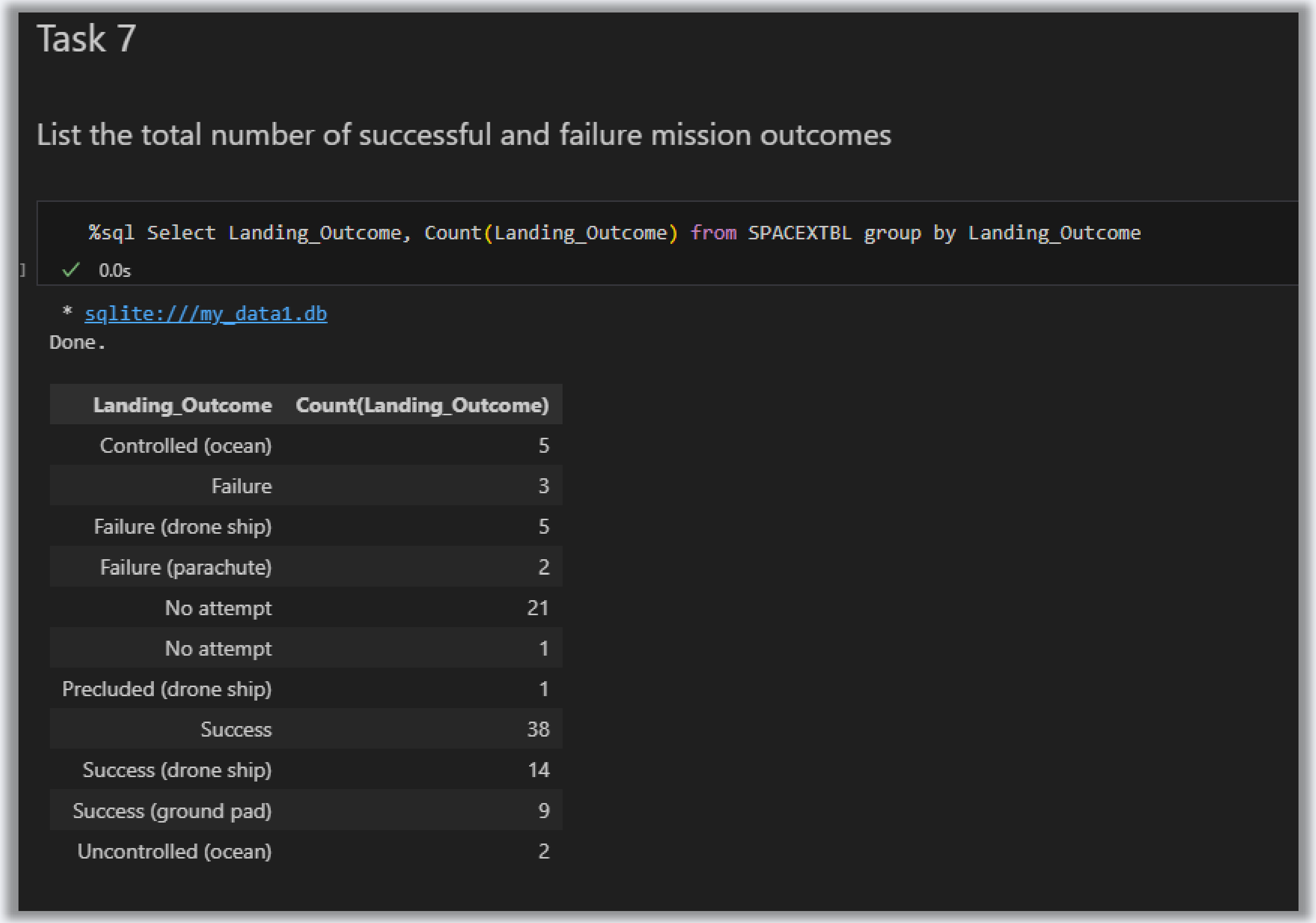
[14] ✓ 0.0s

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

Total Number of Successful and Failure Mission Outcomes

We used the count function on the landing outcome and then grouped the data set based on the Landing outcomes

```
%sql Select Landing_Outcome,
Count(Landing_Outcome) from SPACEXTBL
group by Landing_Outcome
```



Task 7

List the total number of successful and failure mission outcomes

```
%sql Select Landing_Outcome, Count(Landing_Outcome) from SPACEXTBL group by Landing_Outcome
✓ 0.0s
```

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

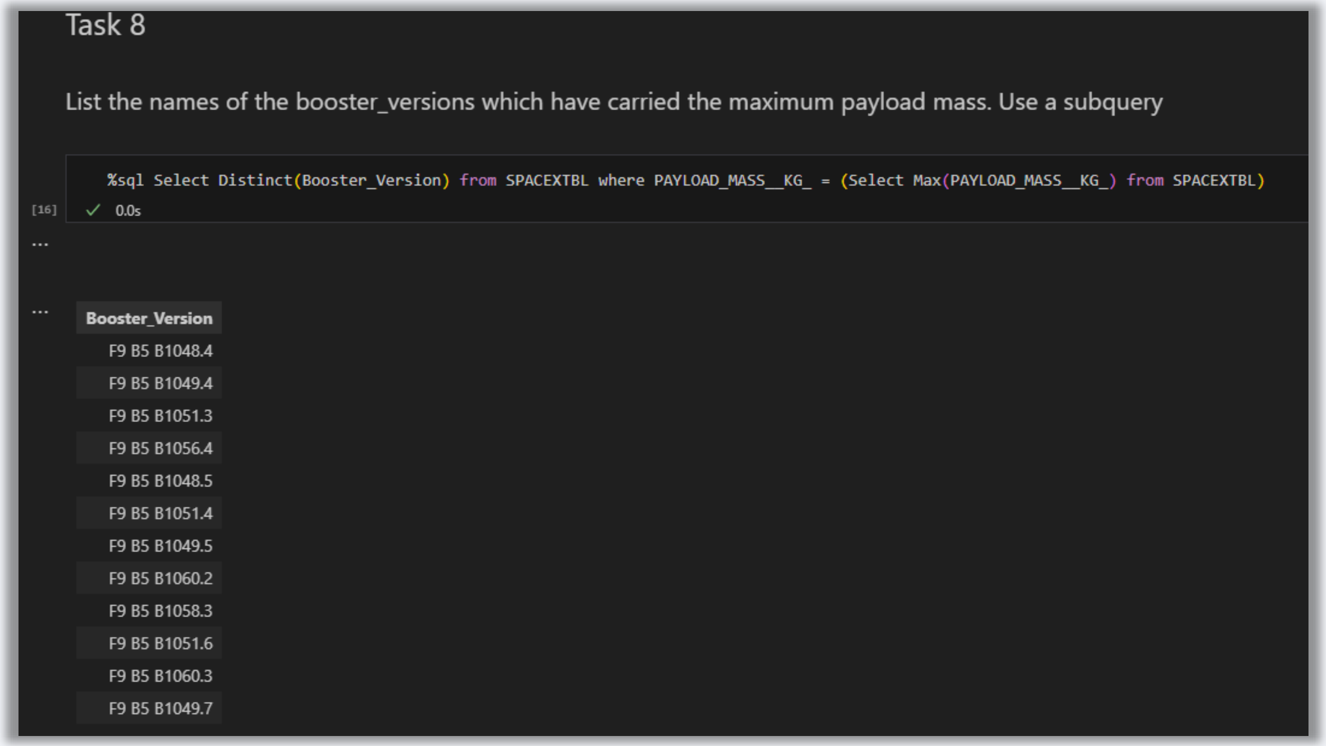
Landing_Outcome	Count(Landing_Outcome)
Controlled (ocean)	5
Failure	3
Failure (drone ship)	5
Failure (parachute)	2
No attempt	21
No attempt	1
Precluded (drone ship)	1
Success	38
Success (drone ship)	14
Success (ground pad)	9
Uncontrolled (ocean)	2

Boosters Carried Maximum Payload

We used a subquery for this problem set.

The sub query first extracts the maximum payload mass for the dataset. The result from this query is used to fetch all the booster versions which have the payload mass on the max payload.

```
%sql Select Distinct(Booster_Version)
from SPACEXTBL where PAYLOAD_MASS__KG_
= (Select Max(PAYLOAD_MASS__KG_) from
SPACEXTBL)
```



Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

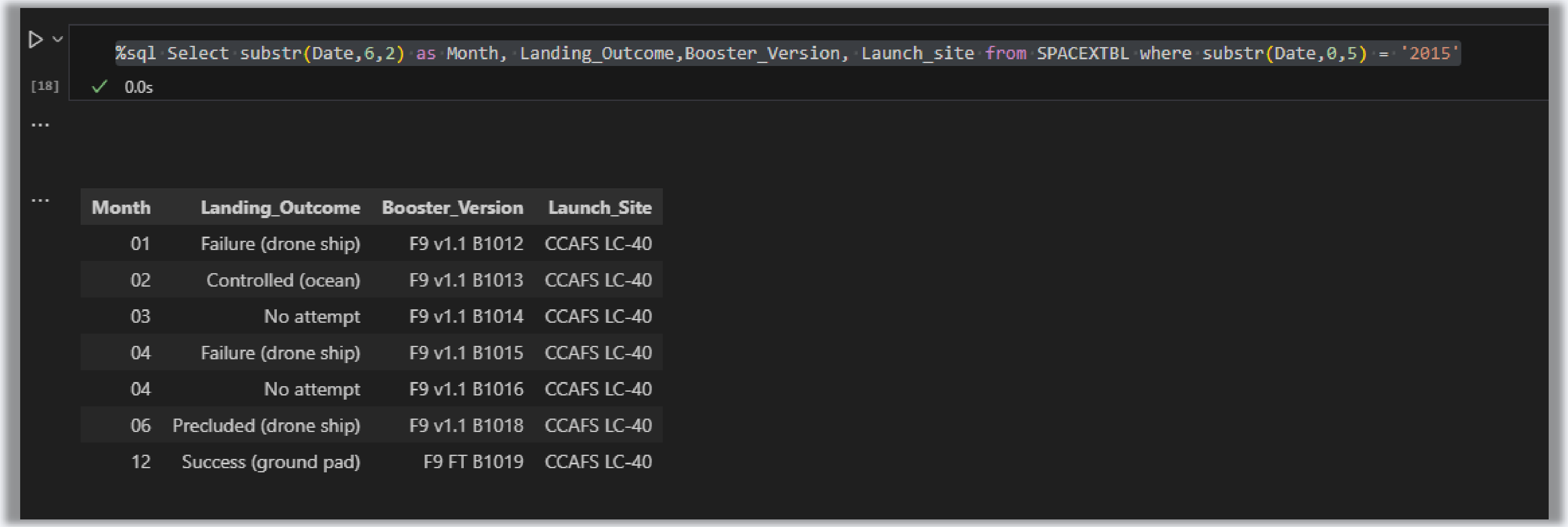
```
%sql Select Distinct(Booster_Version) from SPACEXTBL where PAYLOAD_MASS__KG_ = (Select Max(PAYLOAD_MASS__KG_) from SPACEXTBL)
[16] ✓ 0.0s
```

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

For finding the answer for this problem set, we used the substring method on the Date column.

```
%sql Select substr(Date,6,2) as Month, Landing_Outcome,Booster_Version, Launch_site from SPACEXTBL where substr(Date,0,5)
= '2015'
```

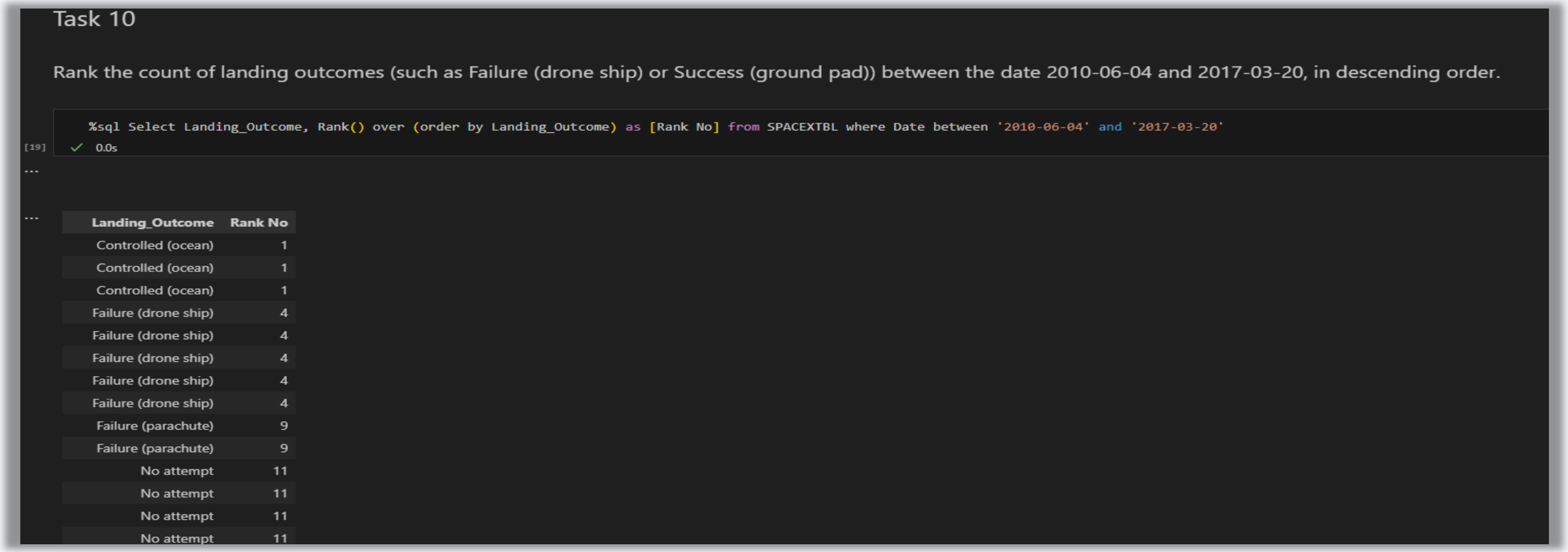


```
%sql Select substr(Date,6,2) as Month, Landing_Outcome,Booster_Version, Launch_site from SPACEXTBL where substr(Date,0,5) = '2015'
[18] ✓ 0.0s
```

Month	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
02	Controlled (ocean)	F9 v1.1 B1013	CCAFS LC-40
03	No attempt	F9 v1.1 B1014	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40
04	No attempt	F9 v1.1 B1016	CCAFS LC-40
06	Precluded (drone ship)	F9 v1.1 B1018	CCAFS LC-40
12	Success (ground pad)	F9 FT B1019	CCAFS LC-40

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

```
%sql Select Landing_Outcome, Rank() over (order by Landing_Outcome) as [Rank No] from SPACEXTBL where Date between
'2010-06-04' and '2017-03-20'
```



Task 10

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

```
%sql Select Landing_Outcome, Rank() over (order by Landing_Outcome) as [Rank No] from SPACEXTBL where Date between '2010-06-04' and '2017-03-20'
```

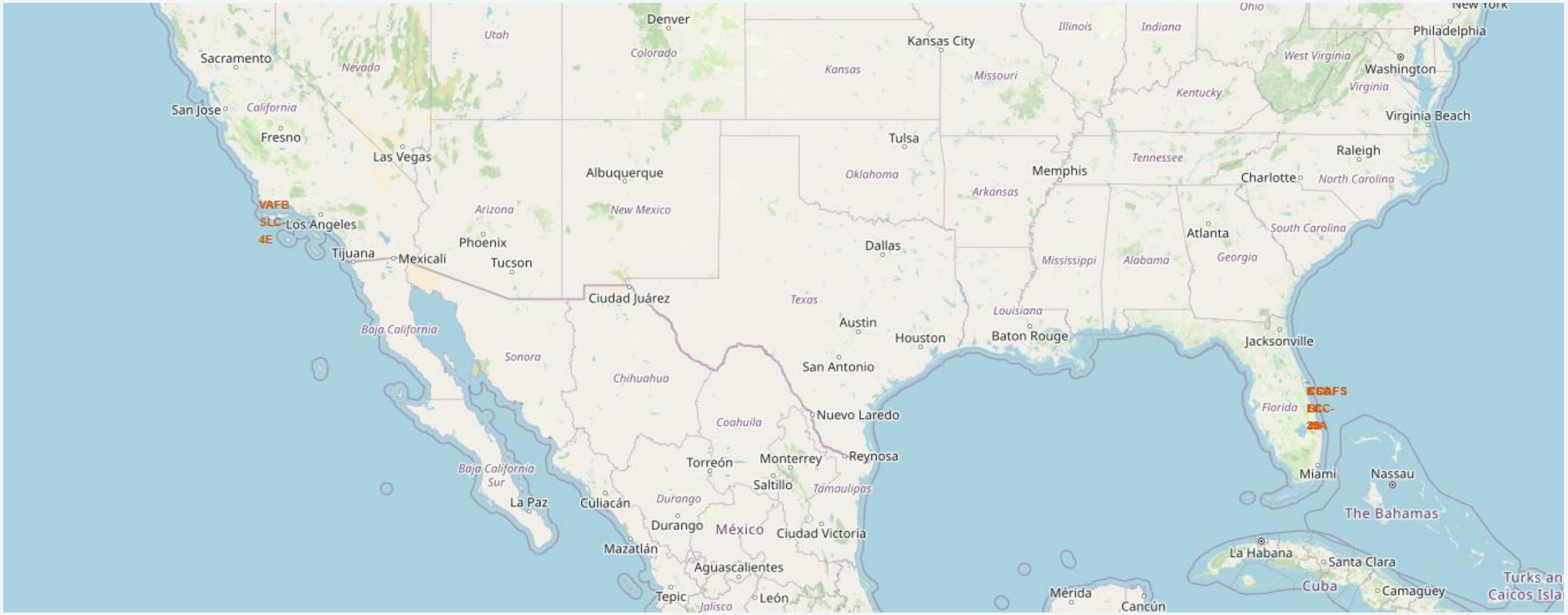
[19] ✓ 0.0s

Landing_Outcome	Rank No
Controlled (ocean)	1
Controlled (ocean)	1
Controlled (ocean)	1
Failure (drone ship)	4
Failure (drone ship)	4
Failure (drone ship)	4
Failure (drone ship)	4
Failure (drone ship)	4
Failure (parachute)	9
Failure (parachute)	9
No attempt	11
No attempt	11
No attempt	11
No attempt	11

Section 3

Launch Sites Proximities Analysis

All launch sites global map markers



The SpaceX launch sites are located on the coasts of California and Florida in United States of America

Markers showing launch sites with color labels

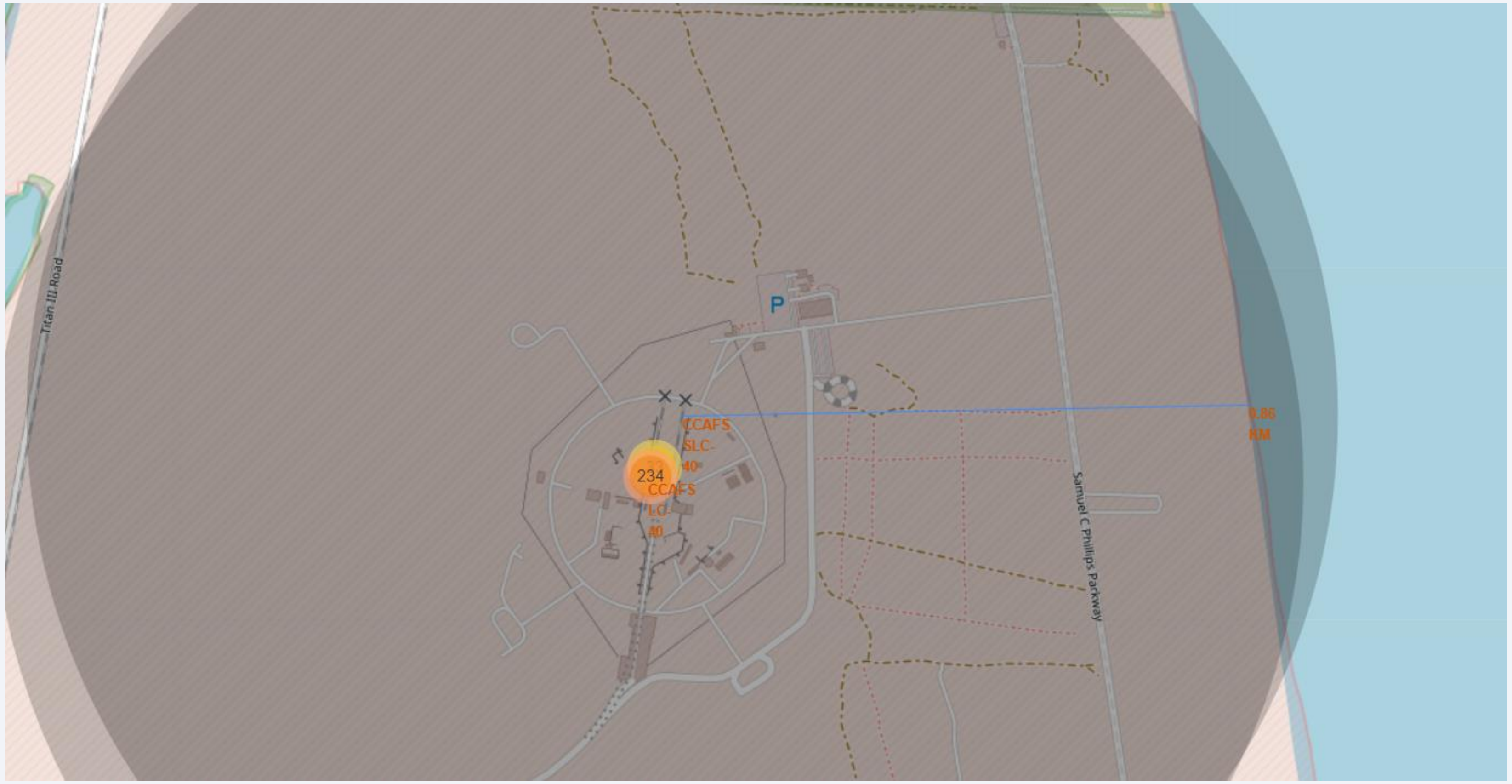
Florida Launch sites and its proximities

7
26
Centaur Road
Samuel C Phillips Parkway

Dix Road
Coast Road
Surf Road
Kelp Road
Agena Way
Space Launch Complex 4
10
Surf Road
Spring Canyon Road

California Launch sites and its proximities

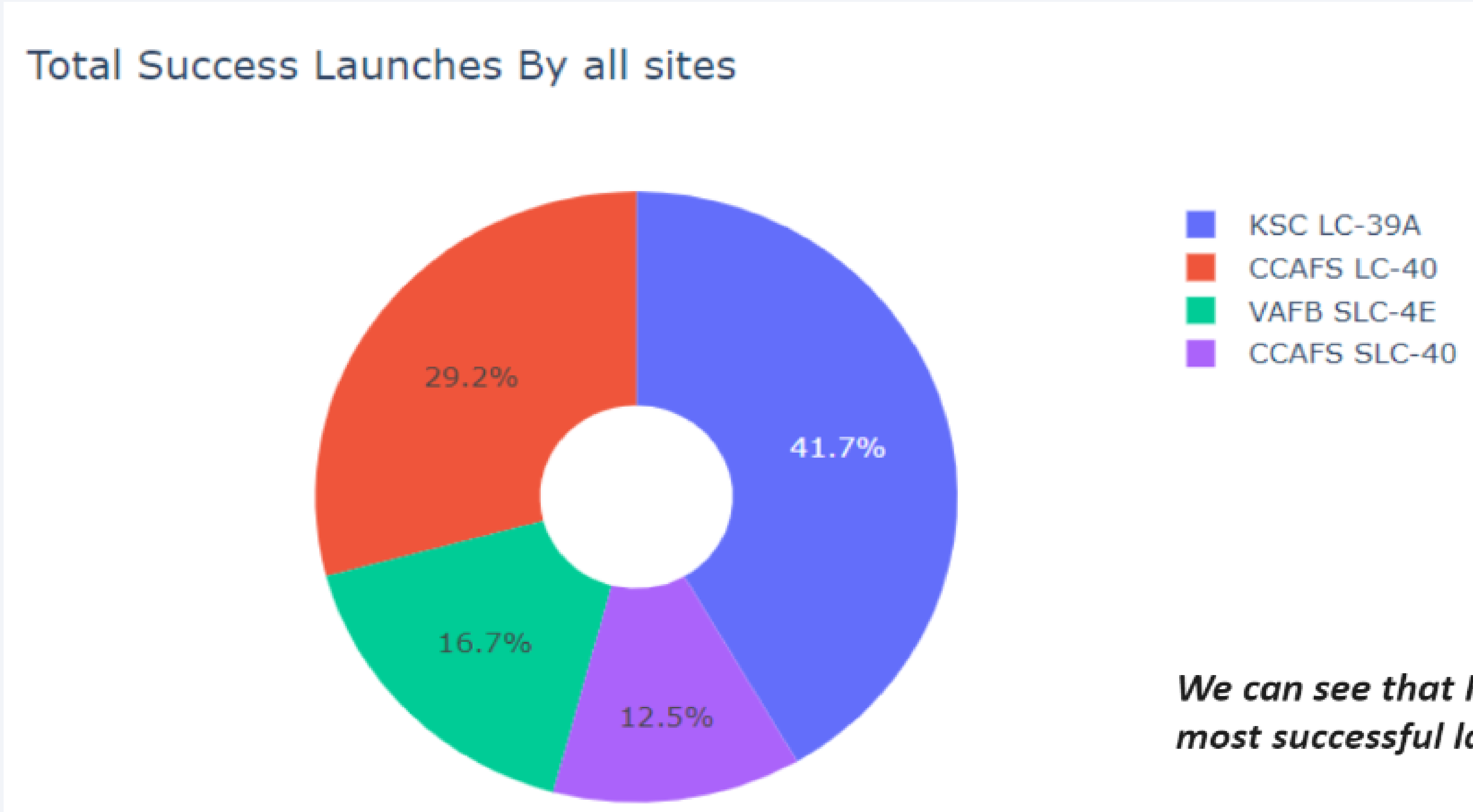
Launch Site distance to landmarks



Section 4

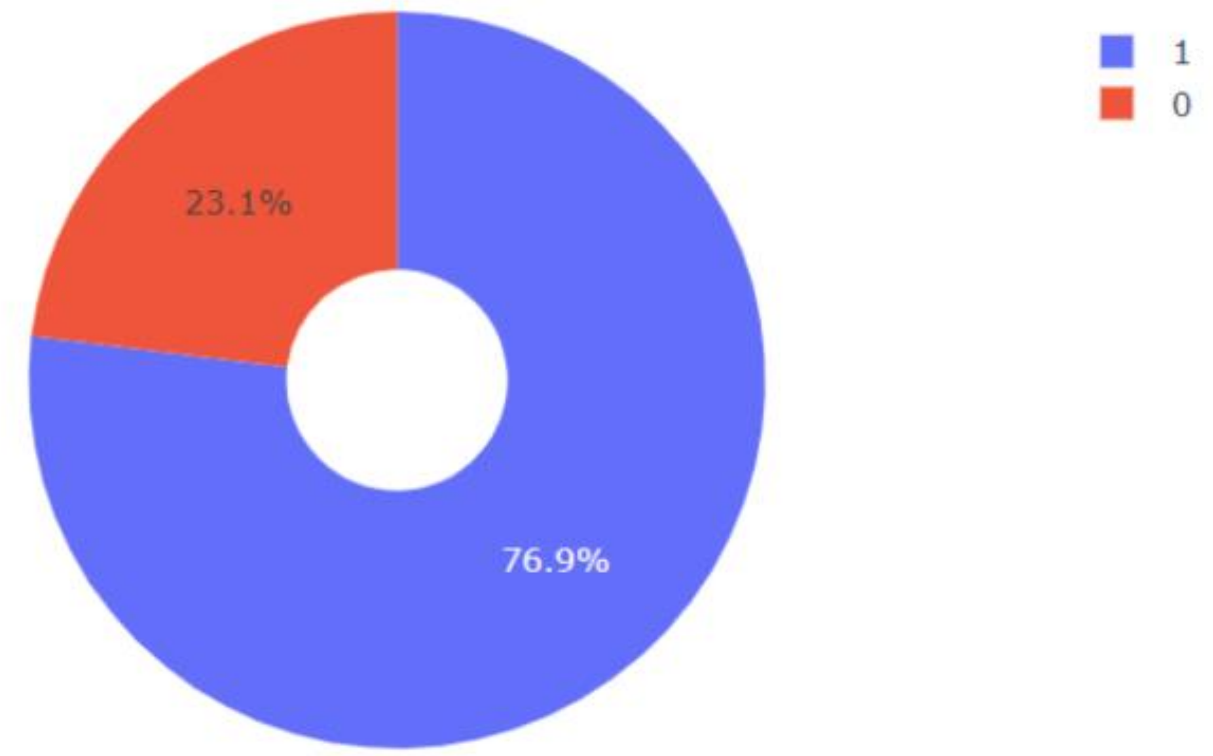
Build a Dashboard with Plotly Dash

Pie Chat showing success percentage of each launch site



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

Launch Site with Highest Success Ratio

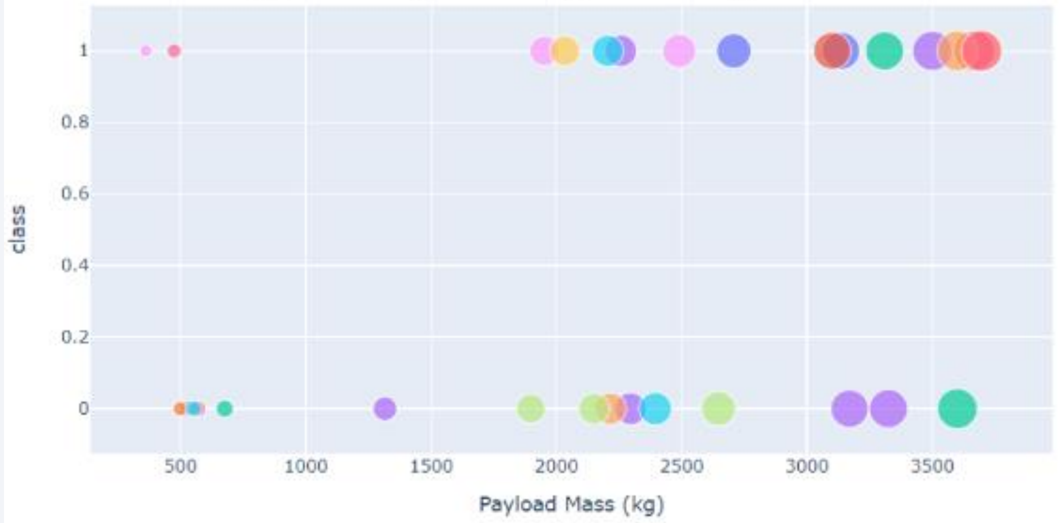


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

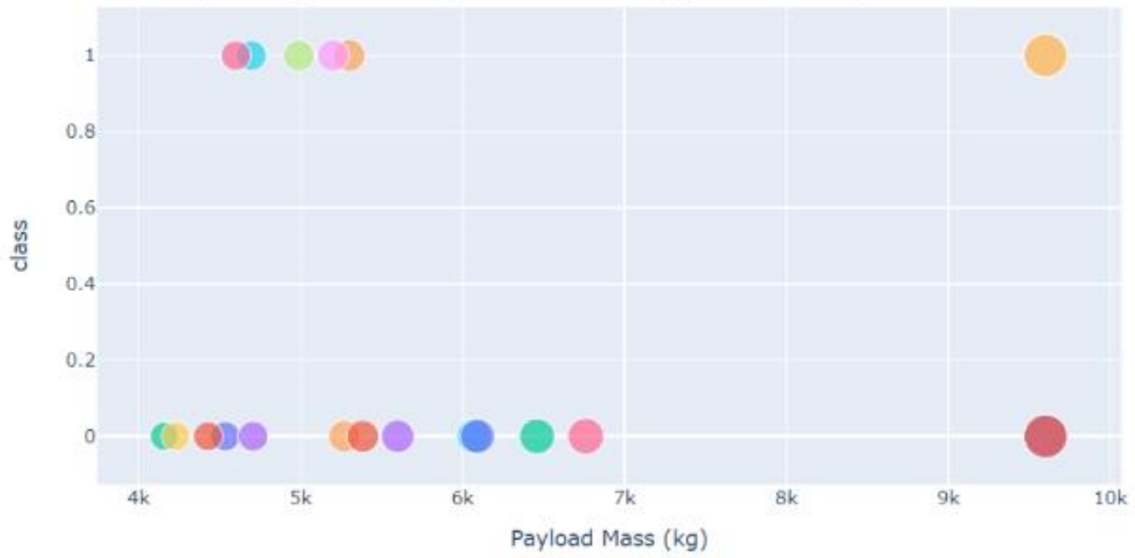
Scatter Plot of Payload vs Launch Outcome for all sites

With different payload selected in the range slider

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg



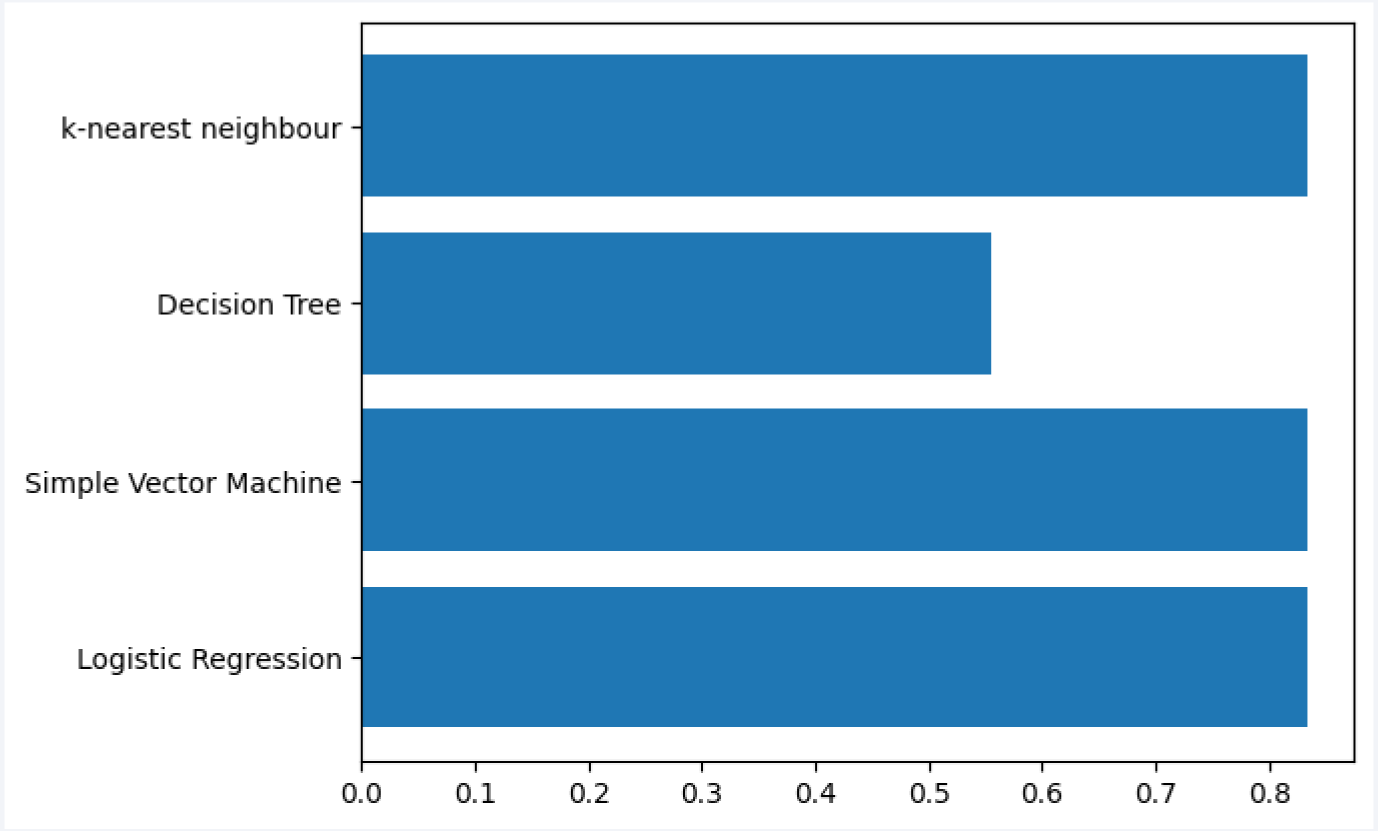
We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 5

Predictive Analysis (Classification)

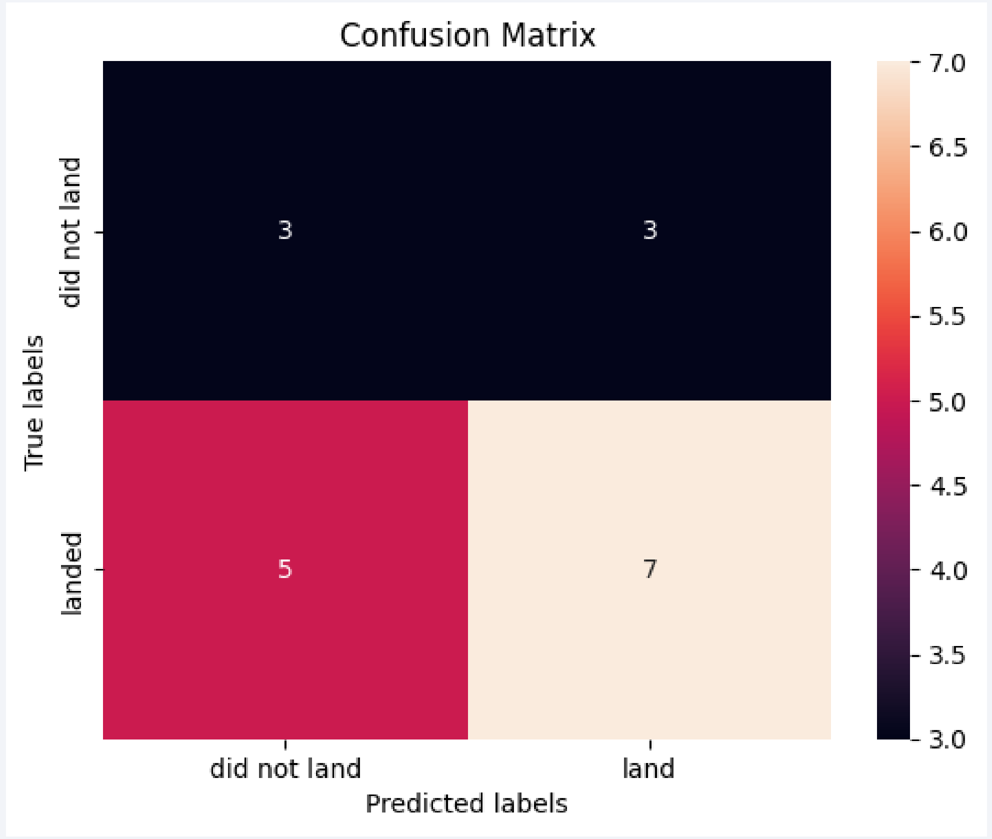
Classification Accuracy

The decision Tree classifier is the model with lowest Root Mean Square Error and highest accuracy.



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

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Appendix

Happy to share the link for my GitHub repository containing almost all Labs and notebooks that were created during the practice and assignments during the IBM Data Science Course.

GitHub_Ahad_Khan

Thank you!