IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Rameson Seraphin
14 October 2024

Outline

Executive Summary

- Summary of methodologies

  - SpaceX data Collection using SpaceX API

  - SpaceX Data Collection using Web Scraping

  - SpaceX Data Wrangling

  - SpaceX Exploratory Data Analysis using SQL

  - Data EDA With Visualization With Pandas and Matplotlib

  - SpaceX Launch Sites Analysis with Folium-Interactive Visual Analytics and Plotly Dash

  - SpaceX Machine Learning Landing Prediction

- Summary of all results

  - EDA results

  - Interactive Visual Analytics and Dashboards

Executive Summary Cont….

- Predictive analysis

Introduction

- Project background and context

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings 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.

- Problems you want to find answers

In this capstone, we will predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

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

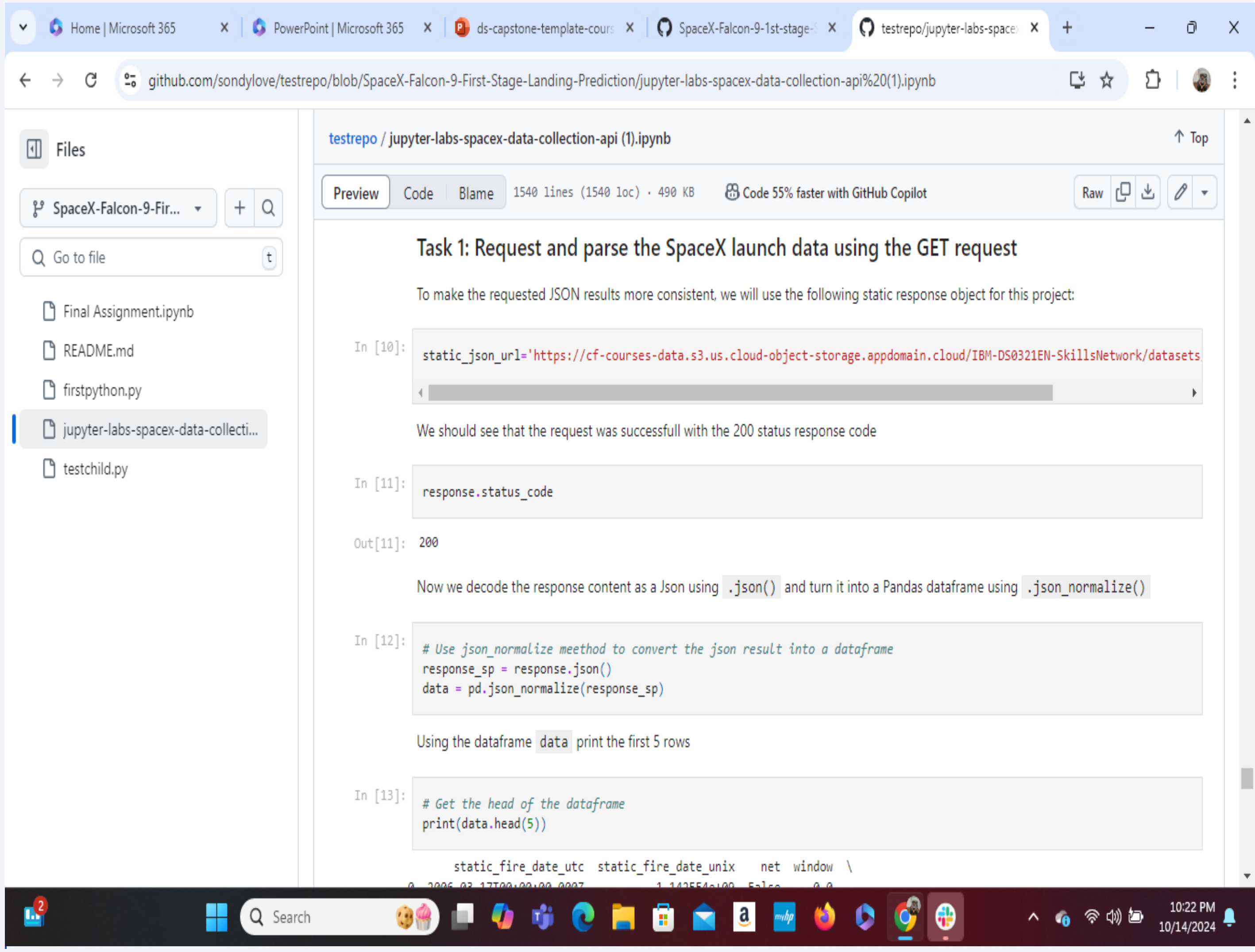
- Describe how SpaceX Falcon 9 data sets were collected.
    - Data was first collected using SpaceX API by making a get request to the SpaceX API. This was done by first defining a series of helper functions that would help in the use of the API to extract information using identification numbers in the launch data from the SpaceX API url.
    - However to make the requested JSON results more consistent, the SpaceX launch data was requested and parsed using the GET request and then decoded the response content as a JSON result which was then converted into a Pandas data frame.
    - Also performed web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled "List of Falcon 9 and Falcon Heavy Launches" of the launch records are stored in a HTML. Using BeautifulSoup and request Libraries, we extract the Falcon 9 launch HTML table records from the Wikipedia page, Parsed the table and converted it into a Pandas data frame.

Data Collection – SpaceX API

- The Data was collected using SpaceX API(Restful API) by making a get request to the SpaceX API then requested and parsed the SpaceX launch data using the GET request and decoded the response content as a JSON file which was then converted into a Pandas data frame.

- Here is the GitHub link:

https://github.com/sondylove/testrepo/blob/SpaceX-Falcon-9-First-Stage-Landing-Prediction/jupyter-labs-spacex-data-collection-api%20(1).ipynb


github.com/sondylove/testrepo/blob/SpaceX-Falcon-9-First-Stage-Landing-Prediction/jupyter-labs-spacex-data-collection-api%20(1).ipynb
Files
Final Assignment.ipynb
README.md
firstpython.py
jupyter-labs-spacex-data-collecti...
testchild.py
testrepo / jupyter-labs-spacex-data-collection-api (1).ipynb
Preview Code Blame 1540 lines (1540 loc) · 490 KB

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:

```
In [10]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets
```

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

```
In [11]: response.status_code

Out[11]: 200
```

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

```
In [12]: # Use json_normalize meethod to convert the json result into a dataframe
         response_sp = response.json()
         data = pd.json_normalize(response_sp)
```

Using the dataframe `data` print the first 5 rows

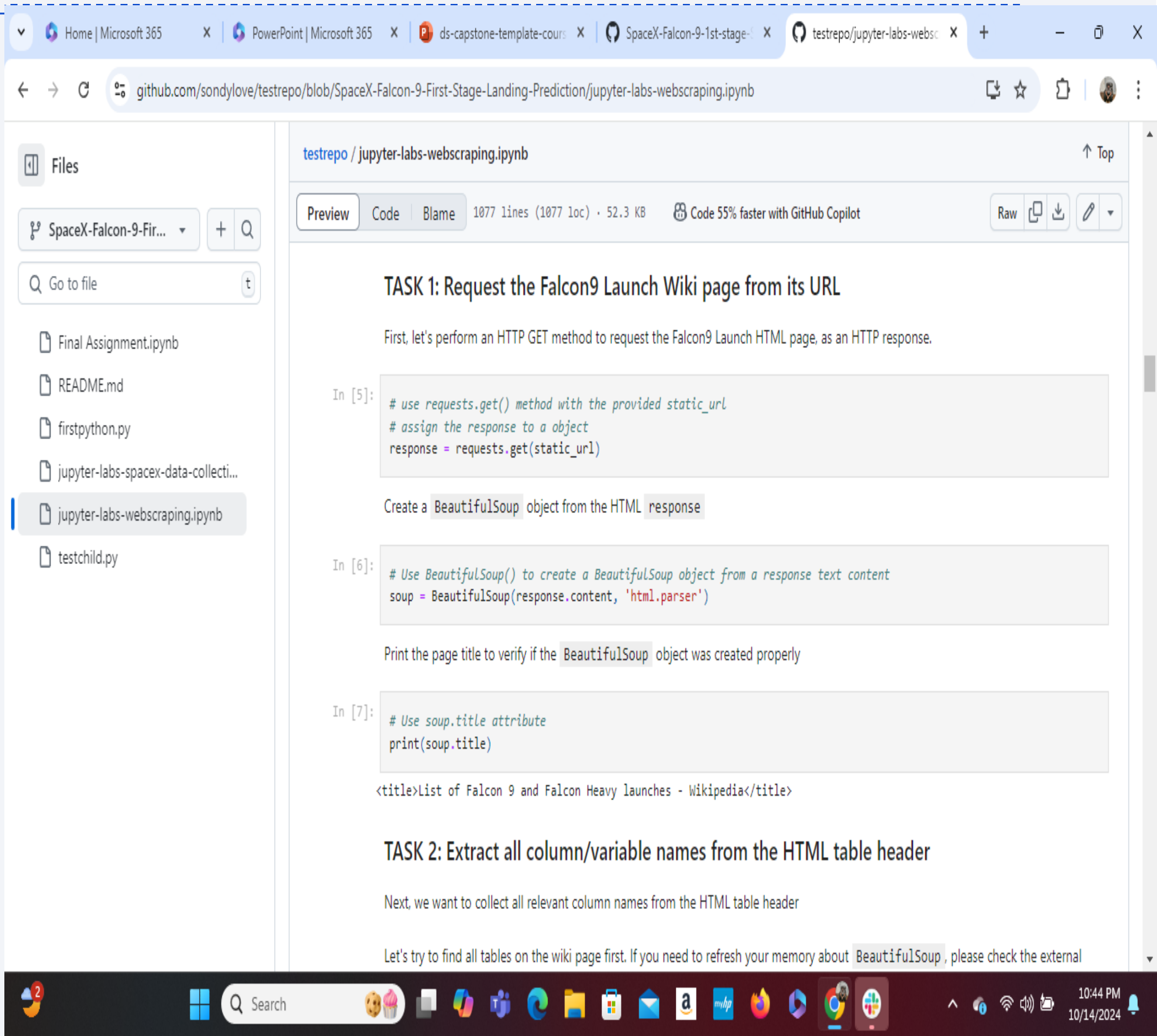
```
In [13]: # Get the head of the dataframe
         print(data.head(5))

      static_fire_date_utc  static_fire_date_unix    net  window  \
```

Data Collection - Scraping

- Web scraping was performed to collect Falcon 9 historical launch records from a Wikipedia using BeautifulSoup and request, to extract the Falcon 9 launch records from HTML table of the Wikipedia page, then creating a data frame by parsing the launch HTML.

- Here is the GitHub URL

- https://github.com/sondylove/testrepo/blob/SpaceX-Falcon-9-First-Stage-Landing-Prediction/jupyter-labs-webscraping.ipynb


Home | Microsoft 365
PowerPoint | Microsoft 365
ds-capstone-template-cours
SpaceX-Falcon-9-1st-stage-
testrepo/jupyter-labs-websc
github.com/sondylove/testrepo/blob/SpaceX-Falcon-9-First-Stage-Landing-Prediction/jupyter-labs-webscraping.ipynb
Files
SpaceX-Falcon-9-Fir...
Go to file
Final Assignment.ipynb
README.md
firstpython.py
jupyter-labs-spacex-data-collecti...
jupyter-labs-webscraping.ipynb
testchild.py
testrepo / jupyter-labs-webscraping.ipynb
Top
Preview
Code
Blame
1077 lines (1077 loc) · 52.3 KB
Code 55% faster with GitHub Copilot
Raw
Search
10:44 PM
10/14/2024

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

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

```
In [6]: # 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 [7]: # Use soup.title attribute
        print(soup.title)

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

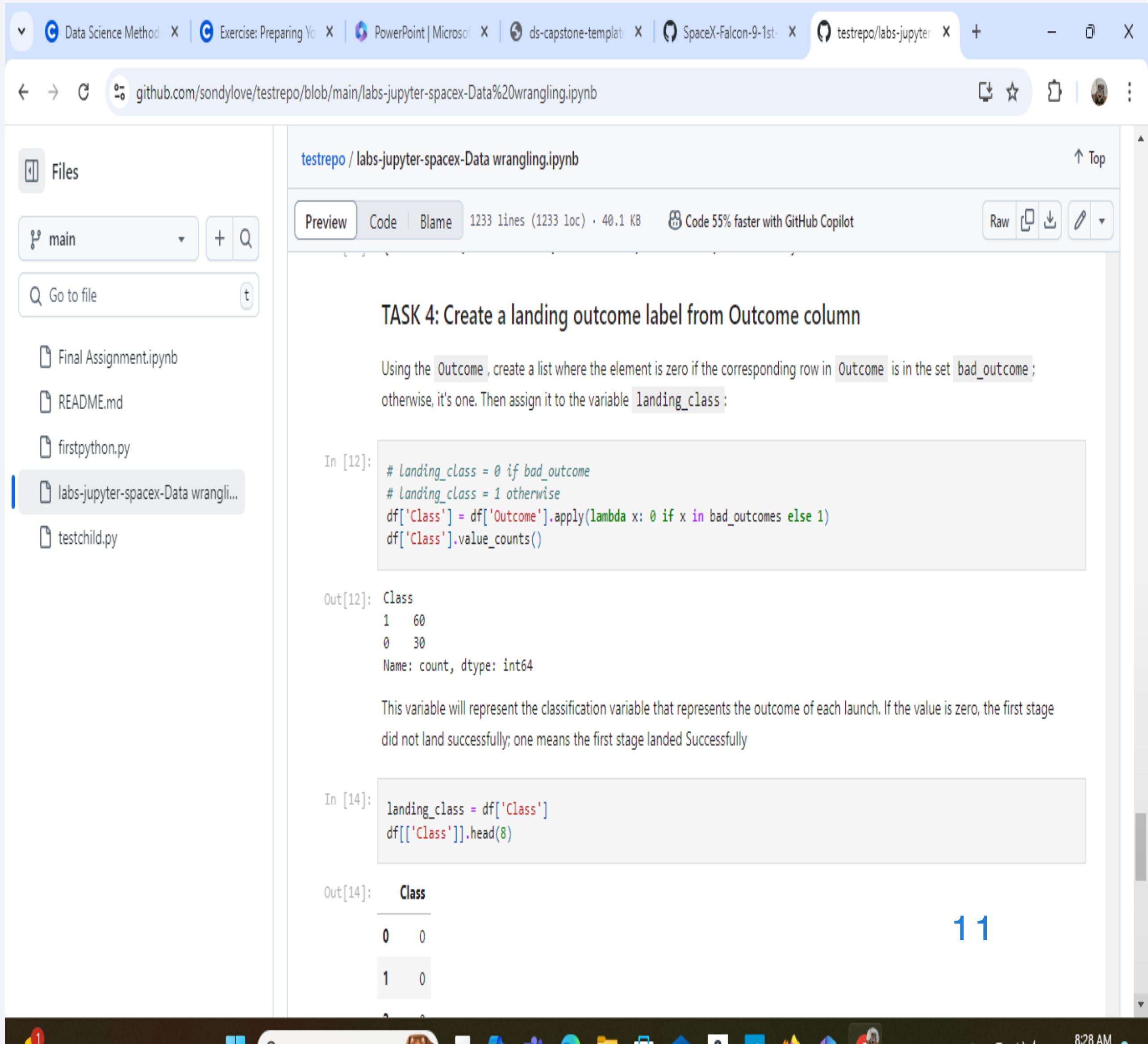
TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external

Data Wrangling

- After Obtaining and creating a Pandas DF from the collected data, data was filtered using the "BoosterVersion" column to only keep the Falcon 9 launches, then we dealt with the missing values in the "LendingPad" and "PayLoadMass" columns. For the PayLoadMass we replace the missing values with the mean of the columns.
- Also performed some Exploratory Data Analysis(EDA) to find some patterns in the and determine what would be the label for training supervised models.
- The following is the URL of the completed data
- https://github.com/sondylove/testrepo/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb


Data Science Method
Exercise: Preparing Y
PowerPoint | Microso
ds-capstone-templat
SpaceX-Falcon-9-1st
testrepo/labs-jupyter
github.com/sondylove/testrepo/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb
Files
main
Go to file
Final Assignment.ipynb
README.md
firstpython.py
labs-jupyter-spacex-Data wrangli...
testchild.py
testrepo / labs-jupyter-spacex-Data wrangling.ipynb
Top
Preview
Code
Blame
1233 lines (1233 loc) · 40.1 KB
Code 55% faster with GitHub Copilot
Raw

TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

```
In [12]:
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
df['Class'] = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1)
df['Class'].value_counts()
```

```
Out[12]:
Class
1    60
0    30
Name: count, dtype: int64
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

```
In [14]:
landing_class = df['Class']
df[['Class']].head(8)
```

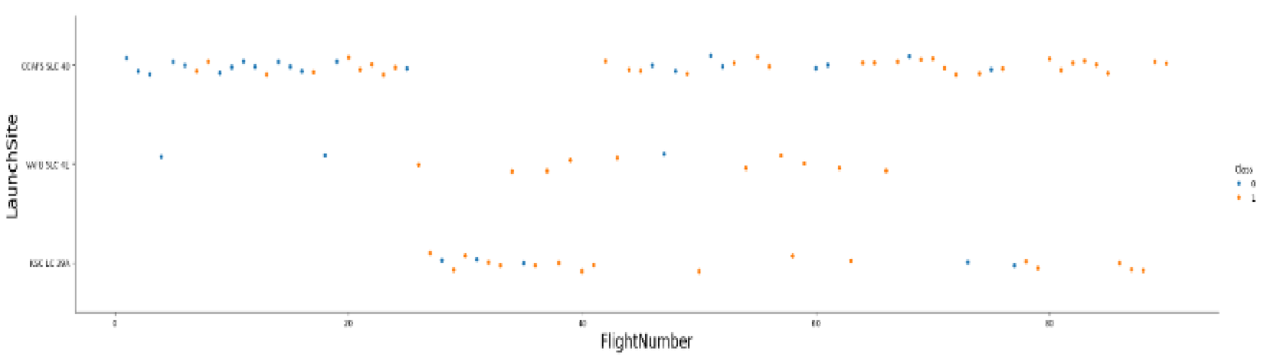
```
Out[14]:
   Class
0  0
1  0
```

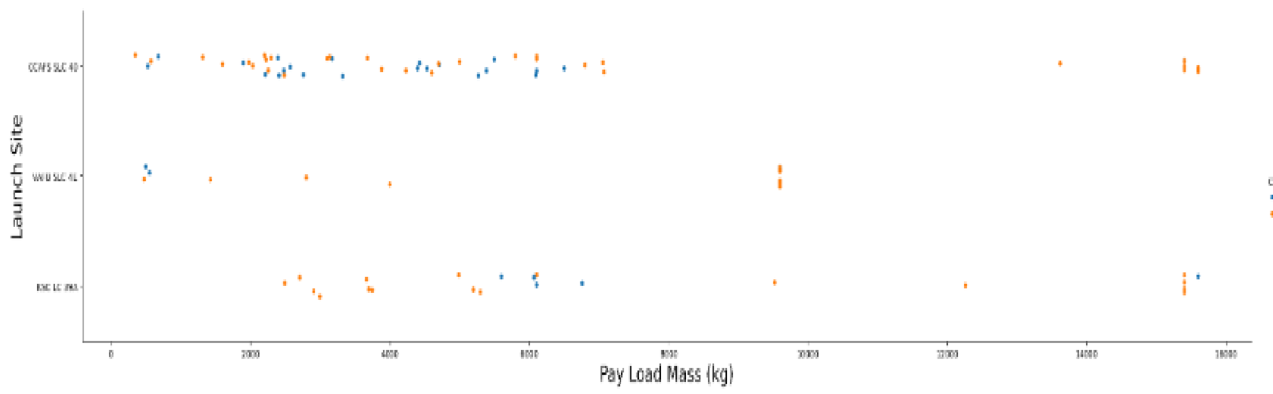
EDA With Data Visualization

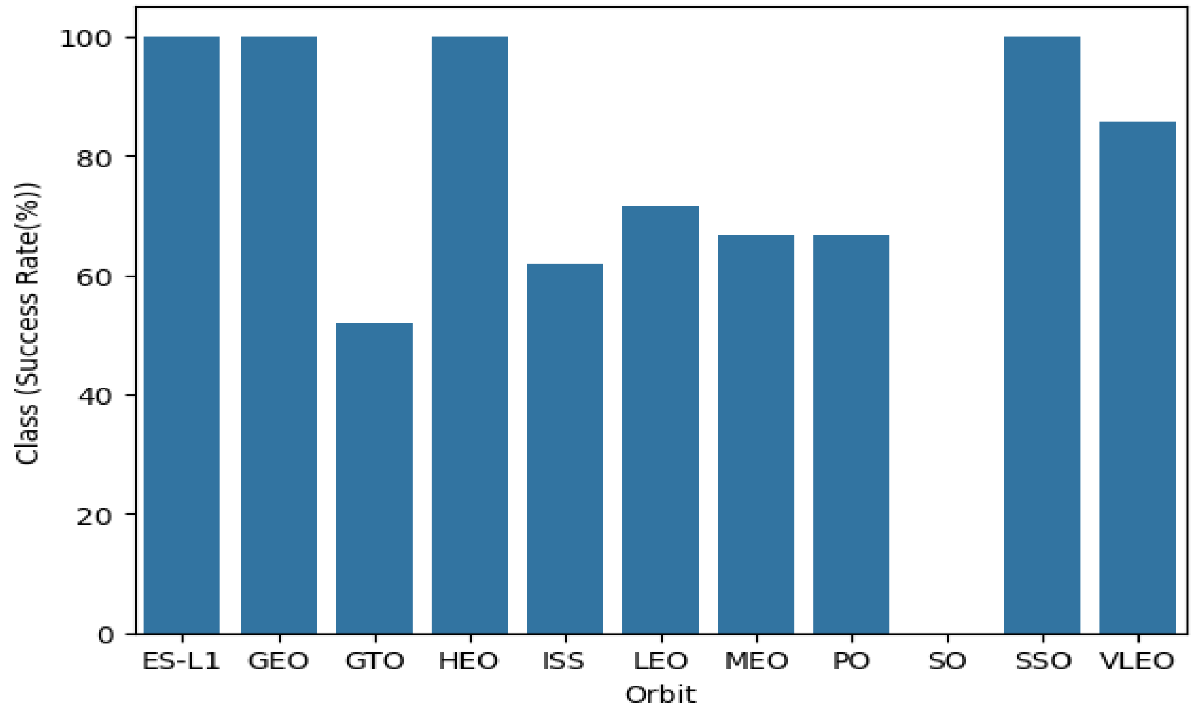
- Performed Data Analysis and Feature Engineering using Pandas and Matplotlib:
  - Exploratory Data Analysis
  - Preparing Data Feature Engineering
- Use Scatter Plots to Visualize the relationship between Flight Number and Launch Site, Payload and Launch Site, Flight Number and Orbit Type, Payload and Orbit Type.
- Use Bar chart to visualize the relationship between success rate and each Orbit Type.
- Line plot to visualize the launch success yearly trend.
- The following is the GitHub URL:

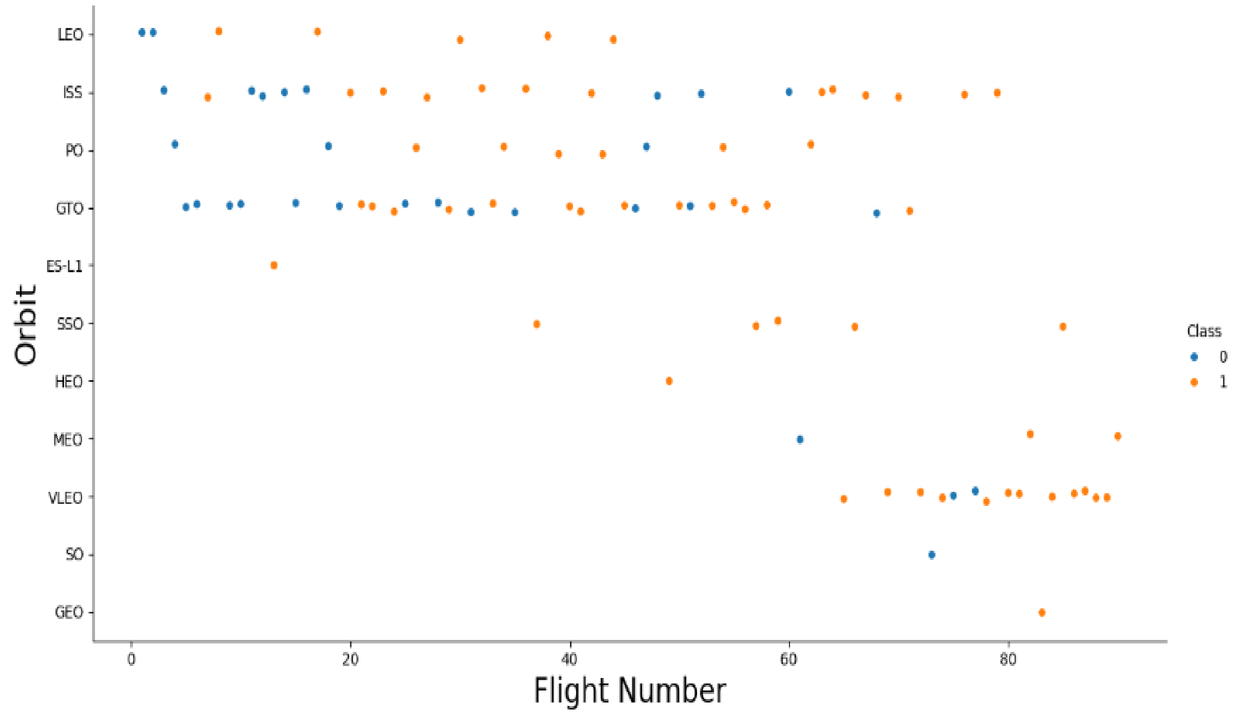
https://github.com/sondylove/testrepo/blob/main/edadataviz.ipynb

EDA With Data Visualization









EDA with SQL

The SQL queries performed are the following:

- Display the names of the unique launch sites from the space mission

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" from SPACEXTBL;
```

- Display 5 records where launch sites begin with the string "CCA"

```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site like 'CCA%' LIMIT 5;
```

- Display the total payload mass carried by boosters launched by NASA

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)",Customer FROM
'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Average Payload Mass", Booster_Version
from 'SPACEXTBL' Where Booster_Version Like 'F9 v1.1%';
```

EDA with SQL (Cont...)

- List the date when the first successful landing outcome in ground pad was achieved

```
%sql SELECT MIN(Date) from 'SPACEXTBL' WHERE
"Landing_Outcome" = "Success (ground pad)";
```

- List the name 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_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

- List the total number of successful and failure missions

```
%sql SELECT "Mission_Outcome", count("Mission_Outcome") as Total from
'SPACEXTBL' GROUP BY "Mission_Outcome";
```

https://github.com/sondylove/testrepo/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

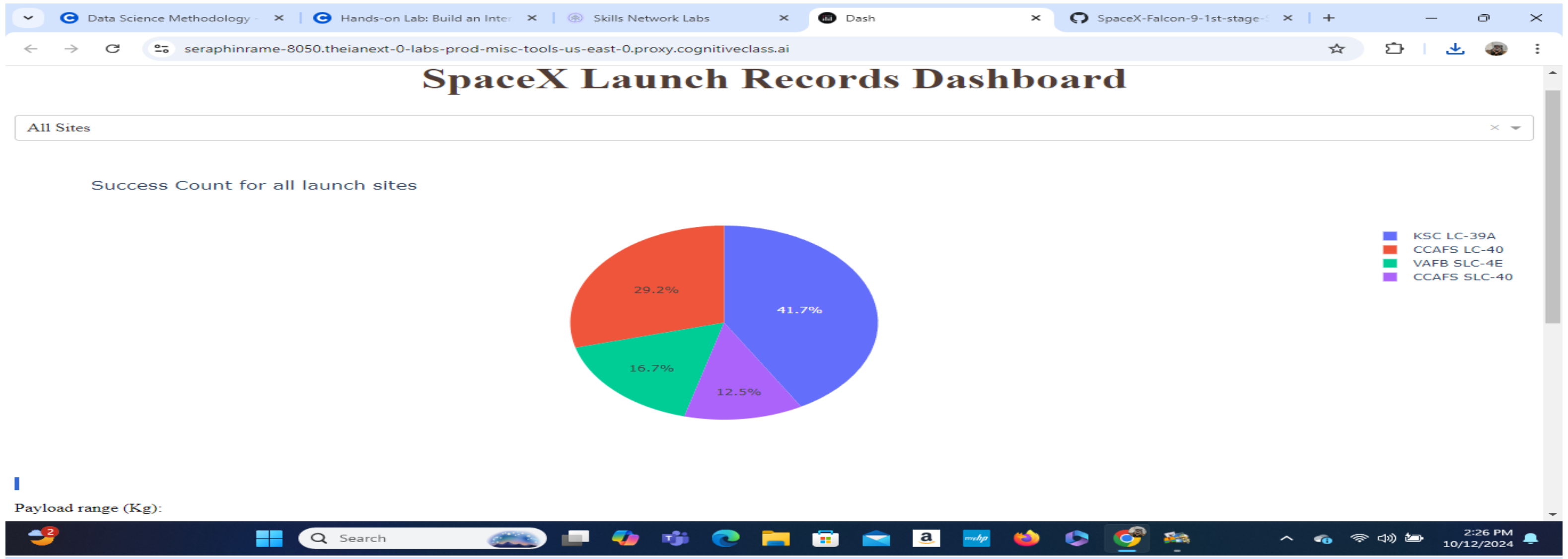
Build an Interactive Map with Folium

- We created a folium map to marked all the launch sites, and we also created map objects such as markers, circles, lines to mark the success or failure of launches of each site.
- We created a launch set outcomes where success = 1, and failure = 0.
- Here is the GitHub URL of the completed interactive map with Folium map, as an external reference and peer-review purpose(https://github.com/sondylove/testrepo/blob/main/lab_jupyter_launch_site_location%20(1).ipynb)

Build a Dashboard with Plotly Dash

- An interactive dashboard application is built with Plotly dash by:
  - Adding a Launch Site Drop-down Input Component
  - Adding a callback function to render success-pie-chart based on selected site dropdown
  - Adding a Range Slider to select Payload
  - Adding a callback function to render the success-payload-scatter-chart scatter plot
- Here the GitHub URL of the completed Plotly Dash lab, as an external reference and peer-review purpose(https://github.com/sondylove/testrepo/blob/main/7.%20Build%20an%20Interactive%20Dashboard%20with%20Ploty%20Dash%20-%20spacex_dash_app.py)

SpaceX Dash App



Data Science Methodology | Hands-on Lab: Build an Inte | Skills Network Labs | Dash | SpaceX-Falcon-9-1st-stage-

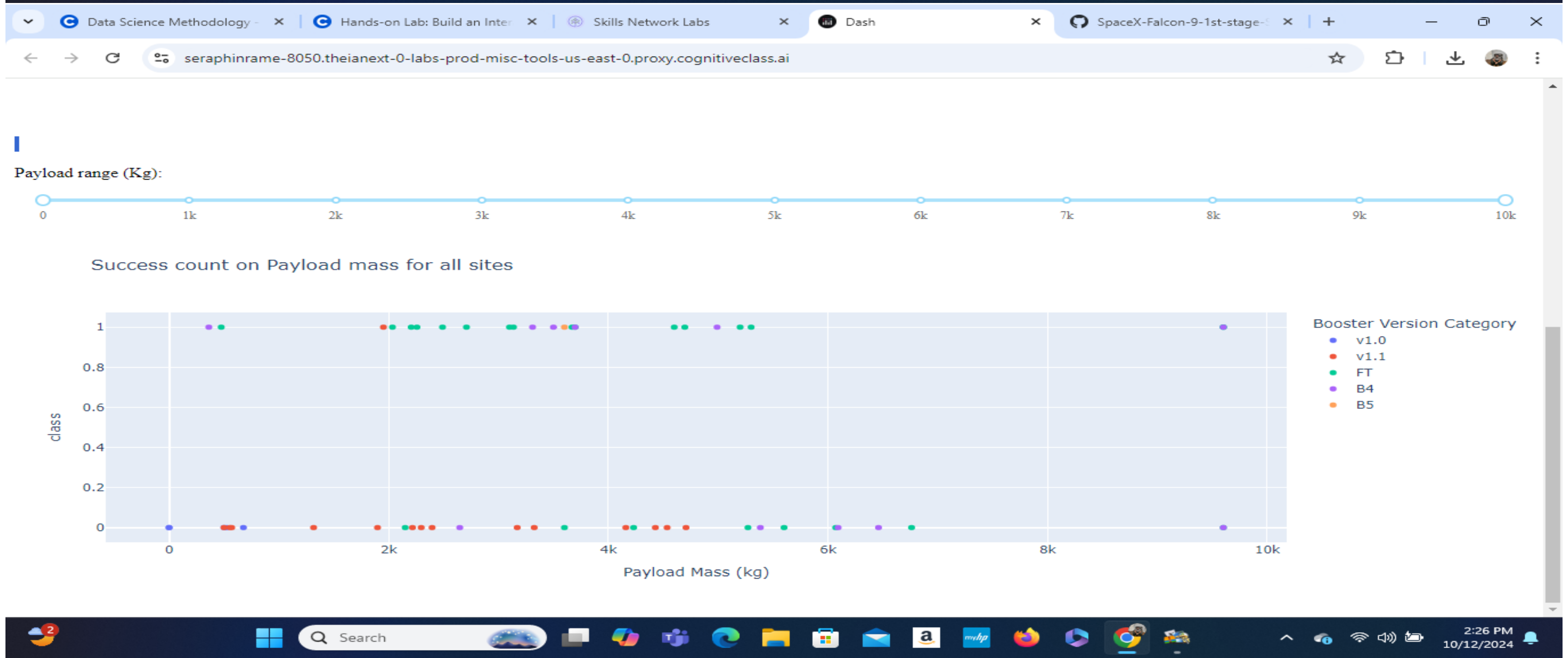
seraphinrame-8050.theianext-0-labs-prod-misc-tools-us-east-0.proxy.cognitiveclass.ai

SpaceX Launch Records Dashboard

All Sites

Success Count for all launch sites

Payload range (Kg):



Data Science Methodology | Hands-on Lab: Build an Inte | Skills Network Labs | Dash | SpaceX-Falcon-9-1st-stage-

seraphinrame-8050.theianext-0-labs-prod-misc-tools-us-east-0.proxy.cognitiveclass.ai

Payload range (Kg):

Success count on Payload mass for all sites

Predictive Analysis (Classification)

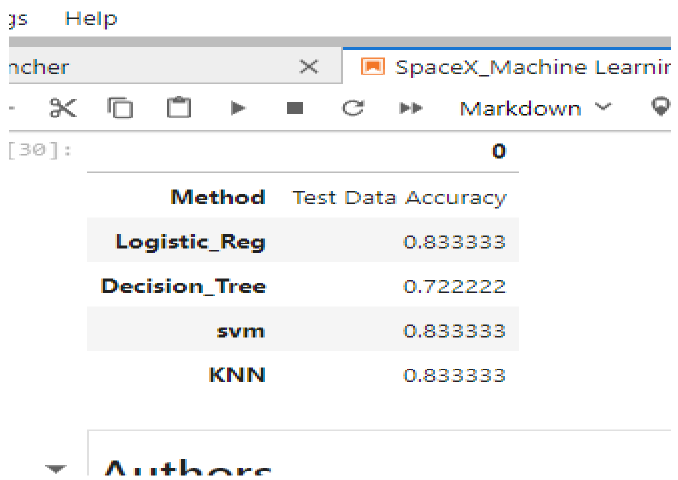
- Summary how I built, evaluated, improved, and found the best performing classification model
- After loading the data as a Pandas dataframe, I performed exploratory data analysis on the dataset and determine training labels by:
  - Creating a Numpy array with the column Class by applying the method to_numpy() then assigned it to the variable Y as the outcome variable
  - Standardized the feature dataset (x) by transforming it using preprocessing, StandardScaler() function from Sklearn.
  - The data was then split into training and testing sets using the function train_test_split from sklearn.model_selection with the test_size parameter set to 0.2 and random_state to 2.
- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

Predictive Analysis(Classification)

- For us to find the best Machine Learning(ML) model that would perform best using the test data between SVM, Classification Trees, K Nearest Neighbor and Logistic Regression:
  - We created an object for each of the algorithms then created a GridSearchCV object and assigned them a set of parameters for each model.
  - For each of the model under evaluation, the GridSearchCV object was created with cv=10, then fit the training data into the GridSearch object for each to find best Hyperparameter
  - After fitting the training set, we output GridSearchCV object for each of the models, then displayed the best parameters using the data attribute best_score_.
  - At the end using the method score to calculate the accuracy on the test data for each model and plotted a confusion matrix for each using the test and predicted outcomes.

Predictive Analysis(Classification)

- The following table shows the test data accuracy score for each of the mrethods comparing them to show which performed best using the test data between SVM, Classification Trees, K Nearest Neighbors and Logistic Regression:



```
[30]:
                                   0
        Method   Test Data Accuracy
   Logistic_Reg            0.833333
  Decision_Tree            0.722222
            svm            0.833333
            KNN            0.833333
```

Authors

- GitHub URL of the completed predictive analysis lab(https://github.com/sondylove/testrepo/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5%20(1).ipynb)

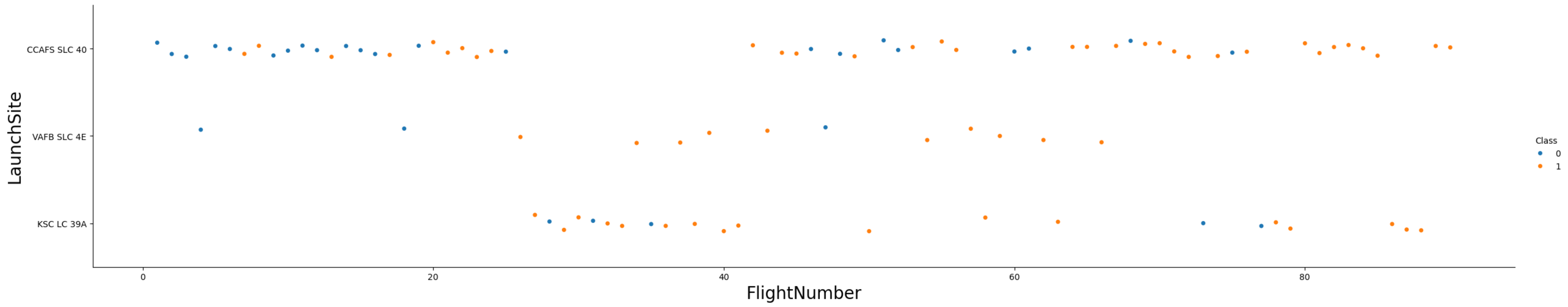
Results

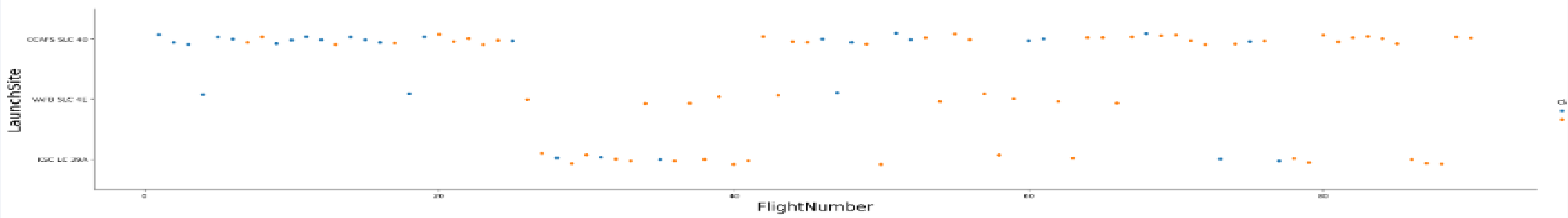
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

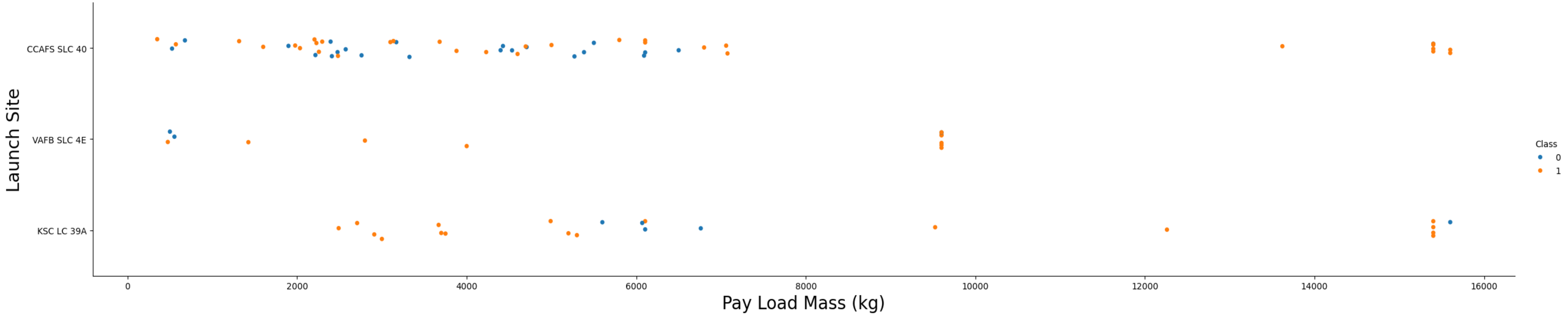


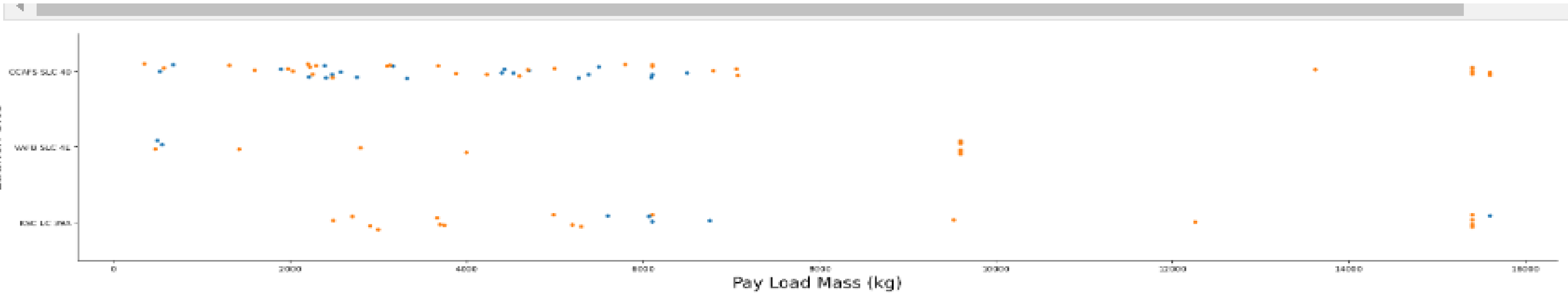


Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

```
# From the plot above, we can deduce that as the flight number increases so is the success rate.
```

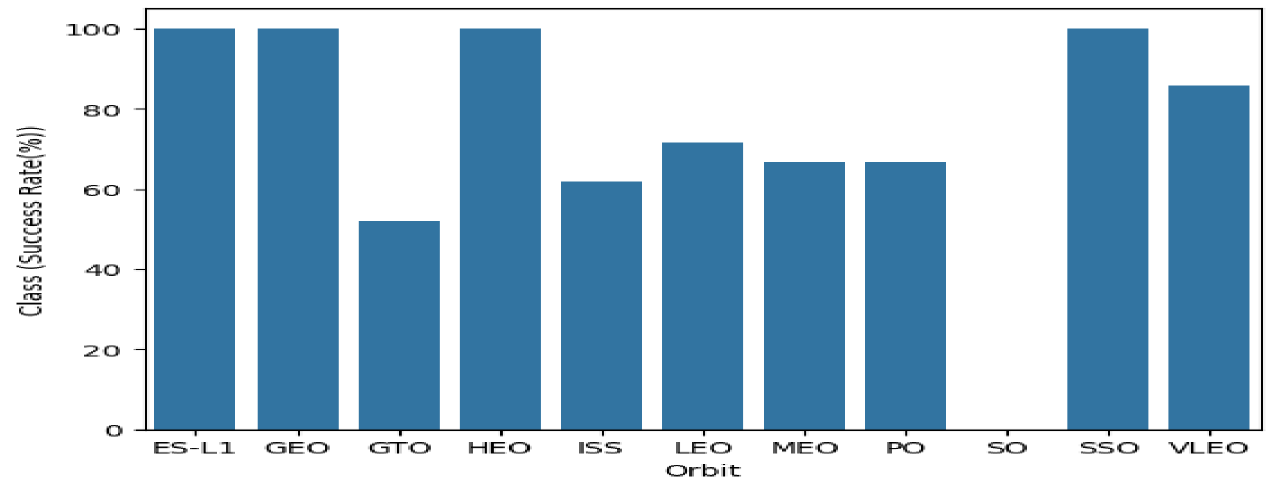
Payload vs. Launch Site

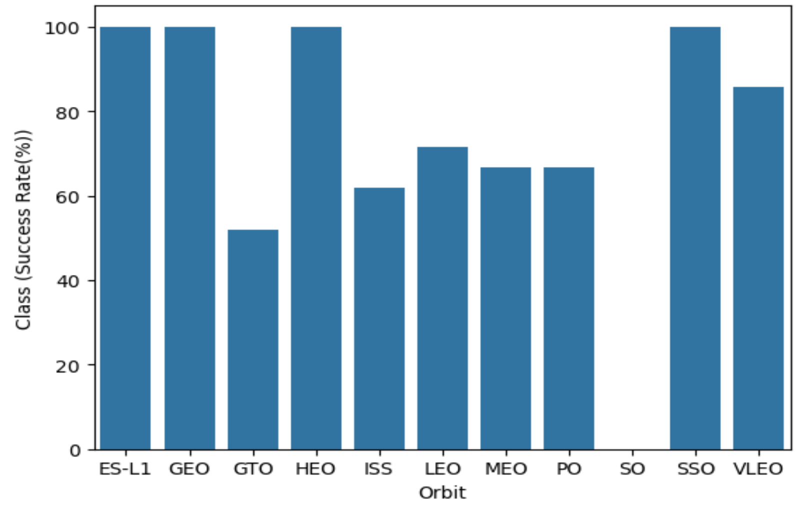




Now if you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

Success Rate vs. Orbit Type

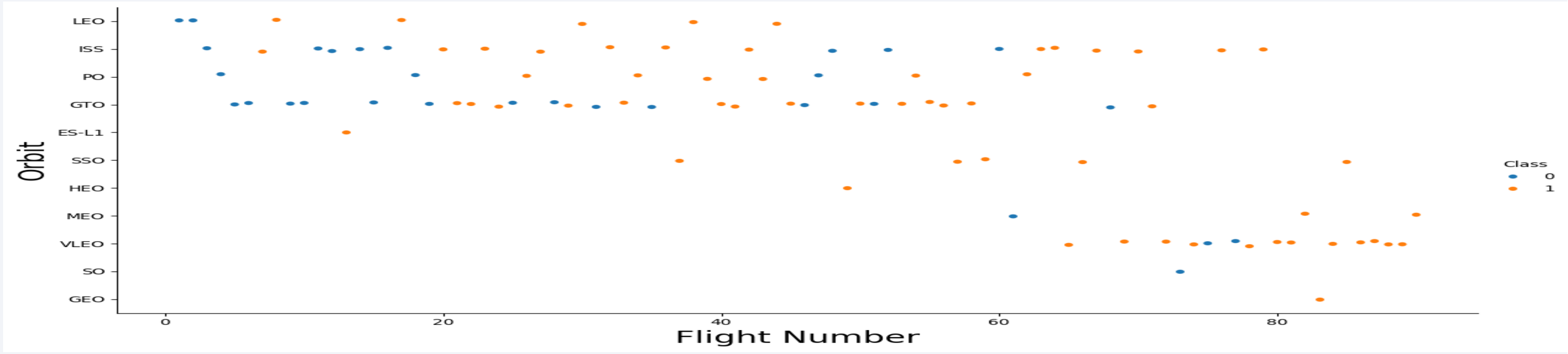


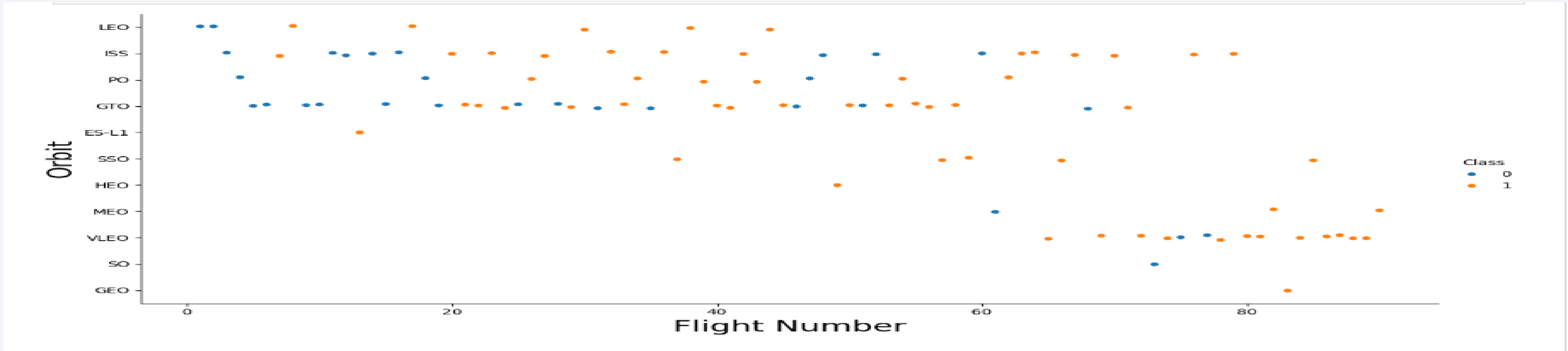


Analyze the plotted bar chart to identify which orbits have the highest success rates.

```
[ ]: # Orbits ES-L1, GEO, HEO and SSO have the highest success rate (100%). Orbit GTO has the lowest success rate(~50%). Fin
```

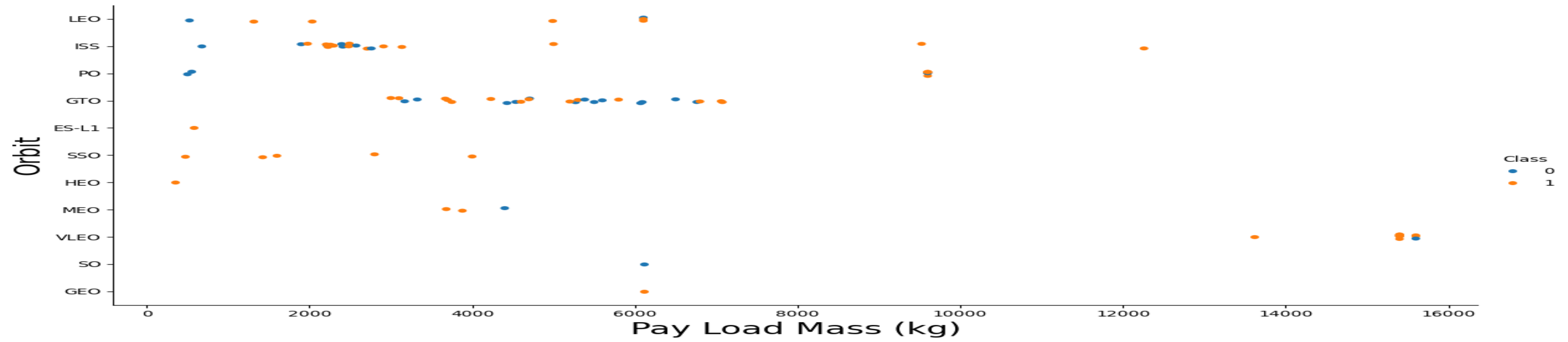
Flight Number vs. Orbit Type

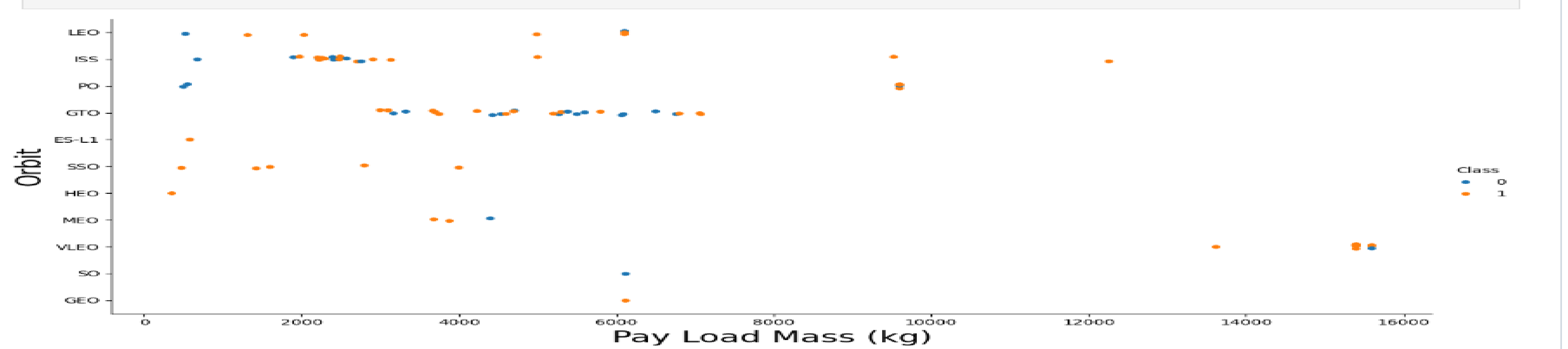




You can observe that in the LEO orbit, success seems to be related to the number of flights. Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.

Payload vs. Orbit Type

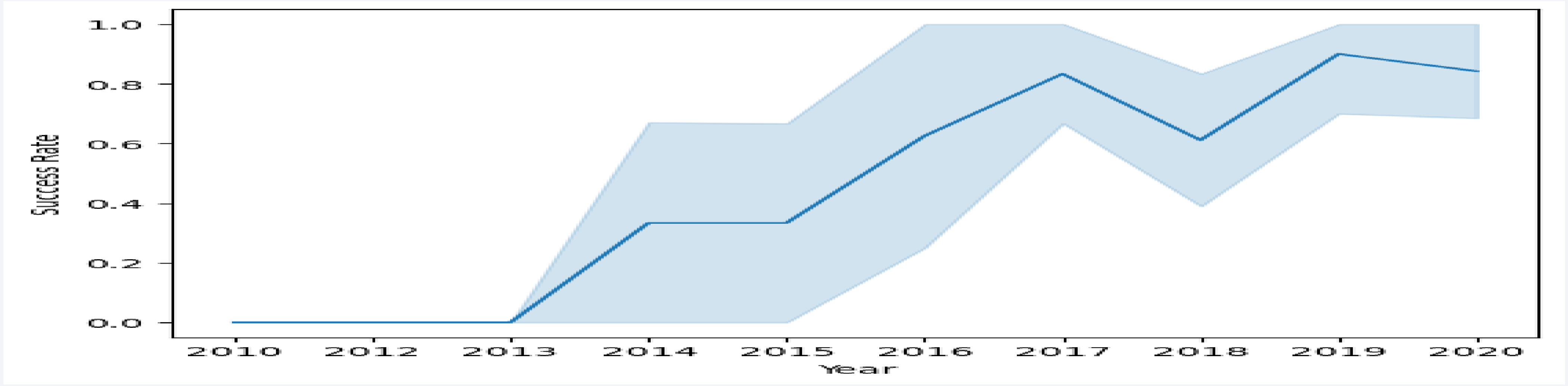


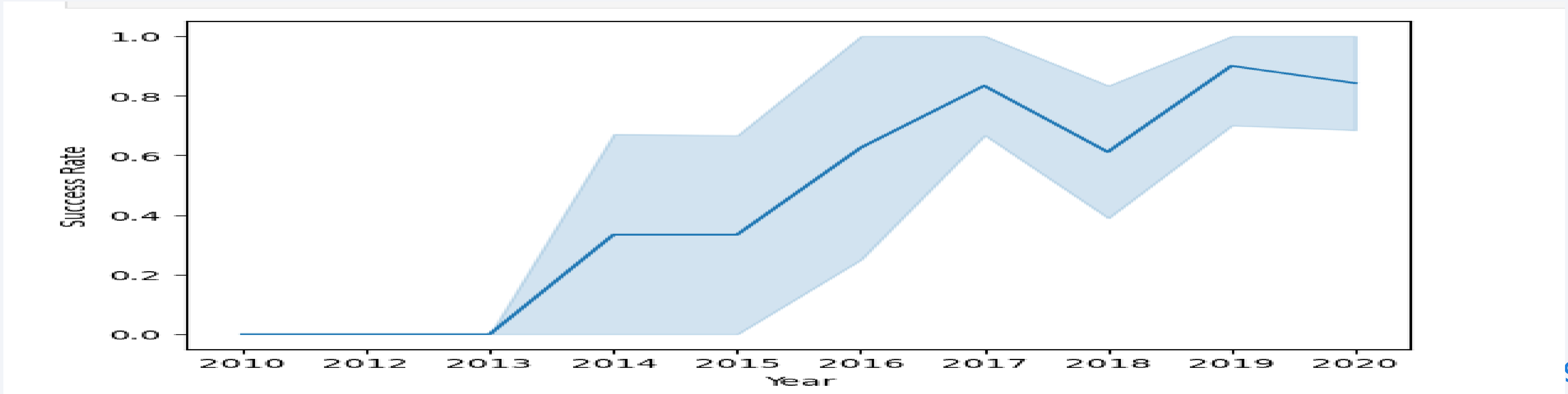


With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.

Launch Success Yearly Trend

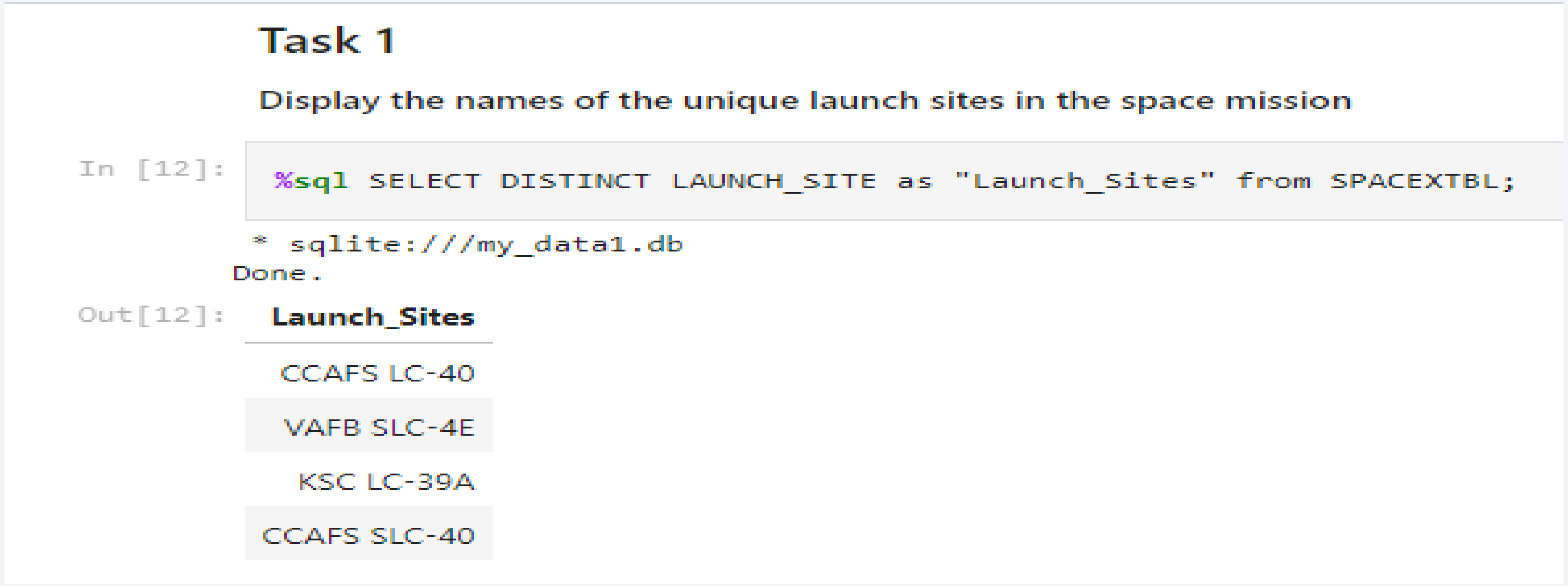




you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

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



Task 1

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

```
In [12]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" from SPACEXTBL;
```

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

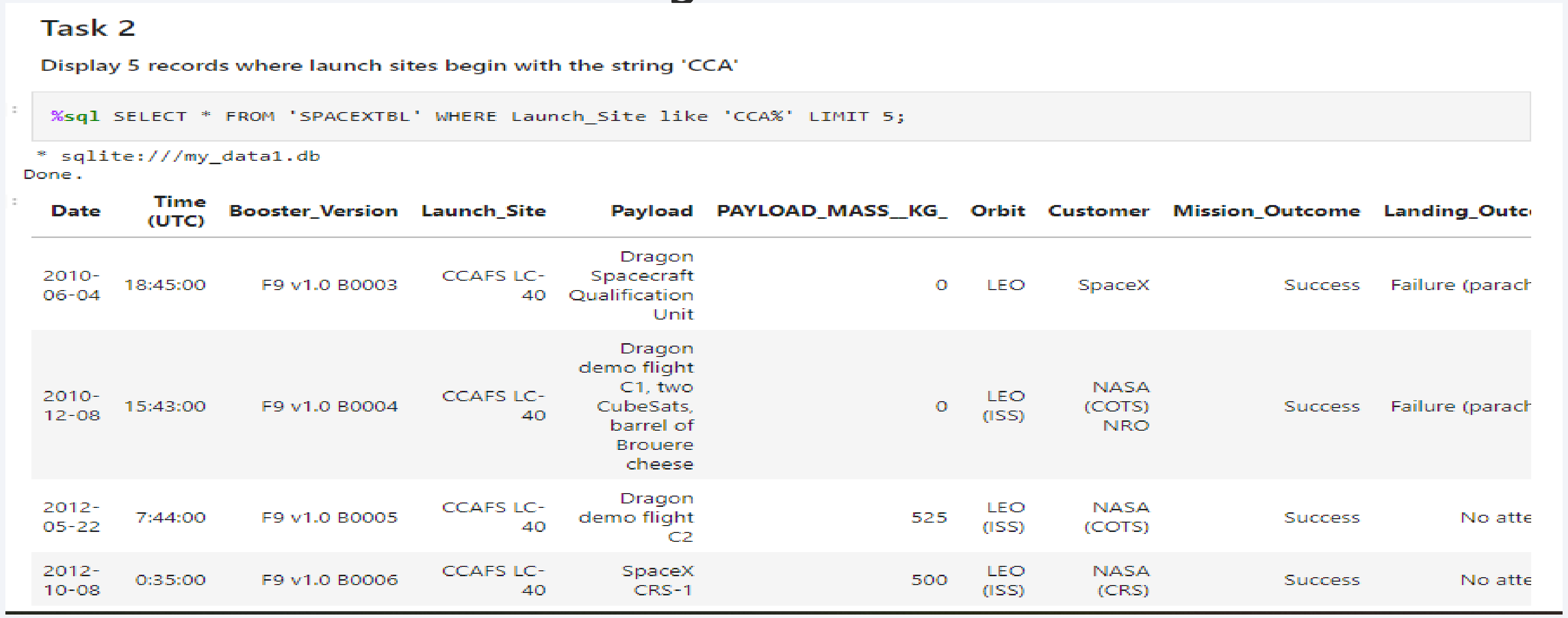
Out[12]:

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

- We use SELECT DISTINCT to return only the unique launch sites from the SPACEXTBL table.

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`



Task 2

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

```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site like 'CCA%' LIMIT 5;
```

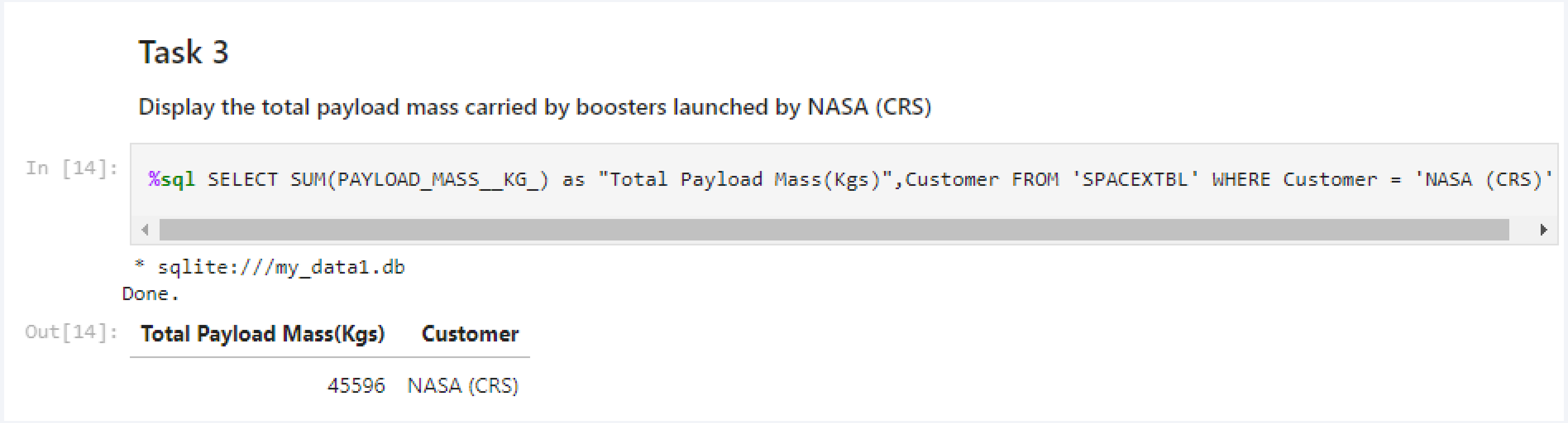
```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outc
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parach
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 (parach
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No atte
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No atte

- We used "Like" command with "%" wildcard in "Where" clause to select and display a table of all records where launch sites begin with the string "CCA".

Total Payload Mass

- Calculate the total payload carried by boosters from NASA



Task 3

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

```
In [14]: %sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)",Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)'
```

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

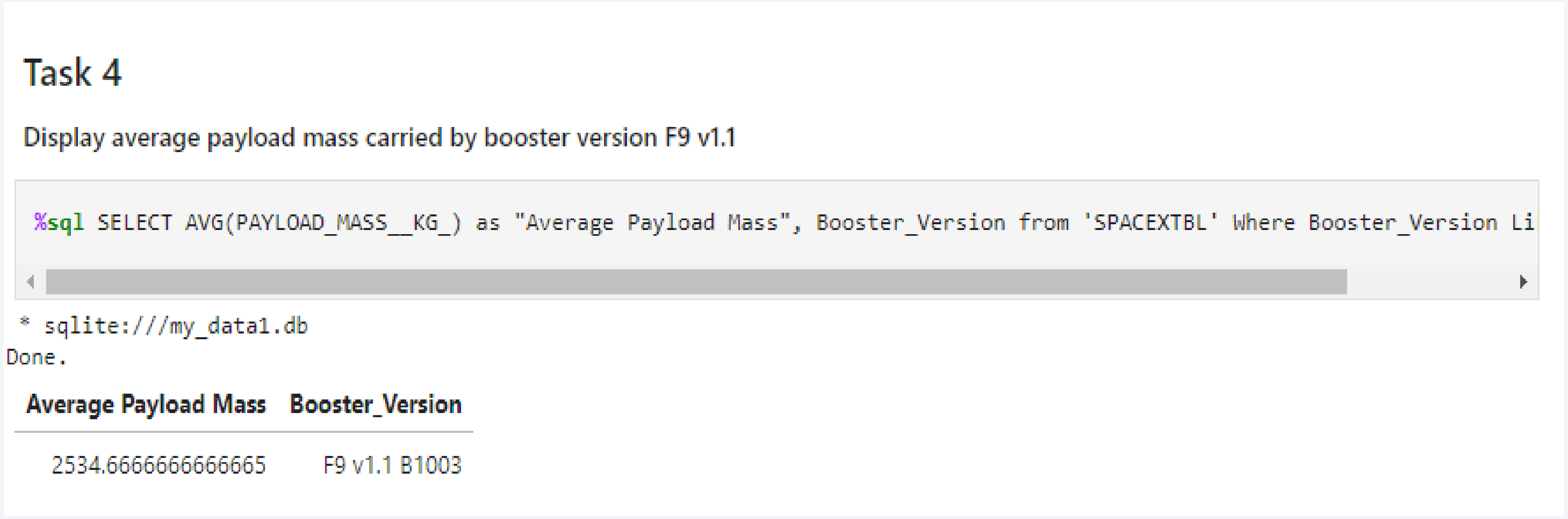
Out[14]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

- We use the SUM() function to return and didsplay the total sum of 'PAYLOAD_MASS_KG' column for customer NASA.

Average Payload Mass by F9 v1.1

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



Task 4

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Average Payload Mass", Booster_Version from 'SPACEXTBL' Where Booster_Version Li
```

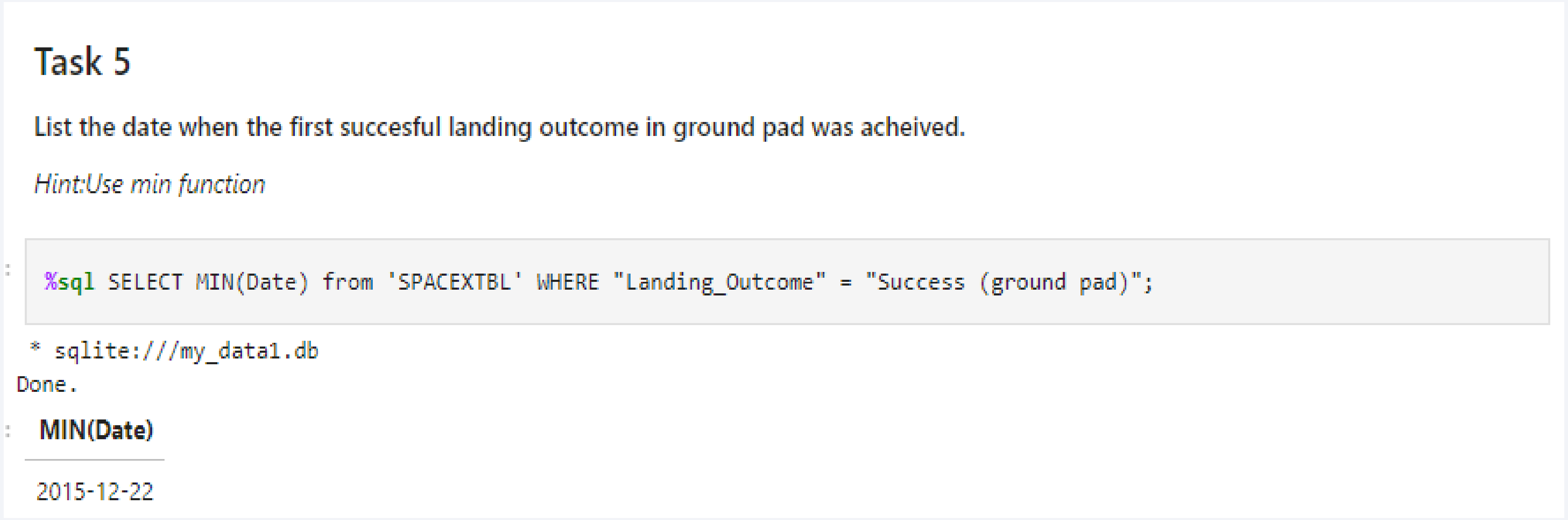
```
 * sqlite:///my_data1.db
Done.
```

Average Payload Mass	Booster_Version
2534.6666666666665	F9 v1.1 B1003

- We use the AVG() function to return and display the average payload mass carried by booster version F9 v1.1

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on 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) from 'SPACEXTBL' WHERE "Landing_Outcome" = "Success (ground pad)";
```

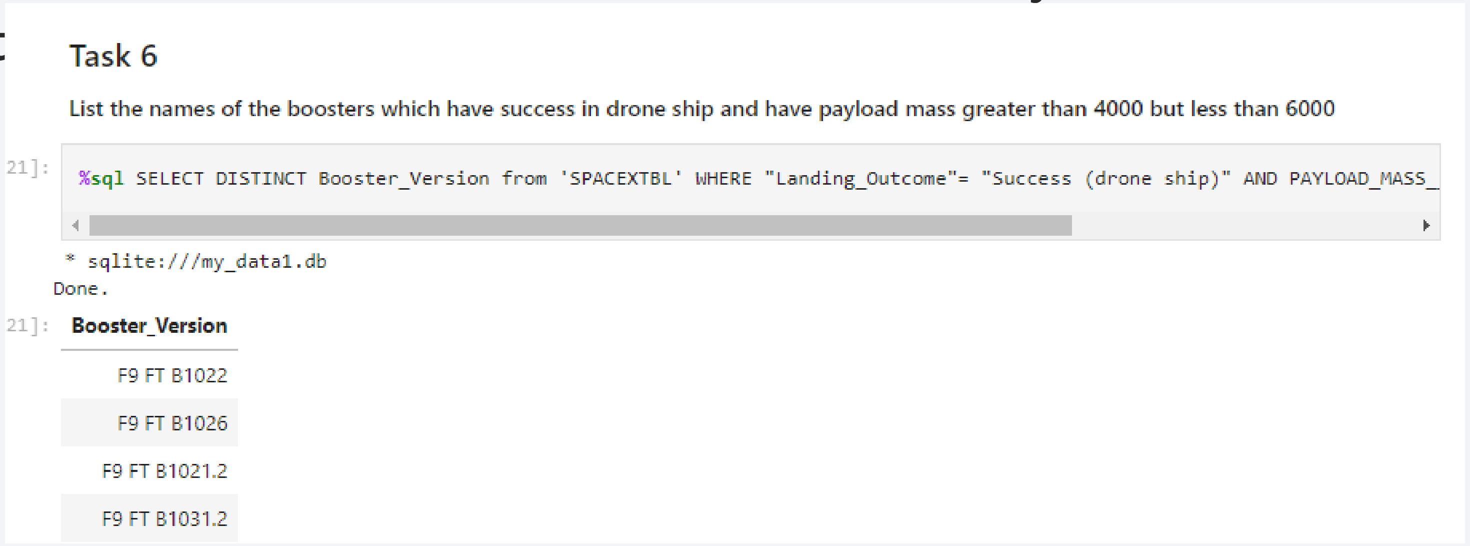
```
 * sqlite:///my_data1.db
Done.
```

MIN(Date)
2015-12-22

- We use the MIN() function to return and display the first (oldest) date when first successful landing outcome on ground pad happened.

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater t



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

```
21]: %sql SELECT DISTINCT Booster_Version from 'SPACEXTBL' WHERE "Landing_Outcome"= "Success (drone ship)" AND PAYLOAD_MASS_
```

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

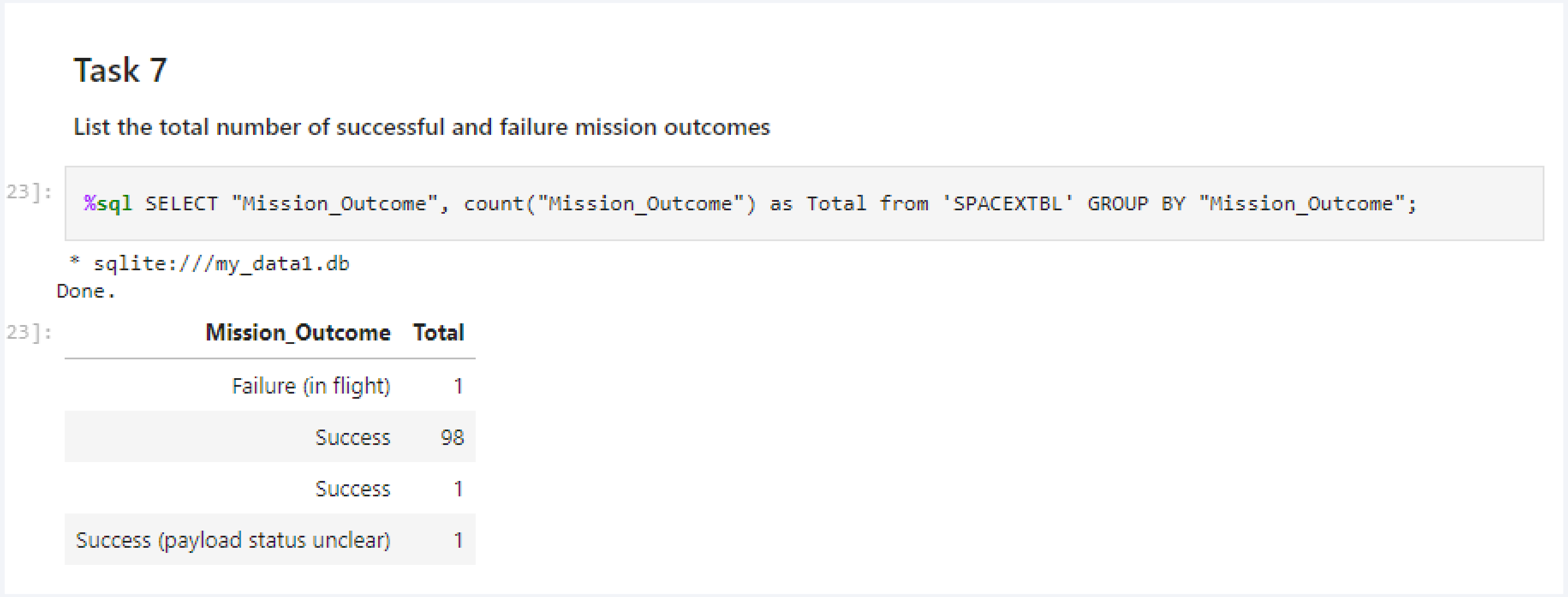
21]:

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

- We use 'SELECT DISTINCT' to return and list the uniques names of boosters with operators >4000 and <6000 to only list boosters with payloads between 4000-6000 with landing outcome of 'Success(drone ship)'.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes



Task 7

List the total number of successful and failure mission outcomes

```
23]:  %sql SELECT "Mission_Outcome", count("Mission_Outcome") as Total from 'SPACEXTBL' GROUP BY "Mission_Outcome";
```

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

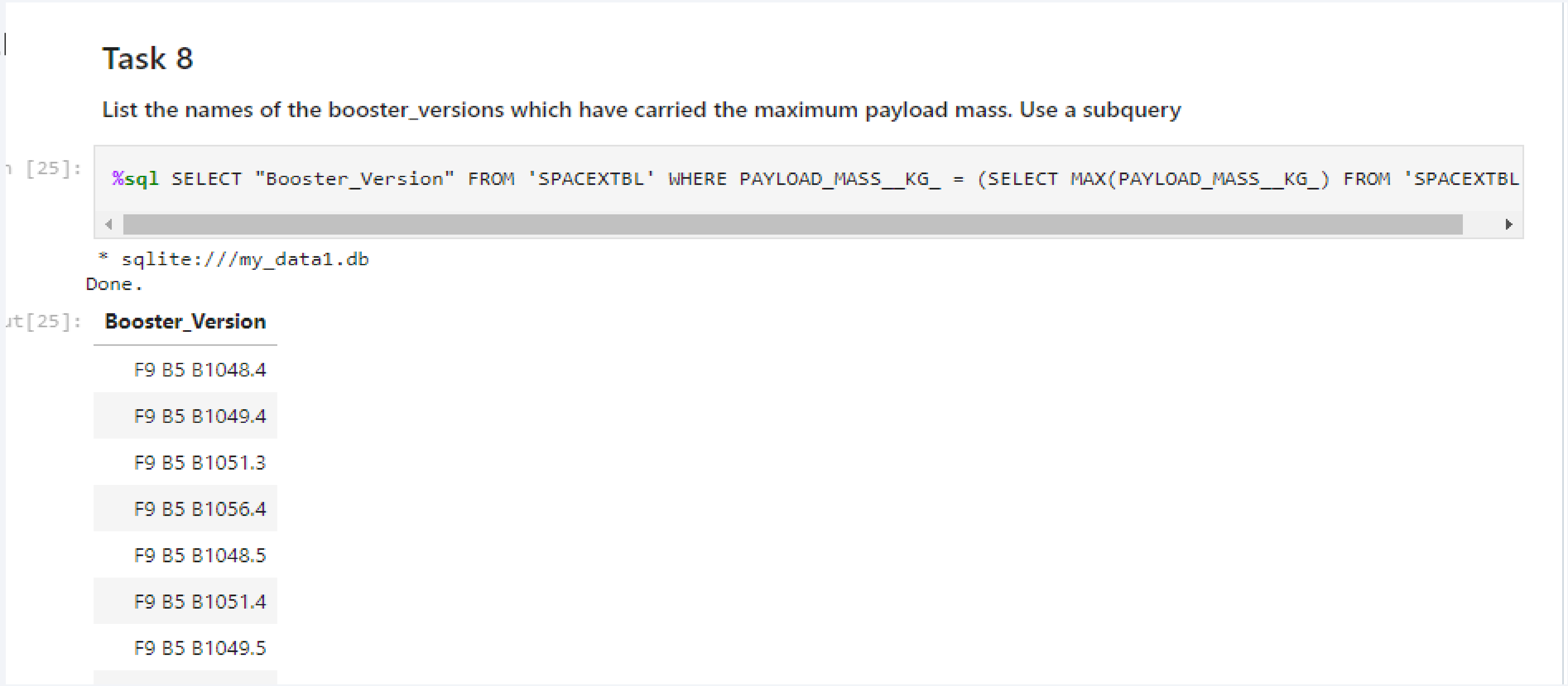
23]:

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

- We use the COUNT() with the 'Group By' to return the total number of mission outcomes.

Boosters Carried Maximum Payload

- List the na



Task 8

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

```
In [25]: %sql SELECT "Booster_Version" FROM 'SPACEXTBL' WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM 'SPACEXTBL
```

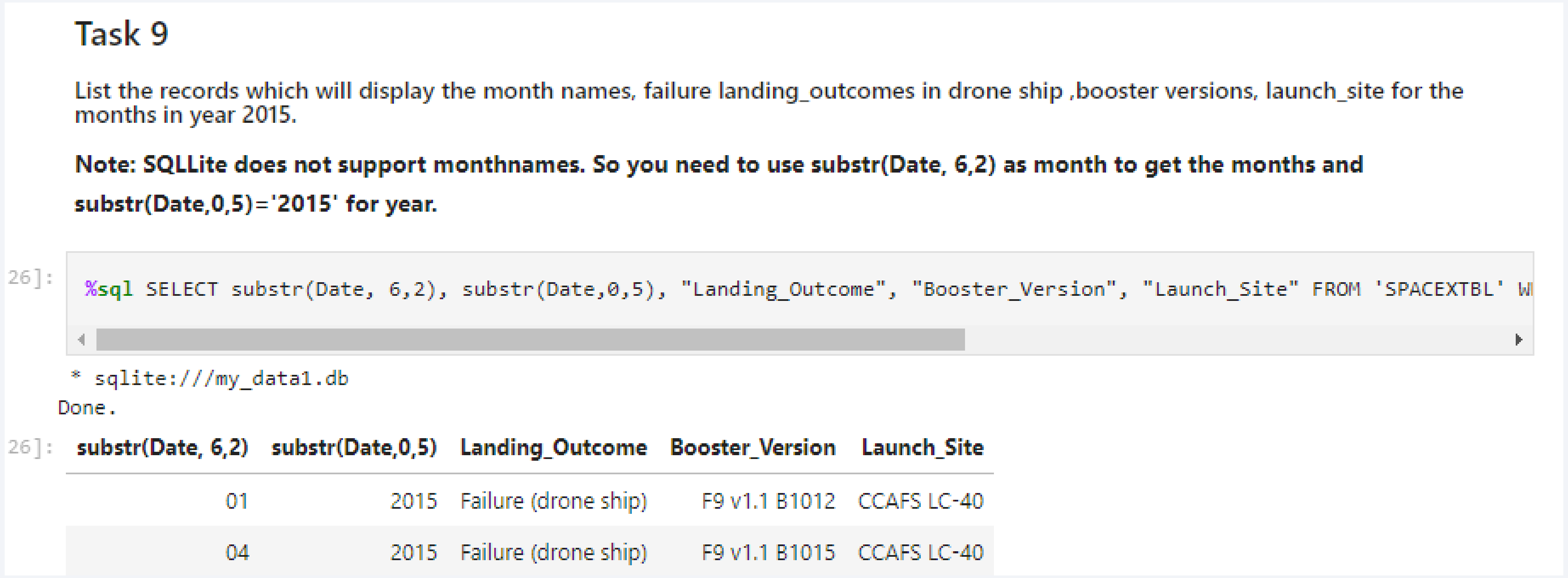
```
 * sqlite:///my_data1.db
Done.
```

Out[25]:

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

- We use a subquery to return and pass the Max payload and used. It list all the boosters that carried the Max payload of 15600kgs.

2015 Launch Records



Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

```
26]:  %sql SELECT substr(Date, 6,2), substr(Date,0,5), "Landing_Outcome", "Booster_Version", "Launch_Site" FROM 'SPACEXTBL' WH
```

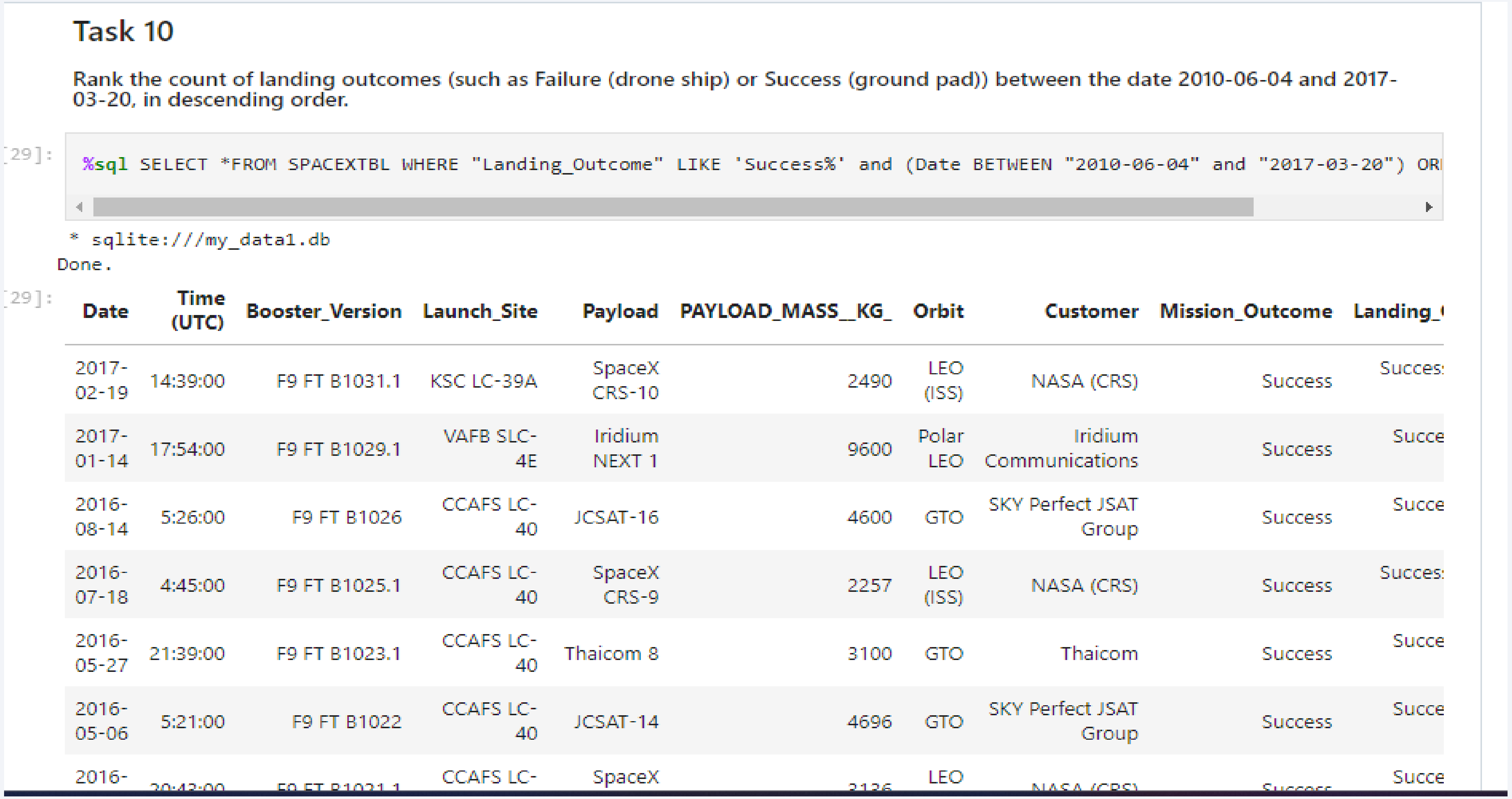
```
 * sqlite:///my_data1.db
Done.
```

26]:

substr(Date, 6,2)	substr(Date,0,5)	Landing_Outcome	Booster_Version	Launch_Site
01	2015	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	2015	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- We used 'substr()' function in the SELECT statement to get the month and year from the date column where substr(Date,7,4)='2015' for year and Landing_outcome was 'Failure (drone ship)' and return the records matching the filter.

Rank Landing Outcomes 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.

```
[29]:  %sql SELECT *FROM SPACEXTBL WHERE "Landing_Outcome" LIKE 'Success%' and (Date BETWEEN "2010-06-04" and "2017-03-20") OR
```

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

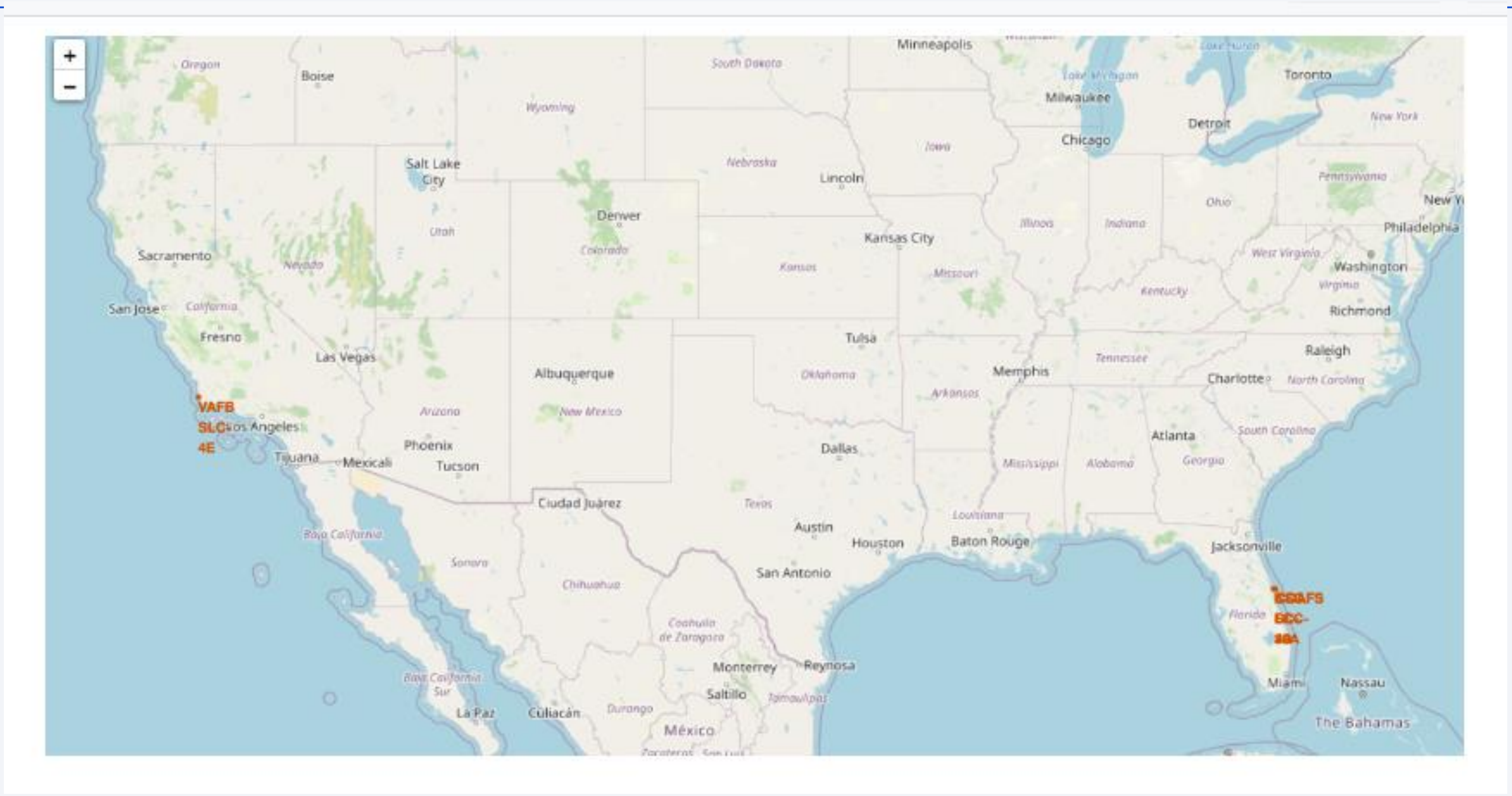
[29]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_(
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Succes
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Succe
2016-08-14	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Succe
2016-07-18	4:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Succes
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Succe
2016-05-06	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Succe
2016-	20:43:00	F9 FT B1021.1	CCAFS LC-	SpaceX	3136	LEO	NASA (CRS)	Success	Succe

Section 3

Launch Sites Proximities Analysis

Markers of All Launch Sites on Global Map



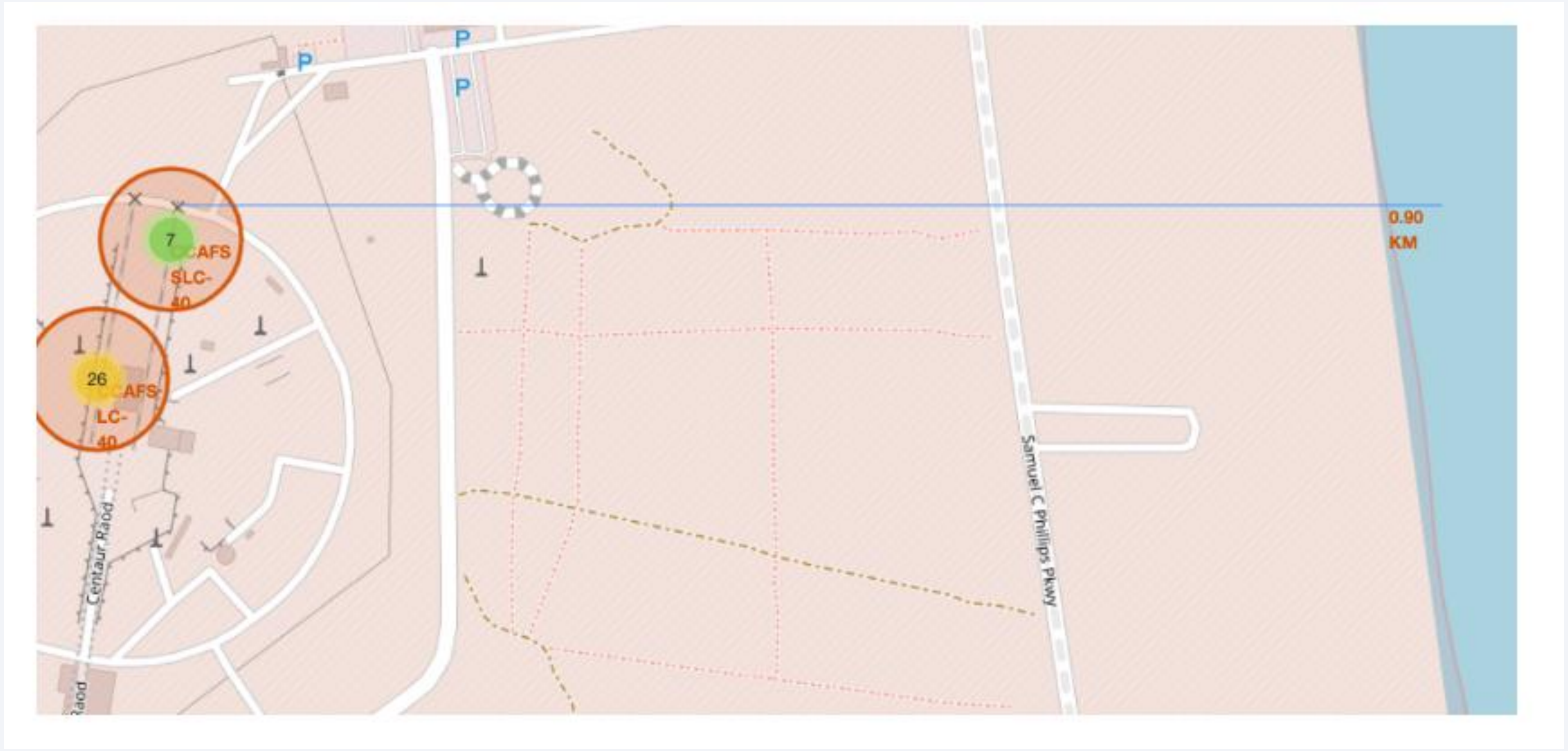
- All launch sites are in proximity to the equator, also all launch sites are in very close proximity to the coast.

Launch Outcomes for Each Site with Color Markers

CCAFS SLC-40
26
CCAFS LC-40
P

Launch site KSC LC-39A has relatively high success rates compared to CCAFS SLC-40 & CCAFS LC-4

Distance Between a Launch Site to its Proximities

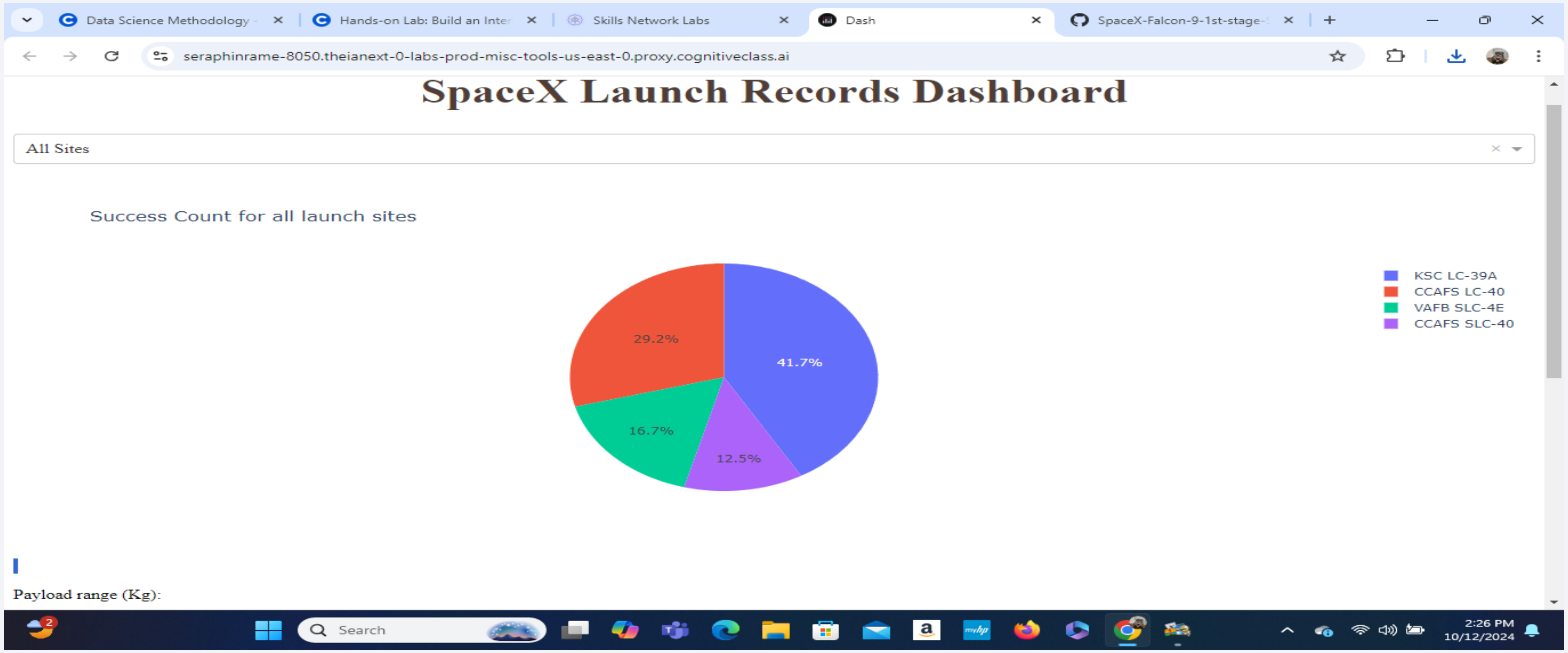


- Launch site CCAFS SLC-40 proximity to coastline is 0.86km.

Section 4

Build a Dashboard with Plotly Dash

Pie Chart for Launch Success Count for All sites


Data Science Methodology
Hands-on Lab: Build an Inter
Skills Network Labs
Dash
SpaceX-Falcon-9-1st-stage-
seraphinrame-8050.theianext-0-labs-prod-misc-tools-us-east-0.proxy.cognitiveclass.ai
Search
2:26 PM
10/12/2024

SpaceX Launch Records Dashboard

All Sites

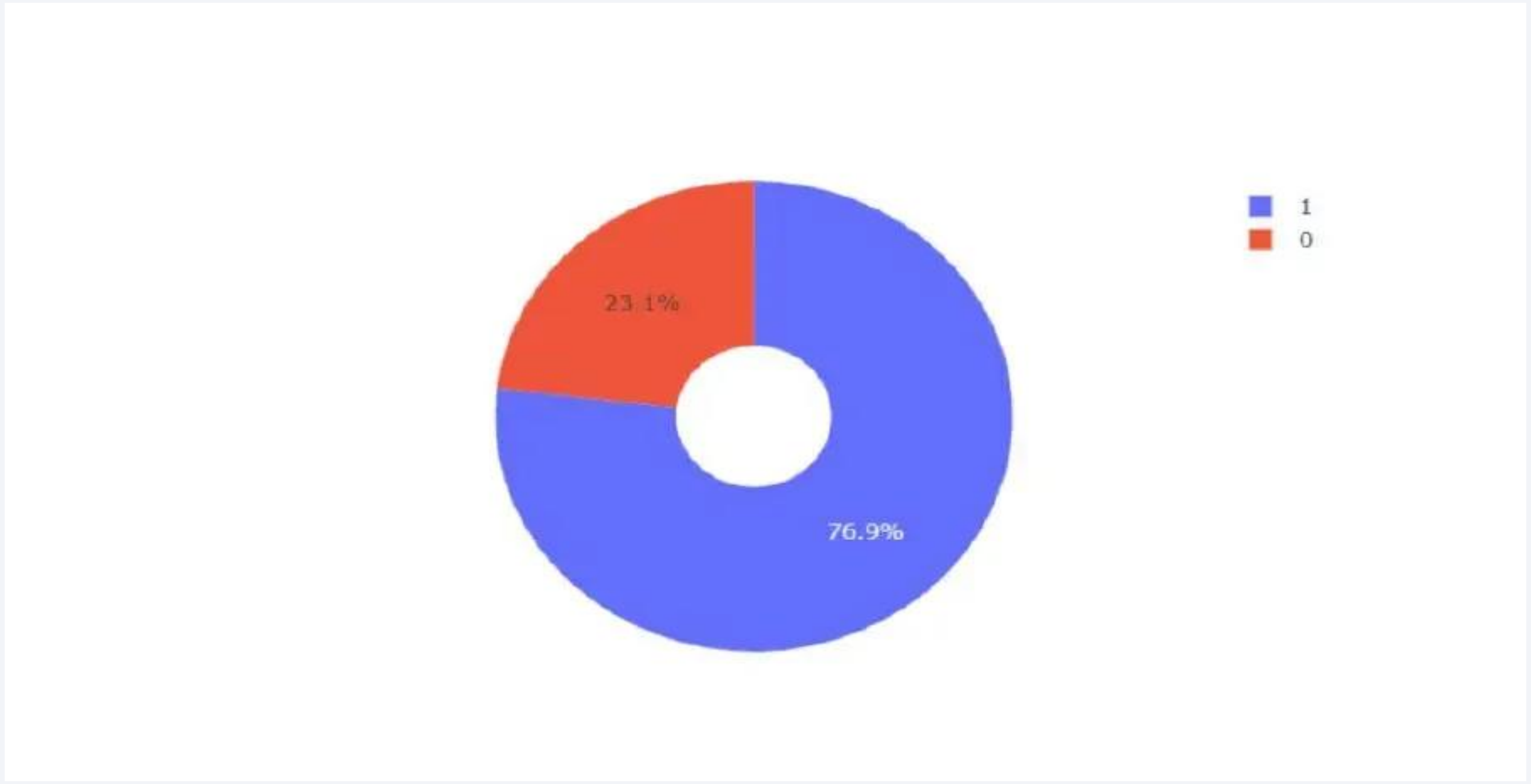
Success Count for all launch sites

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

Payload range (Kg):

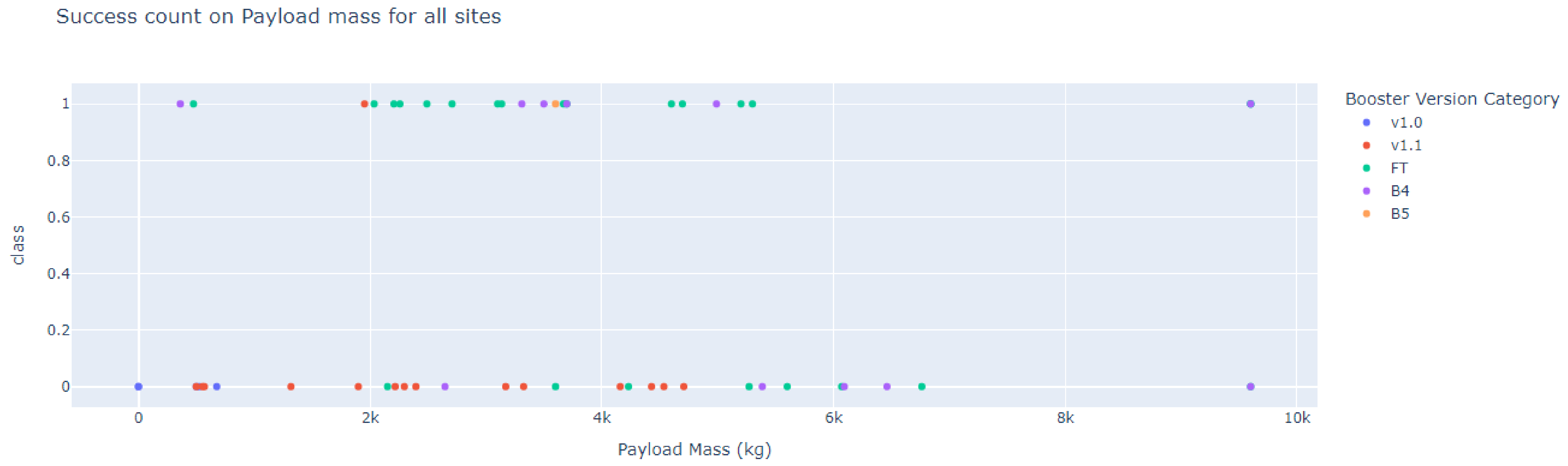
- Launch site KSC LC-39A has the highest success rate 42%, followed by CCAFS LC-40 29%, VAFB SLC-4E at 17% and finally CCAFS SLC_40 with a success rate of 13%.

Pie Chart For The Launch Site With Highest Launch Success



- Launch Site CCAFS LC-40 had the highest success ratio compared to all the others

Payload vs. Launch Outcome Scatter Plot For All Sites

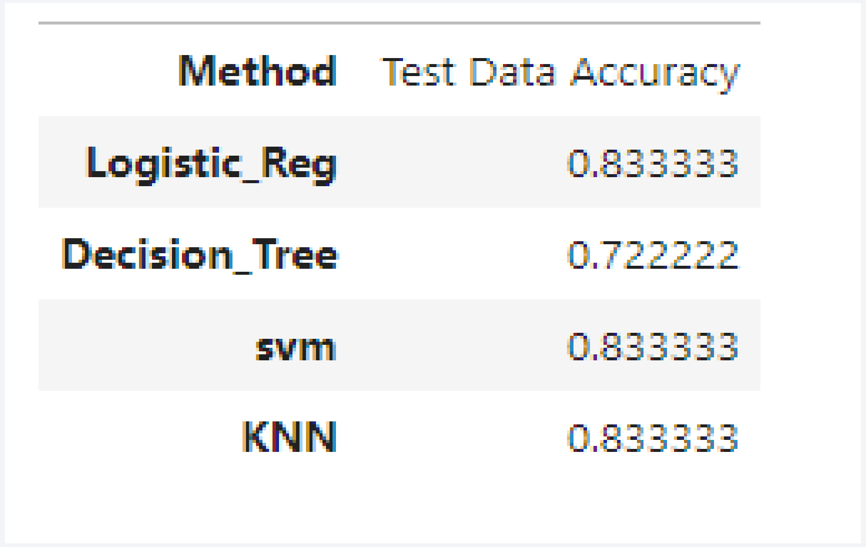


- For launch site CCFAS LC-40, the booster version FT has the larger success rate for Payload mass greater than 2000 kg (>2000kg)

Section 5

Predictive Analysis (Classification)

Classification Accuracy

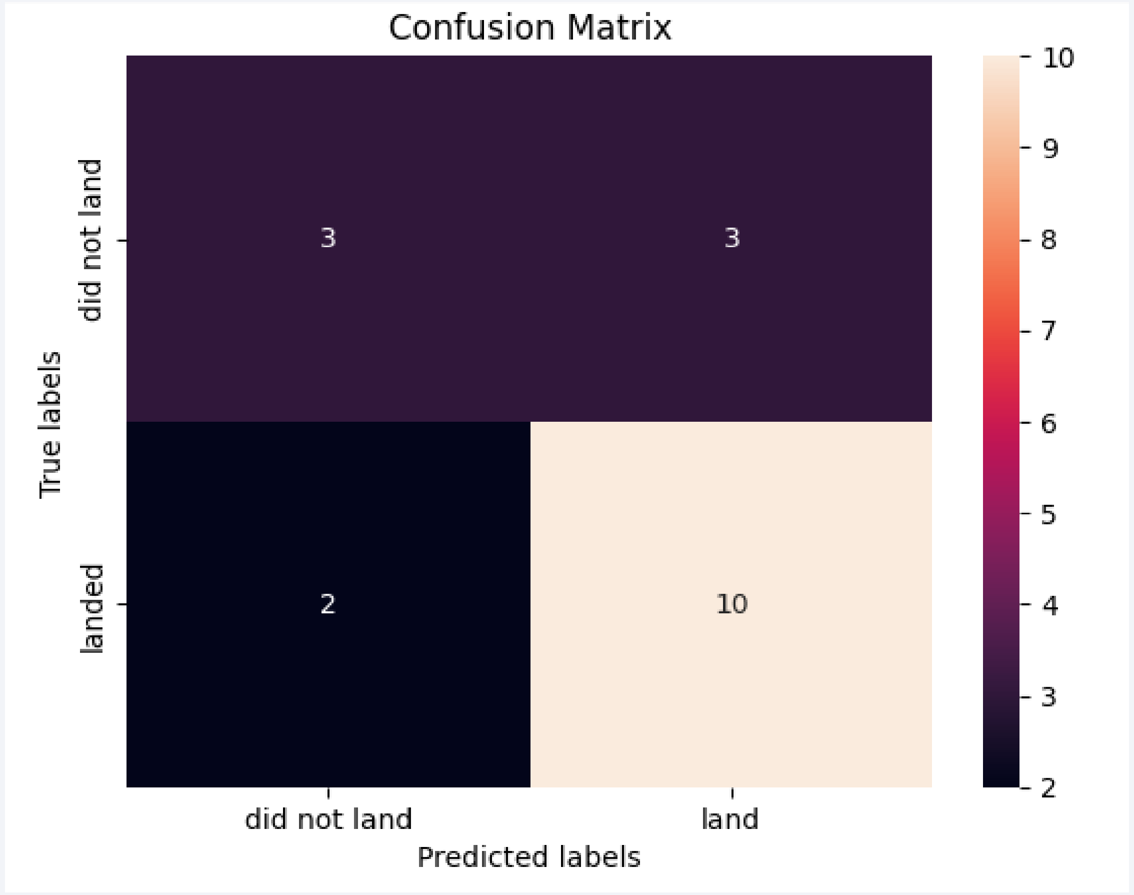


Method	Test Data Accuracy
Logistic_Reg	0.833333
Decision_Tree	0.722222
svm	0.833333
KNN	0.833333

- All models nearly performed equally on the test data with an accuracy of 0.83333 except for the decision tree with an accuracy of 0.722222.

Confusion Matrix

All The classification models have the same confusion matrices and are able to distinguish between classes.



Conclusions

- Different launch sites have different success rates. CCFAS LC-40 has a success rate of 60%, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.
- As the flight number increases in each of the launch sites, so does the success rate. Like VAFB SLC 4E which has a success rate of 100% after flight number 50. And both CCFAS LC-40 and KSC LC-39A have a success rate of 100% after flight number 80.
- There are no rockets launched for heavy payload mass (>10000 kg) in the VAFB SLC 4E site when observing the payload vs. Launch site scatter plot.
- Orbit ES-L1, GEO, HEO and SSO have a success rate of 100%. Orbit SO has The lowest success rate which is close to 0%.
- For the LEO orbit, the success rate seemed related to the number of flights. Which seemed not to be the case for the GTO orbit.
- With heavy payload, The successful land are more for POLAR, LEO and ISS.
- The success rate is increasing with time.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!