IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Abhijna S
26-02-2024

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

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

  Space X 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 Space X 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- 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

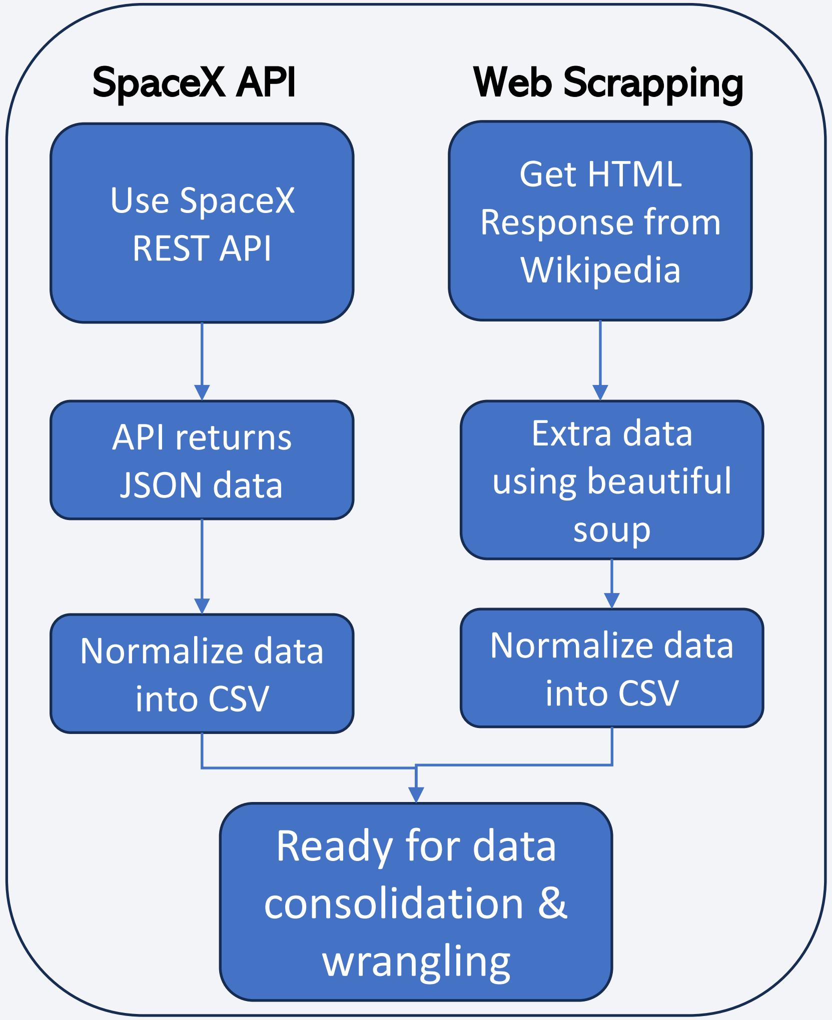
Methodology

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
  - LR, KNN, SVM, DT models have been built and evaluated to find the best classifier

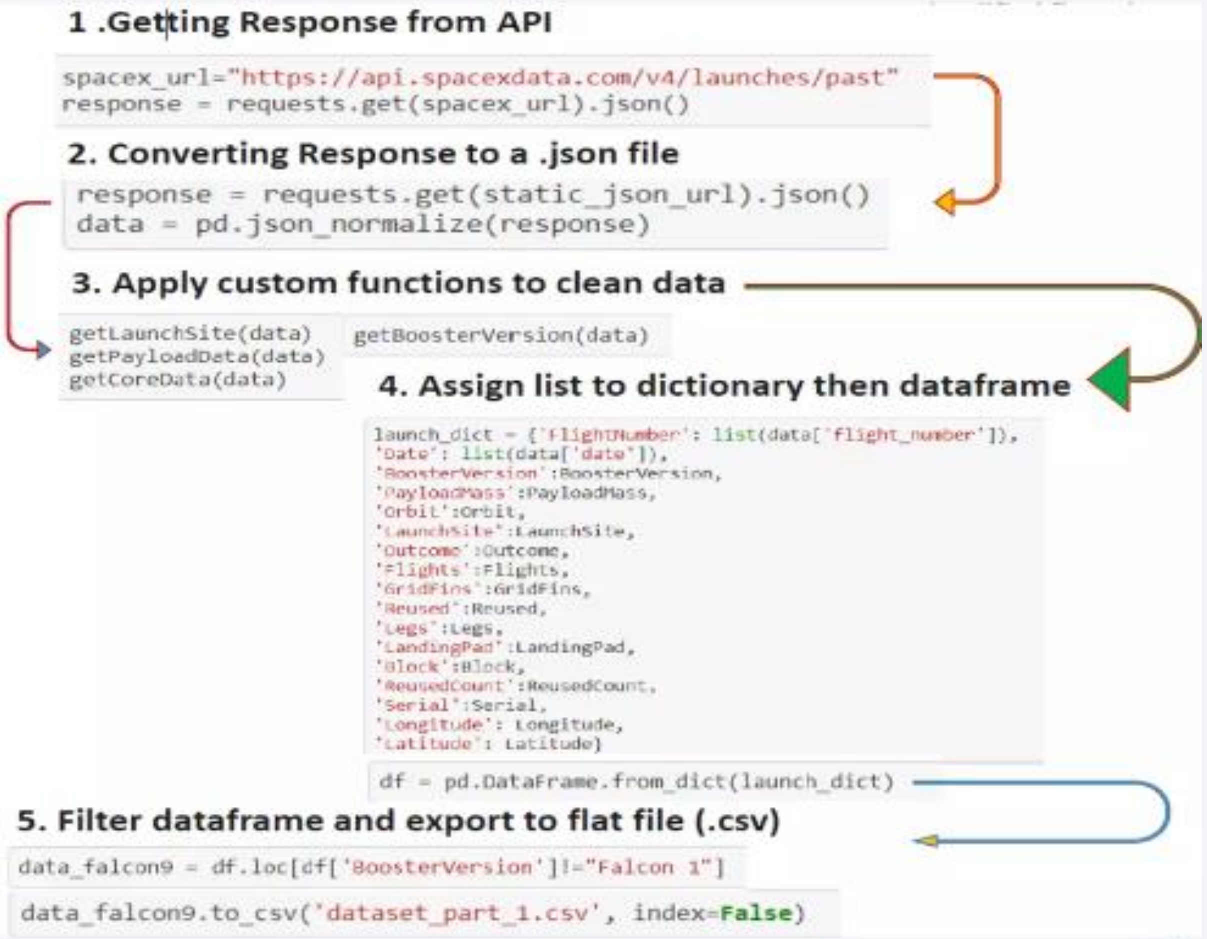
Data Collection

- 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 dataframe 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 dataframe for future analysis.



Data Collection – SpaceX API

- Used the get request to the SpaceX API to collect data, clean the requested data and did some basic manipulation and exported it as a CSV file.
- [IBM-Applied-Data-Science-Capstone/00_SpaceX_Data_Collection.ipynb at main · abhijna123/IBM-Applied-Data-Science-Capstone · GitHub]()



1 .Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url).json()
```

2. Converting Response to a .json file

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Apply custom functions to clean data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
```

```
getBoosterVersion(data)
```

4. Assign list to dictionary then dataframe

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

```
df = pd.DataFrame.from_dict(launch_dict)
```

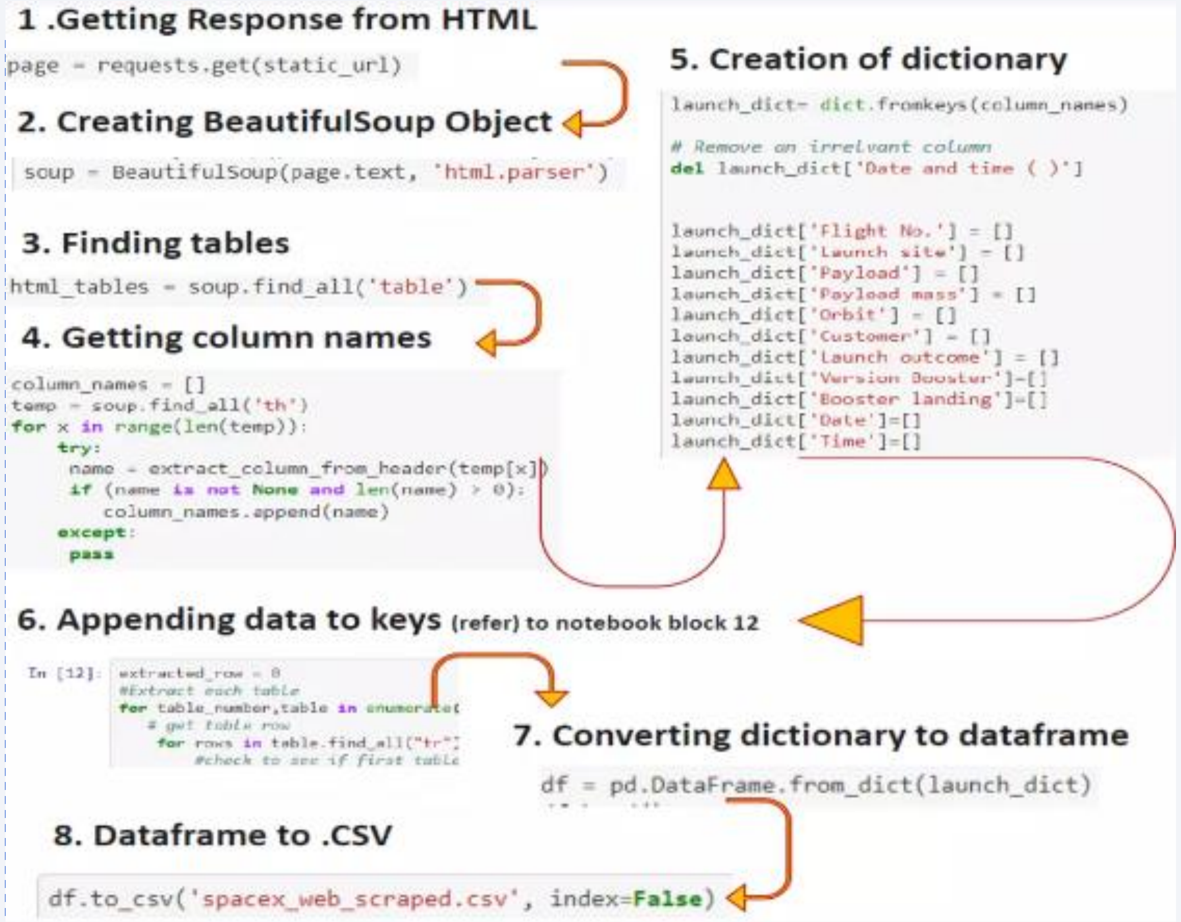
5. Filter dataframe and export to flat file (.csv)

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

- Applied web scraping to extract Falcon 9 launch records with BeautifulSoup
- Parsed the table and converted it into a pandas dataframe.
- IBM-Applied-Data-Science-Capstone/01_SpaceX_Data_Webscraping.ipynb at main · abhijna123/IBM-Applied-Data-Science-Capstone · GitHub



1 .Getting Response from HTML

```
page = requests.get(static_url)
```

2. Creating BeautifulSoup Object

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

3. Finding tables

```
html_tables = soup.find_all('table')
```

4. Getting column names

```
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

5. Creation of dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove on irrelvant column
del launch_dict['Date and time ( )']


launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Appending data to keys (refer) to notebook block 12

```
In [12]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(
             # get table row
              for rows in table.find_all("tr"):
                  #check to see if first table
```

7. Converting dictionary to dataframe

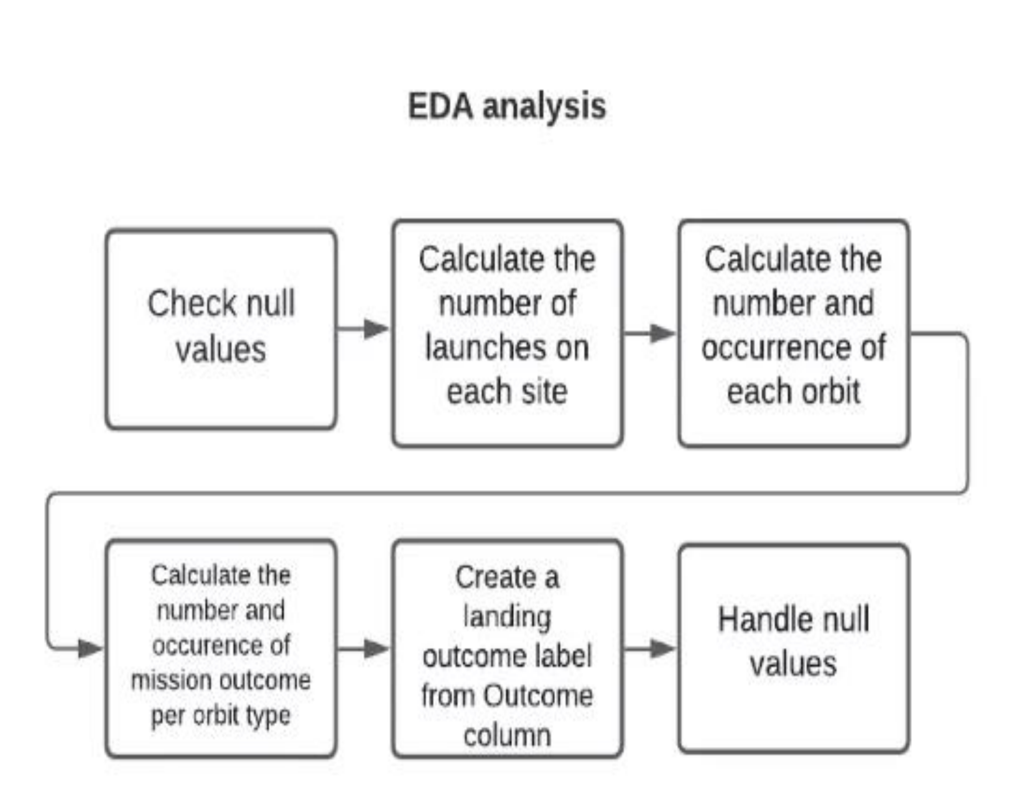
```
df = pd.DataFrame.from_dict(launch_dict)
```

8. Dataframe to .CSV

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

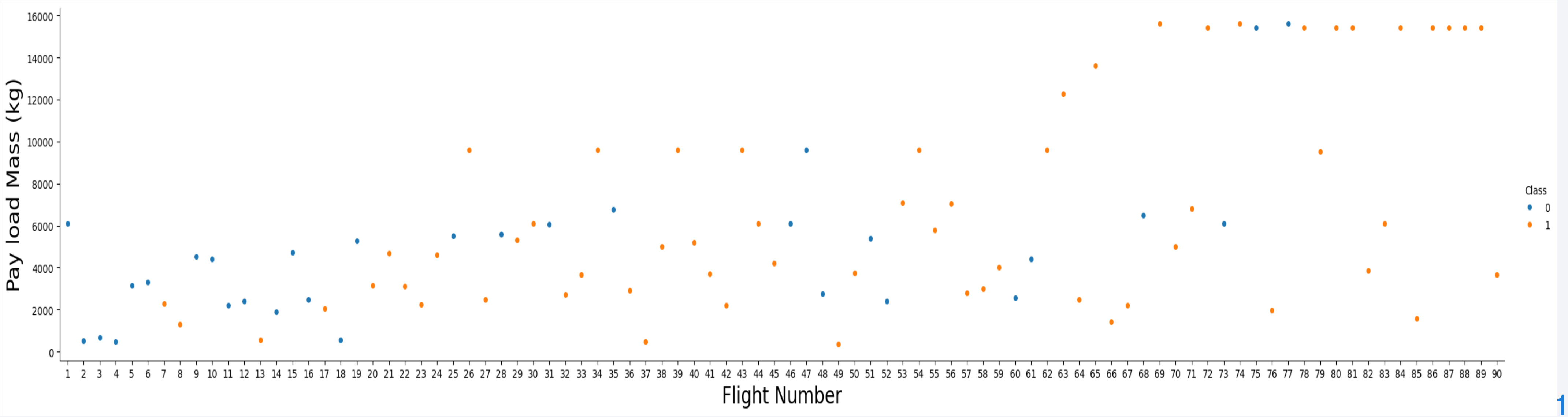
Data Wrangling

- Performed exploratory data analysis (EDA) and determined the training labels.
- Calculated the number of launches at each site, and the number and occurrence of each orbits
- Created landing outcome label from outcome column and exported the results to CSV.
- [IBM-Applied-Data-Science-Capstone/02_SpaceX_Data_Wrangling.ipynb at main · abhijna123/IBM-Applied-Data-Science-Capstone · GitHub]()

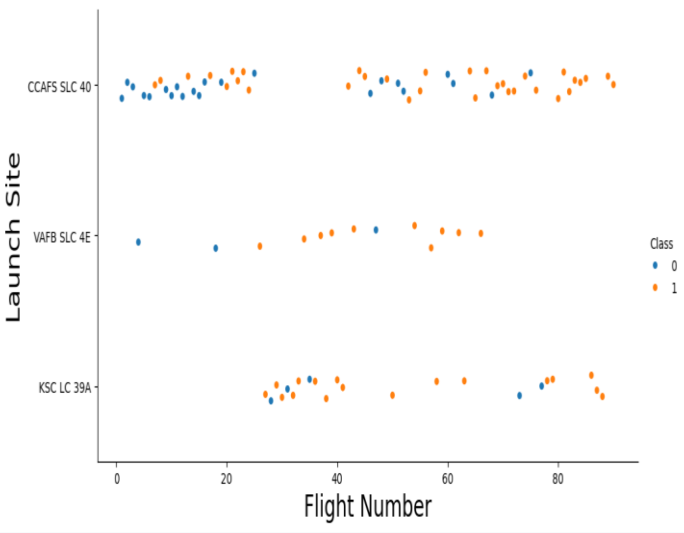

EDA analysis
Check null values
Calculate the number of launches on each site
Calculate the number and occurrence of each orbit
Calculate the number and occurence of mission outcome per orbit type
Create a landing outcome label from Outcome column
Handle null values

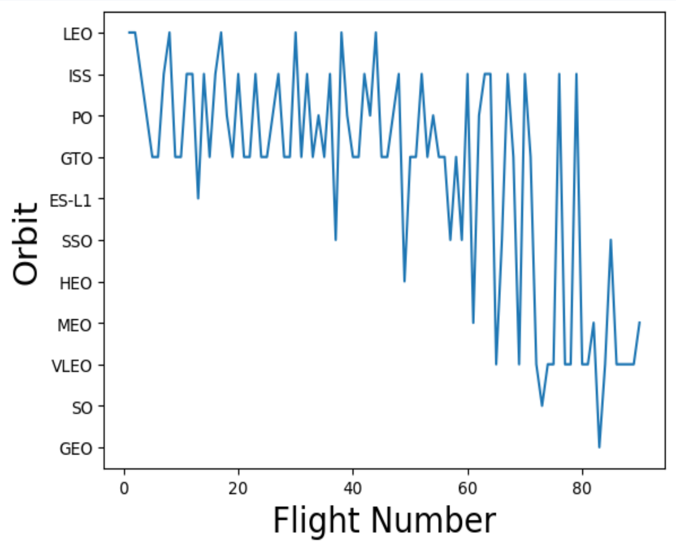
EDA with Data Visualization

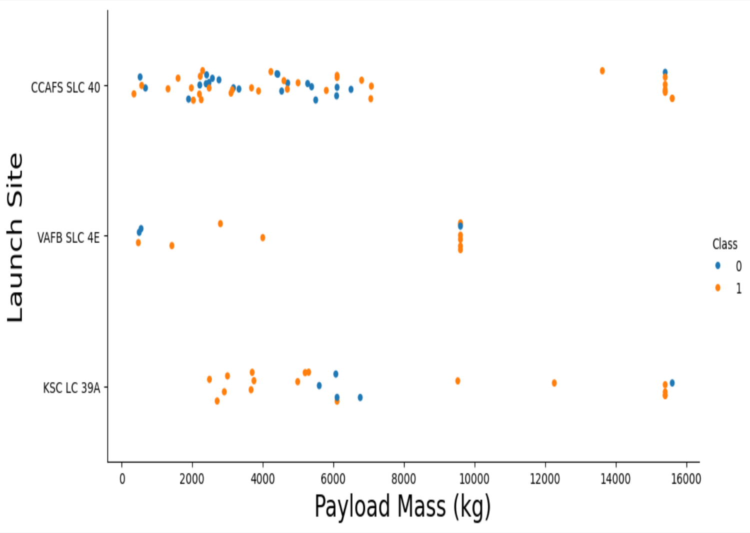
- Explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend
- IBM-Applied-Data-Science-Capstone/04_SpaceX_EDA_Visualization.ipynb at main · abhijna123/IBM-Applied-Data-Science-Capstone · GitHub

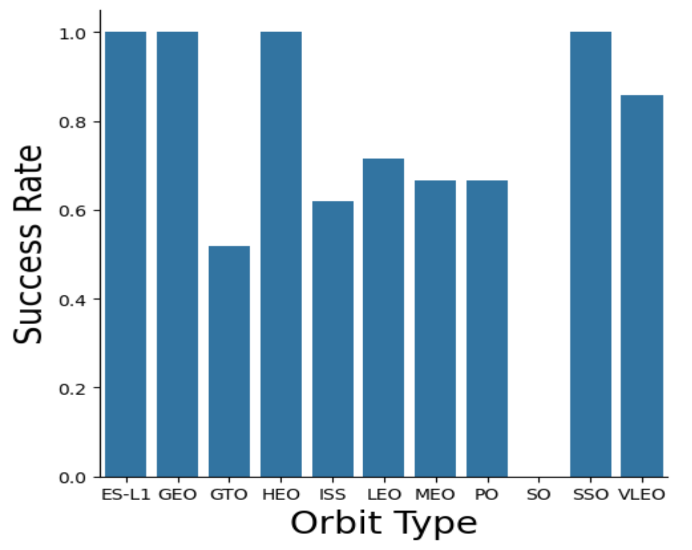


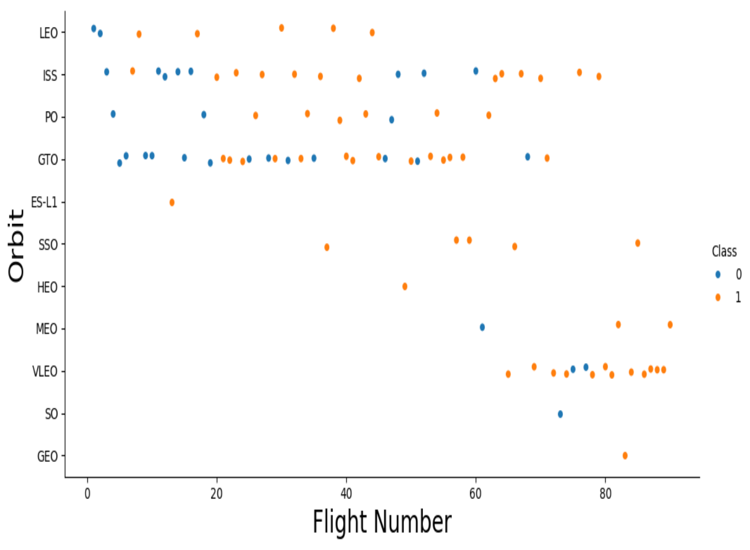
EDA with Data Visualization



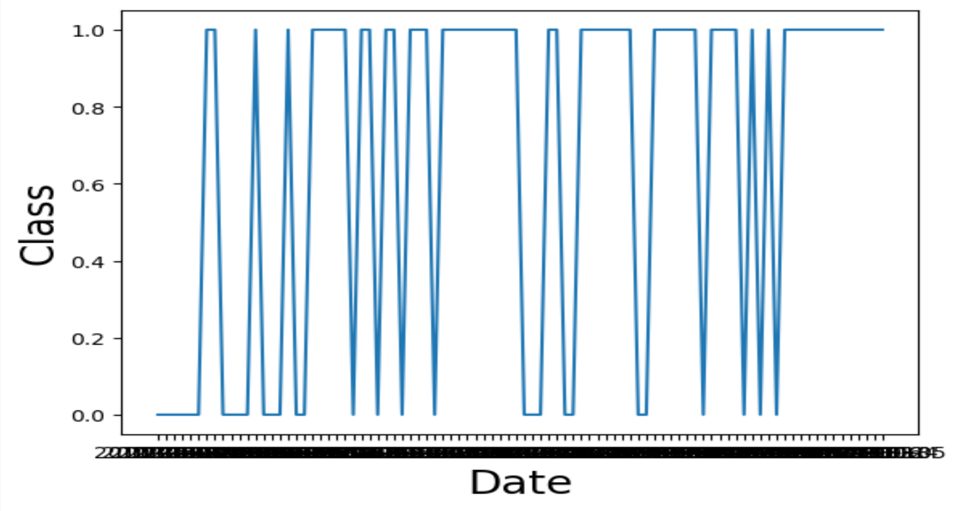


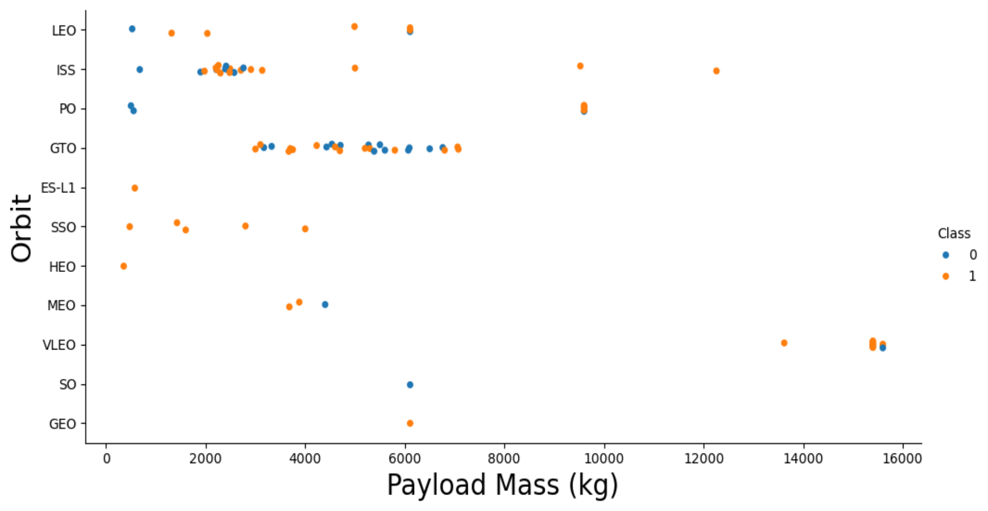


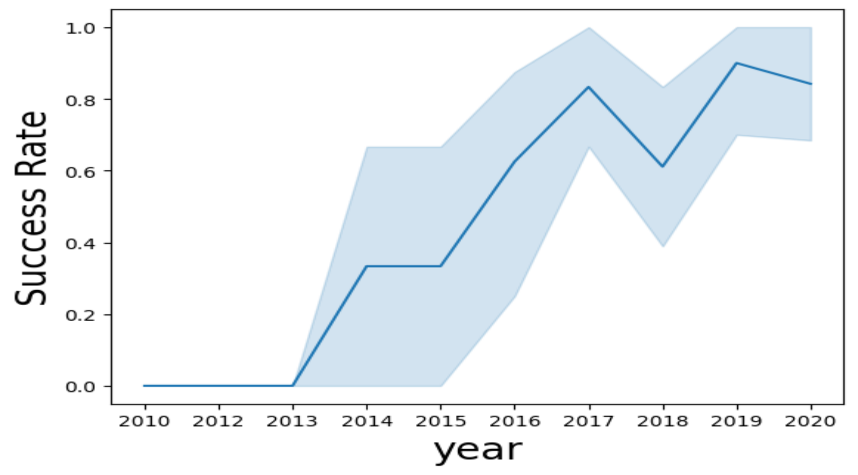




EDA with Data Visualization







EDA with SQL

- Applied EDA with SQL to get insight from the data. Wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - 5 records where launch sites begin with the string 'CCA'
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The date when the first successful landing outcome in ground pad was achieved
  - The names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - The total number of successful and failure mission outcomes
  - The names of the booster_versions which have carried the maximum payload mass.
  - The failed landing outcomes in drone ship, their booster version and launch site names.
  - Rank the count of landing outcomes between date 2010-06-04 and 2017-03-20 in descending order.
- IBM-Applied-Data-Science-Capstone/03_SpaceX_EDA_SQL.ipynb at main · abhijna123/IBM-Applied-Data-Science-Capstone · GitHub

Build an Interactive Map with Folium

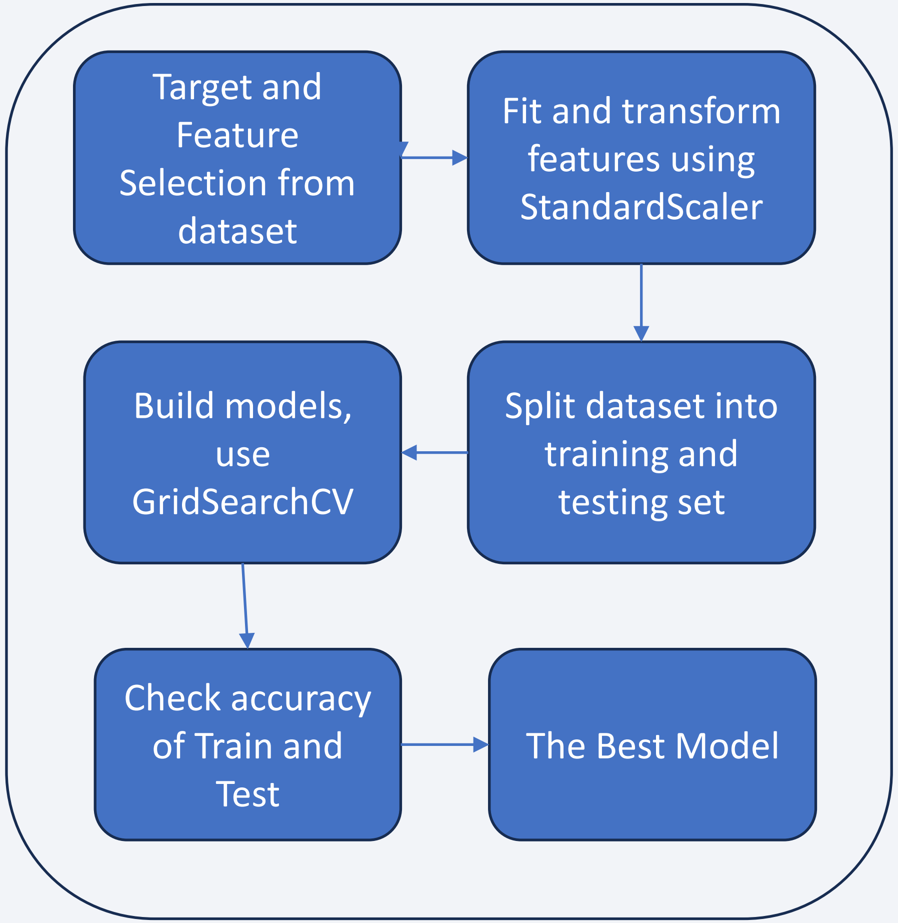
- Marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- Assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, identified which launch sites have relatively high success rate.
- 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.
- IBM-Applied-Data-Science-Capstone/05_SpaceX_Interactive_Visual_Analytics_Folium.ipynb at main · abhijna123/IBM-Applied-Data-Science-Capstone · GitHub

Build a Dashboard with Plotly Dash

- Built an interactive dashboard with Plotly dash
- Plotted pie charts showing the total launches by a certain sites
- Plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- IBM-Applied-Data-Science-Capstone/06_SpaceX_Interactive_Visual_Analytics_Plotly.py at main · abhijna123/IBM-Applied-Data-Science-Capstone · GitHub

Predictive Analysis (Classification)

- Loaded the data using NumPy and pandas, transformed the data, split our data into training and testing.
- Built different machine learning models and tune different hyperparameters using GridSearchCV.
- Used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- Found the best performing classification model.
- IBM-Applied-Data-Science-Capstone/07_SpaceX_Predictive_Analytics.ipynb at main · abhijna123/IBM-Applied-Data-Science-Capstone · GitHub

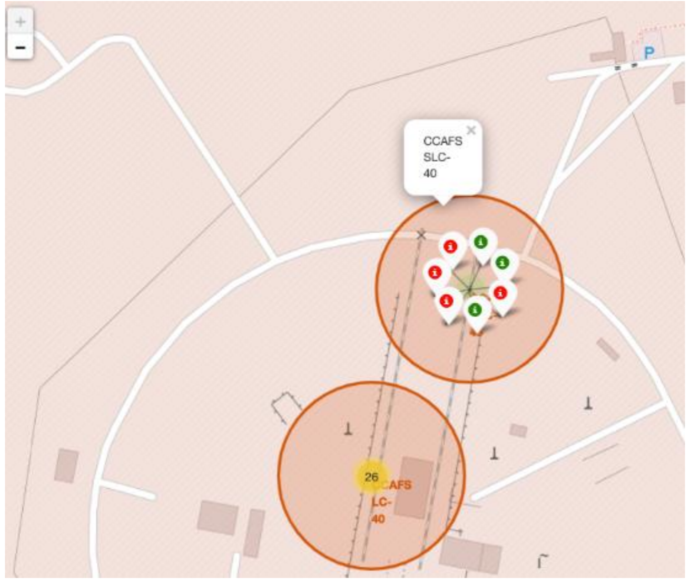

Target and Feature Selection from dataset
Fit and transform features using StandardScaler
Split dataset into training and testing set
Build models, use GridSearchCV
Check accuracy of Train and Test
The Best Model

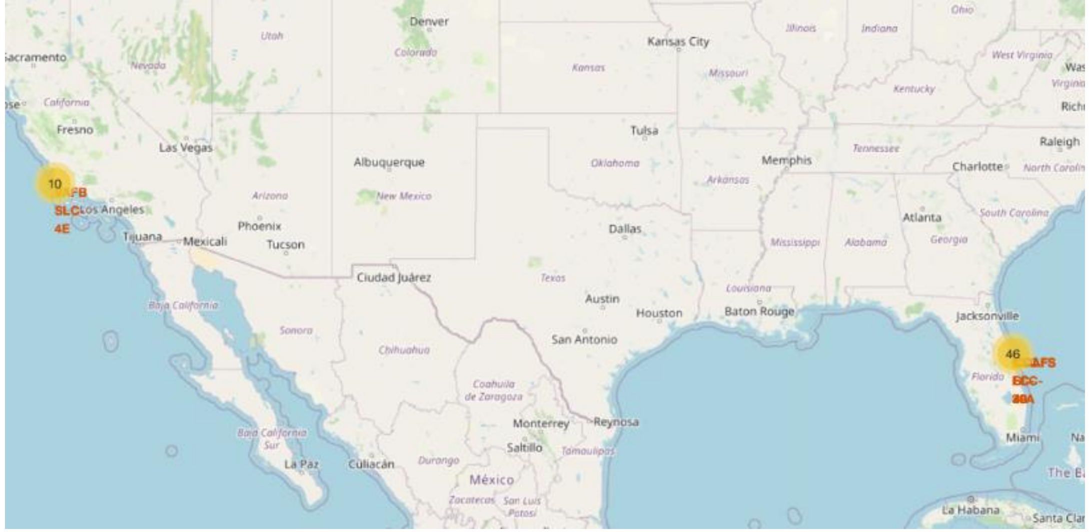
Results

- Exploratory data analysis results
  - SpaceX uses 4 different launch sites
  - The first launches were done to Space X itself and NASA;
  - The average payload of F9 v1.1 booster is 2,928 kg;
  - The first success landing outcome happened in 2015 five year after the first launch;
  - Many Falcon 9 booster versions were successful at landing in drone ships having payload above the average;
  - Almost 100% of mission outcomes were successful;
  - Two booster versions failed at landing in drone ships in 2015: F9 v1.1 B1012 and F9 v1.1 B1015;
  - The number of landing outcomes became as better as years passed.

Results

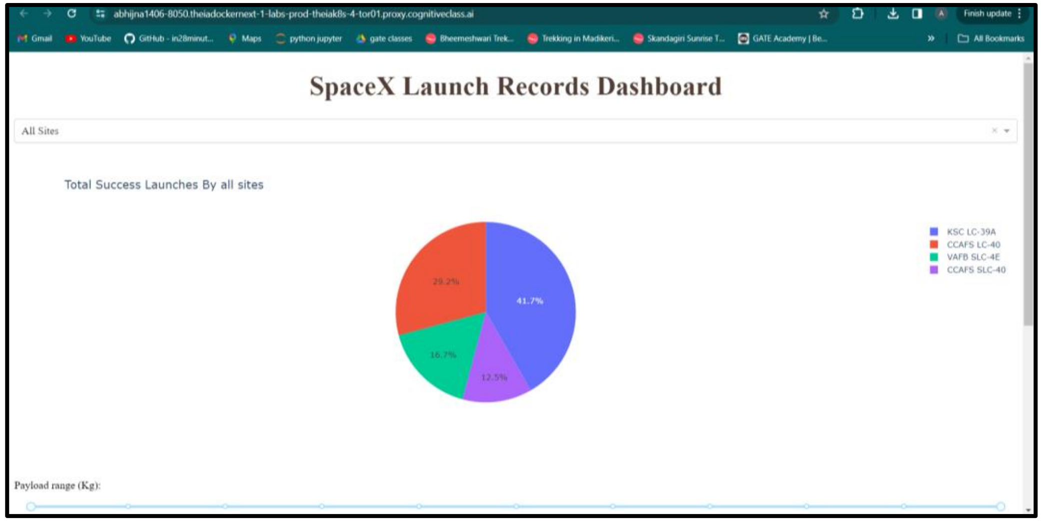
- Interactive analytics demo in screenshots
  - Interactive analytics allowed for the identification of the fact that launch sites historically had solid logistical infrastructure and were located in secure areas, such as close to the sea.
  - Most launches take place at East Coast launch locations
  - The following slides will show Interactive Map with Folium.





Results

- This is a preview of the Plotly dashboard. The following sides will show the results through screenshots.



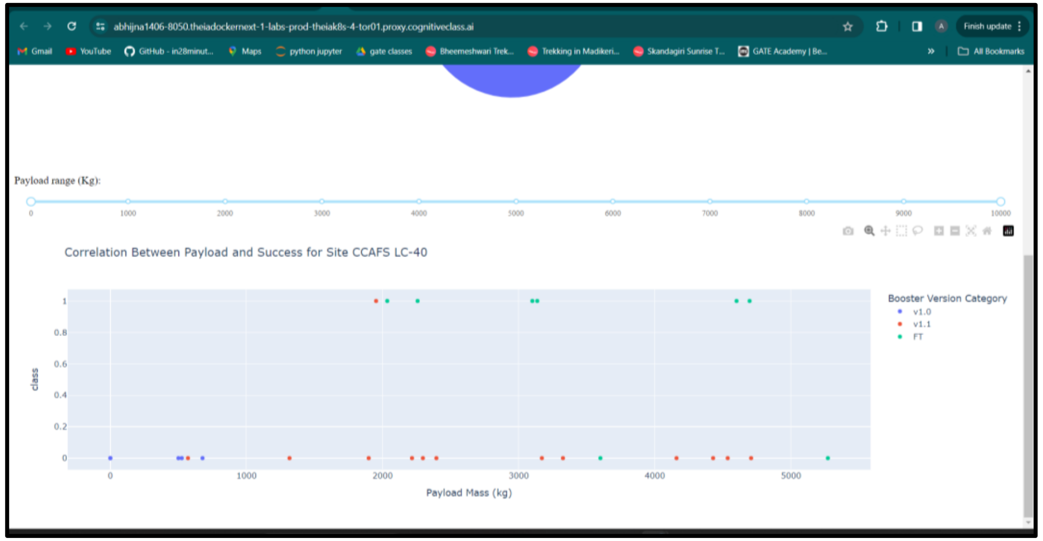
abhijna1406-8050.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai

SpaceX Launch Records Dashboard

All Sites

Total Success Launches By all sites

Payload range (Kg):



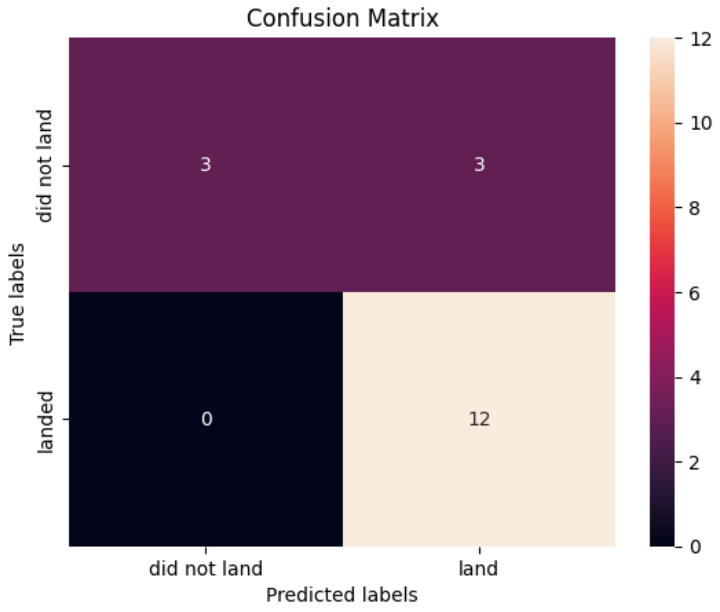
abhijna1406-8050.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai

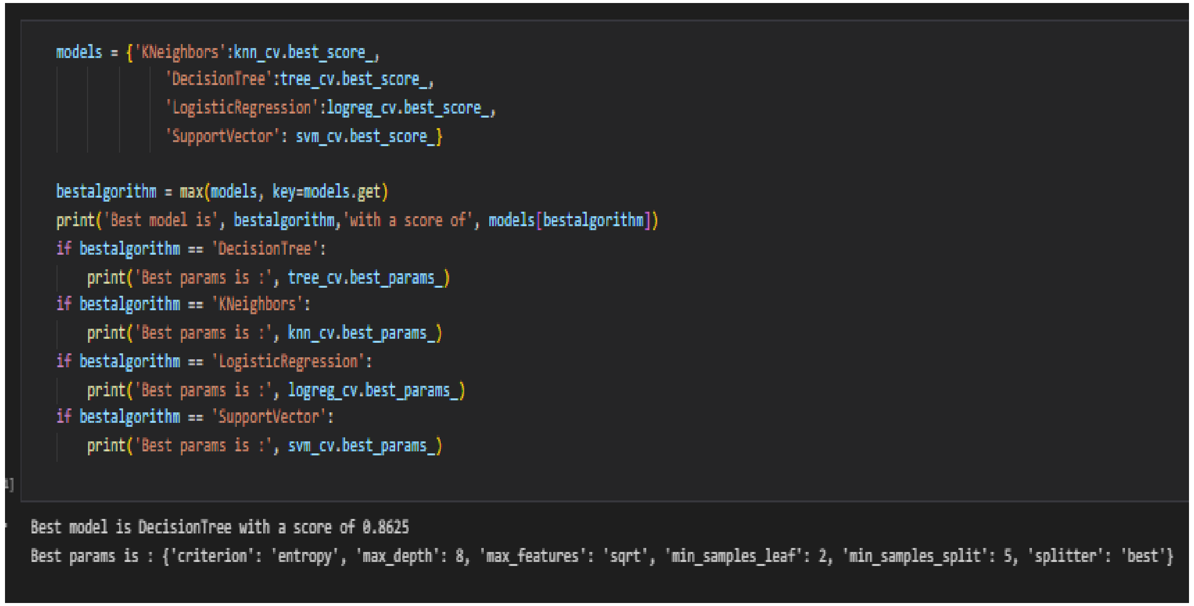
Payload range (Kg):

Correlation Between Payload and Success for Site CCAFS LC-40

Results

- Predictive analysis results:
  - This is a preview of the few models and their plots
  - The decision tree classifier is the model with the highest classification accuracy





```
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_)
```

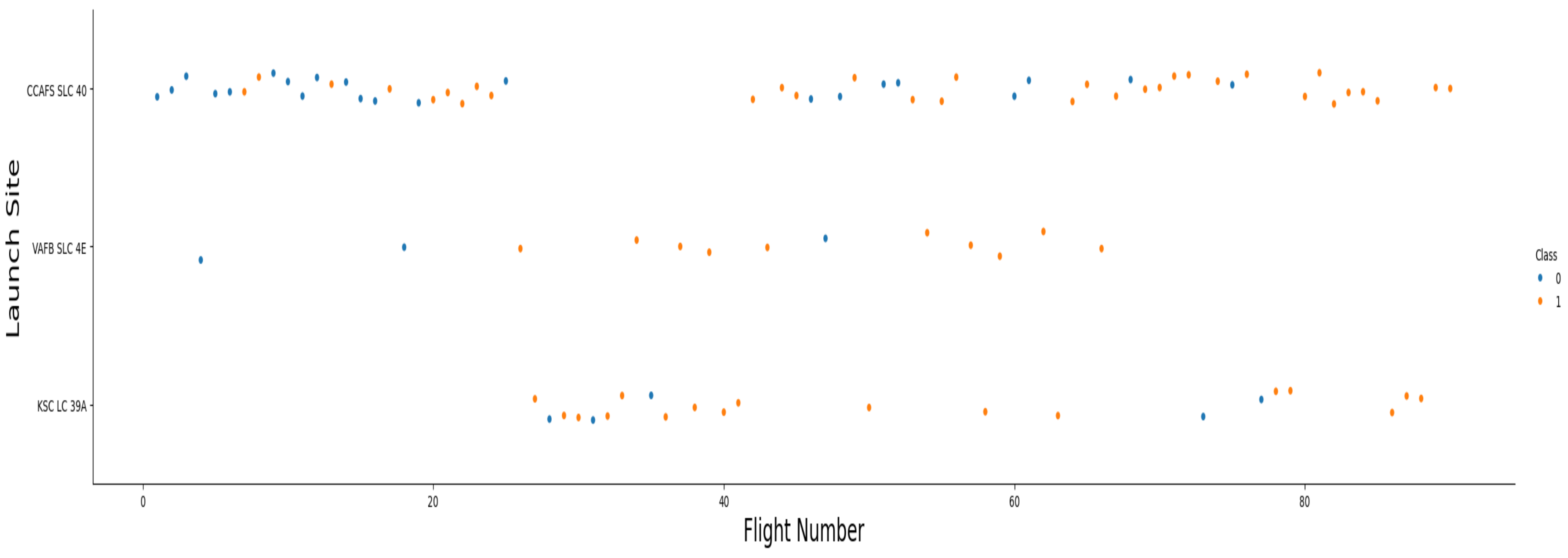
```
Best model is DecisionTree with a score of 0.8625
Best params is : {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'best'}
```

Section 2

Insights drawn from EDA

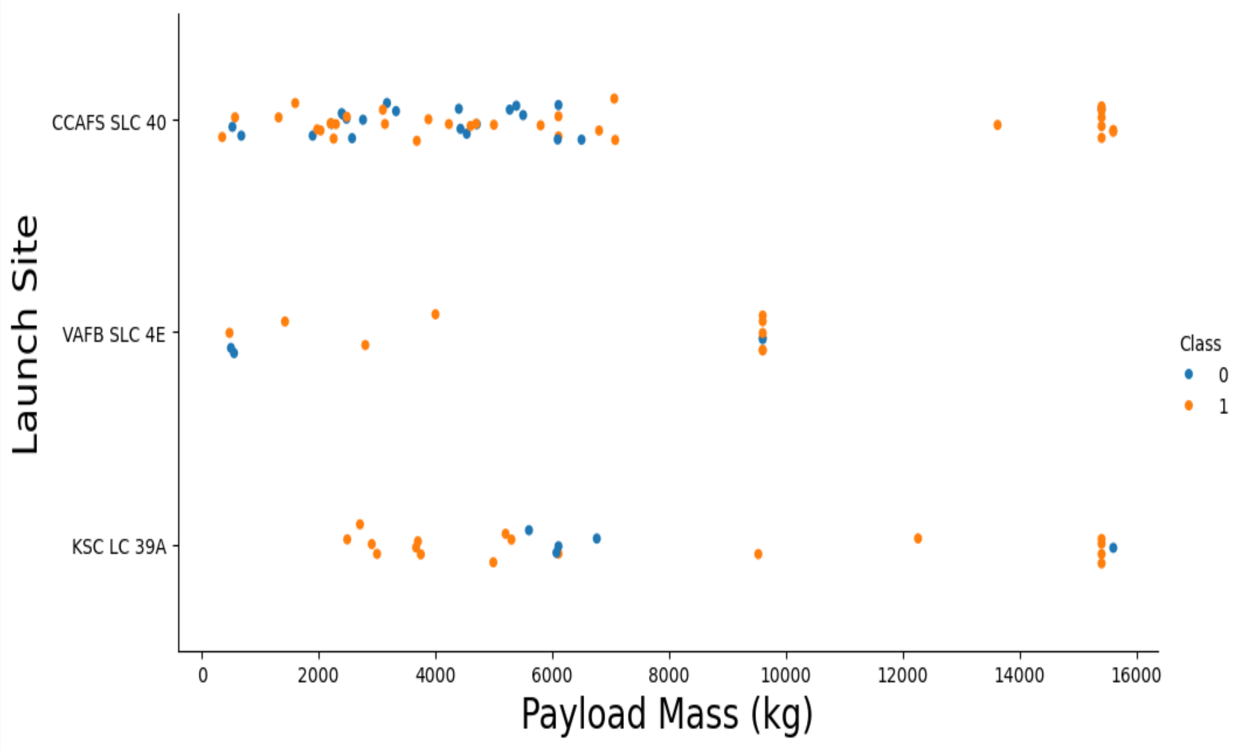
Flight Number vs. Launch Site

- The best launch site (nowadays) is CCAFS SLC 40 where most of the successful launches have taken place recently.
- Graphic suggests an increase in success rate over time (indicated in Flight Number).
- Most likely, there was a huge development around flight 20 that greatly improved the success rate.



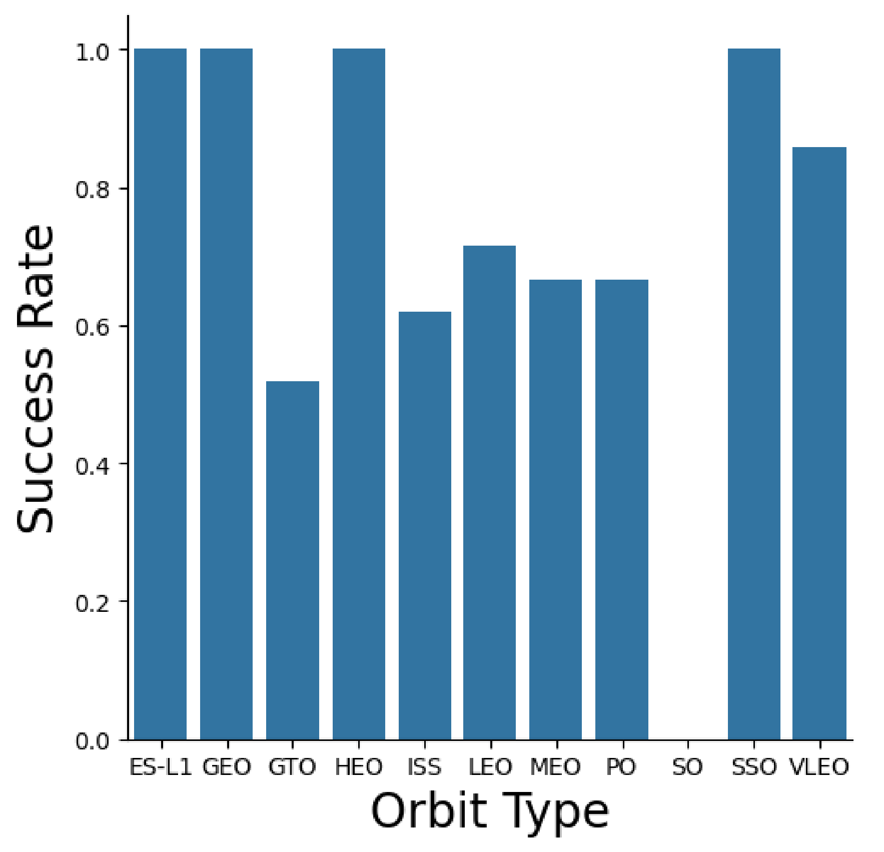
Payload vs. Launch Site

- Payload mass appears to fall mostly between 0-7000 kg.
- Payloads over 12,000kgs seems to be possible for only CCFAS SLC 40 and KSC LC 39A.



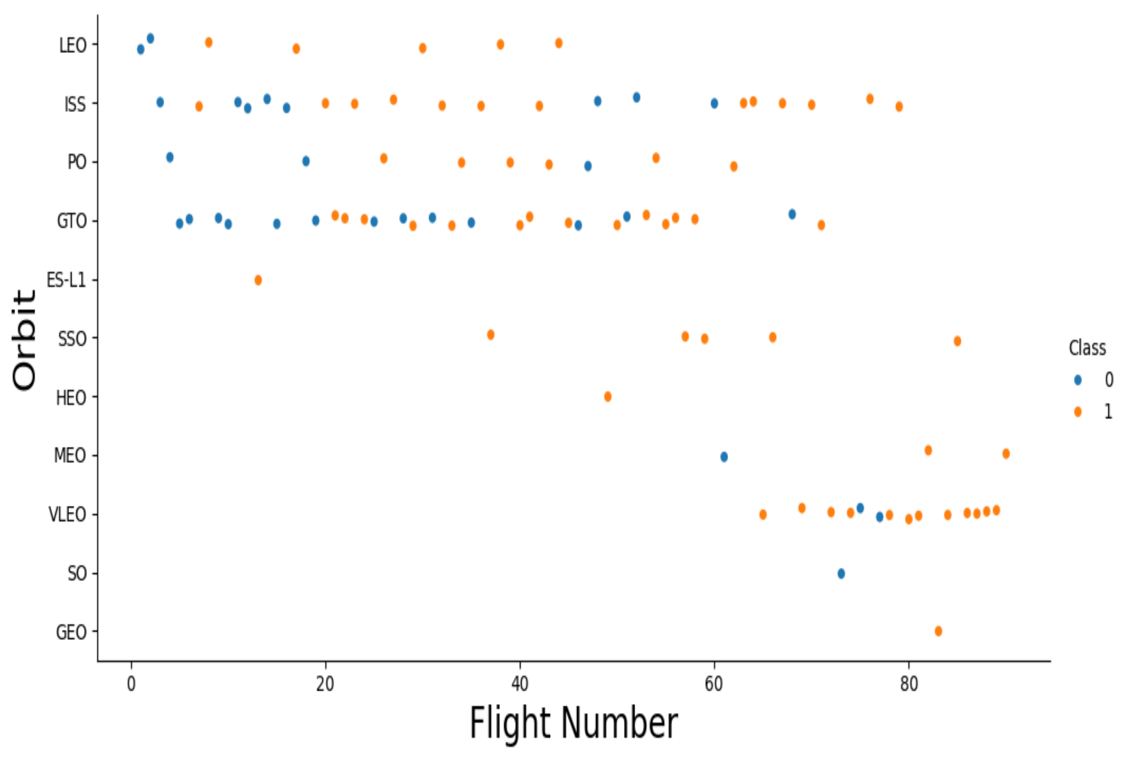
Success Rate vs. Orbit Type

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



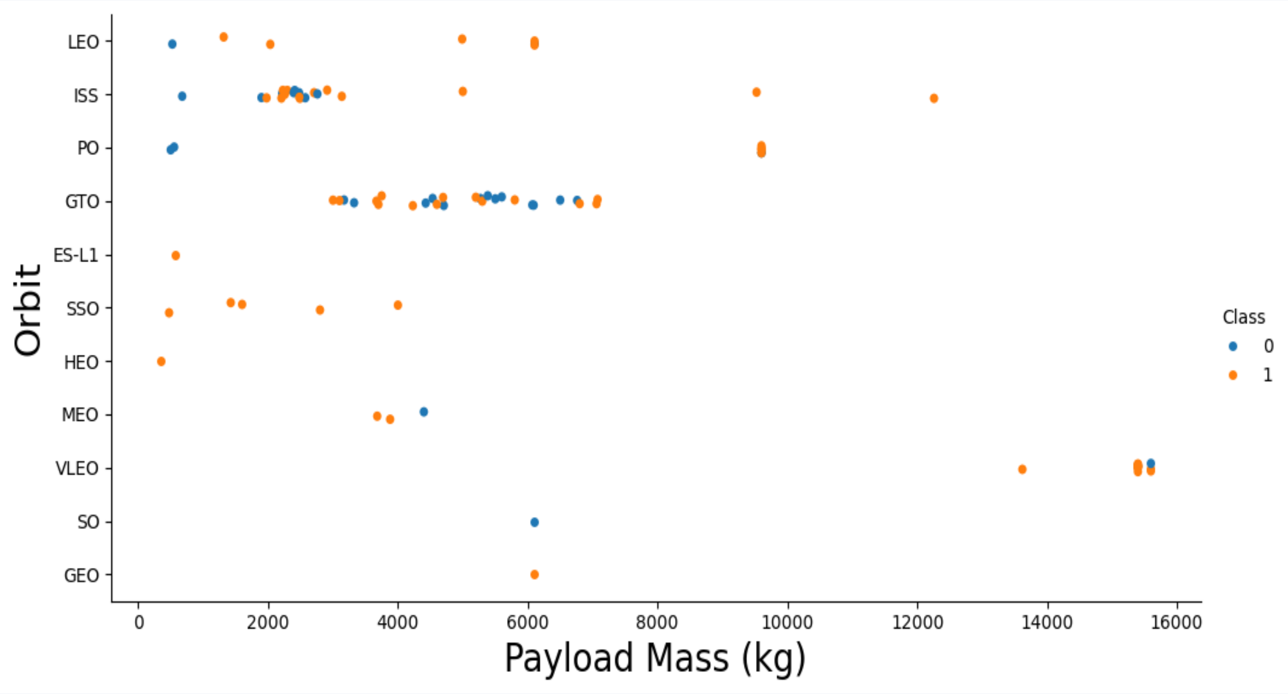
Flight Number vs. Orbit Type

- 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.



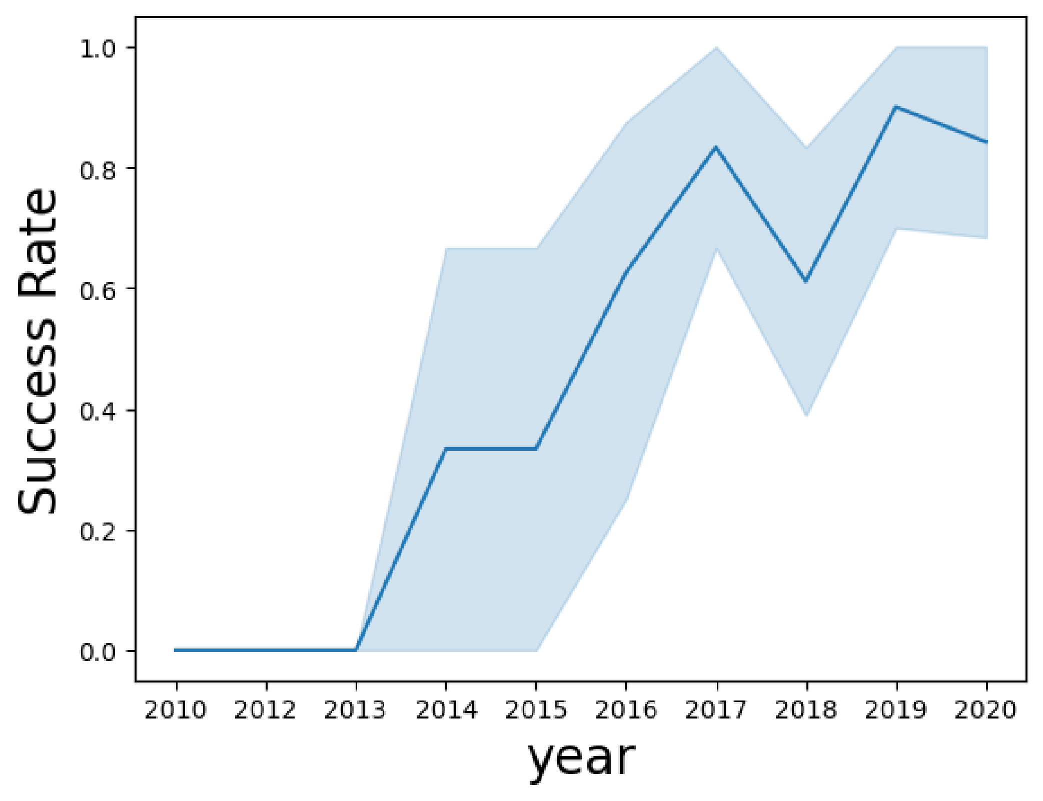
Payload vs. Orbit Type

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



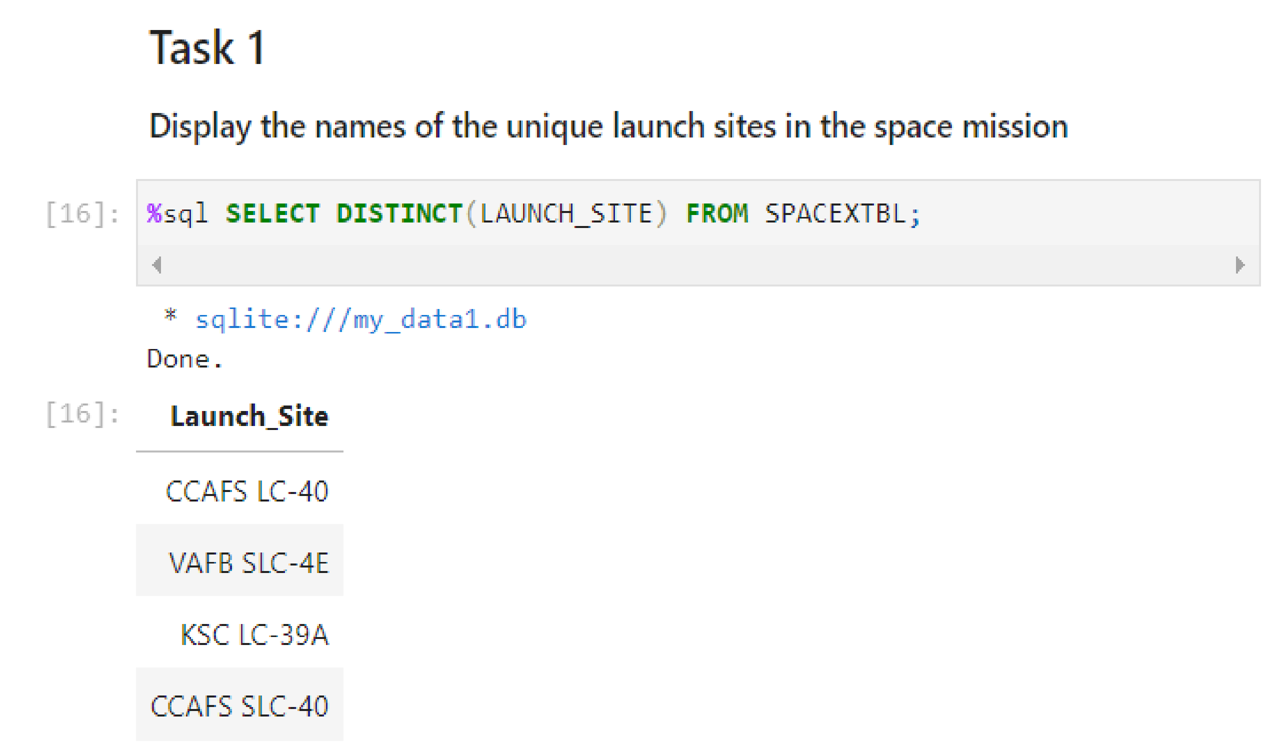
Launch Success Yearly Trend

- 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.



Task 1

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

```
[16]: %sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL;
```

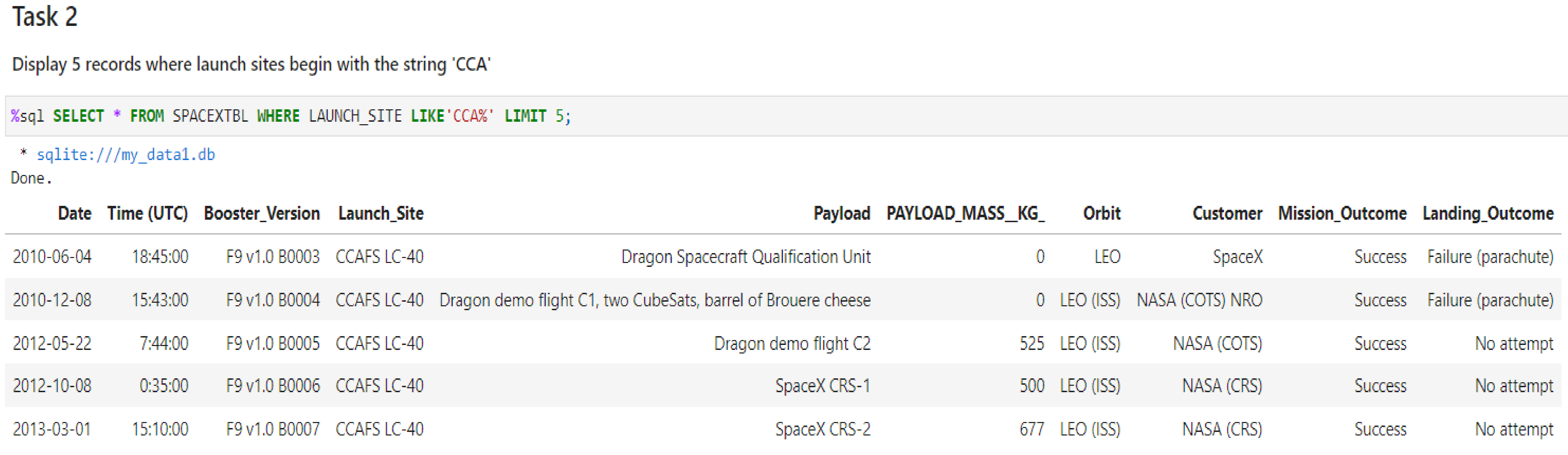
```
 * sqlite:///my_data1.db
Done.
```

[16]:

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

Launch Site Names Begin with 'CCA'

- The following query was used to find 5 records where launch sites begin with `CCA`, using the constraint LIMIT with the WHERE command:



Task 2

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

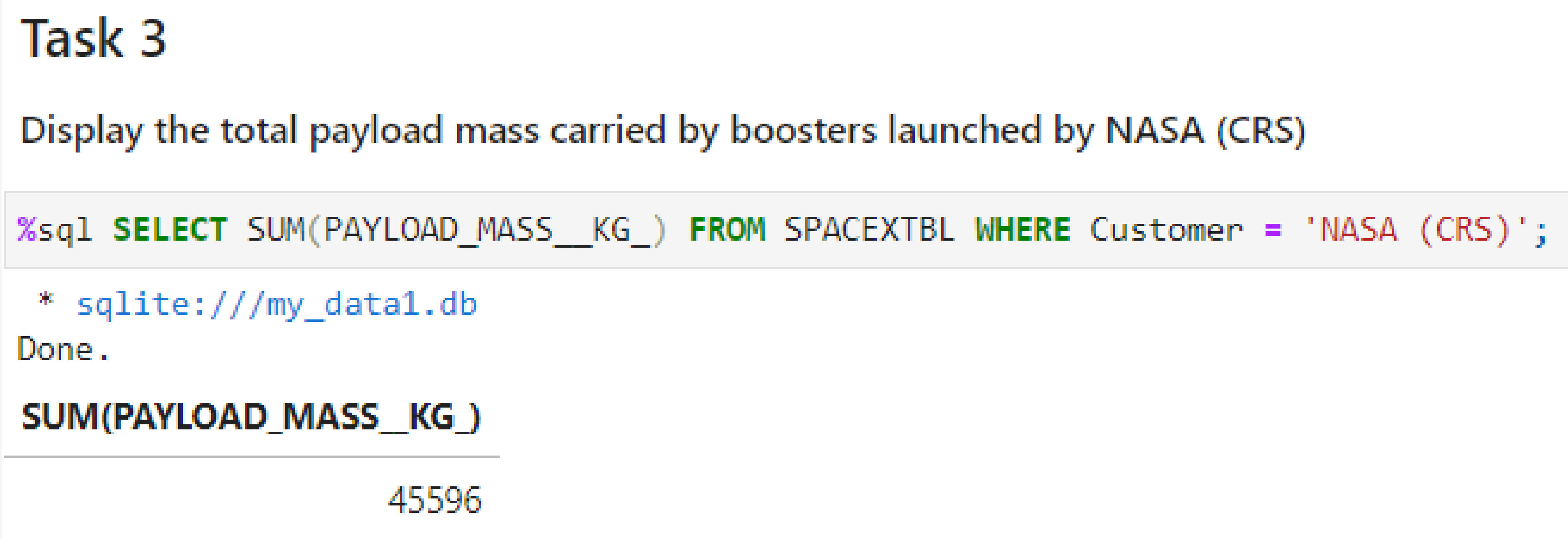
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE'CCA%' LIMIT 5;
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_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

To calculate the total payload mass carried by boosters launched by NASA, it was necessary to use the SUM function and GROUP BY operator:



Task 3

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

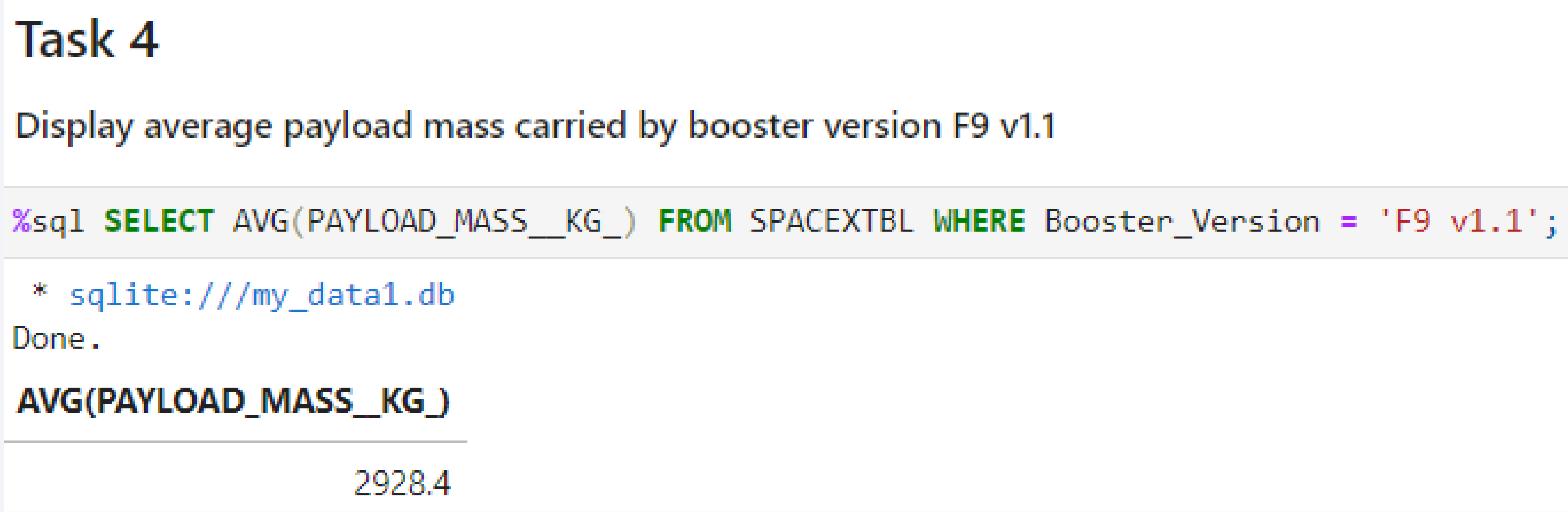
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Customer = 'NASA (CRS)';
```

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- The average payload mass caried by booster version F9 v1.1 was calculated using the AVG function combined with WHERE clause



Task 4

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

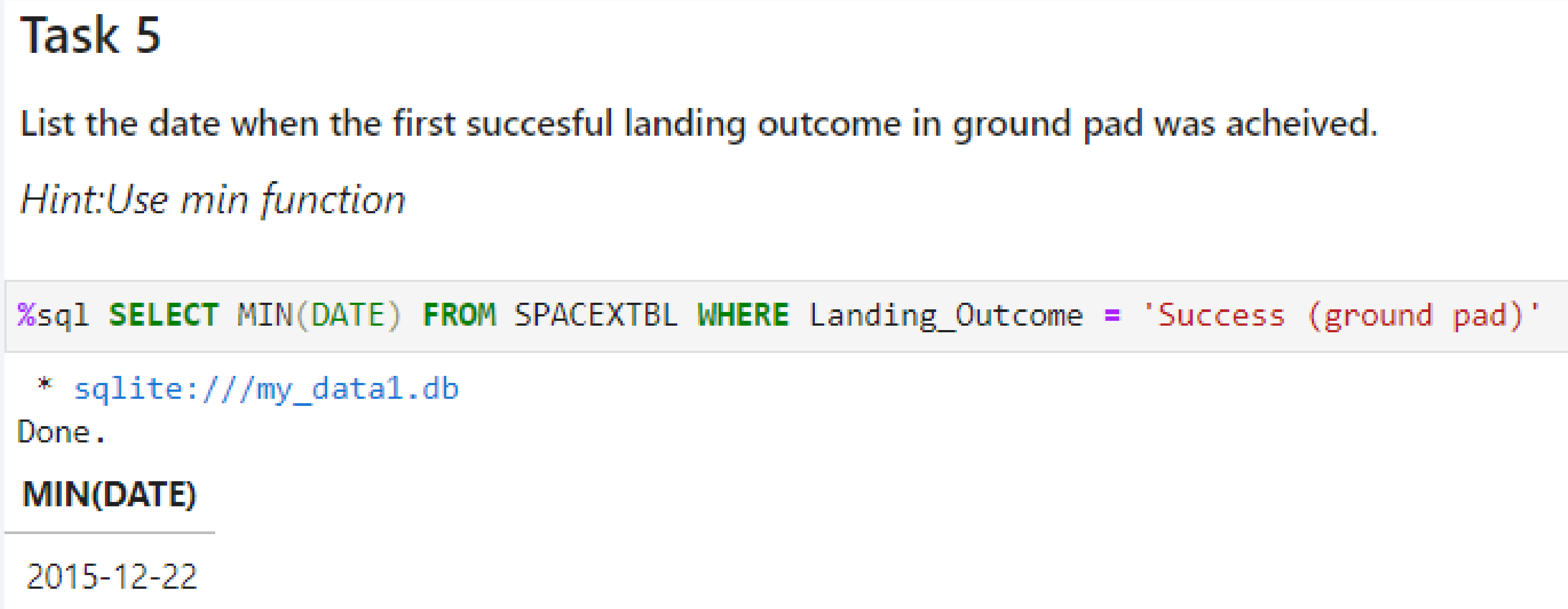
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1';
```

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

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- The first successful landing outcome in ground was achieved in 2015-12-22. To find this date, the MIN function was used:



Task 5

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

Hint:Use min function

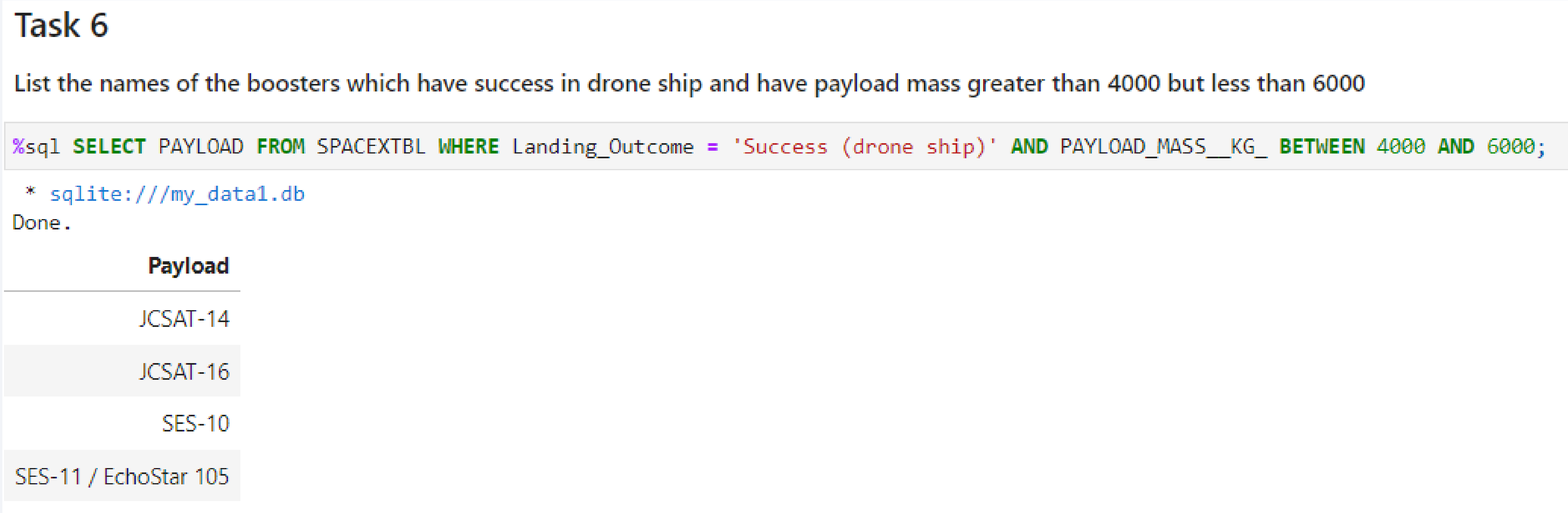
```
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)'
```

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

MIN(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- To list the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 the AND operator was used to combine two restrictions in the WHERE clause:



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

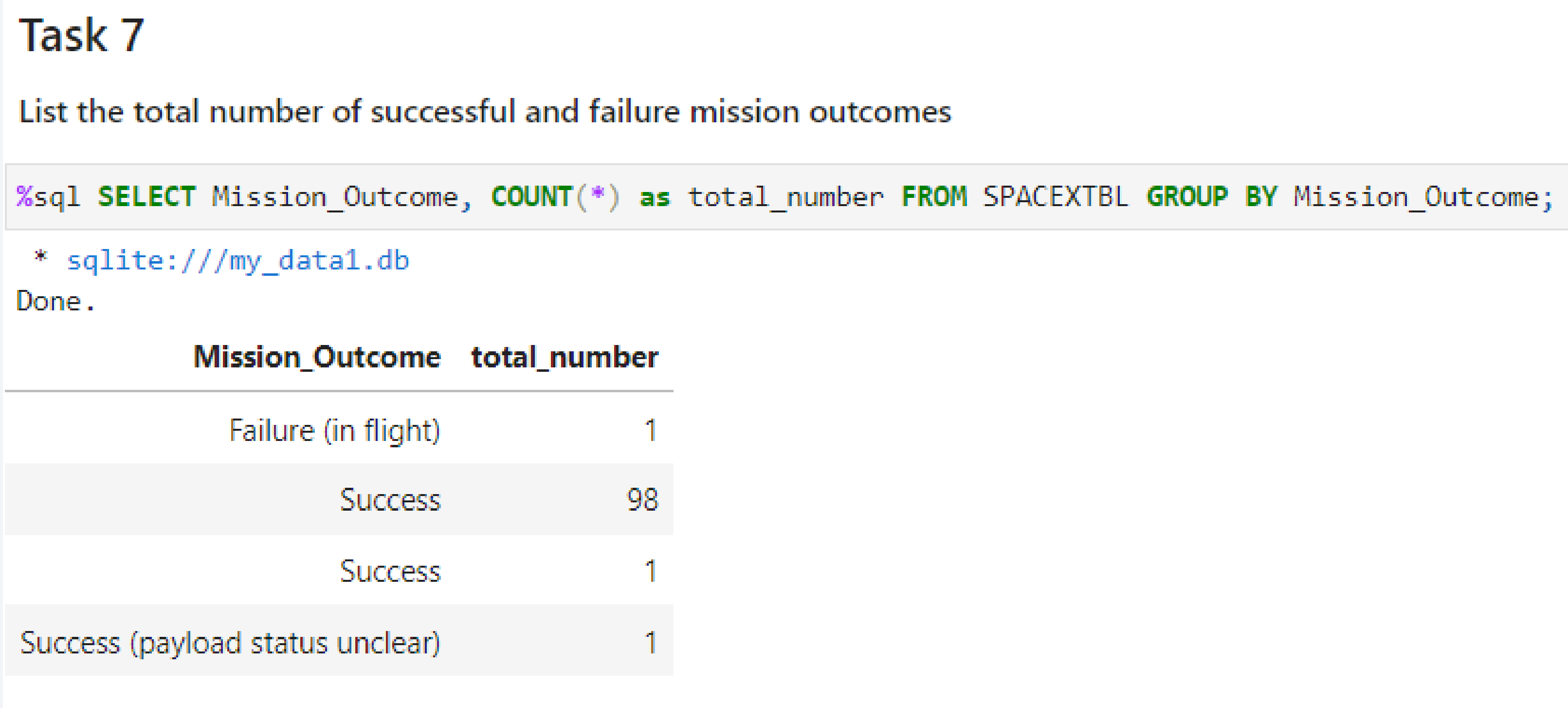
```
%sql SELECT PAYLOAD FROM SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

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

Payload
JCSAT-14
JCSAT-16
SES-10
SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

- To calculate the total number of successful and failure mission outcomes, it was necessary to use the COUNT function with the GROUP BY operator:



Task 7

List the total number of successful and failure mission outcomes

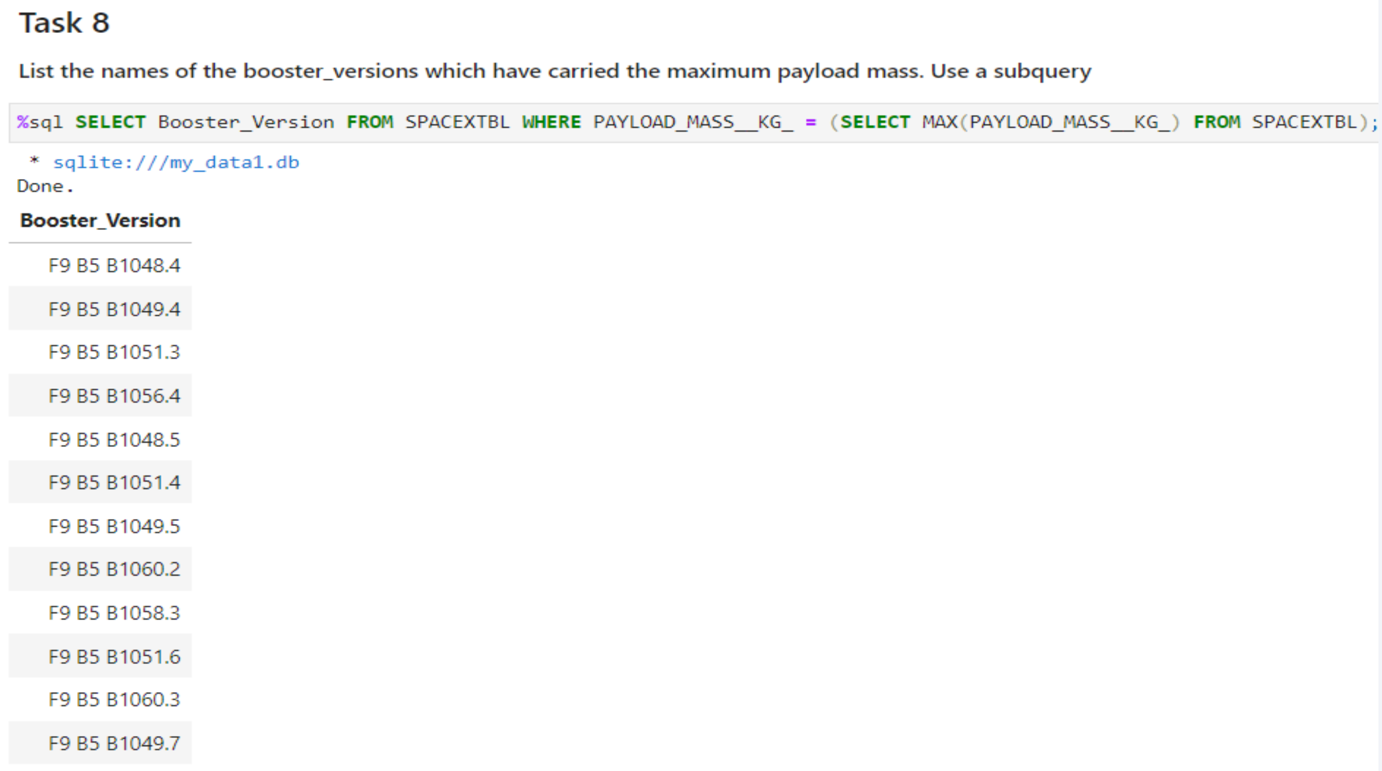
```
%sql SELECT Mission_Outcome, COUNT(*) as total_number FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

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

Boosters Carried Maximum Payload

- To list the names of the booster which have carried the maximum payload mass, the solution consists of using a subquery in the WHERE clause;



Task 8

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

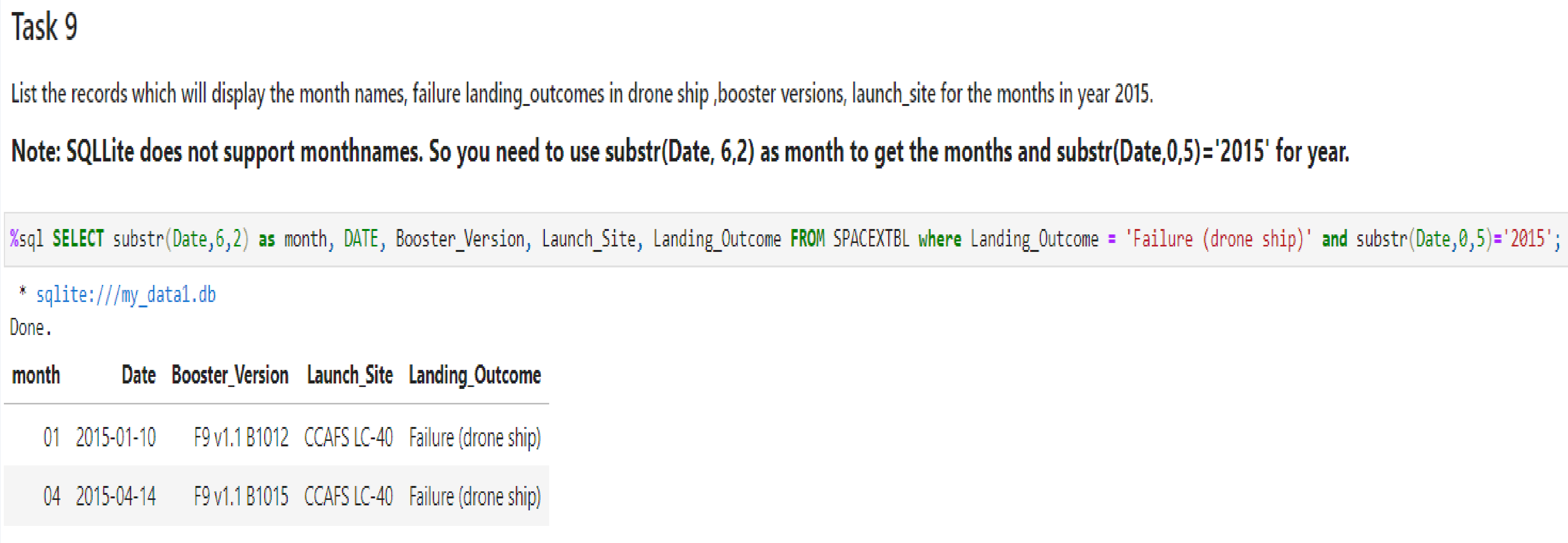
```
%sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

```
 * 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

- To list the failed landing outcomes in drone ship, their booster versions, and launch site names in 2015 the YEAR function was combined with the AND operator in the WHERE clause:



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.

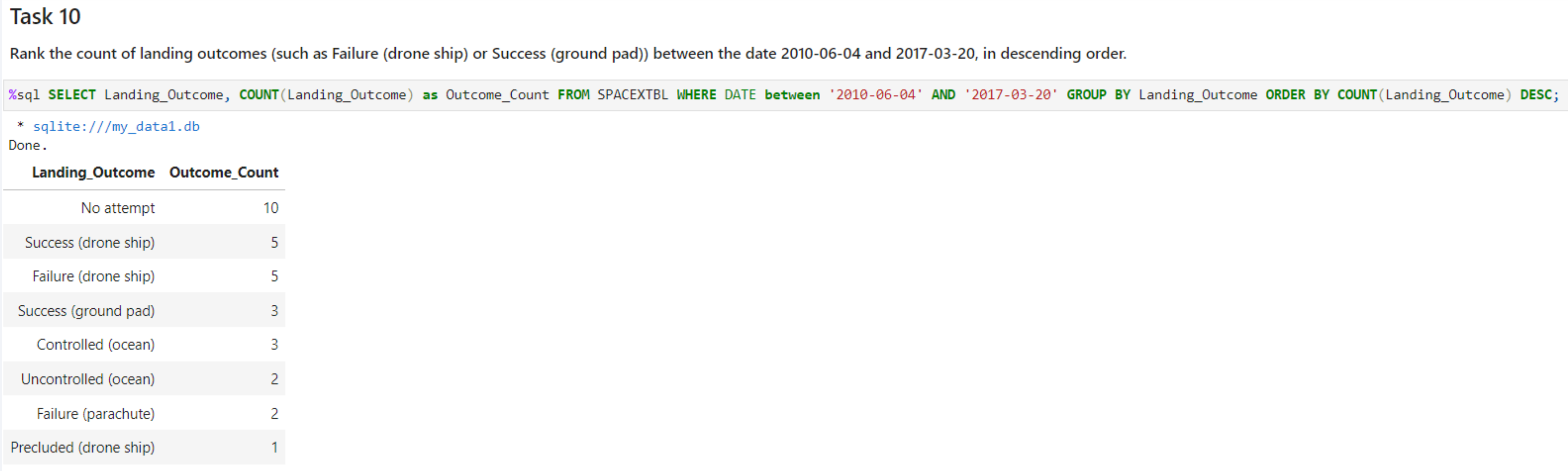
```
%sql SELECT substr(Date,6,2) as month, DATE, Booster_Version, Launch_Site, Landing_Outcome FROM SPACEXTBL where Landing_Outcome = 'Failure (drone ship)' and substr(Date,0,5)='2015';
```

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

month	Date	Booster_Version	Launch_Site	Landing_Outcome
01	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- To get a rank of the count of landing outcomes between the date 2010-06- 04 and 2017-03-20, in descending order, it was used the COUNT function with the operators GROUP BY and ORDER BY:



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, COUNT(Landing_Outcome) as Outcome_Count FROM SPACEXTBL WHERE DATE between '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY COUNT(Landing_Outcome) DESC;
```

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

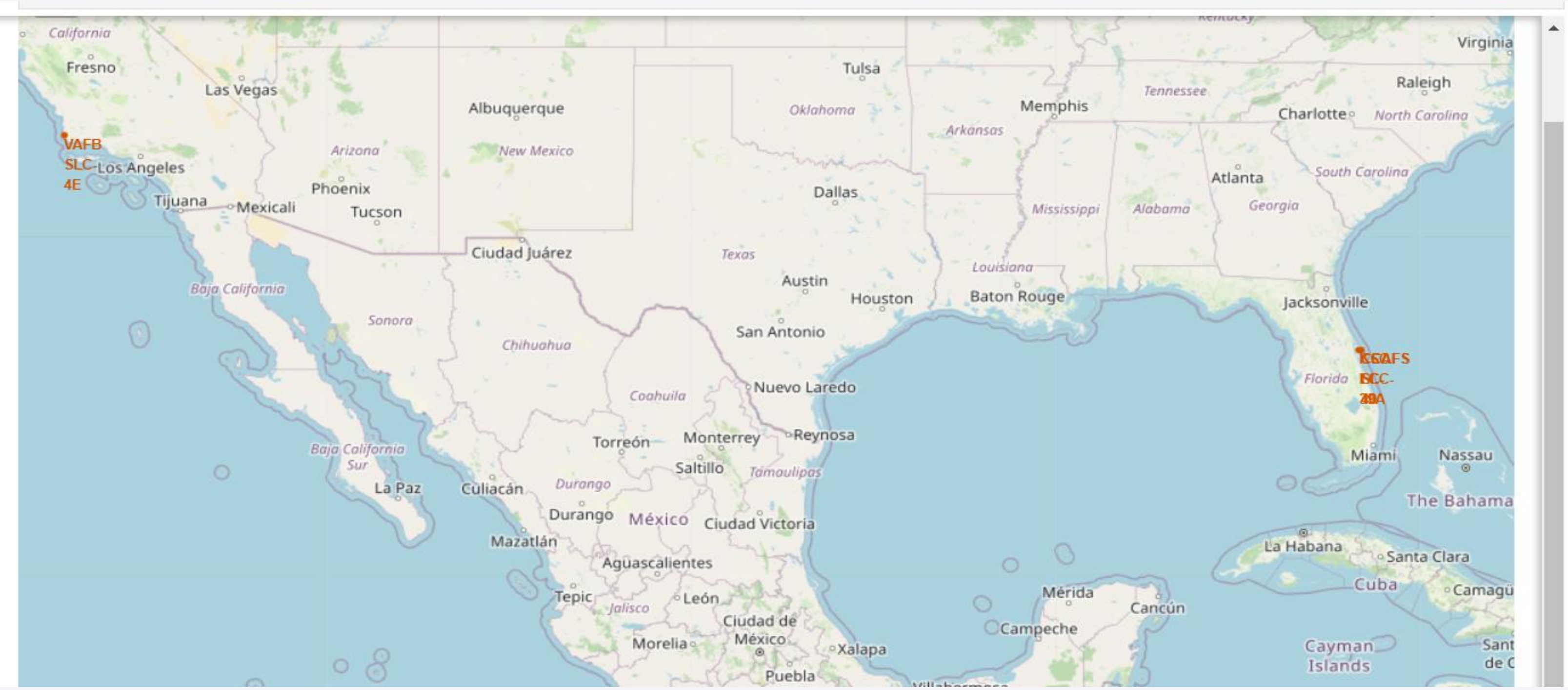
Landing_Outcome	Outcome_Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

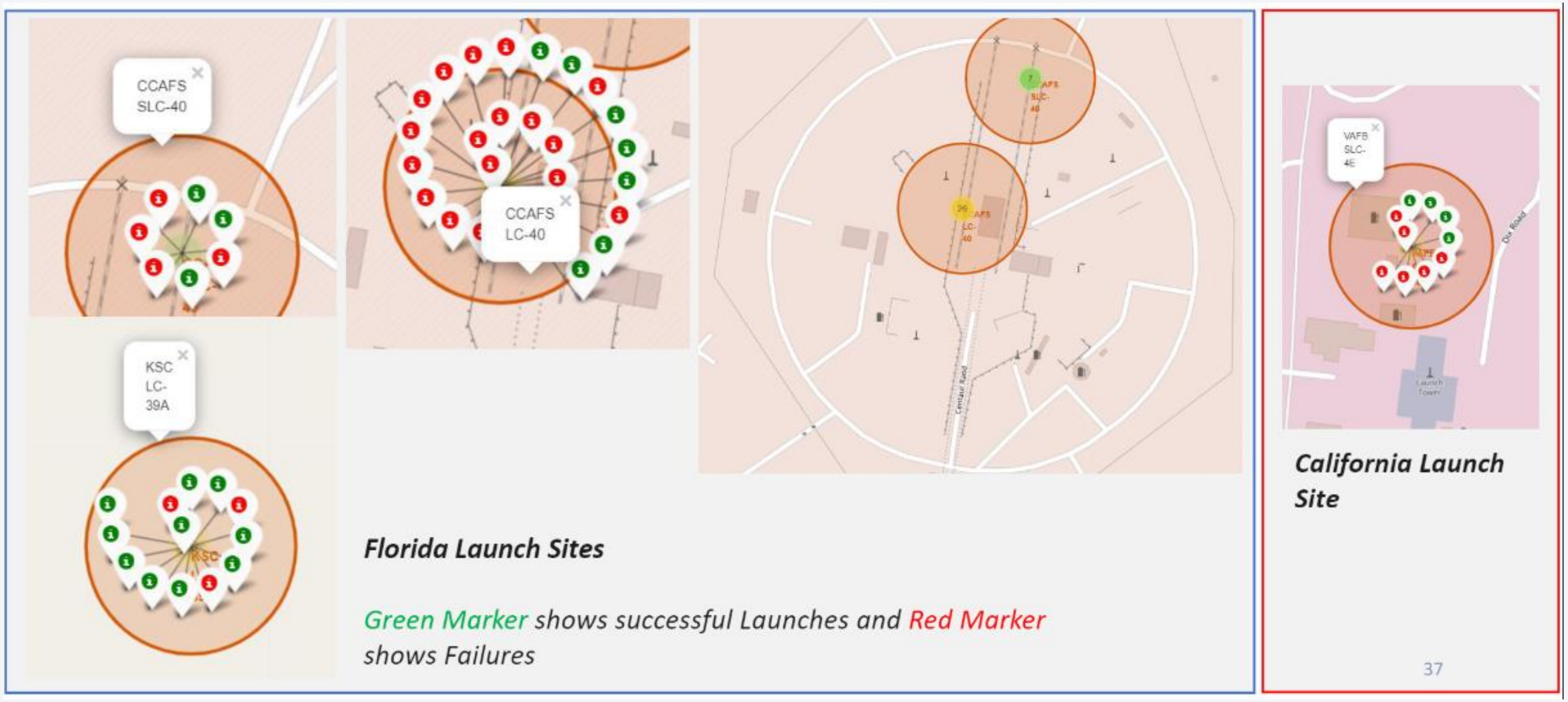
Folium map with all launch sites

- It is possible to see in the map that all launch sites are located in the United States coasts, one in California and two in Florida;
- The proximity with the cost can be explained by the fact that launches toward the sea minimizes the risk of damage from falling debris;



Map with outcome labeled markers

- To each launch it was placed a marker, with color to distinguish failure (red) from success (green)



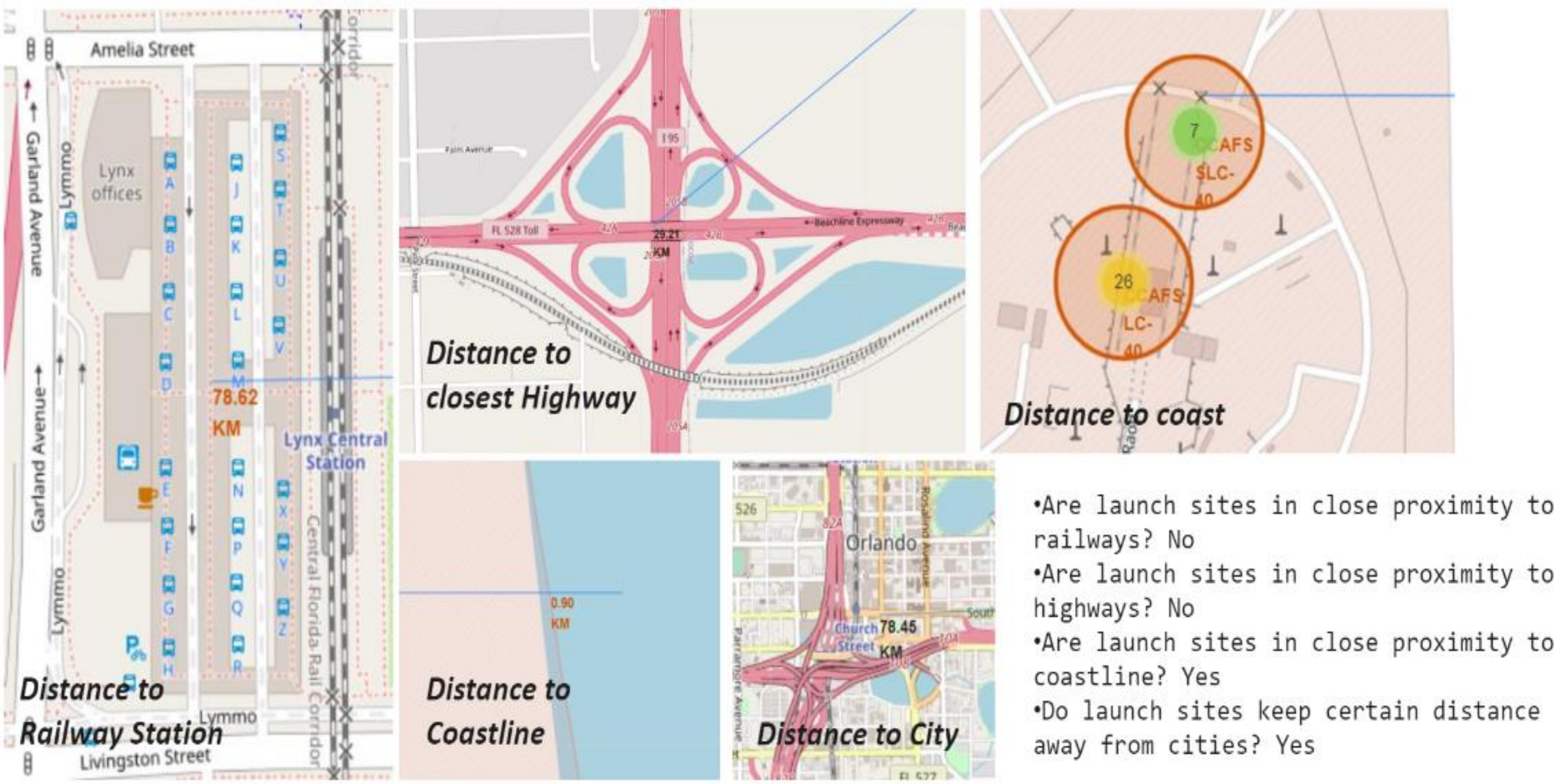
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

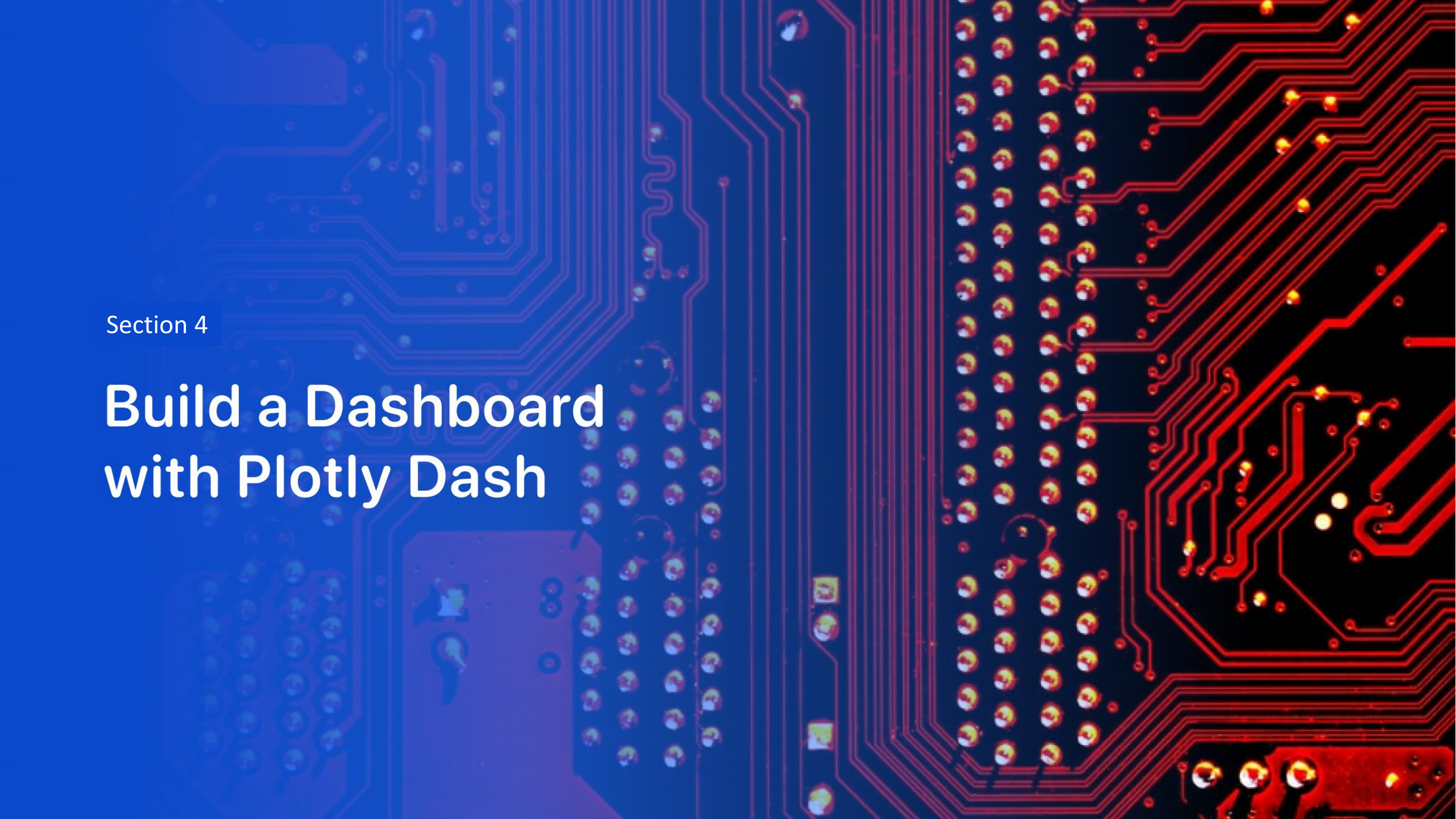
California Launch Site

37

Launch Site distance to landmarks



- •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

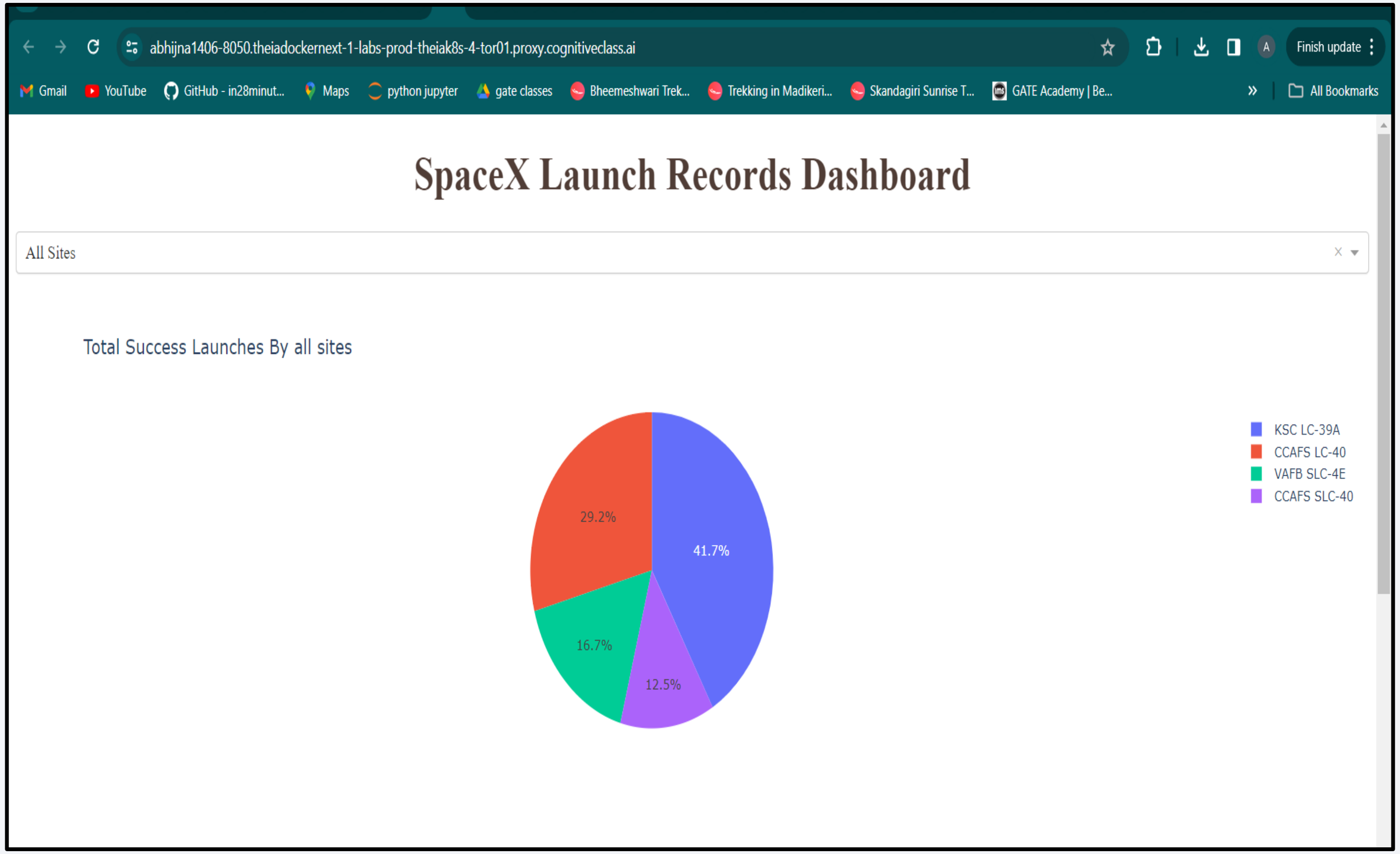


Section 4

Build a Dashboard with Plotly Dash

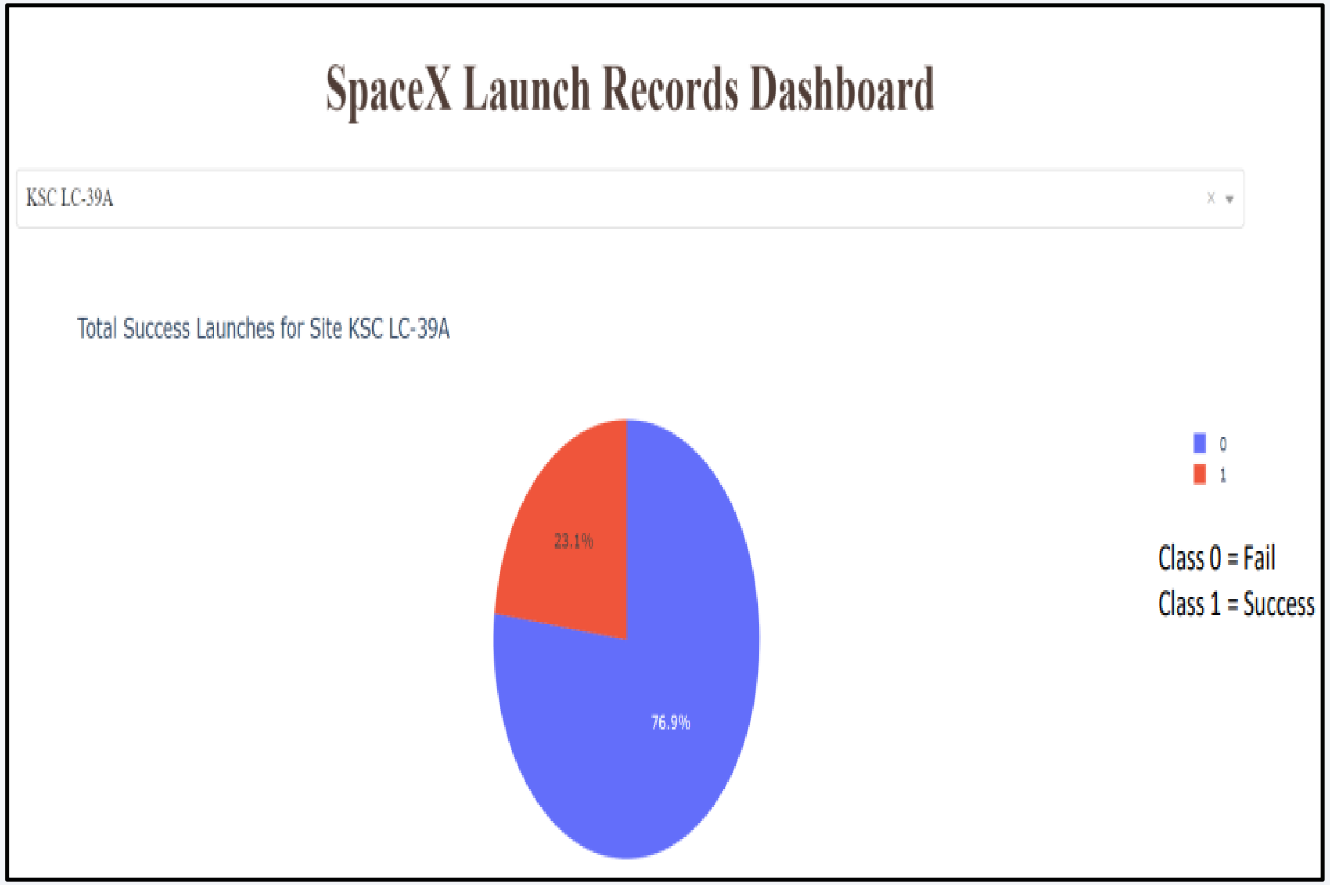
Pie chart showing the success percentage achieved by each launch site

- Using Plotly Dash, a dashboard was created
- The first graph is a piechart containing the count of successful launches by site
- There is also a dropdown where is possible to chose a specific Site
- From all sites, KSC LC-39A had the most successful launches



Highest Success Rate Site

- Using the dashboard interactivity it was possible to conclude that KSC LC-39A has the highest success rate, with 76,9% of successful launches
- The success rate of other sites:
  - CCAFS LC-40: 73,1%
  - CCAFS SLC-40: 57,1%
  - VFAB SLC-4E: 60%



SpaceX Launch Records Dashboard

KSC LC-39A

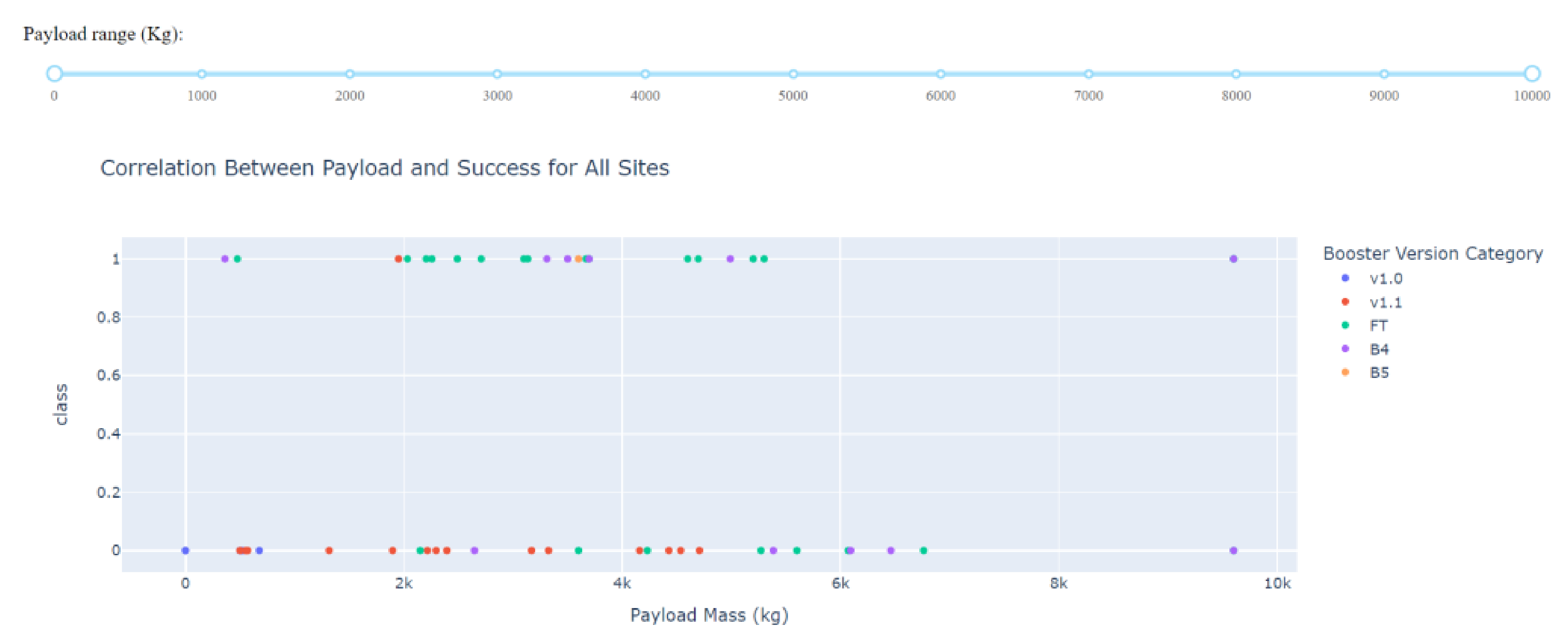
Total Success Launches for Site KSC LC-39A

Class 0 = Fail

Class 1 = Success

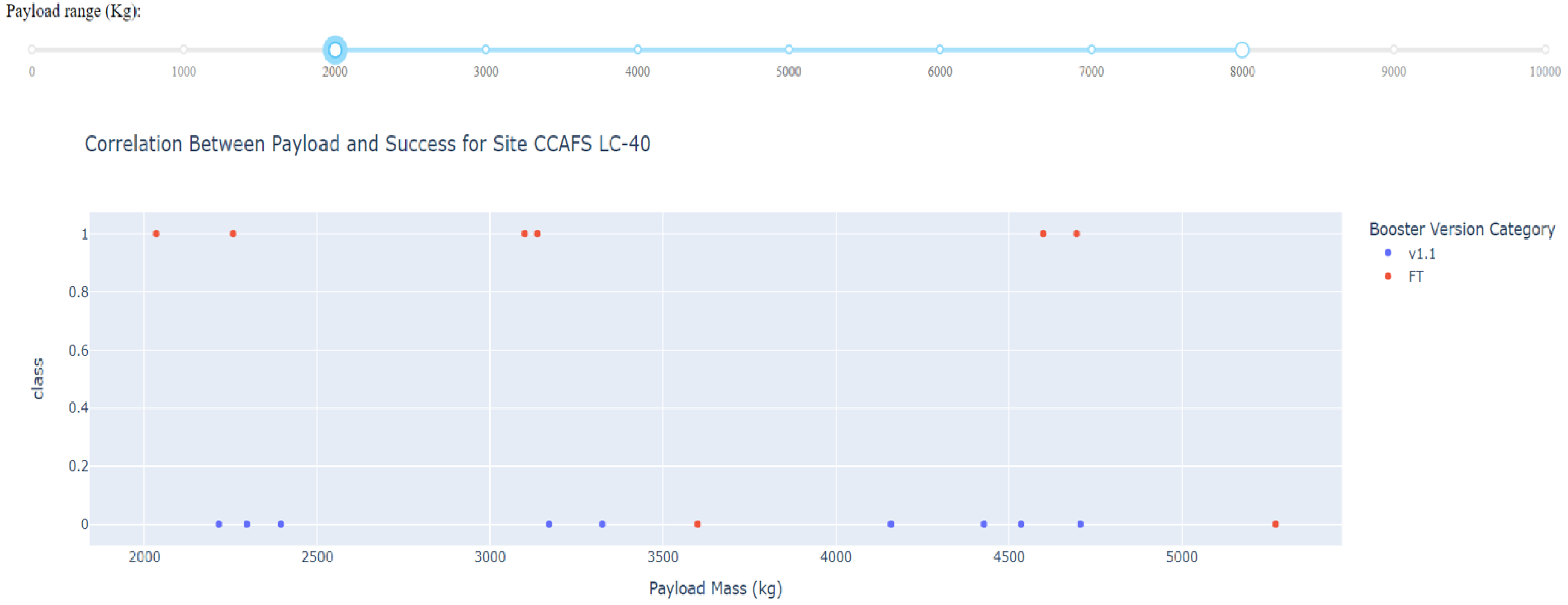
Scatterplot Payload vs Launch Outcome

- The chart of the dashboard is a scatterplot of the payload mass vs launch outcome, colored by booster version category;
- There is a range slider where is possible to change the payload range;



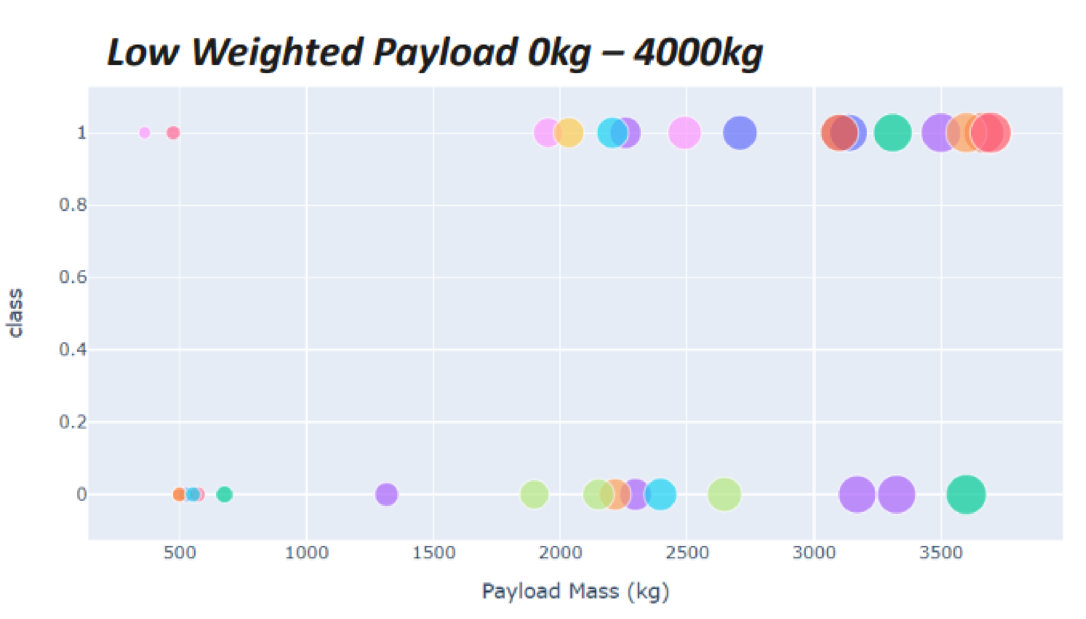
Scatterplot Payload vs Launch Outcome

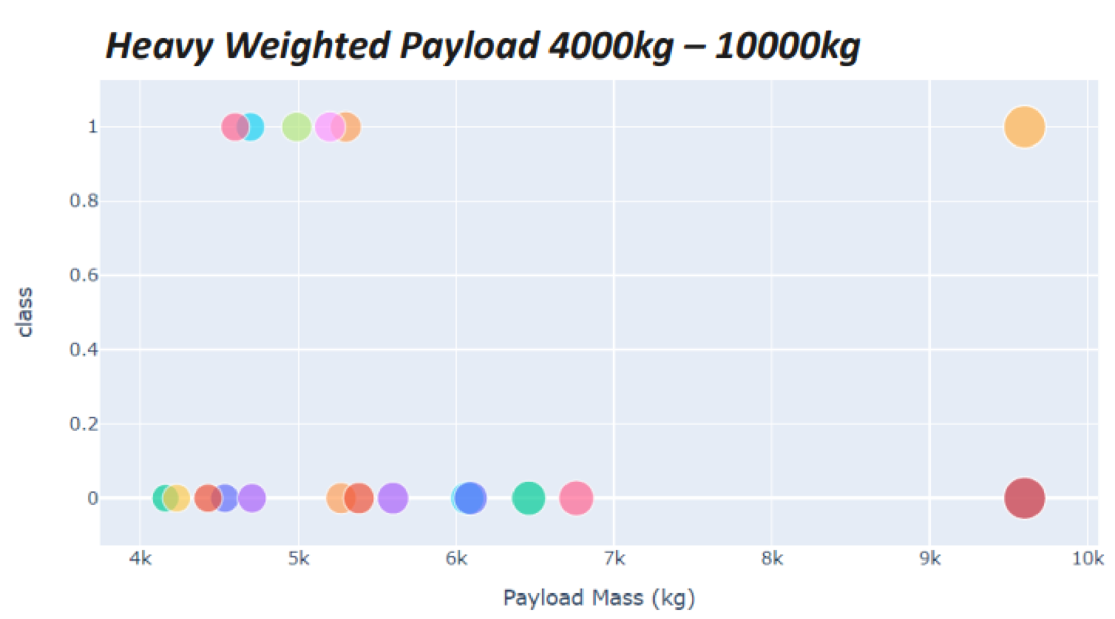
- Payloads between 2,000 kg and 5,000 kg have the highest success rate



Scatterplot Payload vs Launch Outcome

- Success rates for low weighted payloads is higher than heavy weighted payloads



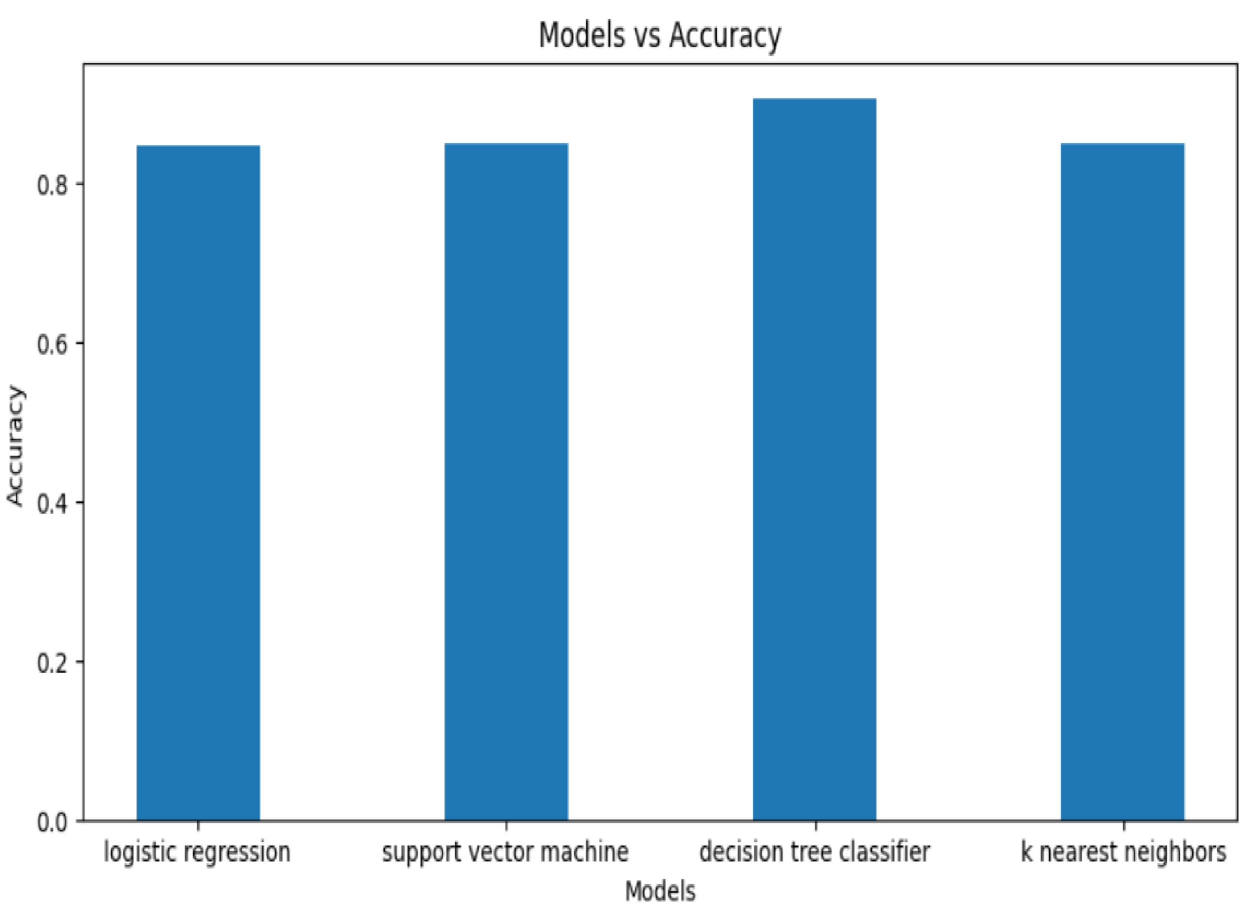


Section 5

Predictive Analysis (Classification)

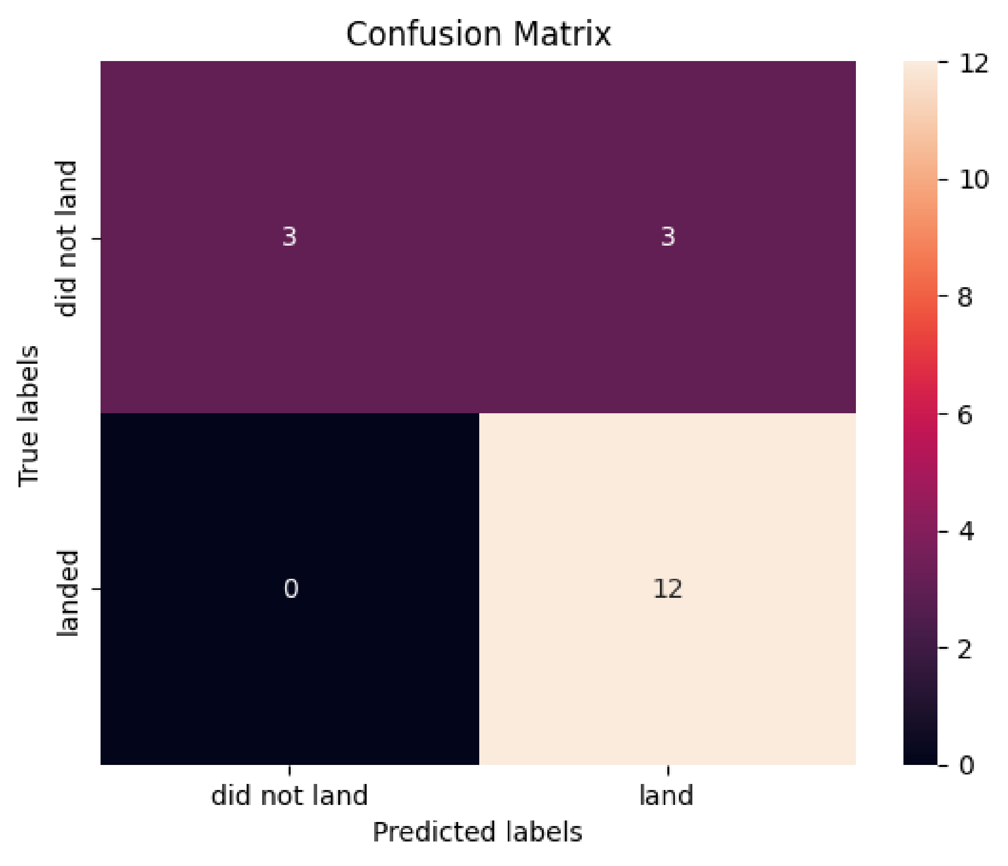
Classification Accuracy

Best model is Decision Tree with the highest accuracy of 0.90.



Confusion Matrix

- Decision Tree Model is best fit.
- The model predicted 12 successful landings when the true label was successful landing.
- The model predicted 0 unsuccessful landings when the true label was landed.
- The model predicted 3 unsuccessful landings when the true label was unsuccessful landings
- The model predicted 3 successful landings when true label was unsuccessful landing.
- The models over predict successful landings.



Conclusions

Problem: To develop a machine learning model for Space Y who wants to compete against SpaceX.

- The goal of model is to predict whether Stage 1 will successfully land or fail to land.
- Different data sources (API and Wiki Page) were analyzed.
- The best launch site is KSC LC 39-A
- Equator: Most of the launch sites are near the equator for an additional natural boost - due to the rotational speed of earth - which helps save the cost of putting in extra fuel and boosters
- Coast: All the launch sites are close to the coast

Conclusions

- Launch Success: Increases over time
- KSC LC-39A: Has the highest success rate among launch sites. Has a 100% success rate for launches less than 5,500 kg
- Orbits: ES-L1, GEO, HEO, and SSO have a 100% success rate
- Payload Mass: Across all launch sites, the higher the payload mass (kg), the higher the success rate
- Launches above payloads 7000kg are more successful.
- Decision Tree Classifier can be used to predict successful landings and increase profits.

Thank you!