IBM Developer
SKILLS NETWORK

IBM DATA SCIENCE CAPSTONE PROJECT PRESENTATION

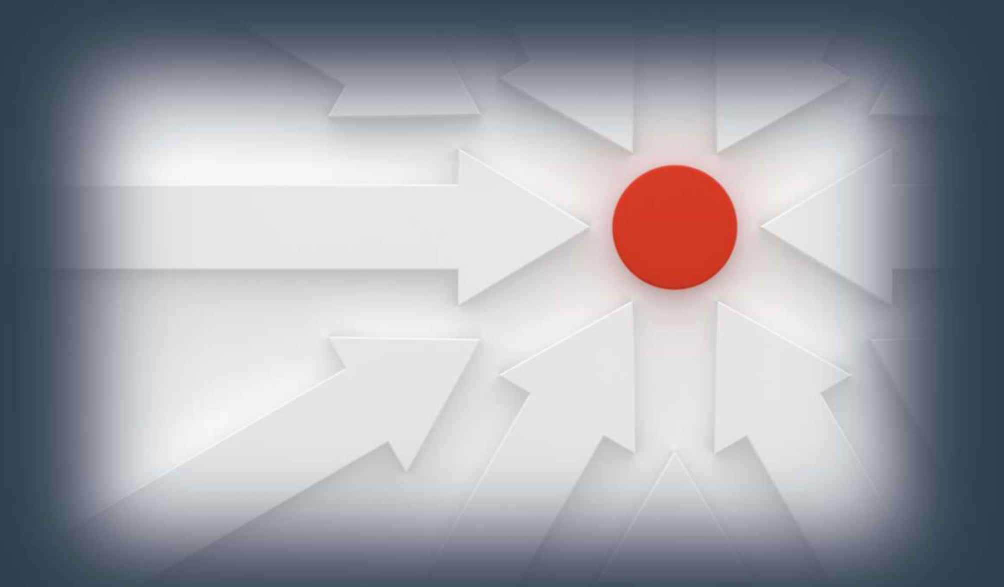
Nana Abena Kumi
November 2024

Outline

Executive Summary

- Summary of methodologies
  - Data Collection via API & Web scraping
  - Data Wrangling
  - Exploratory Data Analysis (EDA) using SQL
  - EDA with Visualization
  - Interactive maps using Folium
  - Dashboards with Ploty Dash
  - Using Predictive Analysis

- Summary of all results
  - Results of Exploratory Data Analysis
  - Dashboards and Interactive Maps
  - Results of predictive Analytics Performed



Introduction

- Project background and context
    - SpaceX is a space technology company that is known for its role in reducing the cost of space travel and making space more accessible. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; while other providers cost upward of 165 million dollars each. The aim of the project is to predict if the first stage will land successfully and determine the cost of the launch using public data and machine learning models. The results of this project may be useful for an alternate company.
- Problem Statement
    - Most space travel providers rocket launches cost upwards of $165m, about 100 million dollars less that what SpaceX is providing. How? This is because the company can reuse the first stage.
    - If we can determine which factors contribute to a successful landing and how they contribute to it, we will be able to replicate similar cost savings for an alternate company, say SpaceY.

Section 1

METHODOLOGY

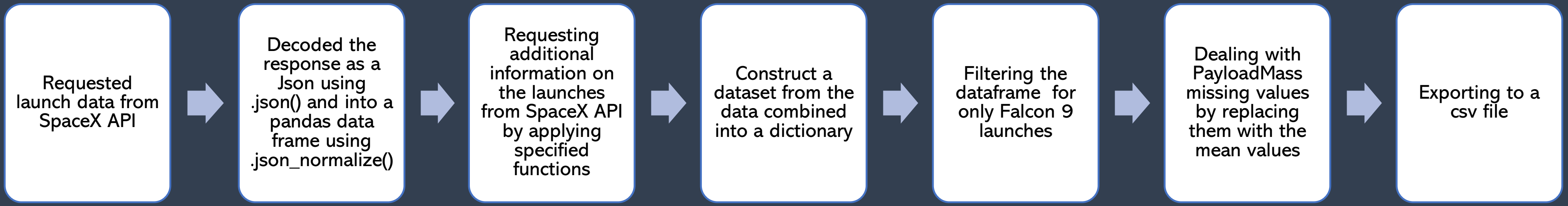
Methodology

Executive Summary

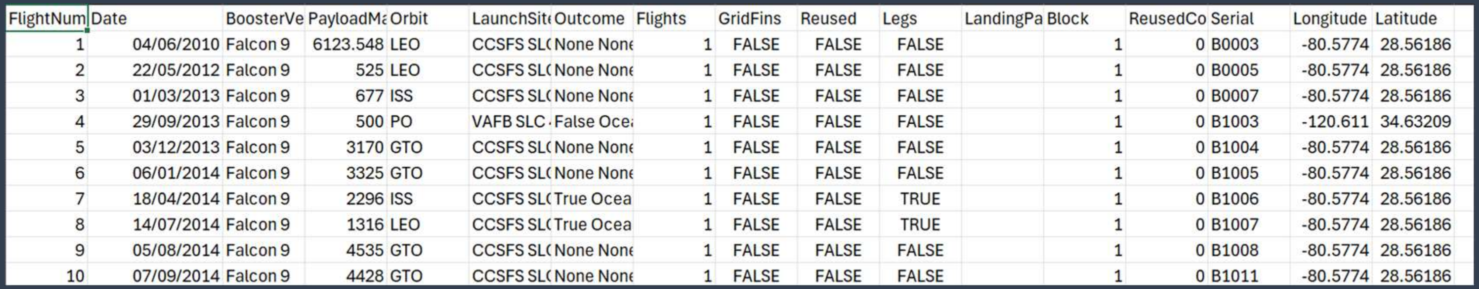
- Data collection methodology:
    - Data collected using SpaceX Rest API
    - Web scraping from Wikipedia (https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches)
- Perform data wrangling
    - Handling missing values
    - Using One Hot Encoding to classify the landing outcomes
- 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 – SpaceX API

Data collection steps


Requested launch data from SpaceX API
Decoded the response as a Json using .json() and into a pandas data frame using .json_normalize()
Requesting additional information on the launches from SpaceX API by applying specified functions
Construct a dataset from the data combined into a dictionary
Filtering the dataframe for only Falcon 9 launches
Dealing with PayloadMass missing values by replacing them with the mean values
Exporting to a csv file

Extract of output file below

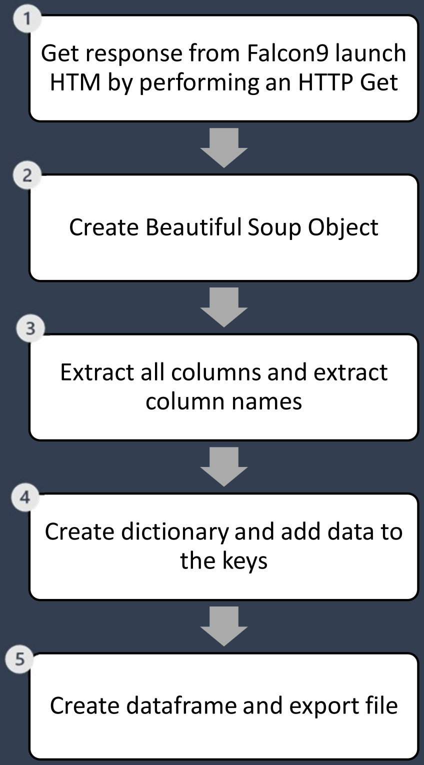


FlightNum	Date	BoosterVe	PayloadMa	Orbit	LaunchSit	Outcome	Flights	GridFins	Reused	Legs	LandingPa	Block	ReusedCo	Serial	Longitude	Latitude
1	04/06/2010	Falcon 9	6123.548	LEO	CCSFS SL(	None None	1	FALSE	FALSE	FALSE		1	0	B0003	-80.5774	28.56186
2	22/05/2012	Falcon 9	525	LEO	CCSFS SL(	None None	1	FALSE	FALSE	FALSE		1	0	B0005	-80.5774	28.56186
3	01/03/2013	Falcon 9	677	ISS	CCSFS SL(	None None	1	FALSE	FALSE	FALSE		1	0	B0007	-80.5774	28.56186
4	29/09/2013	Falcon 9	500	PO	VAFB SLC -	False Oce	1	FALSE	FALSE	FALSE		1	0	B1003	-120.611	34.63209
5	03/12/2013	Falcon 9	3170	GTO	CCSFS SL(	None None	1	FALSE	FALSE	FALSE		1	0	B1004	-80.5774	28.56186
6	06/01/2014	Falcon 9	3325	GTO	CCSFS SL(	None None	1	FALSE	FALSE	FALSE		1	0	B1005	-80.5774	28.56186
7	18/04/2014	Falcon 9	2296	ISS	CCSFS SL(	True Ocea	1	FALSE	FALSE	TRUE		1	0	B1006	-80.5774	28.56186
8	14/07/2014	Falcon 9	1316	LEO	CCSFS SL(	True Ocea	1	FALSE	FALSE	TRUE		1	0	B1007	-80.5774	28.56186
9	05/08/2014	Falcon 9	4535	GTO	CCSFS SL(	None None	1	FALSE	FALSE	FALSE		1	0	B1008	-80.5774	28.56186
10	07/09/2014	Falcon 9	4428	GTO	CCSFS SL(	None None	1	FALSE	FALSE	FALSE		1	0	B1011	-80.5774	28.56186

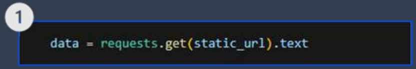
Github Link : Data Collection

https://github.com/Nana-ty/ML-Project/blob/main/dataset_part_1.csv

Data Collection – Web Scraping

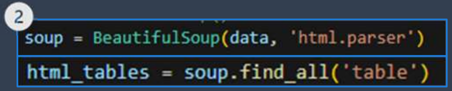


1 Get response from Falcon9 launch HTM by performing an HTTP Get
2 Create Beautiful Soup Object
3 Extract all columns and extract column names
4 Create dictionary and add data to the keys
5 Create dataframe and export file



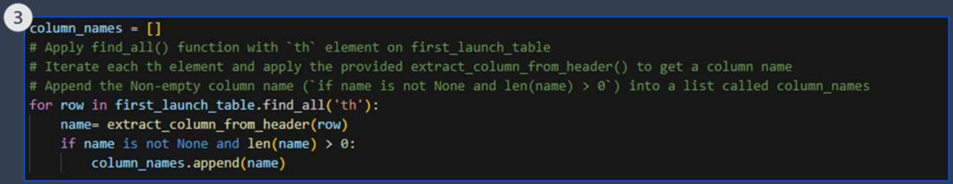
1

```
data = requests.get(static_url).text
```



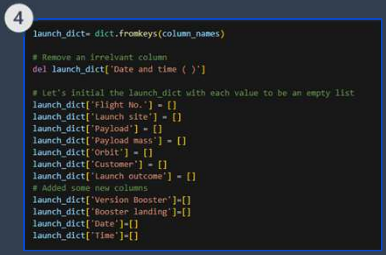
2

```
soup = BeautifulSoup(data, 'html.parser')
html_tables = soup.find_all('table')
```



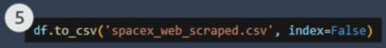
3

```
column_names = []
# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
for row in first_launch_table.find_all('th'):
    name= extract_column_from_header(row)
    if name is not None and len(name) > 0:
        column_names.append(name)
```



4

```
launch_dict= dict.fromkeys(column_names)

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

# Let's initial the launch_dict with each value to be an empty list
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'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```



5

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

Webscraping Link

Data Wrangling

Using the dataset from the previous step (webscraping), pandas is used to perform data analysis

1

Using the *.value_counts()* method to:

1. Determine the number of launches per launch site
2. Calculate the number and occurrence of each orbit and
3. Calculate the number & occurrence of mission outcome per orbit type

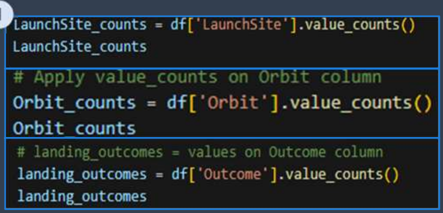
2

Create landing outcome label from the outcome column. An outcome is assigned 0, if in the set bad_outcome, otherwise it's a 1 to a list called landing_class

3

Create a column called Class that contains the values from landing_class and export to csv

1

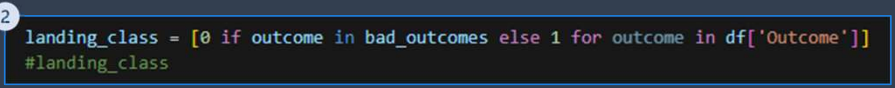


```
LaunchSite_counts = df['LaunchSite'].value_counts()
LaunchSite_counts
```

```
# Apply value_counts on Orbit column
Orbit_counts = df['Orbit'].value_counts()
Orbit_counts
```

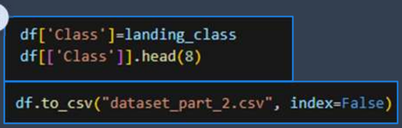
```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

2



```
landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome']]
#landing_class
```

3



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

```
df.to_csv("dataset_part_2.csv", index=False)
```

Data Wrangling Link   dataset_part_2.csv

EDA with Data Visualization

Three types of charts were used for plotted for the data visualization section

Scatter plots were used to visualise the relationships between:
- Payload Mass and Flight Number
- Launch Site and Flight Number
- Payload Mass and Launch Site
- Orbit and Flight Number

The scatter plot was used to visualise the relationship/correlation between two variables

Bar Charts was used to visualise the success rate of each orbit.
- Average Class and Orbit

The bar chart was used to compare the numerical value (class) to a categorical value (orbit)

Line Charts was used to visualise the launch success yearly trend.
- Class and year

The line chart was used to determine trends over time

Data Visualization   dataset_part_3.csv

EDA with SQL

SQL queries performed on the dataset were:

- Display the names of the unique launch sites in the space mission
- Display 5 record where launch sites begin with the string 'KSC'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display the average payload mass carried by booster version F9 v1.1
- List the date where the successful landing outcome in drone ship was achieved
- List the names of boosters which have success in ground pas and have payload mass greater than 4000 but less than 6000kg
- List the total number/count of successful and failed mission outcomes
- List the names of booster_versions which have carried the maximum payload mass
- List the month names, successful (ground pad) landing outcomes, booster version and launch sites for the months in 2017
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between 04/06/2010 and 20/03/2017 in descending order

Github SQL Link

Build an Interactive Map with Folium

The following steps were to visualize the launch site locations and find some geographical patterns.

- TASK 1 : Mark all launch sites on a map.
  - Create a folium Map object with NASA Johnson Space Center to be the initial center location
  - Used folium.Circle and folium.Marker to add a highlighted circle and text label for each launch site

- TASK 2 : Mark the success/fails launches for each site on the map
  - Added marker colors to the launch sites to indicate if the launch site was a success or failure

- TASK 3 : Calculate the distances between a launch sites
  - Calculated the distance between the coastline point and the launch sites to its proximities
  - Used folium.PolyLine to display the line between the launch site and the coastline. For ex. We can see that launchsite CCAFS LC-40 is relatively close to a highway

Github Link

Build a Dashboard with Plotly Dash

Plots/Graphs & Interactions added include:

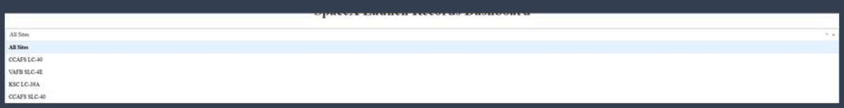
1. Pie chart (Success Pie Chart) to display/visualize successful launches:

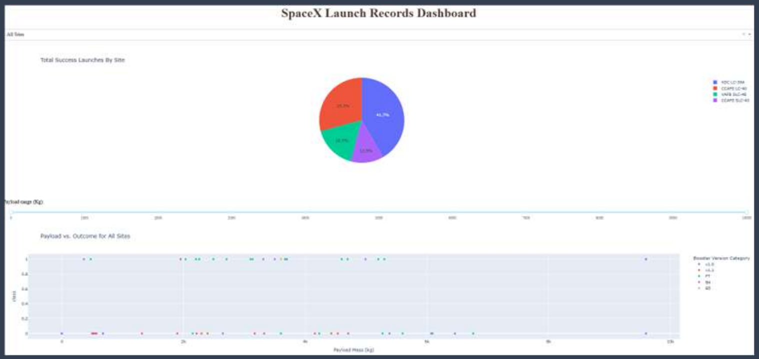
- At an overall level for all sites
- Per each site showing the split between successful and failed launches. A dropdown option (interaction added) to give the user the option to toggle between all sites and a specified site.

2. Scatter Plot (Success-Payload Scatter Plot) to visualize & understand the relationship between the payload(kg) and the launch outcomes:

- At an overall level for all sites
- Per each site . A payload range slider (interaction) was added to adjust the chart by range of payload masses.
- The chart can also be filtered by booster version category

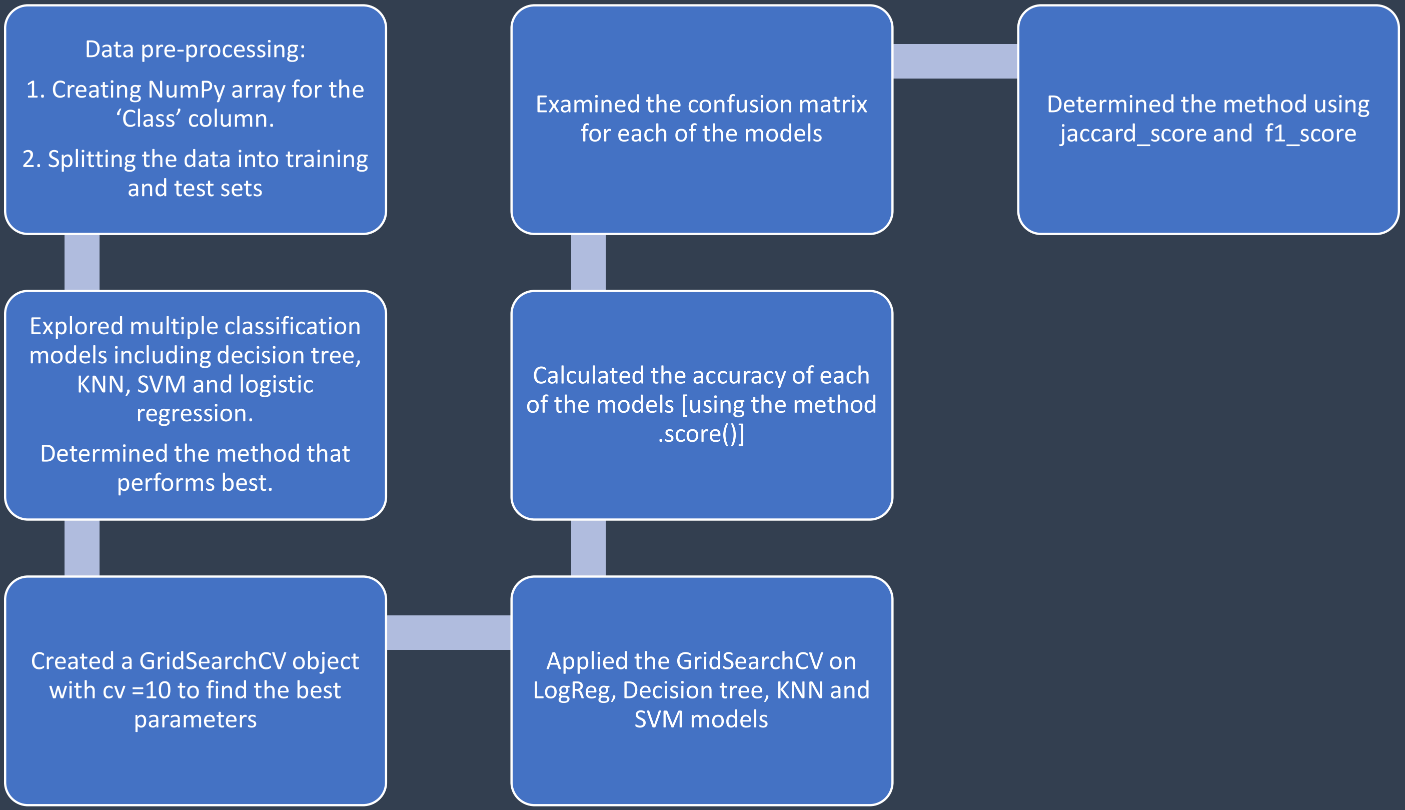
[spacex_dash_app github link]





SpaceX Launch Records Dashboard
All Sites
Total Success Launches By Site
Payload range (Kg):
Payload vs. Outcome for All Sites
Booster Version Category
Payload Mass (kg)

Predictive Analysis (Classification)



Data pre-processing:

1. Creating NumPy array for the 'Class' column.
2. Splitting the data into training and test sets

Explored multiple classification models including decision tree, KNN, SVM and logistic regression.

Determined the method that performs best.

Created a GridSearchCV object with cv =10 to find the best parameters

Applied the GridSearchCV on LogReg, Decision tree, KNN and SVM models

Calculated the accuracy of each of the models [using the method .score()]

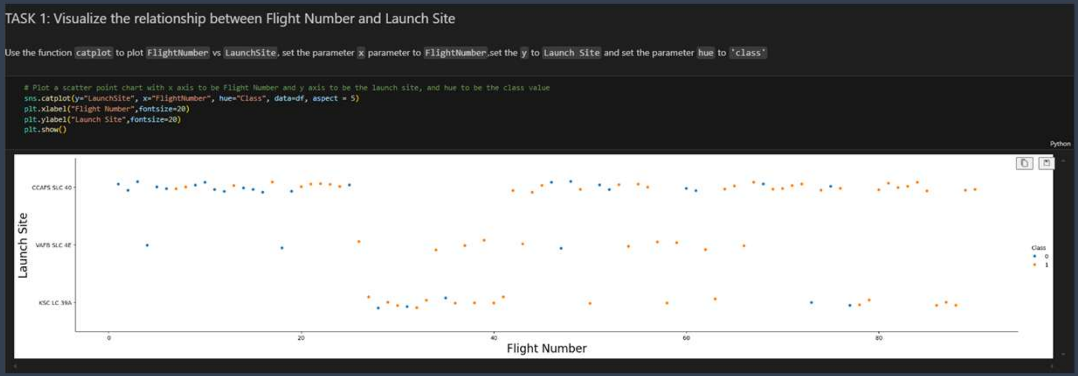
Examined the confusion matrix for each of the models

Determined the method using jaccard_score and f1_score

GitHub URL : ML Prediction

Results

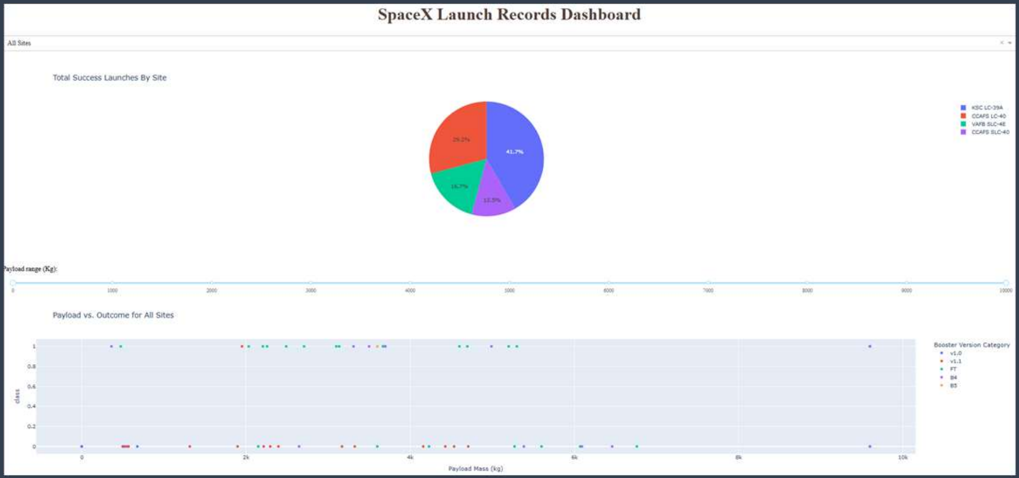
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

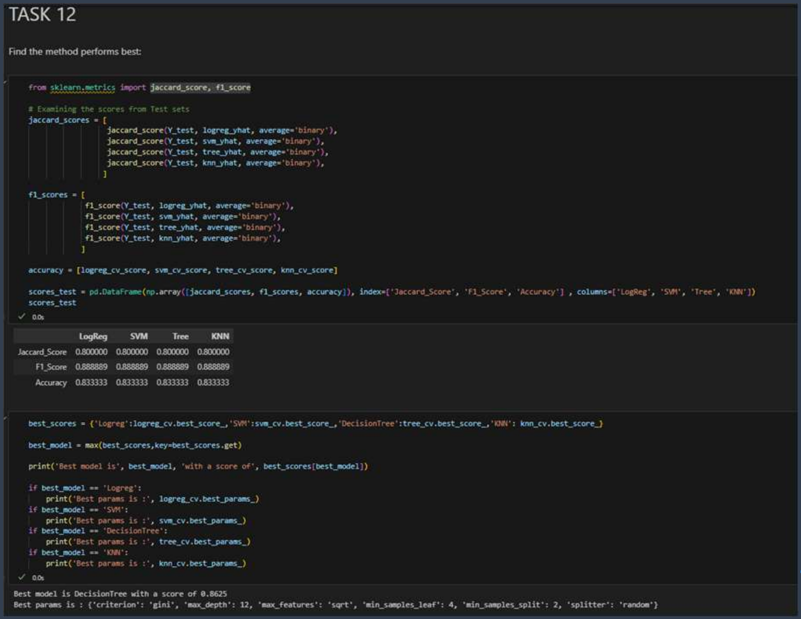


TASK 1: Visualize the relationship between Flight Number and Launch Site

Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

```python
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```





TASK 12

Find the method performs best:

```python
from sklearn.metrics import jaccard_score, f1_score

# Examining the scores from Test sets
jaccard_scores = [
                  jaccard_score(Y_test, logreg_yhat, average='binary'),
                  jaccard_score(Y_test, svm_yhat, average='binary'),
                  jaccard_score(Y_test, tree_yhat, average='binary'),
                  jaccard_score(Y_test, knn_yhat, average='binary'),
                 ]

f1_scores = [
             f1_score(Y_test, logreg_yhat, average='binary'),
             f1_score(Y_test, svm_yhat, average='binary'),
             f1_score(Y_test, tree_yhat, average='binary'),
             f1_score(Y_test, knn_yhat, average='binary'),
            ]

accuracy = [logreg_cv_score, svm_cv_score, tree_cv_score, knn_cv_score]

scores_test = pd.DataFrame(np.array([jaccard_scores, f1_scores, accuracy]), index=['Jaccard_Score', 'F1_Score', 'Accuracy'] , columns=['LogReg', 'SVM', 'Tree', 'KNN'])
scores_test
```

✓ 0.0s

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.800000	0.800000	0.800000	0.800000
F1_Score	0.888889	0.888889	0.888889	0.888889
Accuracy	0.833333	0.833333	0.833333	0.833333

```python
best_scores = {'Logreg':logreg_cv.best_score_,'SVM':svm_cv.best_score_,'DecisionTree':tree_cv.best_score_,'KNN': knn_cv.best_score_}

best_model = max(best_scores,key=best_scores.get)

print('Best model is', best_model, 'with a score of', best_scores[best_model])

if best_model == 'Logreg':
    print('Best params is :', logreg_cv.best_params_)
if best_model == 'SVM':
    print('Best params is :', svm_cv.best_params_)
if best_model == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if best_model == 'KNN':
    print('Best params is :', knn_cv.best_params_)
```

✓ 0.0s

```
Best model is DecisionTree with a score of 0.8625
Best params is : {'criterion': 'gini', 'max_depth': 12, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 2, 'splitter': 'random'}
```

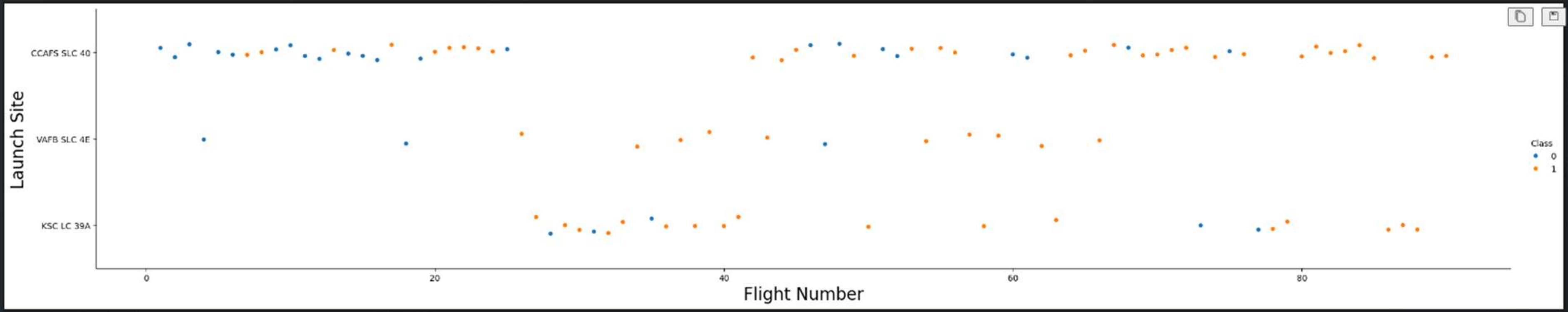
Section 2

INSIGHTS DRAWN FROM EDA

Flight Number vs. Launch Site

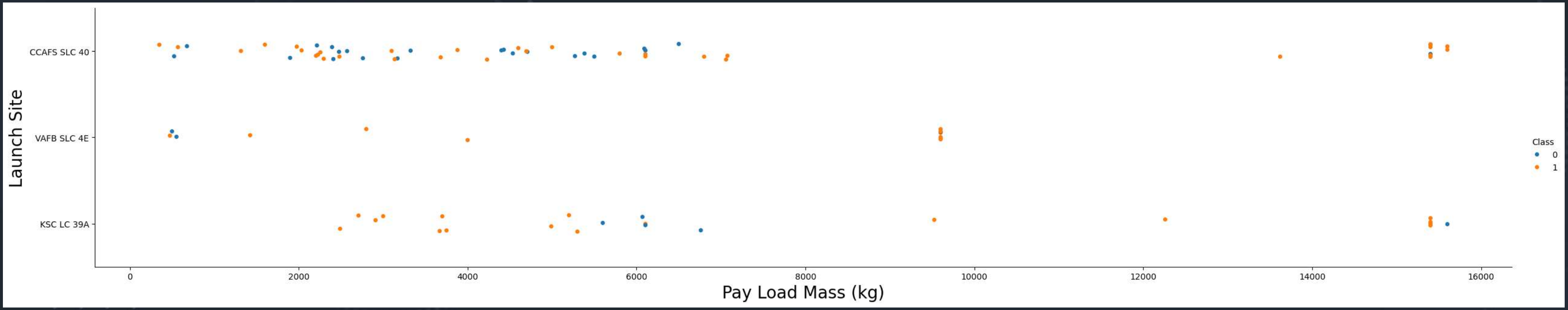
Explanation

- From the plot, at the higher flight numbers there were more successful landings across all three launch sites.



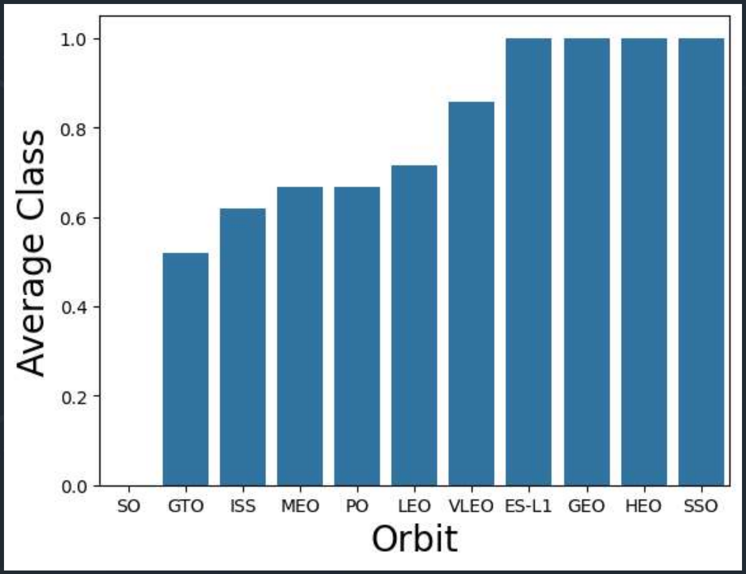
Payload vs. Launch Site

- Explanation
- Overall, it appears that the higher the payload mass the higher the success rate of landing
- Particularly, over 8,000kg there were more successful landings
- There is variability of success at CCAFS SLC 40 launch site during the lower payload masses.



Success Rate vs. Orbit Type

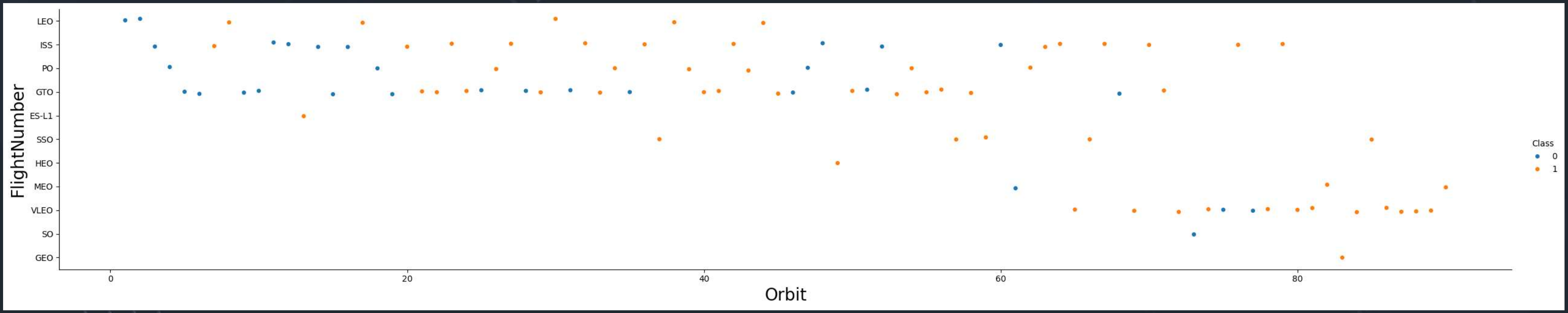
- Explanation
- The orbits ES-L1, GEO, HEO and SSO had 100% success rates with other orbits with lower success rates.
- Orbit SO had 0% success rate.



Flight Number vs. Orbit Type

Explanation

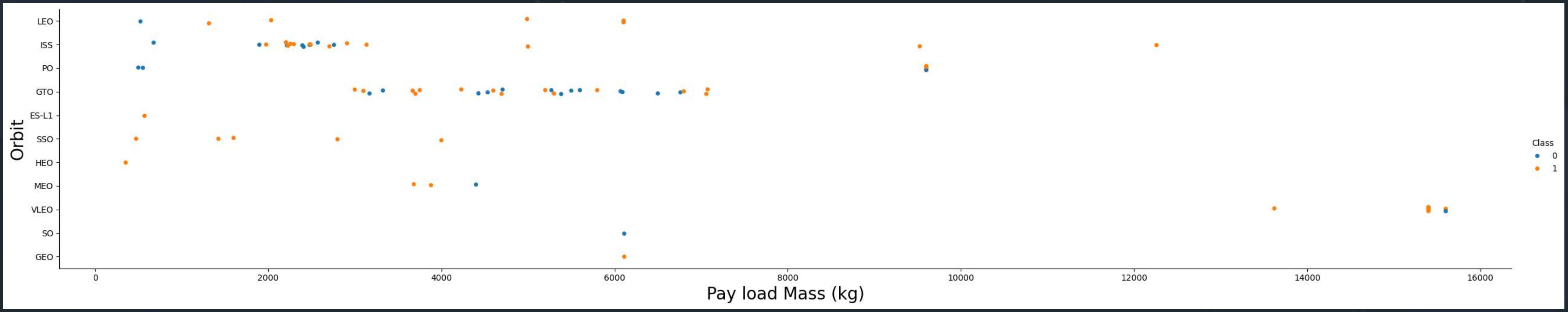
- The success rates of launches increases over the flight numbers with VLEO and ISS having very high success rates
- Launches to GTO, PO, ISS & LEO have mixed results at the earlier launches.
- * Note: the x & y labels should be the other way round



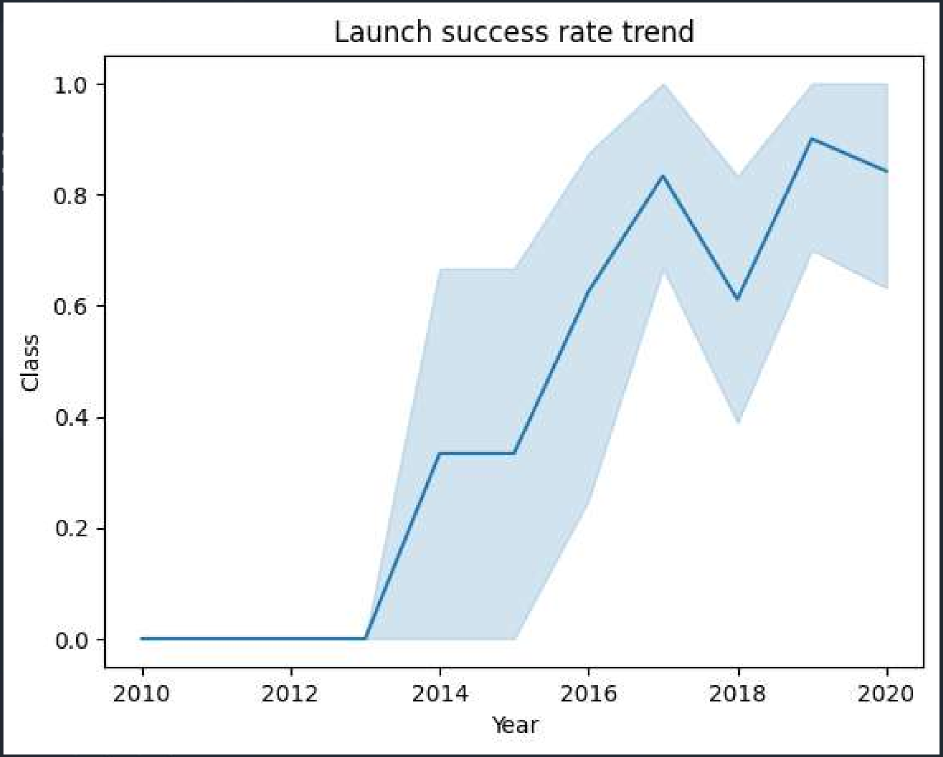
Pay Load Mass vs. Orbit Type

Explanation

- The lighter pay loads have mixed results
- There are fewer launches t heavier payloads, however more successful.



Launch Success Yearly Trend



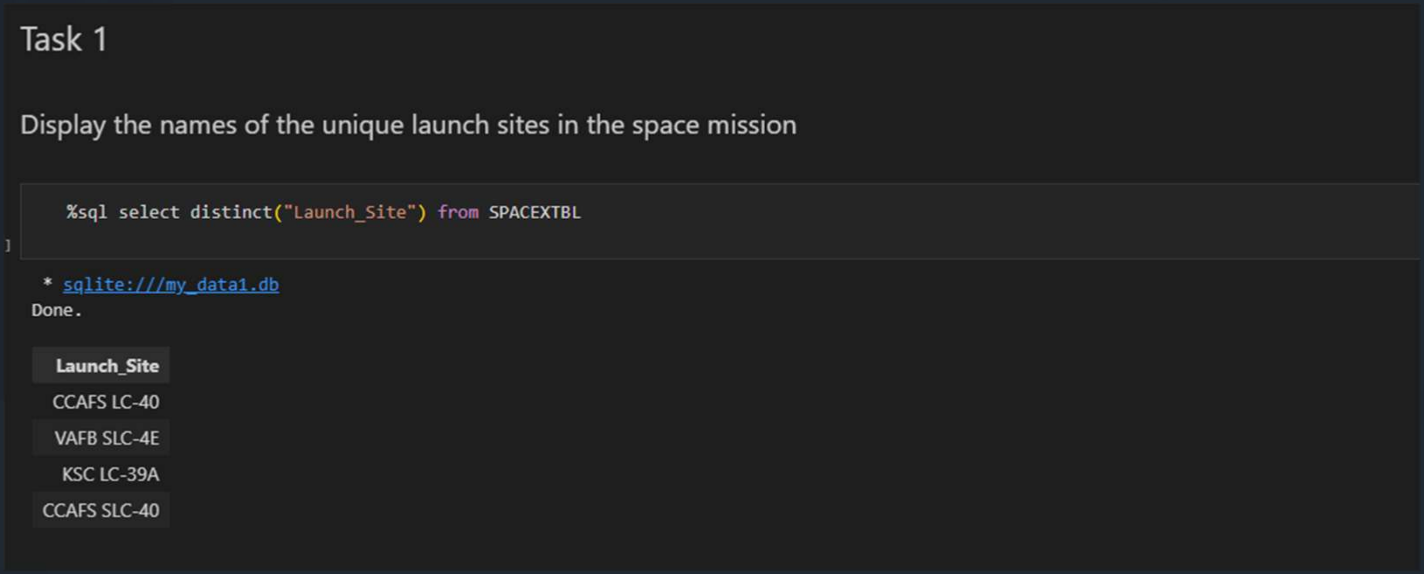
Explanation

- The success rate increased over time with a dip in 2018 and picked up in 2020

All Launch Site Names

Explanation

- Used “distinct” to determine the unique site names



Task 1

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

```
%sql select distinct("Launch_Site") from SPACEXTBL
```

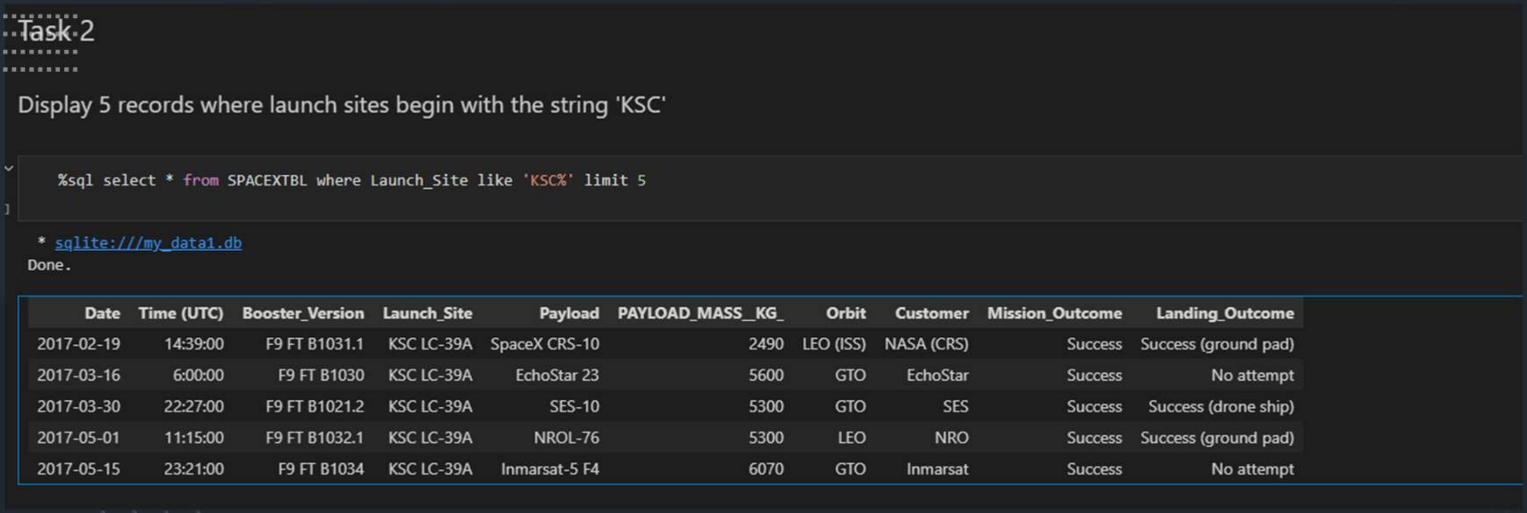
```
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'KSC'

Explanation:

- Used the query below.



Task 2

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

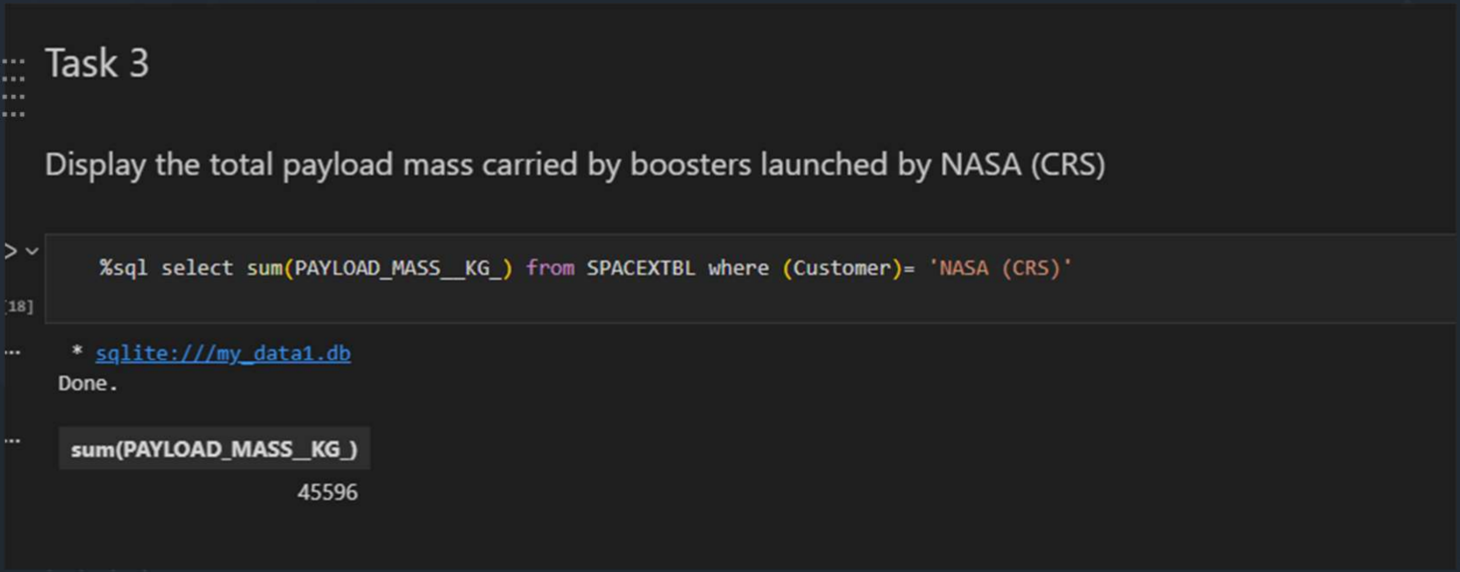
```
%sql select * from SPACEXTBL where Launch_Site like 'KSC%' limit 5
```

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-03-16	6:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	No attempt
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
2017-05-01	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
2017-05-15	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	No attempt

Total Payload Mass

- Used the function below to calculate the total payload mass



Task 3

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

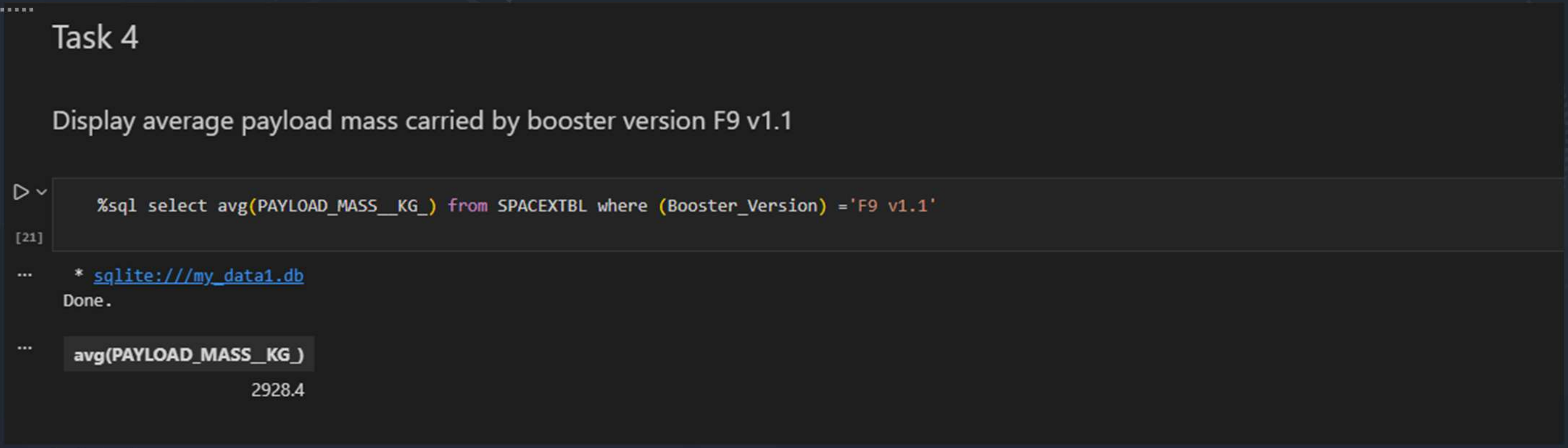
```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where (Customer)= 'NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- The query below was used to calculate the average payload mass



Task 4

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

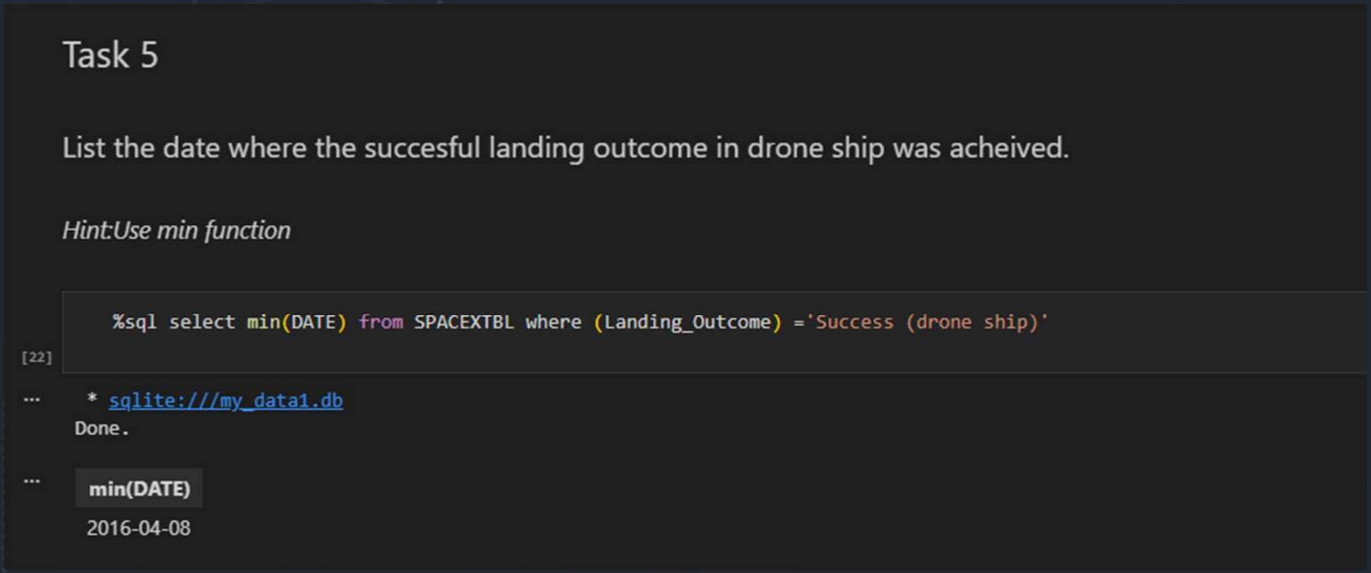
```
[21]
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where (Booster_Version) ='F9 v1.1'
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- Query below was used. The use of min function results in the earliest date where the landing outcome was a success



Task 5

List the date where the succesful landing outcome in drone ship was acheived.

Hint:Use min function

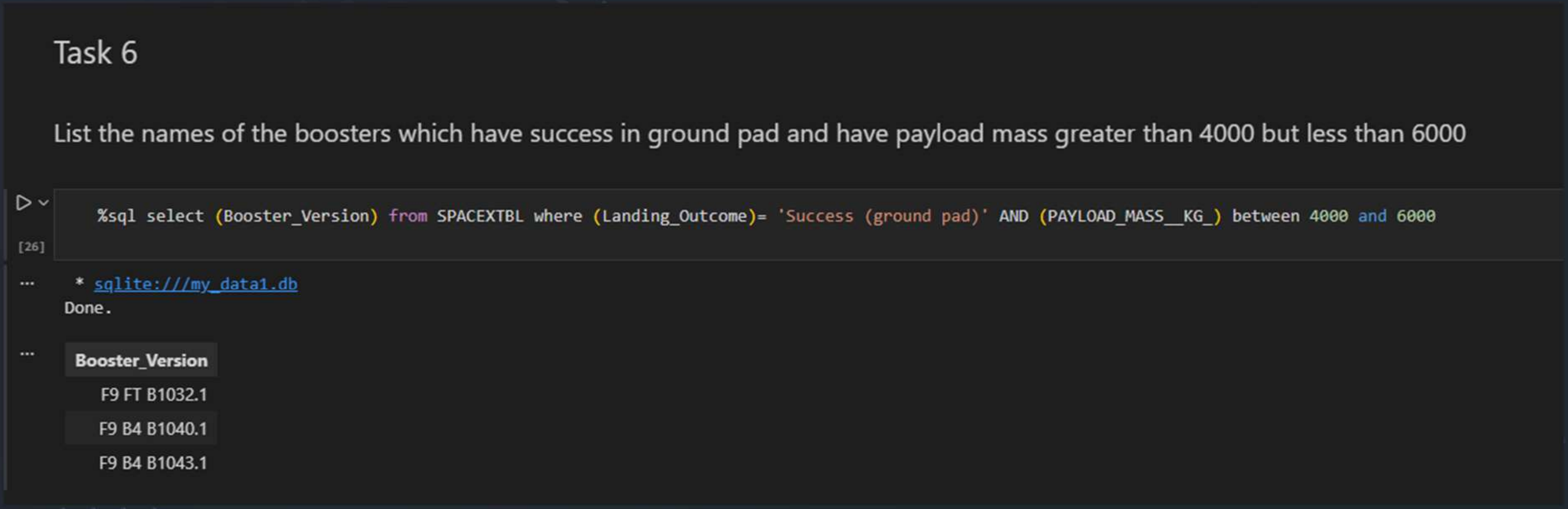
```
%sql select min(DATE) from SPACEXTBL where (Landing_Outcome) ='Success (drone ship)'
```

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

min(DATE)
2016-04-08

Successful Drone Ship Landing with Payload between 4000 and 6000

- Query used is in below screenshot.



Task 6

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

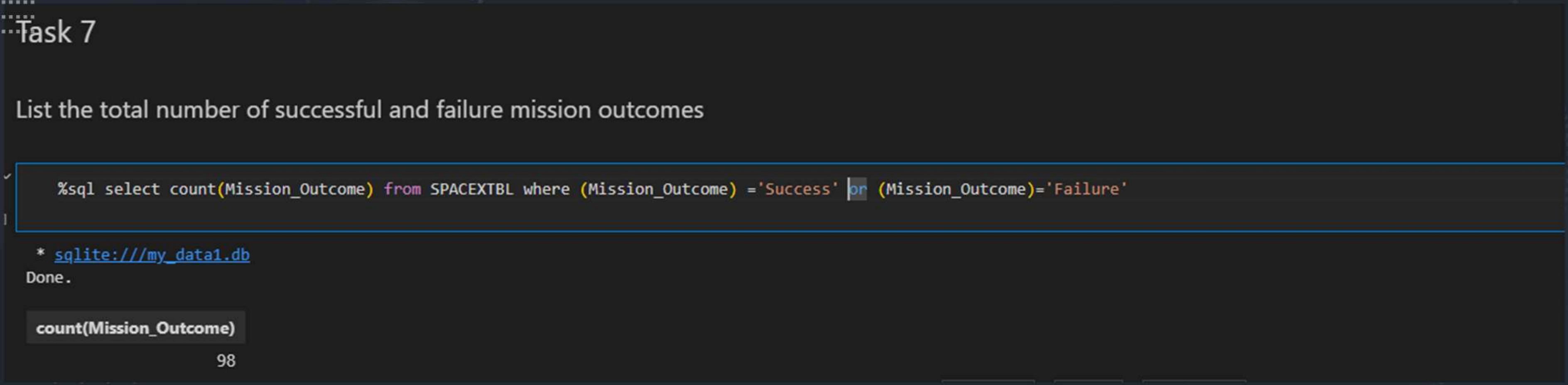
```
%sql select (Booster_Version) from SPACEXTBL where (Landing_Outcome)= 'Success (ground pad)' AND (PAYLOAD_MASS__KG_) between 4000 and 6000
```

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

Booster_Version
F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1

Total Number of Successful and Failure Mission Outcomes

- Used the query below to determine the total number of mission outcomes



Task 7

List the total number of successful and failure mission outcomes

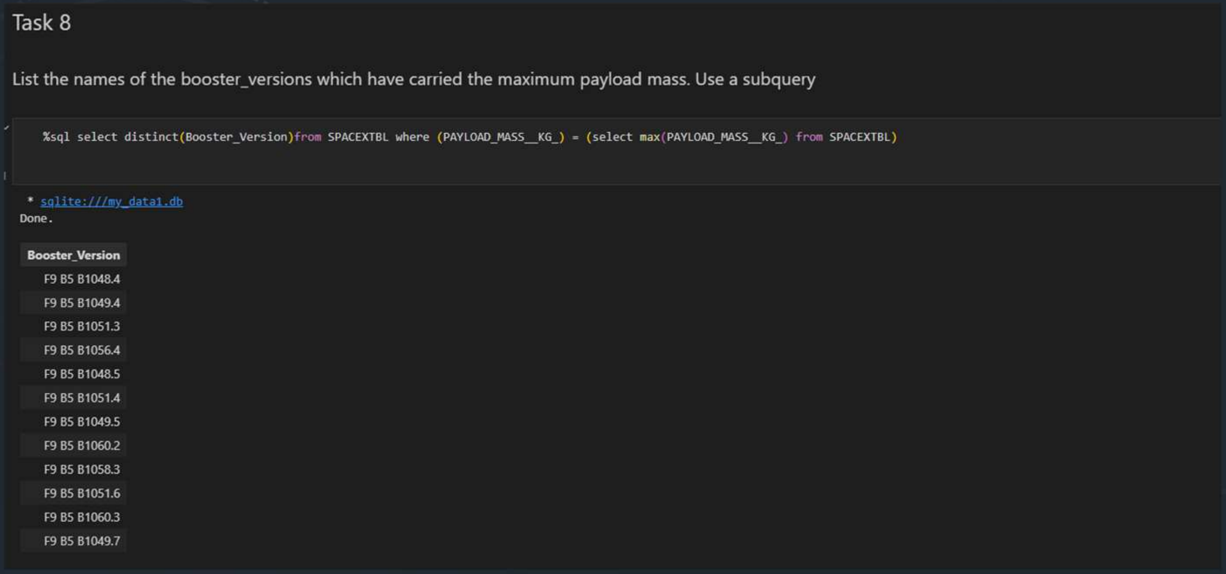
```
%sql select count(Mission_Outcome) from SPACEXTBL where (Mission_Outcome) ='Success' or (Mission_Outcome)='Failure'
```

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

count(Mission_Outcome)
98

Boosters Carried Maximum Payload

- Used a subquery to determine the maximum payload and then set the booster versions that carried that maximum



Task 8

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

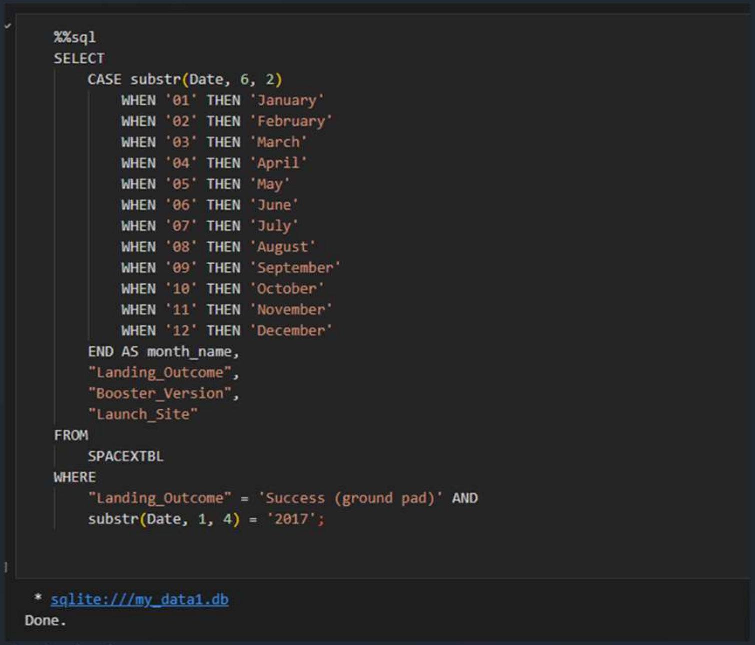
```
%sql select distinct(Booster_Version)from SPACEXTBL where (PAYLOAD_MASS__KG_) = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

```
 * 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

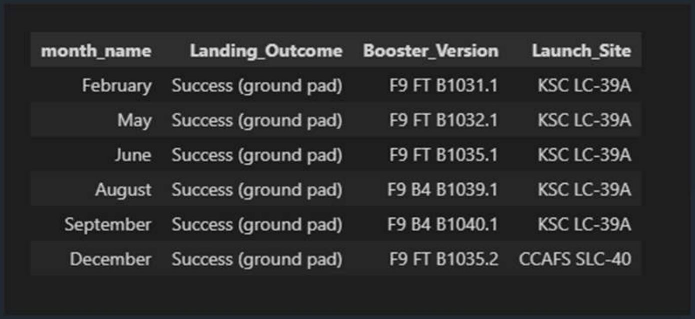
2017 Launch Records

- Query and results in screenshot below. SQLLite at the time of running the code doesn't support month names so use substr.



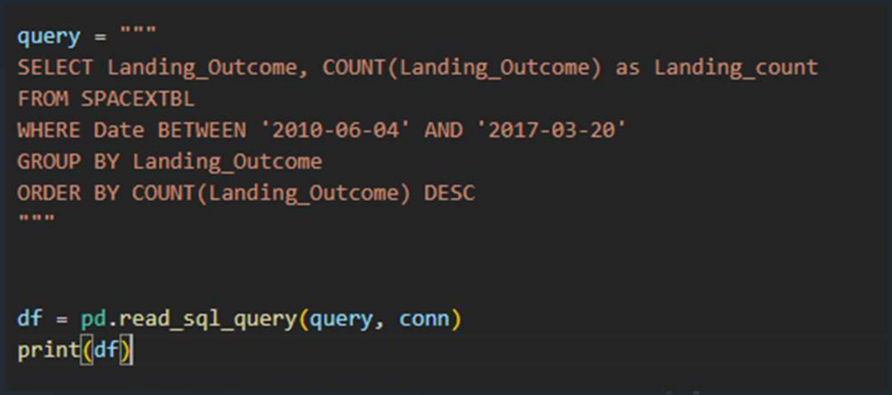
```
%%sql
SELECT
    CASE substr(Date, 6, 2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END AS month_name,
    "Landing_Outcome",
    "Booster_Version",
    "Launch_Site"
FROM
    SPACEXTBL
WHERE
    "Landing_Outcome" = 'Success (ground pad)' AND
    substr(Date, 1, 4) = '2017';
```

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

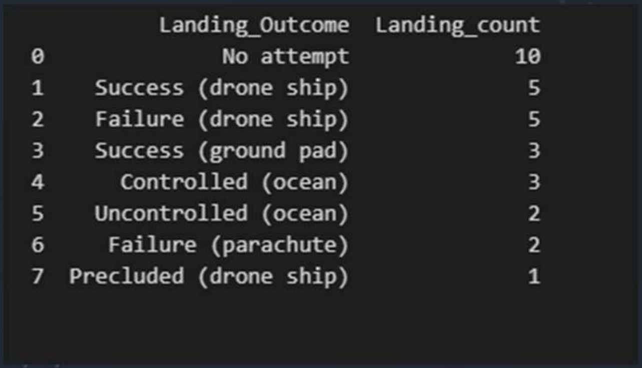


month_name	Landing_Outcome	Booster_Version	Launch_Site
February	Success (ground pad)	F9 FT B1031.1	KSC LC-39A
May	Success (ground pad)	F9 FT B1032.1	KSC LC-39A
June	Success (ground pad)	F9 FT B1035.1	KSC LC-39A
August	Success (ground pad)	F9 B4 B1039.1	KSC LC-39A
September	Success (ground pad)	F9 B4 B1040.1	KSC LC-39A
December	Success (ground pad)	F9 FT B1035.2	CCAFS SLC-40

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



```
query = """
SELECT Landing_Outcome, COUNT(Landing_Outcome) as Landing_count
FROM SPACEXTBL
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY COUNT(Landing_Outcome) DESC
"""


df = pd.read_sql_query(query, conn)
print(df)
```



```
          Landing_Outcome  Landing_count
0              No attempt             10
1      Success (drone ship)            5
2      Failure (drone ship)            5
3     Success (ground pad)             3
4       Controlled (ocean)             3
5     Uncontrolled (ocean)             2
6      Failure (parachute)             2
7   Precluded (drone ship)             1
```

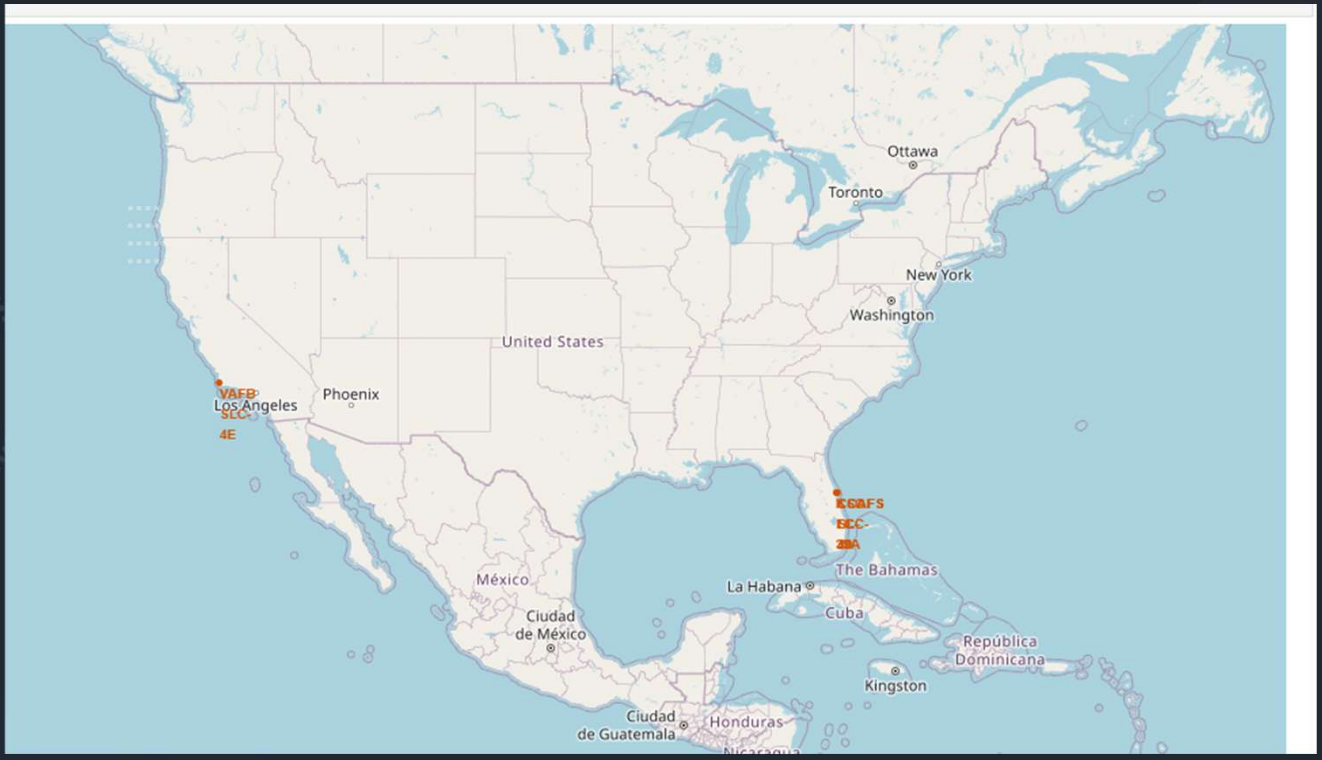
- Query is ranking the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

Section 3

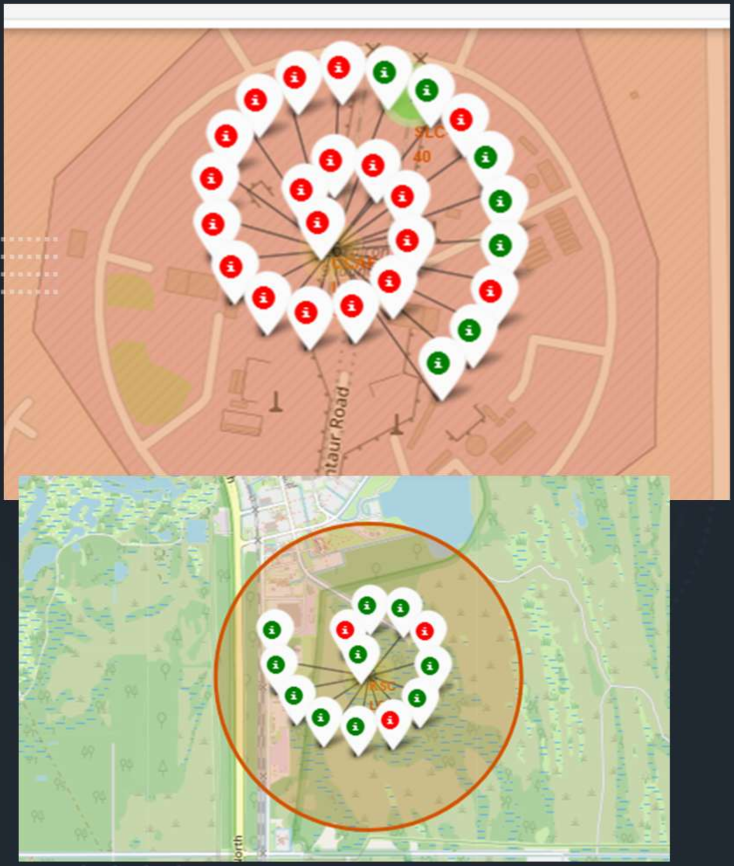
Launch Sites Proximities Analysis

All launch sites' location markers on a global map

- The launch sites are in the United Sates along the coast in Florida and California.
- The sites are not really close to the equator which has a latitude of 0.

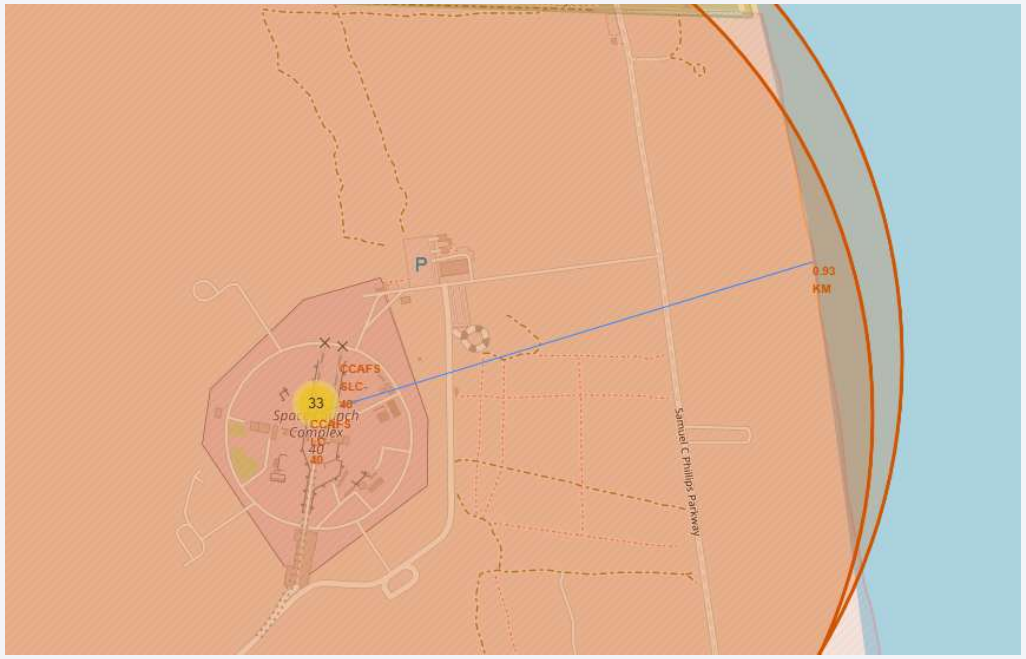


Color-labeled launch outcomes of sites



- The color labeled markers show the success rates at the launch sites.
- Green markers represents a successful launch; RED representing a failed launch.
- Launch Site KSC LC-39A has a higher success rate.

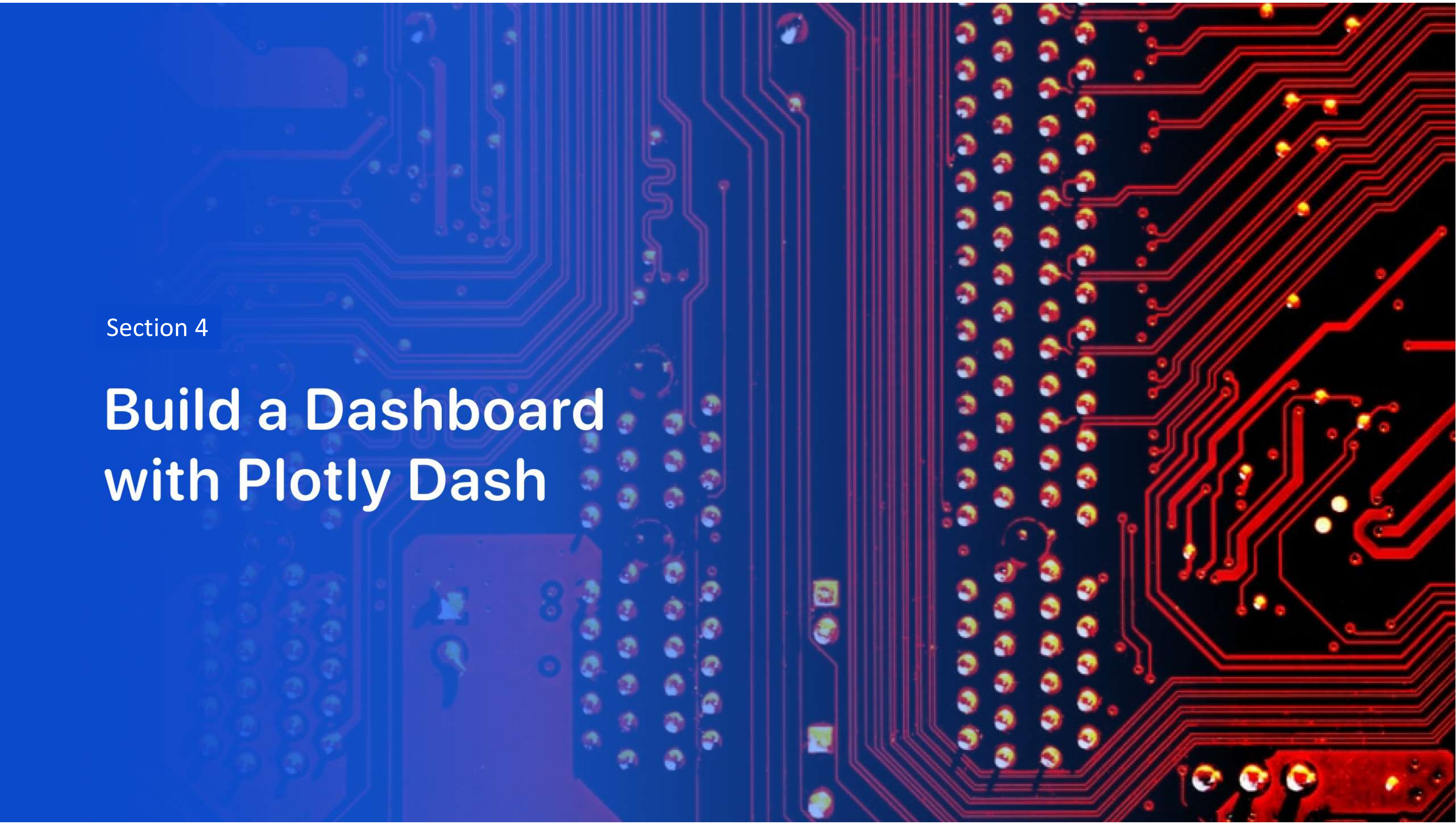
Launch site distance to coastline



- The screenshot shows the distance to the coastline is about 0.9km

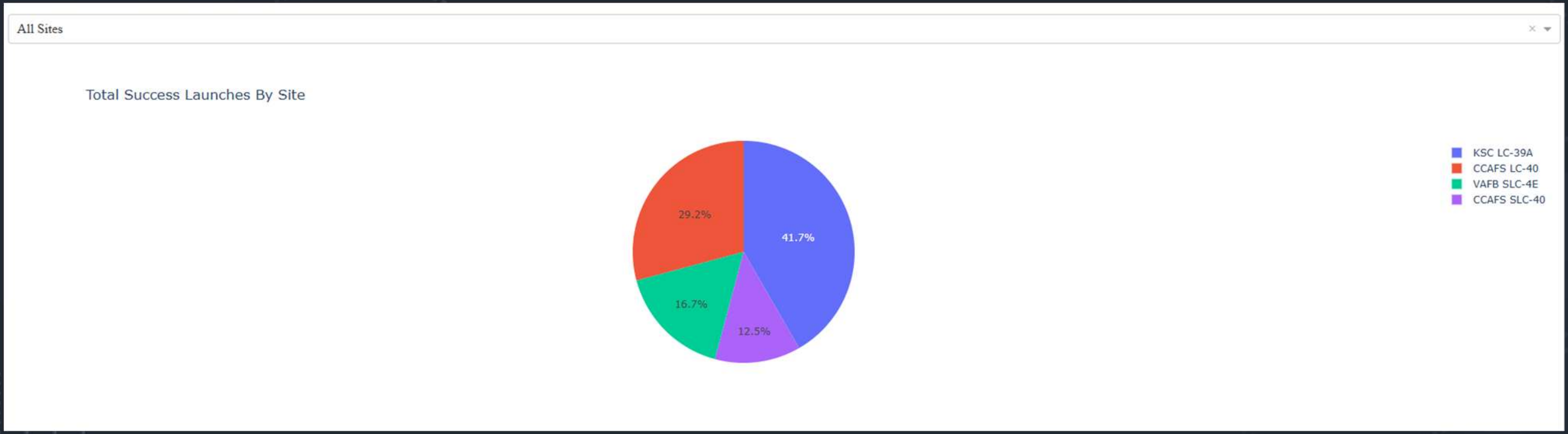
Section 4

Build a Dashboard with Plotly Dash



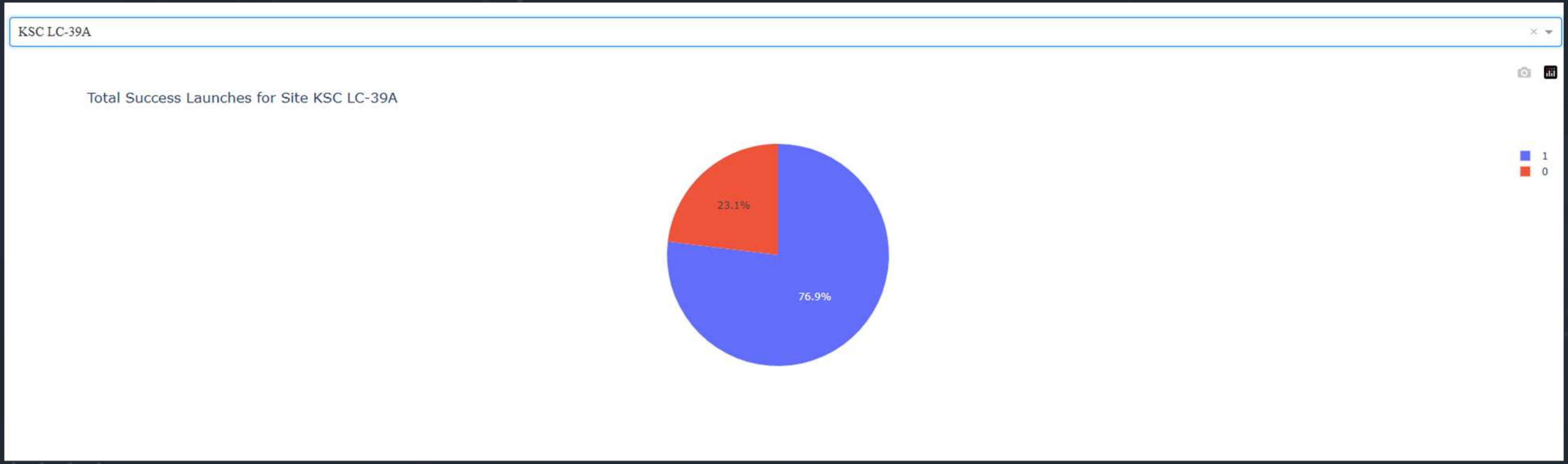
Launch Success for all Sites

- This chart shows the success rate for all the sites

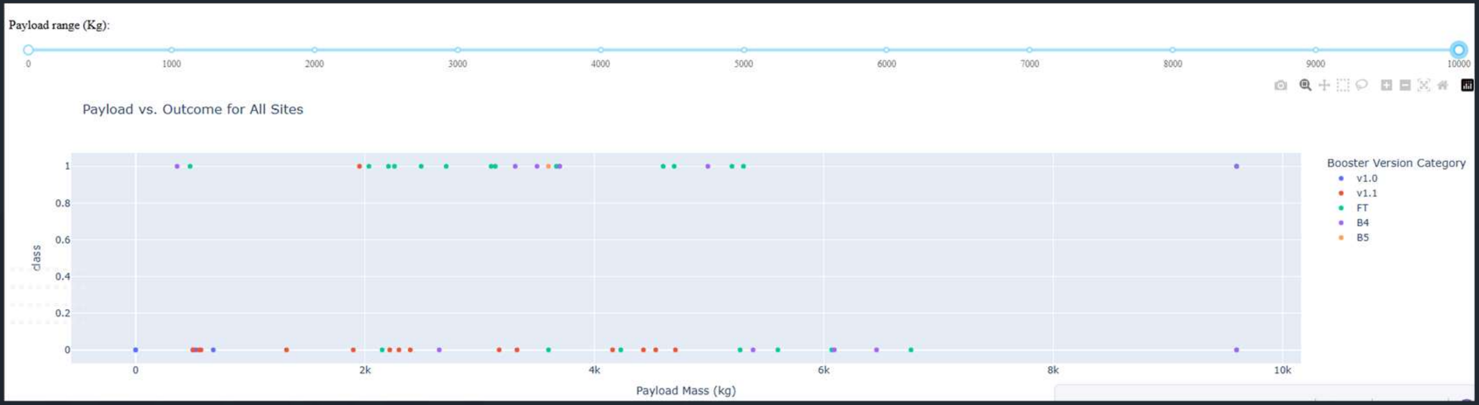


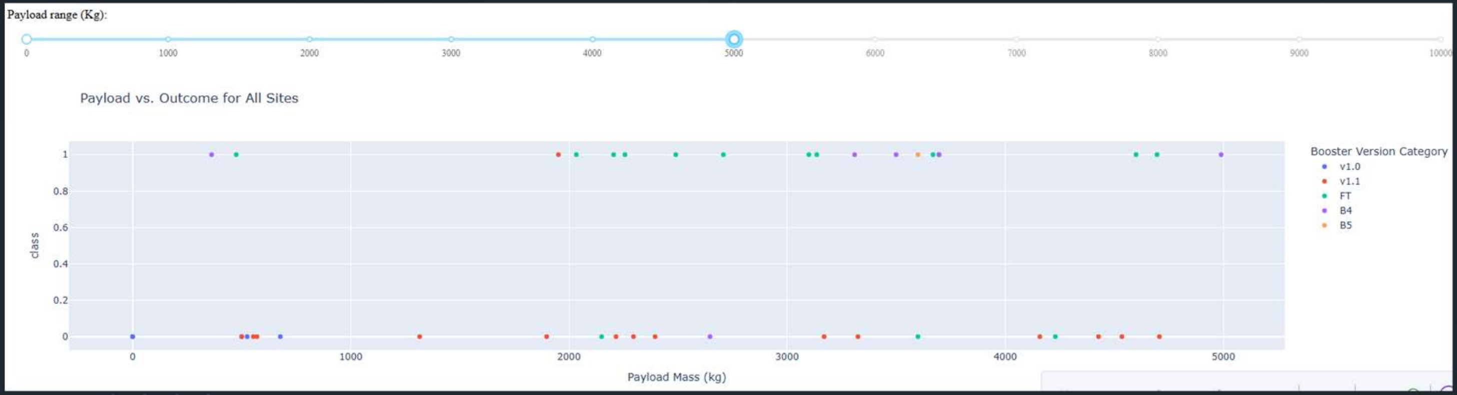
Piechart for the launch site with highest launch success ratio

- KSC LC-39A has the most successful launches



All sites Payload vs. Launch Outcome scatter plot, with different payload





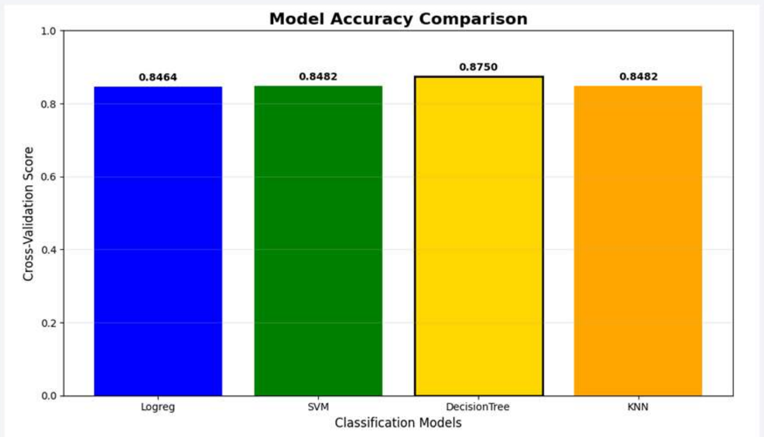
- The charts show the payload vs class at different payloads

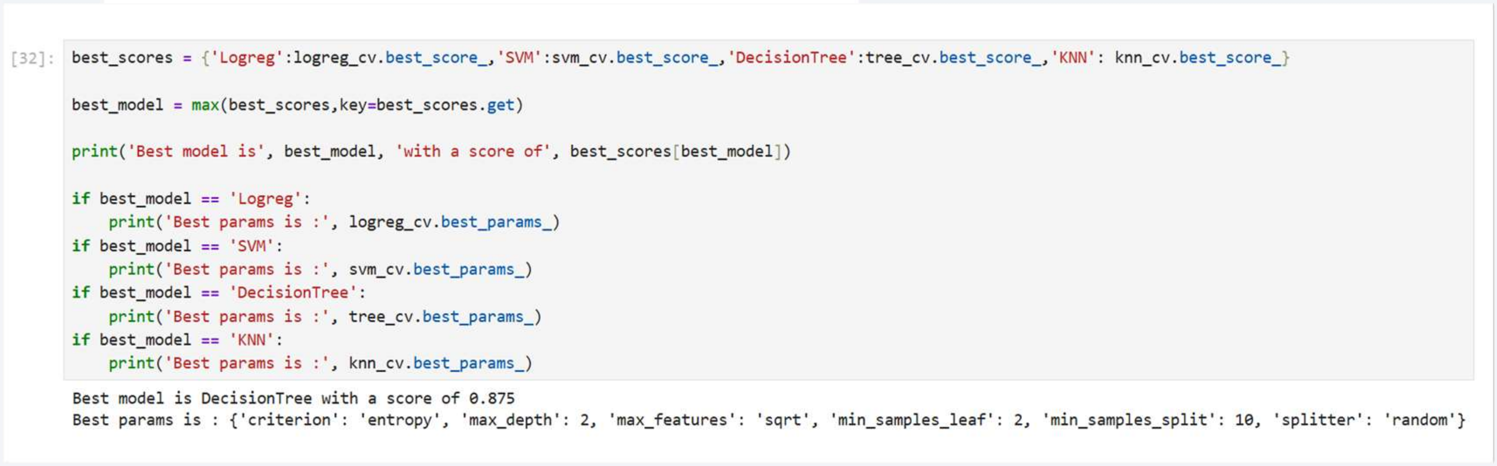
- Payloads between 2,000 and 6,000 have the most success rates

Section 5

Predictive Analysis (Classification)

Classification Accuracy



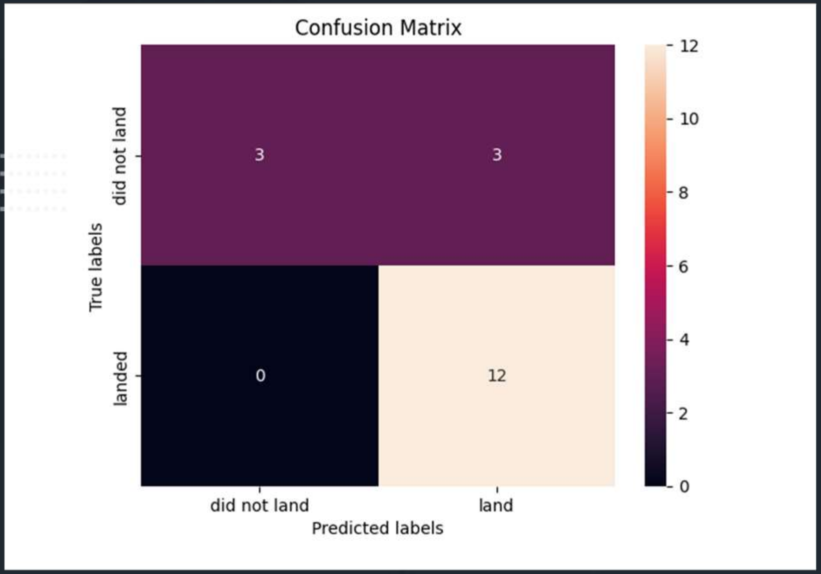


```
[32]: best_scores = {'Logreg':logreg_cv.best_score_,'SVM':svm_cv.best_score_,'DecisionTree':tree_cv.best_score_,'KNN': knn_cv.best_score_}

      best_model = max(best_scores,key=best_scores.get)

      print('Best model is', best_model, 'with a score of', best_scores[best_model])

      if best_model == 'Logreg':
          print('Best params is :', logreg_cv.best_params_)
      if best_model == 'SVM':
          print('Best params is :', svm_cv.best_params_)
      if best_model == 'DecisionTree':
          print('Best params is :', tree_cv.best_params_)
      if best_model == 'KNN':
          print('Best params is :', knn_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.875
Best params is : {'criterion': 'entropy', 'max_depth': 2, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'random'}
```

- The Decision tree has the highest classification accuracy

Confusion Matrix



The confusion for the decision tree can distinguish between the various classes, however there are still false positives

Conclusions

- The Decision tree has the highest classification accuracy
- Launches that had payloads between 2,000 and 6,000 had more success
- The launch sites are mainly along the coast
- KSC LC-39A has the most successful launches
- The success rate increased over time

Thank you!