IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Author: Nick Andersen
Date:07/04/24

Outline

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

Executive Summary

- Summary of methodologies

Interactive dashboards

API and Webscraping Data collection

Data Analytics SQL

Model prediction with machine learning

Data Visualization

Folium Interactive mapping

- Summary of all results

Predicitive Model Analysis

Preliminary data analysis

Visuals with interactive capabilities

Introduction

- Project background and context

SpaceX shows Falcon 9 launches on their website. You can see the discrepancy in their cost with competitors with SpaceX only spending 62 million dollars, almost 100 million less than other providers. This was due to the reuse of the first stage of the launch. If we know if the first stage will land, we can determine the cost of the launch. This can make SpaceX very competitive against other bids against them.

- Problems you want to find answers

How can we determine the success rate of the landing?

What can we do to make a successful landing assured or likely?

How can we tell if the a launch was successful?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was gathered using webscraping from the website “Wikipedia”
- Perform data wrangling
  - Used One-Hot for 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
  - Classification model development and validation

Data Collection

- Describe how data sets were collected.

The data was collected by SpaceX API

Decoded the response using the json() function and converted it into pandas dataframe using the .json_normalize() function.

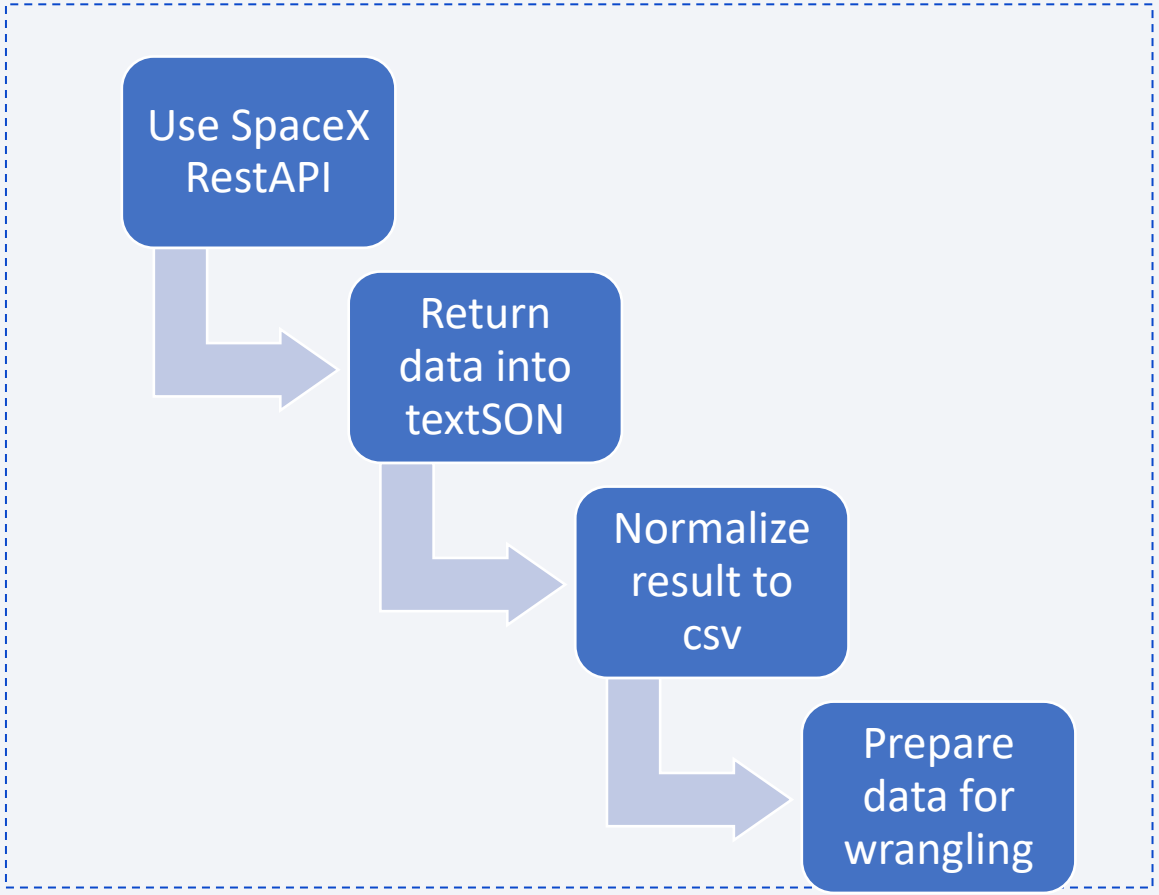
Performed data cleaning process of all missing values and zeros

Webscraping of SpaceX for Falcon 9 from Wikipedia

Data Collection – SpaceX API

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

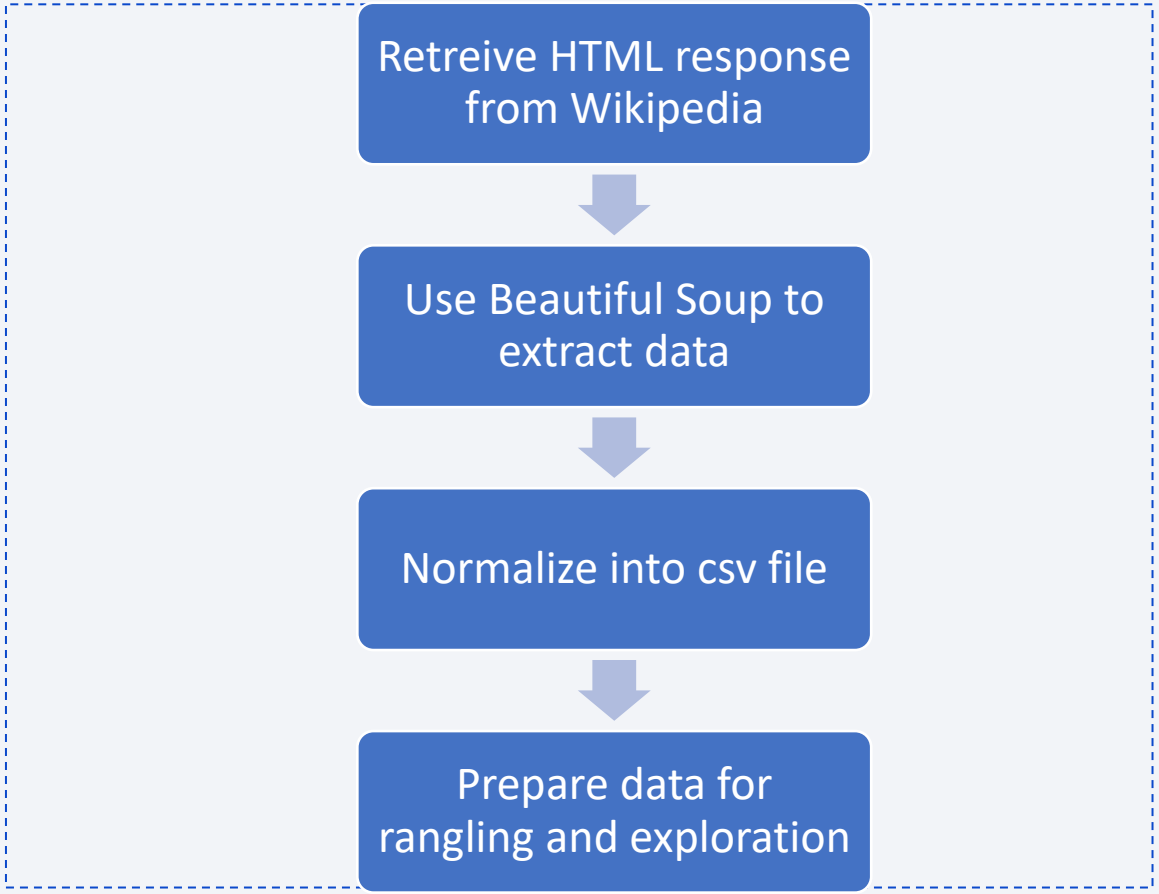
- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose



Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose



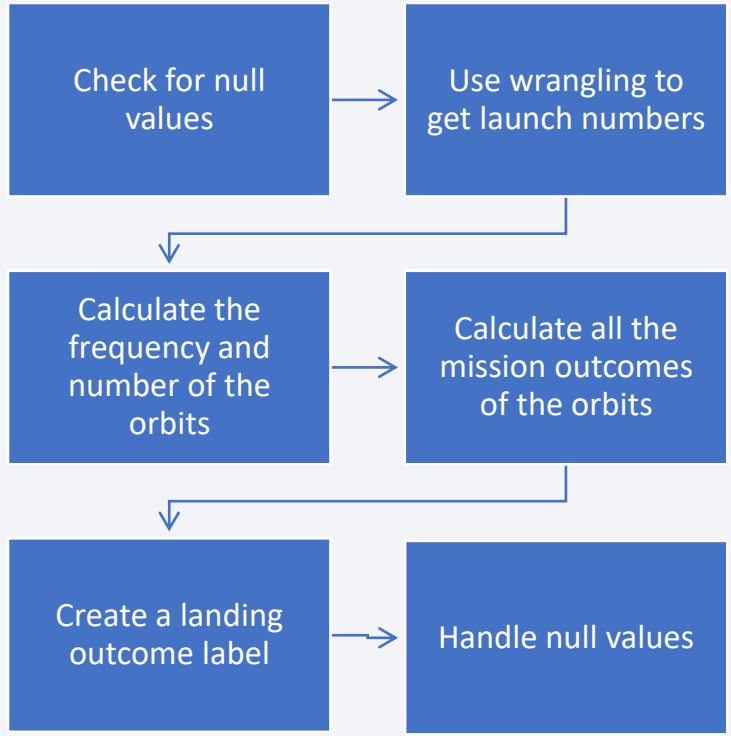
Data Wrangling

Performed EDA, looking for patterns in data and create labels

Many different outcomes were populated which were Either labeled a 1 for a successful landing or a 0 for a failed One.

Label Meanings

True Ocean = Successful landing in the ocean
False Ocean = Failed landing in ocean
RTLS = Successful landing on ground pad
False RTLS = Failed landing on ground pad
True ASDS = Successful landing on drone ship



EDA with Data Visualization

- EDA was used to make various plots to analyze the data set

- Scatter plots was an easy way to see the relationship between two variables. It was used to show the relationship between flight number and launch site, payload and launch site, flight number and orbit type, and payload and orbit type.
- Bar charts more illustrated multiple groups and made it clear which groups performed the best. Bar chart was used to visualize the success rate between each orbit type.
- Line charts were used to illustrate the changes of a single quantity of a period of time. This was used to show the average launch success over a year.

EDA with SQL

- SQL Query List

- Gives total payload mass from boosters by NASA
- Gives average payload mass carried by F9 v 1.1
- List successful drone ship landings
- List of successful boosters with payload between 4000 and 6000
- Names of booster_versions with maximum payload mass
- 5 records where launch sites began with ‘KSC’
- Names of the unique launch sites in space mission
- Records which display month names, successful landings in ground pad, booster versions, and launch sites for 2017
- Ranking amount of successful landing outcomes between 06/04/10 and 03/20/17

Build an Interactive Map with Folium

- Using interactive maps with Folium helps us assess geospacial data allowing for more interactive visuals. This allows us to comprehend factors like launch proximity sites that affect launch success rate.
- Some examples would be..
- Mark all launch sites and highlight them with folium.circle and folium.markerwhich allowed the user to see the sites on the interactive map.
- Calculate distance between launch sites to important landmarks such as cities and railroads
- folium.Marker() to calculate the distance between landmarks on the map
- MousePosition() to show gps coordinates on map where cursor is.
- folium.Polyline() draw line between a point and a launch site
- Questions answered
- Are launch sites in close proximity to railways? Yes
- Are launch sites close to highways? Yes
- Are launch sites outside a certain perimeter from cities? Yes
- Are launch sites close to coastlines? Yes

Build a Dashboard with Plotly Dash

- Dash was used to create an interactive visual analysis on the spaceX launch data in real time.
- Created a dropdown menu that can filter the dashboard by all launch sites or a particular site.
- Created a pie chart referencing successful/failed launches for all sites or a particular site.
- Added a slider to filter the launches by payload range
- Scatter chart was added to visualize correlations between successful or failed launches for selected sites. Color label booster
- Questions answered

- Site with the most launches: KSC LC-39A with 10 launches
- Payload range of site with highest launch success rate: 2000-5000 kg
- Payload with the lowest success rate: 0-2000 and 5500-7000

Highest launch success rate: KSC LC-39A

F9 Booster with the highest lauch success rate: FT

Predictive Analysis

Read SpaceX into dataframe and create a class array → Standardize the data and transform → Train/Test/Split data into training and test data sets → Make models and optimize them → Find model that works best

Read SpaceX into dataframe:

Load the dataframe

Load the data

```
from js import fetch
import io

URL1 = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_2.csv"
resp1 = await fetch(URL1)
text1 = io.BytesIO((await resp1.arrayBuffer()).to_py())
data = pd.read_csv(text1)
```

Standardize the data and transform

```
1, 1], dtype=int64)
```

TASK 2

Standardize the data in `X` then reassign it to the variable `X` using the transform provided below.

```
# students get this
transform = preprocessing.StandardScaler()
```

```
X = transform.fit_transform(X)
```

We split the data into training and testing data using the function `train test split` . The training data is divided into validation data, a second set used for

Train/test/split data into training and test sets

and test data should be assigned to the following labels.

`X_train, X_test, Y_train, Y_test`

```
X_train, X_test, Y_train, Y_test = train_test_split( X, Y, test_size=0.2, random_state=2)
print ('Train set:', X_train.shape,  Y_train.shape)
print ('Test set:', X_test.shape,  Y_test.shape)
```

```
Train set: (72, 83) (72,)
Test set: (18, 83) (18,)
```

```
Y_test.shape
```

Logistic regression object and set train dataset Into GridSearchCV object

TASK 4

Create a logistic regression object then create a GridSearchCV obje `parameters` .

```
parameters ={'C':[0.01,0.1,1],
             'penalty':['l2'],
             'solver':['lbfgs']}
```

```
lr=LogisticRegression()
```

```
logreg_cv = GridSearchCV(lr,parameters,cv=10)
logreg_cv.fit(X_train, Y_train)
```

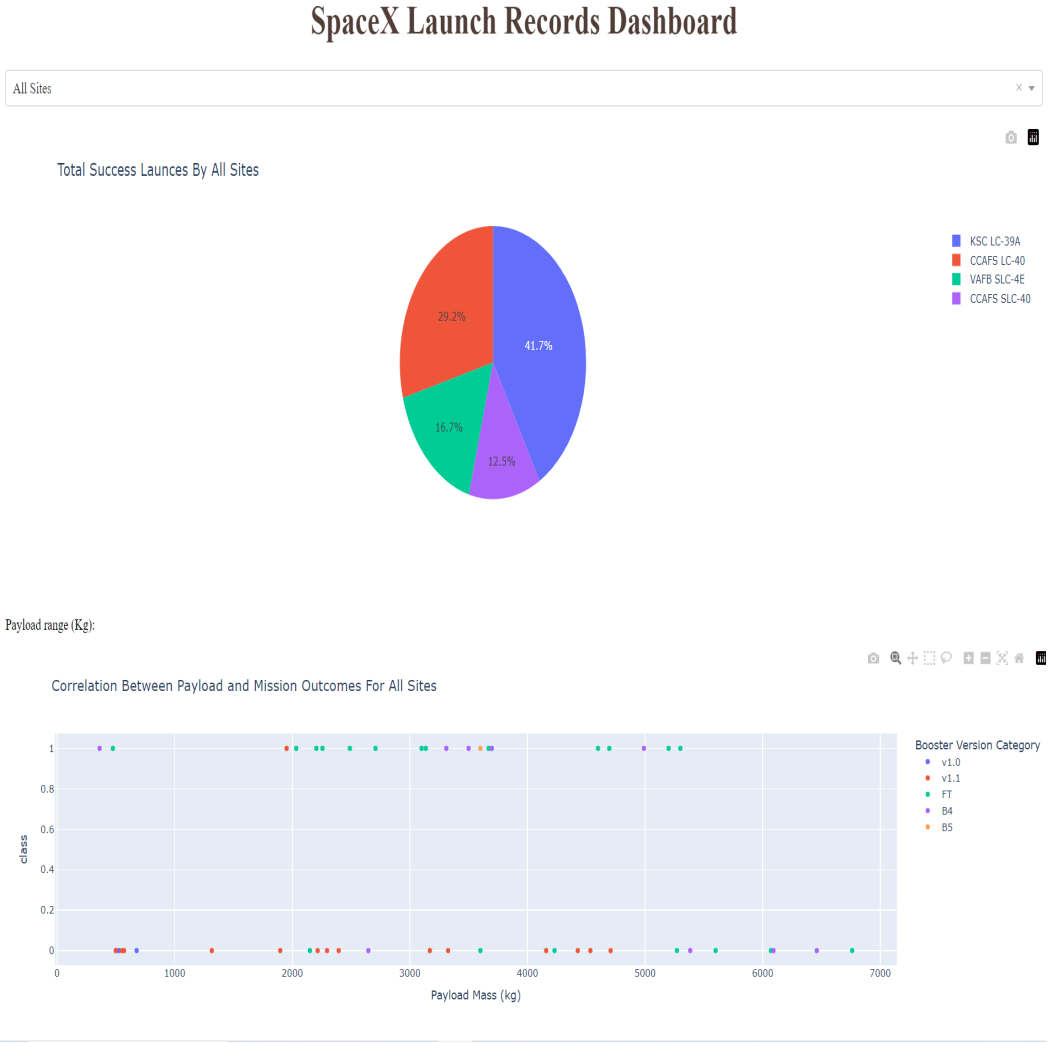
GridSearchCV
estimator: LogisticRegression
LogisticRegression

Search for hyperparameters and print them

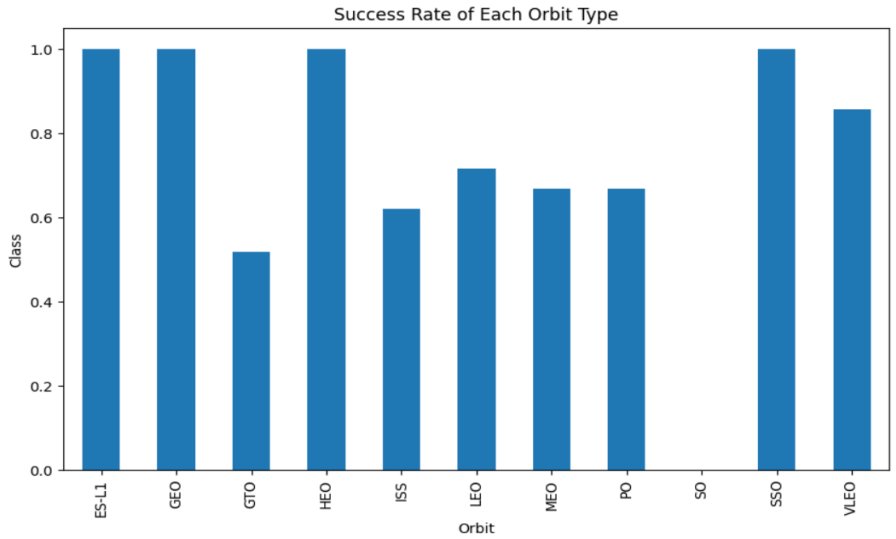
```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

Create confucsion matrix and check For data accuracy.

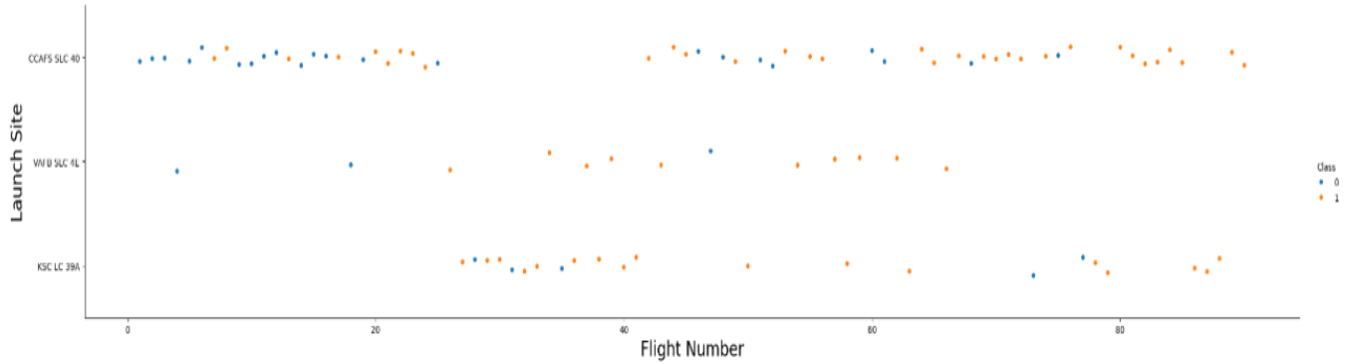
Results:



Ploty Dashboard results:




```
[5]: ### TASK 1: Visualize the relationship between Flight Number and Launch Site
     # 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()
```



Exploratory data analysis results:

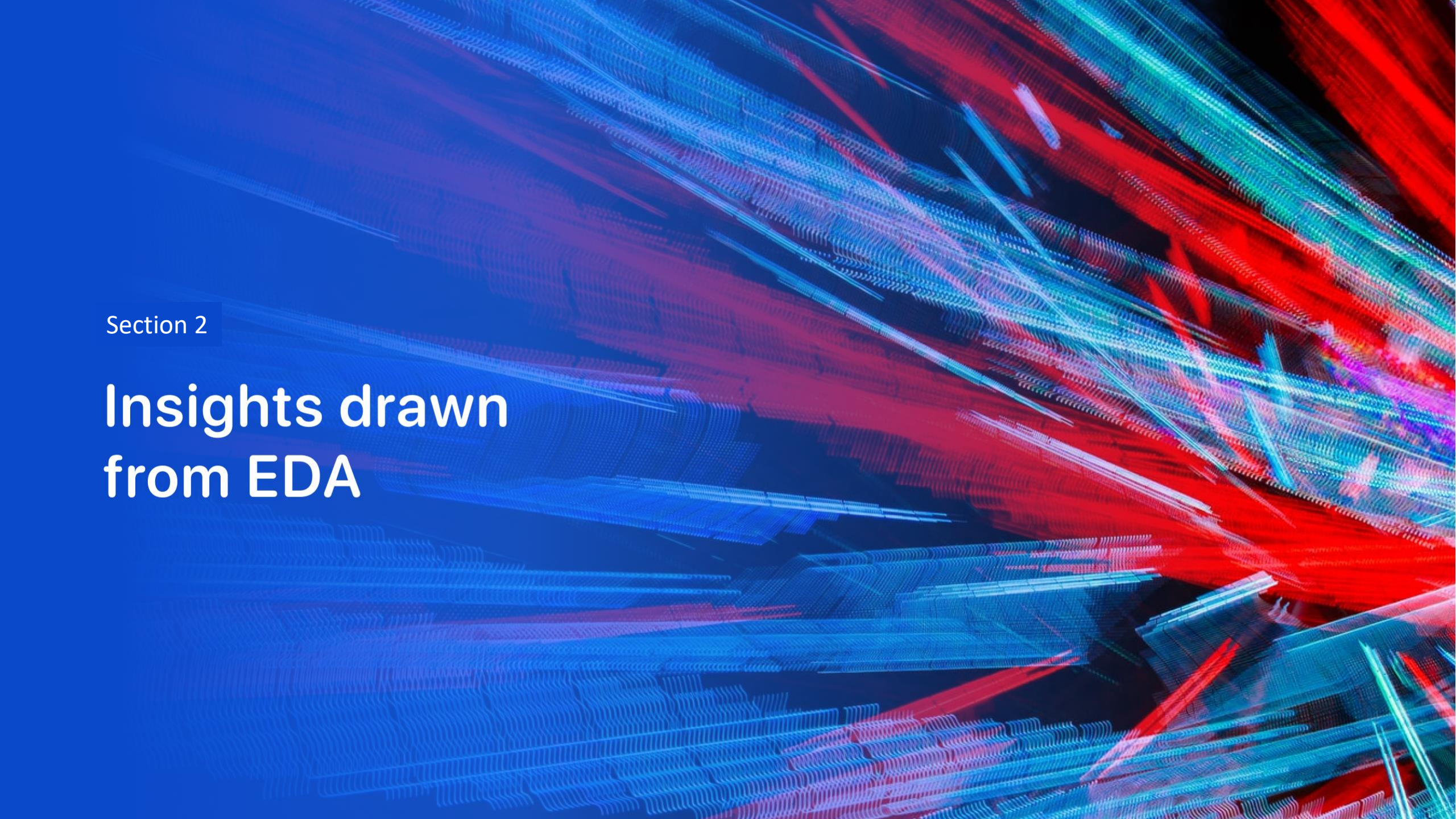
Results Continued

```
]: Model_Performance_df
```

]:

	Algo Type	Accuracy Score	Test Data Accuracy Score
2	Decision Tree	0.862500	0.833333
3	KNN	0.848214	0.833333
1	SVM	0.848214	0.833333
0	Logistic Regression	0.846429	0.833333

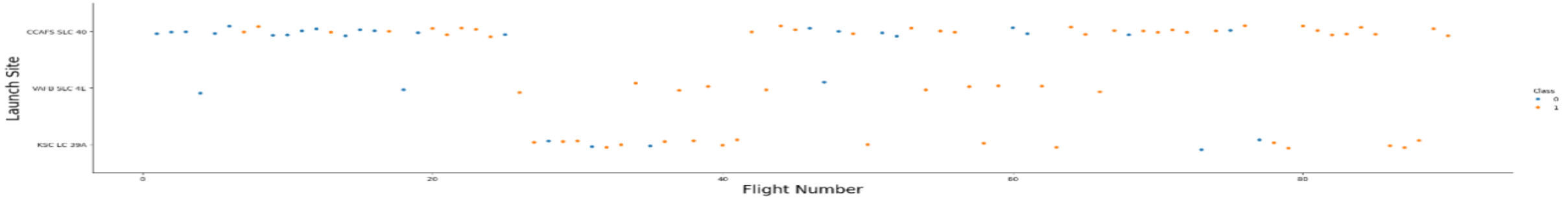
Predictive Analysis Results


Section 2
Insights drawn from EDA

Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type

[5]:
```
### TASK 1: Visualize the relationship between Flight Number and Launch Site
# 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()
```

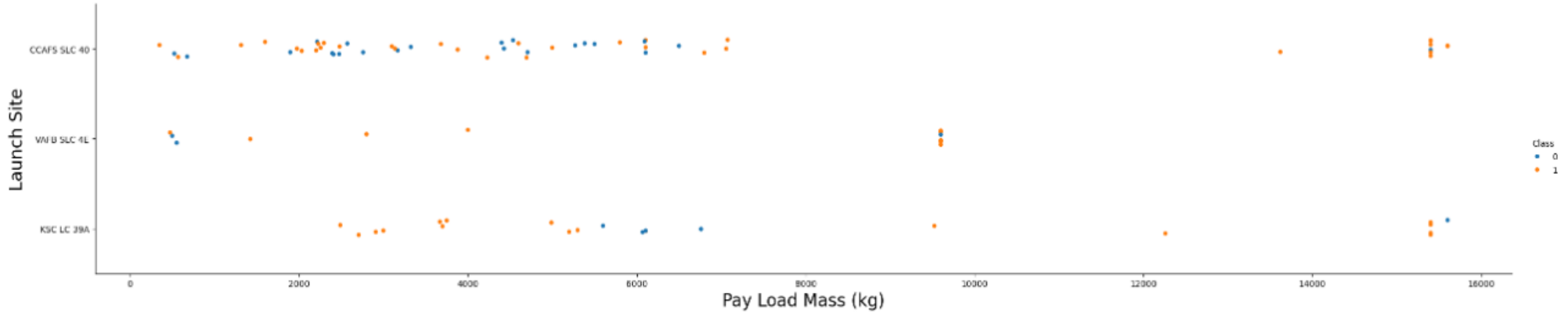


- The launches that increase in success rate correlate to the increased number of flights.
- After the threshold of 20 launches was broke, there were far more successful launches.

Payload vs. Launch Site

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

[7]:
```
### TASK 2: Visualize the relationship between Payload and Launch Site
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay Load Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



We also want to observe if there is any relationship between launch sites and their payload mass.

VAFB SLC 4E didn't have any launches over a payload over 10000kg

As payload increased the success rate of the launch increased as well

Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type

```
]: %sql select Booster_Version from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
 * sqlite:///my_data1.db
Done.
```

]:

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

- Show the screenshot of the scatter plot with explanations: 100 successful launches

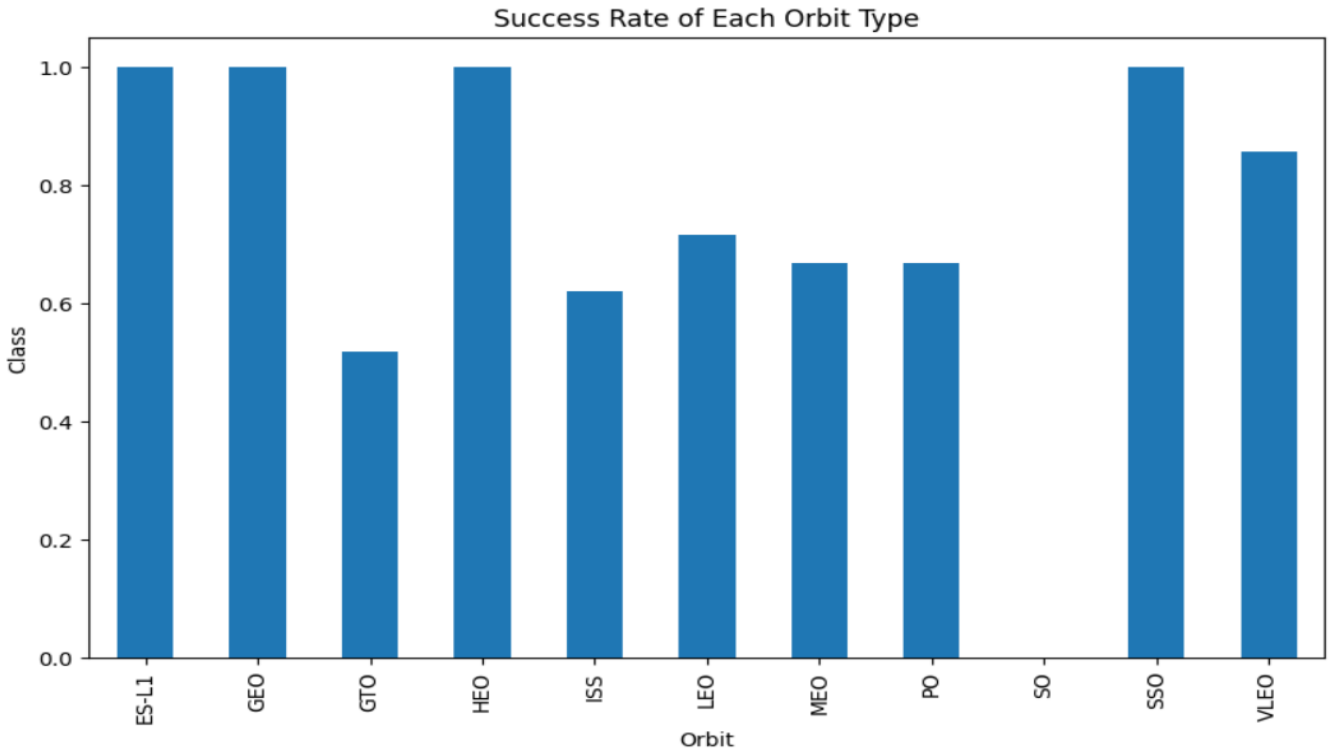
List the total number of successful and failure mission outcomes

```
25]: %sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
 * sqlite:///my_data1.db
Done.
```

25]:

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

Success Rate vs Orbit Type

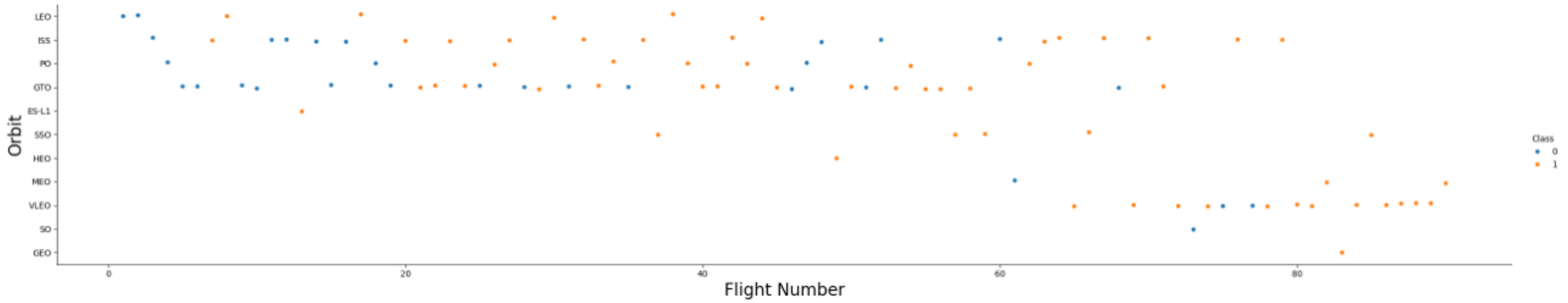


The highest success rates
Were for ES-L-1, GEO, HEO, and SSO.

The lowest success rate was SO

Flight number vs Orbit Type

[12]:
```
### TASK  4: Visualize the relationship between FlightNumber and Orbit type
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



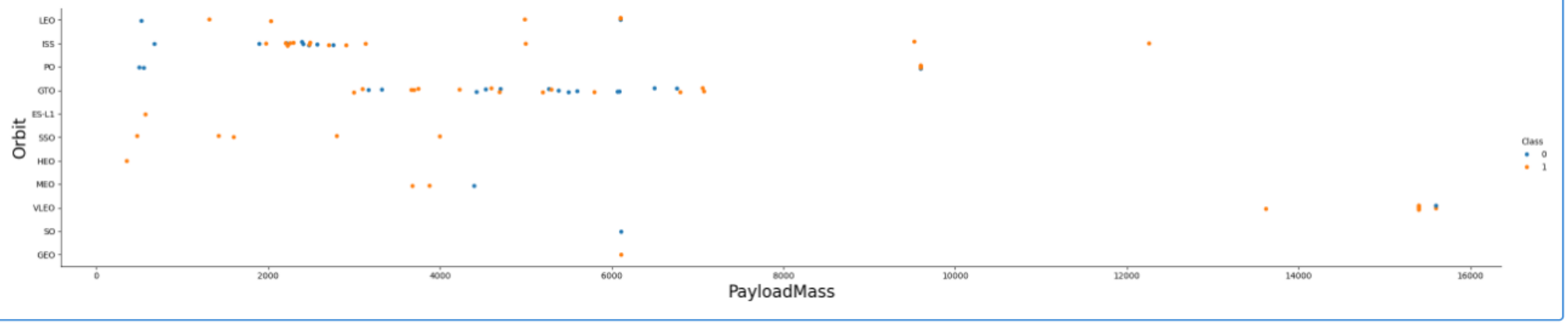
ES-L1, SSO, HEO, and GEO had only successful landings

After 80 launches there were only successful flights

GTO seemed the most inconsistent and was constantly oscillating between success and failure

Payload vs Orbit Type

```
plt.show()
```



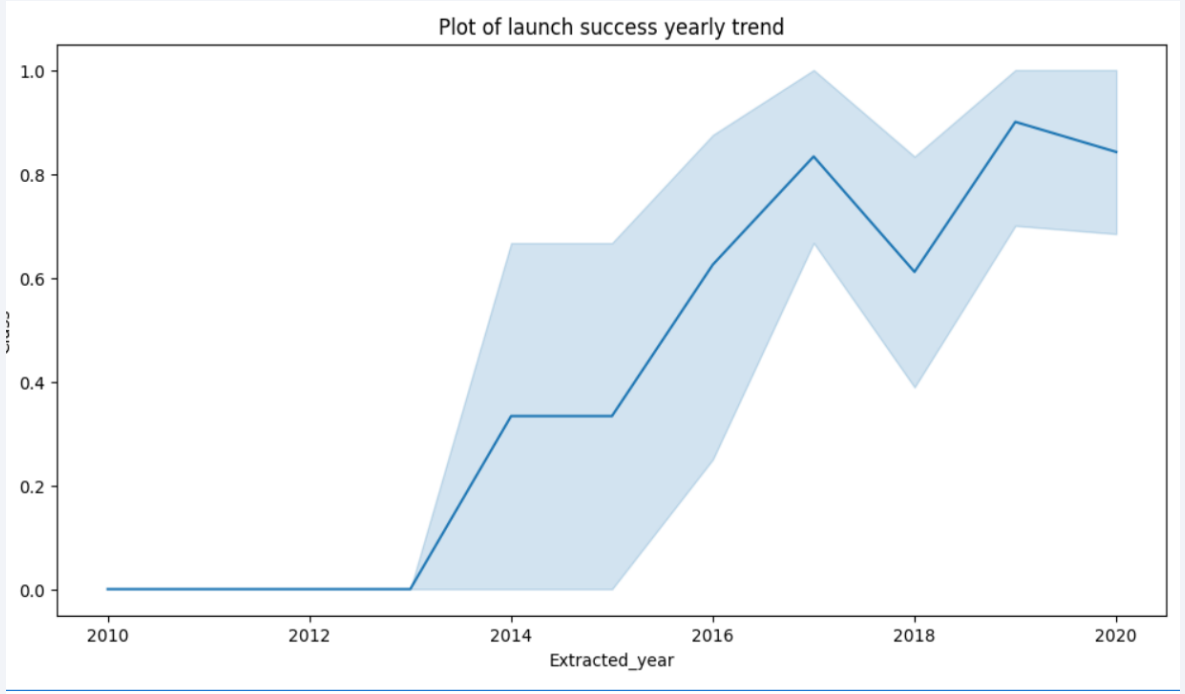
Similarly, we can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type

The higher the payload the higher the rate of success for mainly LEO,ISS,PO,SSO. It seems that once the payload exceeds 5000 kg, there were almost no failed launches.

GTO has no clear pattern

All the others have an insufficient amount of launches to conclude anything.

Launch Success Yearly Trend



From the span of 2013 to 2016 the success rate greatly increased to about 80%

First success was in 2013

Rates began to fall between 2017-2018

All Launch Site Names

- Provided below is the query and the outcome of the site names,

```
[17]: %sql select distinct Launch_Site from SPACEXTABLE
 * sqlite:///my_data1.db
Done.
```

[17]:

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

Launch Site Names Begin with 'CCA'

Task 2

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

[18]:
```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5
```

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

[18]:

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

Would you like to receive official Jupyter

Payload vs. Launch Site

- The total payload mass for the launch was

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

```
[19]: %sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
 * sqlite:///my_data1.db
Done.
```

[19]:

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

- The average payload mass carried by booster version f9 v 1.1

45596 NASA (CRS)

Task 4

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

```
[20]: %sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
 * sqlite:///my_data1.db
Done.
```

[20]:

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

- Date of first successful landing

Hint:Use min function

```
[10]: %sql select min(Date) from SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)'
 * sqlite:///my_data1.db
Done.
```

[10]:

min(Date)
2015-12-22

Total Payload Mass

Task 3

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

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

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

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

Use sum to get the total of a column, the total mass. Total mass was 45596. Only returns Sum for customers with name ‘NASA (CRS)’.

Average Payload Mass by F9 v1.1

Task 4

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

Use the AVG() function to return the average of the column where the booster_version is ‘F9 v1.1’

Average 2534.67

First Successful Ground Landing Date

```
[10]: %sql select min(Date) from SPACEXTABLE WHERE Landing_Outcome = 'Success (ground pad)'
 * sqlite:///my_data1.db
Done.
[10]:  min(Date)
       2015-12-22
```

Use min(Date) function to get the earliest date in the set. Filter by making the condition ‘Success (ground pad)’ to

Make the earliest date of a successful ground pad landing

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

[24]:
```
%sql select Booster_Version from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_
```

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

[24]:

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

Use the operator and to create two conditions to be met to be displayed. Greater than 4000 And less than 6000.

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

[25]:
```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

[25]:

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

Using the group by operator to group data of the same condition into one catagory. 100 successes And 1 failure.

Boosters Carried Maximum Payload

Task 8

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

```
[26]: %sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SP
```

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

[26]:

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

Query uses the max function to get the maximum payload. Returns the boosters with their Payload mass max. They are all maxing at 15,600.

2015 Launch Records

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site

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

[25]:
```
%sql select substr(Date, 6,2) as month, Landing_Outcome, Booster_Version, Launch_Site from SPACEXTABLE\
where Landing_Outcome = 'Failure (drone ship)' and substr(Date,0,5)='2015'
```

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

[25]:

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

Uses substr() to get the months from the column since month names are not supported. Conditions Where the landing was a failure as well.

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 descer order.

[28]:
```
%sql select Landing_Outcome, count(1) as total_landing_outcome from SPACEXTABLE\
where Date between '2010-06-04' and '2017-03-20' group by Landing_Outcome order by count(1) desc
```

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

[28]:

Landing_Outcome	total_landing_outcome
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

Reference Links

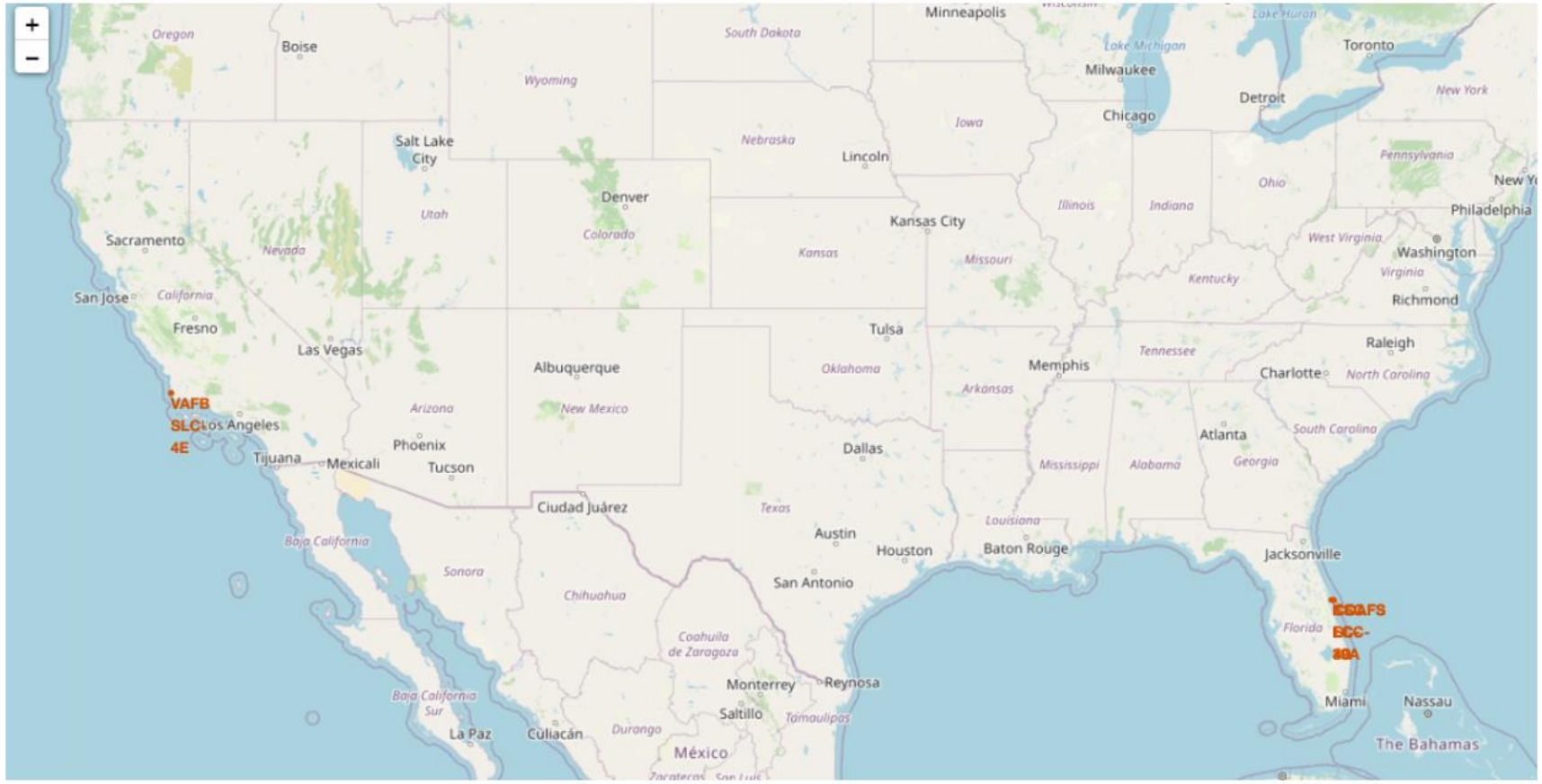
Uses order by count desc to rank to outcomes. Uses where statement to provide landings between Two certain dates.

Section 3

Launch Sites Proximities Analysis

Folium Launch site map

The generated map with marked launch sites should look similar to the following:

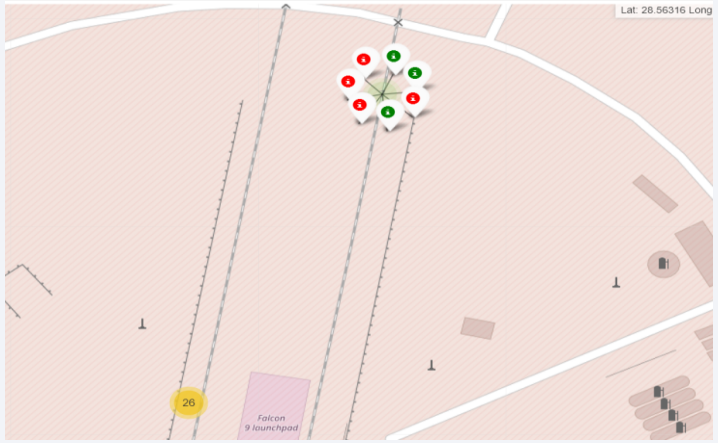


Map with Launch sites

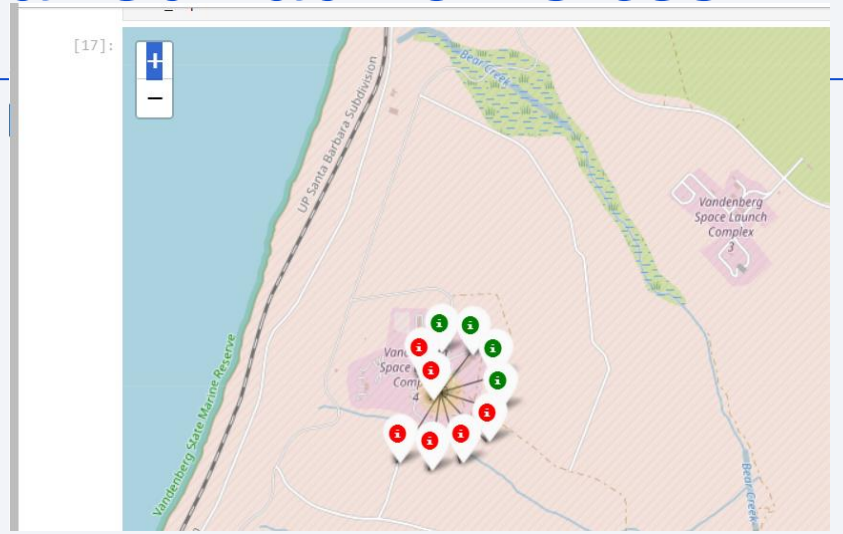
Now, you can explore the map by zoom-in/out the marked areas, and try to answer the following questions:

de) | Idle Mode: Command Ln 1, Col 1 lab_jupyter_launch_site_location.ipynb

Successful and failed launch sites map

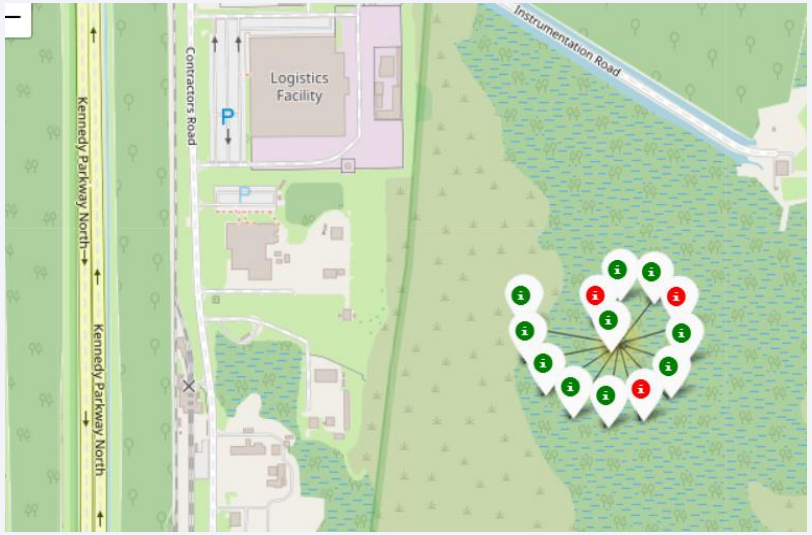


CCAFS SLC 40 LAUNCH SITE

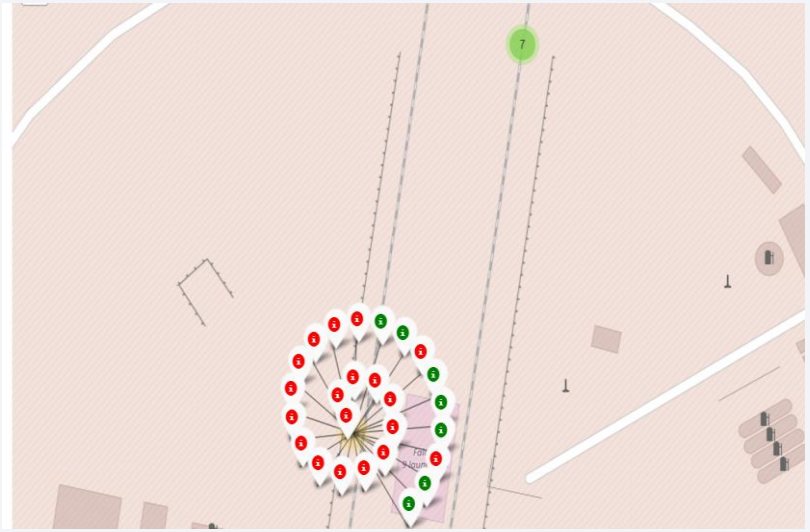


VAFB LAUNCH SITE

Success and fail
Markers where green
Is a success and red is
A failed launch.



KSC LC-39A LAUNCH SITE



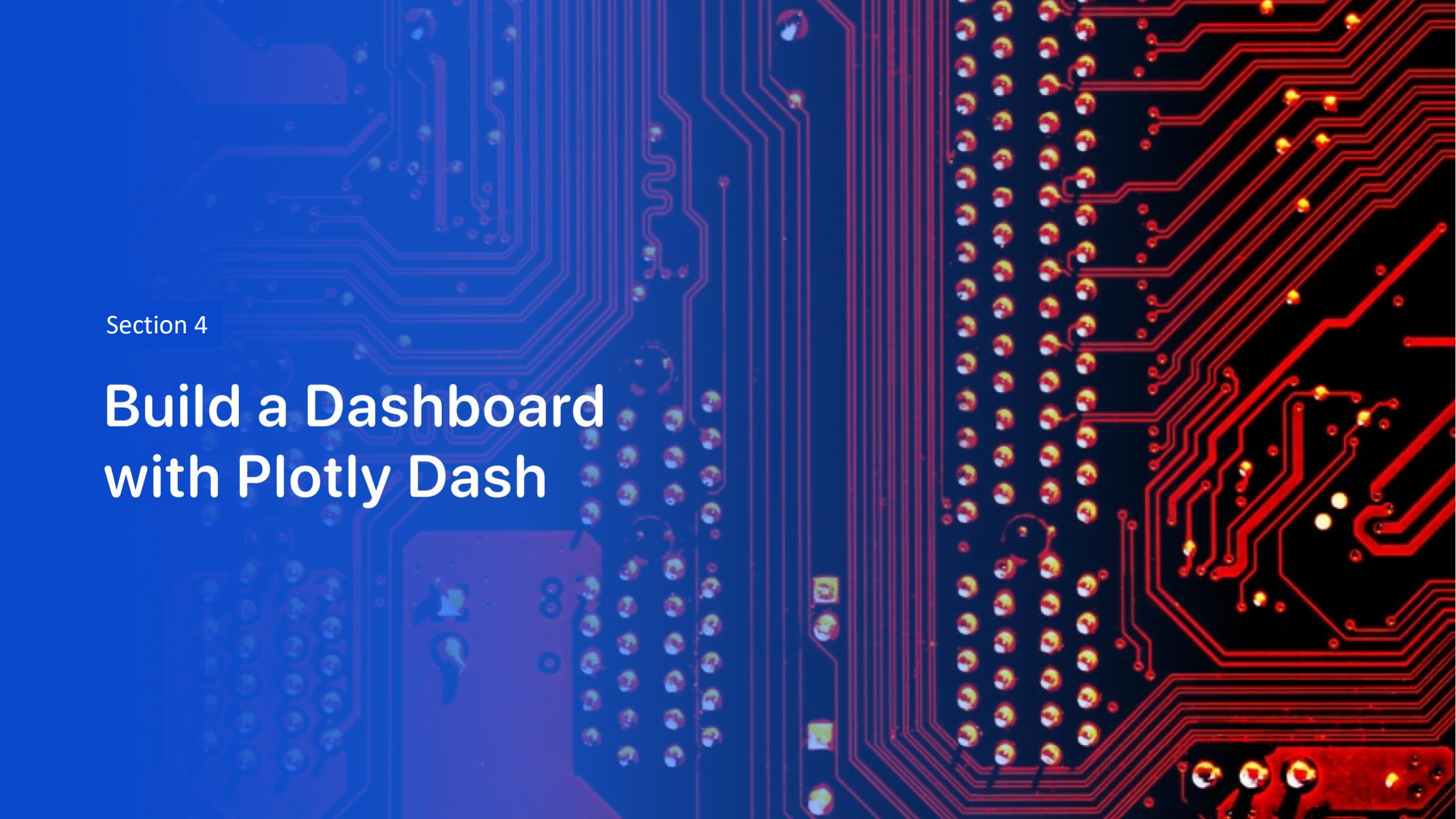
CCAFS SLC-40 LAUNCH SITE

Proximity Map

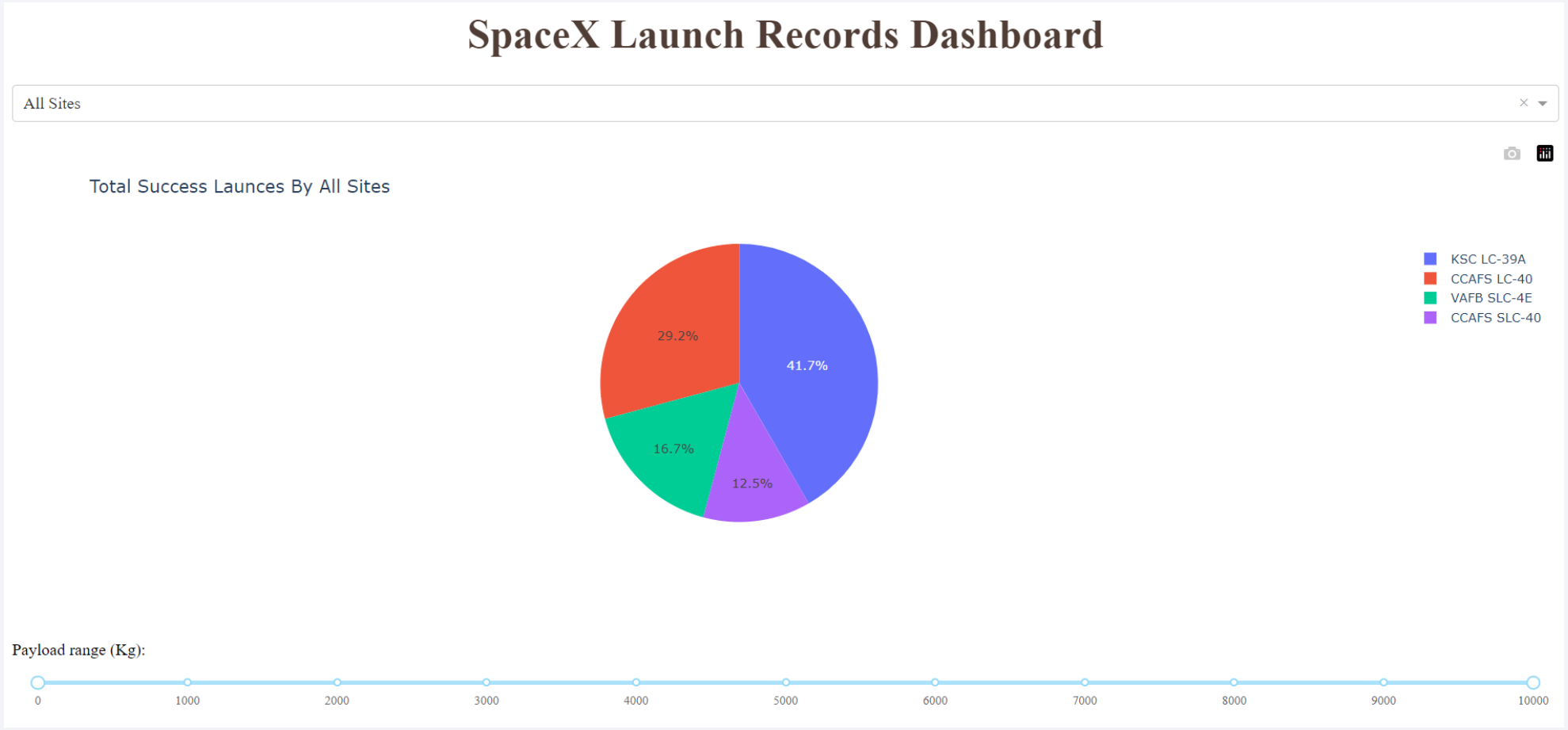
Picture displays the distance
from proximity sites such as the coastline
Railroads and highways.

This puts on display that launch sites
Are typically further away from cities just
In case there is a failure and debree becomes
A hazard for the surrounding area.

In order to quicken the projects progress there
Is a need for resources so the sites have been
Placed near railroads, coastlines, and highways
So those resources can be transported to the site
Very easily.

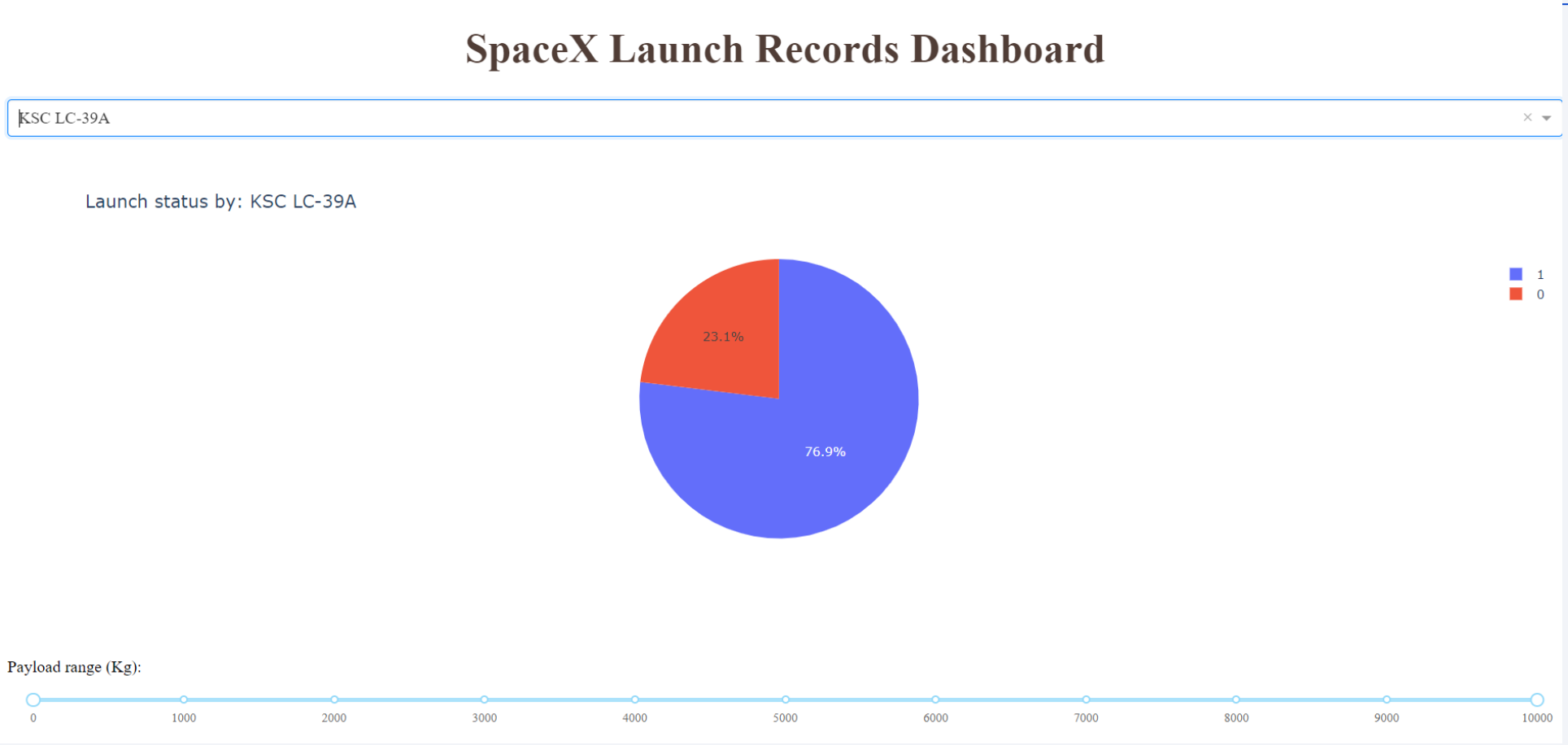

Section 4
Build a Dashboard with Plotly Dash

Launch success rate dashboard



KSC-LC-39A has the highest success rate with 41.7%
CCAFS LC-40 has the second highest with 29.2%'
VAFB SLC-4E has the third highest with 16.7%
Lowest is CCAFS SLC-40 with 12.5%

KSC LC-39A Dashboard



Highest success rate KSC LC-39A only displayed. 0 is a successful launch and 1 is a failed one. 76.9% success rate.

<Dashboard Screenshot 3>

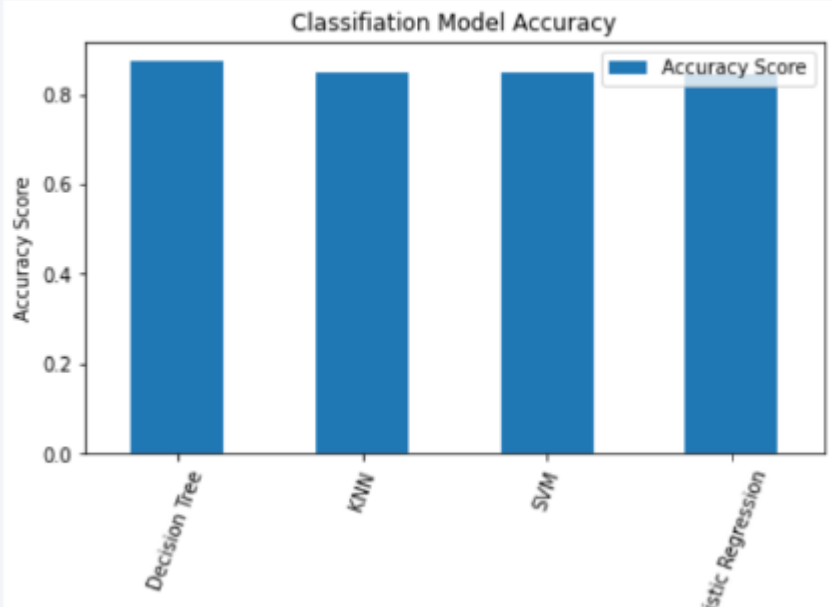
Best period of launches ranged from 2000kg to 5500kg. Booster FT had the most successful launches.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- The decision tree had the highest accuracy score with 87.5% accuracy.
- Test data was the same for all of the models at 83.333
- The accuracy scores are very close to the point that we need other variables to determine which one of the models is more effective.



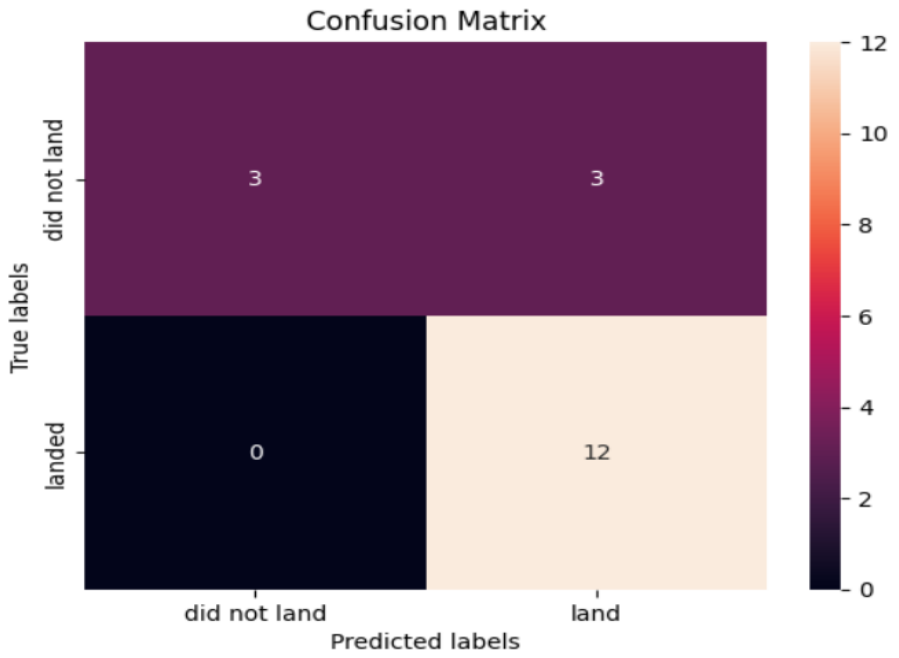

```
[49]: Model_Performance_df
```

[49]:

	Algo Type	Accuracy Score	Test Data Accuracy Score
2	Decision Tree	0.862500	0.833333
3	KNN	0.848214	0.833333
1	SVM	0.848214	0.833333
0	Logistic Regression	0.846429	0.833333

Confusion Matrix

```
[41]: yhat = tree_cv.predict(X_test)
      plot_confusion_matrix(Y_test,yhat)
```



18 total predictions were made.

3 were predicted to not land and Didn't land. (True negative)
3 were predicted to not land and Landed. (False Positive)
12 were predicted to land and Landed (True positive)
0 were predicted to land and Did not land.

(True positive + True negative)/ Total

16.5% error rate.

Conclusions

- While trying to see if the first stage of the Falcon 9 would launch we determined that the more flights a site has, the more likely the first launch will be successful
- The site with the best launch success was KSC LC-39A with a success rate while the site with the lowest was CCFAS SLC-40.
- Launches above a threshold of 7000kg are more likely to succeed
- The best method is the decision tree classifier to predict results more accurately but only marginally compared to other methods.

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!