IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Joseph Maya
29/11/2024

Outline

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

- All files are in Github in this url:
- GingerBeer12/SpaceX-Project

Executive Summary

- Data visualization with Matplotlib, Seaborn and Plotly were implemented.
- For Data Wrangling I used Pandas, numpy any many other Libraries.
- For data Prediction I used KNN, SVM, Decision Tree Classifier and Logistic Regression.
- Summary of all results:
- Model Logistic Regression: was: 88%
- Support Vector Machine was: 83.2%
- Decision Tree Classifier was : 94.44%
- K-nearest neighbors was: 83.2%

Introduction

- Space X Data Project

- This project extract information through Data analysis for better understand operation of Space-X
- Finally it will conclude what are the chances for Space-X to be successful in launching craft into Space and Land the Launcher successfully for reuse.

Section 1

Methodology

Methodology

Executive Summary

- Data was collected by the use of Space X rest API.
- Data was processed in Python EDA and downloaded as json file, later to be transformed into a python file.

- Performed exploratory data analysis (EDA) using visualization in matplotlib and using SQL to answer data related questions.
- Performed interactive visual analytics using Folium and Plotly Dash
- Performed predictive analysis using classification models : build, tune and evaluate classification models.
- I used Four Classification models in Machine learning :

Logistic Regression, SVM (support Vector Machine) , KNN(K nearest neighbors) and Decision-Tree Classifier, by facilitating Gridsearch CV to tune up the parameters.

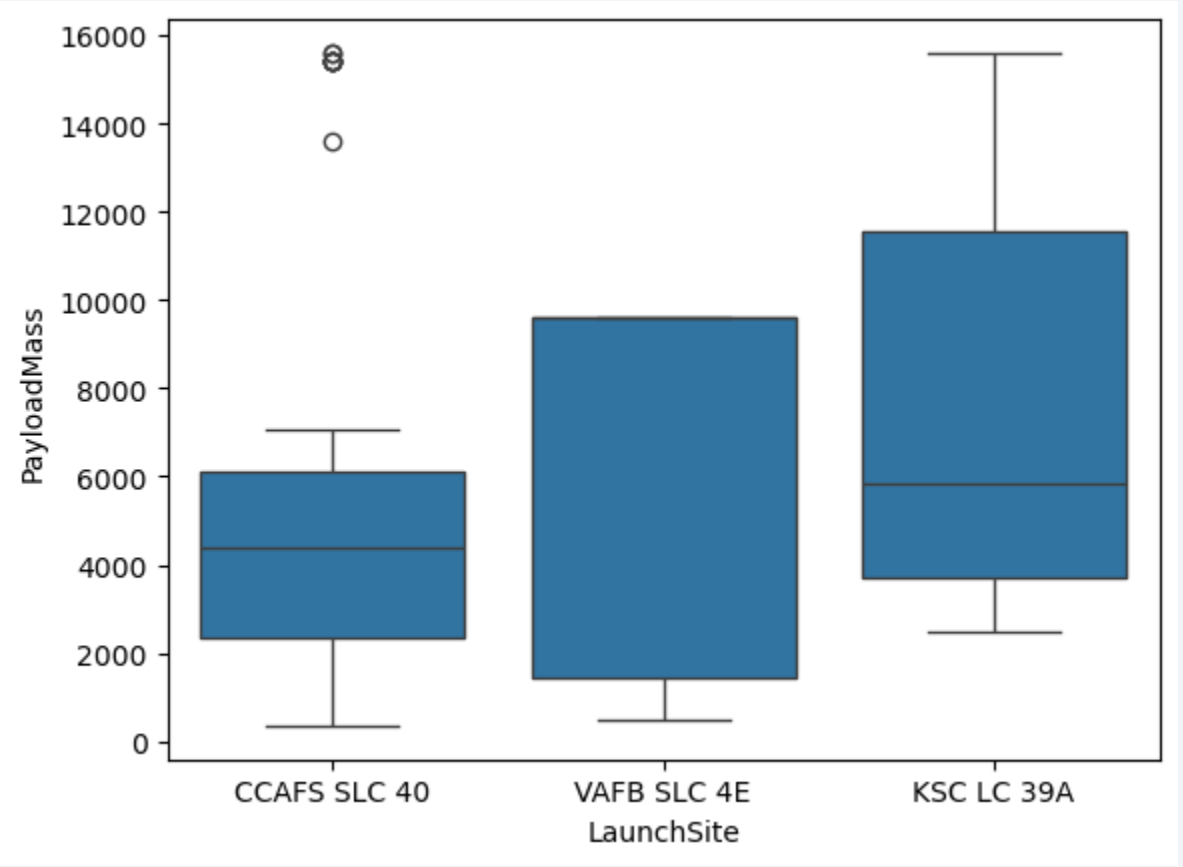
Data Collection

- Data was collected as follows:
- Stages:
- Imported libraries : Pandas, Requests, Numpy and datetime
- 1. response = requests.get(static_url)
- 2. DataFrame = pd.json_normalize(response.json())

Data Collection – SpaceX API

- Also used is BeutifulSoup Library
- response = requests.get(static_url)
- soup = BeautifulSoup(response.text, 'html.parser’)

- In the Box plot we can see Example of the success rate of 3 launch sites with different PayLoadMass



Data Collection - Scraping

- Data collection flow chart can be seen in the files in Github URL:
- GingerBeer12/SpaceX-Project

Data Wrangling

- Describe how data were processed:
- 1. find null values by df.info(), and df.isnull().sum()/len(df)*100
- 2. find out the types of the df columns by :df.dtypes
- 3. used the average of the of selected columns to replace Nones
- 4. Delete rows where necessary. (and not effecting the Results)
- 4. Use dummies to convert success and failure to 0 and 1.
- 5. drill into the statistics of the data by: df.describe() method.

EDA with Data Visualization

Plot for the analysis were made by the use of Matplotlib and Seaborn:

- Box Plot
- Bar plots
- Scatter plots and Regression
- Confusion Metrics
- Line plots

EDA with SQL

Few of the SQL Statements used for analysis:

- %sql select distinct Launch_Site from SPACEXTBL
- %%sql select distinct Launch_Sitefrom SPACEXTBL

 where Launch_Site like 'CCA%’

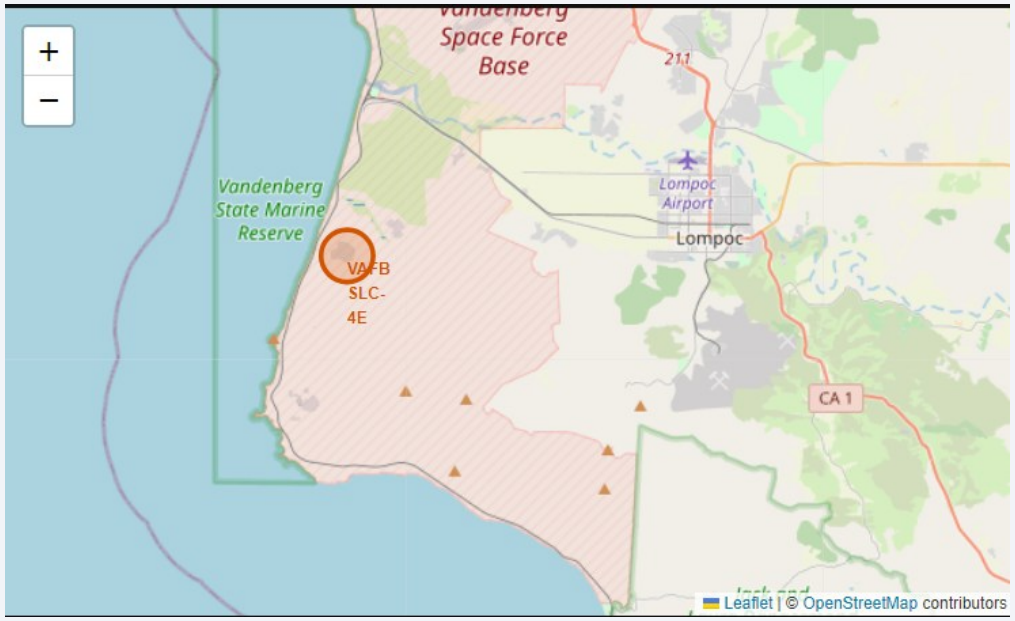
- %sql select count(PAYLOAD_MASS__KG_) from SPACEXTBL
- %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL

where Booster_Version like 'F9 v1.1%'

- %%sql SELECT DISTINCT Booster_VersionFROM SPACEXTBLWHERE Landing_Outcome = 'Success (drone ship)'AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
- %sql select count(*), Mission_Outcome from SPACEXTBL group by Mission_Outcome;

Build an Interactive Map with Folium

- site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
- Markers were made to see the launch site Locations in the United States.



Build a Dashboard with Plotly Dash

- I used Dash to create Dashboard.
- The Dashboard include a drop down list of Sites
- payload visualization, Pie chart and Scatter plot.

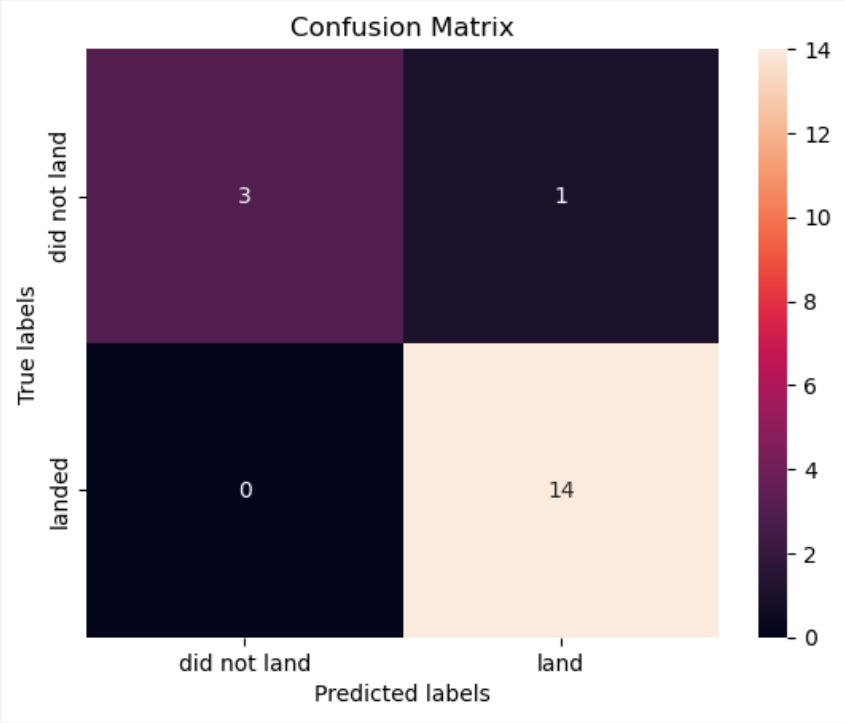
The plots made to assess success rate of launches in different locations and Payloads.

Predictive Analysis (Classification)

- I used Machine Learning methods to get the best Prediction outcome:
- The method that were used are: Logistic Regression, SVM and KNN.
- First I made a train/test split data using sklearn preprocessing and
- sklearn.model_selection - train_test_split
- I used model object to train the Data, and apply GridSearchCV.
- I then used Pipelines and fitted the models,
- Prediction were made for each model.
- Get scores: d-Square, MSE, MAE and Confusion Matrics.

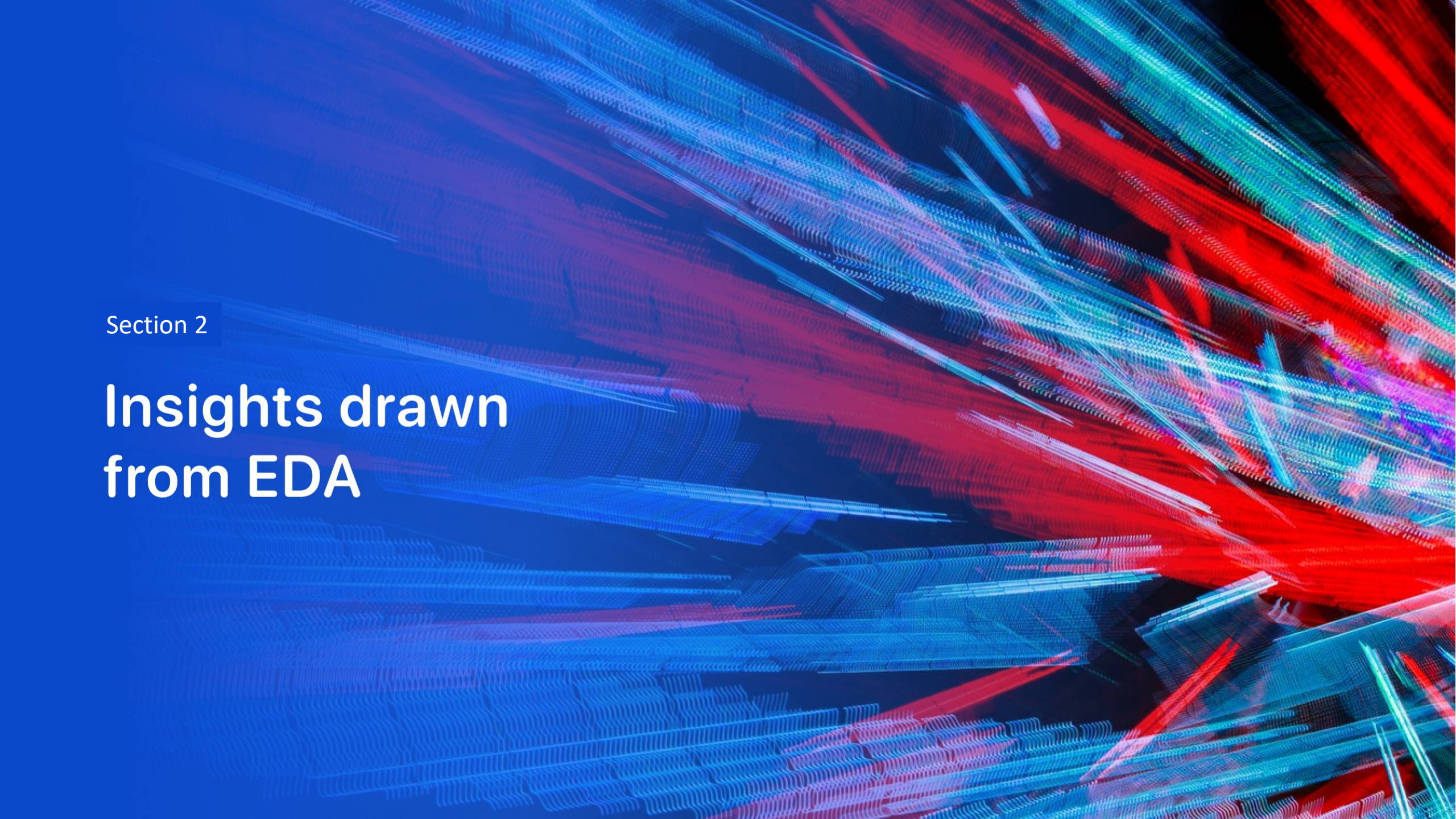
Results

- The Result for the first Model Logistic Regression: was: 88%
- The result for the Support Vector Machine was: 83.2%
- The result for Decision Tree Classifier was : 94.44%
- The result for the K-nearest neighbors was: 83.2%



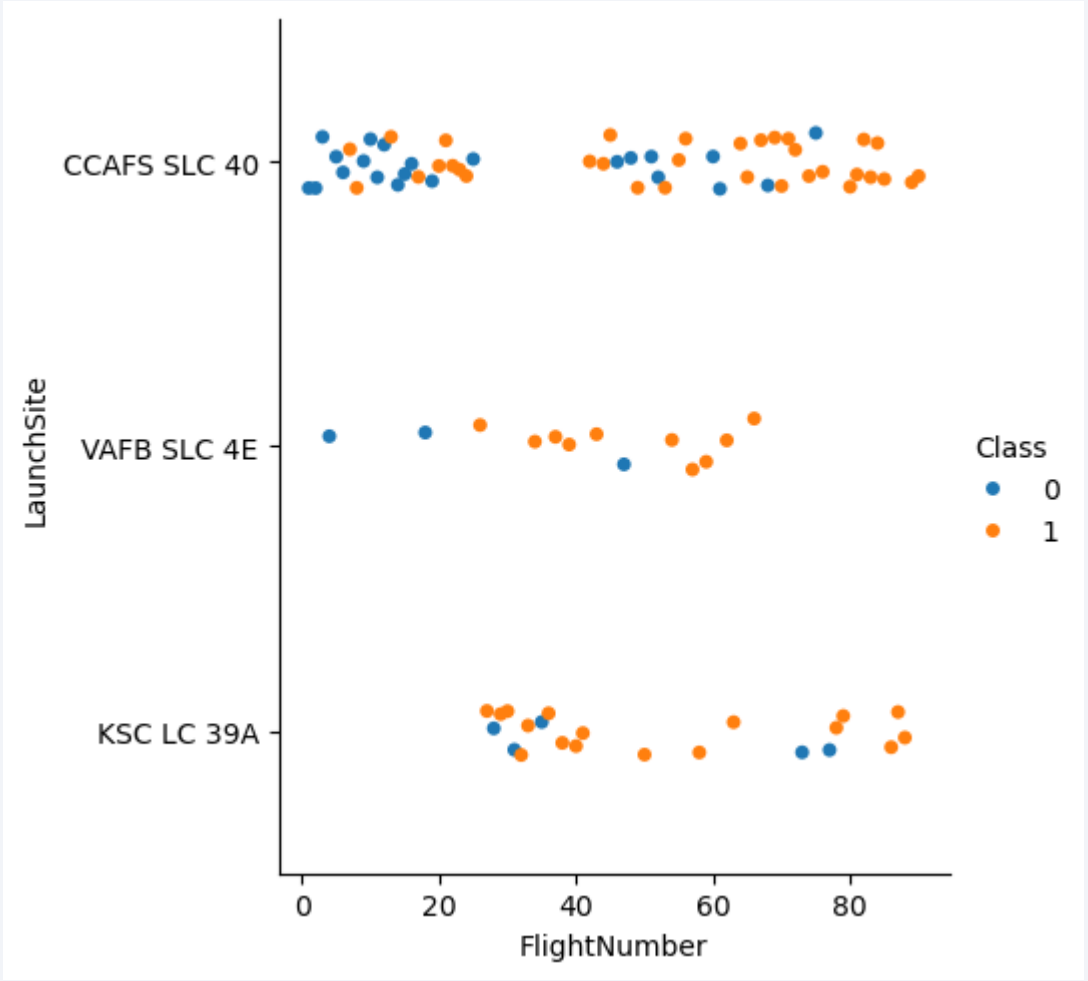
Section 2

Insights drawn from EDA



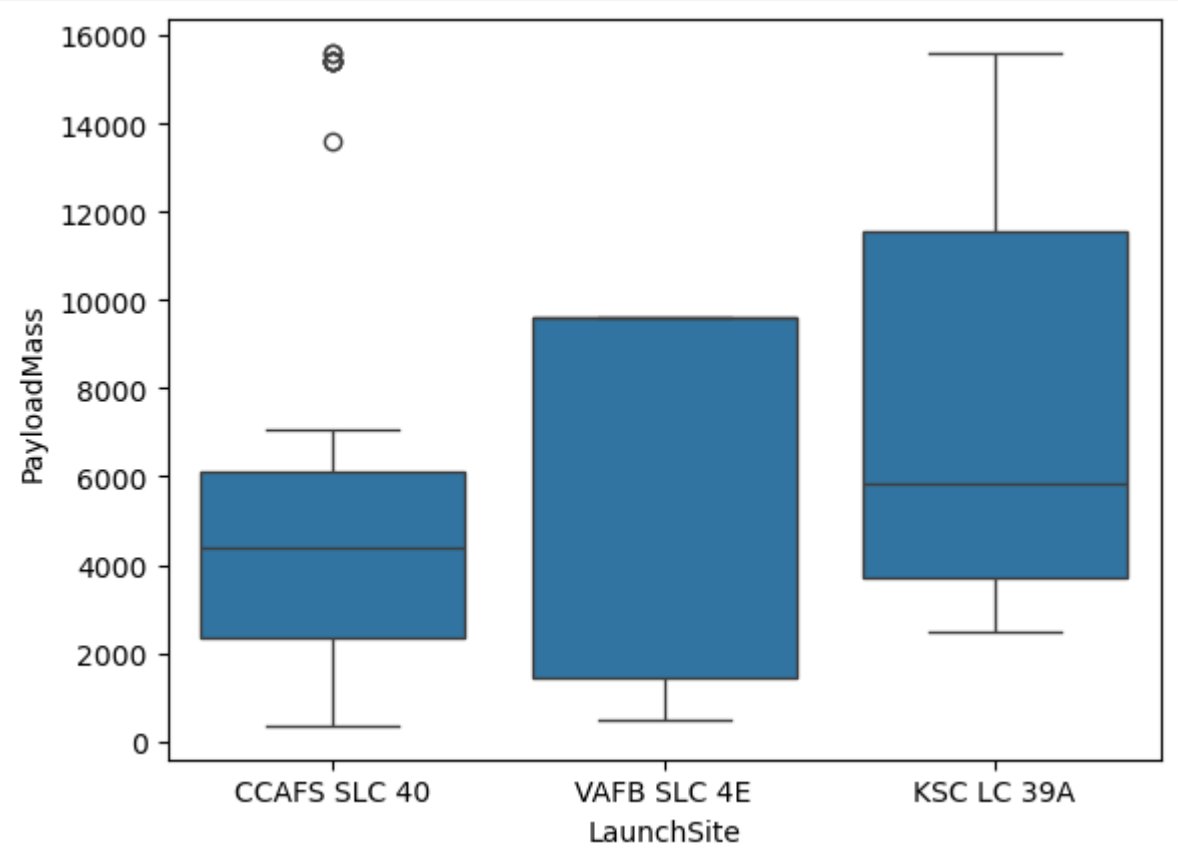
Flight Number vs. Launch Site

- a scatter plot of Flight Number vs. Launch Site
- The Orange/Blue dots shows successful or failure of launches
- We see that there is an improvement to the most recent launches for the three Sites, which were not significant in pervious launches.
- The scores are meant for successful or failure in the landing of the Launcher.



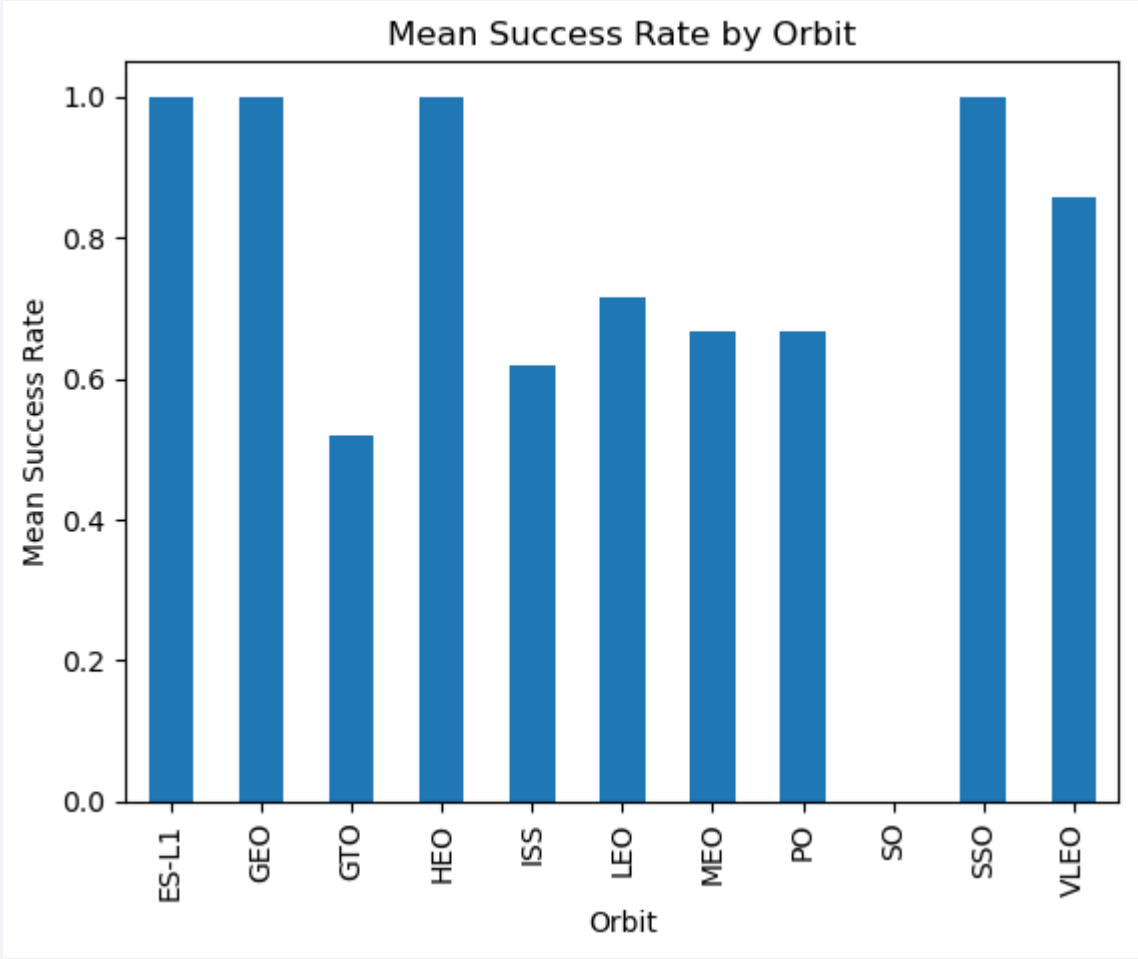
Payload vs. Launch Site

- a scatter plot of Payload vs. Launch Site
- We see that the site with the most successful Launches with highest payload is KSC LC 39A.
- We can determine that for little payloads the best location for launch is CCAFS SLC 40



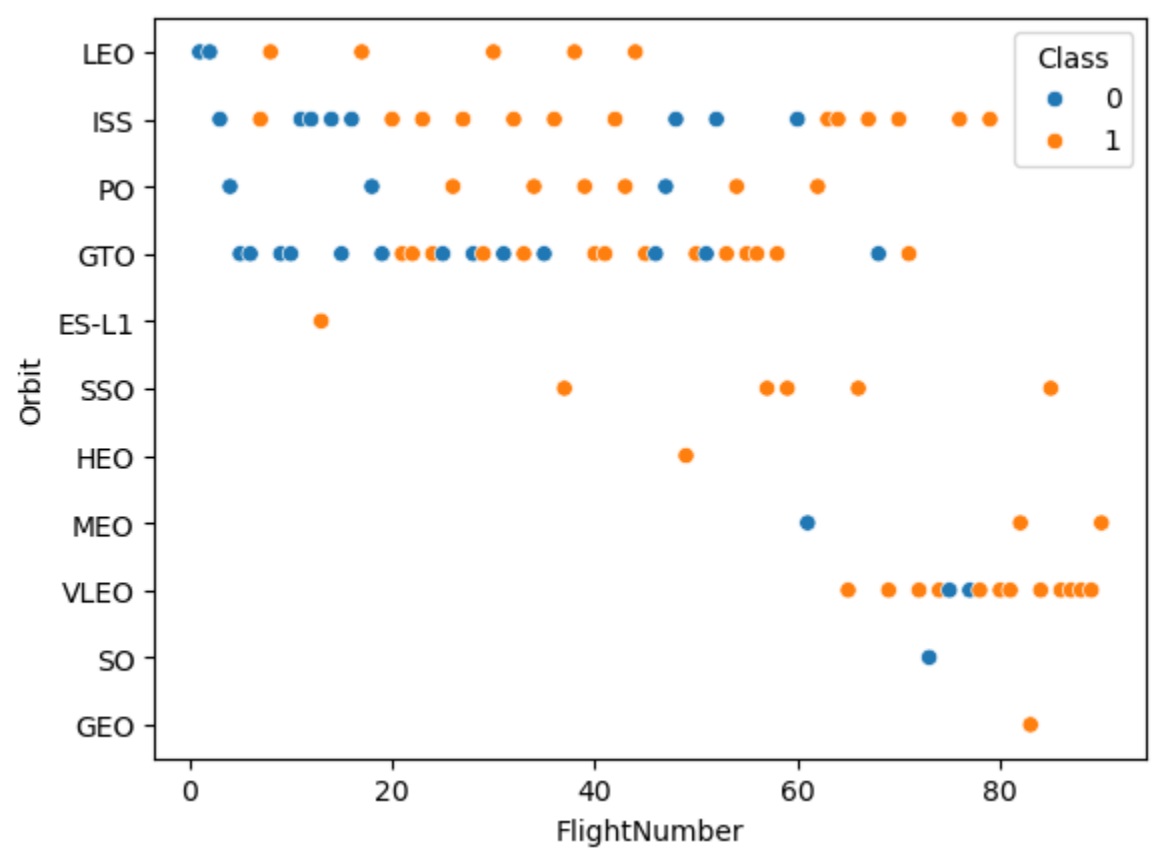
Success Rate vs. Orbit Type

- We can observe from the Chart the rate of success in ES-L1, GEO, HEO and SSO
- Are 1.0 meaning all the launches to this orbit were successful.
- Others were slightly less, but all of them were above 50% and several orbits were 60% or 70% success.



Flight Number vs. Orbit Type

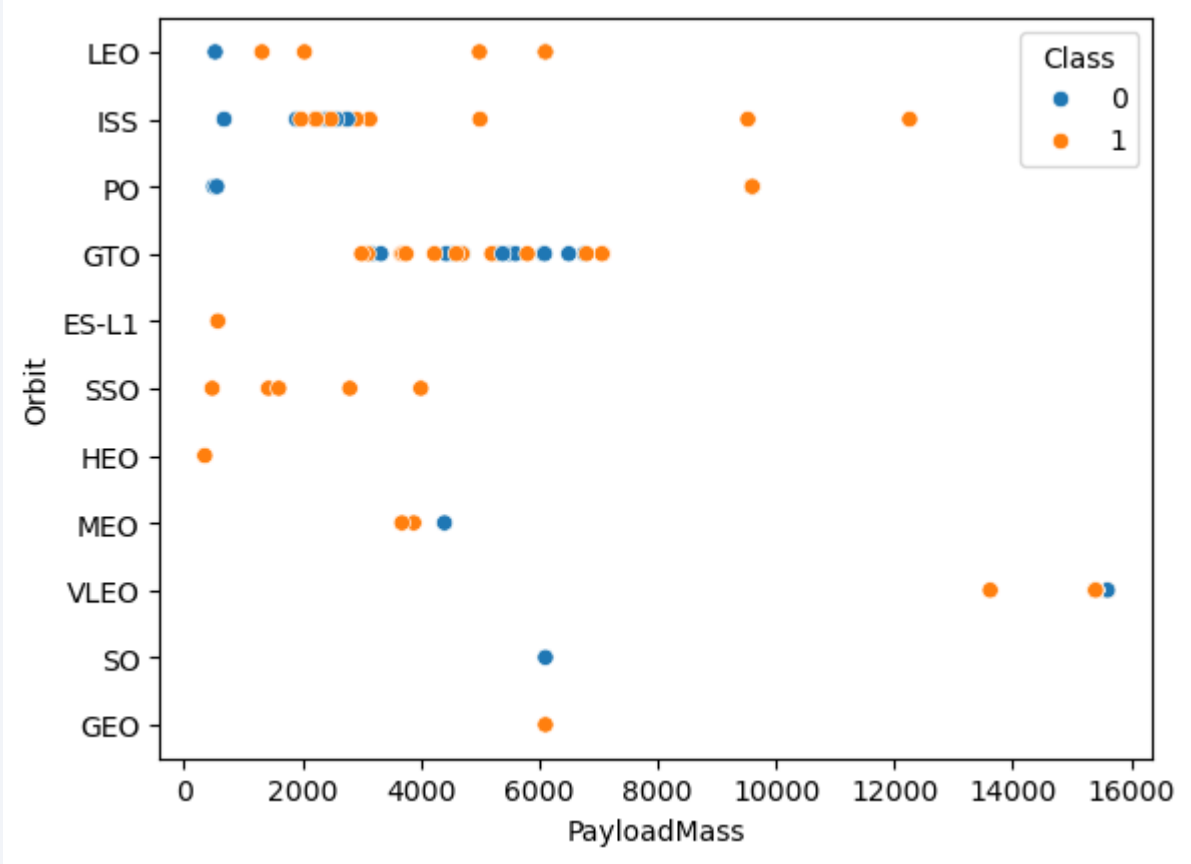
- a scatter point of Flight number vs. Orbit type
- Here the most successful orbit was VLEO, with high proportion of success.
- Generally, in most of launches from 60 Launches and further we see improvement.



Payload vs. Orbit Type

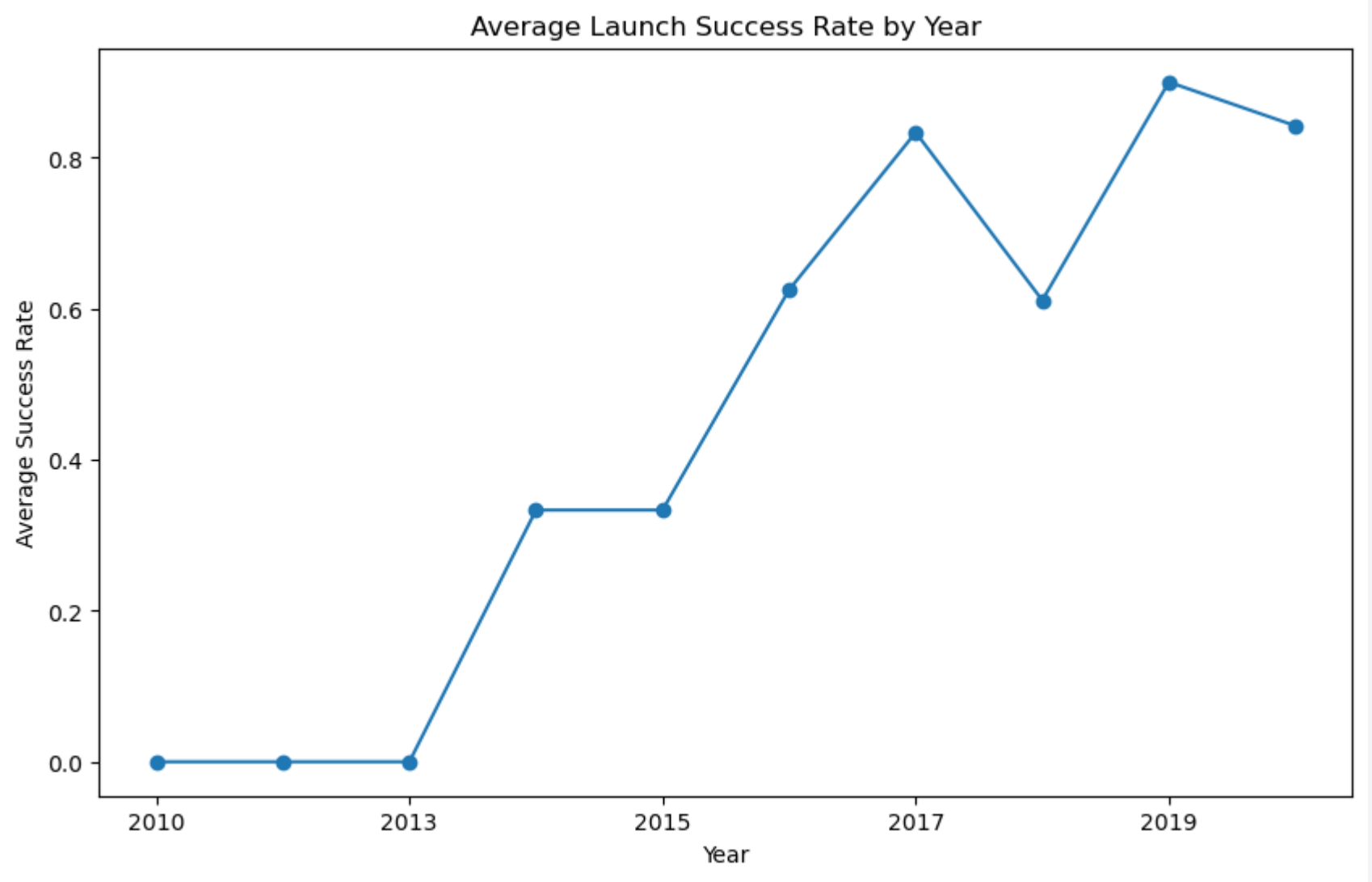
- a scatter point of payload vs. orbit type

- We can see here that most launches with high payload were successful, aside for one failure in the VLEO 15500.



Launch Success Yearly Trend

- a line chart of yearly average success rate
- From this line chart we can see that Space-X has improved dramatically over the years. They now look stable at 80%., after a small decline from year 2018.



All Launch Site Names

- names of the unique launch sites

- The list shows that Launch site CCAFS-40 is the most bussy and CCAFS SLC-40 is the less bussiest Launch Site.

```
spacex_df['Launch Site'].value_counts()
```

```
Launch Site
CCAFS LC-40     26
KSC LC-39A      13
VAFB SLC-4E     10
CCAFS SLC-40     7
Name: count, dtype: int64
```

Launch Site Names Begin with 'CCA'

- 5 records where launch sites begin with `CCA`

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

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

Launch_Site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- The total payload carried by boosters from NASA

6199 kg. were loaded on to space missions.

```
df['PAYLOAD_MASS__KG_'].sum()
```

```
619967
```

Average Payload Mass by F9 v1.1

The Average Payload Mass is:

2534.66 kg

```
filtered_df = spacex_df[spacex_df['Booster Version'].str.contains('F9 v1.1')]
print(filtered_df)
```

```
    Flight Number        Date Time (UTC) Booster Version  Launch Site  \
5               7  2013-12-03   22:41:00         F9 v1.1  CCAFS LC-40
6               8  2014-01-06   22:06:00         F9 v1.1  CCAFS LC-40
7               9  2014-04-18   19:25:00         F9 v1.1  CCAFS LC-40
8              10  2014-07-14   15:15:00         F9 v1.1  CCAFS LC-40
9              11  2014-08-05    8:00:00         F9 v1.1  CCAFS LC-40
10             12  2014-09-07    5:00:00   F9 v1.1 B1011  CCAFS LC-40
11             13  2014-09-21    5:52:00   F9 v1.1 B1010  CCAFS LC-40
12             14  2015-01-10    9:47:00   F9 v1.1 B1012  CCAFS LC-40
13             15  2015-02-11   23:03:00   F9 v1.1 B1013  CCAFS LC-40
14             16  2015-03-02    3:50:00   F9 v1.1 B1014  CCAFS LC-40
15             17  2015-04-14   20:10:00   F9 v1.1 B1015  CCAFS LC-40
16             18  2015-04-27   23:03:00   F9 v1.1 B1016  CCAFS LC-40
17             19  2015-06-28   14:21:00   F9 v1.1 B1018  CCAFS LC-40
26              6  2013-09-29   16:00:00  F9 v1.1  B1003  VAFB SLC-4E
27             21  2016-01-17   18:42:00   F9 v1.1 B1017  VAFB SLC-4E

                                Payload  Payload Mass (kg)        Orbit  \
```

```
filtered_df['Payload Mass (kg)'].mean()
```

```
2534.6666666666665
```

First Successful Ground Landing Date

```
%sql select Date from SPACEXTBL where Mission_Outcome='Success' limit 1
```

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

Date
2010-06-04

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select Booster_Version, Landing_Outcome from SPACEXTBL where Landing_Outcome='Success (drone ship)' and PAYLOAD_MASS__KG_ Between 4000 and 6000
```

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

Booster_Version	Landing_Outcome
F9 FT B1022	Success (drone ship)
F9 FT B1026	Success (drone ship)
F9 FT B1021.2	Success (drone ship)
F9 FT B1031.2	Success (drone ship)

Total Number of Successful and Failure Mission Outcomes

- From the Outcome we can see there were 80 successes and 10 failures

```
df['Booster landing'].value_counts()

Booster landing
Success          80
No attempt       18
Failure          10
Controlled        5
No attempt\n      4
Uncontrolled      2
Failure           1
Precluded         1
Name: count, dtype: int64
```

Boosters Carried Maximum Payload

```
%sql select Booster_Version, max(PAYLOAD_MASS__KG_) from SPACEXTBL
```

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

Booster_Version	max(PAYLOAD_MASS__KG_)
F9 B5 B1048.4	15600

2015 Launch Records

```
%%sql SELECT Booster_Version, Launch_Site, Landing_Outcome
FROM SPACEXTBL
WHERE Landing_Outcome = 'Failure (drone ship)'
AND strftime('%Y', Date) = '2015';
```

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

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

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

```
%sql select Landing_Outcome, Date from SPACEXTBL where Date Between '2010-06-04' and '2017-03-20' Order by Date Desc
```

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

Landing_Outcome	Date
No attempt	2017-03-16
Success (ground pad)	2017-02-19
Success (drone ship)	2017-01-14
Success (drone ship)	2016-08-14
Success (ground pad)	2016-07-18
Failure (drone ship)	2016-06-15
Success (drone ship)	2016-05-27
Success (drone ship)	2016-05-06
Success (drone ship)	2016-04-08
Failure (drone ship)	2016-03-04
Failure (drone ship)	2016-01-17
Success (ground pad)	2015-12-22


Section 3
Launch Sites
Proximities Analysis

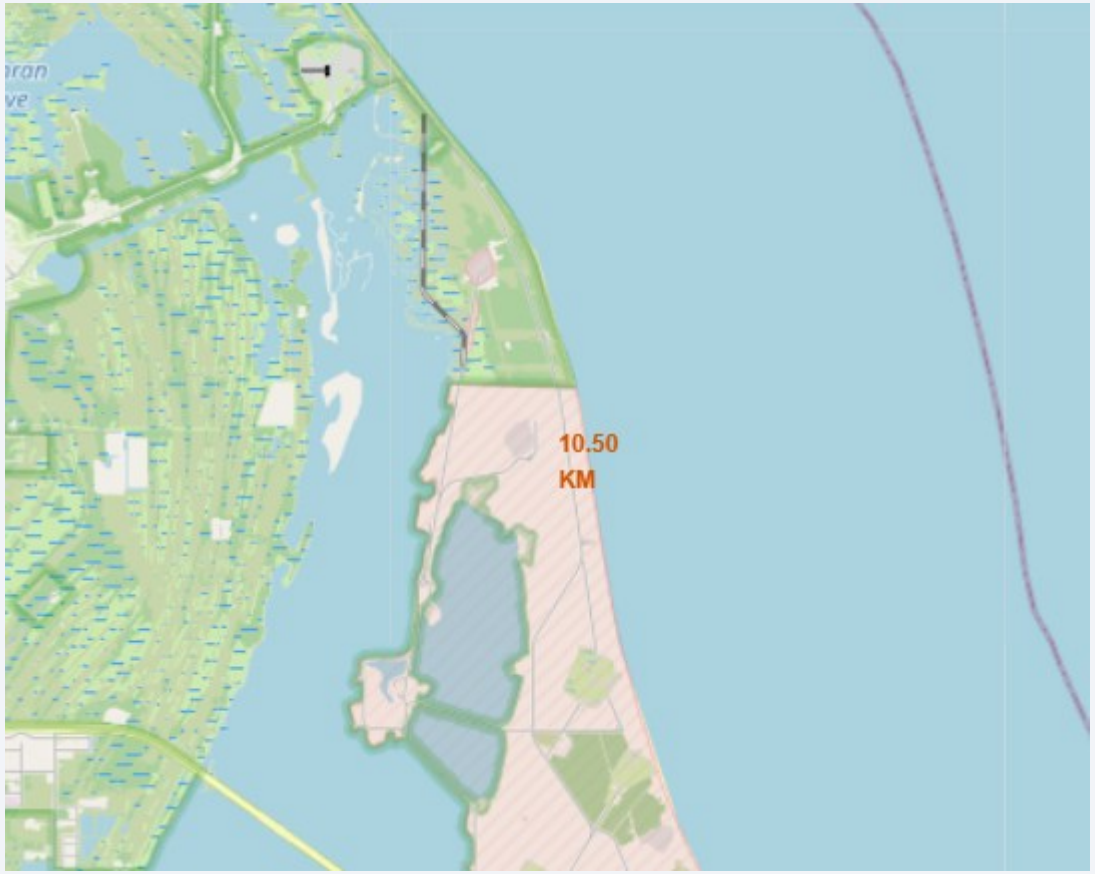
Folium Map – Space Launches

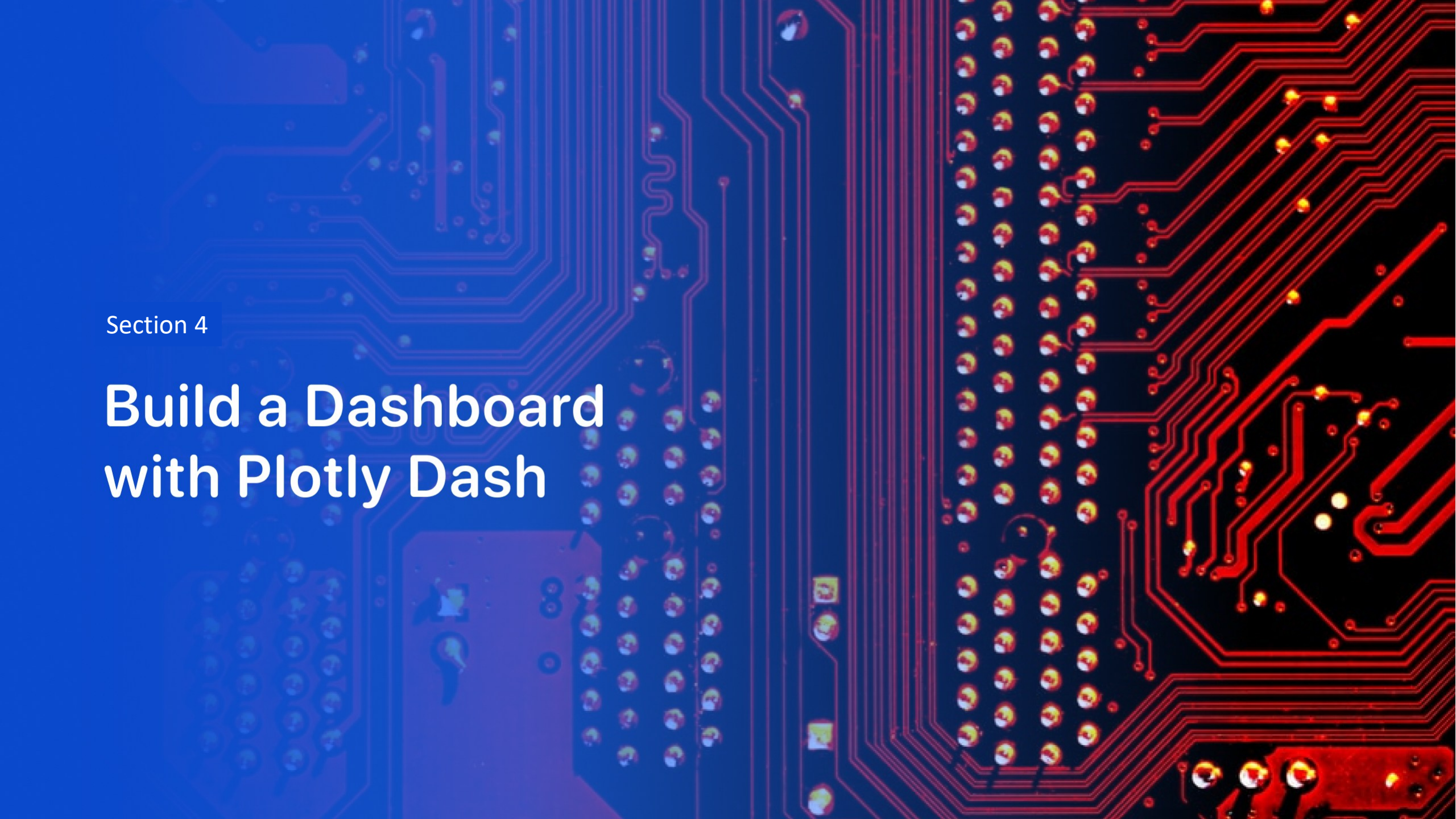
- We can see that there are 3 location in Cape Canavral NASA Center and 1 in Los Angeles

Folium Map – Colored Markers

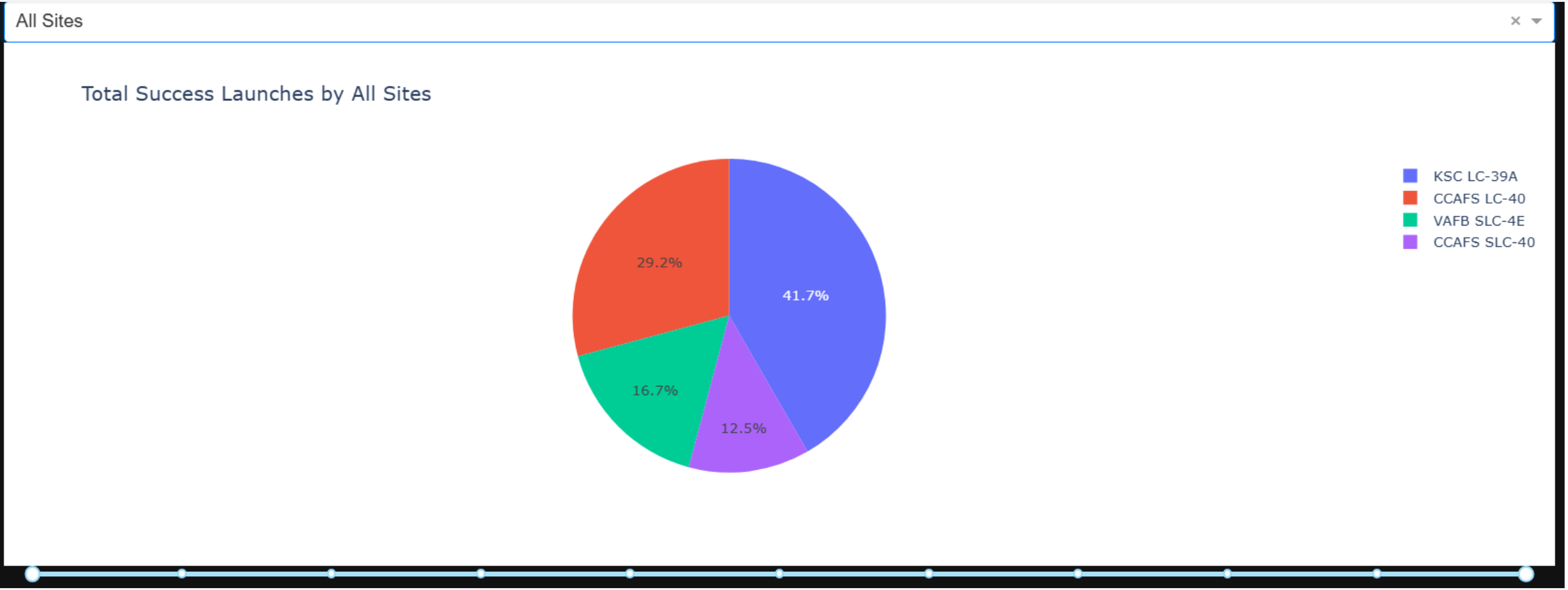
Those 3 sites are in Cape Canavral NASA Center Near Florida

Folium Map – Distance of Site from Coastline

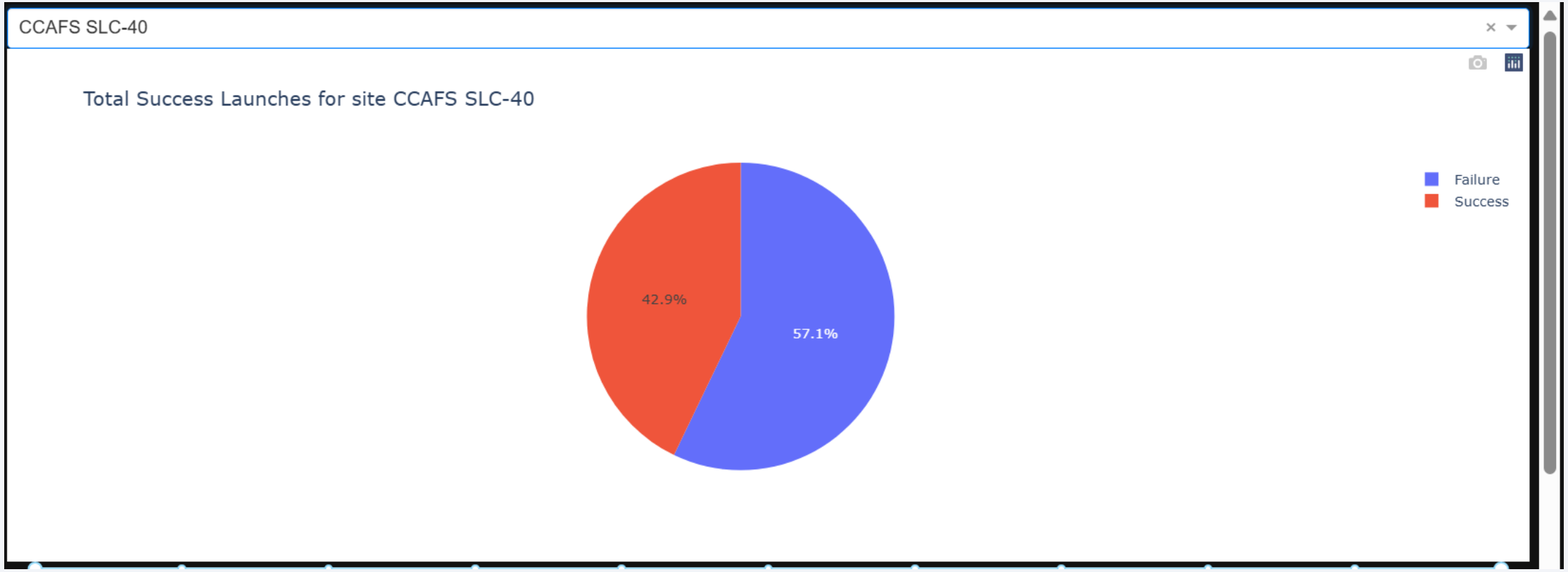



Section 4
Build a Dashboard with Plotly Dash

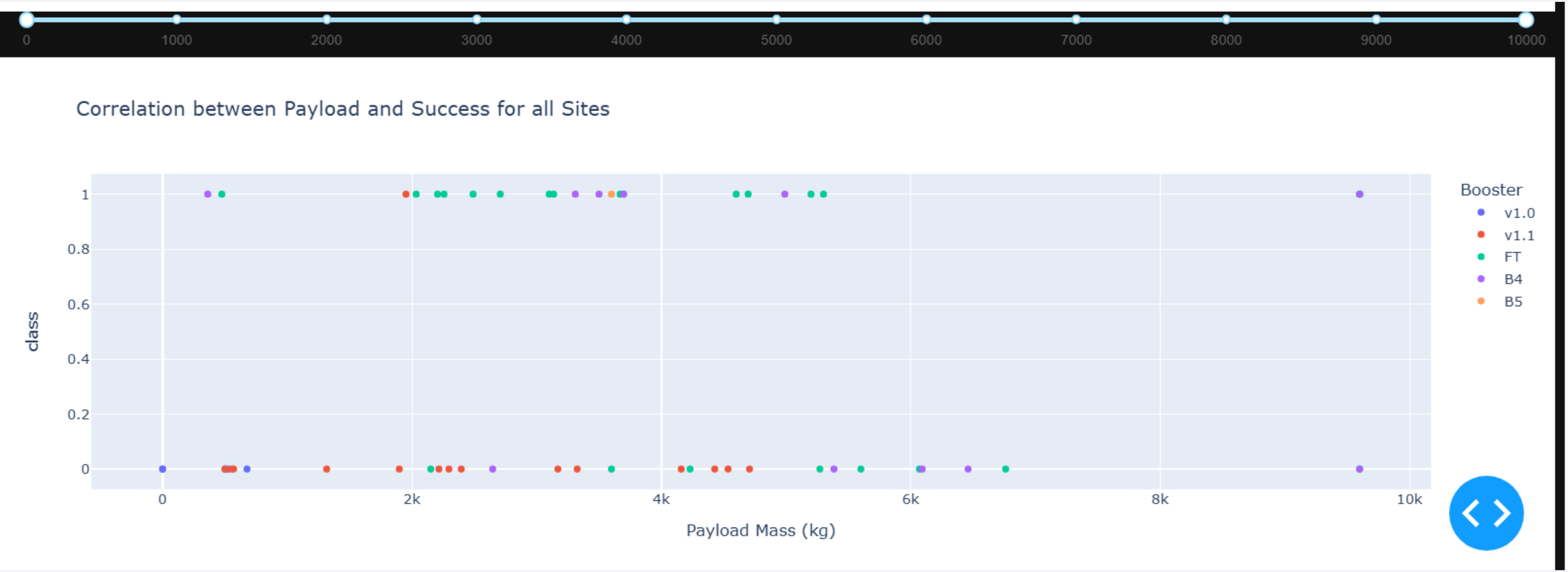
Dashboard – Total Success Launches by site



Dashboard 2-piechart for the launch site with highest launch success



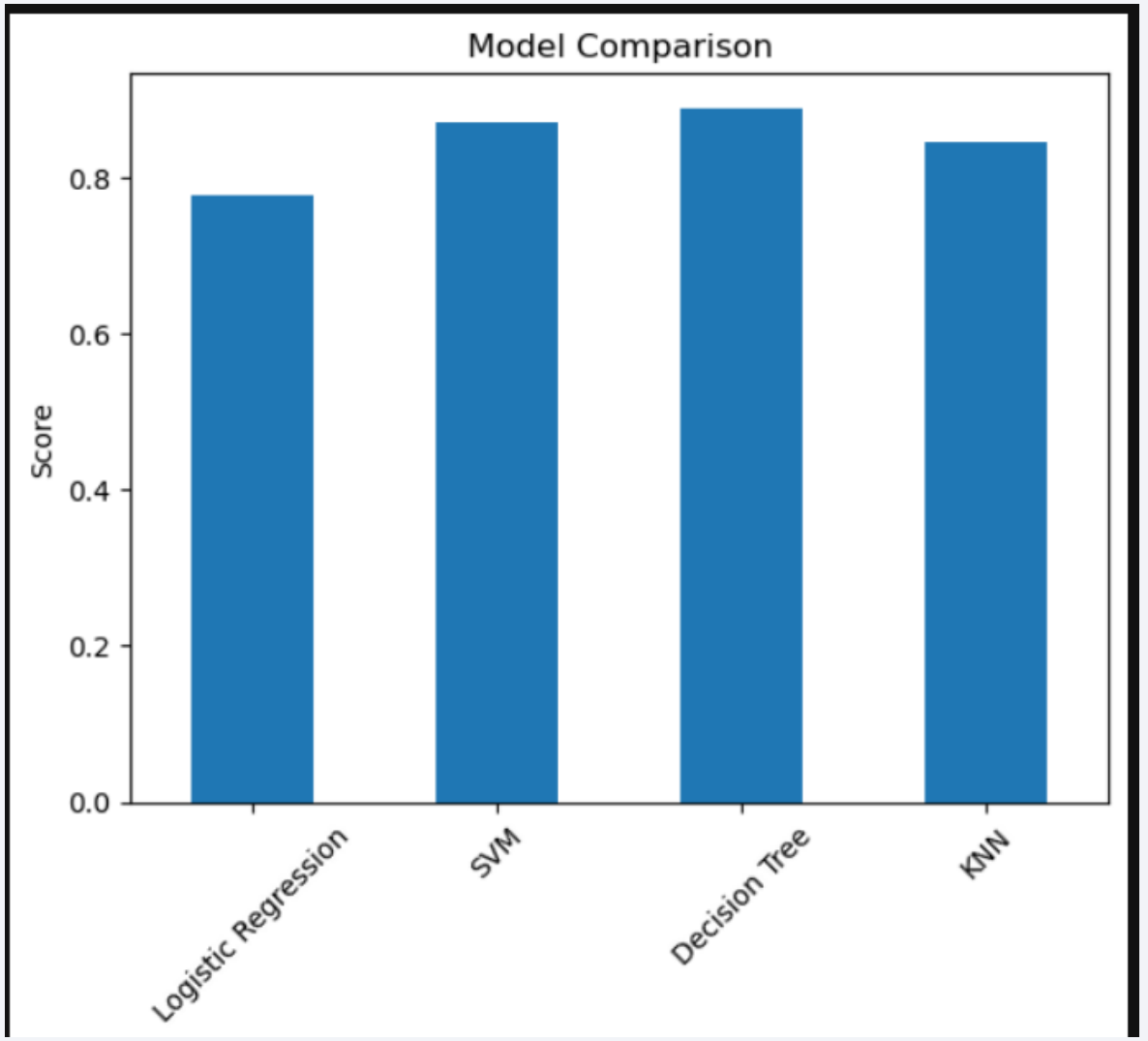
Dashboard- Payload vs. Launch Outcome scatter plot



Section 5

Predictive Analysis (Classification)

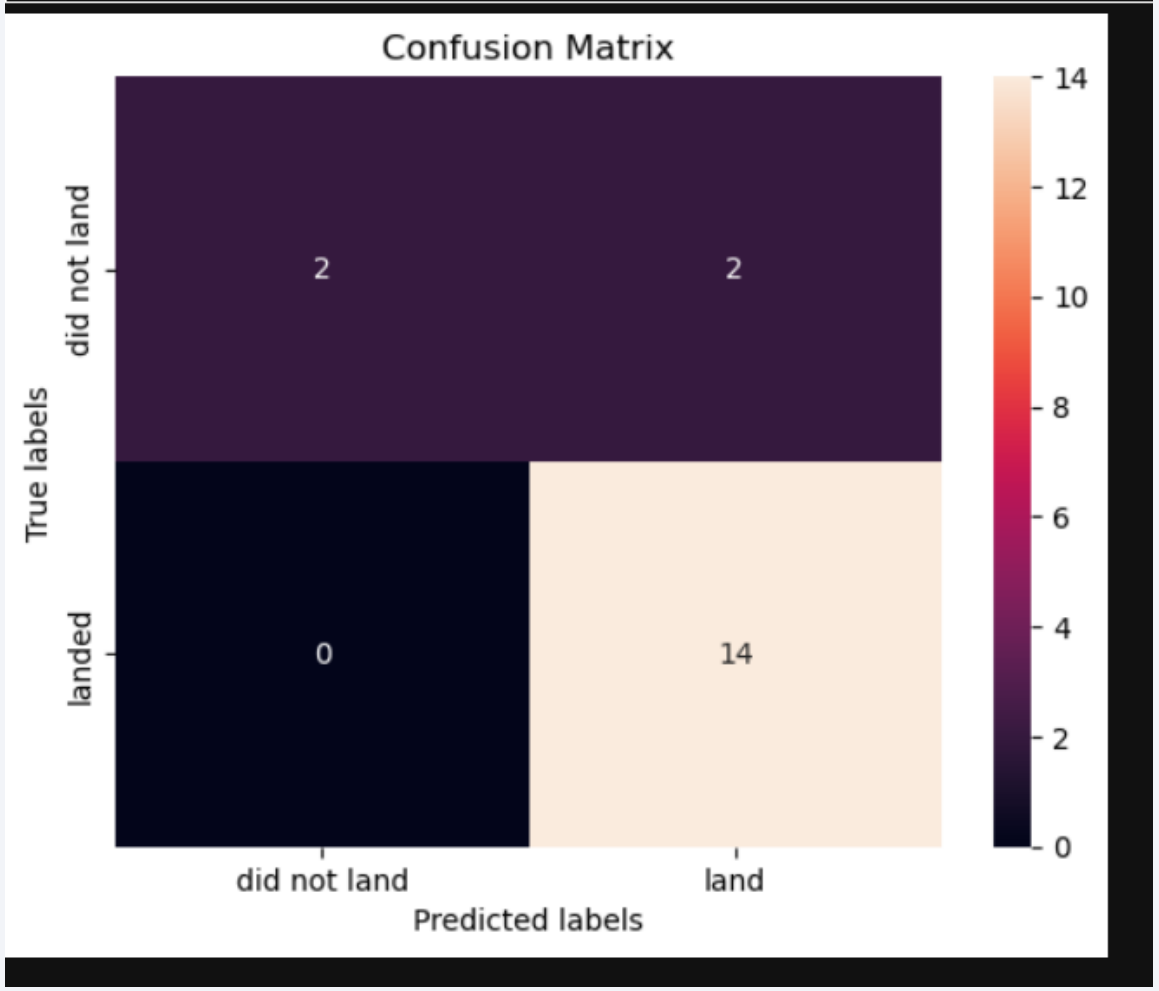
Classification Accuracy Bar chart



Confusion Matrix – Decision Tree Classifier

- We can see here that the number of mistake in the prediction is very low.
- There are 2 false-positives and 0 True negatives.

```
yhat = tree_cv.predict(X_test_scaled)
plot_confusion_matrix(y_test,yhat)
```



Conclusions

- I believe the best prediction for success in launching the next launcher is 88.8%
- Off course this prediction is by numbers and does not reflect all the elements of success for space-X.
- The chosen model I selected is Decision tree Classifier with the relevant Parameters:

```
Best Estimator: DecisionTreeClassifier(max_depth=14, max_features='sqrt', min_samples_leaf=5,
                       min_samples_split=5)
Training Accuracy: 0.8472222222222222
Test Accuracy: 0.8888888888888888
```

Thank you!