IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Atabo Mohammed Isah

29th November, 2024

Outline

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

Executive Summary

The aim of this project is to use SpaceX's Launch data to determine SpaceX rocket launch prices by predicting whether or not the rocket will land successfully and reuse the first stage.

Methodology

SpaceX's Launch data was scraped through an API from Wikipedia, on which **data wrangling, Exploratory Data Analysis (EDA) with SQL, visual analytics with folium and dash and prediction analysis** were performed to understand prior relationships between the many variable and visualize the different sites locations on the map and also the proximities of launch sites with other infrastructure such as highway, railway, etc.., and to also predict landing outcomes

Result

Result showed that landing success has increased since 2013 and prediction analysis favors more success rocket landings and reuse of the first stage. Other factors that significantly affect success landing outcomes were also discovered.

Introduction

We are in the commercial space age, with companies making space travel and investments more accessible. However, high rocket launch costs, often exceeding $165 million, remain a bottleneck. SpaceX reduces costs to $62 million by reusing the Falcon 9's first stage.

The Problem:

Not all landings are successful, as they depend on factors like orbit, payload mass, and customer requirements. The problems here is: how do we determine launch prices, by predicting whether SpaceX will land and reuse the first stage?

I will use SpaceX's public launch data to answer those questions

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - The data was collected from SpaceX's launch data that was gathered from an API—specifically SpaceX REST API. Which contained information about launches including information such as type of rocket used, payload delivered, launch and landing specifications and outcome.
- Data wrangling
  - The data collected was cleaned and formatted using Pandas and Numpy Libraries, to make analysis easy and accurate. For example;
  - NaN were removed or encoded, others were such as Landing Outcome were formatted and stored as Class to indicate landing success (1) or failure (2), The number of launch sites as well as the number of launches made from each site and the new dataset was saved to be used for EDA

Methodology

- Performing EDA using visualization and SQL:
  - I performed EDA using Pandas, Numpy, Matplotlib and Seaborn libraries to view the relationships between different relevant variables such as: Flight Number vs Payload Mass, Launch Site vs Flight Number, Launch Site vs PayLoad Mass, and also Success Rate vs Orbit Type
- Performing interactive visual analytics using Folium and Plotly Dash
  - Interactive visual analytics was done using Pandas, Folium and Plotly Libraries. Folium plugins such as MarkerCluster, MousePosition and DivIcon were used to create markers, lines and positions on the map. Plotly Dash was used to create a dashboard of the plots
- Perform predictive analysis using classification models
  - I used Pandas, Numpy and Sickit-learn libraries for the prediction. I made predictions by building Regression, Decision Tree, SVM, KNN models for training and testing, as well measured using confusion matrix.

Data Collection

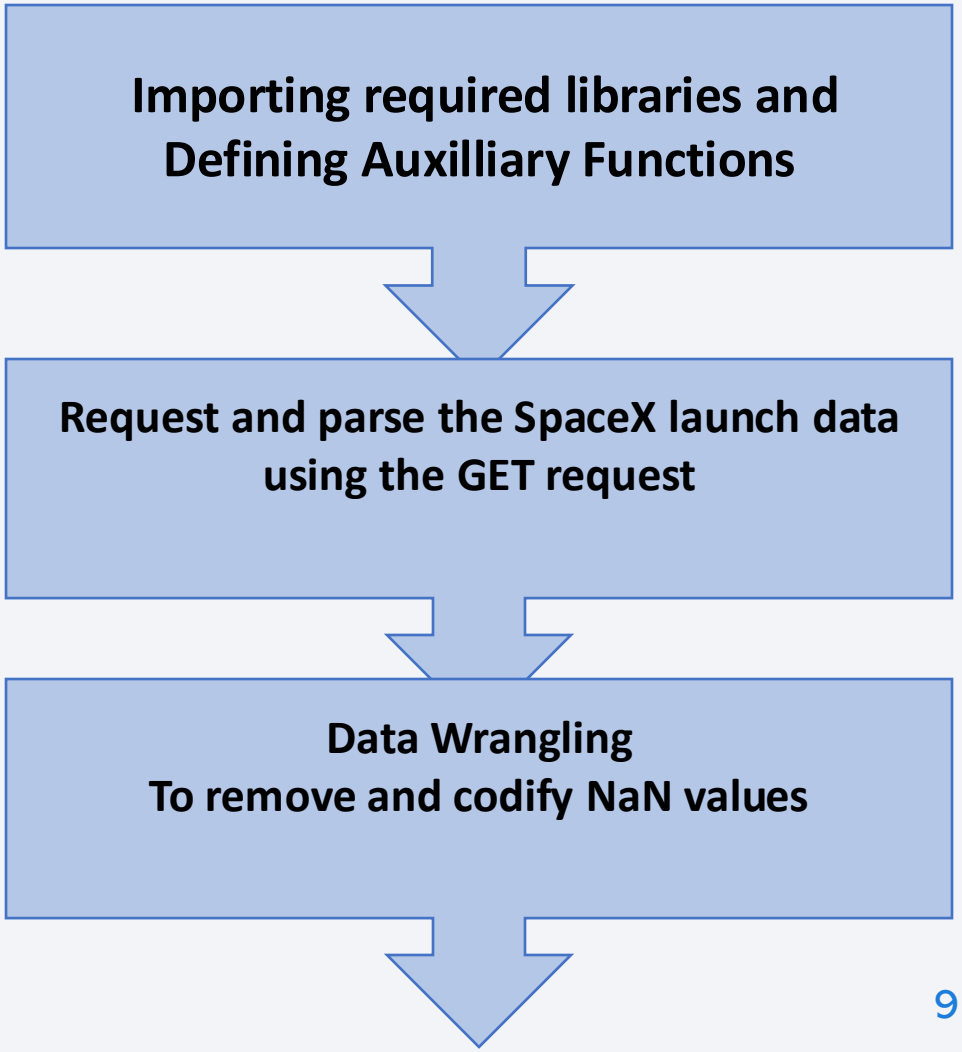
I placed a get request using the requests library to obtain the past launch data from the SpaceX REST API. The response came as a list of JSON objects of which each represented a list. I then converted the JSON into a dataframe using json_normalize function. Here's a breakdown:

- I started by importing required libraries: requests, pandas, numpy and datetime.
- Then I defined a series of helper functions to help us use the APIs to extract required information using ID numbers of the launch data such as LaunchSites, BoosterVersion, PayLoad data, etc
- I placed the request and then parsed the launch data, decoded the content using .json() and finally normalized it to a Pandas dataframe for wrangling

Data Collection – SpaceX API

- My data collection with SpaceX REST calls are represented in the flowcharts on the right.
- GitHub url:

https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/Spacex_api_request.ipynb

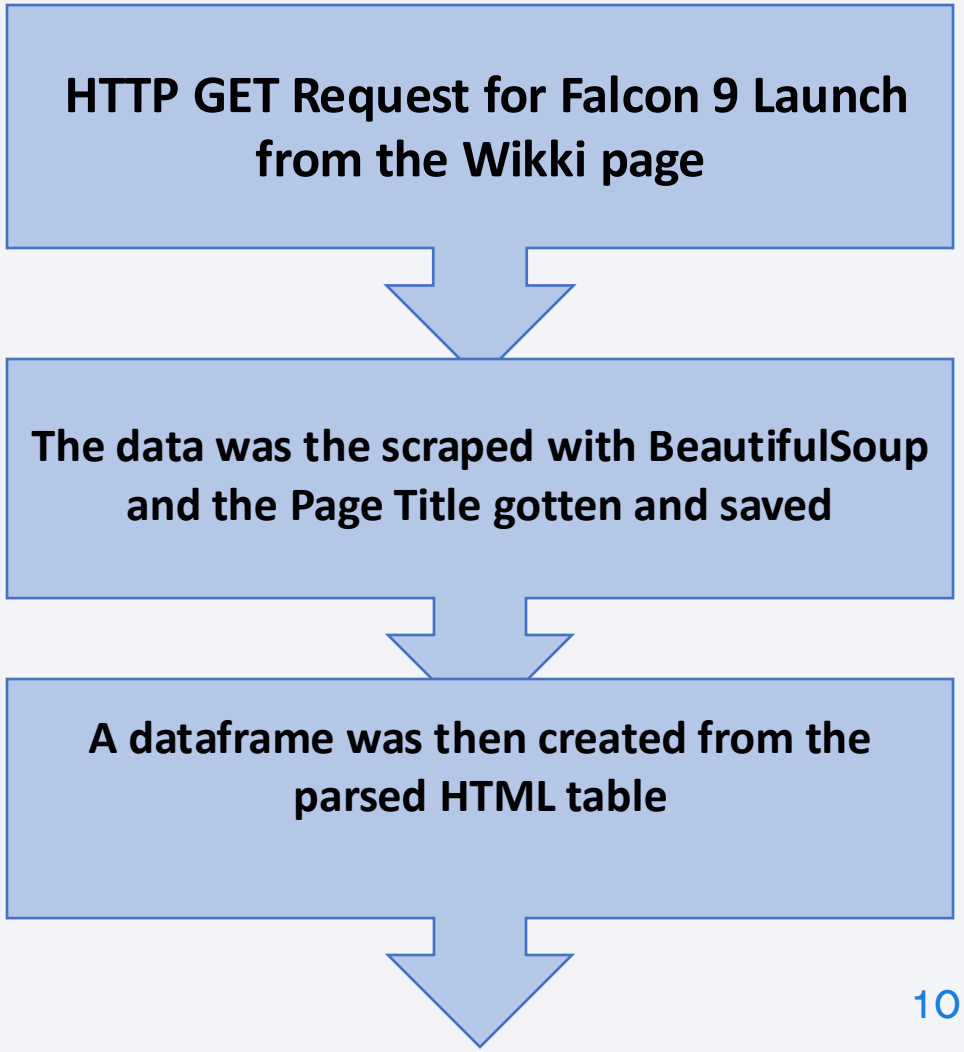


Data Collection - Scraping

- The SpaceX Launch data was Scraped from wikipedia

Github url:

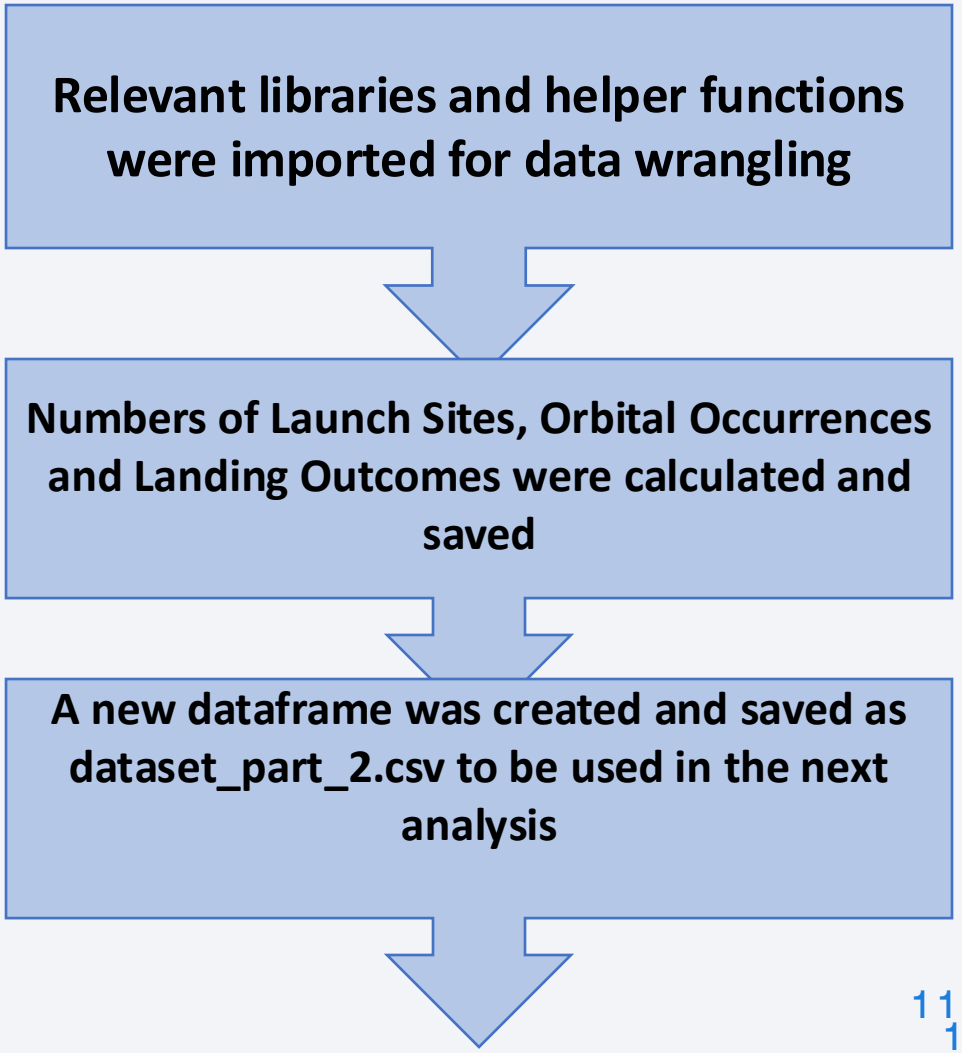
https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/labs-SpaceX_webscraping.ipynb



Data Wrangling

Data wrangling was performed on the data to improve formats for proper evaluation

Github url:
https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/SpaceX-DataWrangling.ipynb



11

EDA with Data Visualization

I performed EDA and Feature Engineering to have a view of the relationships between some important variables. Here's a summary of the plots and why:

1. A scatter plot of **Flight Number** and **Launch Site, it's** to see the number of launch attempts per site;
2. A scatter plot of **PayloadMass** and **LaunchSite**; to see the launch sites lthat aunched rockets with heavier payloads
3. A bar Plot of **Success Rate** By **Orbit** type; to see orbits with higher success rates
4. **Orbit** type and **Flight Number**, to see whether or not success and flight number can be related
5. **Orbit** and **PayloadMass**, to se if sucessful landing has something to do with orbit type
6. **Success rate** by **Year,** to understand how successful launches are with time

Finally, F**eature Engineering** was done, with creating dummy variable and hot-encoding

Github Url :

https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/edadataviz.ipynb

EDA with SQL

A summary of my SQL Queries

After installing and importing relevant libraries, I loaded the SQL extension and established a connection with the database (DB2), I then performed the following queries

- Dropped all existing SpaceX tables using the DROP TABLE IF EXISTS query to remove blank rows.
- Displayed the names of the unique launch sites in the space mission
- Displayed 5 records where launch sites begin with the string 'CCA'
- Displayed the total payload mass carried by boosters launched by NASA (CRS)Display average payload mass carried by booster version F9 v1.

Github url: https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science Capstone/blob/main/EDA_with_SQL.ipynb

EDA with SQL

A summary of my SQL Queries Contd.

- Listed the date when the first successful landing outcome in ground pad was acheived.
- Listed the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- Listed the total number of successful and failure mission outcomes
- Listed the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- Listed the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Github url: https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/EDA_with_SQL.ipynb

Build an Interactive Map with Folium

Summary of all map objects I created and Added to Folium map:

- MarkerCluster: to group clusters of LaunchSite into one big pile/cluster. And why I used it? To make maps more efficient, for when zoomed in, the cluster breaks apart to show individual sites.
- MousePosition: to display latitudes and longitudes of sites on the map. And why I used it? To get positions of relevant locations such as nearest coast lines, railway tracks, highways, etc relative to Launch Sites
- Divicon: This tool added custom dynamic markers to my map, such as colorful circles, etc. I used it to display rich content on the map, such as the sites location with yellow colors and count (for number of sites in a cluster), displaying the distances from sites to nearest landmarks such as railway, etc

GitHub URL: https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/Visual%20Analytics%20with%20Folium2.ipynb

Build a Dashboard with Plotly Dash

A summary of my dashboard plots/graphs

- I created a dropdown bar to list the different Launch Sites; this is ideal for easier selection and viewing of the different Launch site data
- A Pie chart was plotted to display the success count for all launch sites. I created it because pie charts are used to display part of a whole, making it easy to compare the relative sizes of different parts
- A scatter plot for the payload mass and launch Success (Class); This is to the relationship between launch success with increasing payload mass

GitHub URL: https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/spacex_dash_app%20(1).ipynb

Predictive Analysis (Classification)

A machine learning pipeline was created to predict whether the First Stage will land. This is done via the following steps:

- Standardizing the data
- Splitting and Training the data
- Find the best hyperparameter for SVM, Classification Trees and logistic Regression models and
- Found the method that performs best using the test data.

GitHub URL: https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/Spacex_landing_prediction.ipynb

IMPORTING LIBRARIES AND USER FUNCTIONS

- Imported relevant libraries and
- the confusion matrix function

CLEANING, STANDARDIZATION AND SPLITTING

- I performed one-hot-encoding on some columns,
- created a numpy array for y data
- Standardize the X data
- Split the data into training and testing

LOGISTIC REGRESSION AND EVALUATION

- I created Logistic Regression and GridSearch Objects
- Trained the model, and
- Evaluated using Best Cross-Validation Accuracy and Test accuracy
- I displayed the result with Confusion Matrix

Predictive Analysis (Classification)

A machine learning pipeline was created to predict whether the First Stage will land. This is done via the following steps:

- **Standardizing the data**
- **Splitting and Training the data**
- **Find the best hyperparameter for SVM, Classification Trees and logistic Regression models and K-Nearest Neighbour**
- **Found the method that performs best using the test data.**

GitHub URL: https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/Spacex_landing_prediction.ipynb

SUPPORT VECTOR MACHINE AND EVALUATION

- I created SVM and GridSearch Objects
- Trained the model, and
- Evaluated using Best Cross-Validation Accuracy and Test accuracy
- I displayed the result with Confusion Matrix

CLASSIFICATION TREES AND EVALUATION

- I created Decision Tree Classifier and GridSearch Objects
- Trained the model, and
- Evaluated using Best Cross-Validation Accuracy and Test accuracy
- I displayed the result with Confusion Matrix

Predictive Analysis (Classification)

A machine learning pipeline was created to predict whether the First Stage will land. This is done via the following steps:

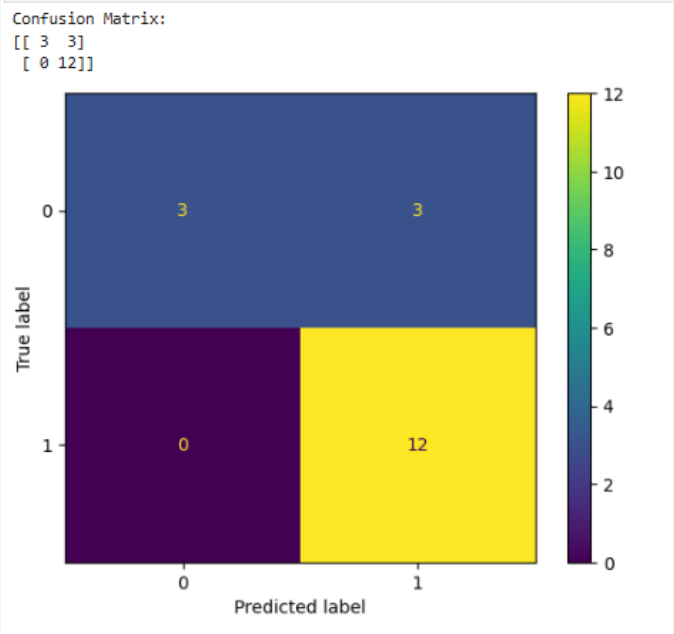
- Standardizing the data
- Splitting and Training the data
- Find the best hyperparameter for SVM, Classification Trees and logistic Regression models and K-Nearest Neighbour
- Found the method that performs best using the test data.

GitHub URL: https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/Spacex_landing_prediction.ipynb

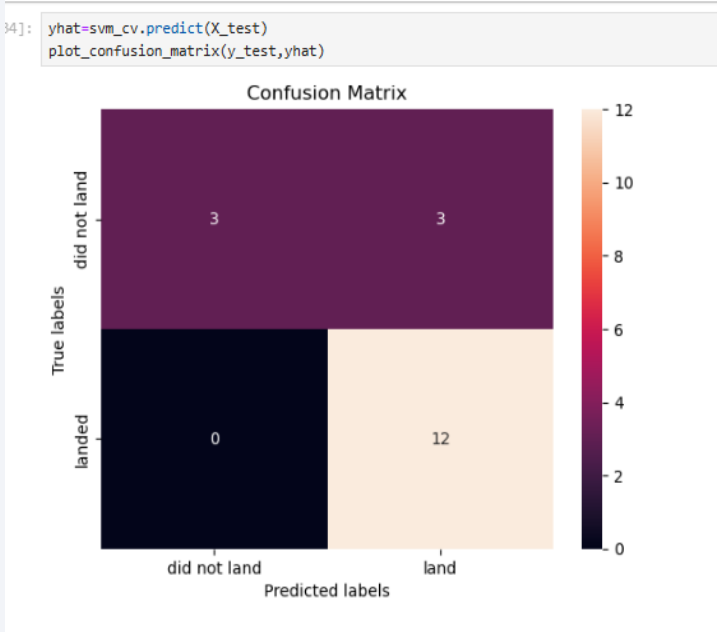
KNN AND EVALUATION

- I created knn and GridSearch Objects
- Trained the model, and
- Evaluated using Best Cross-Validation Accuracy and Test accuracy
- I displayed the result with Confusion Matrix

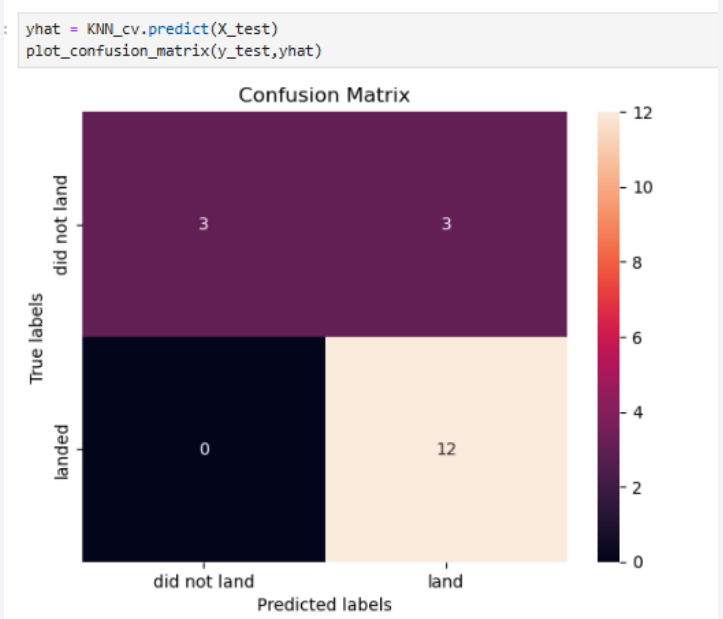
Results



Confusion Matrix for Logistic Regression. We can see that 12 True Labed landed of the predicted label; and 3 False positive Labels



Confusion Matrix for SVM showing12 True Labed landed of the predicted label; and 3 False positive Labels

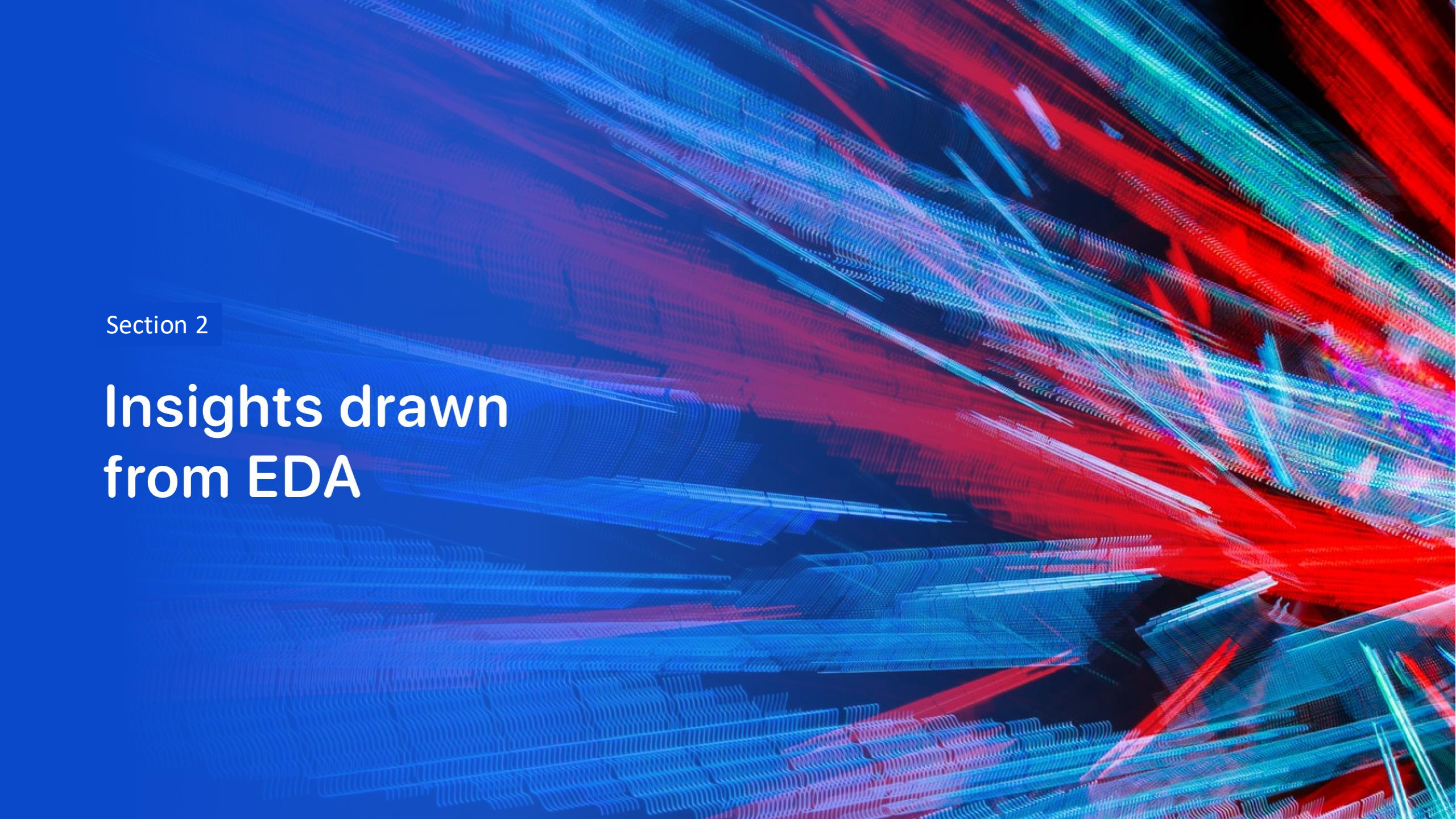


Confusion Matrix for KNN, showing 12 True Labed landed of the predicted label; and 3 False positive Labels

Predictive analysis results

- The Accuracy score for Decision Trees is closer to 1 (0.875) relative to the others (0.83392). The value suggested that the model perform better on the training data in terms of accuracy. Closer to 1 means the model is making fewer errors. t's important to compare test accuracy in addition to cross-validation accuracy. The models performs great, especially with the results of the Test and Cross-Validation scores which are all close to 1.

- GitHub URL: https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/blob/main/Spacex_landing_prediction.ipynb

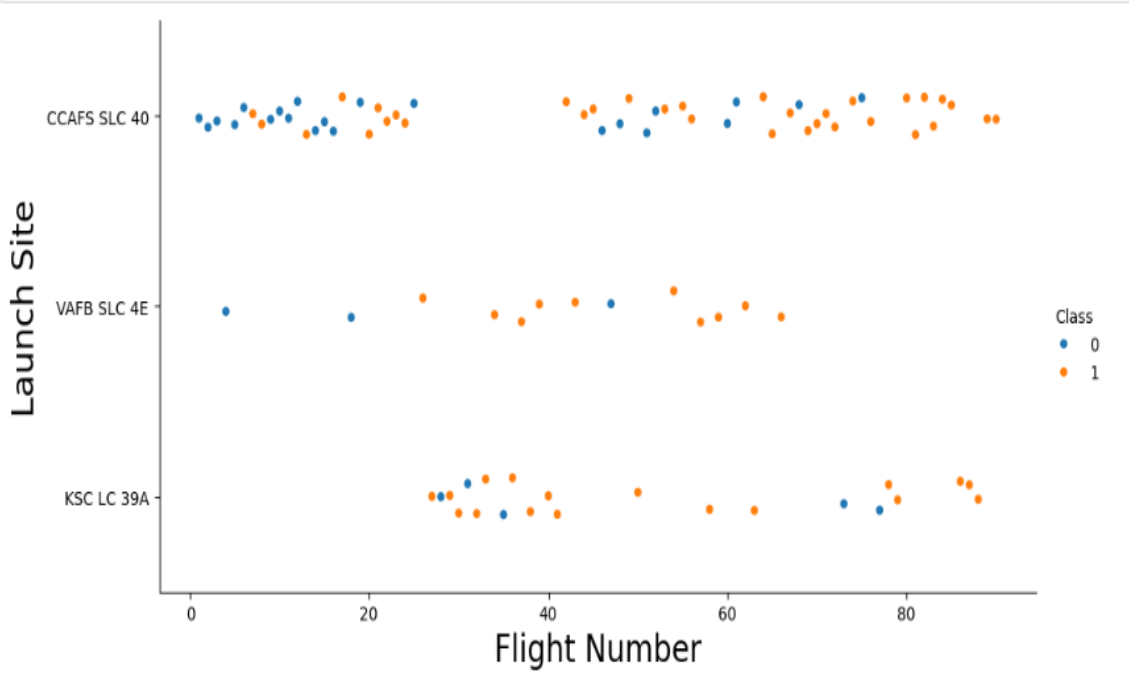

Section 2
Insights drawn from EDA

Launch Site vs. Fligh Number

Insights

This plot illustrates the relationship between Lauch Site and Flight Number.

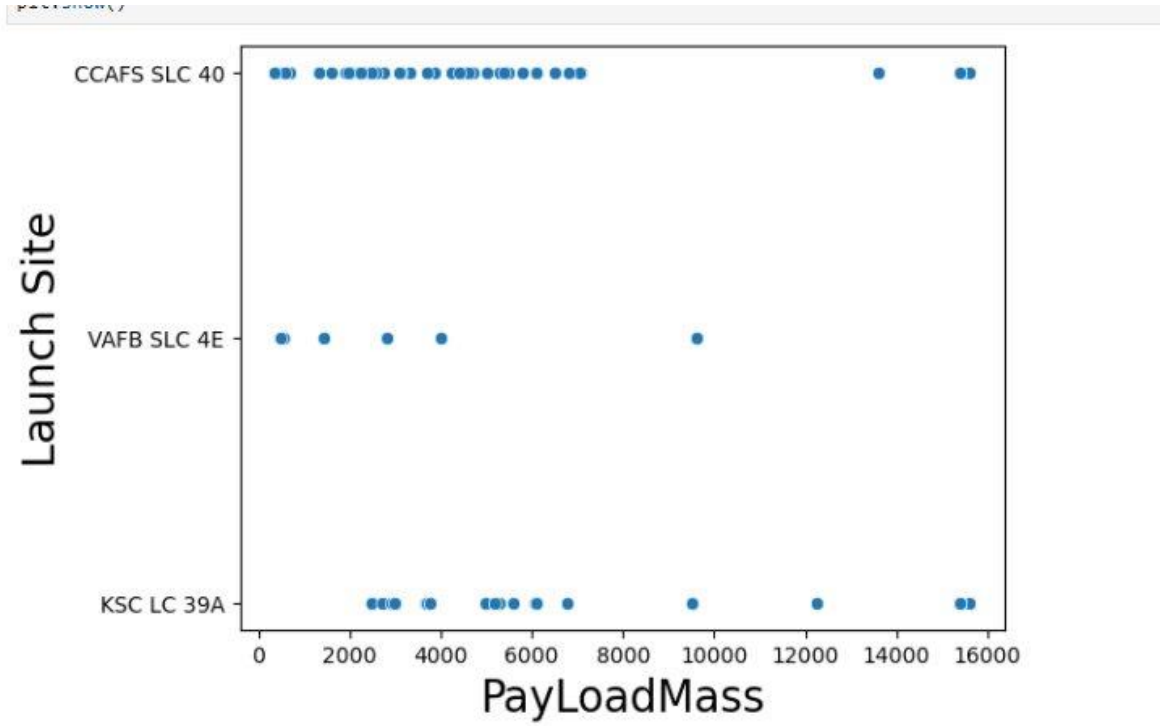
As you can see, of the Three Space X's Lauch Facilities, CCAFS SLC 40 have seen more launch attempts than the rest. It is followed by KSC LC 39A. VAFB SLC 4E has seen the least launch attempts



Launch Site vs. PayLoadMass

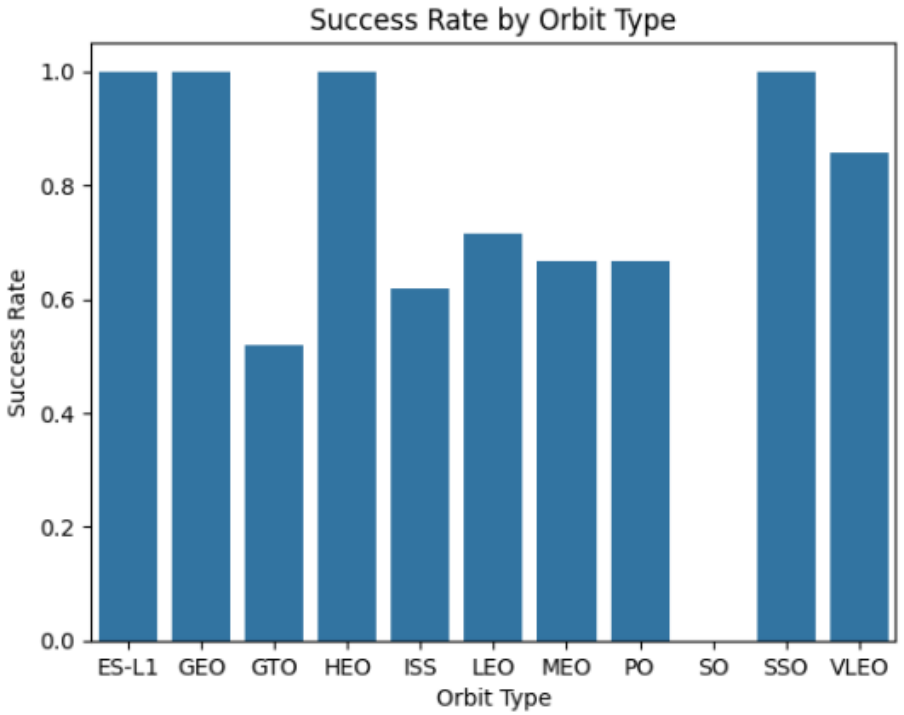
Insights

If you observe Payload Mass Vs. Launch Site scatter point chart you will find for the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).



Success Rate vs. Orbit Type

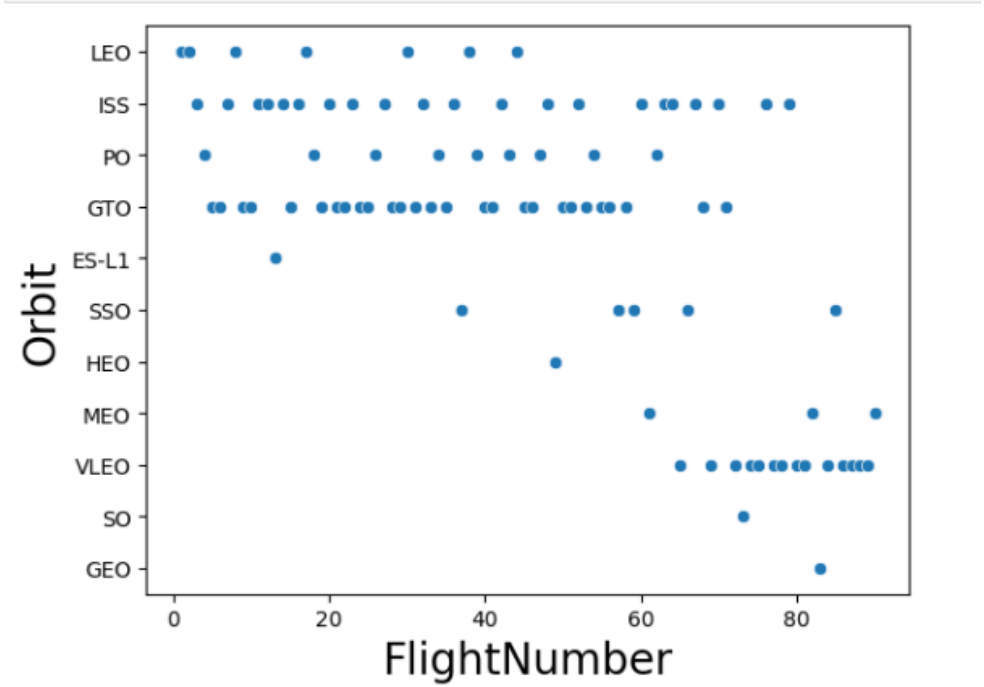
- From the plot, we can see that the orbits have the highest success rates are
- 1. ELS-L1
- 2. GEO
- 3. GTO and
- 4. SSO



Flight Number vs. Orbit Type

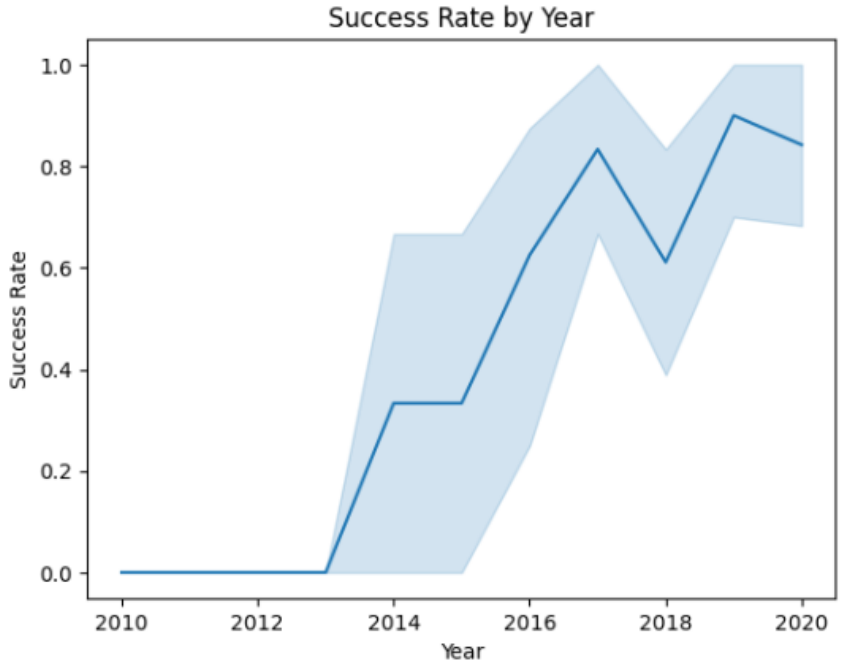
You can observe that in the LEO orbit, success seems to be related to the number of flights.

Conversely, in the GTO orbit, there appears to be no relationship between flight number and success.



Launch Success Yearly Trend

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



All Launch Site Names

Displayed are the names of the unique launch sites in the space mission

- CCAFS LC-40 (Cape Canaveral Air Force Station Launch Complex 40) - Located in Cape Canaveral, Florida.
- VAFB SLC-4E (Vandenberg Space Force Base, Space Launch Complex 4E) - Located in Lompoc, California
- KSC LC-39A (Kennedy Space Center Launch Complex 39A) - Located in Merritt Island, Florida.
- CCAFS SLC-40 (Cape Canaveral Air Force Station, Space Launch Complex 40) - Also in Cape Canaveral, Florida. (This is the same location as the first entry)

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

Launch Site Names Begin with 'CCA'

Here are the 5 records where launch sites begin with `CCA`

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

These Sites show the progression from testing to operational space missions, indicating SpaceX's early use of CCAFS as a key launch site for both demonstration and critical ISS supply missions

Total Payload Mass

I calculated the total payload carried by boosters from NASA with a simple SQL query (left)

```
%sql select Customer,
sum(PAYLOAD_MASS__KG_) as Total_Payload_Mass
from SPACEXTBL
where Customer = 'NASA (CRS)' group by Customer
```

Customer	Total_Payload_Mass
NASA (CRS)	45596

There are several space companies in the Customer columns and NASA is one of them. So I filtered through the column using the 'WHERE' clause and specified NASA as the customer I needed. The Total Payload Mass carried by NASA booster came to 45596kg (right)

Average Payload Mass by F9 v1.1

I calculated the average payload mass carried by booster version F9 v1.1 using a simple SQL query (left);

```
%sql select
avg(PAYLOAD_MASS__KG_) as Average_Payload_Mass
from SPACEXTBL
where Booster_Version = 'F9 v1.1';
```

```
 * sqlite:///my_data1.db
Done.
 Average_Payload_Mass
                2928.4
```

I used the avg() function to get the average Payload mass of the Booster-version Column and then filtered through the column using the 'WHERE' clause and specified *F9 v1.1.* The AveragePayload Mass carried by NASA booster came to 29284Kg (right)

First Successful Ground Landing Date

I found the dates of the first successful landing outcome on ground pad using a simple SQL query shown below:

%sql select min(Date) as First_Success_Date from SPACEXTBL where Mission_Outcome = 'Success';

The output is shown below:

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

First_Success_Date
2010-06-04

I used the min() function to filter the earliest date where the Mission_outcome variable is 'success'.

Successful Drone Ship Landing with Payload between 4000 and 6000

Below is the list of booster names which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```
%sql select Booster_Version from SPACEXTBL
where Mission_Outcome = 'Success' and
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

The filtering is a little moderately complex with mulitiple filtering conditions; but it was able to give us the list we needed displayed on the right.

Total Number of Successful and Failure Mission Outcomes

The total number of successful and failure mission outcomes was claculated using the sql query below

```
%sql select Mission_Outcome, count(*) as Total_mission

from SPACEXTBL

GROUP BY Mission_Outcome;
```

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

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

The result is shown on the right. The filtering argument is GROUPBY, this function groups the like outcomes into similar groups.

Boosters Carried Maximum Payload

To the right is a List of the names of the booster which have carried the maximum payload mass. Below is the sql query;

%sql *select Booster_version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL);*

As you can see, the query is complex having had a sub-query and aggregate function. This is necessary for acuracy.

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

Here's the List of the failed landing_outcomes in drone ship with their booster versions, and launch site names for in year 2015

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

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

There are 2 dron ship landing failure records in 2015, one in the month of January and the other in April. And both happened in CCAFS LLC-40 Launch sites

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

On the right is a rank of 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:

Dron ship seem to have the highest outcome count with equal successes and failures (5 in each case).

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

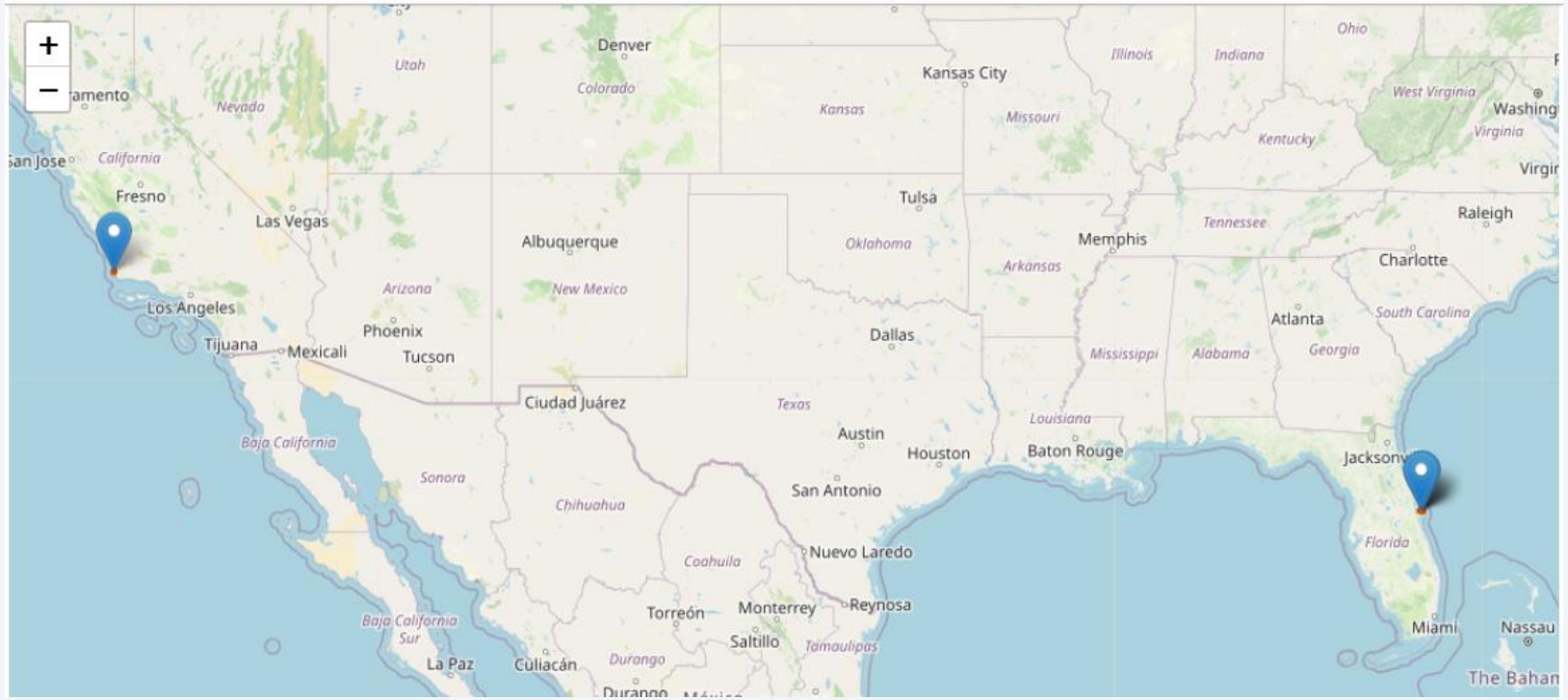
Landing_Outcome	outcome_count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

Map of All Launch Sites

The map of all launch sites is displayed below:



There are 4 launch sites, three of which are located by the cost of Florida and the other on the coast of Los Angeles

CCAFS SCL-40 Launch Outcomes

On the right is a color-labeled launch outcomes for CCAFS SCL-40 Launch Site

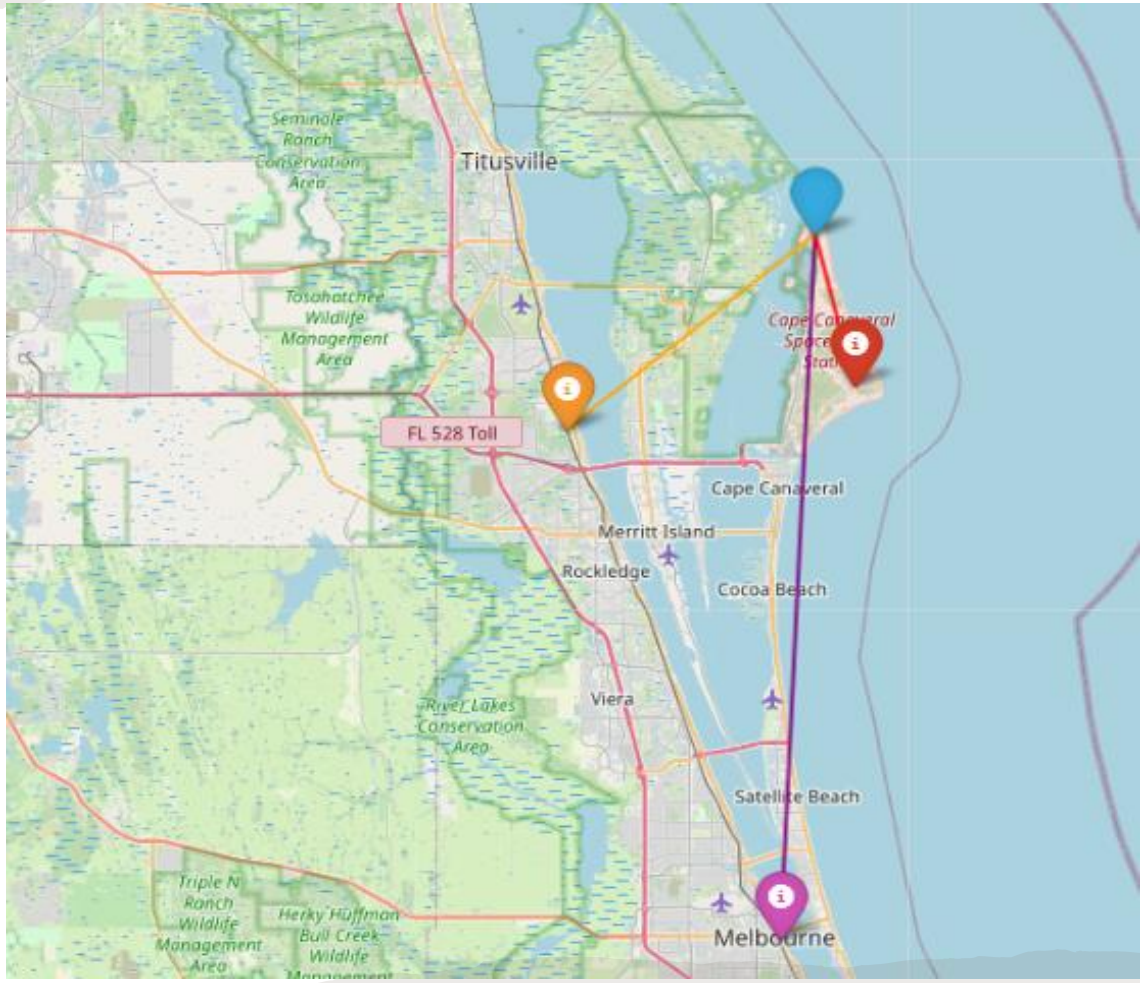
The red color labels indicate failure while the green indicates success. So there are 4 failures and 3 success launch outcome here.

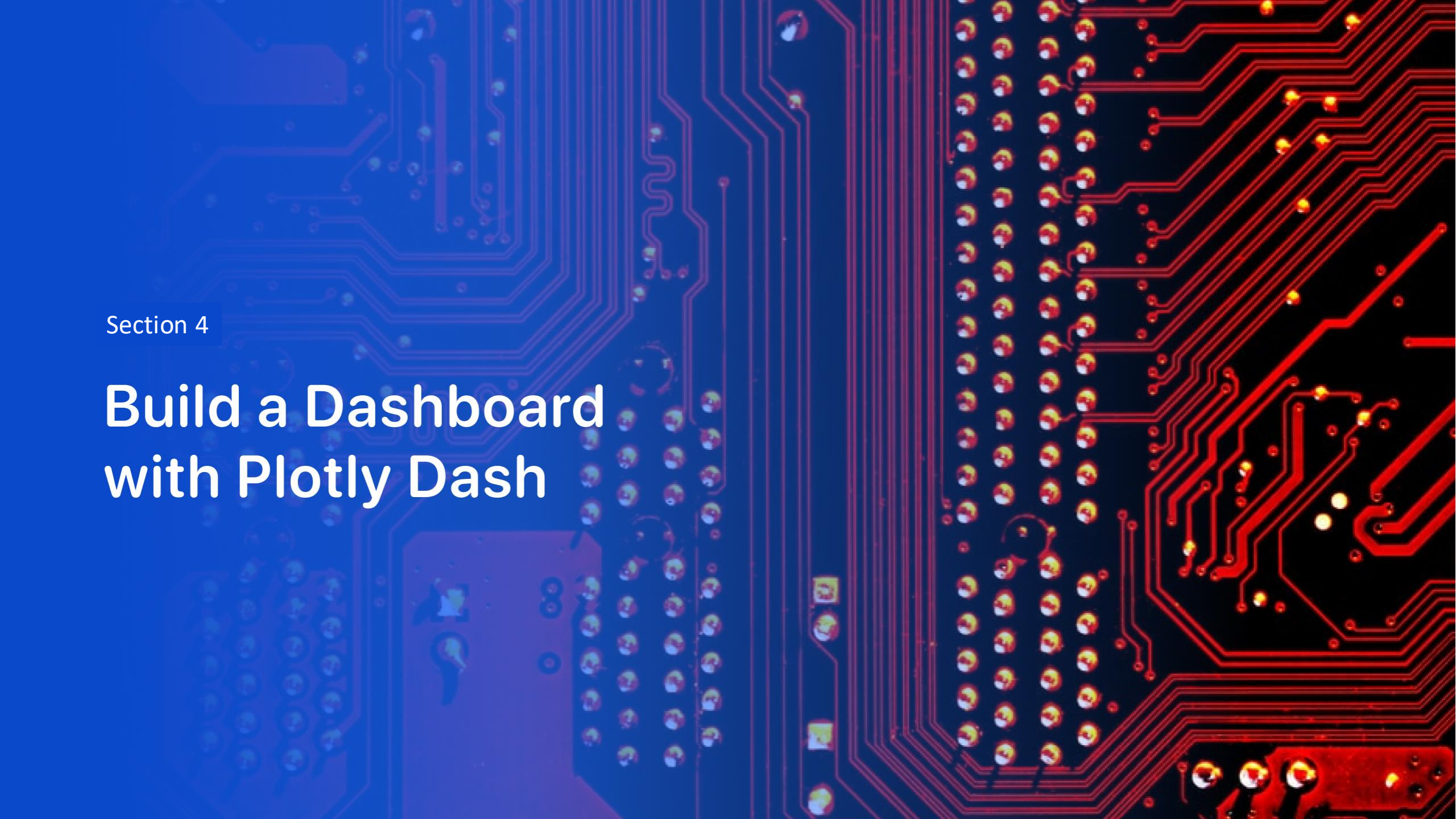
Launch Site Proximity to City, Railway and Highway

The launch site (CCAF SCL-40) location has proximity with a railway station, a highway and a city. The Site's distance to some infastructure are as follows:

- Closes Railwa Distance = 23.58km
- Closest City = 53.91km
- closest Highway = 11.85km

The approximate distances provides for easier logistic operation as well as safety.

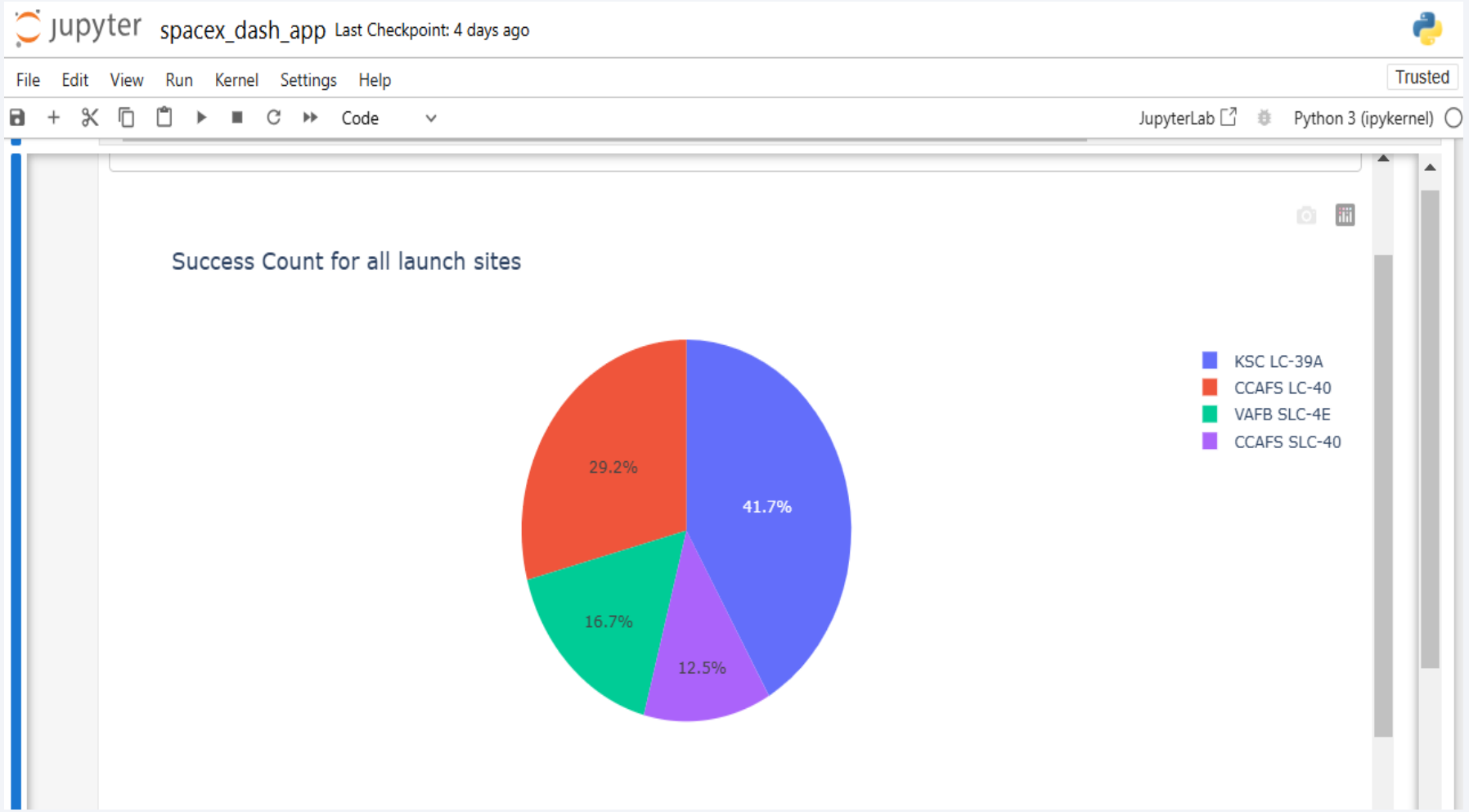



Section 4
Build a Dashboard with Plotly Dash

Dashboard 1: Launch Success Count

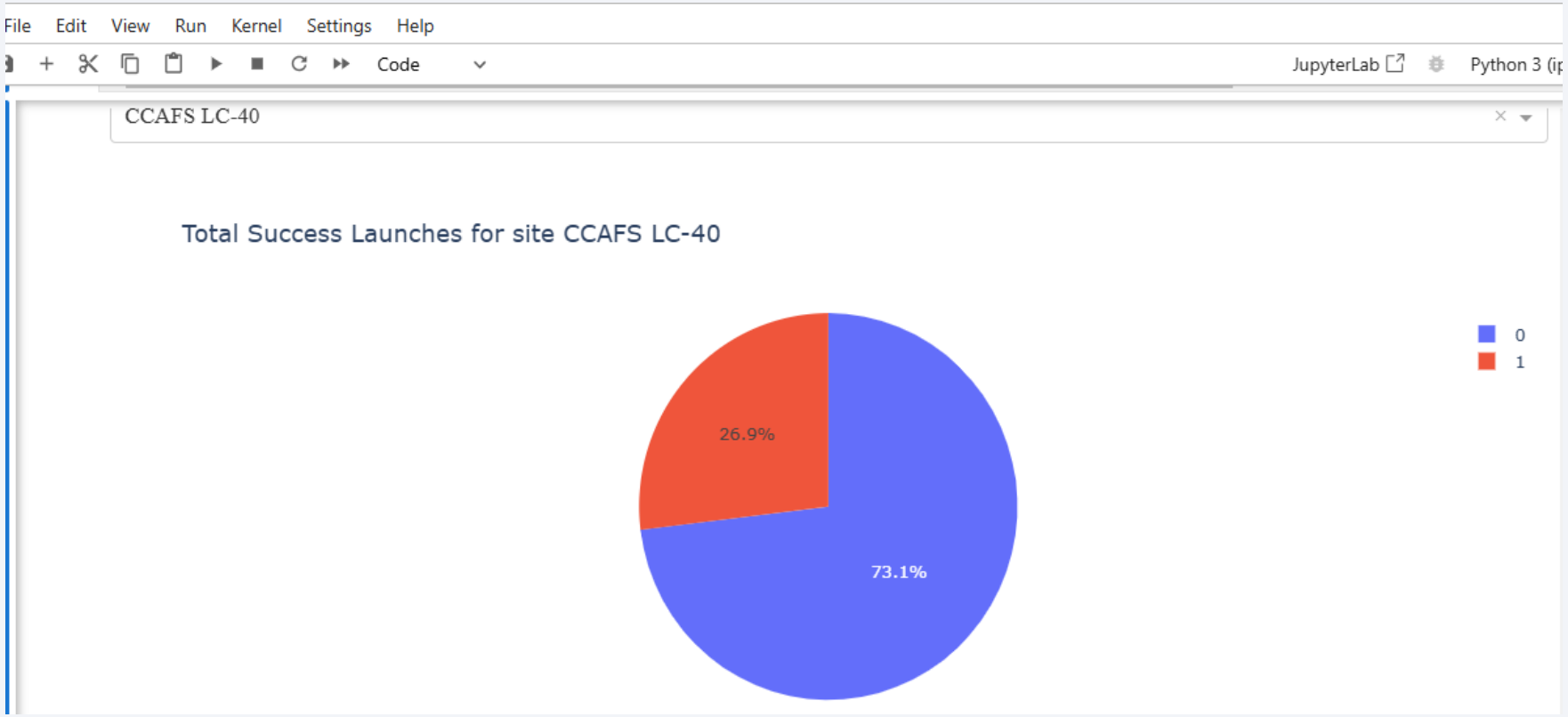
The figure on the right shows the screenshot of launch success count for all sites.

- KSC LC-39A has the most success launch (47.7%); while the site with the least successful Launch is CCAFS SLC-40



Success Launch Ration

In terms of success ration, CCAFS LC-40 is the launch site with highest launch success ratio

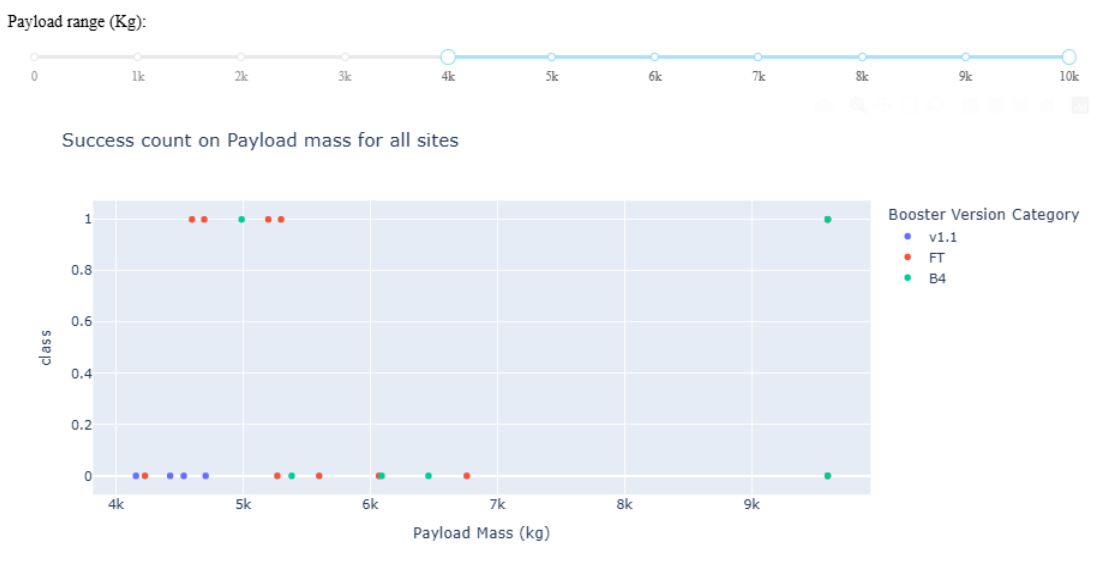


Success Count on Payload Mass

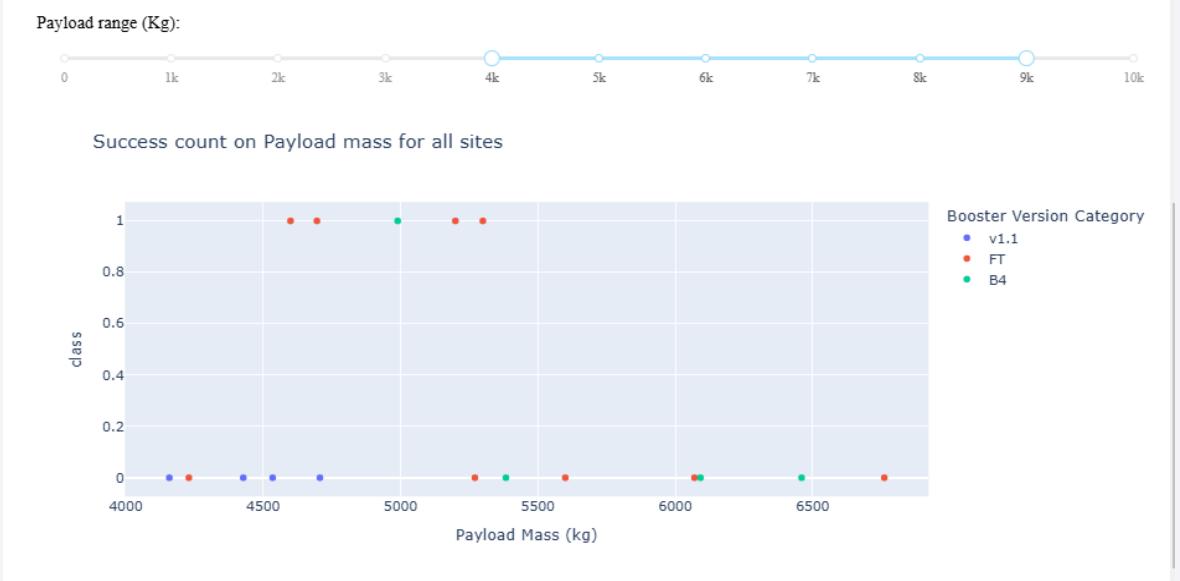
We can see that FT recorded success with lower payload mass. This is followed by B4 that recorded a success even with higher payload mass of over 9000kg

With a low payload mass of 1000kg, FT Booster Version has more success count. This is followed by the B4 of which recorded success with payload mass of over 9000kg.

Success Count on Payload Mass



FT oster Version still leads in the success count as payload mass reaches above 3000kg.



The success count keeps declining as the payload mass reaches above 9000kg.

CONCLUSION

Payload Mass has significant impact on launch success count. The data indicate a declining success count with an increase in payload mass.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- We can see the Decision Tree Classifier happens to be the model with the highest accuracy score of 8.7.
- The Score is more closer to 1 than the others. This means that the model is making fewer errors and performs better on the training data in terms of accuracy

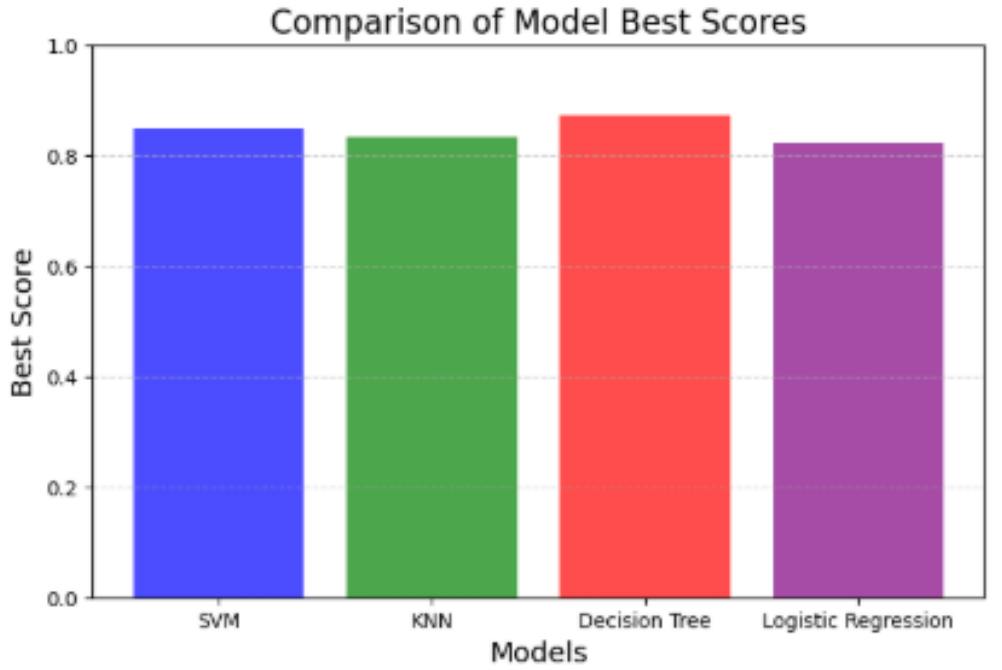
Visualize the built model accuracy for all built classification models, i

[72]:

```
import matplotlib.pyplot as plt

# Labels and scores
models = ['SVM', 'KNN', 'Decision Tree', 'Logistic Regression']
scores = [svm_cv.best_score_, KNN_cv.best_score_, tree_cv.best_score_, lr_cv.best_score_]

# Plotting the bar chart
plt.figure(figsize=(8, 5))
plt.bar(models, scores, color=['blue', 'green', 'red', 'purple'], alpha=0.7)
plt.title('Comparison of Model Best Scores', fontsize=16)
plt.xlabel('Models', fontsize=14)
plt.ylabel('Best Score', fontsize=14)
plt.ylim(0, 1)  # Assuming scores are between 0 and 1
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.show()
```



Confusion Matrix

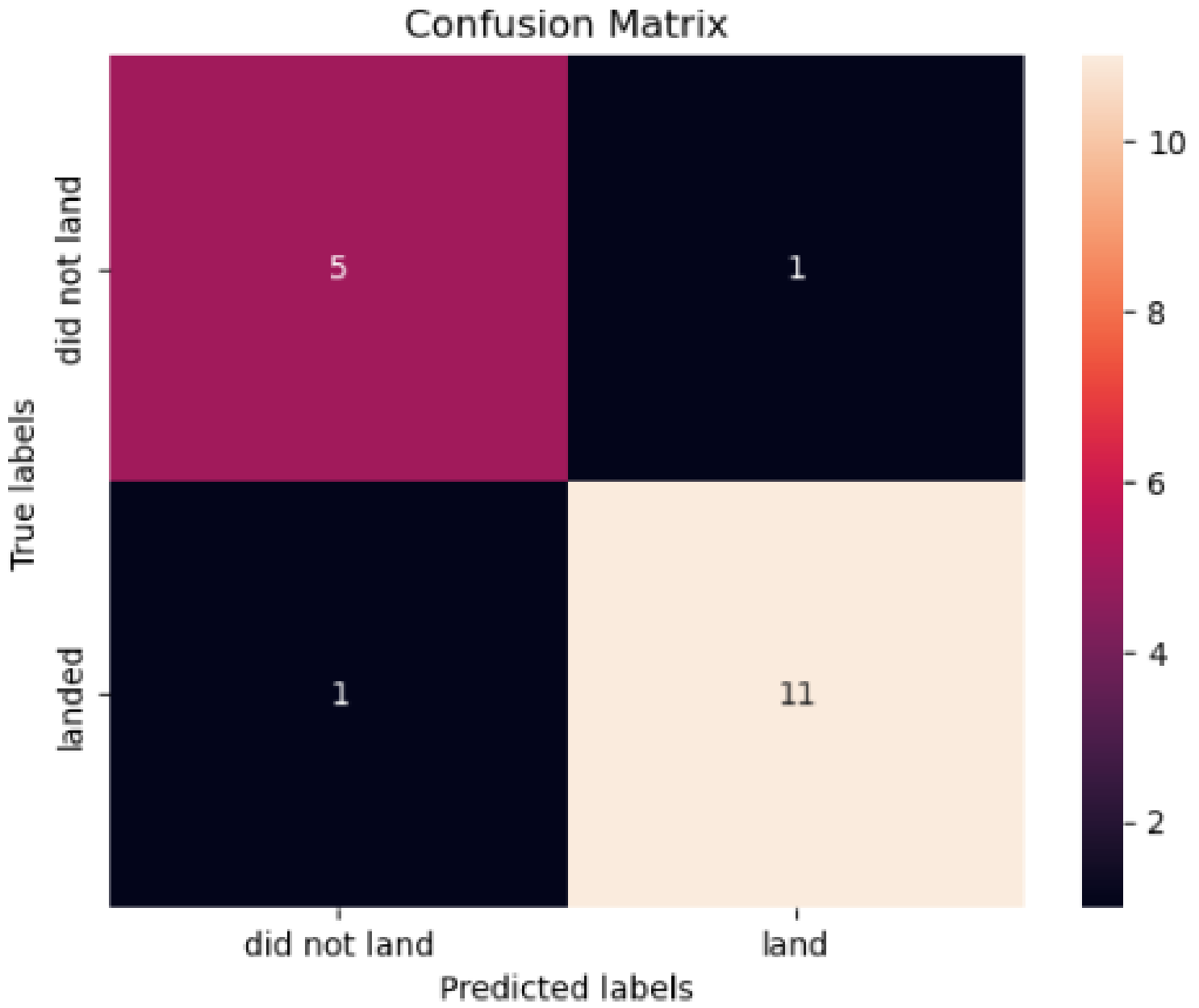
Examining the confusion matrix, we see that the Decision Tree Classifier can distinguish between the different classes.

Overview:

True Postive - 11 (True label is landed, Predicted label is also landed)

False Postive - 5 (True label is not landed, Predicted label is landed)

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



Conclusions

Analysis of the rocket launch data shows that SpaceX's rocket launch cost is justified due to the following findings;

1. **The rate of successful launch landing has increased steadily since 2013. This suggest exponential rise in learning curve**
2. **Launch sites proximity has improved the rate of successful landing as it ensures less manouvers, fuel efficiency and landing precision among other things**
3. **Landing Success rates is also influenced by other factors such as, payload mass, orbit type, Flight Number, Booster Version, etc.,.**
4. **On landing prediction analysis using 4 prediction models, i found out that accuracy score is close to 1; meaning more successful landing is predicted compared to failed landing, thus, saving the first stage. This justifies the cost of Space X's rocket launches**

Appendix

Kindly view all relevant assets such as Python code snippets, SQL queries, charts, Notebook outputs, or data sets I created during this project in the link below:

https://github.com/Mohzsmanuscripts/IBM-Applied-Data-Science-Capstone/tree/main

Thank you

Thank you!