IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Derrick Juda>
<2022/09/18>

Outline

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

Executive Summary

- Summary of methodologies

  -Data Collection through API

  -Data Collection with Web Scraping

  -Data Wrangling

  -Exploratory Data Analysis with SQL

  -Exploratory Data Analysis with Data Visualization

  -Interactive Visual Analytics with Folium

  -Machine Learning Prediction

- Summary of all results

  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context
- Problems you want to find answers

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logistic regression, SVM, Decision Tree and K-Nearest Neighbour are deployed

Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
  - the data then was cleaned, checked for missing values and replaced where necessary.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

- Using get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

- The link to the notebook is https://github.com/DerrickJuda/coursera_capstone_project/blob/61214fe5f484908555f7ee6ca834d05cd26a6f09/jupyter-labs-spacex-data-collection-api_ed.ipynb

API Collection

Now let's start requesting rocket launch data from SpaceX API with the following URL:

```
[6]: spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
[7]: response = requests.get(spacex_url)
```

Check the content of the response

```
[ ]: print(response.content)
```

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

JSON Normalising

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
[11]: # Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
[12]: # Get the head of the dataframe
data.head()
```

Cleaning to get Falcon 9 data only

Finally we will remove the Falcon 1 launches keeping only the Falcon 9 launches. Filter the data dataframe using the `BoosterVersion` column to only keep the Falcon 9 launches. Save the filtered data to a new dataframe called `data_falcon9`.

```
[ ]: # Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data_launch[data_launch['BoosterVersion'] != 'Falcon 1']
data_falcon9
```

Now that we have removed some values we should reset the FlgihtNumber column

```
[ ]: data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

Data Collection - Scraping

- Applying web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- Parsing the table and converted it into a pandas dataframe.
- The link to the notebook is https://github.com/chuksoo/IBM-Data-Science-Capstone-SpaceX/blob/main/Data%20Collection%20with%20Web%20Scraping.ipynb.

BeautifulSoup application

To keep the lab tasks consistent, you will be asked to scrape the data from a snapshot of the `List of Falcon 9 and Falcon Heavy launches` Wikipage updated on `9th June 2021`

[4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

Next, request the HTML page from the above URL and get a `response` object

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

[6]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
data  = requests.get(static_url).text
```

Create a `BeautifulSoup` object from the HTML `response`

[7]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data,"html.parser")
```

Print the page title to verify if the `BeautifulSoup` object was created properly

[8]:
```
# Use soup.title attribute
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

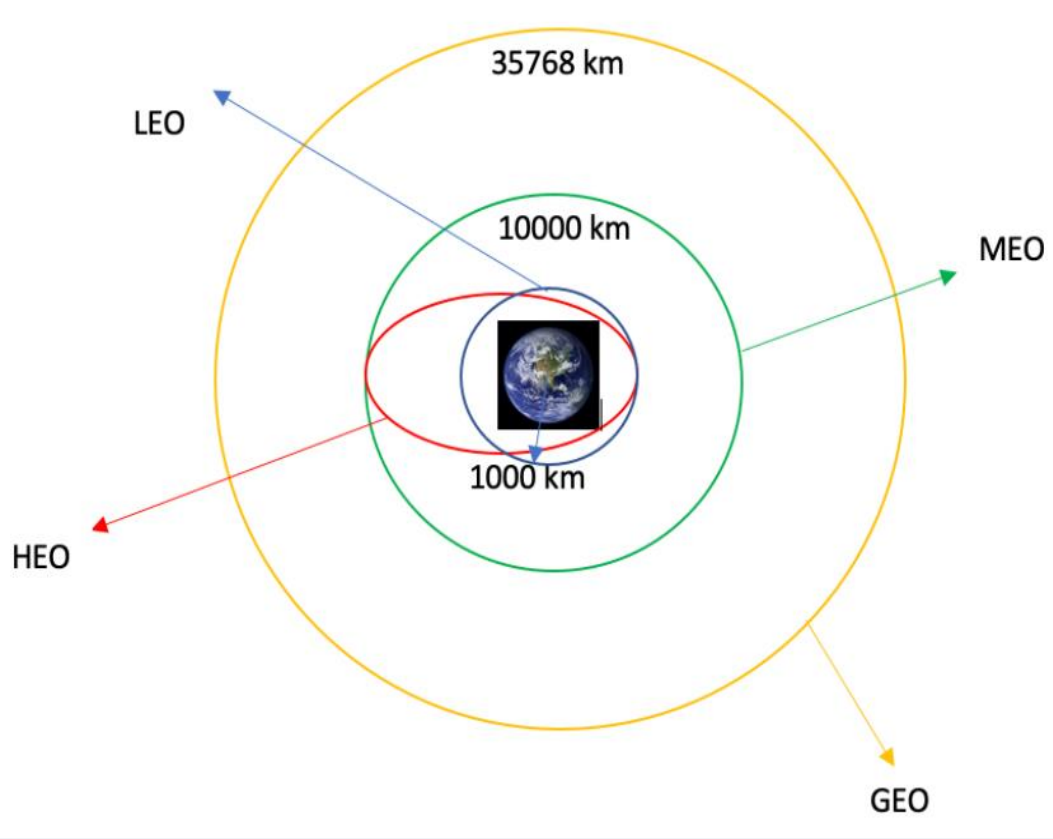
Converting to dataframe

Next, we just need to iterate through the `<th>` elements and apply the provided `extract_column_from_header()` to extract column name one by one

[11]:
```
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 != None and len(name) > 0):
        column_names.append(name)
```

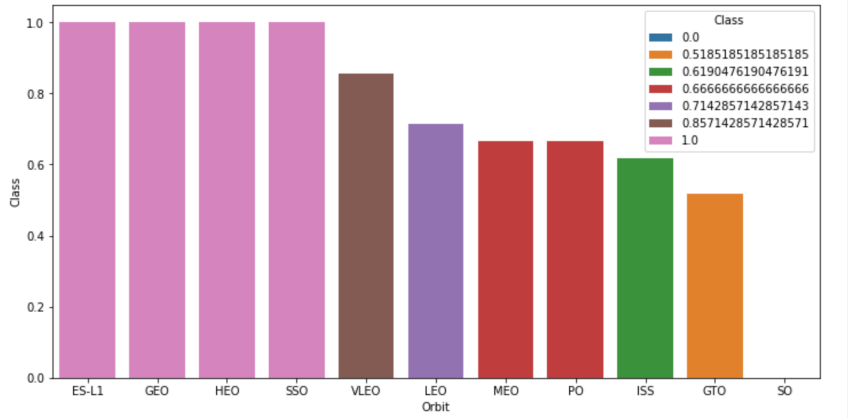
Data Wrangling



- Performing exploratory data analysis and determining the training labels.
- Calculating the number of launches at each site, and the number and occurrence of each orbits
- Creating landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is https://github.com/DerrickJuda/coursera_capstone_project/blob/61214fe5f484908555f7ee6ca834d05cd26a6f09/jupyter-labs-webscraping_ed.ipynb

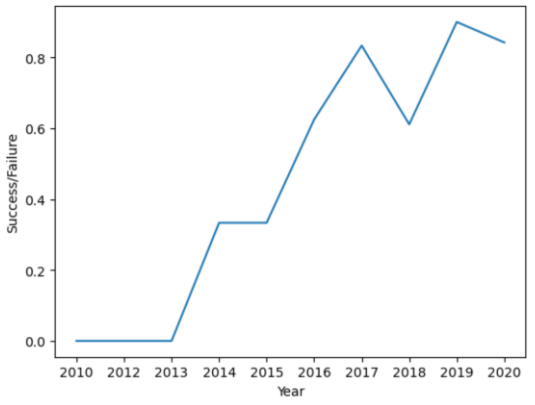
EDA with Data Visualization

```
# HINT use groupby method on Orbit column and get the mean of Class column
grouped_orbits = df.groupby(by=['Orbit'])['Class'].mean().sort_values(ascending=False).reset_index()
fig, ax=plt.subplots(figsize=(12,6))
ax = sns.barplot(x="Orbit",y="Class",data=grouped_orbits,hue='Class', dodge=False)
```



"ES-L1, GEO, HEO, SSO, VLEO orbits have high success rate"

```
In [14]:
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
plt.plot(average_by_year["Year"],average_by_year["Class"])
plt.xlabel("Year")
plt.ylabel("Success/Failure")
plt.show()
```



"Launch success rate exceeds 80% in 2020"

- Exploring the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.
- The link to the notebook is https://github.com/DerrickJuda/coursera_capstone_project/blob/61214fe5f484908555f7ee6ca834d05cd26a6f09/jupyter-labs-eda-dataviz_ed.ipynb

EDA with SQL

- Loading the SpaceX dataset into a SQLite without leaving the jupyter notebook.
- Applying EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook is https://github.com/DerrickJuda/coursera_capstone_project/blob/61214fe5f484908555f7ee6ca834d05cd26a6f09/jupyter-labs-eda-sql-coursera_sqllite_ed.ipynb

Build an Interactive Map with Folium

- Marking all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- Assigning the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- Calculating the distances between a launch site to its proximities. We answered some question for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.

Build a Dashboard with Plotly Dash

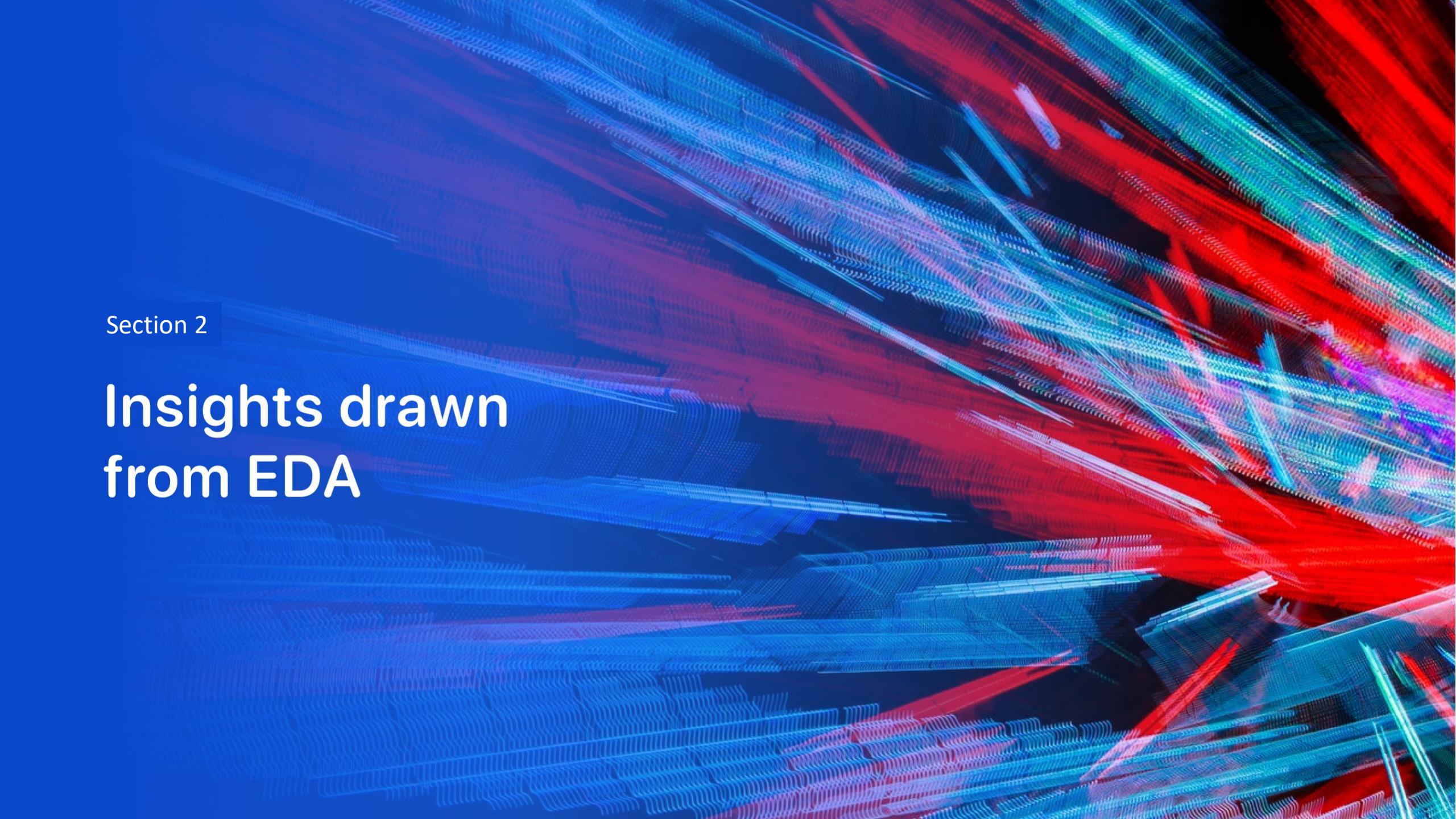
- Building an interactive dashboard with Plotly dash
- Plotting pie charts showing the total launches by a certain sites
- Utilising scatter graph to show the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The link to the notebook is https://github.com/DerrickJuda/coursera_capstone_project/blob/36a398ac7add6130972b4ff152c4dda9634742f1/spacex_dash_app_ed.py

Predictive Analysis (Classification)

- Loading the data using numpy and pandas, transformed the data, split our data into training and testing.
- Building different machine learning models and tune different hyperparameters using GridSearchCV.
- Using accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- Finding the best performing classification model.
- The link to the notebook is https://github.com/DerrickJuda/coursera_capstone_project/blob/36a398ac7add6130972b4ff152c4dda9634742f1/SpaceX_Machine%20Learning%20Prediction_Part_5_ed.ipynb

Results

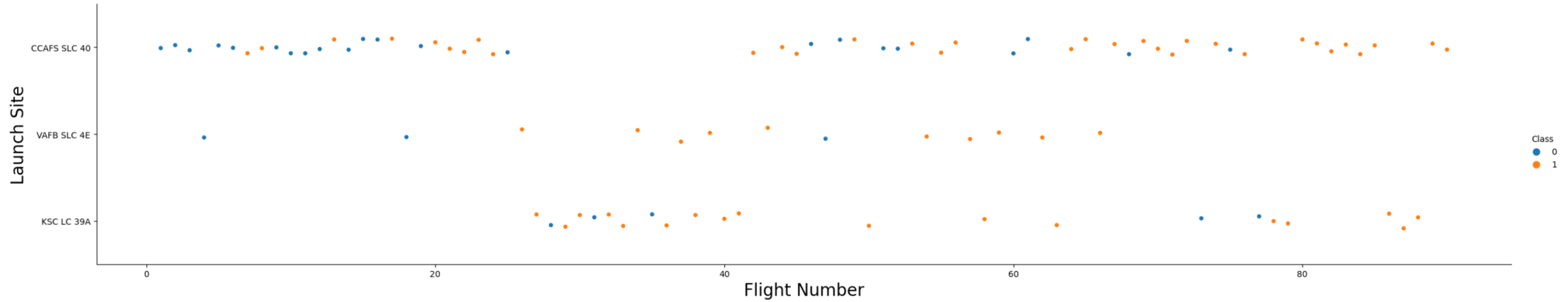
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results


Section 2
Insights drawn from EDA

Flight Number vs. Launch Site

- We can see that the larger the flight amount at a launch site, the greater the success rate at a launch site.

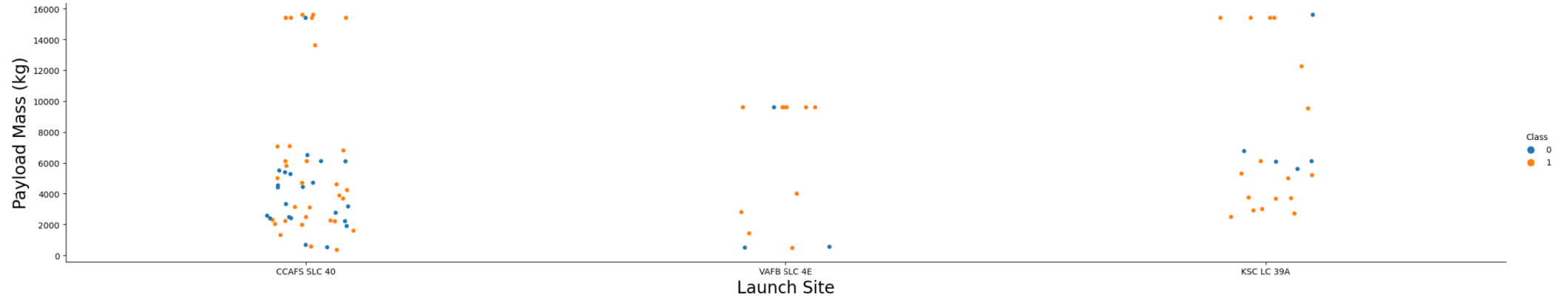
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(x="FlightNumber",y="LaunchSite",hue='Class',data=df, aspect=5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Payload vs. Launch Site

- We can see that the greater the payload mass for CCAFS SLC 40 launch site, the higher the success rate for a rocket. KSC LC 39A launch site works better for any payload but needs more tests to confirm

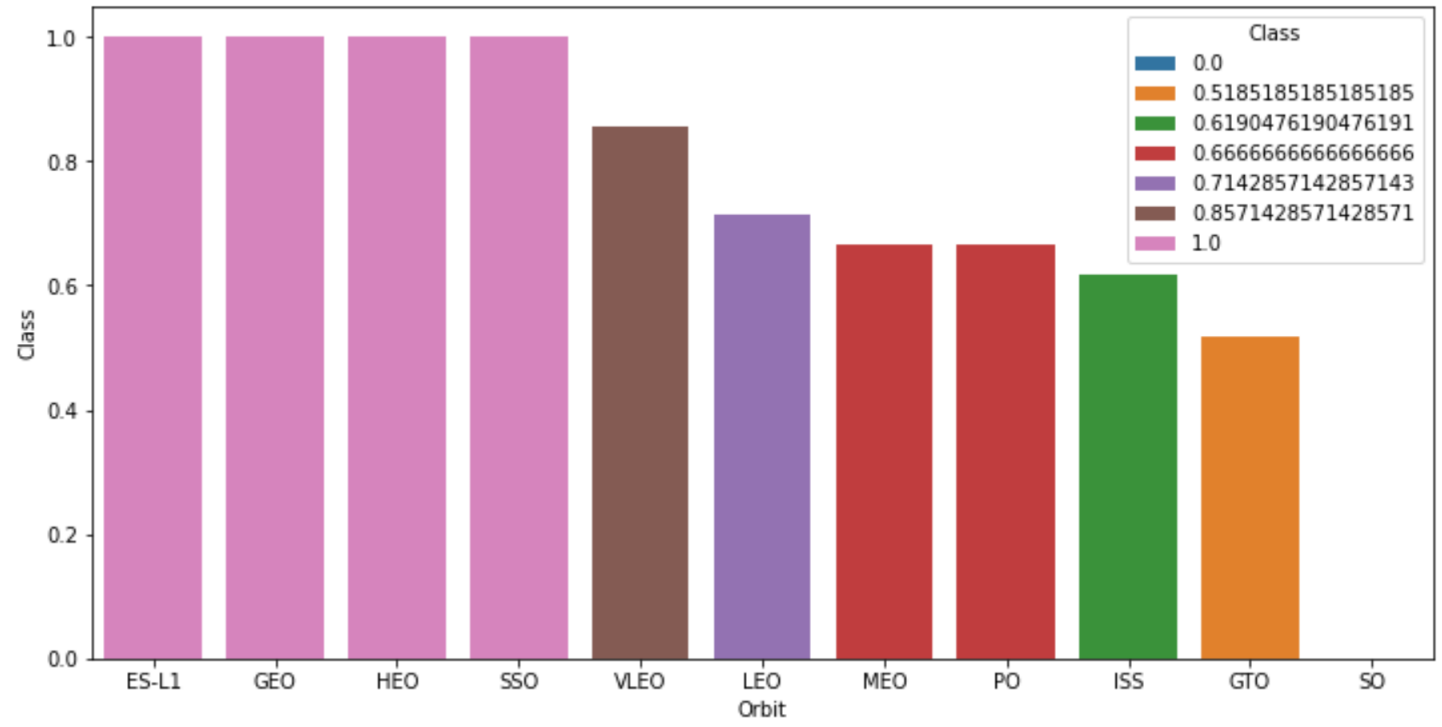
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="PayloadMass",x="LaunchSite",hue='Class',data=df, aspect=5)
plt.ylabel("Payload Mass (kg)",fontsize=20)
plt.xlabel("Launch Site",fontsize=20)
plt.show()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



Success Rate vs. Orbit Type

- We can see that ES-L1, GEO, HEO, SSO and VLEO orbits have high success rate

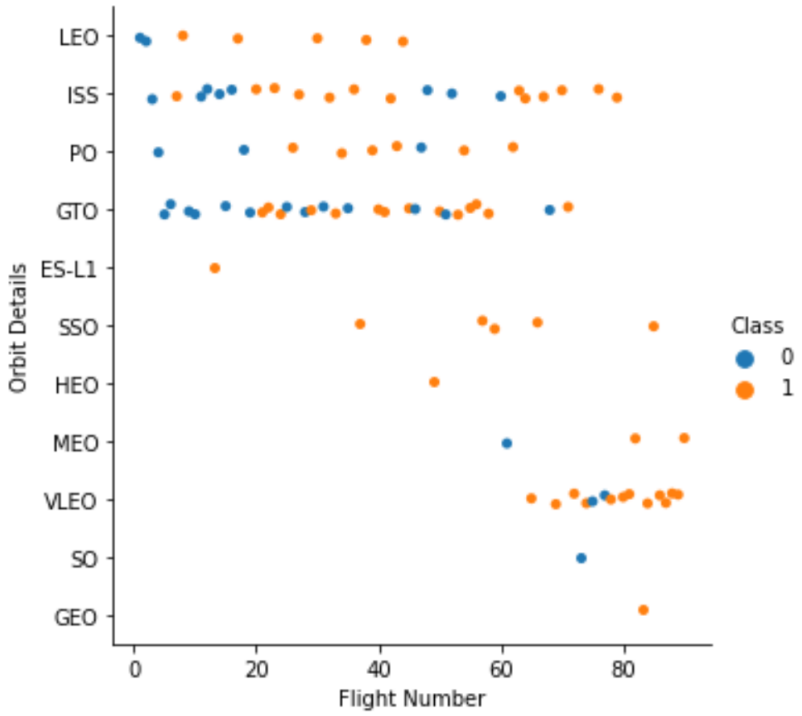
```
# HINT use groupby method on Orbit column and get the mean of Class column
grouped_orbits = df.groupby(by=['Orbit'])['Class'].mean().sort_values(ascending=False).reset_index()
fig, ax=plt.subplots(figsize=(12,6))
ax = sns.barplot(x="Orbit",y="Class",data=grouped_orbits,hue='Class', dodge=False)
```



Flight Number vs. Orbit Type

- We can see that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

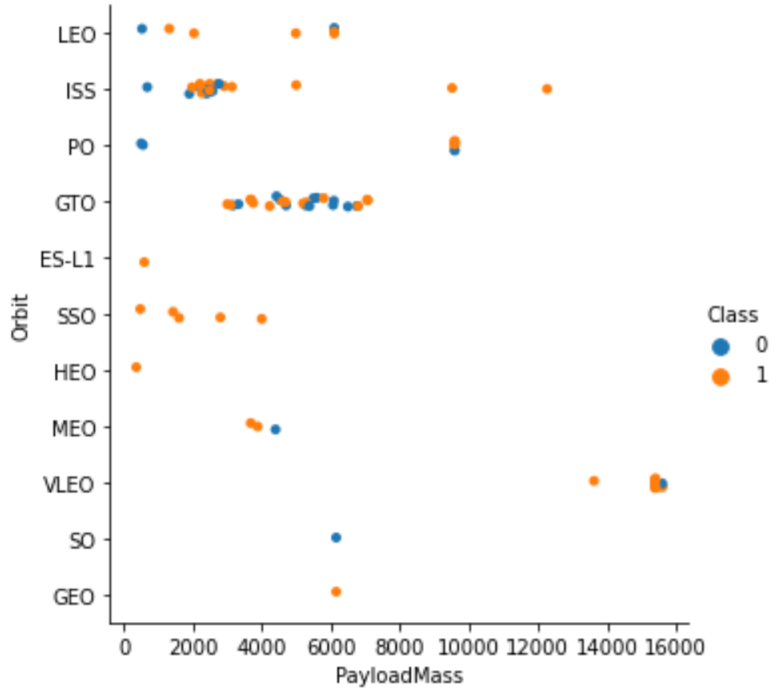
```
sns.catplot(x='FlightNumber',y='Orbit',data=df,hue='Class')
plt.xlabel('Flight Number')
plt.ylabel('Orbit Details')
plt.show()
```



Payload vs. Orbit Type

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

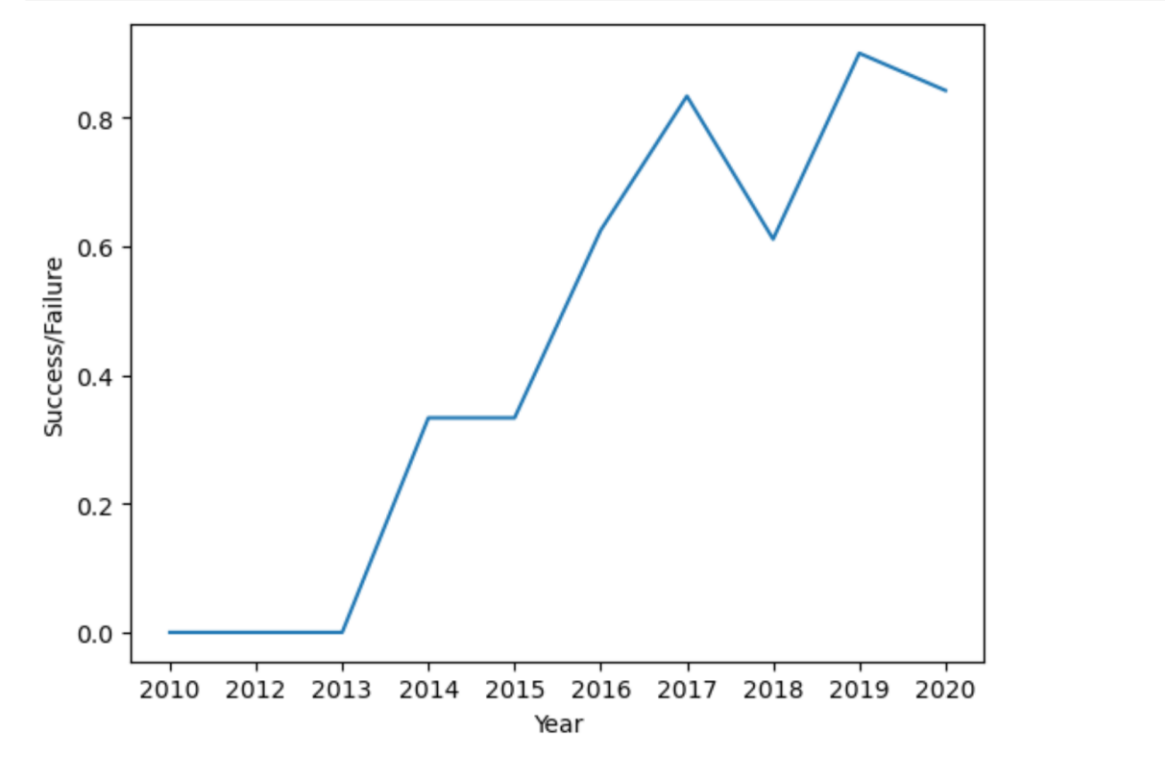
```
sns.catplot(x='PayloadMass',y='Orbit',data=df,hue='Class')
plt.xlabel('PayloadMass')
plt.ylabel('Orbit')
plt.show()
```



Launch Success Yearly Trend

- we can observe that success rate since 2013 has been increasing to more than 80% in 2020

```
plt.plot(average_by_year["Year"],average_by_year["Class"])
plt.xlabel("Year")
plt.ylabel("Success/Failure")
plt.show()
```



All Launch Site Names

- Using the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

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

```
[13]: %sql SELECT distinct(Launch_Site) FROM SPACEXTBL;
```

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

[13]:

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

Launch Site Names Begin with 'CCA'

- Using the query below to display 5 records where launch sites begin with `CCA`

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

[24]:
```
%sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

[24]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Calculating the total payload carried by boosters from NASA as 45596 using the query below

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

```
[17]: %sql SELECT sum(PAYLOAD_MASS__KG_) as payload_mass_kg FROM SPACEXTBL WHERE Customer == 'NASA (CRS)';
```

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

[17]:

payload_mass_kg
45596

Average Payload Mass by F9 v1.1

- Calculating the average payload mass carried by booster version F9 v1.1 as 2928.4 using the query below

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

```
[18]: %sql SELECT avg(PAYLOAD_MASS__KG_) as avg_payload_mass_kg FROM SPACEXTBL WHERE Booster_Version == 'F9 v1.1';
```

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

[18]:

avg_payload_mass_kg
2928.4

First Successful Ground Landing Date

- Observing that the dates of the first successful landing outcome on ground pad was 1st May 2017

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

Hint:Use min function

```
[23]: %sql SELECT min(Date) as launch_date FROM SPACEXTBL WHERE [Landing _Outcome] == 'Success (ground pad)';
```

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

[23]:

launch_date
01-05-2017

Successful Drone Ship Landing with Payload between 4000 and 6000

- Using the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

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

```
[45]: %sql SELECT Booster_Version FROM SPACEXTBL WHERE [Landing _Outcome] == 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000
```

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

[45]:

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

Total Number of Successful and Failure Mission Outcomes

- Using **DISTINCT, COUNT AND GROUP BY** to the MissionOutcome column to determine the total number of launch success and failure.

List the total number of successful and failure mission outcomes

```
[48]: %sql SELECT distinct(Mission_Outcome) FROM SPACEXTBL;
```

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

[48]:

Mission_Outcome
Success
Failure (in flight)
Success (payload status unclear)
Success

```
[47]: %sql SELECT count(Mission_Outcome) FROM SPACEXTBL GROUP BY Mission_Outcome;
```

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

[47]:

count(Mission_Outcome)
1
98
1
1

Boosters Carried Maximum Payload

- Determining the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

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

[49]:
```
%sql select BOOSTER_VERSION as boosterversion from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL);
```

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

[49]:

boosterversion
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

- Using a combinations of the **WHERE** and **substr** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

```
[51]: %sql SELECT substr(Date, 4, 2),MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXTBL where substr(Date,7,4)='2015';
 * sqlite:///my_data1.db
Done.
```

[51]:

substr(Date, 4, 2)	Mission_Outcome	Booster_Version	Launch_Site
01	Success	F9 v1.1 B1012	CCAFS LC-40
02	Success	F9 v1.1 B1013	CCAFS LC-40
03	Success	F9 v1.1 B1014	CCAFS LC-40
04	Success	F9 v1.1 B1015	CCAFS LC-40
04	Success	F9 v1.1 B1016	CCAFS LC-40
06	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

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

- Selecting Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- Applying the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

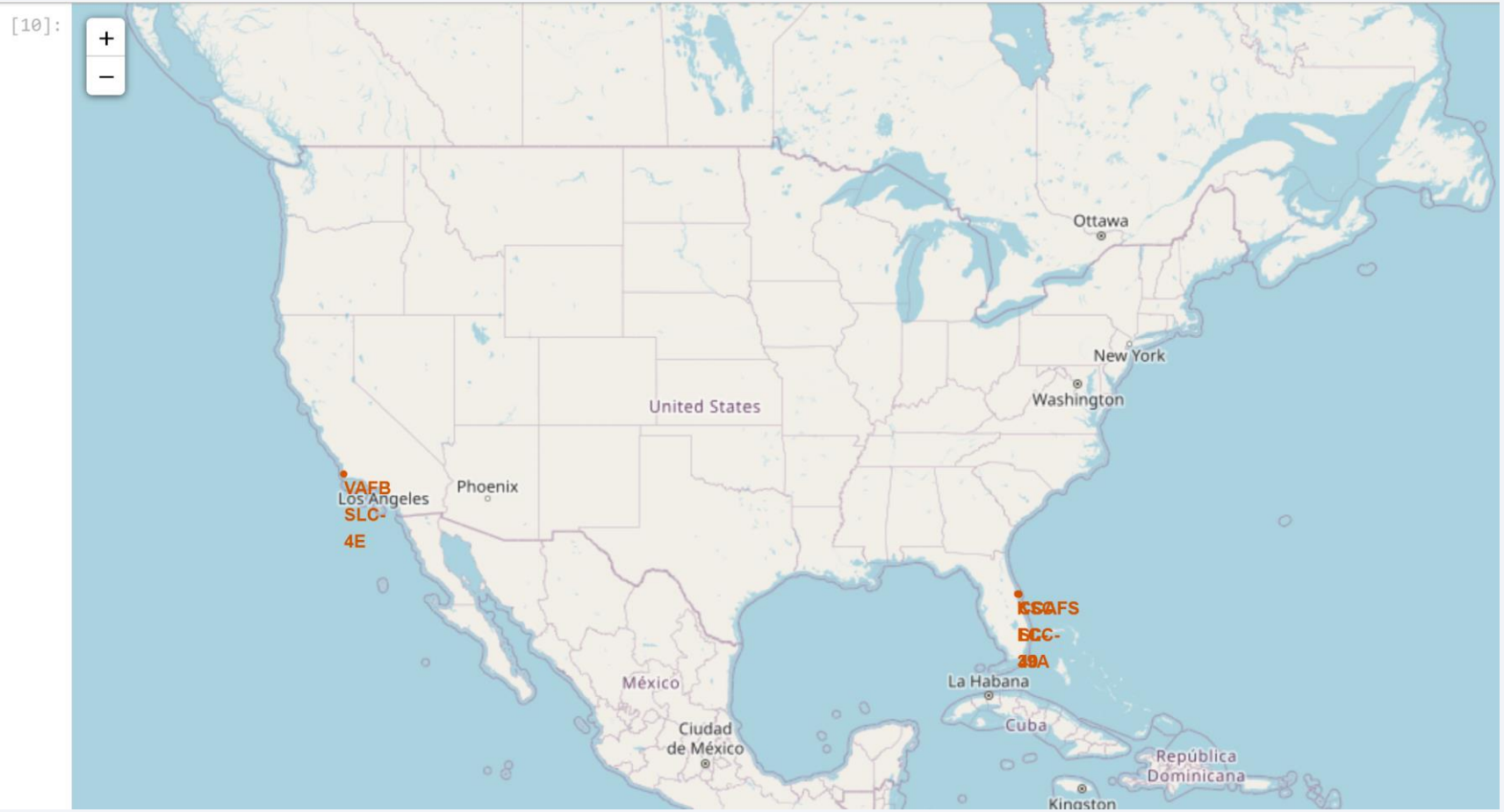
[10]:

Landing_Outcome	count(Landing_Outcome)
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1


Section 3
Launch Sites
Proximities Analysis

All launch sites map markers

- The SpaceX launch sites are located both along the East and West Coast of USA in Florida and California



Markers showing launch sites with color labels

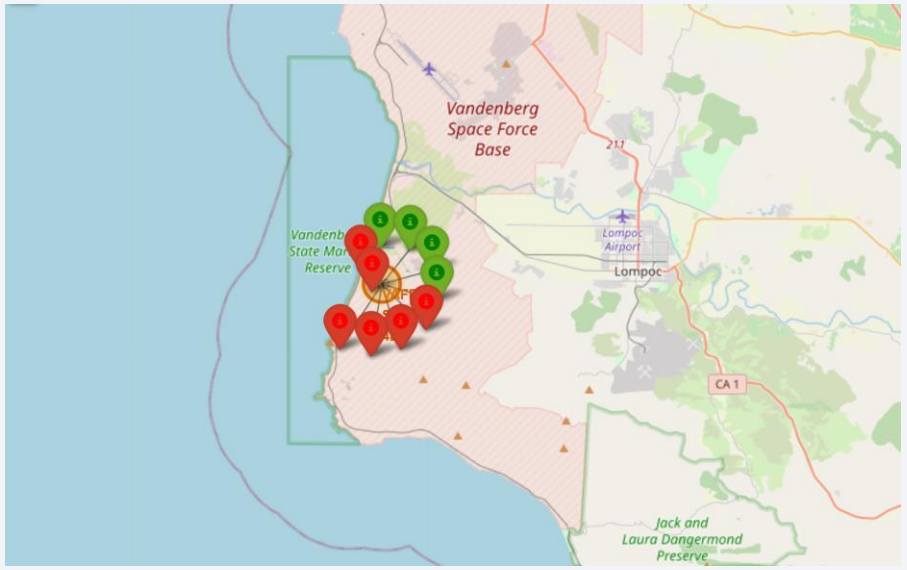
- Green markers show successful launches and red markers show fail launches
- Florida Launch Sites

KSC LC-39A

CCAFS LC-40

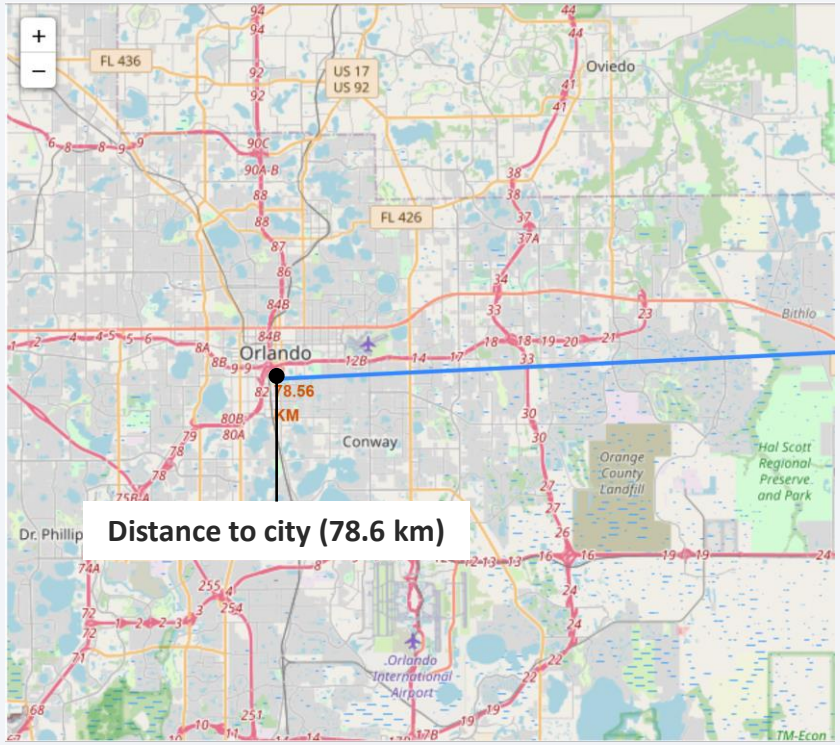
CCAFS SLC-40

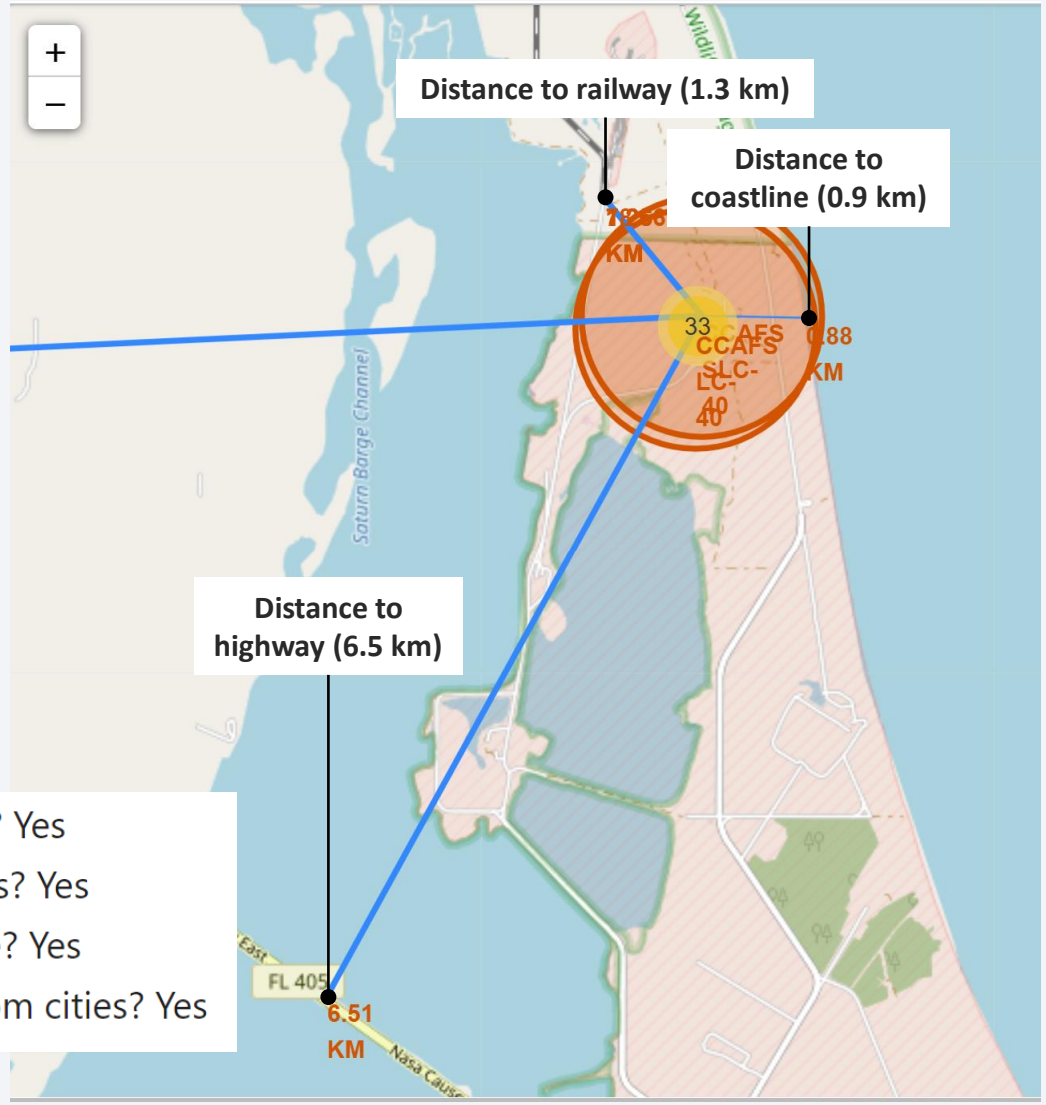
- California Launch Sites



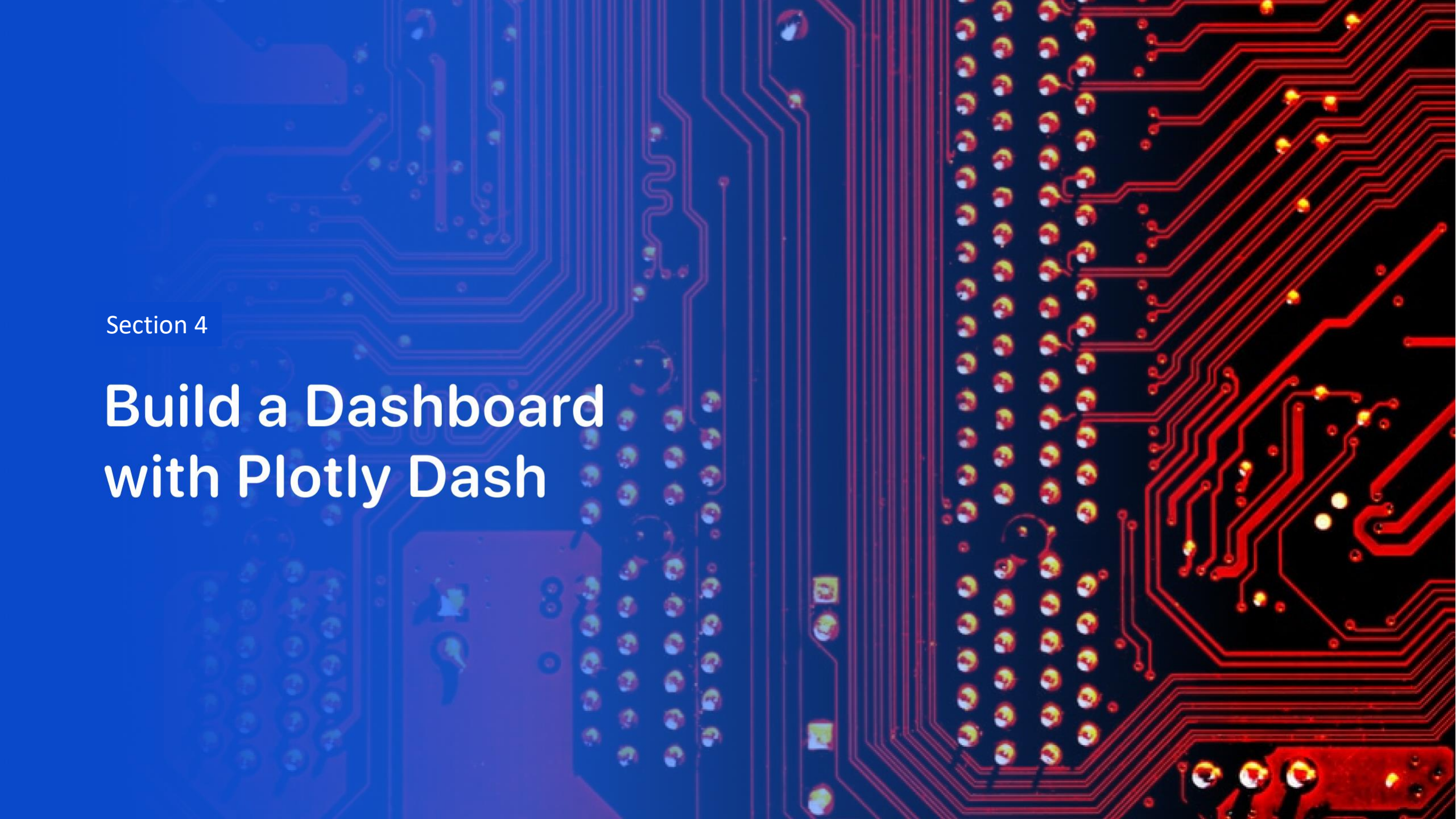
VAFB SLC-4E

Launch Site distance to landmarks



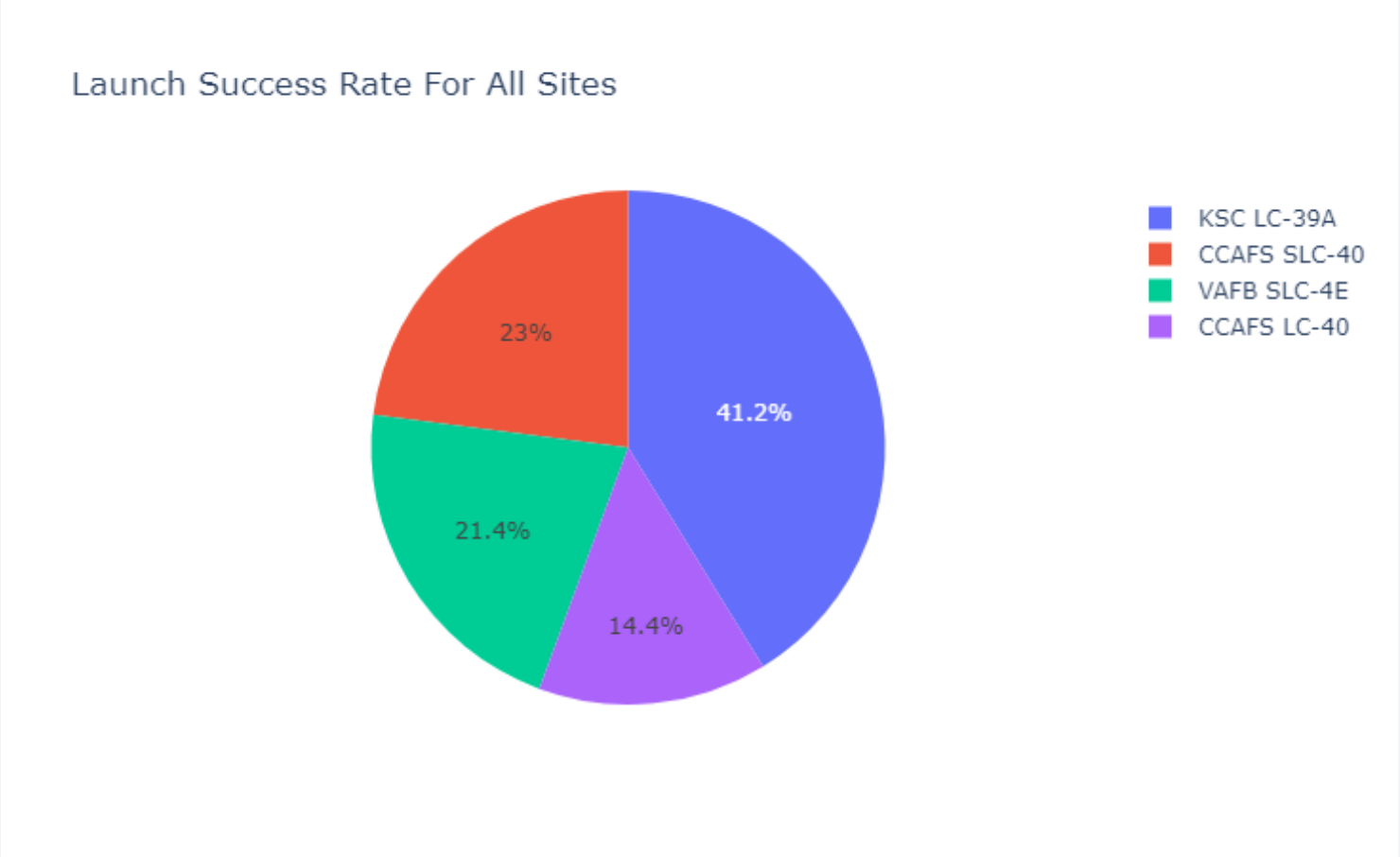


- Are launch sites in close proximity to railways? Yes
- Are launch sites in close proximity to highways? Yes
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes


Section 4
Build a Dashboard with Plotly Dash

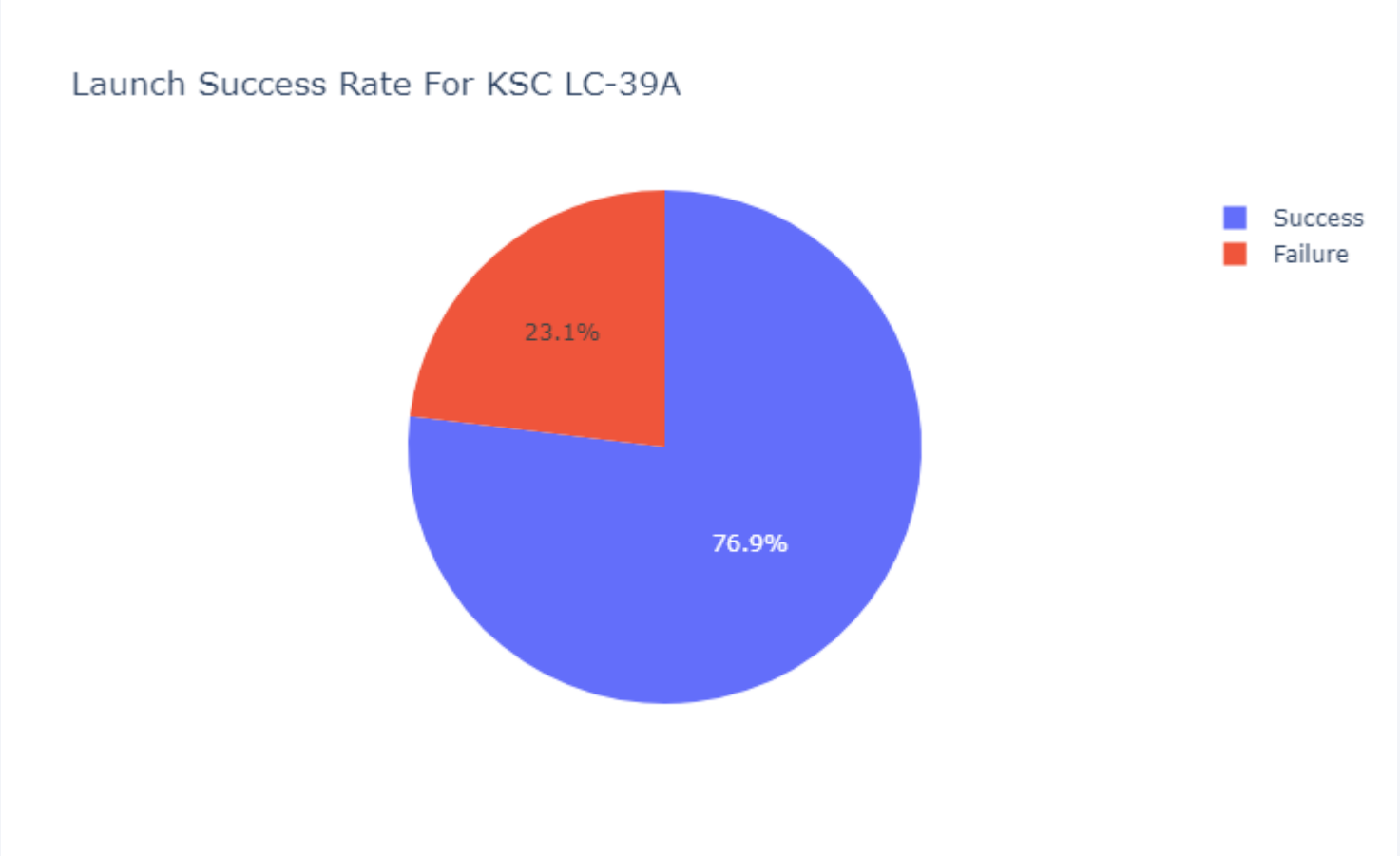
Success percentage achieved by each launch site

- We can see that KSC LC-39A is the most successful launch site



Launch site with the highest launch success ratio

- KSC LC-39A had 76.9% success rate and 23.1% failure rate



Scatter plot of Payload vs Launch Outcome for all sites

- With different payload selected in the range slider, we can see that the success rates for launches with lighter payload is higher than the ones with heavier payload.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

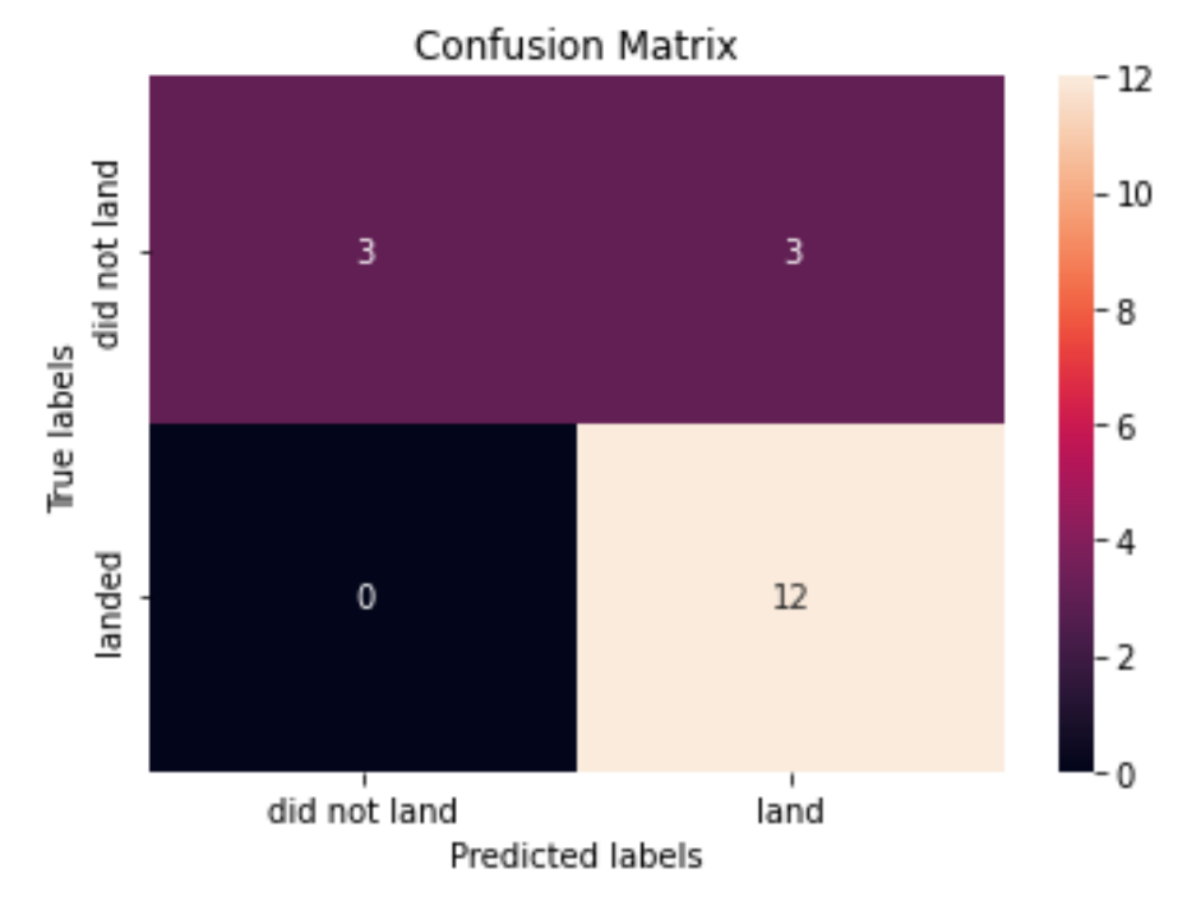
- All models have the same accuracy at 83.3%

[112]:
```
print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
print('Accuracy for K nearest neighbors method:', knn_cv.score(X_test, Y_test))
```

```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.8333333333333334
Accuracy for K nearest neighbors method: 0.8333333333333334
```

Confusion Matrix

- The confusion matrix shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- KNN, SVM, Decision Tree and Logistic Regression can be used for this task as all of them giving the same accuracy percentage.

Thank you!