IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Denis Boeck
26.07.2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data was collected using the SpaceX API & web scraping (Wikipedia)
  - Exploratory Data Analysis
  - Data Wrangling
  - Data Visualization / Interactive Visual Analytics
  - Machine Learning Prediction (Classification Modelling)
- Summary of all results
  - Data could be collected through various sources – all of which available to the public
  - Through EDA best features could be recognized to predict successful landing outcomes
  - ML Classification Modelling showed characteristics to have a successful landing

Introduction

- Project background and context
  - The aim is the prediction of a successful landing of Falcon 9 since the rocket is a lot cheaper in comparison to competitors.
- Problems you want to find answers
  - Can you determine which rocket features will lead to a successful outcome?
  - What is the correlation between different features concerning successful or unsuccessful landings
  - Which are best features you must keep in line to have a successful landing program?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX REST API
  - Web Scraping SpaceX data on Wikipedia
- Perform data wrangling
  - Unnecessary columns were dropped
  - Columns were prepared using One Hot Encoding
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash

Methodology

Executive Summary

- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
  - First step was using the SpaceX REST API (https://api.spacexdata.com/v4/launches/past)
  - Next step was using available data from Wikipedia using web scraping (https://en.wikipedia.org/wiki/List_of_Falcon/_9/_and_Falcon_Heavy_launches)

Data Collection – SpaceX API

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

- Notebook link: [IBM_Data_Science-Capstone/IBM_Data_Science_Capstone-spacex-data-collection-api.ipynb at main · dnsbck/IBM_Data_Science-Capstone (github.com)]

1. Get request for rocket launch data using API

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

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

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Notebook link:
  [IBM_Data_Science-Capstone/IBM_Data_Science_Capstone-webscraping.ipynb at main · dnsbck/IBM_Data_Science-Capstone (github.com)]

1. Apply HTTP Get method to request the Falcon 9 rocket launch page

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

In [5]:
```python
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url)
html_data.status_code
```

Out[5]: 200

2. Create a BeautifulSoup object from the HTML response

In [6]:
```python
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text, 'html.parser')
```

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

In [7]:
```python
# Use soup.title attribute
soup.title
```

Out[7]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

3. Extract all column names from the HTML table header

In [10]:
```python
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

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

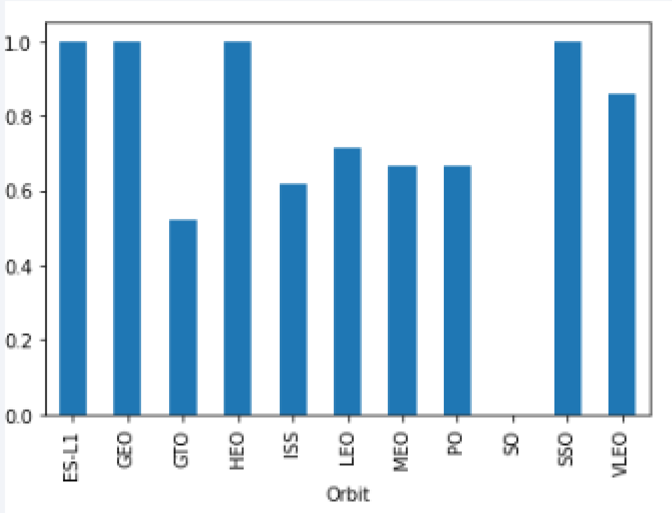
4. Create a dataframe by parsing the launch HTML tables
5. Export data to csv

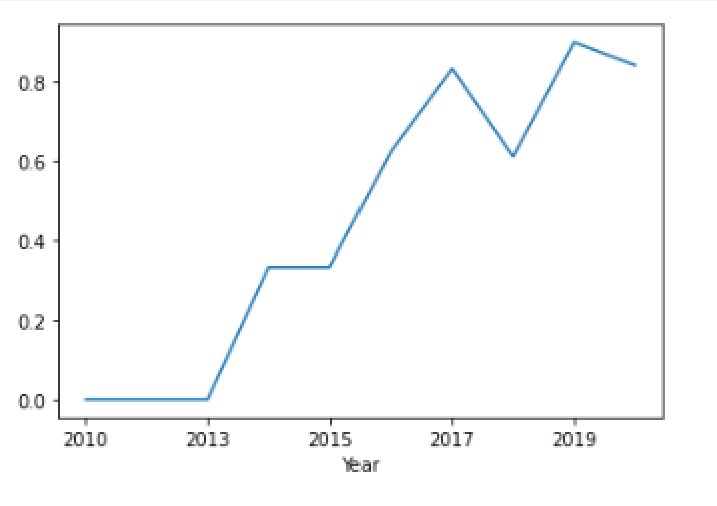
Data Wrangling

- At first EDA was performed and training labels were given
- Calculations were: # of launches per site, # and occurrence for orbits
- The landing outcome was calculated and labeled
- Notebook link:
  [IBM_Data_Science-Capstone/IBM_Data_Science_Capstone-spacex-Data wrangling.ipynb at main · dnsbck/IBM_Data_Science-Capstone (github.com)]

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts





- Notebook link:
  [IBM_Data_Science-Capstone/IBM_Data_Science_Capstone-eda-dataviz.ipynb at main · dnsbck/IBM_Data_Science-Capstone (github.com)]

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - Finding out distinct launch sites
  - Calculate total payload mass by “NASA (CRS)”
  - Calculate average payload mass by booster F9 v1.1
  - Group and calculate total number of successful and unsuccessful missions
- Notebook link: IBM_Data_Science-Capstone/IBM_Data_Science_Capstone-eda-sql-coursera_sqllite.ipynb at main · dnsbck/IBM_Data_Science-Capstone (github.com)

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
  - Markers, circles, lines and clusters were used
- Explain why you added those objects
  - Those objects were used to indicate points on the map for the launch sites, to highlight certain areas and to group events
- Notebook link:
  IBM_Data_Science-Capstone/IBM_Data_Science_Capstone-launch_site_location.ipynb at main · dnsbck/IBM_Data_Science-Capstone (github.com)

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
  - Pie charts show the total launches per launch site
  - Scatter plots show relationships between Outcome / Payload for different booster versions
- Notebook link: IBM_Data_Science-Capstone/IBM_Data_Science_Capstone-launch_site_location.ipynb at main · dnsbck/IBM_Data_Science-Capstone (github.com)

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
    - Data was loaded and used with Pandas and NumPy and split into a training set and a testing set
    - The best performing model was used and evaluated by parameters and accuracy
- Notebook link: [IBM_Data_Science-Capstone/IBM_Data_Science_Capstone-SpaceX_Machine Learning Prediction_Part_5.ipynb at main · dnsbck/IBM_Data_Science-Capstone (github.com)]

Results

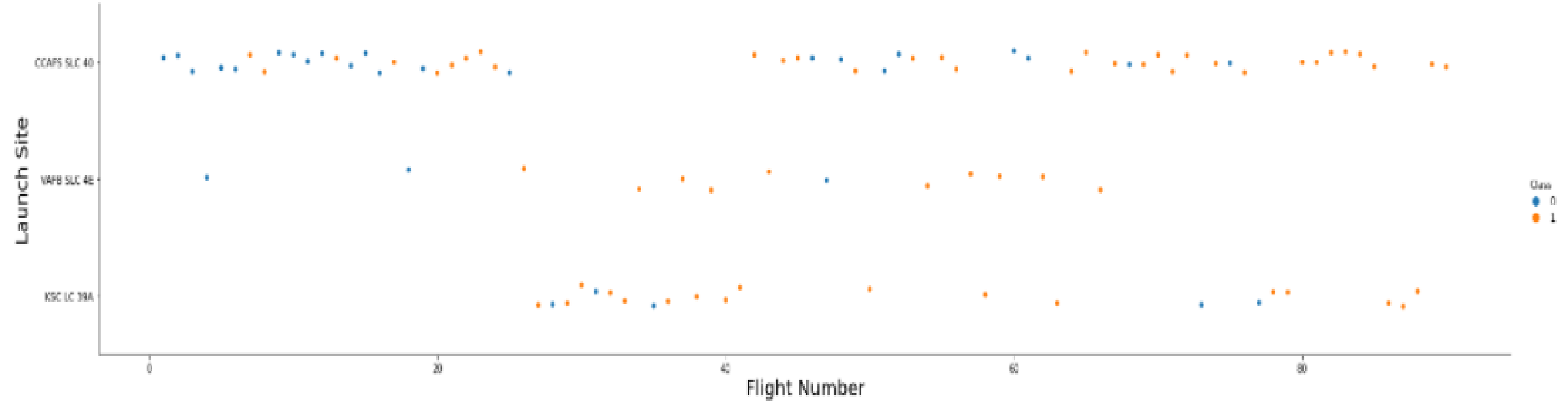
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

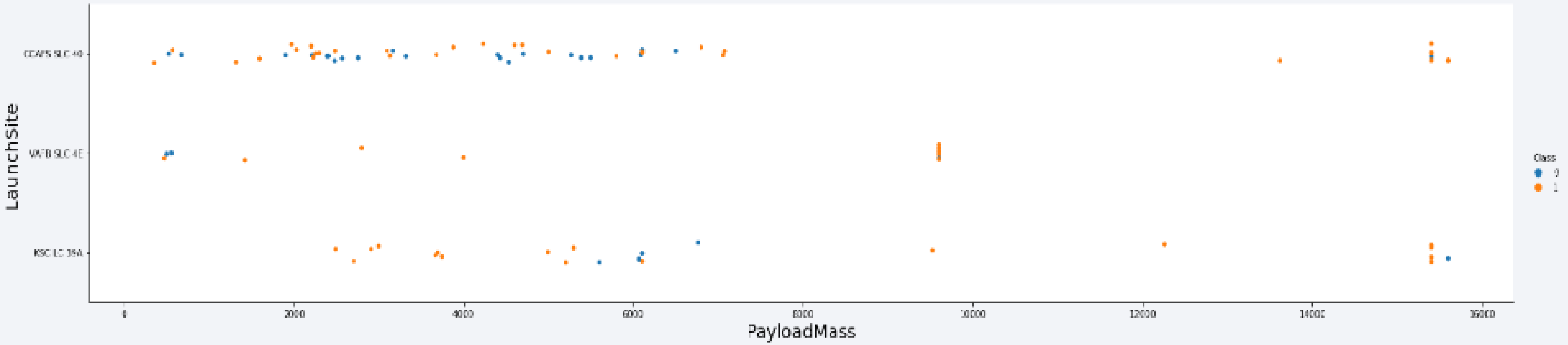
- Show a scatter plot of Flight Number vs. Launch Site



- Show the screenshot of the scatter plot with explanations

Payload vs. Launch Site

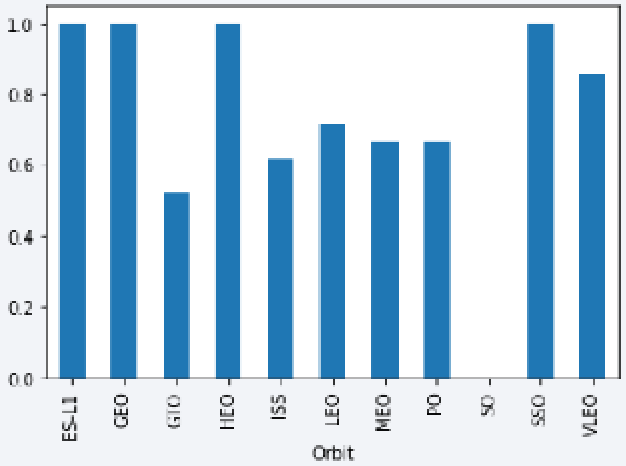
- Show a scatter plot of Payload vs. Launch Site



- Show the screenshot of the scatter plot with explanations

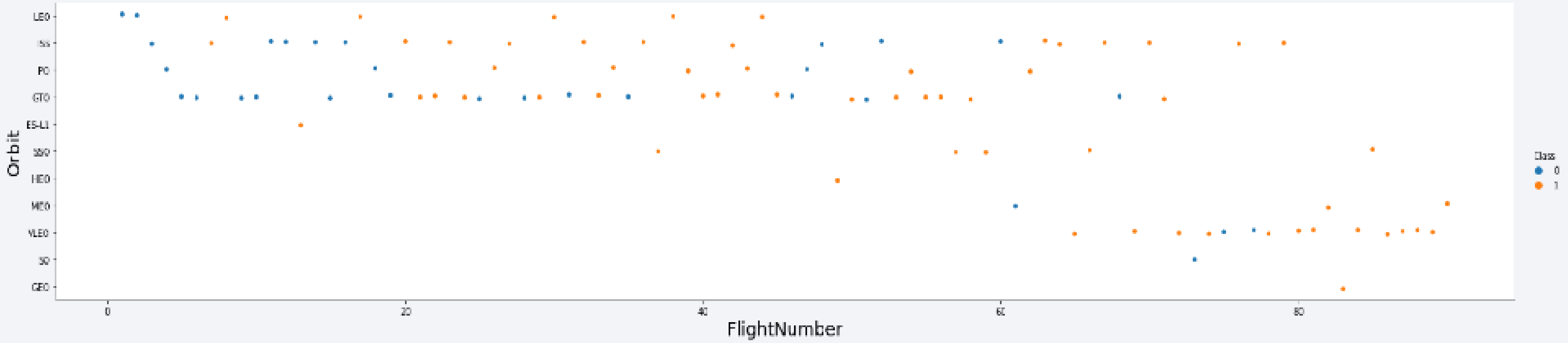
Success Rate vs. Orbit Type

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



Flight Number vs. Orbit Type

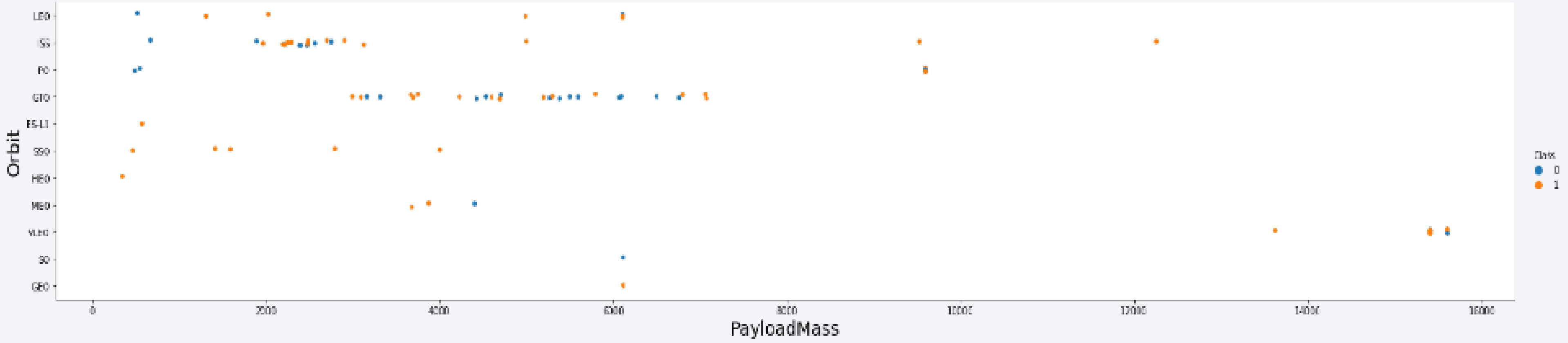
- Show a scatter point of Flight number vs. Orbit type



- Show the screenshot of the scatter plot with explanations

Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type

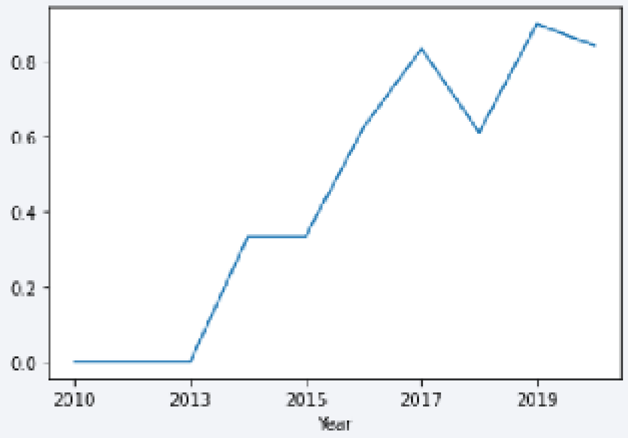


- Show the screenshot of the scatter plot with explanations

Launch Success Yearly Trend

- Show a line chart of yearly average success rate

- Show the screenshot of the scatter plot with explanations



All Launch Site Names

- Find the names of the unique launch sites
- Present your query result with a short explanation here

In [10]:
```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Present your query result with a short explanation here

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

In [11]:
```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Present your query result with a short explanation here

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

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here

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

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here

```
In [14]:
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

In [15]:
```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here

List the total number of successful and failure mission outcomes

In [16]:
```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

The total number of successful mission outcome is:

	successoutcome
0	100

The total number of failed mission outcome is:

Out[16]:

	failureoutcome
0	1

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here

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

In [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [18]:

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

- 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 descending order

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad))

In [19]:
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

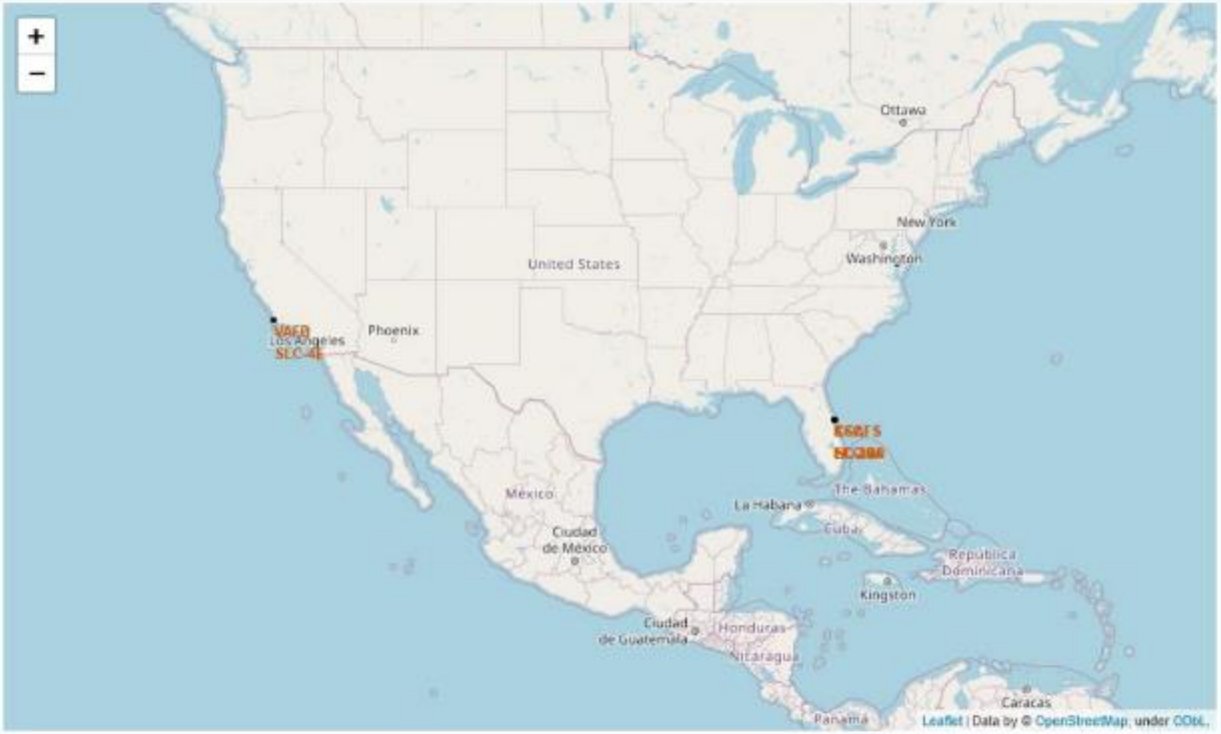
Out[19]:

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

Section 3

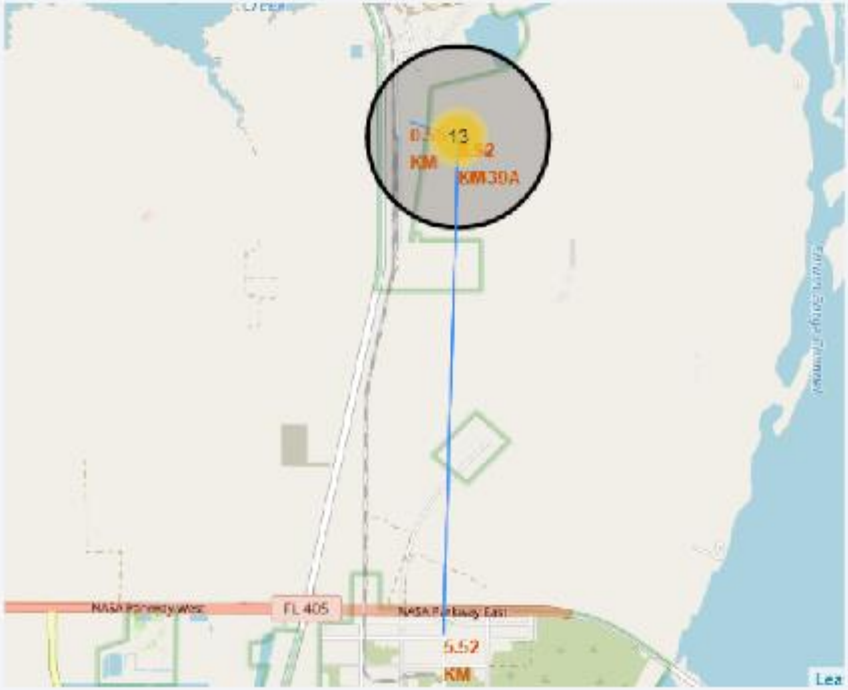
Launch Sites Proximities Analysis

Launch Sites global markers



Outcomes by Launch Site

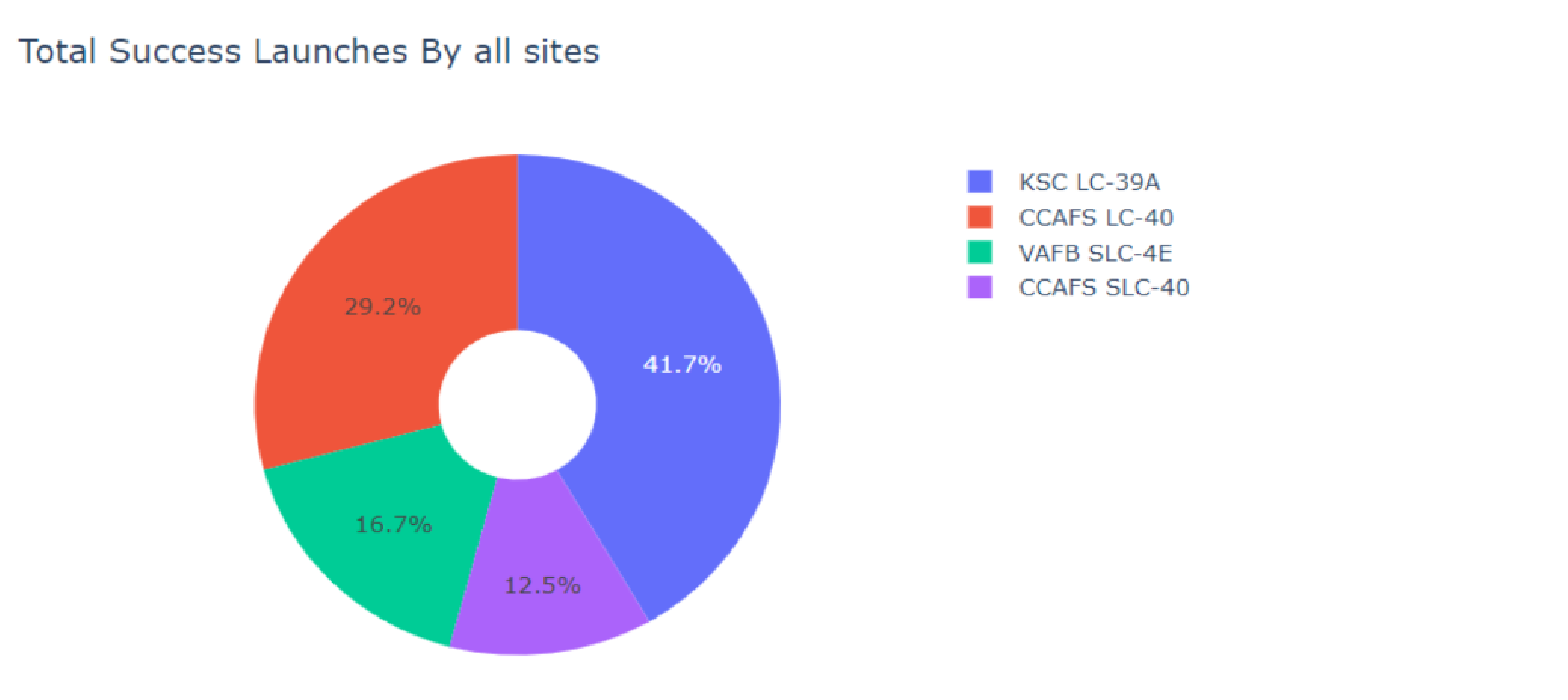
Logistics and Safety



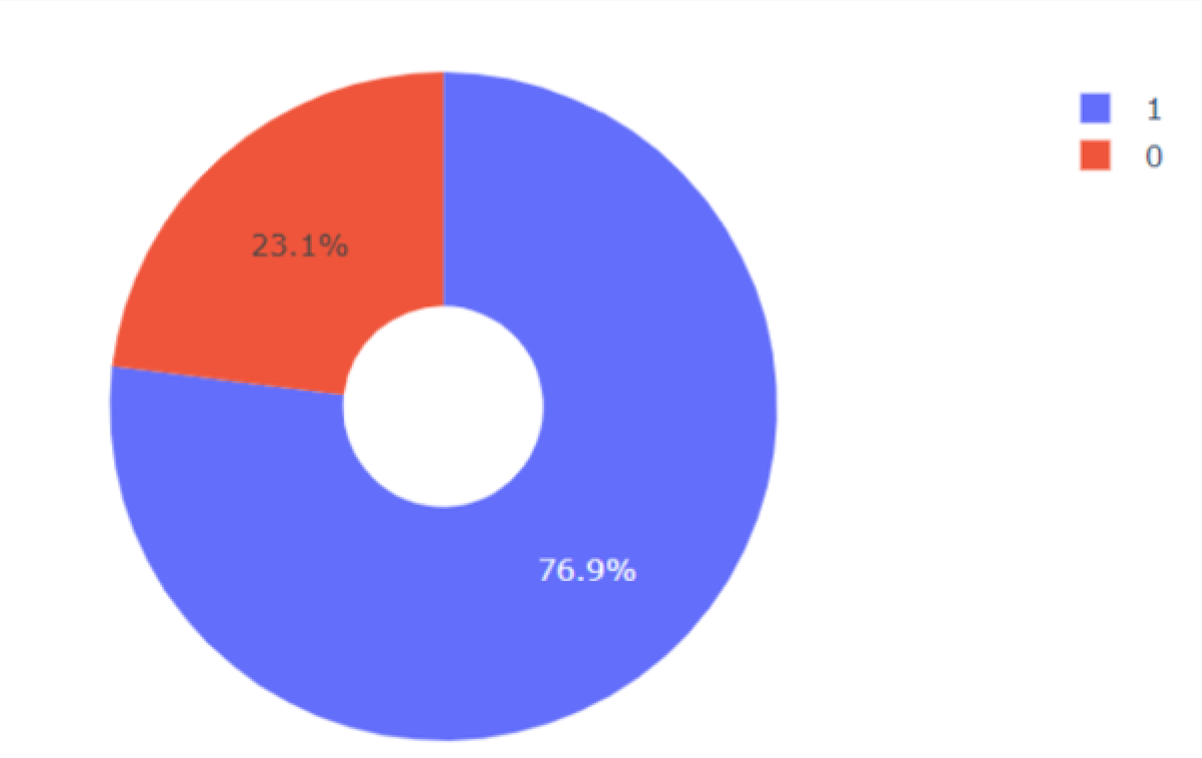
Section 4

Build a Dashboard with Plotly Dash

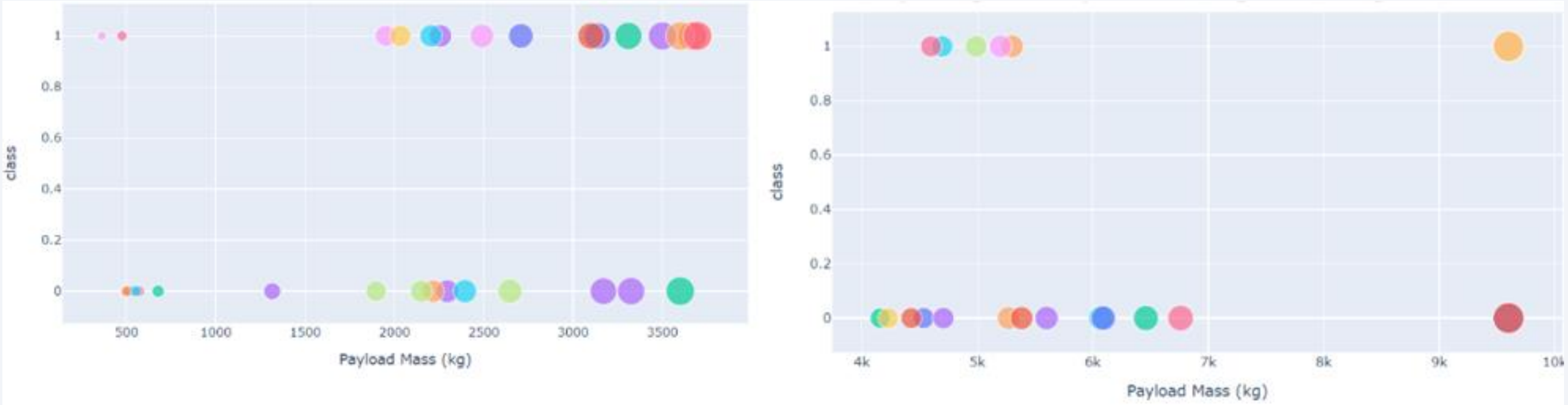
Launch success rates by launch site



Highest launch success ratio (KSC LC-39A)



Payload vs Launch Outcome for launch sites



Section 5

Predictive Analysis (Classification)

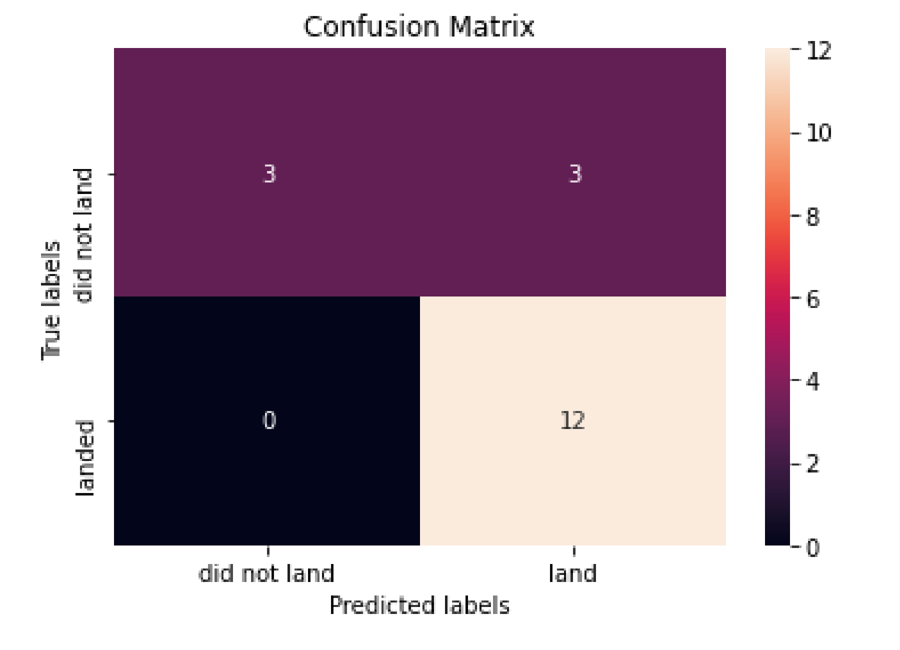
Classification Accuracy

```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix



Conclusions

- A larger number of launches per site seem to lead to a greater success rate
- Launch success rate seemed to increase over the years
- ES-L1, GEO, HEO, SSO, VLEO had best success rates
- KSC LC-39A had the most amount of success launches
- Decision tree as a predictive model seemed to be the best classifier for ML

Thank you!