IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Yoann Almeras
March 9, 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection (API)
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis using SQL
  - Exploratory Data Analysis using Visualization
  - Interactive Visual Analytics using Folium and Plotly Dash
  - Machine Learning Prediction
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demo in screenshots
  - Predictive analysis results

Introduction

- Project background and context

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch

In this project, we will train a machine learning model and use public information to predict if the Falcon 9 first stage will land successfully.

- Problems you want to find answers

- What are the key success factors of rocket landing ?
- What technical features determine the outcome of rocket landing ?
- What launch site offers the best conditions for success ?

Section 1

Methodology

Methodology

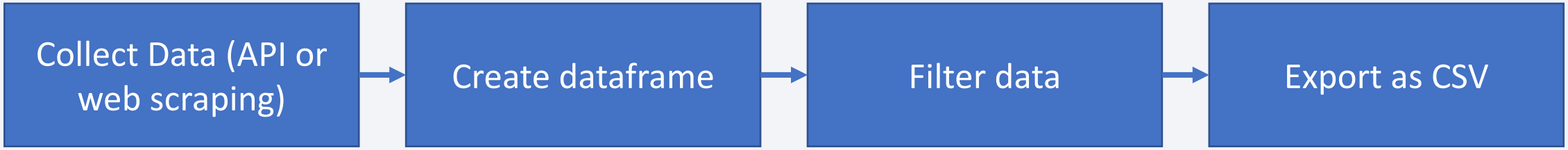
Executive Summary

- Data collection methodology:
  - Data collection using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - Data wrangling applying one-hot encoding to 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
  - How to build, tune, evaluate classification models

Data Collection

Data collection techniques :

- REST API
- Web scraping



Data Collection – SpaceX API

1. Collect Data

We used the .json() method to get the response

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

```
response = requests.get(spacex_url)
```

2. Convert response to JSON

We used the json_normalize function

```
# Use json_normalize meethod to convert the json result
data = pd.json_normalize(response.json())
```

3. Clean Data

We used auxiliary functions to extract information

```
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])
```

4. Create dictionary and create pandas dataframe

We used the BoosterVersion column to only keep the Falco 9 launches

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
```

```
# Create a data from launch_dict
data = pd.DataFrame(launch_dict)
```

5. Filter Data

We used the BoosterVersion column to only keep the Falco 9 launches

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data[data.BoosterVersion == 'Falcon 9']
data_falcon9
```

6. Export File

We exported the file as csv

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

GitHub URL of the notebook : https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/33f06986-b5d2-4e93-86d6-af583e336687/view?access_token=3ce61d214d8d0a38bc55d4bca15402053ab247e5ba7d14450d5be21c7de0723b

Data Collection - Scraping

1. Request the Falcon9 Launch Wiki page

We used the HTTP get method

```
# use requests.get() method with the provided
# assign the response to a object
page = requests.get(static_url)
page.status_code
```

2. Create Beautiful Soup object

We created a BeautifulSoup object from the HTML response

```
# Use BeautifulSoup() to create a BeautifulSoup object
soup = BeautifulSoup(page.text, 'html.parser')
```

3. Find tables on Wiki page

We used auxiliary functions to extract information

```
# Use the find_all function in the BeautifulSoup object
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

4. Extract column and variable names

We used the BoosterVersion column to only keep the Falcon 9 launches

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

5. Create empty dictionary

We created a dictionary that will be later used to create a dataframe

```
launch_dict= dict.fromkeys(column_names)
```

6. Parse launch records values

We extracted each row in each table

```
xtracted_row = 0
Extract each table
or table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collaps
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
```

7. Create dataframe

We created a pandas dataframe

```
df=pd.DataFrame(launch_dict)
```

8. Export File

We exported the file as csv

```
df.to_csv('spacex_web_scraped.csv', index=False)
```


GitHub URL of the notebook :

https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/642b310f-5133-4f29-9274-cb544a1fa426/view?access_token=44b75136b48f08e2ddb2cc89c47dbf9ce28cc90f92b5a6dfdcf0009d735bf241

Data Wrangling

Calculate number of launches on each site

```
In [5]: # Apply value_counts() on column LaunchSite
        df["LaunchSite"].value_counts()

Out[5]: CCAFS SLC 40    55
        KSC LC 39A      22
        VAFB SLC 4E     13
        Name: LaunchSite, dtype: int64
```

Calculate the number and occurrence of each orbit

```
In [6]: # Apply value_counts on Orbit column
        df["Orbit"].value_counts("Orbit")

Out[6]: GTO      0.300000
        ISS      0.233333
        VLEO     0.155556
        PO       0.100000
        LEO      0.077778
        SSO      0.055556
        MEO      0.033333
        ES-L1    0.011111
        HEO      0.011111
        SO       0.011111
        GEO      0.011111
        Name: Orbit, dtype: float64
```

Calculate landing outcome per orbit type

```
In [15]: # landing_outcomes = values on Outcome column
         landing_outcomes = df["Outcome"].value_counts()
```

Create a landing outcome label from Outcome column

```
In [11]: # landing_class = 0 if bad_outcome
         # landing_class = 1 otherwise
         landing_class = []
         for key,value in df["Outcome"].items():
             if value in bad_outcomes:
                 landing_class.append(0)
             else:
                 landing_class.append(1)
```

Export File

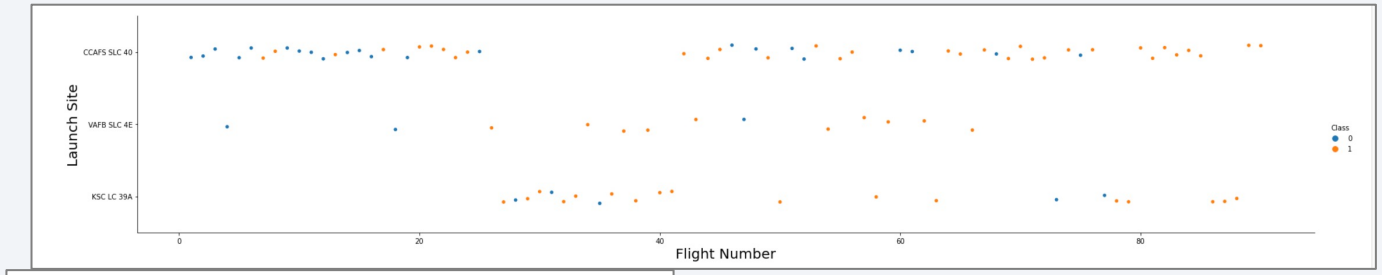
```
df.to_csv("dataset_part_2.csv", index=False)
```

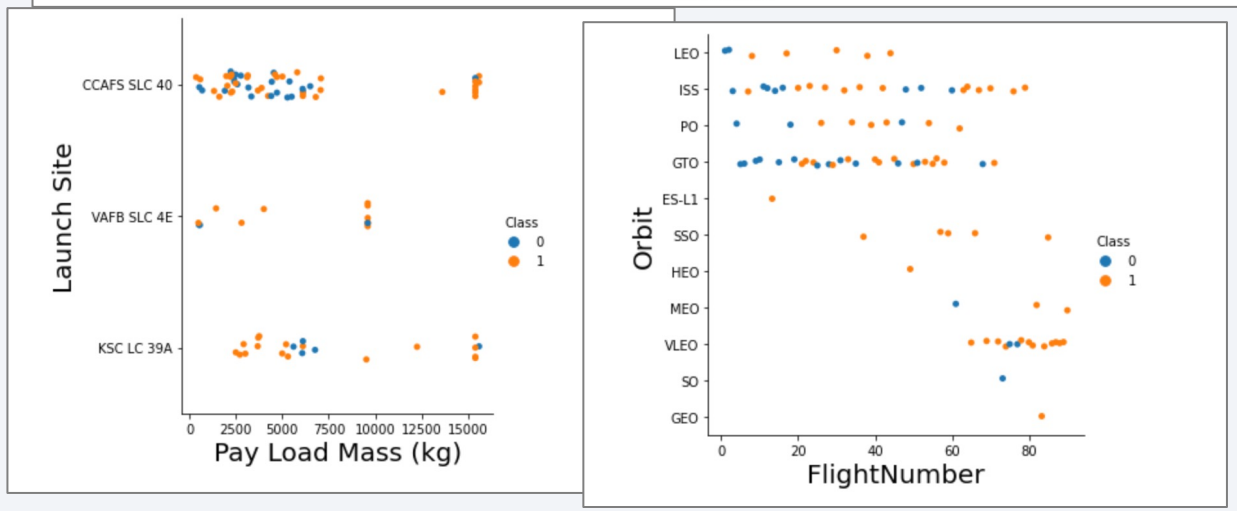

GitHub URL of the notebook :

https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/162d4e0b-10ce-449b-b20e-63914e51ddc8/view?access_token=4b1bf5fc4ec4f23e946fb2e66ccc505c084ac87e48a8ce4515bccc06399bd173

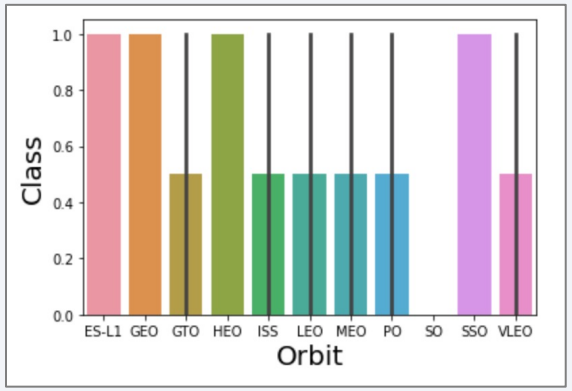
EDA with Data Visualization

We used **Scatter point charts** to observe relationships between numerical variables

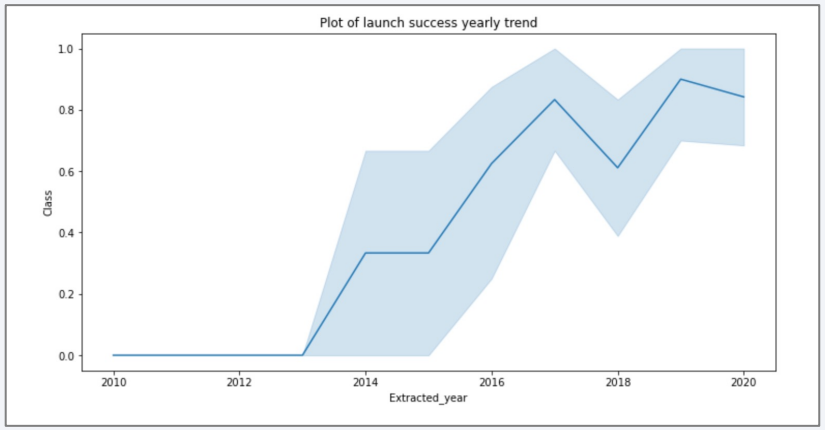




We used a **bar chart** to observe the relationship between success and orbit type (categorical variable)



We used a **line chart** to observe the trend in time



GitHub URL of the notebook :

https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/785c8a33-b47a-4097-ad2a-34ec7712dee4/view?access_token=767bfa23eddcf10e24eb3a0ae72c99f1b17e2e39199efc2186567a3143d5c9e9

EDA with SQL

- We performed the following queries :

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

```
%sql select distinct(LAUNCH_SITE) from SPACEXTBL
```

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

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

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

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
```

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

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
```

List the date when the first successful landing outcome in ground pad was achieved

```
%sql select min(DATE) from SPACEXTBL where Landing__Outcome = 'Success (ground pad)'
```

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

```
%sql select BOOSTER_VERSION from SPACEXTBL where Landing__Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS
```

List the total number of successful and failure mission outcomes

```
%sql select count(MISSION_OUTCOME) from SPACEXTBL where MISSION_OUTCOME = 'Success' or MISSION_OUTCOME = 'Failure (in flight)'
```

List the names of the booster_versions which have carried the maximum payload mass

```
%sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

```
%sql select BOOSTER_VERSION, LAUNCH_SITE from SPACEXTBL where year(DATE) = '2015' AND LANDING__Outcome = 'Failure (drone ship)'
```

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

```
%sql select * from SPACEXTBL where Landing__Outcome like 'Success%' and (DATE between '2010-06-04' and '2017-03-20') order by date desc
```


GitHub URL of the notebook :

https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0402e38c-261a-4618-9a33-5de76f676767/view?access_token=6dc6412277a7aeb59ab54719dd6514120e3f735cf36d189a63dfa40decc23215

Build an Interactive Map with Folium

Map objects	Purpose
Circle	Generate a circle to highlight where the marker is
Marker	Make a mark on the map
Icon	Create an icon on the map
PolyLine	Draw a line between two points (launch site and city, railway, highway, etc.)

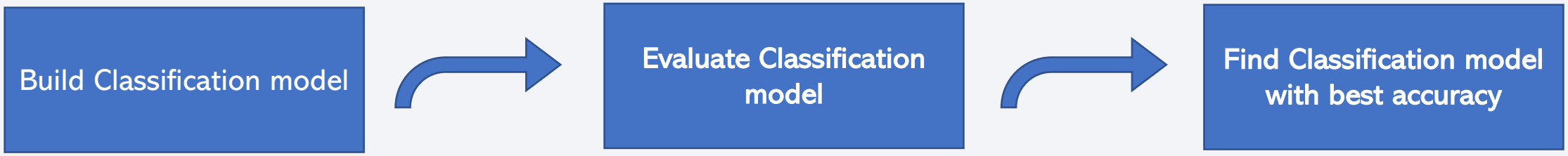
GitHub URL of the notebook :
https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/ae33a996-56a8-41c2-a184-fa727cd384b5/view?access_token=212a8aefea425f19db12faf72308c8c263d972d37310c72fc33df23b96feae89

Build a Dashboard with Plotly Dash

Elements of the dashboard	Purpose
Dropdown menu	Select launch site
Callback function	Get the selected launch site to generate specific graph
Pie chart	Visualize success rate
Range slider	Select different payload range to visualise results more easily
Scatter plot	Visualize correlation bewteen payload and success

GitHub URL of the notebook :
https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/ae33a996-56a8-41c2-a184-fa727cd384b5/view?access_token=212a8aefea425f19db12faf72308c8c263d972d37310c72fc33df23b96feae89

Predictive Analysis (Classification)



- Load dataframe
- Create NumPy array from the column Class
- Standardize data
- Split the data (train/test)

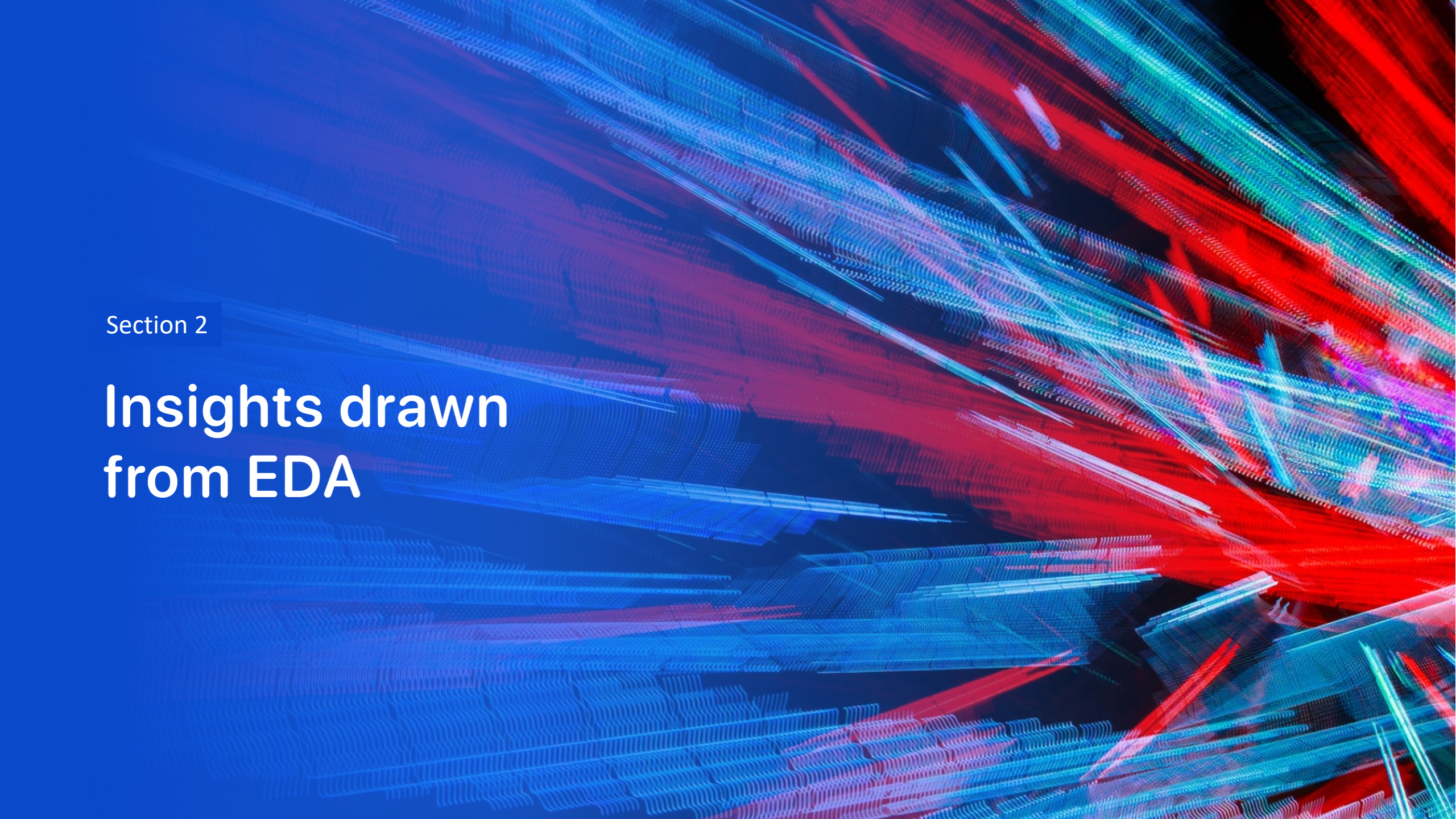
- Create logistic regression, SVM, Decision Tree and KNN objects
- Find the best parameters for each
- Plot confusion matrix and calculate accuracy on the test data for each

- Find the model that performs best

GitHub URL of the notebook :
https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/5a2e7762-212c-43a6-96c9-de4ba9eabef1/view?access_token=426d2b9b2d5302f37326456d7f891b77ff0e2108bde206562435946cd1dd41a6

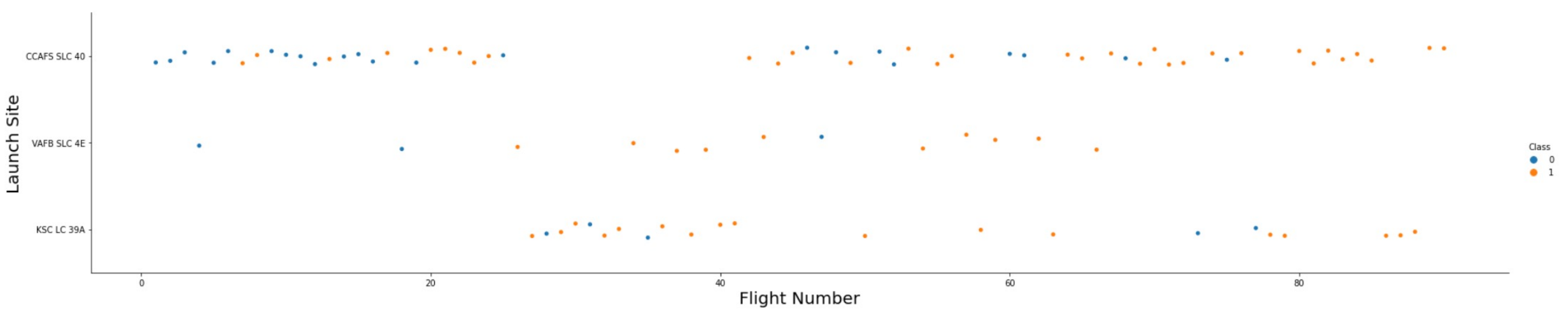
Results

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


Section 2
Insights drawn from EDA

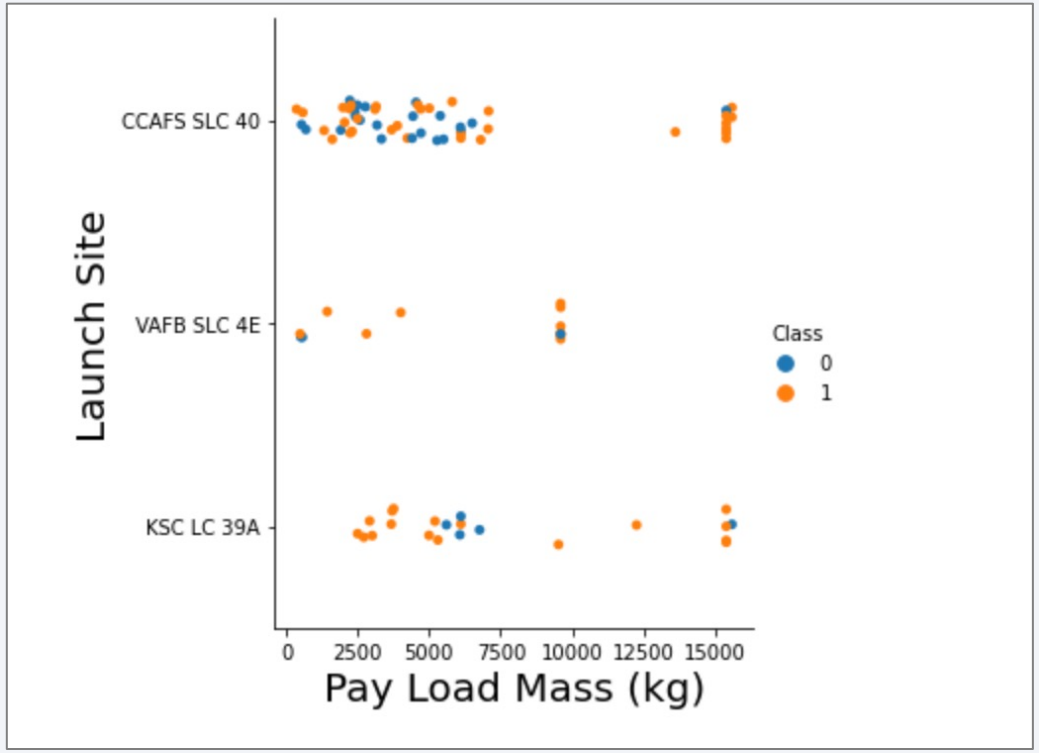
Flight Number vs. Launch Site

From the scatter plot, the higher number of flights at a launch site, the higher chance of success.



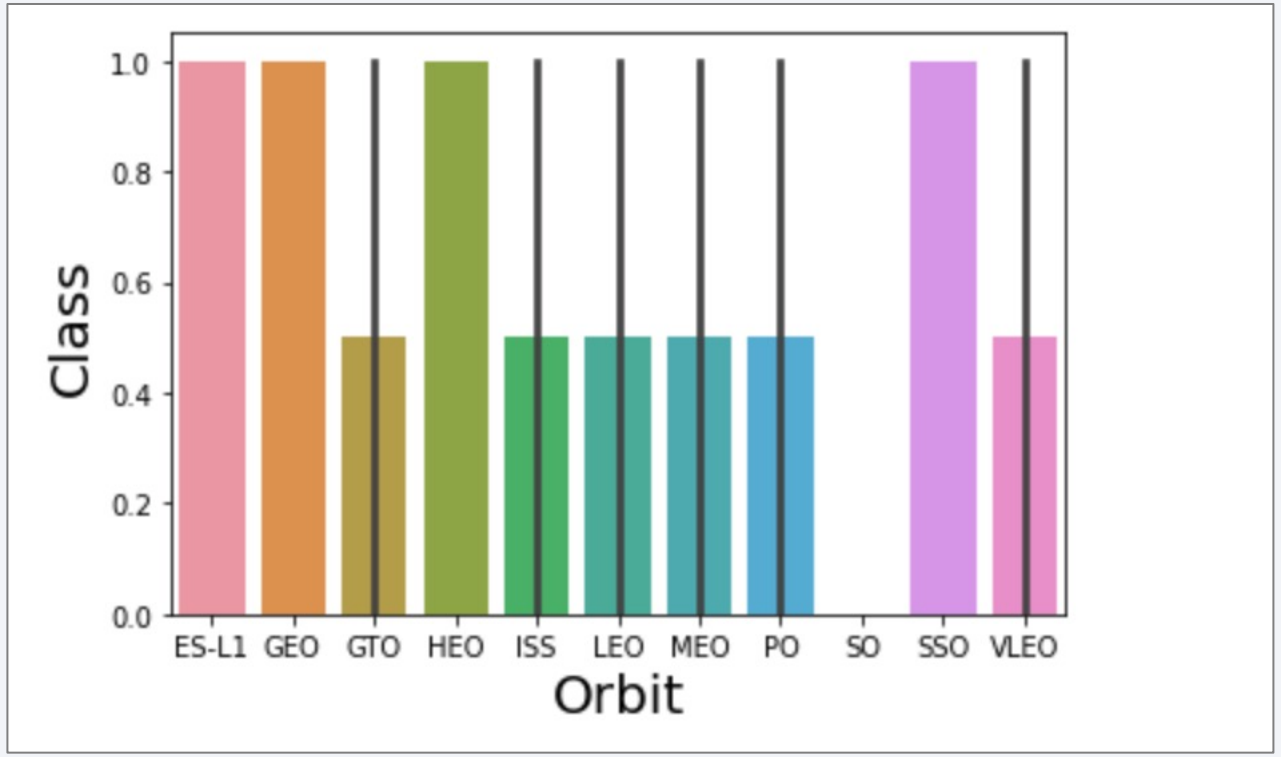
Payload vs. Launch Site

From the scatter plot, the higher the Pay Load Mass, the higher chance of success. However, when we zoom in site by site, the pattern becomes unclear. The correlation would need to be further investigated before making a decision.



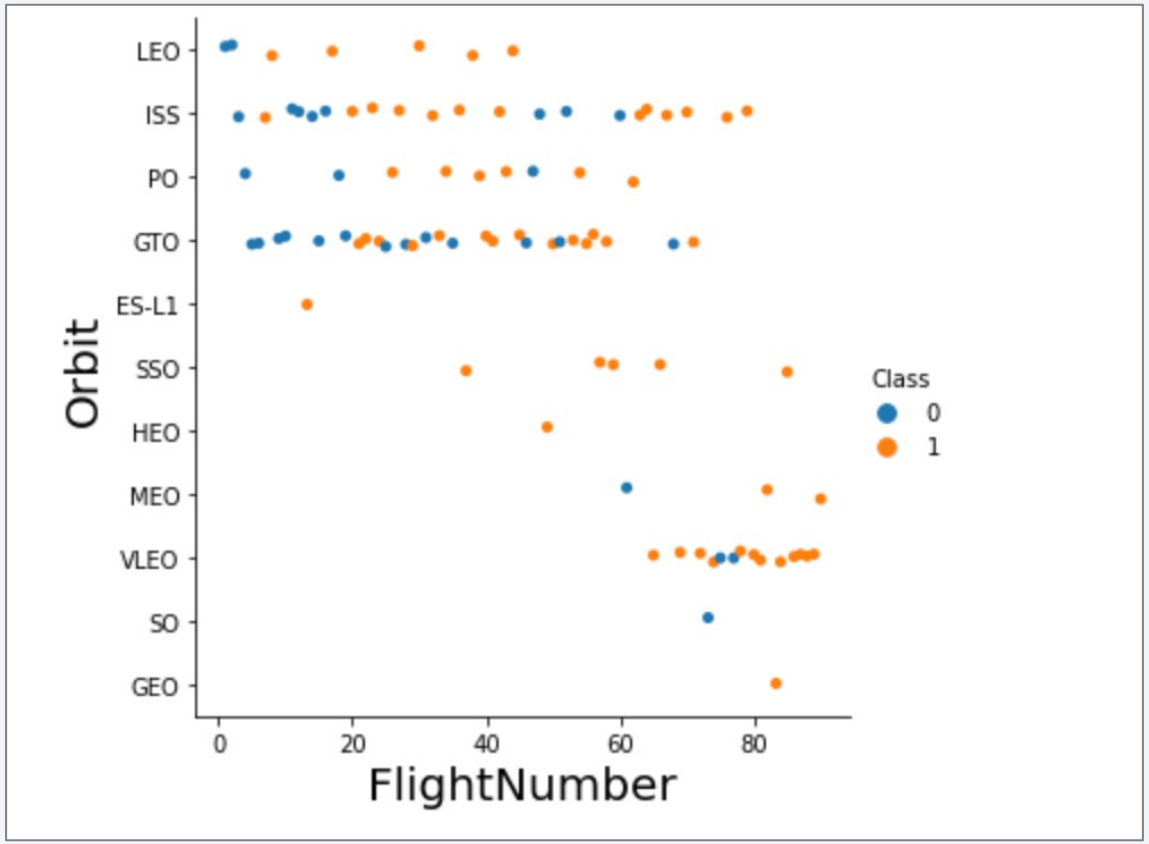
Success Rate vs. Orbit Type

From the bar chart, ES-L1, GEO, HEO and SSO have the higher chance of success.



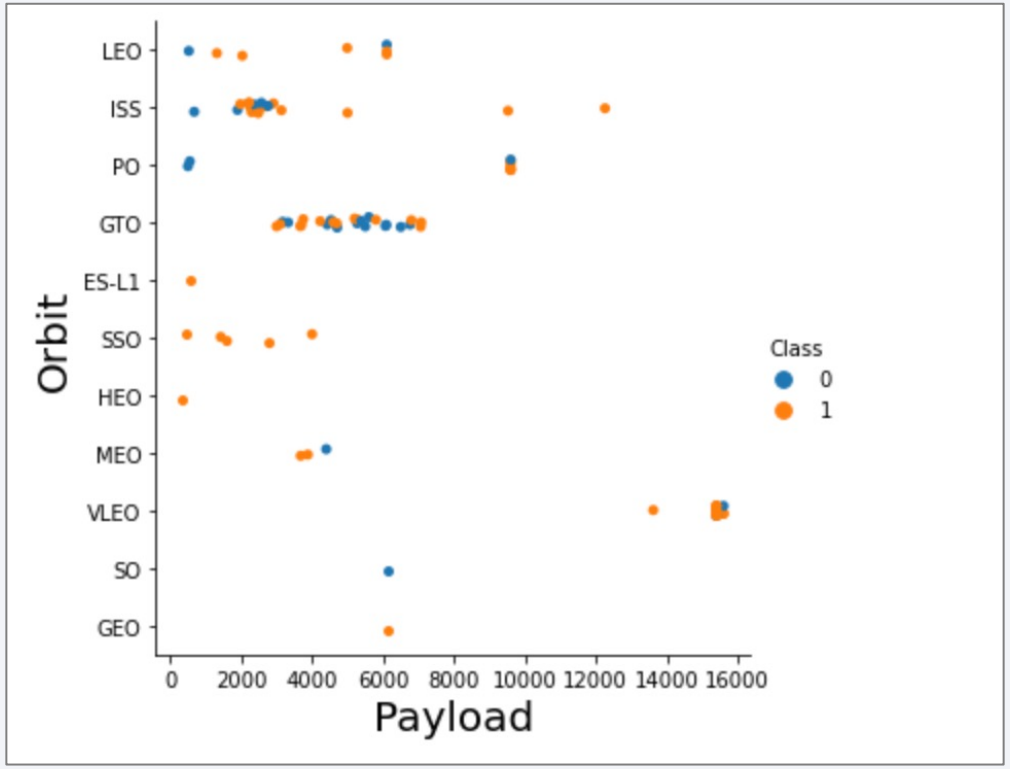
Flight Number vs. Orbit Type

From the scatter plot, we see there might be a correlation for some Orbit type like LEO.
We don't visualize this correlation for other types like ISS or GTO.



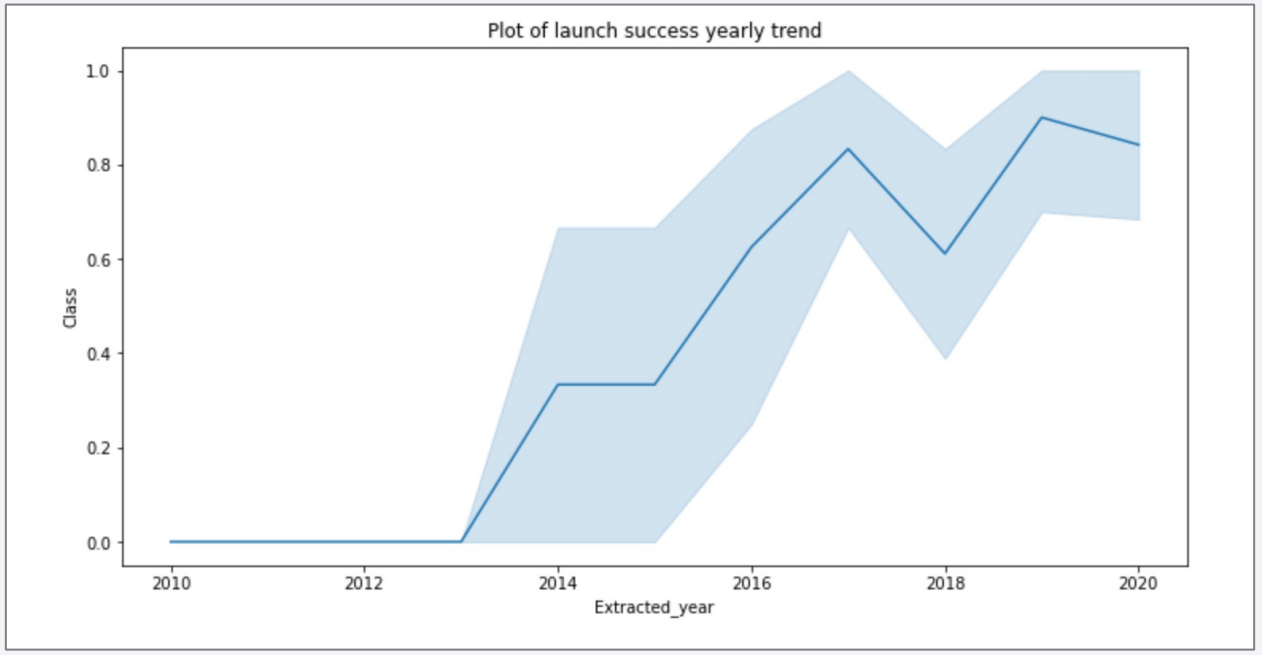
Payload vs. Orbit Type

From the scatter plot, we can see that Payload has a positive impact on some orbit types like ISS, but a negative one on others like GTO.



Launch Success Yearly Trend

From the line chart, we observe that launch success has started to increase in 2013 and has now reached a rate of approximately 80%.



All Launch Site Names

We used a select distinct query to output the different site names.

```
In [13]: %sql select distinct(LAUNCH_SITE) from SPACEXTBL
          * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8
         Done.
```

Out[13]:

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

Launch Site Names Begin with 'CCA'

- We used the following query to display the names of the first 5 launch site beginning with ‘CCA’.

```
In [14]: %sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
 * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498/bludb
Done.
```

Out[14]:

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

Total Payload Mass

- We used the following query to display the total Payload Mass carried by boosters :

```
In [15]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
 * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8
/bludb
Done.
```

Out[15]:

1
45596

Average Payload Mass by F9 v1.1

- We used the following query to display the Average Payload Mass by Falcon9 v1.1 :

```
In [16]: %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
 * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.
Done.
Out[16]:
1
2928
```

First Successful Ground Landing Date

- We used the following query to display the First Successful Ground Landing Date :

```
In [17]: %sql select min(DATE) from SPACEXTBL where Landing__Outcome = 'Success (ground pad)'
          * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.dat
        Done.
Out[17]:
        1
        2015-12-22
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used the following query to display the names of the Successful Drone Ship Landing with Payload between 4000 and 6000

```
In [18]: %sql select BOOSTER_VERSION from SPACEXTBL where Landing__Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS
          * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498/bludb
         Done.
```

Out[18]:

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

Total Number of Successful and Failure Mission Outcomes

- **We used the following query to calculate the Total Number of Successful and Failure Mission Outcomes :**

```
In [19]: %sql select count(MISSION_OUTCOME) from SPACEXTBL where MISSION_OUTCOME = 'Success' or MISSION_OUTCOME = 'Failure (in flight)'
          * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498/bludb
         Done.
Out[19]:  1
          100
```

Boosters Carried Maximum Payload

- We used the following query to list the names of the booster which have carried the maximum payload mass :

```
In [20]: %sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
 * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498
/bludb
Done.
```

Out[20]:

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

- We used the following query to list the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
In [22]: %sql select BOOSTER_VERSION, LAUNCH_SITE from SPACEXTBL where year(DATE) = '2015' AND LANDING__Outcome = 'Failure (drone ship)'
 * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498/bludb
Done.
```

Out[22]:

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

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

- We used the following query to rank the count of landing outcomes between 2010-06-04 and 2017-03-20, in descending order :

```
In [22]: %sql select * from SPACEXTBL where Landing__Outcome like 'Success%' and (DATE between '2010-06-04' and '2017-03-20') order by date desc
```

```
 * ibm_db_sa://jrd71896:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498/bludb
Done.
```

Out[22]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Success (drone ship)
2016-08-14	05:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-07-18	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Success (drone ship)
2016-05-06	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)

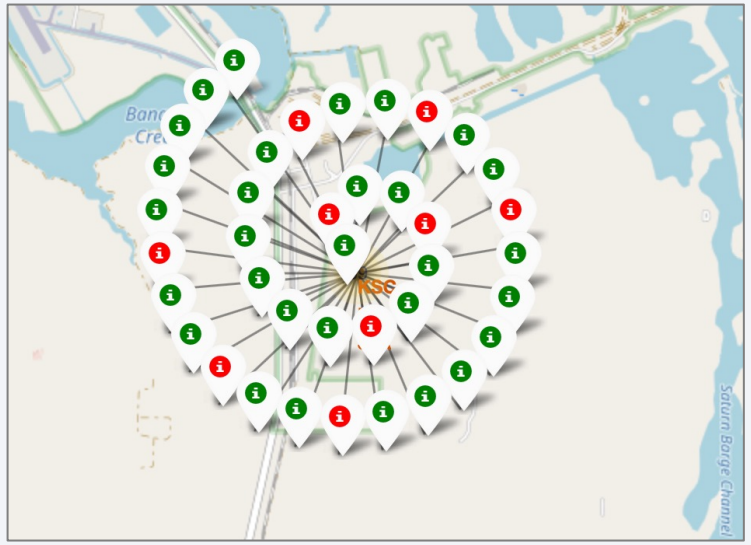

Section 3
Launch Sites
Proximities Analysis

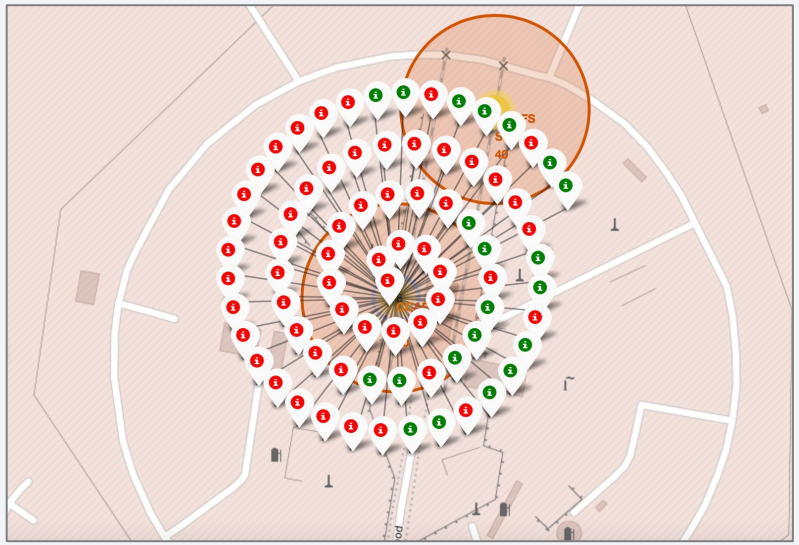
Launch sites on global Folium map

We can see from the map that all launch sites are located nearby California and Florida coasts, in the United States.

Map with markers showing launch sites with color labels

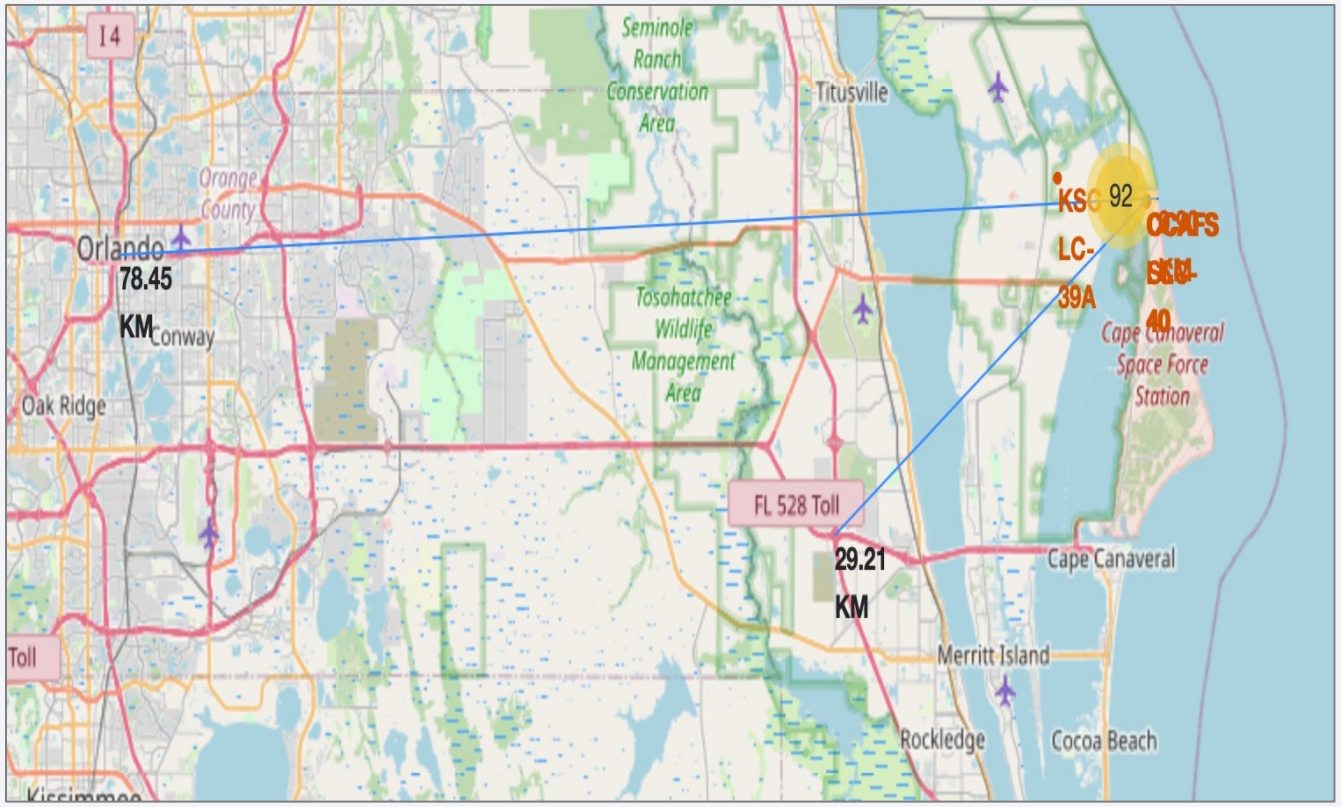
- The green labels show the successful launches for each site, the red markers show the failures.
- The KSC LC-32A has the most success, as show on the bottom-left screenshot.

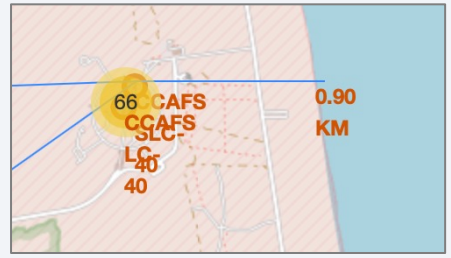




Launch site proximity to landmarks

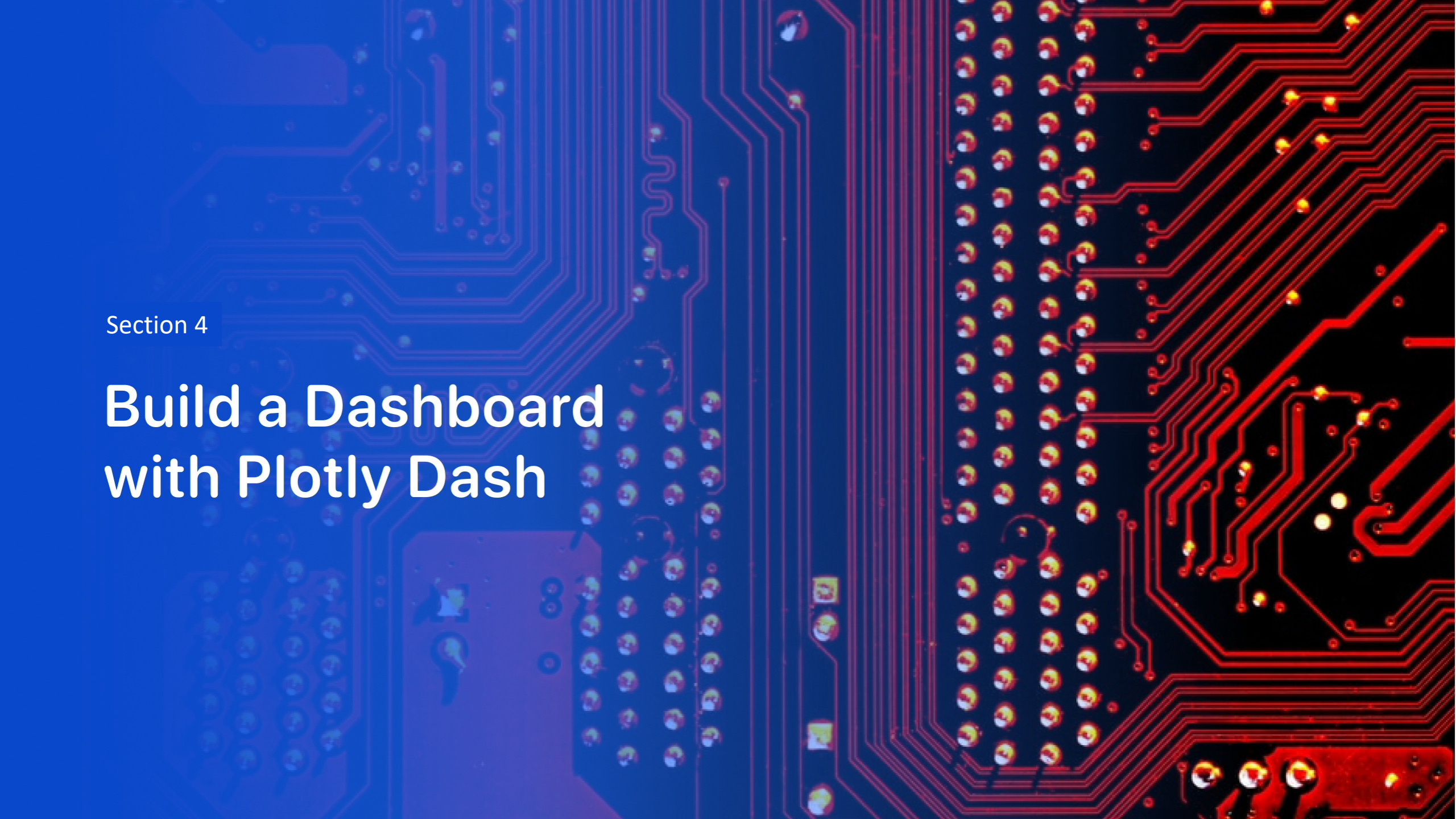
Distance to city :
78,45km





Distance to coastline
: 0,90km

Distance to freeway :
29,41km

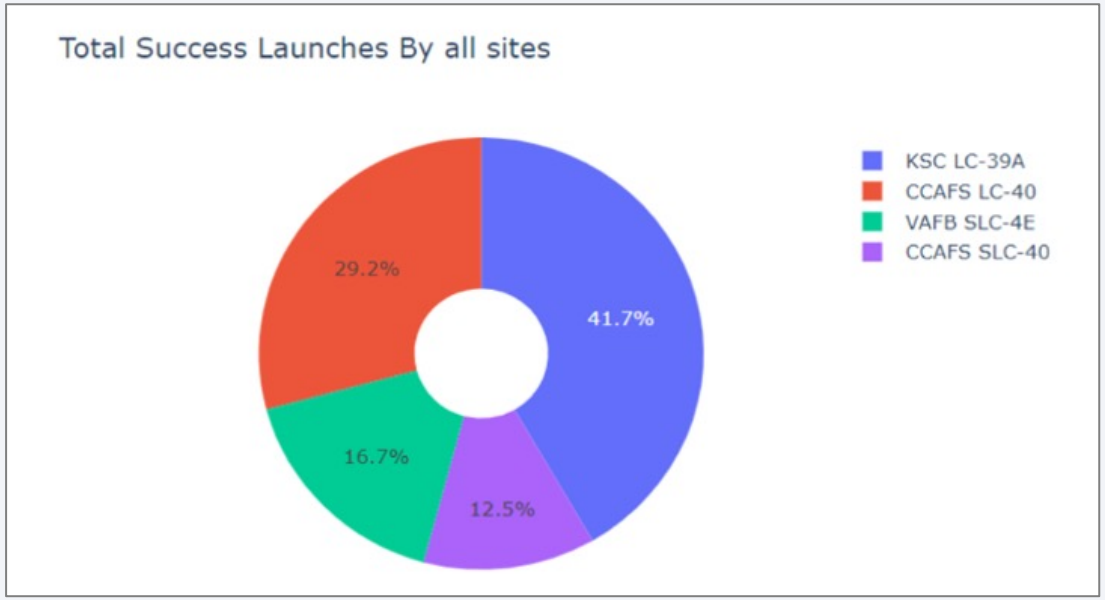


Section 4

Build a Dashboard with Plotly Dash

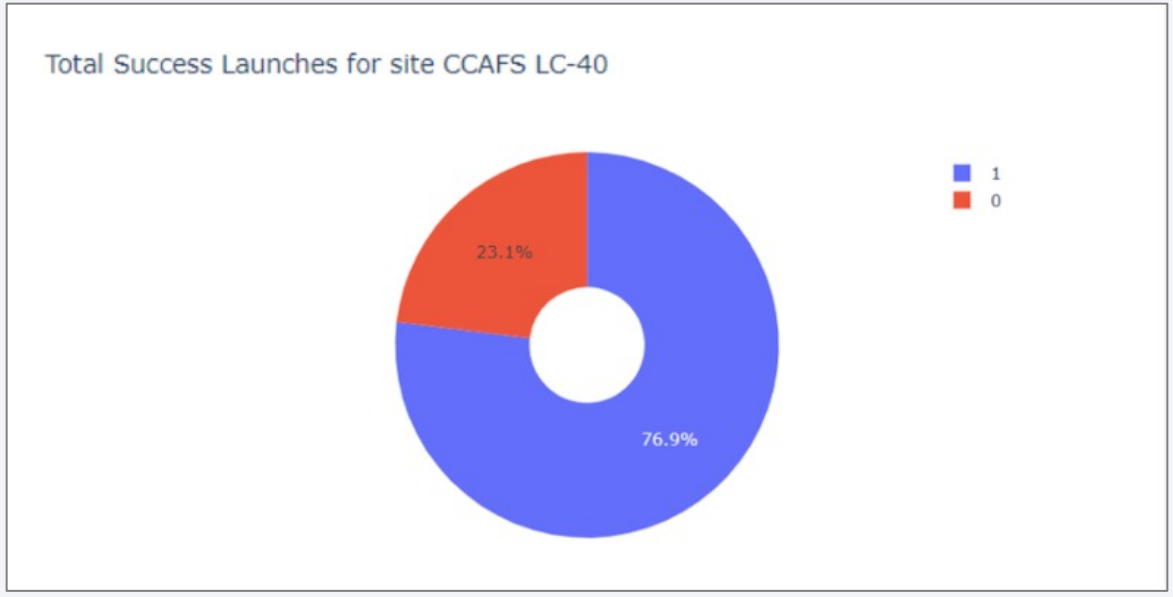
Total Success launches by all sites

From the pie chart, we can see that site KSC LC-39A has had the most successful launches in volume, with 41,7%.



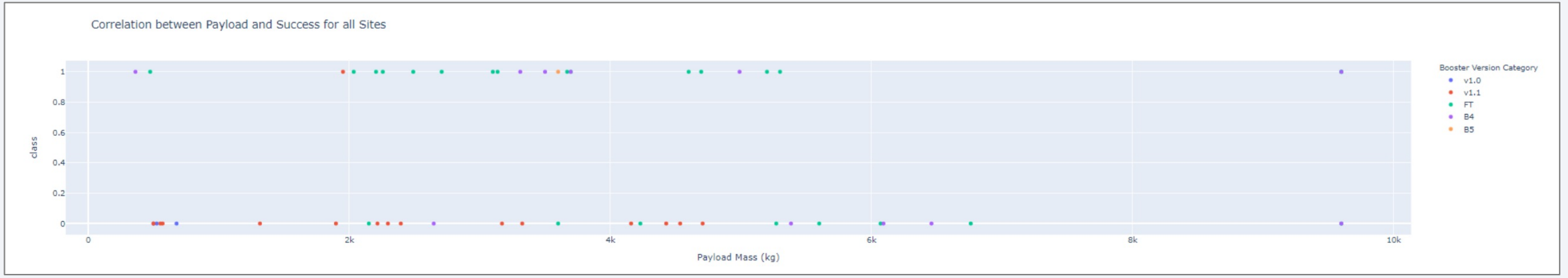
Site with higher Success ratio

From the pie chart, we can see that site CCAFS LC-40 has had a success ratio of 76.9%



Payload Mass vs. Success for all launch sites

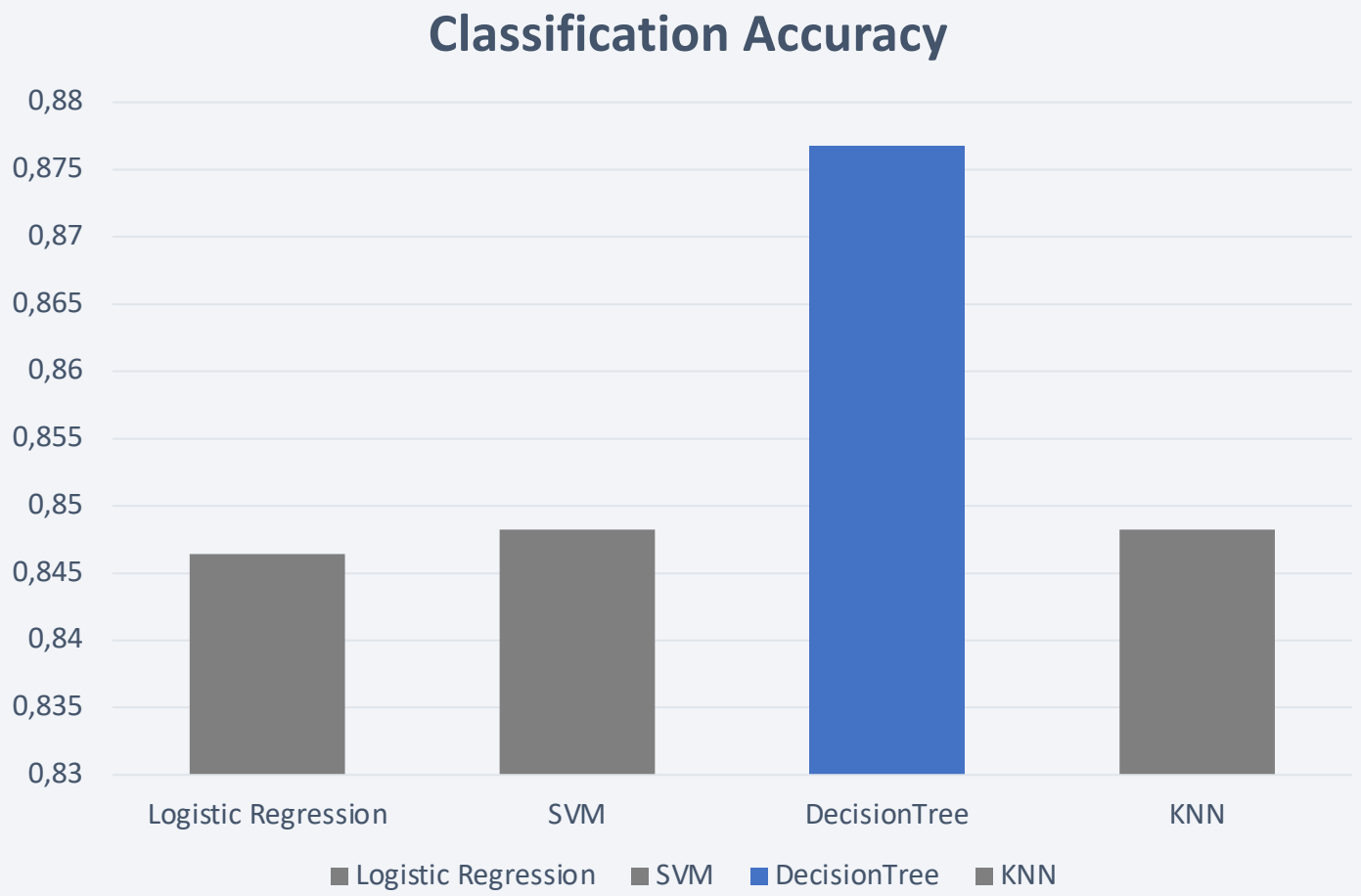
From the scatter plot, we observe that the launch success rate is higher for low weighted payloads < 4000 kgs.



Section 5

Predictive Analysis (Classification)

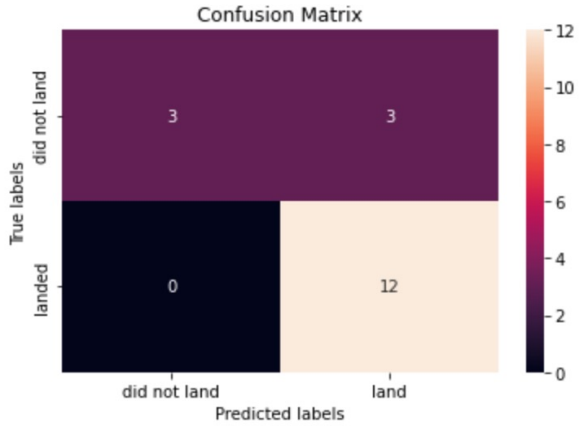
Classification Accuracy



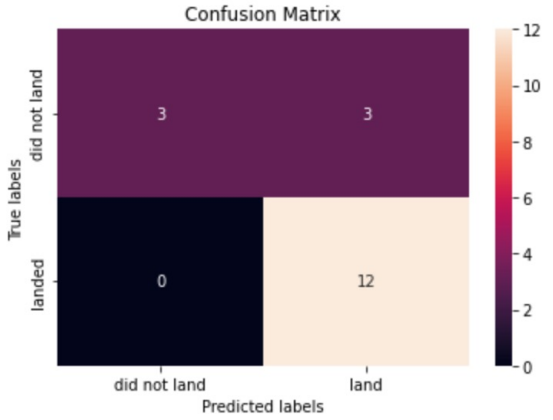
The Decision Tree performs best compared to the other algorithms.

Confusion Matrix

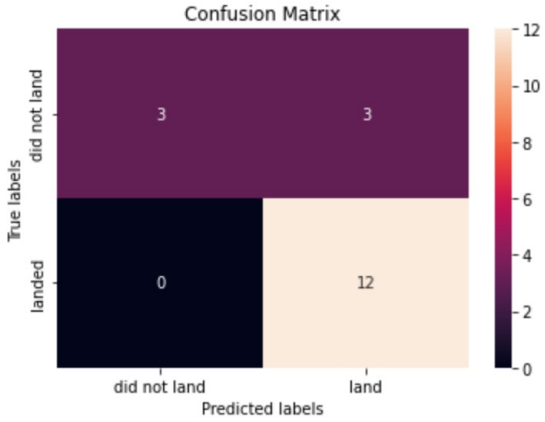
```
In [19]: yhat=logreg_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```



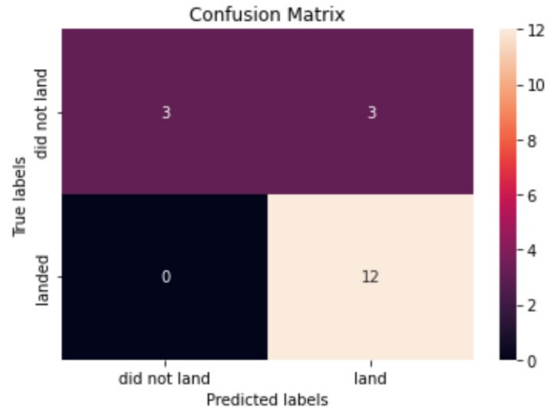

```
In [24]: yhat=svm_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```




```
In [29]: yhat = svm_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```




```
In [34]: yhat = knn_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```



The 4 confusion matrix are identical with accuracy of 0,8333.

Conclusions

- The launch success rates has been improving over time
- Launch site KSC LC-32A has had the most launch success
- Orbits type ES-L1, GEO, HEO and SSO have the higher chance of success
- For other orbits type such as GTO, low weight Payload increases chance of success
- Our Decision Tree model is the one that performs best.

Thank you!