IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Cristian Saavedra
16/oct/2023

Outline

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

Executive Summary

- **Summary of methodologies**
  - Data collection overview
  - Data Wrangling
  - EDA – Exploratory Data Analysis
  - Interactive Visual Analytics
  - Machine Learning Predictions
- **Summary of all results**
  - Were founded the critical variables, and their characteristics to have a successful landing. These characteristics was evaluated in different models, give us a high accuracy (>83%). Some of these critical variables are:
    - Use “high” Payload Mass (i.e. >8.000 kg)
    - Use the Launch Site KSC LC-39A
    - Choose the Orbits: ES-L1, GEO, HEO, SSO, VLEO
    - Use “medium range” booster (i.e. 2.000 – 5.000 kg)

Introduction

- Project background and context

Our company, SPACE Y, are in the market of making space travel affordable for everyone. To reach that, one of the main projects is saving cost by made “reusable rockets”.

Our CEO, Allon Musk, give us the mission to understand the variables that can made reusable a rocket.

- Problems you want to find answers

Based on the data of the previous launches, we will determine the attributes to make a successful landing of the first stage rocket.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - One part of data was collected from API REST of SpaceX (launch data, type of rocket, payload, etc). Another part of data was extracted from WIKI through web scraping.
- Perform data wrangling
  - The data was processed according to separate the successful landing (class 1) from failed (class 0).
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash

Methodology

- Perform predictive analysis using classification models
  - The first step is the **Preprocessing of data**, which will allow us to standardize our data.
  - The second it's the **Train Test Split**, which will allow us to split it into training and test data.
  - After we will train the model and perform a **Grid Search**, allowing us to find the hyperparameters that allow a given algorithm to perform best.
  - With the best hyperparameter values, we will determine the model with the highest accuracy using the training data.
  - **Evaluate and test** logistic regression, support vector machines, decision tree classifier, and K-nearest neighbors.
  - Finally, we will generate the **Confusion Matrix**.

Data Collection

Code LAB1: GitHub

- Data from API
  - The dataset of the rockets

```
In [2]: # 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']:
       if x:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])
```

  - The dataset of the launchpads

```
In [3]: # Takes the dataset and uses the launchpad column to call the API and append the data to the list
def getLaunchSite(data):
    for x in data['launchpad']:
       if x:
         response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
         Longitude.append(response['longitude'])
         Latitude.append(response['latitude'])
         LaunchSite.append(response['name'])
```

  - The dataset of the payloads

```
In [4]: # Takes the dataset and uses the payloads column to call the API and append the data to the lists
def getPayloadData(data):
    for load in data['payloads']:
       if load:
        response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
        PayloadMass.append(response['mass_kg'])
        Orbit.append(response['orbit'])
```

  - Rocket launch data from SpaceX API

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

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

Data Collection - Scraping

Code LAB2: GitHub

- Data from WIKI

  - Scrape the data from a snapshot of the List of Falcon 9 and Falcon Heavy launches Wikipage updated on 9th June 2021
  - Perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response
  - Create a BeautifulSoup object from the HTML response
  - Find all tables on the wiki page first
  - Starting from the third table is our target table contains the actual launch records

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

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

```
In [7]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
        soup = BeautifulSoup(data)
```

```
In [9]: # Use the find_all function in the BeautifulSoup object, with element type `table`
        # Assign the result to a list called `html_tables`
        html_tables = soup.find_all('table')
```

```
In [10]: # Let's print the third table and check its content
         first_launch_table = html_tables[2]
         print(first_launch_table)
```

Data Wrangling

Code LAB3: GitHub

- The data was processed according to separate the successful landing (class 1) from failed (class 0).

```
In [7]: # landing_outcomes = values on Outcome column
        landing_outcomes = df['Outcome'].value_counts()
        landing_outcomes

Out[7]: True ASDS      41
        None None      19
        True RTLS      14
        False ASDS      6
        True Ocean      5
        False Ocean     2
        None ASDS       2
        False RTLS      1
        Name: Outcome, dtype: int64
```

```
In [10]: # landing_class = 0 if bad_outcome
         # landing_class = 1 otherwise
         landing_class = df['Outcome'].replace({'True ASDS': 1, 'None None': 0, 'True RTLS': 1, 'False ASDS': 0, 'True Ocean': 1, 'False O
         df['Outcome'] = df['Outcome'].astype(int)
         df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 17 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   FlightNumber    90 non-null     int64
 1   Date            90 non-null     object
 2   BoosterVersion  90 non-null     object
 3   PayloadMass     90 non-null     float64
 4   Orbit           90 non-null     object
 5   LaunchSite      90 non-null     object
 6   Outcome         90 non-null     int64
 7   Flights         90 non-null     int64
 8   GridFins        90 non-null     bool
 9   Reused          90 non-null     bool
 10  Legs            90 non-null     bool
 11  LandingPad      64 non-null     object
 12  Block           90 non-null     float64
 13  ReusedCount     90 non-null     int64
 14  Serial          90 non-null     object
 15  Longitude       90 non-null     float64
 16  Latitude        90 non-null     float64
dtypes: bool(3), float64(4), int64(4), object(6)
memory usage: 10.2+ KB
```

EDA with SQL

Code LAB4:

- Use SQL to looking for the following data:
  - 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 v1.1
  - Listed the date when the first successful landing outcome in ground pad was achieved
  - List 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 boosters versions which have carried the maximum payload mass
  - Listed the records which will display the month names, failure landing outcomesin drone ship , booster versions, launch site for the months in year 2015;
  - 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.

EDA with Data Visualization

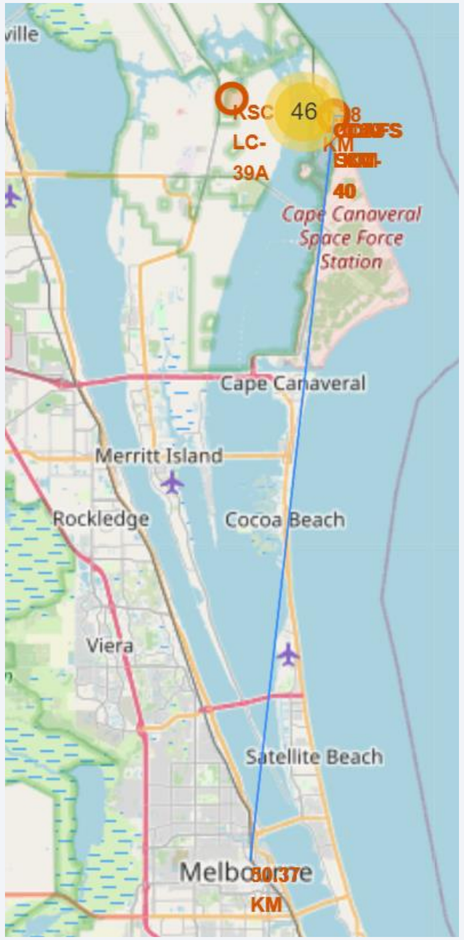
Code LAB5:

- Analyze the data to visualize relationships between:
  - Flight Number and Launch Site
  - Payload and Launch Site
  - Relationship between success rate of each orbit type
  - Flight Number and Orbit type
  - Payload and Orbit type
  - Launch success yearly trend

Build an Interactive Map with Folium

Code LAB6:

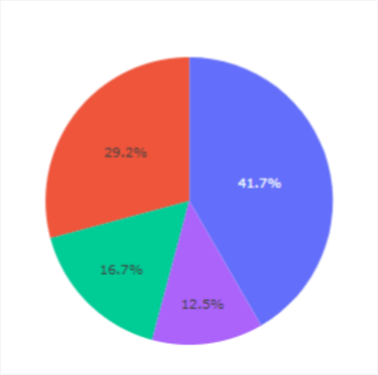
- Building a Interactive Map, including the following marks:
  - Mark all launch sites on a map
  - Mark the success/failed launches for each site on the map
  - Calculate the distances between a launch site to its proximities



Build a Dashboard with Plotly Dash

Code Dash:

- Building a Dashboard with Plotly Dash in order to include:
  - A **Pie Chart,** based on selected site dropdown, which show us the Total Successful Launches rate by sites.
  - A **Scatter Chart**, based on the payload and launch outcome. This chart help us to visually observe how payload may be correlated with mission outcomes for selected sites.



Predictive Analysis (Classification)

Code LAB7: GitHub

- This are the steps in which made the predictive Analysis:
  - Step 1: Load the data
  - Step 2: Standardize the data
  - Step 3: Split the data into training and testing data
  - Step 4: Create a logistic regression object, then create a *GridSearchCV*
  - Step 5: Fit the object to find the best parameters from the dictionary *parameters*.
  - Step 6: Looking for the best parameters using the data attribute *best_params_* and the accuracy on the validation data using the data attribute *best_score_*
  - Step 7: Calculate the accuracy on the test data using the method *score*
  - Step 8: Plot the Confusion Matrix
  - Step 9: Repeat the Step 4 to 9 with: SVM, Decision Tree and KNN

Results

- According the **EDA**:
  - There are positive relations between the successful ratio of launches and some characteristic like:
    - Payload Mass (high Payload, i.e. >8.000 kg, better successful launch ratio)
    - Launch Site (KSC LC-39A have the best launch ratio)
    - Orbit Type (orbits ES-L1, GEO, HEO, SSO, VLEO are successful)
    - Booster version ("medium range" booster, i.e. 2.000 – 5.000 kg, are successful)

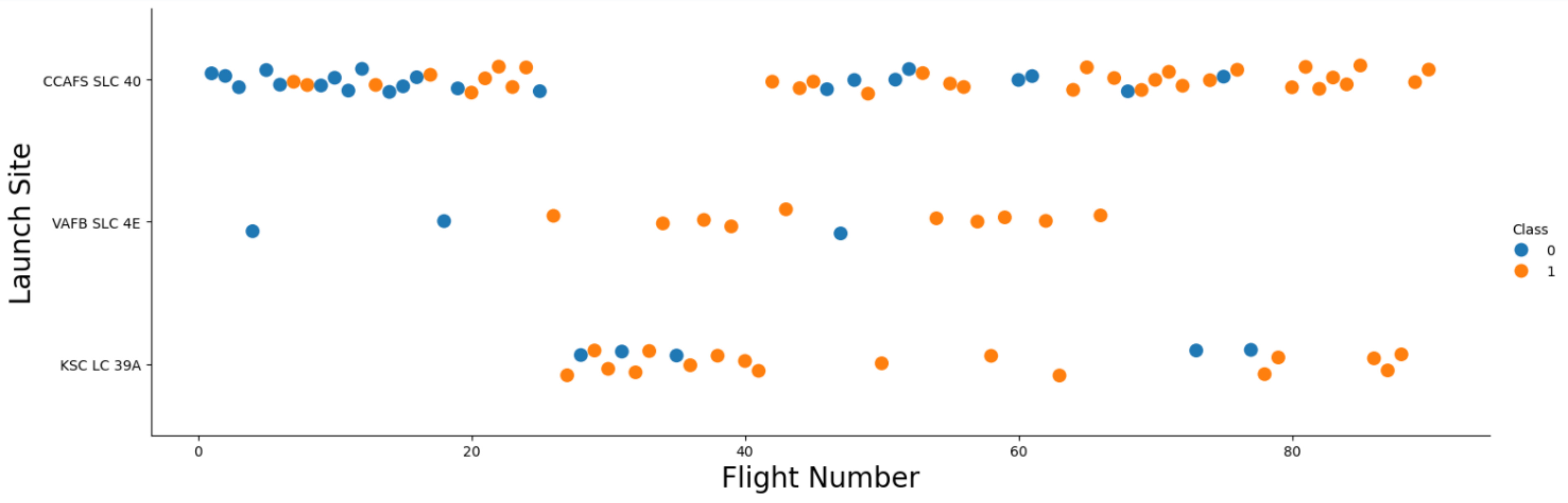
- According the **Predictive Analysis**:
  - The models: Logistic Regression, SVM and KNN have the best and the same accuracy (83,3%) to predict the successful of a launch.

Section 2

Insights drawn from EDA

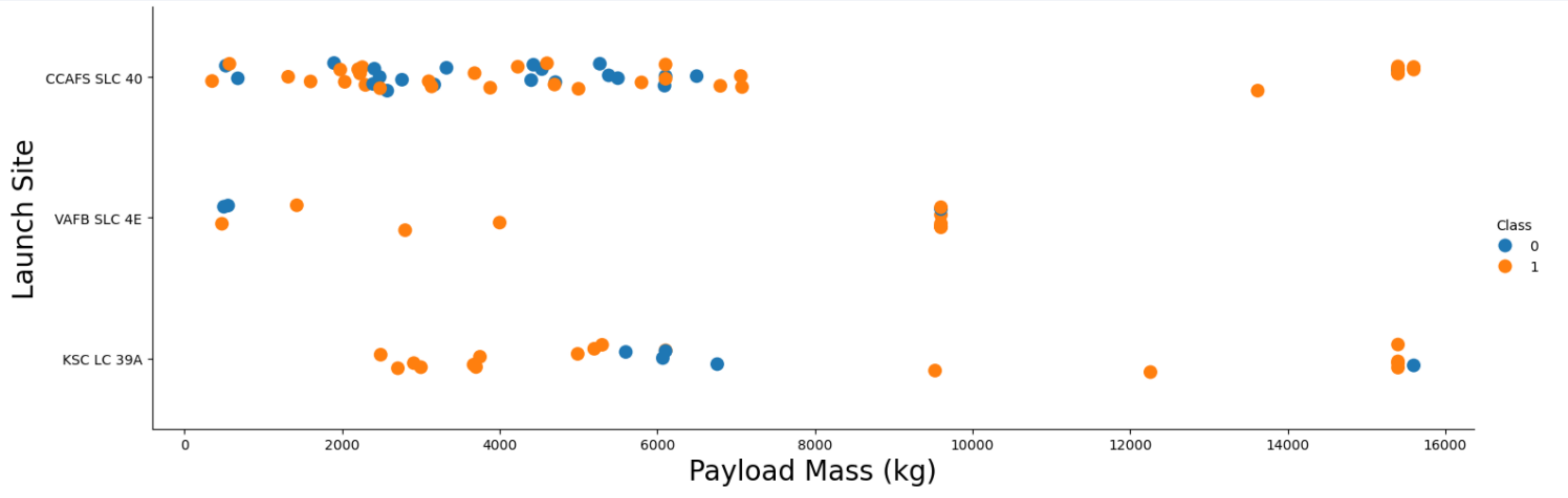
Flight Number vs. Launch Site

- The lates flights was more successful than previous. This tendency start since the 80th flight approximately.
- The success ratio is different across the Launch Sites.
- The Launch Site VAFB SCL 4E, was the more successful compare with others.



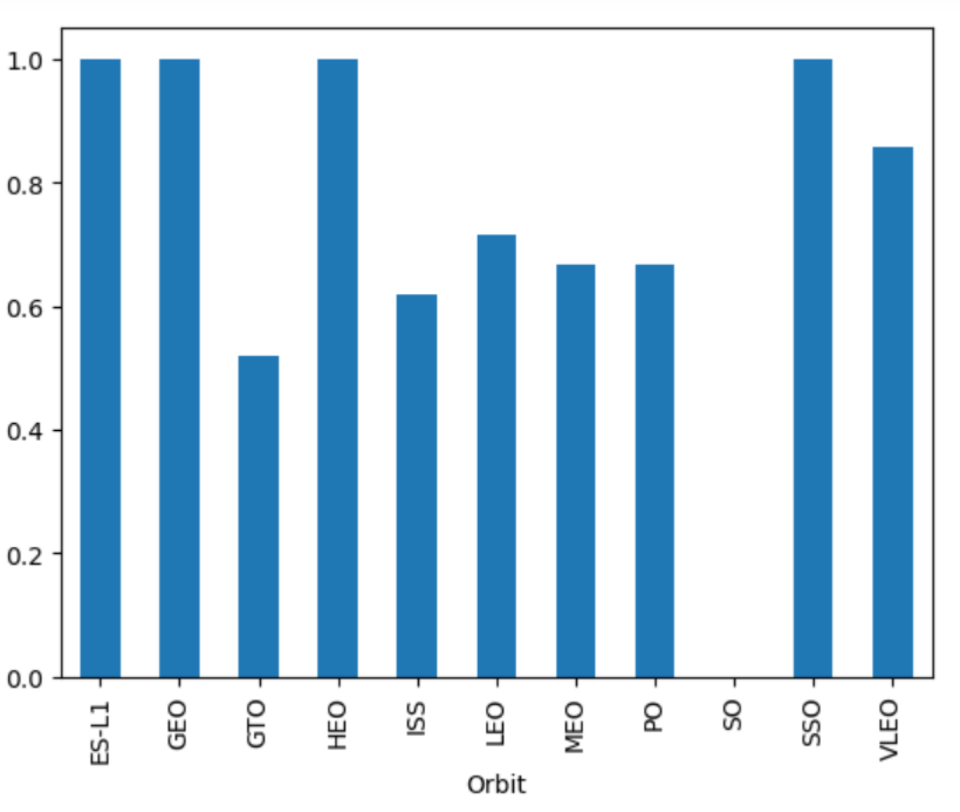
Payload vs. Launch Site

- The flights with higher Payload (above 8.000 kg) was more successful than the others.
- The success ratio is different across the Launch Sites.
- With “low” Payload (< 8.000 kg) the Launch site CCASF SCL 40 had the worst numbers.



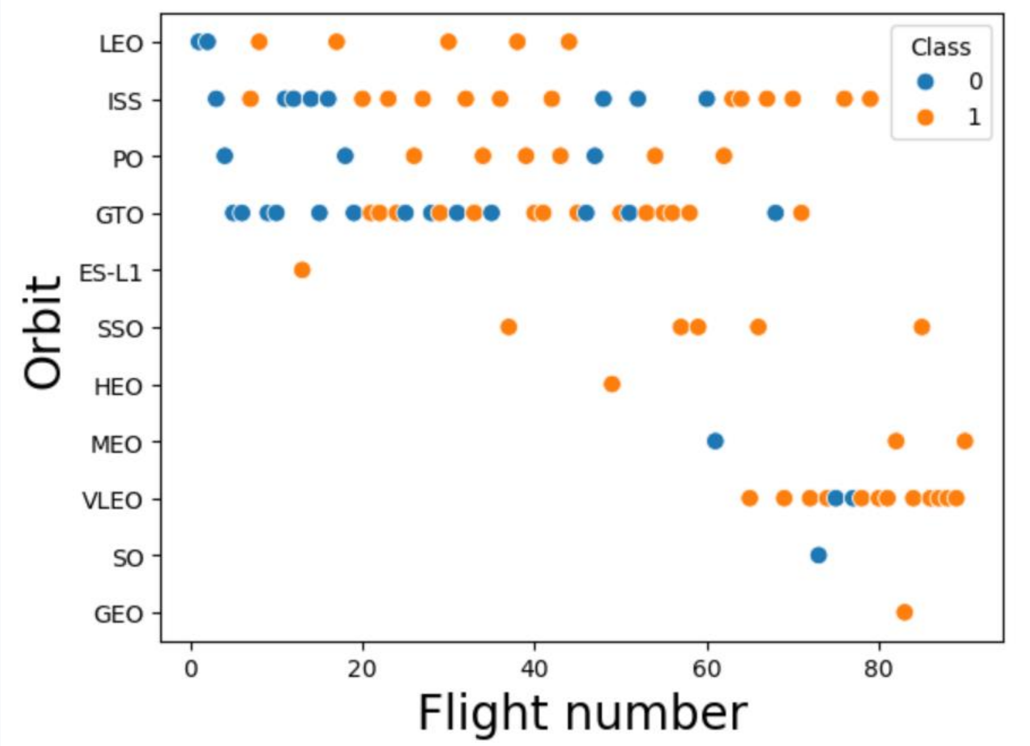
Success Rate vs. Orbit Type

- The success ratio its “very high” (>85%) in the following orbits:
  - ES-L1, GEO, HEO, SSO, VLEO
- The worsts success ratio (<60%) are in:
  - GTO, ISS, SO
- In particular, the orbit SO never has been a success.



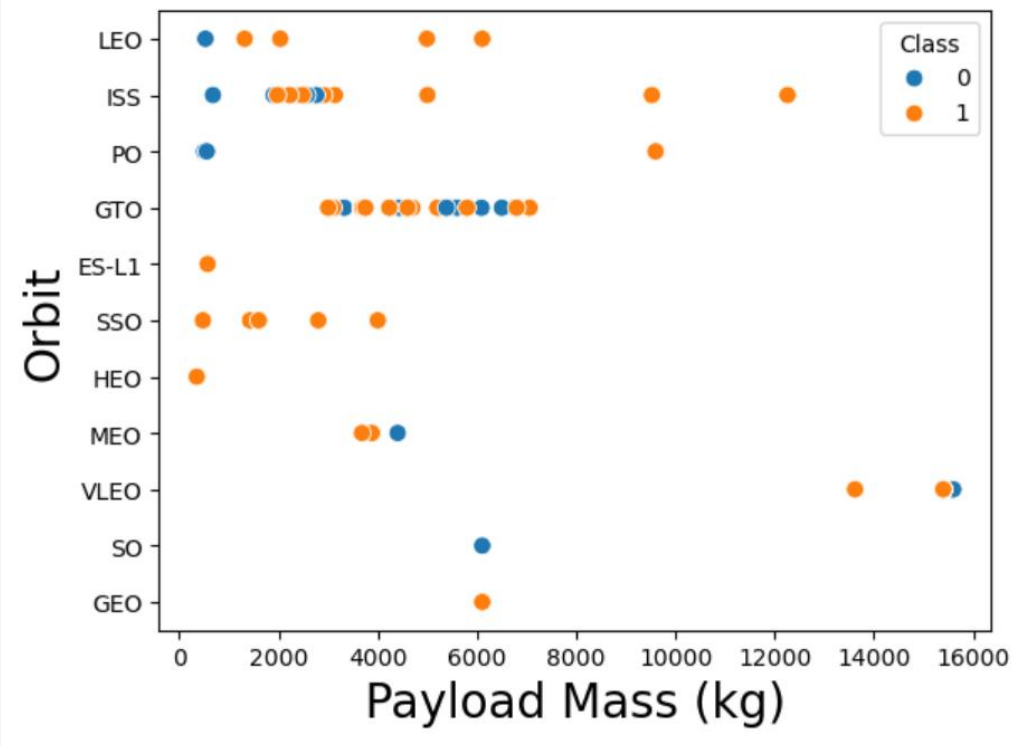
Flight Number vs. Orbit Type

- The lates flights are in general more successful. In particular in the Orbit VLEO, but success too in others, like: GEO, MEO, SSO, ISS.
- The orbit GTO has the worst success ratio.



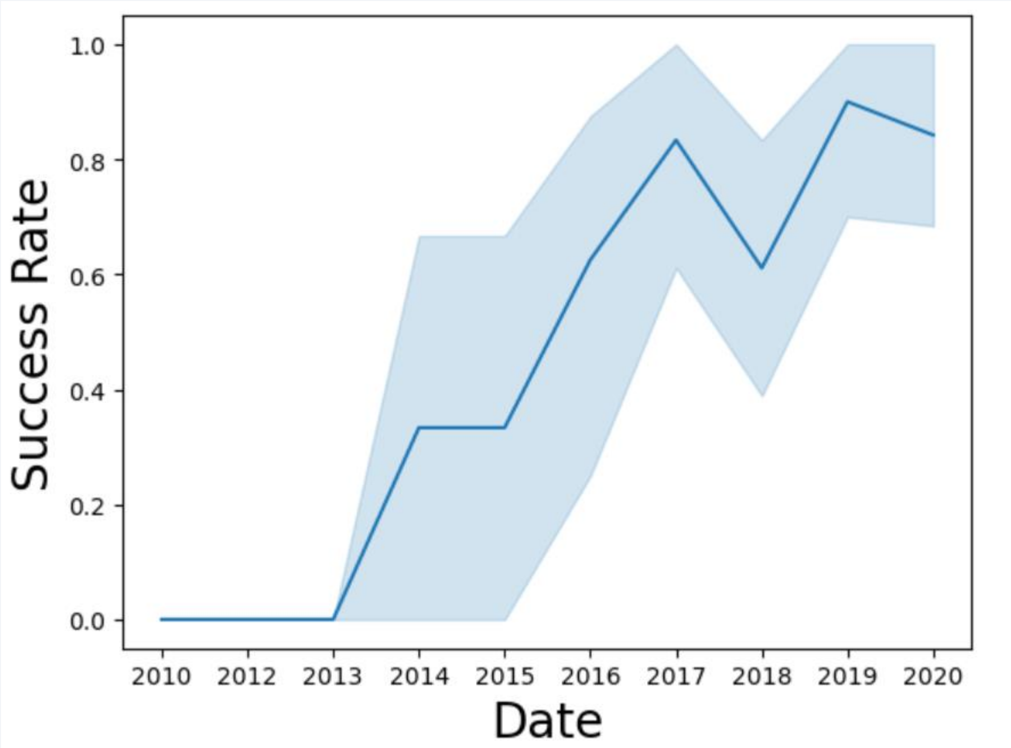
Payload vs. Orbit Type

- There is an orbit with high success ratio, independent the Payload Mass. Is the ISS.
- For “low” Payload Mass, the more success orbit was the SSO.
- For “high” Payload Mass, the success orbits was the ISS.
- Most of the orbits never has been tested with “high” Payload Mass (> 8.000 kg) Like:
    - GTO, SSO, LEO, MEO, HEO, etc



Launch Success Yearly Trend

- The success ratio is increasing since 2013 in a continuous way, except some problem in 2018.



All Launch Site Names

- There are four Launch Sites:

In [8]:
```
%sql SELECT DISTINCT Launch_Site FROM SPACEXTBL;
```

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

Out[8]:

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

Launch Site Names Begin with 'CCA'

- This Query show the first 5 sites which name begin with CCA:

```
In [27]: %sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5

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

Out[27]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
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

- The total Payload Mass carried by boosters launched, sum above 48 ton:

```
In [10]: %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER LIKE "%NASA (CRS)%"

 * sqlite:///my_data1.db
Done.

Out[10]:
```

SUM(PAYLOAD_MASS__KG_)
48213

Average Payload Mass by F9 v1.1

- The average Payload Mass carried by booster F9 v1.1 its close to 3 ton:

```
In [11]: %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version = "F9 v1.1"

 * sqlite:///my_data1.db
Done.

Out[11]:
```

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- The date of the first successful landing in ground pad was achieved on dic 2015:

```
In [18]: %sql SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome = "Success (ground pad)"

 * sqlite:///my_data1.db
Done.

Out[18]:
```

MIN(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- The name of boosters which have success (with Payload between 4 and 6 ton) was on the following query:

```
In [21]: %sql SELECT booster_version FROM SPACEXTBL WHERE "Landing_Outcome" = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AN

 * sqlite:///my_data1.db
Done.

Out[21]:
```

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

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failure mission, ie the total missions was 100:

```
In [29]: %sql select count(Mission_Outcome) from SPACEXTBL WHERE Mission_Outcome= 'Success' or Mission_Outcome = 'Failure (in flight)' or

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

Out[29]:

count(Mission_Outcome)
100

Boosters Carried Maximum Payload

- The following Query show us the name of boosters version which have carried the maximum payload mass:

```
In [30]: %sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)

 * sqlite:///my_data1.db
Done.

Out[30]:
```

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

- In 2015, on October (10) and April (4) was months with failure landing outcome in drone ship, booster versions or launch site:

In [33]:
```
%sql SELECT SUBSTR(Date,6,2) AS Month, Booster_Version, Launch_site FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Failure%drone%' AN
```

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

Out[33]:

Month	Booster_Version	Launch_Site
10	F9 v1.1 B1012	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40

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

- Between 2010-06-04 and 2017-03-20 there was the same number o Success (ground pad) and Failures (drone ship) landing:

```
In [47]: %sql SELECT Landing_Outcome, COUNT(*) AS Numbers FROM SPACEXTBL WHERE Landing_Outcome='Failure (drone ship)' or Landing_Outcome='

 * sqlite:///my_data1.db
Done.

Out[47]:
```

Landing_Outcome	Numbers
Success (ground pad)	5
Failure (drone ship)	5

Section 3

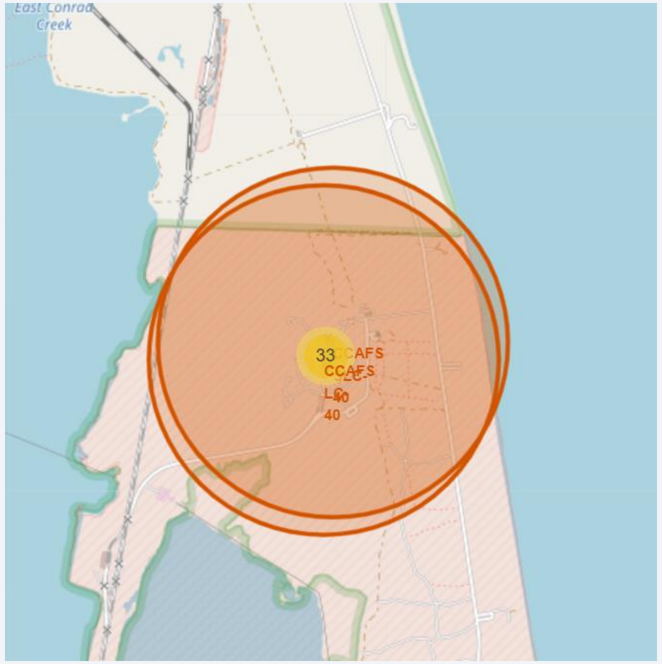
Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

- Map with Marked Launch Sites
  - One in the west coast (VAFB SCL 4E), two on the East Coast (KSC LC 39A and CCAFS SCL 40) and the others in the sea.

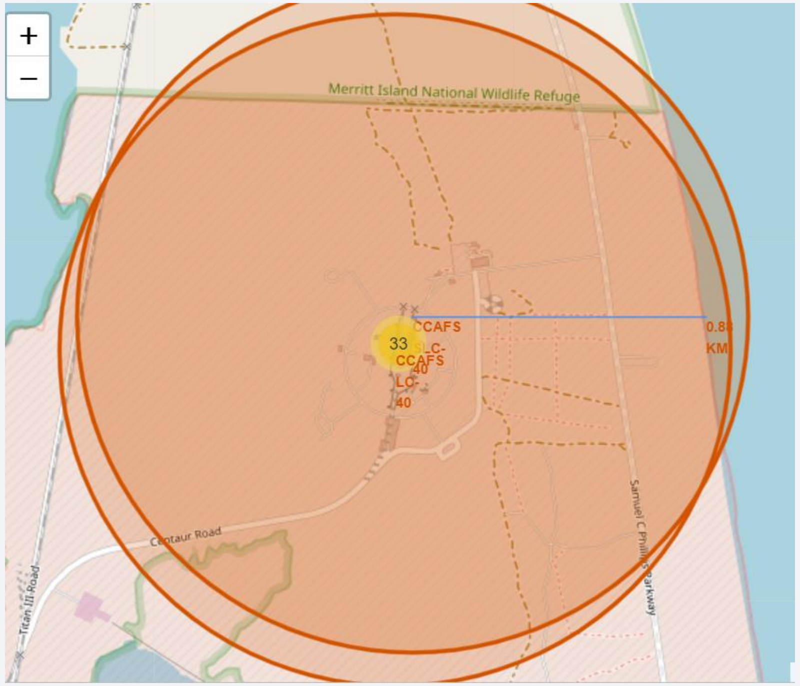
<Folium Map Screenshot 2>

- Map with Marked Successful and Failed Launches
  - The clusters in green show where was the successful Launches.



<Folium Map Screenshot 3>

- Map with the distances between a Launch Site to closer coastline
  - Launch Site – Coast : 0.87 km.




```
[15]: # find coordinate of the closet coastline
      # e.g.,: Lat: 28.56367  Lon: -80.57163
      # distance_coastline = calculate_distance(launch_site_lat, launch_site_lon, coastline_lat, coastline_lon)
      launch_site_lat, launch_site_lon = 28.563197, -80.576820
      coastline_lat, coastline_lon = 28.56319, -80.56785

      distance_coastline = calculate_distance(launch_site_lat, launch_site_lon, coastline_lat, coastline_lon)
      print(distance_coastline,' km')

      0.8762983388668404  km
```

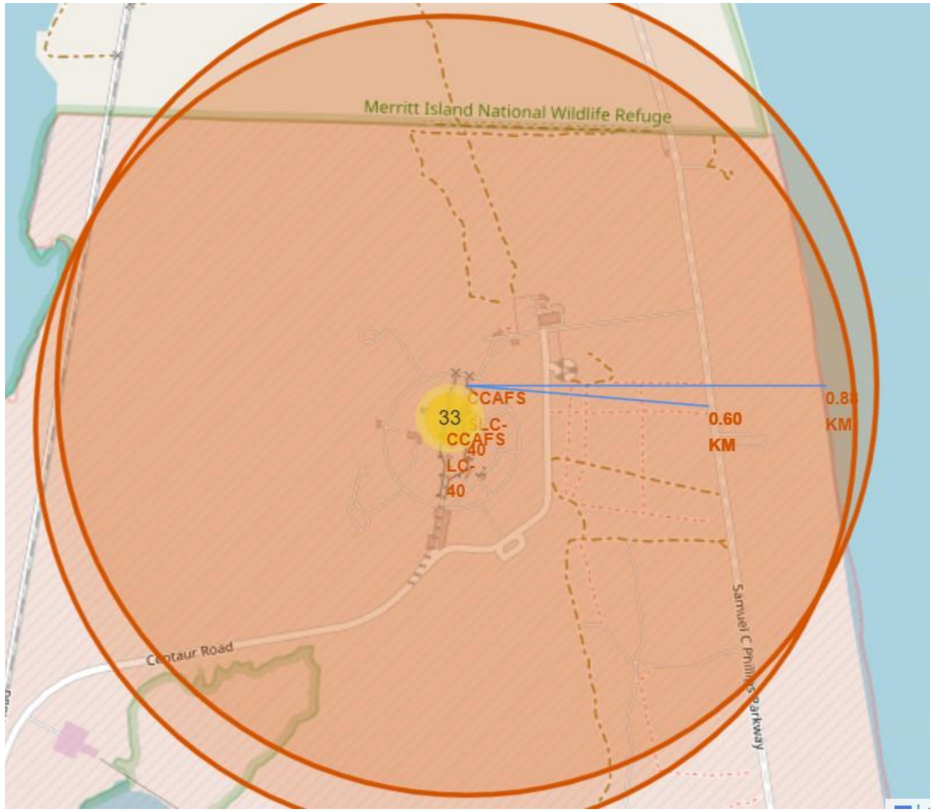
<Folium Map Screenshot 4>

- Map with the distances between a Launch Site to closer Highway
  - Launch Site – Highway : 0.59 km.

```
[20]: # Create a marker with distance to a closest city, railway, highway, etc.
# Draw a line between the marker to the launch site
launch_site_lat, launch_site_lon = 28.563197, -80.576820
highway_lat, highway_lon = 28.56272, -80.57075

distance_highway = calculate_distance(launch_site_lat, launch_site_lon, highway_lat, highway_lon)
print(distance_highway,' km')
```

```
0.595361102710318  km
```



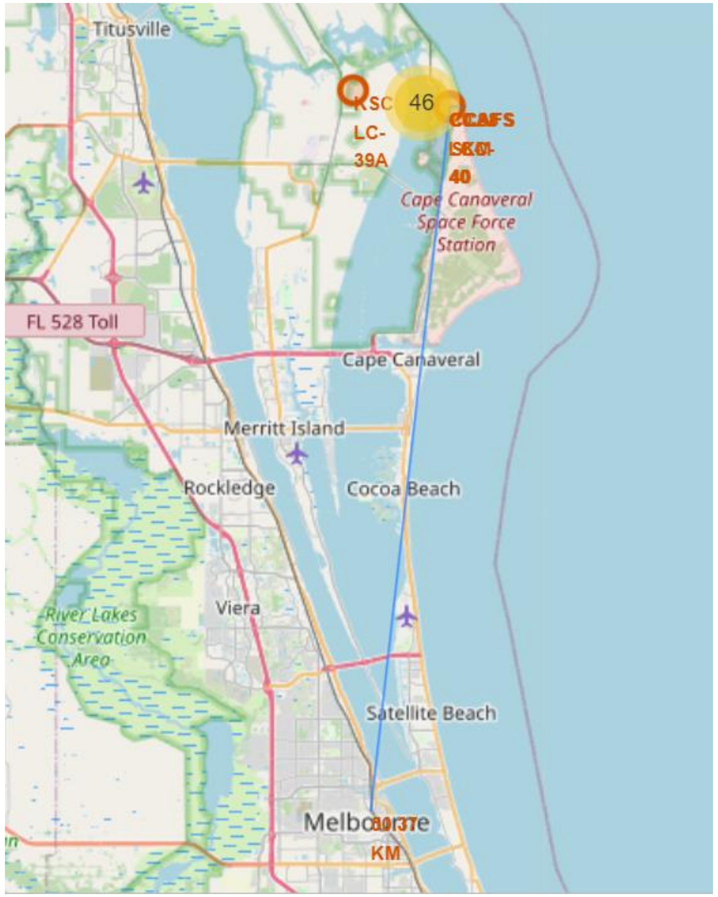
<Folium Map Screenshot 5>

- Map with the distances between a Launch Site to Melbourne
  - Launch Site – Melbourne : 50.3 km.

```
[26]: # Melbourne
launch_site_lat, launch_site_lon = 28.563197, -80.576820
Melbourne_lat, Melbourne_lon = 28.1132, -80.6340

distance_Melbourne = calculate_distance(launch_site_lat, launch_site_lon, Melbourne_lat, Melbourne_lon)
print(distance_Melbourne,' km')
```

```
50.3651514746636  km
```



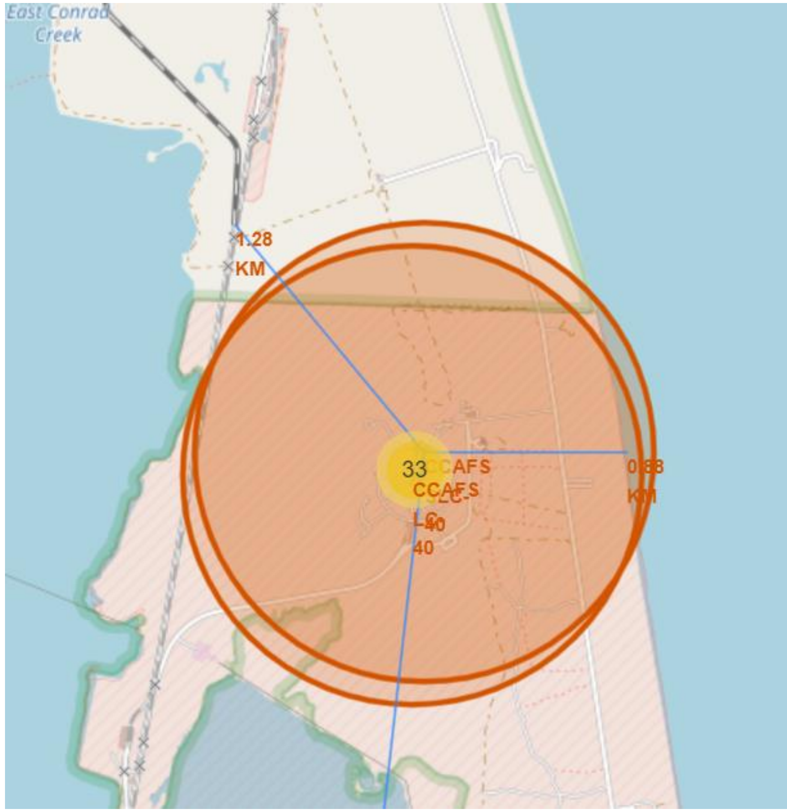
<Folium Map Screenshot 6>

- Map with the distances between a Launch Site to closer Railway
  - Launch Site – Railway : 1.28 km.

```
[20]: # Railway
launch_site_lat, launch_site_lon = 28.563197, -80.576820
Railway_lat, Railway_lon = 28.57205, -80.58527

distance_Railway = calculate_distance(launch_site_lat, launch_site_lon, Railway_lat, Railway_lon)
print(distance_Railway,' km')
```

```
1.2849353031381632  km
```

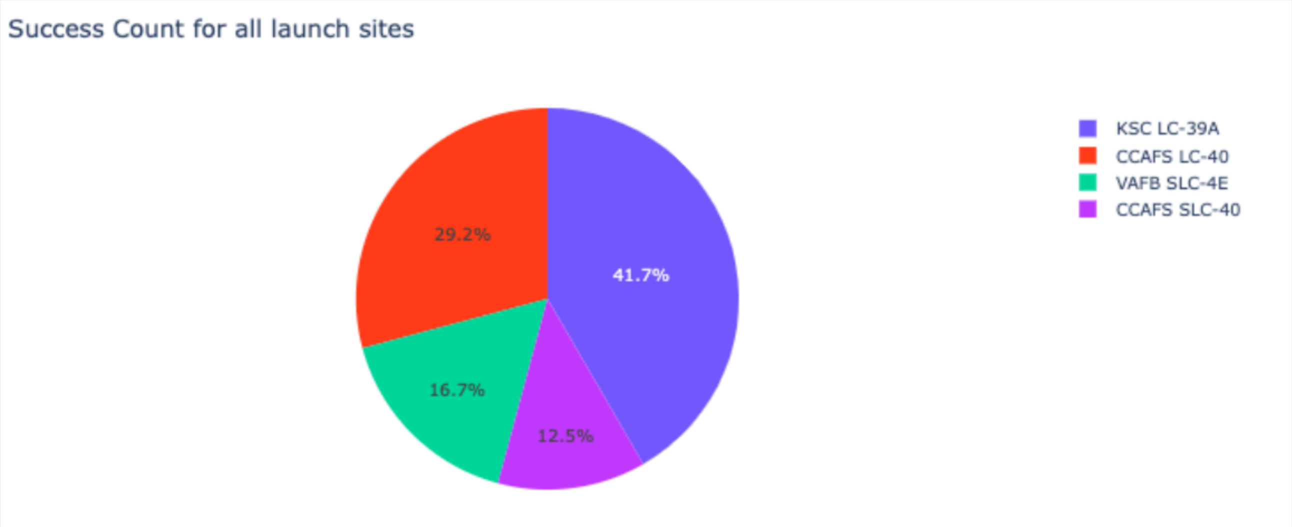


Section 4

Build a Dashboard with Plotly Dash

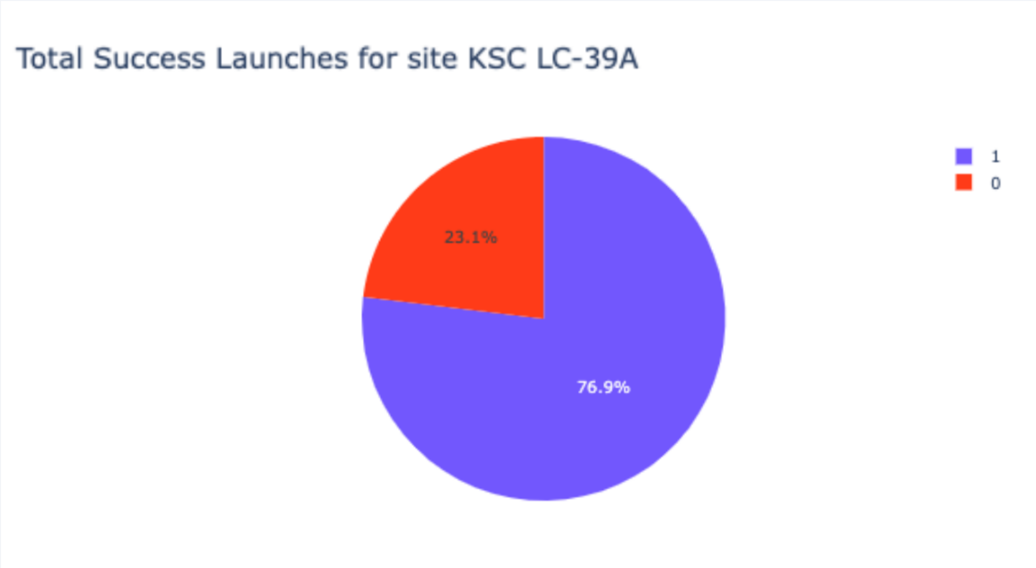
<Dashboard Screenshot 1>

- The Pie chart shows the success count for all launch sites.
- Based on that information, the site KSC LC-39A has the higher number of successful flight, followed by the site CCAFS LC-40



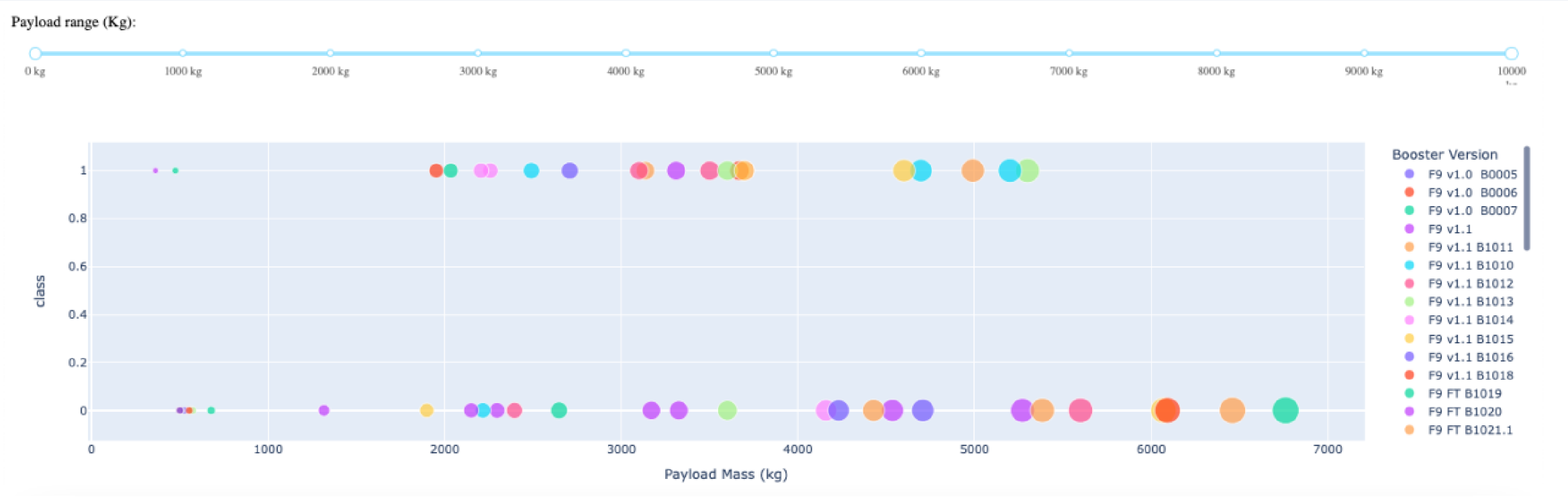
<Dashboard Screenshot 2>

- The Launch site KSC LC-39A had a success ratio of 77%.



<Dashboard Screenshot 3>

- The “light boosters” (< 2.000 kg) and the “heavy booster” (> 5.000 kg) have bad results in successful.
- In the “medium range” (2.000 – 5.000 kg) the results are not focus on particular booster version.

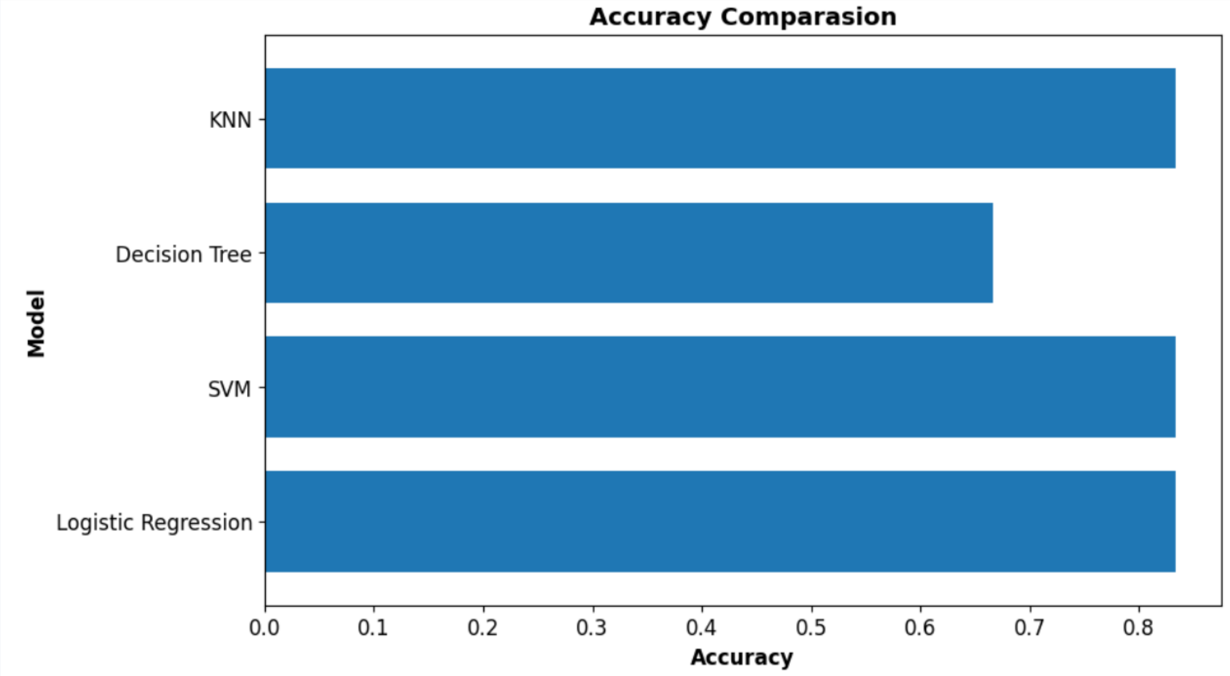


Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Comparing all models, 3 of them have the same accuracy (83,3%): Logistic Regression, SVM and KNN.
- Only Decision Tree was poor with 66,7% of accuracy




```
[39]: #We print the results to compare:
print('Logistic Regression', logreg_cv.score(X_test, Y_test))
print('SVM', svm_cv.score(X_test, Y_test))
print('Decision Tree', tree_cv.score(X_test, Y_test))
print('KNN', knn_cv.score(X_test, Y_test))

Logistic Regression 0.8333333333333334
SVM 0.8333333333333334
Decision Tree 0.6666666666666666
KNN 0.8333333333333334
```

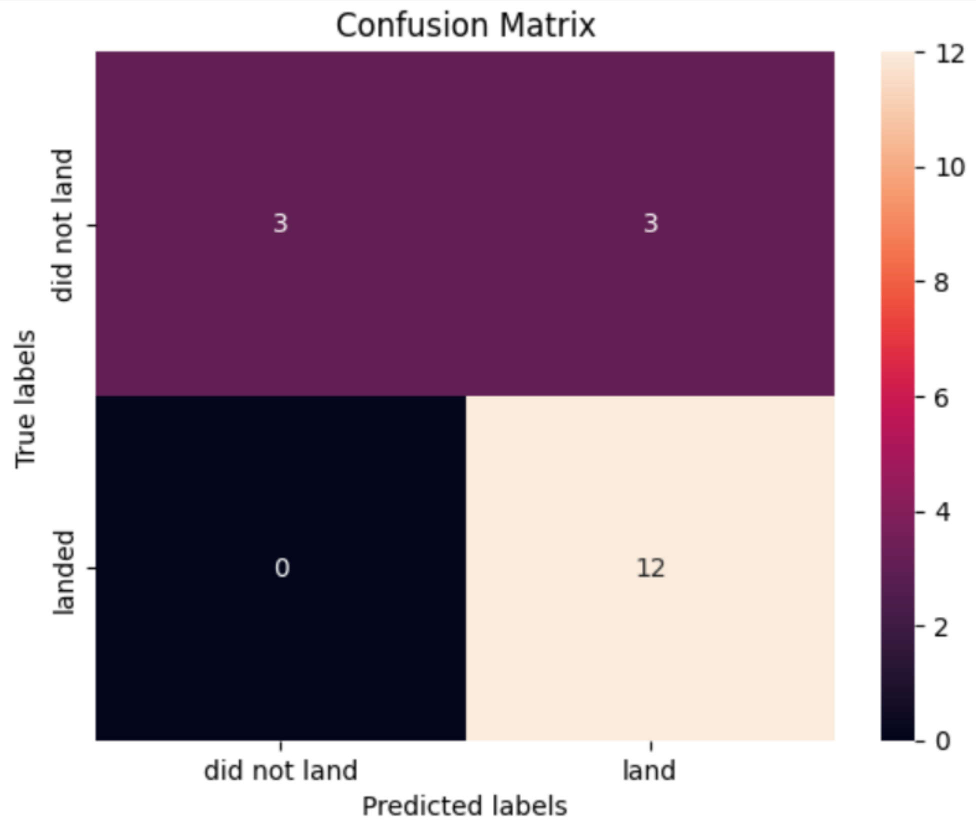
```
[40]: accuracy_dict = {'Model': ['Logistic Regression', 'SVM','Decision Tree','KNN'], 'Accuracy': [logreg_cv.score(X_test,
df_accuracy = pd.DataFrame(accuracy_dict )

from matplotlib import pyplot as plt

df_accuracy.plot(x="Model", y="Accuracy", kind="barh", width=0.75, figsize=(10, 6), fontsize=12, legend=None )
plt.title('Accuracy Comparasion', fontsize=14, fontweight='bold')
plt.ylabel('Model', fontsize=12, fontweight='bold')
plt.xlabel('Accuracy', fontsize=12, fontweight='bold')
plt.savefig('Accuracy_models')
```

Confusion Matrix

- In that case, the Confusion Matrix is the same for 3 models: Logistic Regression, SVM and KNN.
- The result show an accuracy of 83,3% with a 16% of failure prediction in the Landing of Rocket.



Conclusions

- Based on the data of the previous launches, and with the use of EDA and Predictive Analysis, we can determine the attributes to make a successful landing of the first stage rocket. The recommendations are:

  - Use “high” Payload Mass (i.e. >8.000 kg)
  - Use the Launch Site KSC LC-39A
  - Choose the Orbits: ES-L1, GEO, HEO, SSO, VLEO
  - Use “medium range” booster (i.e. 2.000 – 5.000 kg)

- With the characteristics give before, different models (Logistic Regression, SVM and KNN) make an successful prediction of launch with an accuracy of 83,3%.

Thank you!