IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Rolando Ferrer
15-June-2024

Outline

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

Executive Summary

This project demonstrates how data science methodologies can be applied to make space travel more competitive and affordable, a goal pursued by Space Y.

Summary of methodologies used in this Project

Data Collection from API

Web scraping

Exploratory Data Analysis

Interactive Visual Analytics & Dashboard

Predictive Analysis

Summary of key results in this project

Visualization of relationship between data points or features to be used for machine learning prediction

Prediction of first stage successful land based on given data to determine the cost of a launch

Introduction

Space Y, led by entrepreneur Allon Musk, aims to revolutionize space travel by making rocket launches significantly cheaper. Their strategy focuses on cost reduction through data analysis of each launch.

To achieve this, we leveraged data from SpaceX's Falcon 9 launches to develop a machine learning model. This model predicts the success of the first-stage landing, a crucial factor in enabling the reuse of these expensive components and ultimately driving down launch costs.

.

Problem Statement:

What are the key factors that makes a rocket land successfully?

What machine algorithms can be used to predict the outcome of the first landing of the future launches?

Section
1

Methodology

Methodology

Executive Summary

- **Data collection methodology:**
  - Data collection from SpaceX API and web scraping from Wikipedia launch tables
- **Data wrangling methodology:**
  - To prepare the data for analysis and visualization, we have used Pandas and NumPy for data cleaning and data transformation
- **Exploratory data analysis (EDA) using visualization and SQL:**
  - We leveraged visualization libraries like Matplotlib and Seaborn to explore and visualize the data
- **Interactive visual analytics using Folium and Plotly Dash:**
  - We built interactive dashboards for visual data analysis with Folium and Plotly Dash
- **Predictive analysis using classification models:**
  - We employed the scikit-learn library (Scikit-learn) for building predictive models. We utilized various algorithms like Logistic Regression, Decision Trees, and KNN. To assess the models' performance, we used confusion matrices, which visually compare the predicted results against the actual test data.

Data Collection

Data collection from SpaceX API and web scraping from Wikipedia launch tables

- Request from SpaceX API and use data from Spacex_url source
- Decoded response content to JSON format
- Normalized and convert JSON to Pandas dataframe
- Web scraping of Falcon 9 launch records from Wikipedia using Beautiful Soup
- Took a subset of the dataframe, including the features we will use for data analysis: rocket, payloads, launch pad, cores, flight number, and date columns

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"
In [7]: response = requests.get(spacex_url)
```

Check the content of the response

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
In [9]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datas
```

We should see that the request was successfull with the 200 status response code

```
In [10]: response.status_code
Out[10]: 200
```

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

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

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.

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

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

```
[6]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
     if response.status_code == 200:
       soup = BeautifulSoup(response.content, 'html.parser')
       # Now you can use the 'soup' object to navigate and extract data from the webpage
       # ...
     else:
       print("Error: Could not retrieve the content.")
```

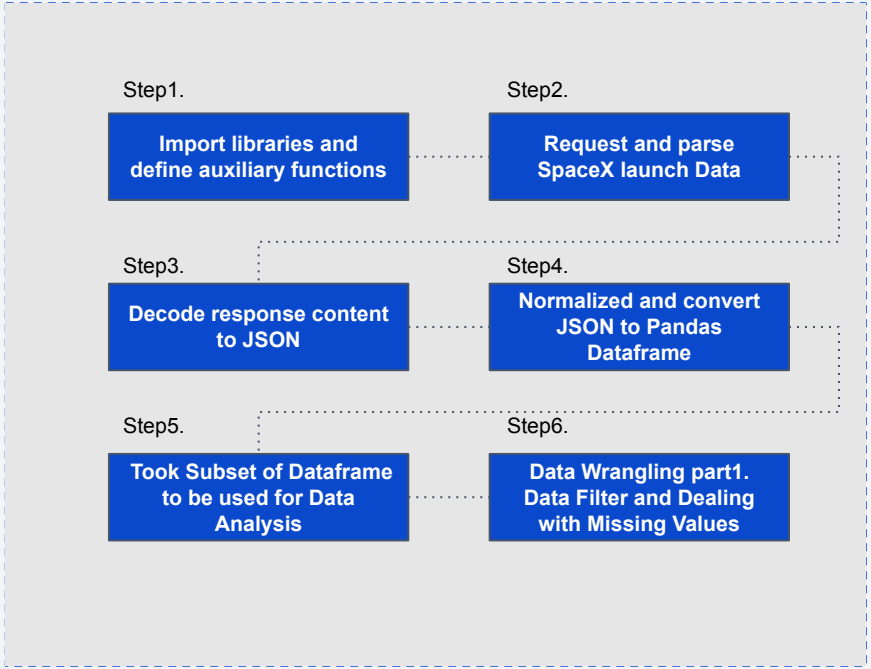
Print the page title to verify if the `BeautifulSoup` object was created properly

```
[7]: # Use soup.title attribute
     soup.title
:[7]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Data Collection – SpaceX API

Summary:

- Request from SpaceX API and use data from Spacex_url source
- Decoded response content to JSON format
- Normalized and convert JSON to Pandas dataframe
- Took a subset of the dataframe, including the features we will use for data analysis: rocket, payloads, launch pad, cores, flight number, and date columns



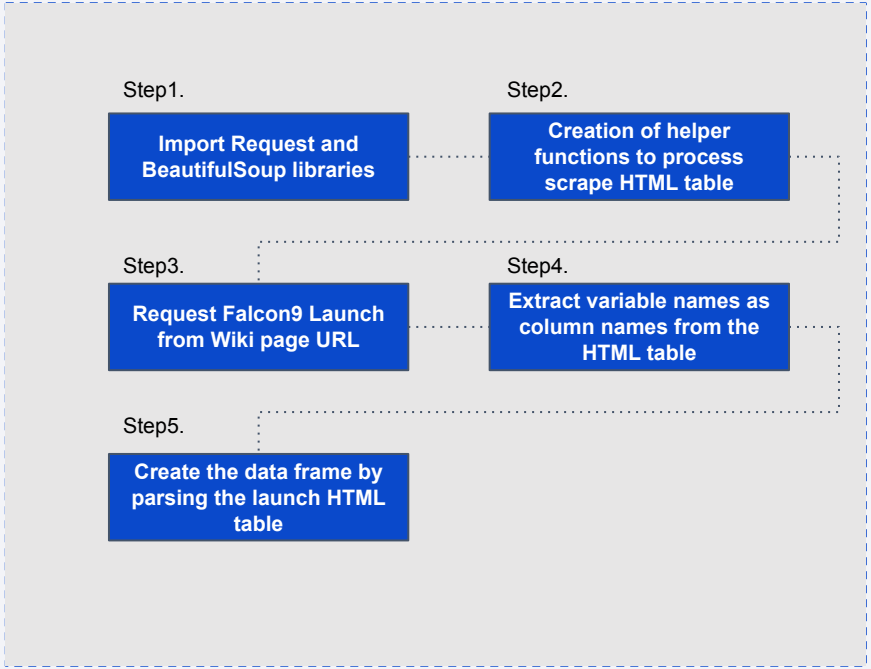
[My Github URL]

Data Collection - Scraping

Summary:

- Import Request and BeautifulSoup libraries
- Process scrape HTML data into tables with helper functions
- Extract column names from HTML table header
- Parse launch HTML table and create the data frame

[My Github URL]

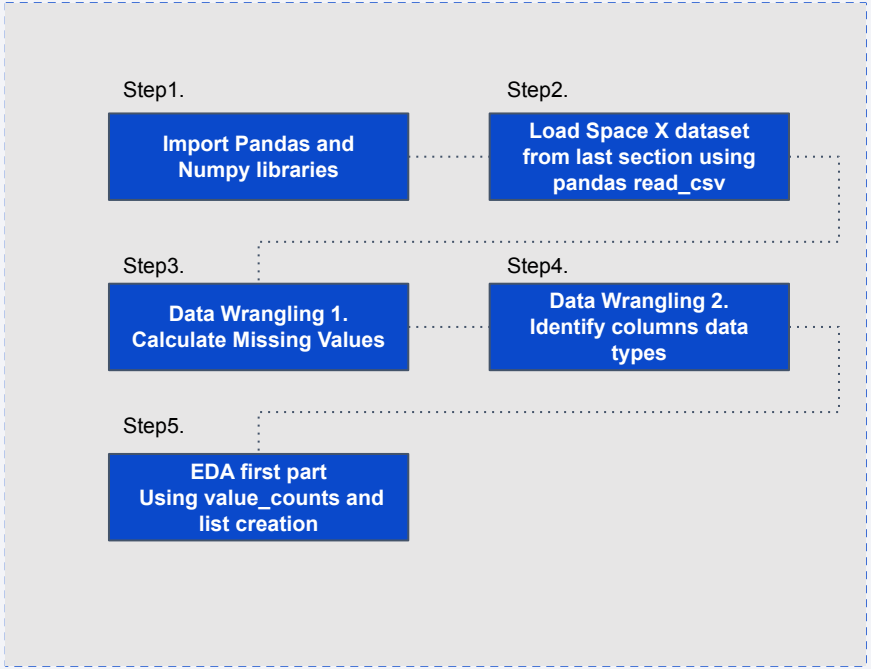


Data Wrangling

Summary:

- Import Pandas and Numpy libraries and define auxiliary functions
- Load Space X dataset
- Data cleansing, analysis and exploration using value_counts and creating list

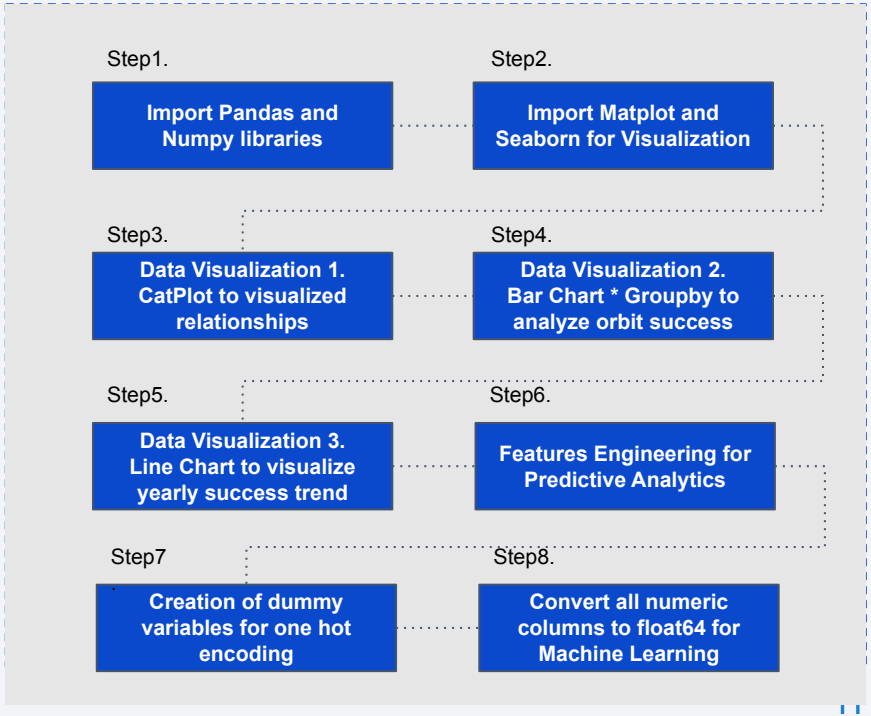
My Github



EDA with Data Visualization

Summary:

- Import Pandas, Numpy libraries and visualization libraries such as Matplotlib and SEAborn
- Use of CATPLOT to visualize relationships between features to be used for machine learning development to predict outcome
- Use of group Method and Bar Chart to identity success of each orbit
- Use of Line chart to visualize launch success yearly trend
- Feature engineering to be used for success prediction

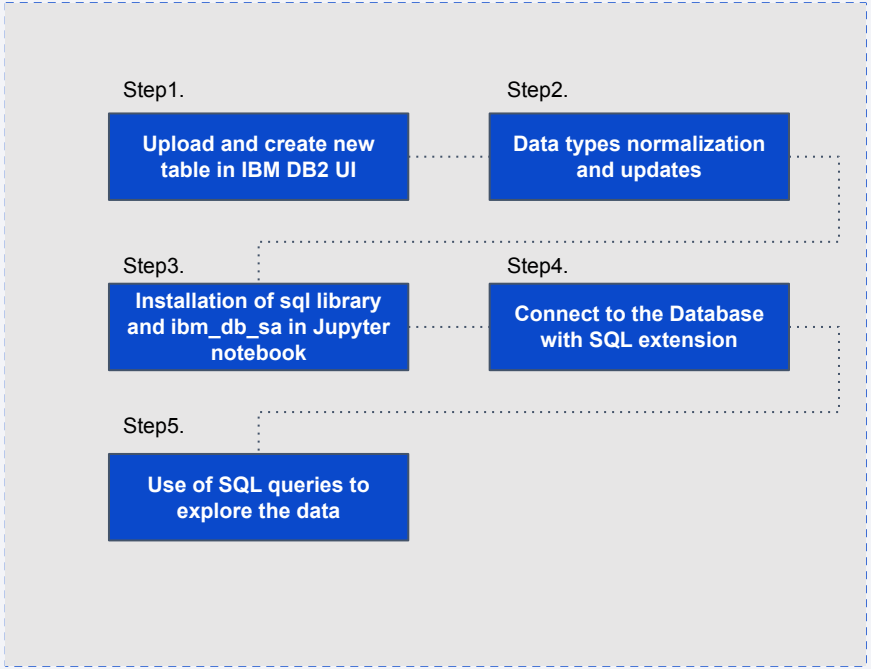


My Github URL

EDA with SQL

Summary:

- Load and create new table in IBM DB2 UI and name SPACEXTBL
- Change of Data types i.e., Date format and characters to integers
- Connect to DB using credentials
- Using SQL queries like Select, Unique, Where, Like, Limit, Between and calculations like Sum and Average



My Github URL

Build an Interactive Map with Folium

Map Objects:

- **Folium.Circle**
  To highlight the area of NASA Johnson Space Center
- **Folium.Map**
  To identify proximities within the equator line and coast
- **MarkCluster**
  To identify launch sites with high success rates
- **MousePosition**
  To highlight areas like railway, highway and coastlines
- **PolyLine**
  To highlight or visualize distance of launch site from selected coastline point

Sample Screenshot of adding MarkCluster into the map to identify sites with high success rates

From the color-labeled markers in marker clusters, you should be able to easily identify which launch sites have relatively high success rates.

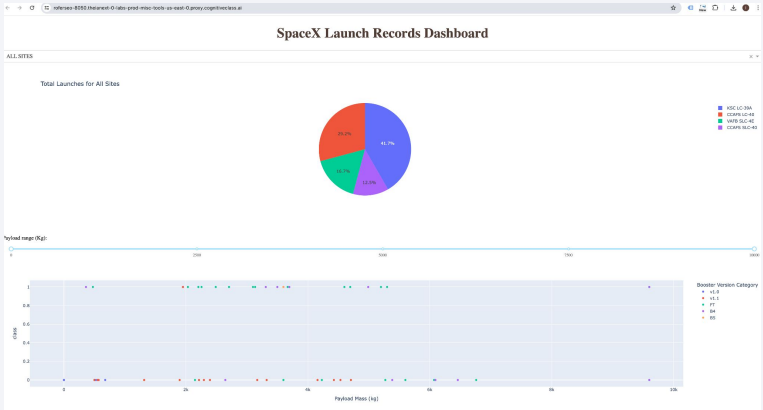
[My Github URL]

Build a Dashboard with Plotly Dash

Dashboard Objects:

- **Input Scroll Down**
  Filter to select launch sites
- **Pie Chart**
  Distribution of first stage landing between sites
- **Input Slider**
  Filter the payload masses for the scatter plot chart
- **Scatter Plot**
  Distribution of firsts stage landing by payload mass and outcome

Sample Screenshot of SpaceX Launch Dashboard

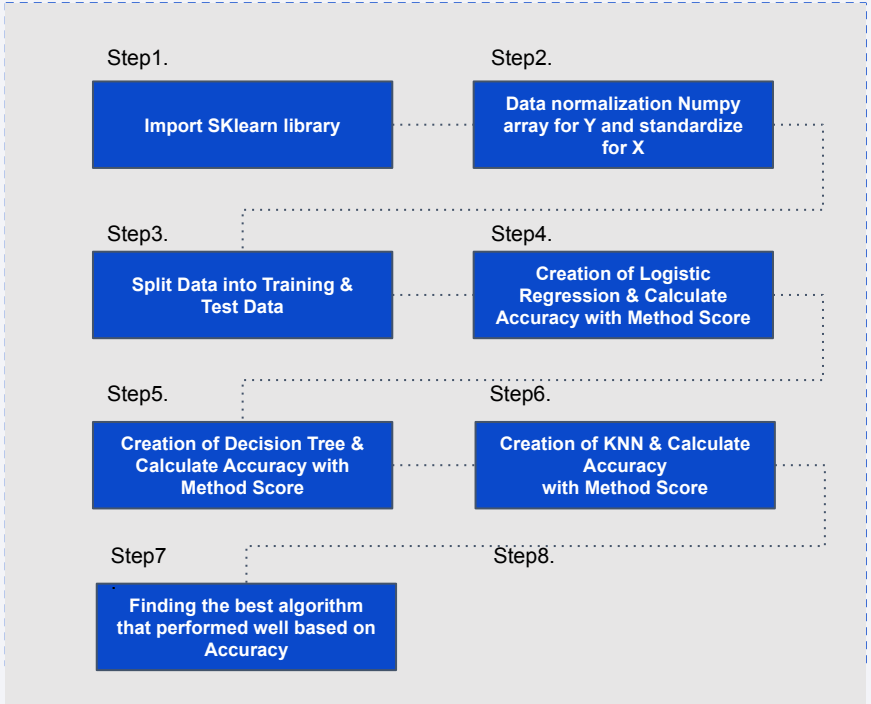


[My Github URL]

Predictive Analysis (Classification)

Models & Accuracy Test:

- **Import of SKlearn library**
  Import SKlearn for model creation
- **Create Numpy array for target variable Y**
  Applying the method to_numpy() for the output Pandas series
- **Standardize data in X**
  Standardize the data in X using fit_transform
- **Split Data into training and Test Data**
  Models are trained using the training data and validated using test data
- **Application of Classification ML algorithms**
  Use of Logistic Regression, SVM, Decision Tree and KNN and test accuracy using method score and visualize with confusion matrix chart



My Github URL

Results

EDA Results

[1] As the flight number increases, the first stage is more likely to land successfully

[2] ES-L1, GEO, HEO and SSO are obits with high success rate

[3] Success Rate Increasing from 2013 to 2017.

Interactive Analytics Results

[1] SpaceX launch sites are all located in United States America cost i.e., California and Florida

[2] Map markers shows mission outcomes (success/failure) for Falcon 9 first stage landings

[3] Distance from launch sites to proximities

Predictive Analysis Results

[1] All models performed well at 83% accuracy

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

Findings:

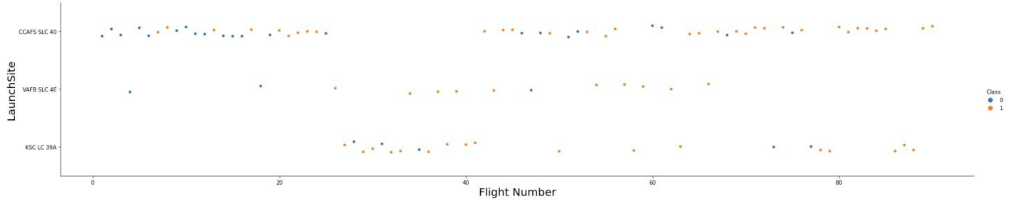
- Success rate varies per launch sites
- As seen in the graph there is a correlation between more flights and high success rates

Flight Number and Launch Site

TASK 1: Visualize the relationship between Flight Number and Launch Site

Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

In [7]:
```
# 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 cla
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



0 = failed landings | 1 = successful landings

Payload vs. Launch Site

Findings:

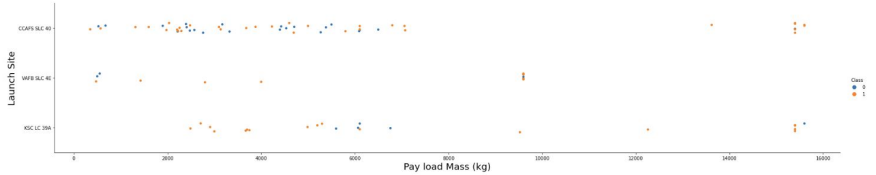
- VAFB-SLC launch site have no rocket launch for heavy payload exceeding 10000
- The heavy the payload for CCAFS SLC40 and KSC LC 39A the higher the success rate

Payload and Launch Site

TASK 2: Visualize the relationship between Payload and Launch Site

We also want to observe if there is any relationship between launch sites and their payload mass.

In [8]:
```
# 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 th
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



0 = failed landings | 1 = successful landings

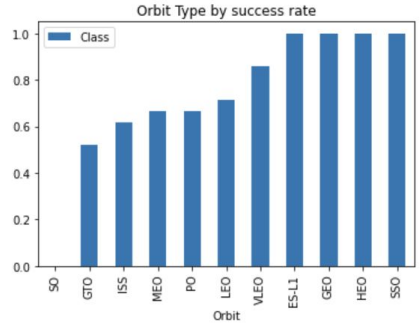
Success Rate vs. Orbit Type

Findings:

- ES-L1, GEO, HEO and SSO are obits with high success rates

Success Rate vs Orbit Type

```
In [17]: # HINT use groupby method on Orbit column and get the mean of Class column
         orbit_success = df.groupby(["Orbit"], as_index=False).agg({"Class":"mean"})
         orbit_success = orbit_success.sort_values(by="Class")

In [18]: orbit_success.plot.bar(x='Orbit', y='Class', title='Orbit Type by success rate')
         plt.show()
```



Flight Number vs. Orbit Type

Findings:

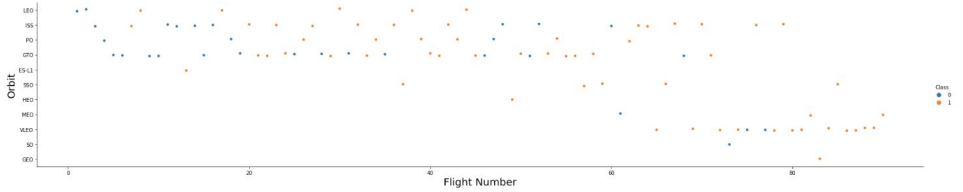
- LEO and VLEO orbit have high success rate based on the number of flights
- GTO shows no relationship between number of flights and success rates

Flight Number and orbit Type

TASK 4: Visualize the relationship between FlightNumber and Orbit type

For each orbit, we want to see if there is any relationship between FlightNumber and Orbit type.

In [19]:
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class valu
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



0 = failed landings | 1 = successful landings

Payload vs. Orbit Type

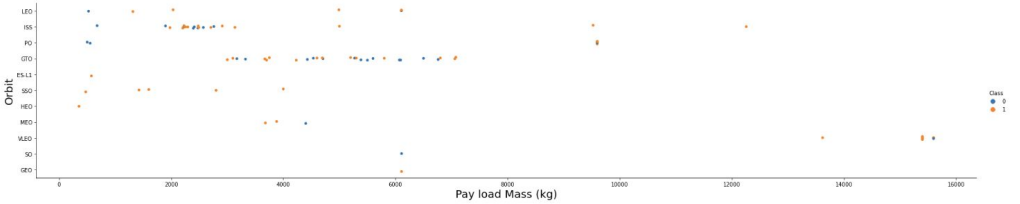
Findings:

- Polar, LEO and ISS have high landing success rate with heavy payload
- GTO shows no relationship between Payload Mass and Orbit Type

Payload and orbit Type

Similarly, we can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type

In [20]:
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



0 = failed landings | 1 = successful landings

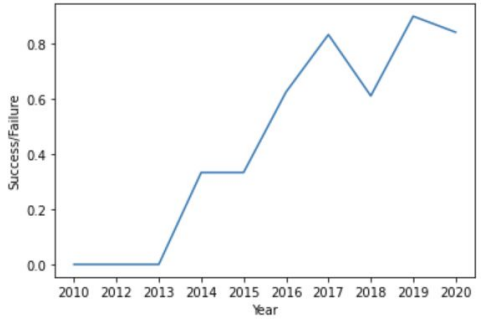
Launch Success Yearly Trend

Findings:

- Success Rate Increasing from 2013 to 2017. Rebounded in 2019 but slightly dropped in 2020

Success Rate YoY 2010 to 2020

```
In [29]: plt.plot(yearly_success_rate["Year"],yearly_success_rate["Class"])
         plt.xlabel("Year")
         plt.ylabel("Success/Failure")
         plt.show()
```



All Launch Site Names

Process:

- Use of the command **Unique** to show only distinct launch site from the SPACEX table

```
In [6]: %sql select Unique(LAUNCH_SITE) from SPACEXTBL;
```

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

Launch Site Names Begin with 'CCA'

Process:

- Use of the command **Select**, **Where** and **Limit** to show only launch site names that begins with CCA and display the first five records

```
In [7]: %sql SELECT LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

Process:

- Us of the command **SELECT, SUM function and LIKE** to get the total payload carried by boosters from NASA (CRS)
- Answer: **48213**

```
%sql select sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL WHERE Customer LIKE '%NASA (CRS)%';
```

payloadmass
48213

Average Payload Mass by F9 v1.1

Process:

- Use of the command **SELECT, avg function and LIKE to** get the average payload carried by boosters version F9 v1.1
- Answer: **6138**

```
%sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL WHERE Booster_Version LIKE 'F9 v1.1';
```

payloadmass
6138

First Successful Ground Landing Date

Process:

- Use of the command **SELECT, avg function and LIKE to** get the average payload carried by boosters version F9 v1.1
- Answer: **2015-12-22**

```
: %sql select min(DATE) as "First Succesful Landing Outcome Date" from SPACEXTBL WHERE Landing_Outcome LIKE 'Success (
```

First Successful Landing Outcome Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Process:

- Use of the command **SELECT, WHERE AND BETWEEN** to get **list of Booster Versions**
- Answer:

Launch_Site
CCAFS LC-40
CCAFS LC-40
KSC LC-39A
KSC LC-39A

Connection from DB2 interrupted. Manually check the results from excel file

Query must be SELECT Booster_Version from SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS_KG BETWEEN 4000 and 6000;

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

Connection from DB2 interrupted. No Answer for this part

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

Connection from DB2 interrupted. No Answer for this part

2015 Launch Records

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

- Present your query result with a short explanation here

Connection from DB2 interrupted. No Answer for this part

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

- Present your query result with a short explanation here

Connection from DB2 interrupted. No Answer for this part

Section 3

Launch Sites Proximities Analysis

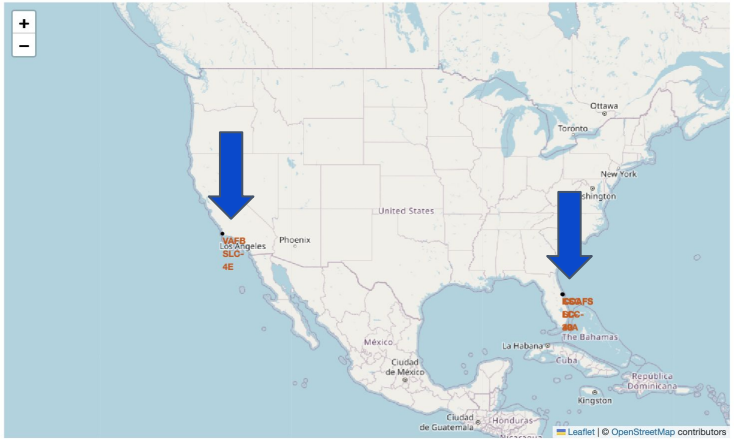
All Launch Sites

Summary:

- SpaceX launch sites are all located in United States America cost i.e., California and Florida

In [8]:
```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch site
for index, row in launch_sites_df.iterrows():
    coordinate = [row['Lat'], row['Long']]
    folium.Circle(coordinate, radius=1000, color='#000000', fill=True).add_child(folium.Popup(row['Launch Site'])).a
    folium.map.Marker(coordinate, icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0), html='<div style="font-size: 12;
site_map
```

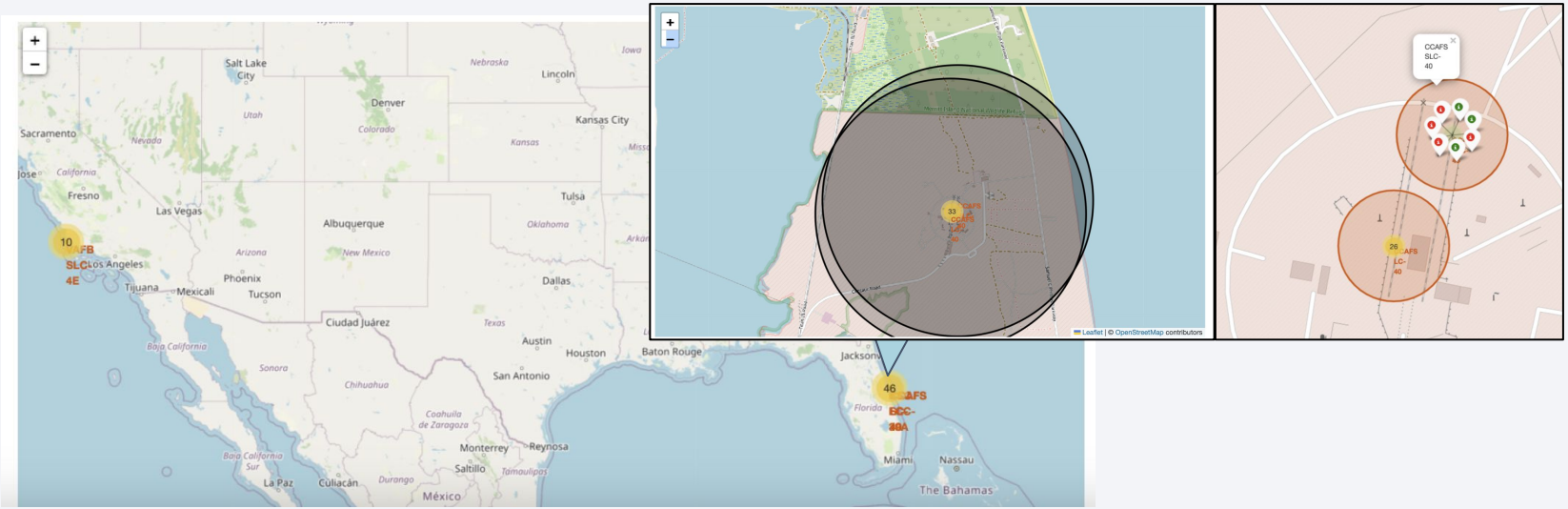
Out[8]:



Markers showing launch sites with fail & success labels

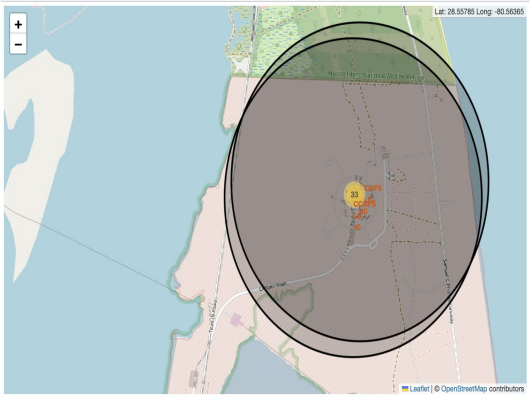
Zoom-in

Successful VS Failed launches

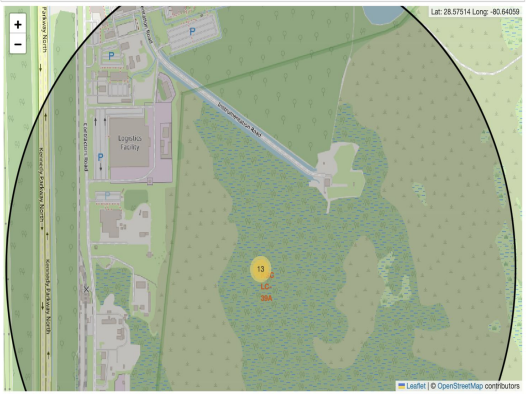


Launch Site Proximity to Landmarks

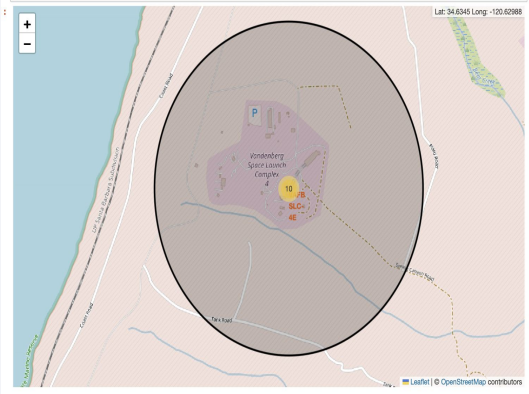
Below are the launch sites with close proximity to cities, railways and coast lines



Close Proximity to Coastlines & Cities



Close Proximity to Railways



Close Proximity to Highways

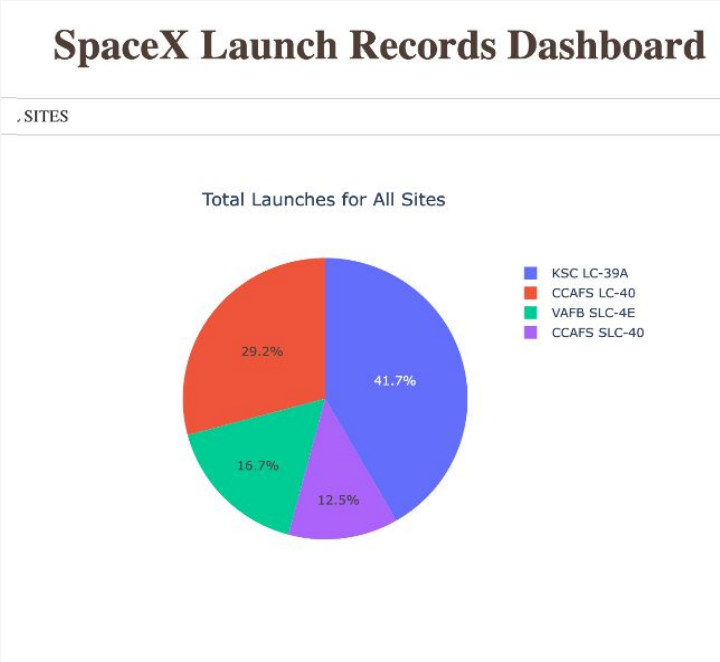
Section 4

Build a Dashboard with Plotly Dash

Success PCT each launch sites

Summary:

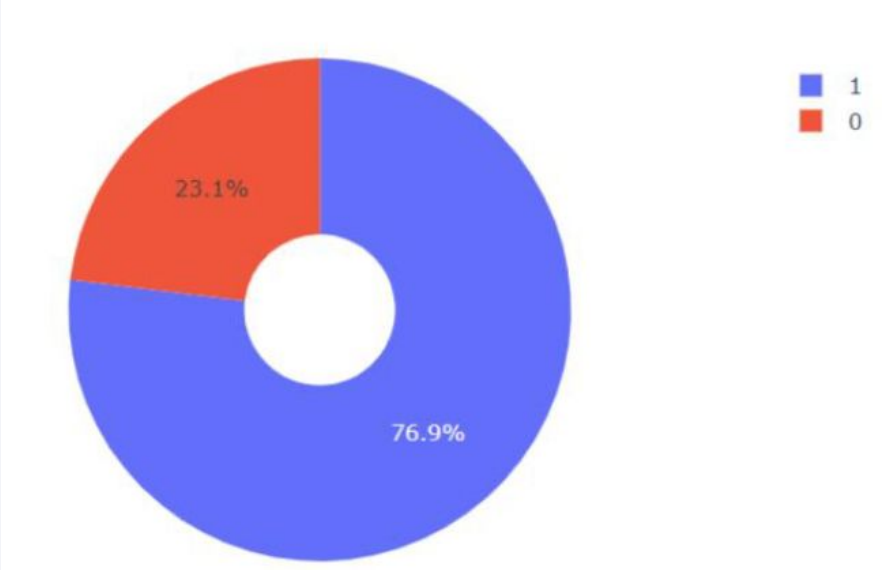
- This pie chart shows the success percentage of each launch site where KSC LC-39A shows the highest with 41.7% followed by CCAFS LC-40 which us at 29%



KSC LC-39A success rate

Summary:

- KSC LC-39A success rate is at 76.9%



Payload vs Launch Outcome

Highest success rate from KSC LC-39A with majority belongs to FT Category and B4 Category

Section
5

Predictive Analysis (Classification)

Classification Accuracy

Summary:

- All models performed well at 83% accuracy

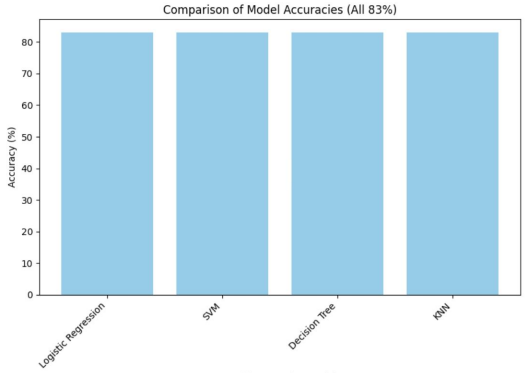
```
[32]: knn_cv.score(X_test, Y_test)
[32]: 0.8333333333333334
```

```
[18]: logreg_cv.score(X_test, Y_test)
[18]: 0.8333333333333334
```

```
[27]: tree_cv.score(X_test, Y_test)
[27]: 0.8333333333333334
```

```
[22]: svm_cv.score(X_test, Y_test)
[22]: 0.8333333333333334
```

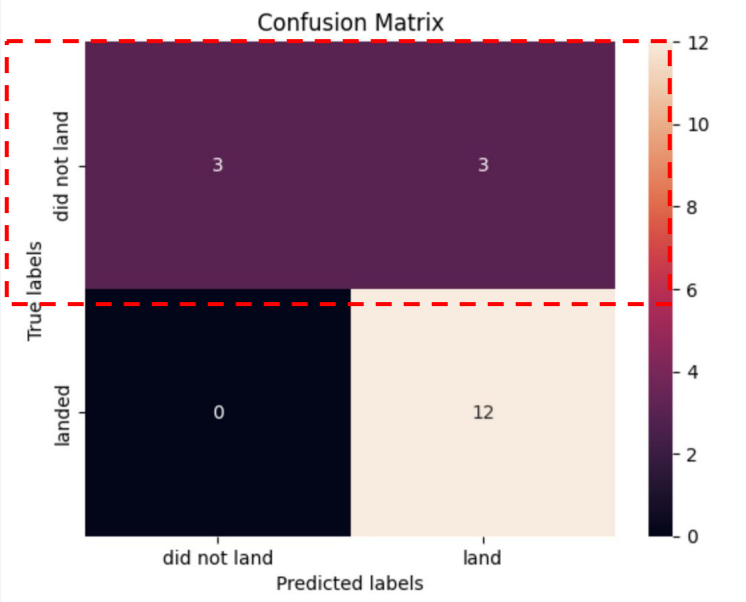
```
import pandas as pd
import matplotlib.pyplot as plt

# Create a DataFrame
data = {'Model': ['Logistic Regression', 'SVM', 'Decision Tree', 'KNN
df = pd.DataFrame(data)

# Create a bar chart
plt.figure(figsize=(8, 6))
plt.bar(df['Model'], df['Accuracy'], color='skyblue')
plt.xlabel('Machine Learning Model')
plt.ylabel('Accuracy (%)')
plt.title('Comparison of Model Accuracies (All 83%)')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()

# Display the chart
plt.show()
```



Confusion Matrix

Summary:

- All models performed at 83% accuracy with identical Confusion Matrix results
- Further fine tuning of algorithms is needed due to 3 unsuccessful lands shows successful landing



Conclusions

- As the flight number increases, the first stage is more likely to land successfully
- ES-L1, GEO, HEO and SSO are orbits with high success rate
- Success Rate Increasing from 2013 to 2017, rebounded in 2019 but slightly drop in 2020
- SpaceX launch sites are all located in United States America coast i.e., California and Florida, and there are launch sites that are close to city areas
- KSC LC-39A shows the highest successful launches versus other launch sites
- All models performed well (LG, SVM, Decision Tree and KNN) with 83% accuracy rate.
- All models can be used to predict success of the first launches and determine the cost for a competitive pricing for Space Y

Thank you!