IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sanghamitra Senapati
01/02/2024

Outline

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

Executive Summary

This capstone project focuses on predicting the successful landing of the SpaceX Falcon 9 first stage using various machine learning classification algorithms. The key steps involved are:

- Data Collection
- Data Wrangling
- Exploratory Data Analysis (EDA) with Visualization
- Exploratory Data Analysis using SQL
- Interactive Mapping with Folium
- Dashboard Creation with Plotly Dash
- Predictive Analysis (Classification)

Our analysis reveals that certain features of rocket launches correlate with their outcomes—whether they succeed or fail. Furthermore, the decision tree algorithm emerges as the most effective model for predicting the successful landing of the Falcon 9 first stage.

Introduction

- Project background and context: SpaceX is the most successful company of the commercial space age, making space travel affordable. The company 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 SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. Based on public information and machine learning models, we are going to predict if SpaceX will reuse the first stage.
- Most unsuccessful landings are planned. Sometimes, SpaceX will perform a controlled landing in the ocean.
- Questions to be answered - How do variables such as payload mass, launch site, number of flights, and orbits affect the success of the first stage landing? - Does the rate of successful landings increase over the years? - What is the best algorithm that can be used for binary classification in this case?

Section 1

Methodology

Methodology

- The overall methodology includes:
  - Data collection, wrangling, and formatting, using:
    - SpaceX API
    - Web scraping
  - Exploratory data analysis (EDA), using:
    - Pandas and NumPy
    - SQL
  - Data visualization, using:
    - Folium
    - Dash
  - Machine learning prediction, using
    - Logistic regression
    - Decision tree
    - K-nearest neighbors (KNN)

Data Collection

- The data collection process utilized a combination of SpaceX's REST API and web scraping from a table on SpaceX's Wikipedia page. This dual approach was necessary to compile comprehensive information on launches for a detailed analysis.
- SpaceX API
  The API endpoint used was: https://api.spacexdata.com/v4/rockets/.
  This API provides extensive data on various rocket launches by SpaceX. To focus the analysis, the data was filtered to include only Falcon 9 launches.
- Any missing values in the dataset were imputed using the mean value of the respective column to ensure data completeness.
- For further details, refer to the github link: https://github.com/greatcourse/Final_Capstone/blob/main/LAB1_API_Collect%20the%20data_part1.ipynb

Next 2 slides will provide the flow chart.

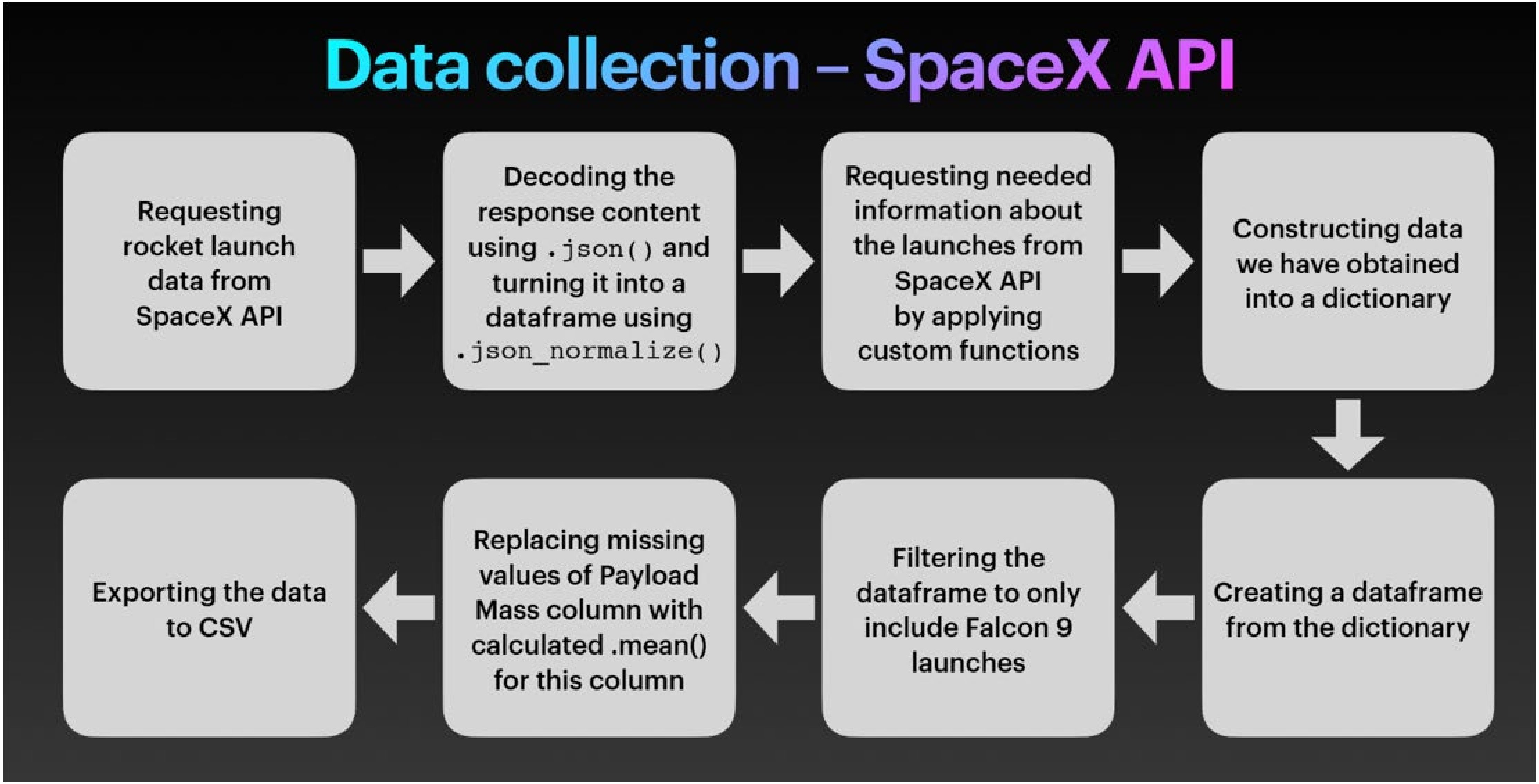


Figure Source: Internet

SpaceX API

1. Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

2. Convert Response to JSON File

```
data = response.json()
data = pd.json_normalize(data)
```

3. Transform data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
getBoosterVersion(data)
```

4. Create dictionary with data

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

5. Create dataframe

```
data = pd.DataFrame.from_dict(launch_dict)
```

6. Filter dataframe

```
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

7. Export to file

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

Figure Source: Internet

Data Collection:Web Scraping

- The data was collected by scraping information from the following webpage: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922.
- This page exclusively contains details about Falcon 9 launches. After processing the data, we obtained a dataset with 121 rows (instances) and 11 columns (features).
- For more information, refer to the GitHub repository: https://github.com/greatcourse/Final_Capstone/blob/main/jupyter-labs-webscraping_final.ipynb

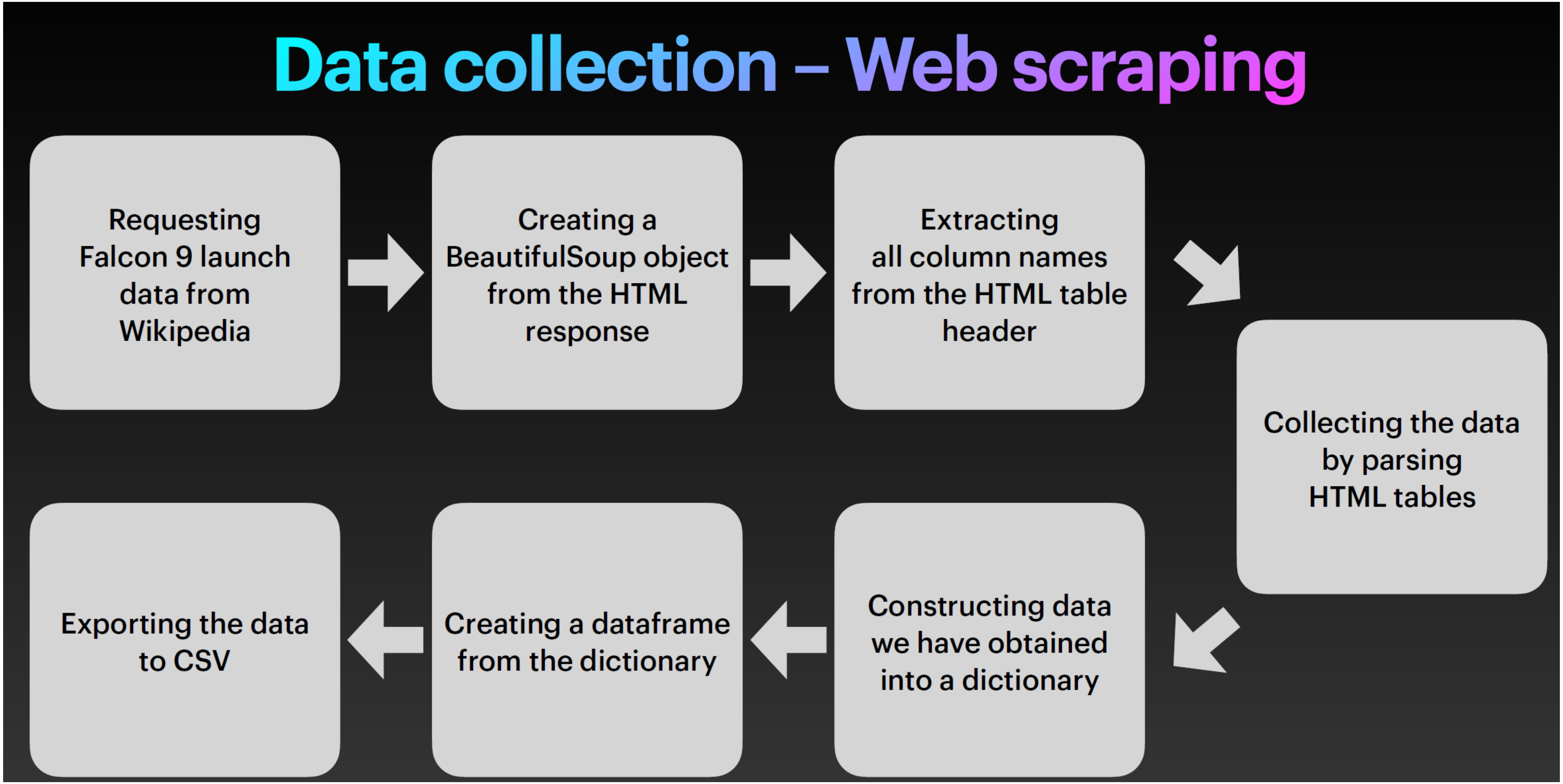


Figure Source: Internet

Data Collection - Scraping

1. Getting Response from HTML

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

2. Create BeautifulSoup Object

```
soup = BeautifulSoup(response.text, "html5lib")
```

3. Find all tables

```
html_tables = soup.findAll('table')
```

4. Get column names

```
for th in first_launch_table.find_all('th'):
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0 :
        column_names.append(name)
```

5. Create dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

6. Add data to keys

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is a
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.stri
                flag=flight_number.isdigit()
```

See notebook for the rest of code

7. Create dataframe from dictionary

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

8. Export to file

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

Figure Source : Internet

Data Wrangling

In the dataset, there are various cases where the booster did not land successfully. For example:

- **True Ocean**: The mission outcome indicates a successful landing in a specific region of the ocean.
- **False Ocean**: The mission outcome indicates an unsuccessful landing in a specific region of the ocean.
- **True RTLS**: The mission outcome indicates a successful landing on a ground pad (Return To Launch Site).
- **False RTLS**: The mission outcome indicates an unsuccessful landing on a ground pad.
- **True ASDS**: The mission outcome indicates a successful landing on a drone ship (Autonomous Spaceport Drone Ship).
- **False ASDS**: The mission outcome indicates an unsuccessful landing on a drone ship.

For simplicity and consistency, these outcomes were converted into training labels:

- **"1"**: Indicates the booster successfully landed.
- **"0"**: Indicates the booster landing was unsuccessful.

Data Wrangling

1. Calculate launches number for each site

```
df['LaunchSite'].value_counts()
```

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

2. Calculate the number and occurence of each orbit

```
df['Orbit'].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
SO        1
ES-L1     1
HEO       1
GEO       1
Name: Orbit, dtype: int64
```

3. Calculate number and occurrence of mission outcome per orbit type

```
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

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

4. Create landing outcome label from Outcome column

```
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
df['Class']=landing_class
```

5. Export to file

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

For further details, refer to the github link:
https://github.com/greatcourse/Final_Capstone/blob/main/mod1_lab3-spacex-Data%20wrangling.ipynb

Figure Source: Internet

EDA with Data Visualization

Pandas and NumPy

- Using functions from the Pandas and NumPy libraries, basic insights about the collected data are derived, including:
- The total number of launches per launch site.
- The frequency of each orbit type.
- The count and frequency of various mission outcomes.

Matplotlib and Seaborn

Functions from the Matplotlib and Seaborn libraries were used to create scatterplots, bar charts, and line charts to visualize the data. These visualizations help in exploring relationships between various features.

EDA with Data Visualization

Various charts were created to visualize the data, including:

- Flight Number vs. Payload Mass
- Flight Number vs. Launch Site
- Payload Mass vs. Launch Site
- Orbit Type vs. Success Rate
- Flight Number vs. Orbit Type
- Payload Mass vs. Orbit Type
- Success Rate Yearly Trend

EDA with Data Visualization

- Scatter plots were used to examine the relationships between variables, which could be leveraged in machine learning models if a significant relationship is identified.

- Bar charts provided comparisons among discrete categories, highlighting the relationship between specific categories and their corresponding measured values.

- Line charts were utilized to illustrate trends over time, offering insights into temporal patterns (time series).

For further details, refer to the github link:
https://github.com/greatcourse/Final_Capstone/blob/main/Data%20Visualization.ipynb

EDA with SQL

SQL queries were utilized to explore and analyze the dataset, answering several key questions, including:

- Identifying the unique launch sites in the space mission dataset.
- Retrieving 5 records where the launch site names start with the string "CCA".
- Calculating the total payload mass carried by boosters launched by NASA (CRS).
- Determining the average payload mass for the booster version F9 v1.1.
- Finding the date of the first successful landing outcome on a ground pad.
- Listing boosters with successful drone ship landings and payload mass between 4000 and 6000.

EDA with SQL

- Counting the total number of successful and failed mission outcomes.
- Identifying the booster versions that carried the maximum payload mass.
- Listing failed drone ship landing outcomes, including booster versions and launch site names, for the year 2015.
- Ranking the count of landing outcomes (e.g., "Failure (drone ship)" or "Success (ground pad)") between June 4, 2010, and March 20, 2017, in descending order.

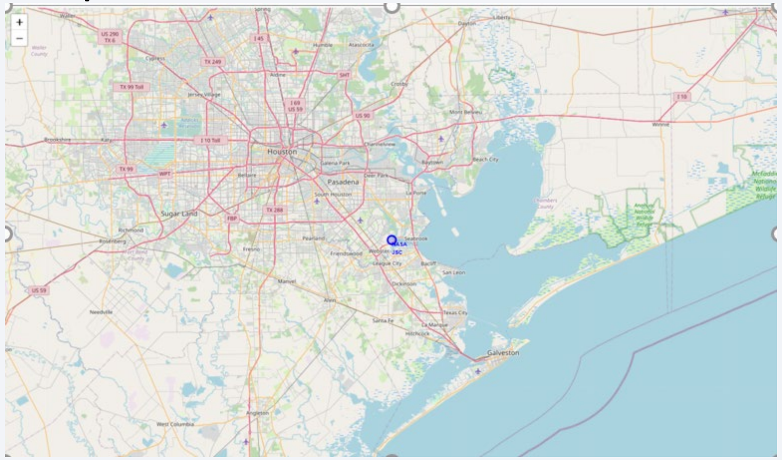
For further details, refer to the github link: https://github.com/greatcourse/Final_Capstone/blob/main/jupyter-labs-eda%20with%20Sql.ipynb

Build an Interactive Map with Folium

Functions from the Folium library were employed to create interactive maps for data visualization. Map objects were created and integrated into the Folium map with the following features:

Markers:

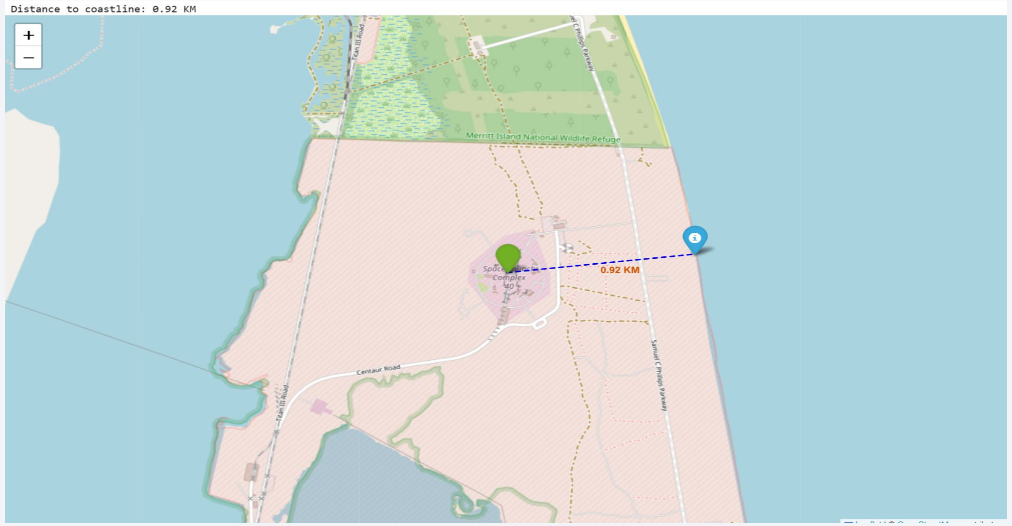
- Added to represent the launch sites and the NASA Johnson Space Center.



- **Circles**:
    - Placed to highlight the launch site locations.
- **Lines**:
    - Used to depict distances to nearby features, including:
        - The distance from CCAFS LC-40 to the coastline.
        - The distance from CCAFS LC-40 to the rail line.
        - The distance from CCAFS LC-40 to the perimeter road.
- Highlighting successful and failed launches for each site on the map.

Build an Interactive Map with Folium

- Plotting the distances from each launch site to nearby landmarks, such as the nearest city, railway, or highway. For example: distances between a launch site to its proximities



For further details, refer to the github link:
Code:https://github.com/greatcourse/Final_Capstone/blob/main/lab_jupyter_launch_site_location_hopefully_final%20(1).ipynb
Figure: https://github.com/greatcourse/Final_Capstone/blob/main/foliumlab_maps.pdf

Build a Dashboard with Plotly Dash

The Plotly Dash dashboard featured interactive elements to enhance data visualization:

Dropdown Menu:

Selecting **'one' launch site** displayed a pie chart showing the distribution of successful and failed Falcon 9 first-stage landings specific to that site.

Selecting **'all' launch sites** displayed a pie chart showing the distribution of successful Falcon 9 first-stage landings across all sites.

Slider Input:

The slider allowed filtering payload masses to refine the scatterplot data.

Scatterplot:

This plot visualized the distribution of Falcon 9 first-stage landings based on payload mass, mission outcomes, and booster version categories.

For further details, refer to the github link:
Code: https://github.com/greatcourse/Final_Capstone/blob/main/Interactive_dashboard_PlotyDash%20(1).py
Figure: https://github.com/greatcourse/Final_Capstone/blob/main/Interactive_dashboard.jpg

Predictive Analysis (Classification)

The classification process involves the following key steps:

Data Preparation

- Load Dataset: The dataset is imported and prepared for analysis.
- Normalize Data: Data normalization ensures uniform scaling for better model performance.
- Split Data: The dataset is divided into training and test sets to enable model evaluation.

Model Preparation

- Algorithm Selection: Suitable machine learning algorithms are chosen.
- Hyperparameter Tuning: GridSearchCV is used to set and optimize parameters for each algorithm.
- Training Models: Models are trained using the training dataset through GridSearchCV.

Predictive Analysis (Classification)

Model Evaluation

- Hyperparameter Optimization: The best hyperparameters for each model are identified.
- Accuracy Measurement: The accuracy of each model is computed using the test dataset.
- Confusion Matrix: A confusion matrix is plotted to evaluate prediction performance.

Model Comparison

- Accuracy Comparison: Models are compared based on their accuracy scores.
- Final Model Selection: The model with the highest accuracy is chosen as the best performer.

For further details, refer to the github link:
https://github.com/greatcourse/Final_Capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_lab_final.ipynb

Results

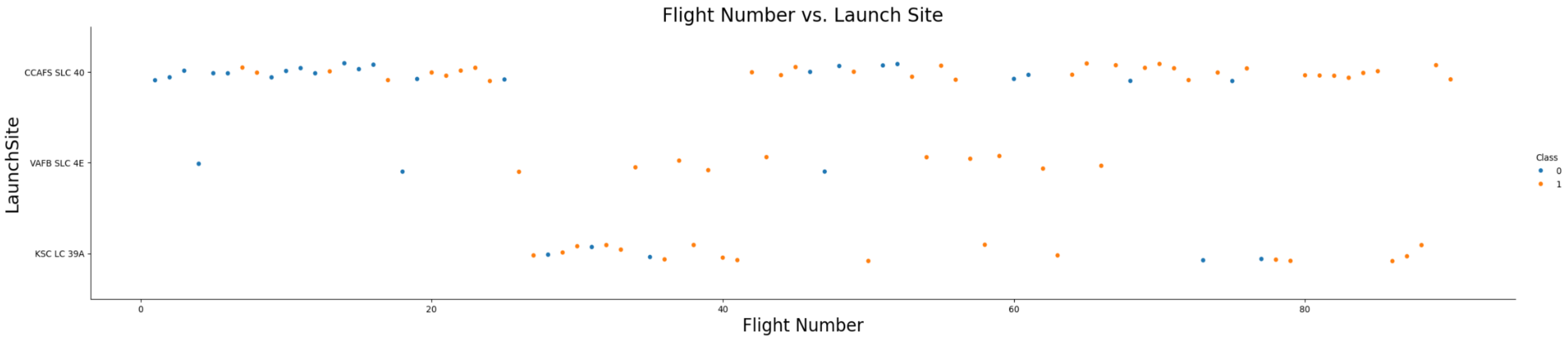
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

[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 class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.title("Flight Number vs. Launch Site", fontsize=22)
plt.show()
```

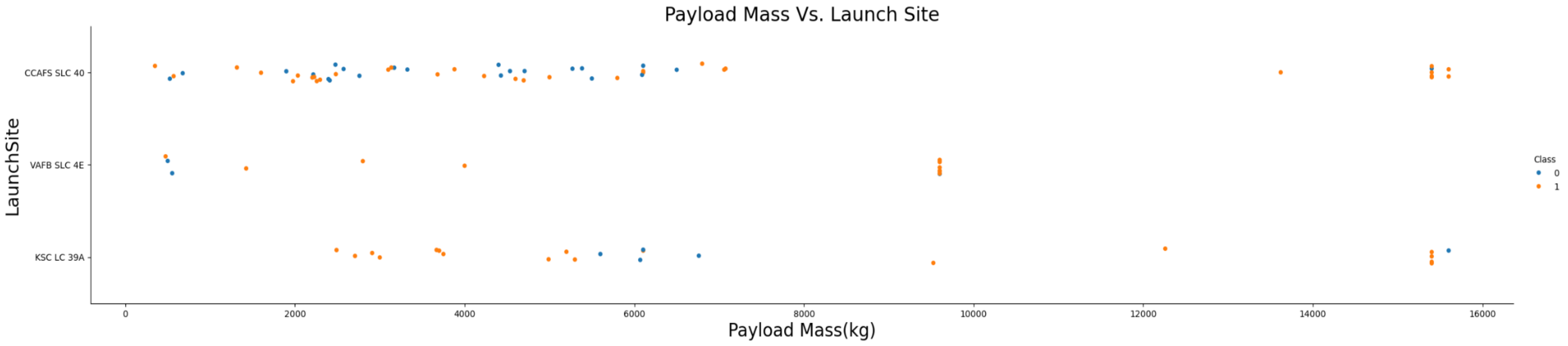


Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

[ ]:
```
The earliest flights all failed while the latest flights all succeeded.
The CCAFS SLC 40 launch site has about a half of all launches.
VAFB SLC 4E and KSC LC 39A have higher success rates.
It can be assumed that each new launch has a higher rate of success.
```

Payload vs. Launch Site

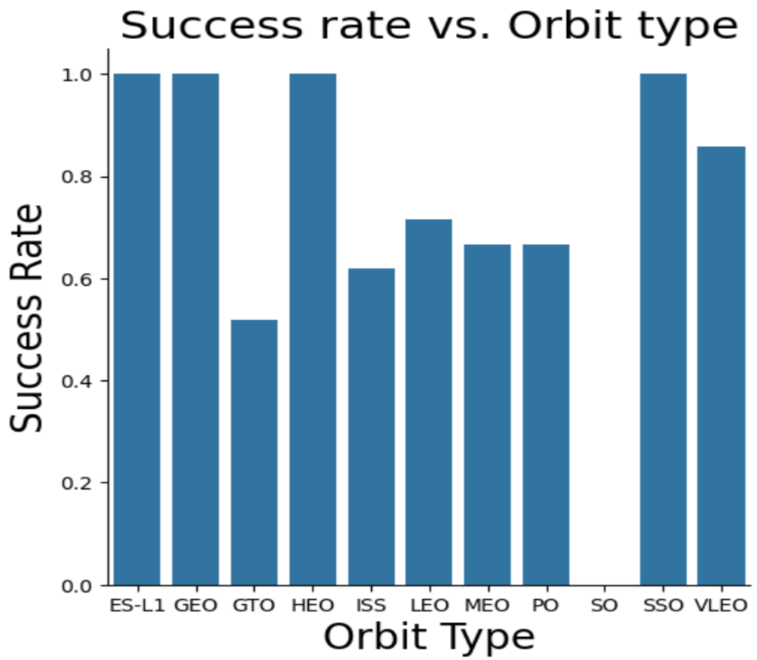
[9]:
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Payload Mass(kg)",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.title("Payload Mass Vs. Launch Site", fontsize=22)
plt.show()
```



[ ]:
```
Explanation:
For every launch site the higher the payload mass, the higher the success
rate.
Most of the launches with payload mass over 7000 kg were successful.
KSC LC 39A has a 100% success rate for payload mass under 5500 kg too.
```

Success Rate vs. Orbit Type

```
sns.catplot(x= 'Orbit', y = 'Class', data = df.groupby('Orbit')['Class'].mean().reset_index(), kind = 'bar')
plt.xlabel('Orbit Type',fontsize=20)
plt.ylabel('Success Rate',fontsize=20)
plt.title("Success rate vs. Orbit type", fontsize=22)
plt.show()
```

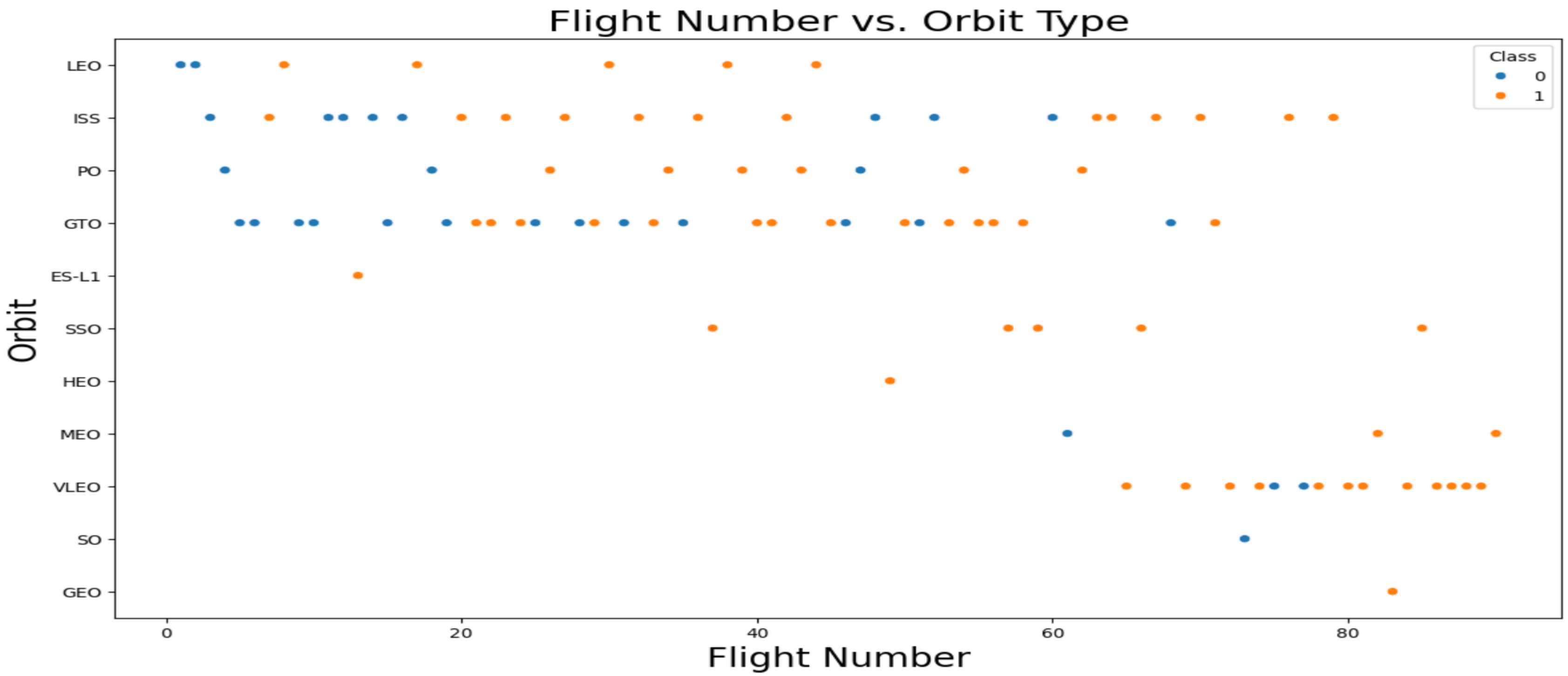


Analyze the plotted bar chart to identify which orbits have the highest success rates.

```
[ ]: Explanation:
Orbits with 100% success rate:
- ES-L1, GEO, HEO, SSO
Orbits with 0% success rate:
- SO
Orbits with success rate
between 50% and 85%:
- GTO, ISS, LEO, MEO, PO
```

Flight Number vs. Orbit Type

```
[14]: plt.figure(figsize=(14,8))
sns.scatterplot(x="FlightNumber",y="Orbit",hue="Class",data = df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.title("Flight Number vs. Orbit Type", fontsize=22)
plt.show()
```

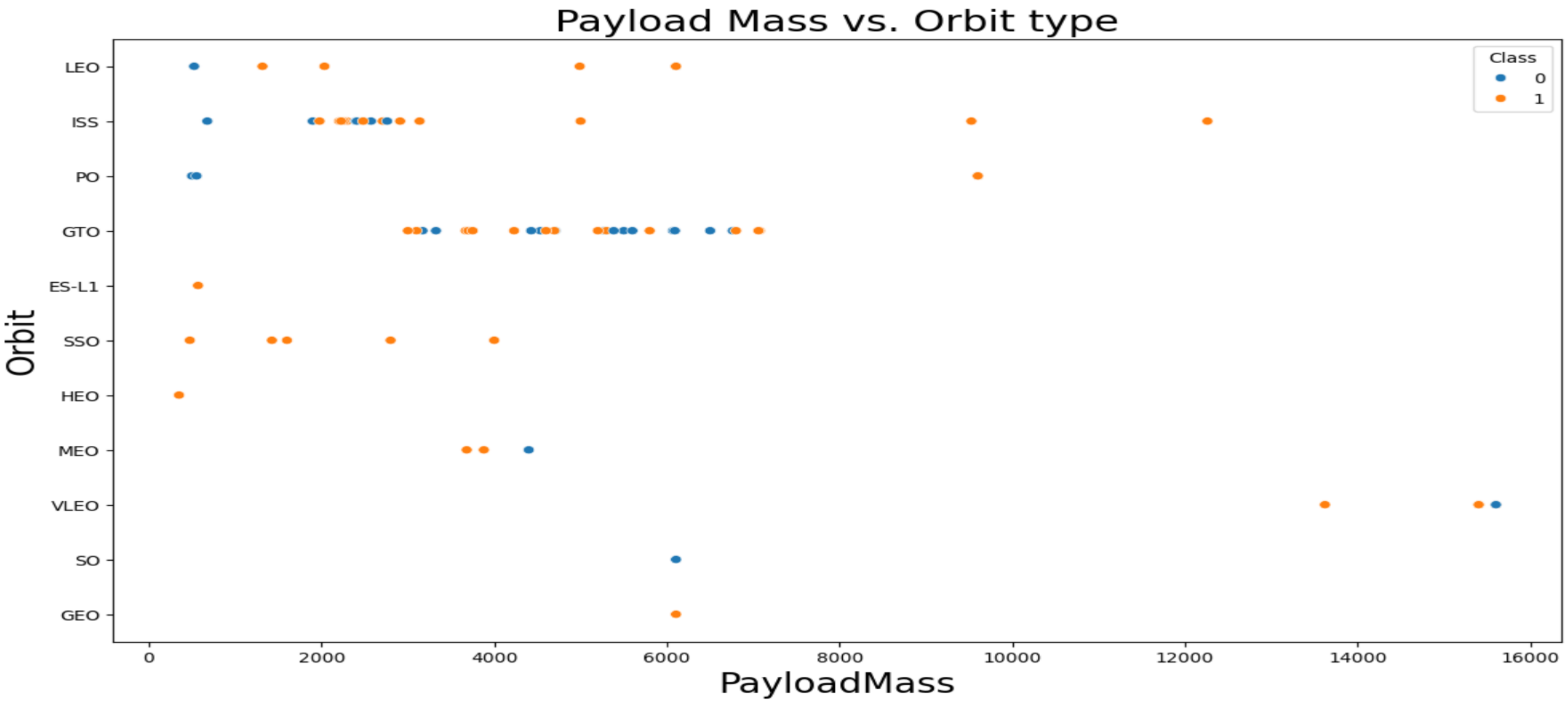



```
[ ]: Explanation:
In the LEO orbit the Success appears related to the number of flights;
on the other hand, there seems to be no relationship between flight number when in GTO orbit.
```

Payload vs. Orbit Type

[15]:
```
# Plot a scatter point chart with x axis to be Payload Mass and y axis to be the Orbit, and hue to be the class value

plt.figure(figsize=(14,8))
sns.scatterplot(x="PayloadMass",y="Orbit",hue="Class",data = df)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.title("Payload Mass vs. Orbit type", fontsize=22)
plt.show()
```



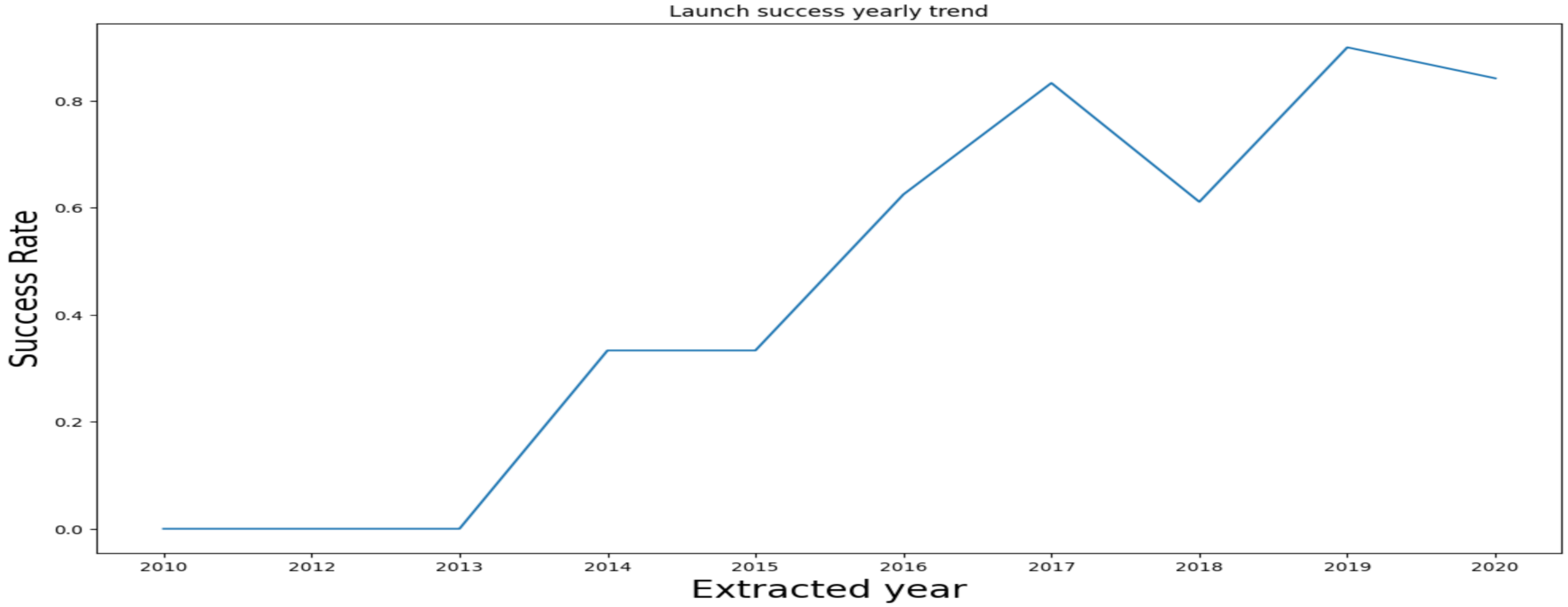
[ ]:
```
Explanation:
Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.
```

Launch Success Yearly Trend

[25]:
```
year=[]
def Extract_year(date):
    for i in df["Date"]:
        year.append(i.split("-")[0])
    return year

df['year']=Extract_year(df["Date"])
df_year_success=df.groupby("year",as_index=False)["Class"].mean()

plt.figure(figsize=(14,8))
sns.lineplot(data=df_year_success, x="year", y="Class" )
plt.xlabel("Extracted year",fontsize=20)
plt.title('Launch success yearly trend')
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```



[ ]:
```
Explanation:
The success rate since 2013 kept increasing till 2020.
```

All Launch Site Names

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

```
[10]: %sql SELECT DISTINCT "LAUNCH_SITE" FROM SPACEXTBL
```

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

[10]:

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

Explanation: There are four unique launch sites in the space mission.

Launch Site Names Begin with 'CCA'

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

[13]:
```
%sql SELECT * FROM SPACEXTBL WHERE "LAUNCH_SITE" LIKE '%CCA%' LIMIT 5
```

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

[13]:

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

Explanation: This is a simple sampling method used to get an overview of the data stored in the database table.

Total Payload Mass

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

[14]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total payload mass by NASA (CRS)" FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';
```

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

[14]:

Total payload mass by NASA (CRS)
45596

Explanation: The total payload carried by boosters from NASA is 45,596 kg

Average Payload Mass by F9 v1.1

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

```
[15]: %sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average payload mass by Booster Version F9 v1.1" FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1';

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

[15]:

Average payload mass by Booster Version F9 v1.1
2928.4

Explanation: The average payload mass carried by booster version F9 v1.1 is 2,928 kg.

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
[23]: %sql SELECT MIN(DATE) AS "Date of first successful landing outcome in ground pad" FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (ground pad)';

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

[23]:

Date of first successful landing outcome in ground pad
2015-12-22

```
[24]: %sql SELECT MIN("DATE") FROM SPACEXTBL WHERE "Landing_Outcome" LIKE 'Success%'

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

[24]:

MIN("DATE")
2015-12-22

Explanation: The first successful landing outcome on ground pad occurred on December 22, 2015.

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

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

```
[26]: %sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

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

[26]:

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

```
[27]: %sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND  PAYLOAD_MASS__KG_ < 6000;
```

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

[27]:

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

Explanation: Listed above are the four booster versions that successfully landed on a drone ship with a payload mass between 4,000 kg and 6,000 kg.

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

List the total number of successful and failure mission outcomes

```
[29]: %sql select mission_outcome, count(*) as total_number from SPACEXTBL group by mission_outcome;
```

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

[29]:

Mission_Outcome	total_number
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

```
[30]: %sql SELECT number_of_success_outcomes, number_of_failure_outcomes FROM (SELECT COUNT(*) AS number_of_success_outcomes FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE 'Success%') success_table, (SELECT COUNT(*) n
```

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

[30]:

number_of_success_outcomes	number_of_failure_outcomes
100	1

Explanation: There were 100 successful and 1 failed mission outcomes.

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

```
[31]: %sql SELECT DISTINCT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);

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

[31]:

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

Explanations: Twelve distinct Falcon 9 boosters have carried the maximum payload mass.

2015 Launch Records

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

```
[35]: %sql SELECT DATE, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE substr(Date,0,5) = '2015' AND LANDING_OUTCOME = 'Failure (drone ship)';

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

[35]:

Date	Booster_Version	Launch_Site
2015-01-10	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	F9 v1.1 B1015	CCAFS LC-40

```
[41]: %sql select substr(Date, 6,2) as month, date, booster_version, launch_site, landing_outcome from SPACEXTBL where landing_outcome = 'Failure (drone ship)' and substr(Date,0,5)='2015'

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

[41]:

month	Date	Booster_Version	Launch_Site	Landing_Outcome
01	2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Explanation: In 2015, there were two failed landing outcomes on a drone ship. Both launches originated from CCAFS LC-40, with one failure occurring in January and the other in April.

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

```
[44]: %%sql select landing_outcome, count(*) as count_outcomes from SPACEXTBL
      where date between '2010-06-04' and '2017-03-20'
      group by landing_outcome
      order by count_outcomes desc;
```

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

[44]:

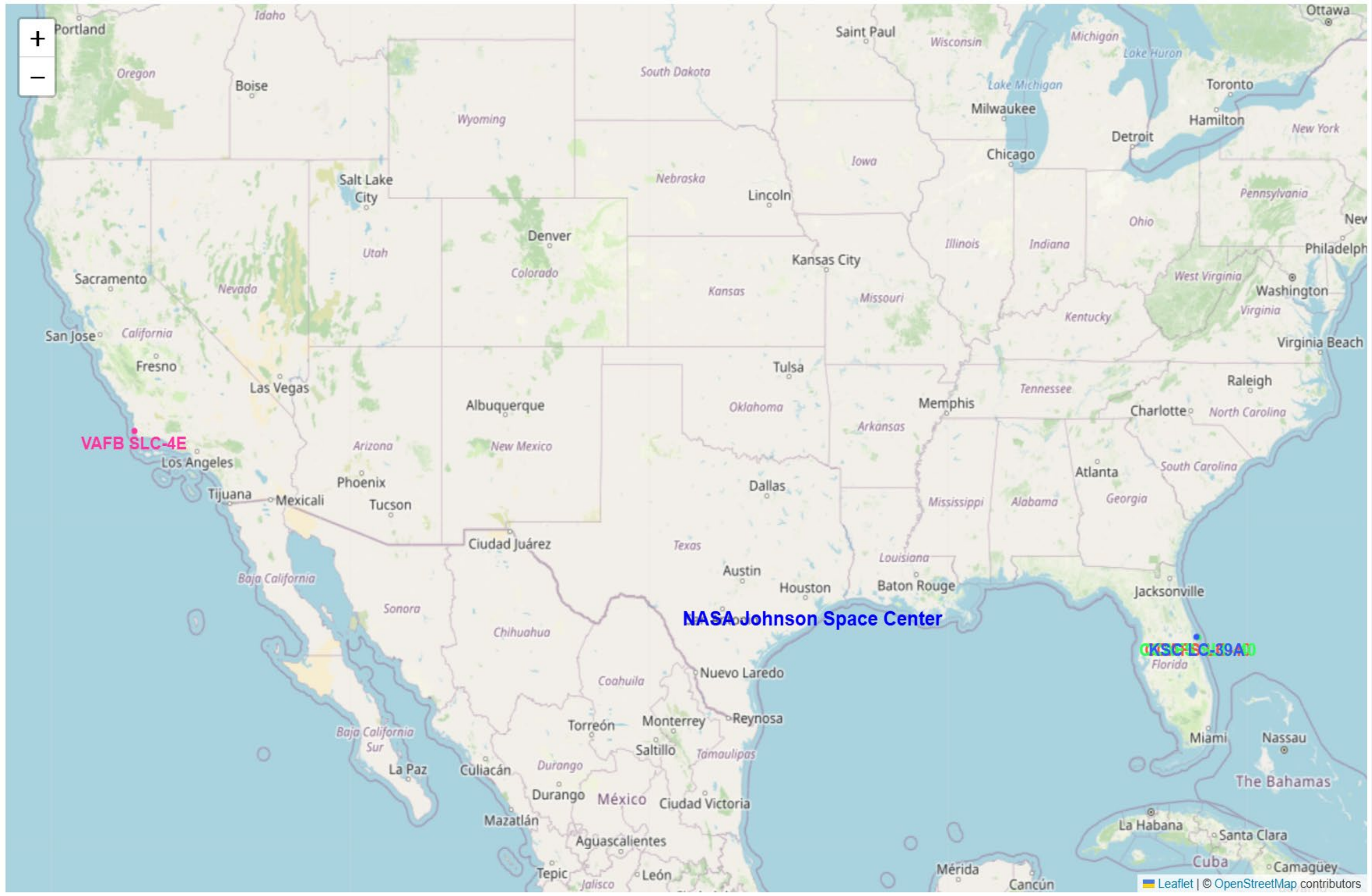
Landing_Outcome	count_outcomes
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Explanation: The most frequent landing outcome observed was ‘No attempt.’

Section 3

Launch Sites Proximities Analysis

All launch sites’ location markers on a global map



All launch sites are located in close proximity to the coast, which helps minimize the risk of debris or explosions affecting nearby populations when rockets are launched towards the ocean.
Both CCAFS LC-40 and CCAFS SLC-40 refer to the same launch site in the data.

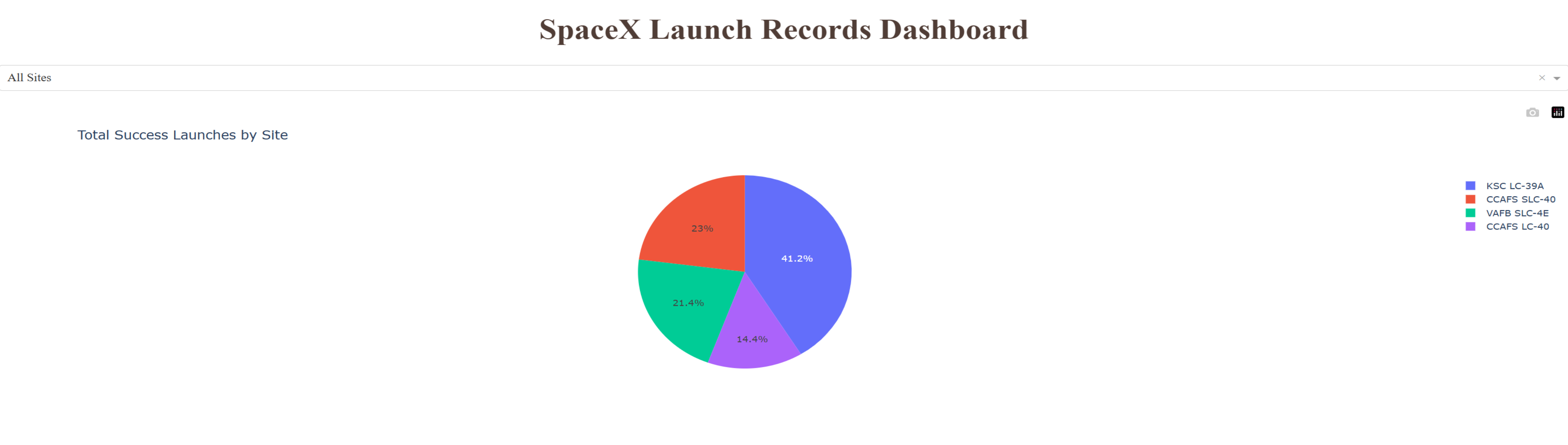
Distances between a launch site to its proximities

- The coastline is 0.92 km away from CCAFS LC-40.
- The rail line is 1.33 km away from CCAFS LC-40.

Section 4

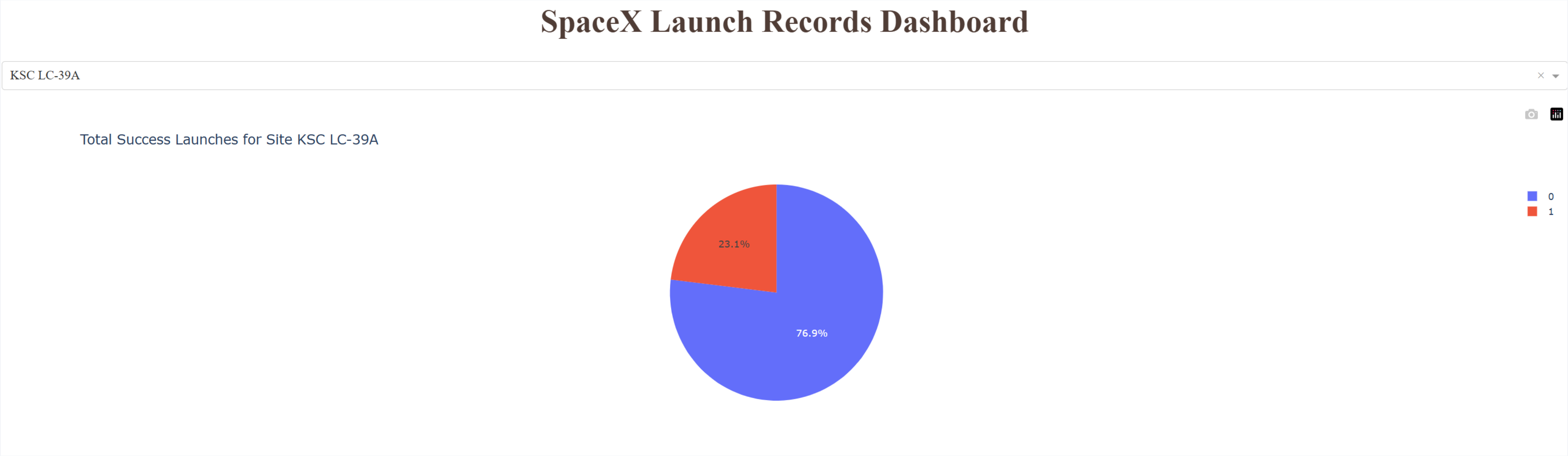
Build a Dashboard with Plotly Dash

Launch success count for all sites



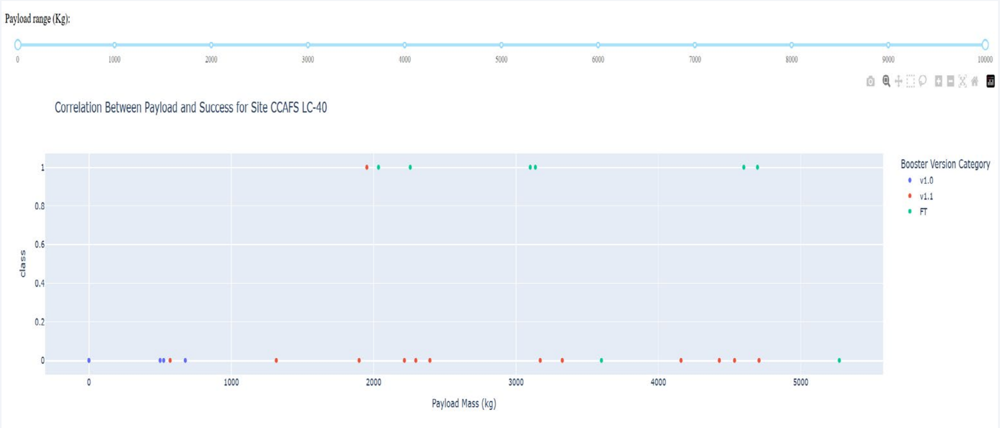
The dropdown menu enabled users to select either a specific launch site or all launch sites.
When all launch sites were selected, the pie chart showcased the distribution of successful Falcon 9 first-stage landing outcomes across the various launch sites.
Notably, the highest proportion of successful Falcon 9 first-stage landings (41.2% of the total) occurred at KSC LC-39A.

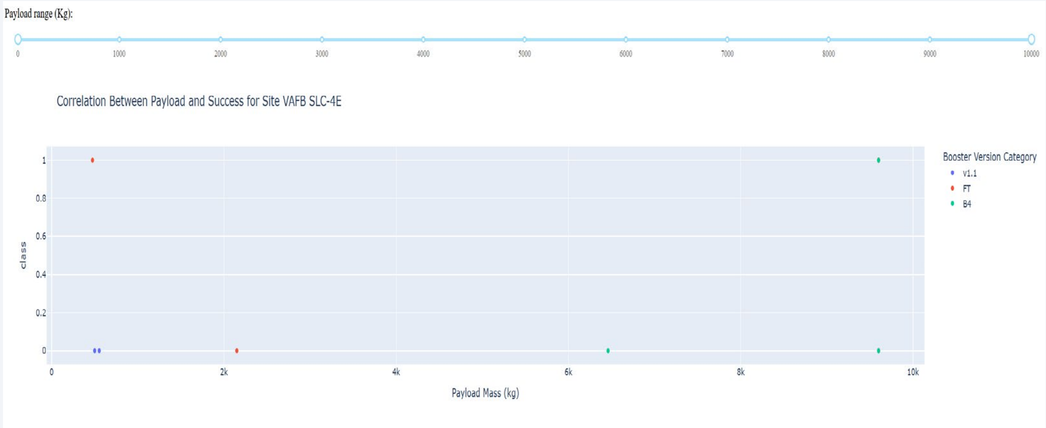
Launch site with highest launch success ratio

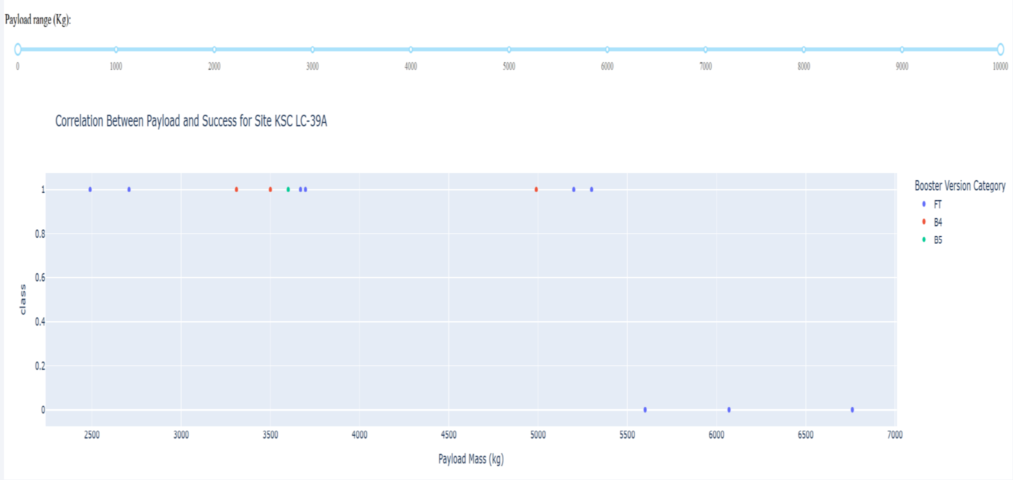


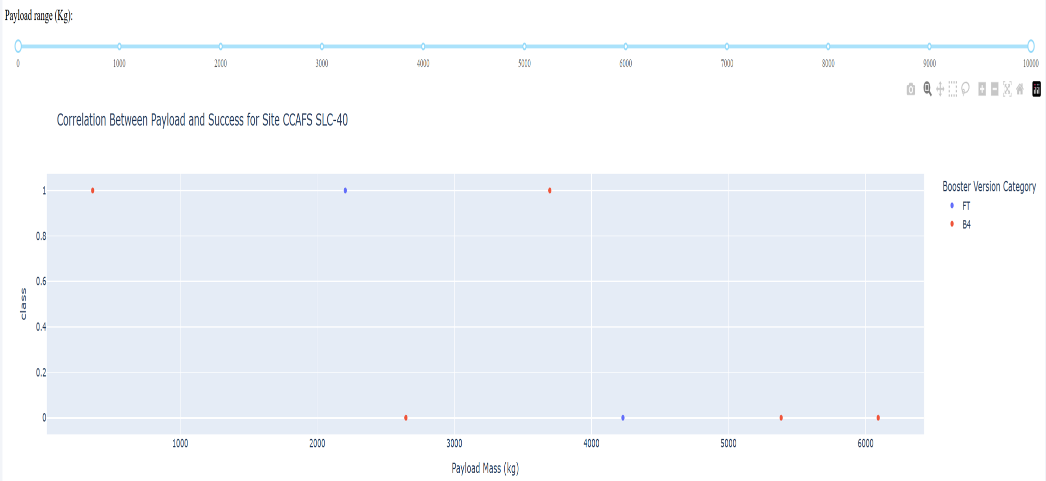
Explanation: KSC LC-39A boasts the highest launch success rate at 76.9%.

Payload Mass vs. Launch Outcome for all sites







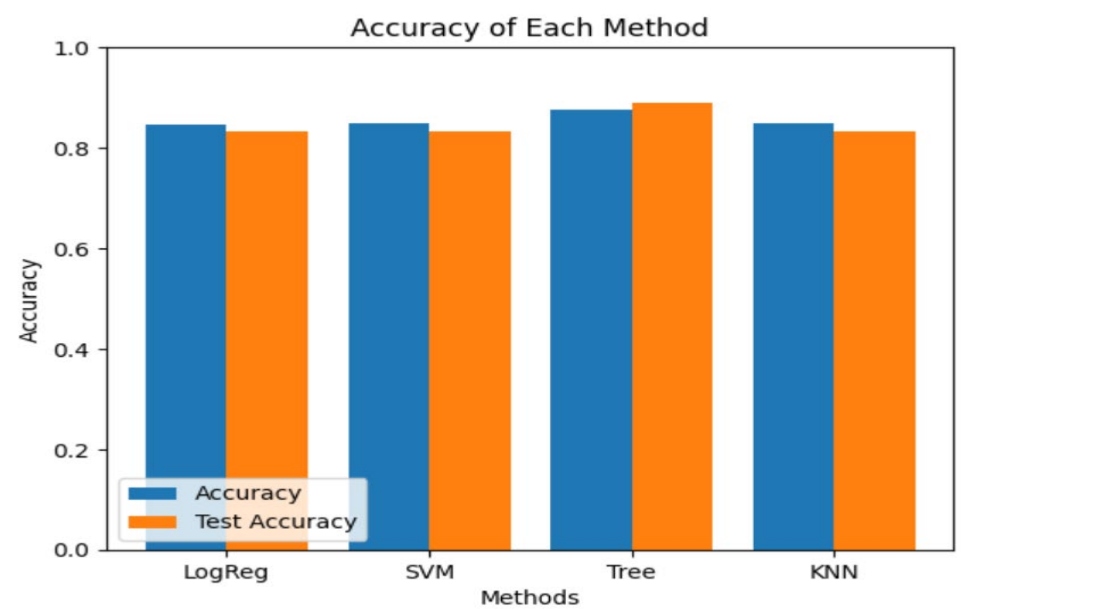


Explanation: The charts indicate that payloads weighing between 2000 and 5500 kg achieve the highest success rate.

Section 5

Predictive Analysis (Classification)

Classification Accuracy

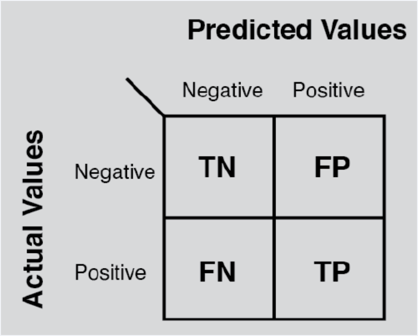


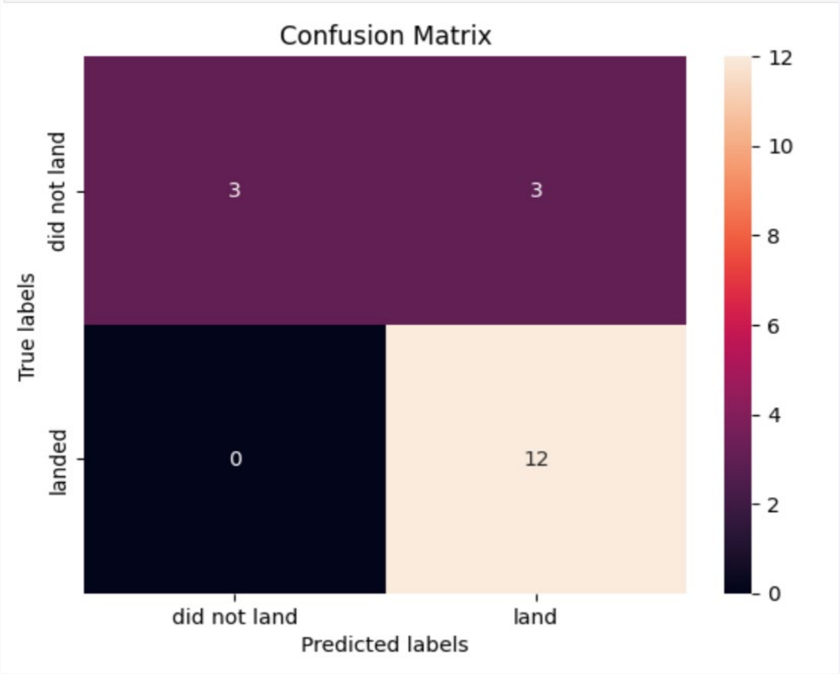
	LogReg	SVM	Tree	KNN
Jaccard_Score	0.800000	0.733333	0.846154	0.750000
F1_Score	0.888889	0.846154	0.916667	0.857143
Accuracy	0.833333	0.777778	0.888889	0.777778

Explanation:
The evaluation scores for the entire dataset confirm that the Decision Tree Model is the best-performing model. It not only achieves higher overall scores but also demonstrates the highest accuracy.

Confusion Matrix

The confusion matrix for the Decision Tree model provides a detailed summary of its predictions:
True Positives (TP): 12 instances where the model correctly predicted the positive class.
True Negatives (TN): 3 instances where the model correctly predicted the negative class.
False Positives (FP): 3 instances where the model incorrectly predicted the positive class for a negative case.
False Negatives (FN): 0 instances where the model incorrectly predicted the negative class for a positive case.





Conclusions

The **Decision Tree Model** is the most effective algorithm for this dataset. Key findings include:

- **Low payload mass** launches tend to have better success rates than those with higher payload masses.
- Most launch sites are near the Equator and close to the coast.
- **Launch success rates** have increased over the years.
- **KSC LC-39A** has the highest success rate of all sites.
- SpaceX's record for **Falcon 9 first-stage landing outcomes** has improved over time.
- There is a general trend toward better performance and success as more launches occur.
- The project aims to predict if the Falcon 9 first stage will land, which helps determine launch costs.
- Features like **payload mass** and **orbit type** affect mission outcomes.
- Several machine learning algorithms were tested, with the **Decision Tree model** outperforming others.
- These models can predict future Falcon 9 landing outcomes.

Appendix

- GitHub URL (Data Collection): https://github.com/greatcourse/Final_Capstone/blob/main/LAB1_API_Collect%20the%20data_part1.ipynb
- GitHub URL (Web Scraping): https://github.com/greatcourse/Final_Capstone/blob/main/jupyter-labs-webscraping_final.ipynb
- GitHub URL (Data Wrangling): https://github.com/greatcourse/Final_Capstone/blob/main/mod1_lab3-spacex-Data%20wrangling.ipynb
- GitHub URL (EDA with SQL): https://github.com/greatcourse/Final_Capstone/blob/main/jupyter-labs-eda%20with%20Sql.ipynb
- GitHub URL (EDA with Data Visualization): https://github.com/greatcourse/Final_Capstone/blob/main/Data%20Visualization.ipynb
- GitHub URL (Folium Maps): https://github.com/greatcourse/Final_Capstone/blob/main/lab_jupyter_launch_site_location_hopefully_final%20(1).ipynb

  maps: https://github.com/greatcourse/Final_Capstone/blob/main/foliumlab_maps.pdf
- GitHub URL (Dashboard File): https://github.com/greatcourse/Final_Capstone/blob/main/Interactive_dashboard_PlotyDash%20(1).py

  dashboard: https://github.com/greatcourse/Final_Capstone/blob/main/Interactive_dashboard.jpg
- GitHub URL (Machine Learning): https://github.com/greatcourse/Final_Capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_lab_final.ipynb

Reference

- Coursera Discussion Forum

Thank you!