IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Craig Murray
January 20, 2024

Outline

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

Executive Summary

- Summary of methodologies
  - Data collection via the Space X API
  - Data collection via web scraping the Space X Wikipedia page
  - Exploratory Data Analysis with SQL and Data Visualization
  - Interactive Visualization with Folium
  - A Plotly based dashboard
- Summary of all results
  - A primary factor in the successful return landing of the reusable module is the type of orbit the mission executed.
  - Other variables analyzed such as flight number and launch do not have a significant effect on mission success.

Introduction

- Project background and context
  - Space X has become more successful over the years as a private company providing a service to put payloads into orbit. While much of the cost of mission is consumed as part of the mission, the first stage boosters are reusable if they are successfully landed after the mission. Any improvement in the reuse of first stage booster would reducxe overall mission costs and profitability.
- Problems you want to find answers
  - Which features have the largest effect on a successful mission.
  - What are the interrelationship of features that could affect mission success.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected from the Space X API and web scraping the Space X Wikipedia page
- Perform data wrangling
  - Categorical variables were one-hot encoded to numerical for models use.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - For this classification task I built 4 different model; logistics regression, a tree based classifier, a support vector machine classifier and k-nearest nieghbor.
  - A grid search was used for each model.
  - K-nearest Neighbor provided the best results

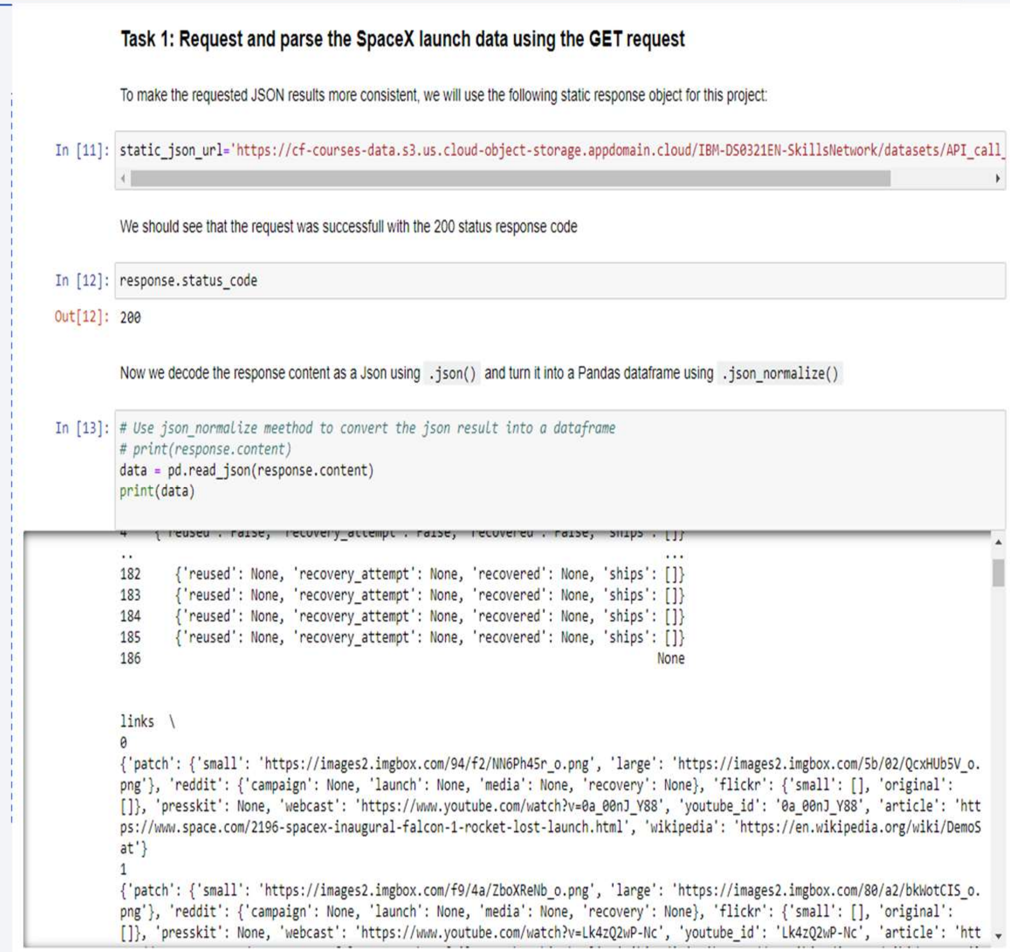
Data Collection

- Describe how data sets were collected.
  - Data for the project was obtained through the Space X API using a get request.
  - Obtained data was in JSON and had to be decoded and imported into a Pandas dataframe.
  - The data was then cleaned looking for and dealing with missing values.
  - Additional data was scraped from the web with Beautiful and combines with the API data.

Data Collection – SpaceX API

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

- Add the GitHub URL of the completed SpaceX API calls notebook https://github.com/cmurray4492/IBM_Capstone_Project/blob/main/Week_1_jupyter-labs-spacex-data-collection-api.ipynb



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 [11]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_
```

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

In [12]:
```
response.status_code
```

Out[12]:
```
200
```

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

In [13]:
```
# Use json_normalize meethod to convert the json result into a dataframe
# print(response.content)
data = pd.read_json(response.content)
print(data)
```


```
..                                                              ...
182    {'reused': None, 'recovery_attempt': None, 'recovered': None, 'ships': []}
183    {'reused': None, 'recovery_attempt': None, 'recovered': None, 'ships': []}
184    {'reused': None, 'recovery_attempt': None, 'recovered': None, 'ships': []}
185    {'reused': None, 'recovery_attempt': None, 'recovered': None, 'ships': []}
186                                                                None


links  \
0
{'patch': {'small': 'https://images2.imgbox.com/94/f2/NN6Ph45r_o.png', 'large': 'https://images2.imgbox.com/5b/02/QcxHUb5V_o.png'}, 'reddit': {'campaign': None, 'launch': None, 'media': None, 'recovery': None}, 'flickr': {'small': [], 'original': []}, 'presskit': None, 'webcast': 'https://www.youtube.com/watch?v=0a_00nJ_Y88', 'youtube_id': '0a_00nJ_Y88', 'article': 'https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-lost-launch.html', 'wikipedia': 'https://en.wikipedia.org/wiki/DemoSat'}
1
{'patch': {'small': 'https://images2.imgbox.com/f9/4a/ZboXReNb_o.png', 'large': 'https://images2.imgbox.com/80/a2/bkWotCIS_o.png'}, 'reddit': {'campaign': None, 'launch': None, 'media': None, 'recovery': None}, 'flickr': {'small': [], 'original': []}, 'presskit': None, 'webcast': 'https://www.youtube.com/watch?v=Lk4zQ2wP-Nc', 'youtube_id': 'Lk4zQ2wP-Nc', 'article': 'htt
```

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- Add the GitHub URL https://github.com/cmurray4492/IBM_Capstone_Project/blob/main/Week_1_jupyter-labs-webscraping.ipynb

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.

In [8]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url).text
```

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

In [9]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response, 'html.parser')
```

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

In [10]:
```
# Use soup.title attribute
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

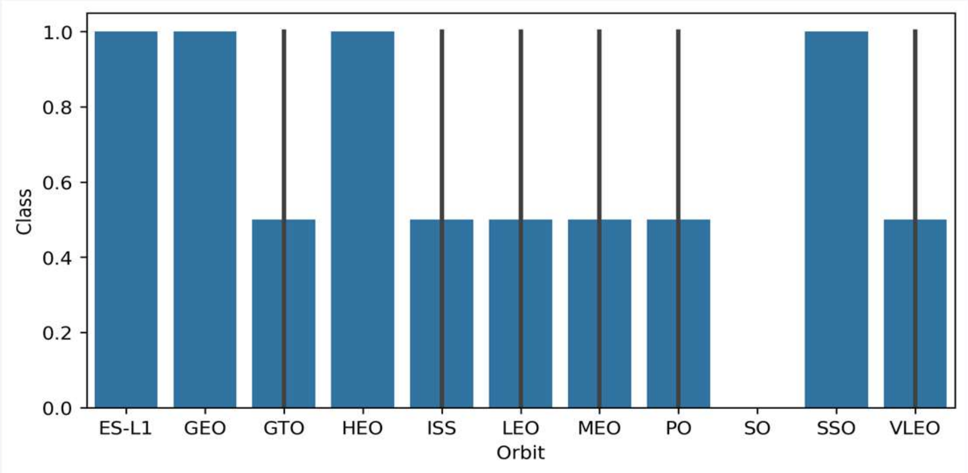
In [11]:
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all("table")
print(html_tables)
```

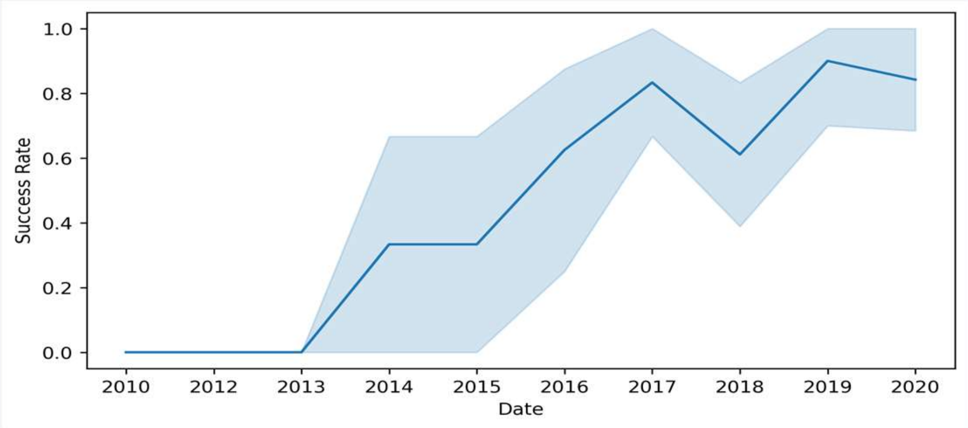
Data Wrangling

- Exploratory data analysis and created training data labels.
- The number of launches for each site were calculated.
- The number of orbit types were calculated.
- Landing outcomes were labeled.
- Notebook link:
  - https://github.com/cmurray4492/IBM_Capstone_Project/blob/main/Week_1_labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Visualization Charts
  - The top chart represents the type of orbit by the class of module.
  - The bottom chart is line chart of the landing success rate of the mission.
- Data visualization notebook link:
  - https://github.com/cmurray4492/IBM_Capstone_Project/blob/main/Week_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb





EDA with SQL

- Study data was loaded into an SQLite3 database.
- Sample of the queries executed for EDA include:
  - Total mass for all missions
  - Average mass per mission
  - Average mass for booster F9 v1.1
  - Date of first successful ground landing
  - 2015 missions and outcomes
- EDA with SQL notebook link:
  - https://github.com/cmurray4492/IBM_Capstone_Project/blob/main/Week_2_jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- The Folium map first listed the launch sites with launch sites.
- Various important infrastructure items so their distances can be measured. The distance from each may have an affect on condition of modules by how long they are exposed during transit.
- Folium notebook link:
  - https://github.com/cmurray4492/IBM_Capstone_Project/blob/main/Week_3_Folium_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

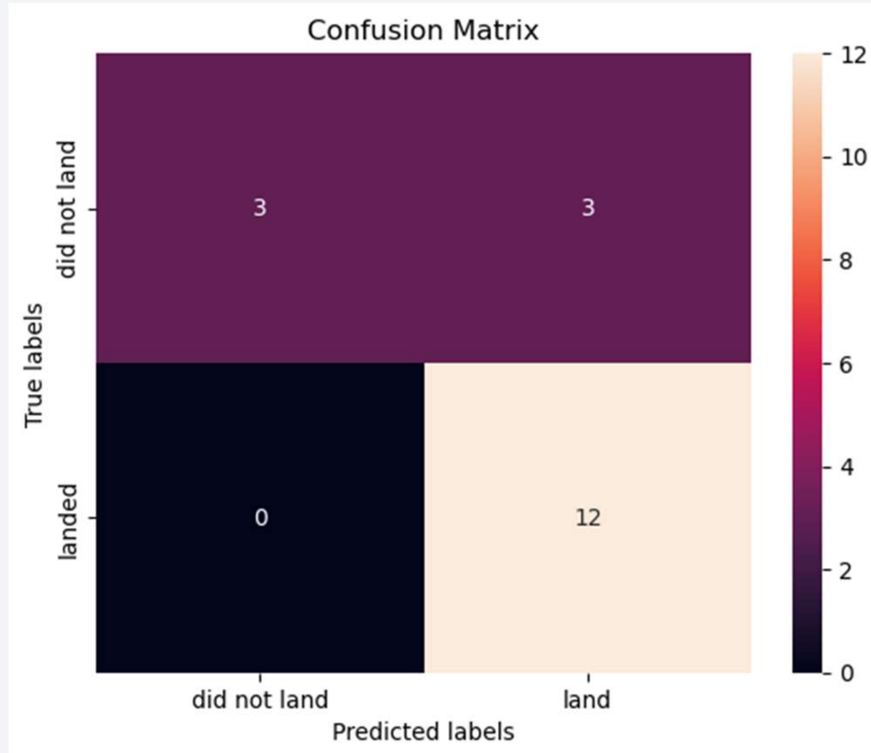
- The Ploty Dash project module included selectors to interactively select elements for review.
- Plotly Dash lab source code link:
  - https://github.com/cmurray4492/IBM_Capstone_Project/blob/main/plotly_dash_board.py

Predictive Analysis (Classification)

- A machine learning classification model was developed using the following steps:
  - The data was scaled using the Standard Scaler
  - A train test split was done with a 20% test size
  - A grid search was setup with selected parameters
  - 4 models were using the following algorithms
    - Logistic Regression
    - Support Vector Machine
    - Decision Tree
    - K Nearest Neighbor
  - The results for each model was evaluated by a confusion matrix
- Predictive analysis lab link:
  - https://github.com/cmurray4492/IBM_Capstone_Project/blob/main/Week_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

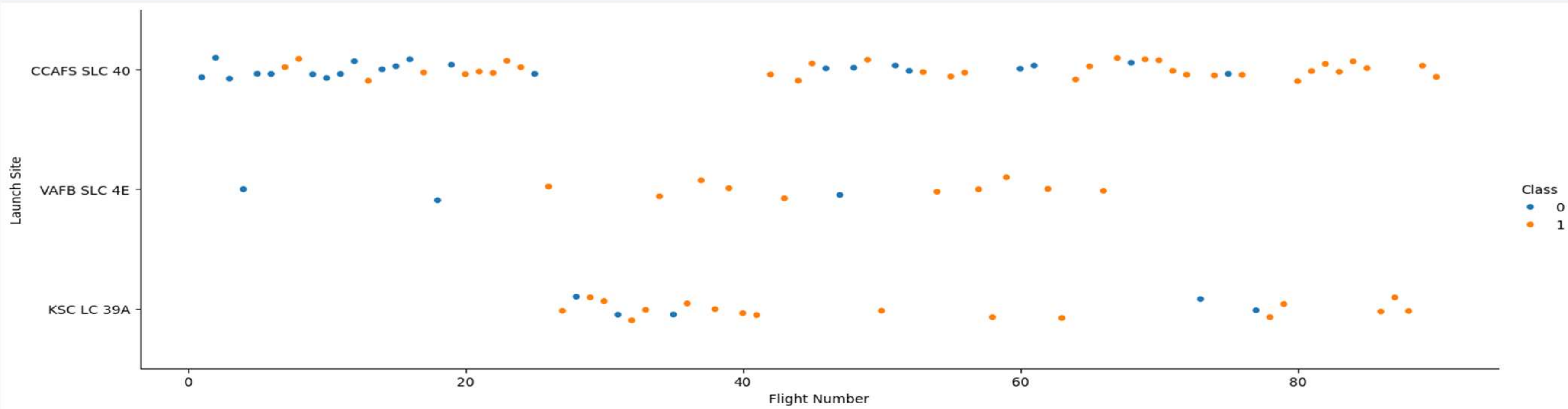
- The best model for predective results was K Nearest Neighbor.
    - Score = 0.8333
- Additional work on hyperparameters may produce a higher score.



Section 2

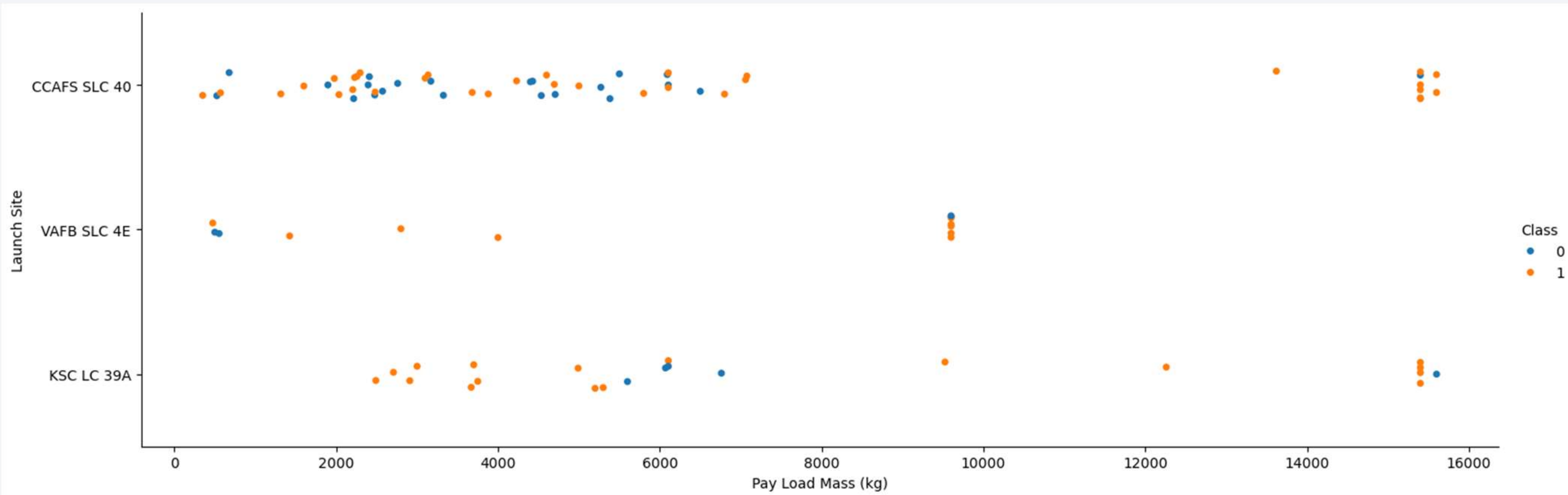
Insights drawn from EDA

Flight Number vs. Launch Site



- This plot shows the flight number by lunch site. Blue dots show class zero and the red show class 1.

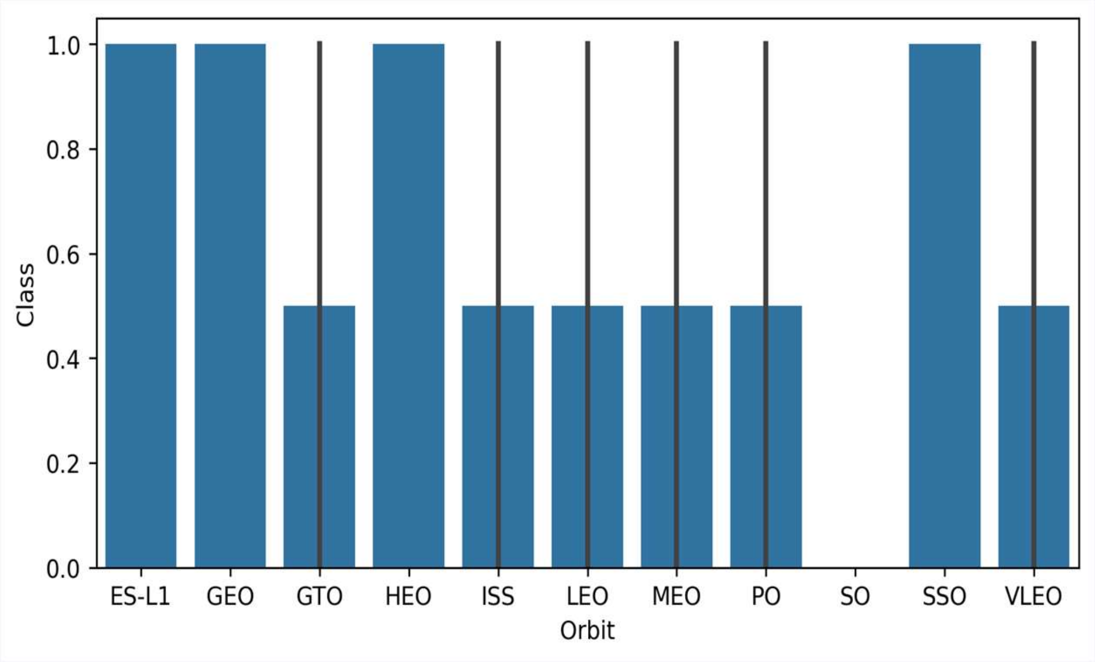
Payload vs. Launch Site



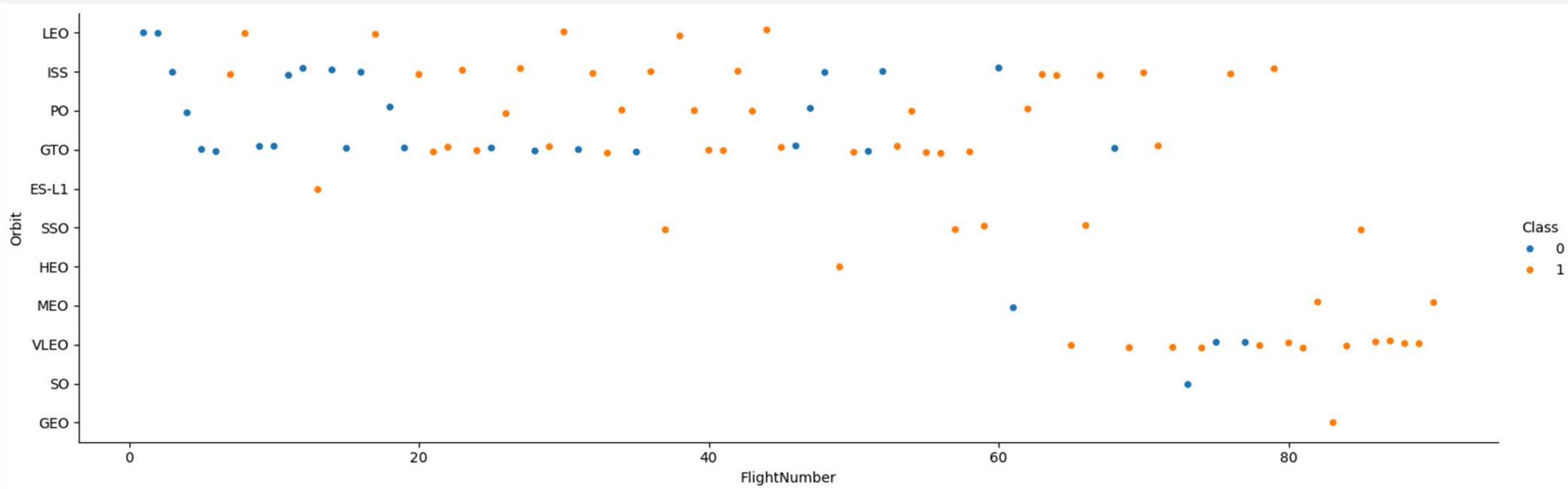
- The scatter plot shows Payload vs. Launch Site. Blue dots show class zero and the red show class 1.

Success Rate vs. Orbit Type

- Orbit success rate. From this bar chart we can see that ES-L1 and and CEO had the best success rate.

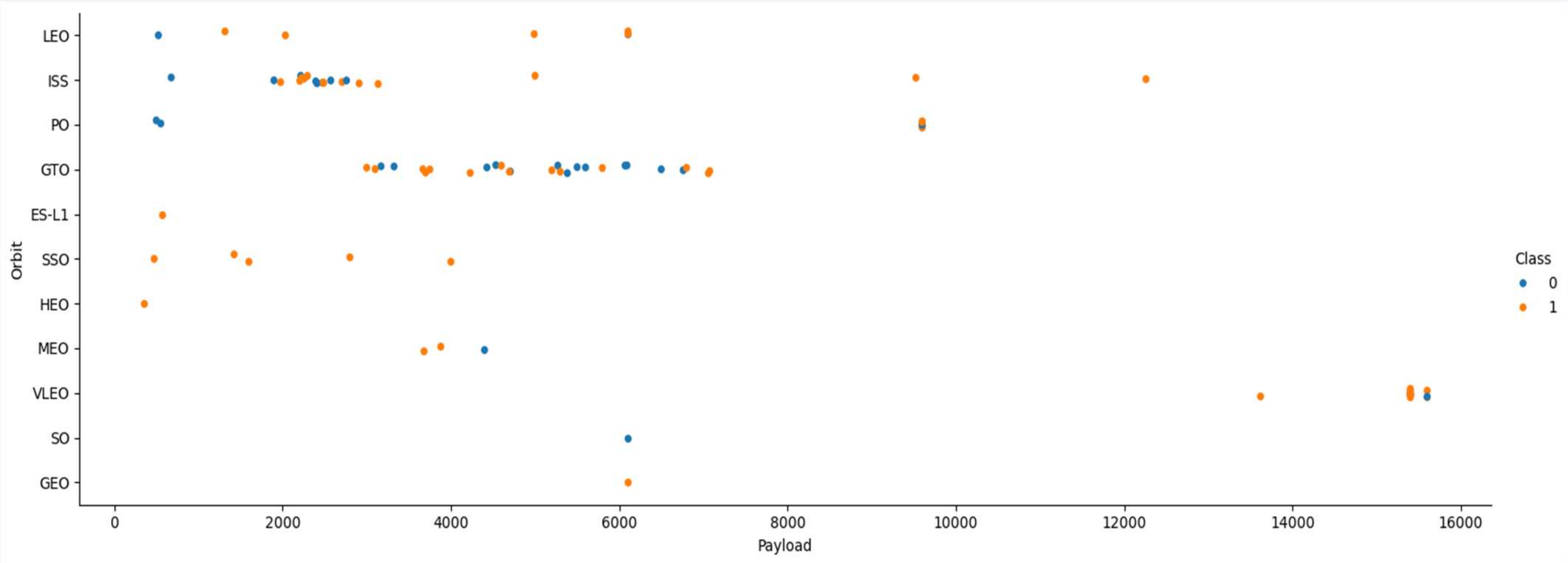


Flight Number vs. Orbit Type



- Scatter Plot of Flight number vs. Orbit type. Blue dots show class zero and the red show class 1.

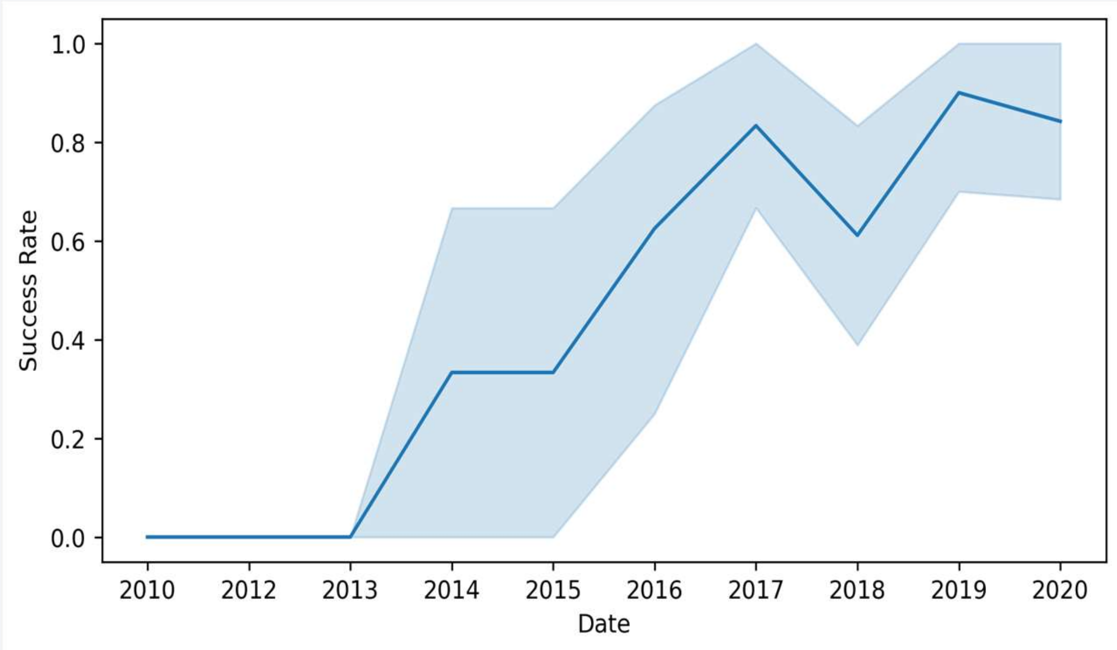
Payload vs. Orbit Type



- Scatter plot of payload vs. orbit type. Blue dots show class zero and the red show class 1.

Launch Success Yearly Trend

- Show a line chart of yearly average success rate.
- Generally, success rates have improved over time.



All Launch Site Names

- To determine all of the names of the launch sites in the dataset, after a connection to the database was established.
- The uses the “distinct” keyword so that only one entry of each type is included.

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

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'

- A query was run to identify the first 5 records that begin with 'CCA'
- The select query where the field 'Launch_Site' like CCA and limit the return to 5 entries.

Task 2

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

In [24]:
```
%sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

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

Out[24]:

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

Total Payload Mass

- Total payload mass is 45596 for NASA
- This select query uses the sum keyword where the customer equals NASA

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

```
n [10]:  %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'

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

ut[10]:

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- The average payload mass for the F9 v1.1 booster is 2928.4.
- The select query for this task uses the AVG keyword where the booster equals F9 v1.1.

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

```
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 ground landing was December 12, 2015.
- To obtain this date the select date looks for the minimum date where the mission outcome and landing outcome both equaled success.

In [17]:

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

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

Out[17]:

min(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

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

In [19]:
```
%sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and
```

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

Out[19]:

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

Total Number of Successful and Failure Mission Outcomes

- Total number of successful and failure mission outcomes

Task 7

List the total number of successful and failure mission outcomes

In [25]:
```
%sql select count(MISSION_OUTCOME) from SPACEXTBL where MISSION_OUTCOME = 'Success' or MISSION_OUTCOME = 'Failure (in fligh
```

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

Out[25]:

count(MISSION_OUTCOME)
99

Boosters Carried Maximum Payload

- Names of the booster which have carried the maximum payload mass
- This SQL query selects booster version which have carry the maximum payload.

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

In [26]:
```
%sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

Out[26]:

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

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

In [33]:

```
%sql SELECT * from SPACEXTBL WHERE substr(Date,0,5)='2015'
```

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

Out[33]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcom
2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone shi
2015-02-11	23:03:00	F9 v1.1 B1013	CCAFS LC-40	DSCOVR	570	HEO	U.S. Air Force NASA NOAA	Success	Controlled (ocea
2015-03-02	3:50:00	F9 v1.1 B1014	CCAFS LC-40	ABS-3A Eutelsat 115 West B	4159	GTO	ABS Eutelsat	Success	No attem
2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failure (drone shi
2015-04-27	23:03:00	F9 v1.1 B1016	CCAFS LC-40	Turkmen 52 / MonacoSAT	4707	GTO	Turkmenistan National Space Agency	Success	No attem
2015-06-28	14:21:00	F9 v1.1 B1018	CCAFS LC-40	SpaceX CRS-7	1952	LEO (ISS)	NASA (CRS)	Failure (in flight)	Precluded (dron shi
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (grour pa

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

In [33]:
```
%sql SELECT * from SPACEXTBL WHERE substr(Date,0,5)='2015'
```

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

Out[33]:

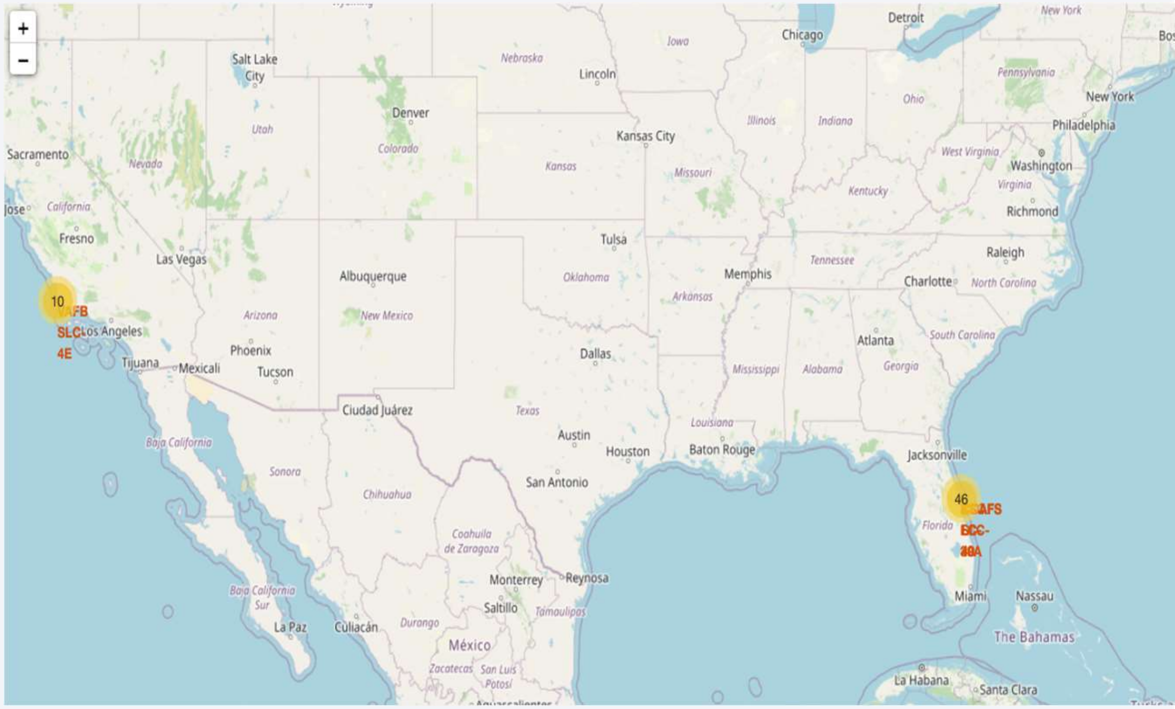
Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcom
2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone shi
2015-02-11	23:03:00	F9 v1.1 B1013	CCAFS LC-40	DSCOVR	570	HEO	U.S. Air Force NASA NOAA	Success	Controlled (ocea
2015-03-02	3:50:00	F9 v1.1 B1014	CCAFS LC-40	ABS-3A Eutelsat 115 West B	4159	GTO	ABS Eutelsat	Success	No attem
2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failure (drone shi
2015-04-27	23:03:00	F9 v1.1 B1016	CCAFS LC-40	Turkmen 52 / MonacoSAT	4707	GTO	Turkmenistan National Space Agency	Success	No attem
2015-06-28	14:21:00	F9 v1.1 B1018	CCAFS LC-40	SpaceX CRS-7	1952	LEO (ISS)	NASA (CRS)	Failure (in flight)	Precluded (dror shi
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (grour pa

Section 3

Launch Sites Proximities Analysis

Launch Site Analysis

- Current launch sites tend to be on or near the coast.



Color Coded Launch Site Map

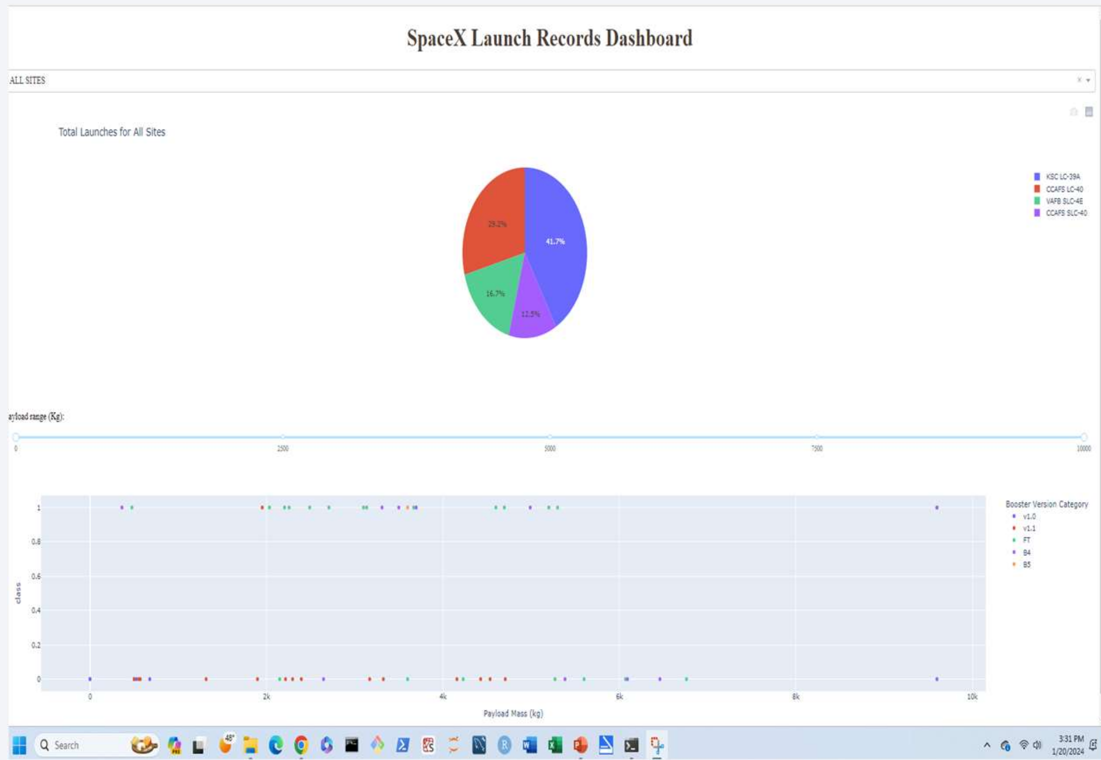
Lanuch Site Infrastructure

Section 4

Build a Dashboard with Plotly Dash

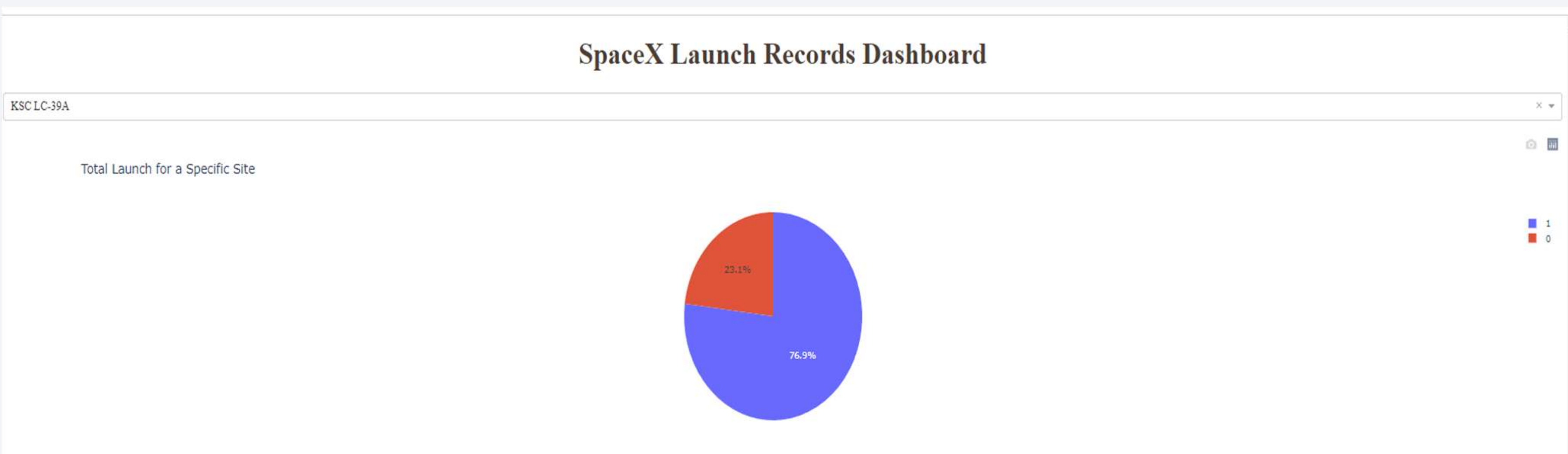
Space X Lanuch Dashboard

- This dashboard has an overview of all launch site and a selection menu to focus in on any active site.



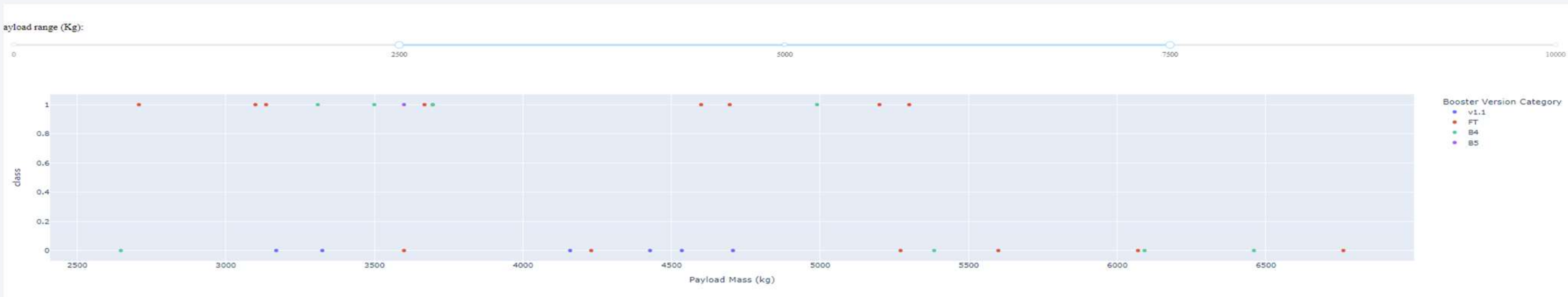
Launch Site With Highest Success Rate

- Screenshot of the piechart for the launch site with highest launch success ratio we KSC LC-39A at 76.9%



Dashboard Slide Data

Slide at 2500



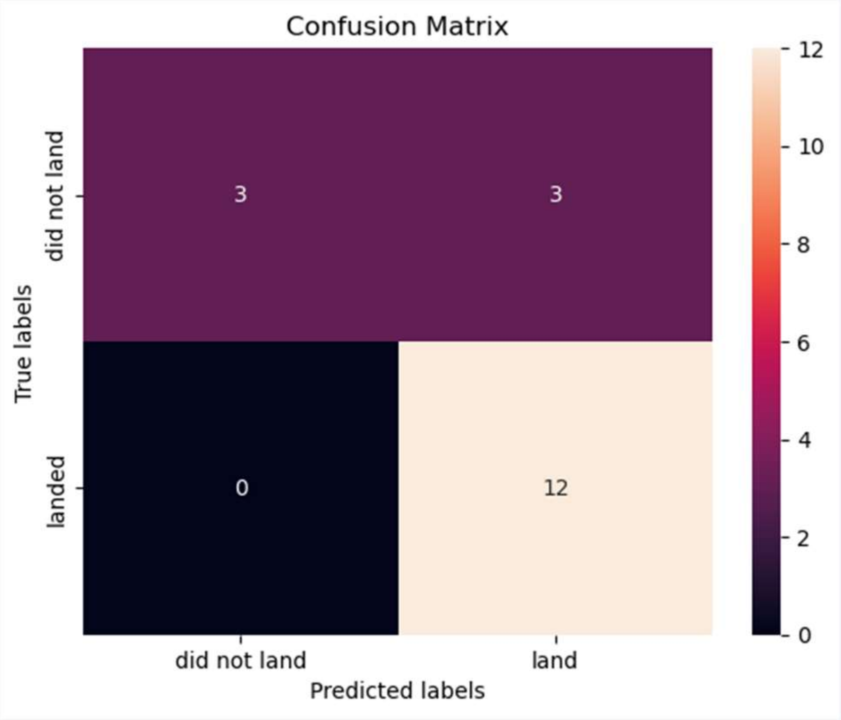
Slide at 7500

Section 5

Predictive Analysis (Classification)

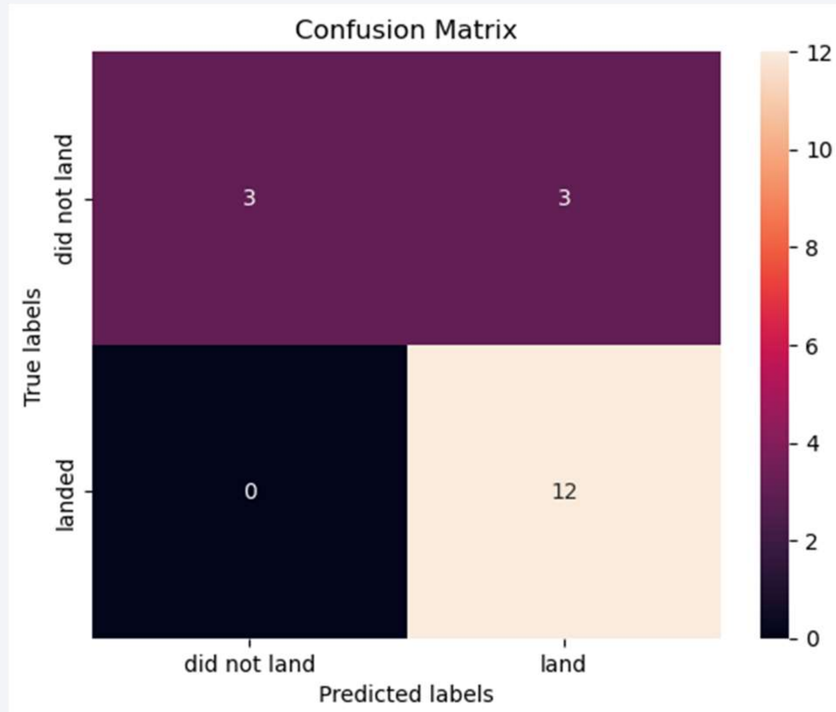
Classification Accuracy

- The Model with the highest accuracy was K-Nearest Neighbors with an accuracy of 0.8333



Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



Conclusions

- Space X is improving mission success over time.
- More launch experience leads to improved results.
- The orbit selected also has an impact on mission success and more research into the casual factors of orbit selection. ES-L1 and GEO have the best results.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project


Thank you!