IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sahil Anand
16/6/2023

Outline

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

Executive Summary

- Summary of methodologies

1. Data collection with API lab
2. Data collection with web scraping lab
3. Data Wrangling
4. Exploratory data analysis with SQL
5. Exploratory data analysis with visualization
6. Visual analytic with folium lab
7. Dash board with ploty dash
8. Machine learning prediction

- Summary of all results

1. Various model provided optimal result
2. Helpful visualization of the data

Introduction

- Project background and context

1. Space X trying develops the model to predict successful landing of falcon rockets

- Problems you want to find answers

1. What is factor affecting failure?
2. What will be the next step to improve landing of the falcon rockets?
3. How to improve the accuracy of the landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

➢ Data is collected with API and Web scraping

- Perform data wrangling

➢ Data transformed and cleaned for analysis and later used for predicted model

- Perform exploratory data analysis (EDA) using visualization and SQL

➢ Find out pattern in data frame and evaluate the result

- Perform interactive visual analytics using Folium and Plotly Dash

➢ Create graphs and maps to help visualize the data

- Perform predictive analysis using classification models

➢ Predictive model created to help predict if future landing will be success

Data Collection

Describe how data sets were collected.

- Data is collect from the URL "https://api.spacexdata.com/v4/rockets/”
- Acquire the data in URL, normalize that data with Json normalization and convert it into proper data frame.
- Update column and row in data frame
- Filter that data to proper analysis.
- Convert that data file to .csv file.

Data Collection – SpaceX API

- Acquiring data from URL

- Normalize the data to json normalize

- Pre processing the data

- Filter data in data frame

- Github= https://github.com/sahil9939/IBM-Data-Science-/blob/3aa7c5f0714e3e4ceaf14dbe7028435883940ba3/question1%20api.ipynb

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

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

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

```
response.status_code
```

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

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe

data.head()
```

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have multiple payloads in a singl
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

```
# Hint
data_falcon9 = df[df.BoosterVersion == 'Falcon 9']
data_falcon9
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
5	8	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857

Data Collection - Scraping

- **Request the Falcon9 Launch Wiki page from its URL**

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

- **Extract all column/variable names from the HTML table header**

```
[8]: # 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')
```

Starting from the third table is our target table contains the actual launch records.

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

- **Create a data frame by parsing the launch HTML tables**

```
]: 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']=[]
```

- Github=https://github.com/sahil9939/IBM-Data-Science-/blob/3aa7c5f0714e3e4ceaf14dbe7028435883940ba3/question2%20webscraping.ipynb

Data Wrangling

- **Calculate the number of launches on each site**
- **Calculate the number and occurrence of each orbit**
- **Calculate the number and occurrence of mission outcome per orbit type**
- **Create a landing outcome label from Outcome column**
- Github=https://github.com/sahil9939/IBM-Data-Science-/blob/3aa7c5f0714e3e4ceaf14dbe7028435883940ba3/question3%20data%20wrangling.ipynb

```
# Apply value_counts() on column LaunchSite
df.value_counts('LaunchSite')
```

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

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then assign it to a variable landing_outcomes.

```
# landing_outcomes = values on Outcome column
landing_outcomes = df.value_counts('Outcome')
```

`True Ocean` means the mission outcome was successfully landed to a specific region of the ocean while `False Ocean` means the mission outcome was unsuccessful
outcome was successfully landed to a ground pad `False RTLS` means the mission outcome was unsuccessfully landed to a ground pad. `True ASDS` means the missic
mission outcome was unsuccessfully landed to a drone ship. `None ASDS` and `None None` these represent a failure to land.

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
# Apply value_counts on Orbit column
df.value_counts('Orbit')
```

```
Orbit
GTO     27
ISS     21
VLEO    14
```

```
# landing_class = 0 if bad_outcome
landing_class=[]
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
# landing_class = 1 otherwise
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land suc

```
df['Class']=landing_class
df[['Class']].head(8)
```

EDA with Data Visualization

- All the chart summary
  - Visualize the relationship between Flight Number and Launch Site
  - Visualize the relationship between Payload and Launch Site
  - Visualize the relationship between success rate of each orbit type
  - Visualize the relationship between Flight Number and Orbit type
  - Visualize the relationship between Payload and Orbit type
  - Visualize the launch success yearly trend
- All the visualization chart help in providing overview of relation of each element help in success or failure of falcon 9 landing.

GitHub=https://github.com/sahil9939/IBM-Data-Science-/blob/3aa7c5f0714e3e4ceaf14dbe7028435883940ba3/question5%20visualization.ipynb

EDA with SQL

- Using SQL to manipulate data frame
- Name of the unique launch sites
- 5 records where sites begin with the string 'CCA'
- Total payload mass carried by boosters.
- Average payload mass carried by booster
- Name of booster which have success in drone ship and have payload mass greater than 4000 but less than 6000.
- Total number of successful and failure mission outcome
- Booster version which have carried the maximum payload mass
- Landing outcomes in drone ship and launch site name for the year 2015
- Rank the count landing outcome between the date 04-06-2010 and 20-03-2017

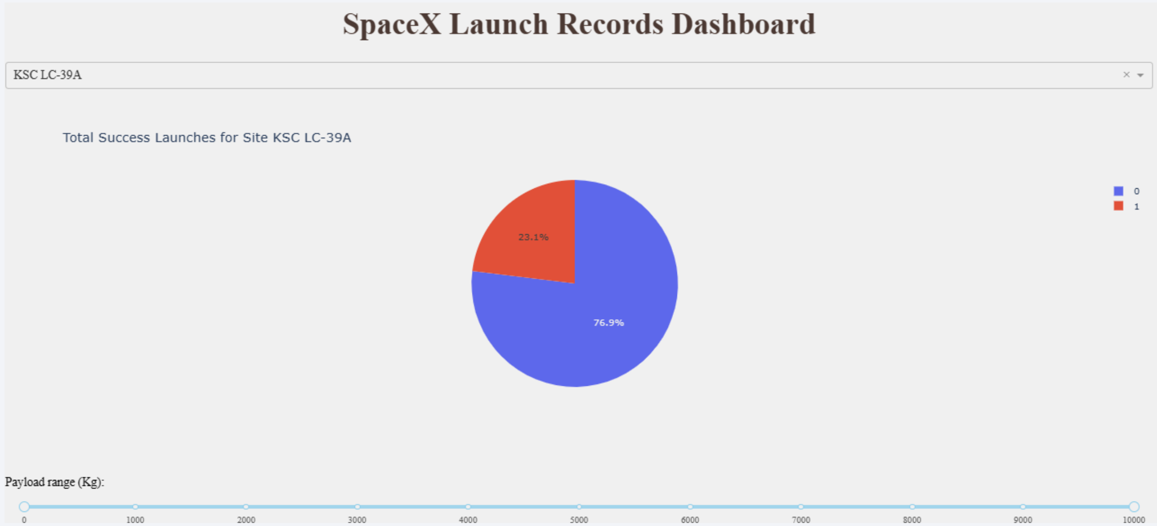
GitHub=https://github.com/sahil9939/IBM-Data-Science-/blob/3aa7c5f0714e3e4ceaf14dbe7028435883940ba3/question4%20sql.ipynb

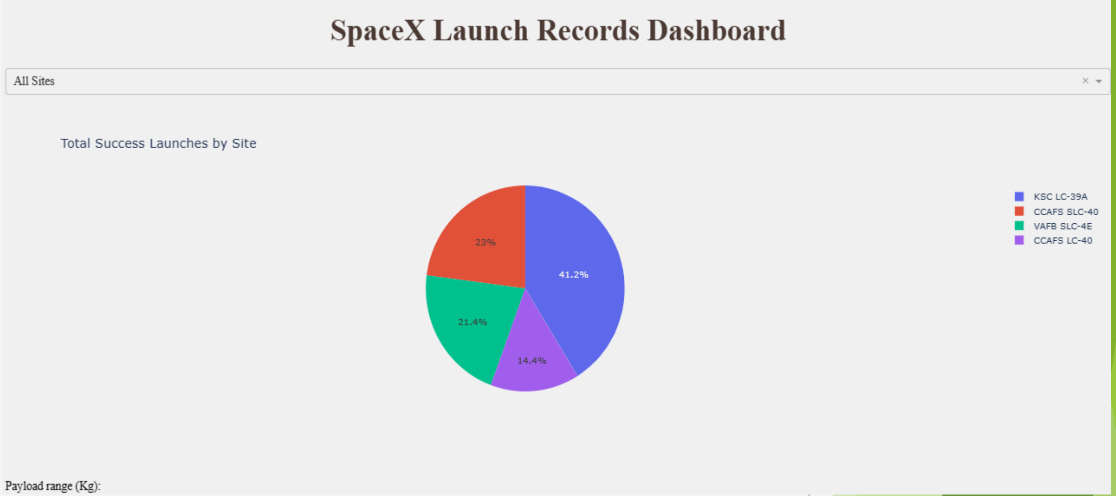
Build an Interactive Map with Folium

- Folium.marker() , folium.cicle(), folium.icon(), folium.polyline(), folium.plugins.antpath() and markercluster() are some function used in for map manipulation.

GitHub=https://github.com/sahil9939/IBM-Data-Science-/blob/3aa7c5f0714e3e4ceaf14dbe7028435883940ba3/question6%20folium%20complete.ipynb

Build a Dashboard with Plotly Dash





GitHub= https://github.com/sahil9939/IBM-Data-Science-/blob/acc443c1a534dca733362f5bafe5ebb6a2b94db1/question8%20plotly.py

Predictive Analysis (Classification)

- Build the model
  - Create column for the class
  - Standardize the data
  - Split the data info train and test stes
  - Build the model and fit the data
- Evaluate the model
  - Calculate the accuracy
  - Calculate the confusion matrixes
  - Plot the result
- Finding the optimal model
  - Find the hyperparameters for the model
  - Find the accuracy
  - Confirm the optimal model

Github= https://github.com/sahil9939/IBM-Data-Science-/blob/3aa7c5f0714e3e4ceaf14dbe7028435883940ba3/question8%20last%20not%20complete.ipynb

Results

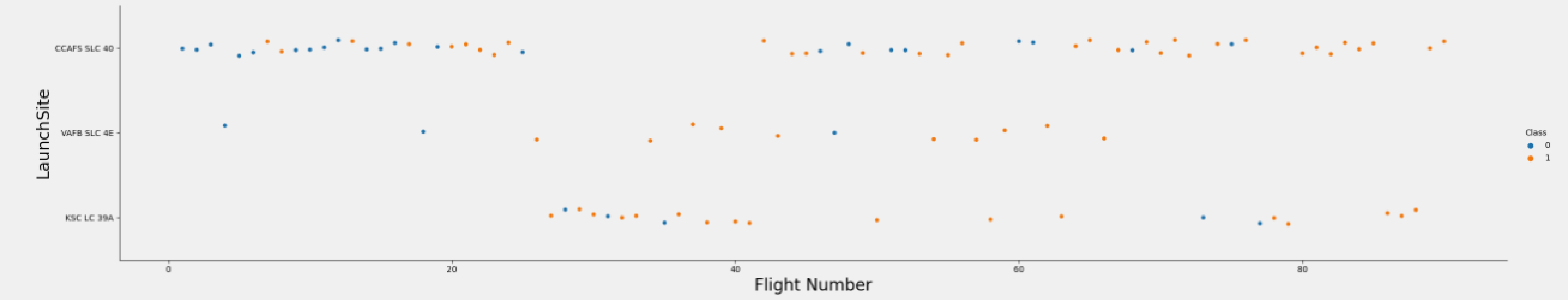
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

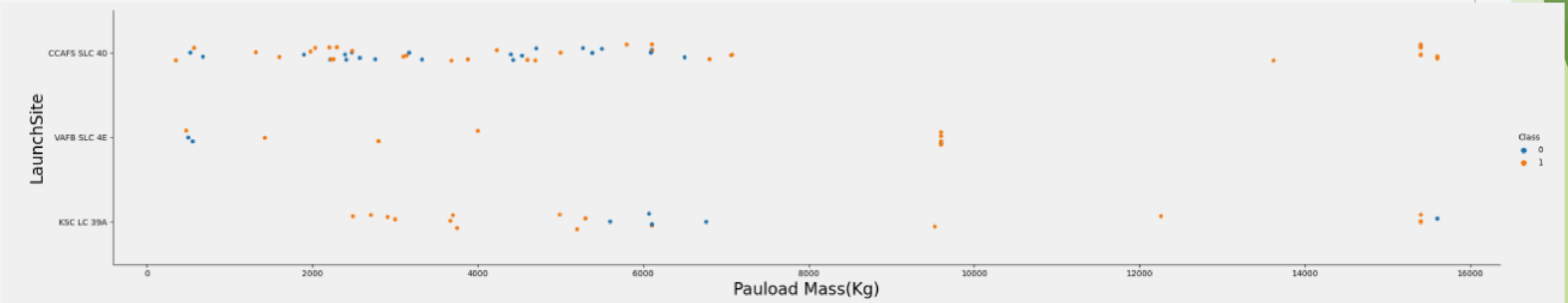
- Scatter plot of Flight Number vs. Launch Site



- Launches from the site of CCAFS SLC 40 are higher than other sites

Payload vs. Launch Site

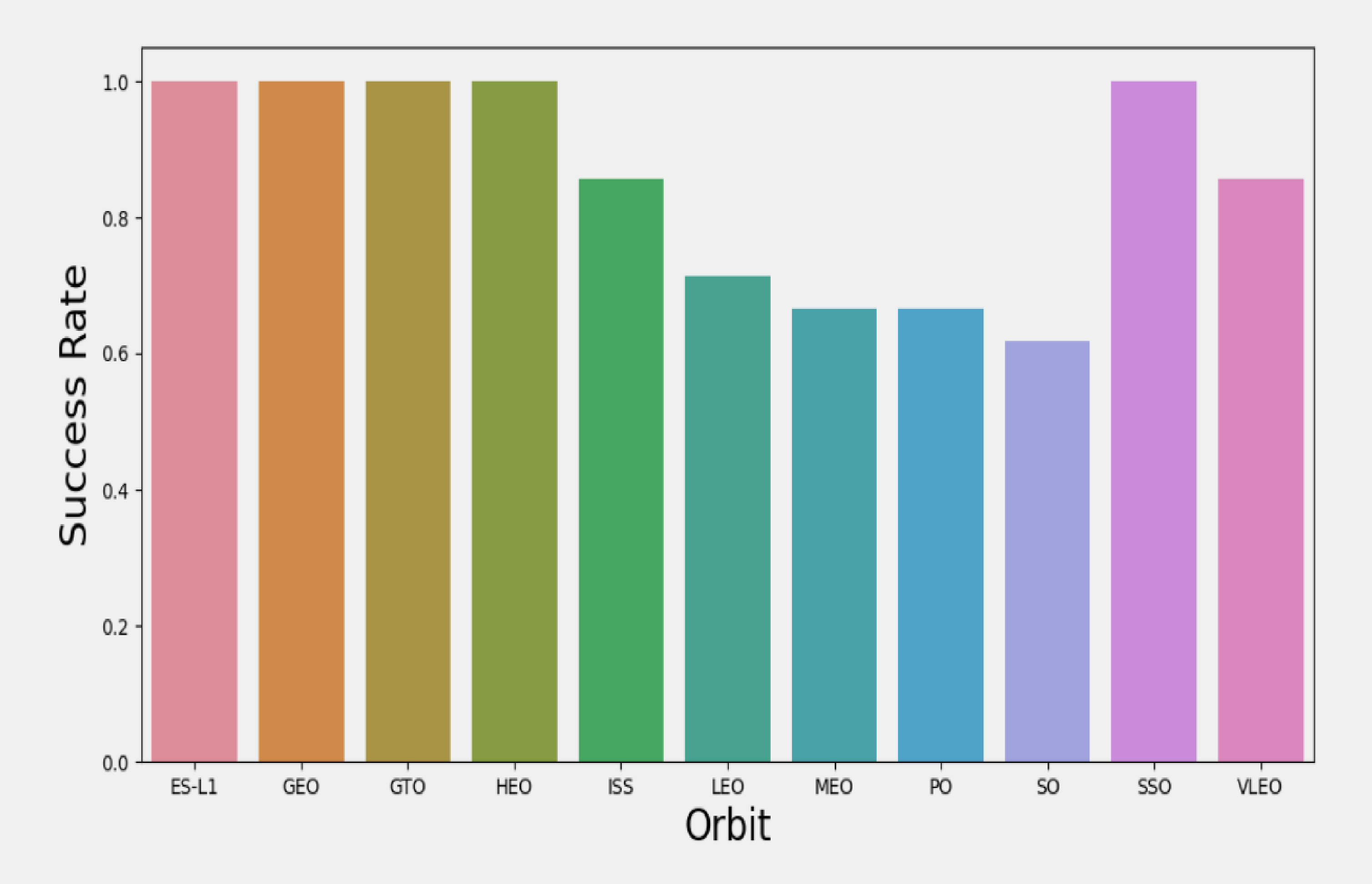
- Scatter plot of Payload vs. Launch Site



- Pay load mass is higher in CCAFS SLC 40 compare to other site

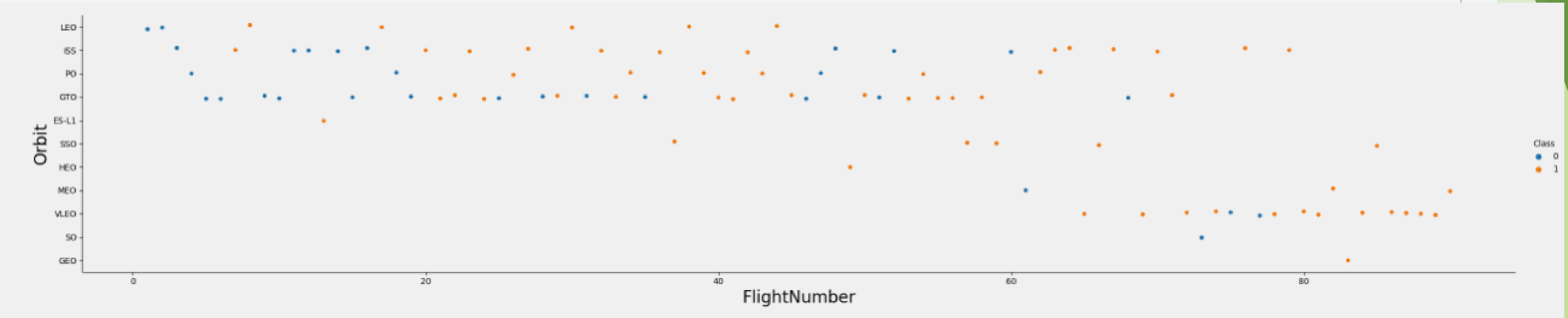
Success Rate vs. Orbit Type

- Bar chart for the success rate of each orbit type
- The orbit of ESL1, GEO, GTO, HEP and SSO have higher success rate



Flight Number vs. Orbit Type

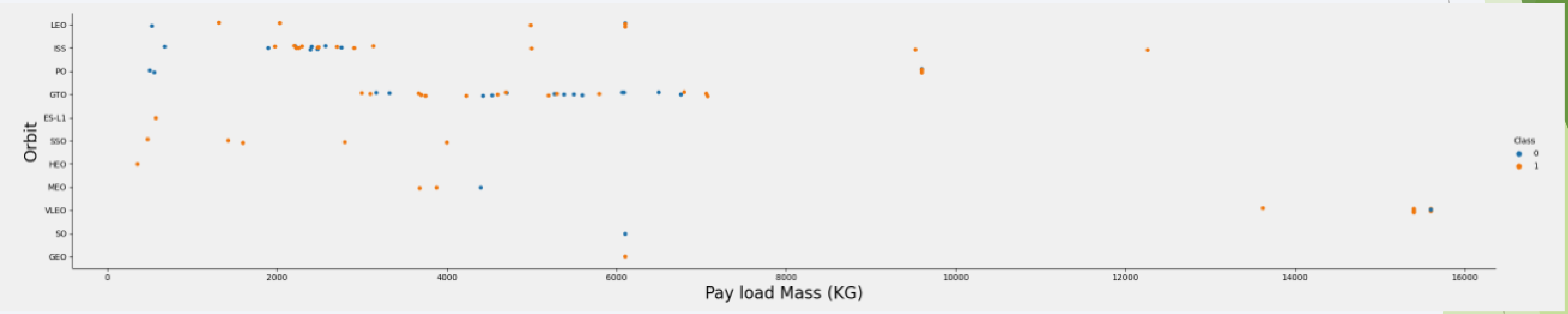
- Scatter point of Flight number vs. Orbit type



- There is shift in trend for VLEO orbit is higher number flight

Payload vs. Orbit Type

- Scatter point of payload vs. orbit type

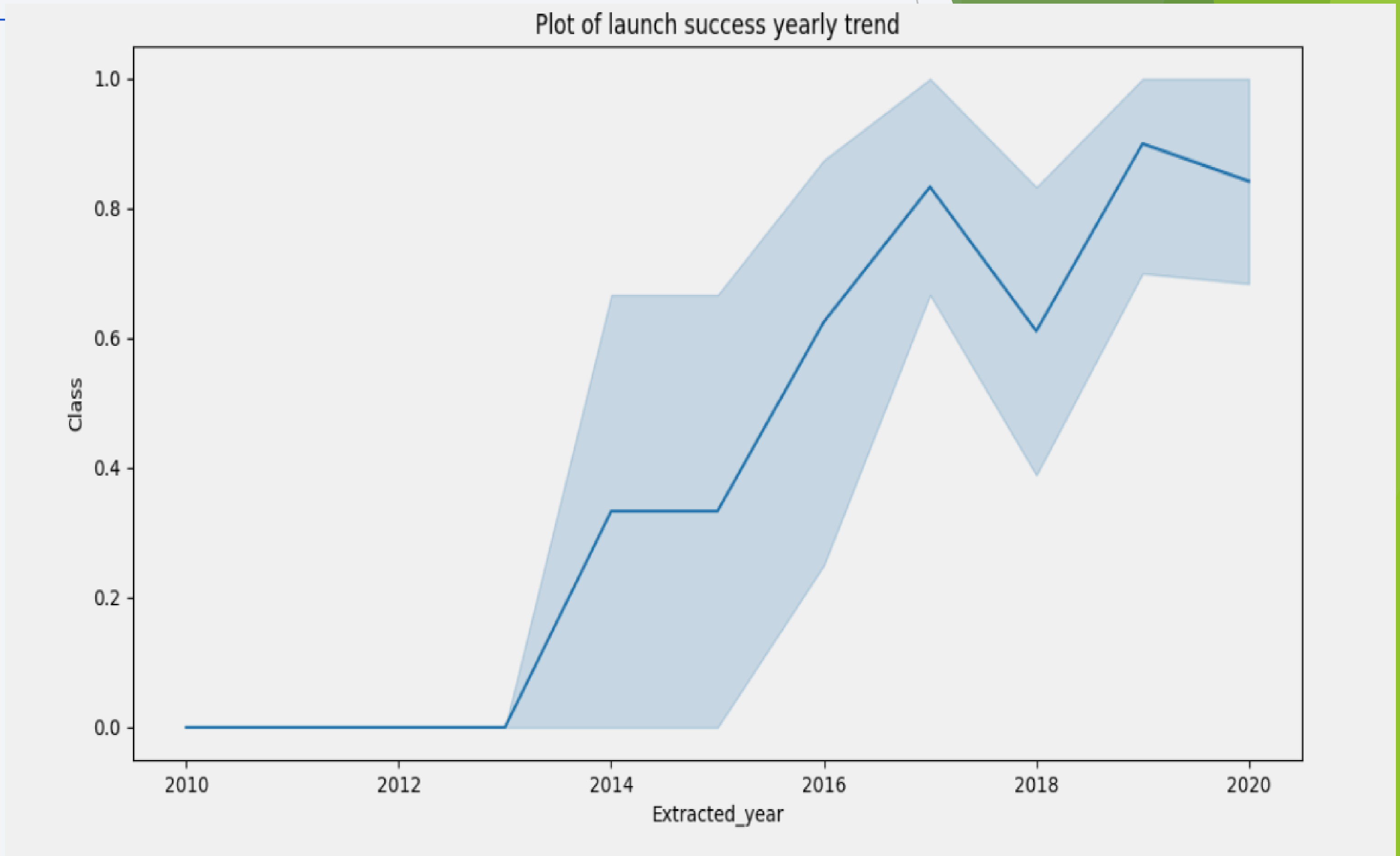


- ISS has Pay load mass from the range of 2000 to 4000kg of mass
- GTO has Pay load mass form the range of 3000 to 7000kg of mass

Launch Success Yearly Trend

- Line chart of yearly average success rate

- Launch success rate is on forward trend form 2013 onwards



All Launch Site Names

- Names of the unique launch sites

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

- Present your query result with a short explanation here

```
[8]: %sql select distinct(Launch_Site) from SPACEXTBL
      * sqlite:///my_data1.db
     Done.
[8]: ,,,,,,,,,,,,,,,,,,,,,,
```

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Launch Site Names Begin with 'CCA'

- 5 records where launch sites begin with `CCA`

1. F9 v1.0 B0003 CCAFS LC-40
2. F9 v1.0 B0004 CCAFS LC-40
3. F9 v1.0 B0005 CCAFS LC-40
4. F9 v1.0 B0006 CCAFS LC-40
5. F9 v1.0 B0007 CCAFS LC-40

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

```
[8]: %sql select Booster_Version, launch_site from SPACEXTBL where launch_site like 'CCA%' limit 5
 * sqlite:///my_data1.db
Done.
```

[8]:

Booster_Version	Launch_Site
F9 v1.0 B0003	CCAFS LC-40
F9 v1.0 B0004	CCAFS LC-40
F9 v1.0 B0005	CCAFS LC-40
F9 v1.0 B0006	CCAFS LC-40
F9 v1.0 B0007	CCAFS LC-40

- Present your query result with a short explanation here

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

Potal payload= 619967.0

- Present your query result with a short explanation here

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

```
[8]: %sql select Booster_Version, launch_site from SPACEXTBL where launch_site like 'CCA%' limit 5
```

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

[8]:

Booster_Version	Launch_Site
F9 v1.0 B0003	CCAFS LC-40
F9 v1.0 B0004	CCAFS LC-40
F9 v1.0 B0005	CCAFS LC-40
F9 v1.0 B0006	CCAFS LC-40
F9 v1.0 B0007	CCAFS LC-40

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

Average payload mass =6138.287128712871

- Present your query result with a short explanation here

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

```
%sql select AVG(PAYLOAD_MASS__KG_) from SPACEXTBL
```

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

AVG(PAYLOAD_MASS__KG_)
6138.287128712871

First Successful Ground Landing Date

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

First successful landing= 01/08/2018

- Present your query result with a short explanation here

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
38]: %sql SELECT MIN(Date) AS FirstSuccessfull_landing_date FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success (ground pad)'
 * sqlite:///my_data1.db
Done.
```

38]:

FirstSuccessfull_landing_date
01/08/2018

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

1. **F9 FT B1022**
2. **F9 FT B1026**
3. **F9 FT B1021.2**
4. **F9 FT B1031.2**

- Present your query result with a short explanation here

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

```
]: %sql  SELECT Booster_Version FROM SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000
```

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

]:

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Successful mission=100
- Failure mission=1
- Present your query result with a short explanation here

Boosters Carried Maximum Payload

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

1. F9 B5 B1048.4 15600.0
2. F9 B5 B1048.5 15600.0
3. F9 B5 B1049.4 15600.0
4. F9 B5 B1049.5 15600.0
5. F9 B5 B1049.7 15600.0
6. F9 B5 B1051.3 15600.0
7. F9 B5 B1051.4 15600.0
8. F9 B5 B1051.6 15600.0
9. F9 B5 B1056.4 15600.0
10. F9 B5 B1058.3 15600.0
11. F9 B5 B1060.2 15600.0
12. F9 B5 B1060.3 15600.0

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

```
%sql  SELECT Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_)FROM SPACEXTBL) ORDER BY Booster_Version
```

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

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600.0
F9 B5 B1048.5	15600.0
F9 B5 B1049.4	15600.0
F9 B5 B1049.5	15600.0
F9 B5 B1049.7	15600.0
F9 B5 B1051.3	15600.0
F9 B5 B1051.4	15600.0
F9 B5 B1051.6	15600.0
F9 B5 B1056.4	15600.0
F9 B5 B1058.3	15600.0
F9 B5 B1060.2	15600.0
F9 B5 B1060.3	15600.0

- Present your query result with a short explanation here

2015 Launch Records

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

1. F9 v1.1 B1012 CCAFS LC-40 Failure (drone ship)
2. F9 FT B1020 CCAFS LC-40 Failure (drone ship)

- Present your query result with a short explanation here

```
56]: %sql  SELECT Booster_Version, Launch_Site, Landing_Outcome from SPACEXTBL WHERE Landing_Outcome LIKE 'Failure (drone ship)' AND Date BETWEEN '01-01-2015' AND '1
```

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

56]:

Booster_Version	Launch_Site	Landing_Outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 FT B1020	CCAFS LC-40	Failure (drone ship)

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

Landing_Outcome	COUNT(Landing_Outcome)
Success	19
No attempt	9
Success (ground pad)	6
Success (drone ship)	5
Failure (drone ship)	3
Failure	3
Failure (parachute)	2
Controlled (ocean)	2
No attempt	1

```
%sql SELECT Landing_Outcome, COUNT(Landing_Outcome) from SPACEXTBL WHERE DATE BETWEEN '06-04-2010' AND '20-03-2017' GROUP BY Landing_Outcome ORDER BY COUNT(Lanc
```

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

Landing_Outcome	COUNT(Landing_Outcome)
Success	19
No attempt	9
Success (ground pad)	6
Success (drone ship)	5
Failure (drone ship)	3
Failure	3
Failure (parachute)	2
Controlled (ocean)	2
No attempt	1

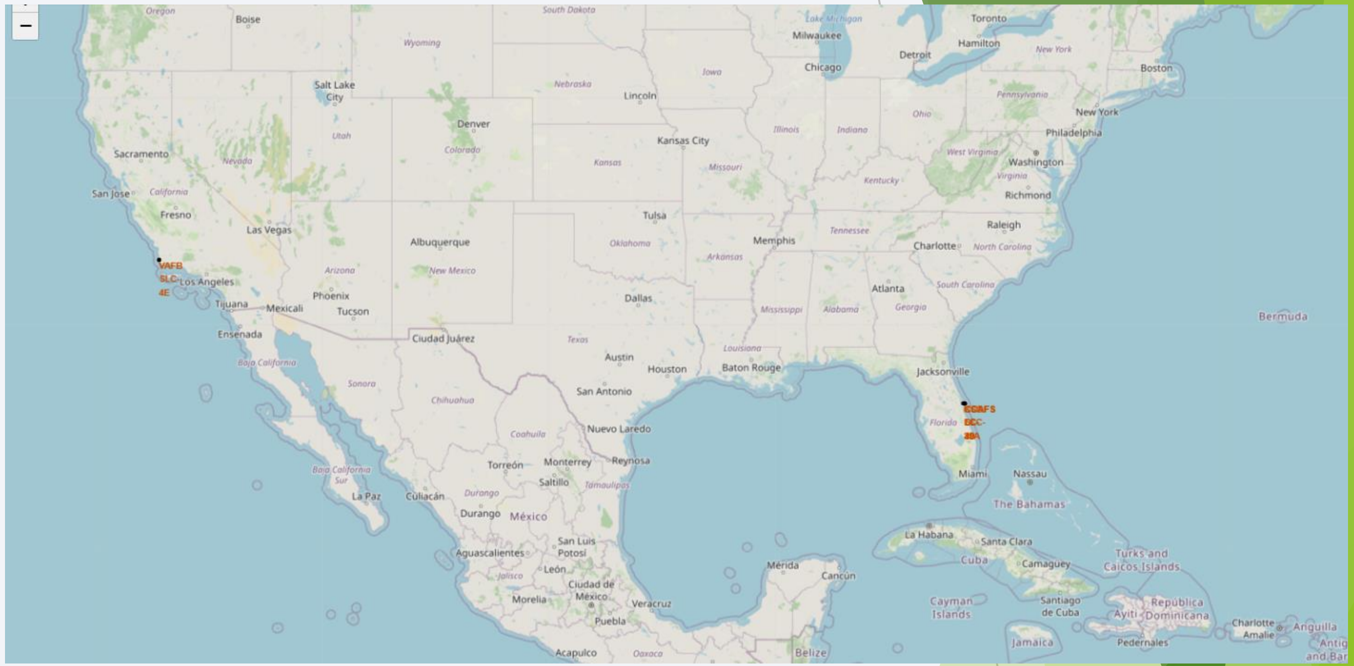
- Present your query result with a short explanation here

Section 3

Launch Sites Proximities Analysis

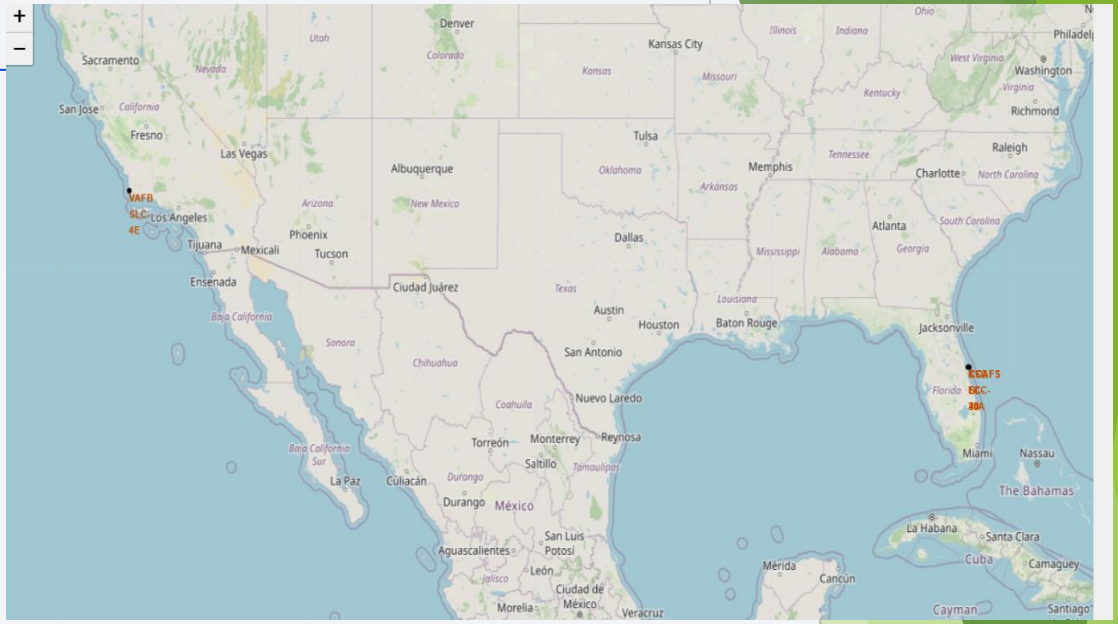
All Launch site Location marker

- All launches near USA, Florida and California.



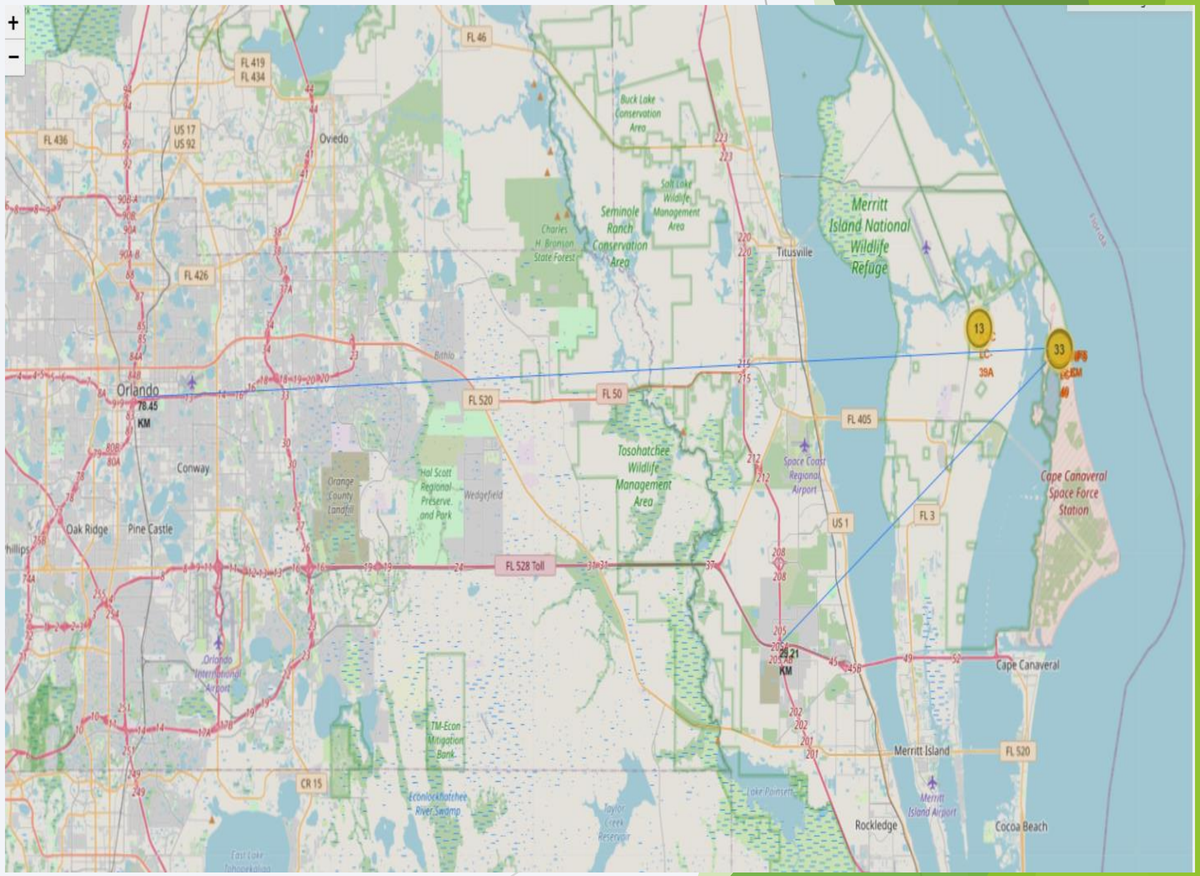
Color label launch outcome

- Color label launch outcome of CCAFS SAF 40



Launch site proximities

- Launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed

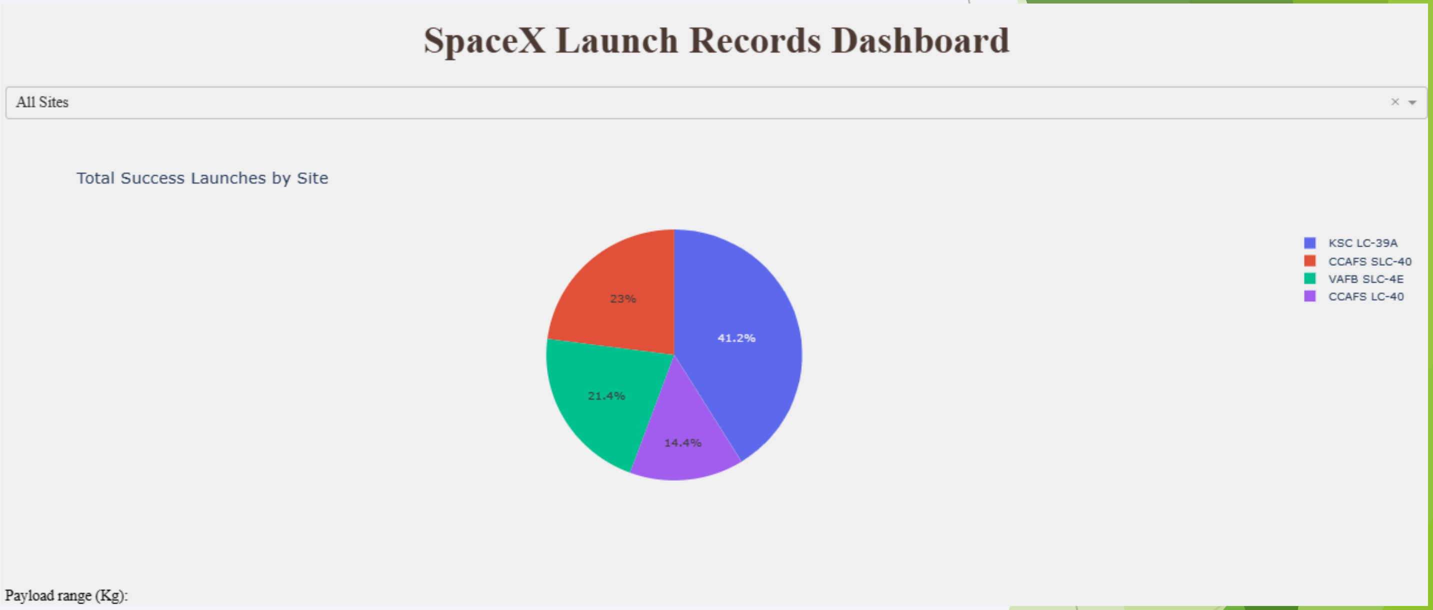


Section 4

Build a Dashboard with Plotly Dash

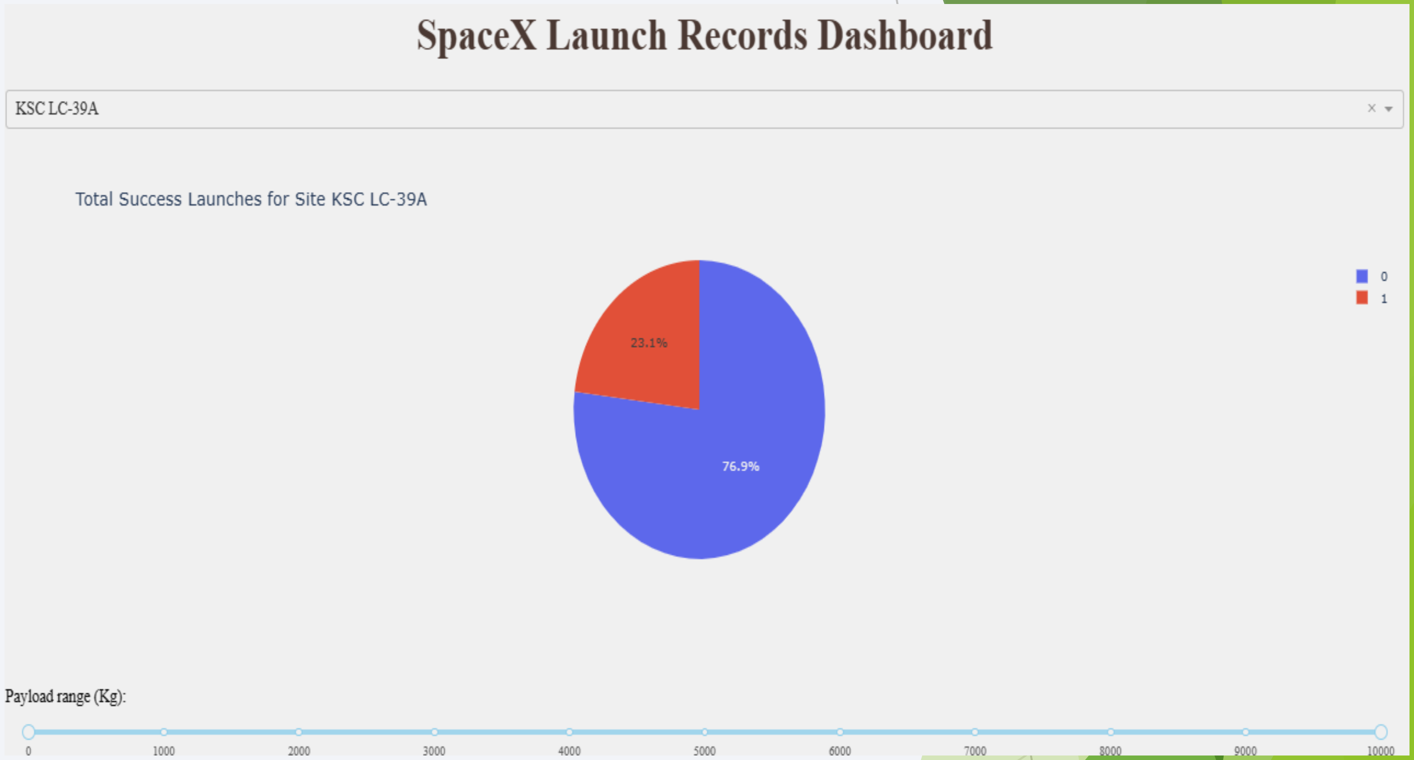
Space X Launch Record Dashboard

This pie chart represent record of all launch site.



SpaceX launch record of highest success rate

- This pie chart represent highest success launch record of all site



Payload vs launch outcome

- **Payload range of 0 to 4000kg**

- **Payload range of 4kg to 10kg**

Section 5

Predictive Analysis (Classification)

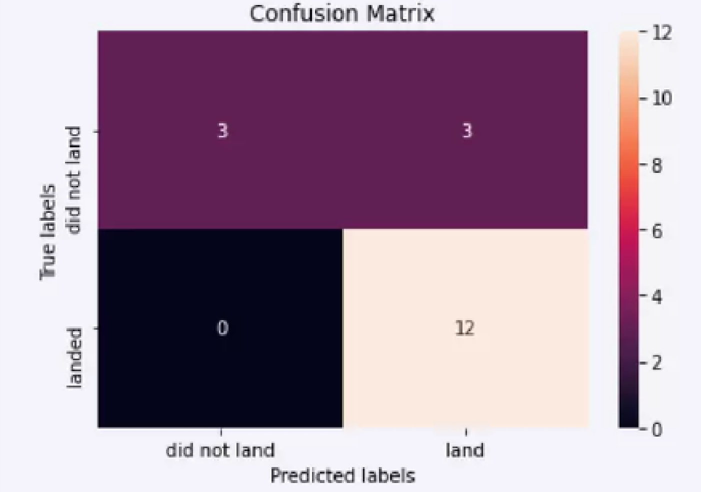
Classification Accuracy

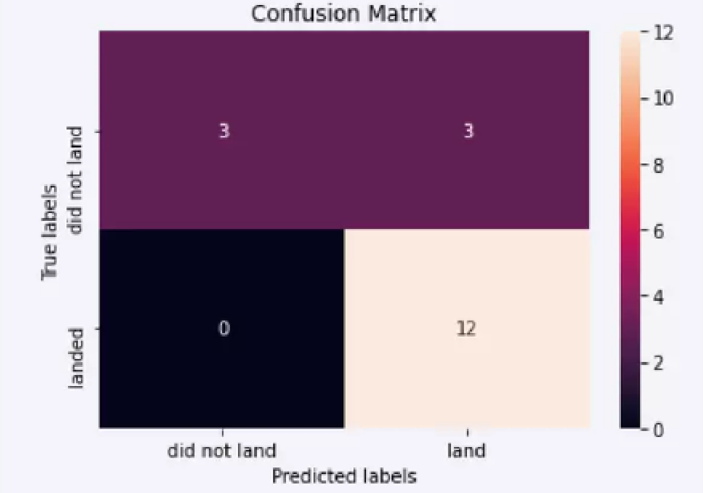
- Visualize the built model accuracy for all built classification models, in a bar chart

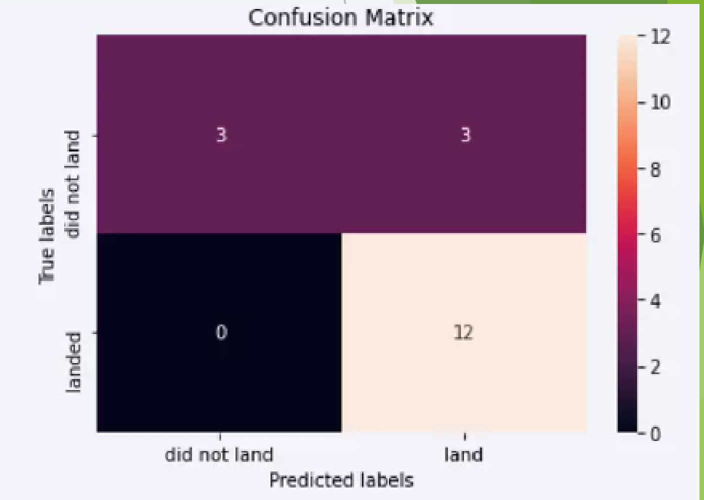
Accuracy	Algorithm
0.833	Logistic Regression
0.833	SVM
0.833	KNN
0.778	Decision Tree

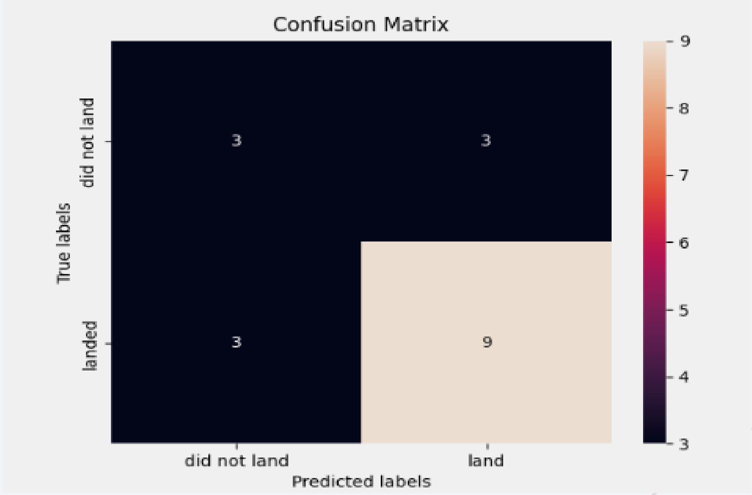
Confusion Matrix

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









Conclusions

- Low weighted payloads perform better than the heavier payloads.
- Launch success rate is on forward trend form 2013 onwards and stabilize at 2019.
- KSC LC-39A had most successful launched from all the sites.
- Orbit GEO,HEO,SSO,ES-L1 has the best success rate.
- Pay load mass is higher in CCAFS SLC 40 compare to other site

Appendix

- Data is collect from the URL https://api.spacexdata.com/v4/rockets/.
- Python reference material from "The Python Language Reference — Python 3.11.4 documentation".
- Coursera IBM Data Science course material.
- SQL reference material form-"SQL Quick Reference (w3schools.com)".
- IBM skill network labs.

Thank you!