IBM Data Science – Applied Data Science Capstone

Space X Falcon 9 Rocket launch/Landing Analysis

Johnson Chong

06-May-2023

Outline

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

Executive Summary

- **Summary of methodologies**
- In this project has used several methods and steps as below:
  - Data Collection
  - Data Wrangling
  - Exploratory Data Analysis with data visualization and SQL
  - Interactive Data Visual Analytics and Dashboard
  - Predictive Analysis (Classification)

- **Summary of all results**
- The project produce outputs and visualization as below:
  - Exploratory data analysis results
  - Geospatial Analytics
  - Interactive Dashboard
  - Predictive Analysis of classification with different Machine learning models

Introduction

Background

- SpaceX Falcon 9 rocket launch cost only $65 million USD compare with other provider $ 165 million USD)
- Greatly save launch cost because SpaceX can landing and reuse the first stage of rocket
- SpaceY company want to bid with SpaceX

Problem

- SpaceY intend to train a machine learning model to predict the successful rate for recovery by landing /reuse at first stage rocket
- Use information for the determine SpaceY company should bid aginst with SpaceX for rocket launch or not

Methodology Summary

Data Collection

- Use SpaceX Rest API with Get request to obtain data
- Also use Wikipedia with Web scraping library to collect data

Data Wrangling

- Remove Nan/ missing values
- Convert outcome into trainable Labels for Successfully./unsuccessful landing

Exploratory Data Analysis (EDA)

- **using Pandas, Matplotlib, Seaborn to visualize the dataset and find relationships between feature and determine the patterns**
- **Using SQL queries to manipulate and evaluate the SpaceX data**

Interactive visual analytics

- Geospatial analysis using Folium
- Create interactive Dashboard by using Plotly and Dash

Predictive Analysis using several classification model

- **Find the base hyperparameter by GridSearchCV for SVM, Decision Tree and logistic Regression**

Data Collection – SpaceX Rest API

- Use SpaceX Rest API with get request to obtain rocket launch data , such as launch data, boosterVersion, palyloadMass, Orbit, Landing outcome and etc.

1.Make Get Response from SpaceX API : Convert response into json format

2. loading into Pandas Dataframe by Json_normalize() function

3.Select relative column/clean data use for further data analysis

4.Create New DataFrame from New created Dictionary

5. FilterDataframe only include Falcon 9 Launches and reset the filghtNumber column --> filling miss value with mean value

GitHub Link

```
spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)

# Use json_normalize meethod to convert the json result into a dataframe
# response = requests.get(static_json_url) # get new data
data = pd.json_normalize(response.json()) # convert to json --> then normalize
```

```
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}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
# Create a data from launch_dict
data2 = pd.DataFrame(launch_dict)
```

```
# 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 paylo
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['BoosterVersion']!='Falcon 1'
data_falcon9 = data2[data2["BoosterVersion"] == 'Falcon 9']
data_falcon9

data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9

# Calculate the mean value of PayloadMass column
meanPM = data_falcon9["PayloadMass"].mean()
meanPM
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9["PayloadMass"].replace(np.nan, meanPM)
data_falcon9
```

Data Collection - Web Scraping

- Use Web scraping to collect Facon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy Launch

1. Request Wikipedia HTML Webpage URL : convert response object into unicode text

2. Create BeautifulSoup with Html5lib parser to decode HTML element in Wikipedia

3. Find launch column header in html table , extract column name one by one

4. use column name to create dictionary and iterate fill the row data record into new dictionary

5. export to new DataFrame

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

# Use BeautifulSoup() to create a BeautifulSoup object
soup = BeautifulSoup(data, 'html.parser')
```

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
for i in first_launch_table.find_all("th"):
    if extract_column_from_header(i) !=None:
        if len(extract_column_from_header(i)) > 0:
            column_names.append(extract_column_from_header(i))
```

```
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']=[]
```

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

GitHub Link

Data Wrangling -- Pandas

- To Determine whether a booster successfull landing or not, we convert the "Outcome" Column into binay value 1 = succuessful or 0= failure outcome
- Mapping Bad outcome : None None, False ASDA, False Ocean, None ASDS, False RTLS --> set to 0 value bad outcome (failure)
- Mapping Success outcome: True ASDS, True RTLS, True Ocean --> set to 1 value Success outcome
- Butput the result of mapping label (Binay value) into new "Class" column add into the dataframe

[GitHub Link]

1.Calculating the number of launches at each site

```
# Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

2.Calculating number and occurrence of each orbit

```
# Apply value_counts on Orbit column
df["Orbit"].value_counts()
```

3. Calculating number of mission Outcome per obit type

```
# landing_outcomes = values on Outcome column
landing_outcomes = df["Outcome"].value_counts()
```

4. Creating landing outcome label from "Outcome" column

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
def onehot(item):
    if item in bad_outcomes:
        return 0
    else:
        return 1
landing_class = df["Outcome"].apply(onehot)
```

5. Calculating the success rate for every landing in dataset

```
df["Class"].mean()

0.6666666666666666
```

6. Exprot DataFrame to CSV file

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

Exploratory Data Analysis (EDA) with Data Visualization

Exploratory Data Analysis and Feature Engineering with Pandas and matplotlib (Plot Chart)

Scatter Chart

- Flight Number Vs Launch SIte , Payload vs Launch Site, Flight Number vs Orbits, Payload vs Orbit type
- Scatter Chart useful to observe the two feature relationshop or correlations

Bar Chart

- Orbits Type vs Success Rate
- Bar chat useful to compare the numberical value to categorical variable , one axis represent a category, other eaxis represent discreate numberical value, Observae the relationship between two axis

Line Chart

- Year vs Success Rate
- Line Chart shows both two axises data variable, show change of variable over time and very clearly to predict the data trend

GitHub Link

Exploratory Data Analysis (EDA) SQL

Load Dataset into SQL Database, use some SQL query to retrival the lanuch data

1. Display the name of unique launch sites in space mission
2. Display 5 records where launch site begin with string "CCA"
3. Display total payload mass carried by booster launched by NASA (CRS)
4. Diplay average payload mass carried by booster version F9 v1.1
5. List the date when the first successful landing outcome in ground pad was achieved
6. List name of boosters which hasd success on a drone shio and a payload mass between 4000 and 6000 kg
7. List the total number of successful and failure mission outcomes
8. List the name of the booster_versions which have carried maximum payload mass.
9. List the month names, failed landing_outcomes on drone ships, their booster_versions, and launch site names for 2015
10. Rank count of successful landing_outcomes between date 04-06-2010 and 20-03-2017 in descending order

GitHub Link

Build Interactive Map - Geospatial Analysis

Use Folium to visualize the launch data at interactive map

Mark all launch sites on a map

- Add a folim circle and folium marker for each launch site on the launch map

Mark success/failed launches for each site on map

- Maker with different colour for indicate the sucessful (class=1) as green , the failure (class=0) as red
- Each launch result in dataframe, add into folium.Marker to Maker_Cluster()
- Add icon as text Label with Maker color for indicate the successful and failure launch result

Calculate the distance between a launch site to its proximities

- proximity of launch sites using Lat and Long values to calucate the distance between the launch site and poximity points
- Use Lat and Long coordinate value mark point by folium Marker show the distance
- Use folium Polyline to draw the distance line between the launch site and poximity point at the map

[GitHub Link]

Interactive Dashboard – Plotly Dash

Use the Plotly and Dash framework to build interactive Dashboard, the dashboard contain several chart to display selective result

Pie chart

- showing the total successful of launches by launch site
- This chart can use drop down menu select/filter all site or different site to show the success vs failure ratio for each launch site

Scatter Plot

- Show correlation between outcome (success or failure) and payload mass (kg)
- This chart can using rangeSlider for select the ranges of payload mass, and then update the chart
- It also can use drop down menu for select/filter booster version

GitHub Link

Predictive Analysis (Classification)

Build the Predictive Analysis classification Model and find the best performance model

Building Model

- Data Preprocessing to standardize the data
- Create (class) column for Target (label) column to train/test the model
- Prepare split the train/test dataset by train_test_split() function
- Use training dataset to train different Machine Learning

Evalution Model Performance and Fine Tune hyperparameter

- Use Accuracy score and best_score for evaluate the training performance
- Use GridSearchCV() Algorithm to tune the hyperparameter for different model to obtain better performance
- Plot the confusion matrix to evalution the prediction classification performance at testing dataset

Comparsion different Model

- Review the Accuracy score for compare all selected machine learning algorithm perform
- Find the best (Highest accurary) score and perform Classification model

GitHub Link

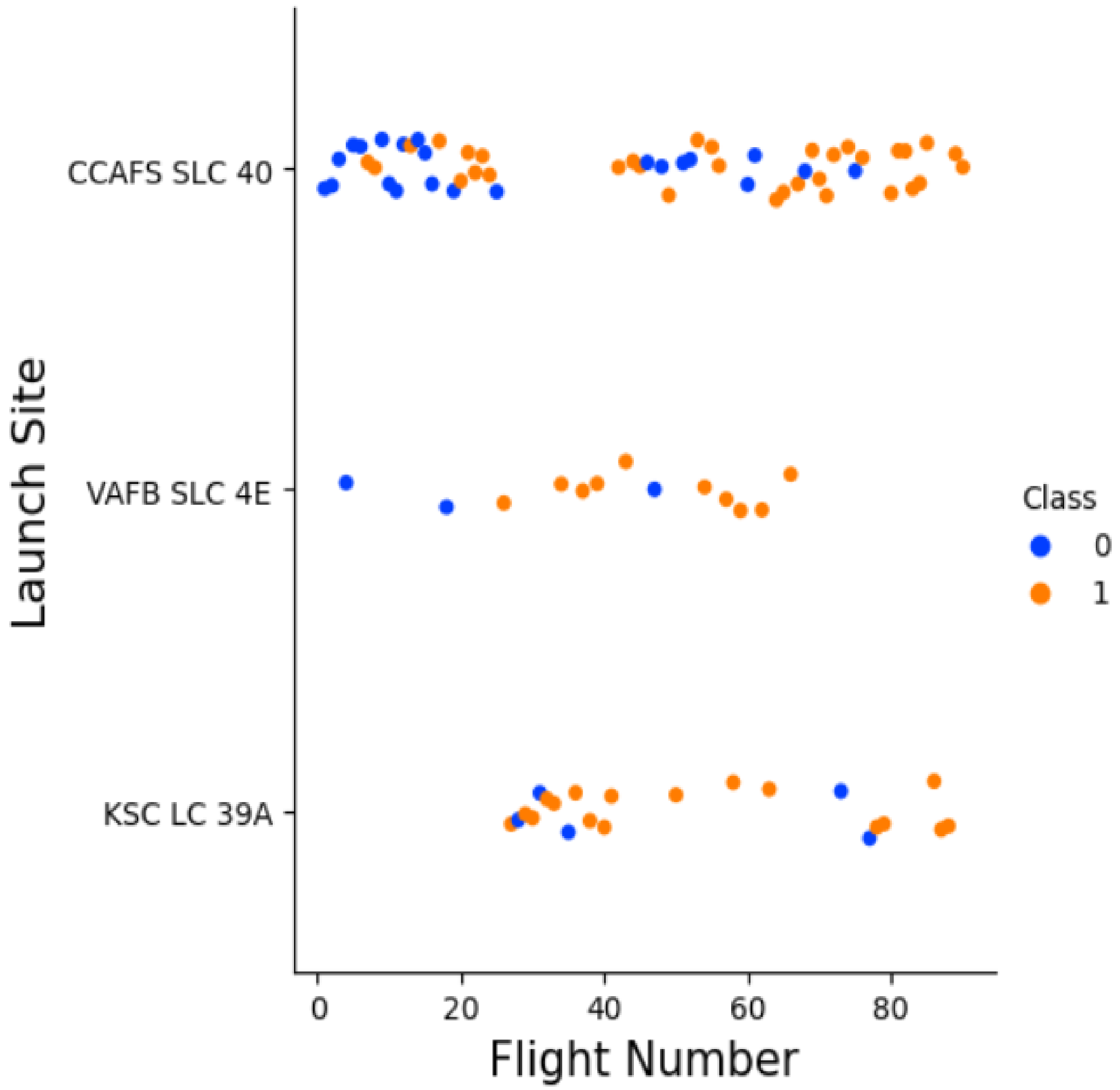
Results

- Exploratory Data Analysis Result
- Interactive Analytics Result
- Predictive Analysis Result

Flight Number vs Launch Stie

Use Scatter plot to show relationship between Flight Number vs Launch Site

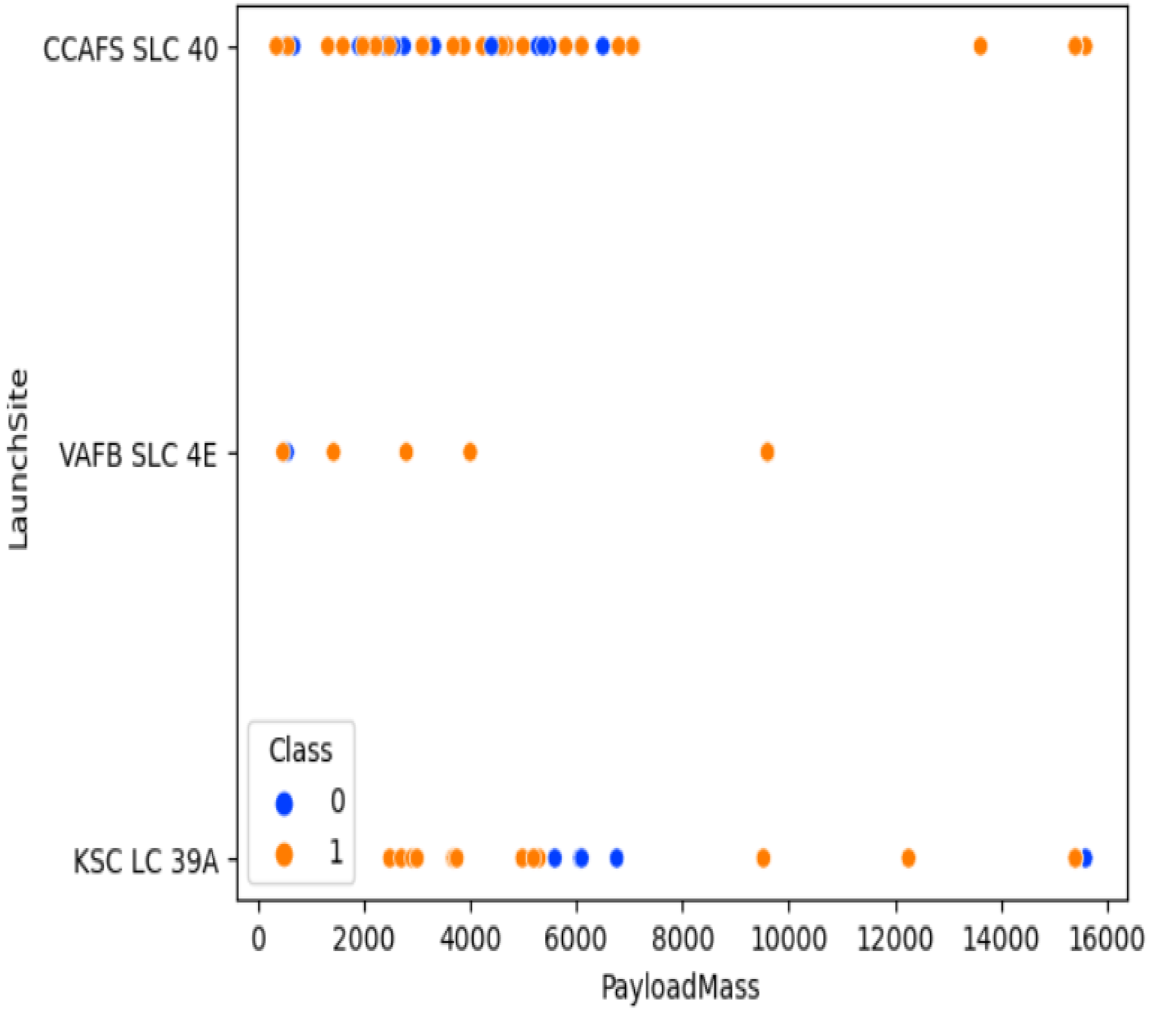
- Class 0 (blue) represent failure launch, Class 1 (orange) represents successful lanuch
- The Result show the succussful rate increase when Flight Number also increase
- specially Flight numbers larger than 20, the success rate has increased significantly
- At CCSFS SLC 40 launch Site the early fight number < 30, the normally higher failure rate
- At KSC LC 39A launch Site no early flight launches from this site, so it is more successful launch rate
- At VAFB SLC 4E launch site show early flight are high failure rate



Payload vs Launch Site

scatter plot to show relationship between Payload vs Launch Site

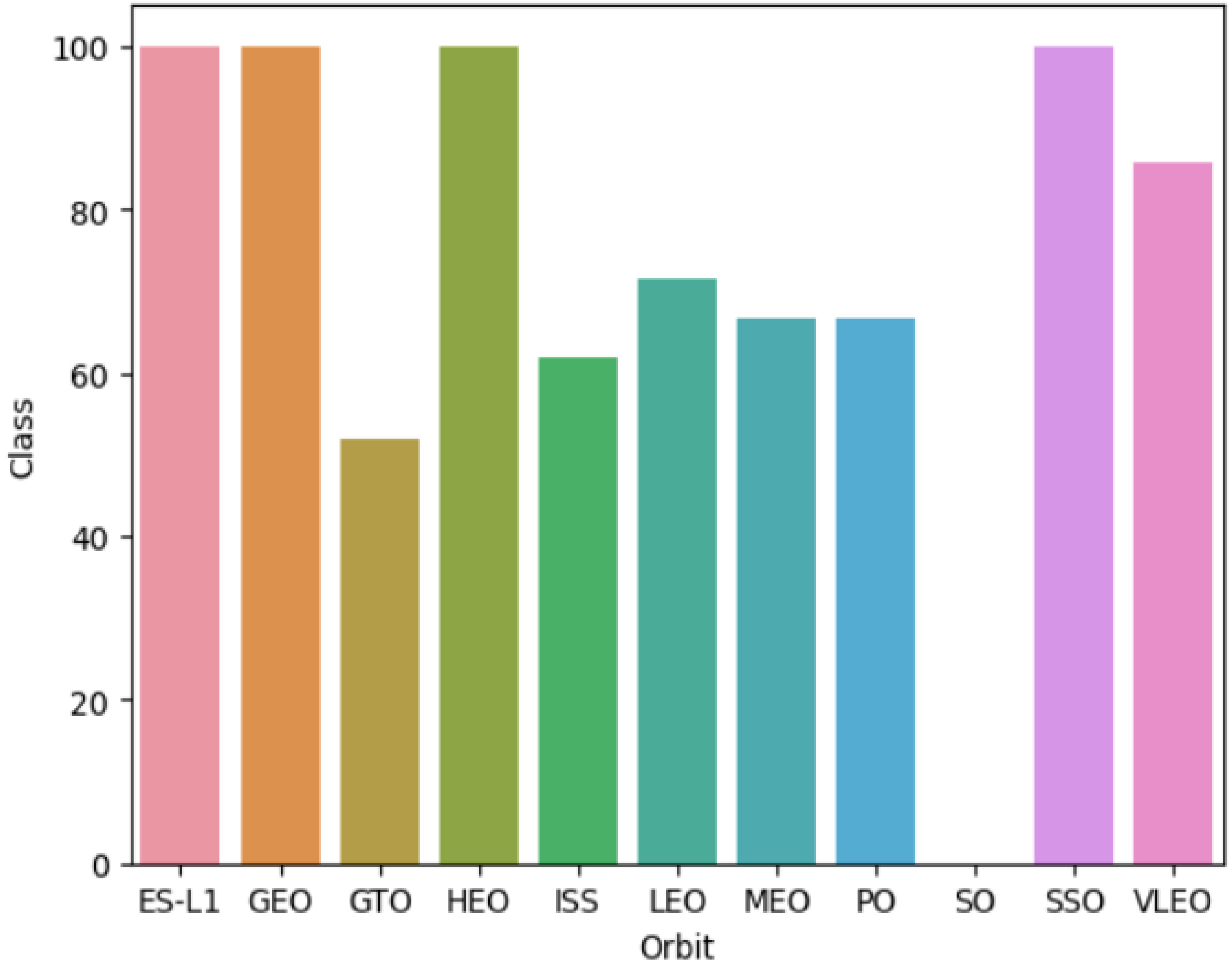
- Class 0 (blue) represent failure launch, Class 1 (orange) represents successful launch
- Most of the payload mass launch between 0-7000kg
- Over 7000kg payload mass ,there are less launch data, and also very few failure landing
- At VAFB SLC 4E launch site, payload mass is lighter than other site
- No clear correlation between Payload mass and success rate with given the launch site



Success Rate vs Orbit Type

Bar Chat to show the Success rate vs Orbit Type

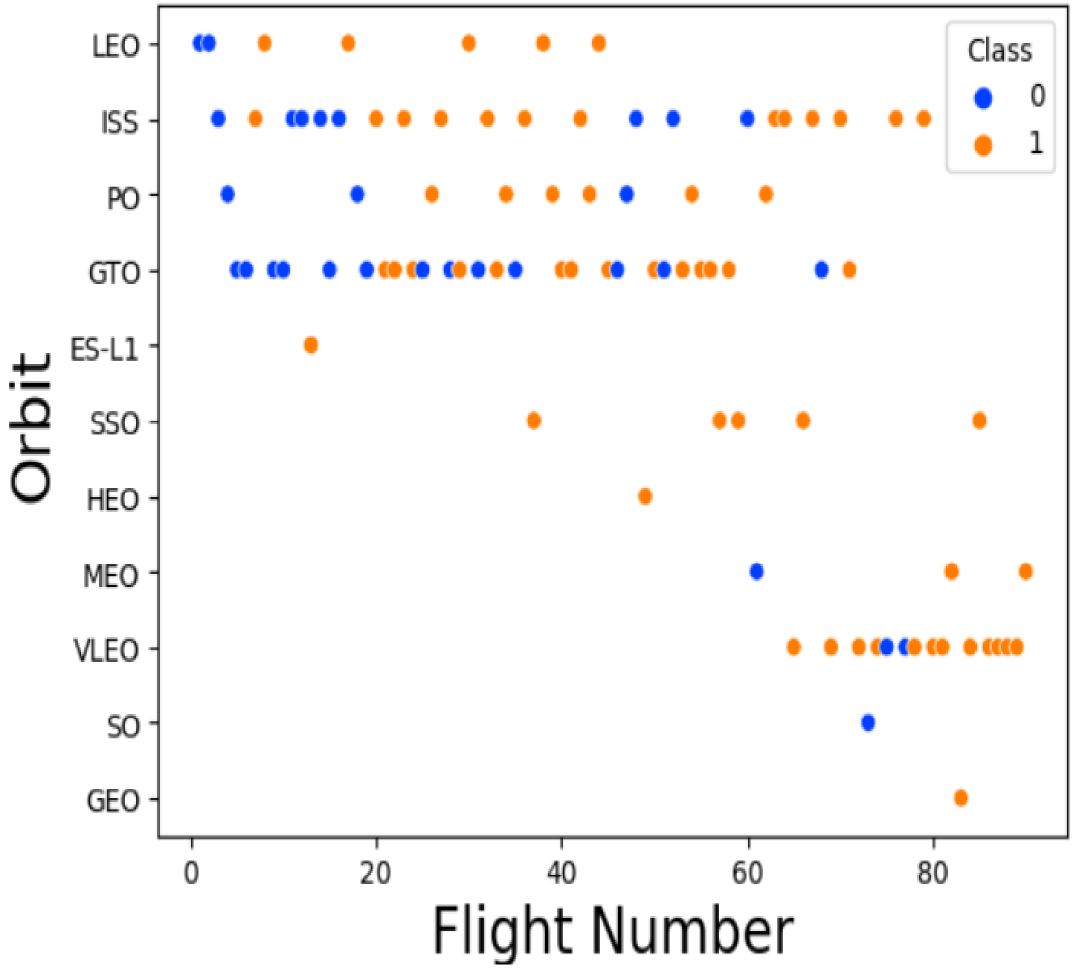
- The hightest (100 %) success rate Orbit type are ES-L1, GEO, HEO, SSO
- The Orbit Type GTO only 50% success rate
- The lowest (0%) success rate Orbit type is SO



Flight Number vs Orbit Type

Use Scatter plot to show relationship between Flight Number vs Orbit Type

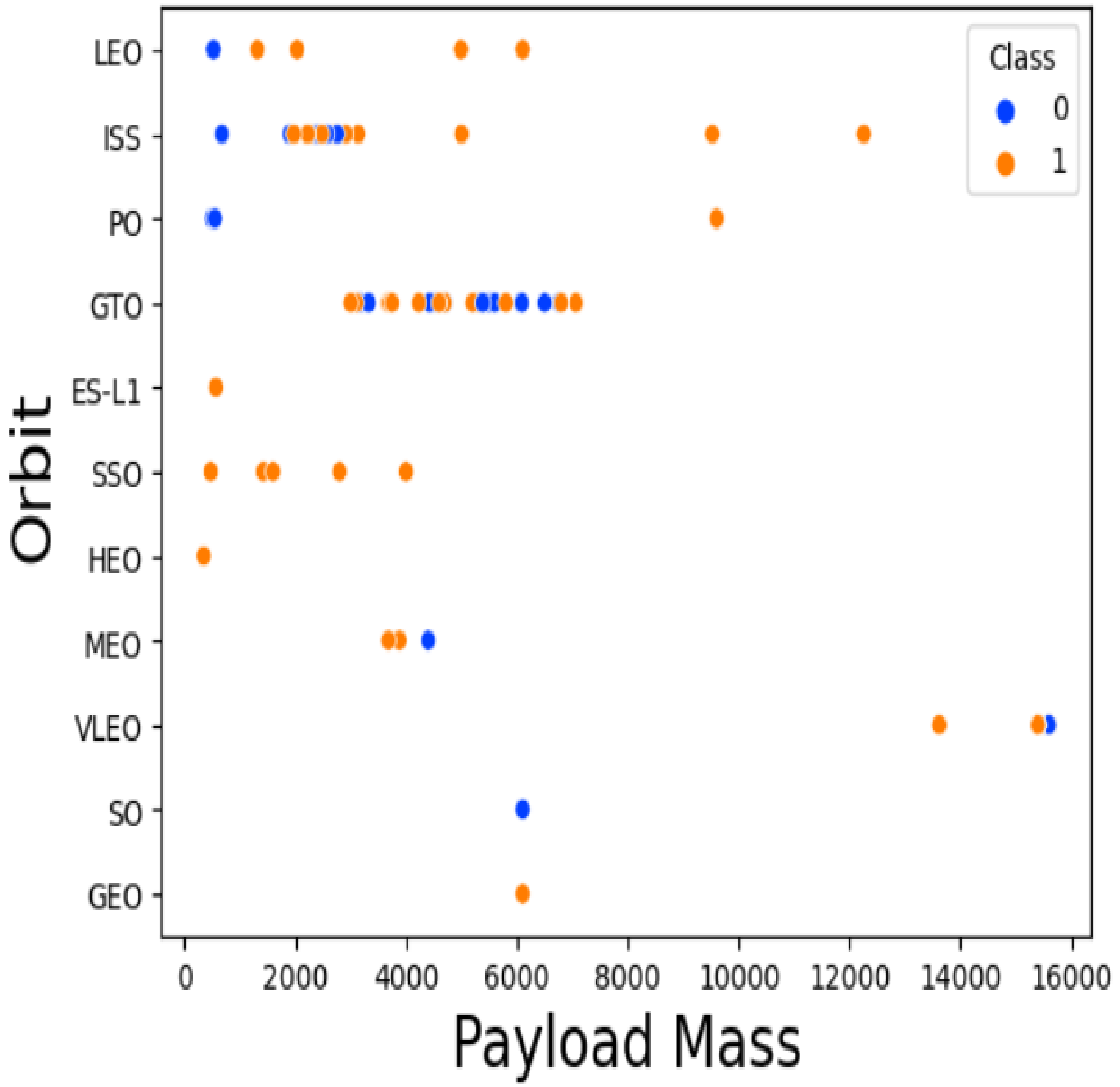
- Class 0 (blue) represent failure launch, Class 1 (orange) represents successful launch
- The Result show most of orbit type the successful outcome correlated with flight number
- Generally, successful rate increase (higher) when the flight number increase
- Orbit Type GEO, HEO, and ES-L1 are 100% successful by only having 1 flight
- SSO Orbit Type is 100% success rate with 5 successful flights
- GTO orbit type no relationship between success rate and flight numbers
- LEO orbit type unsuccessful launch only happen in early (low) flight number



Payload vs Orbit Type

Scatter Plot of Payload Mass vs Orbit Type

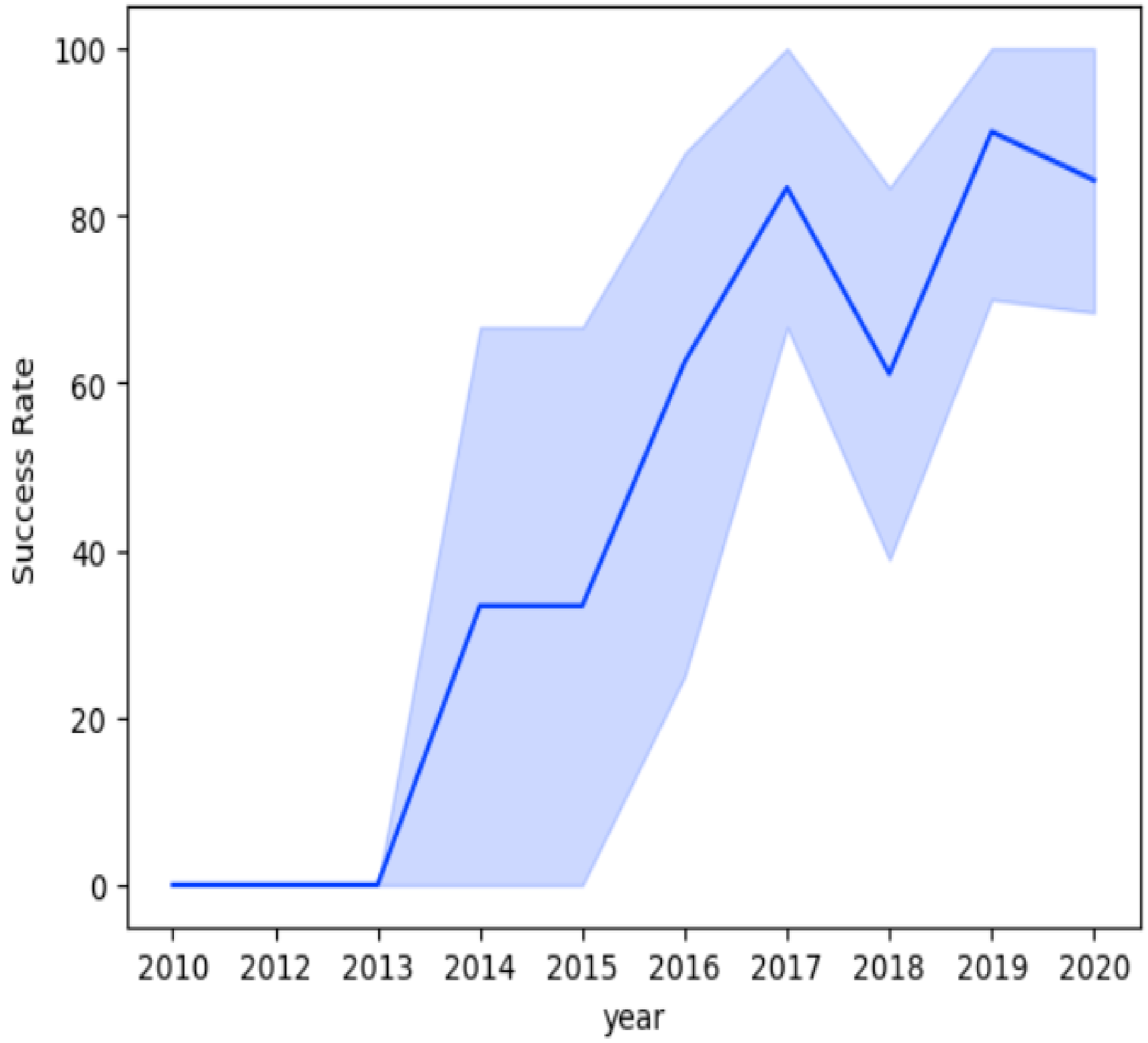
- Class 0 (blue) represent failure launch, Class 1 (ora nge) represents successful launch
- For LEO, ISS and PO orbit type has higher successful rate when the payload mass is heavy
- GTO orbit type seem no relationship between payload mass and orbit type
- SSO, HEO, ES-L1 orbit type are relate with lighter payload mass
- VLEO orbit type are relate with heavier payload mass



Launch Success Yearly Trend

Use Line Chart to show the average launch success rate yearly

- Between 2010 and 2013, all landing success rate is 0% (all failure)
- After 2013, overall trend the success rate are increasing
- After 2015, the success rate increaing to overt 50% chance
- In 2020, the success rate in around 80%



EDA with SQL

mirror_mod.mirror_object
operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True
selection at the end -add
_ob.select= 1
er_ob.select=1
ntext.scene.objects.act
("Selected" + str(modifie
irror_ob.select = 0
bpy.context.selected_obj
ata.objects[one.name].se
int("please select exactly
OPERATOR CLASSES
types.Operator):
X mirror to the selected
ject.mirror_mirror_x"
rror X"

All Launch Site Names

SQL query to find the names of unqiue launch sites

- Use SQL distinct keyword only return unique value from the LAUNCH_SITE column of SPACEXTBL table
- There are four unique launch site: CCAFS LC-40, VAFB SLC-4E, KSC LC-39A, CCAFS SLC-40

```
%sql select distinct LAUNCH_SITE from SPACEXTBL;
```

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

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

Launch Site Name Begin with 'CCA'

SQL query to find lanch site

- Use Limit 5 fetches only 5 record of SpaceX table
- Use like keyword and sign(%) together to search the name staring with 'CCA ' at the Launch_site

```
%sql select * from SPACEXTBL where launch_site like "CCA%" limit 5 ;
```

Python

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

SQL query with calculate Total payload Mass

- Use SQL aggregate function Sum to calculate sum of total payload mass at the Payload_mass_Kg column
- Use where clause to select only customer name equal to NASA (CRS) at Customer Column

```
%sql select sum(PAYLOAD_MASS__KG_) as TOTAL_PAYLOAD_MASS from SPACEXTBL where Customer like 'NASA (CRS)';
```

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

TOTAL_PAYLOAD_MASS
45596

Average Payload Mass by F9 v1.1

SQL query with calculate Average payload Mass by booster version F9 v1.1

- Use SQL aggregate function (avg) to calculate average of payload mass at the Payload_mass_Kg column
- Use where clause to select/filter only Booster Version name equal to "F9 v1.1" at Booster Version Column

```
%sql select avg(PAYLOAD_MASS__KG_) as Average_Payload_Mass from SPACEXTBL where Booster_Version like 'F9 v1.1%';
```

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

Average_Payload_Mass
2534.6666666666665

First Successful Ground Landing Date

SQL query to find the dates of first successful landing outcome on ground pad

- Use min keyword to calculate the mini date column (First date)
- Use where clause to select/filter only LANDING_OUTCOME equal "Success (ground pad)" at Landing_outcome Column

```
# %sql SELECT MIN(DATE) AS FIRST_SUCCESSFUL_GROUND_LANDING FROM SPACEXTBL WHERE "Landing _Outcome" like 'Success (ground pad)';
%sql SELECT MIN(DATE) AS FIRST_SUCCESSFUL_GROUND_LANDING FROM SPACEXTBL WHERE "Landing_Outcome" like 'Success (ground pad)';
```

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

FIRST_SUCCESSFUL_GROUND_LANDING
01/08/2018

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL query to find the boooster version had sucessful landed on Drone Ship with Payload between 4000 and 6000 kg

- Used SQL "where" clause to filter the results only Landing outcoming equal "Success (drone ship)"
- Used SQL "and" , "between" operator to select the Payload_MASS_KEY between 4000 and 6000 kg

```
%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.
```

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

Total Number of Successful and Failure Mission Outcomes

SQL query calculate the total number of succcessful and Failure mission outcome

- Use Count Keyword total number of mission outcomes and use GROUPBY keywords to group the result by the type of mission outcome

```
%sql select "Mission_Outcome", count("Mission_Outcome") as "Total Number" from SPACEXTBL group by "Mission_Outcome";
```

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

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

Boosters Carried Maximum Payload

Use SQL query list the name of booster where carried maximum payload

- Use Distinct keyword to rtiecve ony unique name of booster versions
- Use SQL subquery with "where" and "max" keyword to select the max payload mass from Spacex Table

```
%sql select distinct Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_= (select Max(PAYLOAD_MASS__KG_) from SPACEXTBL);
```

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

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

SQL query to find the Failed Drone Ship Langing Record

- Use SQL where clause to filter the Landing_Outcome equal to "Failure (drone ship)" , use "AND" keyword for select year of 2015 only
- Use substr with Date parameter to extract the Month and Year data from Date column,

```
%%sql select Booster_Version, Launch_Site, (substr(Date,4,2)) as Month from SPACEXTBL
where "Landing _Outcome" = "Failure (drone ship)" and (substr(Date, 7, 4) = '2015');
```

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

Booster_Version	Launch_Site	Month
F9 v1.1 B1012	CCAFS LC-40	01
F9 v1.1 B1015	CCAFS LC-40	04

Rank Counts of Successful Landing Between 2010-06-04 and 2017-03-20

SQL query retrun the list of successful landings and between 2010-06-04 and 2017-03-10

- Use "where" keyword with "Between" keyword to file result of the date within the periods.
- Use "Like" keyword to filter only Success Landing outcome
- Use "group by" and "order by" with "DESC" keyword Rank the result of Successful Landing by the descending order

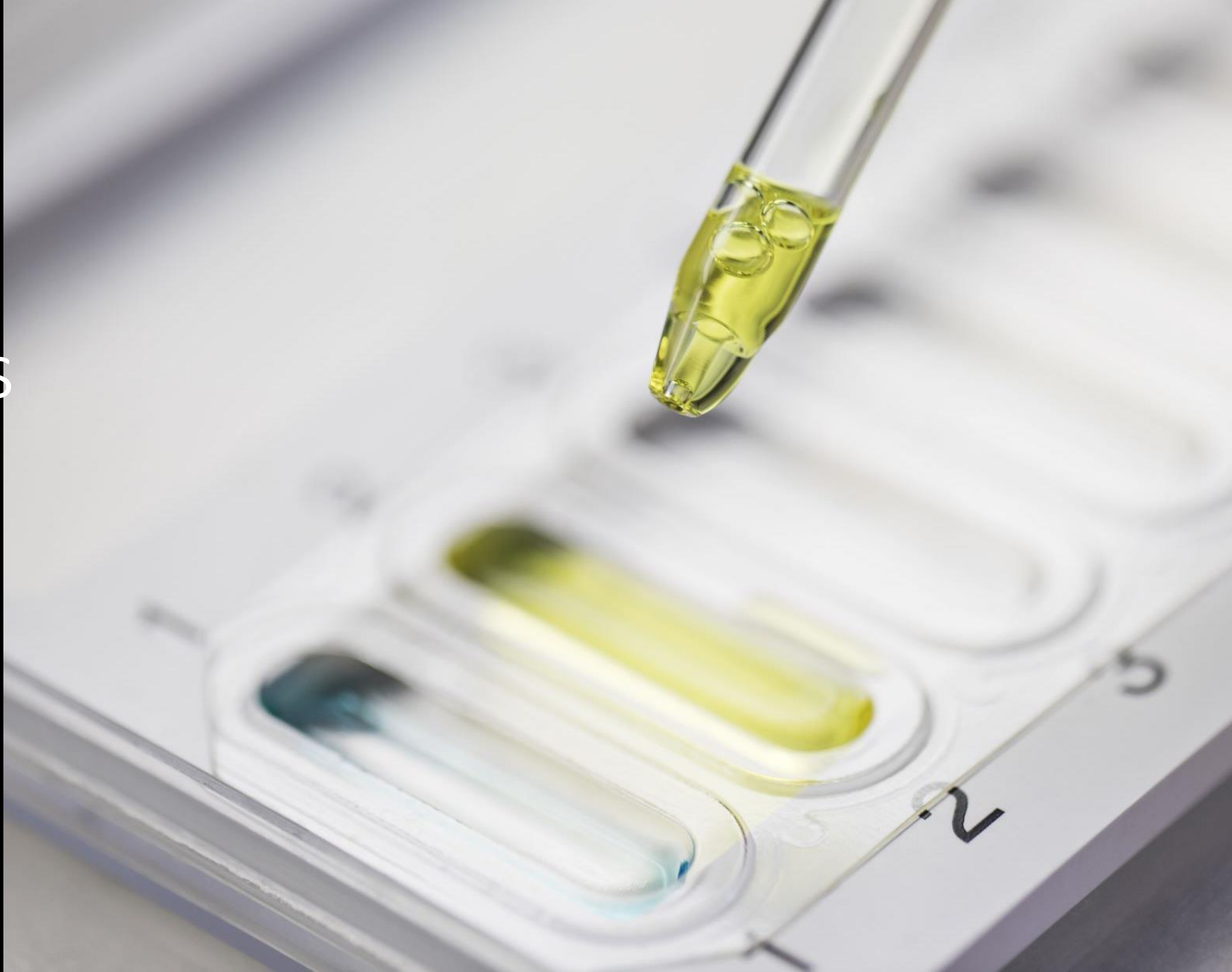
```
%%sql select "Landing_Outcome", count("Landing_Outcome") as Total from SPACEXTBL
where "Landing_Outcome" LIKE "Success%" and
Date between '04-06-2010' AND '20-03-2017' group by "Landing_Outcome" order by Total DESC;
```

✓ 0.1s

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

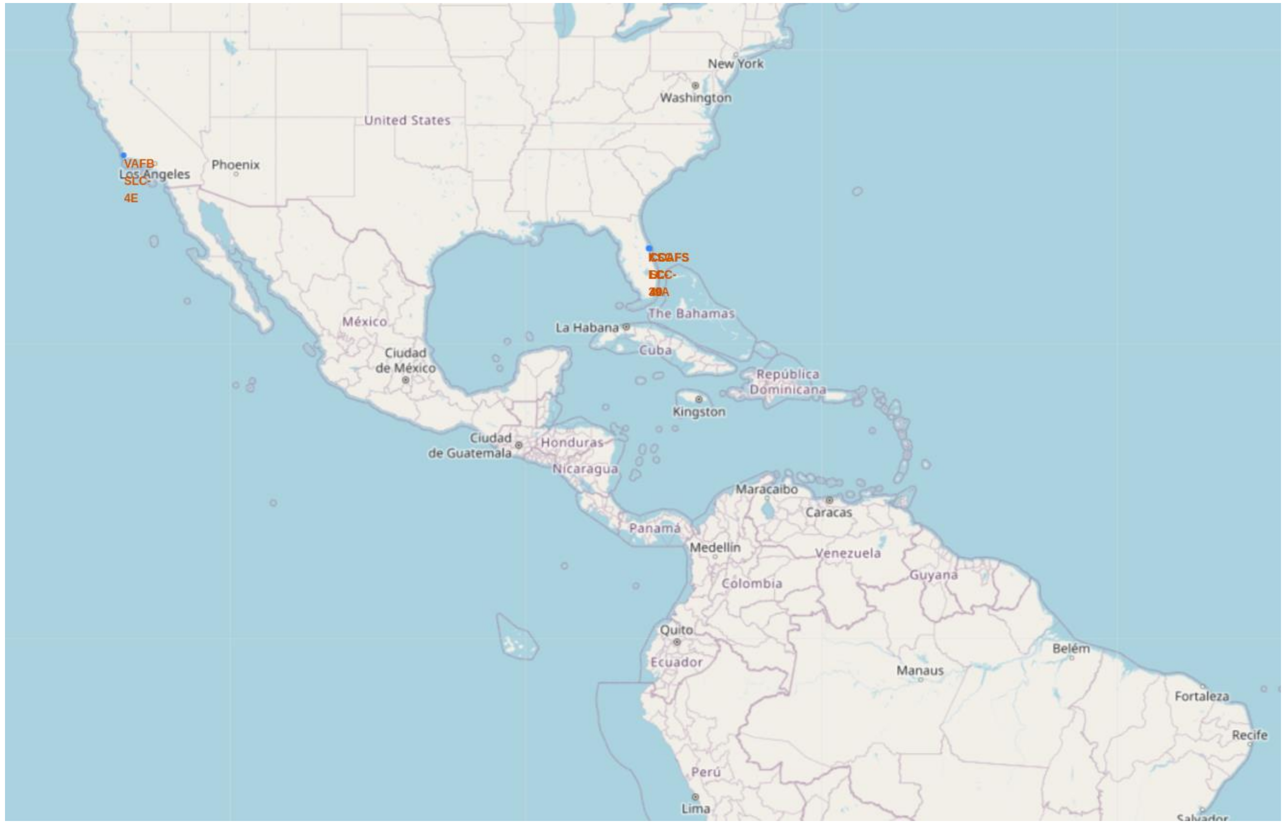
Landing_Outcome	Total
Success	20
Success (drone ship)	8
Success (ground pad)	7

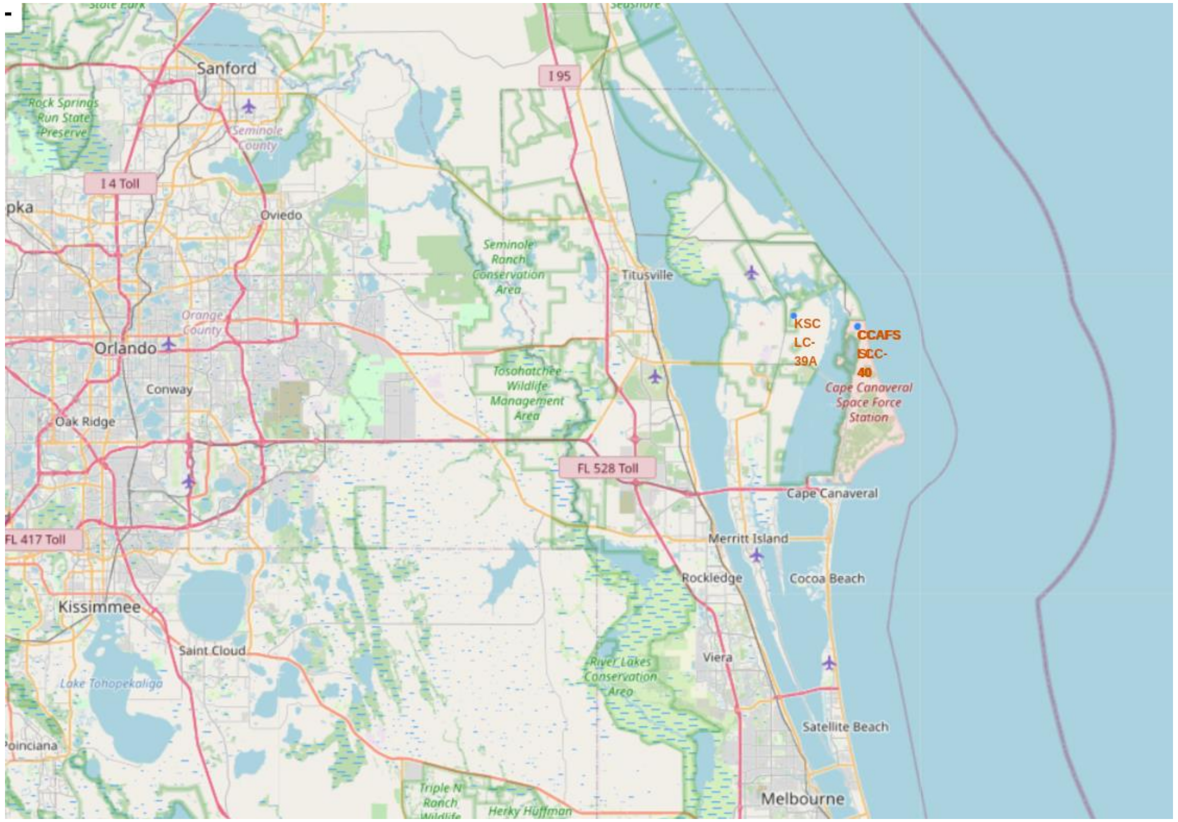
Launch Sites Proximities Analysis



ALL LAUNCH SITES LOCATIONS ON A MAP

- The left map are show all SpaceX lanuch site locate on coast of the United States of America.
- The right map are 2 Florida launch sites very close to each other



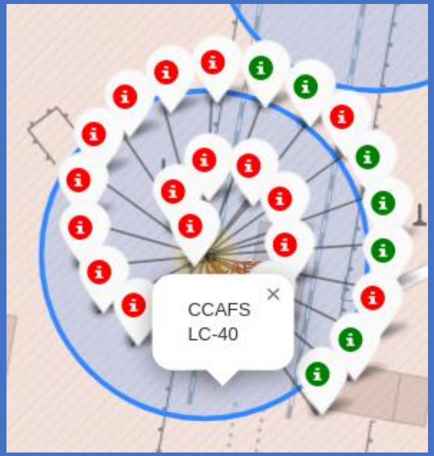


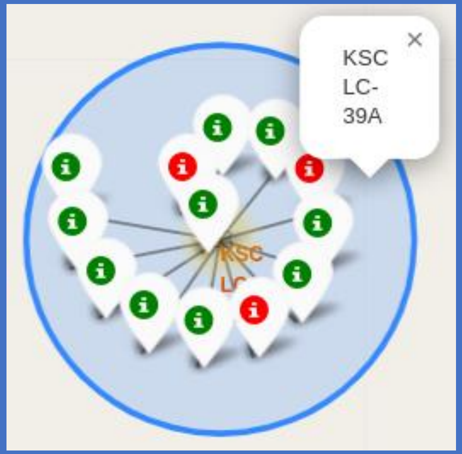
SUCCESS/FAILED LAUNCHES FOR EACH SITE (WITH COLOR LABELED)

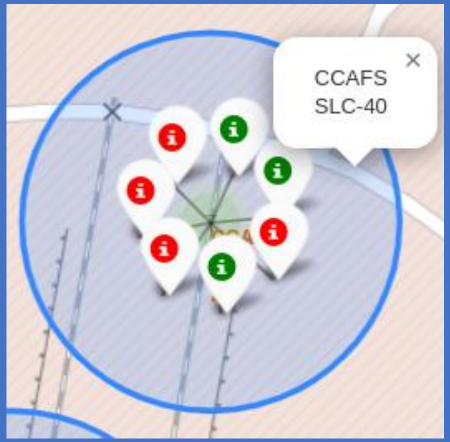
- The Lauches site are group by the clusters to label with green icon for indicate sucessful landding or red icon for failure landing

Launch site in California

Launch Site in Florida

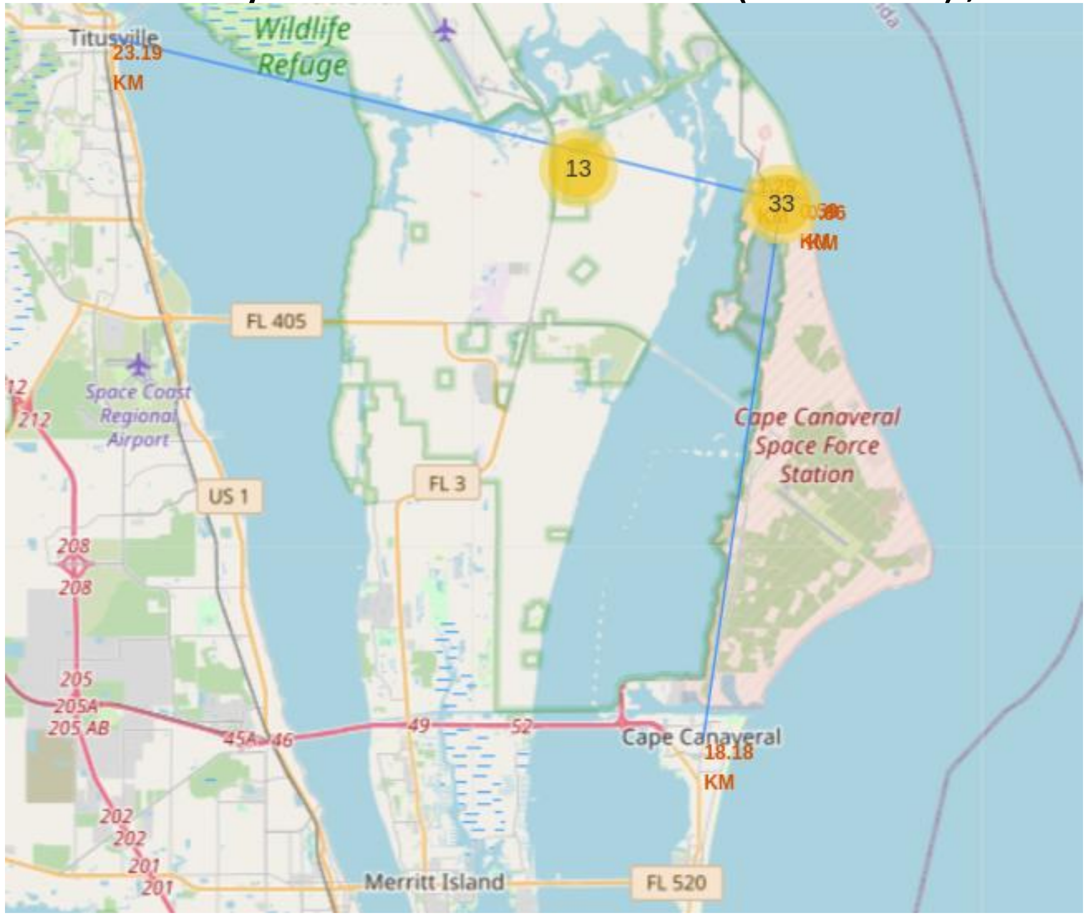


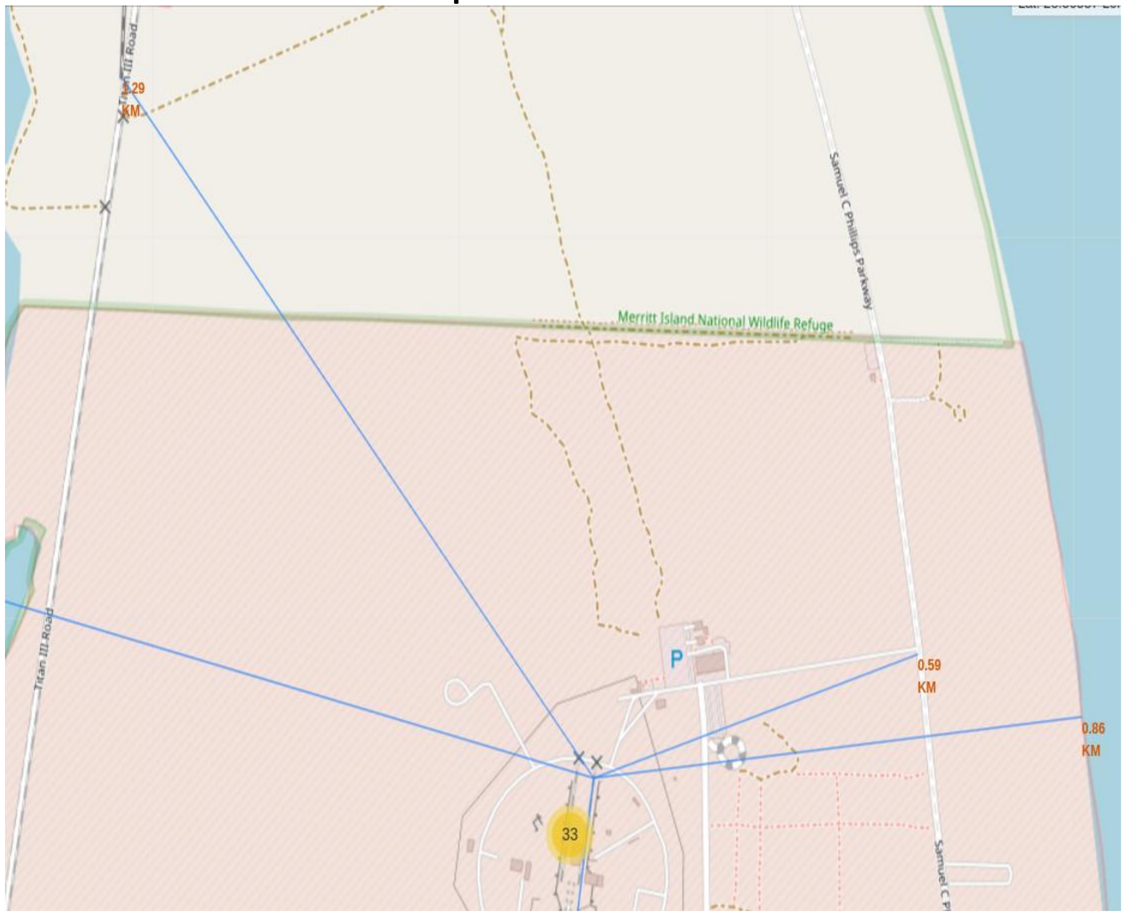




Proximity of Launch sites to other location

- Add Proximity of Launch sites marker to show the location close to railway (1.29 kg) and highway (0.59km) for good transportation of lauch equipment and material, and also close to coastline (0.86km)
- Far away from the other cities (23.19km) , the launch failure does not pose a thread

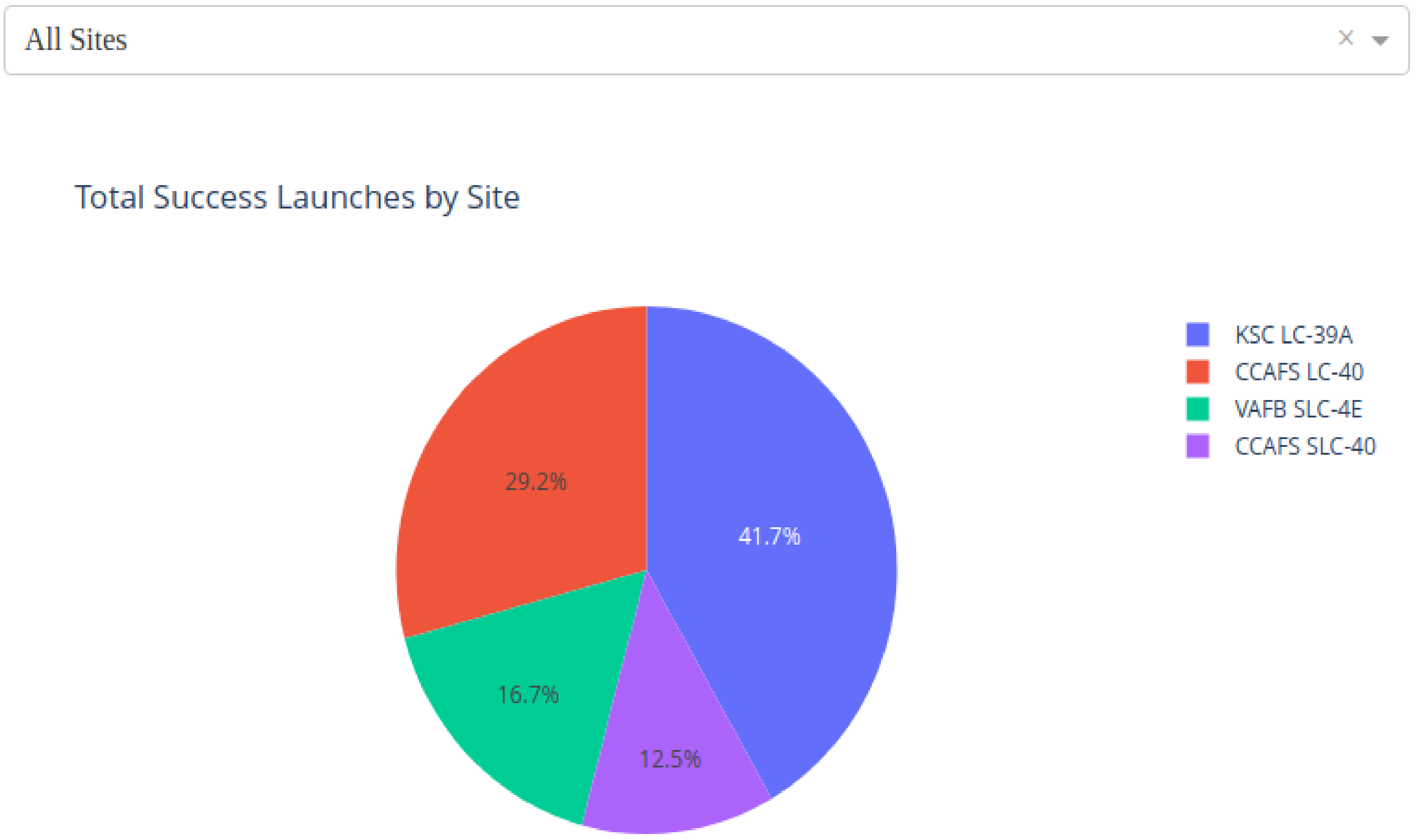




Build a Dashboard With Plotly Dash

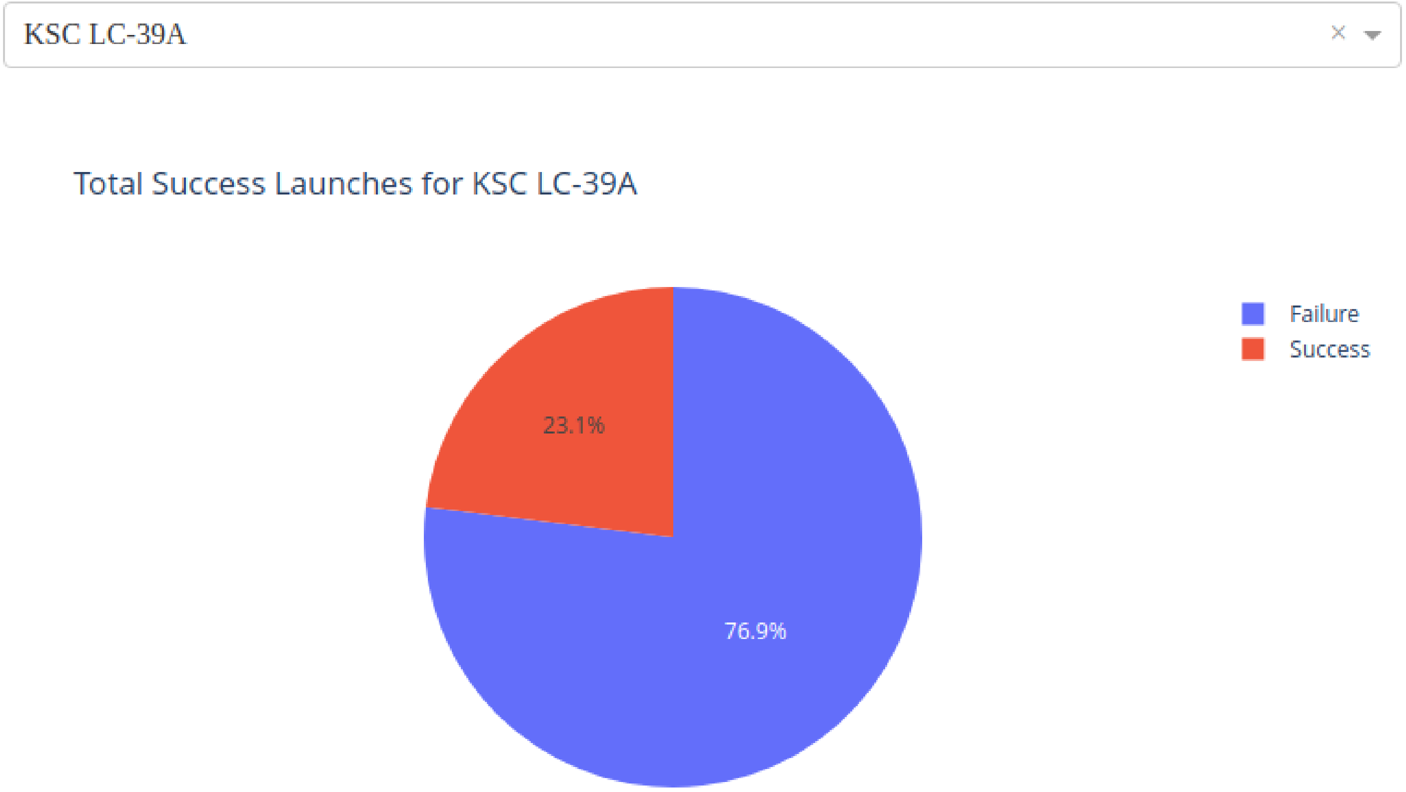
Total Success Launches by all sites

- The launch site KSC LC-39A the most successful launches with 41.7% of the total successful launches



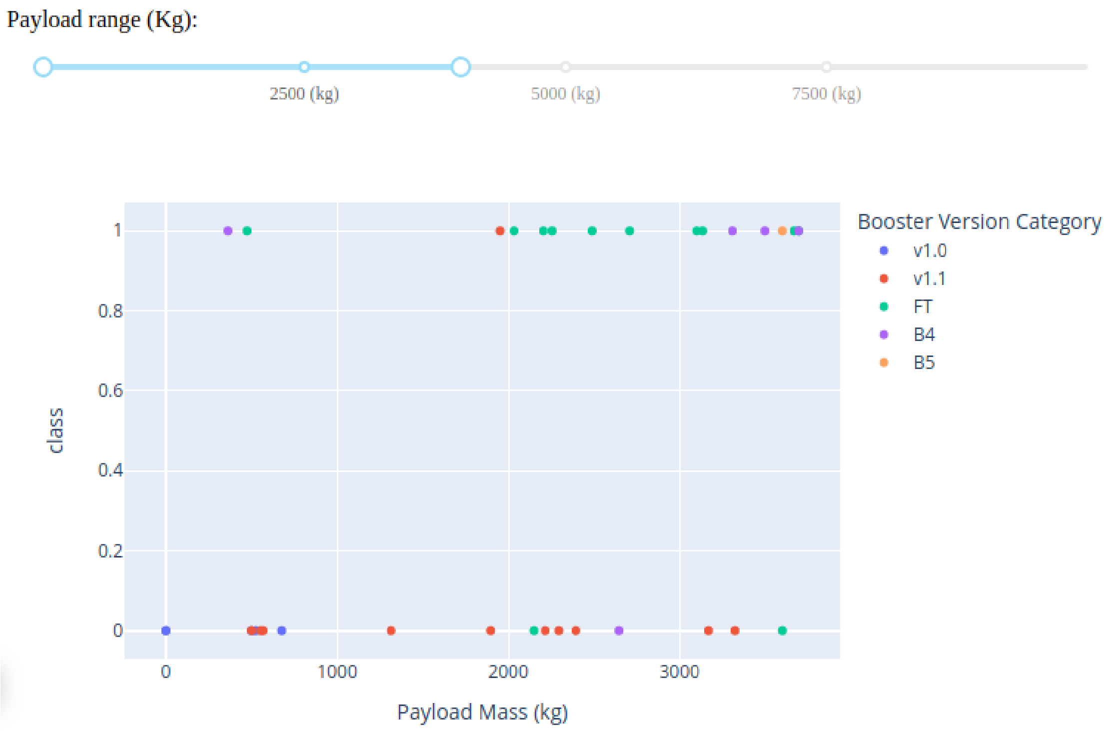
Pie Chart for Launch Stie with highest Launch Success Ratio

- The lanuch site KSC LC-39A had highest rate of successful launch rate 76.9% and 23.1% for failures



Launch Outcome vs Payload Scatter plot for all site

- Split into two payload mass (lower payload 0 to 4000kg) and (heavy payload 4000 to 10000kg) to display the class (1 or 0) success or failure
- THe Launch success rate (class 1) is higher for low weight (0-4000kg) payload than the heavery weight(4000- 10000kg) payload



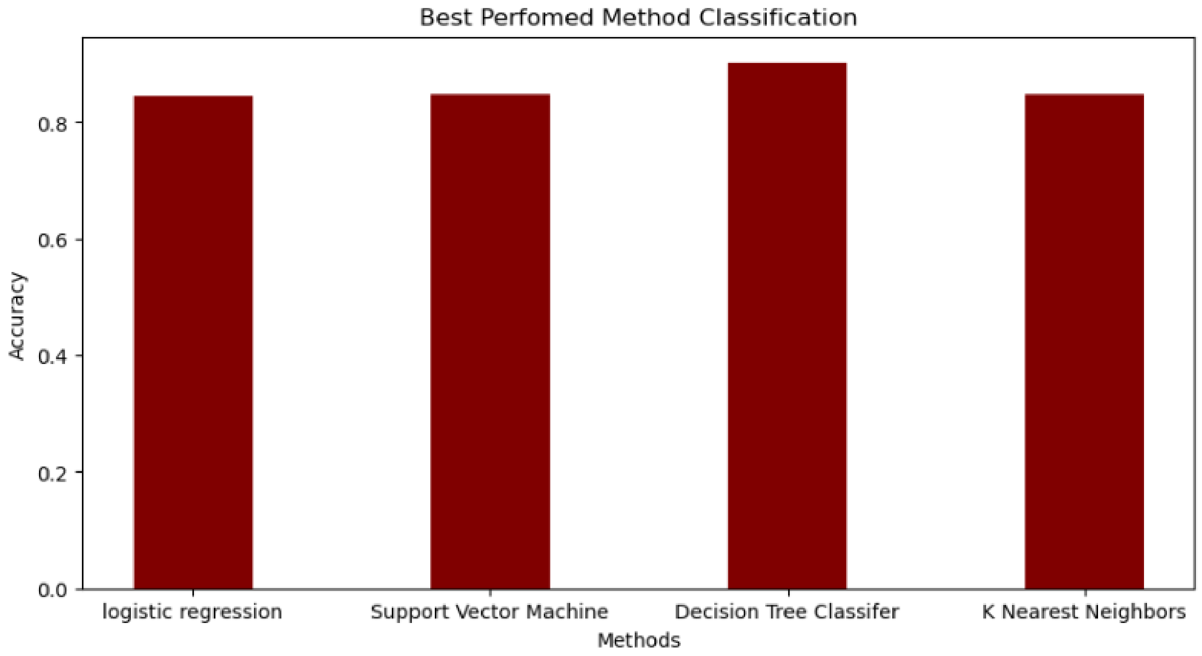
Predicative Analysis Classification

Classification Accuracy

Plotting the Accuracy Score and Best Score for each classification model algorithm

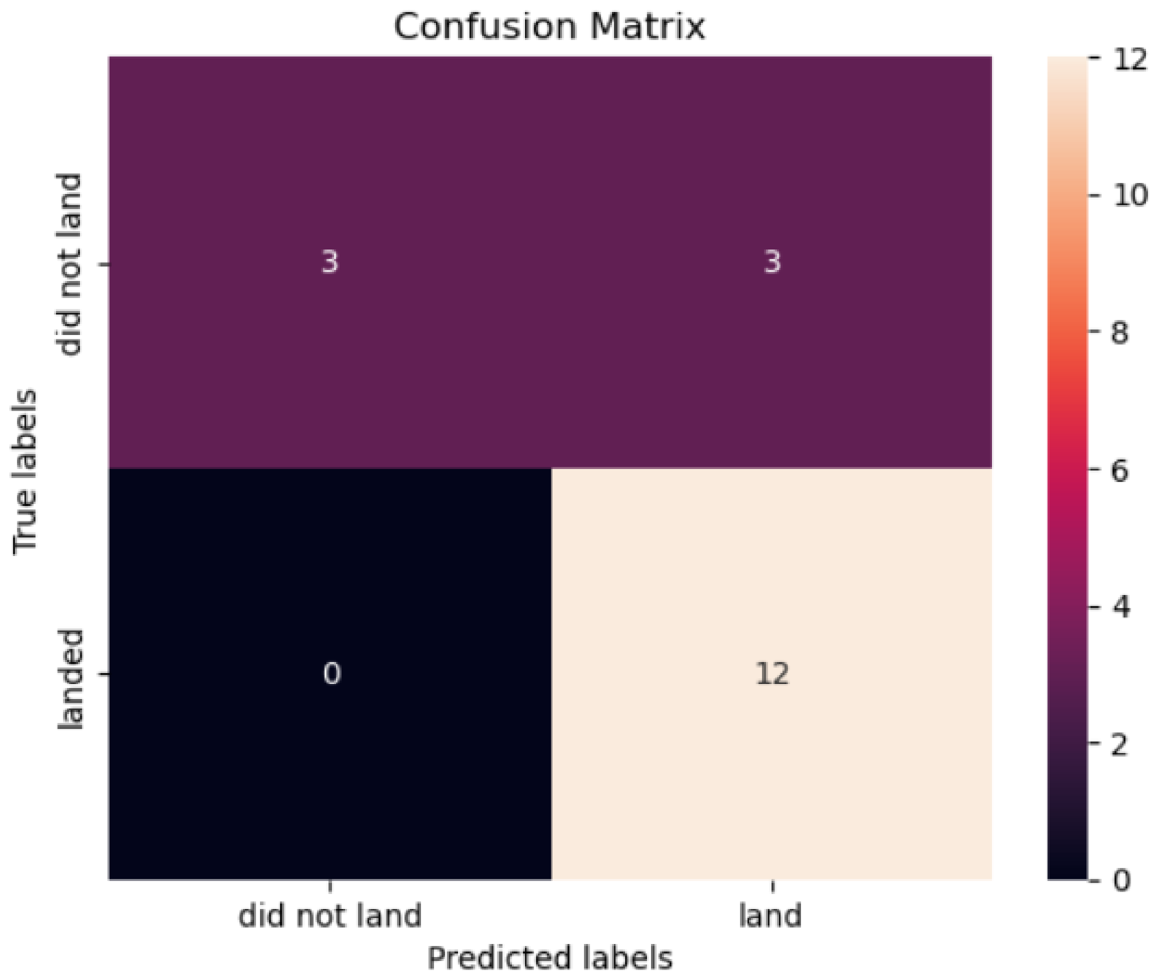
- Decision Tree model has get the highest classifciation accuarcy (Best Score : 90.17%)

	Algorithm	Accuracy Score	Best Score
0	logistic regression	0.833333	0.846429
1	Support Vector Machine	0.833333	0.848214
2	Decision Tree Classifer	0.833333	0.901786
3	K Nearest Neighbors	0.833333	0.848214



Confusion Matrix

- Confustion Matrix for all models performed the same test dataset,
- Confusion martix result for all models is same
- All model predict 12 successful landing (Actuall label = True) was successful and 3 predict for failure (actual label = False) was successful
- All model also has 3 prediction are failure, that predict the outcome successful landing (actual label = False)
- Overall , all models can prediction the launch landing sucessful



Conclusion

Conclusion

- As the number of flights increase, the successful rate also increse. At the early flight, most of outcome unsuccessful. After, obtain more experience, the success rate has increase.
- Between 2010 and 2013, all landing success rate is 0% (all failure) ,After 2013, overall trend the success rate are increasing after 2015, the success rate increaing to over 50% chance
- Orbit Type ES-L1, GEO, HEO and has obtain 100% success rate by only have 1 flight into the respective orbits type. SSO orbit type obtain 100% success rate with 5 successful flight. VLEO orbit type are relate with heavier payload mass
- The launch site KSC LC-39 A had the most successful launches, with 41.7% of the total successful lanuches , also highest rate of successful lanuches, with 76.9% scuccess rate
- The Success of heavy payloads (Over 4000kg) is lower than lighter payload (0 to 4000kg)
- The best performing classification model is the Decision Tree model with accuracy 90.17%

Apprendix

Appprendix

GitHub repository :

https://github.com/johnsonhk88/ibm-Applied-Data-Science-Capstone