IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Kamal Singla
Nov 13, 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Collect the data for analysis from SpaceX API
  - Perform some data wrangling and formatting
  - Perform exploratory data analysis using python libraries like SQL and graphs
  - Perform Interactive data analysis using Folium maps and Dash Application
  - Perform Prediction data analysis using sklearn Maching Learning Models.
- Summary of all results
  - SpaceX launch outcome have improved a lot since 2013, more successful launches are expected in the future.

Introduction

In this report, we will predict if the Falcon 9 first stage will land successfully. SpaceX 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. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

The datasets were taken from spaceX API:

api.spacexdata.com/v4/launches/past

The data was also scraped from the wiki webpages using beautifulSoap library:

https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

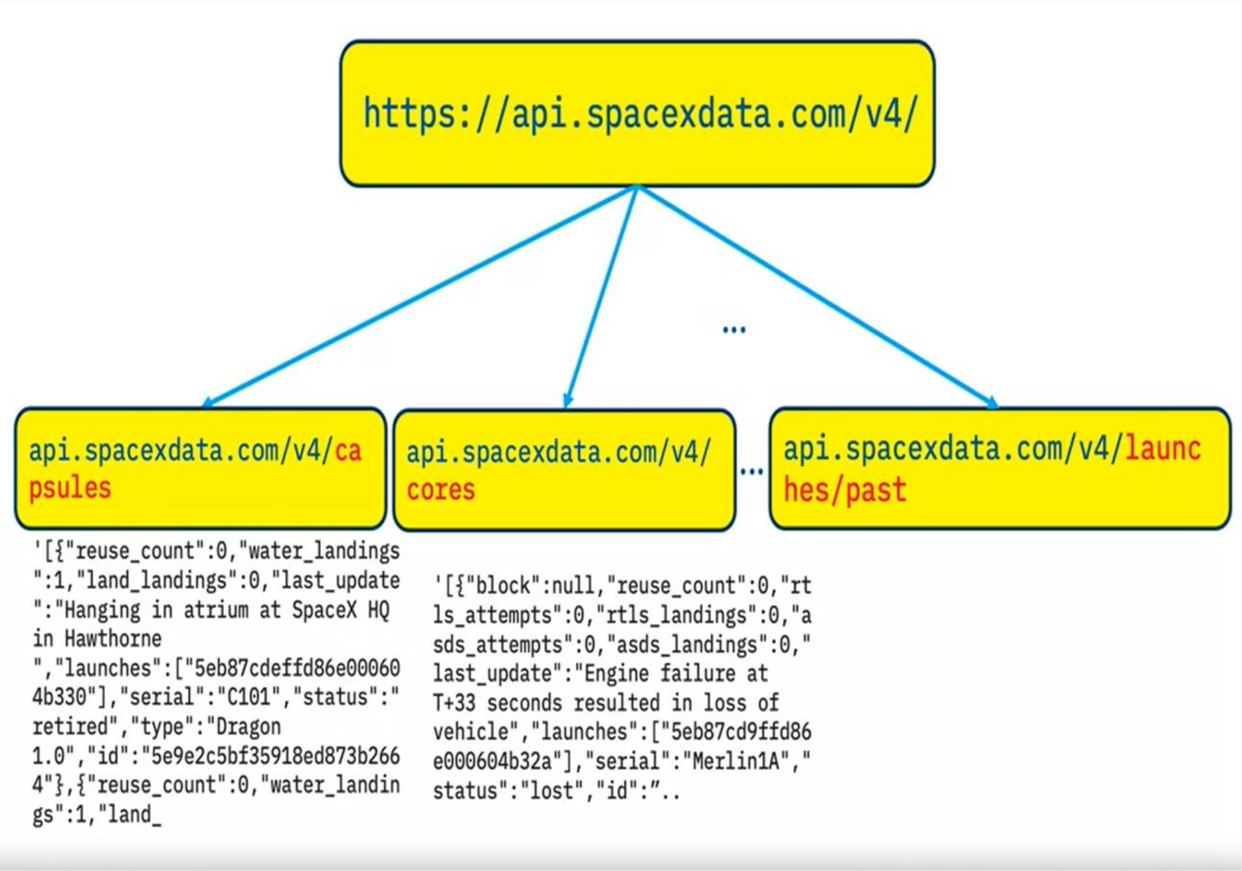
Data Collection – SpaceX API

- The SpaceX REST API endpoints, or URL, starts with **api.spacexdata.com/v4/**.

- We have the different end points, but we will be working with the endpoint **api.spacexdata.com/v4/launches/past** to get past launch data

GitHub URL of the completed SpaceX API calls notebook:

https://github.com/kamalsingla27/SpaceX-DataScience/blob/main/jupyter-labs-spacex-data-collection-api.ipynb



Data Collection - Scraping

- Used the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records
- Parse the data from those tables and convert them into a Pandas data frame for further visualization and analysis

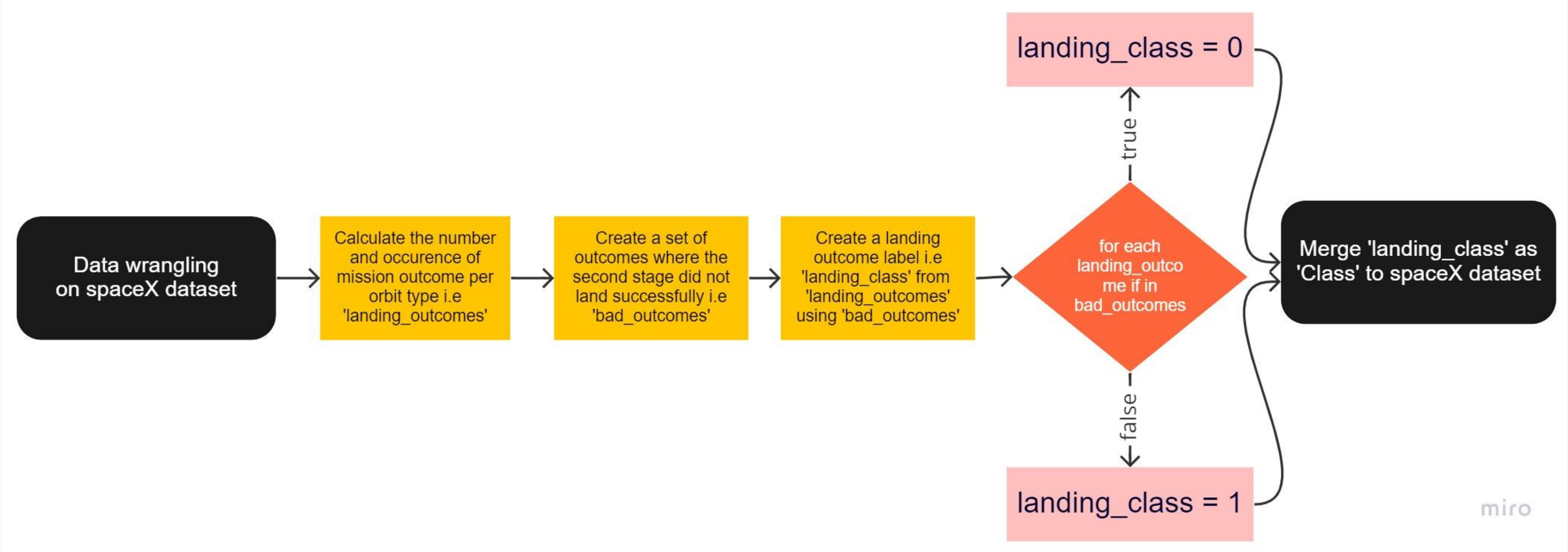
GitHub URL of the completed web scraping notebook:

https://github.com/kamalsingla27/SpaceX-DataScience/blob/main/jupyter-labs-webscraping.ipynb

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1A	167.743129	9.047721
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2A	167.743129	9.047721
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2C	167.743129	9.047721
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3C	167.743129	9.047721
4	6	2010-06-04	Falcon 9	NaN	LEO	CCAFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857

Data Wrangling

Exploratory Data Analysis (EDA) to find patterns in the data and determine the target label for training supervised models



GitHub URL of the completed data wrangling notebook:
https://github.com/kamalsingla27/SpaceX-DataScience/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Visualized the relationship between Flight Number and Launch Site
  - *Plotted 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*
- Visualized the relationship between Payload and Launch Site
  - *Plotted 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*
- Visualized the relationship between success rate of each orbit type
  - *Created a bar chart for the success rate of each orbit*
- *Visualized the relationship between FlightNumber and Orbit type*
  - *Plotted a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value*

EDA with Data Visualization (Continued)

- Visualized the relationship between Payload and Orbit type
  - *Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value*
- Visualize the launch success yearly trend
  - *Plot a line chart with x axis to be the extracted year and y axis to be the success rate*

GitHub URL of your completed EDA with data visualization notebook:

https://github.com/kamalsingla27/SpaceX-DataScience/blob/main/module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

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

```
select distinct(launch_site) from SpaceX
```

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

```
Select * from SpaceX where launch_site like 'CCA%' LIMIT 5
```

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

```
select sum(payload_mass__kg_) "Total Payload Mass"
       from SpaceX where customer = 'NASA (CRS)'
```

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

```
select avg(payload_mass__kg_) "Average Payload Mass"
       from spaceX where booster_version = 'F9 v1.1'
```

- List the date when the first successful landing outcome in ground pad was acheived

```
Select date, landing__outcome from spaceX
       where LCASE(landing__outcome) like '%success%ground pad%' order by date asc LIMIT 1
```

EDA with SQL (Continued)

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

```
Select booster_version, landing__outcome, payload_mass__kg_ from spaceX
        where LCASE(landing__outcome) like '%success%drone ship%'
        and (payload_mass__kg_ BETWEEN 4000 and 6000)
```

- List the total number of successful and failure mission outcomes

```
Select mission_outcome, count(mission_outcome) "Total"
        from spaceX group by mission_outcome
```

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

```
Select booster_version from spaceX
        where payload_mass__kg_ in (select max(payload_mass__kg_) from spaceX)
```

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

```
Select booster_version, launch_site, landing__outcome from spaceX
        where LCASE(landing__outcome) like '%failure%drone%ship%' and YEAR(date) = 2015
```

EDA with SQL (Continued)

- 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
-
- **Select** landing__outcome, **count**(landing__outcome) "Count"
-     **from** spaceX
-     **where** landing__outcome
-     **in** ('Failure (drone ship)', 'Success (ground pad)')
-     **and** (date **between** '2010-06-04' **and** '2017-03-20')
-     **group by** landing__outcome
-     **order by** "Count" **desc**

GitHub URL of your completed EDA with SQL notebook:

https://github.com/kamalsingla27/SpaceX-DataScience/blob/main/jupyter-labs-eda-sql-coursera.ipynb

Build an Interactive Map with Folium

Explain why you added those objects

- folium.map
  Interactive map with Start Location as NASA Johnson Space Center coordinates

- folium.Circle
  Created a blue circle at NASA Johnson Space Center's coordinate. Added as a child to folium.map's object

- folium.Popup
  Popup label showing name. Added as a child to folium.Circle's object

- Folium.map.Marker
  Created a blue circle at NASA Johnson Space Center's coordinate with an icon showing its name. Added as a child to folium.map's object

- DivIcon
  Created an icon as a text label to show the landing site name. Added as icon property to 'Folium.map.Marker' object

- `MarkerCluster()`
  Created a `MarkerCluster` object. Added as child to folium.map's object to store different landing result markers. Userful to show multiple markers with same coordinates.

Build an Interactive Map with Folium (Continued)

- folium.Marker
  Created a Marker object with its coordinate. Show markers for each landing site landing outcome I.e. Either failed or successful. Added as a child to `MarkerCluster` object.
- Folium.Icon
  Marker's icon property to set icon color. Added a an icon property in 'folium.marker' like show green for successful landing and red for failed landing.
- MousePosition
  Mouse Position to get the coordinate (Lat, Long) for a mouse over on the map. Added as child to folium.map's object
- Folium.Polyline
  Create a `folium.PolyLine` object using the coastline coordinates and launch site coordinate to draw a line between two points. Added as child to folium map's object
  e.g. Draw a line between a landing site and nearest places like highway, railway etc.

GitHub URL of your completed interactive map with Folium map:

https://github.com/kamalsingla27/SpaceX-DataScience/blob/main/module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
- Added a Launch Site Drop-down Input Component
  - Added the dash html component 'dcc.Dropdown' to allow the user to select any one of the available launch site for spaceX including the option to Select "All Sites"
- Added a callback function to render `'success-pie-chart'` based on selected site dropdown
  - Based upon the drop-down selection, rendered a pie-chart showing the total success launches launch of all sites or for selected site.
- Add a Range Slider to Select Payload
  - Added a dash html component ''dcc.RangeSlider' to allow the user to select payload between min and max range.
- Add a callback function to render the `'success-payload-scatter-chart'` scatter plot
  - Based on the drop-down selection and range-slider selection, rendered a scatter plot to visually observe how payload may be correlated with mission outcomes for selected site

Build a Dashboard with Plotly Dash (Continued)

- Rendered Pie Chart
  - If ALL sites are selected, return a pie chart graph to show the total success launches (i.e., the total count of `class` column)
  - If a specific launch site is selected, then, render and return a pie chart graph to show the success (`class=1`) count and failed (`class=0`) count for the selected site.
- Rendered Scatter Plot
  - Plotted a scatter plot with the x axis to be the payload and the y axis to be the launch outcome (i.e., `class` column). As such, we can visually observe how payload may be correlated with mission outcomes for selected site(s).
  - In addition, we want to color-label the Booster version on each scatter point so that we may observe mission outcomes with different boosters.

GitHub URL of your completed Plotly Dash lab:

https://github.com/kamalsingla27/SpaceX-DataScience/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- Defined/Created a Target/Dependent Column from the dataset
- Standardized the data using 'sklearn preprocessing'
- Split dataset into training data and test data using sklearn 'train_test_split' i.e. X_train, X_test, Y_train, Y_test
- Defined various models like logical Regression, SVM, KNN, Decision Tree.
- For each Model, trained the model using the training data(X_train and Y_train)
- Find best Hyperparameters for each model i.e. best_params_
- Find the accuracy of each model i.e best_score_
- Get Predicted Results (yhat) from each trained Model by passing the test data (X_test)
- Plotted the Confusion Matrix using Predicted Results (yhat) and actual results(Y_test)
- Plotted a bar graph showing accuracy for each model to graphically present the Best performing Model using seaborn.
- Also analyzed the Confusion Matrix to differentiate Best Performing model to other models.

GitHub URL of your completed predictive analysis lab:

https://github.com/kamalsingla27/SpaceX-DataScience/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

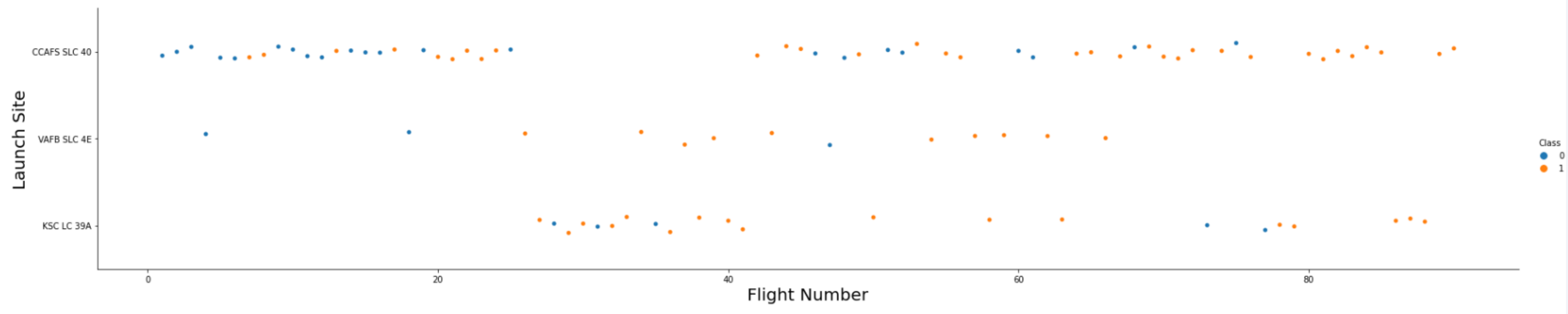
Results

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

Section 2

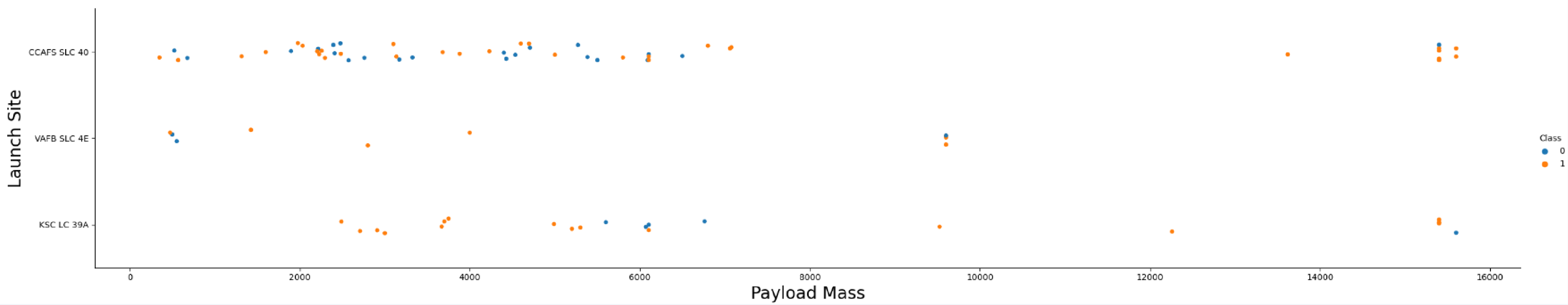
Insights drawn from EDA

Flight Number vs. Launch Site



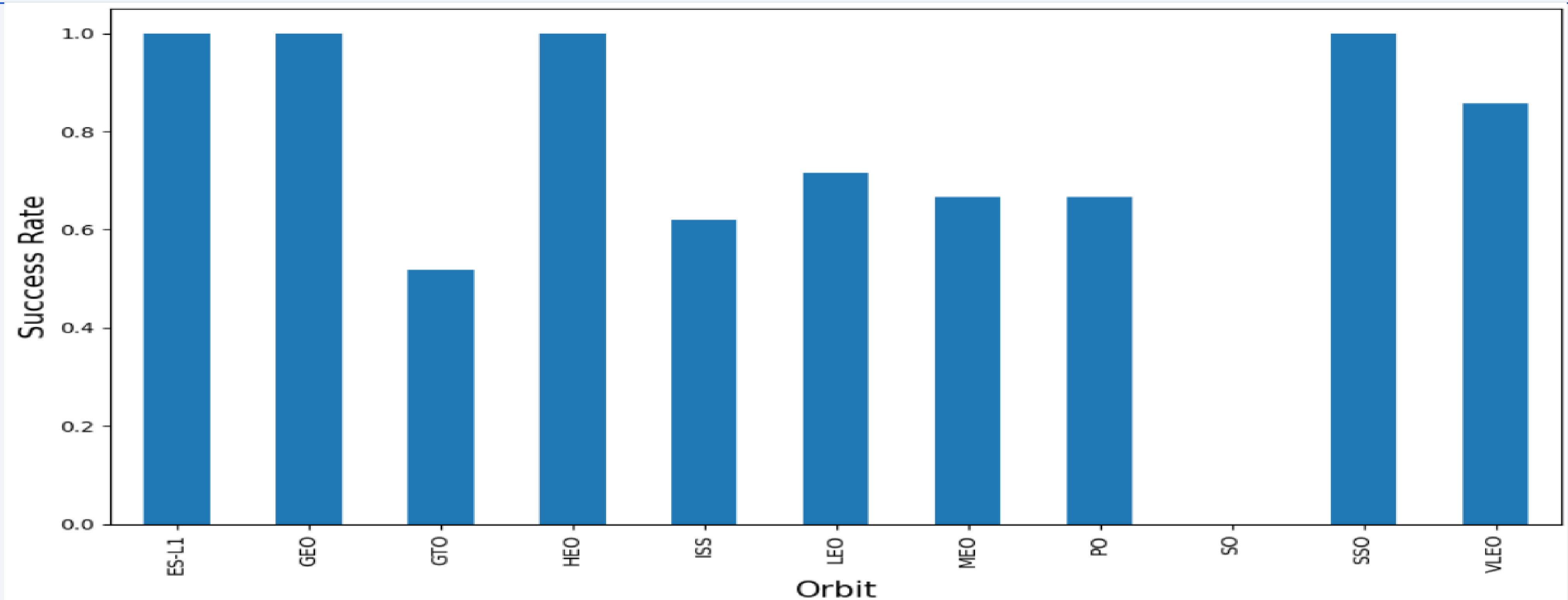
The scatter plot above shows the relationship between Launch Site on Y axis and Flight number on X axis. This graph shows that with the increase in flight number, the success rate is increasing for all Launch Sites. Which means as more and more flights are tested, success rate for all the launch sites are also getting better.

Payload vs. Launch Site



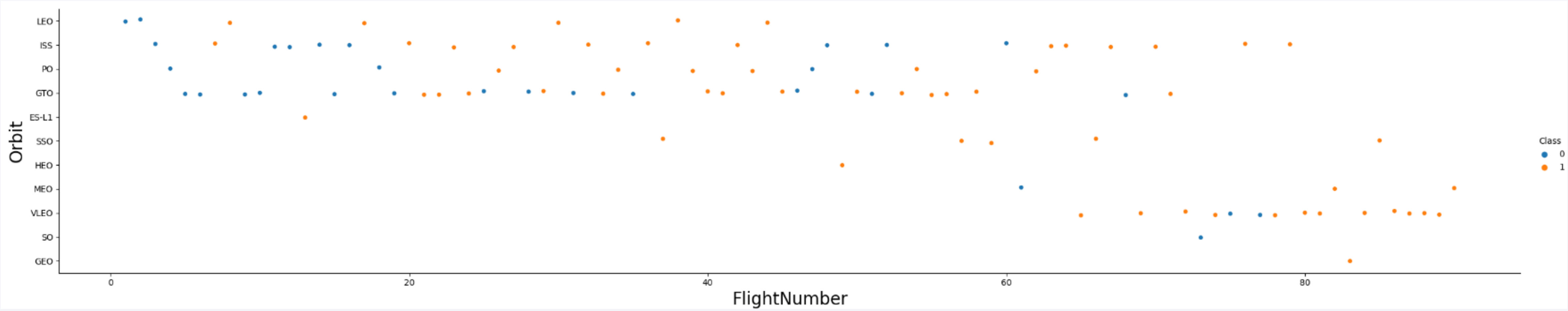
- In the Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).
- Also heavier payloads seems to result in more successful outcome.

Success Rate vs. Orbit Type



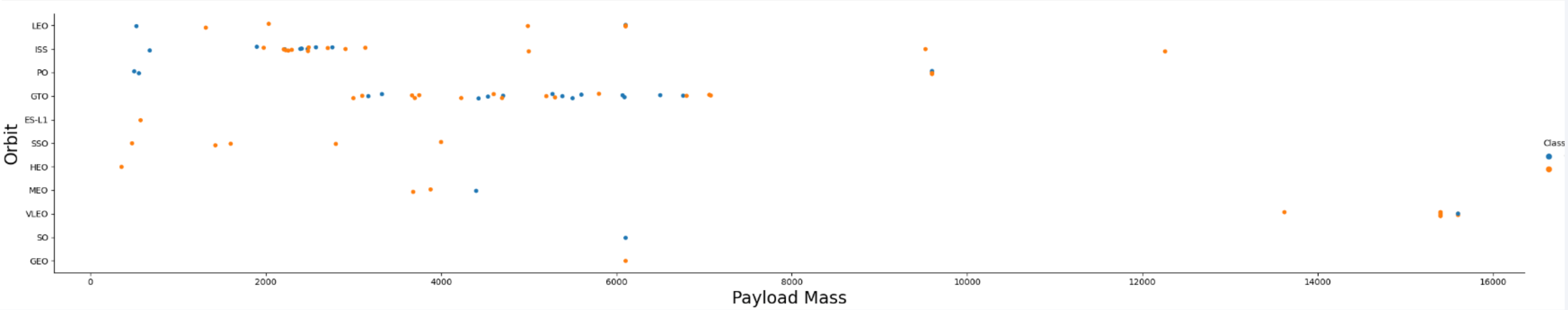
- Bar Chart shows the success rate of each orbit
- ES-L1, GEO, HEO, SSO, VLEO are the orbits with maximum success rate
- Whereas 'SO' seems to have the least to no success

Flight Number vs. Orbit Type



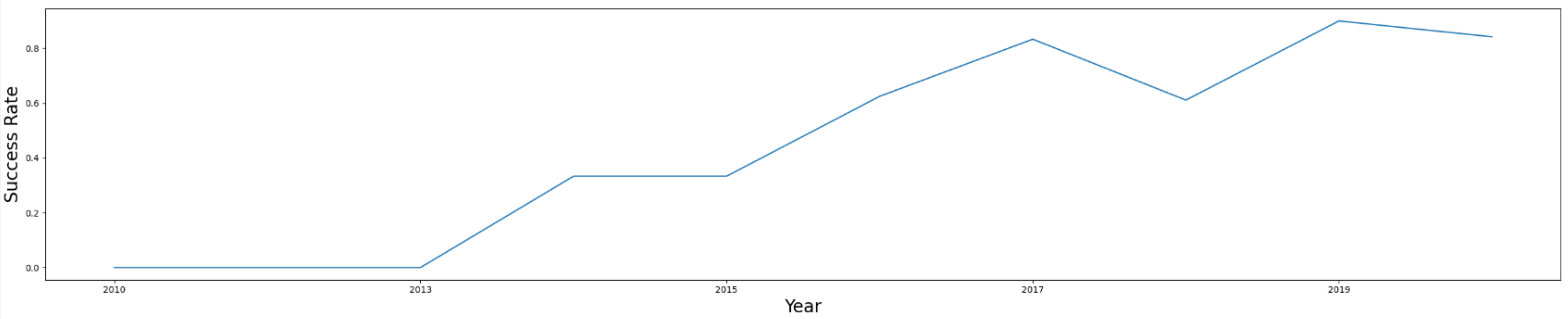
- Scatter point of Flight number vs. Orbit type
- LEO orbit the Success appears related to the number of flights
- But there seems to be no relationship between flight number when in GTO orbit
- In General, Orbit and Flight number doesn't seems to be much related.

Payload vs. Orbit Type



- Scatter point of payload vs. orbit type
- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend



- Line chart of yearly average success rate
- Success Rate is clearly increasing with the increase in years from 2013 to 2020

All Launch Site Names

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

```
%sql select distinct(launch_site) from SpaceX
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

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

- There are 4 distinct launch sites as per the SpaceX Falcon 9 dataset

Launch Site Names Begin with 'CCA'

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

```
%sql Select * from SpaceX where launch_site like 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

- 5 records where launch sites begin with `CCA' from the spaceX Falcon 9 dataset

Total Payload Mass

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

```
%sql select sum(payload_mass__kg_) "Total Payload Mass" from SpaceX where customer = 'NASA (CRS)'
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

Total Payload Mass
45596

- Total payload carried by boosters from NASA (CRS) is 45596

Average Payload Mass by F9 v1.1

Task 4

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

```
%sql select avg(payload_mass__kg_) "Average Payload Mass" from spaceX where booster_version = 'F9 v1.1'
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

Average Payload Mass
2928

- Average payload mass carried by booster version F9 v1.1 is 2928

First Successful Ground Landing Date

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

Hint:Use min function

```
%sql Select date, landing__outcome from spaceX where LCASE(landing__outcome) like '%success%ground pad%' order by date asc LIMIT 1
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

DATE	landing__outcome
2015-12-22	Success (ground pad)

- Dates of the first successful landing outcome on ground pad is 2015-12-22
- We can also achieve this result using min or other ways.

Successful Drone Ship Landing with Payload between 4000 and 6000

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, landing__outcome, payload_mass__kg_ from spaceX where LCASE(landing__outcome) like '%success%drone ship%' and (payload_mass__kg_ BETWEEN 4000 a
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

booster_version	landing__outcome	payload_mass__kg_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

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

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

```
%sql Select mission_outcome, count(mission_outcome) "Total" from spaceX group by mission_outcome
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

mission_outcome	Total
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- Total number of successful and failure mission outcomes
- 1 of the success outcome have 'payload status unclear'
- Overall. There is 1 Failure and 100 Success outcomes

Boosters Carried Maximum Payload

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

```
%%sql

Select booster_version
    from spaceX
    where payload_mass__kg_
    in (select max(payload_mass__kg_)
        from spaceX)
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
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

- List of the names of the booster_versions those carried max payload.

2015 Launch Records

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

```
%%sql

Select booster_version, launch_site, landing__outcome
    from spaceX
    where LCASE(landing__outcome) like '%failure%drone%ship%' and YEAR(date) = 2015
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

booster_version	launch_site	landing__outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

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

```
%%sql

Select landing__outcome, count(landing__outcome) "Count"
    from spaceX
    where landing__outcome
    in ('Failure (drone ship)', 'Success (ground pad)')
    and (date between '2010-06-04' and '2017-03-20')
    group by landing__outcome
    order by "Count" desc
```

```
 * ibm_db_sa://fff06708:***@6667d8e9-9d4d-4ccb-ba32-21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:30376/BLUDB
Done.
```

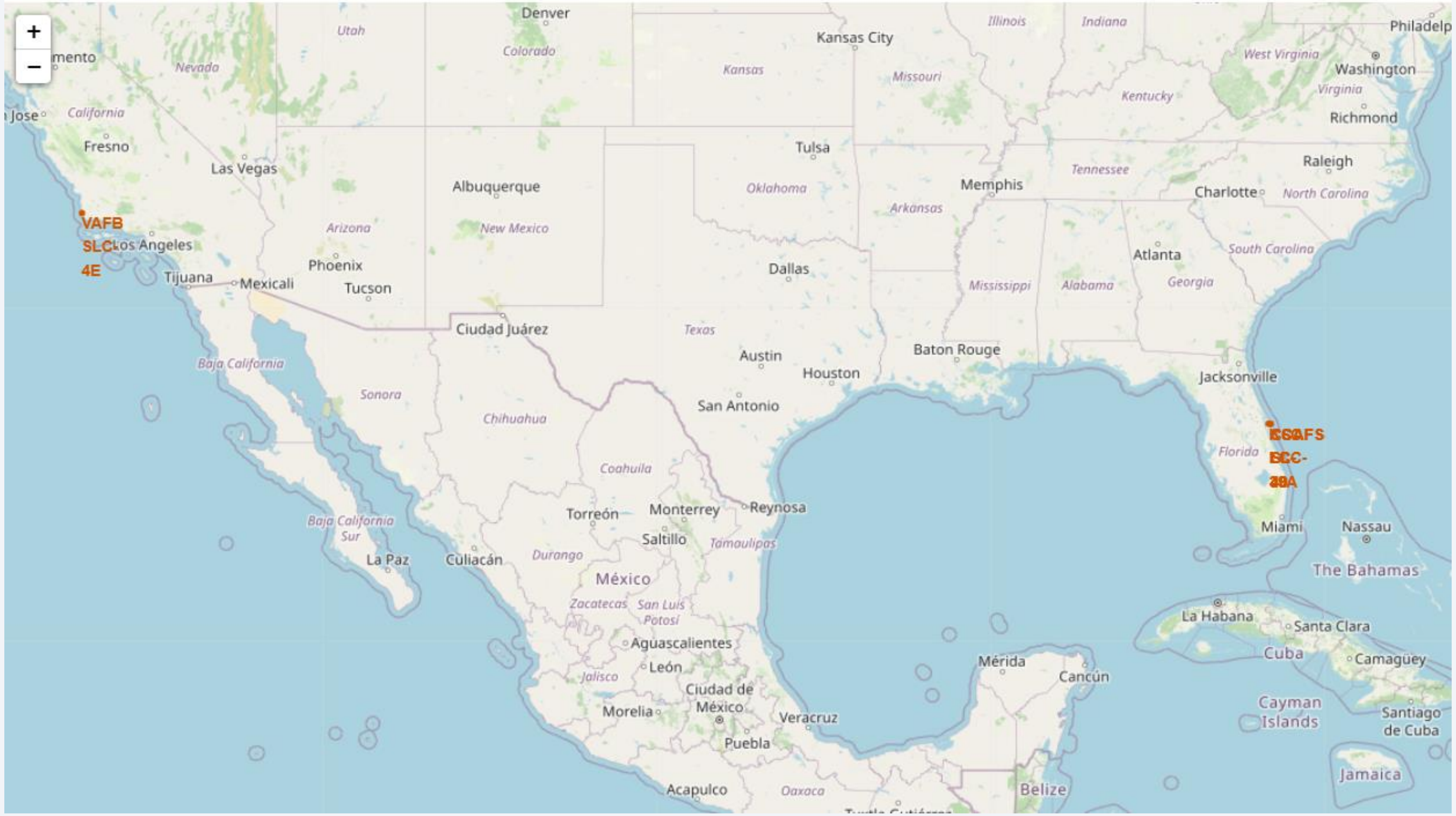
landing__outcome	Count
Failure (drone ship)	5
Success (ground pad)	3

- Ranking 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

Section 3

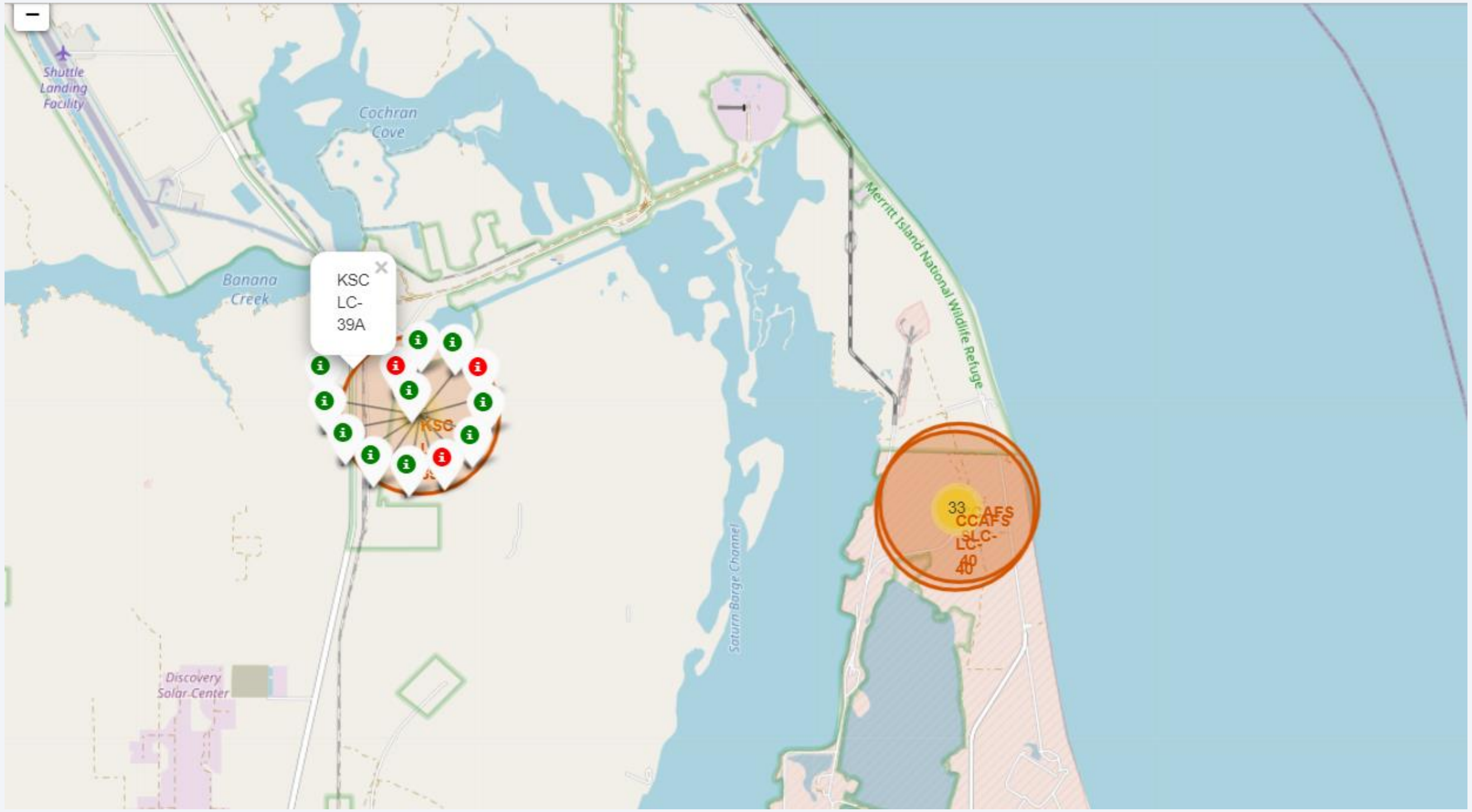
Launch Sites Proximities Analysis

Launch Sites on Folium Map



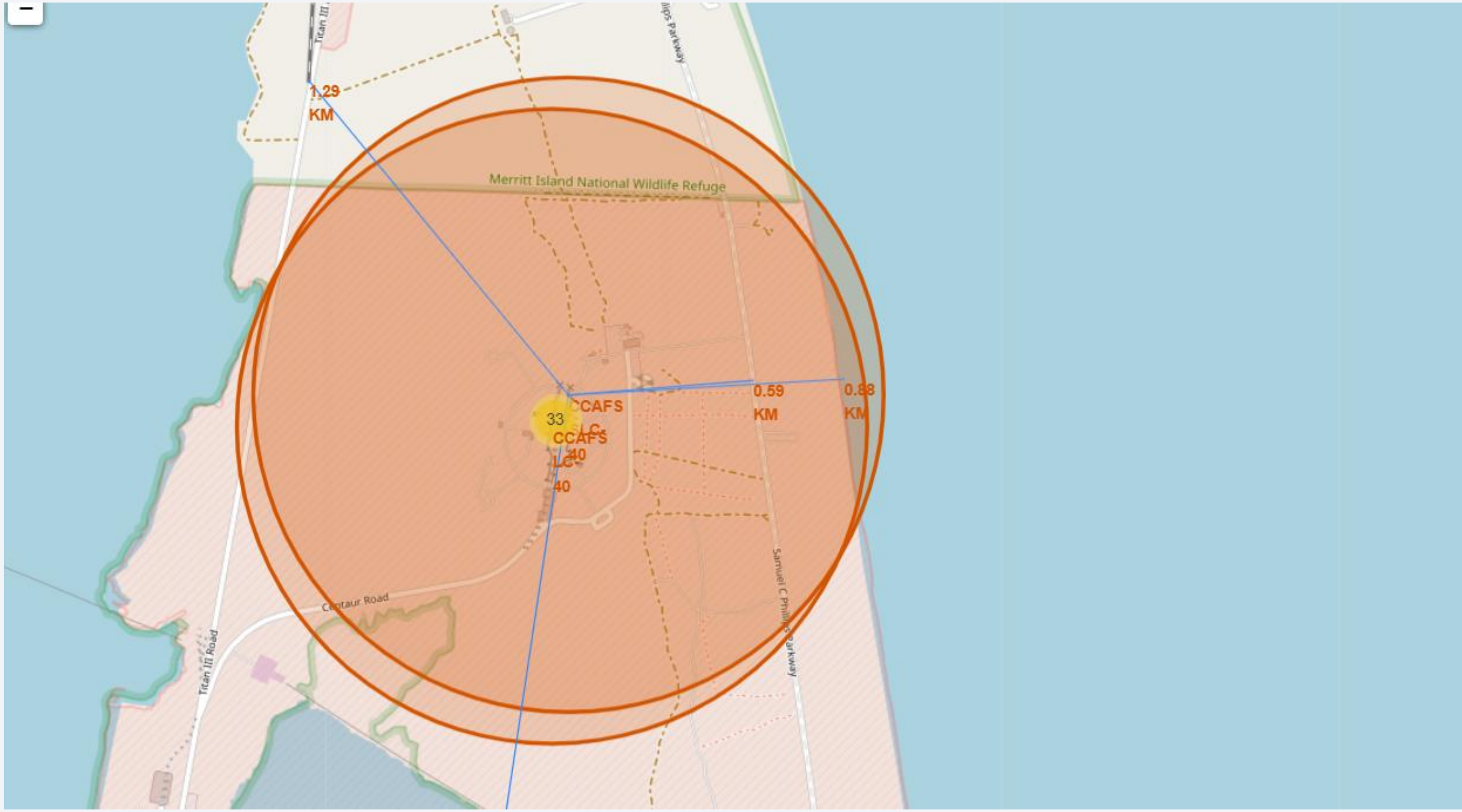
- All the launch sites are close proximity to both the equator and coast lines

Launch Outcomes for Sites



- Showing the color-labelled launch outcomes for site 'KSC LC-39A'
- Red means Failed and Green means Successful launches.

Launch Site Proximities

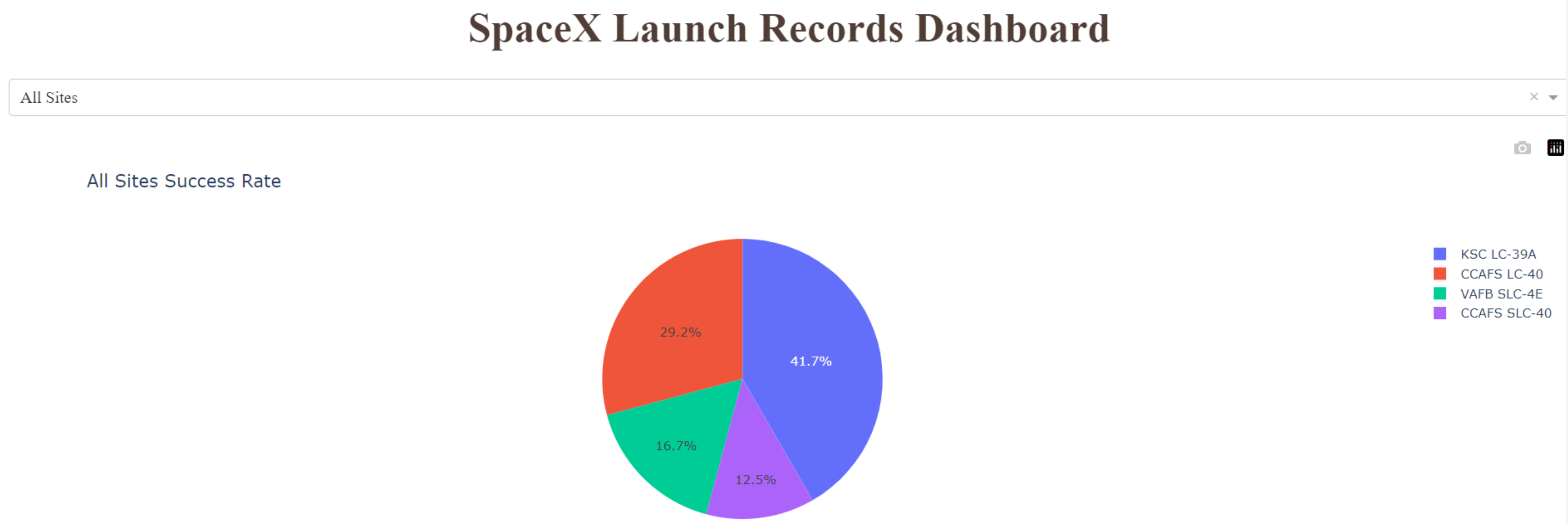


- Map shows a blue line between launch site and nearest proximities like railway, Coastline, highway and city, Whereas the 'KM' marker shows the distance in km between Launch site and proximities.

Section 4

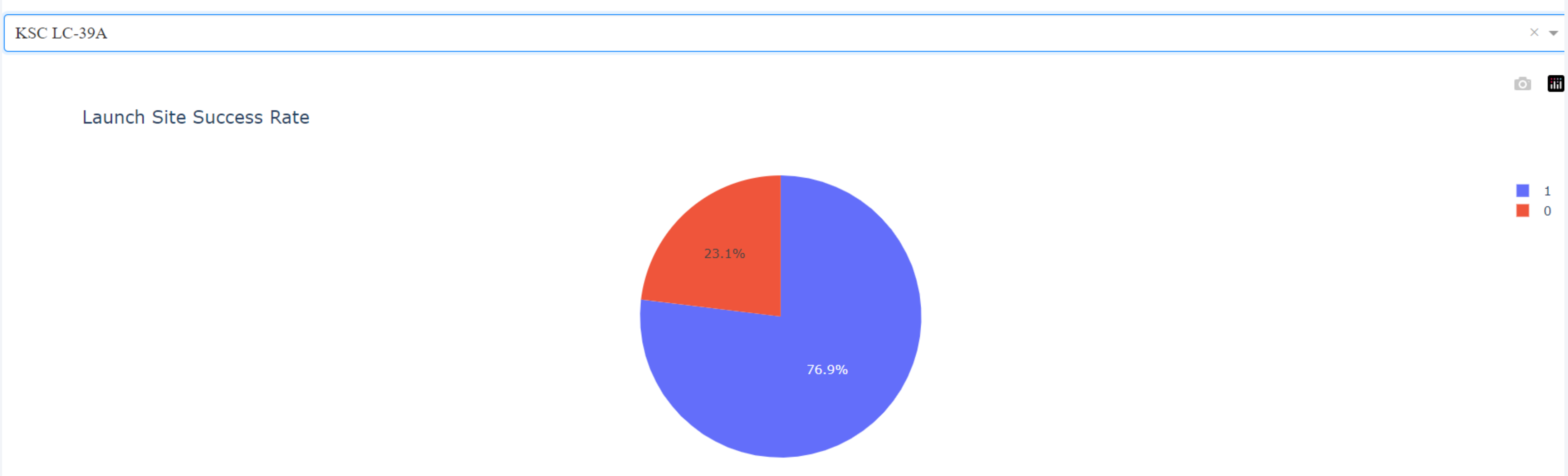
Build a Dashboard with Plotly Dash

Launch Success Rate for Sites



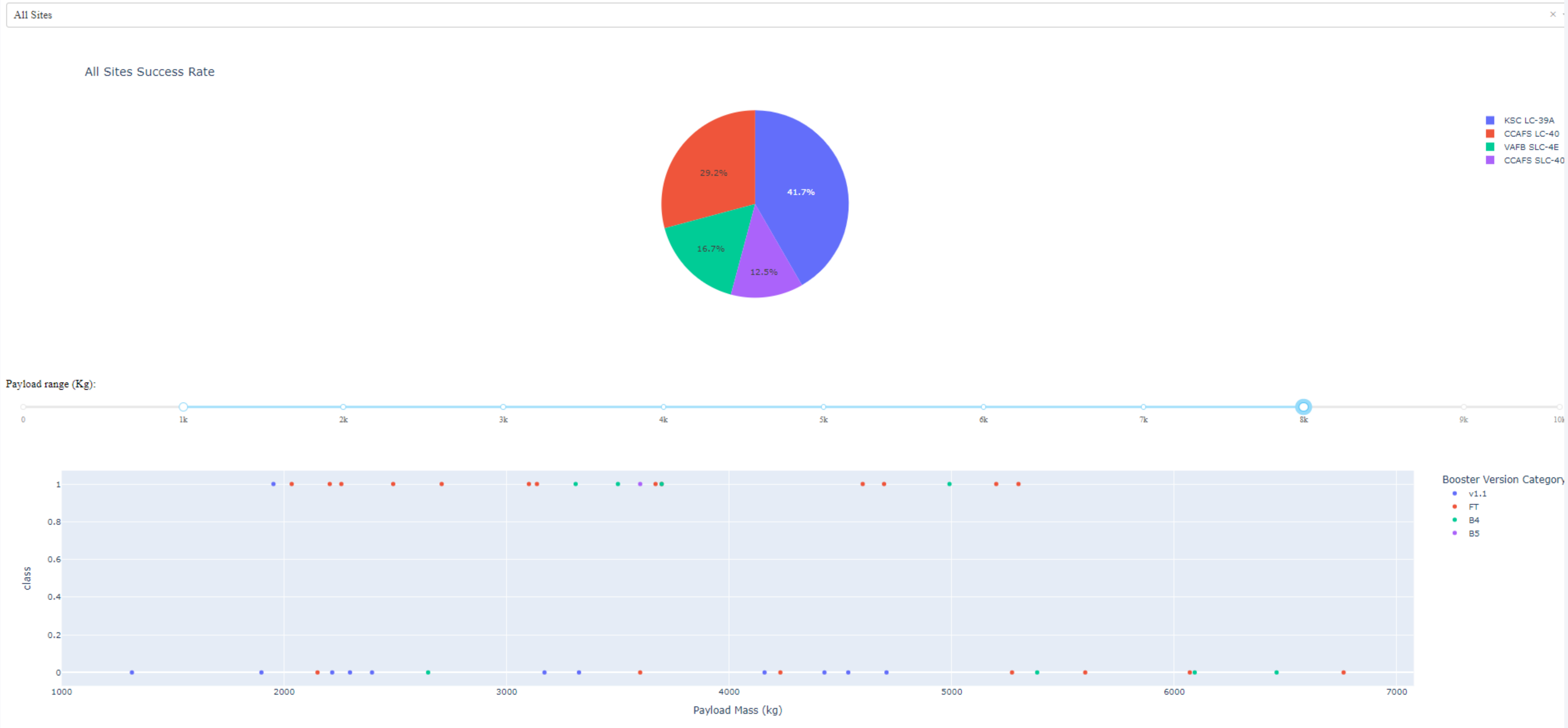
- KSC LC-39A has the highest launch success rate
- Whereas CCAFS SLC-40 has the least success rate

Launch Site With Highest Success Rate Ratio



- KSC LC-39A Launch Site, which has the highest Success Rate, has the success rate ratio of 76.9% and failed 23.1%

Payload Vs Launch Outcome Scatter Plot for All Sites



Payload Vs Launch Outcome Scatter Plot for All Sites

- The above graph shows the Launch outcome for different payloads for different booster version categories for all launch sites

- Which site has the largest successful launches?
  - KSC LC-39A

- Which site has the highest launch success rate?
  - KSC LC-39A

- Which payload range(s) has the highest launch success rate?
  - Between 3000 & 4000

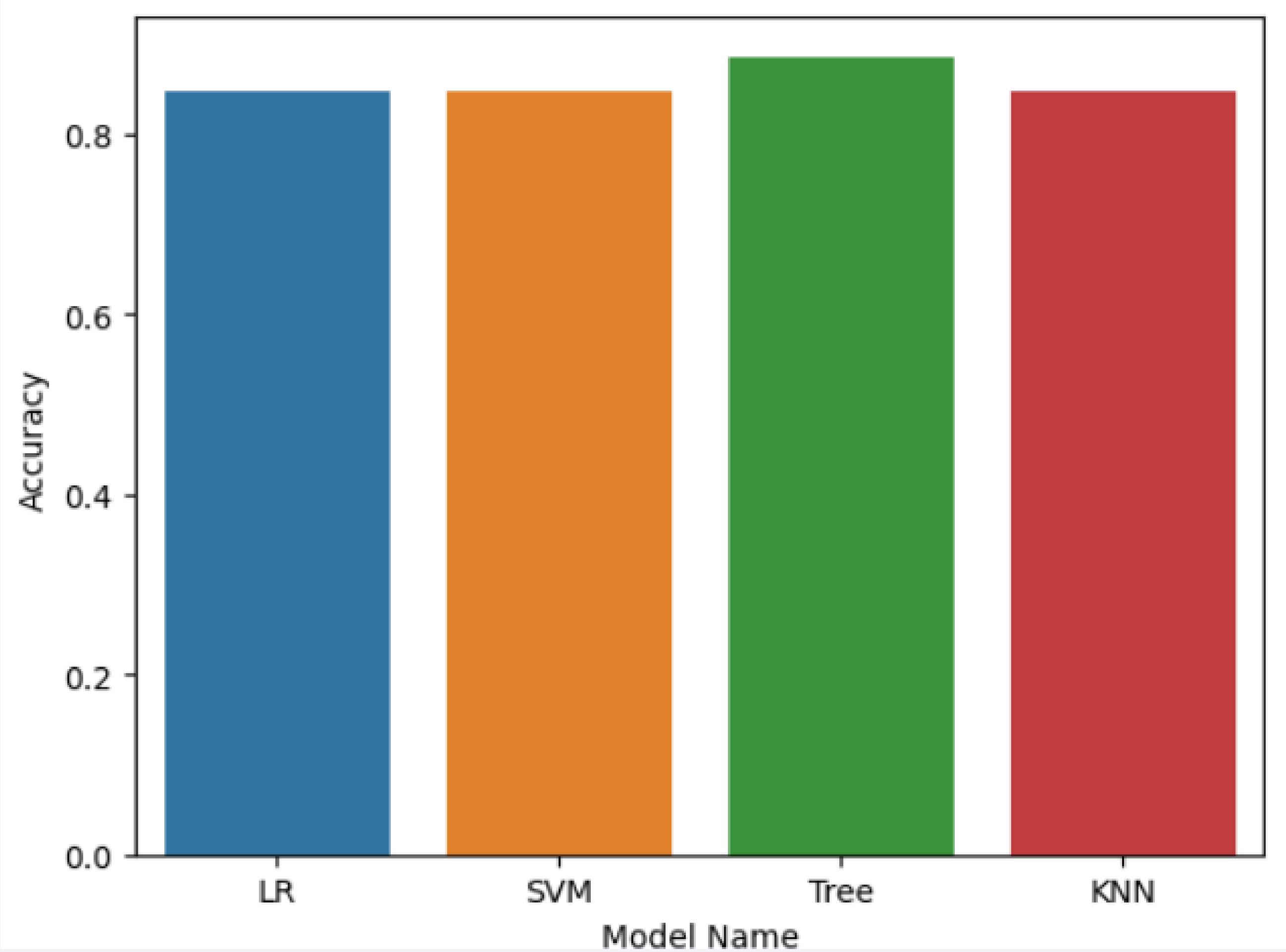
- Which payload range(s) has the lowest launch success rate?
  - Bewteen 6000 & 7000

- Which F9 Booster version (`v1.0`, `v1.1`, FT, B4, B5, etc.) has the highest launch success rate?
  - B5 – 100% - Only 1 Launch and its was a success

Section 5

Predictive Analysis (Classification)

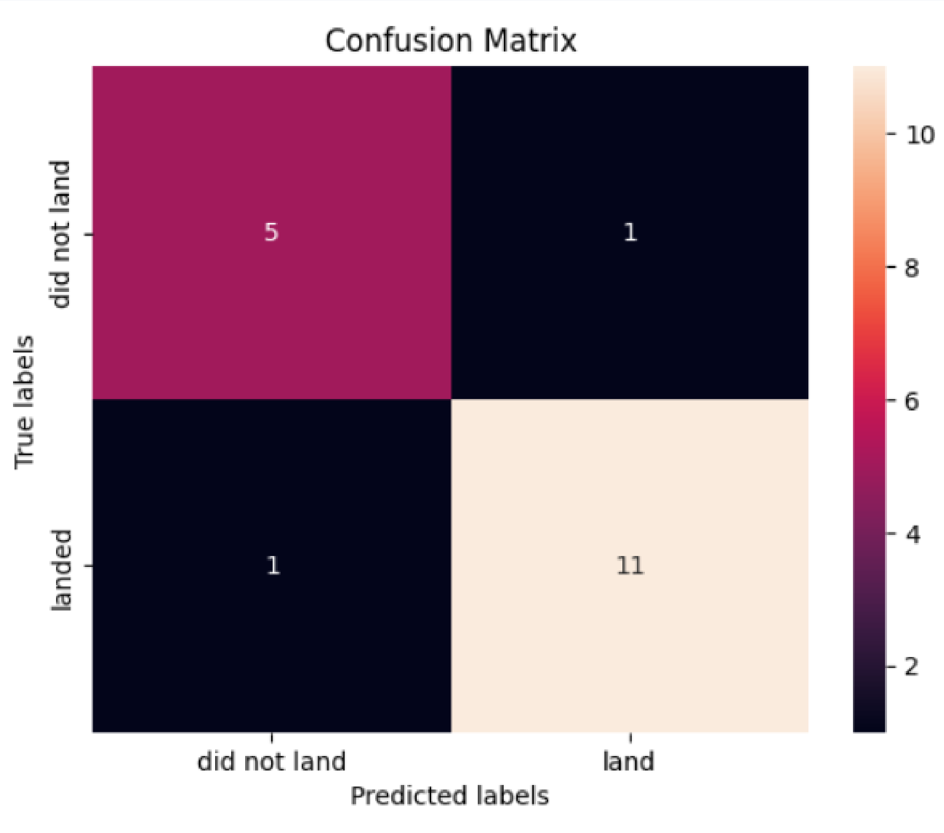
Classification Accuracy



- Decision Tree Model seems to have highest accuracy of around 0.88

Confusion Matrix

- Confusion matrix of the best performing model
- The value of False Positive = 1
- The value of True Positive = 1
- The problem with other Model's confusion Matrix is False Positive i.e. 3
- But in the Decision Tree Model the value is way less and therefore this model has best accuracy and is best performing



Conclusions

- SpaceX technology has improved a lot since 2013.
- The launch success rate has improved drastically between 2013 and 2020
- SpaceX have 4 launching sites and they are near coast lines and equator.
- KSC LC-39A launch site have highest launch success rate.
- ES-L1, GEO, HEO, SSO, VLEO are the orbits with maximum success rate
- Decision Tree proves to be more effective on predictive analysis on SpaceX dataset

Appendix

- Installing Python Libraries

```
import piplite
await piplite.install(['numpy'])
await piplite.install(['pandas']
await piplite.install(['seaborn'])
await piplite.install(['folium'])
```

- **Importing Python Libraries**

Pandas is a software library written for the Python programming language for data manipulation and analysis.

import pandas as pd

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays

import numpy as np

Matplotlib is a plotting library for python and pyplot gives us a MatLab like plotting framework. We will use this in our plotter function to plot data.

import matplotlib.pyplot as plt

#Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics

import seaborn as sns

Preprocessing allows us to standarsize our data

from sklearn import preprocessing

Allows us to split our data into training and testing data

from sklearn.model_selection import train_test_split

Allows us to test parameters of classification algorithms and find the best one

from sklearn.model_selection import GridSearchCV

Logistic Regression classification algorithm

from sklearn.linear_model import LogisticRegression

Support Vector Machine classification algorithm

from sklearn.svm import SVC

Decision Tree classification algorithm

from sklearn.tree import DecisionTreeClassifier

K Nearest Neighbors classification algorithm

from sklearn.neighbors import KNeighborsClassifier

```
# Import folium MarkerCluster plugin
from folium.plugins import MarkerCluster
# Import folium MousePosition plugin
from folium.plugins import MousePosition
# Import folium DivIcon plugin
```

- **Importing Python Libraries**

```python
import folium
# Import folium MarkerCluster plugin
from folium.plugins import MarkerCluster
# Import folium MousePosition plugin
from folium.plugins import MousePosition
# Import folium DivIcon plugin
from folium.features import DivIcon
```

- **Function to plot confusion matrix**

```
def plot_confusion_matrix(y,y_predict):
    "this function plots the confusion matrix"
    from sklearn.metrics import confusion_matrix

    cm = confusion_matrix(y, y_predict)
    ax= plt.subplot()
    sns.heatmap(cm, annot=True, ax = ax); #annot=True to annotate cells
    ax.set_xlabel('Predicted labels')
    ax.set_ylabel('True labels')
    ax.set_title('Confusion Matrix');
    ax.xaxis.set_ticklabels(['did not land', 'land']); ax.yaxis.set_ticklabels(['did not land', 'landed'])
```

- **Download and load a csv file to dataframe**

```
from js import fetch
import io

URL = '<url>'
resp = await fetch(URL)
spacex_csv_file = io.BytesIO((await resp.arrayBuffer()).to_py())
spacex_df=pd.read_csv(spacex_csv_file)
```

Thank you!