Winning Space Race with Data Science

JULIAN A. GRAJALES
10 FEBRUARY 2022

GitHub Project Material URL

Outline

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

Executive Summary

- Summary of methodologies

  This project used public available information to train a machine learning model to predict if a first stage of a launching rocket will land successfully for future use. To do this training we used the following methodologies.

  - Data collection
  - Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
  - Predictive analysis (Classification)

- Summary of results

  - Data collection and data wrangling of raw information
  - Perform exploratory data analysis and visual analytics of findings to train predictive models
  - Display EDA results by means of visualization and formal data presentation to assist in effective decision making processes

Introduction

- ## Background and project context

The SpaceX Falcon 9 is advertised as a two-stage rocket that was engineered to be efficient transporting people and payloads into space. Falcon 9 Is considered a reusable rocket, allowing the company to reuse key expensive rocket stages to reduce cost during launches.

- ## Problems to solve through this project

  - Determine the price of each launch.
  - Gathering of company information to create dashboards for different decision team.
  - Determine if SpaceX will reuse the first stage of Falcon 9 rocket.
  - Determine if the first stage will land successfully.

Section 1

Methodology

Methodology

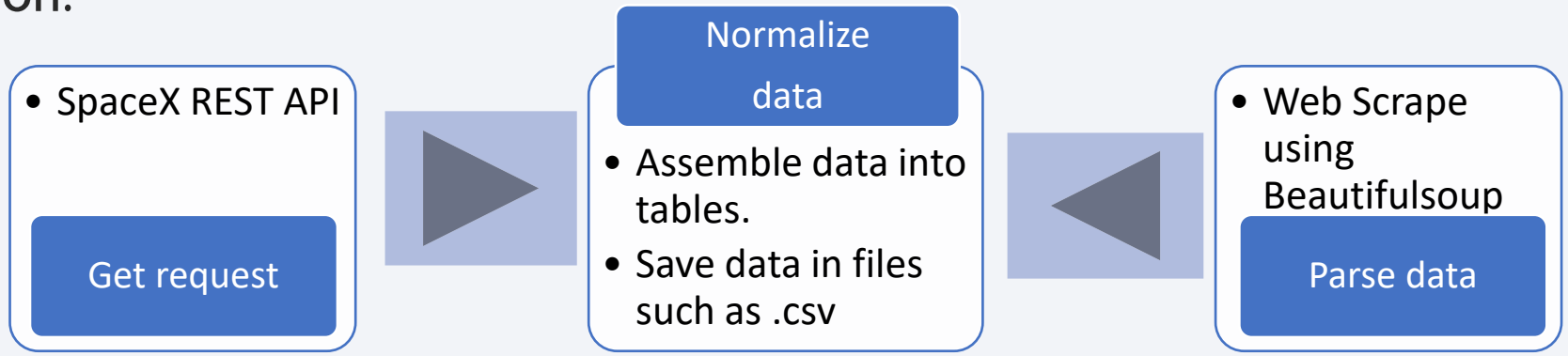
- Data collection methodology:
  - Retrieve SpaceX API data of past results, perform get request to obtain launch data and normalize data to present findings.
  - Perform web scraping from HTML tables, parse the data to convert it into data frames for visualization and analysis
  - Transform raw data to provide meaningful data to solve the addressed questions.
- Perform data wrangling
  - Data analysis to calculate the number of launches on each site
  - Calculate the number of launches on each site
  - Calculate the number and occurence of mission outcome per orbit type

Methodology

- Perform exploratory data analysis (EDA) using visualization and SQL
  - Understand the Spacex DataSet
  - Load the dataset into the corresponding table in a Db2 database
  - Execute SQL queries to answer
- Perform interactive visual analytics using Folium and Plotly Dash
  - Mark all launch sites, success/failed launches for each site and calculate the distances between a launch site to its proximities using Folium
  - Using Plotly Dash to build a real-time application for users to perform interactive visual analytics on SpaceX launch data.
- Perform predictive analysis using classification models
  - Use of machine learning to determine successful landings of the first stage of Falcon 9.
  - Split the data into training data and test data to find the best Hyperparameter for SVM, Classification Trees, and Logistic Regression.
  - Find the method that performs best using test data.

Data Collection

- Working data such as, launches, rocket used, payload delivered, launch and landing specifications, and landing outcome, proceeds from the SpaceX launching data gathered from the SpaceX REST API and public wiki pages.
- The procured data is in the form of a list of JSON objects to represent each launch.
- Data contained in the JSON will be normalized and structured into a flat table.
- BeautifulSoup is used to web scrape html tables containing Falcon9 launch records.
- Data will be parsed, sorted and organized to identify only the Falcon 9 launch information.



Data Collection – SpaceX API

- GitHub Data Collection API

1. Define Get functions: getBoosterVersion(data) getLaunchSite(data)
getPayloadData(data) getCoreData(data)

2. Request rocket launch data from SpaceX API:

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

3. Decode response code as json() and turn it into panda dataframe:

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

4. Construct dataset and convert into dictionary:

```
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}
```

Data Collection - Scraping

- Data Collection with Web Scraping

1. Request Falcon9 launch from Wiki pages:

```
static_url = https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922
data  = requests.get(static_url)
data.status_code
```

2. Create BeautifulSoup object and find all tables:

```
soup = BeautifulSoup(data.text, 'html.parser')
html_tables = soup.find_all('table')
```

3. Apply find_all function using iterate ‘th’ to extract all column names.

4. Create an empty directory to with keys from extracted columns and then fill up the directory with data extracted from table rows.

Data Wrangling

- Data Wrangling

1. Load Space X dataset and identify missing values in each informational attribute.

2. Calculate the number of launches on each site: df.LaunchSite.value_counts()

CCAFS SLC 40 = 55; KSC LC 39A = 22; VAFB SLC 4E = 13

3. Calculate the number and occurrence of each orbit: df.Orbit.value_counts()

- GTO 27 - ISS 21 - VLEO 14 - PO 9 - LEO 7 - SSO 5 - MEO 3 - GEO 1 - ES-L1 1 - SO 1 - HEO 1

Name: Orbit, dtype: int64

4. Calculate the number and occurence of mission outcome per orbit type:

```
Landing_outcomes = df.Outcome.value_counts()
for i,outcome in enumerate(Landing_outcomes.keys()):
 print(i,outcome)
```

4. Create a landing outcome label from Outcome column:

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

EDA with Data Visualization

The following charts are used to present the information:

Scatter plots:

- Flight Number vs. Payload Mass
- Flight Number vs. Launch Site
- Payload vs. Launch Site
- Orbit vs. Flight Number
- Payload vs. Orbit Type
- Orbit vs. Payload Mass

Bar graph:

- Mean vs. Orbit

Line Graph:

- Success Rate vs. Year

The scatter plots display how one variable depends from other parameter.

The bar graph allows the comparison of the successful launch depending on its orbit.

The line graph gives us the trend of successful launches in time.

- EDA with Data Visualization Github Notebook

EDA with SQL

- The exploratory data analysis with SQL loaded the datasets into tables to extract key information and presented in a selective form as follows:
  - Displaying the names of the unique launch sites in the space mission
  - Displaying 5 records where launch sites begin with the string 'KSC'
  - Displaying the total payload mass carried by boosters launched by NASA (CRS)
  - Displaying average payload mass carried by booster version F9 v1.1
  - Listing the date where the successful landing outcome in drone ship was achieved.
  - Listing the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000
  - Listing the total number of successful and failure mission outcomes
  - Listing the names of the booster_versions which have carried the maximum payload mass.
  - Listing the records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017
  - Ranking the count of successful landing_outcomes between the date 2010 06 04 and 2017 03 20 in descending order.
- EDA with SQL notebook

Build an Interactive Map with Folium

To visualize the information with the interactive map, I processed the CSV dataset to extract the geographical coordinates of the sites used to launch the SpaceX Falcon 9 first stage rocket. All sites were marked with circles and labels of each launching sites.

The circles were added to identify the locations in the map regardless of the zooming scale using folium.circle. The visualization helped identify the sites located far from the equator line, but near the tropic where the rotational speed is lesser and safer to perform the launches.

Launches at each site were marked with green (1) or red (0) depending on the success or failure of the launch.

Interactive Map GitHub notebook

Interactive Map with Folium (continued)

- **Task 1: Mark all launch sites on a map**

In [5]:
```
# Select relevant sub-columns: `Launch Site`, `Lat(Latitude)`, `Long(Longitude)`, `class`
spacex_df = spacex_df[['Launch Site', 'Lat', 'Long', 'class']]
launch_sites_df = spacex_df.groupby(['Launch Site'], as_index=False).first()
launch_sites_df = launch_sites_df[['Launch Site', 'Lat', 'Long']]
launch_sites_df
```

Out[5]:

	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610746

Build a Dashboard with Plotly Dash

- I modified the pre-assembled template from Yan Luo and Joseph Santarcangelo to create an interactive dashboard to present the launching records of SpaceX. It has a drop-down menu, listing each of the launching pads, and all as a group, a pie chart with the presenting the success launching rates for the selected site, an slider ruler to customize the result and a scatter plot diagram displaying the payload vs class of booster version.
- This plots enable the decision makers to discriminate the date in a visual manner and customizes the plots to filter the information depending of input parameters such as site, and payloads.
- Interactive Dashboard with Ploty Dash GitHub notebook

Predictive Analysis (Classification)

- To elaborate a predictive analysis, the dataset is used as follows:
  - Load data using Pandas dataframe and Numpy arry.
  - Standardize the data and transform it for analysis.
  - Split our data into training and test data sets, and list the amount of test samples.
  - Create a logistic regression object, a GridSearchCV object logreg, a support vector object, a decision tree classifier object, and a k nearest neighbors object to fit the objects, and find the best parameters.
  - Calculate the accuracy on the test data using the method *score* for each object.
  - Find the method performs best.

- Machine Learning Prediction GitHub notebook

Results

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

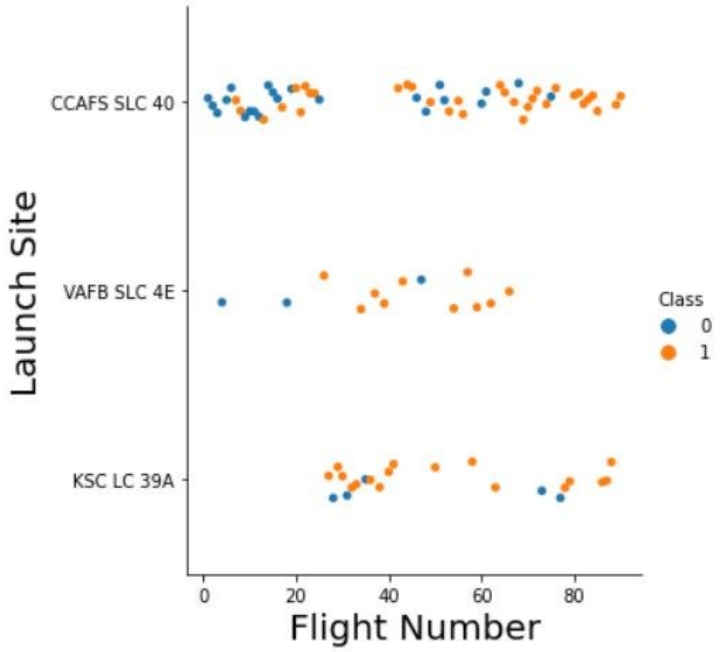
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

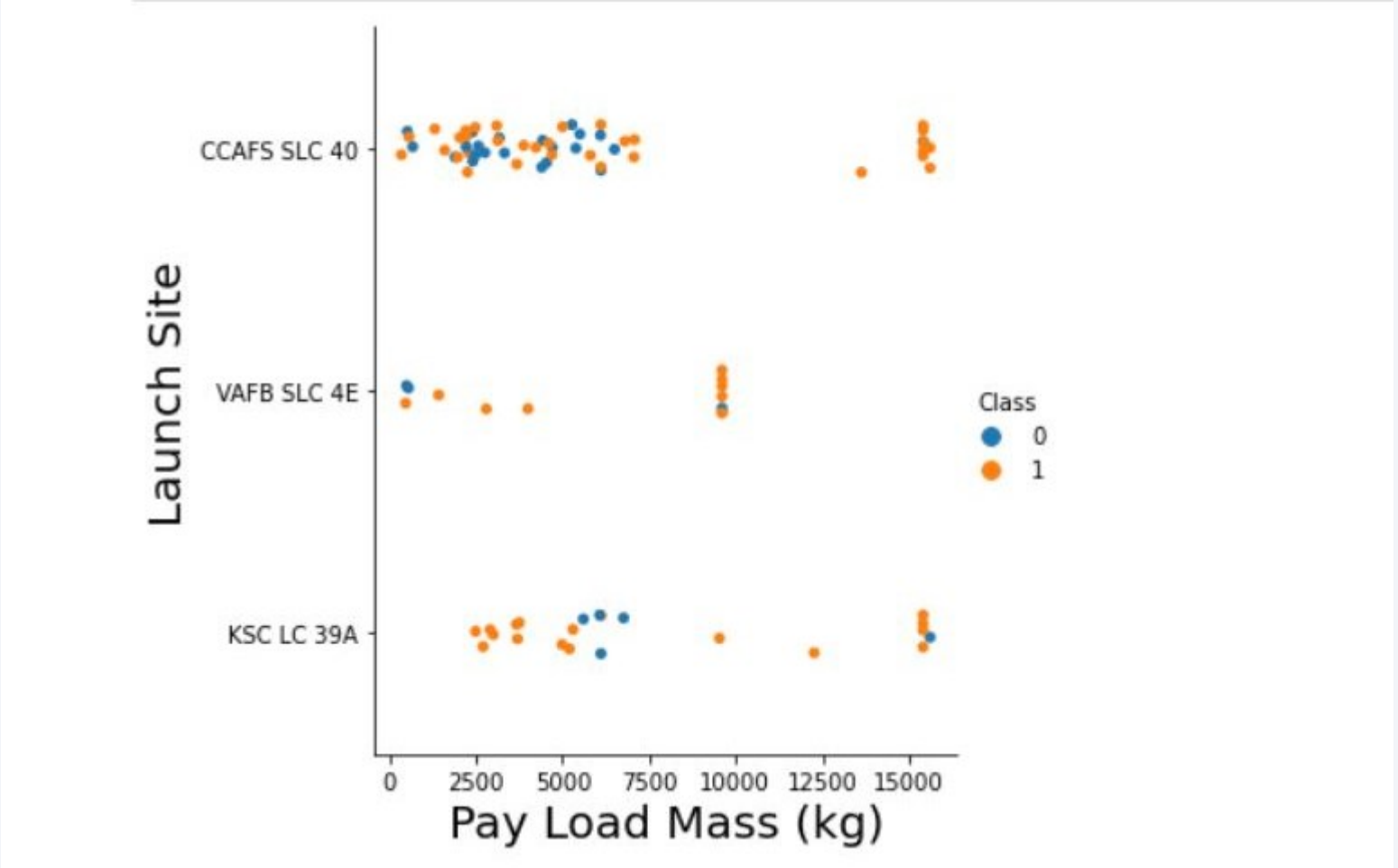
- EDA with Data Visualization Flight number vs. launch site chart:
  - Shows CCAFS SL40 had more launches than the other two sites, and as flights increased, the successful rate of return increased in all sites.

In [15]:
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class valu
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 1)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Payload vs. Launch Site

- EDA with Data Visualization Payload mass vs. launch site chart:
  - Shows the max payload per site and the landing success per pay load at each site.

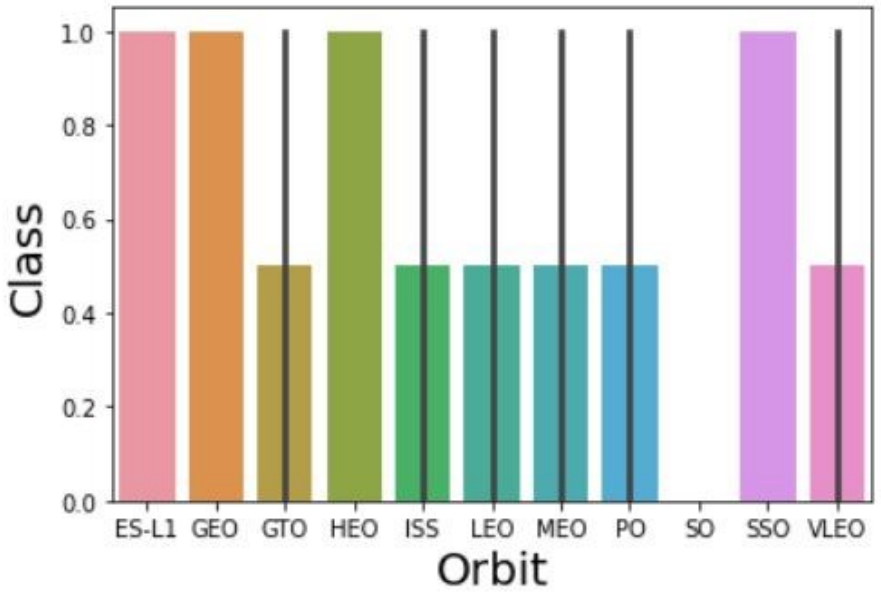


Success Rate vs. Orbit Type

- EDA with Data Visualization Orbit vs. Class chart:
  - Indicates the rate of landing success per orbit.

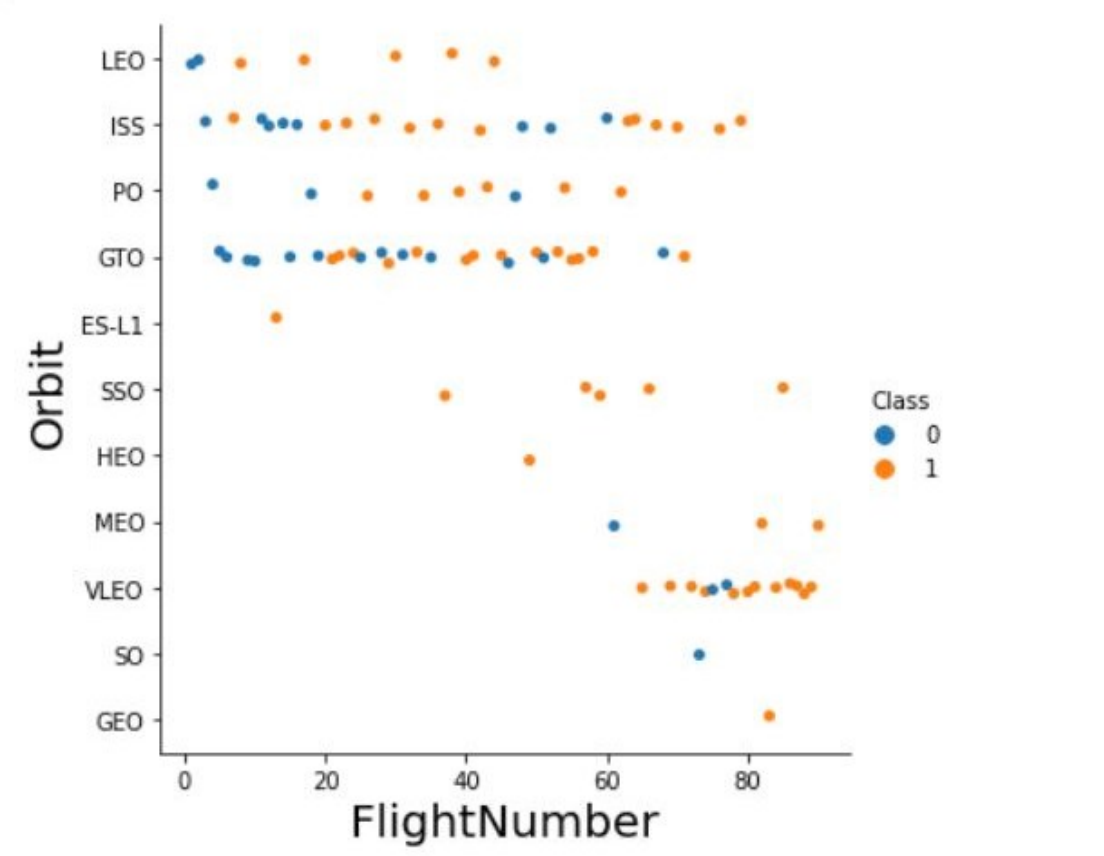
In [6]:

```
# HINT use groupby method on Orbit column and get the mean of Class column
m = df.groupby(['Orbit', 'Class'])['Class'].agg(['mean']).reset_index()
sns.barplot(y="Class", x="Orbit", data=m)

plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Class",fontsize=20)
plt.show()
```



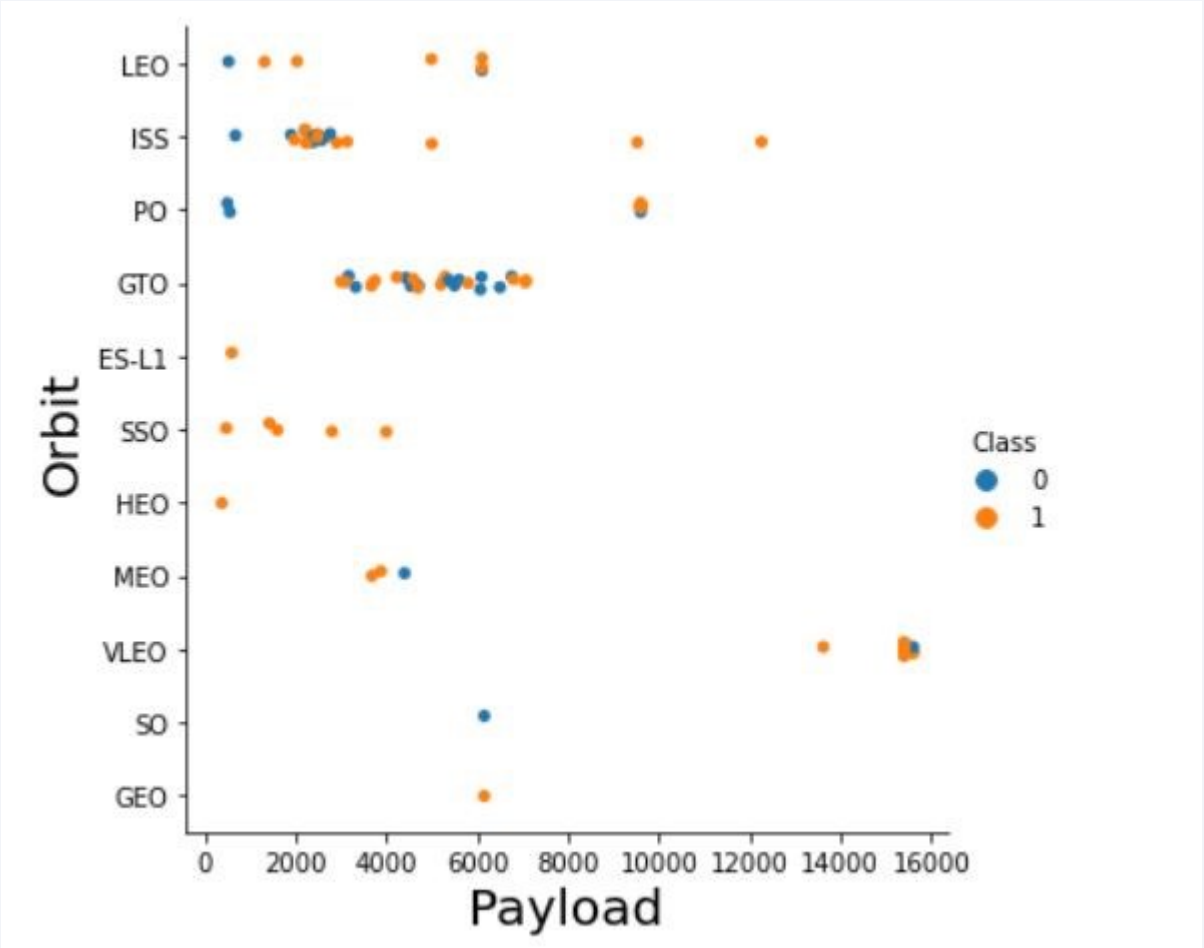
Flight Number vs. Orbit Type

- EDA with Data Visualization Flight number vs. orbit chart:
  - Shows the what progression of landing success per orbit and the increasingly use of different orbit as the flights increases.



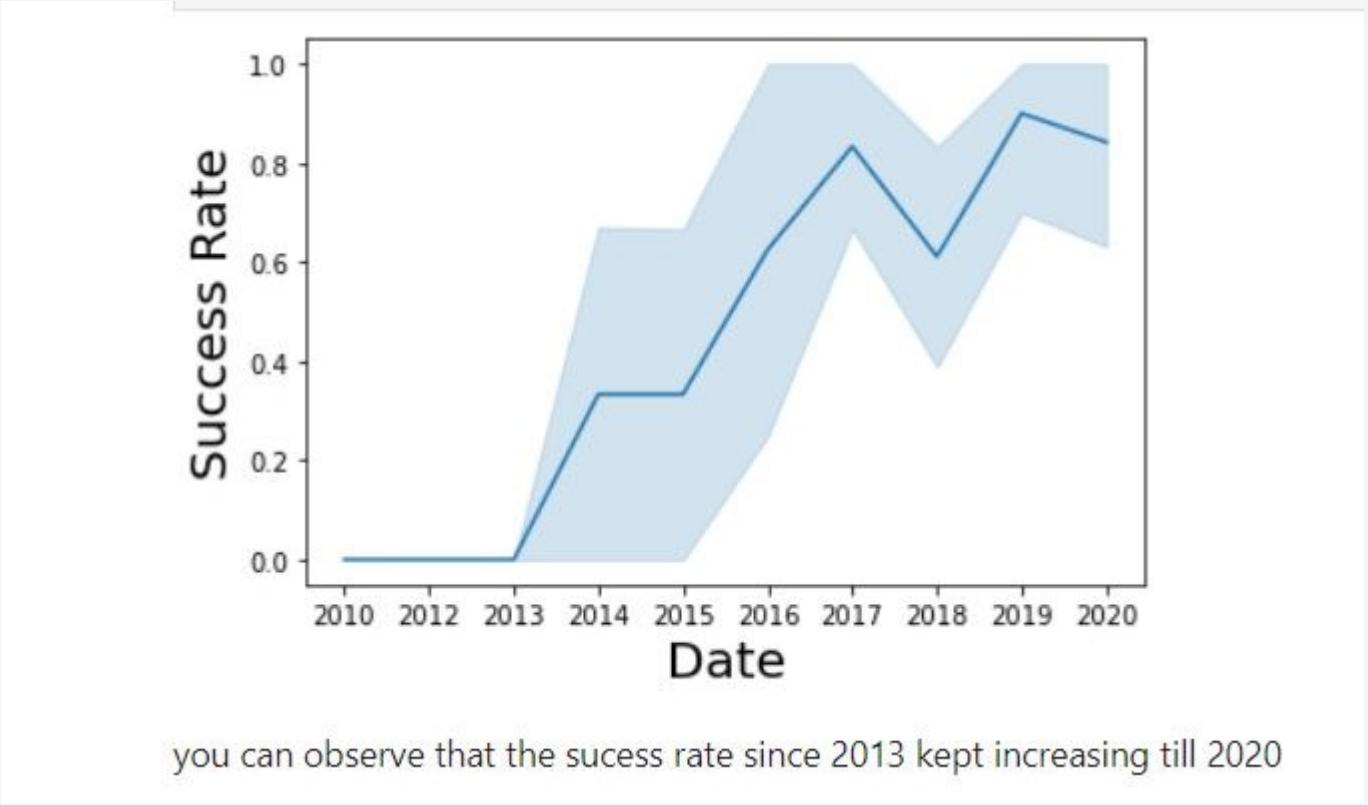
Payload vs. Orbit Type

- EDA with Data Visualization Payload vs. Orbit chart:
  - This chart relates the increase of payload to successful landing per orbit.



Launch Success Yearly Trend

- EDA with Data Visualization Date vs. Success rate chart:
  - This graph shows the increase of average rate of successful landings as the project moves over time.



All Launch Site Names

- Find the names of the unique launch sites
- The query selects all names from the Launch_Site column and removes duplicate values by usint the DISTINCT function.

Task 1

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

In [4]:
```
%%sql
SELECT DISTINCT LAUNCH_SITE From SPACEXDATASET
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[4]:

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

Launch Site Names Begin with 'KSC'

- Find 5 records where launch sites' names start with `KSC`
- The selection processed all rows where the launch site name start with KSC.

Task 2

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

In [5]:
```
%%sql
SELECT * FROM SPACEXDATASET WHERE launch_site LIKE 'KSC%' LIMIT 5
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[5]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	la
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	
2017-03-16	06:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	
2017-05-01	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	
2017-05-15	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- This selection sums all the values in the Payload mass column where the customer name contains CRS

Task 3

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

```
In [6]: %%sql
        SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXDATASET WHERE CUSTOMER LIKE '%CRS%'
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[6]:

1
48213

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- The exercise calculate the average value from all values of Payload mass column on all rows where the booster version is equal to F9 v1.1

Task 4

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

```
In [7]: %%sql
        SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXDATASET WHERE BOOSTER_VERSION LIKE 'F9 v1.1'
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[7]:

1
2928

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- In this task, I selected the minimum (oldest) date from the dataset where both clauses are valid for drone name and successful landing.

Task 5

List the date where the first succesful landing outcome in drone ship was acheived.

Hint:Use min function

In [8]:
```
%%sql
SELECT MIN(DATE) FROM SPACEXDATASET WHERE LANDING__OUTCOME LIKE '%drone ship%' and MISSION_OUTCOM
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databas
es.appdomain.cloud:32731/bludb
Done.
```

Out[8]:

1
2015-01-10

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

Task 6

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

In [9]:
```
%%sql
SELECT BOOSTER_VERSION, LANDING__OUTCOME, PAYLOAD_MASS__KG_ From SPACEXDATASET where LANDING__OUT
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databas
es.appdomain.cloud:32731/bludb
Done.
```

Out[9]:

booster_version	landing__outcome	payload_mass__kg_
F9 FT B1032.1	Success (ground pad)	5300
F9 B4 B1040.1	Success (ground pad)	4990

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

Task 7

List the total number of successful and failure mission outcomes

In [10]:
```
%%sql
select DISTINCT MISSION_OUTCOME, count(MISSION_OUTCOME) as missionoutcomes from SPACEXDATASET GRO
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databas
es.appdomain.cloud:32731/bludb
Done.
```

Out[10]:

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

Boosters Carried Maximum Payload

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

Task 8

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

In [11]:
```
%%sql
select BOOSTER_VERSION as boosterversion from SPACEXDATASET where PAYLOAD_MASS__KG_=(select max(P
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databas
es.appdomain.cloud:32731/bludb
Done.
```

Out[11]:

boosterversion
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 records which will display the month names, succesful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017.

Task 9

List the records which will display the month names, succesful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017

In [12]:
```
%%sql
SELECT MONTH(DATE),MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXDATASET wher
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqd
e00.databases.appdomain.cloud:32731/bludb
Done.
```

Out[12]:

1	mission_outcome	booster_version	launch_site
1	Success	F9 FT B1029.1	VAFB SLC-4E
2	Success	F9 FT B1031.1	KSC LC-39A
3	Success	F9 FT B1030	KSC LC-39A
3	Success	F9 FT B1021.2	KSC LC-39A
5	Success	F9 FT B1032.1	KSC LC-39A
5	Success	F9 FT B1034	KSC LC-39A
6	Success	F9 FT B1035.1	KSC LC-39A
6	Success	F9 FT B1029.2	KSC LC-39A

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

Task 10

Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

In [13]:
```
%%sql
SELECT DATE, LANDING__OUTCOME FROM SPACEXDATASET WHERE DATE BETWEEN '2010-06-04' AND '
```

```
 * ibm_db_sa://lcm86268:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

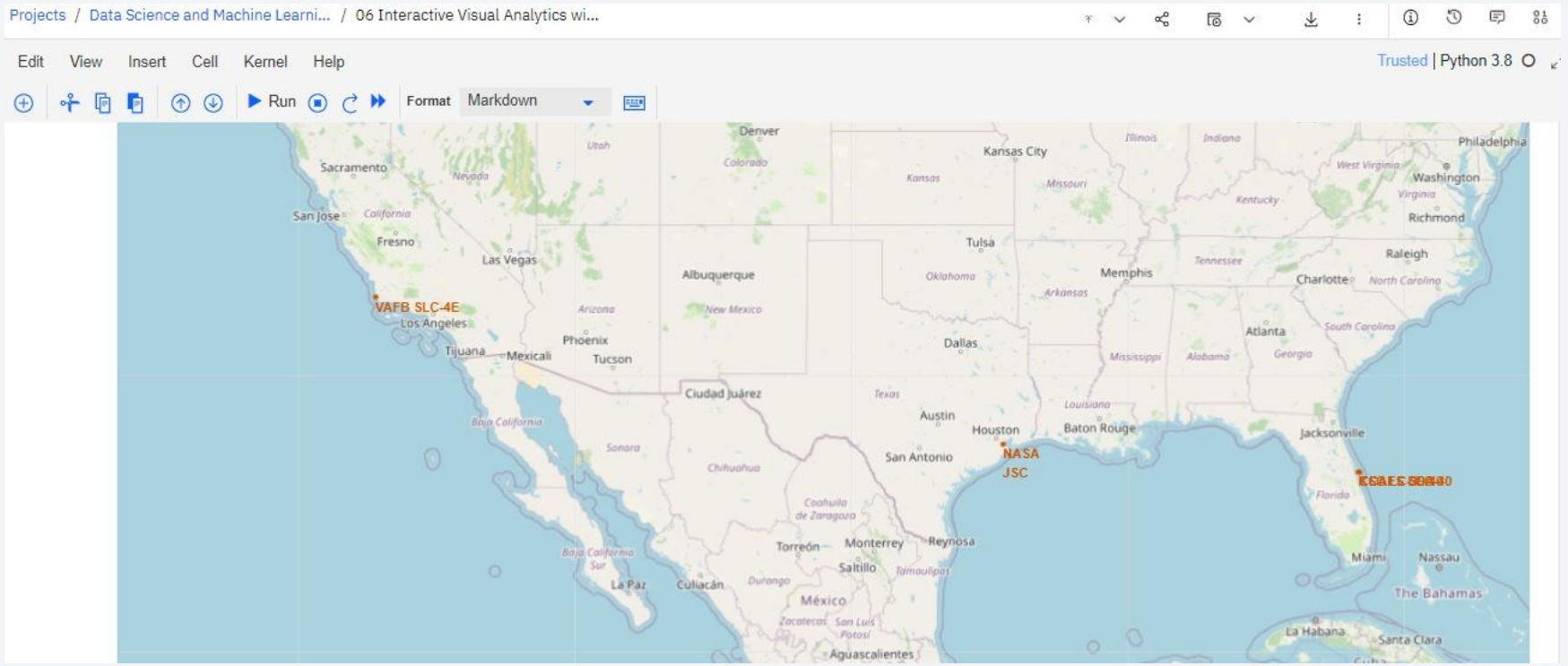
Out[13]:

DATE	landing__outcome
2017-03-16	No attempt
2017-02-19	Success (ground pad)
2017-01-14	Success (drone ship)
2016-08-14	Success (drone ship)
2016-07-18	Success (ground pad)
2016-06-15	Failure (drone ship)
2016-05-27	Success (drone ship)

Section 3

Launch Sites Proximities Analysis

Markers in map for each of the launching sites



- Are all launch sites in proximity to the Equator line? The launch sites are not in proximity to the equator line but to the northern tropic circle because the rotational speed is lesser at the tropic line due to safer winds speeds at the launch pad surfaces.
- Are all launch sites in very close proximity to the coast? Yes, in case of mishaps or launch failures, rockets may be directed to the ocean, instead of populated areas surrounding a launch site in the inner lands.

Mark the success/failed launches for each site on the map

```
In [32]: spacex_df.tail(10)
```

Out[32]:

	Launch Site	Lat	Long	class
46	KSC LC-39A	28.573255	-80.646895	1
47	KSC LC-39A	28.573255	-80.646895	1
48	KSC LC-39A	28.573255	-80.646895	1
49	CCAFS SLC-40	28.563197	-80.576820	1
50	CCAFS SLC-40	28.563197	-80.576820	1
51	CCAFS SLC-40	28.563197	-80.576820	0
52	CCAFS SLC-40	28.563197	-80.576820	0
53	CCAFS SLC-40	28.563197	-80.576820	0
54	CCAFS SLC-40	28.563197	-80.576820	1
55	CCAFS SLC-40	28.563197	-80.576820	0

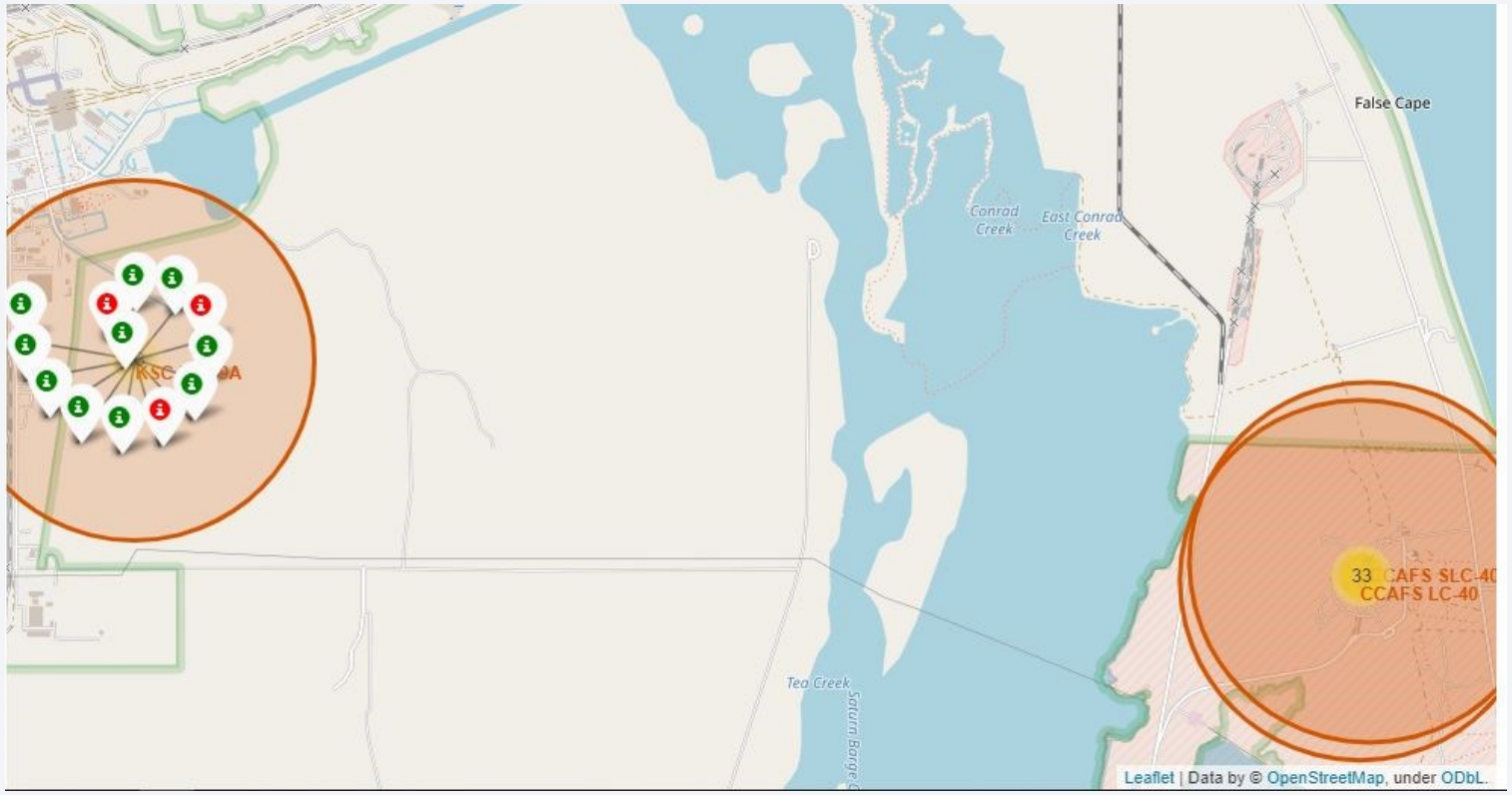
```
In [35]: # Function to assign color to launch outcome
def assign_marker_color(launch_outcome):
    if launch_outcome == 1:
        return 'green'
    else:
        return 'red'

spacex_df['marker_color'] = spacex_df['class'].apply(assign_marker_color)
spacex_df.tail(10)
```

Out[35]:

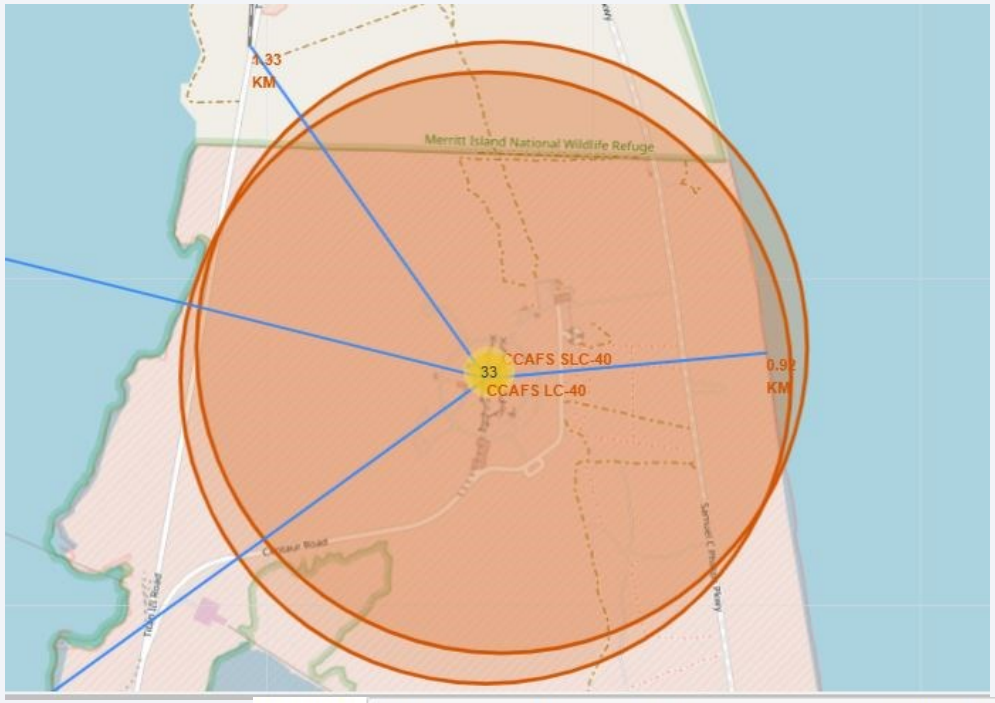
	Launch Site	Lat	Long	class	marker_color
46	KSC LC-39A	28.573255	-80.646895	1	green
47	KSC LC-39A	28.573255	-80.646895	1	green
48	KSC LC-39A	28.573255	-80.646895	1	green
49	CCAFS SLC-40	28.563197	-80.576820	1	green
50	CCAFS SLC-40	28.563197	-80.576820	1	green
51	CCAFS SLC-40	28.563197	-80.576820	0	red
52	CCAFS SLC-40	28.563197	-80.576820	0	red
53	CCAFS SLC-40	28.563197	-80.576820	0	red
54	CCAFS SLC-40	28.563197	-80.576820	1	green
55	CCAFS SLC-40	28.563197	-80.576820	0	red

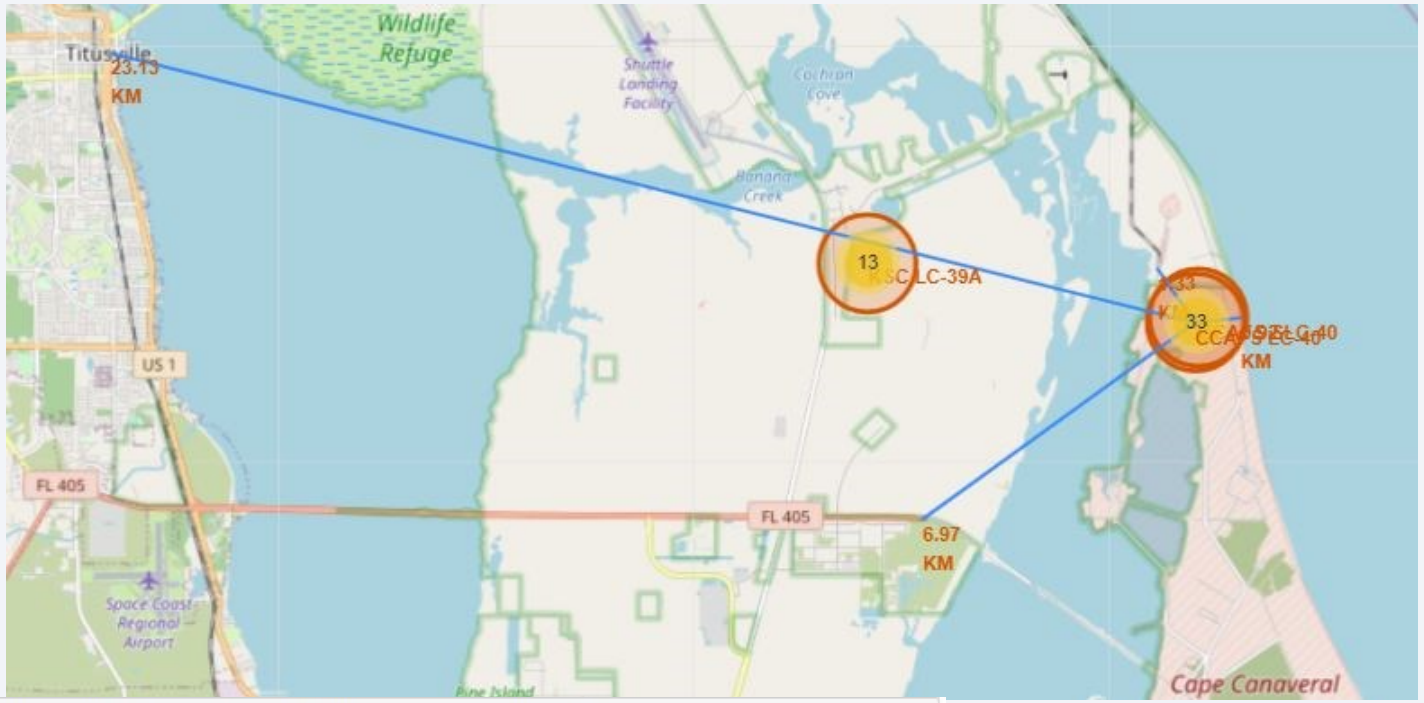
Marker cluster for each of the launching sites

- Each site has been marked with a cluster indicating the amount of launches and a color code designation of the amount of successful and failed launches represented by green and red markers respectively.

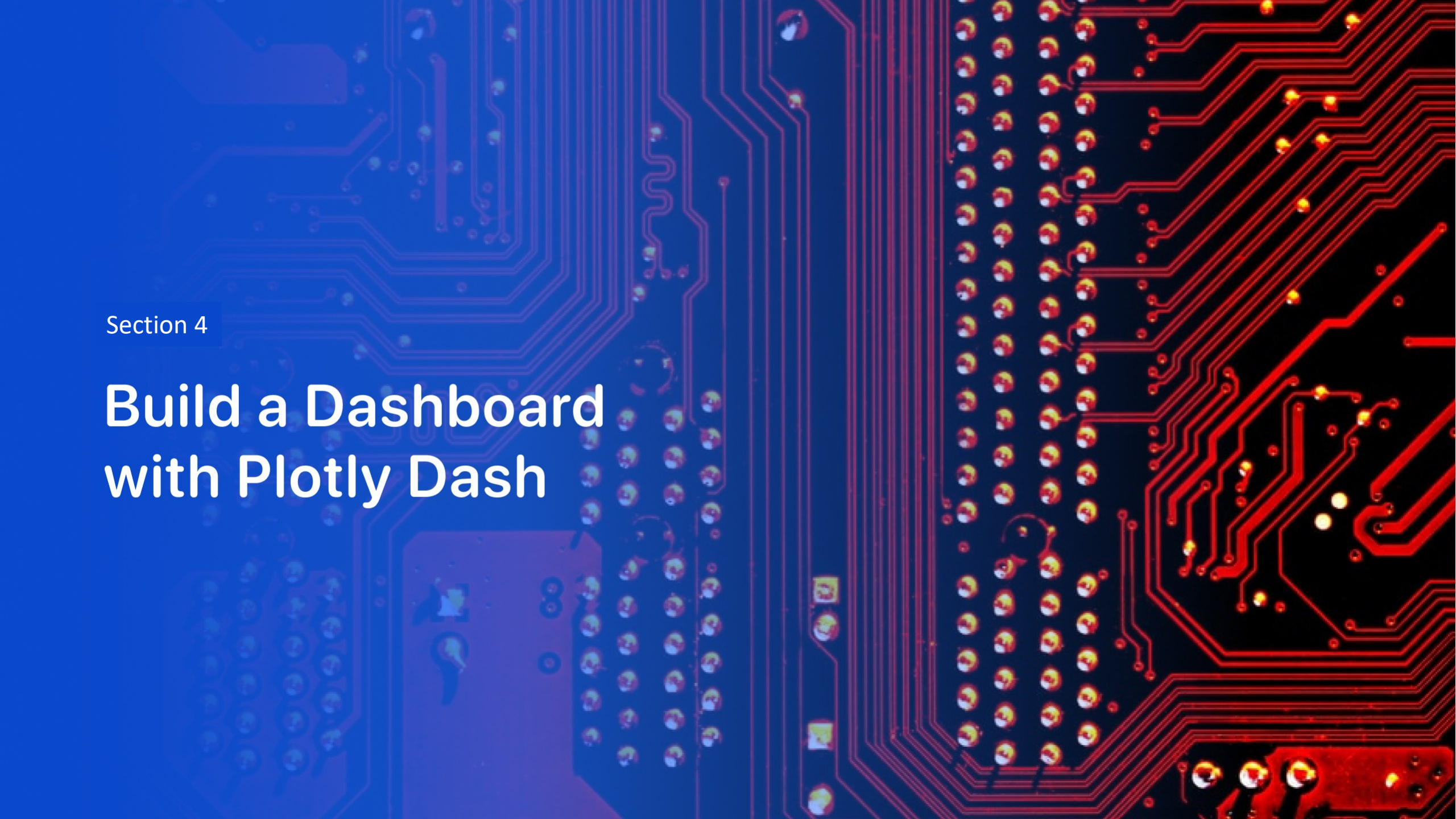


Distances between a launch site to its landmark proximities



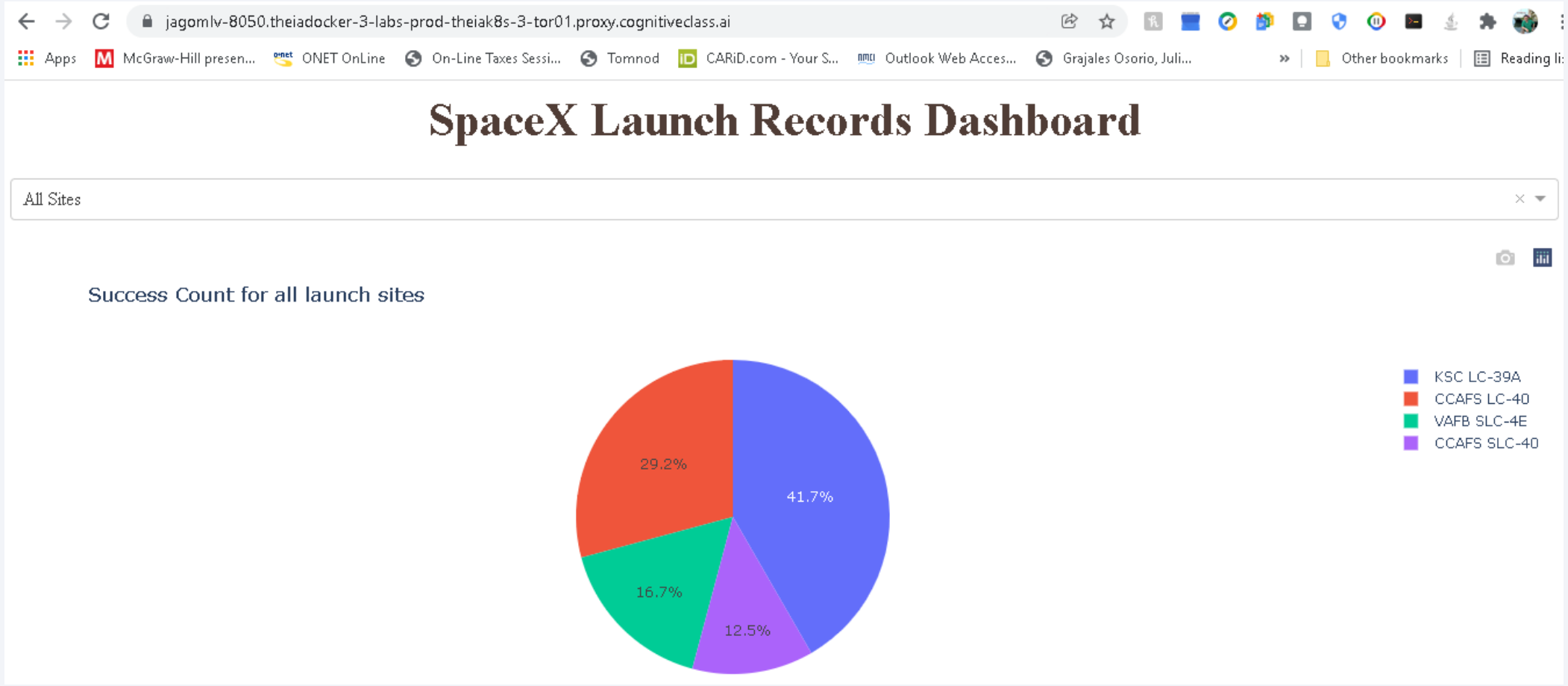



```
In [20]: distance_coastline = calculate_distance(launch_site_lat, launch_site_lon, coastline_lat, coastline_lon)
         print("The distance from the coast to the launch site CCAFS SLC-40 is =", distance_coastline, "km")
         distance_highway = calculate_distance(launch_site_lat, launch_site_lon, closest_highway[0], closest_highway[1])
         print('The distance from the Nasa Parkway to the launch site CCAFS SLC-40 is  =',distance_highway, ' km')
         distance_railroad = calculate_distance(launch_site_lat, launch_site_lon, closest_railroad[0], closest_railroad[1])
         print('The distance from the Nasa railroad to the launch site CCAFS SLC-40 is  =',distance_railroad, ' km')
         distance_city = calculate_distance(launch_site_lat, launch_site_lon, closest_city[0], closest_city[1])
         print('The distance from the Titusville city to the launch site CCAFS SLC-40 is  =',distance_city, ' km')

         The distance from the coast to the launch site CCAFS SLC-40 is = 0.9244036402408913 km
         The distance from the Nasa Parkway to the launch site CCAFS SLC-40 is  = 6.969851083196467  km
         The distance from the Nasa railroad to the launch site CCAFS SLC-40 is  = 1.3287191344977784  km
         The distance from the Titusville city to the launch site CCAFS SLC-40 is  = 23.132922953913646  km
```


Section 4
Build a Dashboard with Plotly Dash

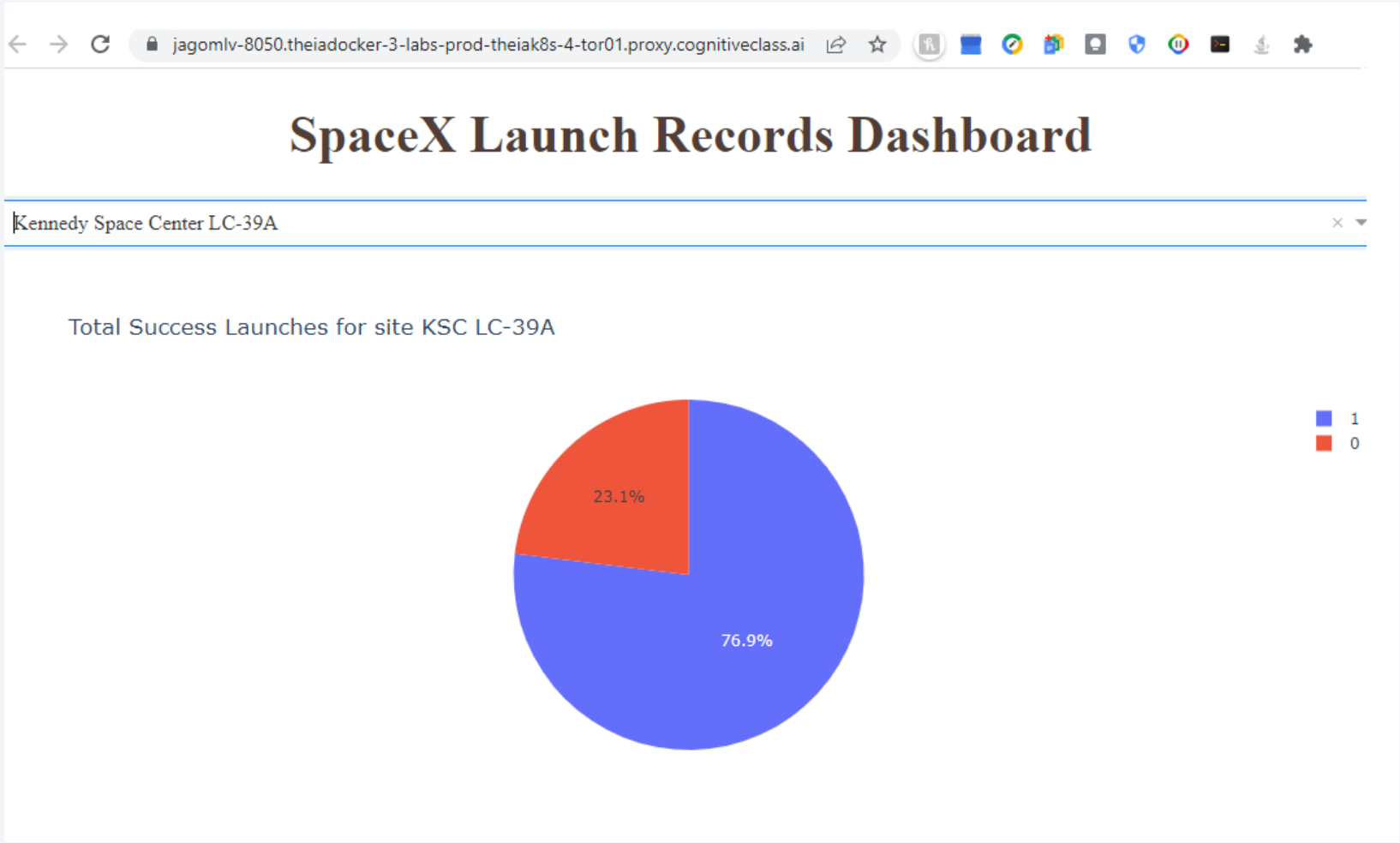
Distribution of success count of launch records for all sites

- TASK 2: Callback function to render success-pie-chart based on selected site dropdown



Space X Launch Records Dashboard.

- Launch site with highest launch success ratio.

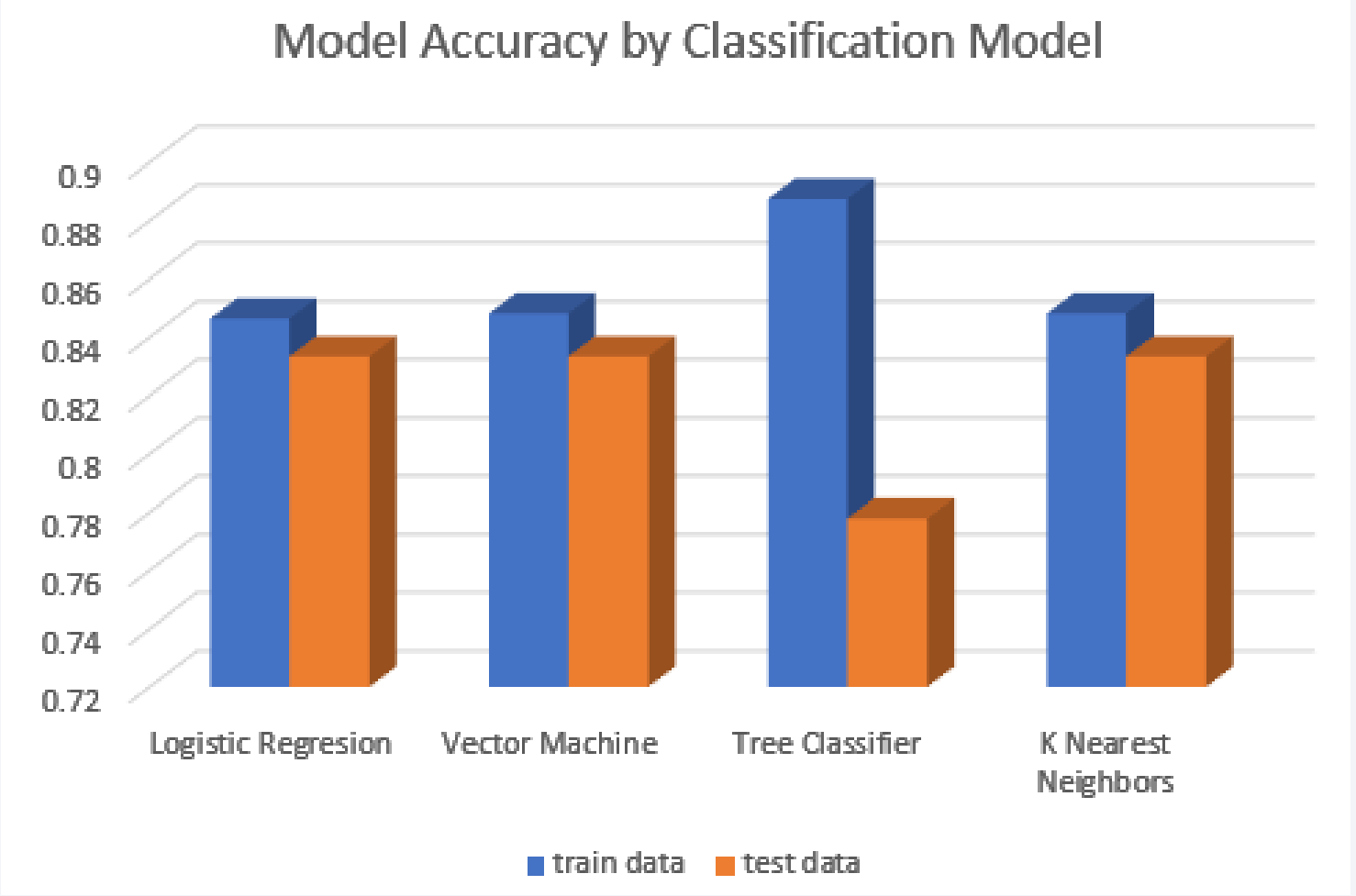


Success count of payload mass between 3000 and 7000 kg from all sites

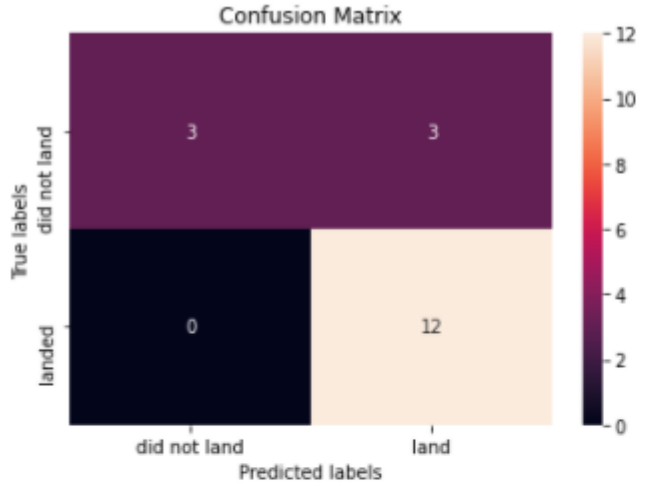
Section 5

Predictive Analysis (Classification)

Classification Accuracy



Confusion Matrix



TASK 12

Find the method performs best:

In [32]:

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.8767857142857143
Best Params is : {'criterion': 'gini', 'max_depth': 14, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 5, 'splitter': 'random'}
```

Conclusions

- Each site had increased success as the flight number count increased.
- Weight is a key element on the successful launches, the lighter the booster is, the better it performs.
- SpaceX launches got better with the passing of time, considering past results as the program evolves in time.
- The sites in Florida had the highest and lowest successful launch rates of all sites. Kennedy Space Center was the most successful and Cape Canaveral was the least successful sites.
- Best Algorithm is Tree classifier algorithm, with a score of 0.8767857142857143
- All sites were located in coastal areas, at about the tropical circle latitude.

Appendix

- Skill Network Lab for plotly notebook.
- IBM Cloud Pack for Data notebooks.
- GitHub training upload and example files.
- CSV files using Microsoft Excel.

Thank you!