IBM

Data Science Capstone Project

SPACE X

Winning Space Race with Data Science

LAK CHEE MENG

26 November 2021

Outline

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

Executive Summary

1. Data collection methodology

Gathering data from SpaceX Rest API

Web Scrapping from Wiki pages

2. Perform data wrangling

Review the attributes of the dataset

Using One Hot Encoding to convert some values where necessary

3. Perform exploratory data analysis (EDA) using visualization and SQL

To show relationships between attributes.

To determine what attributes are correlated with successful landings.

4. Perform interactive visual analytics using Folium and Plotly Dash

To create interactive visuals to allow users to find patterns of dataset faster and more effectively.

To build a dashboard contains input components to interact with charts to provide more insights from the dataset.

5. Perform predictive analysis using classification models

To build a machine learning pipeline using classification models (Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbors) to perform predictive analysis.

Introduction

Project background and context

SpaceX, founded by Billionaire industrialist Allon Musk, is one of the most successful commercial space companies.

SpaceX' main advantage is the ability to launch rockets which are designed to reuse the first stage, in order to reduce the costs significantly.

Objectives

The objectives of this project is to predict any future Falcon 9 rocket launches will be successful by taking into considerations of factors:

- Launch site
- Payload mass
- Booster version
- Orbit type

Section 1

Methodology

Data Collection with an API

The SpaceX launch data is gathered from SpaceX REST API with different endpoints. We will working with the endpoint api.spacexdata.com/v4/launches/past to get past launch data.

The response will be in the form of a JSON and we need to convert into a dataframe by using json-normalize functon.

1. Getting response from API.

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

```
response = requests.get(spacex_url)
```

2. Convert the response to a .json file and turn it into a Pandas dataframe using .json_normalize().

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Using pre-defined functions to extract information from API.

4. The data from requests will be stored in lists and create a new dataframe.

```
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}
```

5. Filter the data and save the filtered data.

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

github.com/cmlak/SpaceX-Falcon-9

Data Collection (web scraping)

The SpaceX launch data can also be gathered by scraping related Wiki pages. We will chose the Python Beautifulsoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records.

1. Request Wiki page from URL

```
page = requests.get(static_url)
```

2. Create a BeautifulSoup Object

```
soup = BeautifulSoup(page.text, 'html.parser')
```

3. Extract all column/variable names from the HTML table header

```
html_tables = soup.find_all('table')
```

4. Extract column names

```
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

5. 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 ( )']

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']=[]
```

6. Appending data to keys

```
extracted_row = 0
#Extract each table
for table_number,table in enume
   # get table row
    for rows in table.find_all(
        #check to see if first
        if rows.th:
            if rows.th.string:
```

7. Converting dictionary to data frame and save

```
df=pd.DataFrame(launch_dict)
df.to_csv('spacex_web_scraped.csv', index=False)
```

github.com/cmlak/SpaceX-Falcon-9

Data Wrangling

github.com/cmlak/SpaceX-Falcon-9

We will perform Exploratory Data Analysis (EDA) to find some patterns in the data and determine the label for training models.

The launch outcomes are keys to labels and need to convert into two valuables.

1. Identify and calculate the missing values in attributes

```
df.isnull().sum()/df.count()*100
```

2. Calculate the number of launches on each site

```
df["LaunchSite"].value_counts()
```

3. Calculate the number and occurrence of each orbit

```
df["Orbit"].value_counts()
```

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

```
Out[18]: True ASDS      41
         None None      19
         True RTLS      14
         False ASDS      6
         True Ocean      5
         False Ocean     2
         None ASDS       2
         False RTLS      1
         Name: Outcome, dtype: int64
```

5. 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)
```

Landing Success Rate:
67%

6. Export data set as .CSV

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

EDA with Data Visualization

We will create visuals to better understand the data and analyze the relationships between difference attributes.

The ultimate aims of data visualization are to find the keys relationships between attributes and labels that will lead us to make final predictive conclusion.

Plot scatter point or line charts

Flight Number vs. Payload
- *To determine how flight number and payload affect the outcome*

Flight Number vs. Launch Site
- *To determine how launch site affect the outcome*

Payload vs. Launch Site
- *To find the patterns of payload assigned to launch site*

Plot a bar chart

Success rate of each orbit
- *To* check the success rate based on each orbit type

Orbit type vs Flight Number
- *To determine the number of flight and outcome based on each orbit type*

Payload vs. Orbit type
- *To determine the number of flight and outcome based on each orbit type*

Plot a line chart

Launch success yearly trend
- *To get the average launch success trend*

github.com/cmlak/SpaceX-Falcon-9

EDA with SQL

github.com/cmlak/SpaceX-Falcon-9

We will perform Exploratory Data Analysis (EDA) by execute SQL queries to understand the dataset.

1. Displaying the names of the unique launch sites in the space mission.
2. Displaying 5 records where launch sites begin with the string 'CCA'.
3. Displaying the total payload mass carried by boosters launched by NASA (CRS).
4. Displaying average payload mass carried by booster version F9 v1.1.
5. Listing the date when the first successful landing outcome in ground pad was acheived.
6. Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
7. Listing the total number of successful and failure mission outcomes.
8. Listing the names of the booster versions which have carried the maximum payload mass. Use a subquery.
9. Listing the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015.
10. 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.

Build an Interactive Map with Folium

We use Folium, a a powerful data visualization library for visualizing geospatial data, to build an interactive map.

We mark the launch site locations with launch outcome information and calculate the distances between different launch sites, this will allow us to find some geographical patterns about launch sites.

1. Mark all launch sites on a map

We took the latitude and longtitude coordinates at each launch site and added a Circle Marker around each launch site with a label of the name of the launch site.

2. Mark the launch outcome for each site on the map

We enhance the map by adding the launch outcomes (success rate) for each site. We create markers for all launch records with green marker represent successful launch (class=1) and red marker for failed to launch (class=0).

3. Calculate the distances between a launch site to its proximities

We calculate the distance between two points on the map based on their Lat and Long values. We mark down a point on the closest landmarks and calculate the distance between the point.

We will be able to answer the following questions:

Are launch sites in close proximity to railways?
Are launch sites in close proximity to highways?
Are launch sites in close proximity to coastline?
Do launch sites keep certain distance away from cities?

github.com/cmlak/SpaceX-Falcon-9

Build a Dashboard with Plotly Dash

We build a dashboard application with the Python Plotly Dash package. The dashboard will allow us to find more insights from the SpaceX dataset more easily than with static graphs.

We define a dash application skeleton and input components, graph and callback function.

We will be able to answer the questions by using the dashboard:

- Which site has the largest successful launches?
- Which site has the highest launch success rate?
- Which payload range(s) has the highest launch success rate?
- Which payload range(s) has the lowest launch success rate?
- Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest launch success rate?

github.com/cmlak/SpaceX-Falcon-9

1. Dashboard component: Dropdown List

The Dropdown List component is linked to a pie chart for four different launch sites, to allow us to find the success rate of launch, by visualizing the launch success counts in an interactive way.

2. Dashboard Component: Range Slider

The Range Slider component is linked to a success-payload-scatter-chart scatter, to allow us to find if variable payload is correlated to mission outcome. We can select different payload range and see if we can identify some visual patterns.

Predictive Analysis (Classification)

Finally, we use the information we have gathered and processed to build machine learning models to predict the outcome whether the first stage will successfully land.

1. Building Model

- Import necessary libraries
- Perform exploratory Data Analysis and determine Training Labels
- create a column for the class
- Standardize the data
- Split into training data and test data
- Create machine learning objects: SVM, Classification Trees and Logistic Regression

2. Evaluating Model

- Check accuracy for each model
- Using GridSearchCV to test parameters of classification algorithms and find the best one
- Find best hyperparameter for SVM, Classification Trees and Logistic Regression
- Plot Confusion Matrix

3. Improving Model

- Feature Engineering
- Algorithm Tuning

4. Selecting Best Model

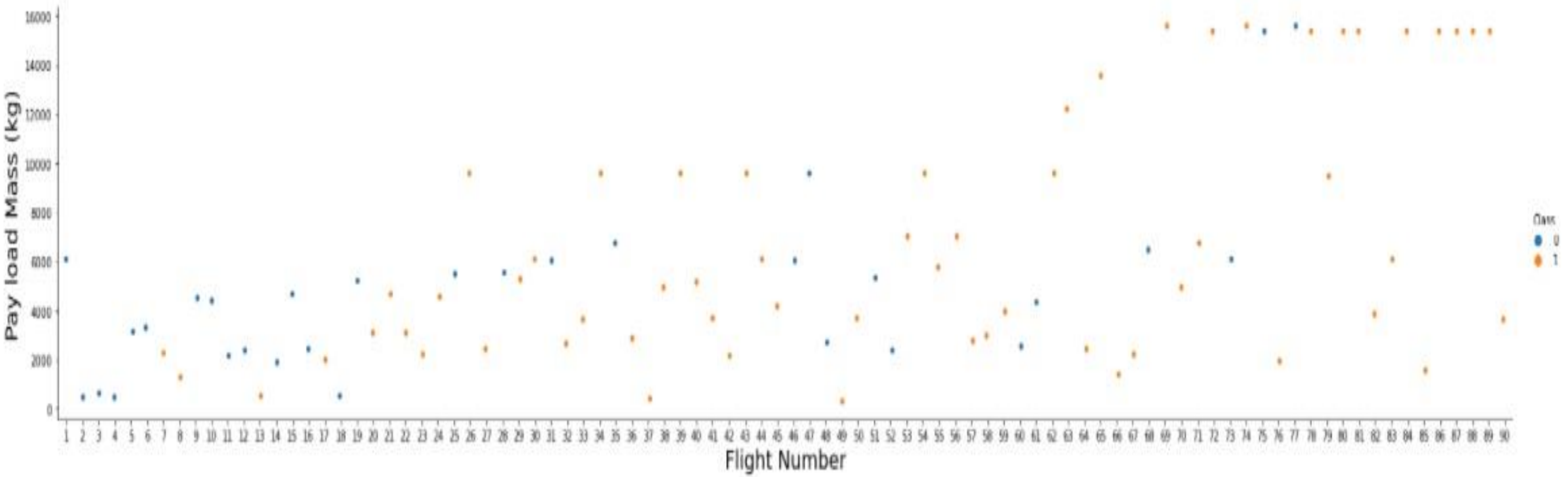
The model with the best accurary score will be selected

github.com/cmlak/SpaceX-Falcon-9

Section 2

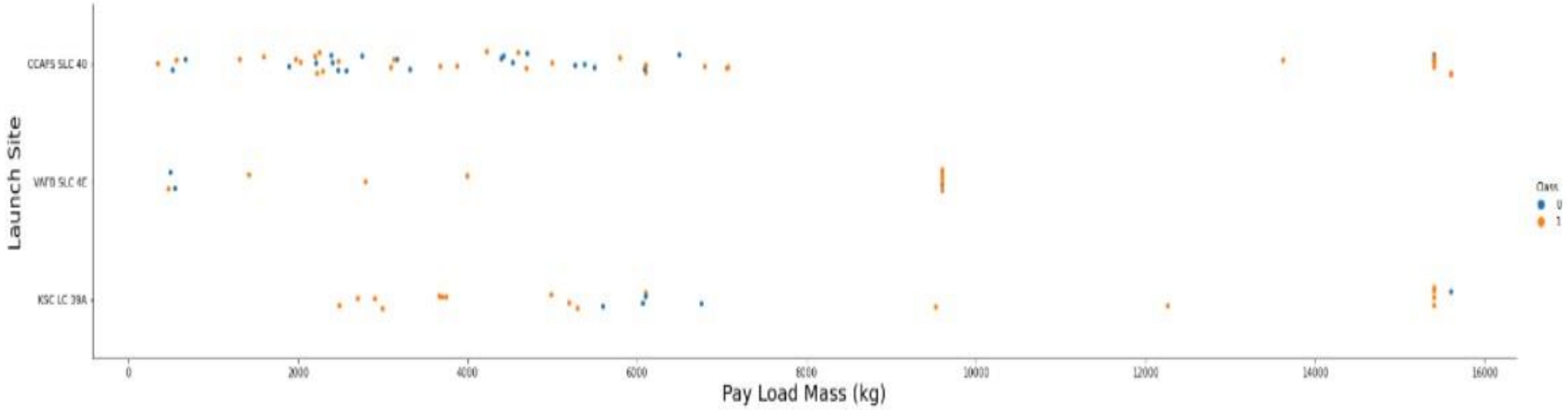
Insights drawn from EDA

2.1 Flight Number vs. Launch Site



We see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.

2.2 Payload vs. Launch Site



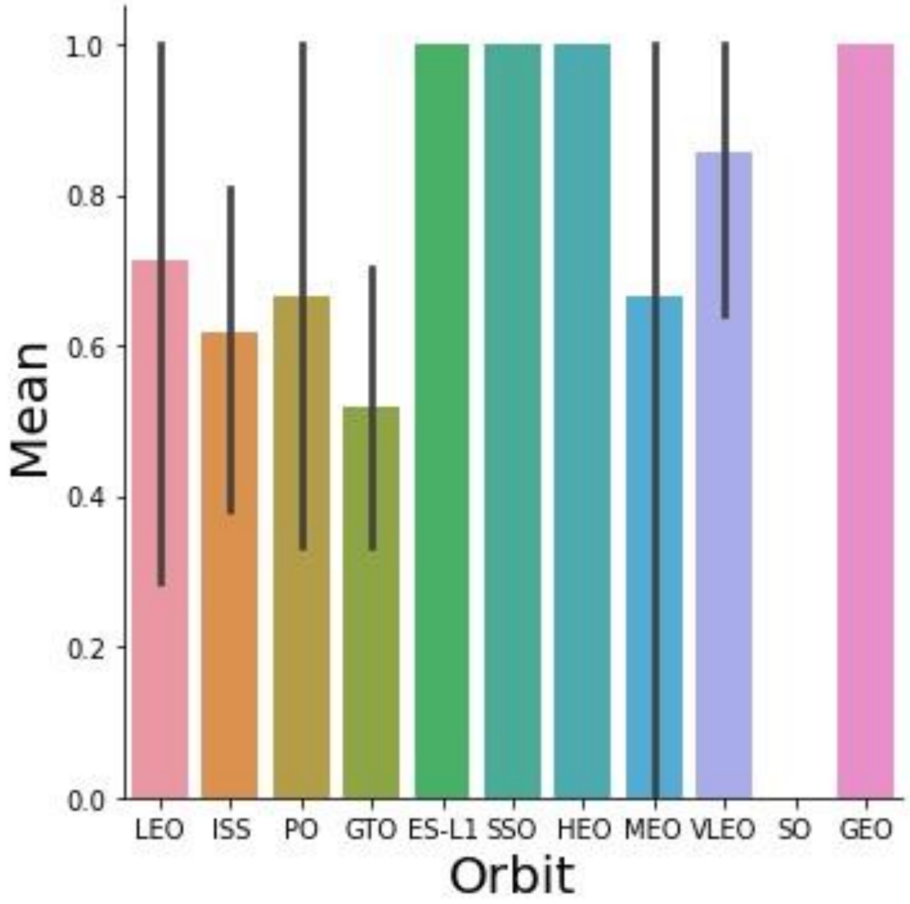
For the VAFB-SLC launch site there are no rockets launched for heavy payload mass (greater than 10000).

2.3 Success Rate vs. Orbit Type

Orbit GEO,HEO,SSO,ES-L1

has

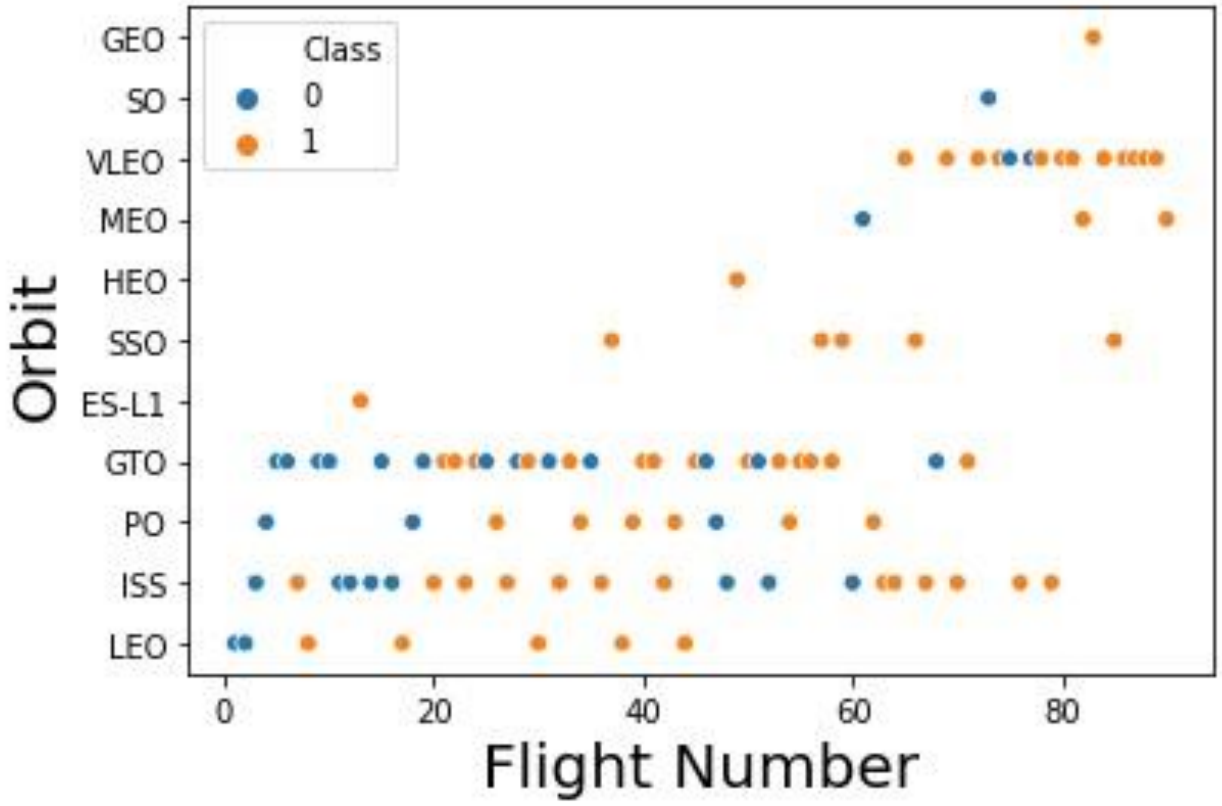
Best Success Rate.



2.4 Flight Number vs. Orbit Type

The success rate of **LEO orbit** appears related to the number of flights.

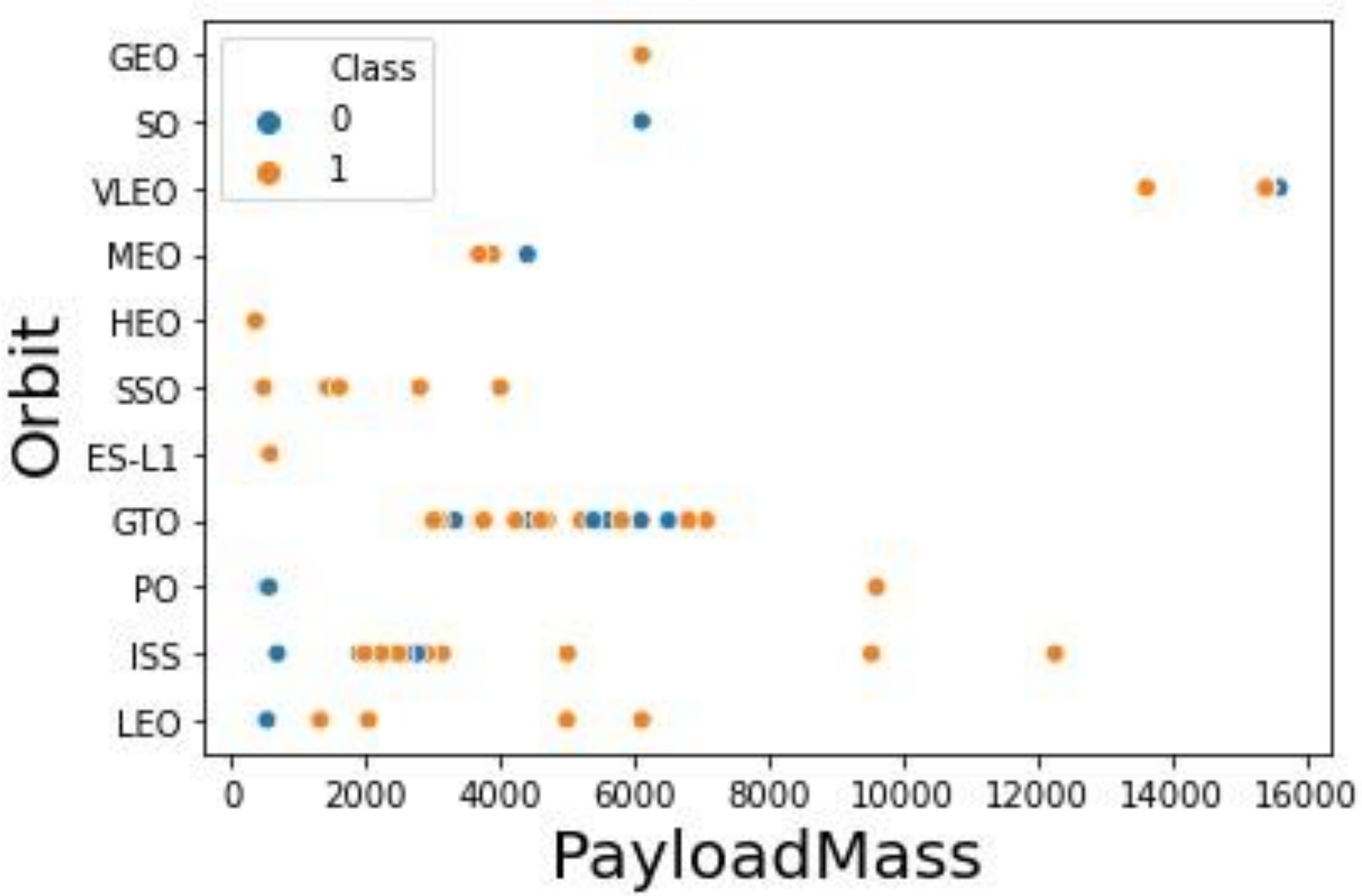
On the other hand, there seems to be no relationship between flight number when in **GTO orbit**.



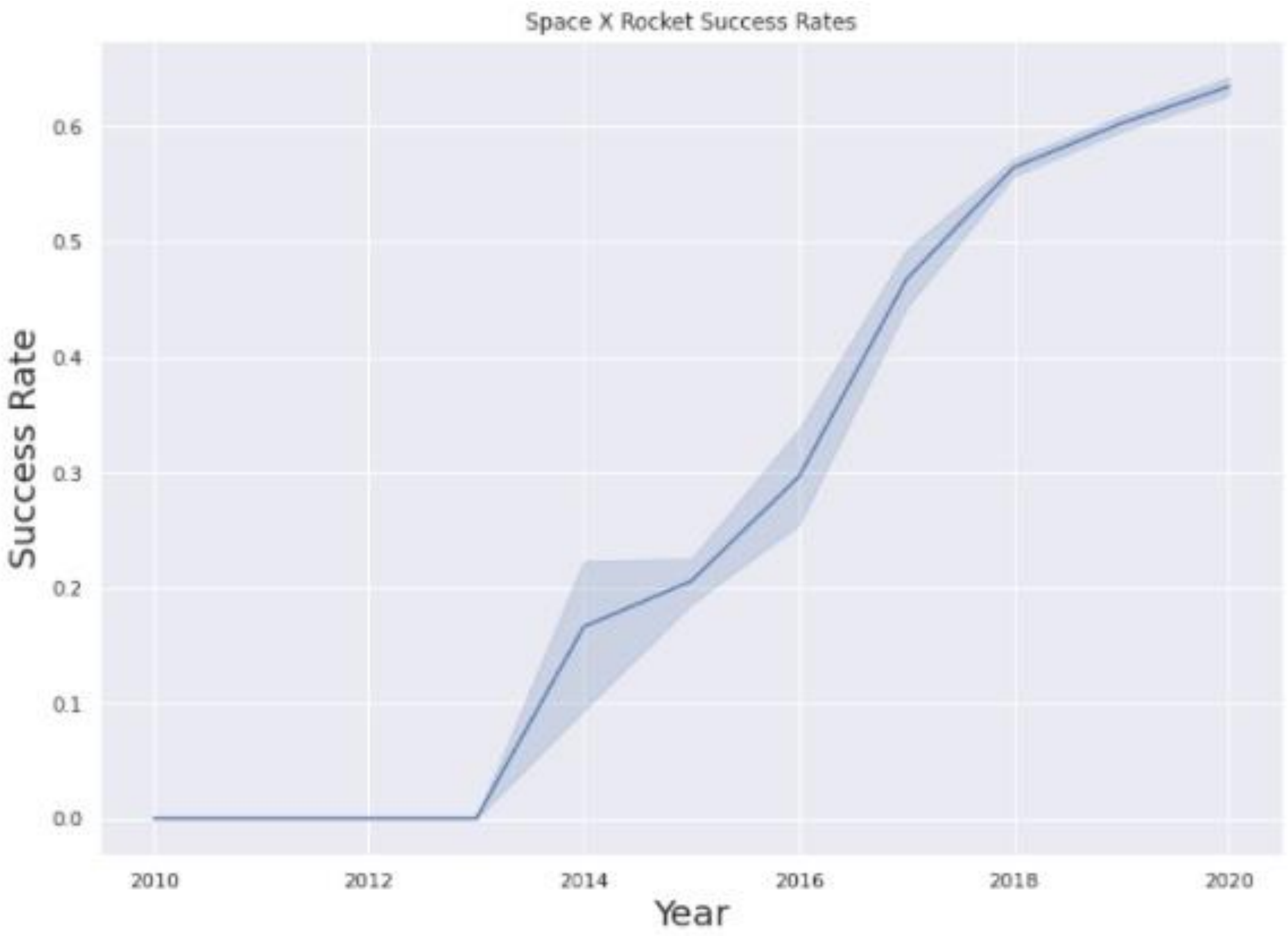
2.5 Payload vs. Orbit Type

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 showed.



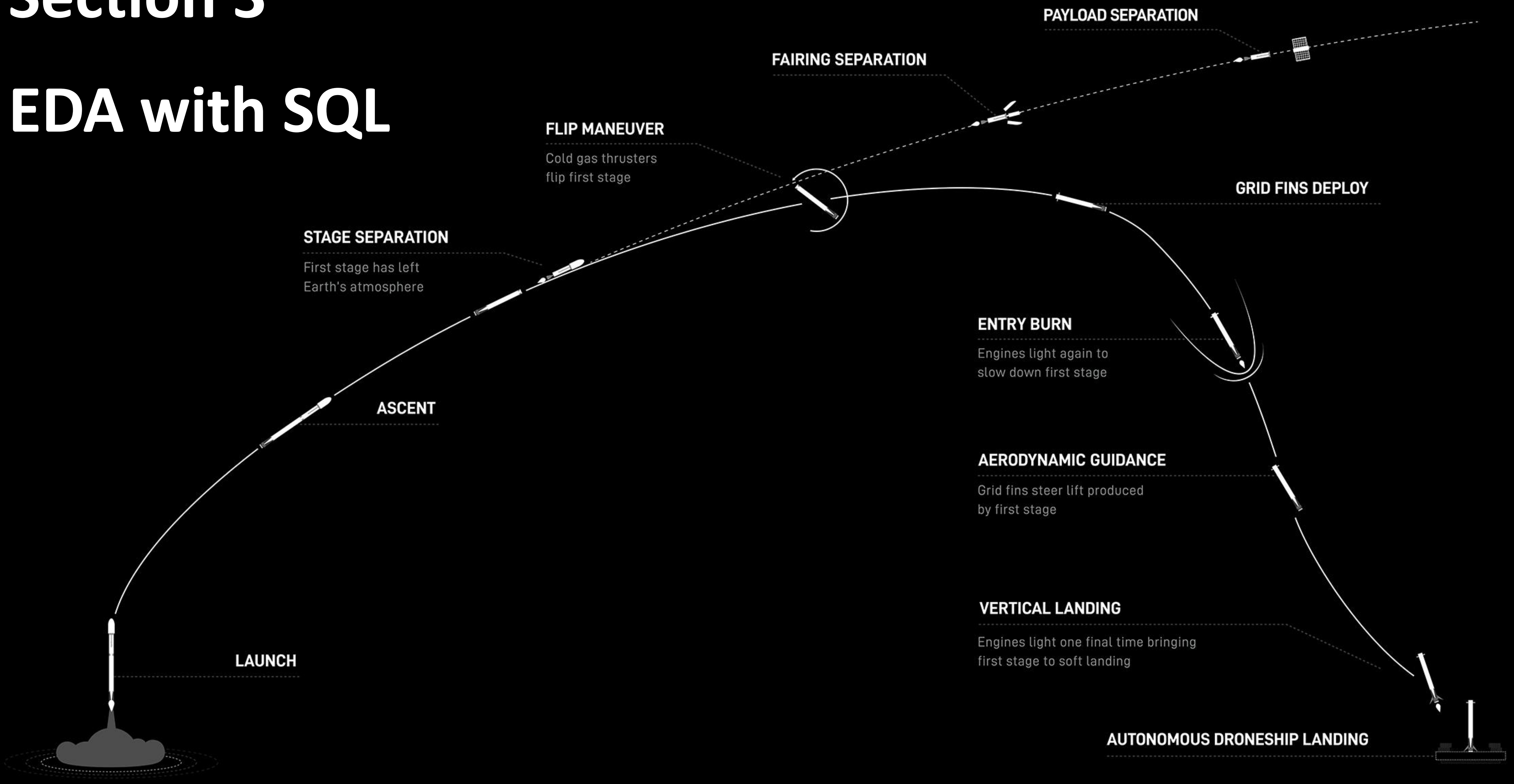
2.6 Launch Success Yearly Trend



The success rate since 2013 kept increasing till 2020.

Section 3

EDA with SQL



3.1 All Launch Site Names

Find the names of the unique launch sites

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

SQL Query

```
SELECT DISTINCT (LAUNCH_SITE) FROM SPACEXTBL;
```

SQL Explanation

Using ***DISTINCT*** query to show unique values in the ***Launch_Site*** column from ***tblSpaceX*** dataset

3.2 Launch Site Names Begin with 'CCA'

Find 5 records where launch sites begin with `CCA`

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

SQL Query

```
SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE
LIKE 'CCA%' limit 5;
```

SQL Explanation

Using Limit 5 in the query to show only first 5 values and the like keyword has a wild card with the words 'CCA%‘. The percentage in the end suggests that the Launch_Site name must start with CCA in the Launch_Site column from tblSpaceX dataset.

3.3 Total Payload Mass

Calculate the total payload carried by boosters from NASA

Total Payload Mass
45596

SQL Query

```
SELECT SUM (PAYLOAD_MASS_KG_) AS “TOTAL PAYLOAD MASS”
FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)‘;
```

SQL Explanation

Using ***SUM*** aggregate function to sum up the total values in the column ***PAYLOAD_MASS_KG_***.

Using ***WHERE*** clause to filter the dataset to show only the value equal to ***Customer NASA (CRS).***

3.4 Average Payload Mass by F9 v1.1

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

Average Payload Mass
2928

SQL Query

```
SELECT AVG (PAYLOAD_MASS_KG_) AS “Average Payload Mass”
FROM SPACEXTBL
WHERE BOOSTER_VERSION = 'F9 v1.1'
```

SQL Explanation

Using the ***AVG*** function to calculate the mean of the column ***PAYLOAD_MASS_KG_.***

Using ***WHERE*** clause to filter the dataset to show only the value equal to ***Booster_version F9 v1.1.***

3.5 First Successful Ground Landing Date

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

SQL Query

```
SELECT MIN (DATE) AS “First Successful Ground Landing Date”
FROM SPACEXTBL
WHERE Landing__Outcome = 'Success (ground pad)‘;
```

First Successful Ground Landing Date
2015-12-22

SQL Explanation

Using the ***MIN*** function to show the minimum date as ***Date which first Successful landing outcome in drone ship*** was achieved in the column ***Date***.

Using ***WHERE*** clause to filter the dataset to show only the value equal to ***‘Success (ground pad)’*** in the ***Landing_Outcome*** column.

3.6 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

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

SQL Query

```
%sql select BOOSTER_VERSION from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS_KG_ > 4000 and PAYLOAD_MASS_KG_ < 6000
```

SQL Explanation

Selecting only ***Booster_Version*** column.

Using ***WHERE*** clause to filter the dataset to show only the value equal to ***'Success (drone ship)'*** in the ***Landing_Outcome*** column.

The ***AND*** clause specifies additional filter conditions ***Payload_Mass_KG_>4000 AND Payload_Mass_KG_<6000.***

3.7 Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

SQL Query

```
SELECT COUNT (MISSION_OUTCOME) AS
"Mission Outcomes" FROM SPACEXTBL;
```

SQL Explanation

Using ***SELECT COUNT*** Statement to retrieve the total number of successful and failure mission outcomes.

Mission Outcomes
101

3.8 Boosters Carried Maximum Payload

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

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

SQL Query

```
SELECT BOOSTER_VERSION FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX (PAYLOAD_MASS_KG_)
FROM SPACEXTBL);
```

SQL Explanation

Using ***SELECT*** statement to retrieve data in the ***Booster_Version*** column from ***tblSpaceX*** dataset.

Using ***WHERE*** clause to filter the dataset to show only the maximum payload mass in the ***PAYLOAD_MASS_KG_*** column.

3.9 2015 Launch Records

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

1	mission_outcome	booster_version	launch_site
1	Success	F9 v1.1 B1012	CCAFS LC-40
2	Success	F9 v1.1 B1013	CCAFS LC-40
3	Success	F9 v1.1 B1014	CCAFS LC-40
4	Success	F9 v1.1 B1015	CCAFS LC-40
4	Success	F9 v1.1 B1016	CCAFS LC-40
6	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

SQL Query

SELECT DATE, BOOSTER_VERSION, Launch_SITE, LANDING__OUTCOME FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND YEAR(DATE) = '2015';

SQL Explanation

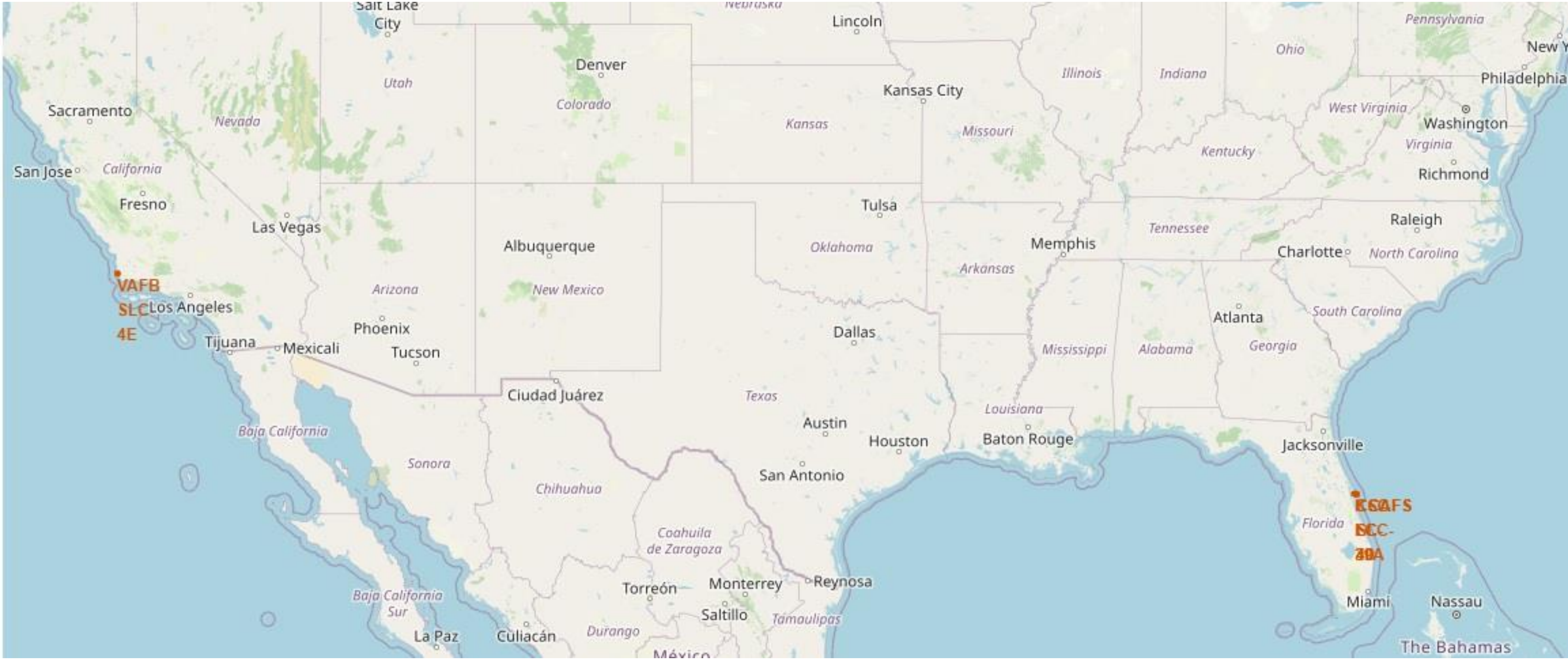
Using ***SELECT*** statement to show ***DATE as Month***, ***Booster_Version, Launch_Site, Landing_Outcome*** columns from ***tblSpaceX*** dataset.

Using ***WHERE*** clause to filter Date columns to show only the value equal to ***2015***.

Section 4

Launch Sites Proximities Analysis

4.1 All launch sites global map markers



All the SpaceX launch sites are situated in the coastal areas of Florida and California, USA.

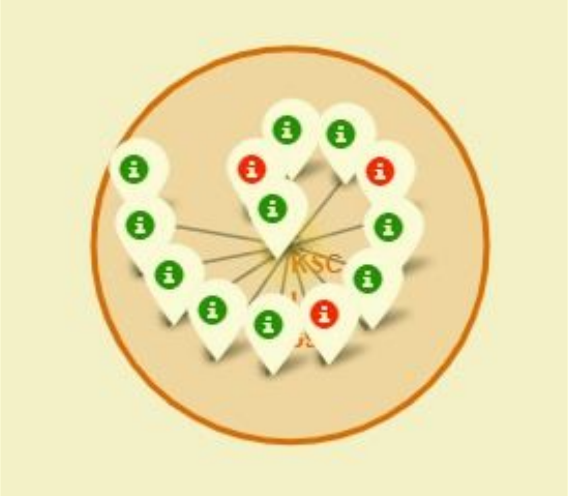
4.2 Color Markers Labelled the Launch Outcome for Each Site

California Launch Site (VAFB SLC-4E)

The color marker shows that there are 4 successful launches and 6 failed launches in this launch site.

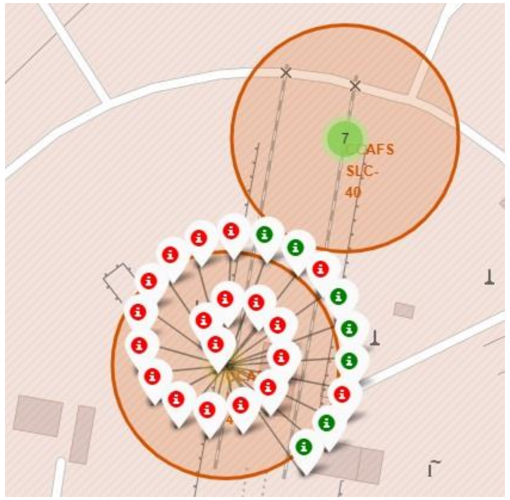
Florida Launch Site (KSC LC-39A)

The color marker shows that there are 10 successful launches and 3 failed launches in this launch site.



Florida Launch Site (CCAFS SLC-40)

The color marker shows that there are 3 successful launches and 4 failed launches in this launch site.



Florida Launch Site (CCAFS LC-40)

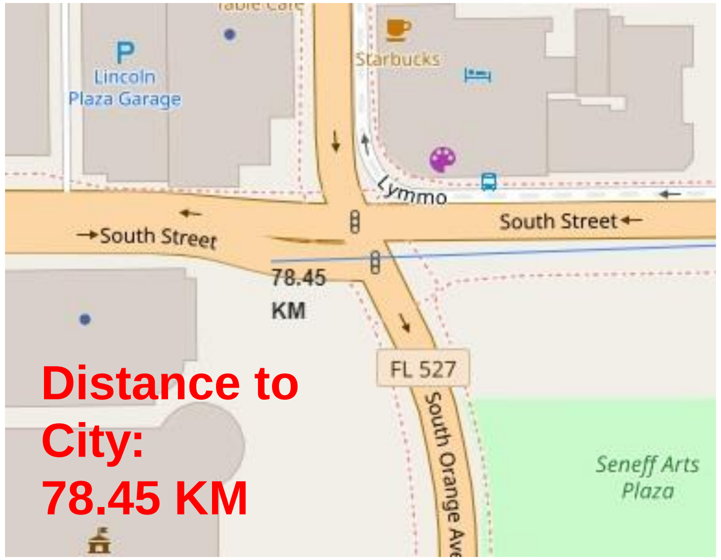
The color marker shows that there are 7 successful launches and 19 failed launches in this launch site.

Color Marker: GREEN for Sucessful
RED for Failed

Based on the color-labeled markers, we can easily identify **KSC LC-39A** have relatively high success rates.

3.3 Launch Sites Distance to Landmarks

CCAFS-SLC-40 as a reference



Trends

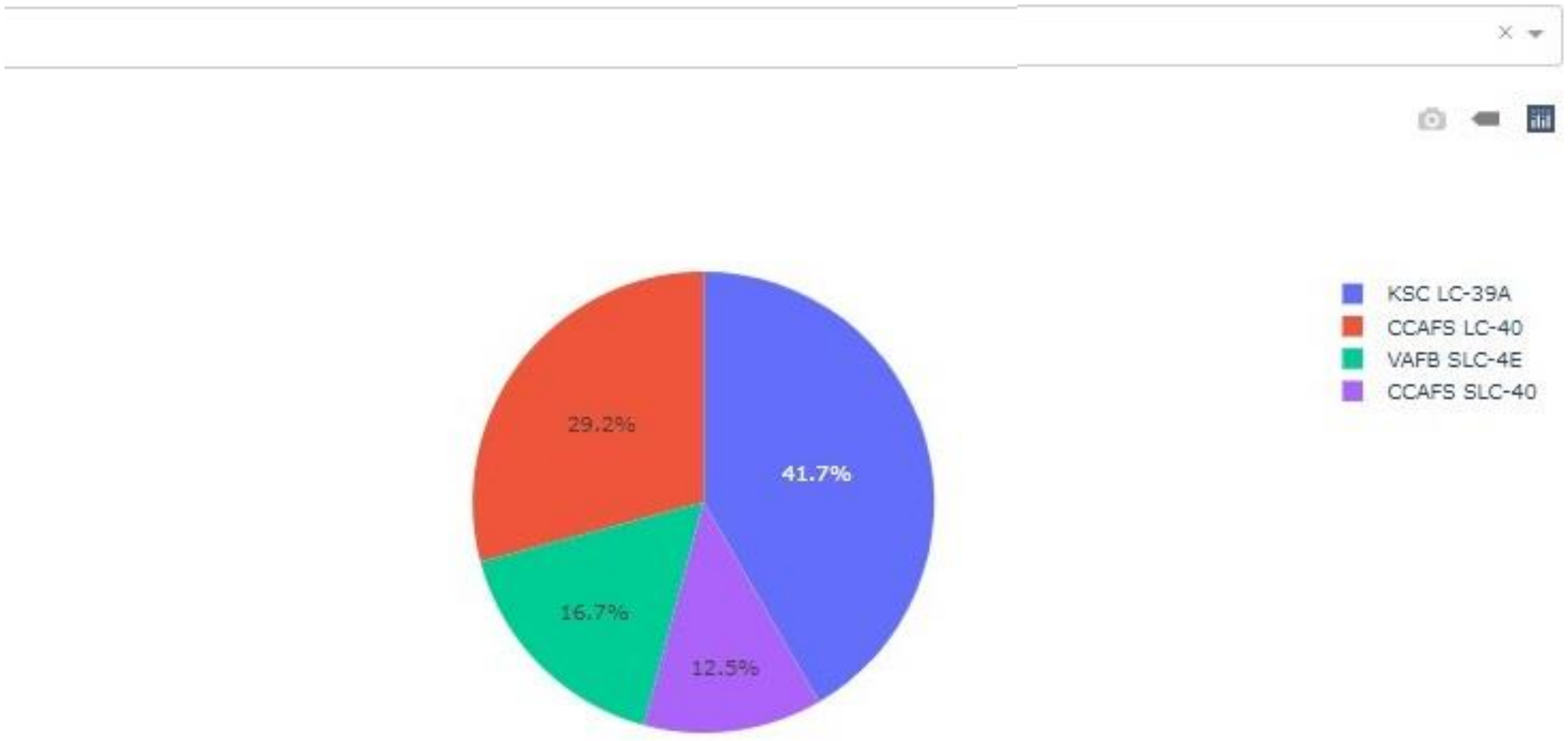
The Launch site is:

- Very close to railways
- Not close to highways
- Next to coastline
- Far away from cities

Section 5

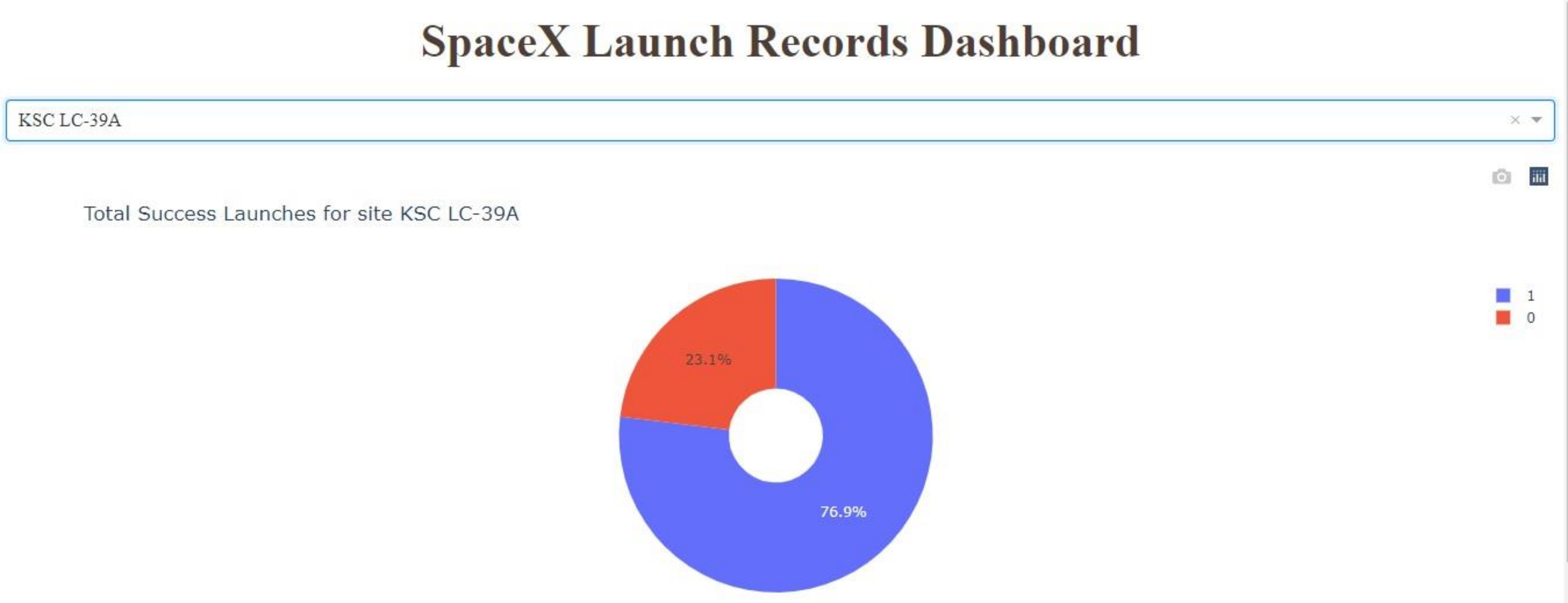
Build a Dashboard with Plotly Dash

5.1 Pie Chart Show the Launch Success Rate for All Sites



After visual analysis using the pie chart, we are able to obtain some insights that the KSC LC-39A has the largest successful launches (41.7%).

5.2 Pie Chart Show the Launch Success Rate for All Sites

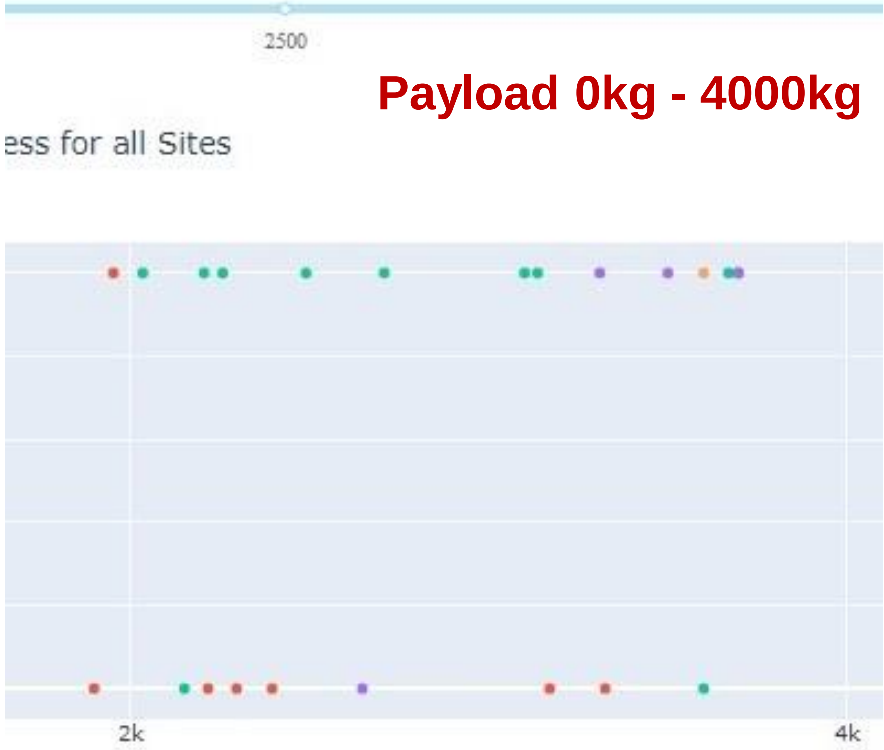


The pie chart shows that the KSC LC-39A has the highest launch success rate with 76.9% and failure rate with 23.1%.

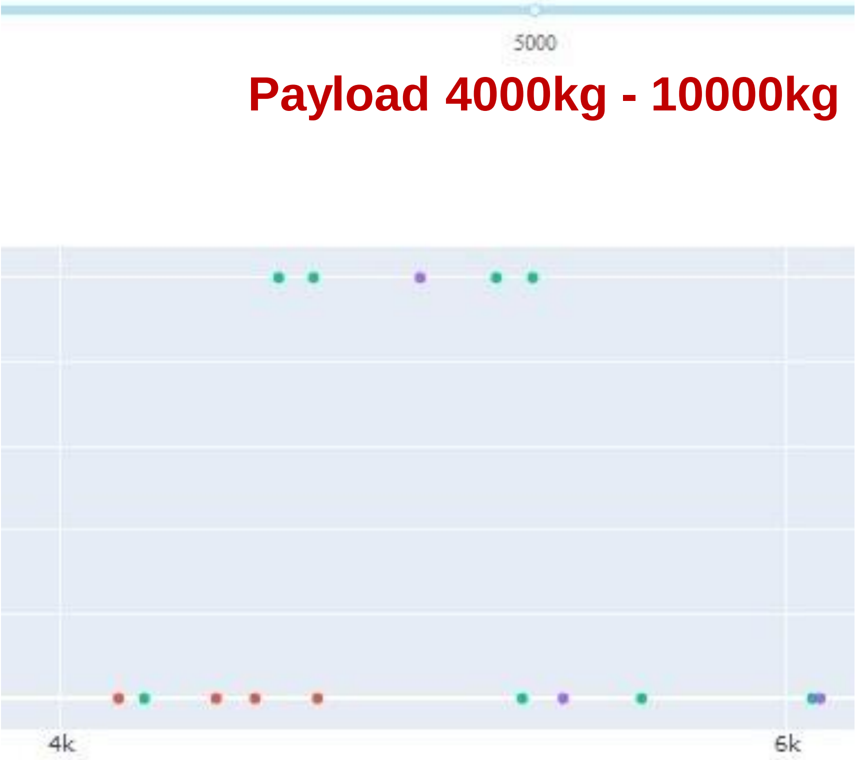
5.3 Payload vs Launch Outcome

Scatter plot shows the relationship between payload and launch outcome.

Payload 0kg - 4000kg



Payload 4000kg - 10000kg



The scatter plot shows that the success rate for low weighted payloads (below 4000kg) is greater that heavy weighted payloads.

Section 6

Predictive Analysis (Classification)

6.1 Classification Model Comparison

The classification accuracy using training data is very extremely close for 4 different Classifier Models: Logistic Regression, Support Vector Machine, Decision Tree and K Nearest Neighbor.

No	Classifier	Accuracy
1	Logistic Regression	84.64%
2	Support Vector Machine (SVM)	84.82%
3	Decision Tree	87.50%
4	K Nearest Neighbour	84.82%

Using **Best Algorithm** method, we can determine which model performs best

```
Best Algorithm is Tree with a score of 0.875
Best Params is : {'criterion': 'gini', 'max_depth': 14, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
```

Best Performer:
The Decision Tree

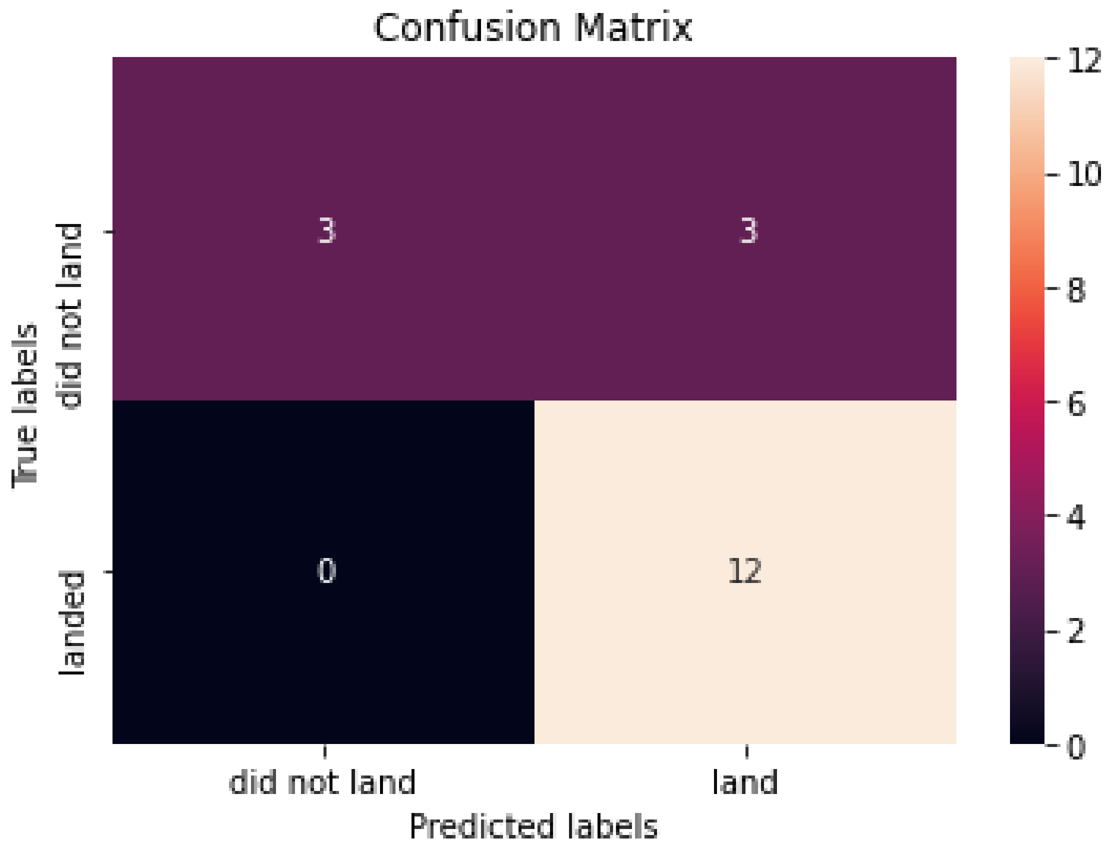
The accuracy of Decision Tree model on test data:

```
accuracy:  0.9444444444444444
```

6.2 Confusion Matrix

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class.

Examining the confusion matrix, we see that Decision Tree can distinguish between the different classes. We see that the major problem is false positives.



Conclusion

The success of any future Falcon 9 rocket launches will depend on the following factors:

Launch site : KSC LC-39A had the most successful launches from all the sites.

Payload mass : Low weighted payloads perform better than heavier payloads.

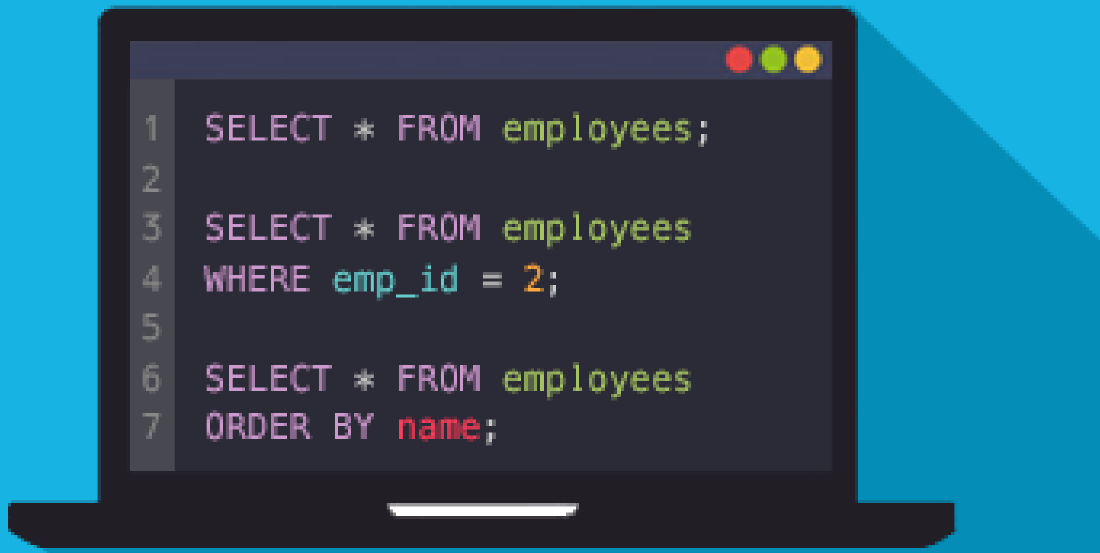
Booster version : F9 FT B1022, F9 FT B1026, F9 FT B1021.2, F9 FT B1031.2.

Orbit type : GEO, HEO, SSO, ES-L1 has the best Success Rate

Classifier: Decision Tree Model is the best Classifier to predict success of future launches for the supplied data.

Appendix

- SQL
- Plotly Dash
- Folium
- Github



SQL (Structured Query Language) is a standardized programming language that's used to manage relational databases and perform various operations on the data in them. Initially created in the 1970s, SQL is regularly used not only by database administrators, but also by developers writing data integration scripts and data analysts looking to set up and run analytical queries.

I am using IBM DB2 cloud service to store my data and using python library (sql magic) to access the data via python notebook. It allow me to ultilize the SQL to explore and manipulate data.

It will also allow me to convert some useful columns into a new dataframe easily and plot charts to visualize the data.

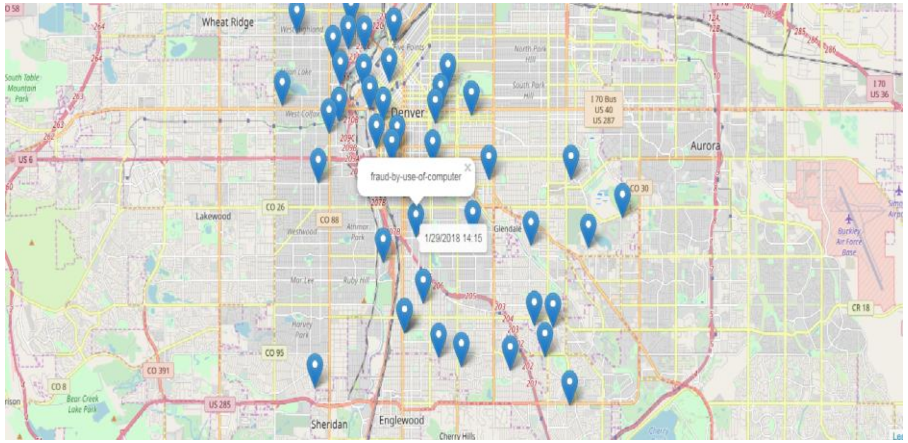
Dash is a python framework created by plotly for creating interactive web applications. Dash is written on the top of Flask, Plotly.js and React.js. With Dash, you don’t have to learn HTML, CSS and Javascript in order to create interactive dashboards, you only need python. Dash is open source and the application build using this framework are viewed on the web browser.

Building a Plotly Dash application will allow me to perform interactive visual analytics on SpaceX launch data in real-time. The dashboard application contains input components such as a dropdown list and a range slider to interact with charts, such as a pie chart and a scatter point chart.

Folium

Folium is a Python library used for visualizing geospatial data. It is easy to use and yet a powerful library.

Folium is a Python wrapper for Leaflet.js which is a leading open-source JavaScript library for plotting interactive maps.



It has the power of Leaflet.js and the simplicity of Python, which makes it an excellent tool for plotting maps.

Folium is designed with simplicity, performance, and usability in mind. It works efficiently, can be extended with a lot of plugins, has a beautiful and easy-to-use API.

Folium allows me to perform more interactive visual analytics to discover some preliminary correlations between Launch Sites, Success Rate and its Proximities.

cmlak / SpaceX-Falcon-9 Public

```
In [4]:
# Download and read the `spacex_launch_geo.csv`
spacex_csv_file = wget.download('https://cf-courses-data.s3.us.cloud-object-storage.appdo
spacex_df=pd.read_csv(spacex_csv_file)
```

Now, you can take a look at what are the coordinates for each site.

```
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
```

GitHub

GitHub is a code hosting platform for version control and collaboration. It lets us work together on projects from anywhere.

It allows us to create repositories, manage branches, make changes to files (as commits), and open and merge a pull request.

I had uploaded all the completed notebooks and Python files to my GitHub repository for reference.

URL: github.com/cmlak/SpaceX-Falcon-9


THANK YOU