IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Reem Alsubaie
2/19/2022

Outline

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

Executive Summary

- Summary of 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 all results

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

Introduction

- Project background and context

We predicted 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.

- Problems you want to find answers

- What influences if the rocket will land successfully?

- The effect each relationship with certain rocket variables will impact in determining the success rate of a successful landing.

- What conditions does SpaceX have to achieve to get the best results and ensure the best rocket success landing rate.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API
  - (Web Scrapping) from Wikipedia
- Perform data wrangling
  - One Hot Encoding data fields for Machine Learning and dropping irrelevant columns
- 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

- We worked with SpaceX launch data that is gathered from the SpaceX REST API.
- This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
- Our goal is to use this data to predict whether SpaceX will attempt to land a rocket or not.
- The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/.
- Another popular data source for obtaining Falcon 9 Launch data is web scraping Wikipedia using BeautifulSoup.

SpaceX API

1. Use SpaceX REST API
2. API returns SpaceX data in .JSON
3. Normalize data into flat data file such as .csv

Web Scrapping

1. Get HTML Response from Wikipedia
2. Extract data using beautiful soup
3. Normalize data into flat data file such as .csv

Data Collection – SpaceX API

Use SpaceX REST API

Filter DF for Falcon 9 only / Clean Data

API returns SpaceX data in .JSON

Normalize data into flat data file such as .csv

1 .Getting Response from API

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

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

2. Converting Response to a .json file

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

3. Apply custom functions to clean data

```
getBoosterVersion(data)
```

```
getLaunchSite(data)
```

```
getPayloadData(data)
```

```
getCoreData(data)
```

4. Assign list to dictionary then 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}
 data1= pd.DataFrame.from_dict(launch_dict)
```

5. Filter dataframe and export to flat file (.csv)

```
data_falcon9= data1[data1['BoosterVersion']!='Falcon 1']
```

```
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```


GitHub URL

Data Collection - Scraping

Get HTML Response from Wikipedia

Extract data using beautiful soup

Parse HTML table into a list dictionary

Normalize data into flat data file such as .csv

1 .Getting Response from HTML

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

2. Creating BeautifulSoup Object

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

3. Finding tables

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

4. Getting column names

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitab
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number correspo
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
```

5. Creation of dictionary

```
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']=[]
```

6. Appending data to keys (refer) to notebook block 12

```
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))
```

7. Converting dictionary to dataframe

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

8. Dataframe to .CSV

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

[GitHub URL](#)

Data Wrangling

In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad. True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship. We mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful

Perform Exploratory Data Analysis EDA on dataset

Calculate the number of launches at each site

Calculate the number and occurrence of each orbit

Calculate the number and occurrence of mission outcome per orbit type

Export dataset as .CSV

Create a landing outcome label from Outcome column

Work out success rate for every landing in dataset

GitHub URL

EDA with Data Visualization

Line Graph being drawn:

- Success Rate VS. Year

Line graphs are useful in that they show data variables and trends very clearly and can help to make predictions about the results of data not yet recorded

Bar Graph being drawn:

- Mean VS. Orbit

A bar diagram makes it easy to compare sets of data between different groups at a glance. The graph represents categories on one axis and a discrete value in the other. The goal is to show the relationship between the two axes. Bar charts can also show big changes in data over time.

Scatter Graphs being drawn:

- 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

Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation . Scatter plots usually consist of a large body of data.

GitHub URL

EDA with SQL

For example of some questions we were asked about the data we needed information about. Which we are using SQL queries to get the answers in the dataset :

- 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.

GitHub URL

Build an Interactive Map with Folium

To visualize the Launch Data into an interactive map. We took the Latitude and Longitude Coordinates at each launch site and added a Circle Marker around each launch site with a label of the name of the launch site.

We assigned the dataframe launch_outcomes(failures, successes) to classes 0 and 1 with Green and Red markers on the map in a MarkerCluster()

Using Haversine's formula we calculated the distance from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns. Lines are drawn on the map to measure distance to landmarks

Example of some trends in which the Launch Site is situated in.

•Are launch sites in close proximity to railways? No

•Are launch sites in close proximity to highways? No

•Are launch sites in close proximity to coastline? Yes

•Do launch sites keep certain distance away from cities? Yes

GitHub URL

Build a Dashboard with Plotly Dash

- Graphs

  - Pie Chart showing the total launches by a certain site/all sites

    - display relative proportions of multiple classes of data.

    - size of the circle can be made proportional to the total quantity it represents.

- Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions

  - It shows the relationship between two variables.

  - It is the best method to show you a non-linear pattern.

  - The range of data flow, i.e. maximum and minimum value, can be determined.

  - Observation and reading are straightforward.

Predictive Analysis (Classification)

BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

EVALUATING MODEL

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

IMPROVING MODEL

- Feature Engineering
- Algorithm Tuning

FINDING THE BEST PERFORMING CLASSIFICATION MODEL

- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

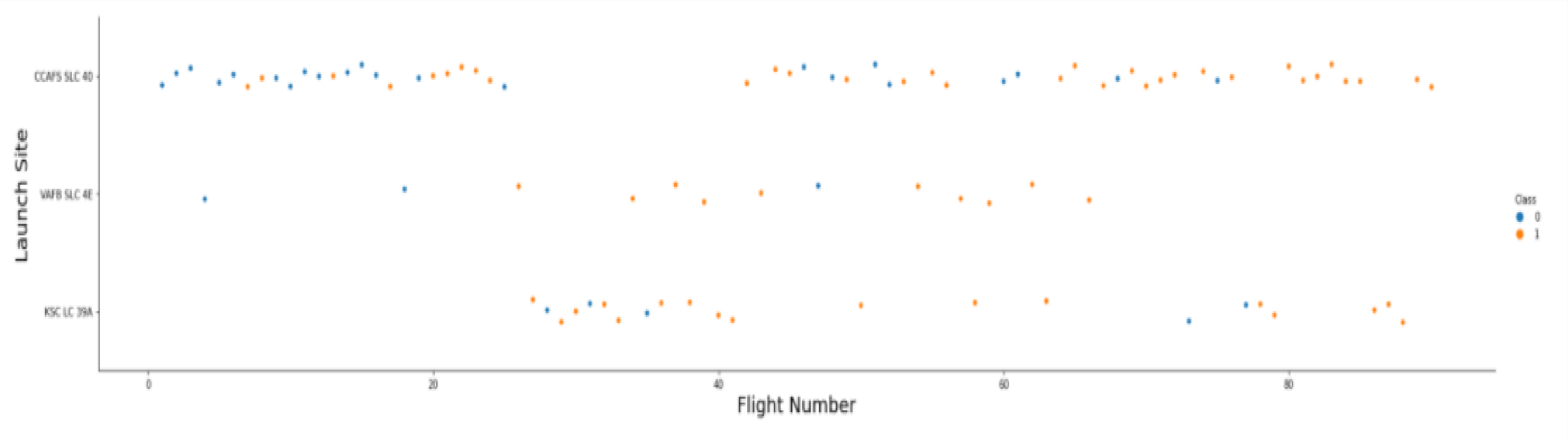
Results

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

Section 2

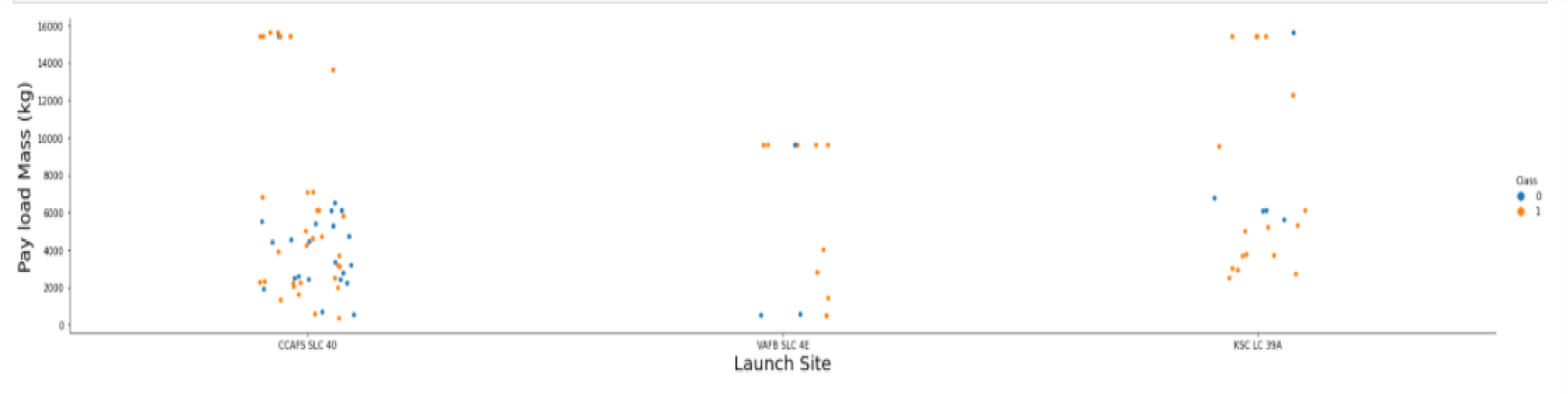
Insights drawn from EDA

Flight Number vs. Launch Site



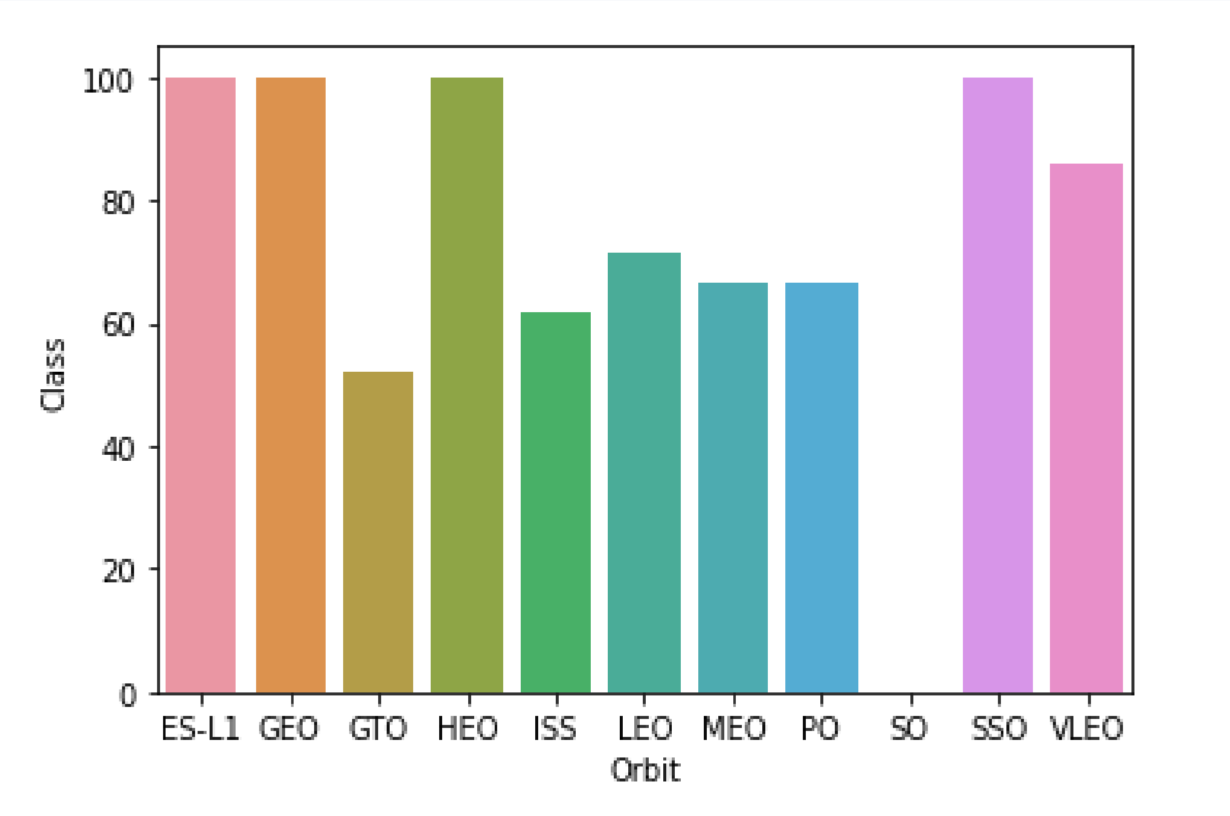
The more amount of flights at a launch site the greater the success rate at a launch site

Payload vs. Launch Site



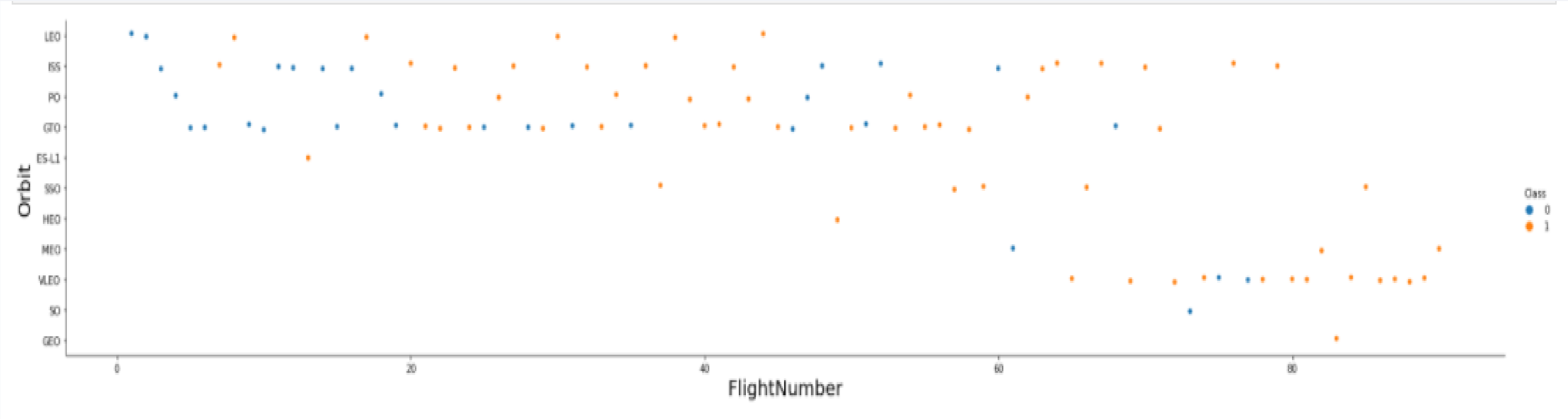
The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket. There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependant on Pay Load Mass for a success launch.

Success Rate vs. Orbit Type



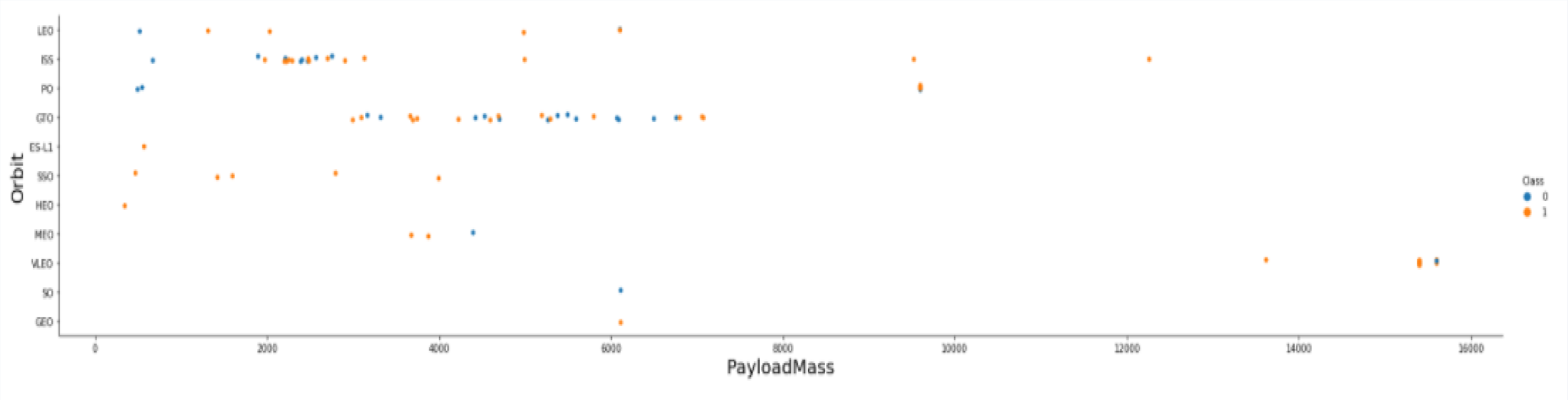
Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Flight Number vs. Orbit Type



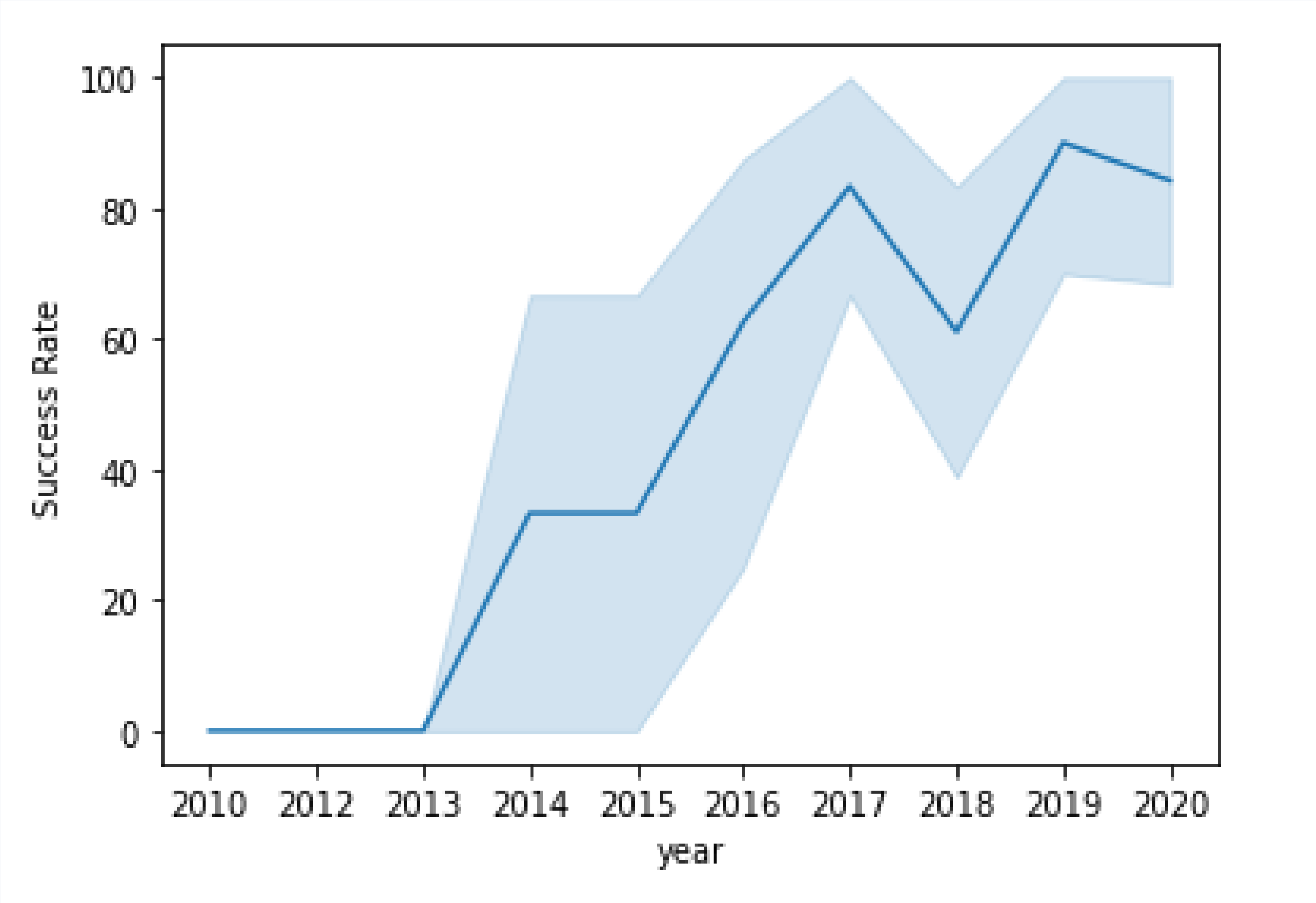
You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



You should observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

Launch Success Yearly Trend



you can observe that the success rate since 2013 kept increasing till 2020

All Launch Site Names

```
%sql select DISTINCT LAUNCH_SITE from SPACEXTBL
```

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

QUERY EXPLAINATION: Using the word **DISTINCT** in the query means that it will only show Unique values in the **Launch_Site** column from **tblSpaceX**

Launch Site Names Begin with 'CCA'

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

DATE	Time (UTC)	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	Landing _Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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)
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-12	22:41:00	F9 v1.1	CCAFS LC-40	SES-8	3170	GTO	SES	Success	No attempt

QUERY EXPLAINATION: Using the word TOP 5 in the query means that it will only show 5 records from **tblSpaceX** and **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.

Total Payload Mass

```
%sql select sum(payload_mass__kg_) as sum from SPACEXTBL where customer like 'NASA (CRS)'
```

SUM
22007

QUERY EXPLAINATION: Using the function **SUM** summates the total in the column **PAYLOAD_MASS_KG_** The **WHERE** clause filters the dataset to only perform calculations on Customer **NASA (CRS)**

Average Payload Mass by F9 v1.1

```
%sql select avg(payload_mass__kg_) as Average from SPACEXTBL where booster_version like 'F9 v1.1%'
```

average
3226

QUERY EXPLAINATION: Using the function **AVG** works out the average in the column **PAYLOAD_MASS_KG_** The **WHERE** clause filters the dataset to only perform calculations on **Booster_version F9 v1.1**

First Successful Ground Landing Date

```
%sql select min(date) as Date from SPACEXTBL where mission_outcome like 'Success'
```

DATE
2010-04-06

QUERY EXPLAINATION: Using the function **MIN** works out the minimum date in the column Date The **WHERE** clause filters the dataset to only perform calculations on **Landing_Outcome Success** (drone ship)

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select booster_version from SPACEXTBL where (mission_outcome like 'Success' ) AND (payload_mass__kg_ BETWEEN 4000 AND 6000) AND (landing__outcome like 'Success (drone ship)' )
```

F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1

QUERY EXPLAINATION: Selecting only **Booster_Version** The **WHERE** clause filters the dataset to **Landing_Outcome = Success (drone ship)** The **AND** clause specifies additional filter conditions **Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000**

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT mission_outcome, count(*) as Count FROM SPACEXTBL GROUP by mission_outcome ORDER BY mission_outcome
```

mission_outcome	COUNT
Success	44
Success (payload status unclear)	1

Boosters Carried Maximum Payload

```
maxm = %sql select max(payload_mass__kg_) from SPACEXTBL
maxv = maxm[0][0]

%sql select booster_version from SPACEXTBL where payload_mass__kg_=(select max(payload_mass__kg_) from SPACEXTBL)
```

booster_version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3

2015 Launch Records

```
%sql select MONTHNAME(DATE) as Month, landing__outcome, booster_version, launch_site from SPACEXTBL where DATE like '2015%' AND landing__outcome like 'Failure (drone ship)'
```

	Month	Booster_Version	Launch_Site	Landing_Outcome
0	January	F9 FT B1029.1	VAFB SLC-4E	Success (drone ship)
1	February	F9 FT B1031.1	KSC LC-39A	Success (ground pad)
2	March	F9 FT B1021.2	KSC LC-39A	Success (drone ship)
3	May	F9 FT B1032.1	KSC LC-39A	Success (ground pad)
4	June	F9 FT B1035.1	KSC LC-39A	Success (ground pad)
5	June	F9 FT B1029.2	KSC LC-39A	Success (drone ship)
6	June	F9 FT B1036.1	VAFB SLC-4E	Success (drone ship)
7	August	F9 B4 B1039.1	KSC LC-39A	Success (ground pad)
8	August	F9 FT B1038.1	VAFB SLC-4E	Success (drone ship)
9	September	F9 B4 B1040.1	KSC LC-39A	Success (ground pad)
10	October	F9 B4 B1041.1	VAFB SLC-4E	Success (drone ship)
11	October	F9 FT B1031.2	KSC LC-39A	Success (drone ship)
12	October	F9 B4 B1042.1	KSC LC-39A	Success (drone ship)
13	December	F9 FT B1035.2	CCAFS SLC-40	Success (ground pad)

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

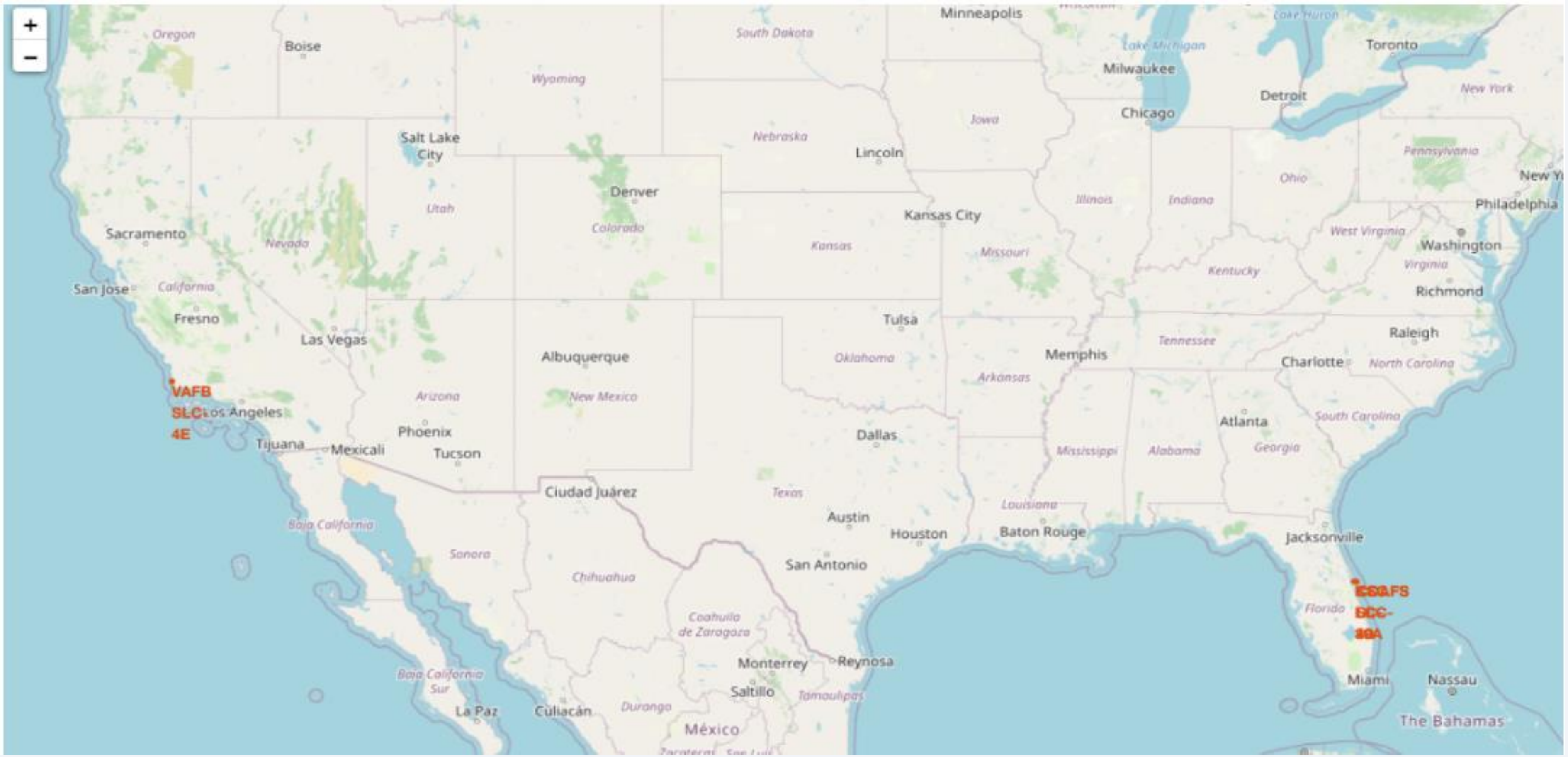
```
%sql select landing__outcome, count(*) as count from SPACEXTBL where Date >= '2010-06-04' AND Date <= '2017-03-20' GROUP by landing__outcome ORDER BY c
```

34

Section 4

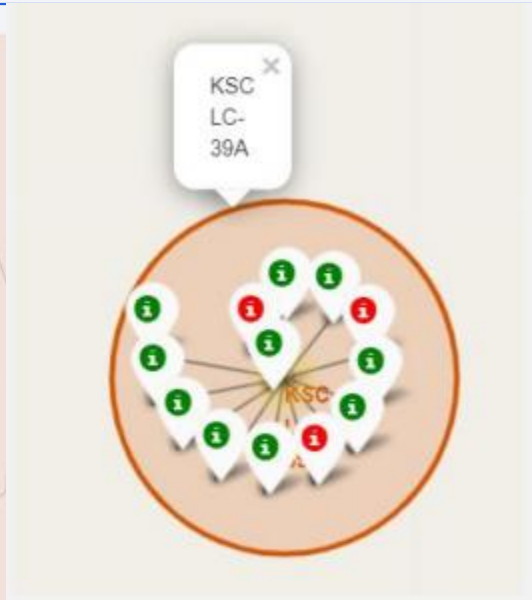
Launch Sites Proximities Analysis

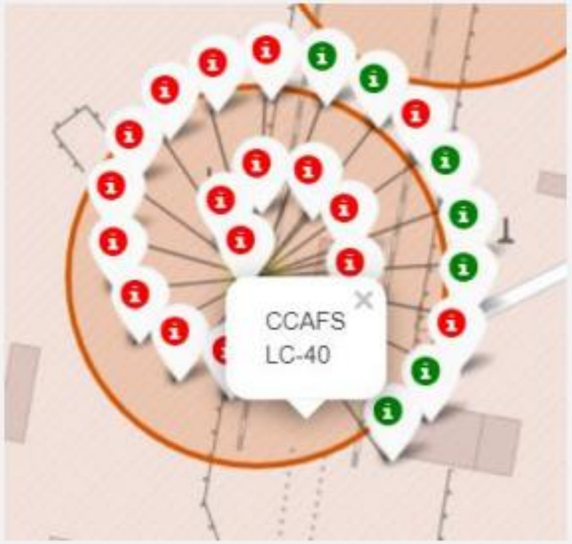
Mark all launch sites on a map

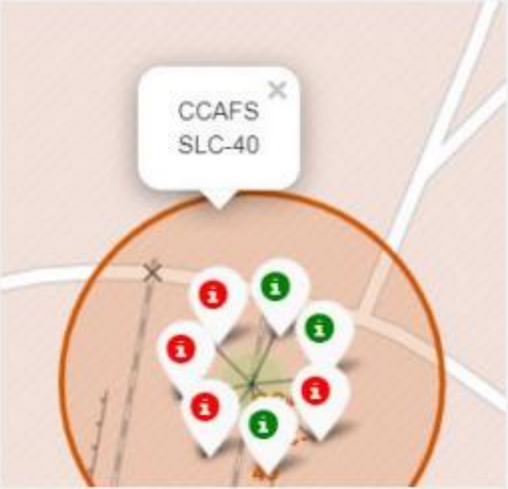


We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California

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







Florida Launch Sites

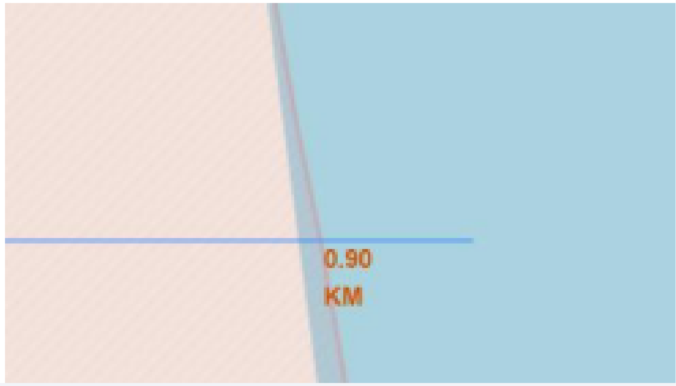
California Launch Site

Green Marker shows successful Launches
and Red Marker shows Failures

Calculate the distances between a launch site to its proximities

Distance to coast

Distance to Railway Station



Distance to Coastline

Distance to closest Highway

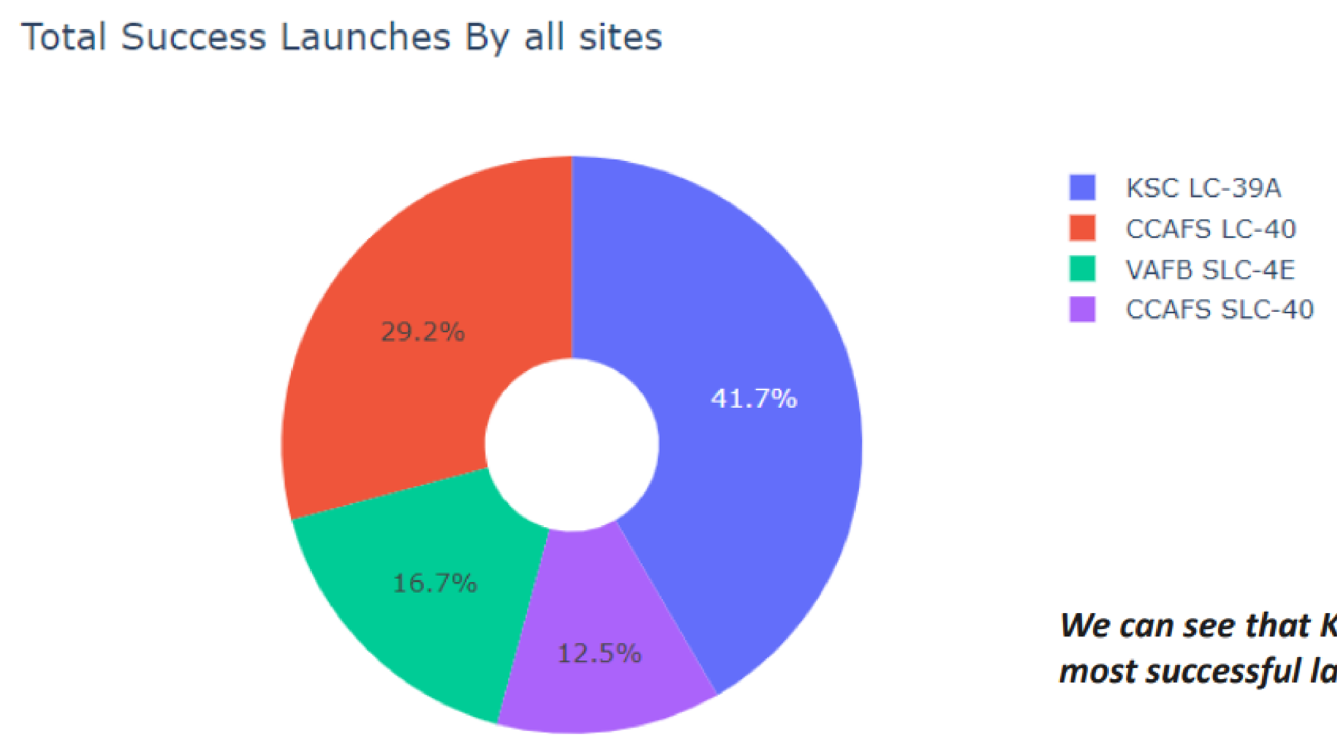
Distance to City

- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 5

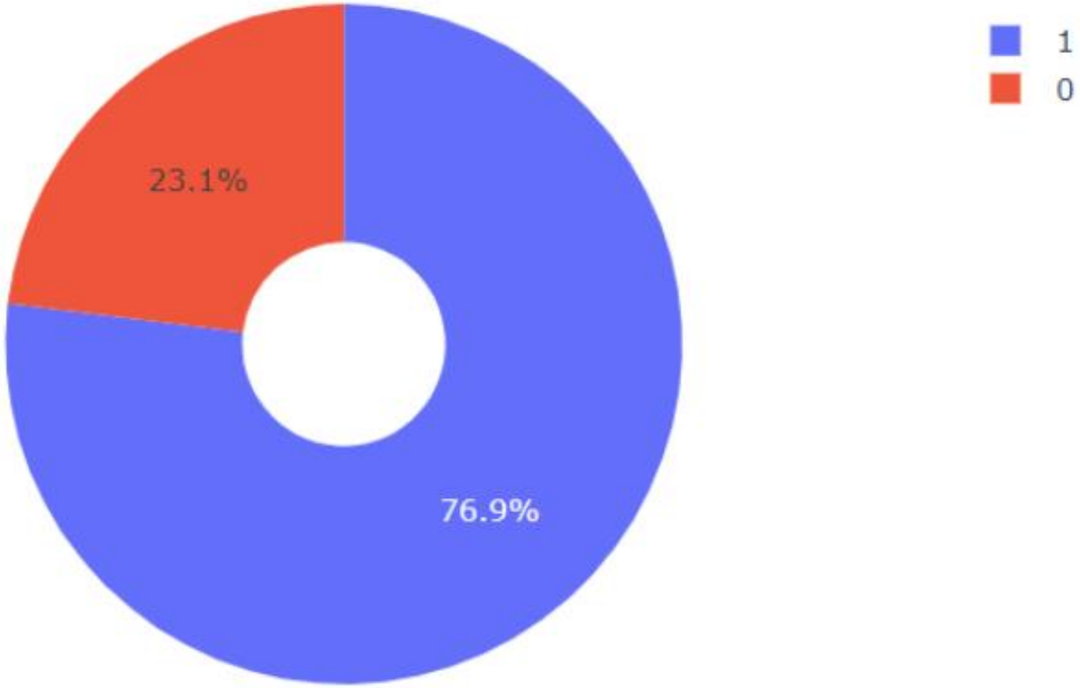
Build a Dashboard with Plotly Dash

DASHBOARD – Pie chart showing the success percentage achieved by each launch site



We can see that KSC LC-39A had the most successful launches from all the sites

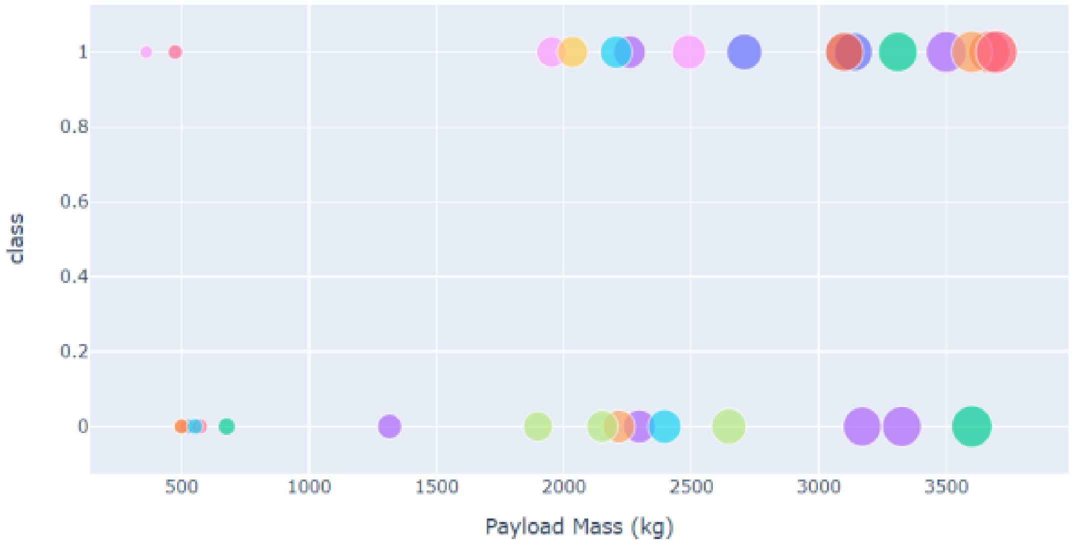
DASHBOARD – Pie chart for the launch site with highest launch success ratio



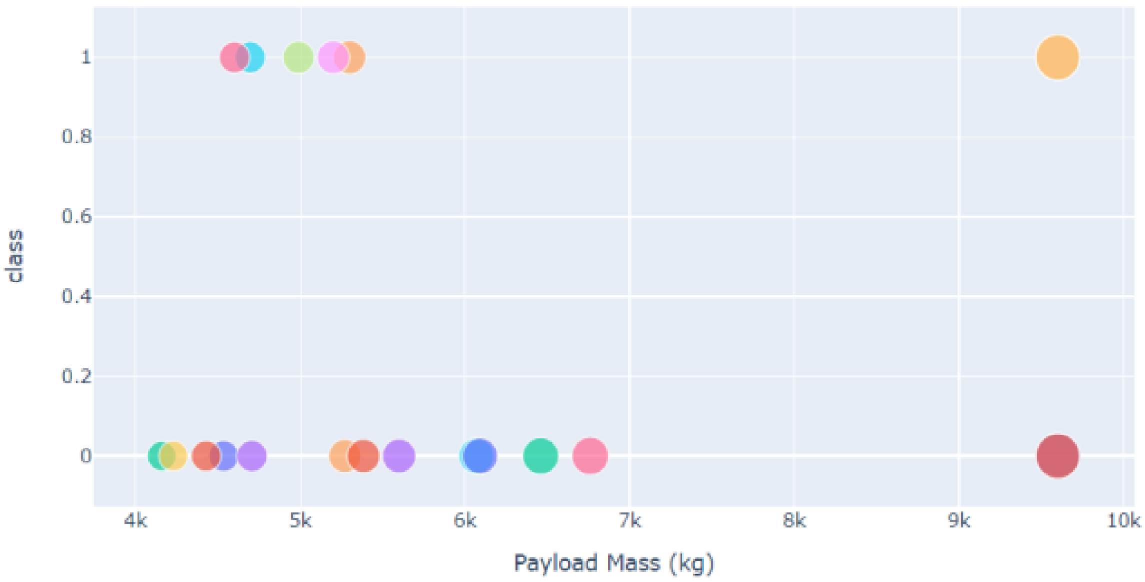
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

DASHBOARD – Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg

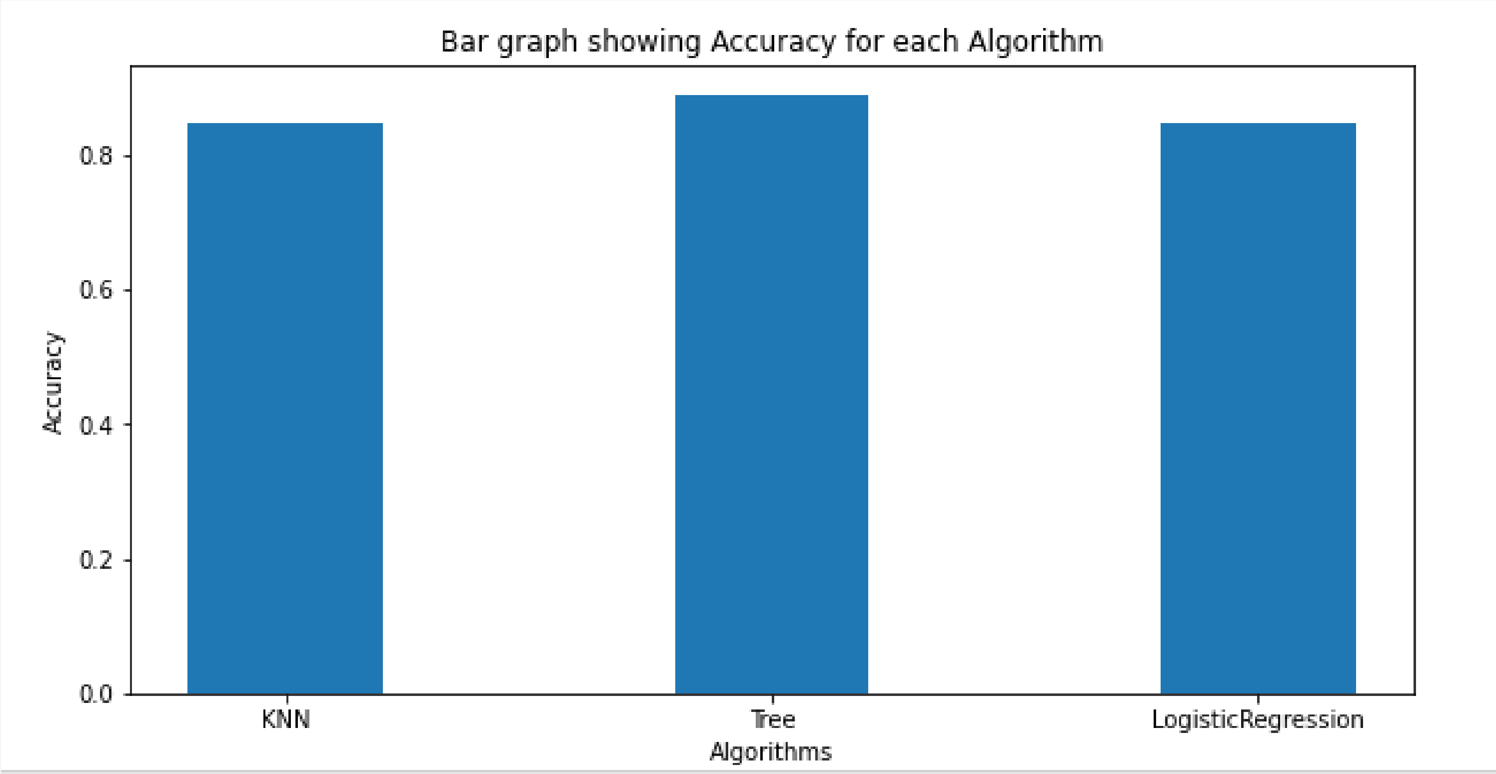


We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Section 6

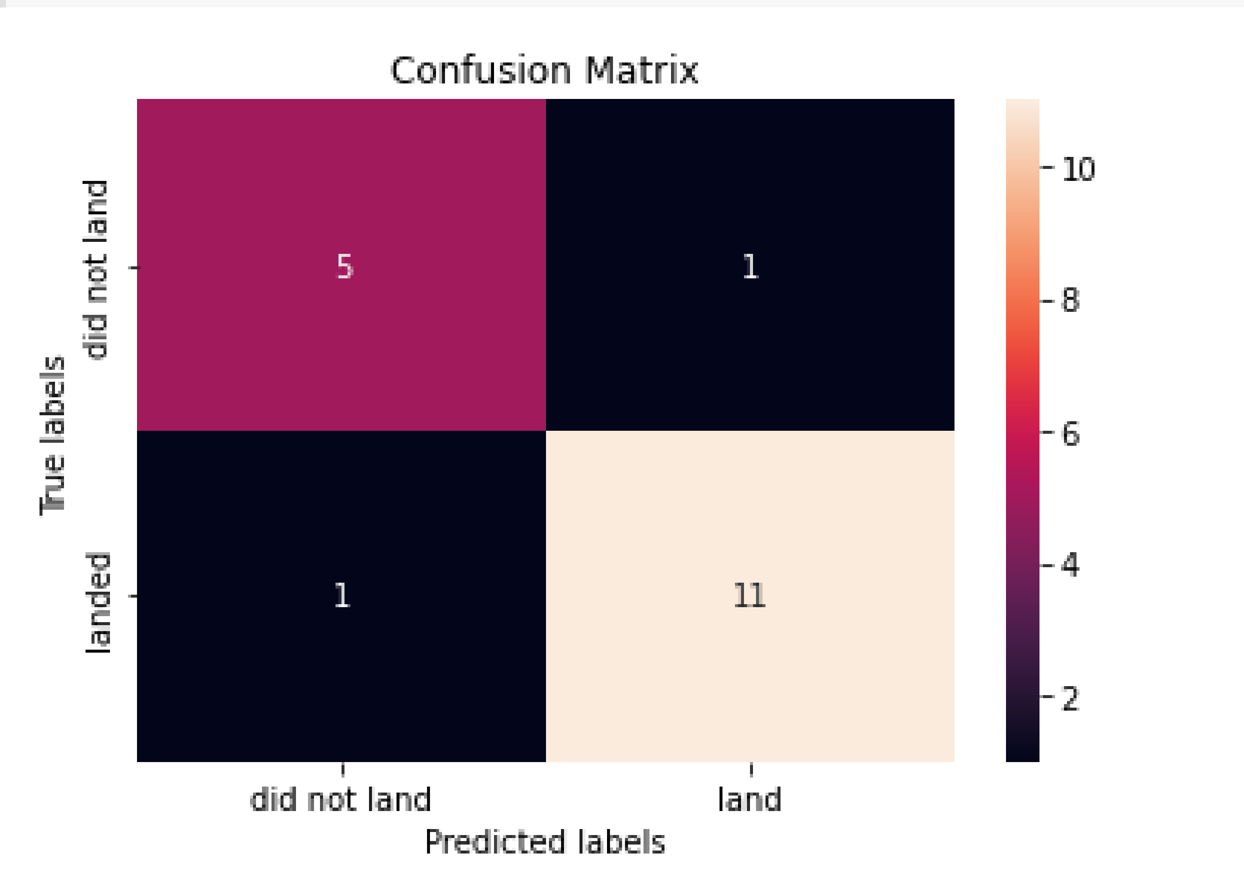
Predictive Analysis (Classification)

Classification Accuracy




```
Best Algorithm is Tree with a score of 0.875
Best Params is : {'criterion': 'entropy', 'max_depth': 2, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitter': 'random'}
```

Confusion Matrix



For Tree Algorithm

Conclusions

- The Tree Classifier Algorithm is the best for Machine Learning for this dataset
- Low weighted payloads perform better than the heavier payloads
- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
- We can see that KSC LC-39A had the most successful launches from all the sites
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Appendix

- Haversine formula
- ADGGoogleMaps Module (not used but created)
- Module sqlserver (ADGSQLSERVER)
- PythonAnywhere 24/7 dashboard

Thank you!