IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Imen KADRI
10/05/2022

Outline

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

Executive Summary

➢ **Summary of methodologies**

- Data collection :
  - from an API
  - with web scraping
- Data wrangling
- Exploratory data Analysis using SQL
- Exploratory data analysis with visualization
- Interactive data visualization :
  - with folium
  - by building a dashboard
- Prediction using machine learning classification

➢ **Summary of all results**

- Results of Exploratory Data Analysis
- Screenshots of the dashboard used for the interactive
- Results of thepPredictive Analytics

Introduction

➢ Background and context of the project

- Space X 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 Space X 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 space X for a rocket launch.
- **Goal :**

-Determine the price of each launch
-Determine if Space X will reuse the first stage
-Create a machine learning model to predict if the first stage will land.

➢ Problems to find answers

- What elements can determine if the rocket will land successfully?
- The interaction amongst various features that determine the success rate of a successful landing.
- What conditions ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected from the API of SpaceX and Wikipedia website
- Perform data wrangling
  - Find some patterns in the data and determine what would be the label for training supervised models.
  - Categorical features are transformed using one-hot encoding
- 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

- Description of datasets collection:
  - Data is collected using get request to the SpaceX API.
  - Then , the response content is decoded as a Json using .json() function call and transformed to a pandas dataframe using .json_normalize().
  - Next, the data is cleaned, checked for missing values and fill in missing values if necessary.
  - In addition, web scraping from Wikipedia for Falcon 9 launch records is performed using BeautifulSoup.
  - The goal was to convert the launch records to a pandas dataframe for future analysis. After extracting him as HTML table and parsing the table

Data Collection – SpaceX API

- Use of the get request to the SpaceX API in order to collect data, clean the requested data and did some basic data wrangling and formatting.
- Github url to the notebook:

  https://github.com/Imen2029/Applieddatascience-capstone-project/blob/1d75f27c74dab5d73cd1d70a8eb6b2449b822d9e/jupyter-labs-spacex-data-collection-api%20(2)%20(1).ipynb

Now let's start requesting rocket launch data from SpaceX API with the following URL:

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

```
[7]: response = requests.get(spacex_url)
```

Check the content of the response

```
[28]: #print(response.content)
```

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
[9]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-Skill
```

Data Collection - Scraping

- Web scrap Falcon 9 launch records with BeautifulSoup:

  - Extract a Falcon 9 launch records HTML table from Wikipedia

  - Parse the table and convert it into a Pandas data frame

- Github url to the notebook:

  https://github.com/Imen2029/Applied-datascience-capstone-project/blob/4a8e57e9850c725c7cac38924bafd7f16aaaf3a7/jupyter-labs-webscraping%20(2).ipynb

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
6]: # use requests.get() method with the provided static_url
response = requests.request(method='GET', url=static_url)
# assign the response to a object
```

Create a `BeautifulSoup` object from the HTML `response`

```
7]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text, "html.parser")

#print(soup.prettify())
```

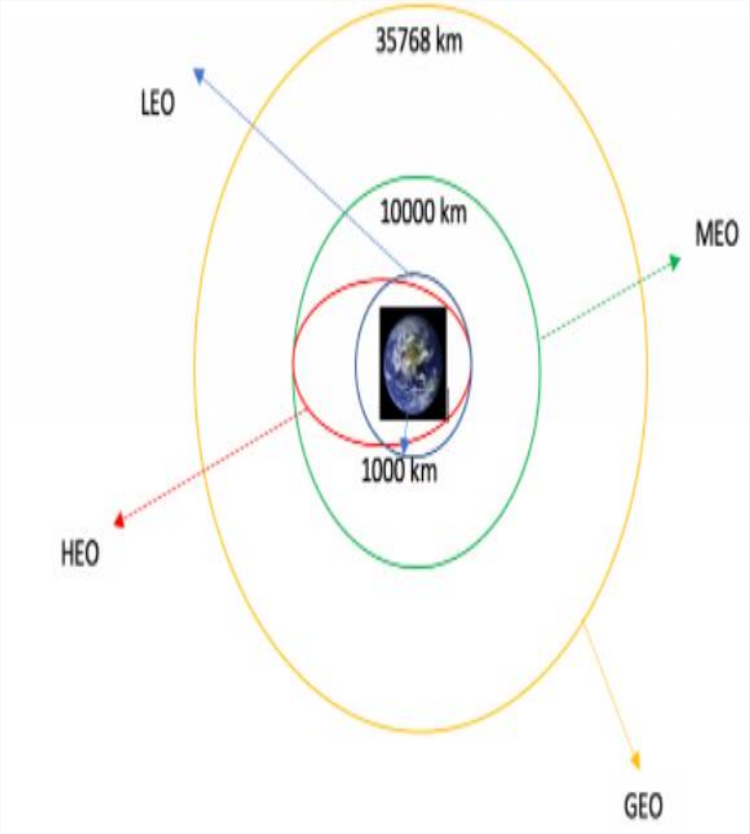
Print the page title to verify if the `BeautifulSoup` object was created properly

```
8]: # Use soup.title attribute
print(soup.title)
```

Data Wrangling

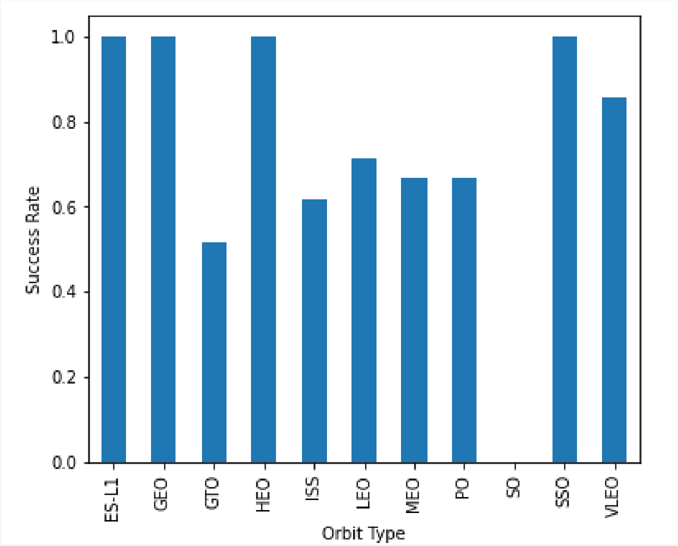
- Exploratory data analysis and determination of the training labels
- Calculation of **the number of launches on each site**
- **Calculation of the number and occurrence of each orbit**
- **Calculationof the number and occurence of mission outcome per orbit type**
- **Creation of a landing outcome label from Outcome column**
- Github url to the notebook:

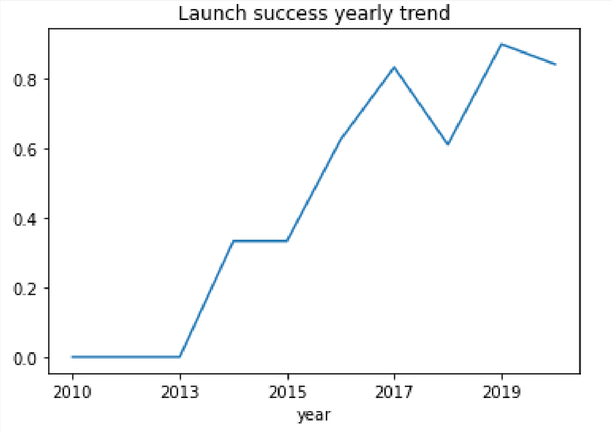
  https://github.com/Imen2029/Applied-datascience-capstone-project/blob/d5eb53f604948a51606be5dbca78e358aba437e2/labs-jupyter-spacex-Data%20wrangling.ipynb

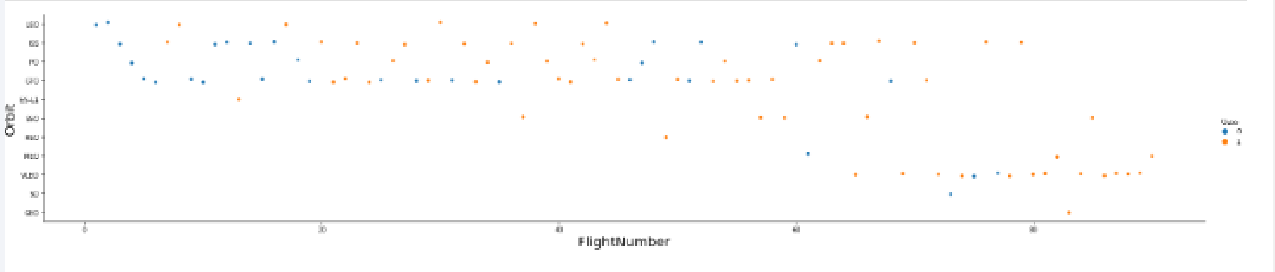


EDA with Data Visualization

- Graphs were for:
  -Visualize the relationship between Flight Number and Launch Site
  -Visualize the relationship between Payload and Launch Site
  -Visualize the relationship between success rate of each orbit type
  - Visualize the relationship between FlightNumber and Orbit type
  - Visualize the relationship between Payload and Orbit type
  -Visualize the launch success yearly trend
- Github url for this notebook:
https://github.com/Imen2029/Applied-datascience-capstone-project/blob/fda61c06b322245750132cd30cbe28613f81861f/edaviz.ipynb







EDA with SQL

- The SpaceX dataset is loaded into a SQL Lite database instead of DB2 as demanded in the original notebook
- Queries are written to find out for instance:

  -The names of unique launch sites in the space mission.

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

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

  -The total number of successful and failure mission outcomes

  -The failed landing outcomes in drone ship, their booster version and launch site names.
- Github url to the notebook:

  https://github.com/Imen2029/Applied-datascience-capstone-project/blob/21c2b14dfd73216f275cfee0ce7b6376841d2174/edasql.ipynb

Build an Interactive Map with Folium

- All launch sites are marked and map objects such as markers, circles, lines are added to mark the success or failure of launches for each site on the folium map.
- Assignment of the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Identifying which launch sites have relatively high success rate using the color-labeled marker clusters
- The distances between a launch site to its proximities is calculated to answer:

  - Are launch sites near railways, highways and coastlines.

  -Do launch sites keep certain distance away from cities.
- GitHub url to this notebook:

https://github.com/Imen2029/Applied-datascience-capstone-project/blob/92a9c70447b3477b0347836652da542724086d9b/launch_site_location_folium.ipynb

Build a Dashboard with Plotly Dash

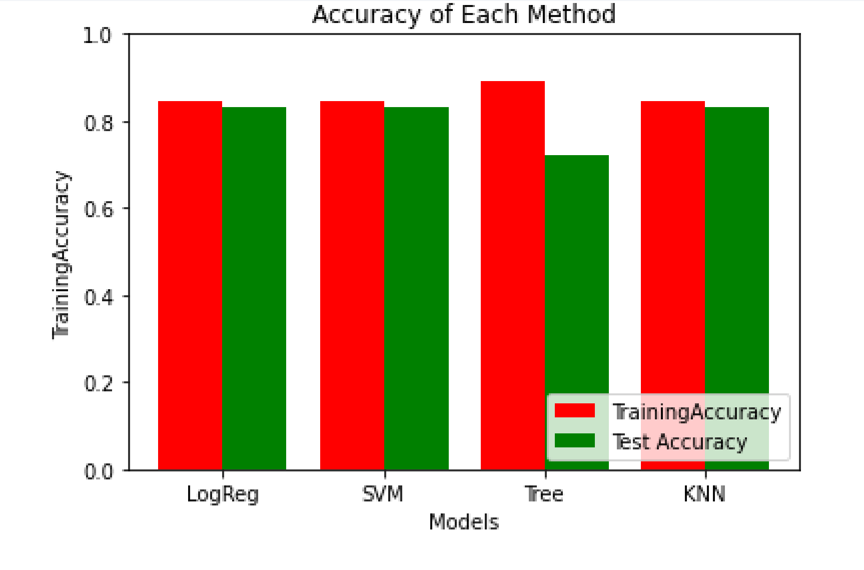
- An interactive dashboard with Plotly dash is built
- Pie charts showing the total launches by a certain sites are plotted
- Scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version are We plotted
- Github url to the notebook :

https://github.com/Imen2029/Applied-datascience-capstone-project/blob/ddac0c2eed78aa39806c7fb987535bf15b97f3ff/spacex_dash_app.py.py

Predictive Analysis (Classification)

- The data is loaded using numpy and pandas, transformed and splitted into training and testing.
- Different machine learning models are builit and different hyperparameters are tuned using GridSearchCV.
- Accuracy is used as the metric for our models
- Best performing classification model are found.
- Github url to this notebook :

https://github.com/Imen2029/Applied-datascience-capstone-project/blob/f8ffc358e92c3f1acc1f1bf3e8baa67226252cdc/Machine%20Learning%20Prediction_Part_5.ipynb



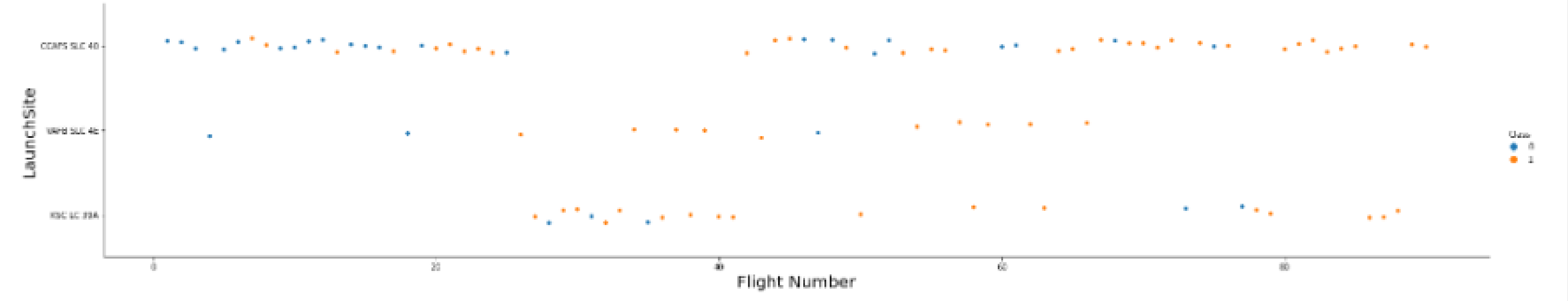
Results

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

Section 2

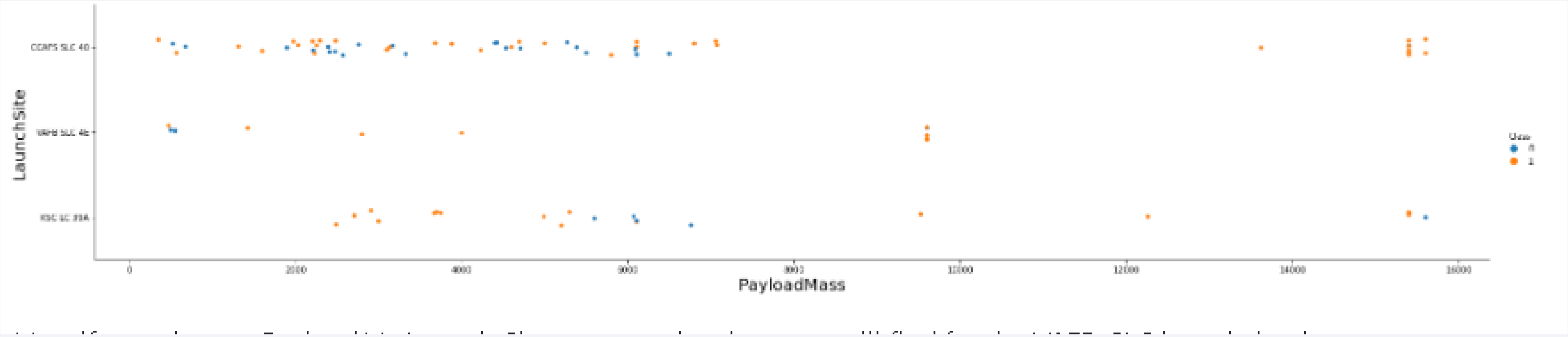
Insights drawn from EDA

Flight Number vs. Launch Site



- At a launch site., the larger the flight amount, the greater the success rate .

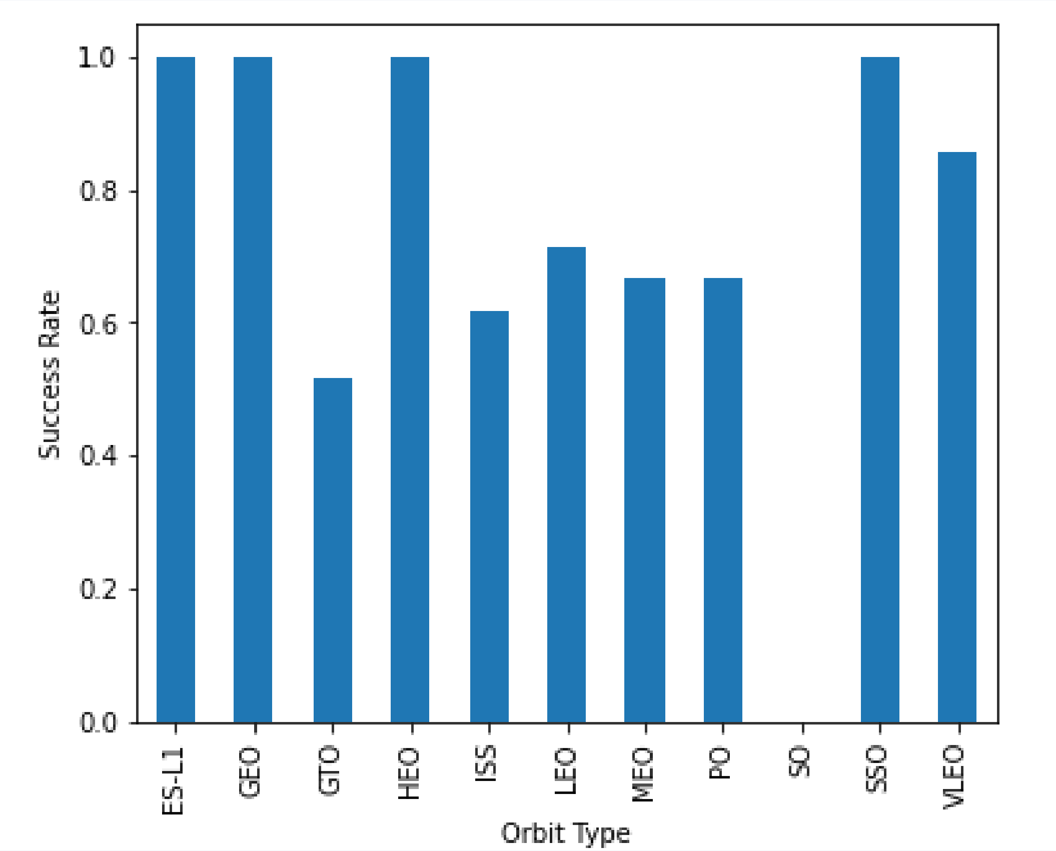
Payload vs. Launch Site



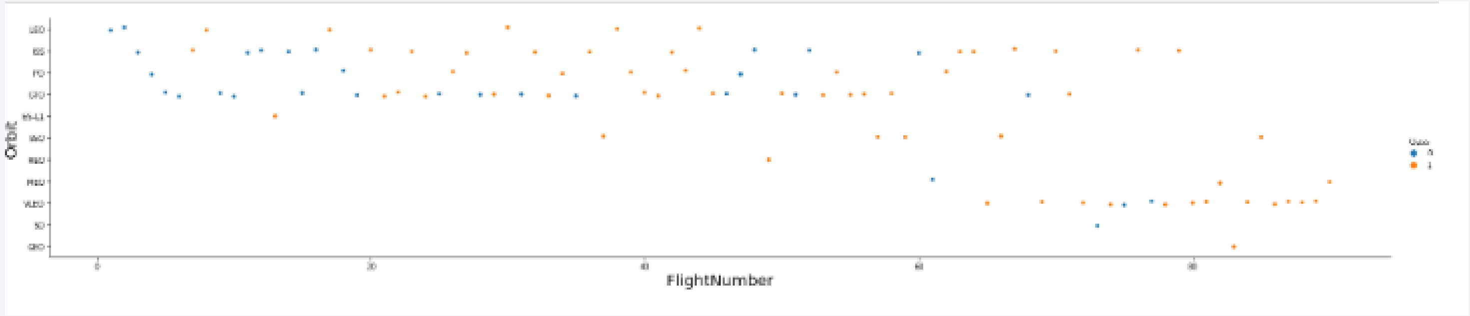
- For the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000)

Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO, and VLEO orbits had the most success rate.

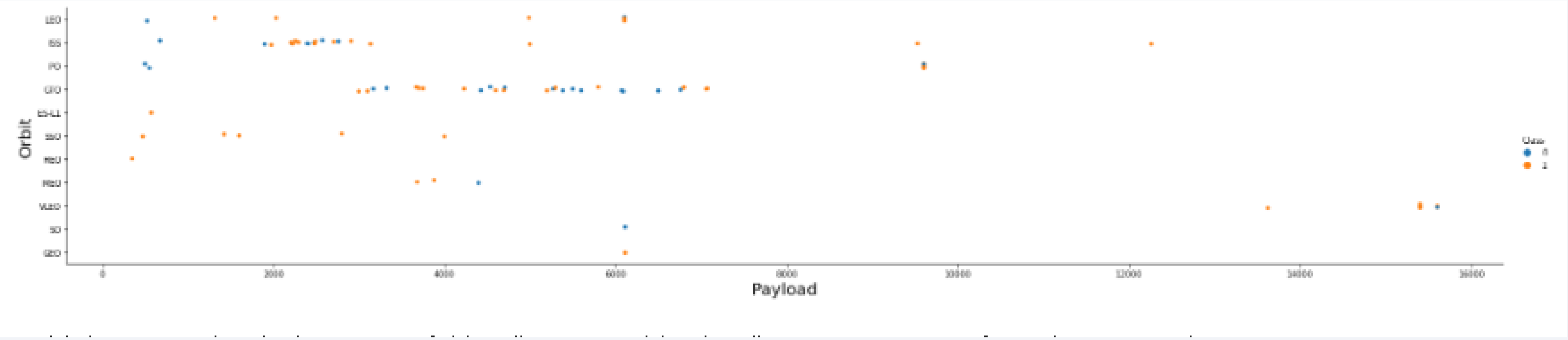


Flight Number vs. Orbit Type



- The LEO orbit success is related to the number of flights
- There is no relationship between flight number and the GTO orbit

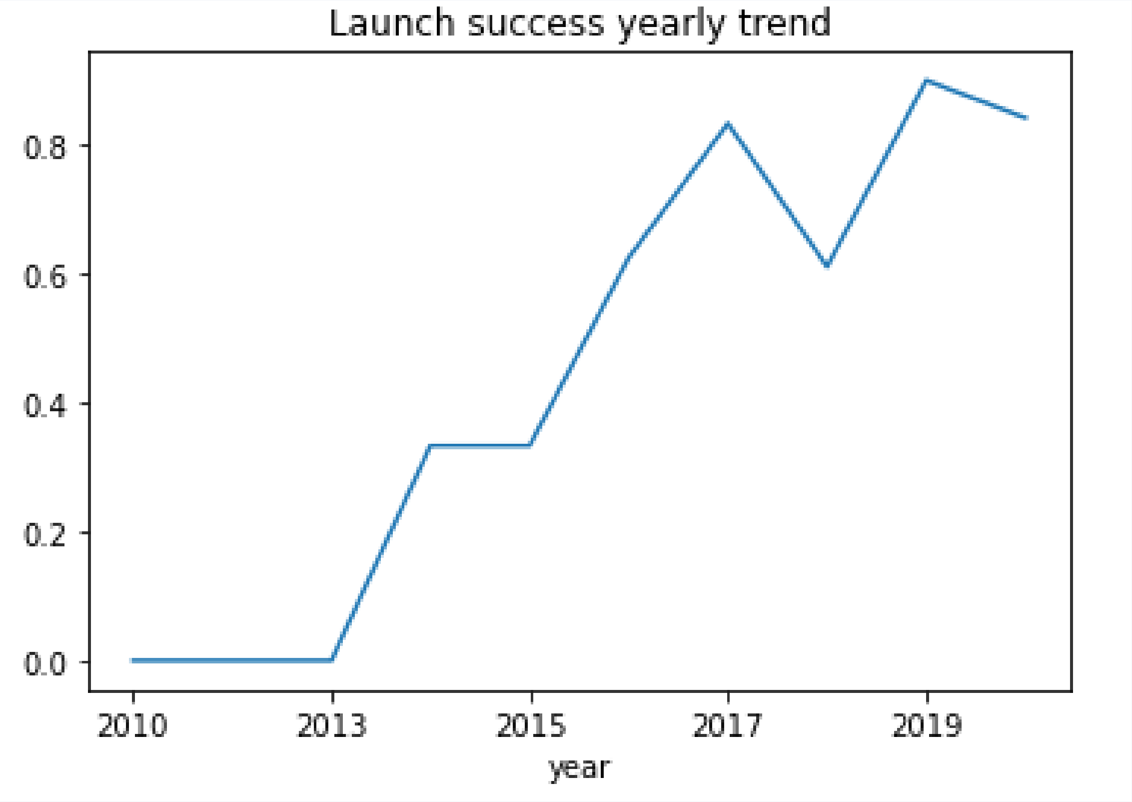
Payload vs. Orbit Type



- 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

- The sucess rate since 2013 kept increasing till 2020



All Launch Site Names

- To find the names of the unique launch sites, we use the key word ‘distinct’

```
[27]: #Display the names of the unique launch sites in the space mission
      c.execute('''SELECT distinct(Launch_Site) FROM space3''')
      c.fetchall()

[27]: [('CCAFS LC-40',), ('VAFB SLC-4E',), ('KSC LC-39A',), ('CCAFS SLC-40',)]
```

Launch Site Names Begin with 'CCA'

- Query to find 5 records where launch sites begin with `CCA` is given by this figure:

```
#Display 5 records where launch sites begin with the string 'CCA'
condi='%CCA,'
q='SELECT * FROM space3 where Launch_Site like ?'
c.execute('''SELECT * FROM space3  where Launch_Site LIKE 'CCAFS%' Limit 5''')

output_all = c.fetchall()
for row_all in output_all:
  print(row_all)
```

```
('04-06-2010', '18:45:00', 'F9 v1.0  B0003', 'CCAFS LC-40', 'Dragon Spacecraft Qualification Unit', '0', 'LEO', 'SpaceX',
'Success', 'Failure (parachute)')
('08-12-2010', '15:43:00', 'F9 v1.0  B0004', 'CCAFS LC-40', 'Dragon demo flight C1, two CubeSats, barrel of Brouere chees
e', '0', 'LEO (ISS)', 'NASA (COTS) NRO', 'Success', 'Failure (parachute)')
('22-05-2012', '07:44:00', 'F9 v1.0  B0005', 'CCAFS LC-40', 'Dragon demo flight C2', '525', 'LEO (ISS)', 'NASA (COTS)', 'S
uccess', 'No attempt')
('08-10-2012', '00:35:00', 'F9 v1.0  B0006', 'CCAFS LC-40', 'SpaceX CRS-1', '500', 'LEO (ISS)', 'NASA (CRS)', 'Success',
'No attempt')
('01-03-2013', '15:10:00', 'F9 v1.0  B0007', 'CCAFS LC-40', 'SpaceX CRS-2', '677', 'LEO (ISS)', 'NASA (CRS)', 'Success',
'No attempt')
```

Total Payload Mass

- The total payload carried by boosters from NASA is calculated as below:

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

```
#Display the total payload mass carried by boosters launched by NASA (CRS)
c.execute('''SELECT sum(PAYLOAD_MASS__KG_) as total_payload FROM space3  where Customer='NASA (CRS)' ''')
c.fetchall()
```

```
[(45596,)]
```

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is calculated as below:

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

```
#Display average payload mass carried by booster version F9 v1.1
c.execute('''SELECT avg(PAYLOAD_MASS__KG_) as avg_payload FROM space3  where  Booster_Version='F9 v1.1' ''')
c.fetchall()
```

```
[(2928.4,)]
```

First Successful Ground Landing Date

- The dates of the first successful landing outcome on ground pad are found as below:

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

Hint:Use min function

```
#List the date when the first successful landing outcome in ground pad was acheived.
c.execute('''select min (Date) from space3 where [Landing _Outcome] Like 'Success %'
''')
c.fetchall()
```

```
[('01-05-2017',)]
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- The names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 is given as below:

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

```
c.execute('''select  * from space3
where [Landing _Outcome]='Success (drone ship)'
and
PAYLOAD_MASS__KG_ between 4000 and 6000''')
output_all = c.fetchall()
for row_all in output_all:
  print(row_all)
```

```
('06-05-2016', '05:21:00', 'F9 FT B1022', 'CCAFS LC-40', 'JCSAT-14', '4696', 'GTO', 'SKY Perfect JSAT Group', 'Success', 'Success (drone ship)')
('14-08-2016', '05:26:00', 'F9 FT B1026', 'CCAFS LC-40', 'JCSAT-16', '4600', 'GTO', 'SKY Perfect JSAT Group', 'Success', 'Success (drone ship)')
('30-03-2017', '22:27:00', 'F9 FT  B1021.2', 'KSC LC-39A', 'SES-10', '5300', 'GTO', 'SES', 'Success', 'Success (drone ship)')
('24-08-2017', '18:51:00', 'F9 FT B1038.1', 'VAFB SLC-4E', 'Formosat-5', '475', 'SSO', 'NSPO', 'Success', 'Success (drone ship)')
('11-10-2017', '22:53:00', 'F9 FT  B1031.2', 'KSC LC-39A', 'SES-11 / EchoStar 105', '5200', 'GTO', 'SES EchoStar', 'Success', 'Success (drone ship)')
```

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failure mission outcomes are given as below:

List the total number of successful and failure mission outcomes

```
c.execute('''select  count(Mission_Outcome) as Mission_outcome_total_success from space3
where Mission_Outcome like 'Success%'
''')
output_all = c.fetchall()
for row_all in output_all:
  print(row_all)
```

```
(100,)
```

```
c.execute('''select  count(Mission_Outcome) as Mission_outcome_total_failure from space3
where Mission_Outcome like 'Failure%'
''')
output_all = c.fetchall()
for row_all in output_all:
  print(row_all)
```

```
(1,)
```

Boosters Carried Maximum Payload

- The names of the booster which have carried the maximum payload mass is given as below:

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

```
c.execute('''select Booster_Version from space3
where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from space3)
''')
output_all = c.fetchall()
for row_all in output_all:
  print(row_all)
```

```
('F9 FT B1029.1',)
('F9 FT B1036.1',)
('F9 B4 B1041.1',)
('F9 FT  B1036.2',)
('F9 B4  B1041.2',)
('F9 B5B1048.1',)
('F9 B5 B1049.2',)
```

2015 Launch Records

- We use here between and like clauses
- The failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015 is given as below:

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

```
c.execute('''select Date, Booster_Version, Launch_Site from space3 as s
where [Landing _Outcome] LIKE 'Failure (drone ship)'
and Date between '01-01-2015' and '31-12-2015'
''')

output_all = c.fetchall()
for row_all in output_all:
  print(row_all)
```

```
('10-01-2015', 'F9 v1.1 B1012', 'CCAFS LC-40')
('14-04-2015', 'F9 v1.1 B1015', 'CCAFS LC-40')
('17-01-2016', 'F9 v1.1 B1017', 'VAFB SLC-4E')
('04-03-2016', 'F9 FT B1020', 'CCAFS LC-40')
('15-06-2016', 'F9 FT B1024', 'CCAFS LC-40')
```

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

- The rank landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order is given as:

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

```
c.execute('''select  * from space3
where [Landing _Outcome] in ('Failure (drone ship)','Success (ground pad)')
and Date between '04-06-2010' and '20-03-2017'
order by Date desc
''')
output_all = c.fetchall()
for row_all in output_all:
  print(row_all)
```

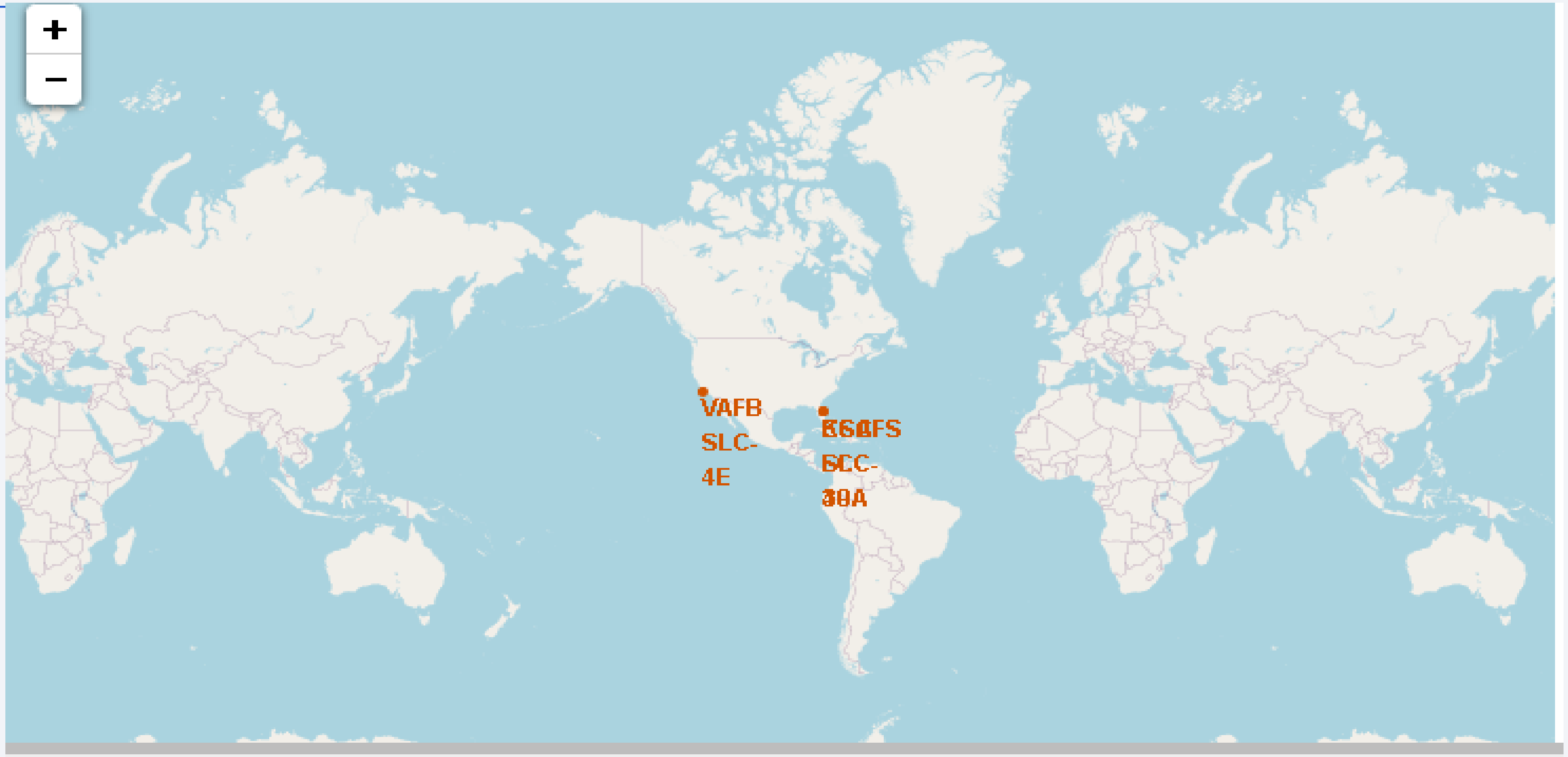
```
('19-02-2017', '14:39:00', 'F9 FT B1031.1', 'KSC LC-39A', 'SpaceX CRS-10', '2490', 'LEO (ISS)', 'NASA (CRS)', 'Success', 'Success (ground pad)')
('18-07-2016', '04:45:00', 'F9 FT B1025.1', 'CCAFS LC-40', 'SpaceX CRS-9', '2257', 'LEO (ISS)', 'NASA (CRS)', 'Success', 'Success (ground pad)')
('17-01-2016', '18:42:00', 'F9 v1.1 B1017', 'VAFB SLC-4E', 'Jason-3', '553', 'LEO', 'NASA (LSP) NOAA CNES', 'Success', 'Failure (drone ship)')
('15-12-2017', '15:36:00', 'F9 FT  B1035.2', 'CCAFS SLC-40', 'SpaceX CRS-13', '2205', 'LEO (ISS)', 'NASA (CRS)', 'Success', 'Success (ground pad)')
('15-06-2016', '14:29:00', 'F9 FT B1024', 'CCAFS LC-40', 'ABS-2A Eutelsat 117 West B', '3600', 'GTO', 'ABS Eutelsat', 'Success', 'Failure (drone ship)')
('14-08-2017', '16:31:00', 'F9 B4 B1039.1', 'KSC LC-39A', 'SpaceX CRS-12', '3310', 'LEO (ISS)', 'NASA (CRS)', 'Success', 'Success (ground pad)')
('14-04-2015', '20:10:00', 'F9 v1.1 B1015', 'CCAFS LC-40', 'SpaceX CRS-6', '1898', 'LEO (ISS)', 'NASA (CRS)', 'Success', 'Failure (drone ship)')
('10-01-2015', '09:47:00', 'F9 v1.1 B1012', 'CCAFS LC-40', 'SpaceX CRS-5', '2395', 'LEO (ISS)', 'NASA (CRS)', 'Success', 'Failure (drone ship)')
('08-01-2018', '01:00:00', 'F9 B4 B1043.1', 'CCAFS SLC-40', 'Zuma', '5000', 'LEO', 'Northrop Grumman', 'Success (payload status unclear)', 'Success (ground pad)')
('07-09-2017', '14:00:00', 'F9 B4 B1040.1', 'KSC LC-39A', 'Boeing X-37B OTV-5', '4990', 'LEO', 'U.S. Air Force', 'Success', 'Success (ground pad)')
```

Section 3

Launch Sites Proximities Analysis

All launch sites on a map

- The Space x launch sites are located in USA(Florida ans California)

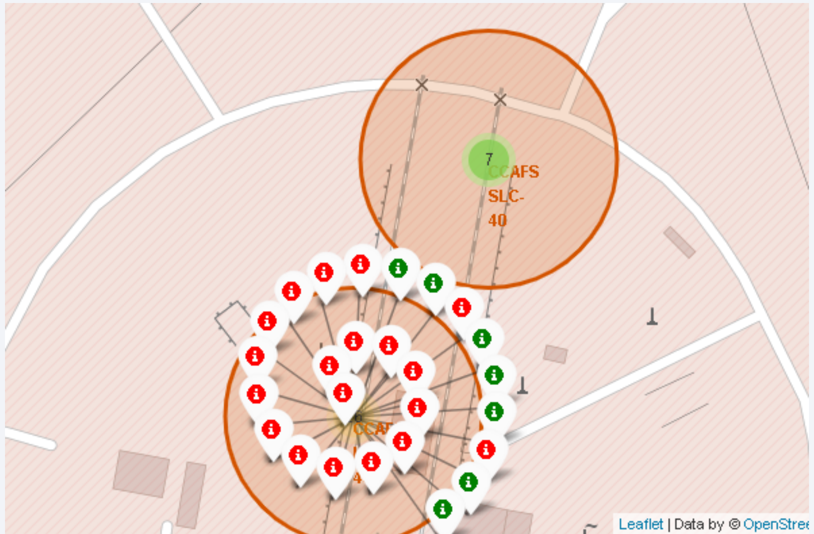


Markers of the success/failed launches for each site on the map

- Florida lauch site:

- Green marker=succes

- Red marker = failure



Distances between a launch site to its proximities

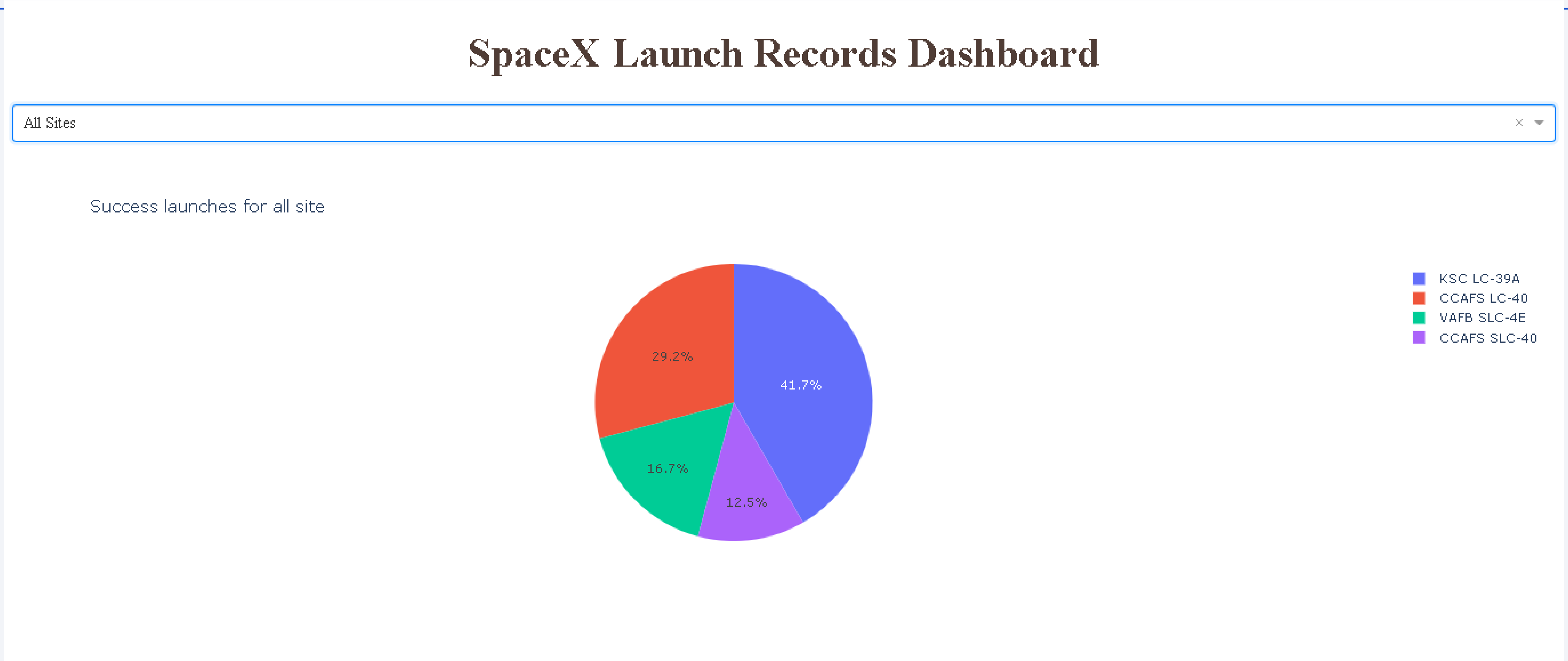
- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed

- Explain the important elements and findings on the screenshot

Section 4

Build a Dashboard with Plotly Dash

Pie chart visualizing launch success counts for all sites



The KSC LC-39A site has the largest successful launches

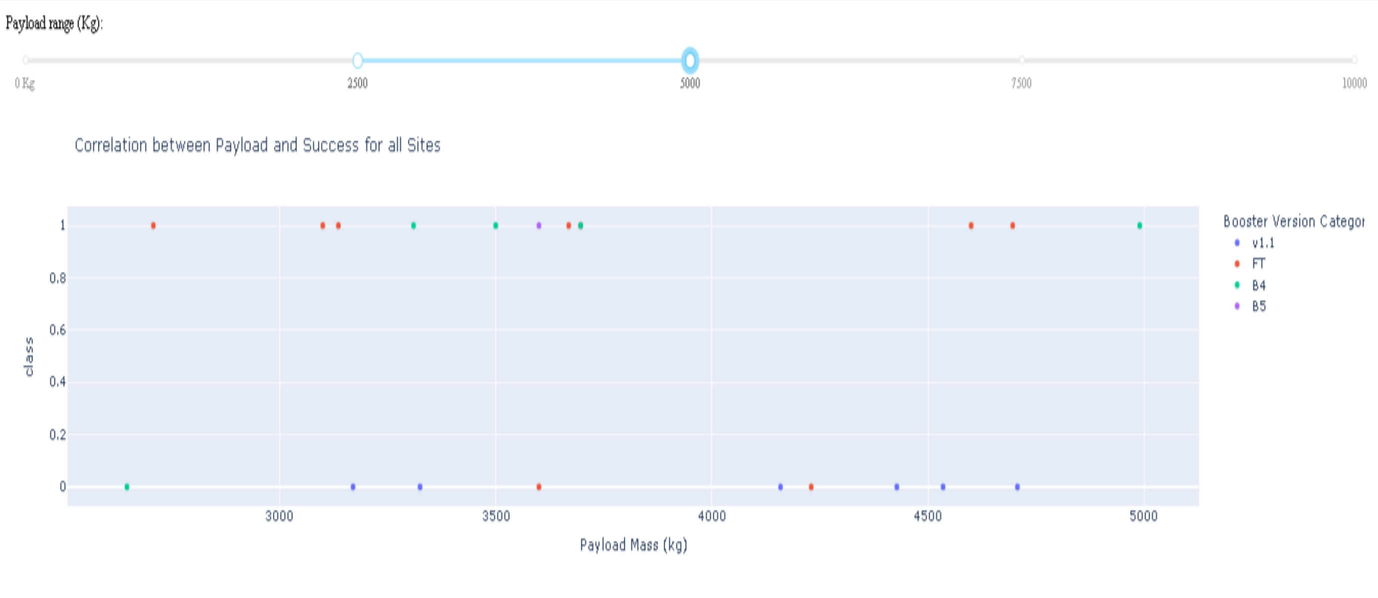
Pie chart of the highest launch site success ratio

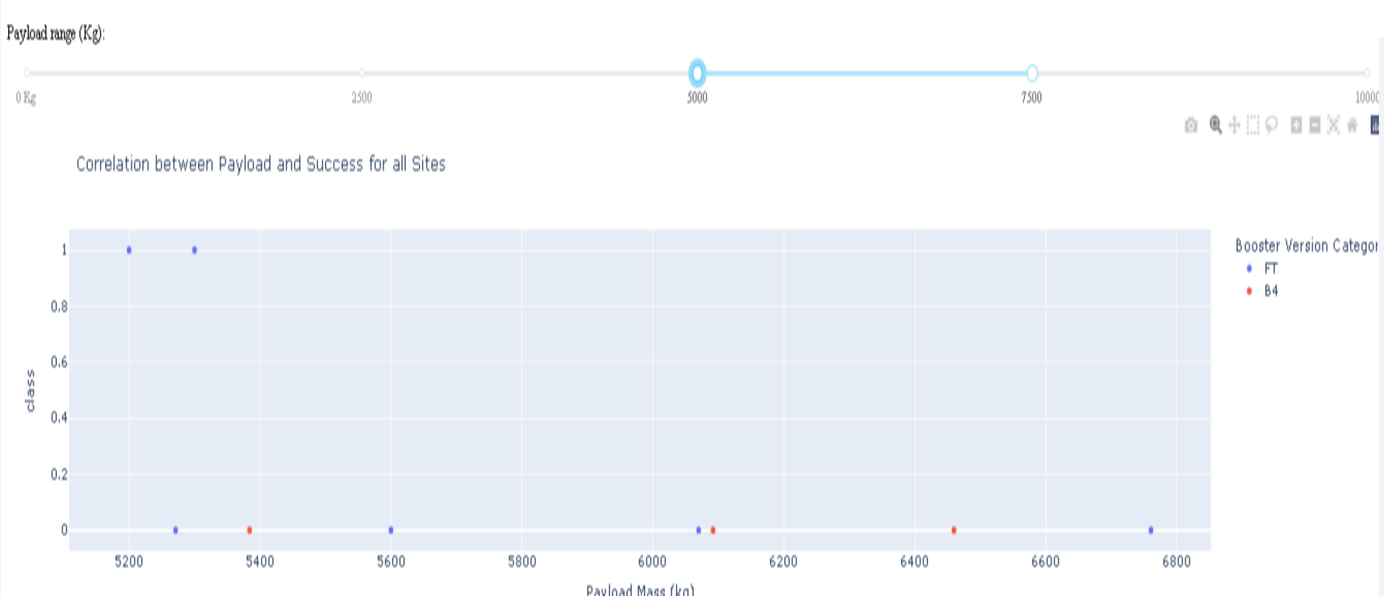
The KSC LC-39A site achieved 76.9% succes rate vs. 23.1% failure rate.

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

Observation:

For low weighted payloads, the success rates is higher than the heavy weigheted payloads



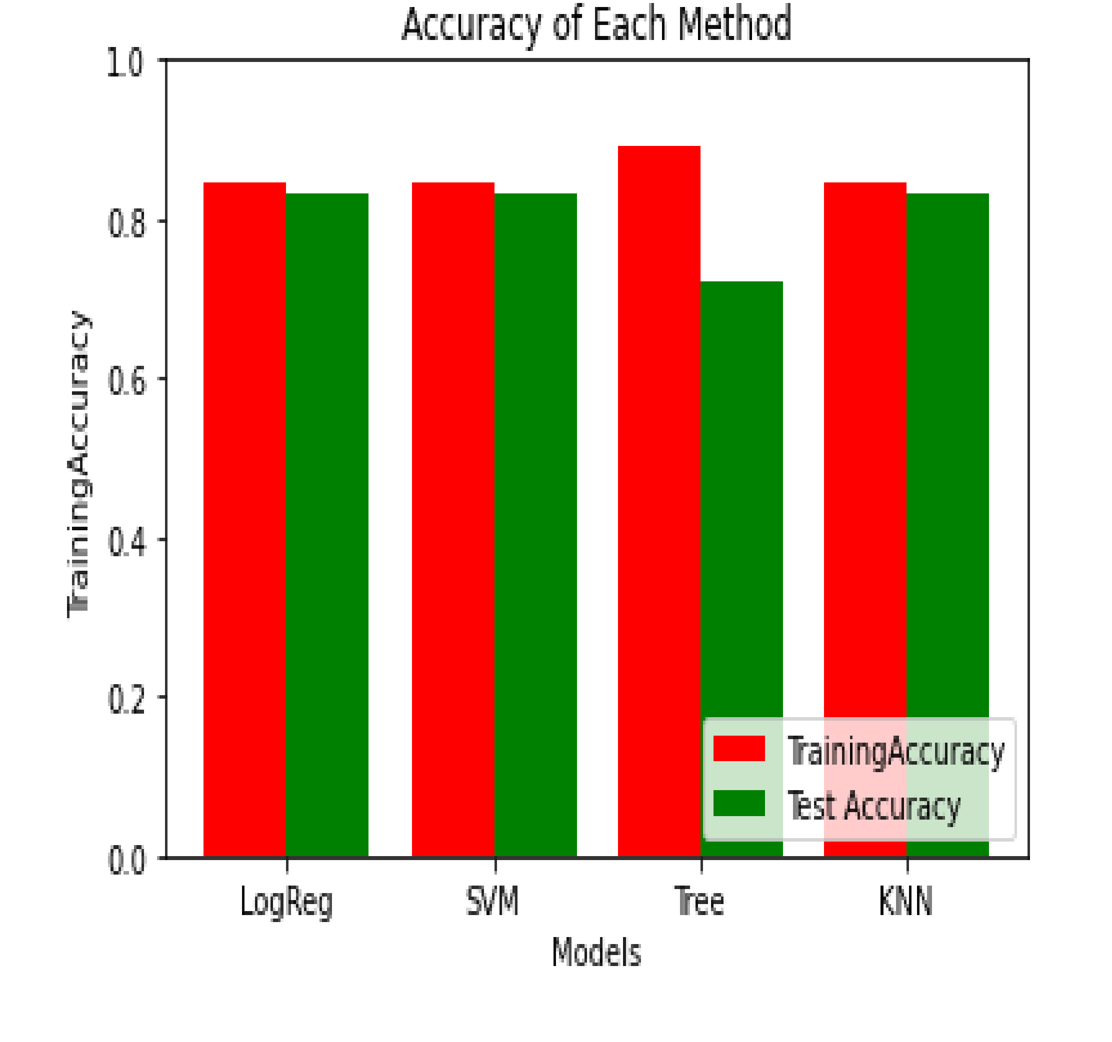


Section 5

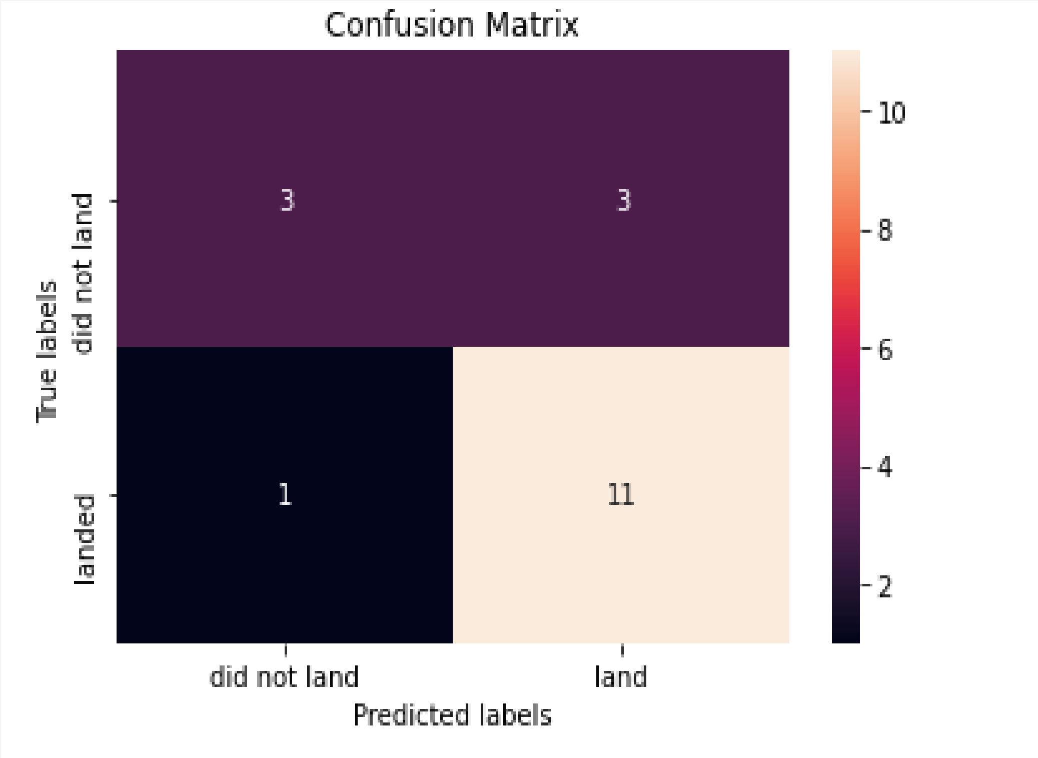
Predictive Analysis (Classification)

Classification Accuracy

- The tree model has the highest classification accuracy



Confusion Matrix



Conclusions

- The flight amount and the success rate at a launch site are proportional. The larger the flight amount, the greater the success rate at a launch site.
- Launch success rate started to increase in 2013 until2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- For this task ,the best machine learning algorithm is the decision tree classifier.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project


Thank you!