IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Li Si Lin
3/7/2023

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collecion via API
  - Exploratory Data Analysis
  - Predictive Analysis

- Summary of all results

Introduction

- Project background and context
  - Aim of the project is to predict if the Falcon 9 will successfully land, given the information of all details in the launch process.
  - This would help the investor to further evaluate the value of SpaceX in terms of its sucess rate and cost–savings.
- Problems you want to find answers
  - What are the main factors that affect the landing process?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data is collected from SpaceX REST API, which is public-accessible information.
  - Furthermore, we scrap external data source such as Wikipedia.
- Perform data wrangling
  - Reduce column size, backfill missing values and perform One-Hot Encoding on catergorical values.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Compare through various classification models, fine-tune the parameters through grid search and evaluated by the recall and precision rate.

Data Collection

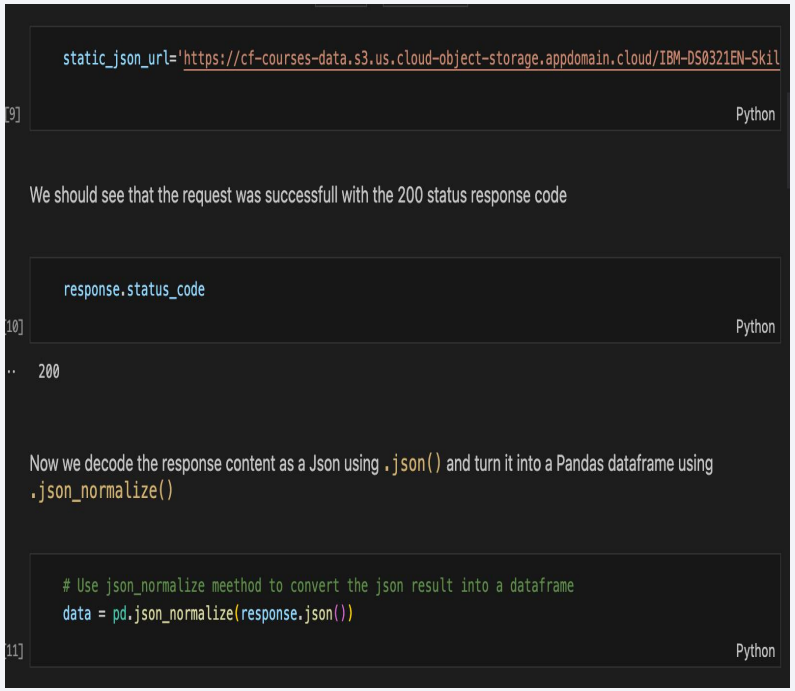
- Launch, Payload and Rocket information is obtained through API

SpaceX REST API --> Data in JSON format -->
Convert to Data Frame --> Clean data

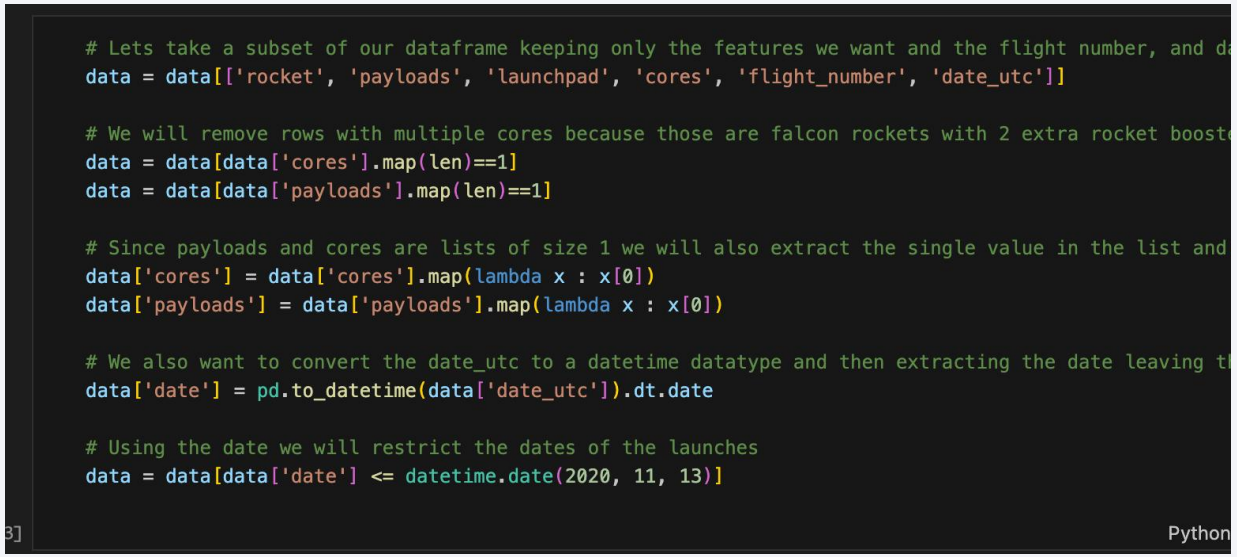
Other relavant information is extracted from Wikipedia

Crawl from webpage --> Use BeautifulSoup to extract important information from HTML format --> Convert to dataframe

Data Collection – SpaceX API



read data from API and convert it to dataframe



perform data clean up process

Data Collection - Scraping

```
# use requests.get() method with the provided static_url
# assign the response to a object

response = requests.get(static_url)
```

[4] Python

Create a BeautifulSoup object from the HTML response

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text)
```

[18] Python

1. Scrap data and convert it into beautiful soup object

```
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 colum

for i in first_launch_table.findAll('th'):
    nm = extract_column_from_header(i)
    if nm is not None and len(nm) > 0:
        column_names.append(nm)
```

Python

Check the extracted column names

```
print(column_names)
```

Python

2. Extract relavant data from the object

```
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']=[]
```

54] Python

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            launch_dict['Flight No.'].append(flight_number)

            #print(flight_number)
            datatimelist=date_time(row[0])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`
            date = datatimelist[0].strip(',')
            #print(date)
            launch_dict['Date'].append(date)

            # Time value
            # TODO: Append the time into launch_dict with key `Time`
            time = datatimelist[1]
            #print(time)
            launch_dict['Time'].append(time)

            # Booster version
            # TODO: Append the bv into launch_dict with key `Version Booster`
            bv=booster_version(row[1])
            if not(bv):
                bv=row[1].a.string
            # print(bv)
            launch_dict['Version Booster'].append(bv)

            # Launch Site
            # TODO: Append the bv into launch_dict with key `Launch Site`
```

2. Converted into the data frame

Data Wrangling

- Describe how data were processed

    1. Calculate launches number for each site
    2. Calcuate the number and occurance of each orbit
    3. Calculate occurance of mission outcome
    4. Create landing outcome label

EDA with Data Visualization

- **Scatter Graphs** is used to explore relationship between variables

- **Bar Plot** is used to compare success rate between categoric values.

EDA with SQL

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first succesful landing outcome in ground pad was acheived.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
- 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.

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
- Explain why you added those objects
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer–review purpose

Build a Dashboard with Plotly Dash

- Dashboard has dropdown, pie chart, range slider and scatter plot components.
  - Dropdown allows user to interact with the map by choosing launch site.
  - Pie chart shows successes/failures for the site chosen.
  - Range slider allows user to choose the range of payload mass.
  - Scatter plot shows relationship between two variables.

Predictive Analysis (Classification)

- **Prepare Data**: after loading the dataset, it is firstly normalized and split into train and test set.

- **Model Tuning**: select a range of models and use Grid Search to fine tune of each model with train set. Calculated the fine-tuned model score using test set.

- **Model selection:** Based on the model score generated (precision, recall, accuracy), to choose the model with the best score.

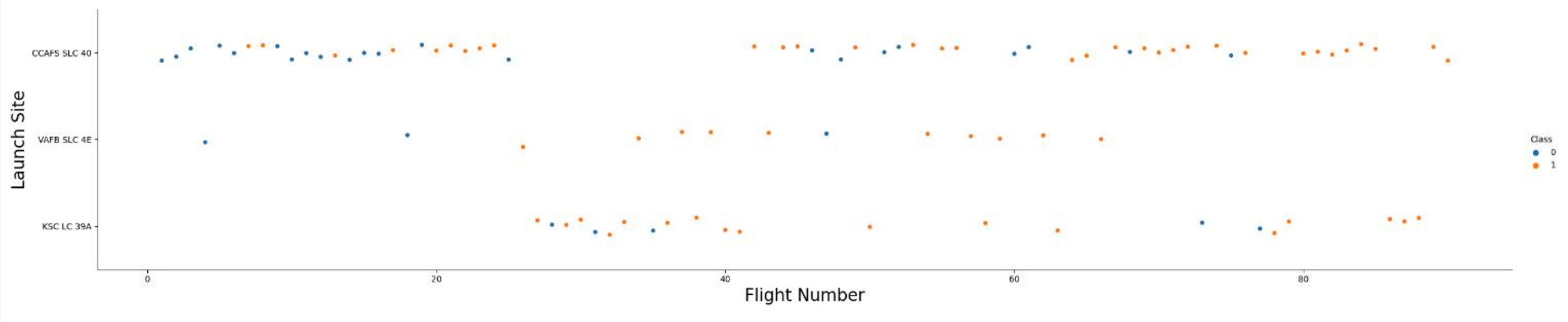
Results

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

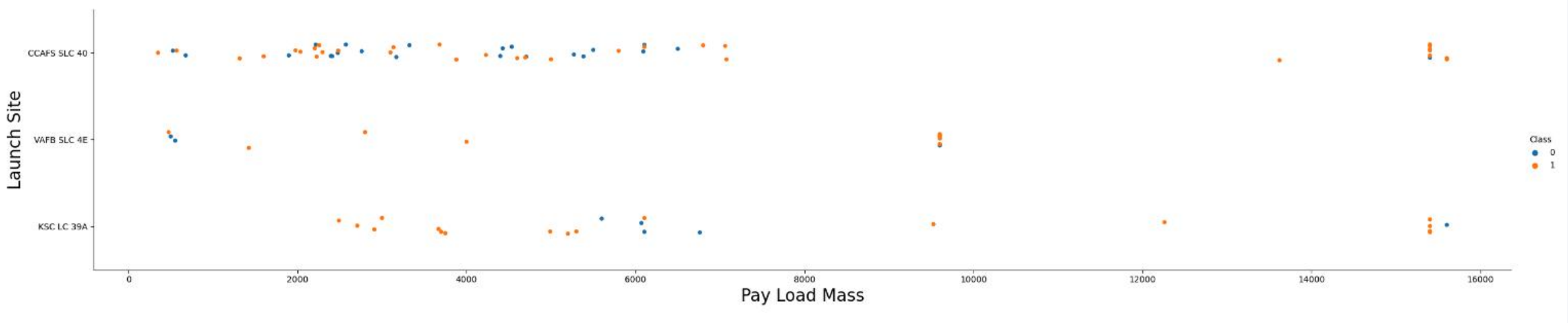
Section 2

Insights drawn from EDA

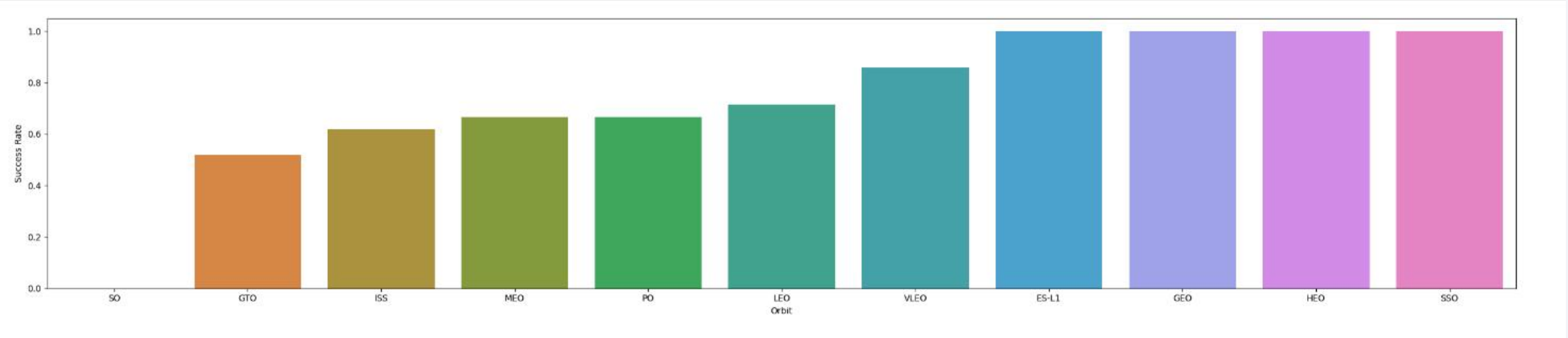
Flight Number vs. Launch Site



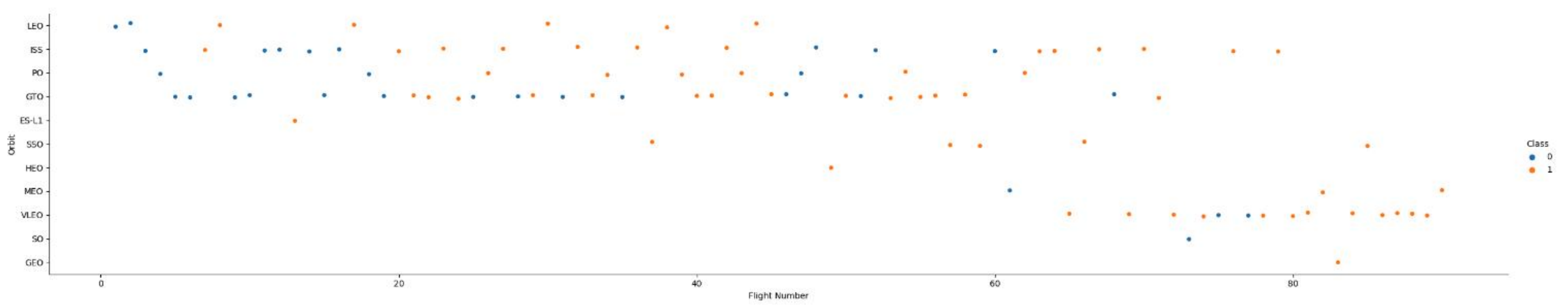
Payload vs. Launch Site



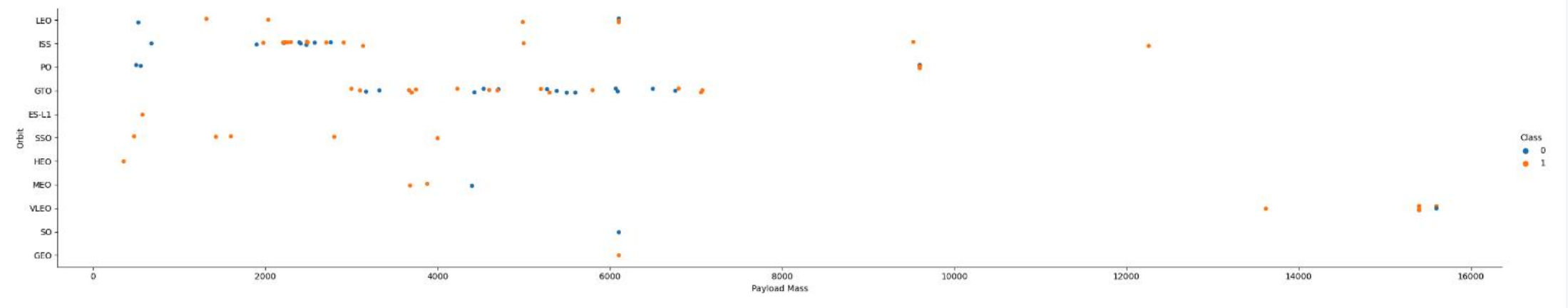
Success Rate vs. Orbit Type



Flight Number vs. Orbit Type



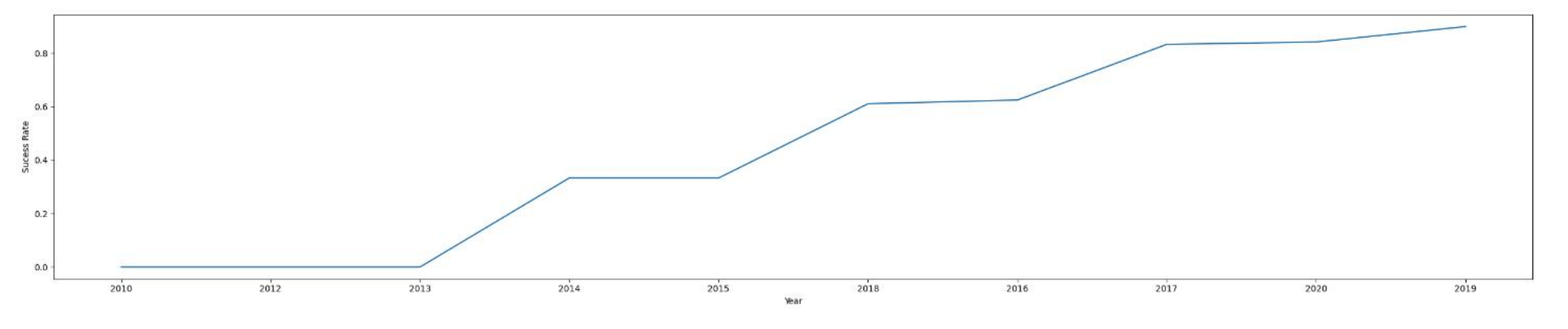
Payload vs. Orbit Type



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

- Find the names of the unique launch sites
- Use ‘Distinct’ to remove duplicated launch_site

```
[13]: %%sql
      SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL
```

```
 * sqlite:///my_data1.db
Done.
```

[13]:

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- Use ‘WHERE’ condition to filter launch sites, and use ‘LIMIT’ to choose top 5 records.

[14]:
```
%%sql
SELECT * FROM SPACEXTBL
WHERE LAUNCH_SITE LIKE '%CCA%'
LIMIT 5
```

```
 * sqlite:///my_data1.db
Done.
```

[14]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Out
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (para
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (para
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No at
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No at
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No at

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- Use ‘SUM()’ to aggregated the payload and use ‘WHERE’ to filter the selected launches

```
[16]: %%sql
      SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL
      WHERE CUSTOMER = 'NASA (CRS)'

       * sqlite:///my_data1.db
      Done.
[16]: SUM(PAYLOAD_MASS__KG_)
      45596.0
```

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Use ‘AVG()’ to calculate the average of payload with selected launches.

```
[19]: %%sql
      SELECT AVG(PAYLOAD_MASS__KG_)
      FROM SPACEXTBL
      WHERE BOOSTER_VERSION LIKE '%F9 v1%'

       * sqlite:///my_data1.db
      Done.
```

[19]:

AVG(PAYLOAD_MASS__KG_)
1986.1

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Use ‘MIN(date)’ to find the earlist date of successful landing launches filtered by ‘WHERE’ clause.

```
[20]: %%sql
      SELECT MIN(DATE) FROM SPACEXTBL
      WHERE Landing_Outcome LIKE '%Success%'

       * sqlite:///my_data1.db
      Done.

[20]: MIN(DATE)
      01/07/2020
```

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

- Use ‘>’ and ‘<’ to filter launches within a payload mass range.

[25]:
```
%%sql
SELECT booster_version FROM SPACEXTBL
WHERE landing_outcome = 'Success (drone ship)'
AND PAYLOAD_MASS__KG_ > 4000
AND PAYLOAD_MASS__KG_ < 6000
```

```
 * sqlite:///my_data1.db
Done.
```

[25]:

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Create a column by ‘CASE...WHEN’ clause to split success and failure cases, and group by the column to count the number of cases in each category.

[31]:
```
%%sql
select
case
when landing_outcome like '%success%' then 'success'
when landing_outcome like '%failure%' then 'failure'
else 'others' end as indicator,
count(1) as cnt
from spacextbl
group by indicator
order by cnt asc
```

```
 * sqlite:///my_data1.db
Done.
```

[31]:

indicator	cnt
failure	10
success	61
others	928

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Use subquery to first select the maximum payload mass, then filter by the value.

```
[33]: %%sql
      select booster_version
      from spacextbl
      where payload_mass__kg_ = (select max(payload_mass__kg_) from spacextbl)
```

```
 * sqlite:///my_data1.db
Done.
```

[33]:

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

2015 Launch Records

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

- Use ‘substr()’ to extract information from a string

[37]:
```
%%sql
select
    substr(date, 4, 2) as month,
    landing_outcome,
    booster_version,
    launch_site
from spacextbl
where
    substr(date,7,4)='2015'
    and landing_outcome like '%Failure (drone ship)%'
```

```
 * sqlite:///my_data1.db
Done.
```

[37]:

month	Landing_Outcome	Booster_Version	Launch_Site
10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

[44]:
```
%%sql
select
    landing_outcome,
    count(1) as cnt
from spacextbl
where
    date between '04/06/2010' and '20/03/2017'
group by
    landing_outcome
order by
    count(1) desc
```

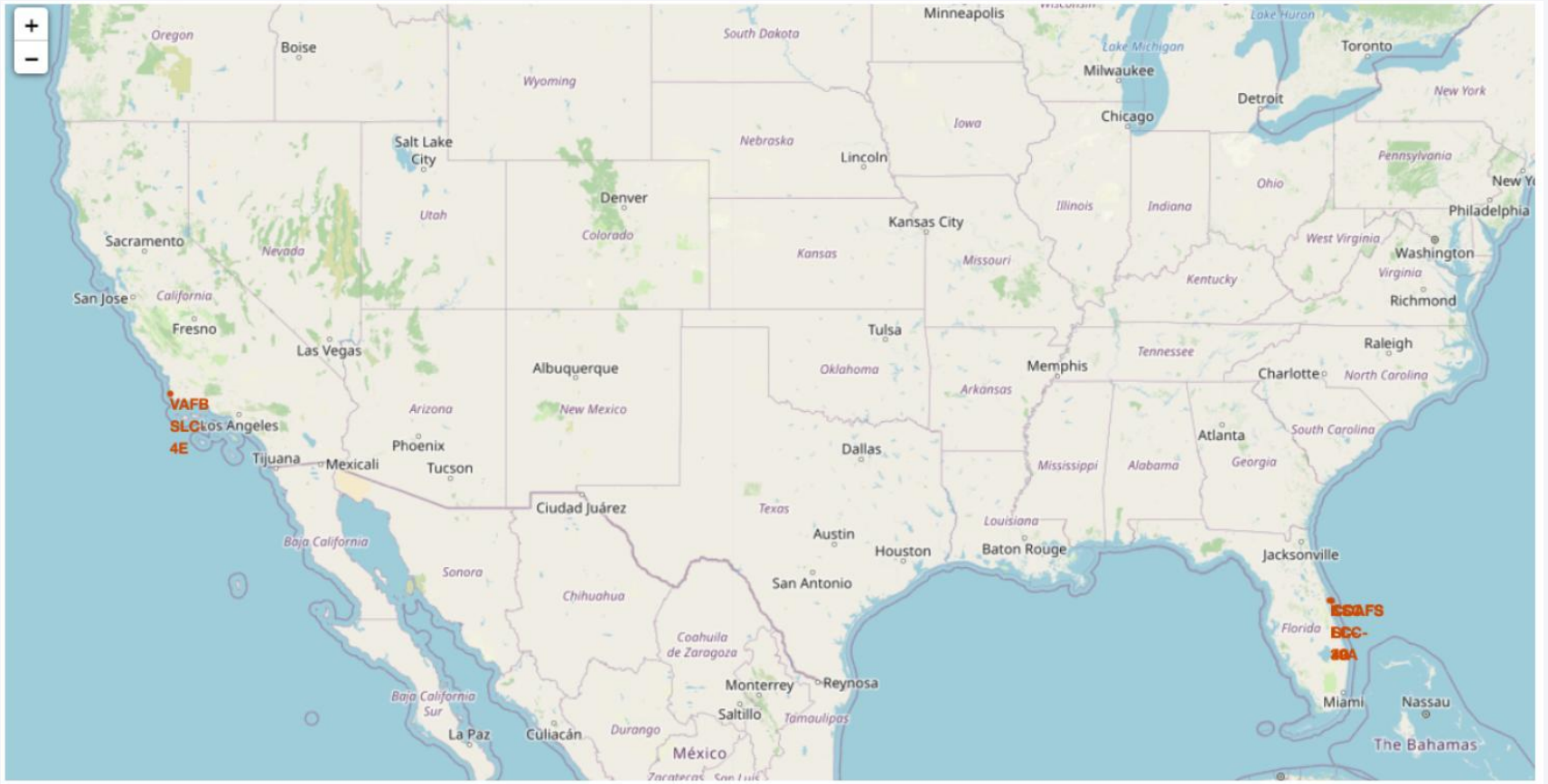
```
 * sqlite:///my_data1.db
Done.
```

[44]:

Landing_Outcome	cnt
Success	20
No attempt	9
Success (drone ship)	8
Success (ground pad)	7
Failure (drone ship)	3
Failure	3
Failure (parachute)	2
Controlled (ocean)	2
No attempt	1

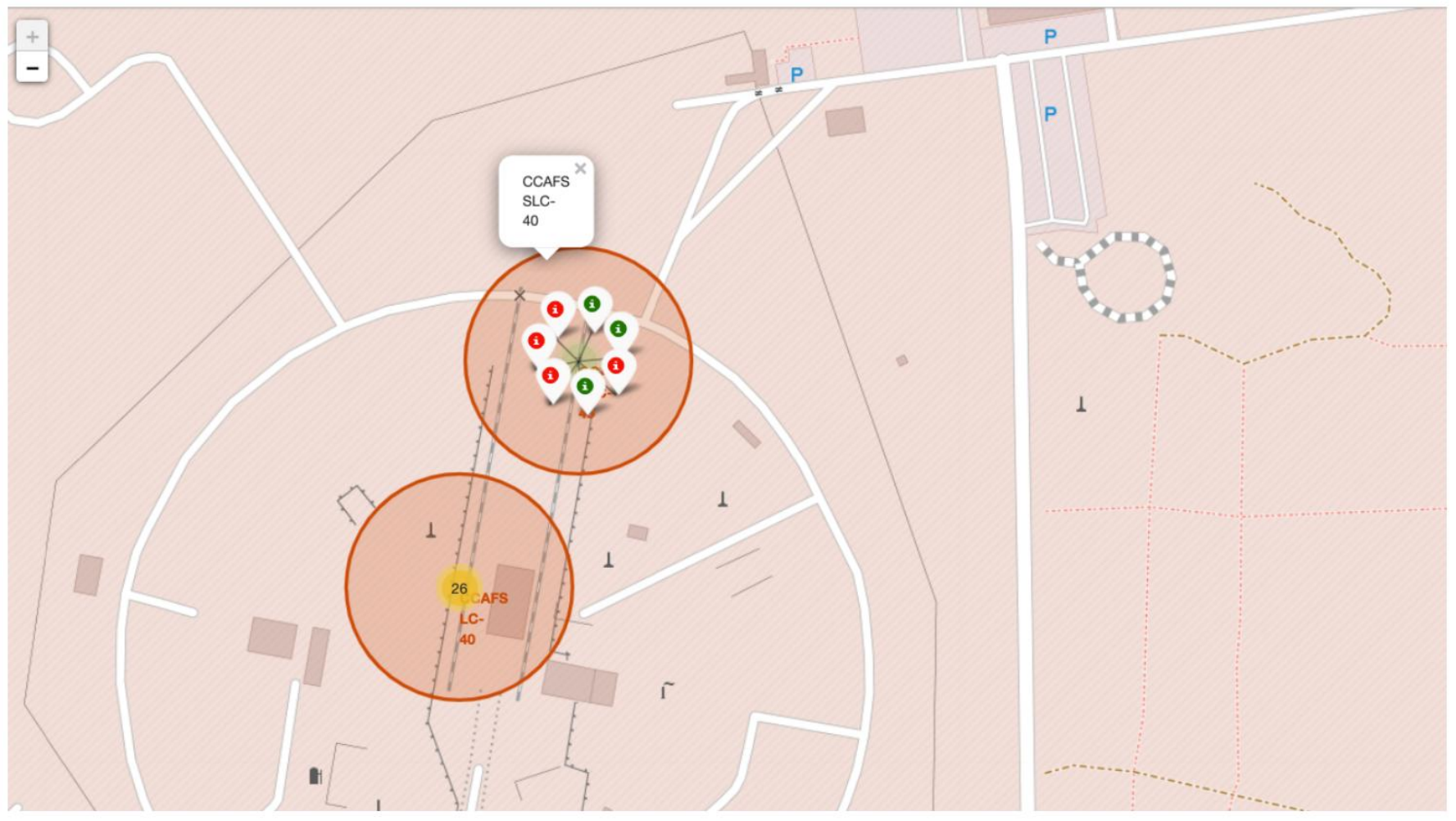

Section 3
Launch Sites
Proximities Analysis

Launch Site Location

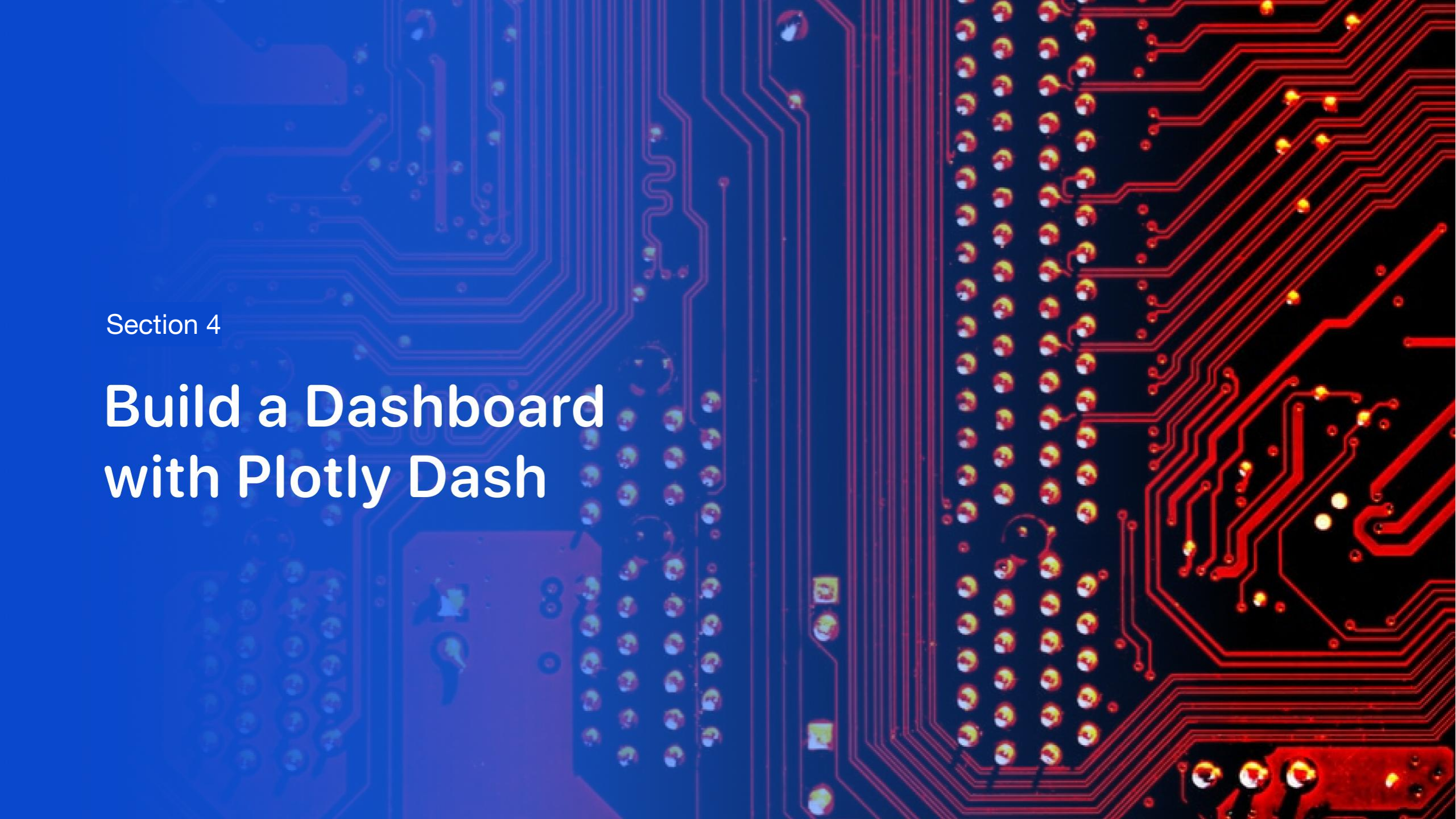


The red circles are launch sites and the grouping of points in a cluster to display multiple and different information.

Color Labelled Folium Map



Distance from Coastline


Section 4
Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>

- Replace <Dashboard screenshot 1> title with an appropriate title
- Show the screenshot of launch success count for all sites, in a piechart
- Explain the important elements and findings on the screenshot

<Dashboard Screenshot 2>

- Replace <Dashboard screenshot 2> title with an appropriate title
- Show the screenshot of the piechart for the launch site with highest launch success ratio
- Explain the important elements and findings on the screenshot

<Dashboard Screenshot 3>

- Replace <Dashboard screenshot 3> title with an appropriate title
- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

Section 5

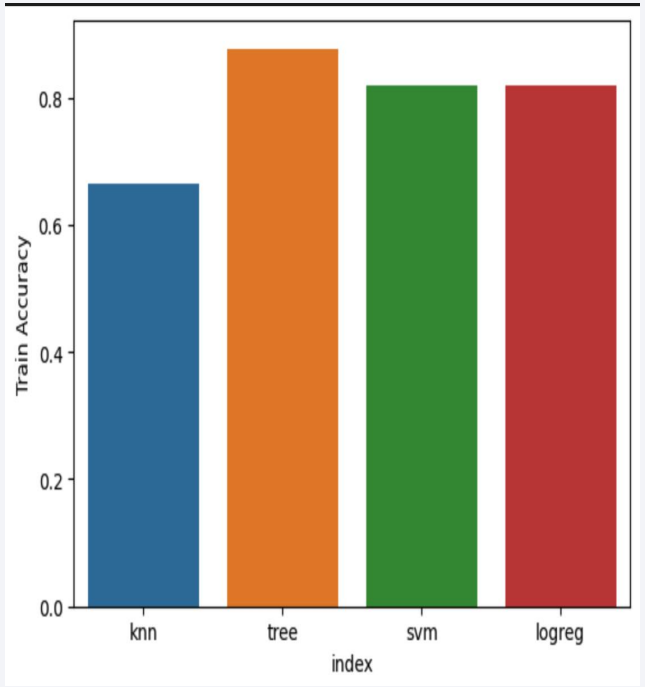
Predictive Analysis (Classification)

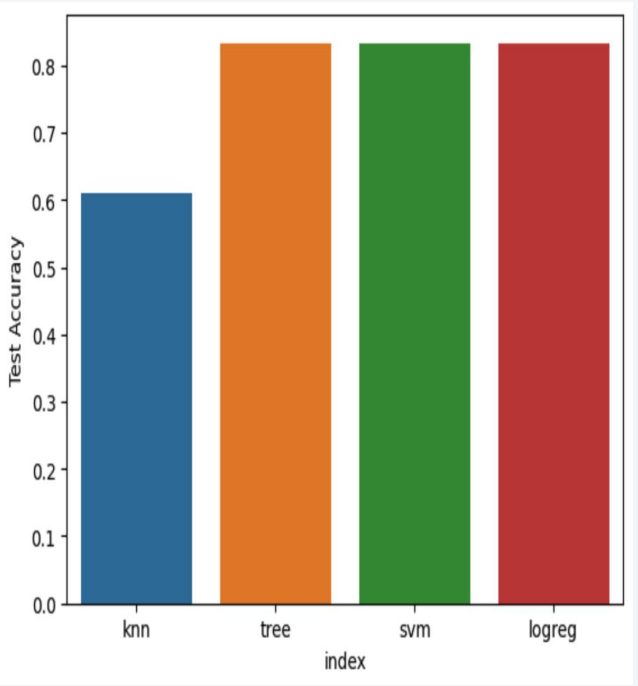
Classification Accuracy

While all models gain the same accuracy on test set, Tree based model outperforms the rest with the highest Accuracin in Train set. The optimal paramerters are as below.

	index	Train Accuracy	Test Accuracy
0	knn	0.664286	0.611111
1	tree	0.876786	0.833333
2	svm	0.819643	0.833333
3	logreg	0.819643	0.833333

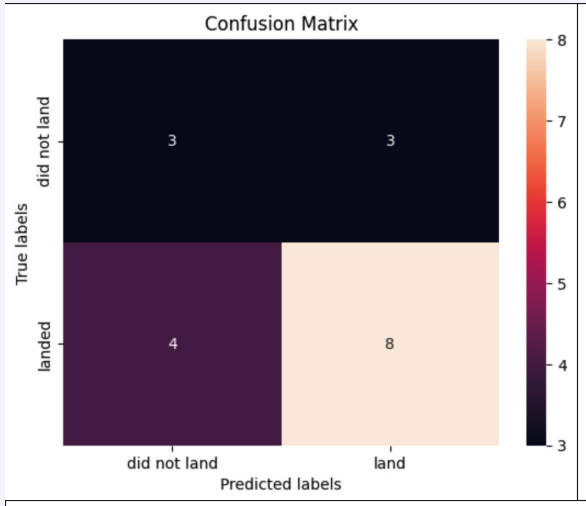
```
tree_cv.best_params_
[63]  ✓ 0.0s
...   {'criterion': 'gini',
       'max_depth': 4,
       'max_features': 'sqrt',
       'min_samples_leaf': 1,
       'min_samples_split': 5,
       'splitter': 'random'}
```



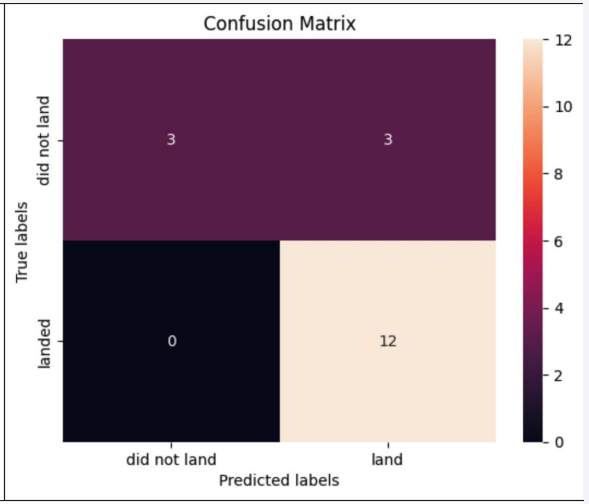


Confusion Matrix

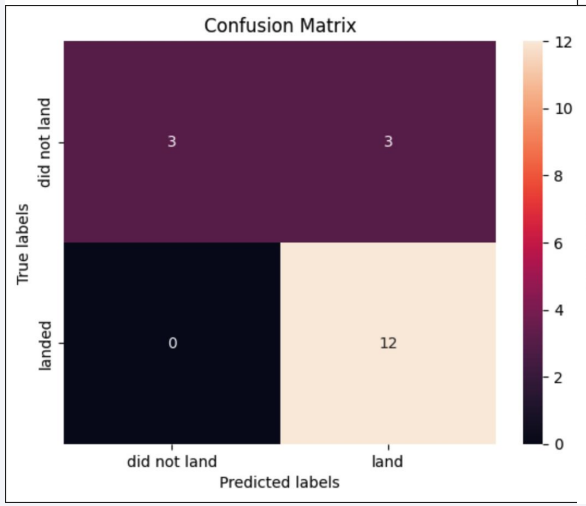
Knn



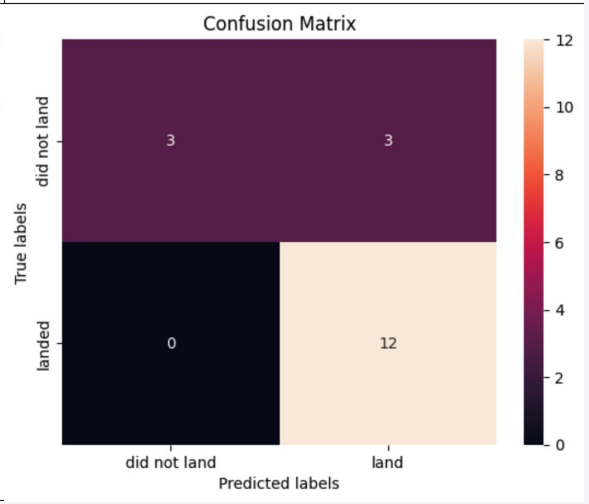
Tree



LogRegression



SVM



The KNN model is underperforms comparing to others.

However the tree model, SVM, Logistic Regression models perform the same in terms of confusion matrix.

Conclusions

- There are several factors that will affect the successu of the launch, which are launch site, orbit, and number of previous launches.
- With current data, the best orbits with highest success rates are GEO.
- Decision Tree model is the selected model for the accuracy of prediction. However, all models achieves same accuracy score since the test size is small. The model performance should be further evaluated with more available data points.

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!