IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Umer Sherdil Paracha
13.07.2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection
  - Data Wrangling
  - EDA with Data Visualization
  - EDA with SQL
  - Interactive Map using Folium
  - Dashboard using Plotly Dash
  - Predictive Analysis (Classification)
- Summary of all results
  - EDA Results
  - Interactive Analysis
  - Predictive Analysis

Introduction

- 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.
- The aim of the analysis is to predict if the landing of the first stage rocket.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data collected through SpaceX REST API and Web Scrapping from Wikipedia.
- Perform data wrangling
  - Exploratory data analysis performed using pandas.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Models built, tuned and evaluated using sklearn (Logistic Regression, SVM, Decision Trees, K-Nearest Neighbors).

Data Collection

- Datasets collected using SpaceX API and Web Scrapping from Wikipedia.
- API provides data about launches. For example, the type of rocket used, payload delivered, launch and landing specifications and landing outcome.
- Web Scrapping from Wikipedia tables was done using Beautiful Soup.

Data Collection – SpaceX API

1. Using **get** method to retrieve data from API.
2. Converting data to JSON.
3. Helper functions to fetch useful data from API.
4. Creating a Python dictionary.
5. Creating Pandas Dataframe.

- [Data_Science_and_ML/jupyter-labs-spacex-data-collection-api.ipynb at main · UmerSherdil/Data_Science_and_ML (github.com)]

1
```
spacex_url="https://api.spacexdata.com/v4/launches/past"

response = requests.get(spacex_url)
```

2
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

3
```
# Call getBoosterVersion
getBoosterVersion(data)

# Call getLaunchSite
getLaunchSite(data)

# Call getPayloadData
getPayloadData(data)

# Call getCoreData
getCoreData(data)
```

4
```
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
```
# Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

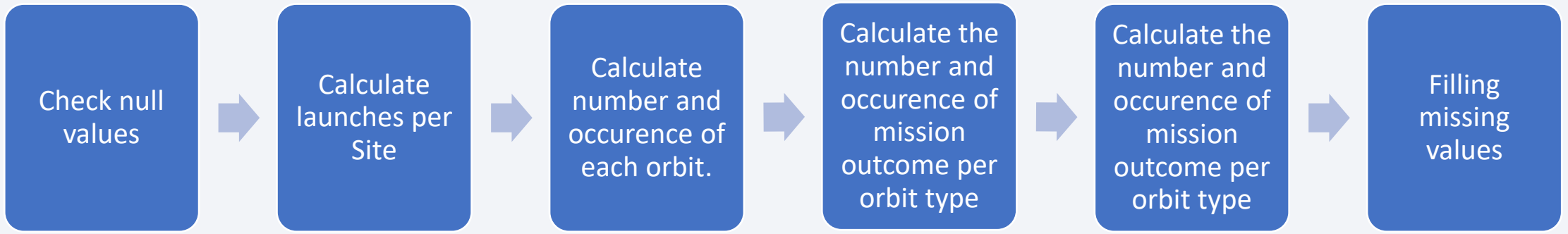
Data Collection - Scraping

1. Using **get** method to retrieve data from HTML.
2. Creating BeautifulSoup object.
3. Finding all tables.
4. Extracting column names.
5. Preparing dictionary
6. Filling dictionary with values from table.
7. Creating Dataframe

1
```
response = requests.get(static_url)
```
2
```
soup = BeautifulSoup(response.text, "html.parser")
```
3
```
html_tables = soup.find_all("table")
```
4
```
column_names = []
column_names_1 = []

temp = first_launch_table.find_all('th')
for item in temp:
    col_name = extract_column_from_header(item)
    if col_name is not None and len(col_name) > 0:
        column_names.append(col_name)
```
5
```
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
```
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)
            datatimelist=date_time(row[0])
```
7
```
df=pd.DataFrame(launch_dict)
```

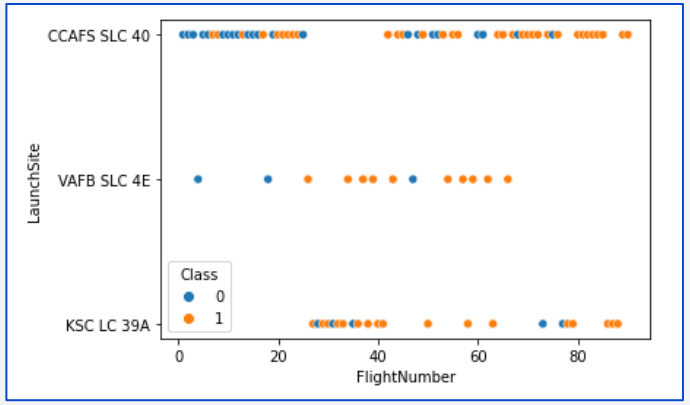
- [Data_Science_and_ML/jupyter-labs-webscraping.ipynb at main · UmerSherdil/Data_Science_and_ML (github.com)]

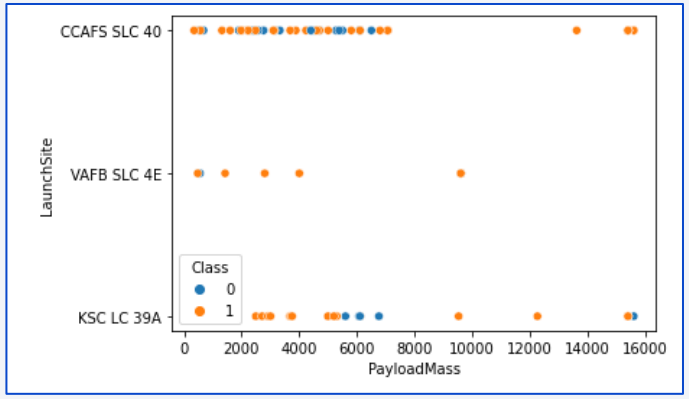
Data Wrangling

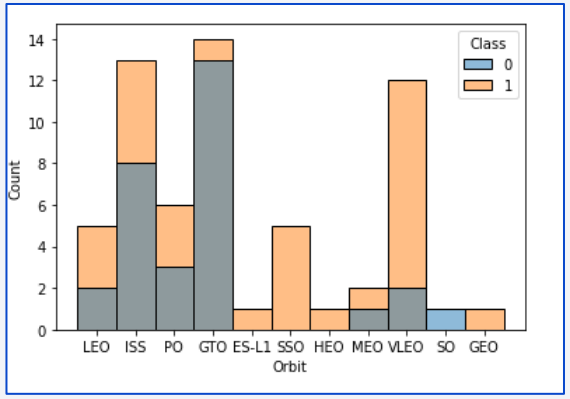


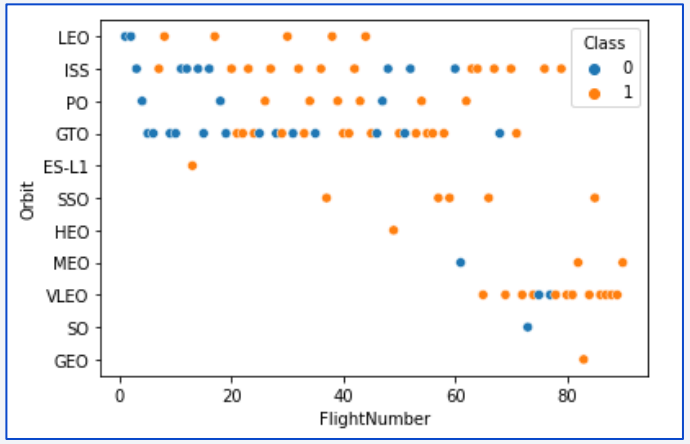
Data_Science_and_ML/labs-jupyter-spacex-Data wrangling.ipynb at main · UmerSherdil/Data_Science_and_ML (github.com)

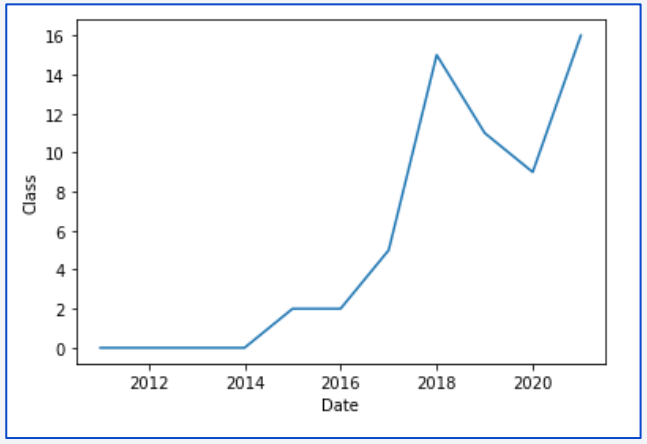
EDA with Data Visualization

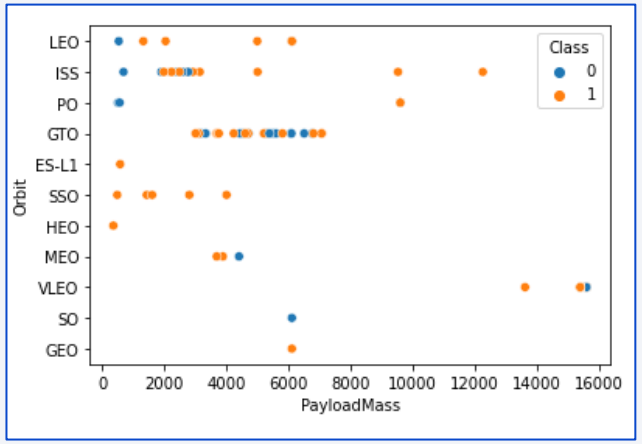












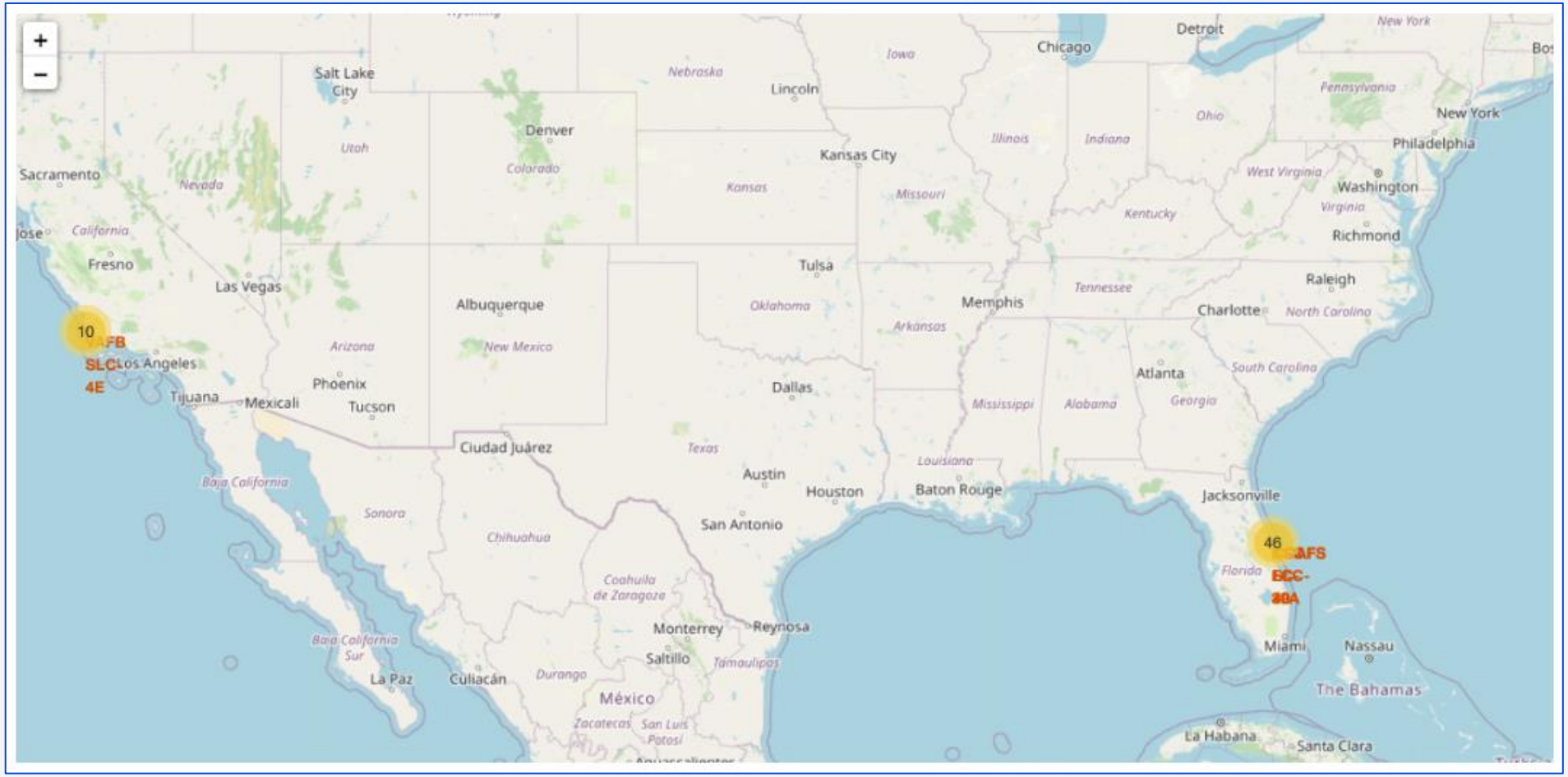
[Data_Science_and_ML/jupyter-labs-eda-dataviz.ipynb at main · UmerSherdil/Data_Science_and_ML (github.com)]

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

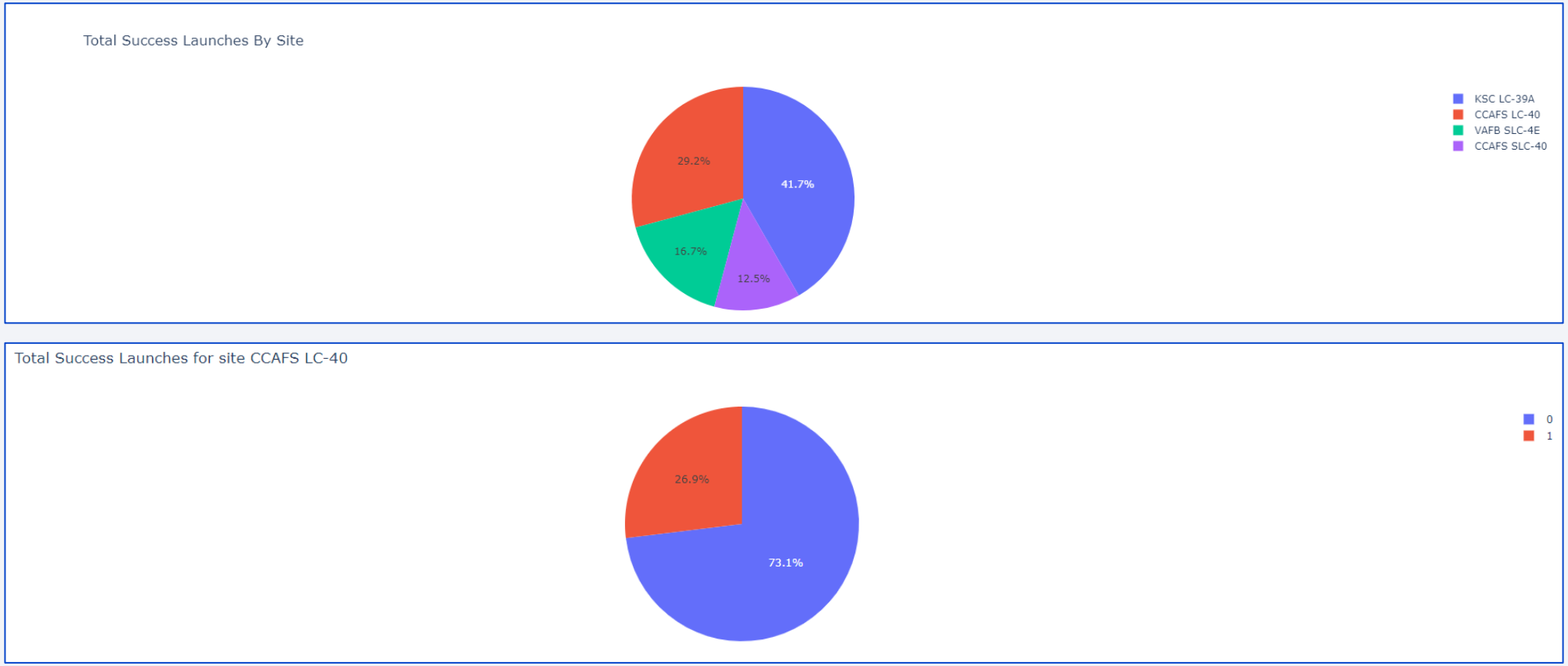
Data_Science_and_ML/jupyter-labs-eda-sql-coursera_sqllite.ipynb at main · UmerSherdil/Data_Science_and_ML (github.com)

Build an Interactive Map with Folium



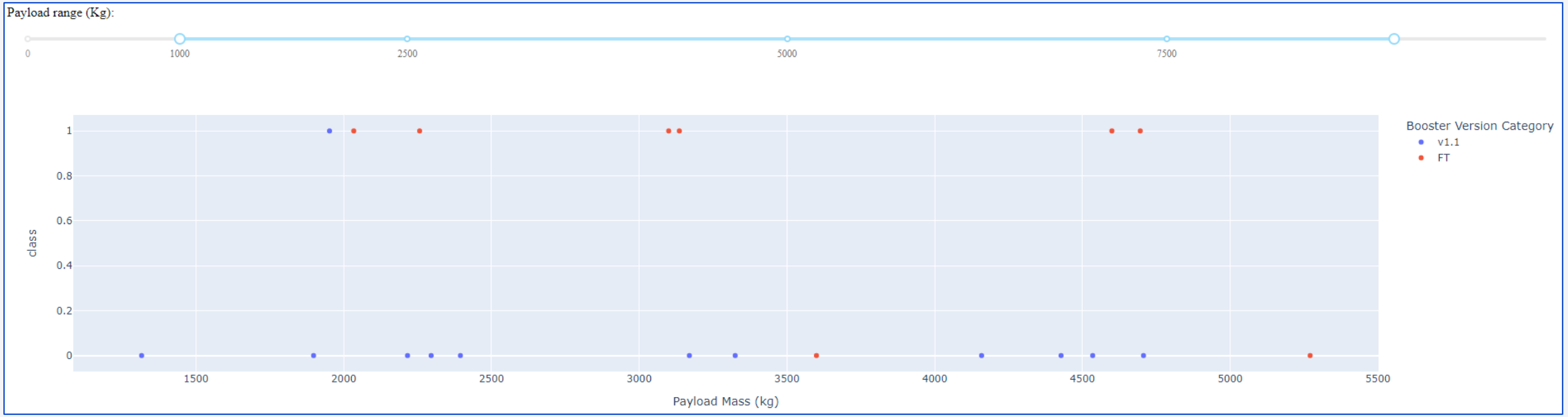
Data_Science_and_ML/lab_jupyter_launch_site_location.ipynb at main · UmerSherdil/Data_Science_and_ML (github.com)

Build a Dashboard with Plotly Dash



Data_Science_and_ML/spacex_dash_app.py at main · UmerSherdil/Data_Science_and_ML (github.com)

Build a Dashboard with Plotly Dash

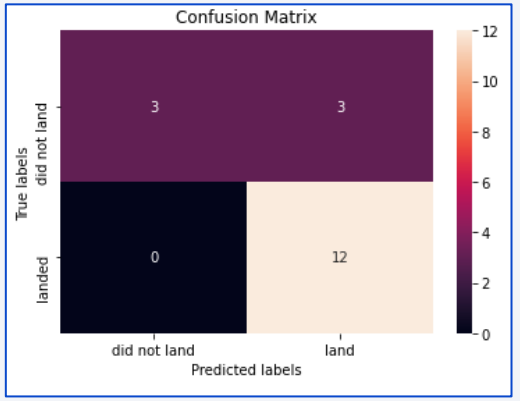


Data_Science_and_ML/spacex_dash_app.py at main · UmerSherdil/Data_Science_and_ML (github.com)

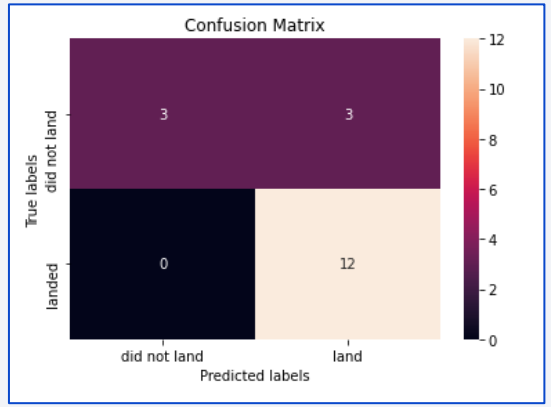
Predictive Analysis (Classification)

- Decision tree performs best on test data (88.89% accuracy).
- The accuracy of Logistic Regression, SVM and KNNs is 83.34%. (On test data).

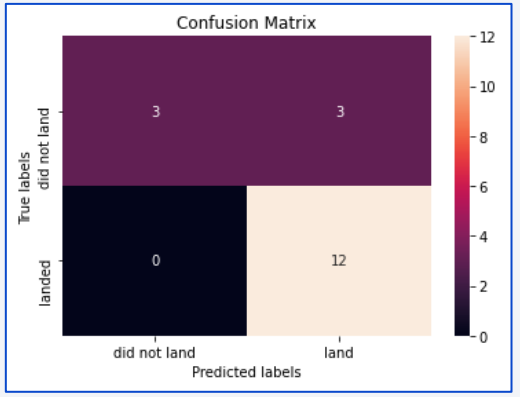
Logistic Regression



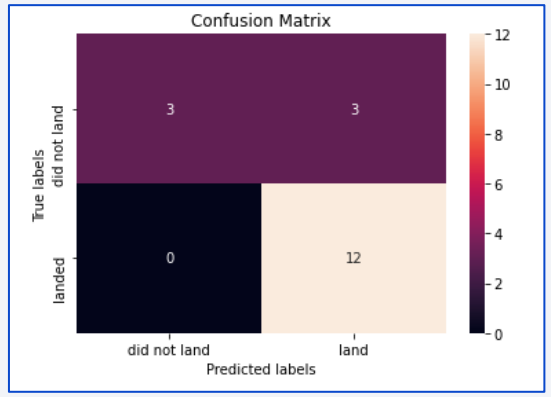
SVM



Decision Tree



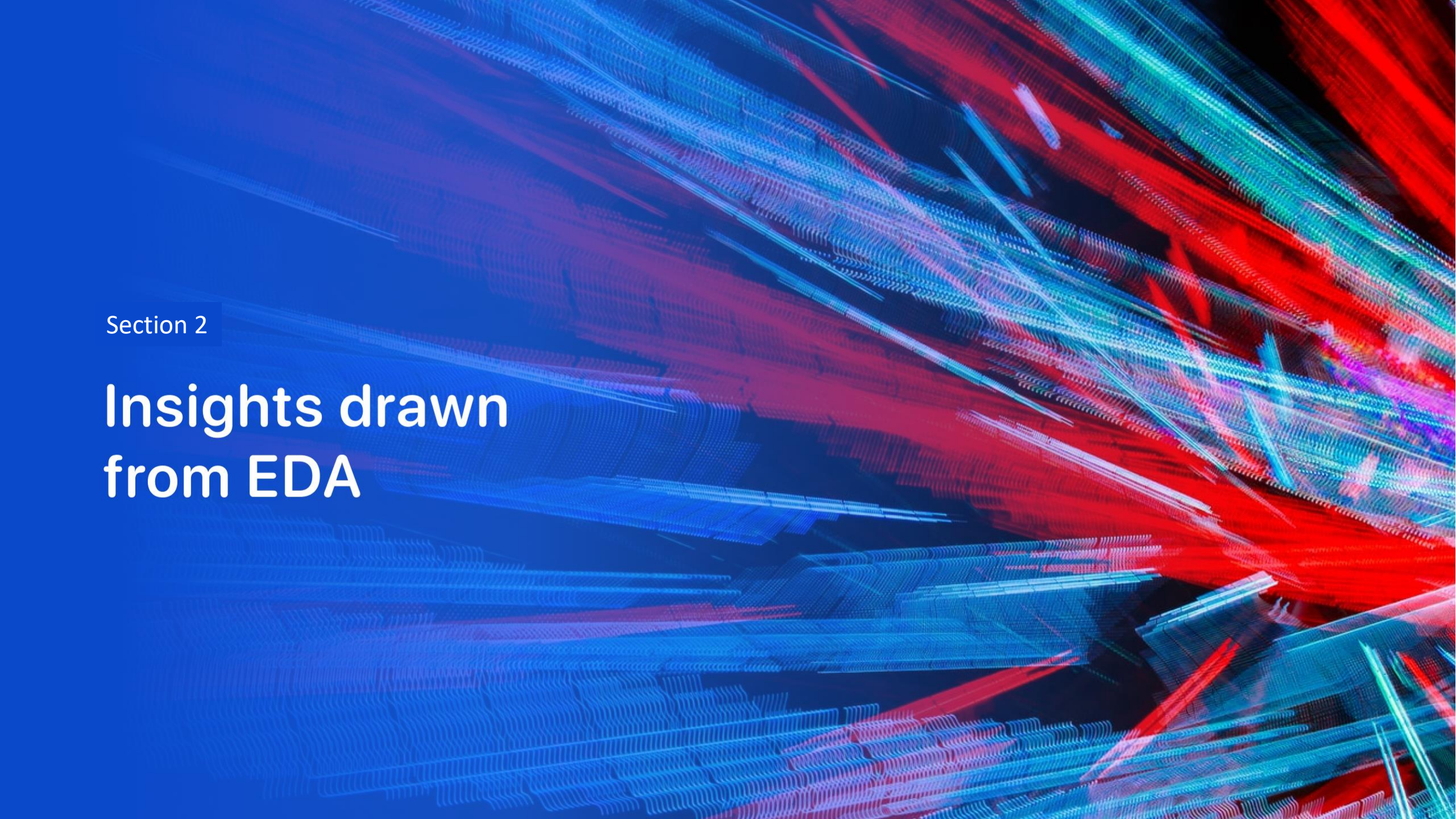
K-Nearest Neighbors (KNNs)



[Data_Science_and_ML/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · UmerSherdil/Data_Science_and_ML (github.com)]

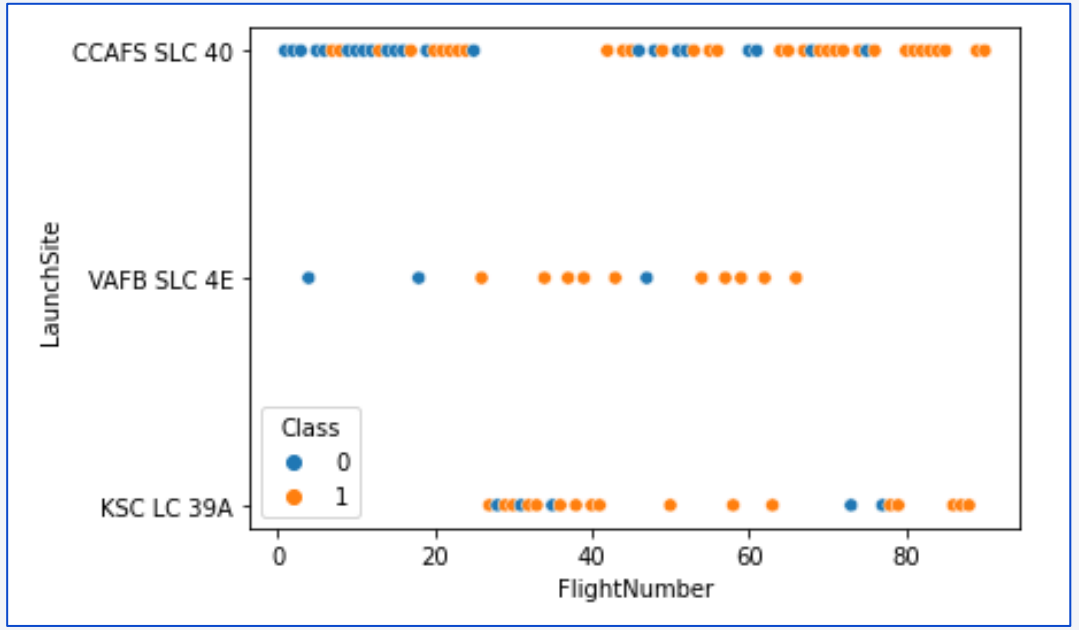
Results

- The success rate of launches increases with years. This makes sense as with time SpaceX improves the rocket design.
- KSC LC-39A launch site has the highest success rate.
- Orbits ES L1, HEO, GEO and SSO have the highest success rates.
- Decision tree performs best on test data with an accuracy of 88.89%.


Section 2
Insights drawn from EDA

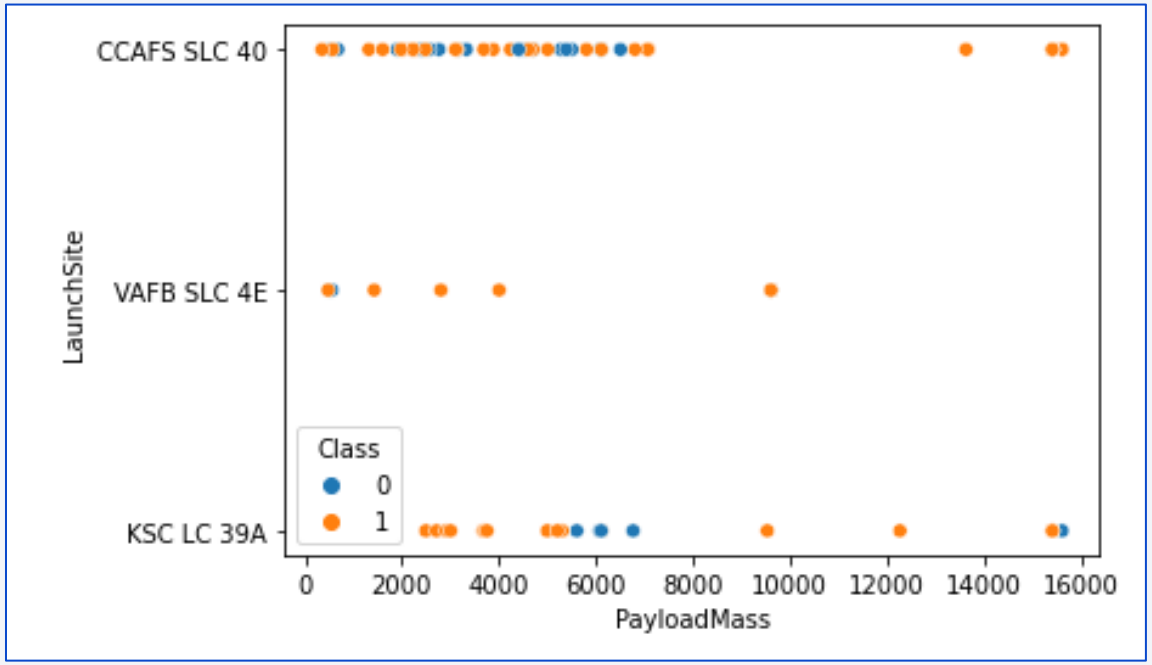
Flight Number vs. Launch Site

- Most launches were made from CCAFS SLC 40 launch site.



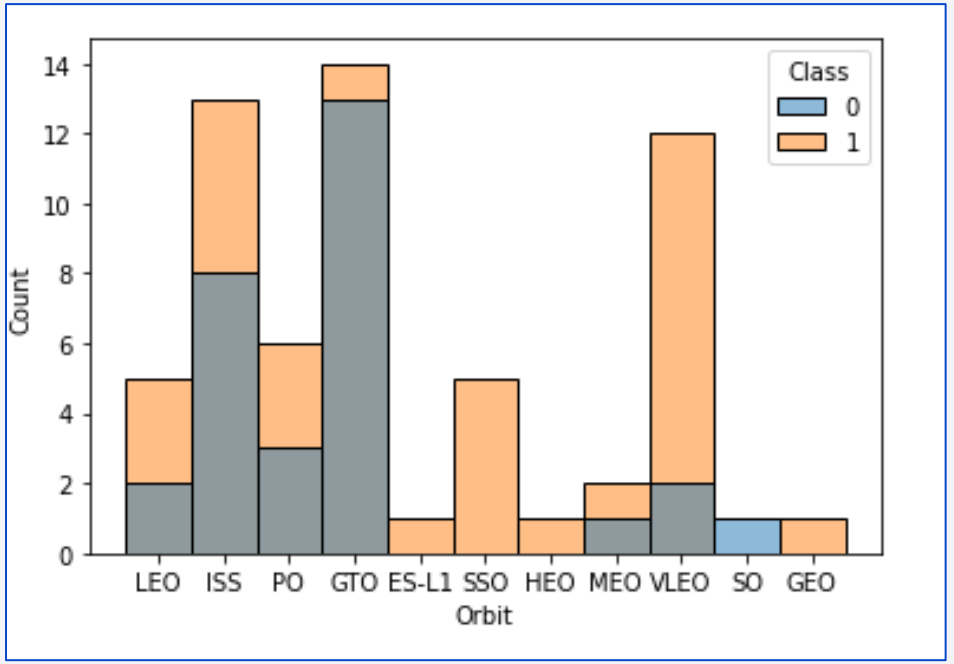
Payload vs. Launch Site

- Most lighter payloads were launched from CCAFS SLC 40.



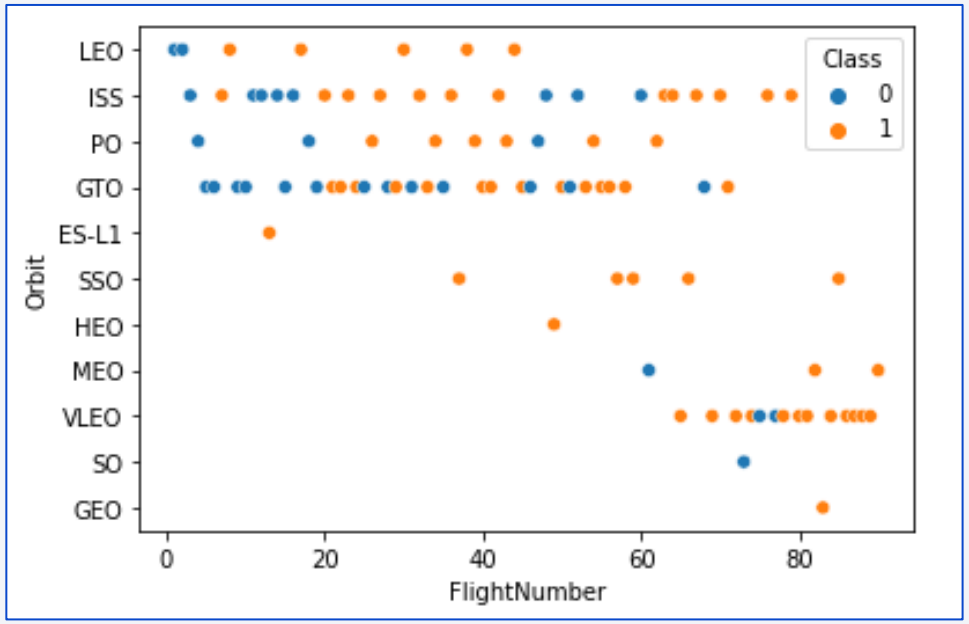
Success Rate vs. Orbit Type

- Orbits ES L1, HEO, GEO and SSO have the highest success rates.



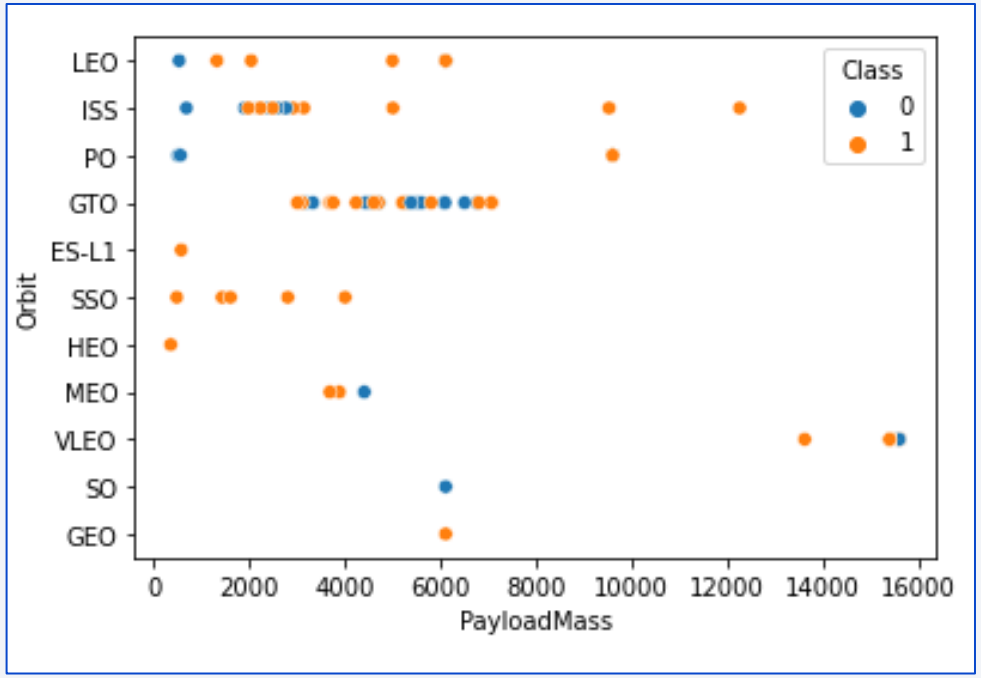
Flight Number vs. Orbit Type

- In the past more launches were made in LEO, ISS, PO and GTO orbit. However, with time the trend shifts towards VLEO orbit.



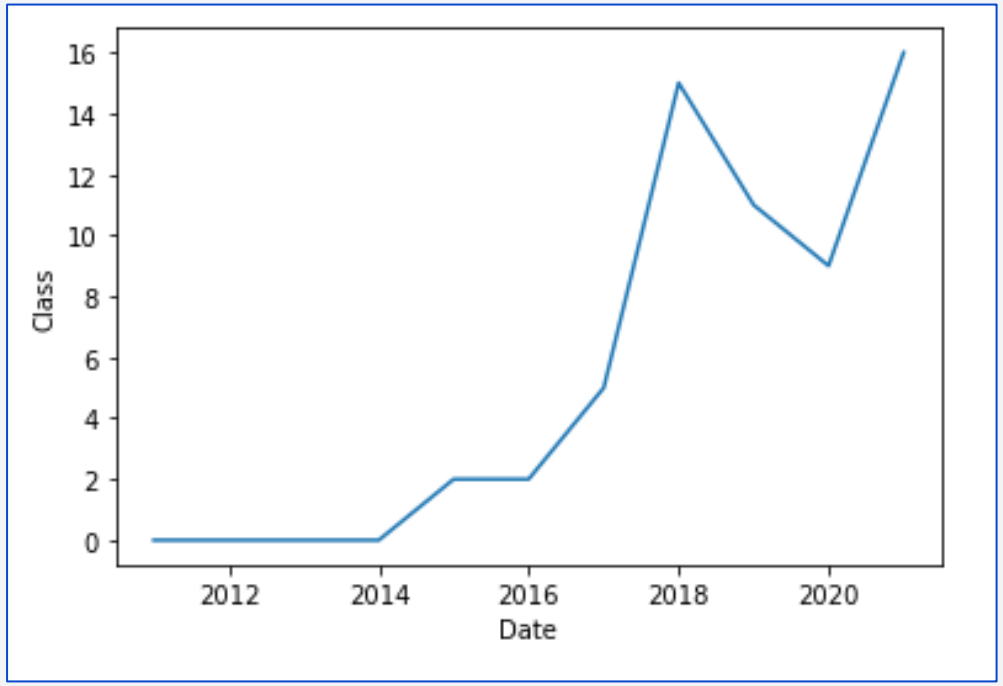
Payload vs. Orbit Type

- Payloads between the range of 2000-4000 kg were mostly sent in ISS orbit; whereas, relatively heavier payloads (4000-8000 kg) were sent in the GTO orbit.



Launch Success Yearly Trend

- Launch success rate increases significantly with time.



All Launch Site Names

```
print(cur.execute('Select distinct "Launch_Site" from SPACEXTBL').fetchall())
```

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

Launch Site Names Begin with 'CCA'

```
print(cur.execute('select * from "spacextbl" where "launch_site" like "cca%"').fetchall()[:5])
```

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

```
print(cur.execute('select sum(payload_mass__kg_) from spacextbl where customer="NASA (CRS)"').fetchall())
```

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

Average Payload Mass by F9 v1.1

```
print(cur.execute('select avg(payload_mass__kg_) from spacextbl where booster_version = "F9 v1.1"').fetchall())
```

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

First Successful Ground Landing Date

```
print(cur.execute("SELECT MIN(substr(Date,7)||'-'||substr(date,4,2)||'-'||substr(date,1,2)) FROM spacextbl WHERE \"Landing _Outcome\" = \"Success (ground pad)\"").fetchall())
```

```
[('2015-12-22',)]
```

Successful Drone Ship Landing with Payload between 4000 and 6000

```
print(cur.execute('SELECT "BOOSTER_VERSION" FROM "SPACEXTBL" WHERE "Landing _Outcome" = "Success (drone ship)" AND "PAYLOAD_MASS__KG_" > 4000 AND "PAYLOAD_MASS__KG_" < 6000').fetchall())
```

```
[('F9 FT B1022',), ('F9 FT B1026',), ('F9 FT  B1021.2',), ('F9 FT  B1031.2',)]
```

Total Number of Successful and Failure Mission Outcomes

```
print(cur.execute('SELECT COUNT("Mission_Outcome") FROM "SPACEXTBL" WHERE "Mission_Outcome" LIKE "Success%" OR "Mission_Outcome" LIKE "Failure%"').fetchall())
```

```
[(101,)]
```

Boosters Carried Maximum Payload

```
print(cur.execute('SELECT "Booster_Version" from "SPACEXTBL" WHERE \
                  "payload_mass__kg_" = (SELECT MAX(payload_mass__kg_) FROM "SPACEXTBL")').fetchall())
```

```
[('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

```
print(cur.execute('SELECT SUBSTR("Date", 4, 2), "Landing _Outcome", "Booster_Version", "Launch_Site" FROM "SPACEXTBL" WHERE "Landing _Outcome" = "Failure (drone ship)" \
                  AND substr(Date,7,4)="2015"').fetchall())
```

```
[('01', '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

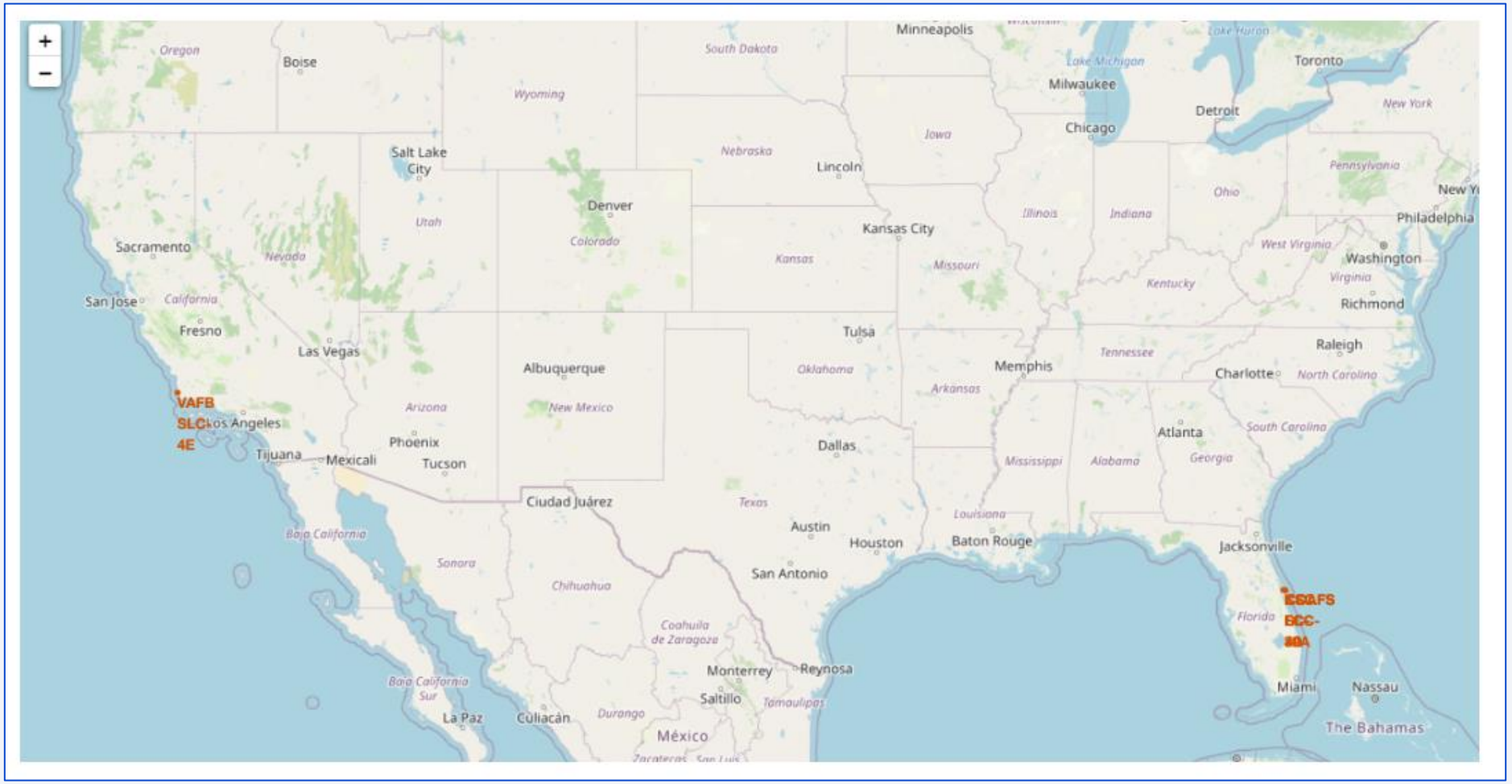
```
# || string concatenation operator in SQL Lite
print(cur.execute("SELECT \"Landing _Outcome\", COUNT(\"Landing _Outcome\") FROM SPACEXTBL \
                  WHERE substr(Date,7)||'-'||substr(date,4,2)||'-'||substr(date,1,2) BETWEEN '2010-06-04' AND '2017-03-20' \
                  GROUP BY \"Landing _Outcome\" \
                  ORDER BY COUNT(\"Landing _Outcome\") DESC").fetchall())
```

```
[('No attempt', 10), ('Success (drone ship)', 5), ('Failure (drone ship)', 5), ('Success (ground pad)', 3), ('Controlled (ocean)', 3), ('Uncontrolled (ocean)', 2), ('Failure (parachute)', 2), ('Precluded (drone ship)', 1)]
```

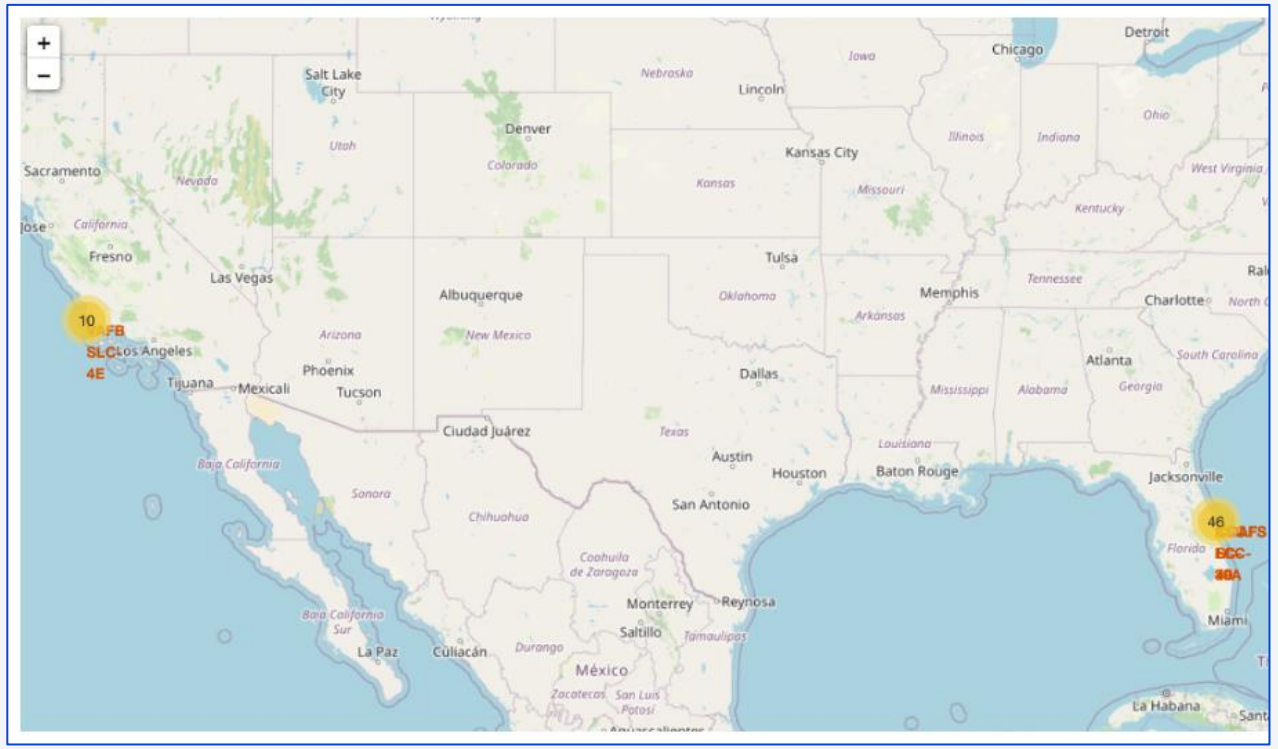
Section 3

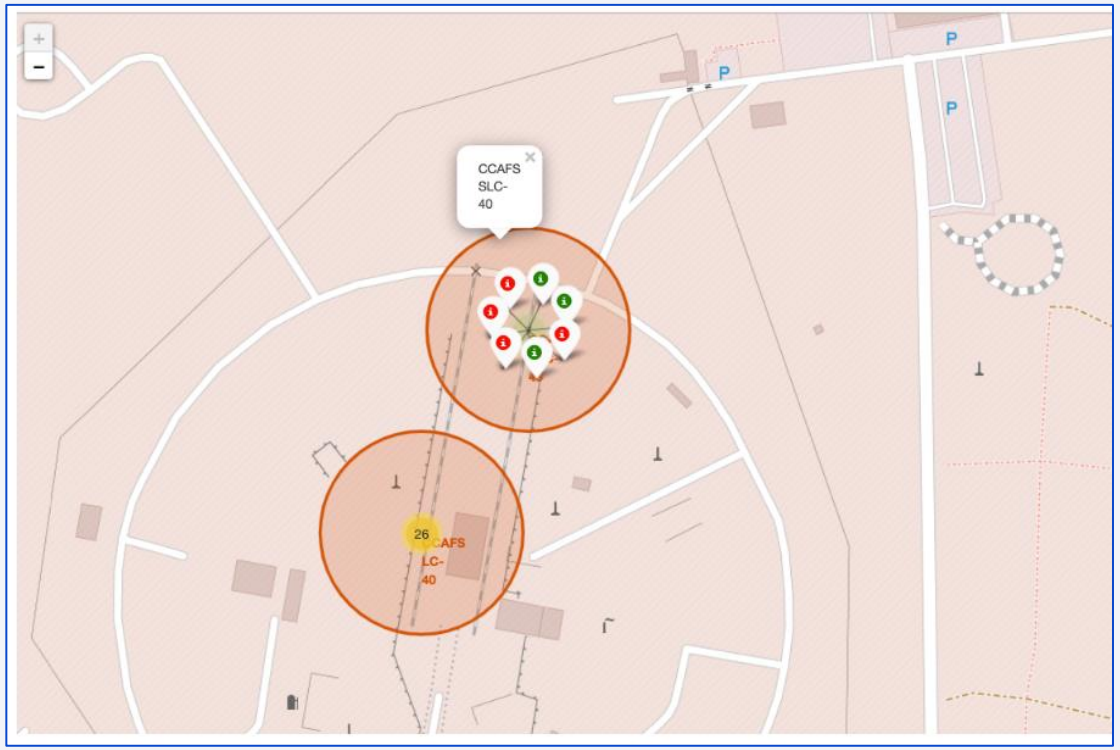
Launch Sites Proximities Analysis

Location of all Launch Sites

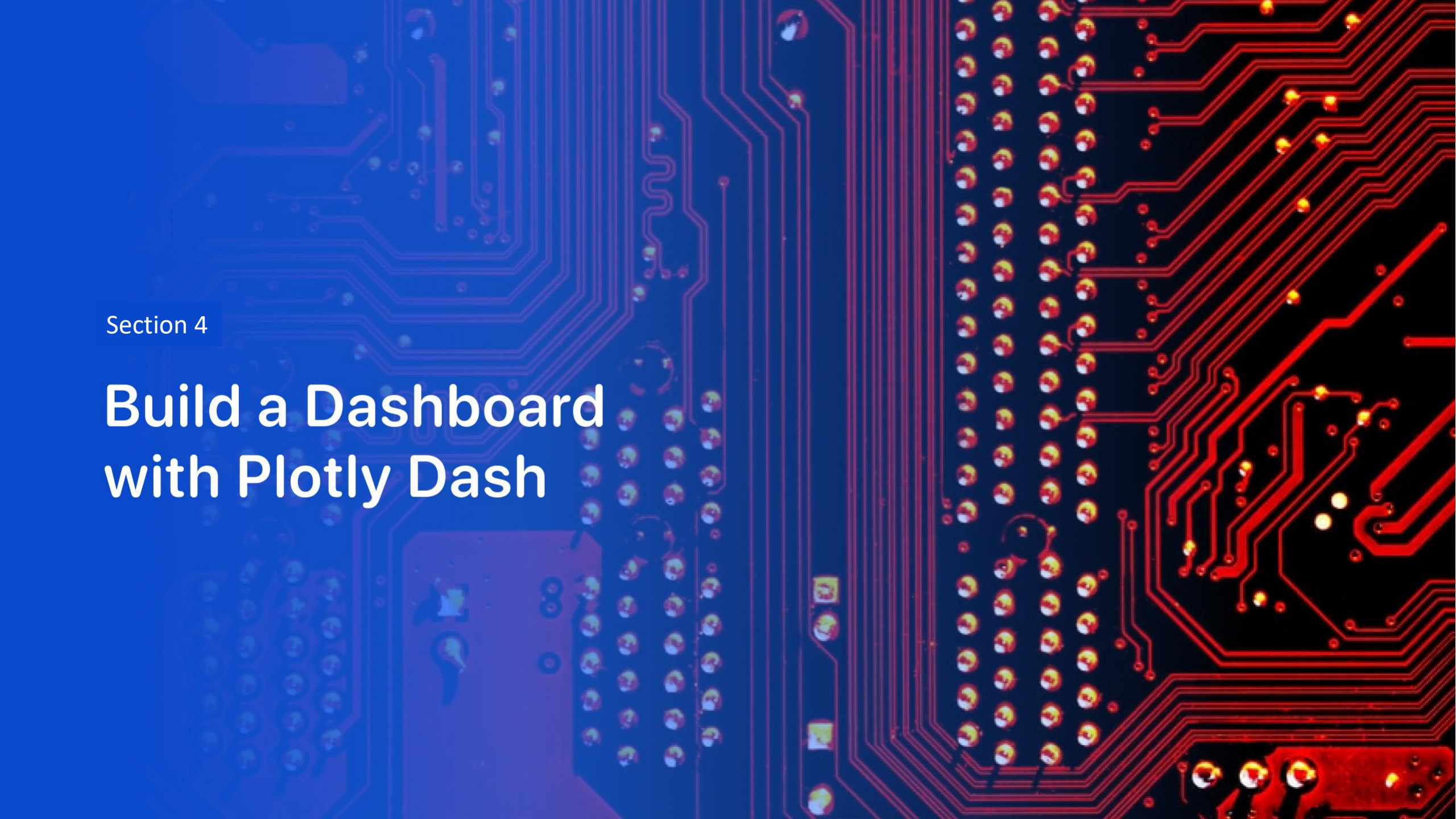


Success/Failed Launches



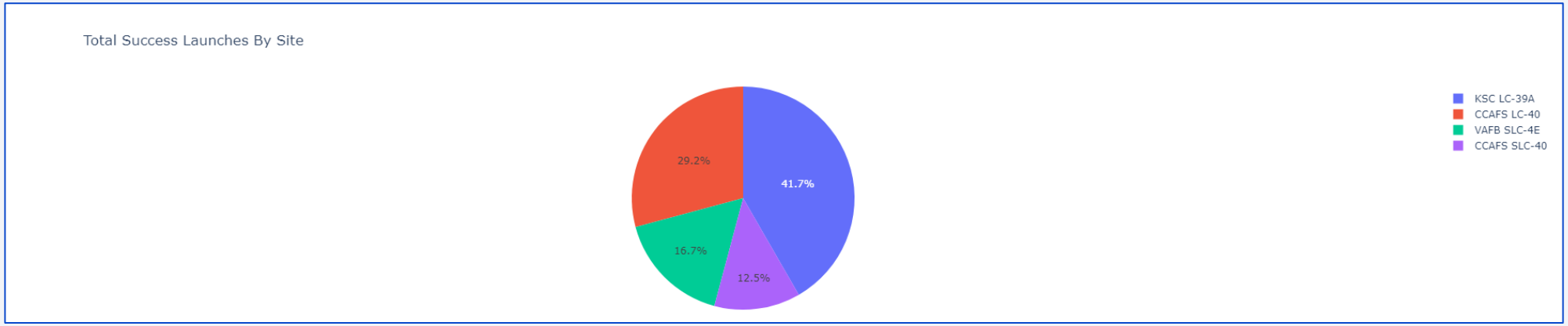


Distance between Launch Sites and its Proximities

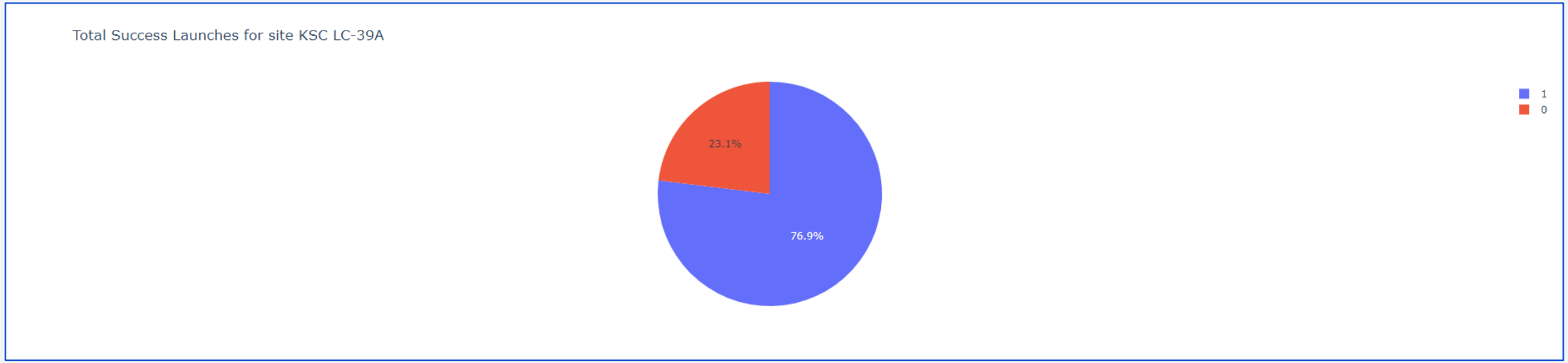

Section 4
Build a Dashboard with Plotly Dash

Success Rate of all Launch Sites

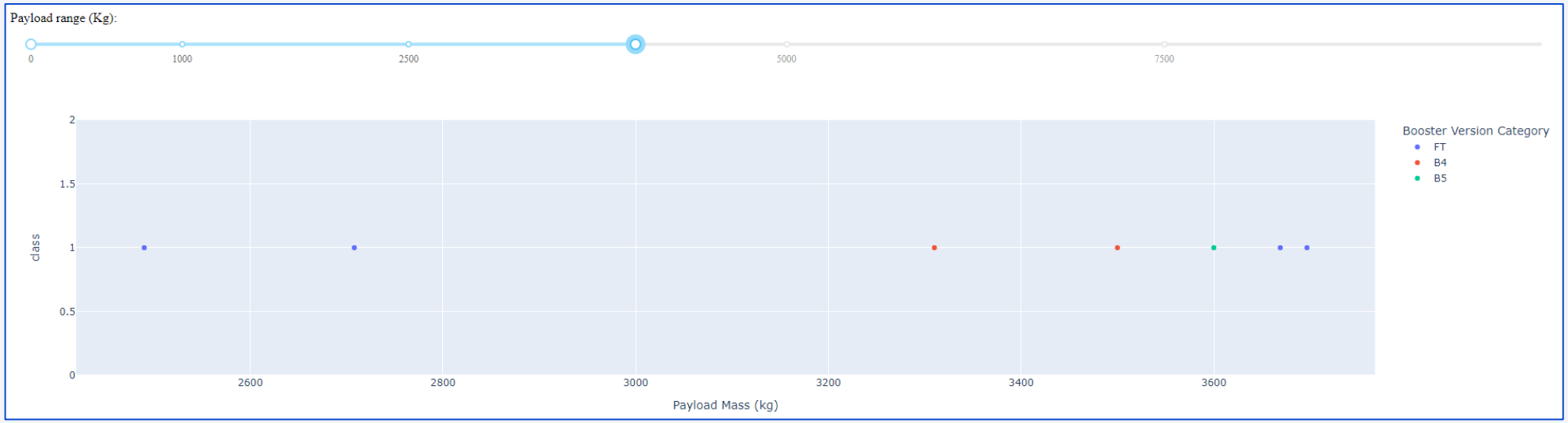
- KSC LC-39A has the most successful launches from all sites.

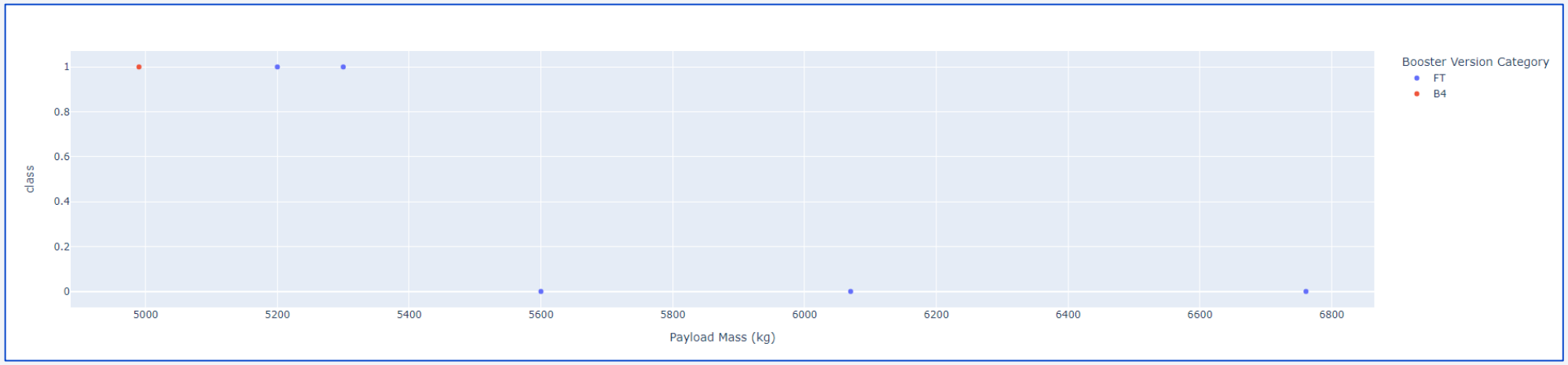


Success Rate of KSC LC-39A Launch Site



Success Rate and Payload Mass





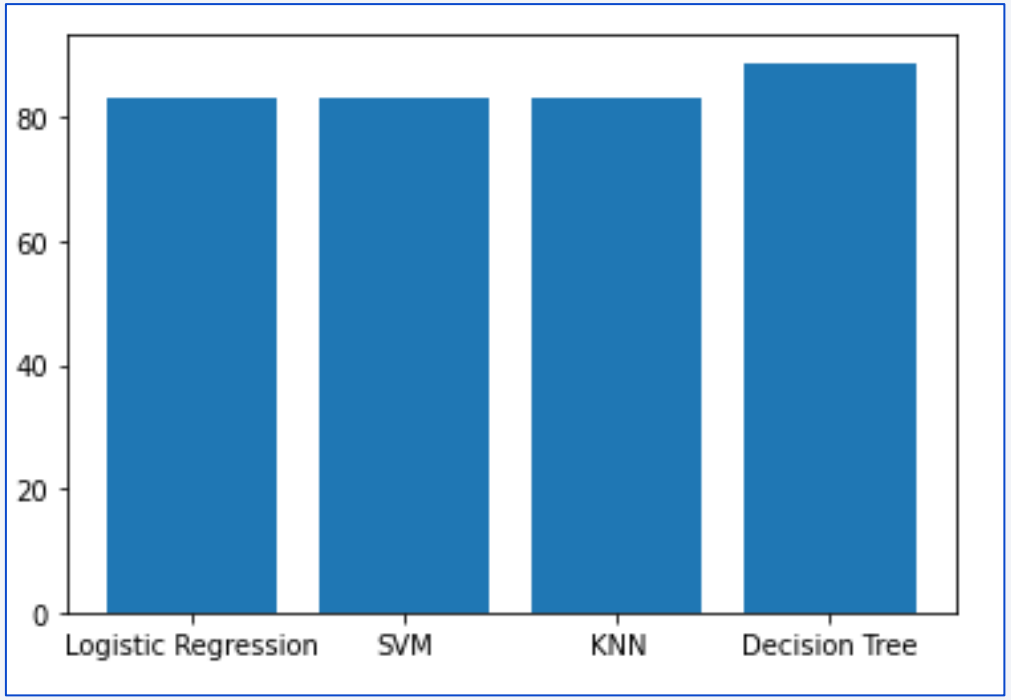
- The success rate of lighter payloads is higher than the heavier payloads.

Section 5

Predictive Analysis (Classification)

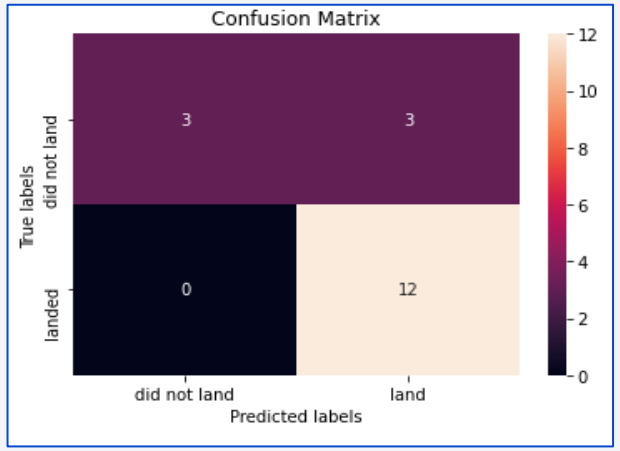
Classification Accuracy

- Decision trees perform the best on test data.

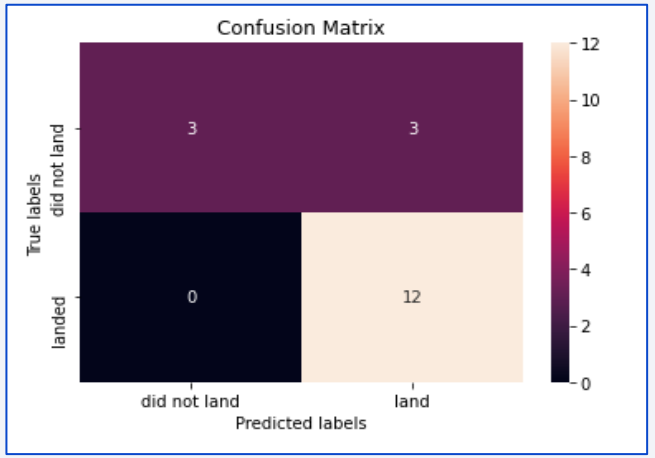


Confusion Matrix

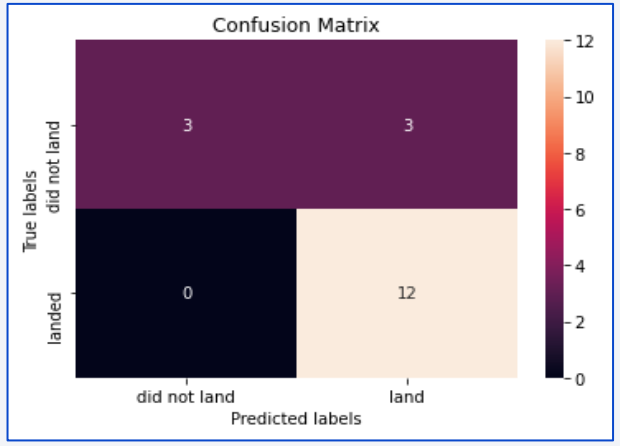
Logistic Regression



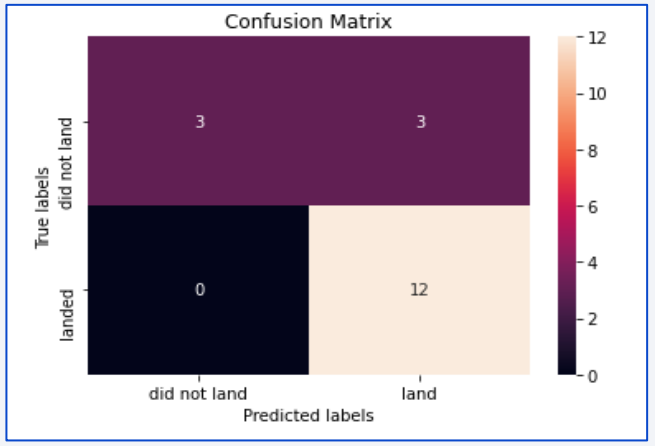
SVM



Decision Tree



K-Nearest Neighbors (KNNs)



Conclusions

- The success rate of launches increases with years. This makes sense as with time SpaceX improves the rocket design.
- KSC LC-39A launch site has the highest success rate.
- Orbits ES L1, HEO, GEO and SSO have the highest success rates.
- Ligher payloads perform better than heavier payloads.
- Decision tree performs best on test data with an accuracy of 88.89%.

Thank you!