IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

IVAN ZHURAVLEV
11/04/2023

Outline

Executive Summary

- Summary of methodologies
  - ❑ Data Collection with API
  - ❑ Data Collection with Web Scraping
  - ❑ Data Wrangling
  - ❑ Exploratory Data Analysis with SQL
  - ❑ Exploratory Data Analysis with Visualization
  - ❑ Interactive Visual Analytics and Dashboard with Folium and PlotlyDash
  - ❑ Machine Learning Predictive Analysis
- Summary of all results
  - ❑ Exploratory Data Analysis results
  - ❑ Interactive Visual Analysis results
  - ❑ Predictive Analysis results

Introduction

- Project background and context:

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.

The goal of the project is to create a data analysis pipeline to predict if the first stage will land successfully.

- Questions we will find answers to:
    - What factors determine if the rocket launch will result in a successful landing?
    - What combination of factors determine the success rate of a successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Request to the SpaceX API & Clean the requested data
  - Extracting Falcon 9 launch records from Wikipedia HTML table
- Perform data wrangling
  - Created variable that represents the outcome of each launch.
- 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

- Brief description on how data sets were collected:
  - ❑ Request to the SpaceX API
    - ➢ https://api.spacexdata.com/v4/
    - ➢ decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
    - ➢ clean the data, check for missing values and fill in missing values.
  - ❑ Extracting launch records from Wikipedia HTML table
    - ➢ web scraping from Wikipedia with BeautifulSoup
    - ➢ extract the launch records as HTML table, parse the table and convert it to a pandas dataframe

Data Collection – SpaceX API

- GitHub URL of the completed SpaceX API calls notebook:

https://github.com/iZotop79/IBM-Project/blob/b7f3d13088679fb647006e9ce4b26dec4223e217/data-collection-api.ipynb

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

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

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

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra rocket boosters and rows that have multiple payloads in a sing
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the list and replace the feature.
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date leaving the time
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

```
# Calculate the mean value of PayloadMass column
mean = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].fillna(mean, inplace=True)
data_falcon9.isnull().sum()
```

Data Collection - Scraping

- GitHub URL of the completed web scraping notebook:

https://github.com/iZotop79/IBM-Project/blob/b7f3d13088679fb647006e9ce4b26dec4223e217/webscraping.ipynb

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

```
response = requests.get(static_url).text
```

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

```
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
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
```

Data Wrangling

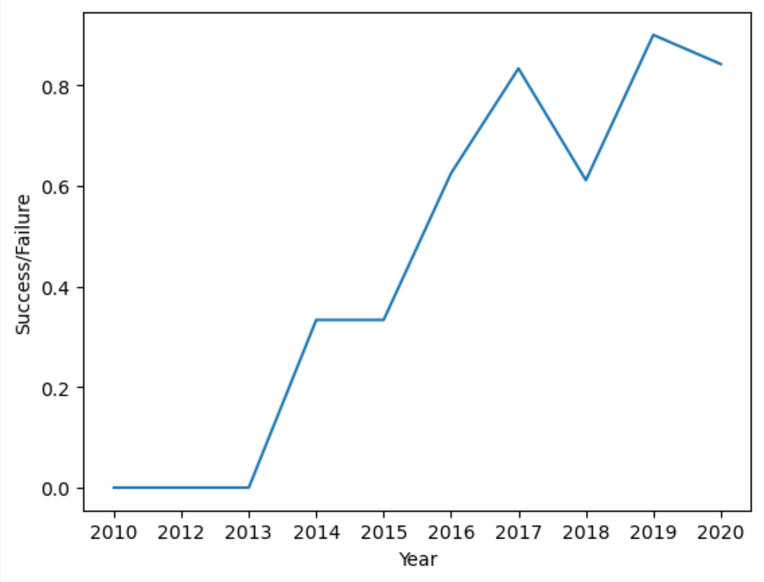
- Performed exploratory data analysis
- Calculated the number of launches on each site
- Calculated the number and occurrence of each orbit
- Calculated the number and occurrence of mission outcome per orbit type
- Created a landing outcome label from Outcome column

- GitHub URL of the completed data wrangling notebook:

https://github.com/iZotop79/IBM-Project/blob/b7f3d13088679fb647006e9ce4b26dec4223e217/data_wrangling_.ipynb

EDA with Data Visualization

- Plot out the Flight Number vs. Payload Mass & overlay the outcome of the launch
- Plot out the Flight Number vs. Launch Site & overlay the outcome of the launch
- Visualize the relationship between Payload and Launch Site
- Visualize the relationship between success rate of each orbit type
- Visualize the relationship between Flight Number and Orbit type
- Visualize the relationship between Payload and Orbit type
- Visualize the launch success yearly trend
- GitHub URL of the completed EDA with data visualization notebook:



https://github.com/iZotop79/IBM-Project/blob/b7f3d13088679fb647006e9ce4b26dec4223e217/EDA%20with%20Visualization.ipynb

EDA with SQL

- the SQL queries that were performed:
  - query1 = "SELECT DISTINCT LAUNCH_SITE FROM SPACEX“
  - query2 = "SELECT * FROM SPACEX WHERE LAUNCH_SITE LIKE 'KSC%' FETCH FIRST 5 ROWS ONLY“
  - query3 = "SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)’”
  - query4 = "SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEX WHERE BOOSTER_VERSION = 'F9 v1.1’”
  - query5 = "SELECT MIN(DATE) FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)’”
  - query6 = "SELECT BOOSTER_VERSION FROM SPACEX WHERE MISSION_OUTCOME = 'Success' AND LANDING__OUTCOME = 'Success (ground pad)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000“
  - query7 = "SELECT MISSION_OUTCOME, COUNT(*) FROM SPACEX GROUP BY MISSION_OUTCOME“
  - query8 = "SELECT BOOSTER_VERSION FROM SPACEX WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX)“
  - query9 = "SELECT TO_CHAR(DATE, 'MONTH'), LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE BETWEEN '2017-01-01' AND '2017-12-31' AND LANDING__OUTCOME = 'Success (ground pad)’”
  - query10 = "SELECT LANDING__OUTCOME, COUNT(*) FROM SPACEX WHERE MISSION_OUTCOME = 'Success' AND DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING__OUTCOME ORDER BY COUNT(*) DESC"
- GitHub URL of your completed EDA with SQL notebook:

https://github.com/iZotop79/IBM-Project/blob/9f1fe9d08e8322053c84f1bb9a9ee7c220d26f24/jupyter-labs-eda-sql-edx%20(1)%20(1).ipynb

Build an Interactive Map with Folium

- marked all launch sites on map and added objects such as markers, circles, lines to mark the success or failure of launches
- assigned the feature launch outcomes (failure or success)
- identified which launch sites have relatively high success rate with color-labeled marker clusters
- calculated the distances between a launch site to its proximities.
- showed if launch sites are near railways, highways and coastlines.
- GitHub URL of the completed interactive map with Folium:

https://github.com/iZotop79/IBM-Project/blob/9f1fe9d08e8322053c84f1bb9a9ee7c220d26f24/Launch%20Sites%20Locations%20Analysis%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- Added a dropdown list to enable Launch Site selection
- Add a pie chart to show the total successful launches count for all sites
- Add a slider to select payload range
- Add a scatter chart to show the correlation between payload and launch success
- GitHub URL of the completed Plotly Dash lab:

https://github.com/iZotop79/IBM-Project/blob/5187194e4f6d4e96e0846f7b648317085c7f0be9/spacex_dash_app.py

Predictive Analysis (Classification)

- Performed exploratory Data Analysis and determined Training Labels:
  - ❑ created a column for the class
  - ❑ Standardized the data
  - ❑ Split into training data and test data
- Built machine learning models and finetuned hyperparameters using GridSearchCV:
  - ❑ for support vector machine,
  - ❑ k nearest neighbors
  - ❑ decision tree classifier
  - ❑ and Logistic Regression
- Determined the method that performs the best in terms of accuracy using test data
- GitHub URL of the completed predictive analysis lab:

https://github.com/iZotop79/IBM-Project/blob/5187194e4f6d4e96e0846f7b648317085c7f0be9/Machine%20Learning%20Predictions.ipynb

Results

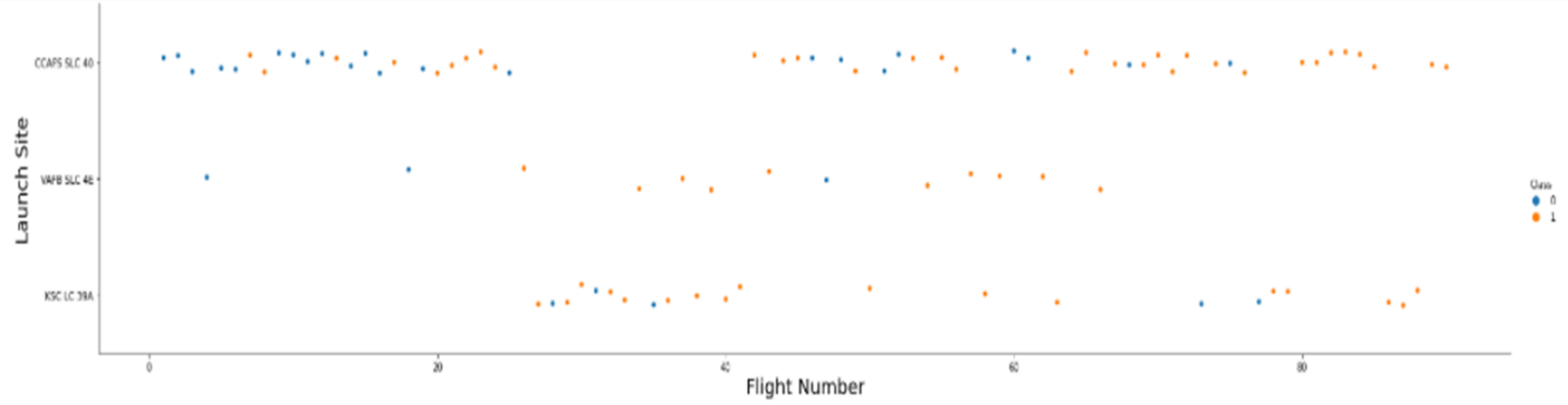
- Exploratory data analysis results
  - https://github.com/iZotop79/IBM-Project/blob/9f1fe9d08e8322053c84f1bb9a9ee7c220d26f24/jupyter-labs-eda-sql-edx%20(1)%20(1).ipynb
- Interactive analytics demo in screenshots
  - https://github.com/iZotop79/IBM-Project/blob/b7f3d13088679fb647006e9ce4b26dec4223e217/EDA%20with%20Visualization.ipynb
- Predictive analysis results
  - https://github.com/iZotop79/IBM-Project/blob/5187194e4f6d4e96e0846f7b648317085c7f0be9/Machine%20Learning%20Predictions.ipynb

Section 2

Insights drawn from EDA

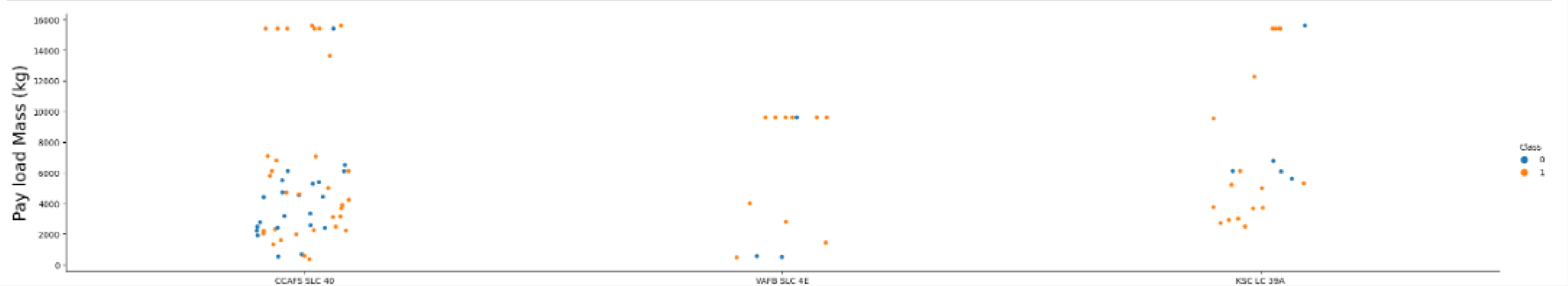
Flight Number vs. Launch Site

- As the Flight Numbers increase so does the success rate



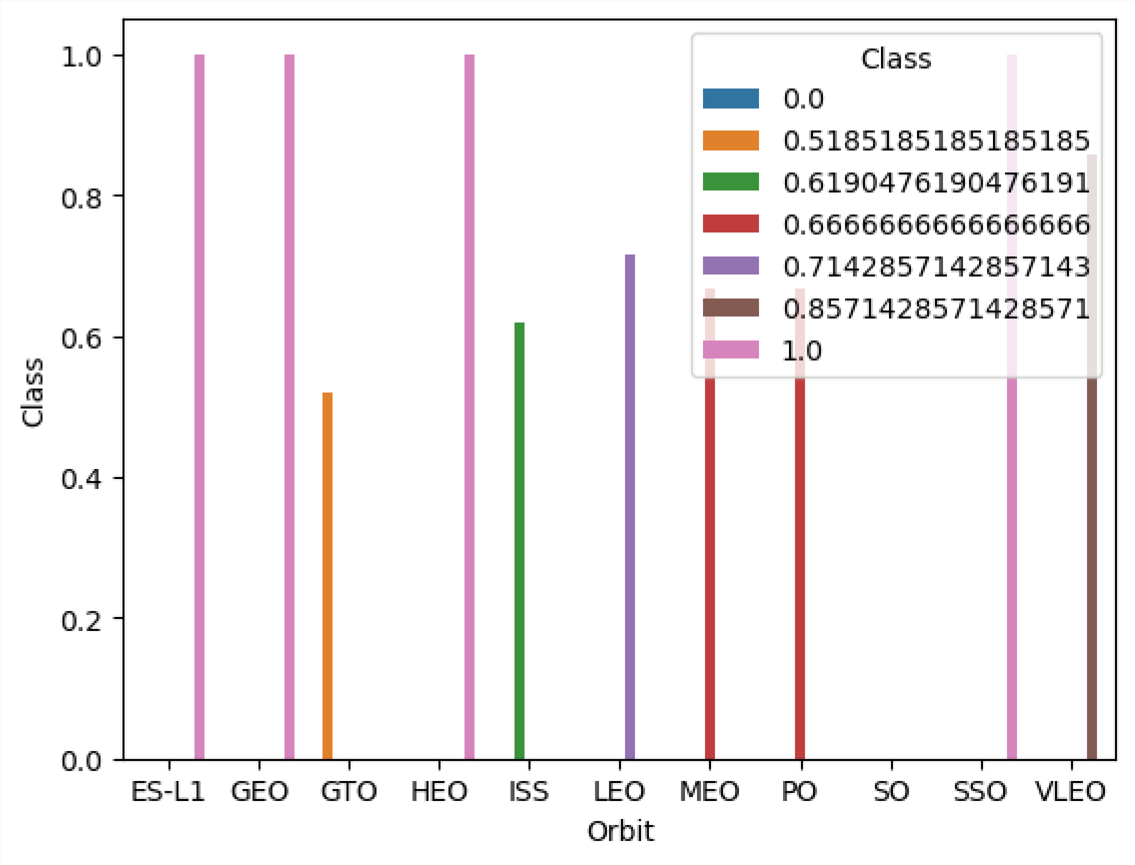
Payload vs. Launch Site

- With higher Payload the flights are more successful on all Launch Sites



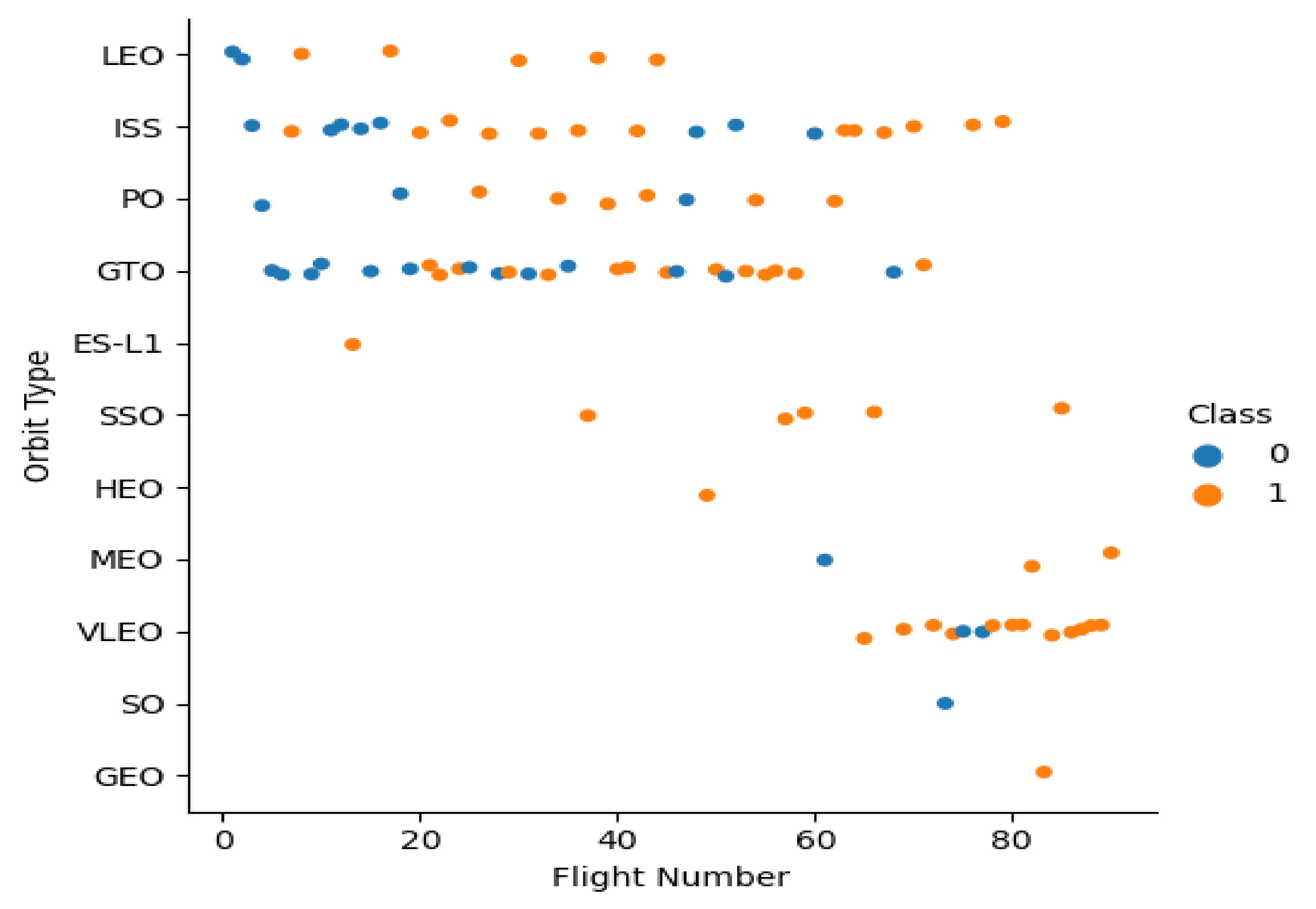
Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



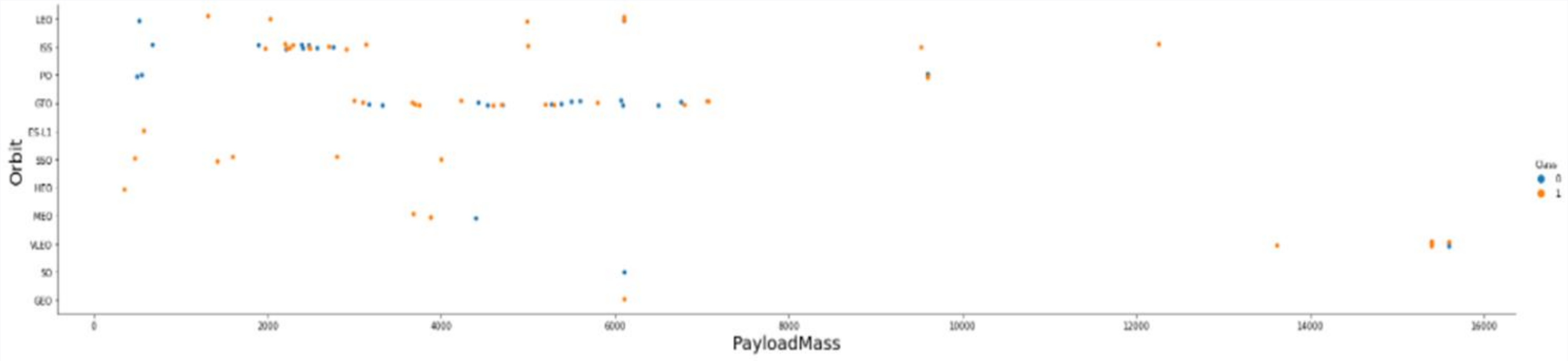
Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type



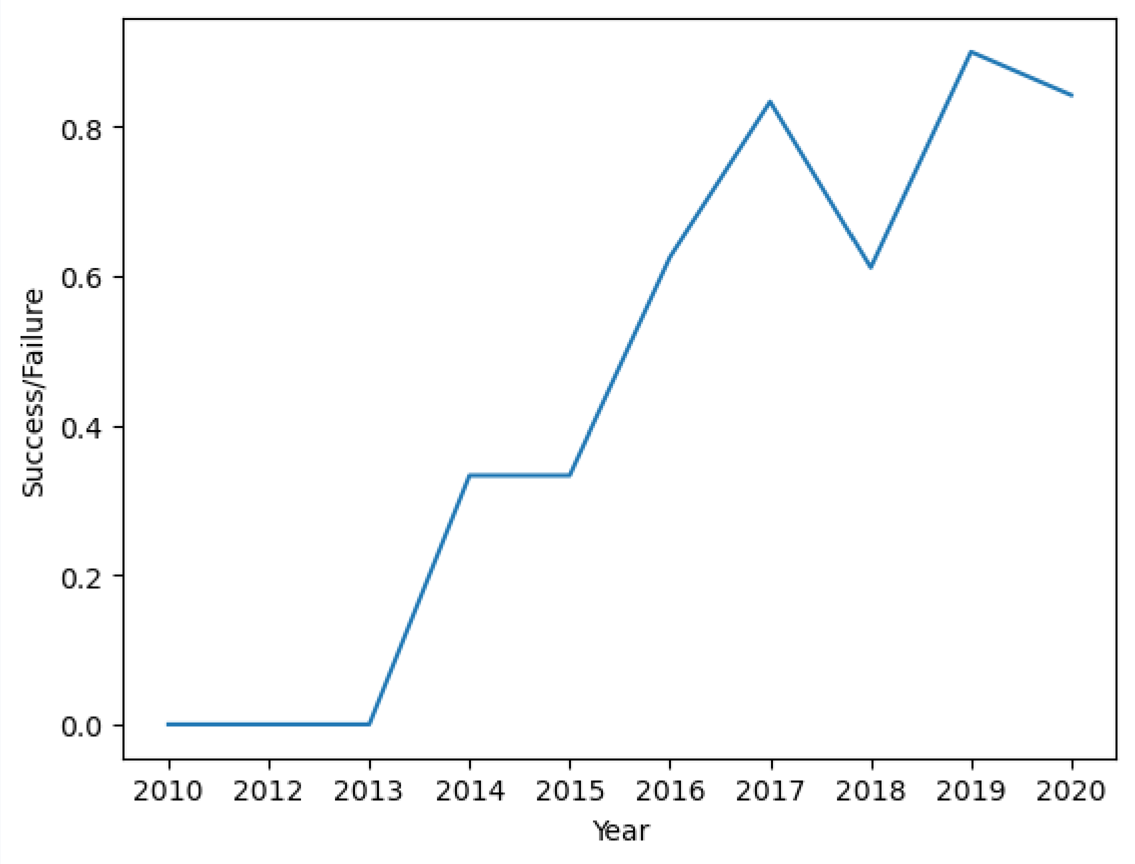
Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type



Launch Success Yearly Trend

- that success rate since 2013 kept an increasing trend



All Launch Site Names

- query = "SELECT DISTINCT LAUNCH_SITE FROM SPACEX“
- names of the unique launch sites:

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

Launch Site Names Begin with 'KSC'

- query = "SELECT * FROM SPACEX WHERE LAUNCH_SITE LIKE 'KSC%' FETCH FIRST 5 ROWS ONLY"
- Find 5 records where launch sites' names start with `KSC`


```
{'DATE': datetime.date(2017, 2, 19), 0: datetime.date(2017, 2, 19), 'TIME__UTC_': datetime.time(14, 39), 1: datetime.time(14, 39), 'BOOSTER_VERSION': 'F9 FT B1031.1', 2: 'F9 FT B1031.1', 'LAUNCH_SITE': 'KSC LC-39A', 3: 'KSC LC-39A', 'PAYLOAD': 'SpaceX CRS-10', 4: 'SpaceX CRS-10', 'PAYLOAD_MASS__KG_': 2490, 5: 2490, 'ORBIT': 'LEO (ISS)', 6: 'LEO (ISS)', 'CUSTOMER': 'NASA (CRS)', 7: 'NASA (CRS)', 'MISSION_OUTCOME': 'Success', 8: 'Success', 'LANDING__OUTCOME': 'Success (ground pad)', 9: 'Success (ground pad)'}
{'DATE': datetime.date(2017, 3, 16), 0: datetime.date(2017, 3, 16), 'TIME__UTC_': datetime.time(6, 0), 1: datetime.time(6, 0), 'BOOSTER_VERSION': 'F9 FT B1030', 2: 'F9 FT B1030', 'LAUNCH_SITE': 'KSC LC-39A', 3: 'KSC LC-39A', 'PAYLOAD': 'EchoStar 23', 4: 'EchoStar 23', 'PAYLOAD_MASS__KG_': 5600, 5: 5600, 'ORBIT': 'GTO', 6: 'GTO', 'CUSTOMER': 'EchoStar', 7: 'EchoStar', 'MISSION_OUTCOME': 'Success', 8: 'Success', 'LANDING__OUTCOME': 'No attempt', 9: 'No attempt'}
{'DATE': datetime.date(2017, 3, 30), 0: datetime.date(2017, 3, 30), 'TIME__UTC_': datetime.time(22, 27), 1: datetime.time(22, 27), 'BOOSTER_VERSION': 'F9 FT  B1021.2', 2: 'F9 FT  B1021.2', 'LAUNCH_SITE': 'KSC LC-39A', 3: 'KSC LC-39A', 'PAYLOAD': 'SES-10', 4: 'SES-10', 'PAYLOAD_MASS__KG_': 5300, 5: 5300, 'ORBIT': 'GTO', 6: 'GTO', 'CUSTOMER': 'SES', 7: 'SES', 'MISSION_OUTCOME': 'Success', 8: 'Success', 'LANDING__OUTCOME': 'Success (drone ship)', 9: 'Success (drone ship)'}
{'DATE': datetime.date(2017, 5, 1), 0: datetime.date(2017, 5, 1), 'TIME__UTC_': datetime.time(11, 15), 1: datetime.time(11, 15), 'BOOSTER_VERSION': 'F9 FT B1032.1', 2: 'F9 FT B1032.1', 'LAUNCH_SITE': 'KSC LC-39A', 3: 'KSC LC-39A', 'PAYLOAD': 'NROL-76', 4: 'NROL-76', 'PAYLOAD_MASS__KG_': 5300, 5: 5300, 'ORBIT': 'LEO', 6: 'LEO', 'CUSTOMER': 'NRO', 7: 'NRO', 'MISSION_OUTCOME': 'Success', 8: 'Success', 'LANDING__OUTCOME': 'Success (ground pad)', 9: 'Success (ground pad)'}
{'DATE': datetime.date(2017, 5, 15), 0: datetime.date(2017, 5, 15), 'TIME__UTC_': datetime.time(23, 21), 1: datetime.time(23, 21), 'BOOSTER_VERSION': 'F9 FT B1034', 2: 'F9 FT B1034', 'LAUNCH_SITE': 'KSC LC-39A', 3: 'KSC LC-39A', 'PAYLOAD': 'Inmarsat-5 F4', 4: 'Inmarsat-5 F4', 'PAYLOAD_MASS__KG_': 6070, 5: 6070, 'ORBIT': 'GTO', 6: 'GTO', 'CUSTOMER': 'Inmarsat', 7: 'Inmarsat', 'MISSION_OUTCOME': 'Success', 8: 'Success', 'LANDING__OUTCOME': 'No attempt', 9: 'No attempt'}
```

Total Payload Mass

- query = "SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)'"
- Calculate the total payload carried by boosters from NASA:

```
Total Payload Mass for NASA (CRS): 45596 kg
```

Average Payload Mass by F9 v1.1

- query = "SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEX WHERE BOOSTER_VERSION = 'F9 v1.1'"
- Calculate the average payload mass carried by booster version F9 v1.1:

```
Average Payload Mass for F9 v1.1: 2928 kg
```

First Successful Ground Landing Date

- query = "SELECT MIN(DATE) FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)'"
- Find the dates of the first successful landing outcome on drone ship:

```
2016-04-08
```

Successful Drone Ship Landing with Payload between 4000 and 6000

- query = "SELECT BOOSTER_VERSION FROM SPACEX WHERE MISSION_OUTCOME = 'Success' AND LANDING__OUTCOME = 'Success (ground pad)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000“
- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000:

```
F9 FT B1032.1
F9 B4 B1040.1
```

Total Number of Successful and Failure Mission Outcomes

- query = "SELECT MISSION_OUTCOME, COUNT(*) FROM SPACEX GROUP BY MISSION_OUTCOME“
- Calculate the total number of successful and failure mission outcomes:

```
Failure (in flight): 1
Success: 99
Success (payload status unclear): 1
```

Boosters Carried Maximum Payload

- query = "SELECT BOOSTER_VERSION FROM SPACEX WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX)“
- List the names of the booster which have carried the maximum payload mass:

```
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
```

2017 Launch Records

- query = "SELECT TO_CHAR(DATE, 'MONTH'), LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE BETWEEN '2017-01-01' AND '2017-12-31' AND LANDING__OUTCOME = 'Success (ground pad)'"
- List the records which will display the month names, succesful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017:

	F	S	F	K
0	MAY	Success (ground pad)	F9 FT B1032.1	KSC LC-39A
1	JUNE	Success (ground pad)	F9 FT B1035.1	KSC LC-39A
2	AUGUST	Success (ground pad)	F9 B4 B1039.1	KSC LC-39A
3	SEPTEMBER	Success (ground pad)	F9 B4 B1040.1	KSC LC-39A
4	DECEMBER	Success (ground pad)	F9 FT B1035.2	CCAFS SLC-40

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

- query = "SELECT LANDING__OUTCOME, COUNT(*) FROM SPACEX WHERE MISSION_OUTCOME = 'Success' AND DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING__OUTCOME ORDER BY COUNT(*) DESC"
- Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order:

```
No attempt: 10
Failure (drone ship): 5
Success (drone ship): 5
Controlled (ocean): 3
Success (ground pad): 3
Failure (parachute): 2
Uncontrolled (ocean): 2
```

Section 3

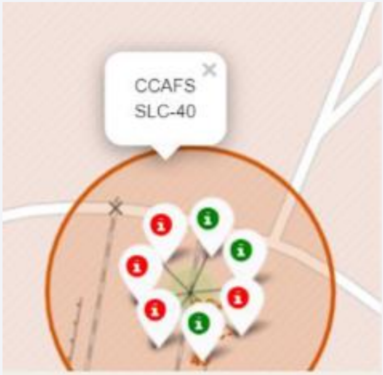
Launch Sites Proximities Analysis

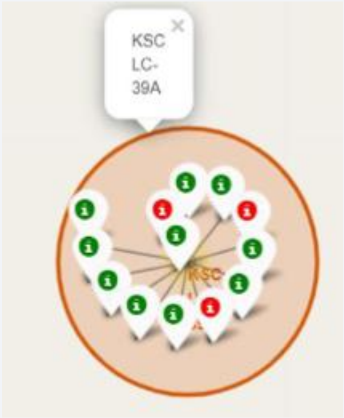
launch sites global map markers

- All launch sites are in the USA coasts, Florida and California.

Showing launch sites with colour labels

- Green Marker shows successful launches and Red Marker shows failures





<Folium Map Screenshot 3>

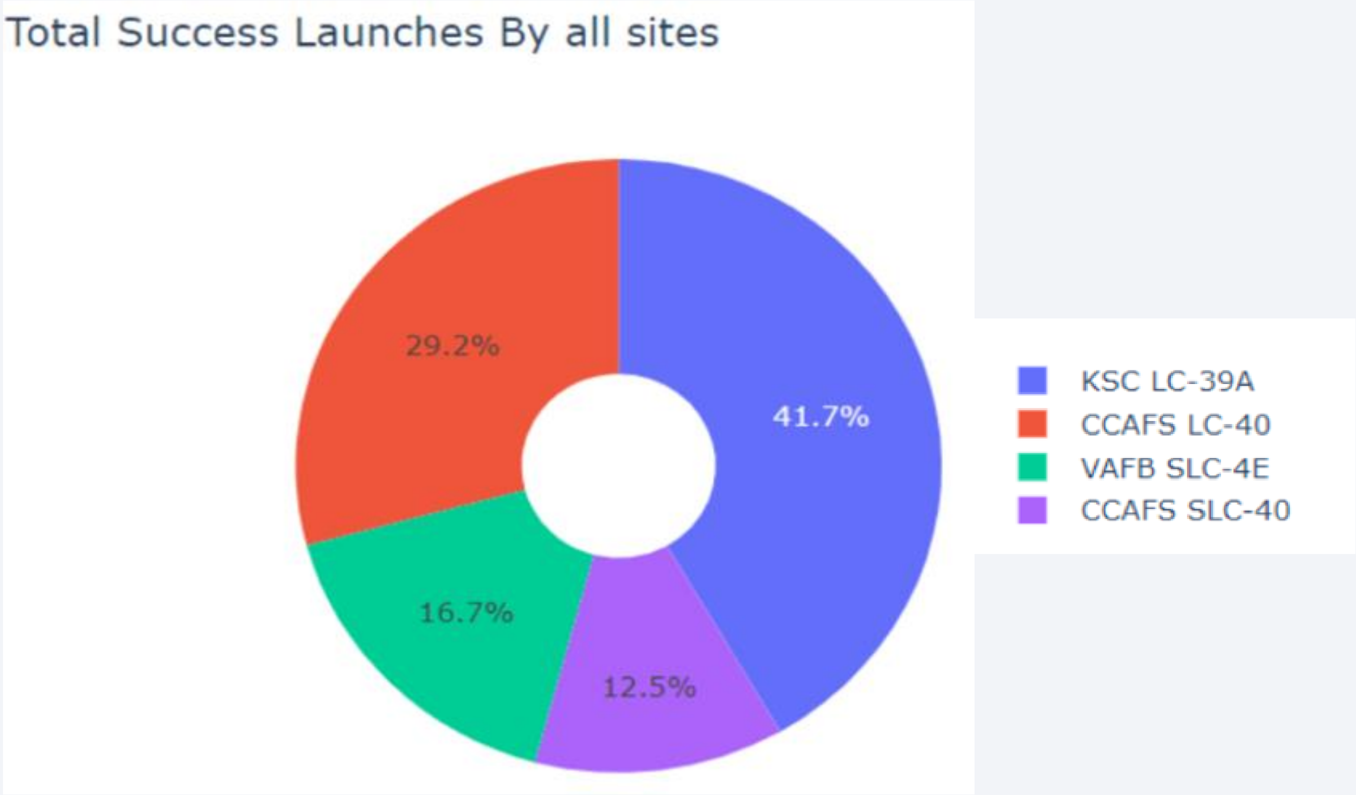
- Replace <Folium map screenshot 3> title with an appropriate title
- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed
- Explain the important elements and findings on the screenshot

Section 4

Build a Dashboard with Plotly Dash

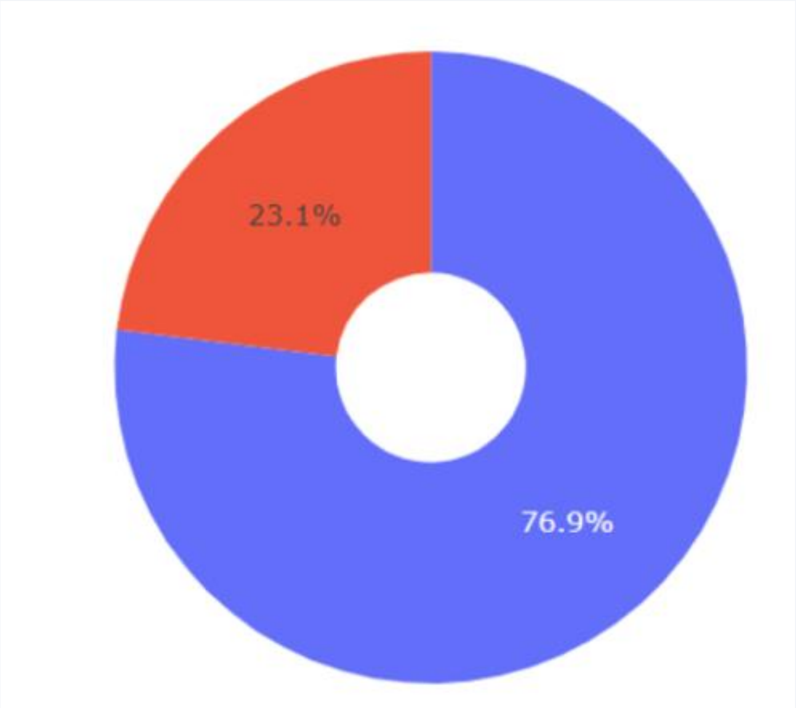
launch success count for all sites

- KSC LC-39A has the most successful launches.



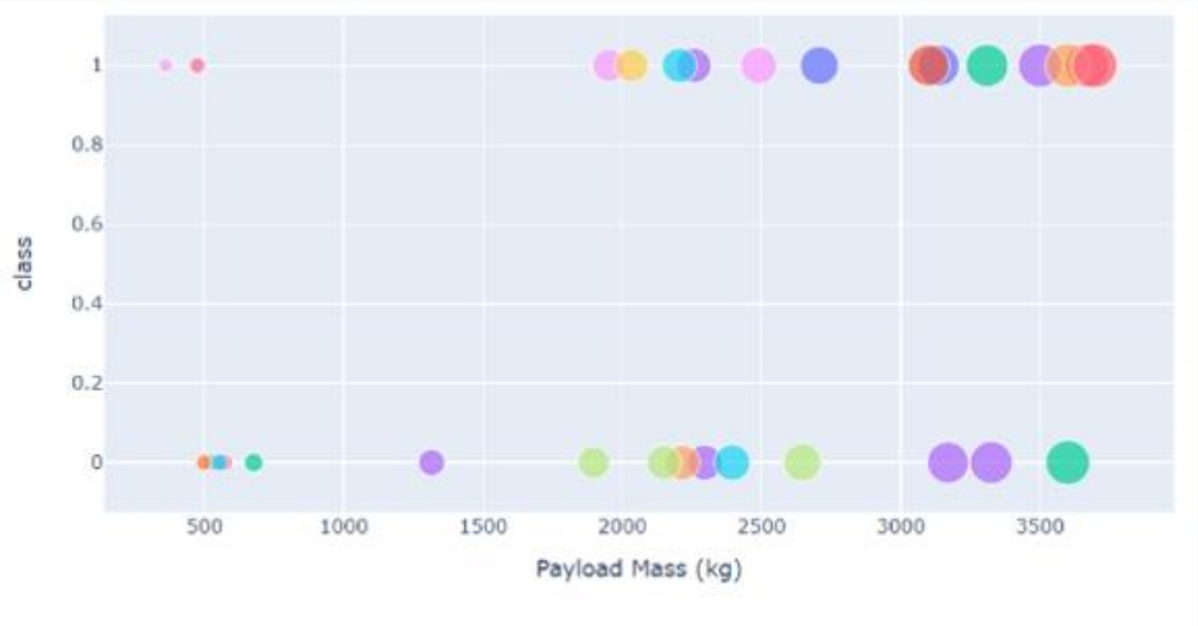
the launch site with highest launch success ratio

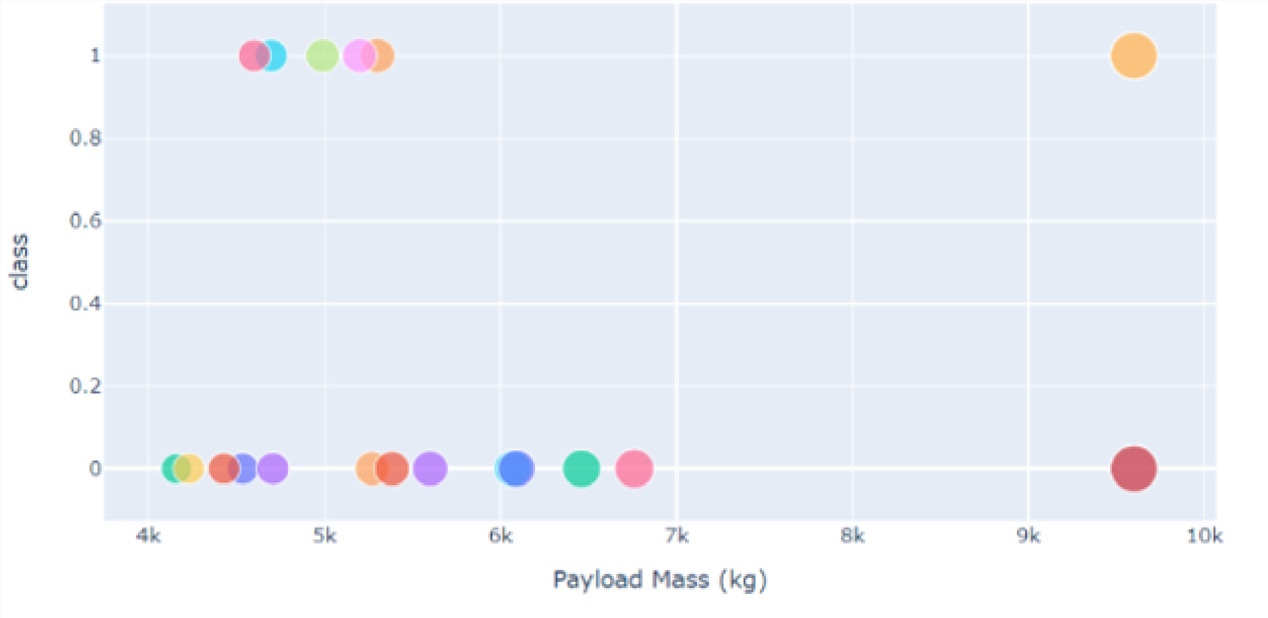
- KSC LC-39A has a 76.9% success rate



Payload vs. Launch Outcome scatter plot

- Success rate for low weighted payloads is higher





Section 5

Predictive Analysis (Classification)

Classification Accuracy

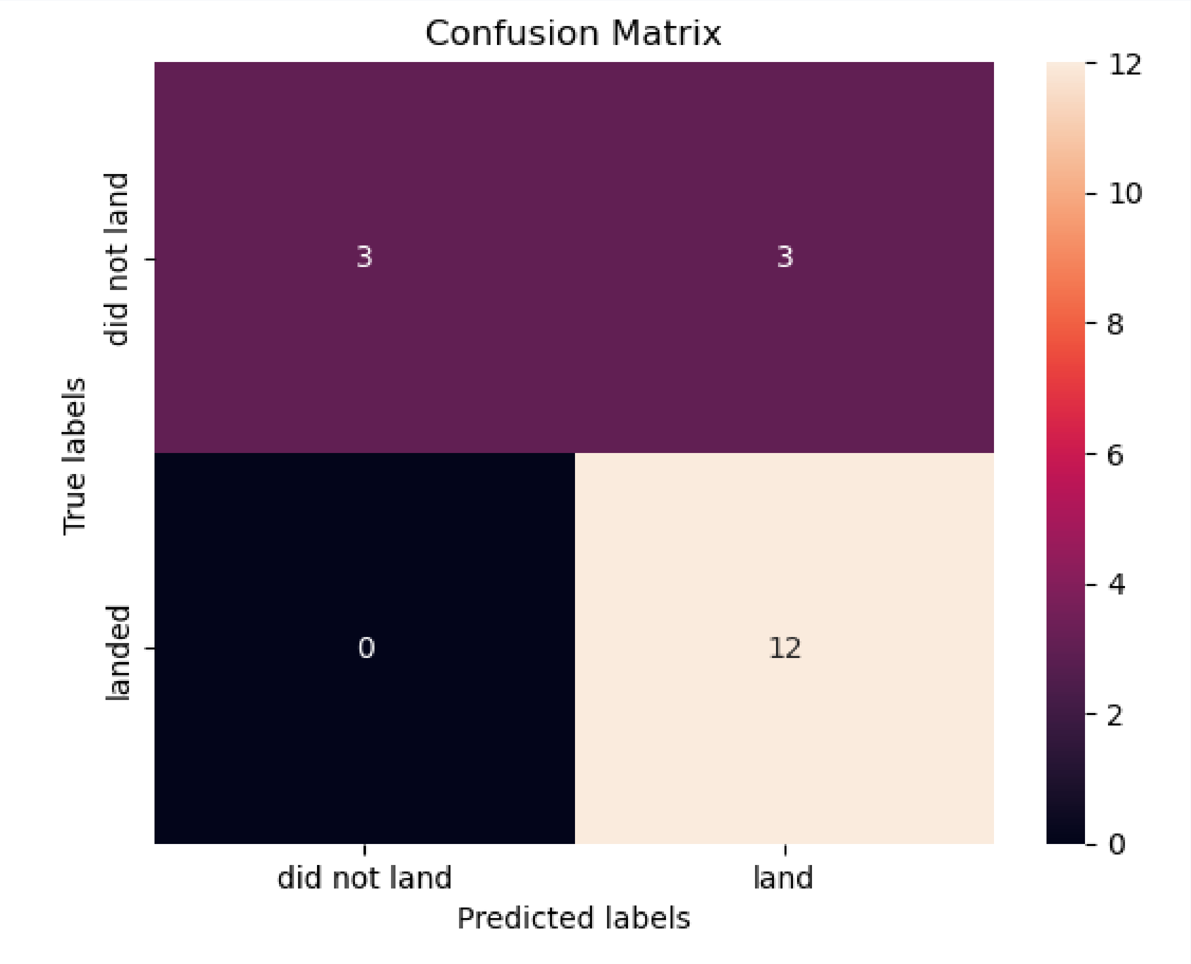
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestmethod = max(models, key=models.get)
print('Best method is', bestmethod,'with a score of', models[bestmethod])
```

```
Best method is DecisionTree with a score of 0.875
```

Confusion Matrix

- The decision tree classifier shows that the decision tree classifier can distinguish between the different classes.
- However, a problem is the false positive, unsuccessful landing marked as successful landing by the classifier.



Conclusions

- Launch success rate started to increase in 2013 till 2020.
- Orbits VLEO, GEO, ES-L1, HEO, SSO had the most success rate.
- KSC LC-39A was the most successful launch sites.
- The Decision tree classifier is the most accurate algorithm to predict the success of a launch.

Thank you!