Winning Space Race with Data Science

MEHMET AYDOĞMUŞ
11/8/2021

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- Summary of methodologies
  - Data collection
  - Data wrangling
  - EDA with SQL
  - EDA with Data Visualization
  - 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 in screenshots
  - Predictive analysis results

Introduction

- Project background and context

Our goal was to predict 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

- Factors that affect the rocket's landing.
- What conditions does SpaceX have to achieve in order to get the best results.

Section 1

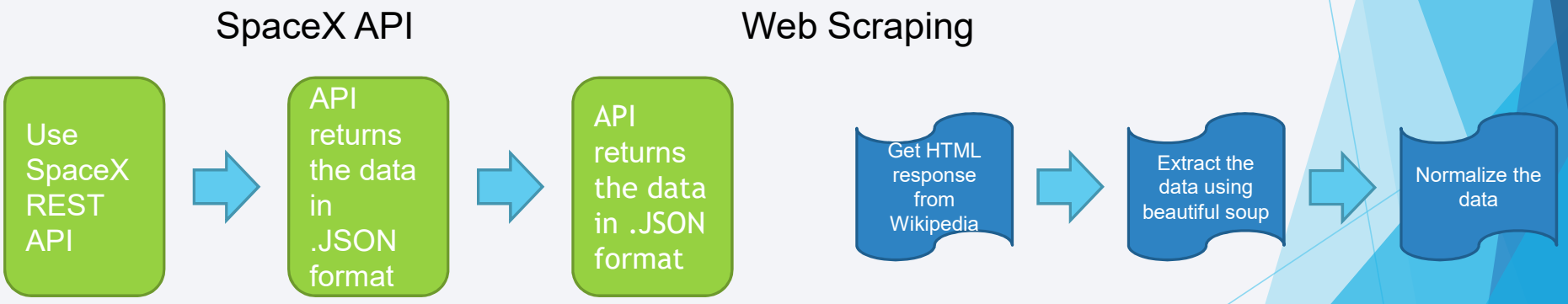
Methodology

Methodology

- Data collection methodology:
  - SpaceX Rest API
  - (Web Scraping) from Wikipedia
- Perform data wrangling (Transforming data for Machine Learning)
  - One Hot Encoding the data fields for machine learning
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Plotting: Graphs to show relationships between variables
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build and evaluate classification models

Data Collection

- Data sets were collected by
  - We worked with SpaceX launch data which is gathered from the SpaceX REST API.
  - This API will give us data about launches, rocket used, payload delivered, launch specs, landing specs and landing outcome.



Data Collection – SpaceX API

1. Getting Response from API

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

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

2. Decoding the response content as a .JSON file

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

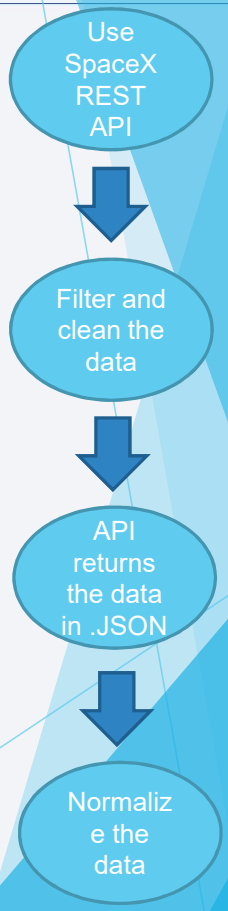
3. Assigning the list to dictionary

```
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}
```

4. Filtering the data and exporting the file (.csv)

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

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



GitHub URL to Notebook

Data Collection - Scraping

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

```
column_names = []
tmp = soup.find_all('th')
for x in range(len(tmp)):
    try:
     name = extract_column_from_header(tmp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

5. Creating the dictionary

```
launch_dict= dict.fromkeys(column_names)

del launch_dict['Date and time ( )']

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'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

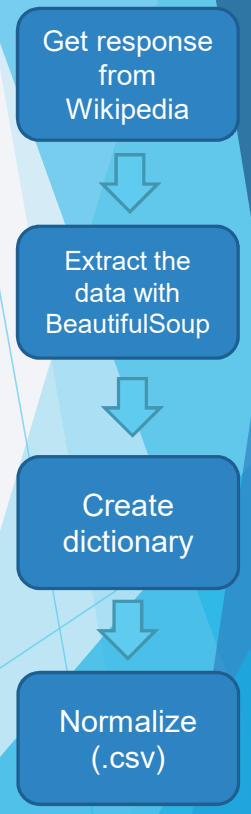
6.Converting dictionary

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

7. Converting dataframe

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

[GitHub URL to Notebook]



Data Wrangling

How the data were processed:

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 will mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

Process:

Perform EDA (Exploratory Data Analysis) on dataset

Calculate the number of launches on each site

Calculate the number and occurrence of mission outcome per orbit type

Calculate the number and occurrence of each orbit

Export dataset as .CSV

GitHub URL to Notebook

EDA with Data Visualization

Scatter Graphs:

Flight Number vs. Launch Site

Payload Mass vs. Launch Site

Flight Number vs. Orbit Type

Payload vs. Orbit Type

A scatter plot can be used either when one continuous variable is under the control of the experimenter and the other depends on it or when both continuous variables are independent.

GitHub URL to Notebook

Bar Graph:

Success Rate vs. Orbit Type

Bar graphs/charts provide a visual presentation of categorical data. Categorical data is a grouping of data into discrete groups, such as months of the year, age group, shoe sizes, and animals. These categories are usually qualitative. In a column (vertical) bar chart, categories appear along the horizontal axis and the height of the bar corresponds to the value of each category.

Line Graph:

Success Rate vs. Year

Line graphs are used to track changes over short and long periods of time. When smaller changes exist, line graphs are better to use than bar graphs. Line graphs can also be used to compare changes over the same period of time for more than one group.

EDA with SQL

- Performed SQL queries
  - Names of unique launch sites in the space mission
  - 5 records where launch sites being with the string ‘CCA’
  - Total payload mass carried by boosters launched by NASA (CRS)
  - Average payload mass carried by boosters version F9 v1.1
  - Listed the date when the first successful landing outcome in ground pad was achieved
  - Listed the names of the booster which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - Listed the total number of successful and failure mission outcomes
  - Listed the names of the booster_versions which have carried the maximum payload mass
  - Listed the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - Ranked the count of successful landing outcomes (ground pad) between the date 2010-06-04 and 2017-03-20 in descending order

GitHub URL to Notebook

Build an Interactive Map with Folium

- We took the latitude and longtitude of each launch site and added a circle around each site with a label of the name of the launch site
- The launch success rate may depend on many factors such as payload mass, orbit type, and so on. It may also depend on the location and proximities of a launch site, i.e., the initial position of rocket trajectories. Finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations.

- 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 to Notebook

Build a Dashboard with Plotly Dash

- Pie Chart:
  - Pie chart allows us to show the total launches by a certain site or all sites
  - showing percentages of successful and failure launches
- Scatter Graph:
  - Scatter graph shows the relationship between Payload Mass (Kg) and outcome such as success and failure
  - The range of data can be determined

GitHub URL to Notebook

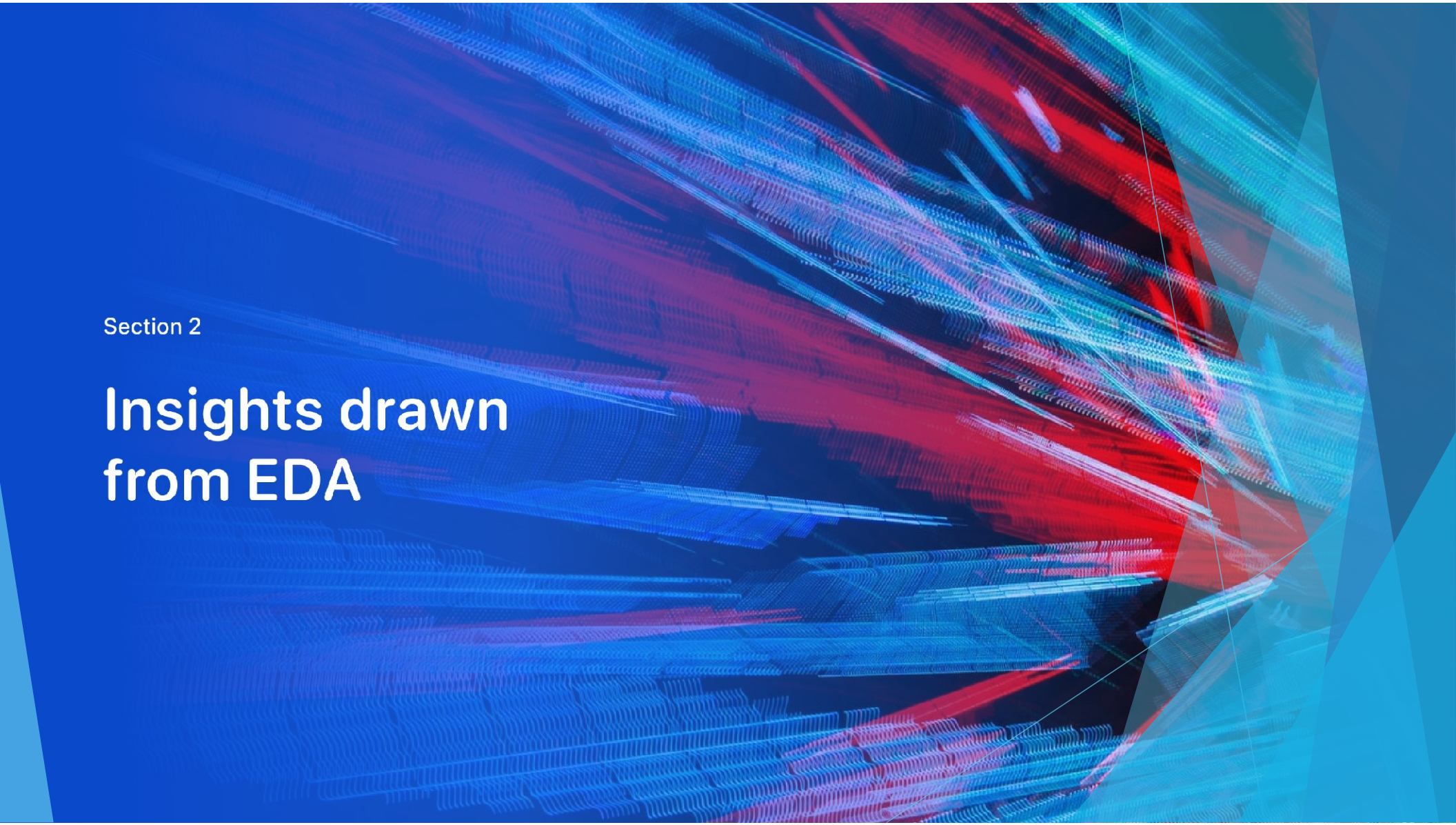
Predictive Analysis (Classification)

- **Building the Model**
  - Load the dataset
  - Transform the data
  - Split the data into training and test data sets
  - Check the number of test samples
  - Find out the best algorithm to use
  - Set parameters and algorithms to GridSearchCV
  - Fit the data into the GridSearchCV objects and train the data
- **Evaluating the Model**
  - Check accuracy for each model
  - Get the tuned hyperparameters for each type of algorithm
  - Plot the Confusion Matrix
- **Improving the Model**
  - Feature Engineering
- **Finding the method performs best using test date**
  - The best performing classification model wins

GitHub URL to Notebook

Results

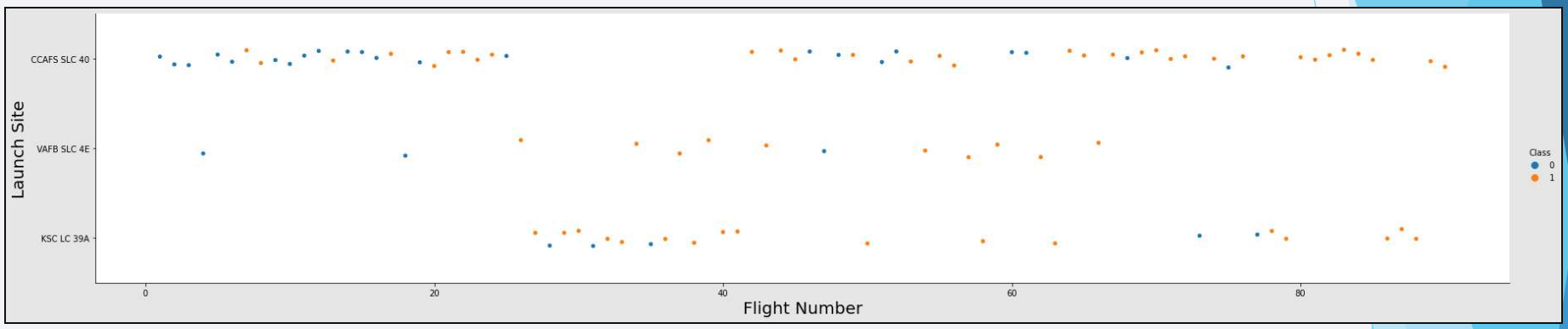
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results



Section 2

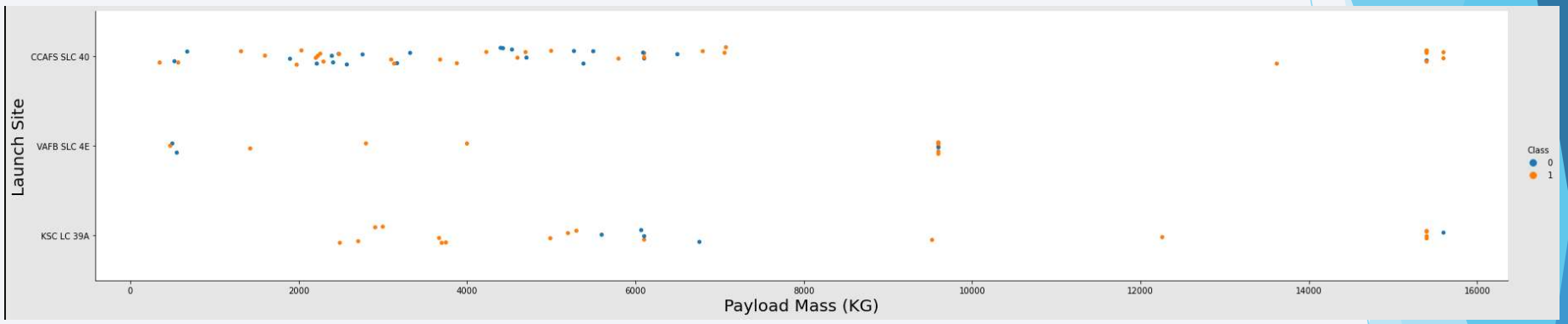
Insights drawn from EDA

Flight Number vs. Launch Site



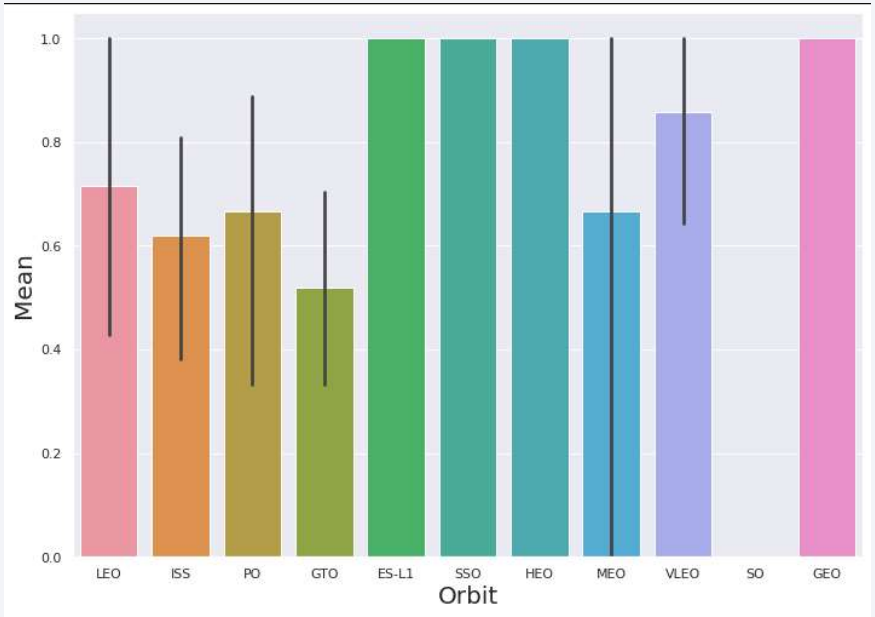
According to the chart above KSC LC-39A has the best success rate.

Payload vs. Launch Site



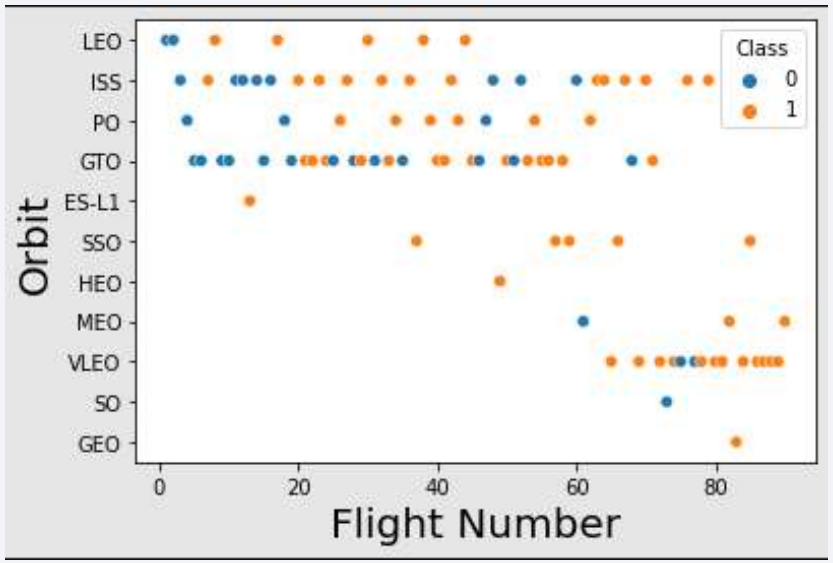
As seen on the Payload vs. Launch Site scatter point chart above for the VAFB-SLC launch site there are no rockets launched for heavy payload mass (greater than 10000).

Success Rate vs. Orbit Type



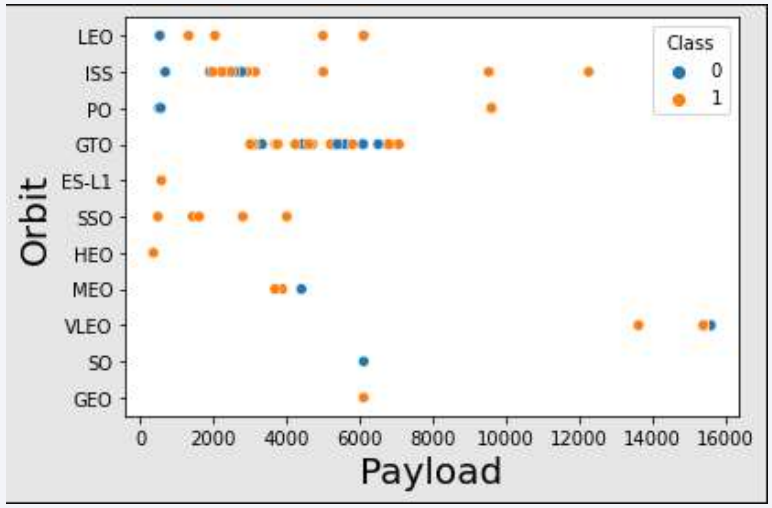
As seen above the ES-L1, SSO, HEO and GEO orbitals have the best success rate

Flight Number vs. Orbit Type



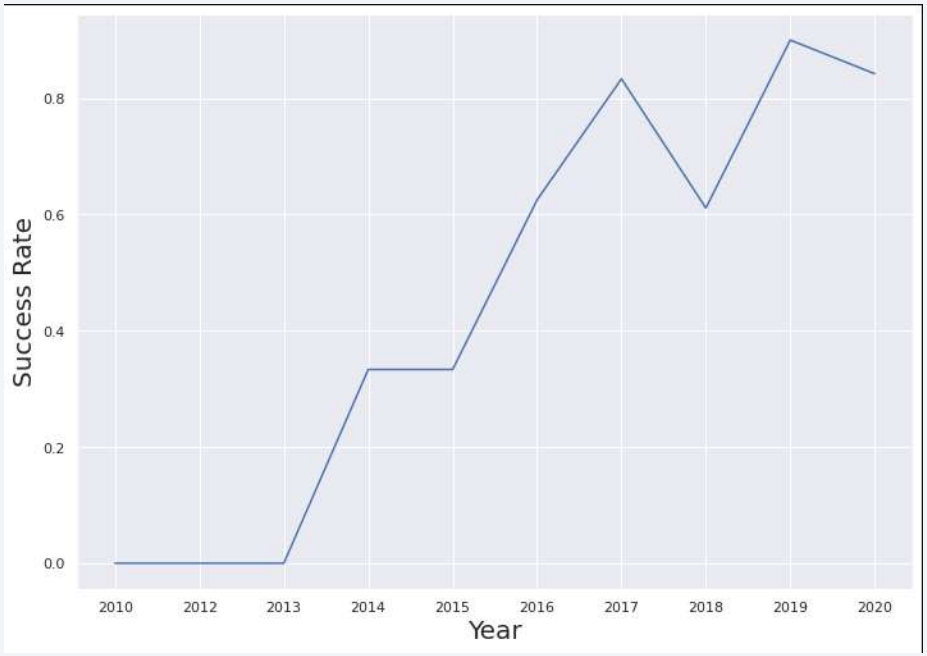
As seen above in the LEO orbit, the success is related to the number of flights; on the other hand, there seems to be no relationship between flight number in GTO orbit.

Payload vs. Orbit Type



- With heavy payloads the successful landing rates are better for PO, LEO and ISS.
  - However for GTO, success and failure rates are close to each other.

Launch Success Yearly Trend



As seen above the success rate kept increasing since 2013 till 2020 except for the year 2018.

All Launch Site Names

%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXDATASET

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

Query Explanation:

DISTINCT means that it will only show unique values in the launch_site column from SPACEXDATASET

Launch Site Names Begin with 'CCA'

```
%sql SELECT * FROM
SPACEXDATASET WHERE
LAUNCH_SITE LIKE '%CCA%' LIMIT 5
```

Query Explanation:

Using the word **LIMIT** 5 in the query means that it will only show 5 records from SPACEXDATASET and **LIKE** keyword has a wildcard with the word '%CCA%' and the percentage suggests that the launch_site name must include CCA word.

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	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)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
%sql SELECT sum(PAYLOAD_MASS__KG_) AS "Total Payload Mass NASA (CRS)" FROM SPACEXDATASET
WHERE CUSTOMER = 'NASA (CRS)'
```

Total Payload Mass NASA (CRS)
45596

Query Explanation:

Using the function **SUM** summates the total in the column PAYLOAD_MASS__KG_

The **WHERE** clause filters the data to only perform calculations on Customer NASA (CRS) table.

Average Payload Mass by F9 v1.1

%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass F9 v1.1" FROM SPACEXDATASET WHERE BOOSTER_VERSION = 'F9 v1.1'

Average Payload Mass F9 v1.1
2928

Query Explanation:

AVG function finds the average in the column PAYLOAD_MASS__KG_

The **WHERE** clause filters the data to only perform calculations on BOOSTER_VERSION 'F9 v1.1' table.

First Successful Ground Landing Date

%sql SELECT MIN(DATE) AS "First Successful Landing" FROM SPACEXDATASET
WHERE LANDING__OUTCOME = 'Success (ground pad)'

First Successful Landing
2015-12-22

Query Explanation:

MIN function finds the minimum date in the DATE column

The **WHERE** clause filters the data to only perform calculations on LANDING__OUTCOME 'Success (ground pad)' table.

Successful Drone Ship Landing with Payload between 4000 and 6000

%sql SELECT BOOSTER_VERSION AS "Successful Boosters" FROM SPACEXDATASET WHERE LANDING__OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000

Successful Boosters
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Query Explanation:

The **AND** clause filters the data on Payload Mass (Kg) column between 4000 and 6000.

The **WHERE** clause filters the data to only perform calculations on LANDING__OUTCOME 'Success (ground pad)‘ table.

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Success and Failure Count" FROM SPACEXDATASET
WHERE MISSION_OUTCOME = 'Success' OR MISSION_OUTCOME = 'Failure (in flight)'
```

Success and Failure Count
100

Query Explanation:

We used subqueries here to produce the result.

The **WHERE** clause filters the data to only perform calculations on MISSION_OUTCOME 'Success' OR MISSION_OUTCOME 'Failure (in flight)' table.

Boosters Carried Maximum Payload

%sql SELECT BOOSTER_VERSION AS "Maximum Payload Mass" FROM SPACEXDATASET WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXDATASET)

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

Query Explanation:

We used a subquery again in order to find the maximum payload carried

The **WHERE** clause filters the data to only perform calculations on PAYLOAD_MASS__KG_ MAX(PAYLOAD_MASS__KG_ table.

2015 Launch Records

%sql SELECT DATE, LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXDATASET WHERE LANDING__OUTCOME LIKE '%Failure (drone ship)%' AND YEAR(DATE) = 2015

DATE	landing__outcome	booster_version	launch_site
2015-01-10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Query Explanation:

We used YEAR(DATE) = 2015 in order to filter the DATE column to show data frames of 2015.

The **WHERE** clause filters the data to only perform calculations on LANDING__OUTCOME and we used LIKE '%Failure (drone ship)%' AND YEAR(DATE) = 2015 to find a specific landing outcome, which is ‘failure (drone ship)’.

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

%sql SELECT * FROM SPACEXDATASET WHERE LANDING__OUTCOME LIKE '%Success (ground pad)%' AND (DATE BETWEEN '2010-06-04' AND '2017-03-20') ORDER BY DATE DESC

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2016-07-18	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2015-12-22	01:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

Query Explanation:

The **ORDER BY** keyword is used to sort the result-set in ascending or descending order.

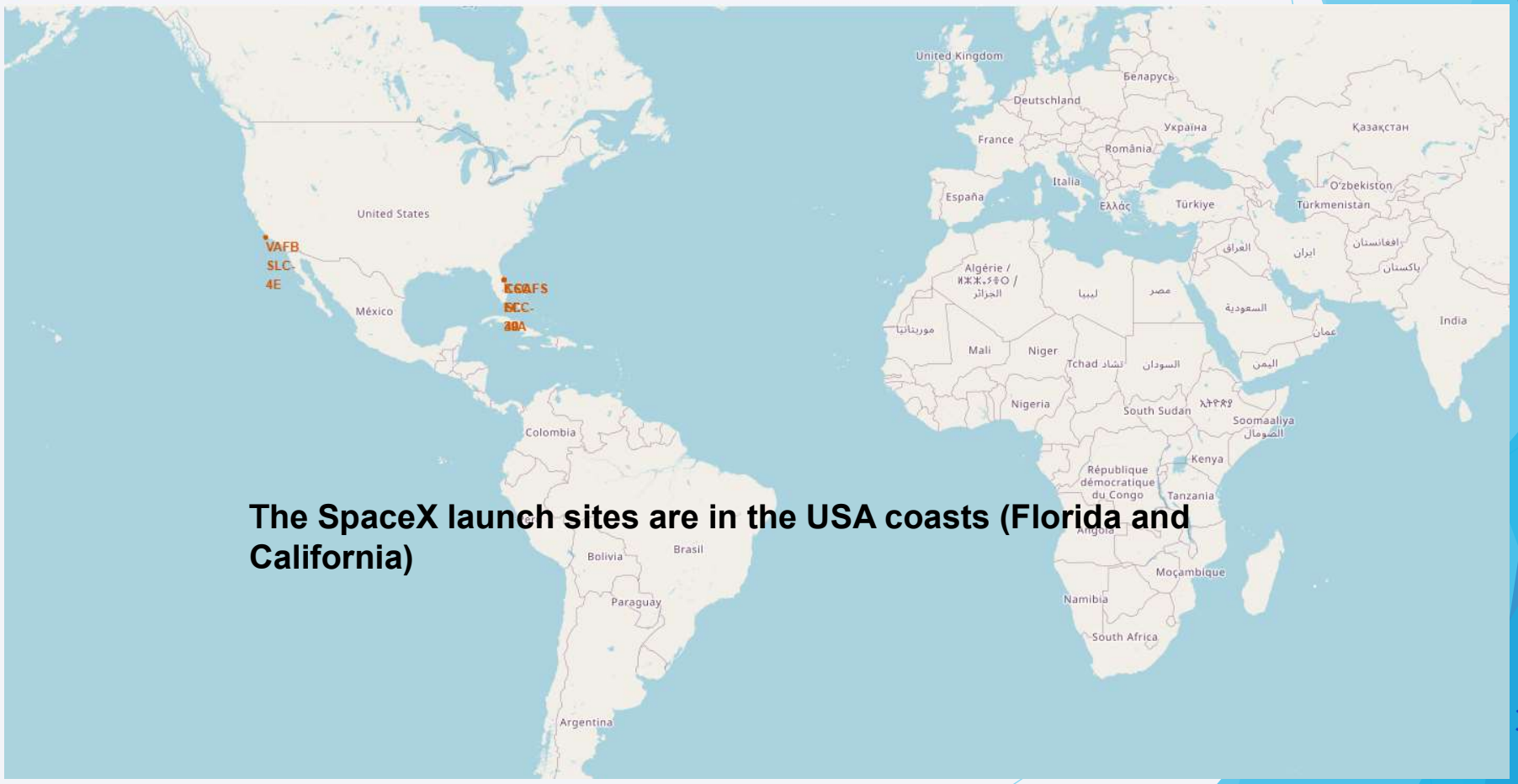
The **BETWEEN** operator selects values within a given range which is '2010-06-04' AND '2017-03-20'. The values can be numbers, text, or dates.

The **WHERE** clause filters the data to only perform calculations on LANDING__OUTCOME LIKE '%Success (ground pad)%' AND (DATE BETWEEN '2010-06-04' AND '2017-03-20') to find a specific landing outcome, which is 'Success (ground pad)'.

Section 4

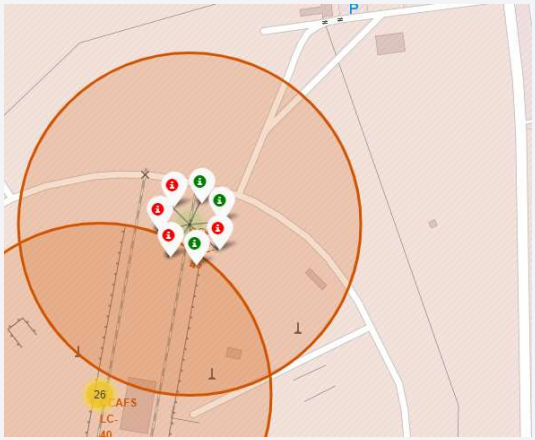
Launch Sites Proximities Analysis

All Launch Sites on Map

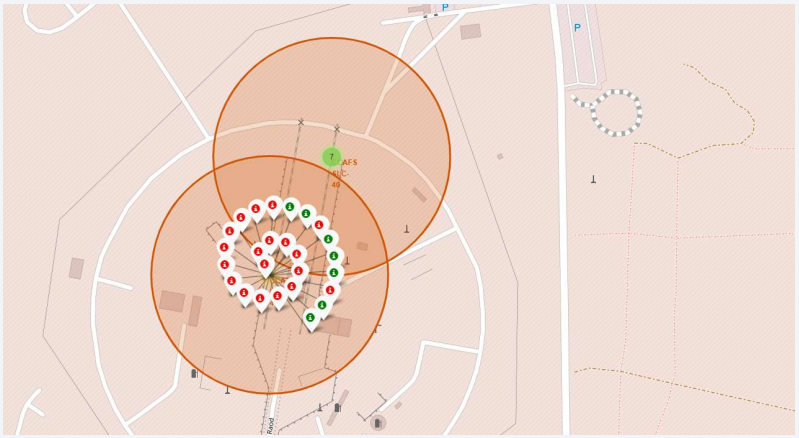


The SpaceX launch sites are in the USA coasts (Florida and California)

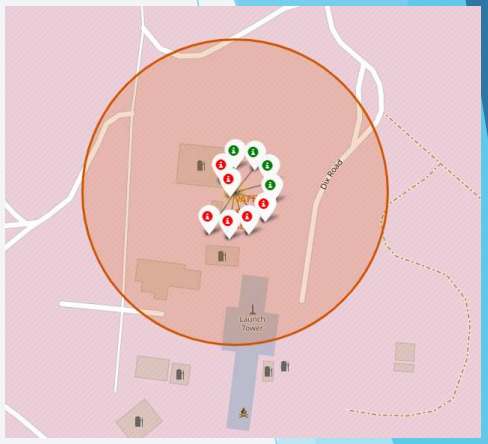
Circles and Markers



CCAFS SLC-40 (Florida Site)



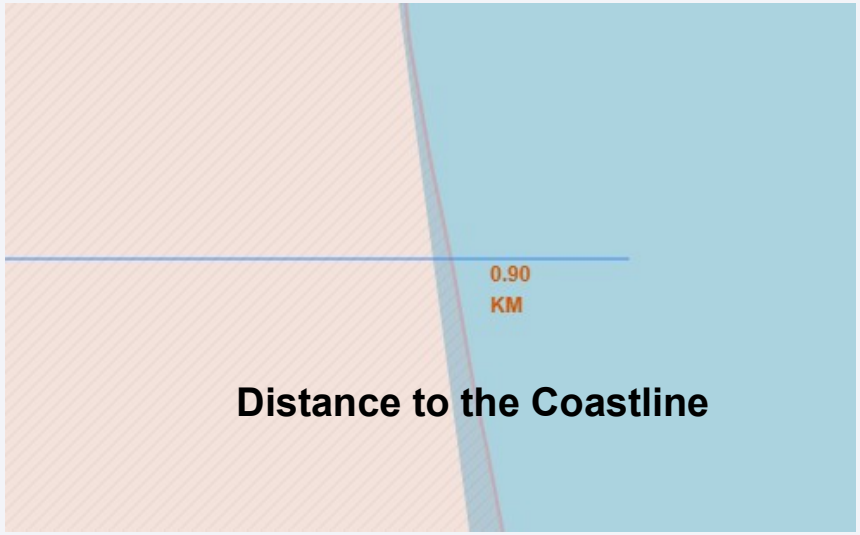
CCAFS LC-40 (Florida Site)



VAFB SLC-4E (California Site)

Green marker shows the successful launches and Red Marker shows the failures

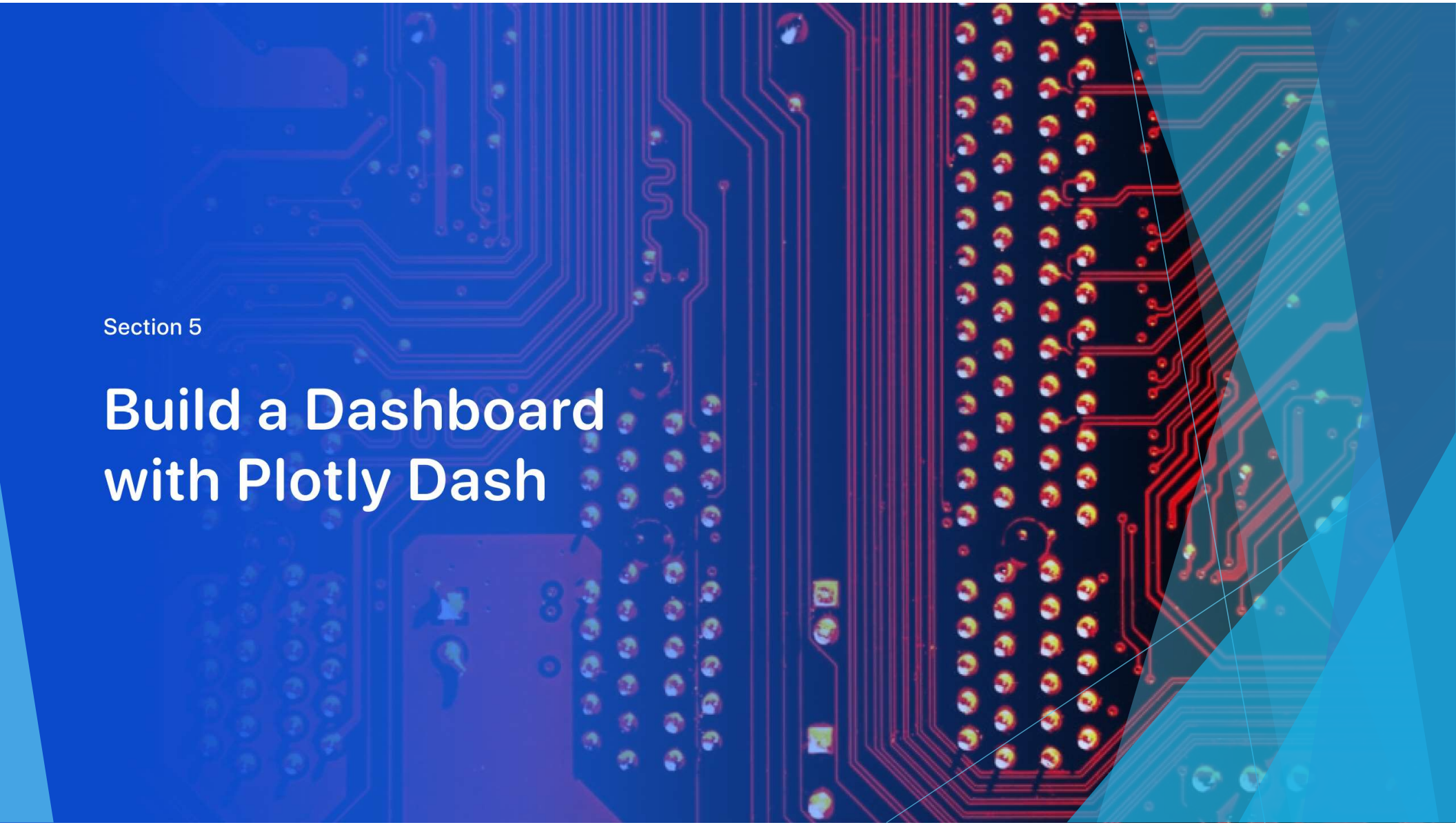
Launch Site distances to landmarks using CCAFS-SLC-40 as a Reference



Distance to the Coastline

Distance to closest highway

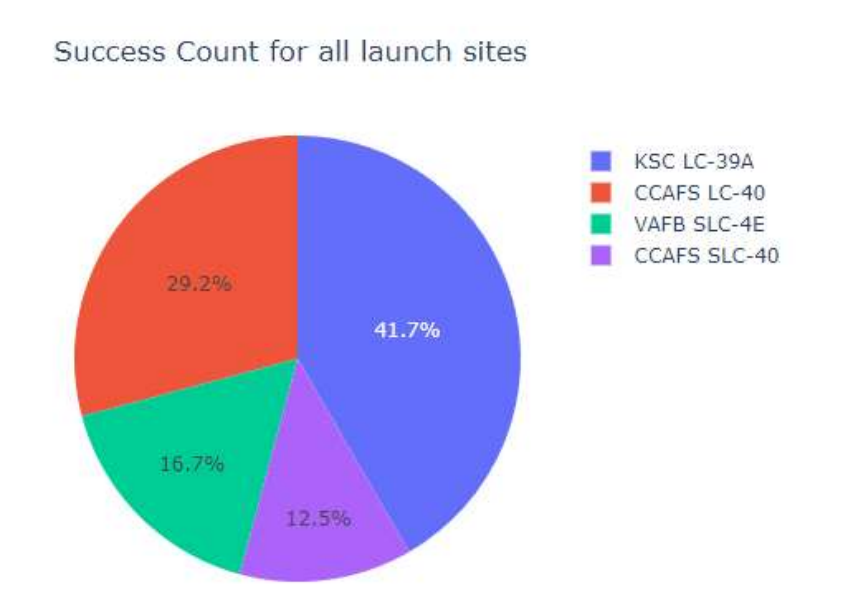
Are launch sites in close proximity to railways? *No*
Are launch sites in close proximity to highways? *No*
Are launch sites in close proximity to coastline? *Yes*
Do launch sites keep certain distance away from cities? *Yes*



Section 5

Build a Dashboard with Plotly Dash

Pie chart for the success percentage by each launch site

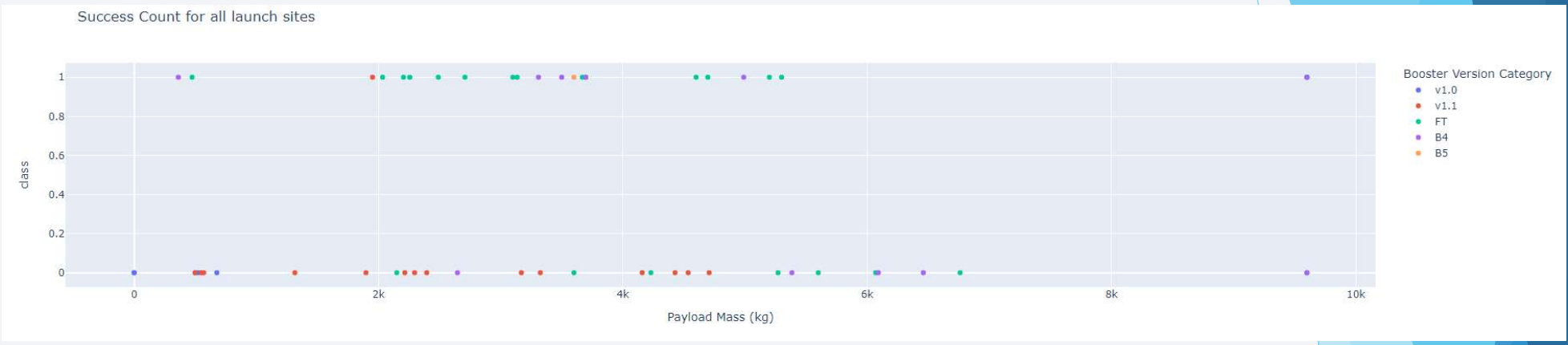


As seen on the above chart KSC LC-39A has the most successful launches

Pie chart for the launch site with highest success ratio

As seen on the above chart KSC LC-39A has a 76.9% success rate

Payload vs. Launch outcome scatter plot



As seen on the above chart success rates between 2000 and 4000 kg payloads are higher than others.

Section 6

Predictive Analysis (Classification)

Classification Accuracy

The tree algorithm is the most successfull algorithm among others with a score of 0.8625 (86.25% success rate).

```
algorithms = {'LogisticRegression': logreg_cv.best_score_, 'Tree': tree_cv.best_score_, 'KNN': knn_cv.best_score_}
best = max(algorithms, key=algorithms.get)
print(f"Logistic Regression: .{logreg_cv.best_score_}\nTree:{tree_cv.best_score_}\nKNN:{knn_cv.best_score_}")
print(f"the Best Algorithm is: {best} with {algorithms[best]} score")
```

```
Logistic Regression: .0.8464285714285713
Tree:0.8625
KNN:0.8482142857142858
the Best Algorithm is: Tree with 0.8625 score
```

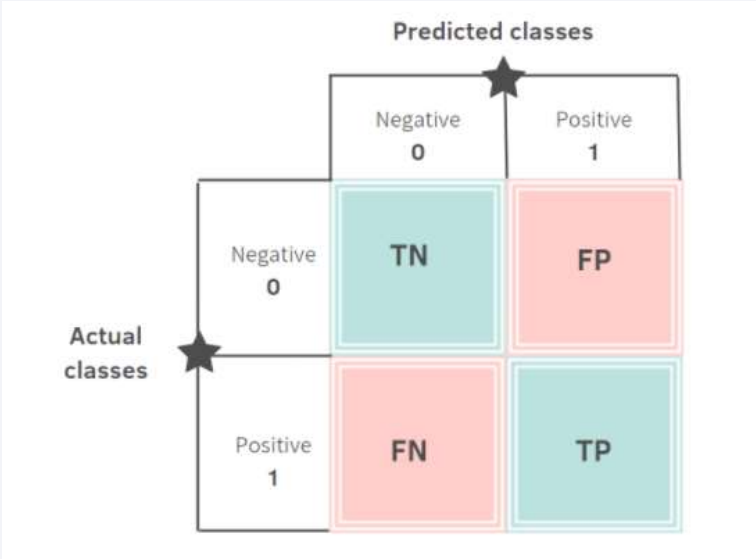
Tree algorithm has a 0.875 (87.5%) accuracy

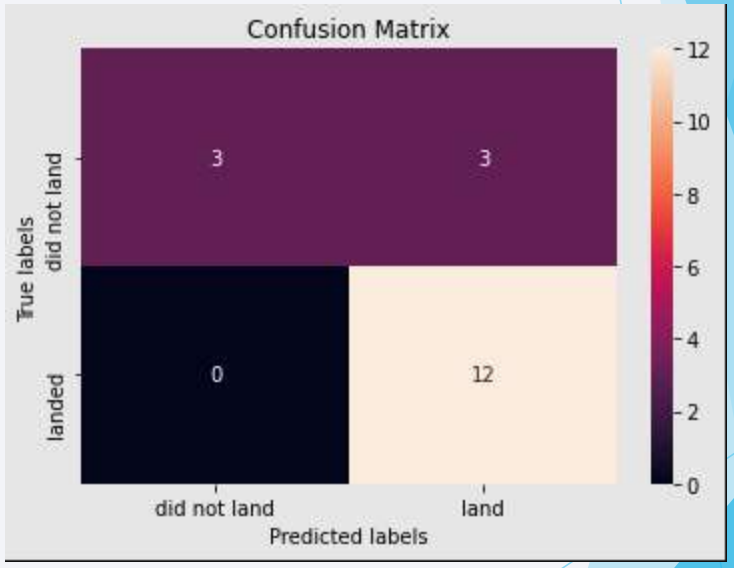
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 4, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 10, 'splitter': 'random'}
accuracy : 0.875
```

Confusion Matrix

As seen on the chart below ***tree algorithm*** has no false negative which is pretty good





Conclusions

- The ***Tree Algorithm*** is the best for this ***Machine Learning*** data set
- ES-L1, SSO, GEO, HEO orbits have the best Success Rate
- Low weighted payloads between 2000 and 4000 kg payloads perform better
- KSC LC-39A has the most successful launches

Thank you!