IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Akshay Babasaheb More
28/10/2021

Introduction

- **SpaceX Falcon 9 first stage Landing Prediction**

- **Request to the SpaceX API**
- **Clean the requested data**

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Qualitative Data Collection-Online Forum, Group Chat ,Web Survey Chat, Online Communication
  - Quantitative Data Collection- Face to Face, Phone ,Mail.
- Perform data wrangling
  - Data Acquisition: Identify and obtain access to the data within your sources.
  - Joining Data: Combine the edited data for further use and analysis.
  - Data Cleansing: Redesign the data into a usable and functional format and correct/remove any bad data.
- Performed exploratory data analysis (EDA) using visualization and SQL

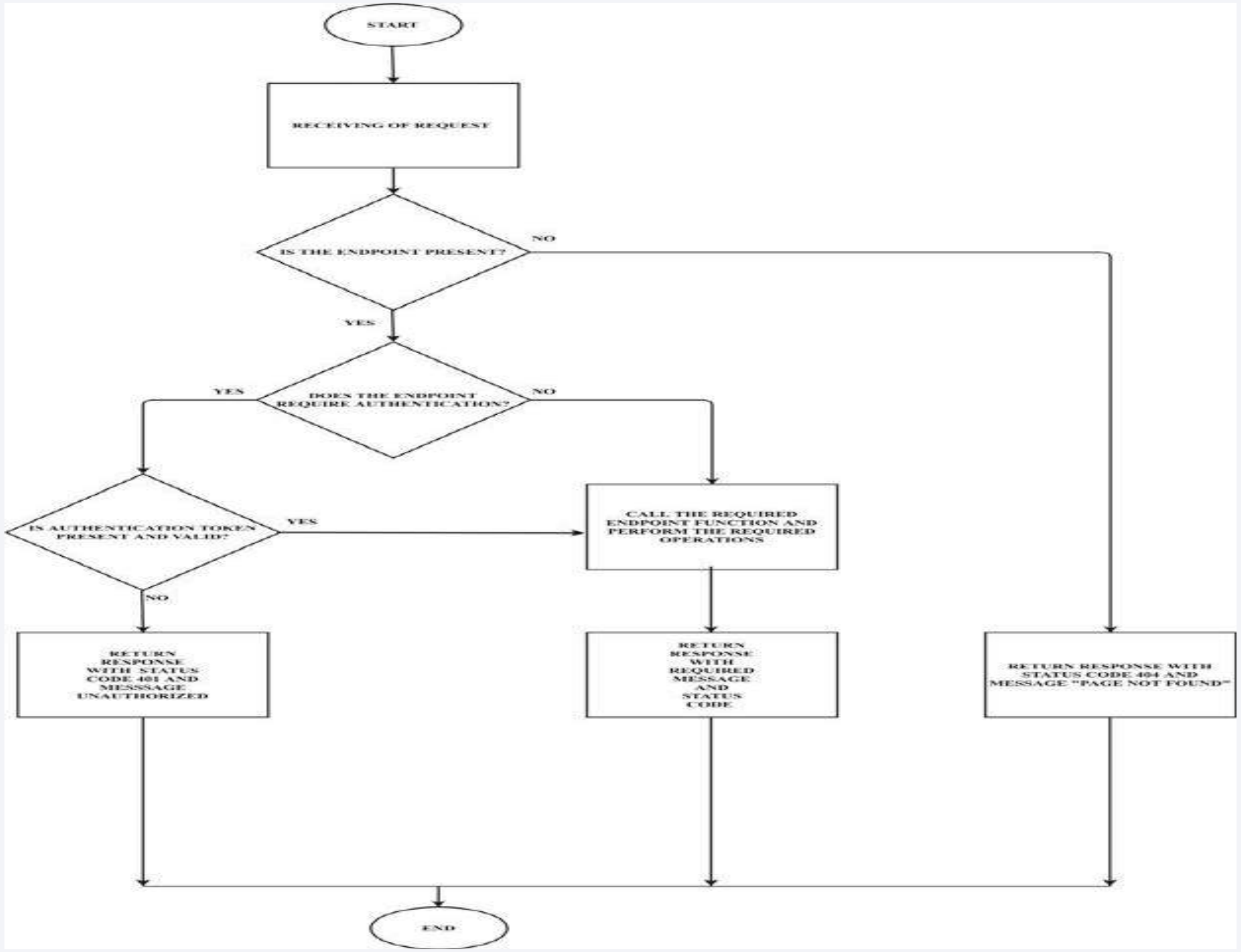
Data Collection

```
def getBoosterVersion(data): for x in data['rocket’]:

  response = requests.get("https://api.spacexdata.com/v4/rockets/"+(x)).json()

  BoosterVersion.append(response['name'])
```

```
def getPayloadData(data):
  for load in data['payloads’]:
    response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
    PayloadMass.append(response['mass_kg’])
    Orbit.append(response['orbit'])
```

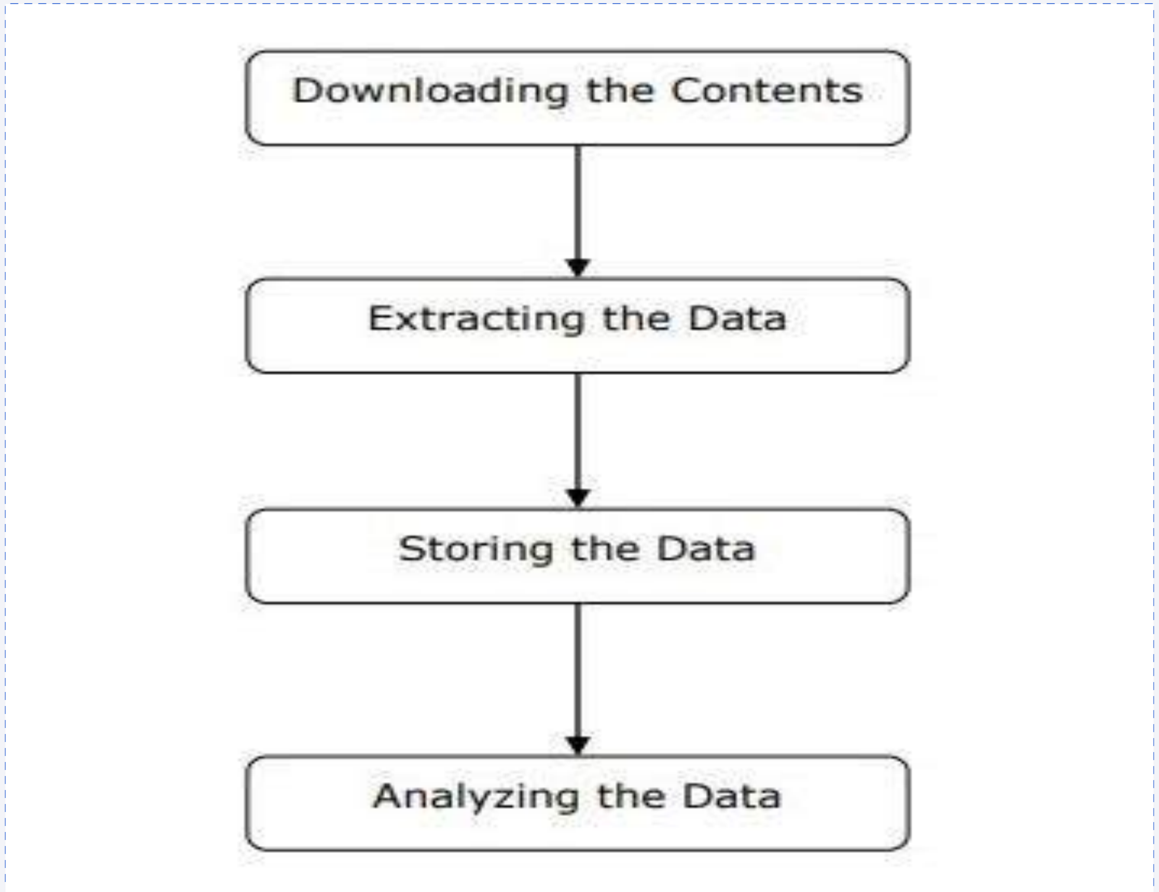
Data Collection – SpaceX API

- SpaceX REST calls using key phrases and flowcharts
- https://github.com/Akshaymore55/Applied-Data-Science/blob/master/Jupyter-labs-SpaceX-data-collection-API.ipynb


START
RECEIVING OF REQUEST
IS THE ENDPOINT PRESENT?
NO
YES
DOES THE ENDPOINT REQUIRE AUTHENTICATION?
YES
NO
IS AUTHENTICATION TOKEN PRESENT AND VALID?
YES
NO
CALL THE REQUIRED ENDPOINT FUNCTION AND PERFORM THE REQUIRED OPERATIONS
RETURN RESPONSE WITH STATUS CODE 401 AND MESSSAGE UNAUTHORIZED
RETURN RESPONSE WITH REQUIRED MESSAGE AND STATUS CODE
RETURN RESPONSE WITH STATUS CODE 404 AND MESSAGE "PAGE NOT FOUND"
END

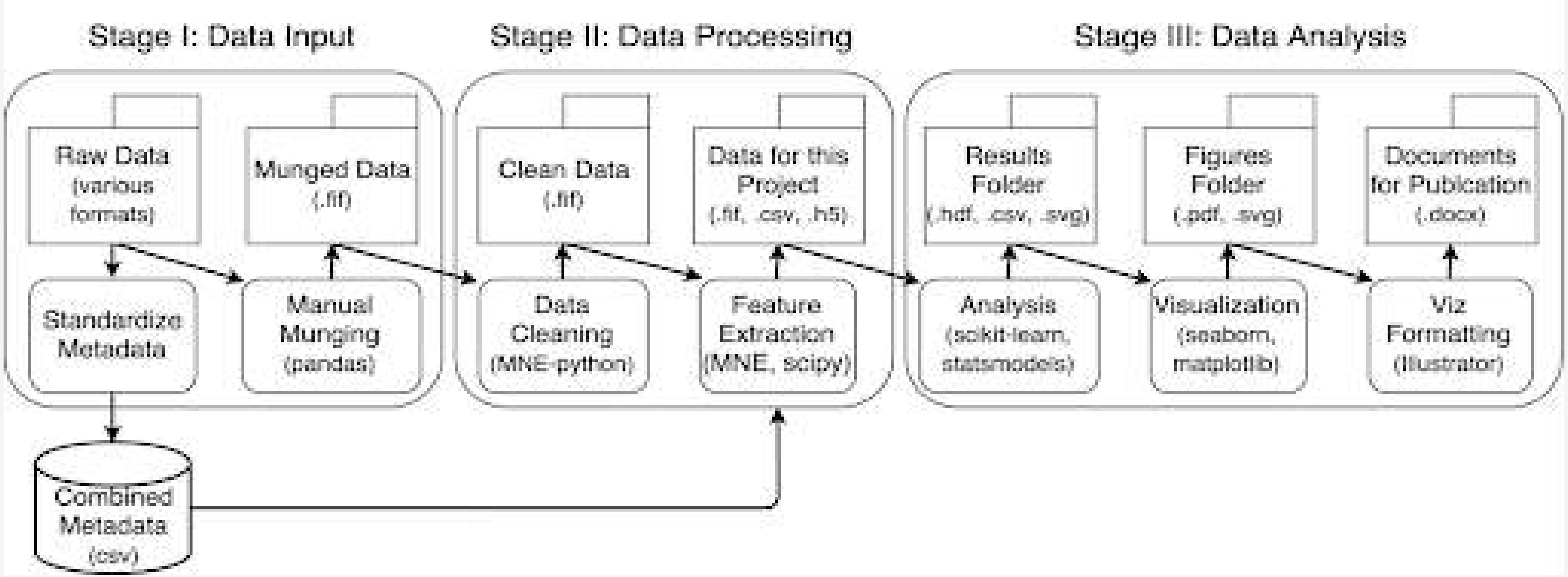
Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts



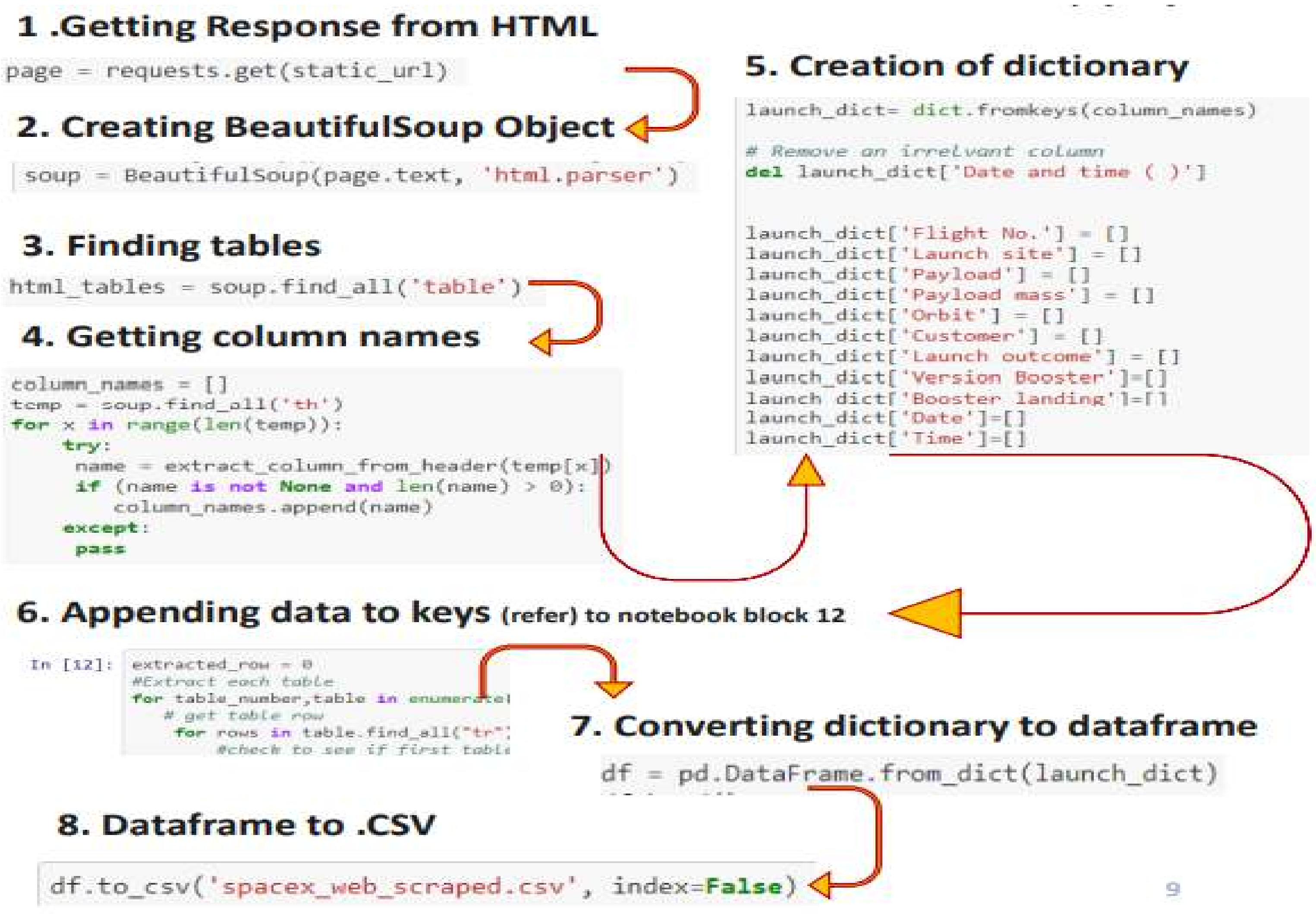
Data Wrangling

- **Data wrangling is the process of cleaning and unifying messy and complex data sets for easy access and analysis.**



Stage I: Data Input
Stage II: Data Processing
Stage III: Data Analysis
Raw Data (various formats)
Munged Data (.fif)
Clean Data (.fif)
Data for this Project (.fif, .csv, .h5)
Results Folder (.hdf, .csv, .svg)
Figures Folder (.pdf, .svg)
Documents for Publication (.docx)
Standardize Metadata
Manual Munging (pandas)
Data Cleaning (MNE-python)
Feature Extraction (MNE, scipy)
Analysis (scikit-learn, statsmodels)
Visualization (seaborn, matplotlib)
Viz Formatting (Illustrator)
Combined Metadata (csv)

Data Collection with API:-



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 = []
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
```

5. Creation of dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
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. Appending data to keys (refer) to notebook block 12

```
In [12]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(
             # get table row
              for rows in table.find_all("tr"):
                  #check to see if first table
```

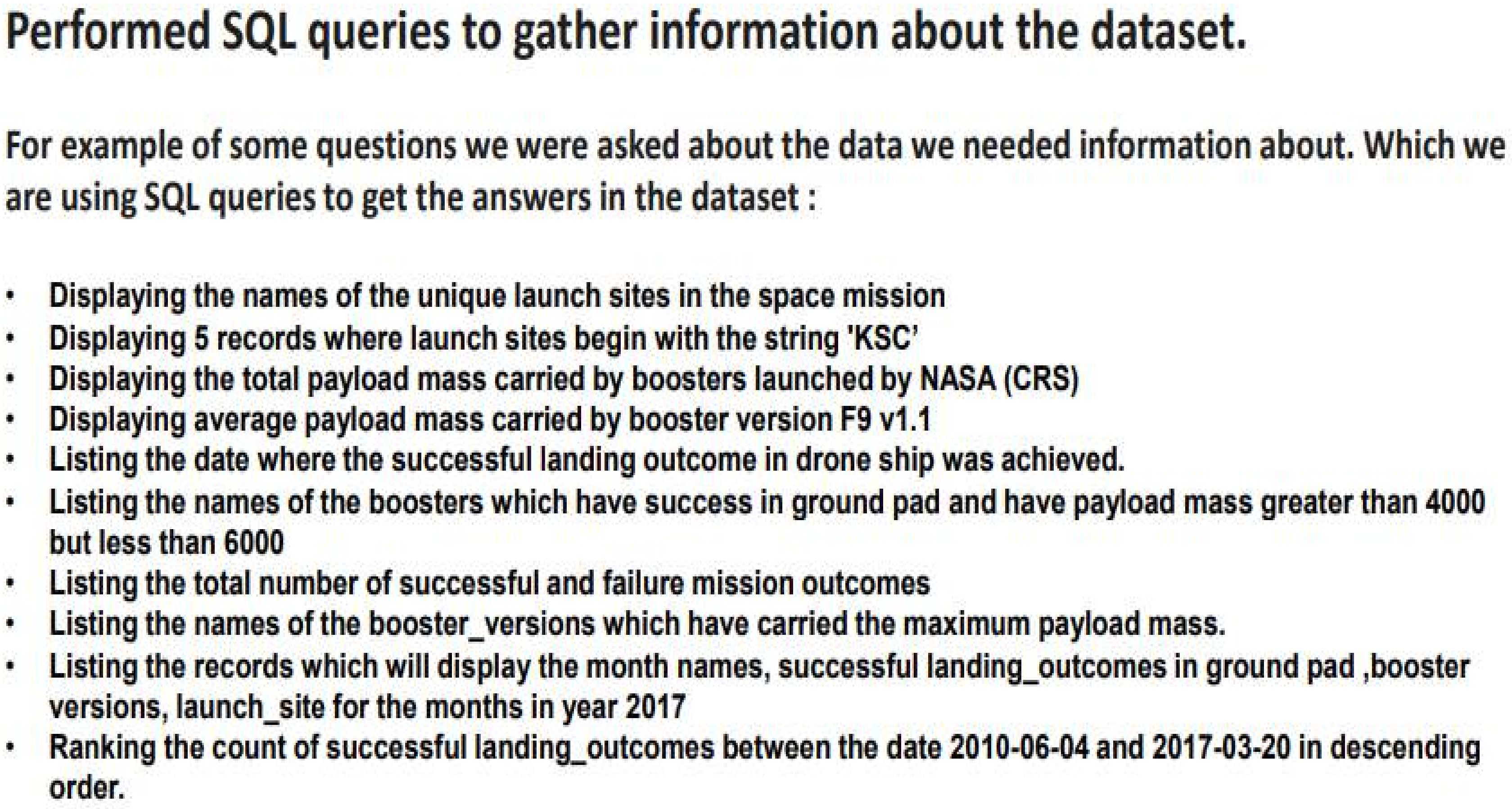
7. Converting dictionary to dataframe

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

8. Dataframe to .CSV

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

EDA SQL Describe:-

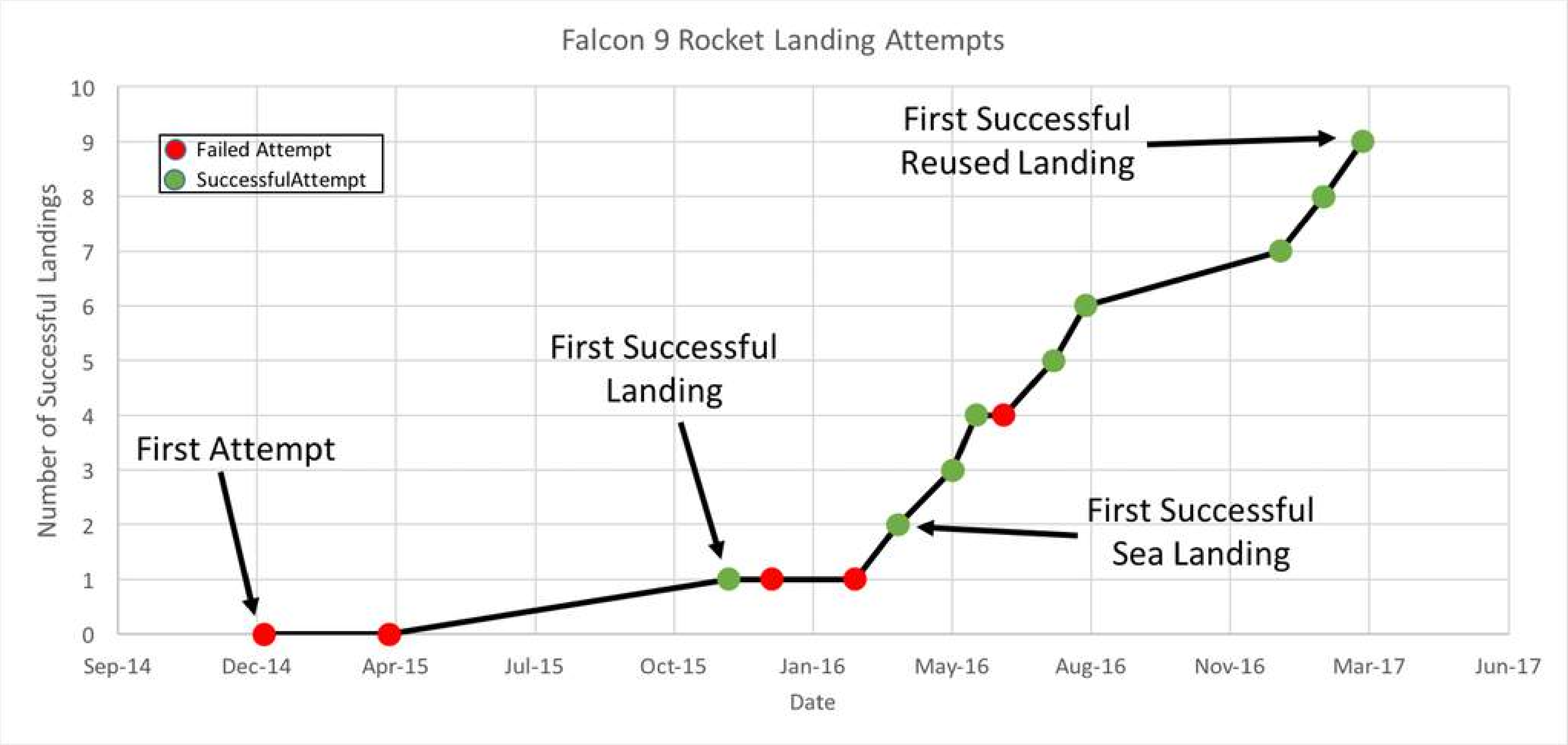


Performed SQL queries to gather information about the dataset.

For example of some questions we were asked about the data we needed information about. Which we are using SQL queries to get the answers in the dataset :

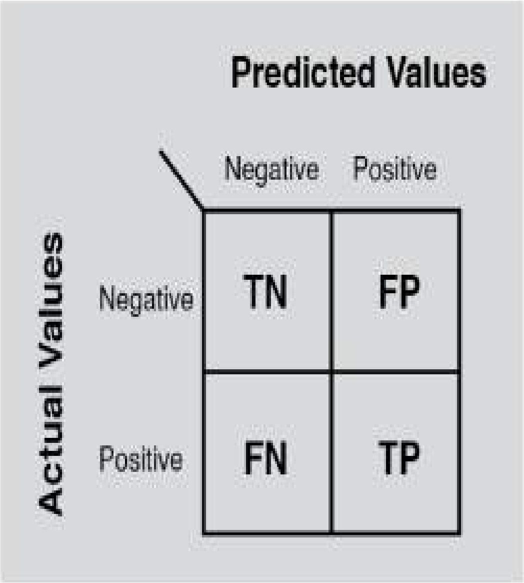
- **Displaying the names of the unique launch sites in the space mission**
- **Displaying 5 records where launch sites begin with the string 'KSC'**
- **Displaying the total payload mass carried by boosters launched by NASA (CRS)**
- **Displaying average payload mass carried by booster version F9 v1.1**
- **Listing the date where the successful landing outcome in drone ship was achieved.**
- **Listing the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000**
- **Listing the total number of successful and failure mission outcomes**
- **Listing the names of the booster_versions which have carried the maximum payload mass.**
- **Listing the records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017**
- **Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.**

EDA with Data Visualization



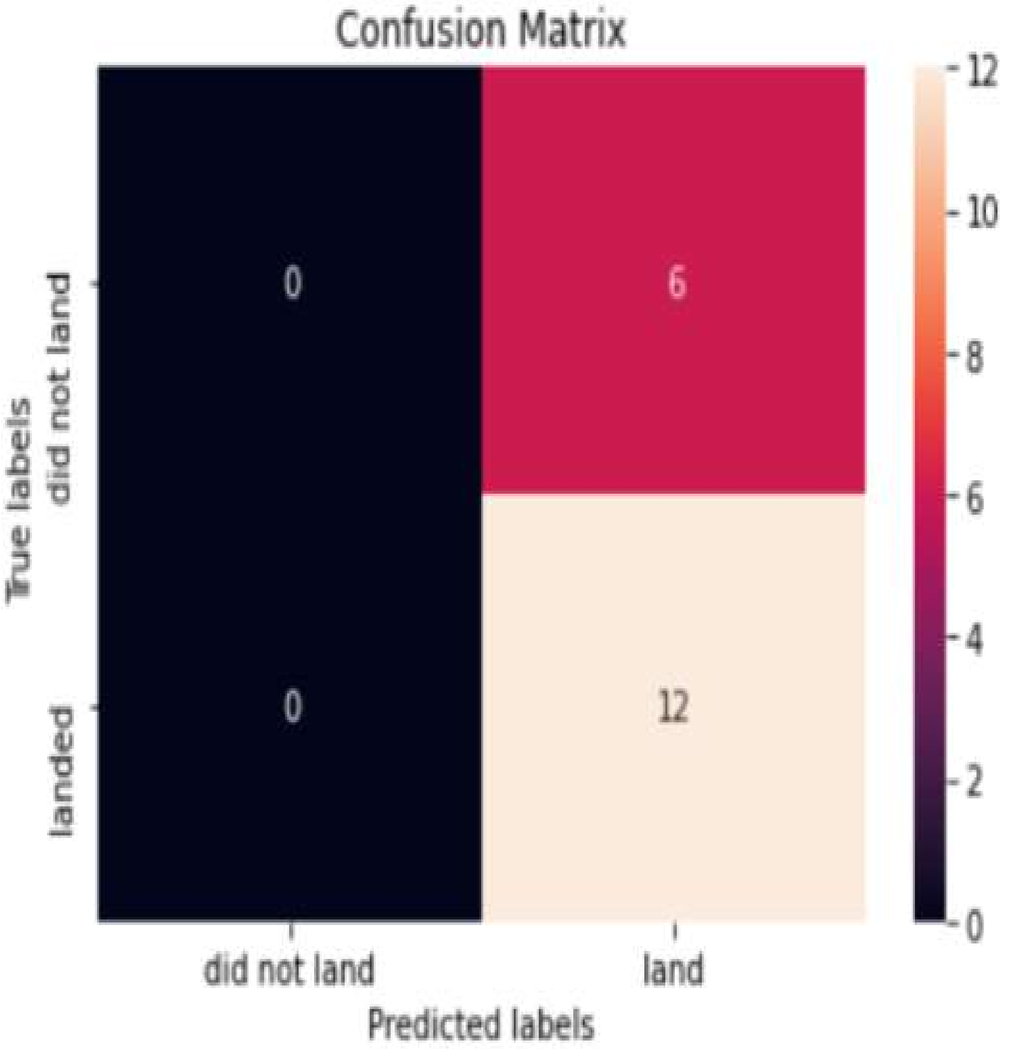
Confusion Matrix
for the Tree

Examining the confusion matrix, we see that Tree can distinguish between the different classes. We see that the major problem is false positives.

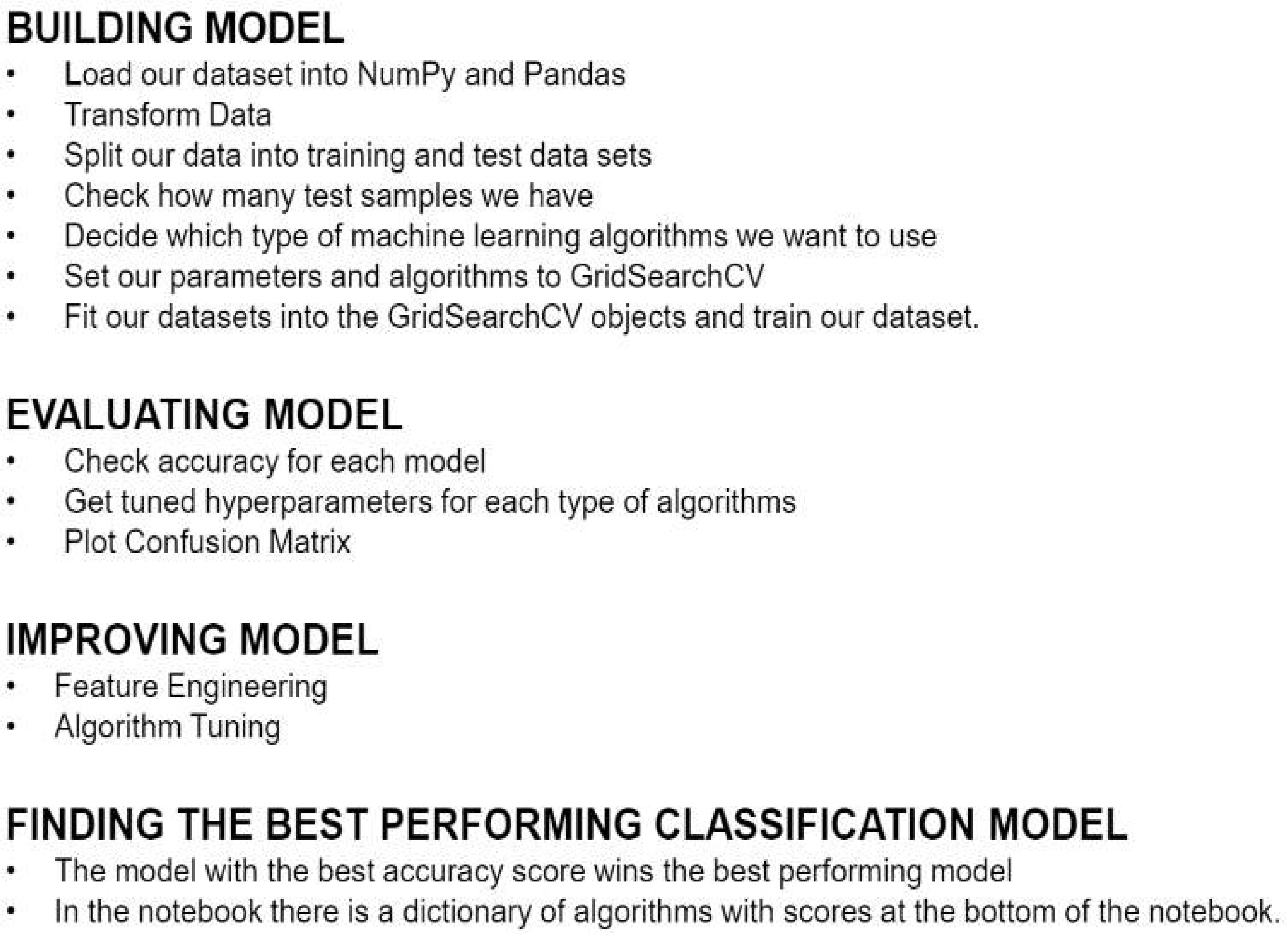


Predicted Values

Actual Values	Negative	Positive
Negative	TN	FP
Positive	FN	TP



Predictive Analysis (Classification)



BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

EVALUATING MODEL

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

IMPROVING MODEL

- Feature Engineering
- Algorithm Tuning

FINDING THE BEST PERFORMING CLASSIFICATION MODEL

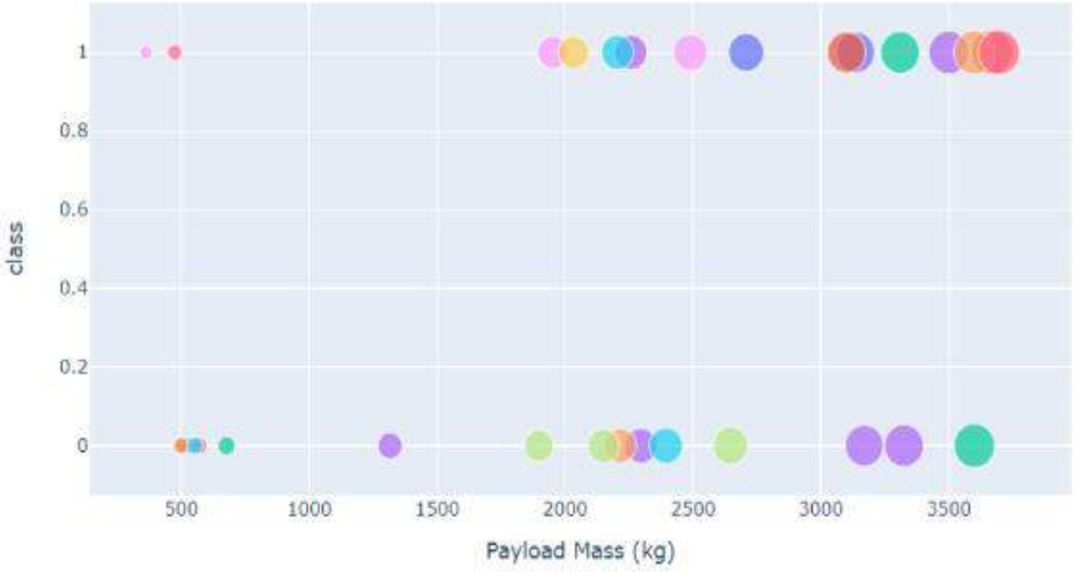
- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

Section 2

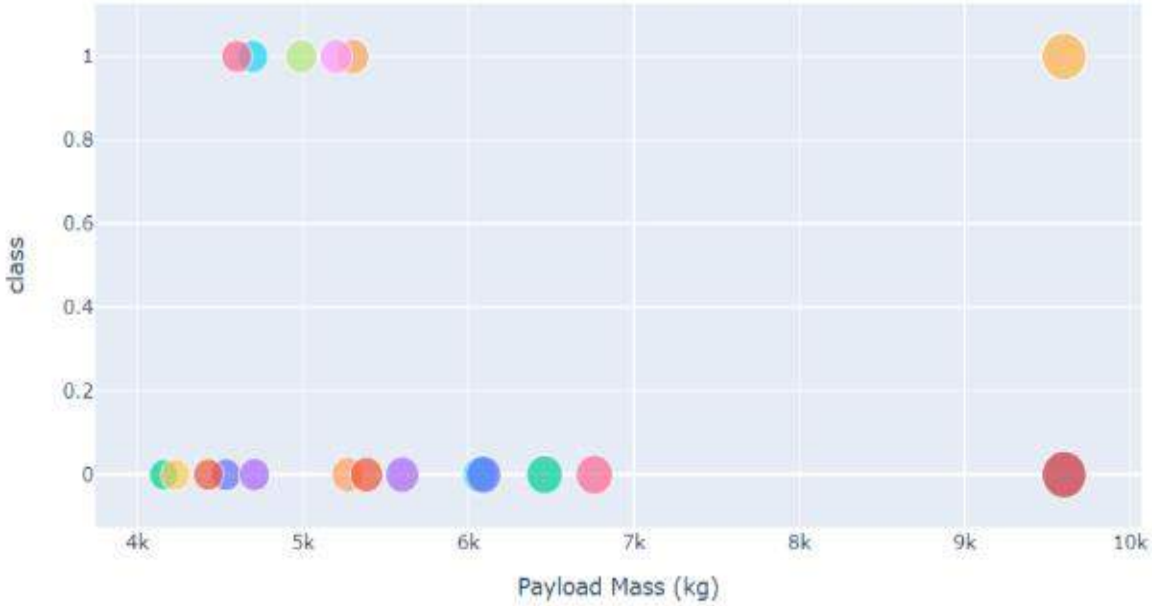
Insights drawn from EDA

Low weighted payload vs Heavy payload

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg

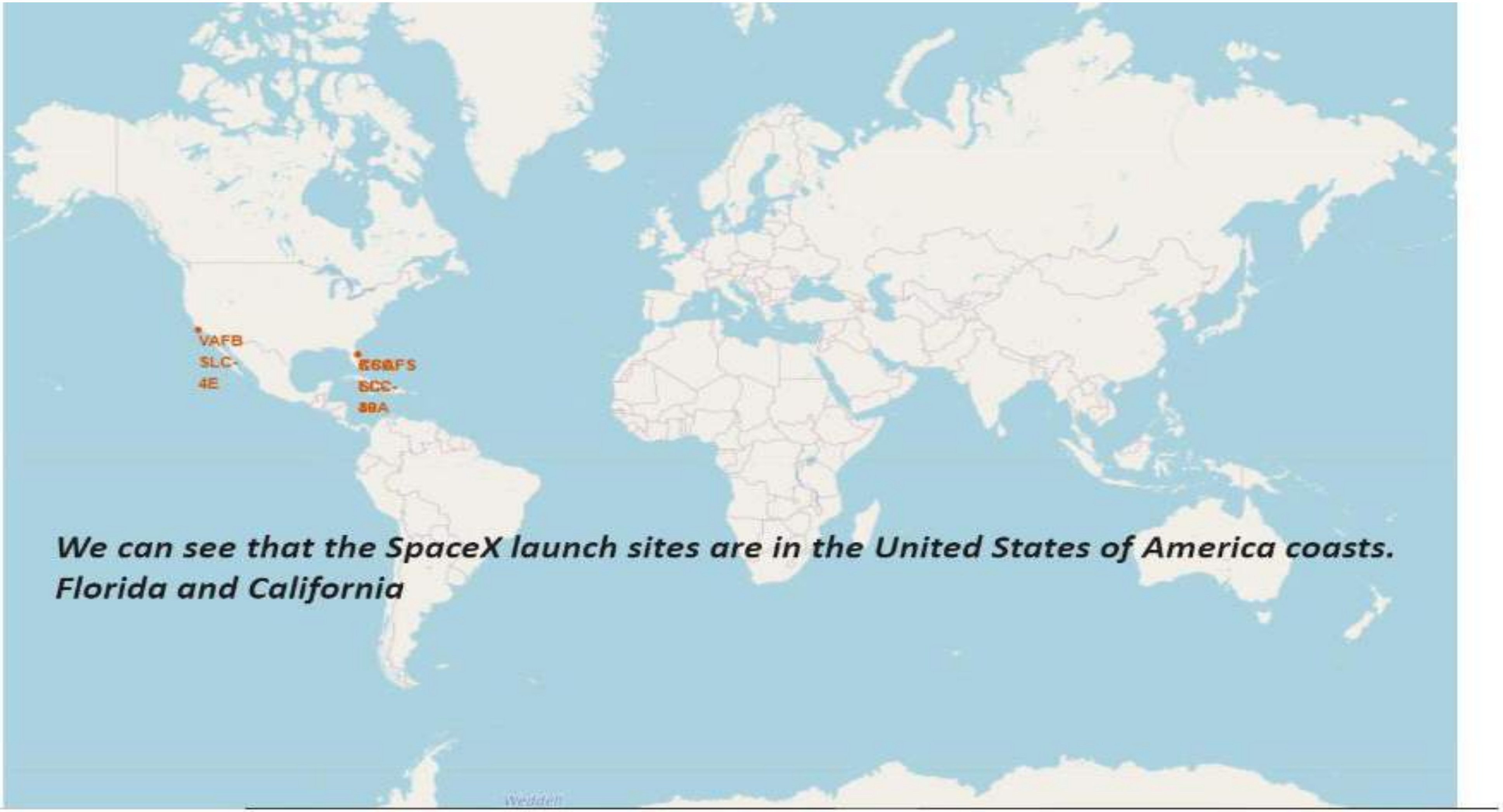


We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

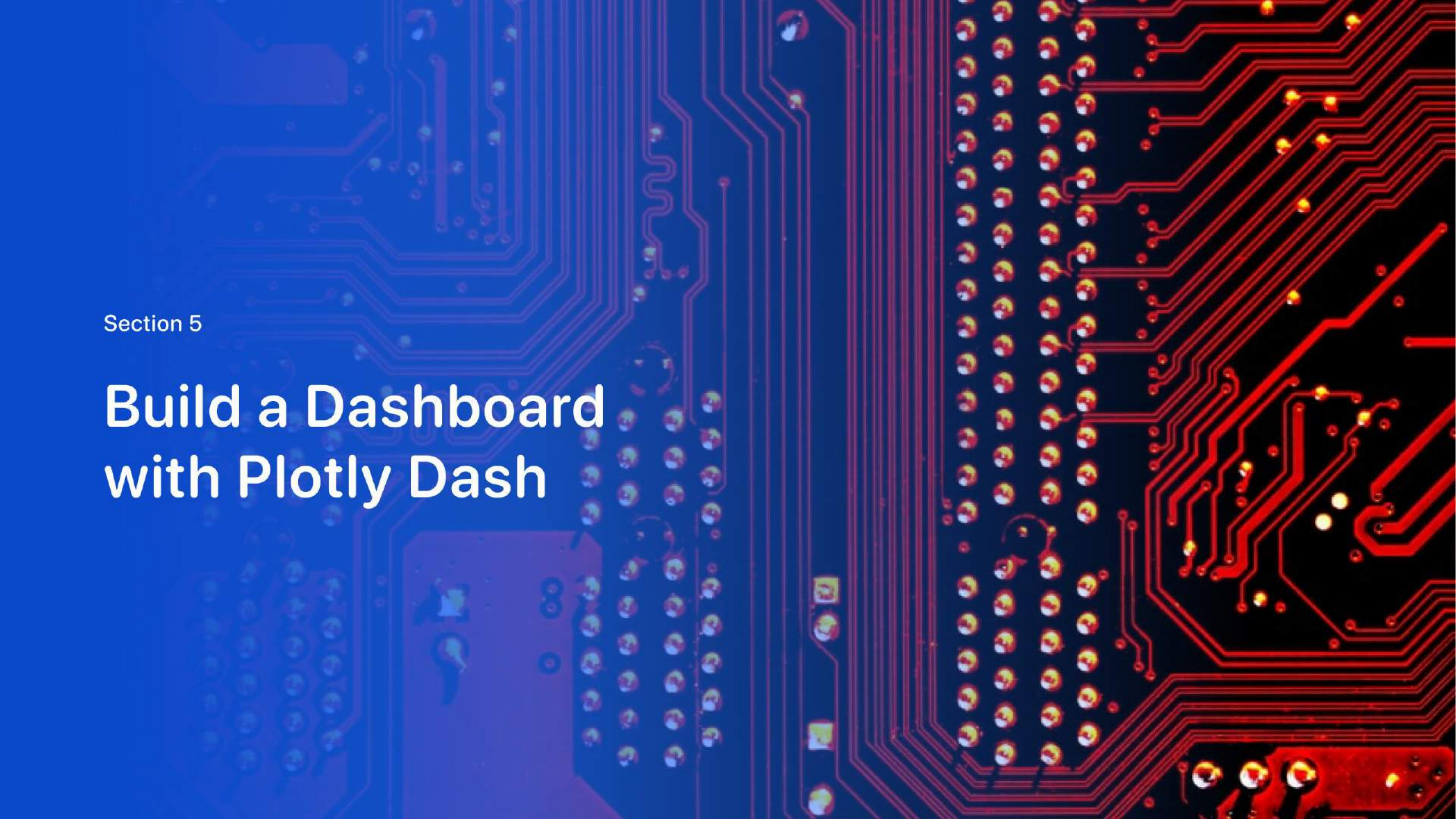
Section 4

Launch Sites Proximities Analysis

All Launch sites:-



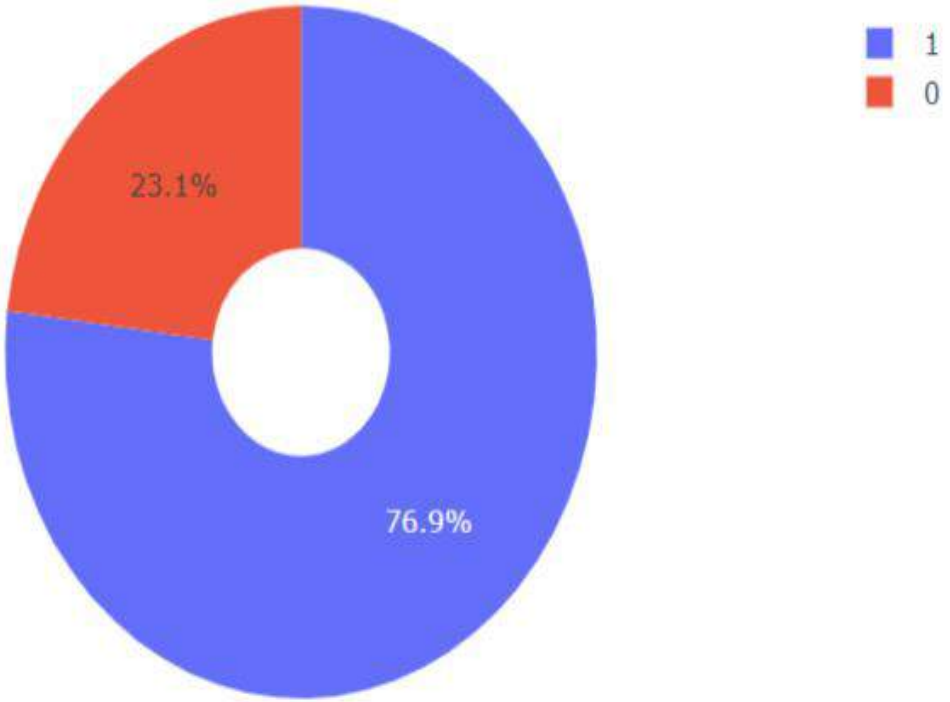
We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California



Section 5

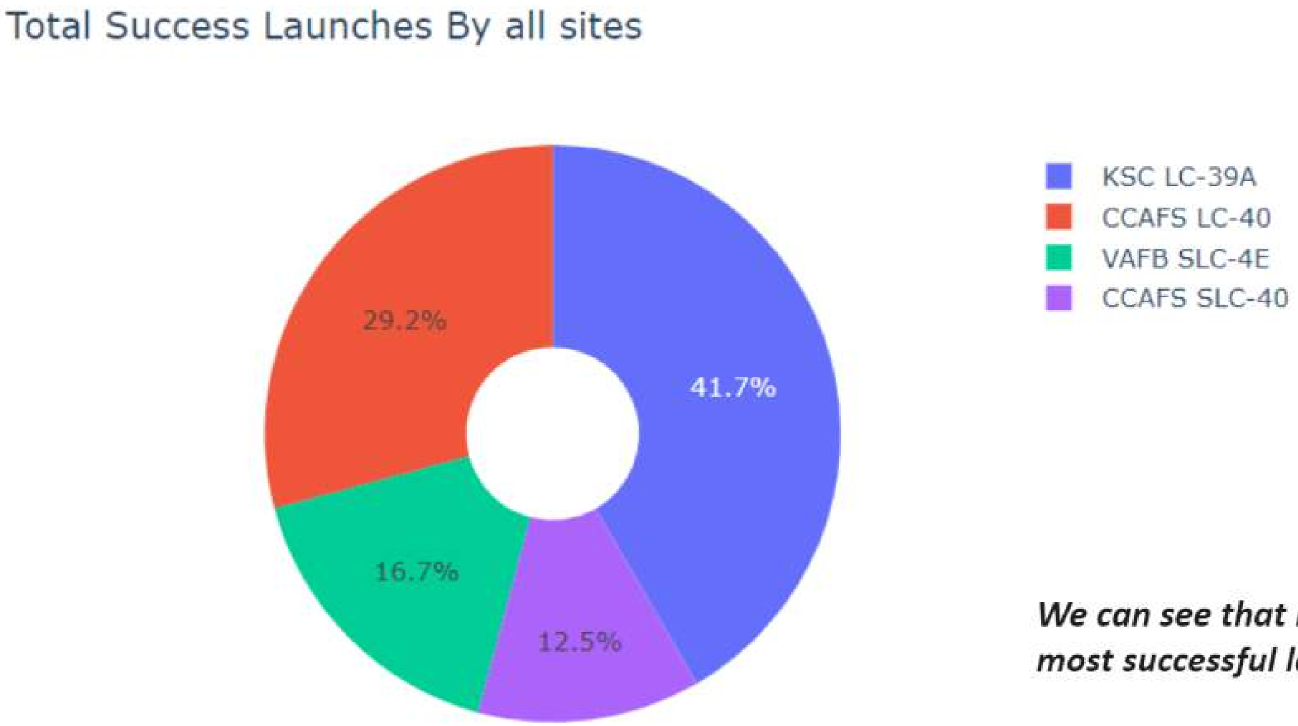
Build a Dashboard with Plotly Dash

Total Success Launch Site:-



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Highest Launch Site:-

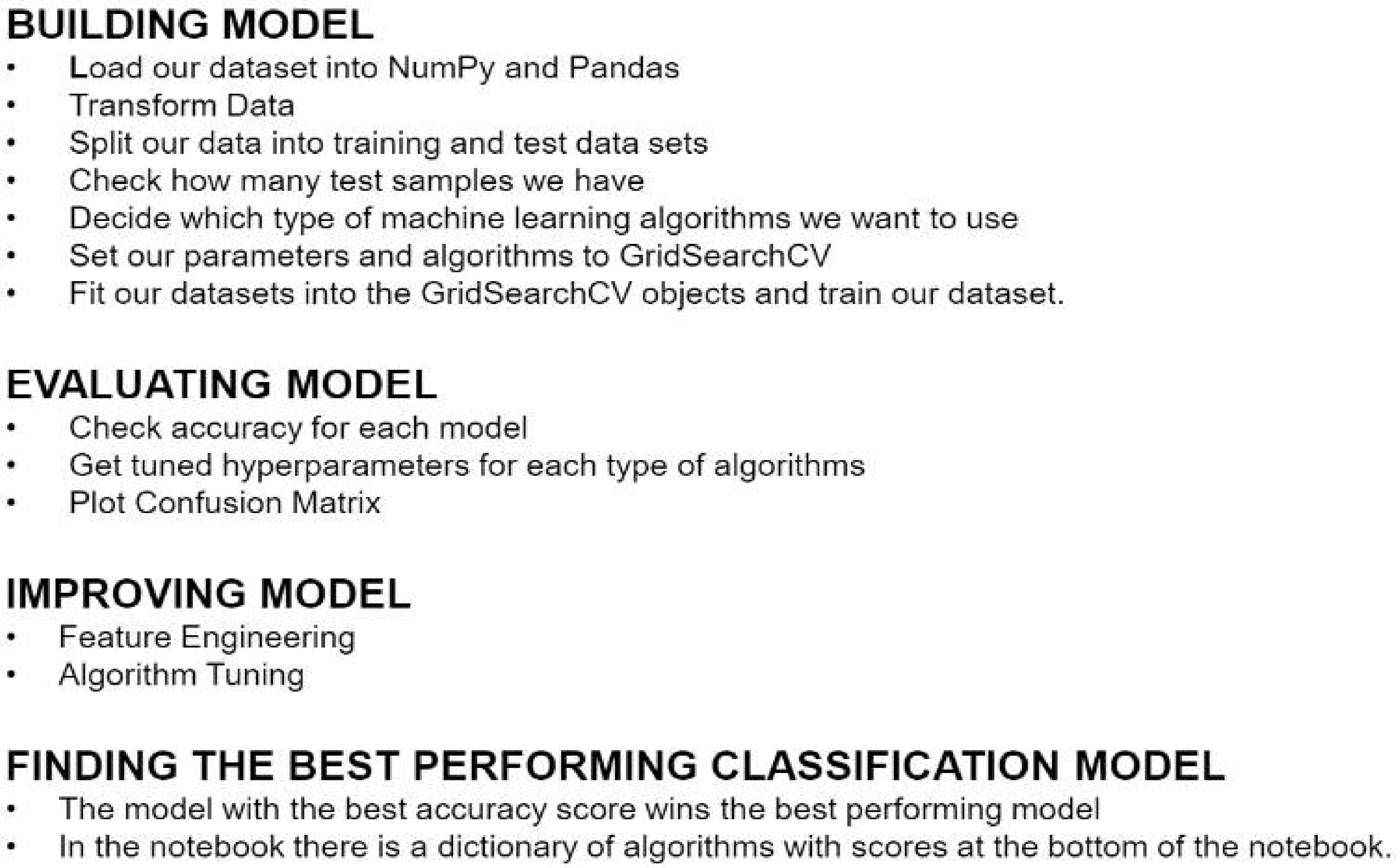


We can see that KSC LC-39A had the most successful launches from all the sites

Section 6

Predictive Analysis (Classification)

Predative Analysis:-



BUILDING MODEL

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

EVALUATING MODEL

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

IMPROVING MODEL

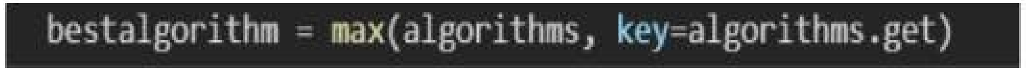
- Feature Engineering
- Algorithm Tuning

FINDING THE BEST PERFORMING CLASSIFICATION MODEL

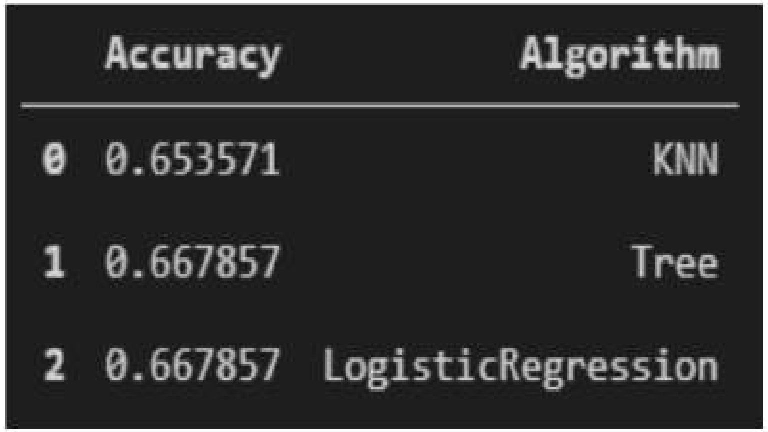
- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

Classification Accuracy using training data

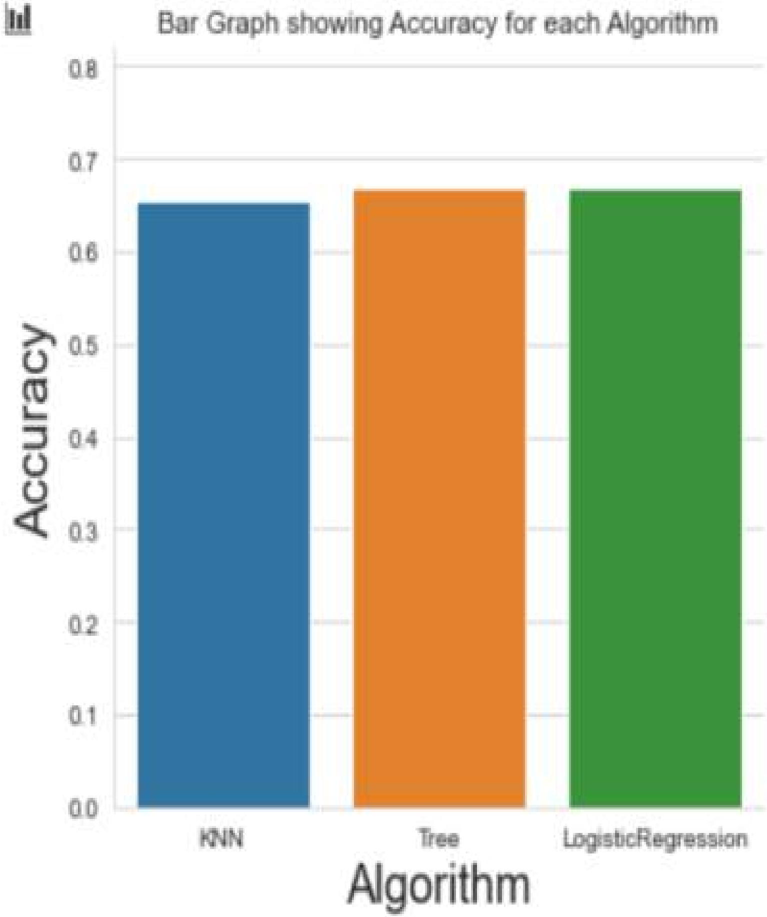
As you can see our accuracy is extremely close but we do have a winner its down to decimal places! using this function



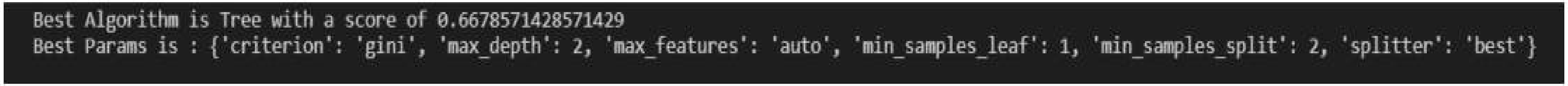
```
bestalgorithm = max(algorithms, key=algorithms.get)
```



	Accuracy	Algorithm
0	0.653571	KNN
1	0.667857	Tree
2	0.667857	LogisticRegression



The tree algorithm wins!!



```
Best Algorithm is Tree with a score of 0.6678571428571429
Best Params is : {'criterion': 'gini', 'max_depth': 2, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
```

After selecting the best hyperparameters for the decision tree classifier using the validation data, we achieved 83.33% accuracy on the test data.

Thank you!