IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Roelof Pieterson
16-01-2022

Outline

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

Executive Summary

Summary of Methodologies

- Data Collection
- Data Wrangling
- EDA with Data Visualization
- EDA with SQL
- Creating an interactive map using Folium
- Creating a Dashboard using Plotly Dash
- Predictive Analysis using Classification

Summary of all Results

- Exploratory Data Analysis
- Predictive Analysis

Introduction

Project background and context

The project focused on predicting whether the first stage of the Falcon 9 would land successfully. Falcon 9 rocket launches have a cost of 62 million dollars as advertised by SpaceX while other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Thus, if we can determine if the first stage will land then we can consequently determine the cost of the launch. This information is useful other companies that want to bid against SpaceX for a rocket launch.

Problems you want to find answers

- Factors that influence the launch success rate such as payload mass, orbit type, location and proximities of launch a site
- Find the best Hyperparameter for different Machine Learning algorithms and find the method that performs best on the test data

Section 1

Methodology

Methodology

Executive Summary

Data collection methodology:

- SpaceX Rest API
- Web Scraping Falcon 9 and Falcon Heavy Launches Records from Wikipedia

Perform data wrangling

- Mapping categorical values to integer values then representing each integer value as binary vector – *one hot encoding*

Perform exploratory data analysis (EDA) using visualization and SQL

- Plotting cat plots, bar charts, scatter plots, line charts to portray the relationships between variables

Perform interactive visual analytics using Folium and Plotly Dash

- Mark launch sites, mark the success/failed launches for each site on the map
- Calculate the distances between a launch site to its proximities

Perform predictive analysis using classification models

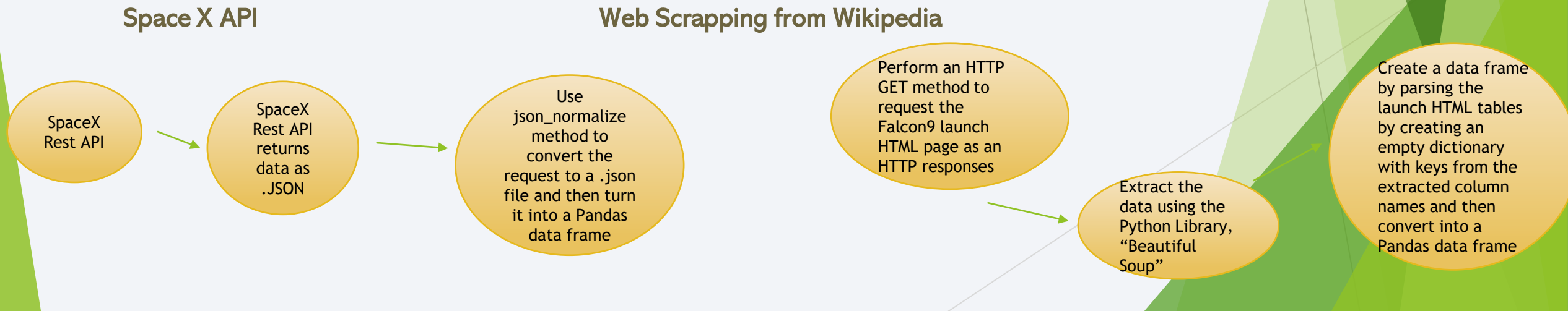
- How to build, tune, evaluate classification models

Data Collection

How were data sets collected?

- Data sets were collected by working with SpaceX launch data that we obtained from the Space Rest API which included a variety of information such as payload mass, orbit type, initial position of rocket trajectories, launch outcomes and booster landings. We request the rocket launch data from the Space X API with the following URL: *"https://api.spacexdata.com/v4/launches/past"*
- Data sets were also collected by Web scraping HTML tables from Wikipedia, parsing the tables and finally converting them to into a Pandas data frame. Here, the Python library, "Beautiful Soup" was used.

Present data collection process using key phrases and flowcharts



Data Collection – SpaceX API

1) Requesting rocket launch data from SpaceX API with the URL shown below:

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

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

```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0
```

```
response.status_code
response = requests.get(static_json_url)
```

2) Convert the request to a .json file

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

3) Apply custom functions to the clean the data

```
getBoosterVersion(data)
getPayloadData(data)
```

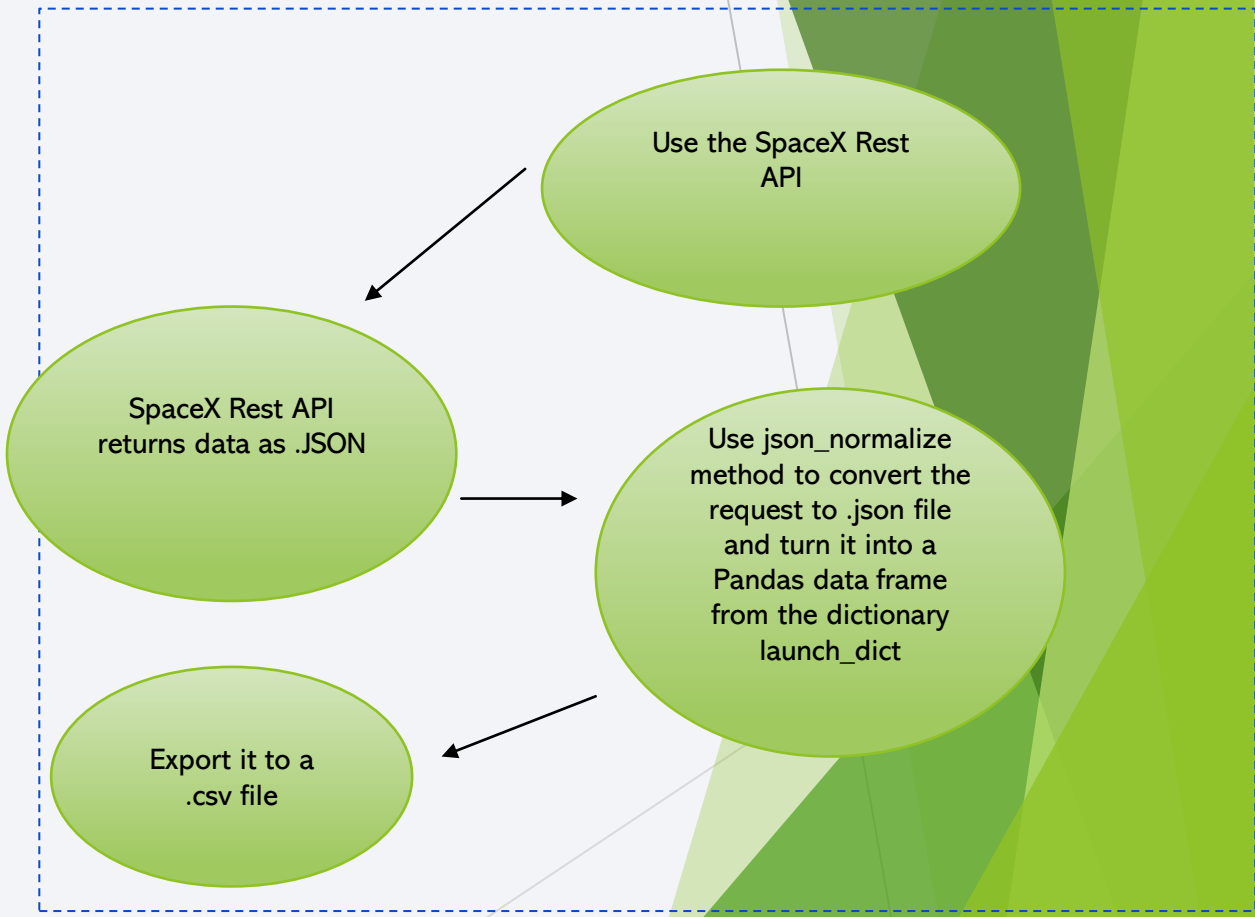
```
getLaunchSite(data)
getCoreData(data)
```

4) Construct dataset using the data we have obtained. Combine the columns into a dictionary and then create a Pandas data frame from the dictionary launch_dict and finally export it to a .csv file

```
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}
```

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

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



https://github.com/roe14/SpaceX-Falcon-9-first-stage-landing-prediction/blob/master/Data%20Collection%20API.ipynb

Data Collection - Scraping

1) Request the Falcon9 Launch Wikipedia page from its URL

```
page=requests.get(static_url)
page.status_code
```

```
Out[5]: 200
```

2) Create a BeautifulSoup object from the HTML response

```
soup=BeautifulSoup(page.text, 'html.parser')
```

3) Extract relevant column names from the HTML table header

- Find all tables on the Wikipedia page

```
html_tables=soup.find_all('table')
```

- Extract, starting from the third table is the target table

```
first_launch_table = html_tables[2]
print(first_launch_table)
```

- Extract relevant column names iterating through the <th> elements

```
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
```

4) Create an empty dictionary/covert it to a dataframe/ export it to a .csv

```
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'] = []
```

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

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

Perform an HTTP GET method to request the Falcon9 launch HTML page as an HTTP responses from Wikipedia

Extract the data using the Python Library, "Beautiful Soup"

Create a data frame by parsing the launch HTML tables into a list and then creating an empty dictionary with keys from the extracted column names and then converting into a Pandas data frame

https://github.com/roe14/SpaceX-Falcon-9-first-stage-landing-prediction/blob/master/Data%20Collection%20with%20Webscraping%20-%20Lab%202.ipynb

Data Wrangling

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 = the mission outcome was successfully landed to a specific region of the ocean
False Ocean = the mission outcome was unsuccessfully landed to a specific region of the ocean
True RTLS = the mission outcome was successfully landed to a ground pad
False RTLS = the mission outcome was unsuccessfully landed to a ground pad
True ASDS = the mission outcome was successfully landed on a drone ship
False ASDS = the mission outcome was unsuccessfully landed on a drone ship
These outcomes will be converted into Training Labels with *1* meaning the booster successfully landed and *0* meaning that it was unsuccessful

1) Exploratory Data Analysis

- Calculate the number of launches on each site
- Calculate the number and occurrence of each orbit
- Calculate the number and occurrence of mission outcome per orbit type

```
# Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

```
df["Orbit"].value_counts("Orbit")
df.Orbit.value_counts()
```

2) Create a Landing outcome from the Outcome column

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class=[]
for key,value in df ["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

```
In [15]: df['Class']=landing_class
         df[['Class']].head(8)
```

Out[15]:

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

```
In [22]: df.head()
         df.Outcome.value_counts()
```

```
Out[22]: True ASDS      41
         None None      19
         True RTLS      14
         False ASDS      6
         True Ocean      5
         None ASDS       2
         False Ocean     2
         False RTLS      1
         Name: Outcome, dtype: int64
```

Perform Exploratory data analysis

Create a landing outcome label from the Outcome column

Export it to a .csv file

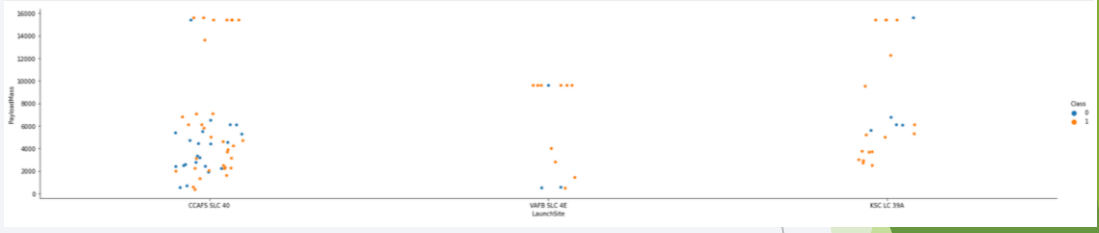
https://github.com/roe14/SpaceX-Falcon-9-first-stage-landing-prediction/blob/master/SpaceX%20Falcon9%20-%20Lab%203%20-%20Data%20Wrangling.ipynb

EDA with Data Visualization

Cat Plots – Scatter Plots were formulated to portray /visualize the following relationships:

- Flight Number (x-axis) **vs.** Pay Load Mass (y-axis)
- Launch Site (x-axis) **vs.** Flight Number (y-axis)
- Launch Site (x-axis) **vs.** Pay Load Mass (y-axis)
- Orbit Type **vs.** Fight Number
- Orbit Type **vs.** Pay Load Mass

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="PayloadMass", x="LaunchSite", hue="Class", data=df, aspect=5)
plt.xlabel("Launch Site")
plt.ylabel("Pay Load Mass in kg")
plt.show()
```



Scatter plots were used to display the relationship between two variables especially how one variable can affect another variable. In other words, is there a correlation between the two variables. As one variable increases does the other?

Bar Chart was formulated to portray/visualize the following relationships:

- Orbit Type **vs.** Success rate (class)

A Bar Chart was used to display the relationship between the two axes. A Bar chart was used to measure the change over time especially when it concerns larger changes.

Line Chart was formulated to portray/visualize the following relationship:

- Year **vs.** Average Success Rate

A Line Graph was used to track changes over periods of time. Moreover, data patterns and trends over time become more evident making it easier to form predictions.

https://github.com/roe14/SpaceX-Falcon-9-first-stage-landing-prediction/blob/master/Space%20x%20-%20Lab%204%20-%20EDA%20-%20Visualisation.ipynb

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 'KSC'
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display the average payload mass carried by booster version F9 v 1.1
- List the date where the successful landing outcome in drone ship was achieved
- List the names of the boosters which have success in ground pad 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 months names, successful landing outcomes in ground pad, booster versions, launch site for the months in the year 2017
- Rank the count of successful landing outcomes between the date 04-06-2010 and 20-03-2017 in descending order

https://github.com/roe14/SpaceX-Falcon-9-first-stage-landing-prediction/blob/master/jupyter-labs-eda-sql-edx_sqllite.ipynb

Build an Interactive Map with Folium

Each site's launch location was added on the map using the respective site's latitude and longitude coordinates. In doing so, *folium.Circle and folium.Marker were* used to add a highlighted circle/marker with a text label on a specific coordinate/launch site displaying the name of the launch site.

The map objects were added to mark the success/failed launches for each site on the map. Using the "class" column, markers were created for all launch records. If a launch was a success (class=1) a green marker was assigned and if it was a failure (class=0) a red marker was assigned.

An analysis was undertaken to explore the proximities of launch sites. By adding *MousePosition()* on the map one could find the distances from the launch sites to any points of interests such as railways, highways or closest coastline. A *PolyLine* can be drawn between the respective launch site to the landmark of interest to measure the distance. From this, one can analyze certain trends of whether launch sites tend to be located near coastlines, highways and so on.

https://eu-gb.dataplatform.cloud.ibm.com/analytics/notebooks/v2/a88c0720-a738-47ee-8ccb-f67a4cfc88a9?projectid=c226f2af-fcab-48dd-8866-9edd0a807ec3&context=cpdaas

Build a Dashboard with Plotly Dash

The Dashboard Applications were built with Plotly Dash. The plots and graphs added to the dashboard were as follows:

1) **Pie Chart** – To show the total successful launches count for all sites. A slider was added to select the payload range. A callback function was added to get the selected launch site from site *dropdown* and render a pie chart visualizing launch success counts

2) **Scatter Chart** – To show the correlation between payload and launch success (class column) so we can visually observe how payload may be correlated with mission outcomes for selected sites A callback function was added to render this plot for site-dropdown and payload slider as inputs and Success-payload scatter chart as output

https://github.com/roe14/SpaceX-Falcon-9-first-stage-landing-prediction/blob/master/Plotly%20-%20DashBoards%20-%20SpaceX.ipynb

Predictive Analysis (Classification)

Building the Model

1) Load our dataframe
2) Create Numpy array from the column *'Class"* (launch success)
3) Standardize the data in X
4) Split our data into training and testing data sets
5) Check how many test samples we have – *"y_test.shape"*
6) Decide which Machine Learning Algorithm you wish to use
7) Hyperparameters are then selected using the function *GridSearchCV*

Evaluating the Model

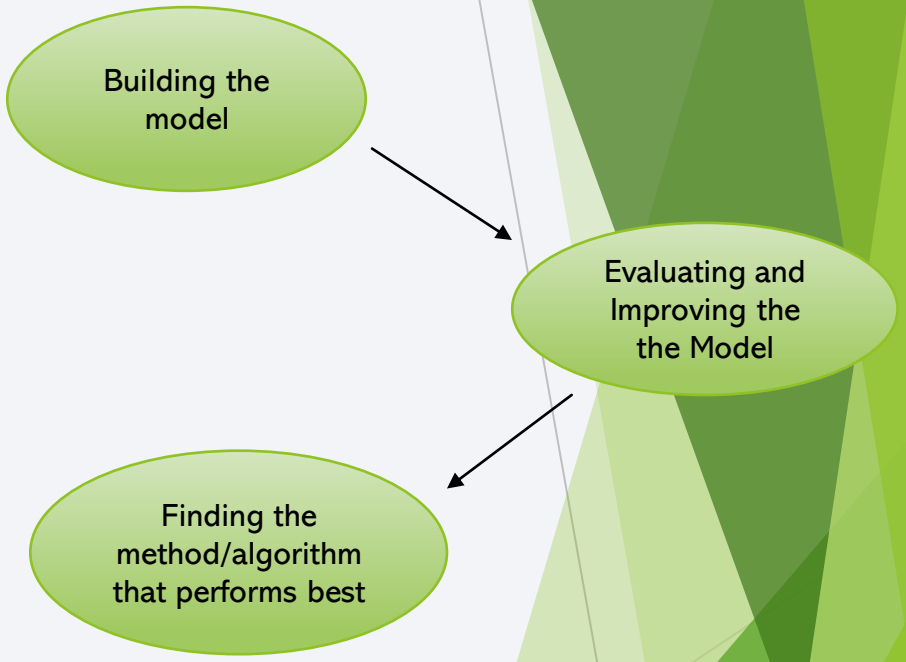
1) Use the method, "score" to calculate the accuracy on the test data (X_test, Y_test)
2) Tune the Hyperparameters for each algorithm
3) Plot and analyze the Confusion Matrix to distinguish between the different classes

Improving the Model

1) Tuning the Hyperparameters for each algorithm
2) Feature Engineering

Finding the method/algorithm that performs best

1) The model with the best accuracy score is the one that performs the best



https://github.com/roe14/SpaceX-Falcon-9-first-stage-landing-prediction/blob/master/Space%20X%20-%20Machine%20Learning.ipynb

Results

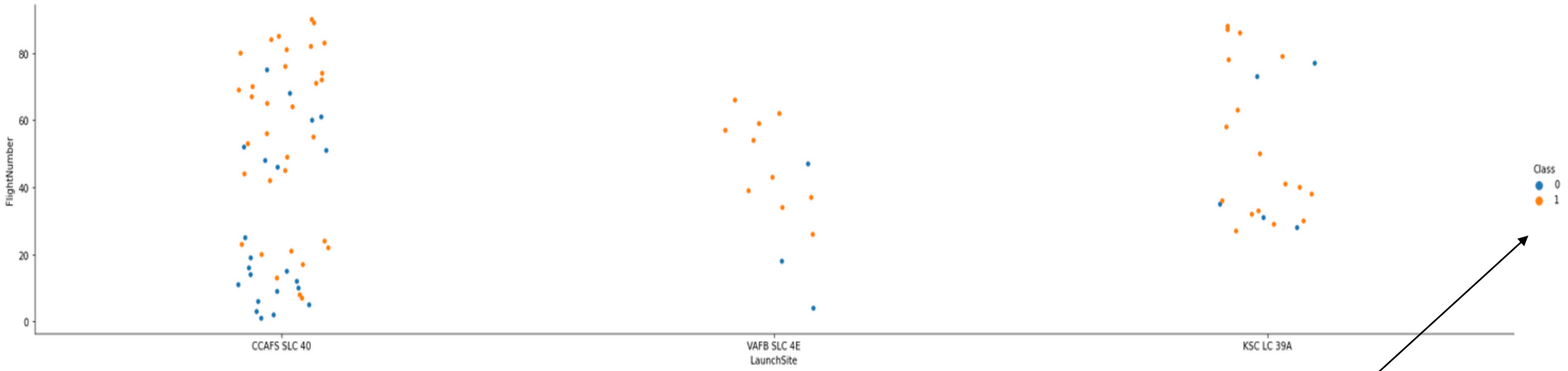
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

Flight Number vs. Launch Site – Scatter Plot

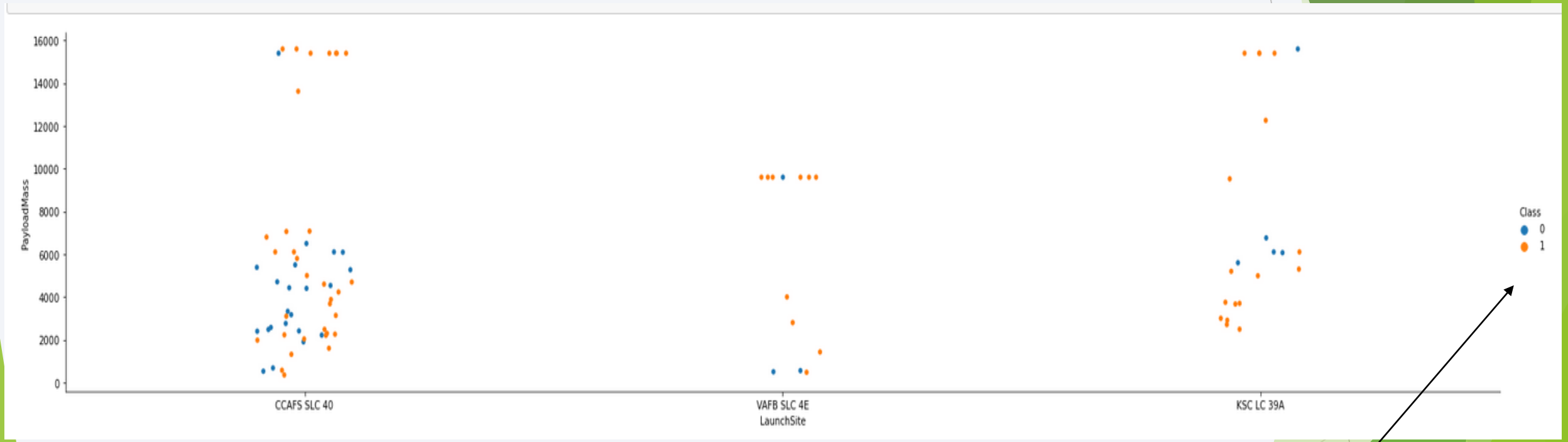


- The Scatter Plot illustrates that if there are more flights at a launch site then the rate of success increases.
- CCAFS SLC 40 is the busiest launch site in terms of the number of flights that launch from there

Key: Class (Launch Outcome)
Blue (0) = Failure
Orange (1) = Success

Payload vs. Launch Site

Payload vs. Launch Site – Scatter Plot

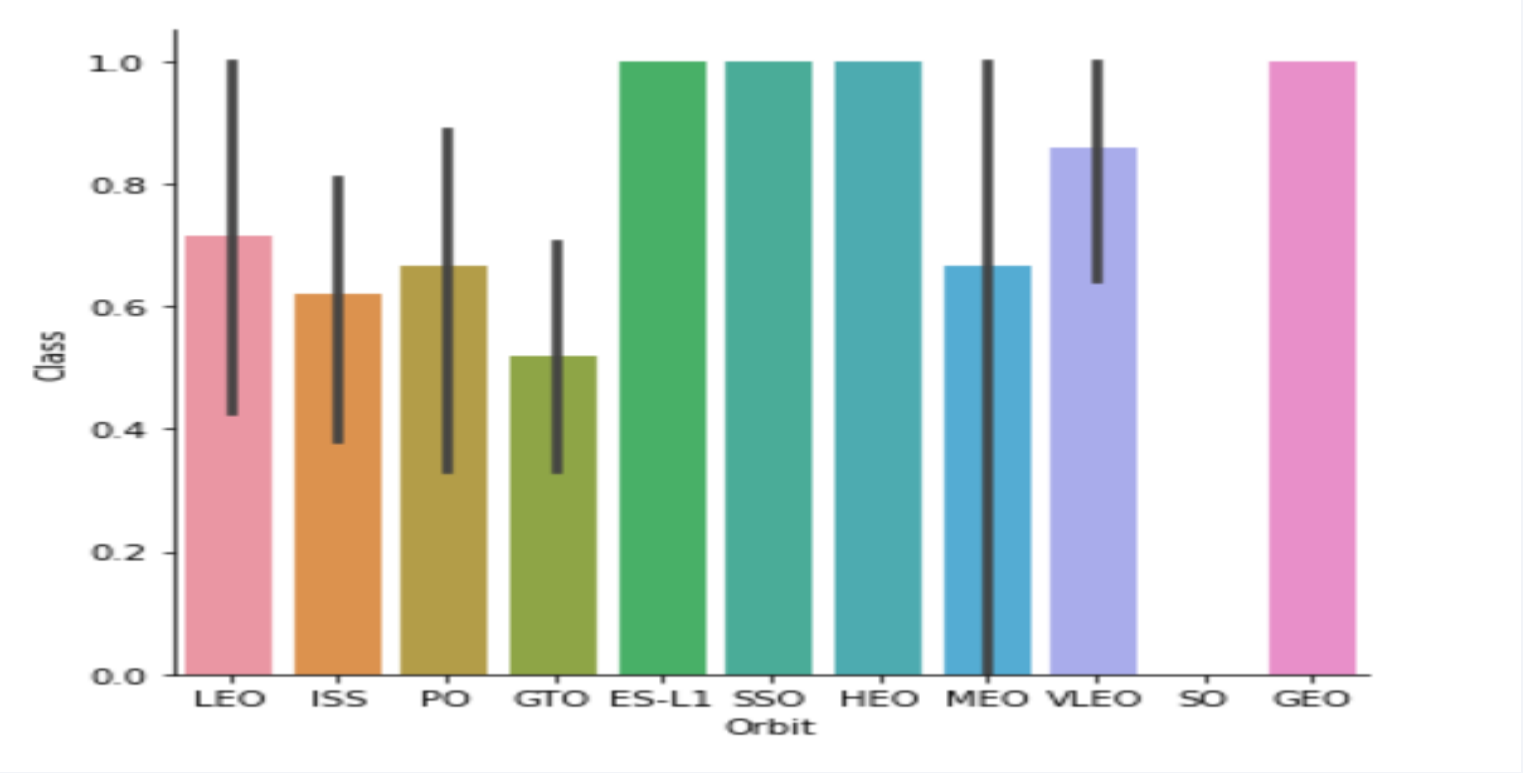


- At the VABF SLC 4E launch site there aren't any rockets launched with a payload mass greater than 10,000
- At the CCAFS SLC 40 launch site, the greater the payload mass, the greater the probability of a successful launch
- There seems to both failed and successful launches of low, mid and high payload masses. Therefore, there isn't a clear pattern to suggest that the success rate of a launch is dependent on the payload mass.

Key: Class (Launch Outcome)
Blue (0) = Failure
Orange (1) = Success

Success Rate vs. Orbit Type

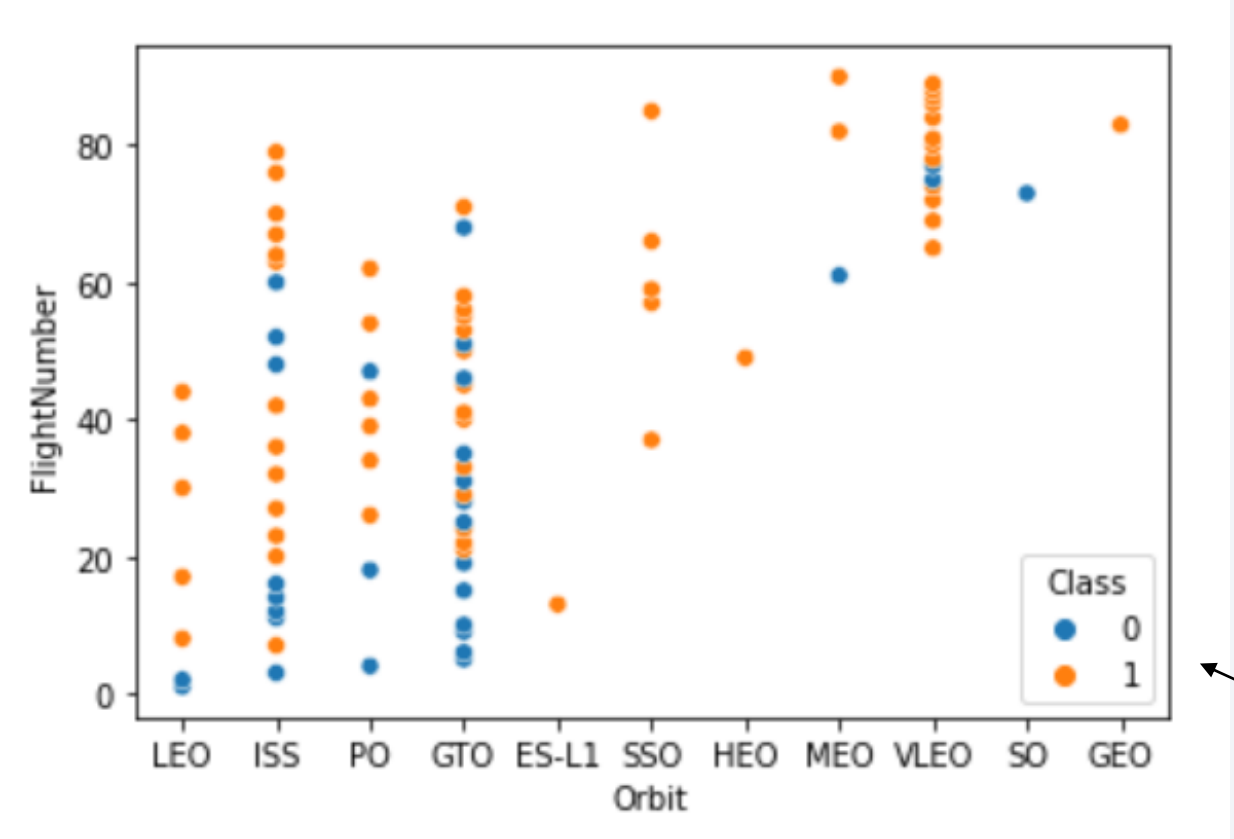
Success Rate(Class) vs. Orbit Type – Bar Chart



- The bar chart portrays that the orbit types; ES-L1, SSO, HEO, GEO have the highest success rates.
- Orbit type, GTO has the lowest success rate

Flight Number vs. Orbit Type

Flight Number vs. Orbit Type – Scatterplot

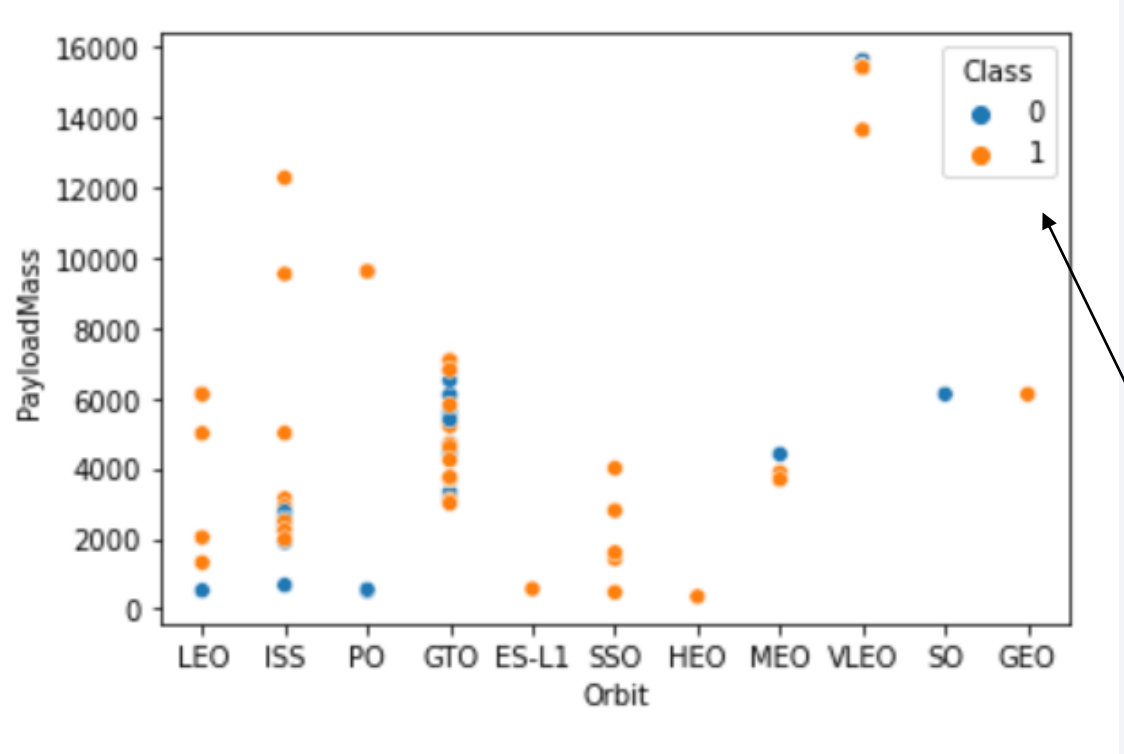


- In LEO orbit the success seems to be related to the number of flights
- However, success is not related to the number of flights in GTO orbit in where there are a number of both failed and successful flights
- In GEO orbit the number of flights exceeds 75 and the success rate is 100 percent

Key: Class (Launch Outcome)
Blue (0) = Failure
Orange (1) = Success

Payload vs. Orbit Type

- Payload vs. Orbit Type – Scatter Plot

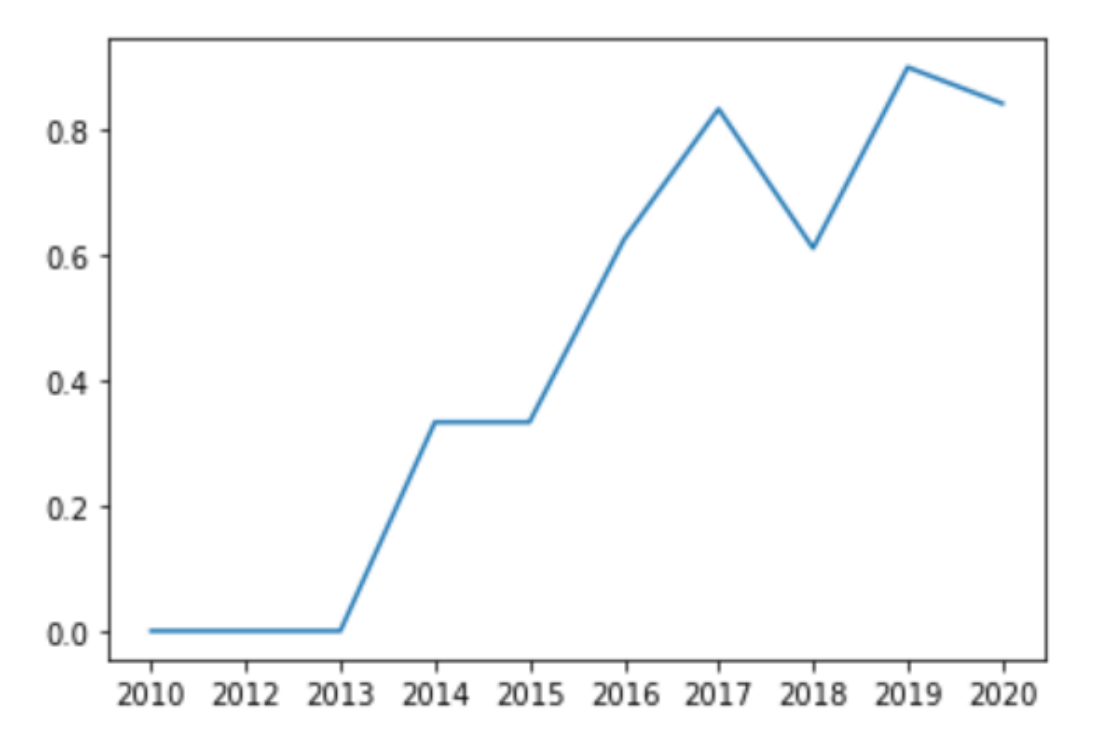


- Orbit Types, LEO, ISS and PO all show that heavy payloads lead to a positive landing rate.
- However, for GTO this positive correlation can not be seen as heavier payloads lead to both successful and failed landing outcomes.
- More heavier payloads ranging from 13,000 to 16,000 for VLEO have failed and successful landing outcomes as well.

Key: Class (Launch Outcome)
Blue (0) = Failure
Orange (1) = Success

Launch Success Yearly Trend

Launch Success Yearly Trend – Line Graph



- There is a constant increase in the Launch Success rate from 2010 till 2020 with a few fluctuations.

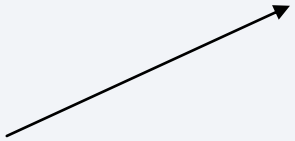
All Launch Site Names

EDA - SQL

%sql select distinct (Launch_Site) from SPACEXTBL

The code shown above displays the names of the unique launch sites in the space mission.

The “select” statement returns a result set of records from the column, “Launch_Site” in this case, the names of all the launch sites.



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

Launch Site Names Begin with ‘KSC '

EDA – SQL

%sql select launch_site from SPACEXTBL where (launch_site) like 'KSC%' limit 5

The Code shown above displays 5 records where launch sites begin with the string, “KSC”

Launch_Site
KSC LC-39A
KSC LC-39A
KSC LC-39A
KSC LC-39A
KSC LC-39A

The “where” clause is used to filter records or rather extract only those records that fulfill a specified condition. The logical operator “like” is then used to determine whether a specific character string matches a specified pattern such as launch sites that start with “KSC.”

The “limit” clause restricts or limits how many rows are returned from a query

Total Payload Mass

EDA – SQL

```
%sql select sum(payload_mass__kg_) as total_payload_mass from SPACEXTBL where (Customer) = "NASA (CRS)"
```

total_payload_mass
45596

The code shown above calculates the total payload carried by boosters from NASA by the rocket CRS

The “sum” function returns the total sum of a numeric column (payload_mass_kg) column.

The “as” statement is a way to rename the column in the output/assign aliases.

The “where” clause is used to filter the customer column to the “NASA (CRS)” customers

Average Payload Mass by F9 v1.1

EDA – SQL

```
%sql select avg(payload_mass__kg_) as average_payload from SPACEXTBL where
(booster_version) = 'F9 v1.1'
```

average_payload
2928.4

The code shown above calculates the average payload mass carried by booster version F9 v 1.1

The “select” statement returns data stored in the payload_mass_kg column while the “avg” function returns the average value of that numeric column.

The “where” clause is then used to filter the average payload to a specific booster version – F9 v1.1

First Successful Ground Landing Date – Drone Ship

EDA – SQL

```
%sql select min(date) from SPACEXTBL where "Landing _Outcome" = 'Success (drone ship)'
```

min(date)
06-05-2016

The code shown above finds and returns the date where the first successful landing outcome in drone ship was achieved.

The “min” function returns the smallest value of the selected column which would be the “date” column.

The “where” clause is used to filter out the landing outcome column and the “==“ operator is used return the matched record.

Successful Drone Ship Landing with Payload between 4000 and 6000

EDA – SQL

```
%sql select Booster_Version from SPACEXTBL where "Landing _Outcome" ='Success (drone ship)' and
payload_mass__kg_ between 4001 and 5999
```

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

The code shown above returns a list of names of boosters which have successfully landed on drone ship with a payload mas between 4000 and 6000.

The Booster_Version column is selected then specifying the criteria with the “where” clause, landing outcomes that were a success and that had a payload mass between 4001 and 5999 were selected.

Total Number of Successful and Failure Mission Outcomes

EDA – SQL

```
%sql select mission_outcome, count(mission_outcome) as outcome from SPACEXTBL group by mission_outcome
```

Mission_Outcome	outcome
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

The code shown above calculates the total number of successful and failed mission outcomes.

The mission_outcome column is selected, “count” is used to calculate the total sum of the mission_outcome column and it is portrayed as “outcome” in the output by using the statement “as.”

The “group by” statement partitions result rows into groups, based on their values in the mission_outcome column

Boosters Carried Maximum Payload

EDA – SQL

```
%sql select booster_version from SPACEXTBL where payload_mass__kg_ = (select
max(payload_mass__kg_) from SPACEXTBL)
```

The code shown above lists the names of the booster which have carried the maximum payload

The booster_version column is selected and then the “where” clause is used to specify the criteria.

The “max” function returns the largest value for that selected column that being the payload_mass_kg_ column or rather the boosters that carry the maximum payload. As we can see, there are 12 booster versions that carry the maximum payload.

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

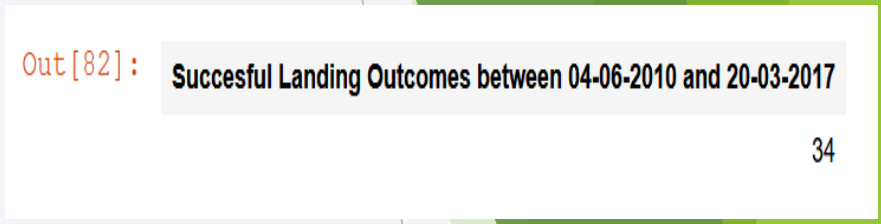
- List the records which will display the month names, succesful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017

- Present your query result with a short explanation here

Rank the count of Successful Landing Outcomes Between 2010-06-04 and 2017-03-20

EDA – SQL

```
%sql select count ("Landing _Outcome") as "Succesful Landing Outcomes between 04-06-2010 and 20-03-2017" from SPACEXTBL where ("Landing _Outcome" like '%Success%') and (date >= "04-06-2010") and (date <= "20-03-2017")
```



The code shown above finds the rank of the count of successful landing outcomes between the date 04-06-2010 to 20-03-2017 in descending order.

After the “where” clause is used to specify the criteria, the logical operator “like” is used to perform pattern matching in this case if the character string matches Success.

Logical operators such as greater than or equal to and lesser than or equal to are used to focus between specific dates.

Section 4

Launch Sites Proximities Analysis

Global Launch Sites

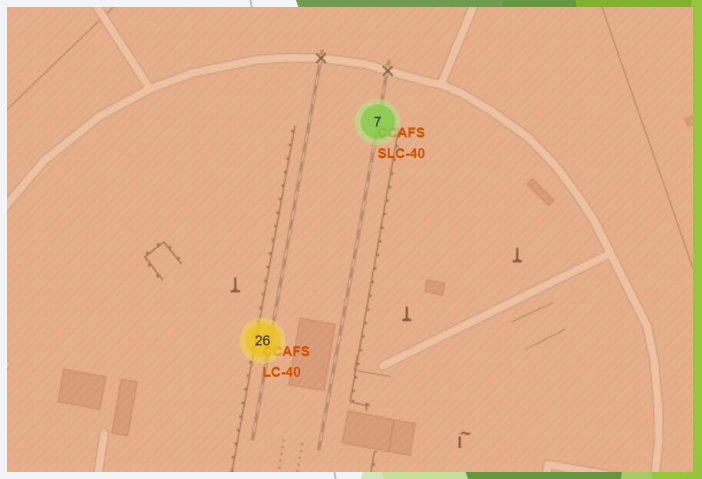
The map depicts the global launch sites. As shown, the SpaceX launch sites are located in the USA, more specifically in the states of California and Florida.

Color-labeled launch outcomes on Folium Map

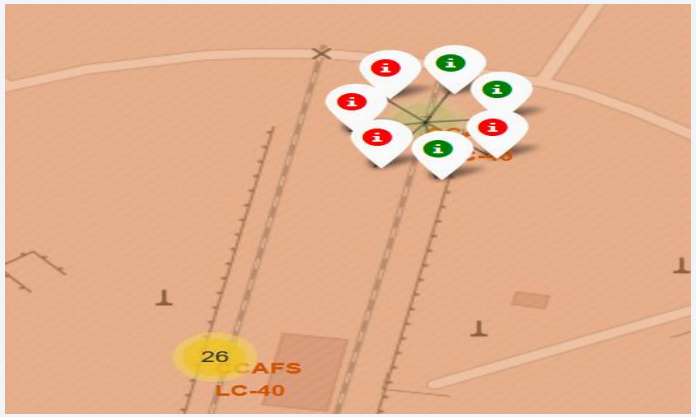
The screen shots on the left portray the **Florida Launch Sites**

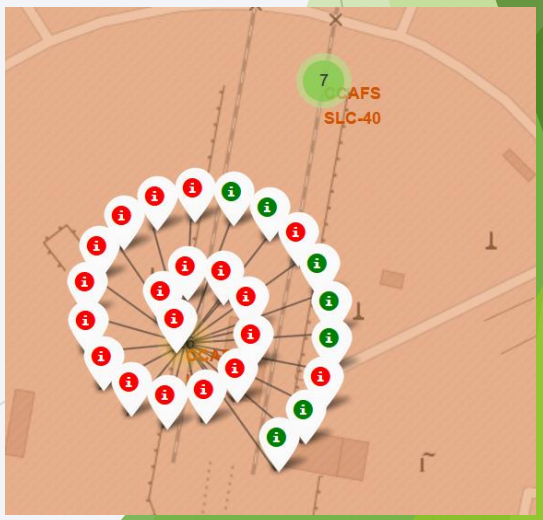
1) Launch site 7:
   - Red Marker = Failed Launch
   - Green Marker = Successful Launch
2) Launch site 26:
   - Red Marker = Failed Launch
   - Green Marker = Successful Launch

Zoom in



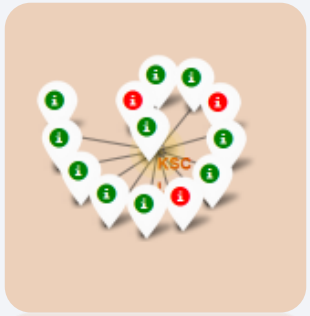
Zoom in





California Launch sites

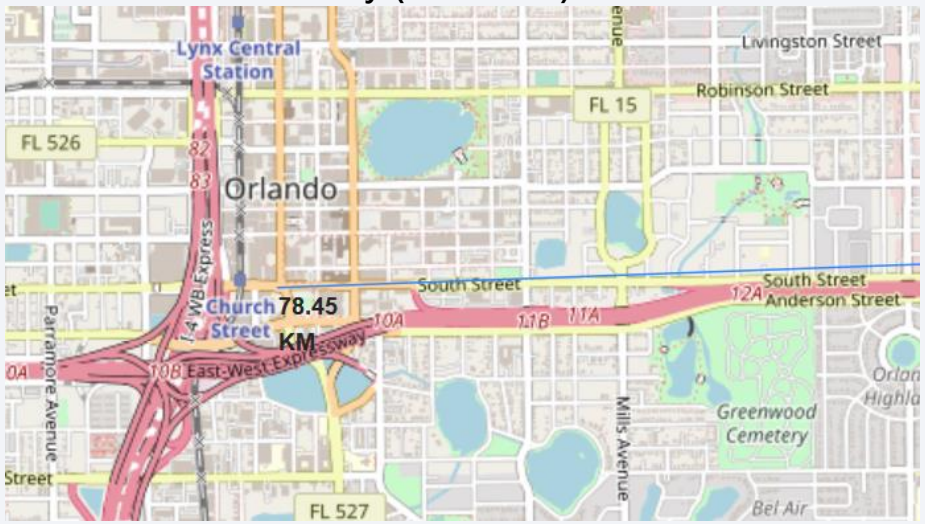
Another Florida Launch Site



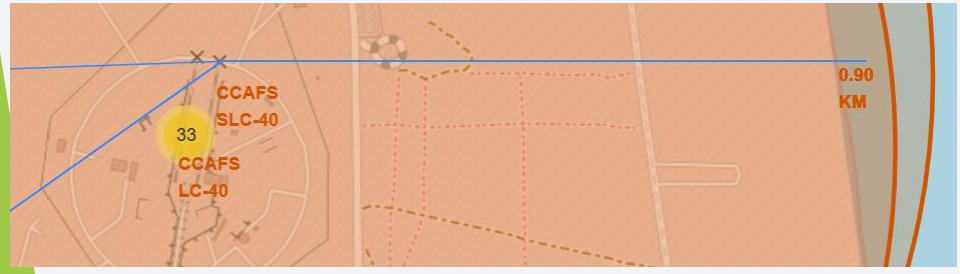
Distances from Launch Sites to Landmarks – Using CCAFS SLC-40 Launch Site as the reference

Distance from Launch Site in Florida to various landmarks (Blue Line represents the distance)

Distance from Launch Site to the nearest city (78.45 km)



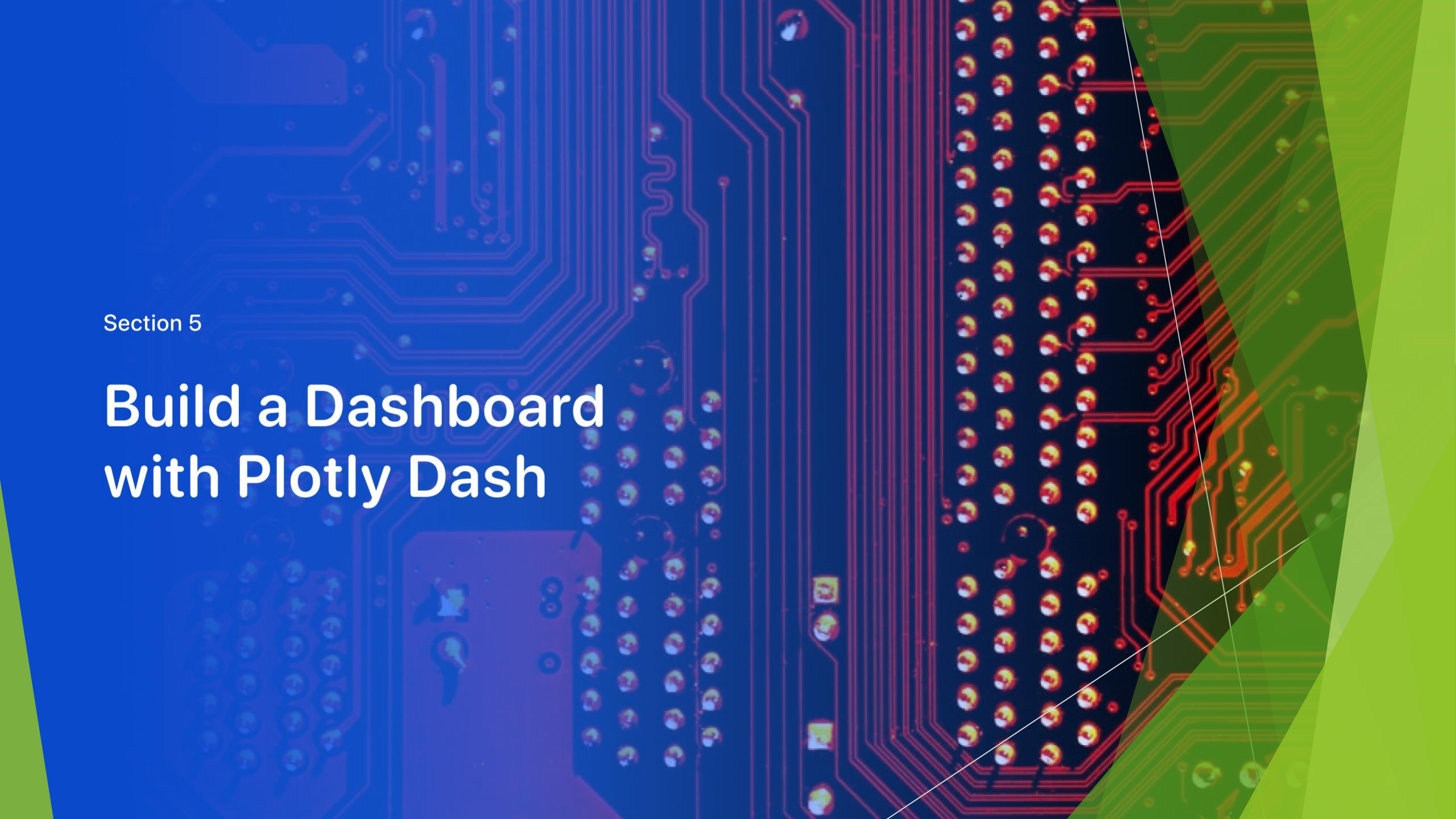
Distance from Launch Site to the nearest coast (0.90 km)



Distance from Launch Site to the nearest highway (29.21 km)

Observations

- Launch sites in Florida **are not** in close proximity to Highways
- Launch sites in Florida **are** in close proximity to the nearest coastline
- Launch sites in Florida **are not** in close proximity the nearest city

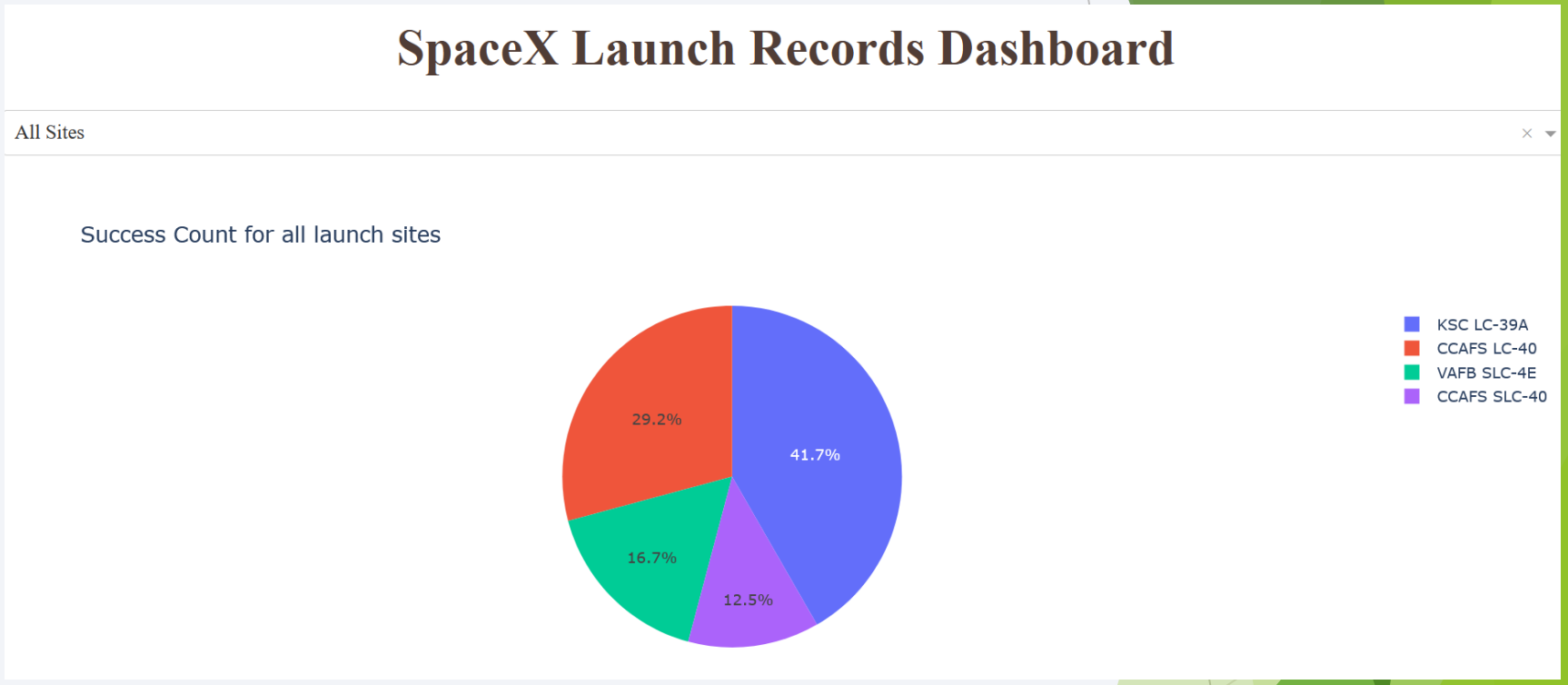

Section 5
Build a Dashboard with Plotly Dash

Pie Chart – Success Count for all launch sites

The Pie Chart on the right shows the success count for all launch sites in a percentage.

KSC-LC-39A had the highest percentage of successful launches.

CCAFS SLC-40 launch site had the lowest percentage of successful launches

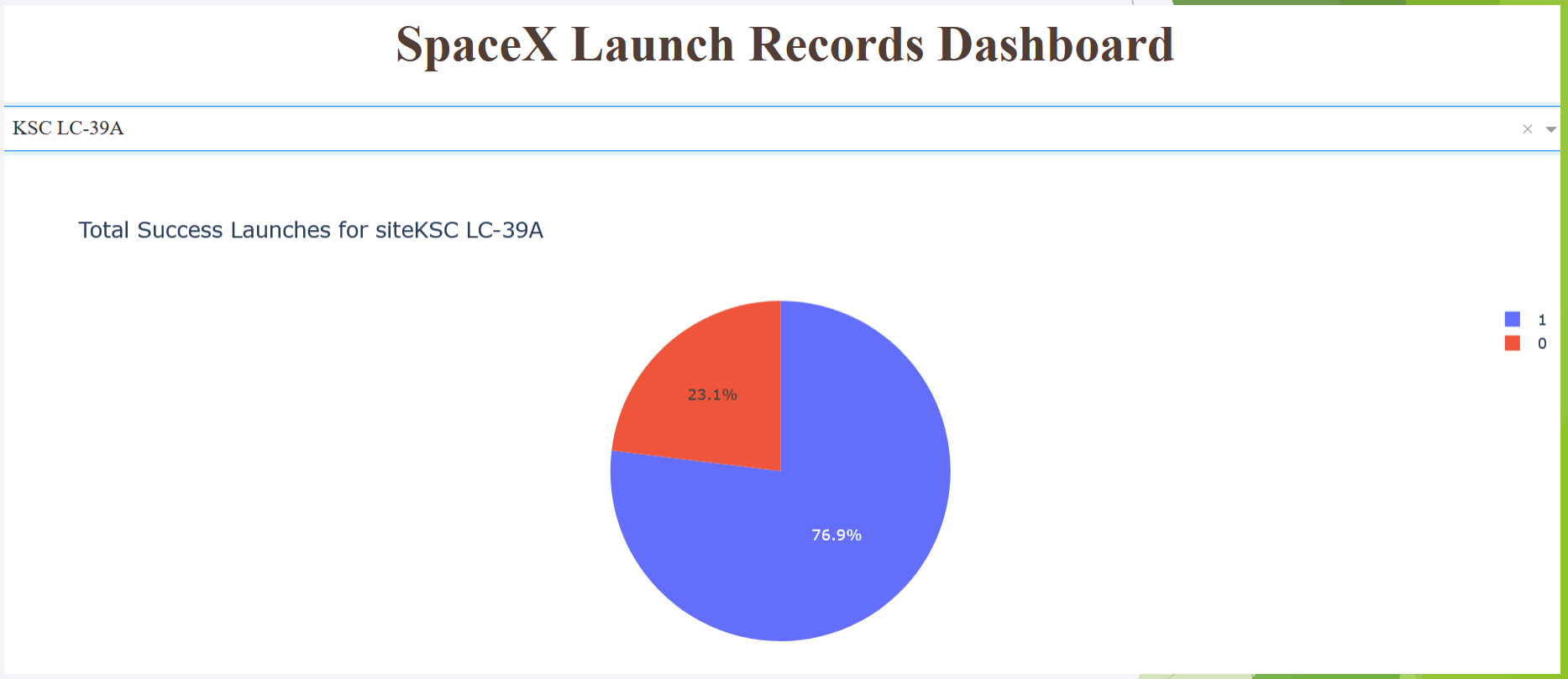


Pie Chart – Launch Site with Highest Launch Success Ratio

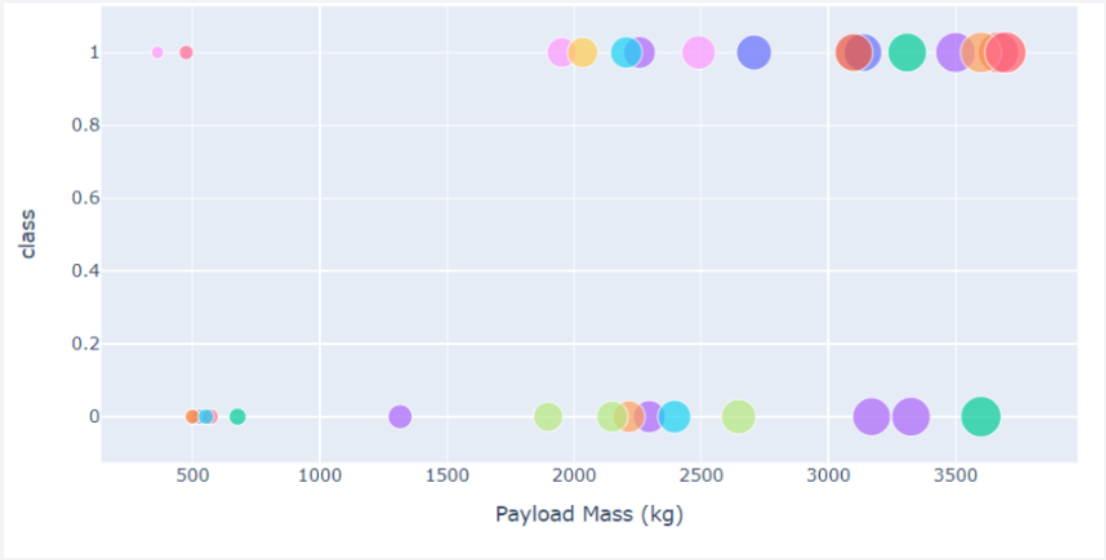
Pie Chart portraying Launch Site, KSC LC-39 which has the highest launch success ratio out of all the sites.

KSC LC-30 has a 23.1& rate of failed launches.

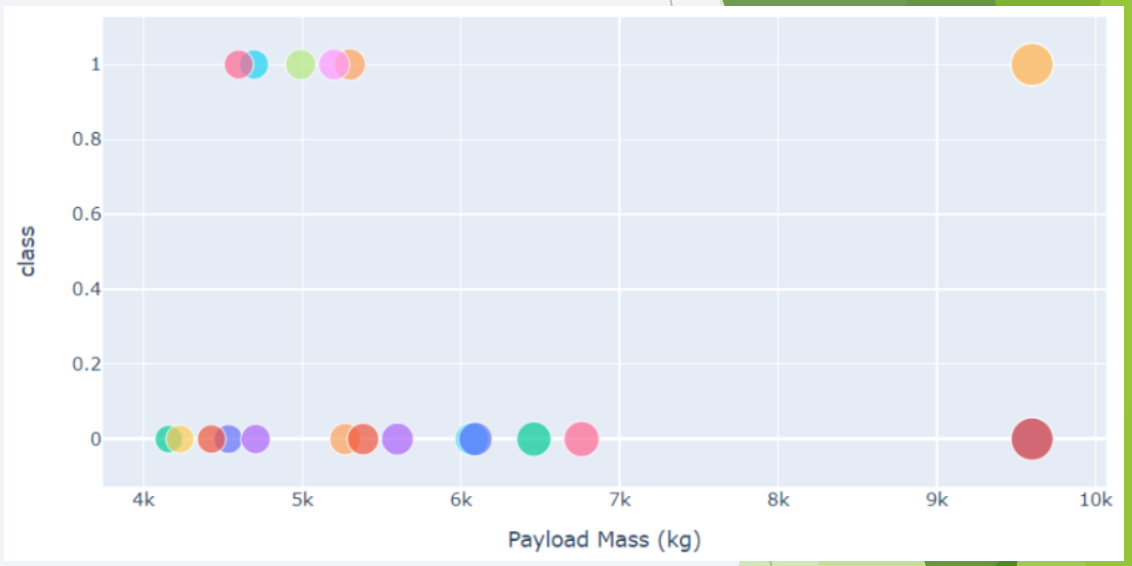
KSC LC-30 has a 76.9 % rate of successful launches.



Dashboard – Payload v Launch Outcome scatter plot for all the launch sites with different payload selected slider.



Payload from 0kg – 4000kg



Payload from 400kg – 10,000kg

The image on the left portrays that rockets with a payload between 0kg and 400kg have a higher degree of success (class)

However, as shown by the image on the left, payloads that range from 400kg to 10,000 kg have a much lower rate of success (class)

Section 6

Predictive Analysis (Classification)

Classification Accuracy

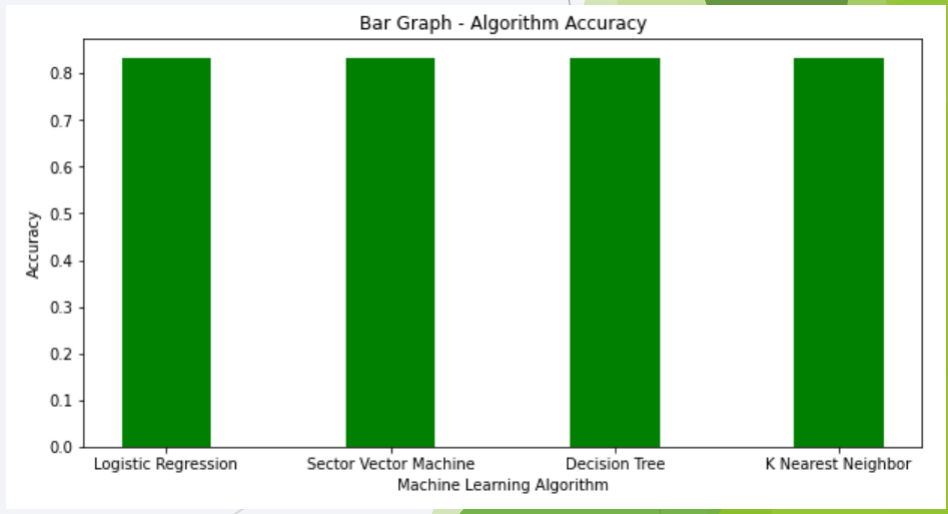
The Best Algorithm is the K-Nearest Neighbor

```
In [33]:
```

```
#Find the method performs best:
scores_accuracy=[lr_accuracy,svm_accuracy,tree_accuracy,knn_accuracy]
print(scores_accuracy)
print(scores_accuracy.index(max(scores_accuracy)))
print("The Best Algorithm is K-Nearest Neighbor:",knn_accuracy)
```

```
[0.8333333333333334, 0.8333333333333334, 0.8333333333333334, 0.8333333333333334]
0
The Best Algorithm is K-Nearest Neighbor: 0.8333333333333334
```

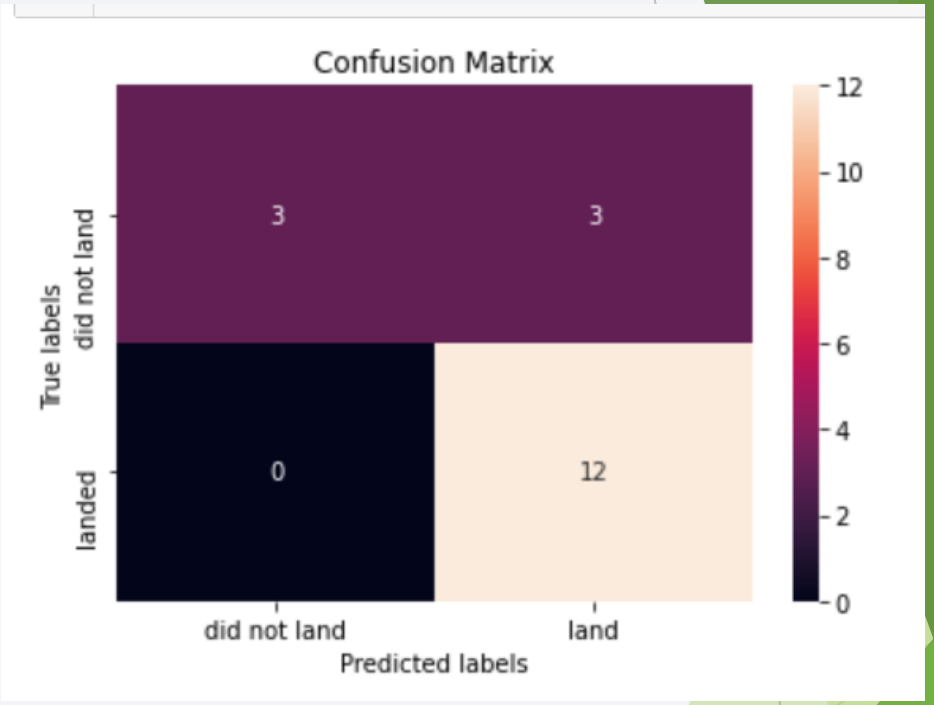
```
data = {"Logistic Regression":0.8333333333333334, 'Sector Vector Machine':0.8333333333333334, 'Decision Tree':0.
courses = list(data.keys())
values = list(data.values())
fig = plt.figure(figsize = (10, 5))

plt.bar(courses, values, color ='green',
        width = 0.4)

plt.xlabel("Machine Learning Algorithm")
plt.ylabel("Accuracy")
plt.title("Bar Graph - Algorithm Accuracy")
plt.show()
```



Confusion Matrix

The image on the right shows the Confusion Matrix for the highest accurate algorithm – K Nearest Neighbour.

- We can see that there are 3 **false positives** (top –right) meaning that we predicted a “yes” but that wasn’t the case
- There are 0 **False negatives** (bottom-left)
- There are 3 **true negatives** (top- left) meaning that we predicted a negative, “no” prediction and in these 3 cases, the predictions were accurate
- However, there are 12 **true positives** (bottom-right) meaning that in these cases our predictions were accurate



Conclusions

- KSC LC – 39A is the launch site that had the most successful launches
- There is a constant increase in the Launch Success rate from 2010 till 2020 with a few fluctuations.
- Orbit Types – GEO, ES-L1, SSO and HEO have the best success rates out of all the orbit types.
- CCAFS SLC 40 launch site - the greater the payload mass, the greater the probability of a successful launch. There seems to both failed and successful launches of low, mid and high payload masses. Therefore, there isn't a clear pattern to suggest that the success rate of a launch is dependent on the payload mass.
- Rockets with a payload between 0kg and 400kg have a higher degree of success than heavier ones
- Launch sites are usually near coasts but far from highways.
- The K-Nearest Neighbor Algorithm is the best and most accurate algorithm.

Appendix

- The library, SQlite was imported for the EDA-SQL lab.

Thank you!