IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Arnav Gupta
10 JULY 2023

Outline

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

Executive Summary

- Summary of methodologies
- Summary of all 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.

What we will be doing

We will predict if the Falcon 9 first stage will land successfully.

Section 1

Methodology

Methodology

Executive Summary

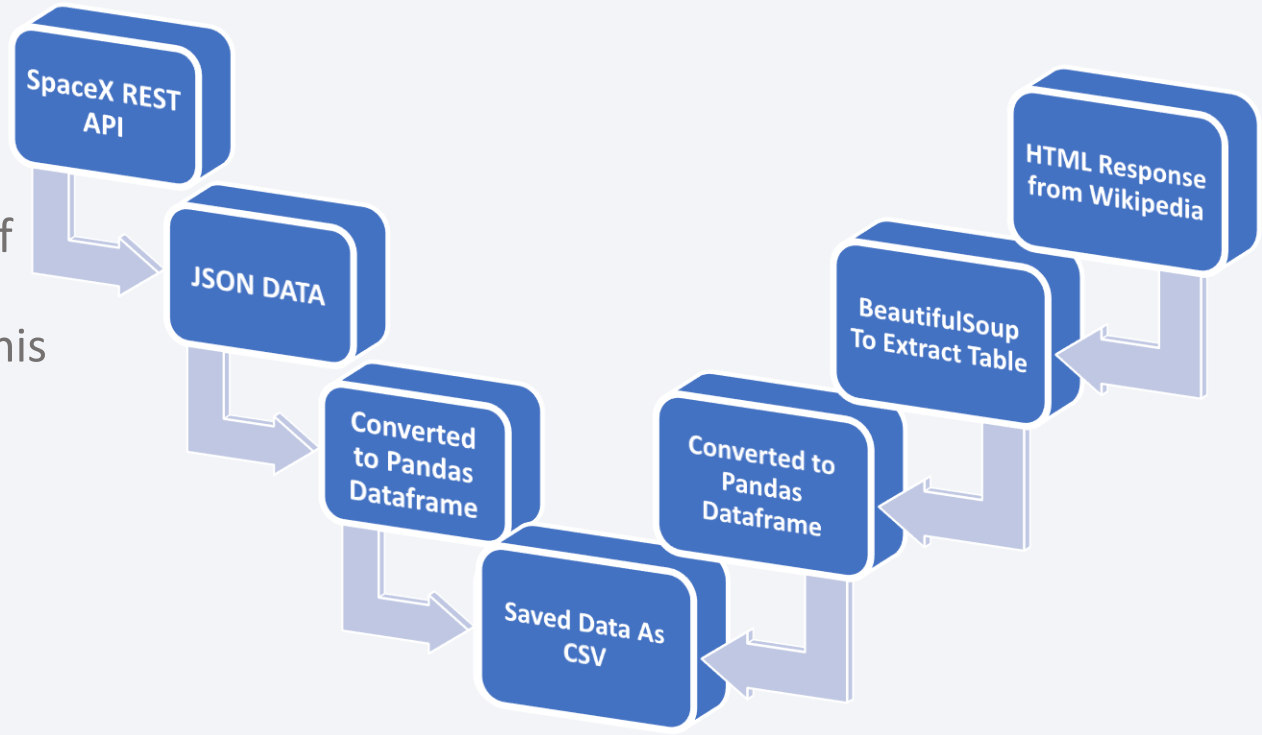
- Data collection methodology:
  - SpaceX Rest API
  - Web scraping data from Wikipedia
- Perform data wrangling
  - Prepared data for the algorithm by data cleaning of NaN values , determined columns necessary for the training model and categorized the data
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Logistic Regression, K-Nearest Neighbors, Support Vector Machines and Decision Tree models have been used and evaluated for the best accuracy

Data Collection

There are two ways of data collection:

- SpaceX REST API :

This API contains launch data which includes the model of rocket used, Orbit of Launched Rocket, Payload Mass, Landing Outcomes, Launch Site and other Launch data. This API can be accessed by accessing the URL starting with https://api.spacexdata.com/v4/ following with the data required. (For ex : /rockets or /payloads)



- Wikipedia Pages :

The Wikipedia page titled ***"List of Falcon 9 and Falcon Heavy launches"*** contains HTML tables from which we can extract Falcon 9 Launch records using the Python library ***BeautifulSoup***. This can then be stored in a Dataframe.

Data Collection – SpaceX API

1

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

response = requests.get(spacex_url)
```

Getting data from API

2

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

Using the dataframe data print the first 5 rows

```
# Get the head of the dataframe
data.head()
```

Converting data to Pandas Dataframe

3

```
BoosterVersion[0:5]
```

```
['Falcon 1', 'Falcon 1', 'Falcon
```

we can apply the rest of the function

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

Getting refined data from API using custom functions

4

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

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

Converting refined data to Pandas Dataframe

5

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = data[data['BoosterVersion']!='Falcon 1']
```

Now that we have removed some values we should reset the FlgihtNumber column

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

Filter the Dataframe to only include Falcon 9 launches

https://github.com/arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

1

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

Getting Html Response from Wikipedia

2

```
# Use BeautifulSoup() to create a BeautifulSoup obj
soup = BeautifulSoup(response.content)
```

Using BeautifulSoup Library

3

```
column_names = []

# Apply find_all() function with `th` element on f
thall = first_launch_table.find_all('th')
# Iterate each th element and apply the provided e
# Append the Non-empty column name (`if name is no
for i in thall:
    name = extract_column_from_header(i)
    if name is not None and len(name)>0:
        column_names.append(name)
```

Getting column names from First table on page

4

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

Creating Dictionary from Column names

5

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_al
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.str
                flag=flight_number.isdigit()
```

Filling Dictionary up with Data from Html Table

6

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

Converting Dictionary to Pandas Dataframe

RESULT

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

GitHub URL FOR Web scraping From Wikipedia Notebook:

https://github.com/arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

1. Checking null values present

```
df.isnull().sum()/df.count()*100
```

2. Calculating number of launches per site

```
# Apply value_counts() on column L
df.value_counts('LaunchSite')
```

3. Calculate the number and occurrence of each orbit

```
# Apply value_counts on Orbit colum
df.value_counts('Orbit')
```

4. Calculate the number and occurrence of Mission outcome per orbit type

```
# landing_outcomes = values on Outcome column
landing_outcomes = df.value_counts('Outcome')
landing_outcomes
```

5. Create a landing outcome label from Outcome column

```
temp=df['Outcome'].values
# landing_class = 0 if bad_outcome
landing_class=[]
for i in temp:
    if i in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

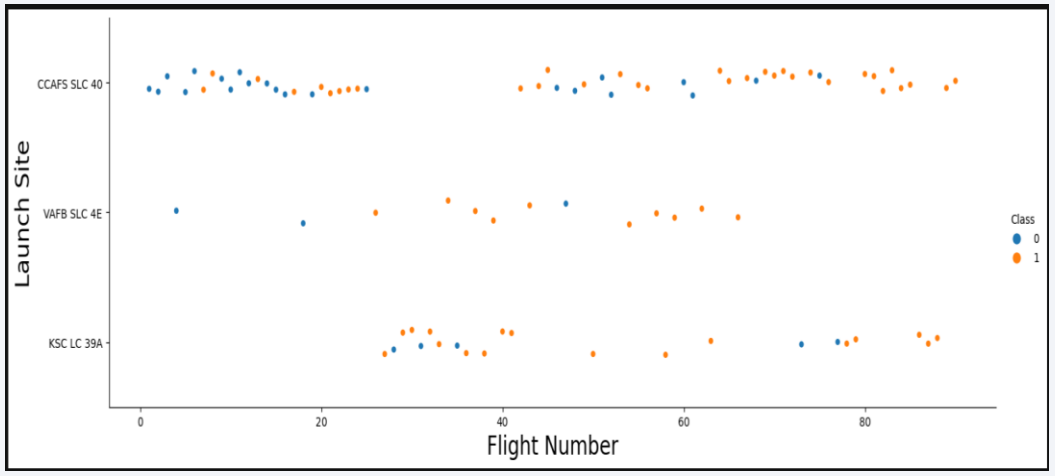
```
# Calculate the mean value of PayloadMass column
payload_df = data_falcon9[data_falcon9['PayloadMass'].notnull()]
mean = payload_df['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9.replace(np.nan,mean)
```

Removing null values(did during data collection)

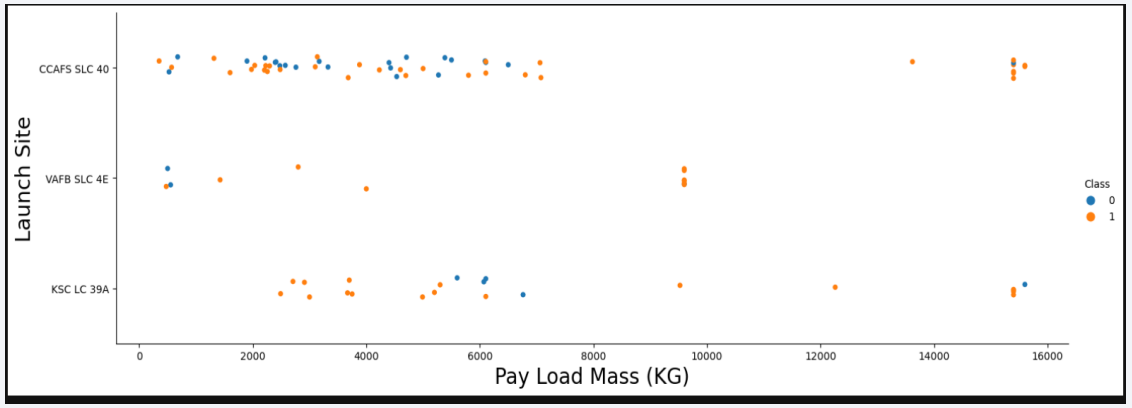
GitHub URL FOR Web scraping From Wikipedia Notebook:

https://github.com/arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

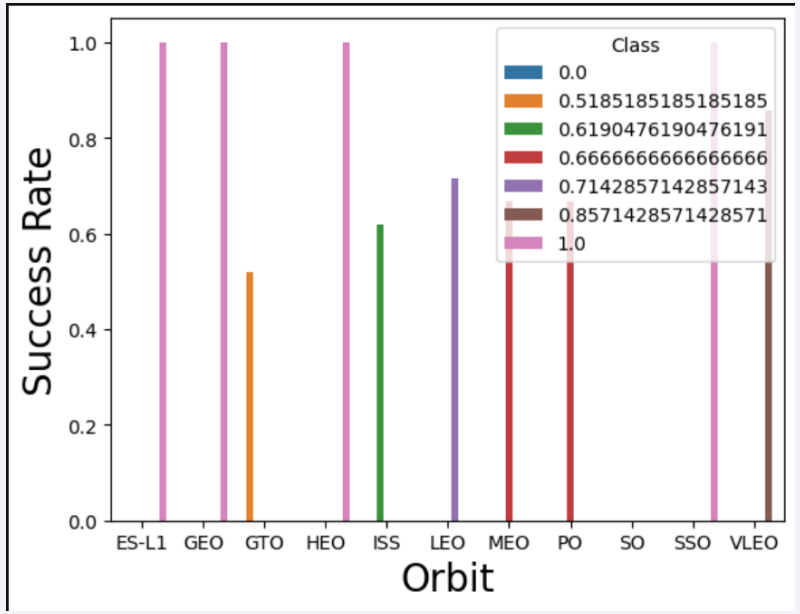


As flight number increases so does the chance of success. Earlier flights were done through CCAFS SLC 40 which were mostly failures

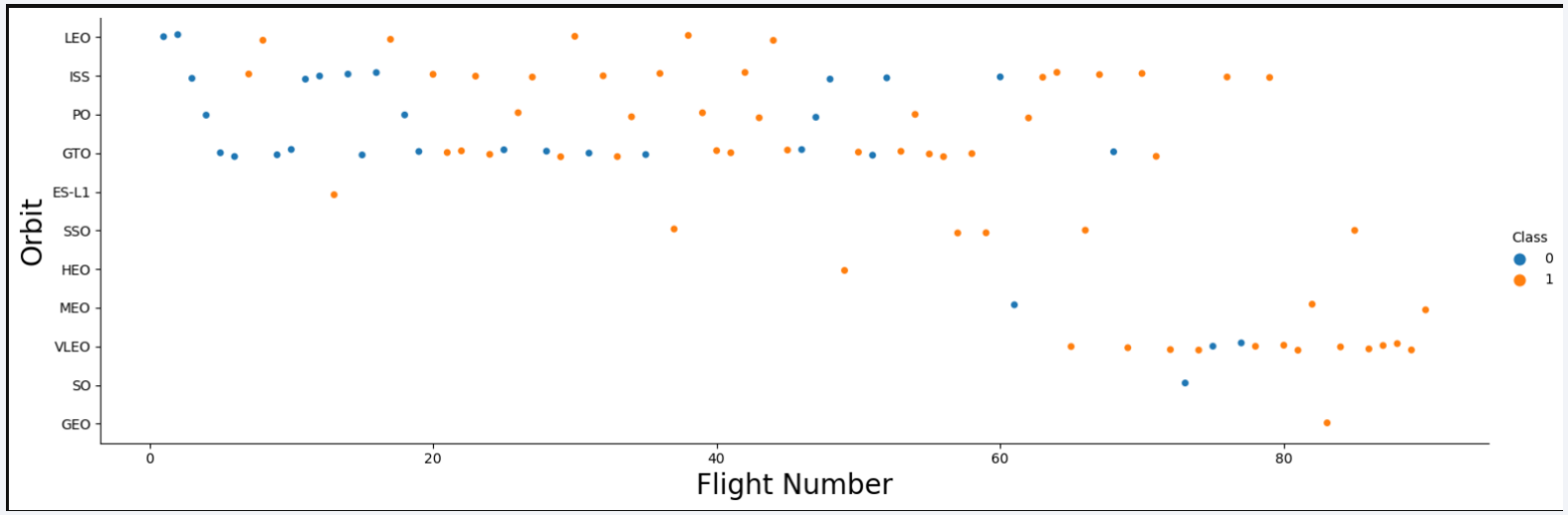


VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).

EDA with Data Visualization

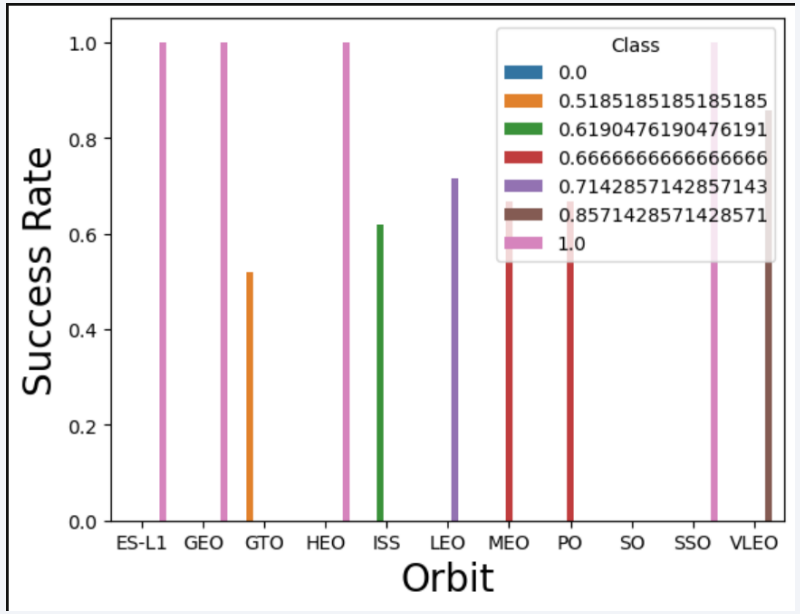


ES-L1 GEO and SSO orbits have a 100% success rate whereas SO AND SSO orbits have a 0% success rate . VLEO also has a relatively high success rate whereas the rest have 50-70% success rate

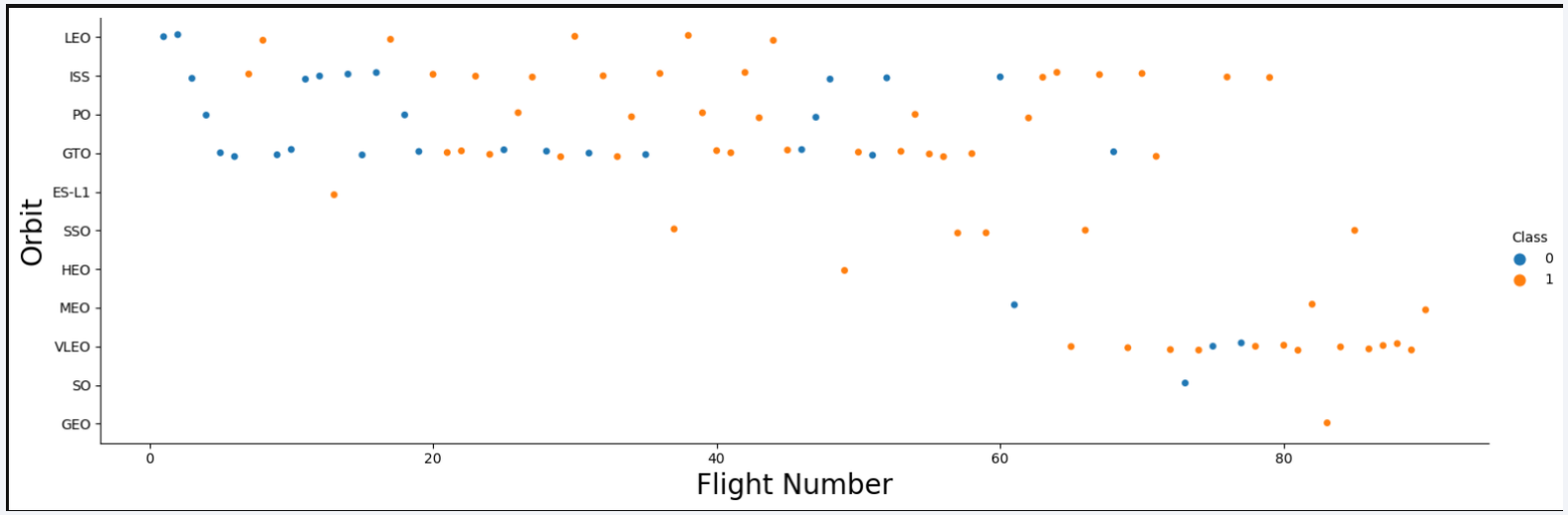


In the VLEO and ISS orbit the success rate increases with the increase in flight number, whereas in the GTO orbit there seems to be no relation in success rate with number of flights. SSO orbit seems to have all launches as success. LEO orbit has been since stopped being used(around flight number 45)

EDA with Data Visualization



ES-L1 GEO and SSO orbits have a 100% success rate whereas SO AND SSO orbits have a 0% success rate . VLEO also has a relatively high success rate whereas the rest have 50-70% success rate



In the VLEO and ISS orbit the success rate increases with the increase in flight number, whereas in the GTO orbit there seems to be no relation in success rate with number of flights. SSO orbit seems to have all launches as success. LEO orbit has been since stopped being used(around flight number 45)

EDA with SQL

- SQL queries performed
  - 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 succesful 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. Use a subquery
  - Listing the records which will display the month names, succesful 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 04-06-2010 and 20-03-2017 in descending order.

GitHub URL for the SQL notebook :

IBM-SpaceX-Prediction-With-Machine-Learning/jupyter-labs-eda-sql-edx_sqllite.ipynb at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub

Build an Interactive Map with Folium

- Following map objects were created and added to the Folium map
  - Marked all launch sites on a map which allowed to visualize the launch sites on the map
    - Circles (folium.Circle) for each launch site with a text label over each launch (folium.Marker)
  - MarkerCluster() was added to add Green marker for successful lunch and Red for un-successful
  - Calculated the distances between a launch site to its proximities (eg coastline , railroad, higway, cities etc)
    - Used MousePosition and calculated the distance between the coastline point and the launch site.
    - Created a folium.Marker() to show the distance
    - Drew a PolyLine between a launch site to the selected coastline point

GitHub URL for the completed interactive map with Folium map:
IBM-SpaceX-Prediction-With-Machine-Learning/lab_jupyter_launch_site_location.ipynb at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub

Build a Dashboard with Plotly Dash

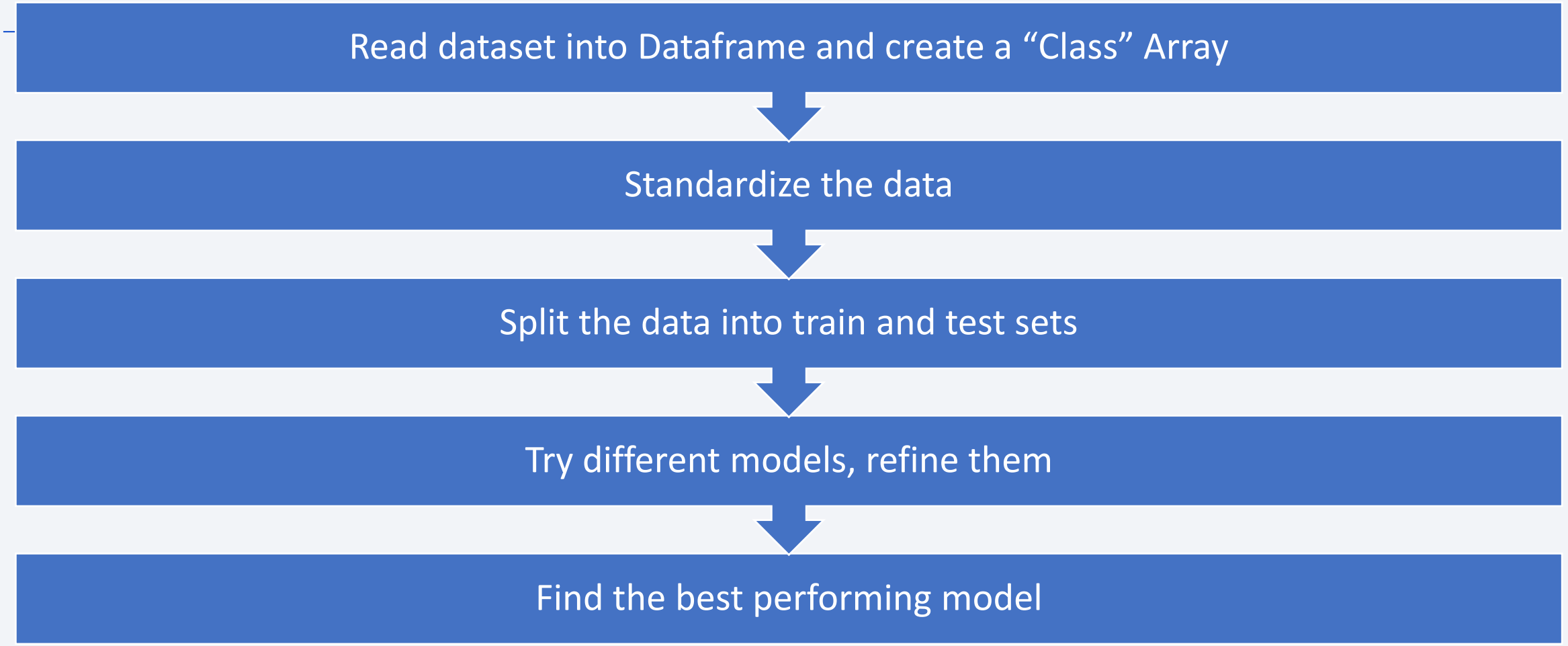
- Built a Plotly Dash web application to perform interactive visual analytics on SpaceX launch data in real-time. Added Launch Site Drop-down, Pie Chart, Payload range slide, and a Scatter chart to the Dashboard.
  - Added a Drop-down list to provide an ability to filter Dashboard visual by all launch sites or a particular launch site
  - Added a Pie Chart to the Dashboard to show total success launches when 'All Sites' is selected and show success and failed counts when a particular site is selected
  - Added a slider to select payload range to identify visual patterns.
  - Added a Scatter chart to observe the relation between mission outcome and payload. The color-label Booster version on each scatter point provided missions outcomes with different boosters.
- Dashboard helped answer following questions:
  1. Which site has the largest successful launches? KSC LC-39A with 10
  2. Which site has the highest launch success rate? KSC LC-39A with 76.9% success
  3. Which payload range(s) has the highest launch success rate? 2000 – 5000 kg
  4. Which payload range(s) has the lowest launch success rate? 0-2000 and 5500 - 7000
  5. Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest launch success rate? FT

GitHub URL for the application - IBM-SpaceX-Prediction-With-Machine-Learning/spacex_dash_app.py at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub

GitHub URL for the dashboards –

- IBM-SpaceX-Prediction-With-Machine-Learning/Screenshot (87).png at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub
- IBM-SpaceX-Prediction-With-Machine-Learning/Screenshot (88).png at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub

Predictive Analysis (Classification)



GitHub URL for the the analysis

IBM-SpaceX-Prediction-With-Machine-Learning/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub

Predictive Analysis (Classification)

1. Read dataset into Dataframe and create a “Class” Array
   A. Load the dataframe

```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_2.csv")
```

   B. Create a NumPy array from the column Class in data

```
Y = data['Class'].to_numpy()
```

2. Standardize the data in X and then re-assign to data in X using the transform below

```
scaler = preprocessing.StandardScaler().fit(X)
X=scaler.transform(X)
```

3. Split data in test and train set

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2,random_state=2)
```

GitHub URL for the the analysis

IBM-SpaceX-Prediction-With-Machine-Learning/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub

Predictive Analysis (Classification)

4. Try different models based on different algorithms and refine them. Below is the example for Logistic Regression

   A. Create a logistic regression object then create a GridSearchCV object

   B. Fit the object to find the best parameters from the dictionary parameters

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()

gv = GridSearchCV(lr,parameters,scoring='accuracy',cv=10)
logreg_cv = gv.fit(X_train,Y_train)
```

   C. Display the best parameters using the data attribute best_params_ and the accuracy on the validation data using the data attribute best_score_

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

   D. Calculate the accuracy on the test data

```
logreg_cv.score(X_test,Y_test)
```

   E. Repeat the steps for models – SVM, Decision Tree and KNN

GitHub URL for the the analysis

IBM-SpaceX-Prediction-With-Machine-Learning/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub

Predictive Analysis (Classification)

5. Find the method that performs best

In [129…

```
df1 = pd.DataFrame({'Model':['Logistic Regression' , 'Vector Machine', 'Descision Tree', 'KNN'], 'Accuracy Score' : acc_lis
df1
```

Out[129…

	Model	Accuracy Score	Test Data Accuracy
0	Logistic Regression	0.846429	0.833333
1	Vector Machine	0.848214	0.833333
2	Descision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333

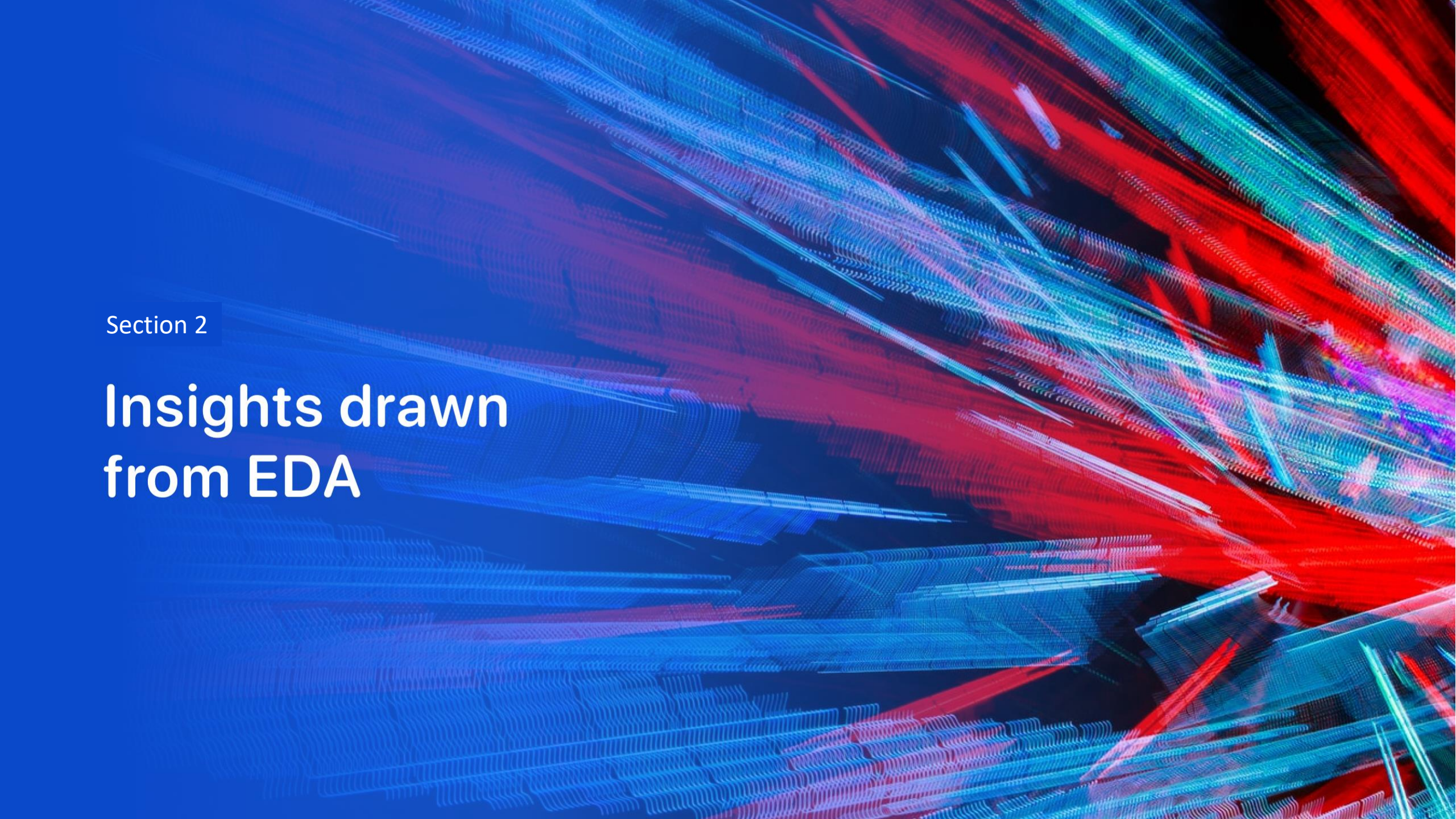
Based on accuracy score calculated above, Decision Tree performs best

GitHub URL for the analysis

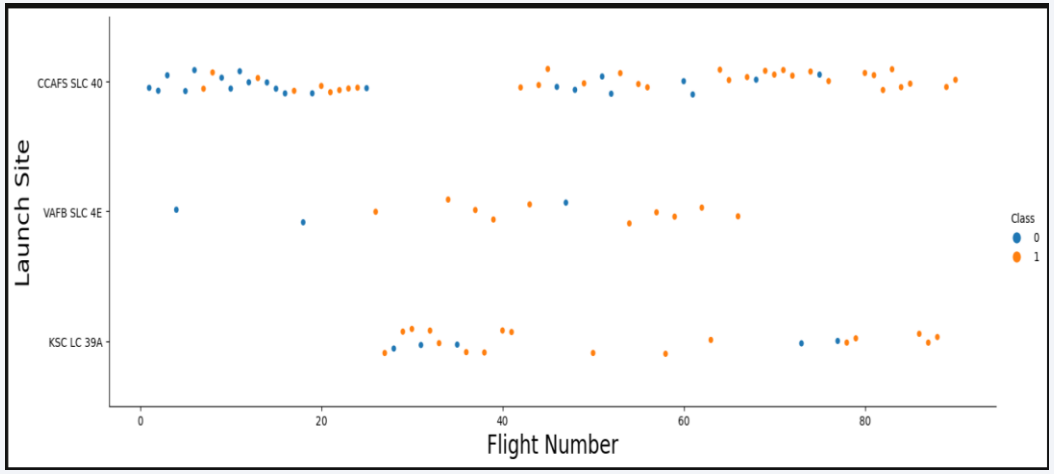
IBM-SpaceX-Prediction-With-Machine-Learning/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · arnav0103/IBM-SpaceX-Prediction-With-Machine-Learning · GitHub

Results

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

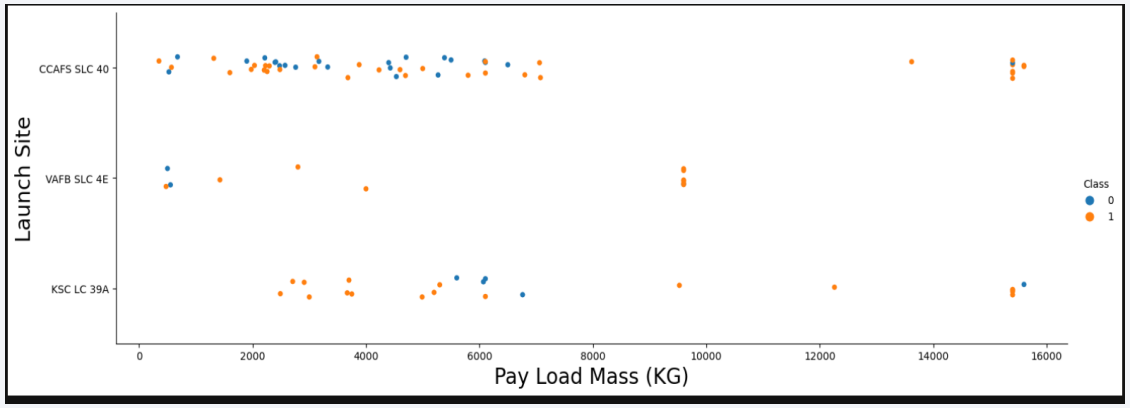

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



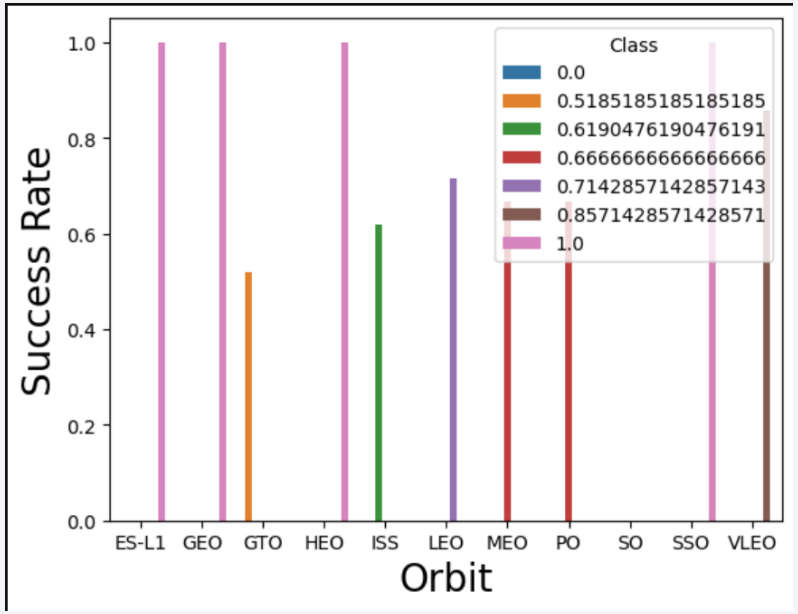
- As number of flights increases so does the chance of success. (Class = 1).
- Earlier flights were done through CCAFS SLC 40 which were mostly failures
- For launch site ‘KSC LC 39A’, it takes at least around 25 launches before a first successful launch

Payload vs. Launch Site



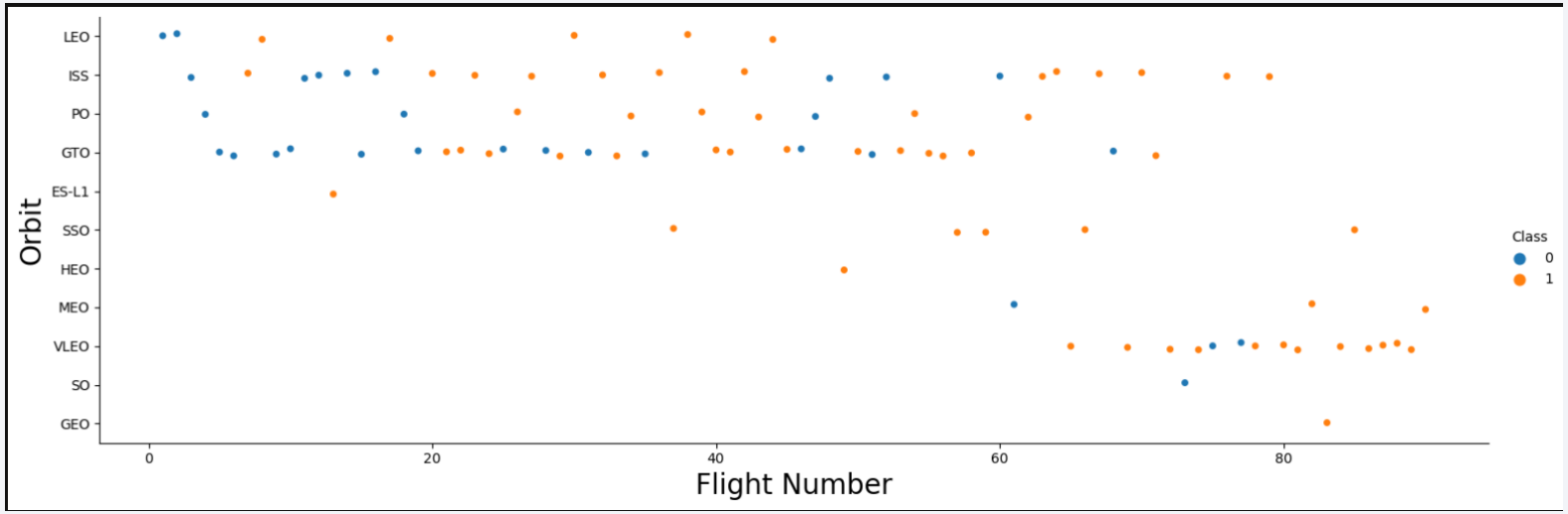
- For VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).
- Percentage of successful launch (Class=1) increases for launch site ‘VAFB SLC 4E’ as the payload mass increases
- There is no clear correlation or pattern between launch site and payload mas

Success Rate vs. Orbit Type



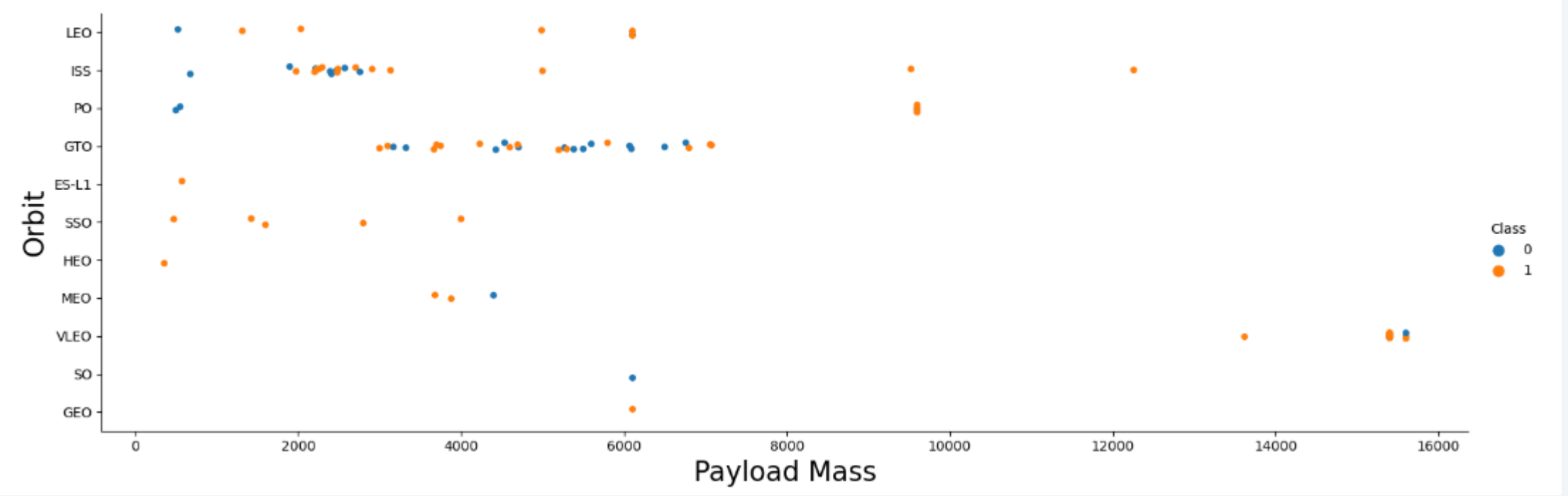
- ES-L1 GEO and SSO orbits have a 100% success rate whereas SO AND SSO orbits have a 0% success rate . VLEO also has a relatively high success rate whereas the rest have 50-70% success rate

Flight Number vs. Orbit Type



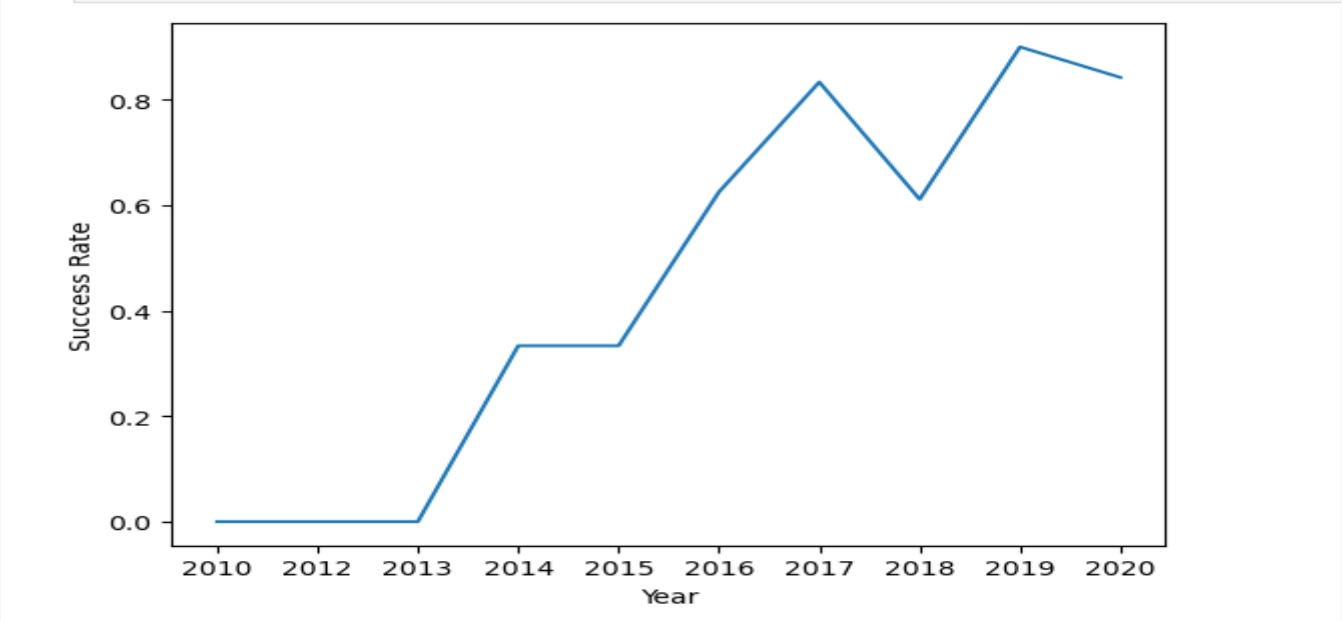
In the VLEO and ISS orbit the success rate increases with the increase in flight number, whereas in the GTO orbit there seems to be no relation in success rate with number of flights. SSO orbit seems to have all launches as success. LEO orbit has been since stopped being used(around flight number 45)

Payload vs. Orbit Type



- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch Success Yearly Trend



- We can observe that the success rate since 2013 kept increasing till 2020
- Success rate (Class=1) increased by about 80% between 2013 and 2020
- Success rates remained the same between 2010 and 2013 and between 2014 and 2015
- Success rates decreased between 2017 and 2018 and between 2019 and 2020

All Launch Site Names

- Query

```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL
```

- DISTINCT would return only unique values
- There are 4 launch sites
- Result

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

Launch Site Names Begin with 'KSC'

- Query

```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'KSC%' LIMIT 5
```

- Using keyword ‘Like’ and format ‘KCS%’, returns records where ‘Launch_Site’ column starts with “KCS”
- Limit 5, limits the number of returned records to 5
- Result :

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
19-02-2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
16-03-2017	06:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	No attempt
30-03-2017	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
01-05-2017	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
15-05-2017	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	No attempt

Total Payload Mass

- Calculate the total payload carried by boosters from NASA (CRS)

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)'
```

- ‘sum’ adds column ‘PAYLOAD_MASS_KG’ and returns total payload mass for customers named ‘NASA (CRS)’
- Result

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1'
```

- ‘AVG’ keyword returns the average of payload mass in ‘PAYLOAD_MASS_KG’ column where booster version is ‘F9 v1.1

- Result

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on drone ship.Present your query result with a short explanation here

```
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE LANDING_OUTCOME='Success (drone ship)'
```

- The keyword “MIN” helps get the earliest/first date with landing outcome SUCCESS for drone ship.
- Result

MIN(DATE)
06-05-2016

Successful Ground pad Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on ground pad and had payload mass greater than 4000 but less than 6000

```
%sql SELECT DISTINCT(BOOSTER_VERSION) FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (ground pad)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```

- The ‘and’ operator in the where clause returns booster versions where both conditions in the where clause are true i.e the landing is on ground pad and payload mass is greater than 4000 but less than 6000
- Result

Booster_Version
F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Query

```
%sql SELECT COUNT(MISSION_OUTCOME) AS SUCCESS FROM SPACEXTBL WHERE MISSION_OUTCOME='Success'
```

```
%sql SELECT COUNT(MISSION_OUTCOME) AS FAILURE FROM SPACEXTBL WHERE MISSION_OUTCOME!='Success'
```

- Result

SUCCESS
98

FAILURE
3

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

```
%%sql
SELECT DISTINCT(BOOSTER_VERSION)
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

- The sub query returns the maximum payload mass by using keywork ‘max’ on the pay load mass column
- The main query returns booster versions where payload mass is maximum with value of 15600 (returned by sub-query)
- Result

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

2017 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**

```
%%sql
SELECT SUBSTR(DATE,4,2) AS MONTHNAMES, BOOSTER_VERSION, LAUNCH_SITE
FROM SPACEXTBL
WHERE LANDING_OUTCOME='Success (ground pad)' AND SUBSTR(DATE,7,4)='2017'
```

- **Query uses substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2017' for year.**

- Result

MONTHNAMES	Booster_Version	Launch_Site
02	F9 FT B1031.1	KSC LC-39A
05	F9 FT B1032.1	KSC LC-39A
06	F9 FT B1035.1	KSC LC-39A
08	F9 B4 B1039.1	KSC LC-39A
09	F9 B4 B1040.1	KSC LC-39A
12	F9 FT B1035.2	CCAFS SLC-40

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

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

```
%%sql
SELECT LANDING_OUTCOME , COUNT(LANDING_OUTCOME) AS AMOUNT
FROM SPACEXTBL
WHERE DATE BETWEEN '04-06-2010' AND '20-03-2017'
AND LANDING_OUTCOME LIKE 'Success %'
GROUP BY LANDING_OUTCOME
ORDER BY AMOUNT DESC
```

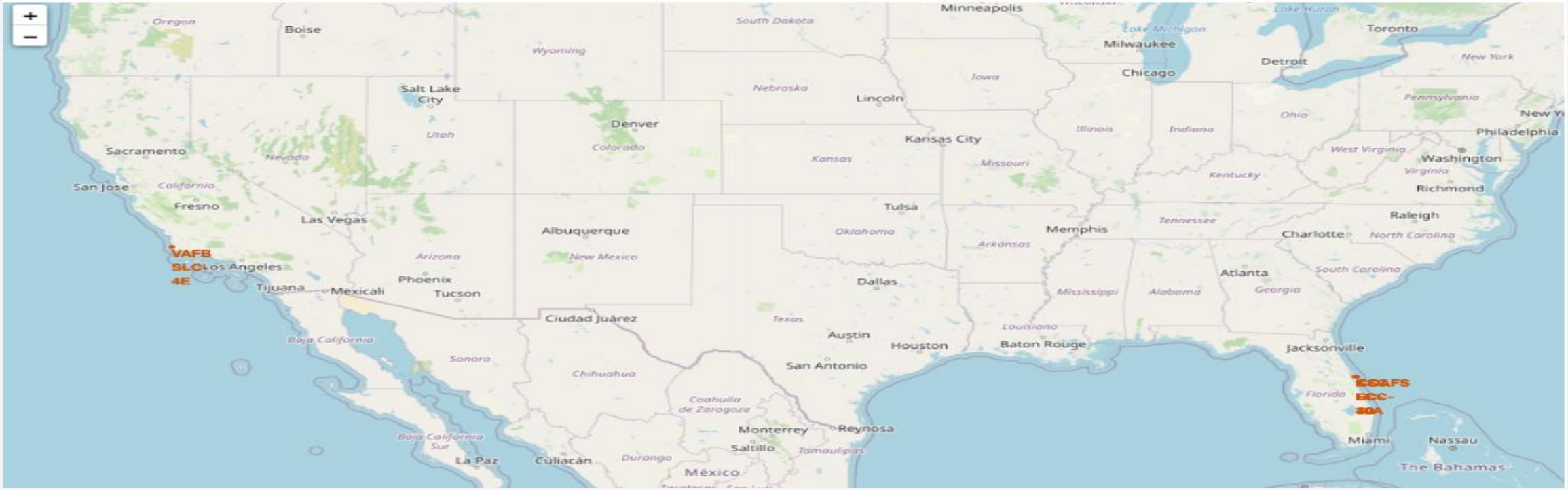
- The ‘group by’ key word arranges data in column ‘Landing__Outcome’ into groups •
- The ‘between’ and ‘and’ keywords return data that is between 2010-06-04 and 2017-03-20
- The ‘order by’ keyword arranges the counts column in descending order •
- The result of the query is a ranked list of landing outcome counts per the specified date range
- **Result**

Landing_Outcome	AMOUNT
Success (drone ship)	8
Success (ground pad)	6

Section 3

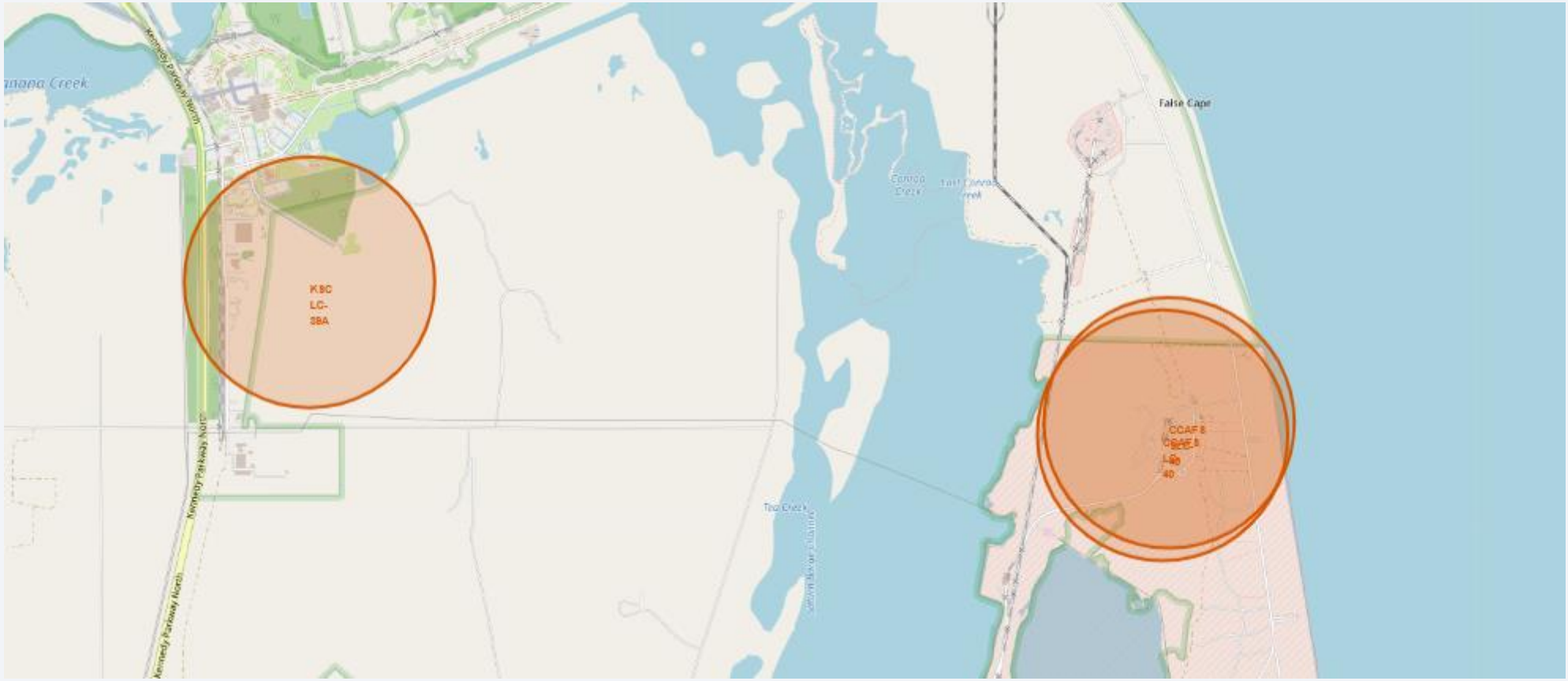
Launch Sites Proximities Analysis

All launch sites on the map



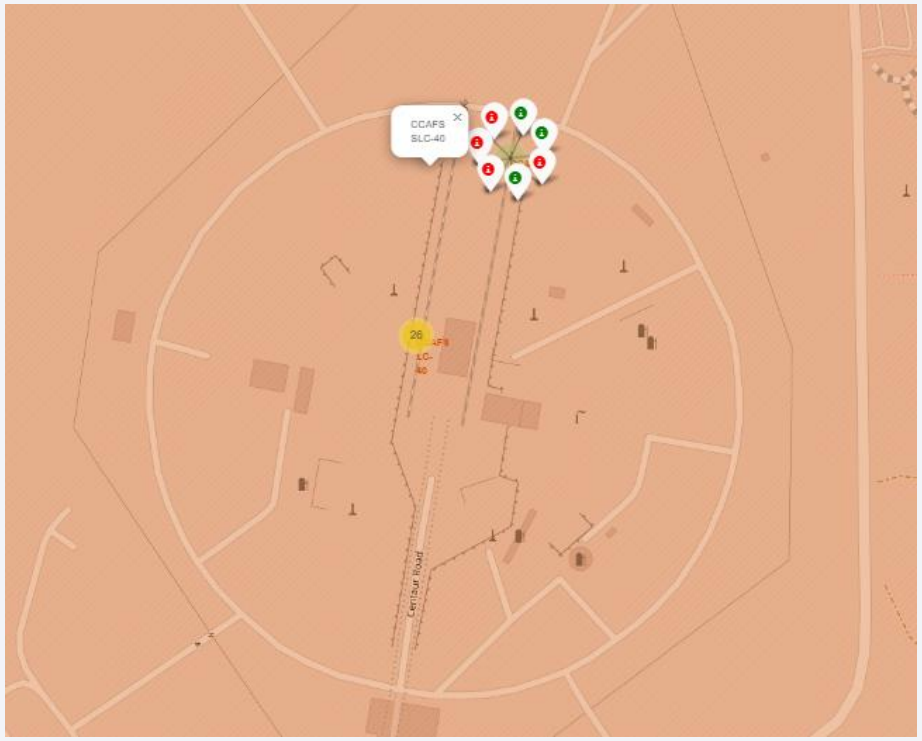
- SpaceX launch sites are in United States of America coasts – Florida and Los Angeles

Florida’s launch sites on the map



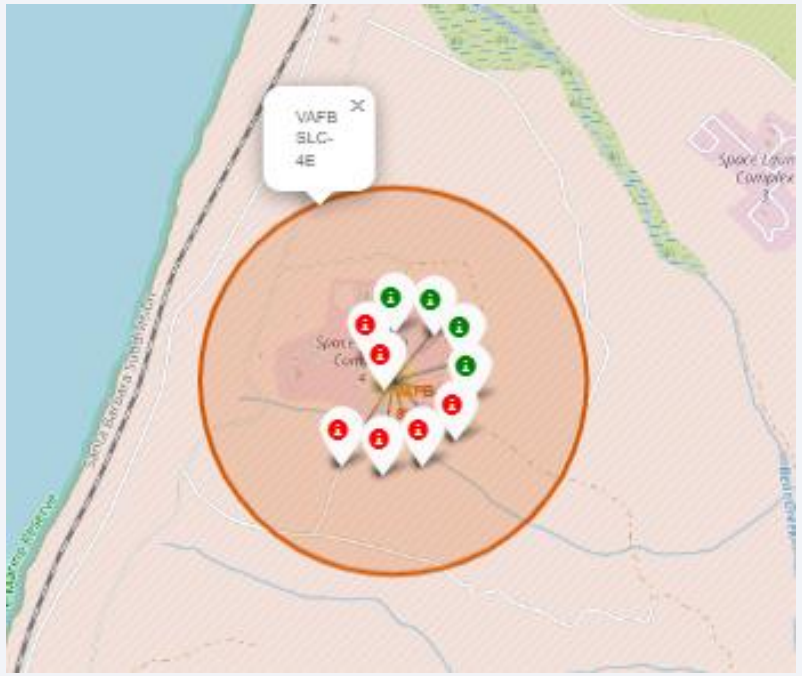
- SpaceX launch sites (zoomed) in United States of America coasts Florida -> KSCLC-39A and CCAFS SLC-40

Color labeled launch records on map



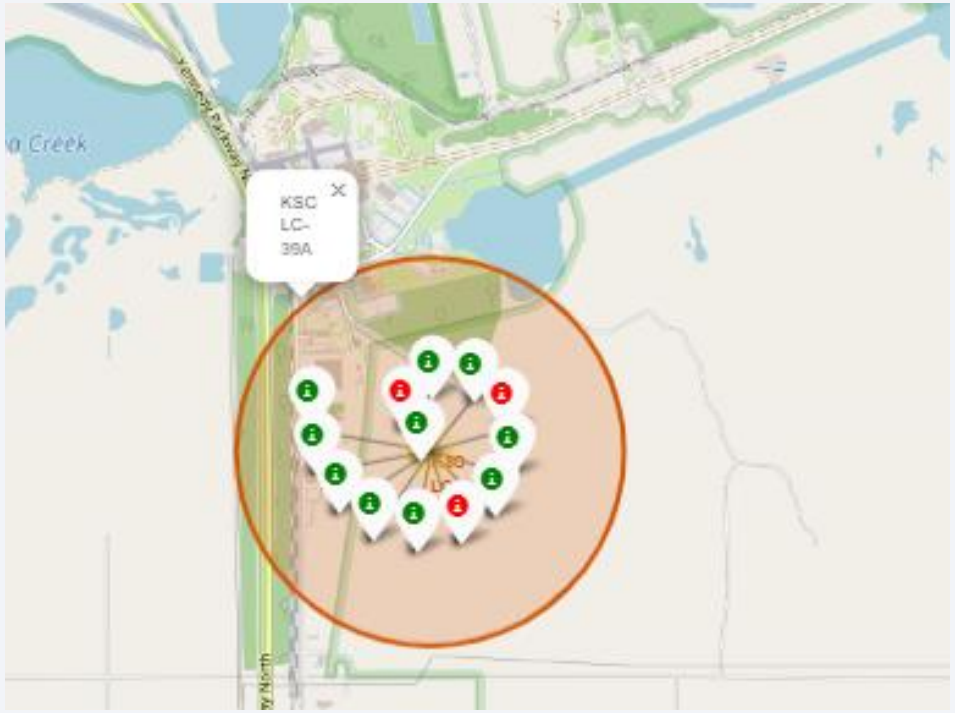
- Green Marker – Successful Launch and Red Marker -> Failed Launch
- The map above us of Launch Site CCAFS LC-40 and CCAFS LC-40

Color labeled launch records on map



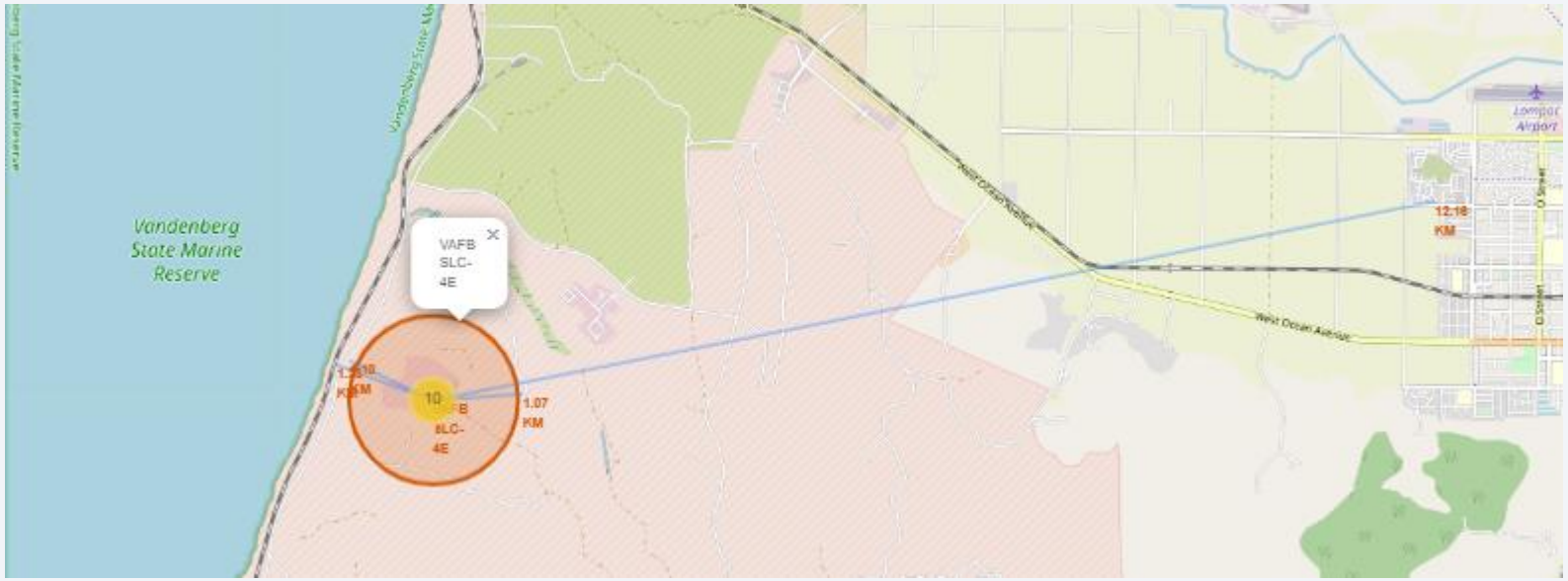
- Green Marker – Successful Launch and Red Marker -> Failed Launch
- The map above us of Launch Site VSFB-SLC-4E (10 launches)

Color labeled launch records on map



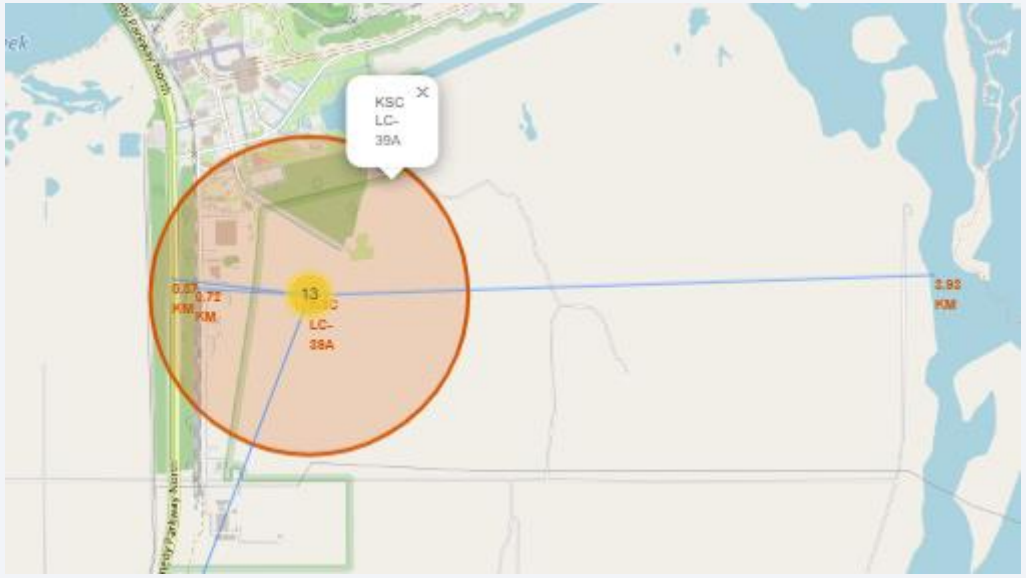
- Green Marker – Successful Launch and Red Marker -> Failed Launch
- The map above us of Launch Site KSCLC-39A (13 launches)

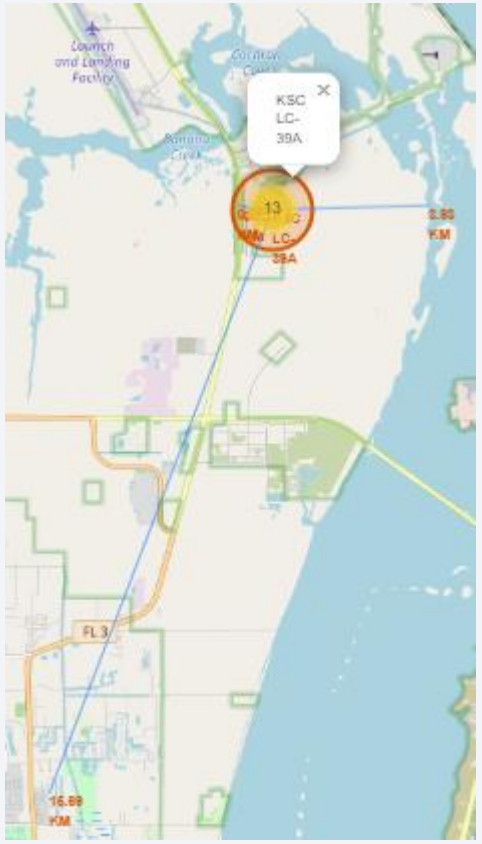
Launch site VAFS SLC-4E proximity



- Distance to railways - 1.23 KM
- Distance to highways – 1.07 KM
- Distance to coastline – 1.10 KM
- Distance to city – 12.18 KM

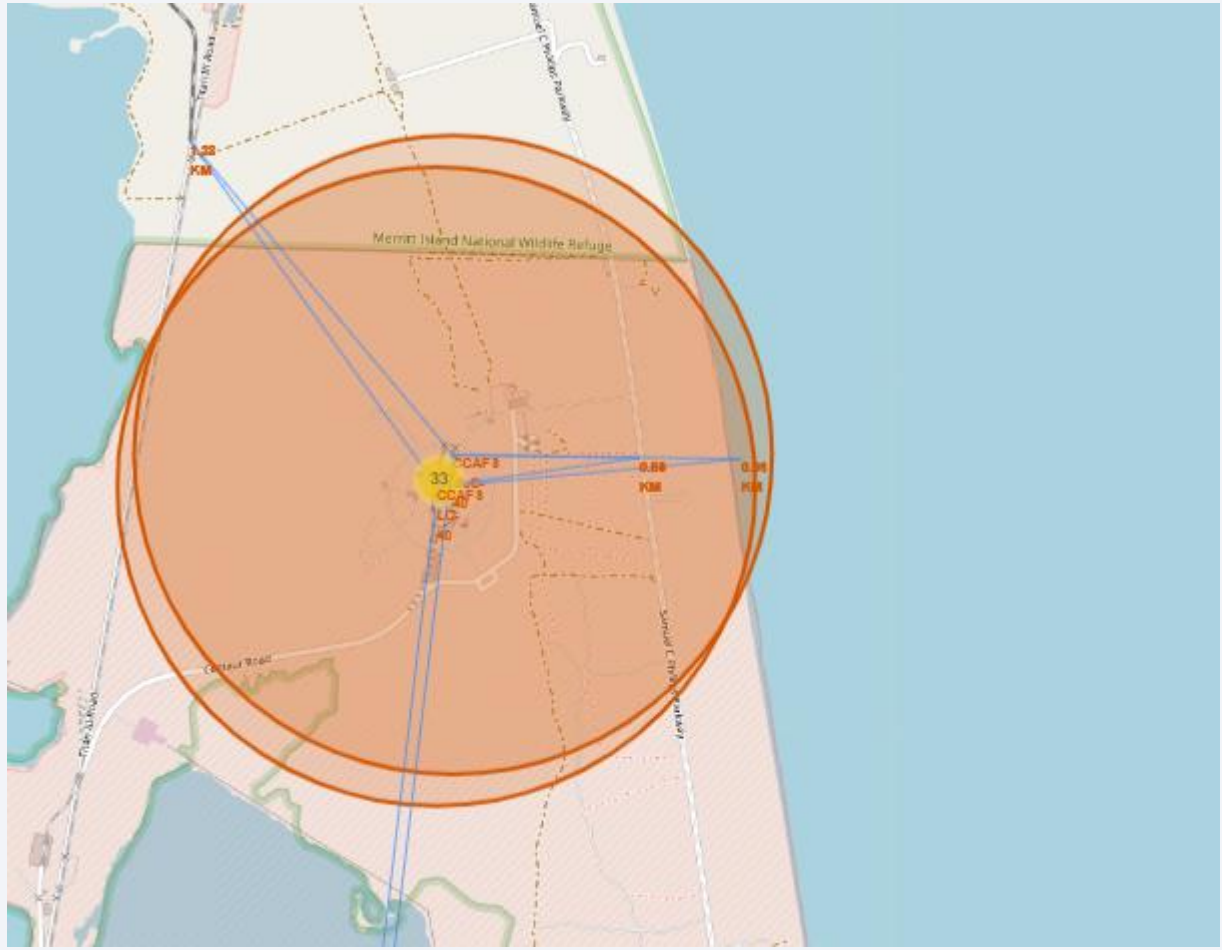
Launch site KSCLC-39A proximity





- Distance to railways - 0.72 KM
- Distance to highways – 0.87 KM
- Distance to coastline – 3.83 KM
- Distance to city – 16.89 KM

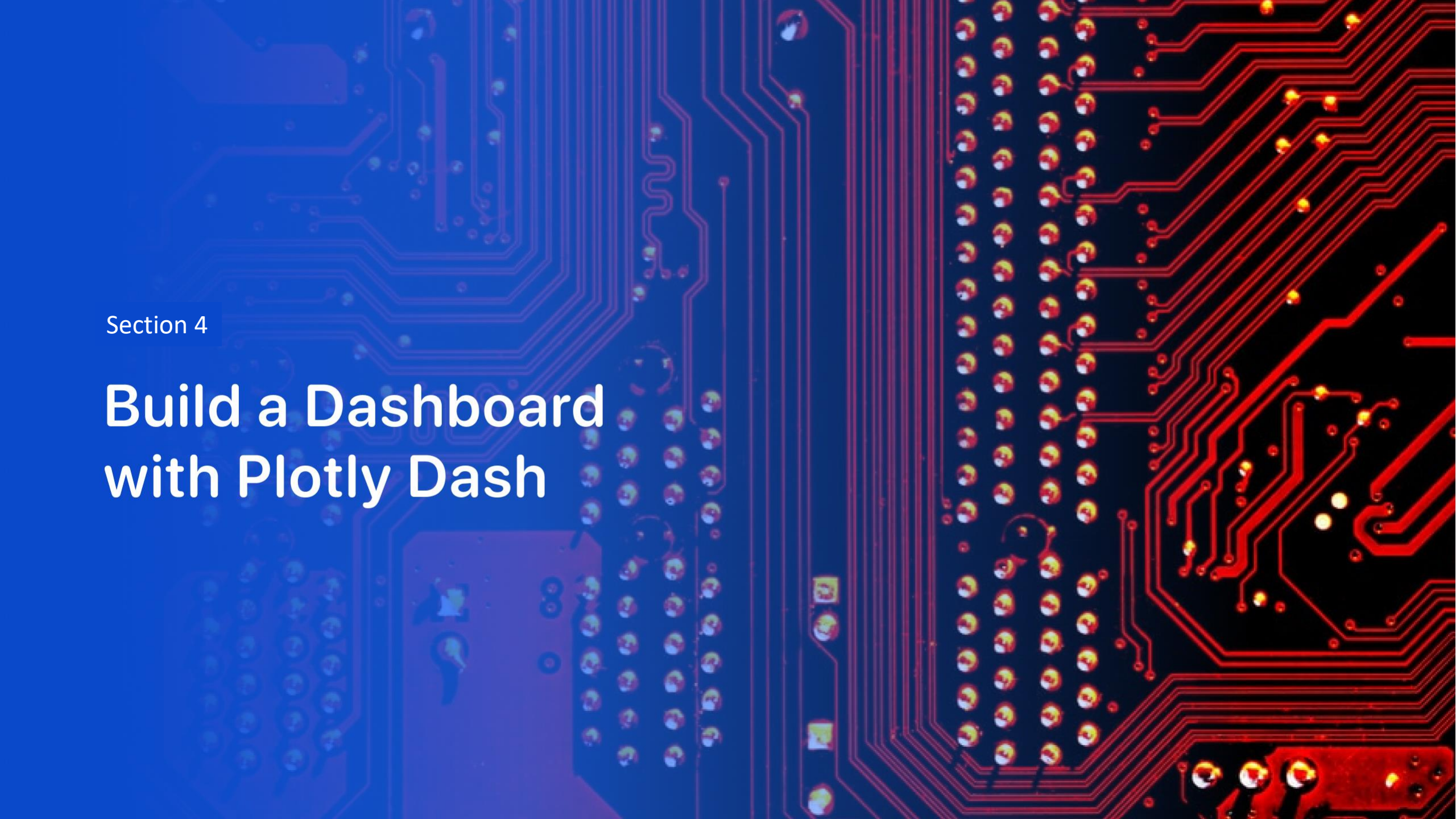
Launch site VAFS SLC-4E proximity



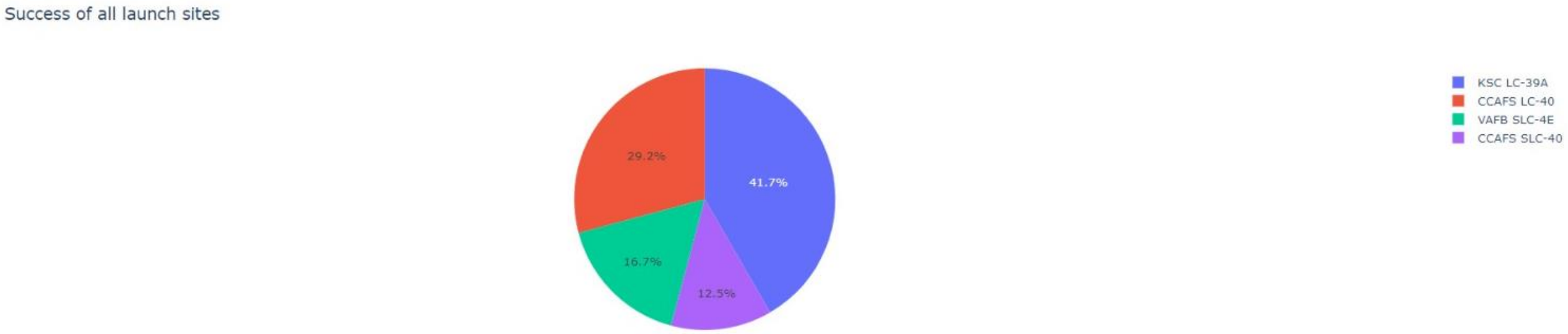
- Distance to railways 1.38 KM
- Distance to highways – 0.68 KM
- Distance to coastline – 0.89 KM
- Distance to city – 17.49 KM

Proximity Questions

- Are launch sites in close proximity to railways? YES
- Are launch sites in close proximity to highways? YES
- Are launch sites in close proximity to coastline? YES
- Do launch sites keep certain distance away from cities? YES (More than 12 KM in the mapped cases)

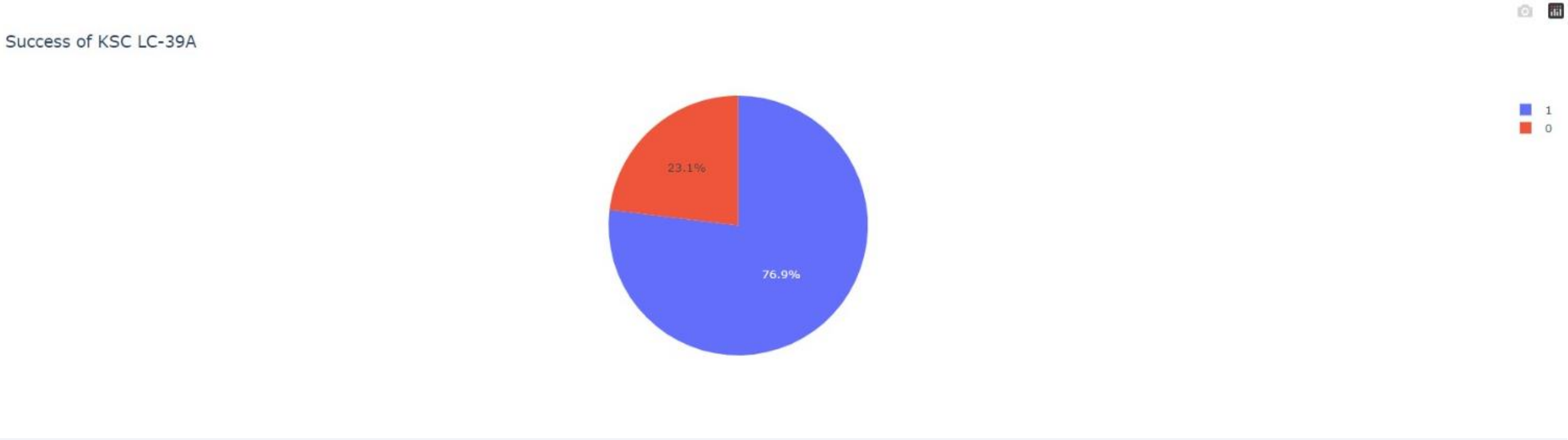

Section 4
Build a Dashboard with Plotly Dash

Launch success count for all sites



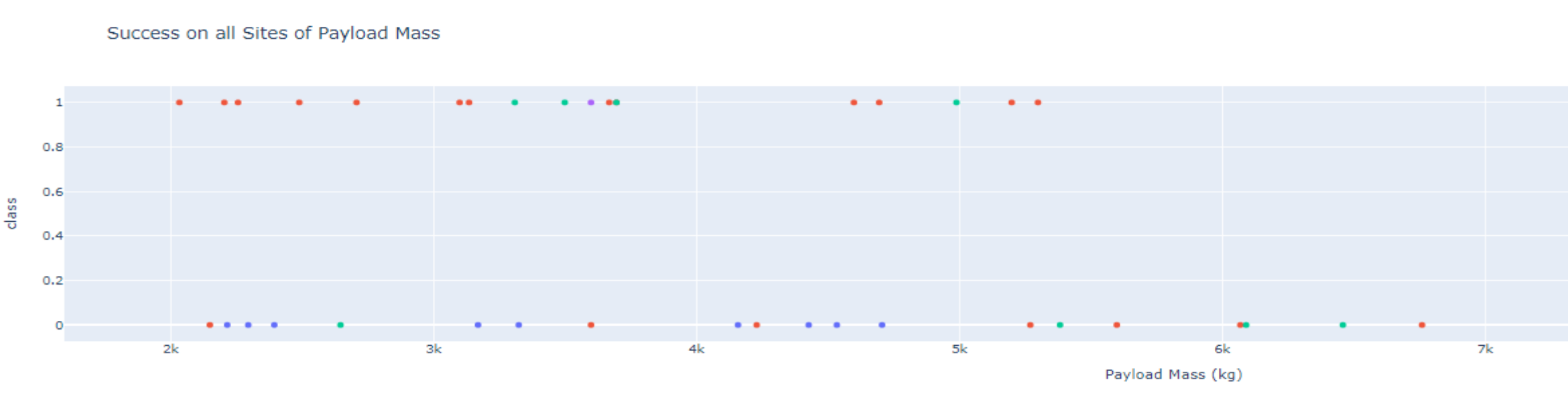
- Out of all sites, KSC LC-39A has the most successful launches

Success rate of KSC LC-39A



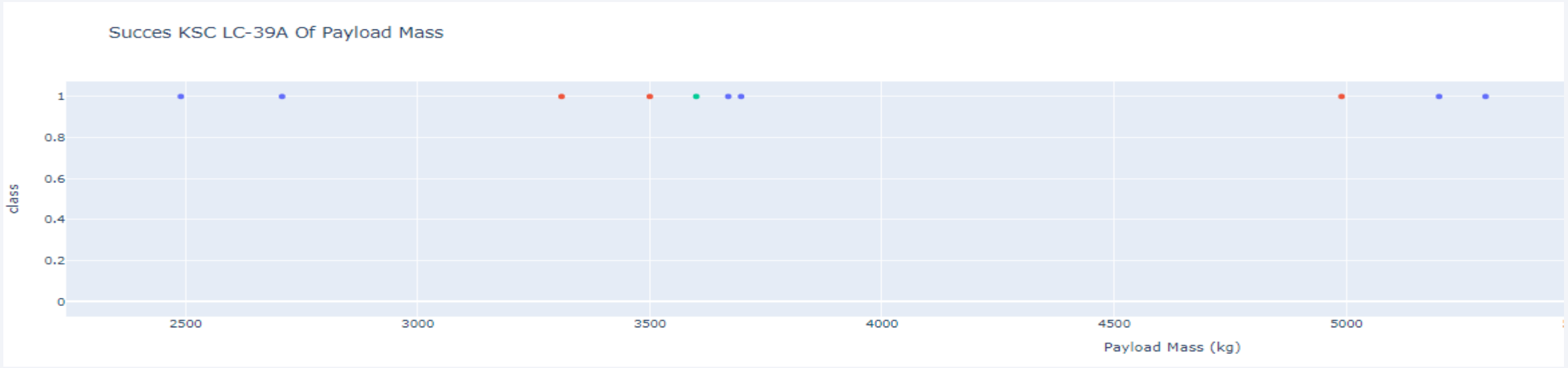
- KSC LC-39A has highest launch success rate with 76.9% success

Scatter plot of Payload vs launch outcome for all sites



- Payloads between 2000 and 5500 Kg have the highest success rate

Scatter plot of Payload vs launch outcome for KSC LC-39A



- Payload between 2500 and 1750 KG have highest success rate

Section 5

Predictive Analysis (Classification)

Classification Accuracy

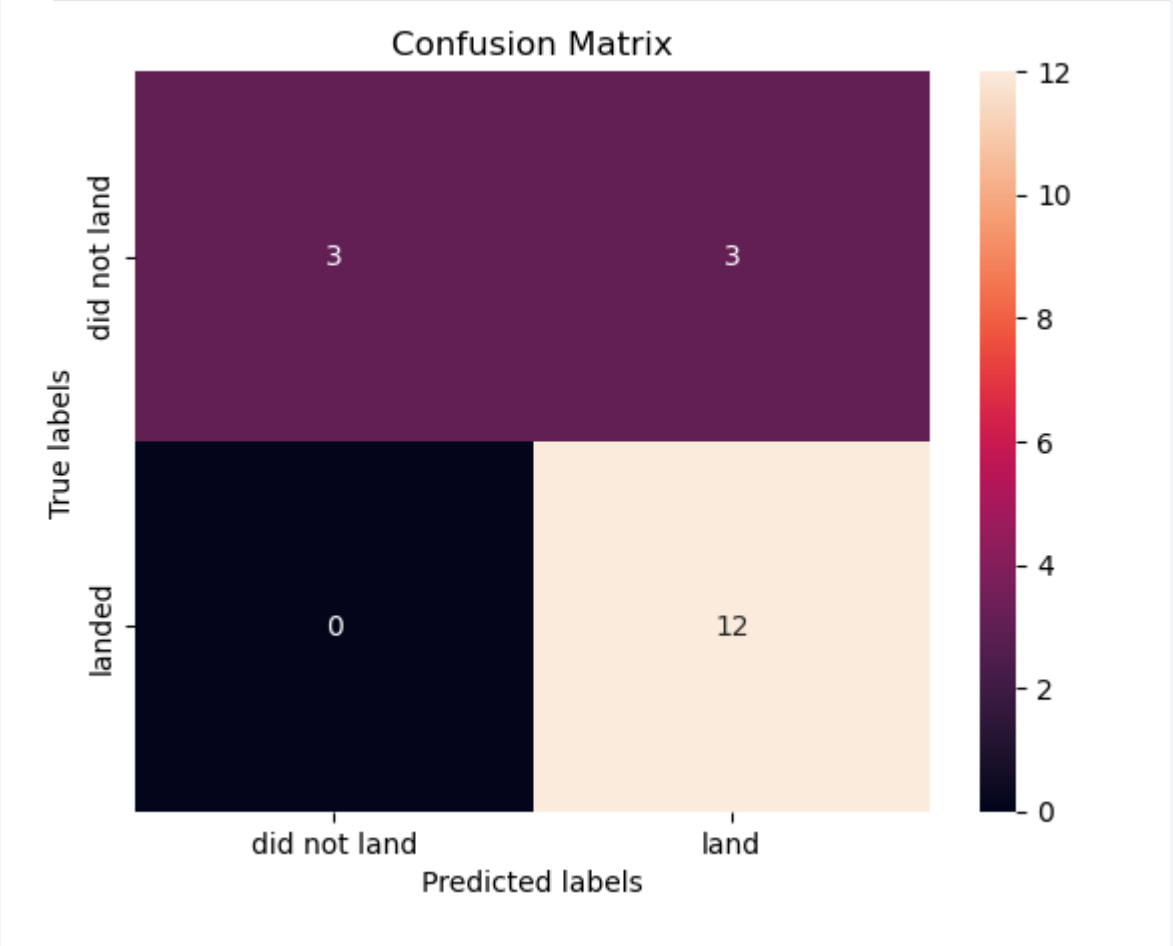
- Based on scores of test set we can't confirm which is the best model.
- Based on accuracy score, Decision Tree model performs best

	Model	Accuracy Score	Test Data Accuracy
0	Logistic Regression	0.846429	0.833333
1	Vector Machine	0.848214	0.833333
2	Descision Tree	0.875000	0.833333
3	KNN	0.848214	0.833333

Confusion Matrix

- The “model” can distinguish between different classes.
- The main issue is the “False Positives” where “Did Not land” is marked as “successful landed” by the classifier.

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP



Conclusions

- Decision Tree is the best model for this dataset
- Launches with low payload mass (~2000 and 5500 kg) show better results than launches with larger payload mass
- KSC LC-39A has the highest success rate of the launches from all sites
- Orbits ES-L1 GEO and SSO orbits have a 100% success rate whereas SO AND SSO orbits have a 0% success rate
- Success Rate of launches has increased over years

Thank you!

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project