Space Race Champion Strategy with Data Science

Daniel Ade Aryono
18th September 2022

Outline

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

Executive Summary

Summary of methodologies

- Data Collection through API
- Data Collection with Web Scraping
- Data Wrangling
- Exploratory Data Analysis with SQL
- Exploratory Data Analysis with Data Visualization
- Interactive Visual Analytics with Folium
- Machine Learning Prediction

Summary of all results

- Exploratory Data Analysis result
- Interactive analytics in screenshots
- Predictive Analytics result

Introduction

Project background and context

Space X 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 Space X 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 space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

Problems you want to find answers

- What factors determine if the rocket will land successfully?
- The interaction amongst various features that determine the success rate of a successful landing.
- What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Data Collection :

Data was collected using SpaceX API and web scraping from Wikipedia.

Data Wrangling :

One-hot encoding was applied to categorical features.

Exploratory data analysis (EDA) :

Perform Exploratory data analysis (EDA) using visualization and SQL.

Data Visualization :

Perform Data Visualization analytics using Folium and Plotly Dash.

Model Development :

Perform predictive analysis using classification models.

Report :

Reporting results to stakeholders.

Data Collection

The historical launch data was collected from Open Source REST API for SpaceX

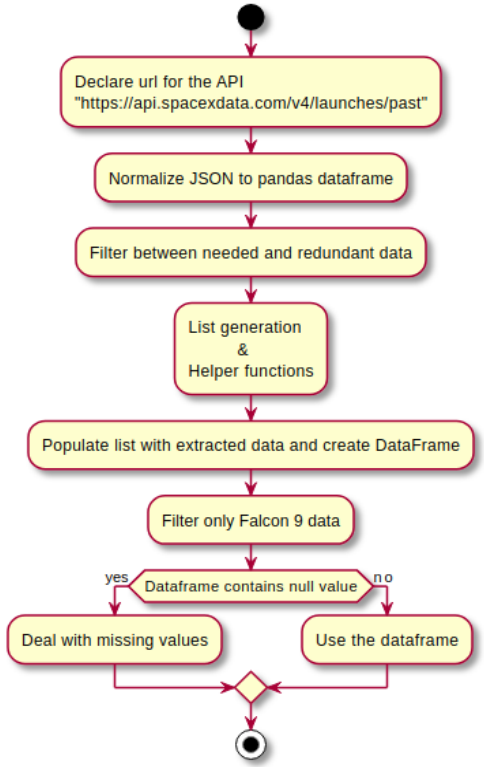
- Requested and parsed the SpaceX launch data using the GET request
- Data decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize()
- Replaced missing payload mass values from classified missions with mean
- Data cleaned and checked for missing values and fill in missing values where necessary
- Performed the web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.

Data Collection – SpaceX API

- Requested and parsed the SpaceX launch data using the GET request
- Filtered the dataframe to only include Falcon 9 launches
- Replaced missing payload mass values from classified missions with mean

The link to the notebook is

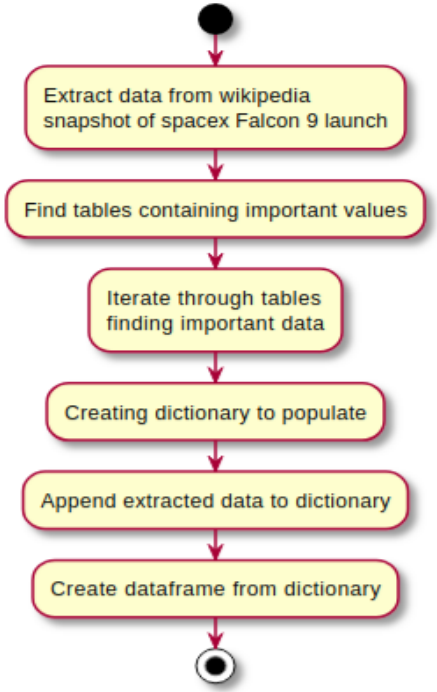
https://github.com/danielaryono/ibmcapstone-datascience/blob/main/Data-Collection.ipynb



Data Collection - Scrapping

- Applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- Parsed the table and converted it into a pandas dataframe.

The link to the notebook is
https://github.com/danielaryono/ibmcapstone-datascience/blob/main/Webscraping.ipynb



Data Wrangling

- Performed exploratory data analysis and determined the training labels.
- Calculated the number of launches on each site
- Calculated the number and occurrence of each orbit
- Calculated the number and occurrence of mission outcome per orbit type
- Created a landing outcome training label from 'Outcome' column and exported the results to csv.

```
In [6]:
1 # Apply value_counts on Orbit column
2 df['Orbit'].value_counts()

Out[6]: GTO     27
        ISS     21
        VLEO    14
        PO       9
        LEO      7
        SSO      5
        MEO      3
        SO       1
        ES-L1    1
        HEO      1
        GEO      1
        Name: Orbit, dtype: int64
```

The link to the notebook is
https://github.com/danielaryono/ibmcapstone-datascience/blob/main/Data-Wrangling.ipynb

Exploratory Data Analysis (EDA) with SQL

Load the SpaceX dataset into Jupyter Notebook using a PostgreSQL database.

- We applied EDA with SQL to get insight from the data.
- Ran SQL queries to display and list information about
  - Launch sites
  - Payload masses
  - Booster versions
  - Mission outcomes
  - Booster landings

In [19]:

```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

The link to the notebook is
https://github.com/danielaryono/ibmcapstone-datascience/blob/main/EDA-with-SQL.ipynb

EDA with Data Visualization

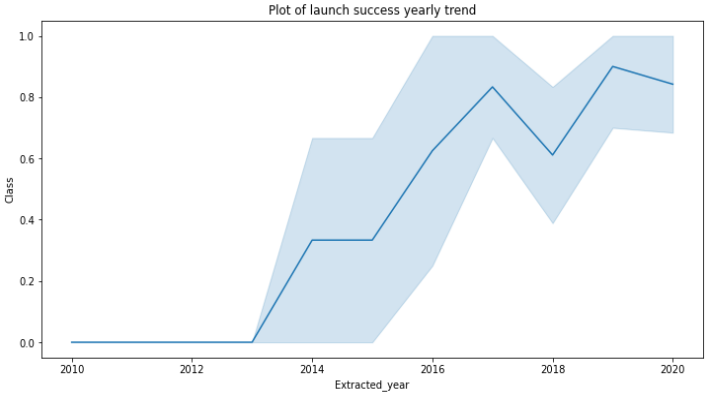
Read the dataset into a Pandas dataframe

Used Matplotlib and Seaborn visualization libraries to plot

- Flight Number vs Payload Mass
- Flight Number vs Launch Site
- Payload vs Launch Site
- Orbit type vs Success rate
- Flight Number vs Orbit type
- Payload vs Orbit type
- Year vs Success rate

The link to the notebook is

https://github.com/danielaryono/ibmcapstone-datascience/blob/main/EDA-with-Data-Visualization.ipynb

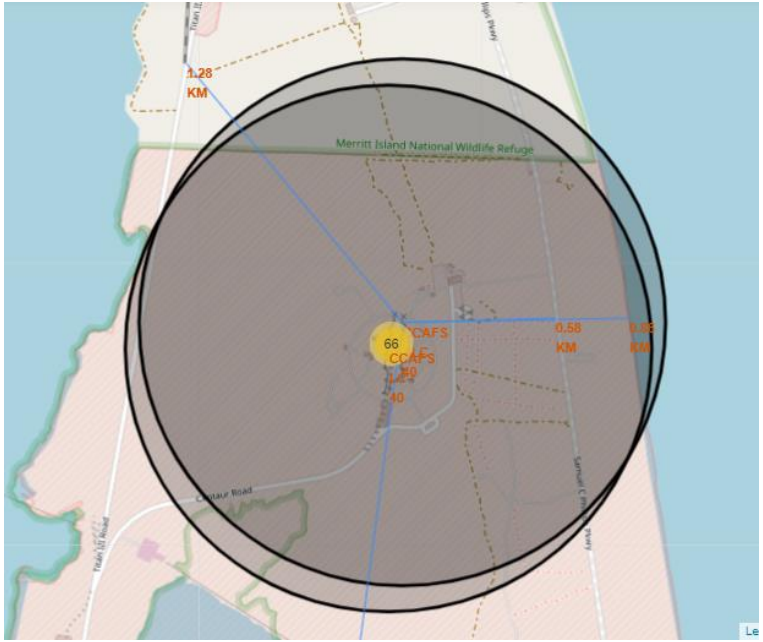


Interactive Map with Folium

Used Python interactive mapping library called Folium for Launch Sites Location Analysis

- Marked all launch sites on a map
- Marked the successful/failed launches for each site on map
- Calculated the distances between a launch site to its proximities
    - Railways
    - Highways
    - Coastlines
    - Cities

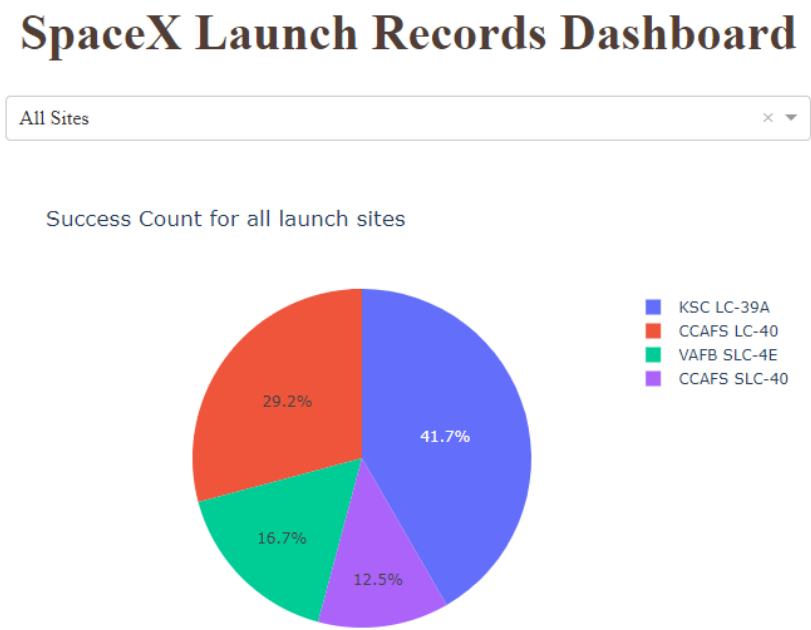
The link to the notebook is
https://github.com/danielaryono/ibmcapstone-datascience/blob/main/Data-Visualization-with-Folium.ipynb



Dashboard with Plotly Dash

Used Python interactive dashboard library called Plotly Dash to explore and manipulate data in an interactive and real-time way

- Pie chart showing success rate
- Color coded by launch site
- Scatter chart showing payload mass vs. landing outcome
- Color coded by booster version
- With range slider for limiting payload amount
- Drop-down menu to choose between all sites and individual launch sites

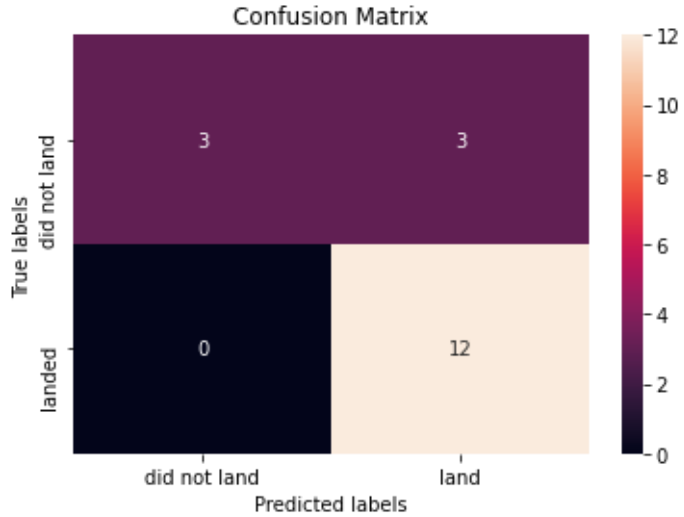


The link to the notebook is

https://github.com/danielaryono/ibmcapstone-datascience/blob/main/Data-Visualization-with-Plotly.ipynb

Predictive Analysis (Classification)

- Import libraries and defined function to create confusion matrix
- Loaded the data frame created during data collection
- Created a column for our training label 'Class' created during data wrangling
- Standardized and Split the data into training data and test data
- Fit the training data to various model types
  - Logistic Regression
  - Support Vector Machine
  - Decision Tree Classifier
  - K Nearest Neighbors Classifier
- Evaluated accuracy of each model using test data to select the best classification model.



The link to the notebook is
https://github.com/danielaryono/ibmcapstone-datascience/blob/main/Machine-Learning-Prediction.ipynb

Results

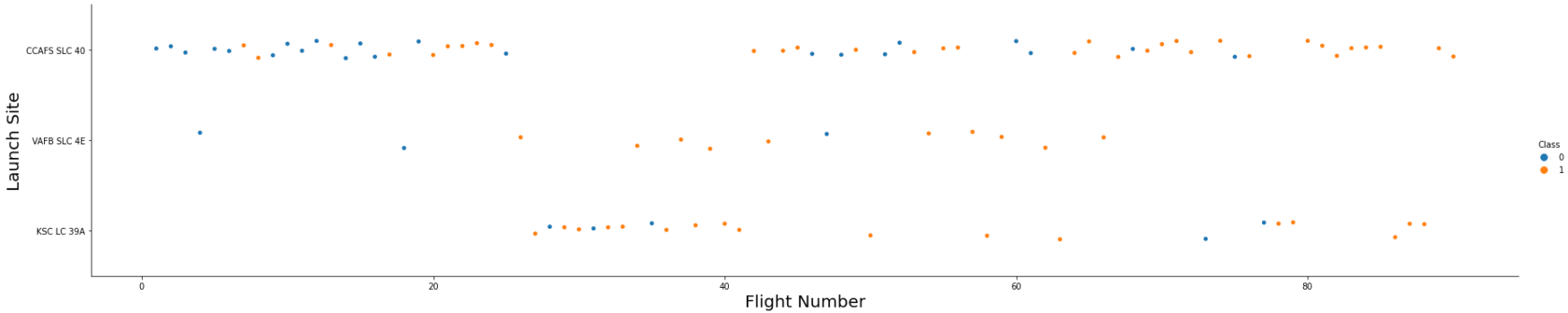
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

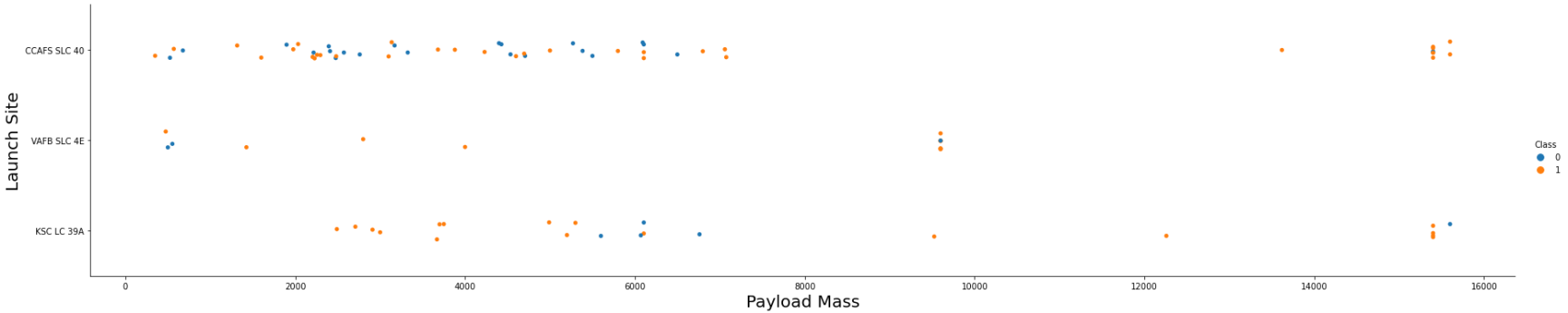
Flight Number vs. Launch Site

From the plot, Flight Number vs Launch Site, CCAFS SLC 40 appears to have been where most of the early 1st stage landing failures took place



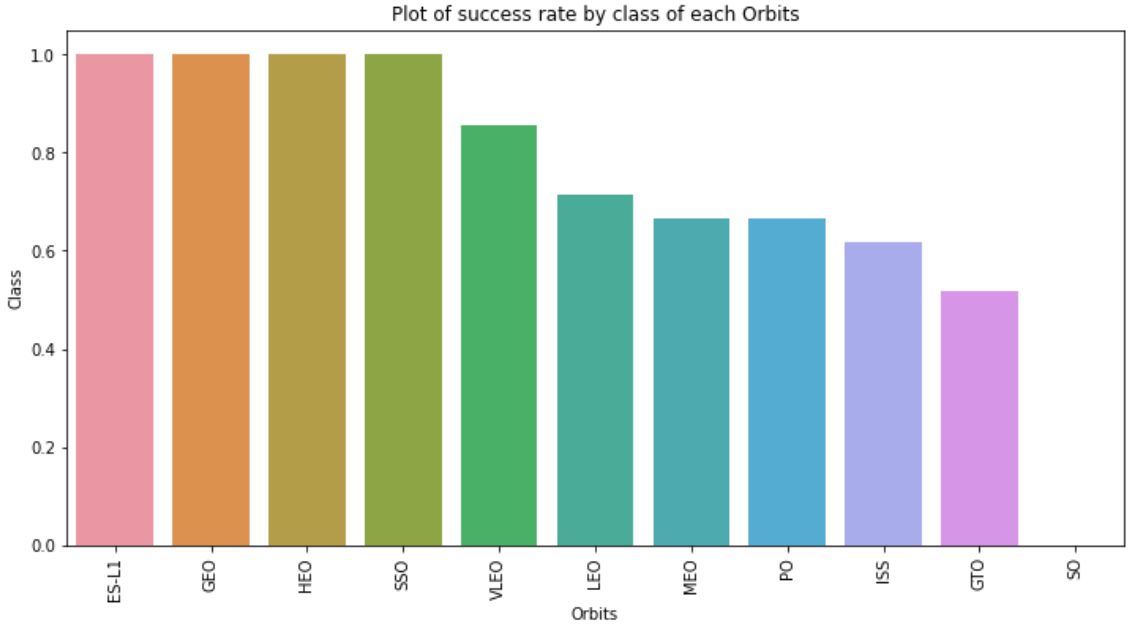
Payload Mass vs. Launch Site

CCAFS SLC 40 and KSC LC 39A appear to be favored for heavier payloads



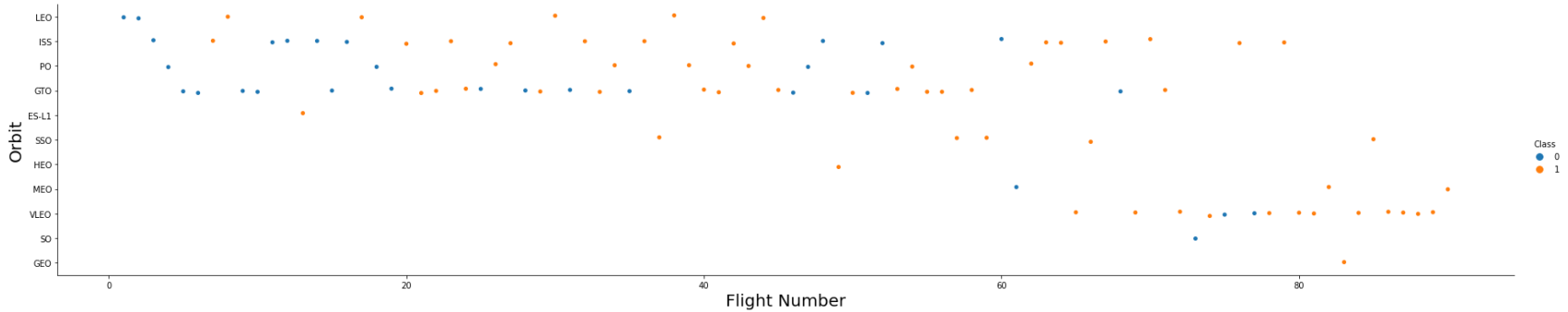
Success Rate vs. Orbit Type

ES-L1, GEO, HEO, SSO, VLEO Orbit type had the most success rate.



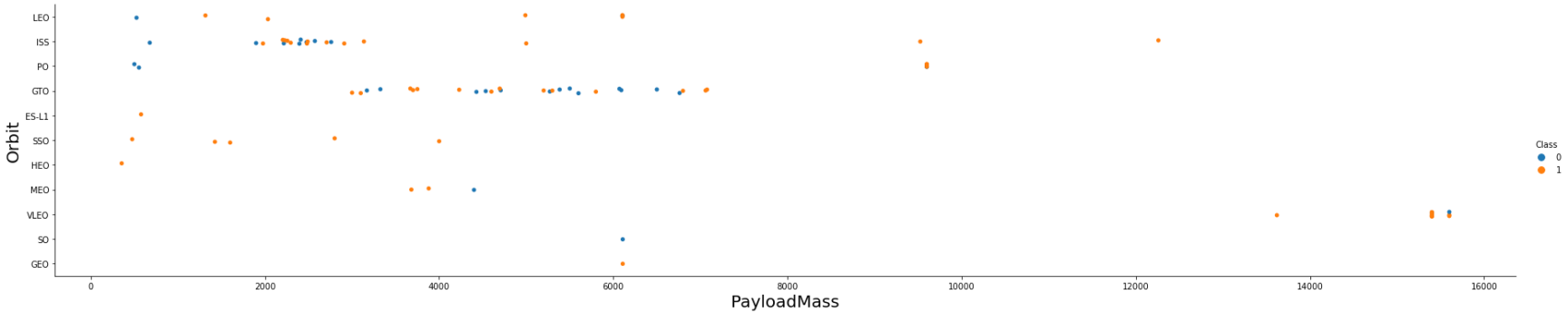
Flight Number vs. Orbit Type

LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no correlation between flight number and the orbit.



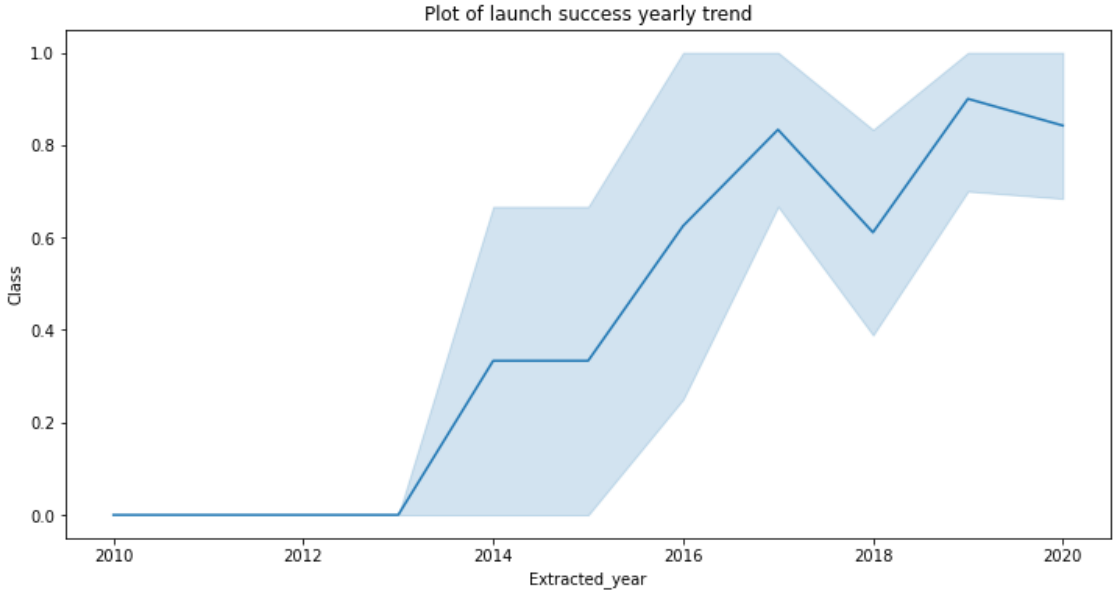
Payload vs. Orbit Type

Heavier payloads have a negative influence on GTO orbits and positive influence on ISS orbits



Launch Success Yearly Trend

The success rate since 2013 kept on increasing till 2020.



All Launch Site Names

Used the key word **DISTINCT** to show only unique launch sites from the SpaceX data.

What launch sites has SpaceX used?

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

Display the names of the unique launch sites in the space mission

In [10]:
```
task_1 = '''
        SELECT DISTINCT LaunchSite
        FROM SpaceX
'''
create_pandas_df(task_1, database=conn)
```

Out[10]:

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'CCA'

Used the query above to display 5 records where launch sites begin with `CCA`

- CCAFS LC-40 and CCAFS SLC-40

Display 5 records where launch sites begin with the string 'CCA'

In [11]:

```
task_2 = '''
        SELECT *
        FROM SpaceX
        WHERE LaunchSite LIKE 'CCA%'
        LIMIT 5
        '''
create_pandas_df(task_2, database=conn)
```

Out[11]:

	date	time	boosterversion	launchsite	payload	payloadmasskg	orbit	customer	missionoutcome	landingoutcome
0	2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	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
3	2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

Total payload carried by boosters from NASA is 45596 using the query

In [12]:
```
task_3 = '''
        SELECT SUM(PayloadMassKG) AS Total_PayloadMass
        FROM SpaceX
        WHERE Customer LIKE 'NASA (CRS)'
        '''
create_pandas_df(task_3, database=conn)
```

Out[12]:

	total_payloadmass
0	45596

Average Payload Mass by F9 v1.1

The average payload mass carried by booster version F9 v1.1 is 2928.4

In [13]:
```
task_4 = '''
        SELECT AVG(PayloadMassKG) AS Avg_PayloadMass
        FROM SpaceX
        WHERE BoosterVersion = 'F9 v1.1'
        '''
create_pandas_df(task_4, database=conn)
```

Out[13]:

	avg_payloadmass
0	2928.4

First Successful Ground Landing Date

The dates of the first successful landing outcome on ground pad was 22nd December 2015

In [14]:

```
task_5 = '''
        SELECT MIN(Date) AS FirstSuccessfull_landing_date
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Success (ground pad)'
        '''
create_pandas_df(task_5, database=conn)
```

Out[14]:

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload
between 4000 and 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

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

In [15]:

```
task_6 = '''
        SELECT BoosterVersion
        FROM SpaceX
        WHERE LandingOutcome = 'Success (drone ship)'
            AND PayloadMassKG > 4000
            AND PayloadMassKG < 6000
        '''
create_pandas_df(task_6, database=conn)
```

Out[15]:

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

Used wildcard like '%' to filter for WHERE Mission Outcome was a success or a failure.

In [16]:

```
task_7a = '''
        SELECT COUNT(MissionOutcome) AS SuccessOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Success%'
        '''

task_7b = '''
        SELECT COUNT(MissionOutcome) AS FailureOutcome
        FROM SpaceX
        WHERE MissionOutcome LIKE 'Failure%'
        '''
print('The total number of successful mission outcome is:')
display(create_pandas_df(task_7a, database=conn))
print()
print('The total number of failed mission outcome is:')
create_pandas_df(task_7b, database=conn)
```

Out[16]:

The total number of successful mission outcome is:

	successoutcome
0	100

The total number of failed mission outcome is:

	failureoutcome
0	1

Boosters Carried Maximum Payload

List the names of the booster versions which have carried the maximum payload mass.

- F9 B5 B1048.4
- F9 B5 B1048.5
- F9 B5 B1049.4
- F9 B5 B1049.5
- F9 B5 B1049.7
- F9 B5 B1051.3
- F9 B5 B1051.4
- F9 B5 B1051.6
- F9 B5 B1056.4
- F9 B5 B1058.3
- F9 B5 B1060.2
- F9 B5 B1060.3

In [17]:

```
task_8 = '''
        SELECT BoosterVersion, PayloadMassKG
        FROM SpaceX
        WHERE PayloadMassKG = (
                                SELECT MAX(PayloadMassKG)
                                FROM SpaceX
                                )
        ORDER BY BoosterVersion
        '''
create_pandas_df(task_8, database=conn)
```

Out[17]:

	boosterversion	payloadmasskg
0	F9 B5 B1048.4	15600
1	F9 B5 B1048.5	15600
2	F9 B5 B1049.4	15600
3	F9 B5 B1049.5	15600
4	F9 B5 B1049.7	15600
5	F9 B5 B1051.3	15600
6	F9 B5 B1051.4	15600
7	F9 B5 B1051.6	15600
8	F9 B5 B1056.4	15600
9	F9 B5 B1058.3	15600
10	F9 B5 B1060.2	15600
11	F9 B5 B1060.3	15600

2015 Launch Records

List the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015

- Failure (drone ship)F9 v1.1 B1012CCAFS LC-40
- Failure (drone ship)F9 v1.1 B1015CCAFS LC-40

In [18]:

```
task_9 = '''
        SELECT BoosterVersion, LaunchSite, LandingOutcome
        FROM SpaceX
        WHERE LandingOutcome LIKE 'Failure (drone ship)'
            AND Date BETWEEN '2015-01-01' AND '2015-12-31'
        '''
create_pandas_df(task_9, database=conn)
```

Out[18]:

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

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

Rank the count of landing outcomes between the date 2010-06-04 and 2017-03-20, in descending order

- 10 -No attempt5 -Failure (drone ship)
- 5 -Success (drone ship)
- 3 -Controlled (ocean)
- 3 -Success (ground pad)
- 2 -Failure (parachute)
- 2 -Uncontrolled (ocean)
- 1 -Precluded (drone ship)

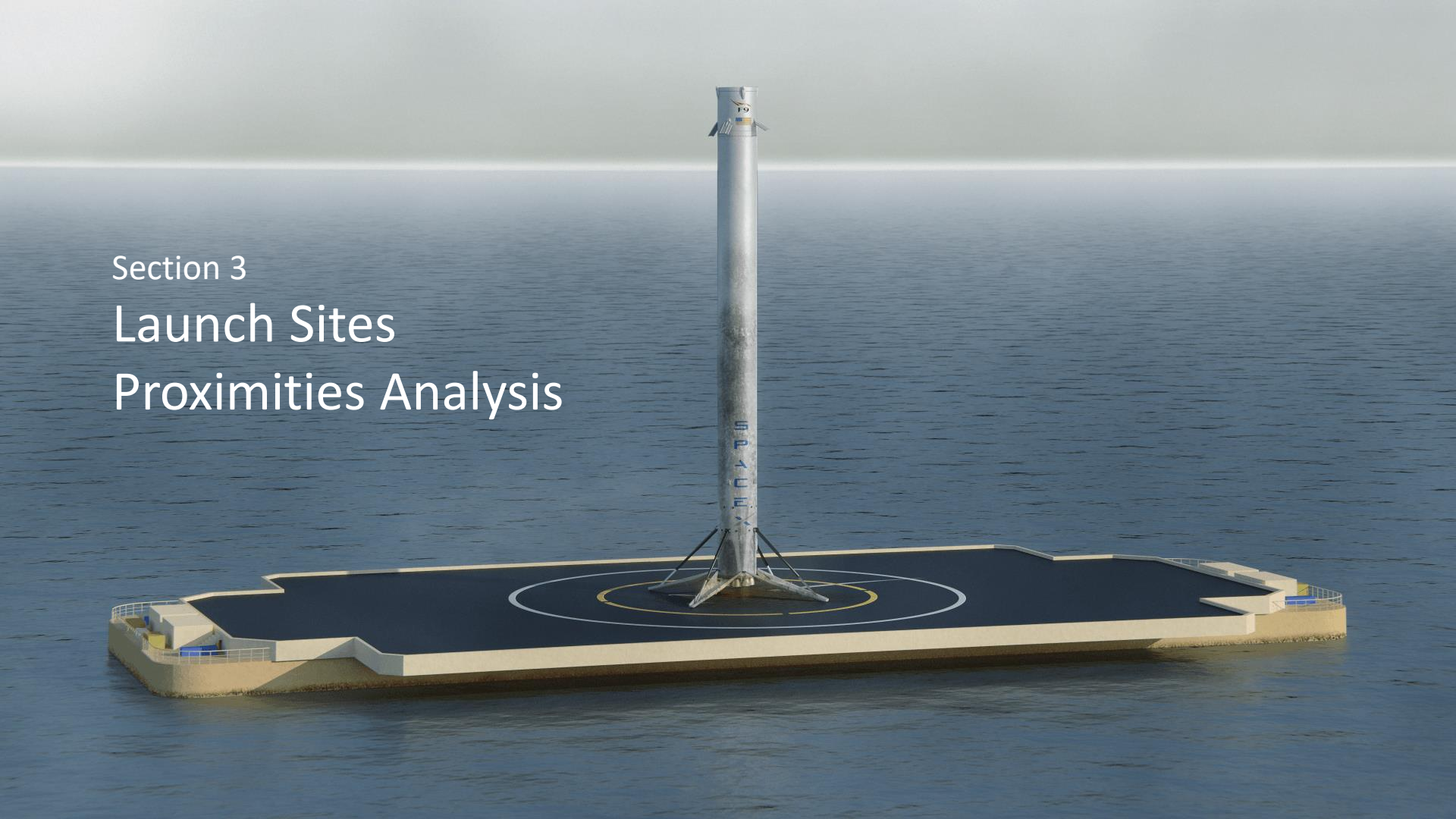
In [19]:
```
task_10 = '''
        SELECT LandingOutcome, COUNT(LandingOutcome)
        FROM SpaceX
        WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
        GROUP BY LandingOutcome
        ORDER BY COUNT(LandingOutcome) DESC
        '''
create_pandas_df(task_10, database=conn)
```

Out[19]:

	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

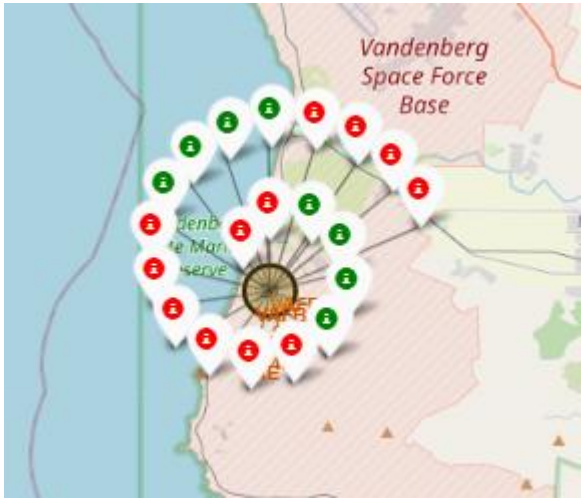
Section 3

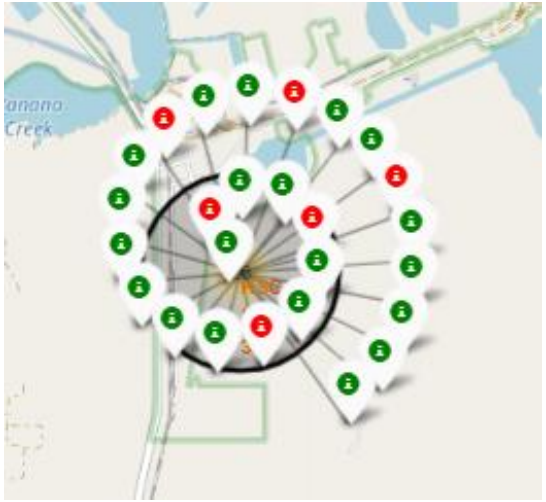
Launch Sites Proximities Analysis

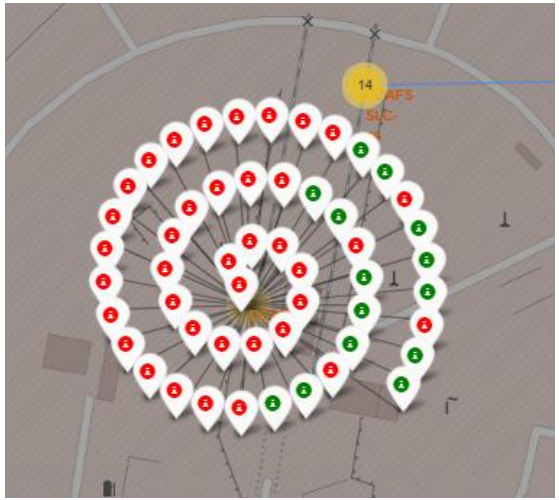


All launch sites global map markers

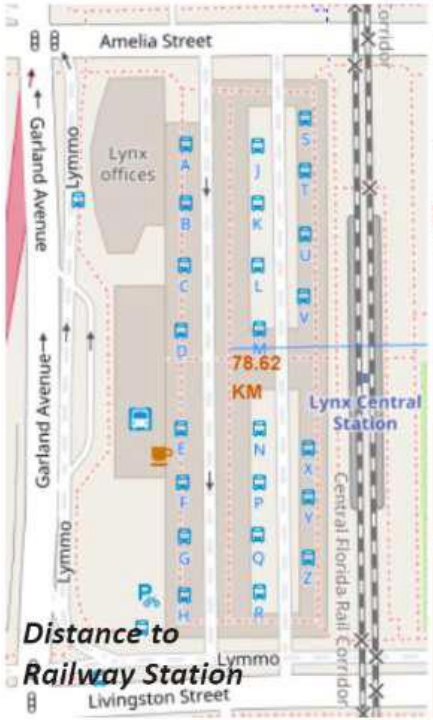
Markers showing launch sites with color labels







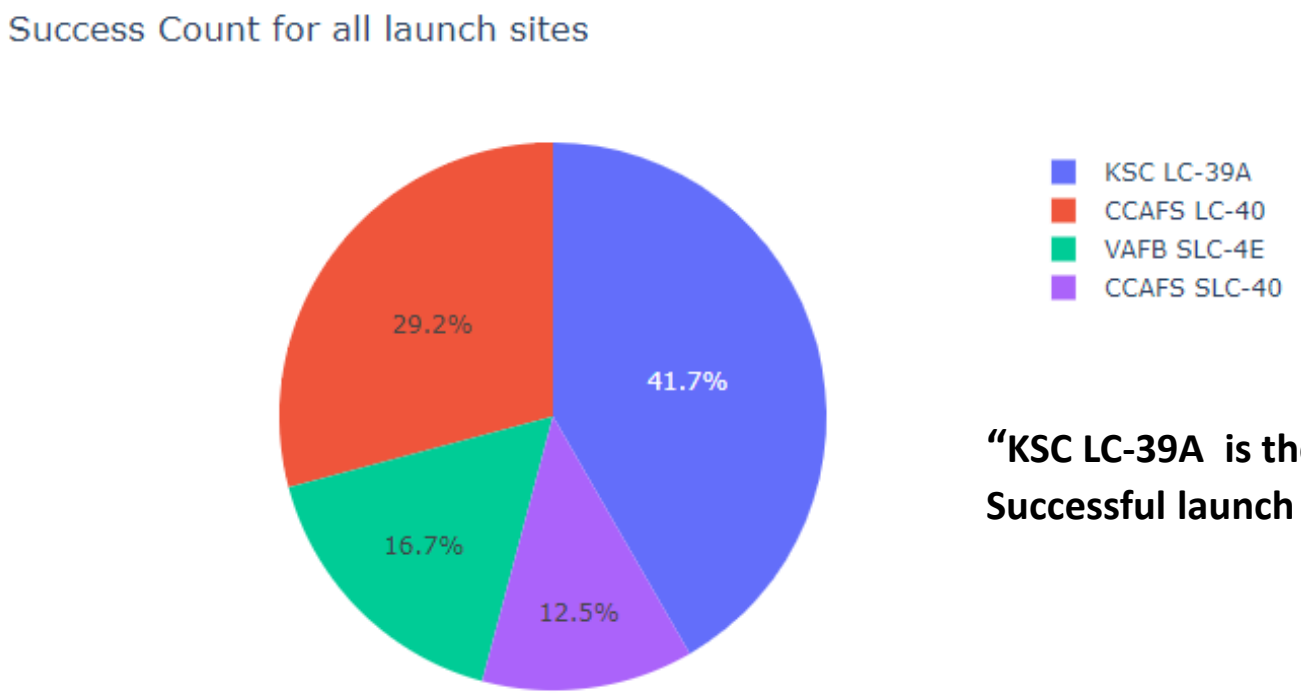
Launch Site distance to landmarks



- 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

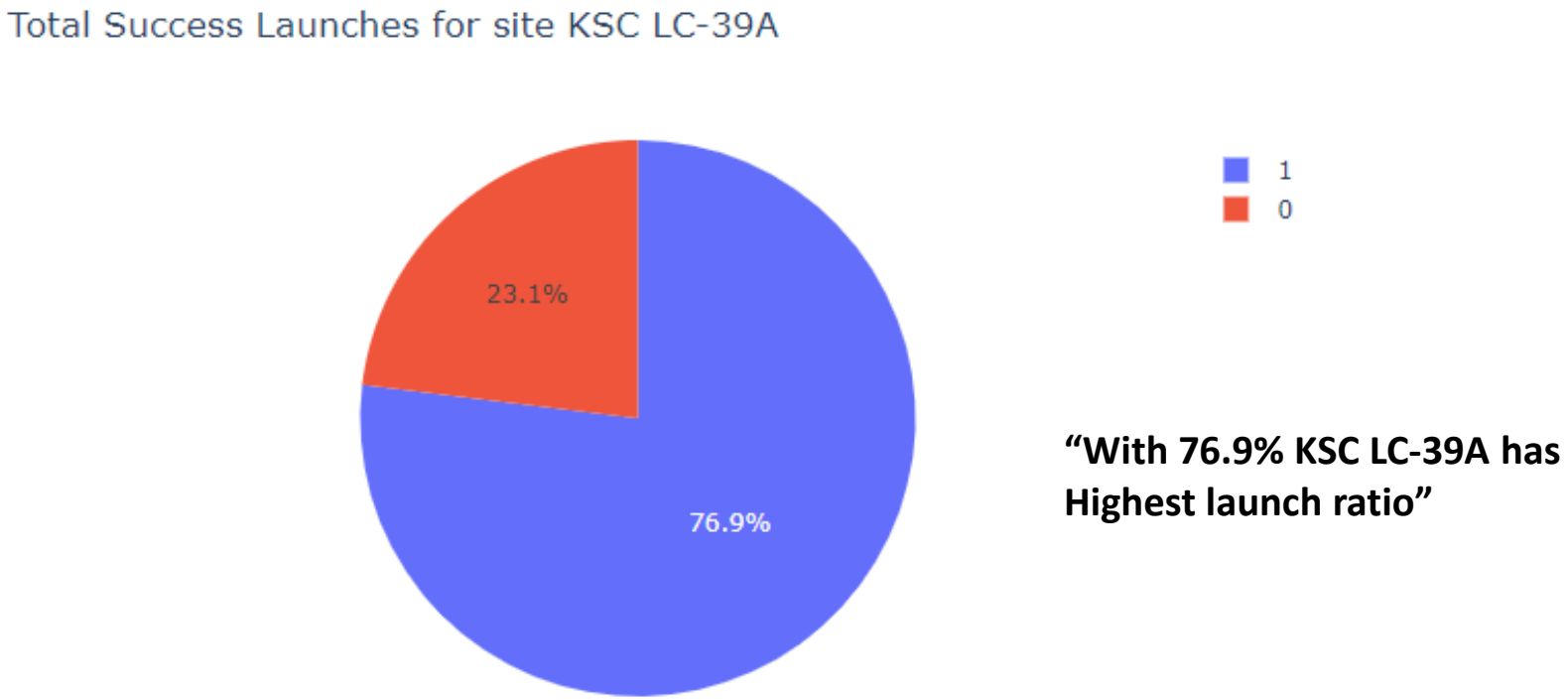

Section 4
Dashboard with
Plotly Dash

Success Percentage achieved by each launch site



“KSC LC-39A is the most Successful launch site.”

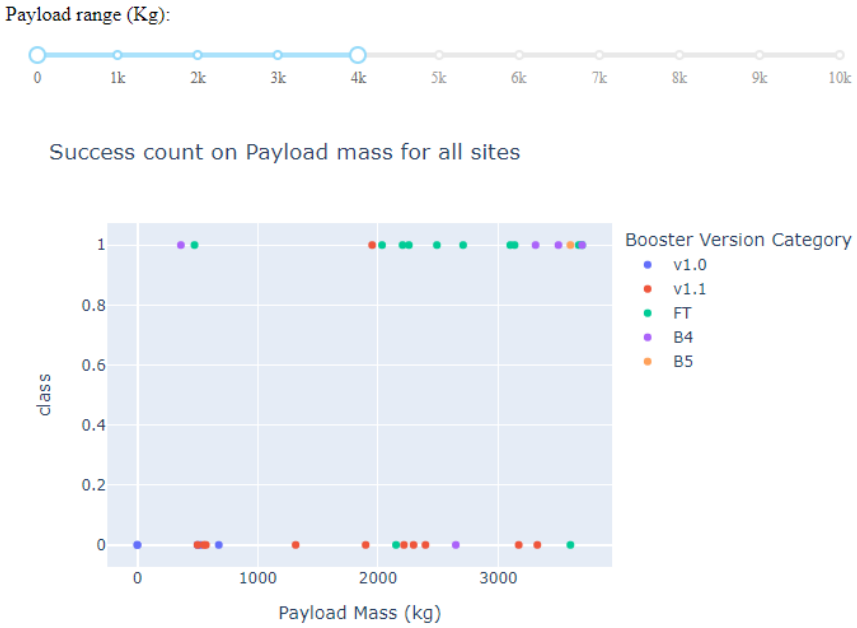
Launch site with the highest launch success ratio

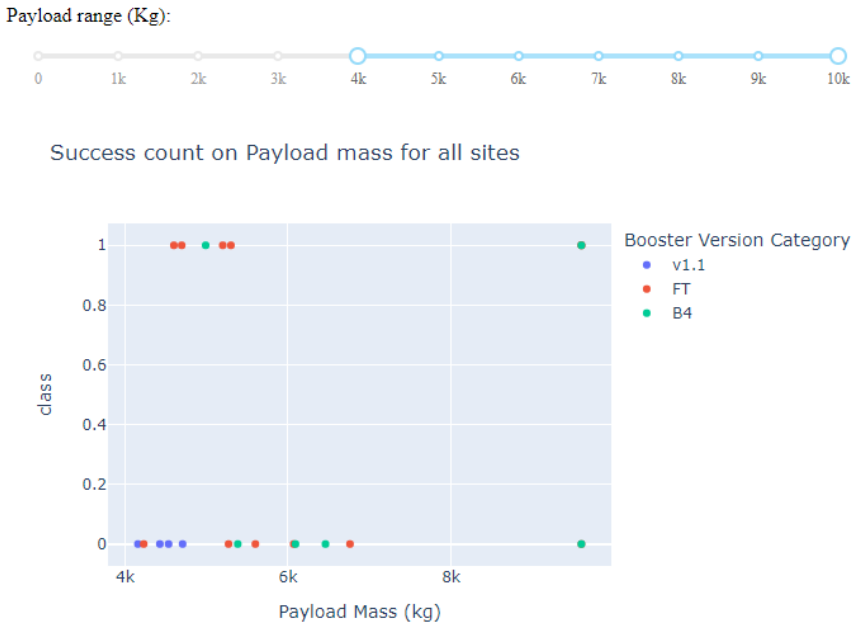


“With 76.9% KSC LC-39A has Highest launch ratio”

Payload vs. Launch Outcome for all sites

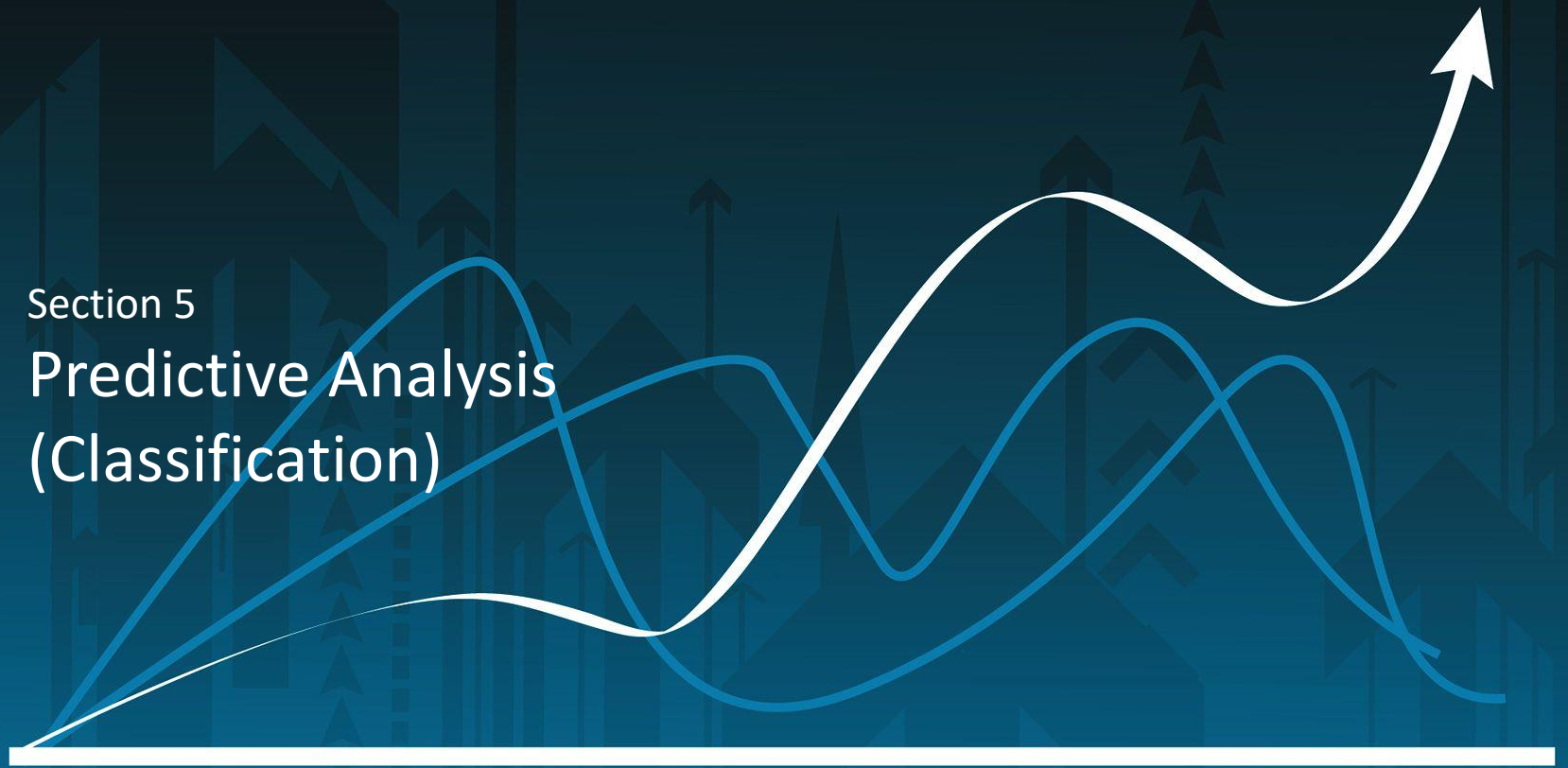
“Success rate for low payload mass is higher than heavy payload mass”





Section 5

Predictive Analysis (Classification)



Classification Accuracy

The decision tree classifier is the model with the highest classification accuracy have 87.3%

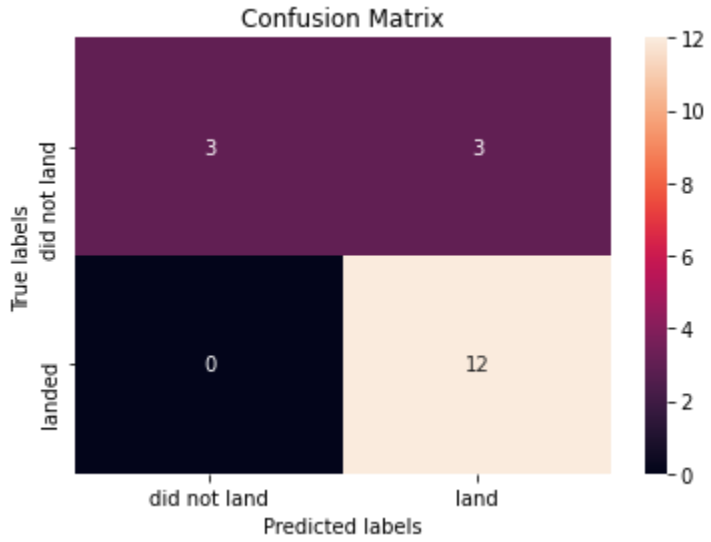
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.8732142857142856
Best params is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 5,
'splitter': 'random'}
```

Confusion Matrix

- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes
- The major problem is false positives as evidenced by the models incorrectly predicting the 1st stage booster to land in 3 out of 18 samples in the test set



Conclusion

- Using the models from this report SpaceY can predict when SpaceX will successfully land the 1st stage booster with 87.3% accuracy.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- Launch success rate started to increase in 2013.
- Success rate for low payload mass is higher than heavy payload mass.
- KSC LC-39A is the most Successful launch site.
- The Decision tree classifier is the best machine learning algorithm with 87.3% accuracy.

Appendix

Notebooks to recreate dataset, analysis, and models:

https://github.com/danielaryono/ibmcapstone-datascience


Thank you!