IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Abdelilah EL HADFAOUI
04/09/2022

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

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
  - We then cleaned the data, checked for missing values and fill in missing values where necessary.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection - SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

- The Jupyter notebook is on the link below :
- https://github.com/Abdelilahelhadfaoui/Datascience-Capstone-project/blob/master/Week%201%20:%20Data%20collection%20with%20webscrapping.ipynb

1. Get request for rocket launch data using API

In [6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

In [7]:
```
response = requests.get(spacex_url)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

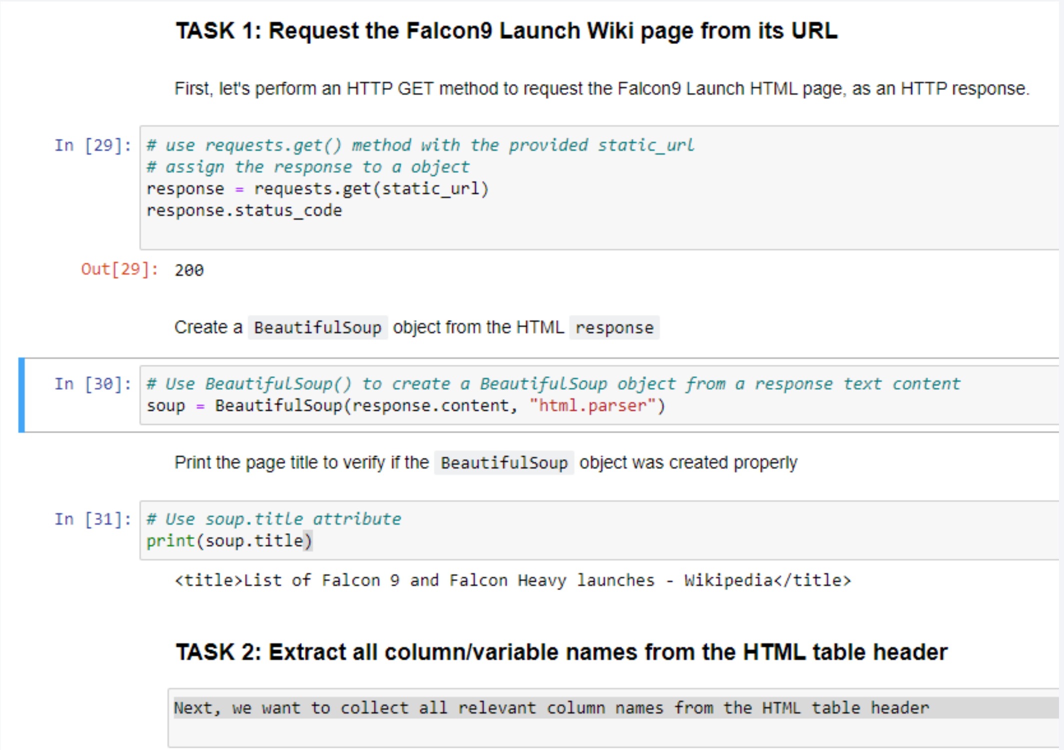
In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The Jupyter notebook is on the link below :
- https://github.com/Abdelilahelhadfaoui/Datascience-Capstone-project/blob/master/Week%201%20:%20Data%20collection%20with%20webscrapping.ipynb



TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

In [29]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
response.status_code
```

Out[29]: 200

Create a `BeautifulSoup` object from the HTML `response`

In [30]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, "html.parser")
```

Print the page title to verify if the `BeautifulSoup` object was created properly

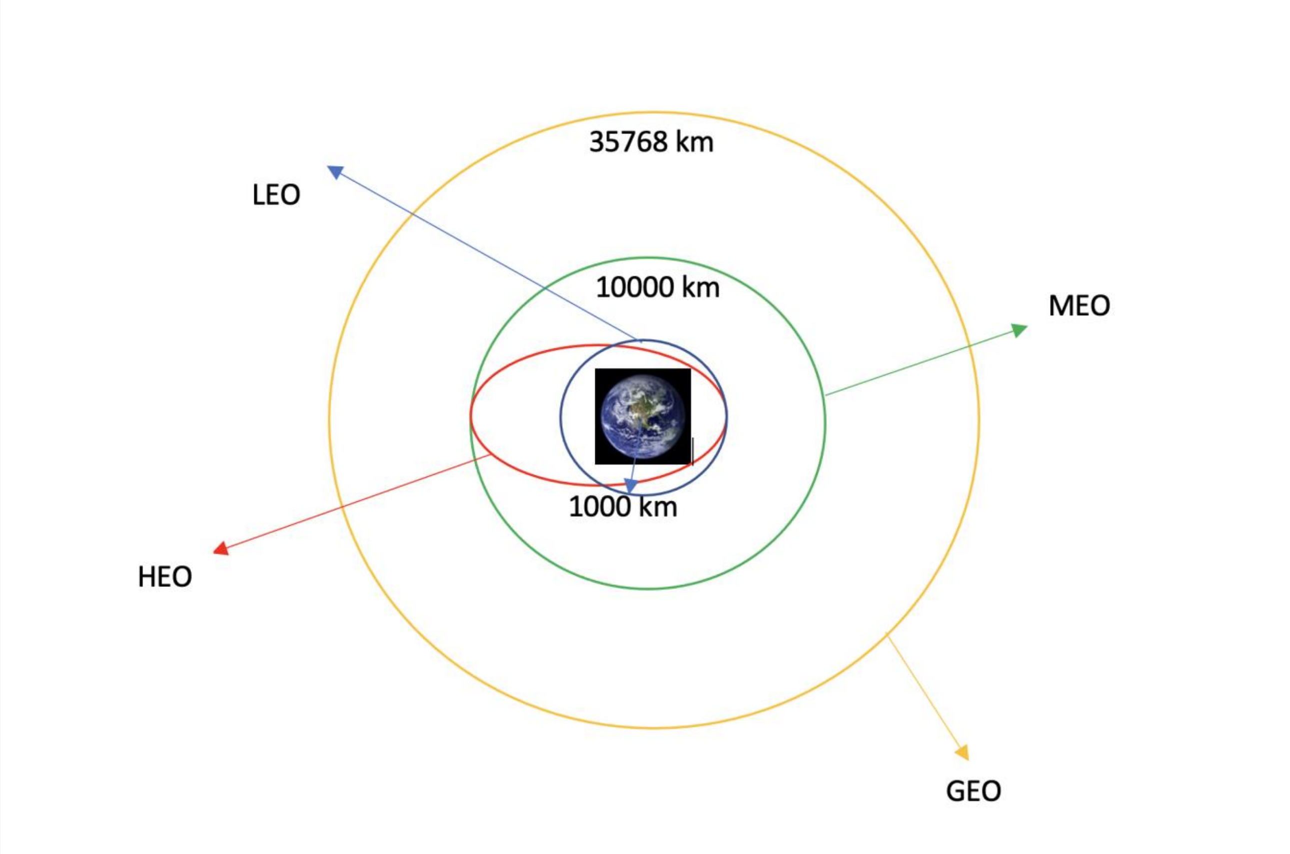
In [31]:
```
# Use soup.title attribute
print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

```
Next, we want to collect all relevant column names from the HTML table header
```

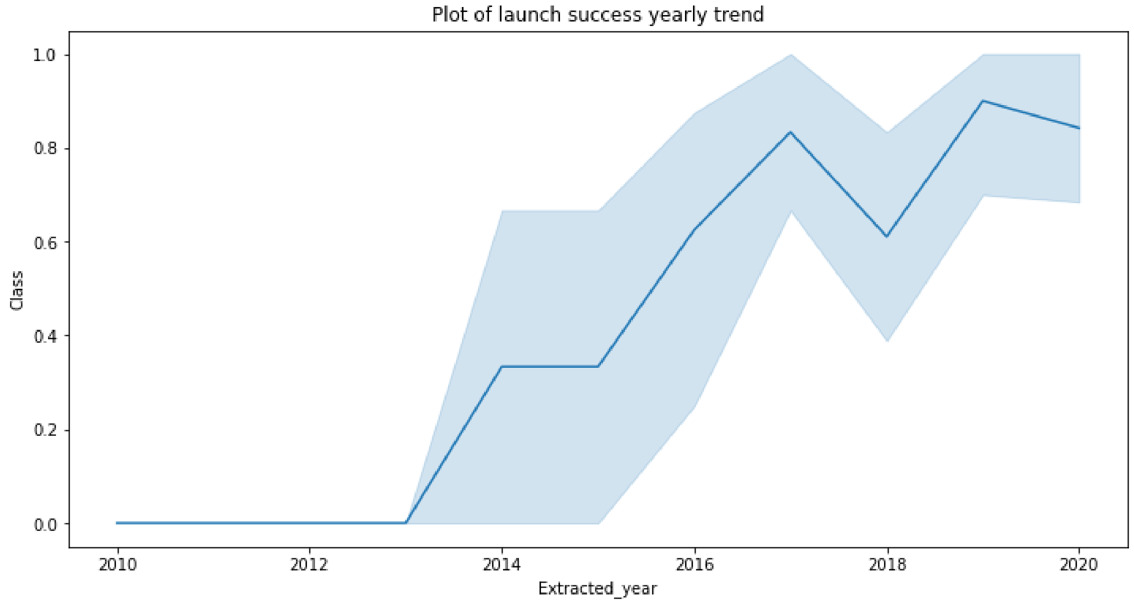
Data Wrangling

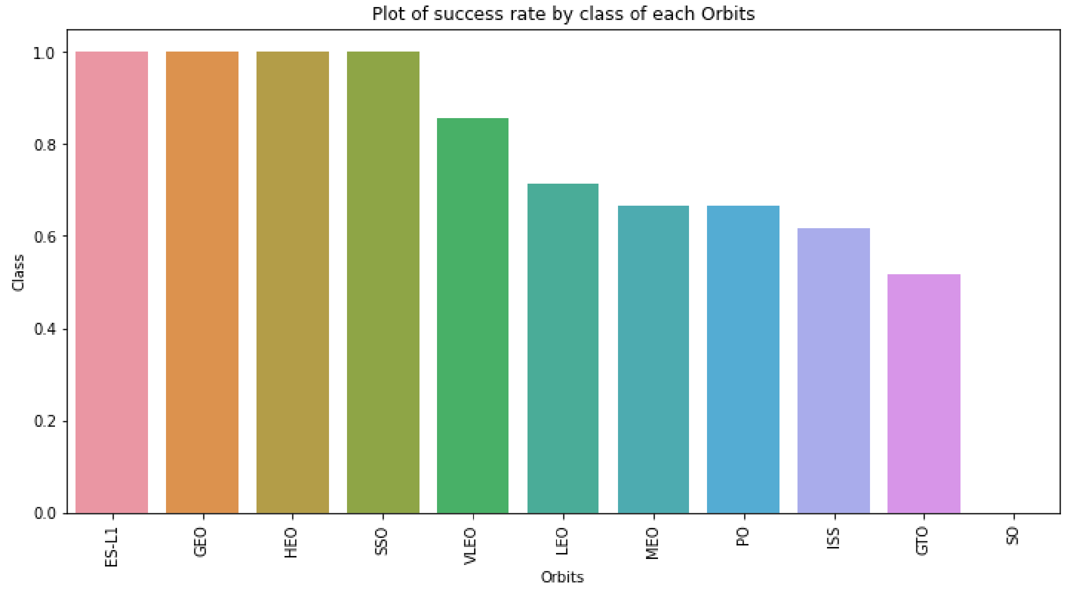


- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is : https://github.com/Abdelilahelhadfaoui/Datascience-Capstone-project/blob/master/Week%201%20-%20Data%20Wrangling.ipynb

EDA with Data Visualization

- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.





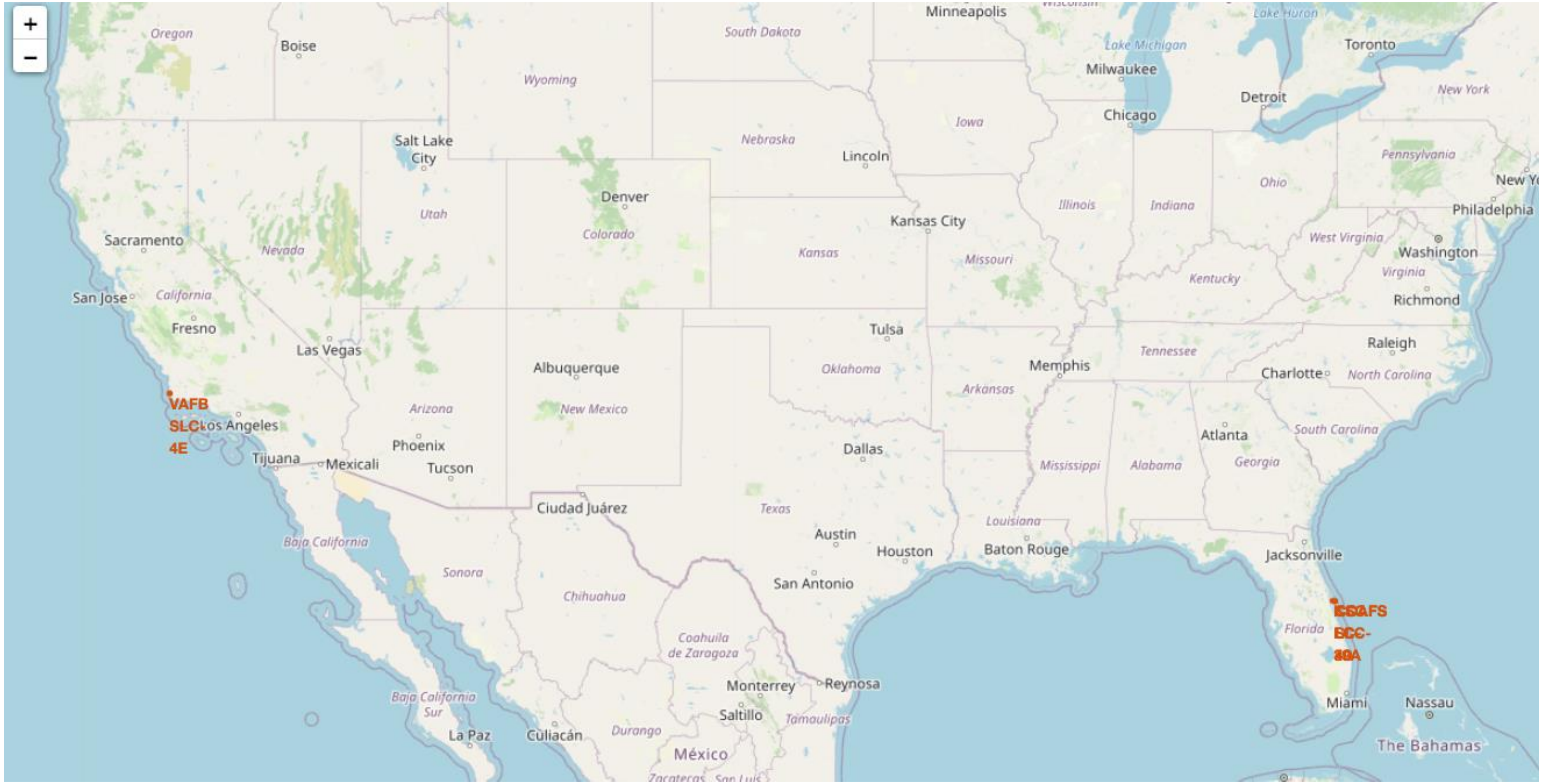
- The link to the notebook is : https://github.com/Abdelilahelhadfaoui/Datascience-Capstone-project/blob/master/Week%202%20-%20EDA%20with%20matplotlib.ipynb

EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook is : https://github.com/Abdelilahelhadfaoui/Datascience-Capstone-project/blob/master/Week%202%20:%20EDA%20with%20SQL%20.ipynb

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.



VAFB
SLC-4E
+
−
Oregon
Boise
Wyoming
South Dakota
Minneapolis
Lake Michigan
Lake Huron
Toronto
Milwaukee
Detroit
New York
Salt Lake City
Nebraska
Iowa
Chicago
Lincoln
Pennsylvania
Ohio
Denver
Utah
Illinois
Indiana
Philadelphia
Sacramento
Nevada
Colorado
Kansas City
Kansas
Missouri
West Virginia
Washington
Virginia
San Jose
California
Kentucky
Richmond
Fresno
Tulsa
Las Vegas
Tennessee
Raleigh
Albuquerque
Oklahoma
Memphis
Charlotte
North Carolina
Arkansas
Arizona
New Mexico
Los Angeles
Atlanta
South Carolina
Phoenix
Dallas
Tijuana
Mexicali
Tucson
Mississippi
Alabama
Georgia
Ciudad Juárez
Texas
Louisiana
Austin
Houston
Baton Rouge
Jacksonville
Baja California
Sonora
San Antonio
Chihuahua
Florida
Coahuila de Zaragoza
Monterrey
Reynosa
Miami
Nassau
Baja California Sur
Saltillo
Tamaulipas
La Paz
Culiacán
Durango
México
The Bahamas

Build a Dashboard with Plotly Dash

- We built an interactive dashboard with Plotly dash
- We plotted pie charts showing the total launches by a certain sites
- We plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.
- The link to the notebook is : https://github.com/Abdelilahelhadfaoui/Datascience-Capstone-project/blob/master/Week%203%20-%20Datascience%20Capstone.ipynb

Predictive Analysis (Classification)

- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- The link to the notebook is : https://github.com/Abdelilahelhadfaoui/Datascience-Capstone-project/blob/master/Week%204.ipynb

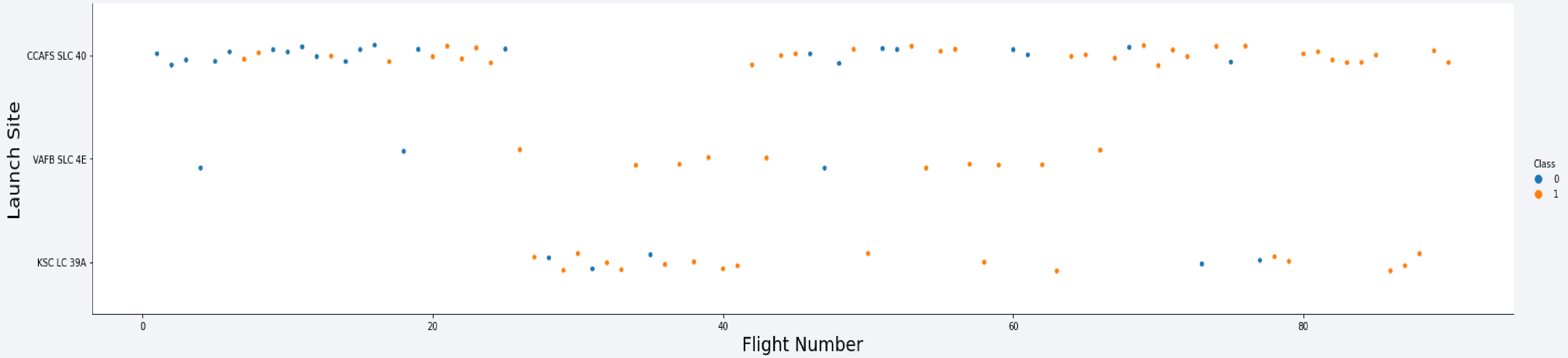
Results

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

Section 2

Insights drawn from EDA

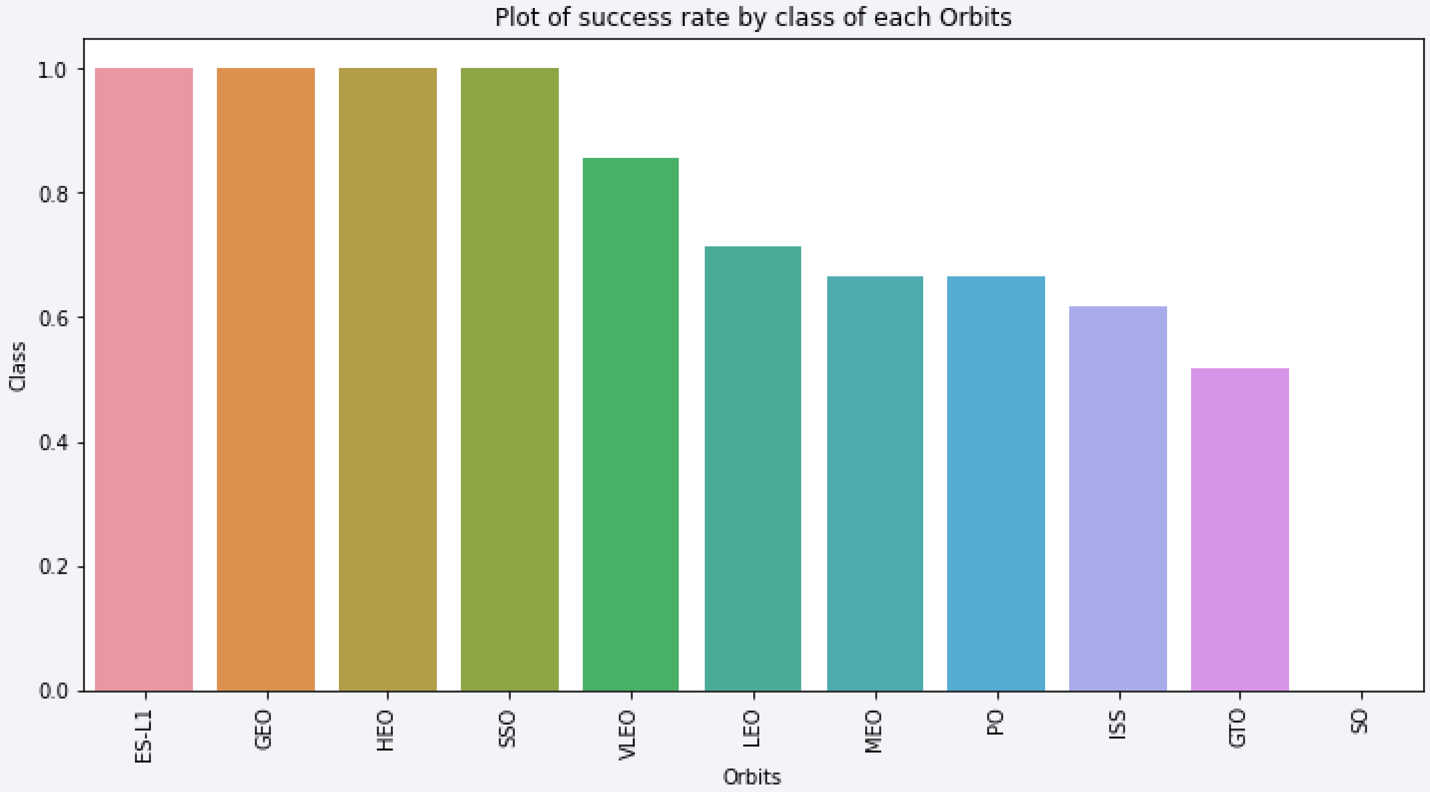
Flight Number vs. Launch Site



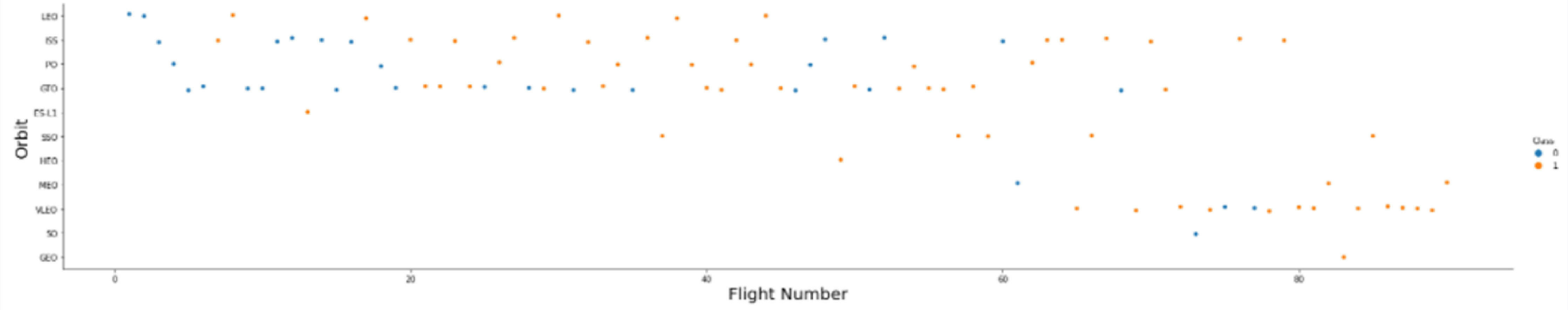
From the plot, we can see that the larger the flight amount at that launch site, the greater the success rate.

Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



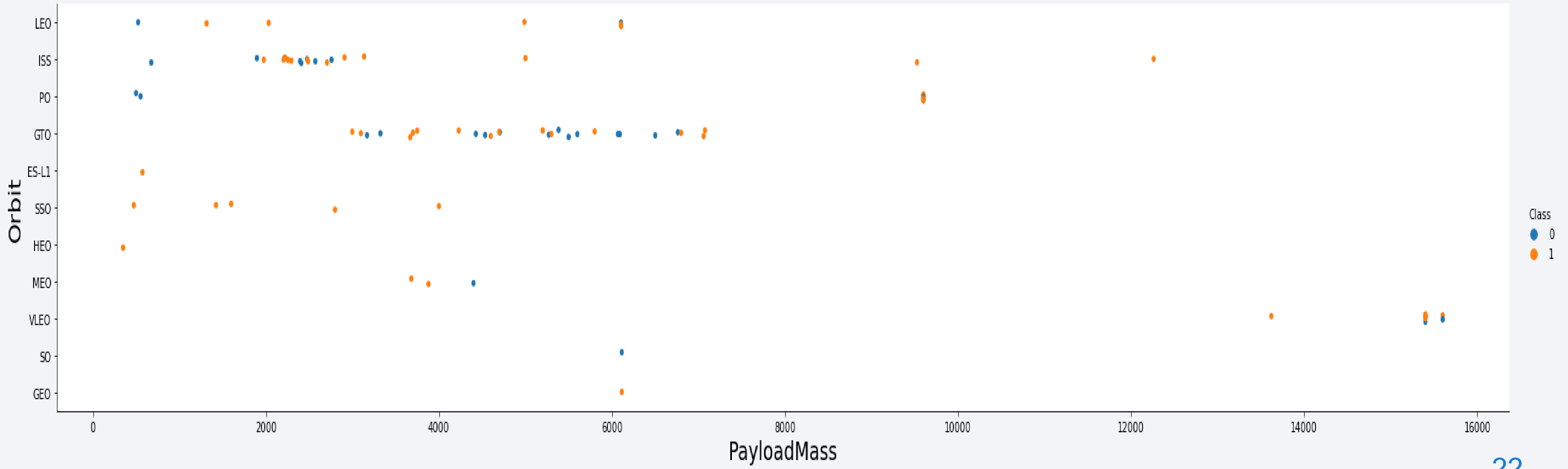
Flight Number vs. Orbit Type



- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

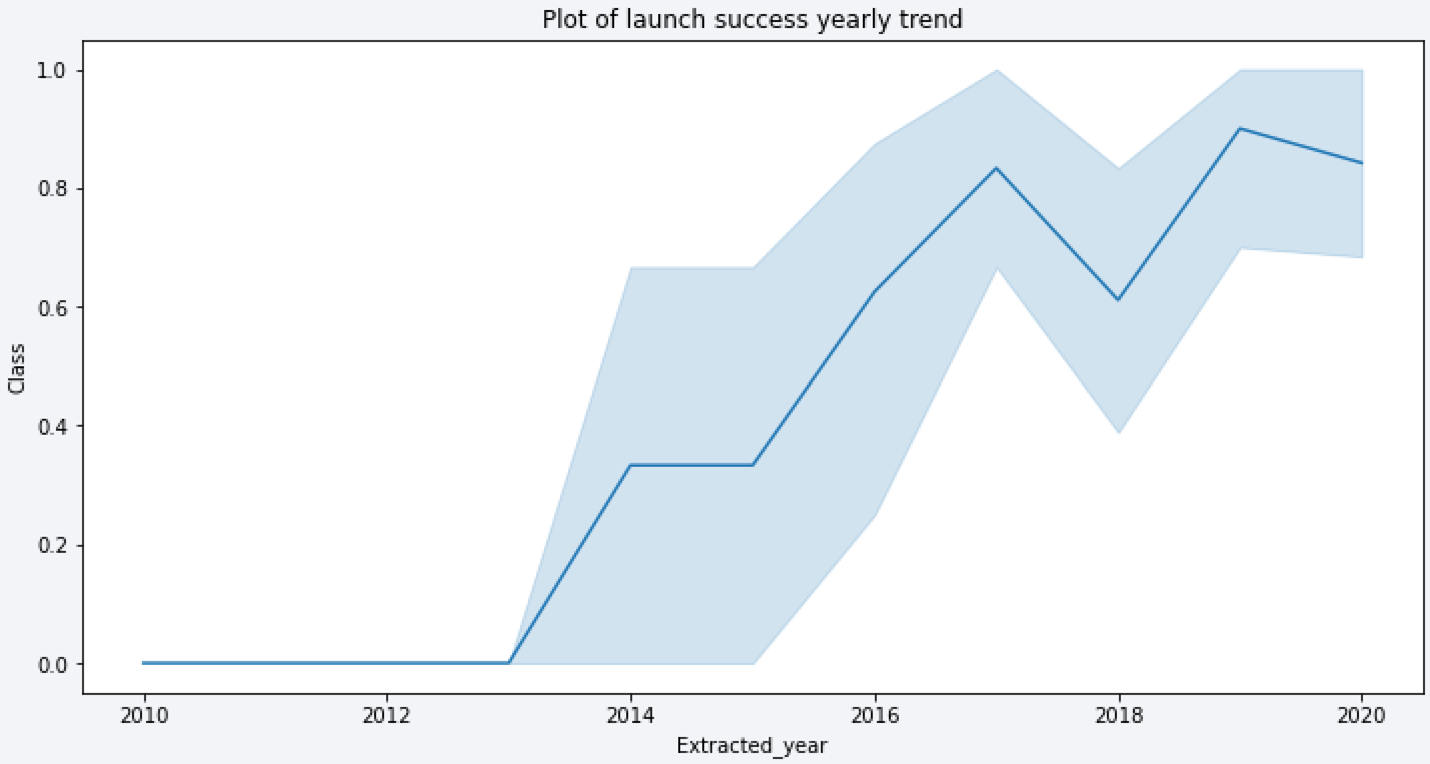
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

Task 1

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

In [11]:
```
%sql SELECT DISTINCT(launch_site)FROM SPACEXTBL
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[11]:

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

- We used the query above to display the names of the unique launch sites.

Launch Site Names Begin with 'CCA'

Task 2

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

In [46]:
```
%sql SELECT * FROM SPACEXTBL WHERE launch_site LIKE 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[46]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-12	22:41:00	F9 v1.1	CCAFS LC-40	SES-8	3170	GTO	SES	Success	No attempt

- We used the query above to display 5 records where launch sites begin with CCA

Total Payload Mass

- We calculated the total payload carried by boosters from NASA

Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

```
In [18]: %sql SELECT SUM(payload_mass__kg_) as Total_PayloadMass FROM SPACEXTBL WHERE customer =  'NASA (CRS)'

 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[18]:

total_payloadmass
22007

Average Payload Mass by F9 v1.1

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

Task 4

Display average payload mass carried by booster version F9 v1.1

```
In [24]: %sql SELECT AVG(payload_mass__kg_) AS Avg_PayloadMass FROM SPACEXTBL WHERE booster_version = 'F9 v1.1'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[24]:

avg_payloadmass
3676

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 2017-01-05

Task 5

List the date when the first successful landing outcome in ground pad was acheived.

Hint:Use min function

In [26]:
```
%sql SELECT MIN(DATE) AS first_success FROM SPACEXTBL WHERE landing__outcome LIKE 'Success (ground pad)'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[26]:

first_success
2017-01-05

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

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

```
In [27]: %sql SELECT booster_version FROM SPACEXTBL WHERE landing__outcome = 'Success (drone ship)' AND payload_mass__kg_ > 4000 AND payload_mass__kg_ < 6000

 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[27]:

booster_version
F9 FT B1022
F9 FT B1031.2

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

- We used wildcard like ‘%’ to filter for **WHERE** MissionOutcome was a success or a failure.

Task 7

List the total number of successful and failure mission outcomes

In [43]:
```
%sql SELECT COUNT(mission_outcome) AS Number_Success FROM SPACEXTBL WHERE mission_outcome LIKE '%Success%'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[43]:

number_success
45

In [44]:
```
%sql SELECT COUNT(mission_outcome) AS Number_Failure FROM SPACEXTBL WHERE mission_outcome LIKE '%Failure%'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[44]:

number_failure
0

Boosters Carried Maximum Payload

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [39]:
```
%sql SELECT booster_version, payload_mass__kg_ FROM SPACEXTBL WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL) ORDER BY booster_version
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[39]:

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600

- We determined the booster that have carried the maximum payload using a subquery in the WHERE clause and the **MAX()** function.

2015 Launch Records

- We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

Task 9

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

In [40]:
```
%sql SELECT booster_version, launch_site, landing__outcome FROM SPACEXTBL WHERE landing__outcome LIKE 'Failure (drone ship)' AND Date BETWEEN '2015-01-01' AND '2015-12-31'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[40]:

booster_version	launch_site	landing__outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)

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

Task 10

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

In [45]:
```
%sql SELECT landing__outcome, COUNT(landing__outcome) FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY landing__outcome ORDER BY COUNT(landing__outcome) DESC
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

Out[45]:

landing__outcome	2
No attempt	7
Failure (drone ship)	2
Success (drone ship)	2
Success (ground pad)	2
Controlled (ocean)	1
Failure (parachute)	1

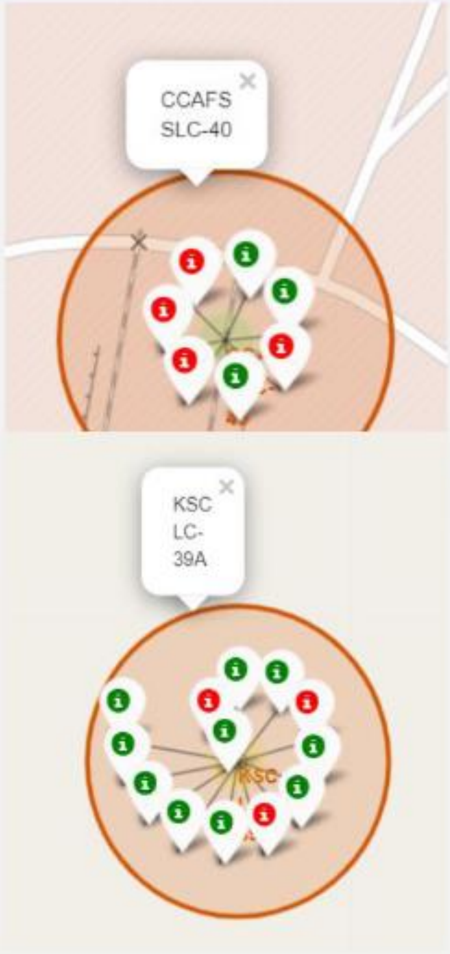
- We used WHERE and GROUP BY and ORDER BY.

Section 3

Launch Sites Proximities Analysis

All launch sites global map markers

Markers showing launch sites with color labels



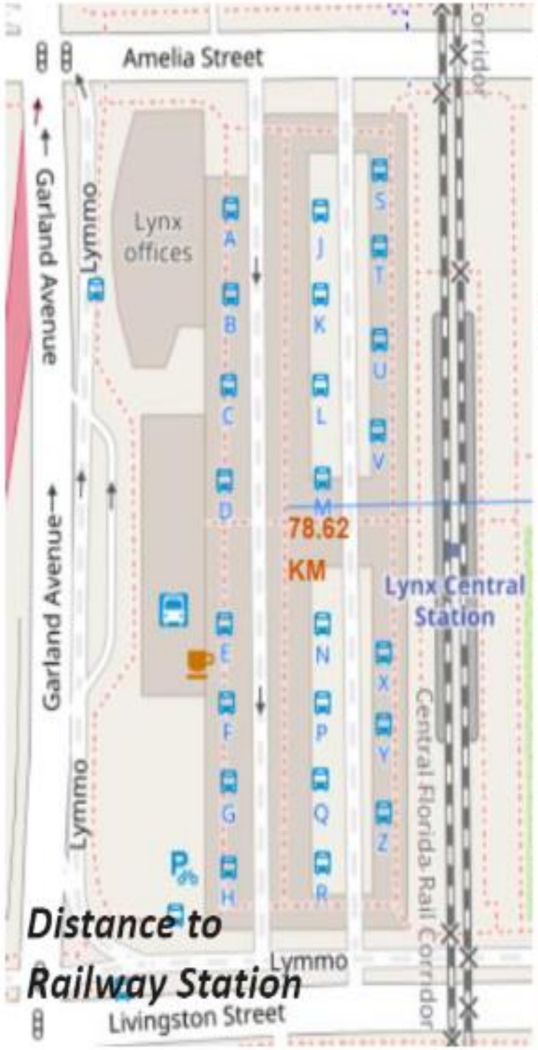
Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

37

Launch Site distance to landmarks

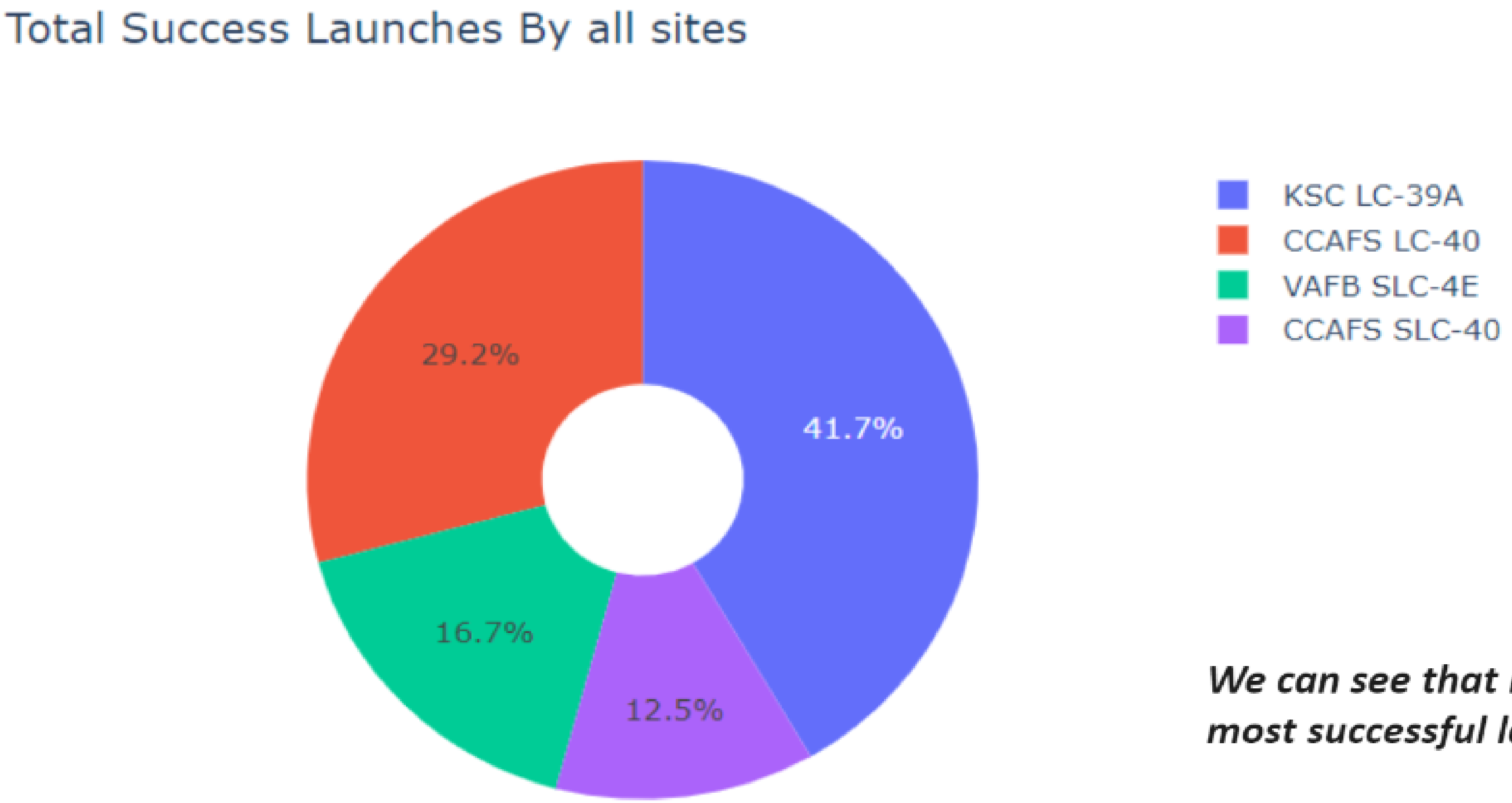


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

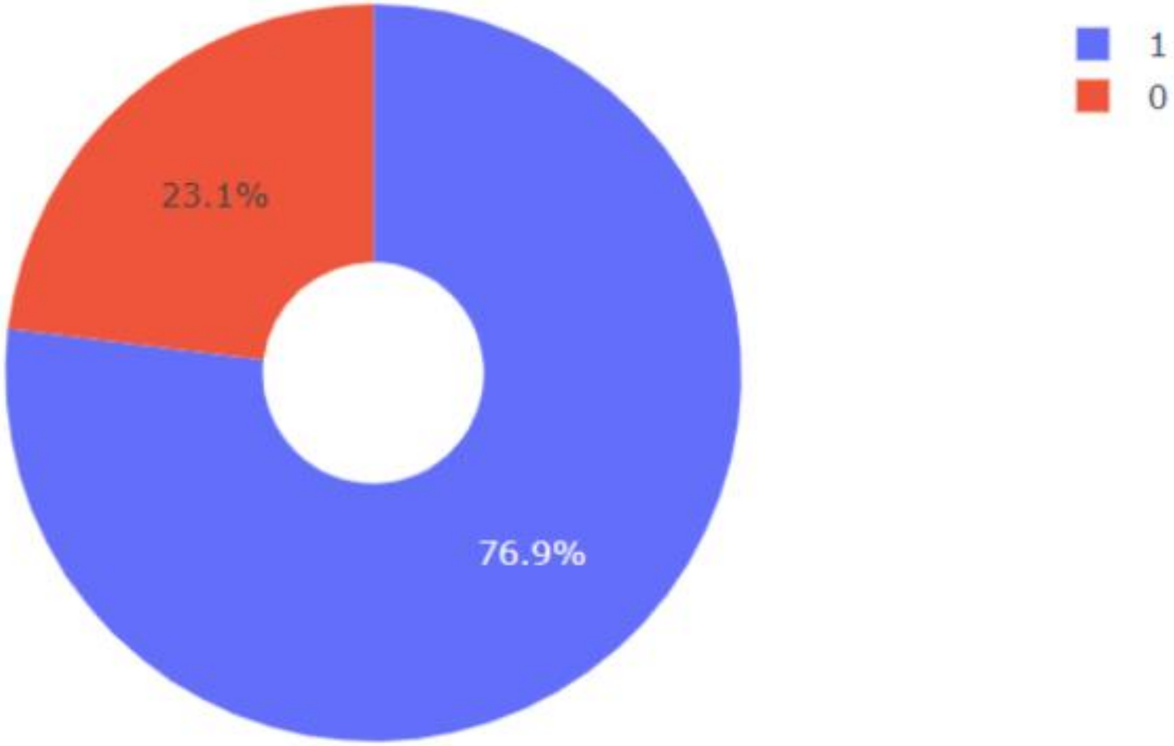
Build a Dashboard with Plotly Dash

Pie chart showing the success percentage achieved by each launch site



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

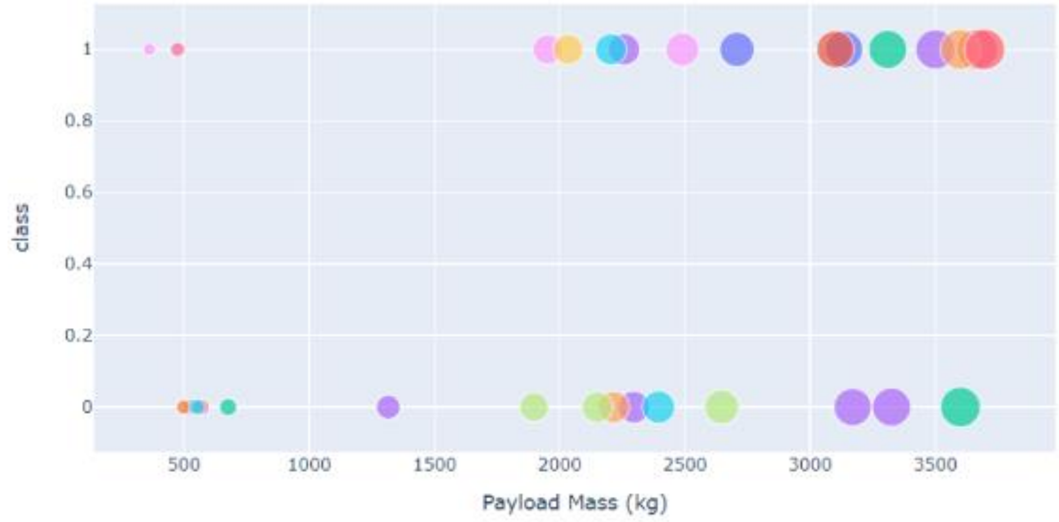
Pie chart showing the Launch site with the highest launch success ratio



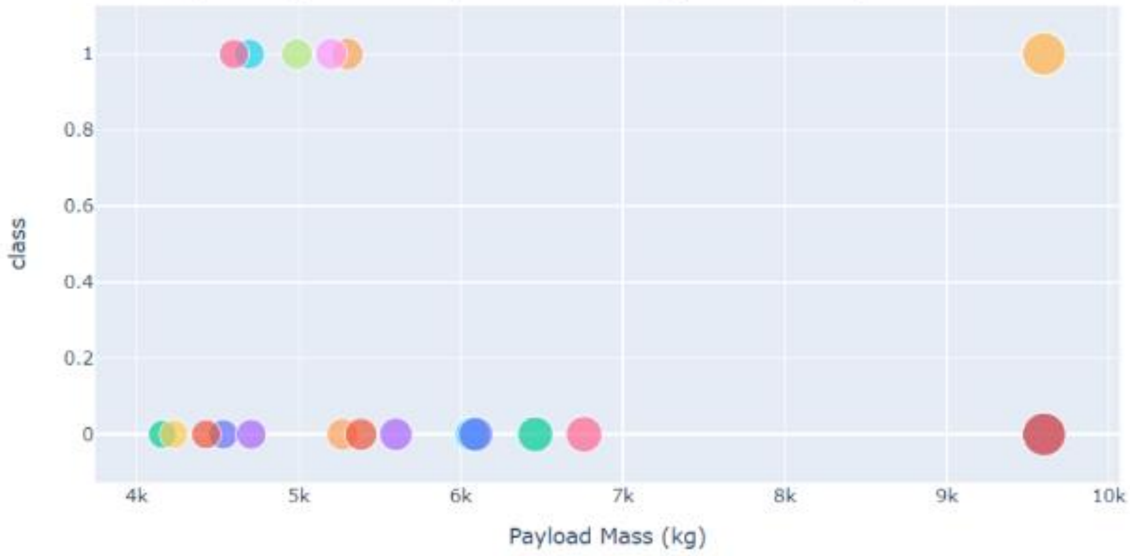
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Scatter plot of Payload vs Launch Outcome for all sites, with different payload selected in the range slider

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg



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

Section 5

Predictive Analysis (Classification)

Classification Accuracy

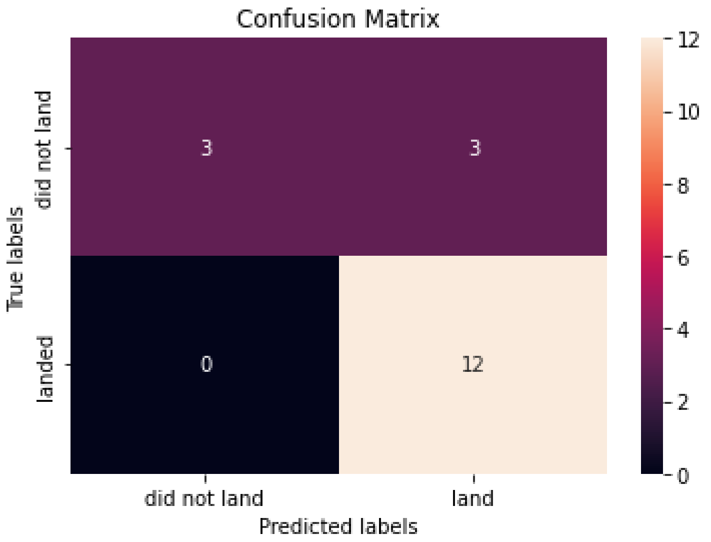
```
In [30]: models = {'KNeighbors':knn_cv.best_score_,
                   'DecisionTree':tree_cv.best_score_,
                   'LogisticRegression':logreg_cv.best_score_,
                   'SupportVector': svm_cv.best_score_}

         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 0.8875000000000002
    Best params is : {'criterion': 'gini', 'max_depth': 16, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
```

- The best model is Decision Tree with an Accuracy of 88.75%

Confusion Matrix

We can plot the confusion matrix

```
In [31]: yhat = tree_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```



- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.

Conclusions

This project and analysis allow us to conclude the following :

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate is in continuous increase since 2013.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!