IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Oladayo Timileyin Ebenezer

24th August, 2022

Outline

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

Executive Summary

- Summary of methodologies
  - ❖ Collecting the Data – API calls
  - ❖ Collecting the Data – Web Scrapping
  - ❖ Data Wrangling
  - ❖ Exploratory Data Analysis using SQL
  - ❖ Exploratory Data Analysis using Pandas and Matplotlib
  - ❖ Interactive Visual Analytics and Dashboard – Folium
  - ❖ Predictive Analysis (Classification)
- Summary of all results
  - ❖ Exploratory Data Analysis Result
  - ❖ Interactive Visual Analytics Dashboards (Screenshots)
  - ❖ Predictive Analysis Results

Introduction

- Project background and context

An alternative company intends to bid against SpaceX for a rocket launch. SpaceX advertises on its website that Falcon 9 rocket launches cost 62 million dollars while for other providers it costs upward of 165million dollars. Much of the savings of SpaceX are because they can reuse the first stage. As a result, if we can predict whether the first stage will land, we can estimate the cost of a launch. This will help the alternative company in its bid. Thus in this project, we will predict if the Falcon 9 first stage will land successfully.

- Problems you want to find answers
  - ✓ What criteria affect whether the rocket will successfully land.
  - ✓ The correlation between numerous features that determines the likelihood of a successful landing.
  - ✓ What operational requirements must be met to achieve a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

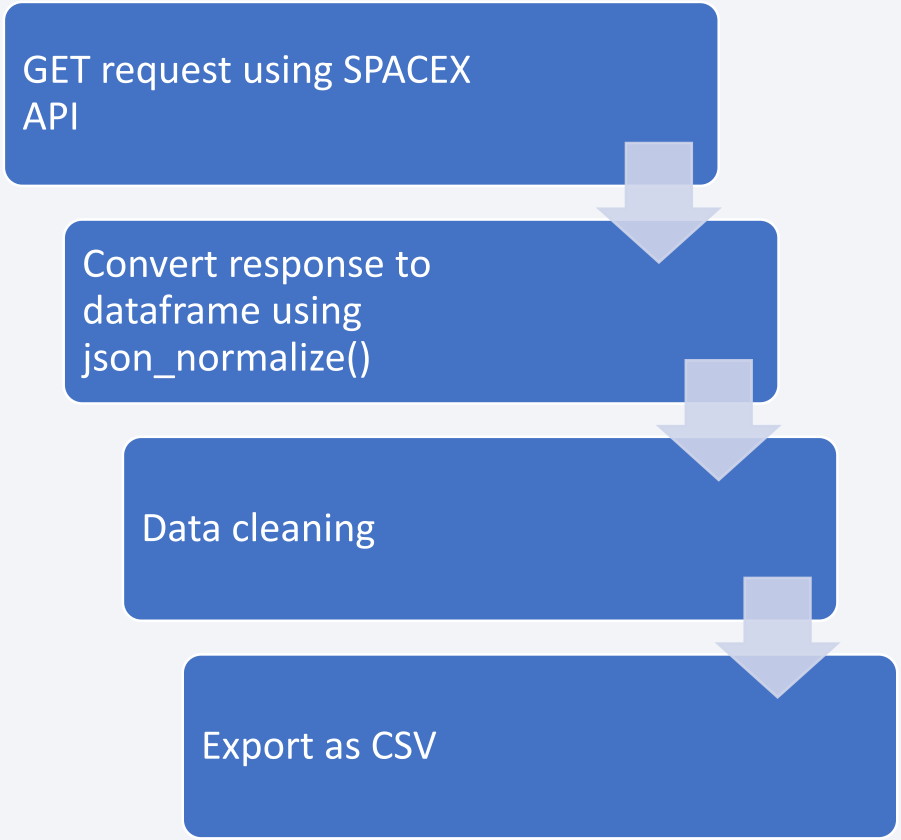
- Data collection methodology:
  - Data was collected using API from SpaceX and then Web Scrapping from Wikipedia.
- Perform data wrangling
  - One-Hot encoding was applied to the categorical features to get dummy numerical values
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Data was collected using API call and Web scrapping.
  - ➢ API call was made to SPACEX API through the getrequest package
  - ➢ We then decode the response content as a json using the json() function and then converted to a pandas dataframe using .json_normalize()
  - ➢ We cleaned the data to remove irrelevant data, check for missing value and replace for the PayloadMass feature
  - ➢ Also, we use web scrapping to get Falcon 9 launch records from Wikipedia using BeautifulSoup
  - ➢ We extracted the records as HTML Tables, parse it and convert to pandas dataframe.

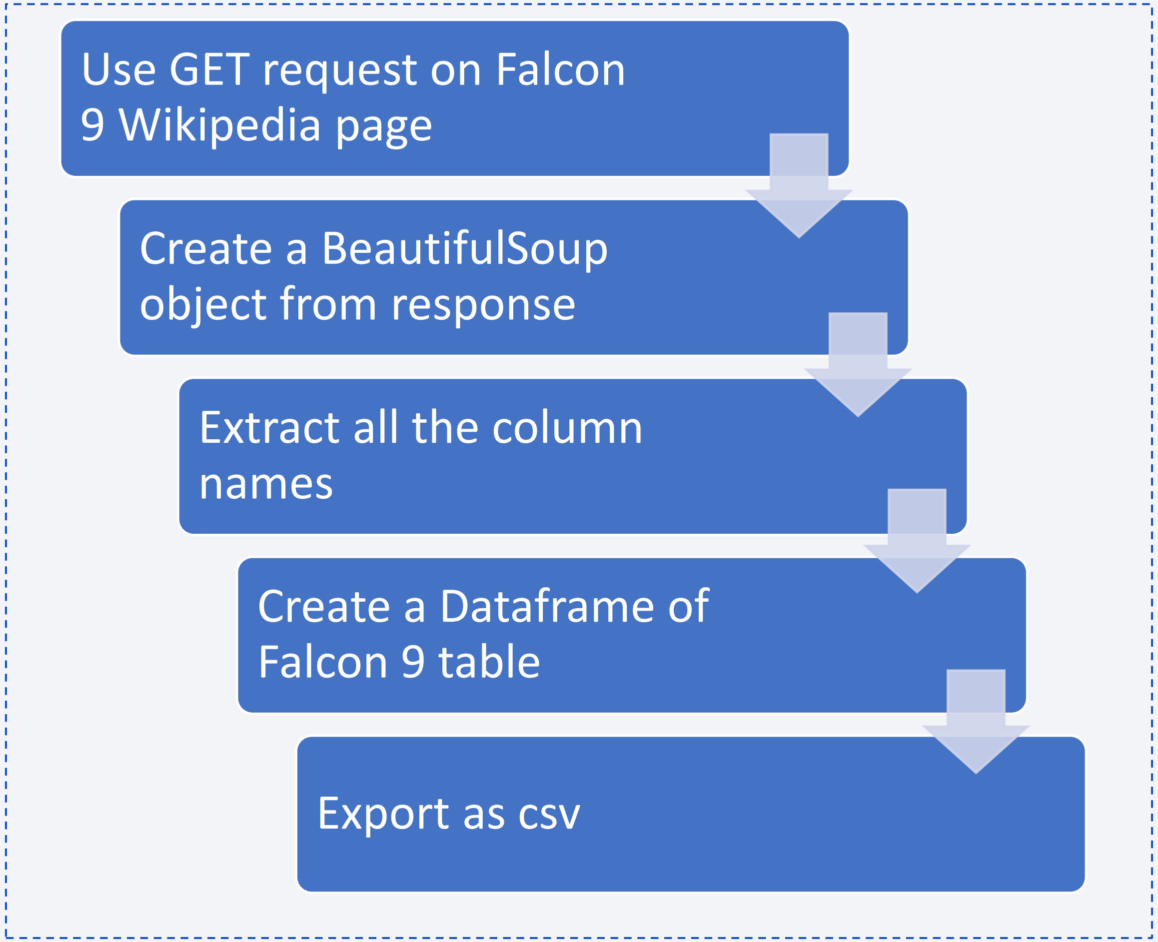
Data Collection – SpaceX API

- GET request was used to collect data from SPACEX API, data then cleaned and saved as csv
- GitHub URL of the notebook is https://github.com/RoyaltyServices/Coursera-Data-Capstone-Project/blob/3426345a9890eafcc14463a7de60350700218434/jupyter-labs-spacex-data-collection-api.ipynb



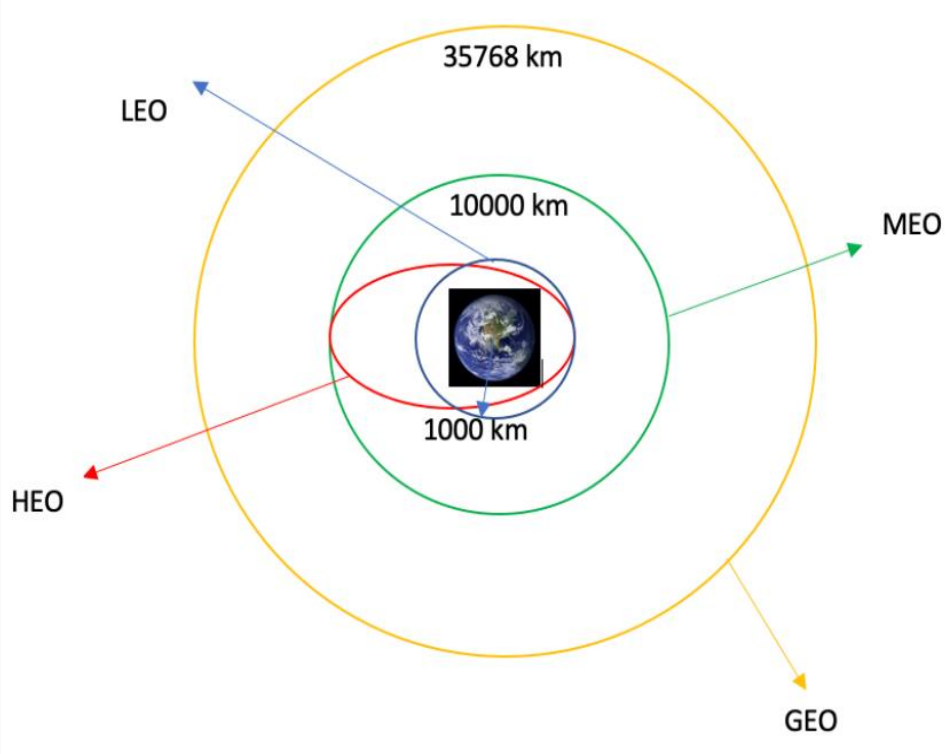
Data Collection - Scraping

- Web Scrapping was used to scrap Falcon 9 launces data from Wikipedia
- The html response was parsed and converted into a pandas dataframe and exported as csv
- GitHub URL of the notebook is https://github.com/RoyaltyServices/Coursera-Data-Capstone-Project/blob/3426345a9890eafcc14463a7de60350700218434/jupyter-labs-webscraping.ipynb



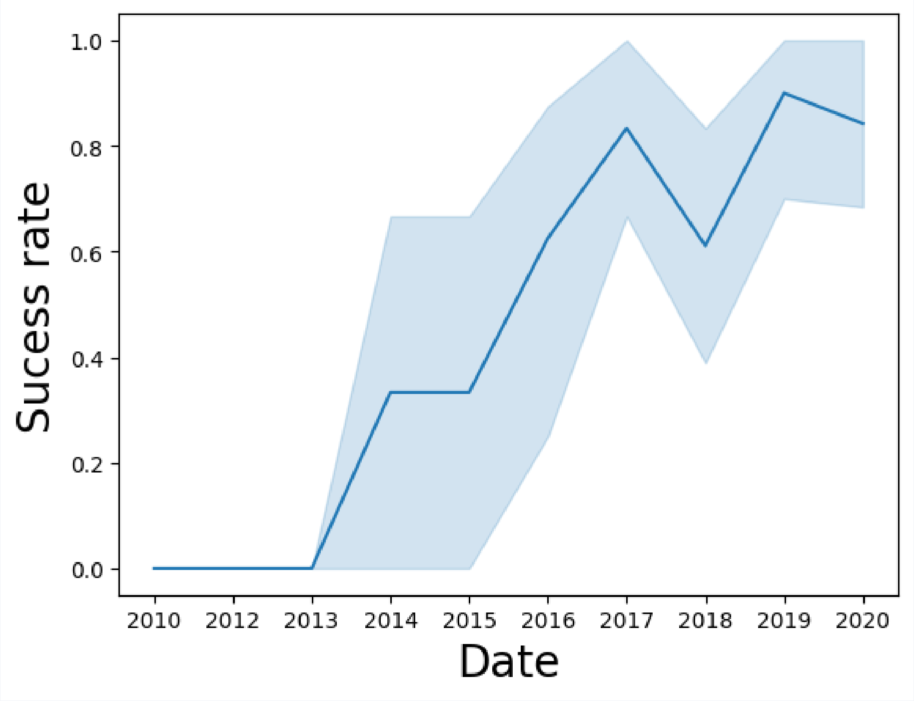
Data Wrangling

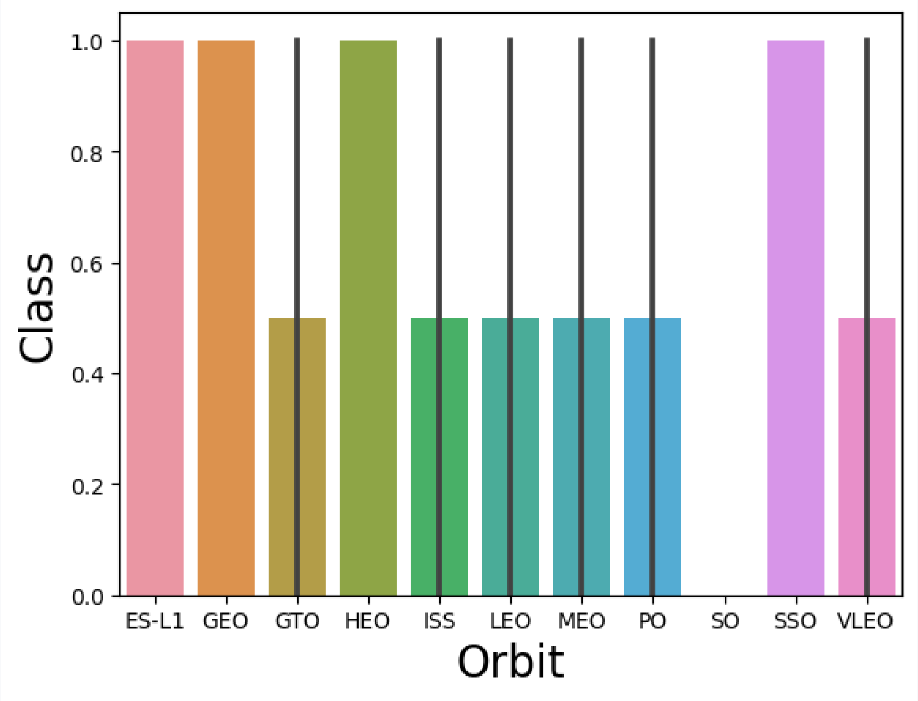
- We first did Exploratory Data Analysis and we the picked training labels
- Then, we calculated the number of launches at each site and the number and occurrence of each orbit
- Later, we created a landing outcome label from outcome column
- The data was exported to csv
- GIT URL link is https://github.com/RoyaltyServices/Coursera-Data-Capstone-Project/blob/3426345a9890eafcc14463a7de60350700218434/labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

- We investigated the data by visualizing the correlation between the flight number and the launch site, the payload and the launch site, the success rate of each orbit type, the flight number and the orbit type, and the yearly launch success.





GitHub URL of notebook
https://github.com/RoyaltyServices/Coursera-Data-Capstone-Project/blob/3426345a9890eafcc14463a7de60350700218434/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- Dataset was loaded into SQL Lite using the in-notebook command magic
- The following queries were then used to get insights from the Data
  - ➢ The names of unique launch sites in the space mission
  - ➢ The total payload mass carried by NASA (CRS) launched boosters
  - ➢ Average Payload Mass carried by booster version F9 v1.1
  - ➢ Total number of successful and failure mission outcomes
  - ➢ Failed landing outcomes in droneships, booster versions, and launch sites
- GitHub URL https://github.com/RoyaltyServices/Coursera-Data-Capstone-Project/blob/3426345a9890eafcc14463a7de60350700218434/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- We marked all launch locations and added map elements such as markers, circles, and lines to the folium map to indicate the success or failure of launches for each site.
- The feature launch outcomes (failure or success) were assigned to classes 0 and 1. That is, a 0 for failure and a 1 for success.
- We found which launch sites had a pretty high success rate using color-labeled marker clusters.
- We measured the distances between a launch location and its surroundings.
- GitHub URL https://github.com/RoyaltyServices/Coursera-Data-Capstone-Project/blob/3426345a9890eafcc14463a7de60350700218434/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- We used plotly dash to build an interactive dashboard
- We used pie charts to show total launches by a certain site
- We plot scatter plot to show correlation between Outcome and Payload Mass for the various booster version
- GitHub URL https://github.com/RoyaltyServices/Coursera-Data-Capstone-Project/blob/3426345a9890eafcc14463a7de60350700218434/spacex_dash_app.py

Predictive Analysis (Classification)

- We imported the data with numpy and pandas, converted it, then divided it into training and testing sets.
- We built different machine learning models (KNN, SVM, DecisionTree, LogisticRegression) and tune different hyperparameters using GridSearchCV to find the best
- We utilized accuracy as our model's measure and increased it through feature engineering and algorithm tweaking.
- We discovered the most effective categorization model.
- GitHub URL https://github.com/RoyaltyServices/Coursera-Data-Capstone-Project/blob/9ab5f7d56ec76bc321af0c7cfb6d7f402e778be2/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb

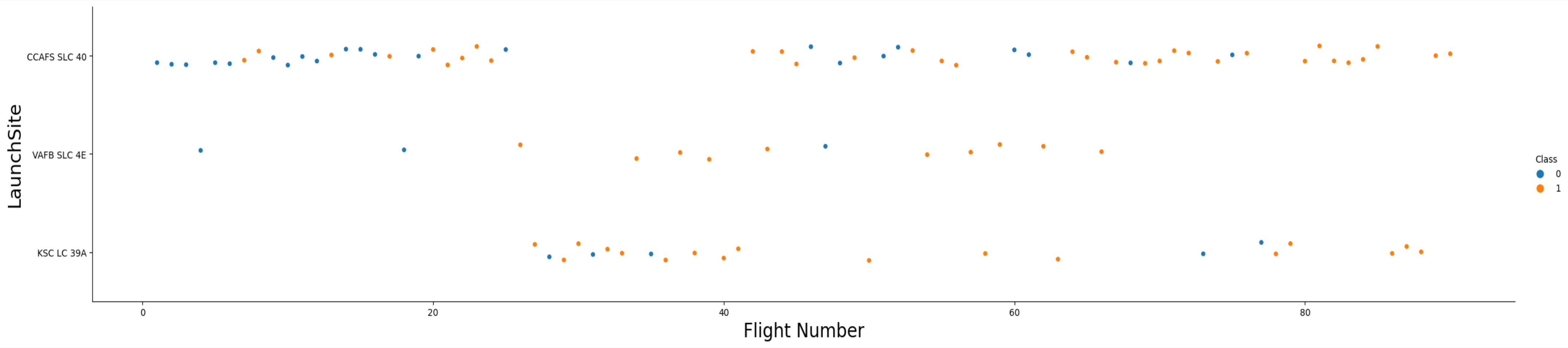
Results

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

Section 2

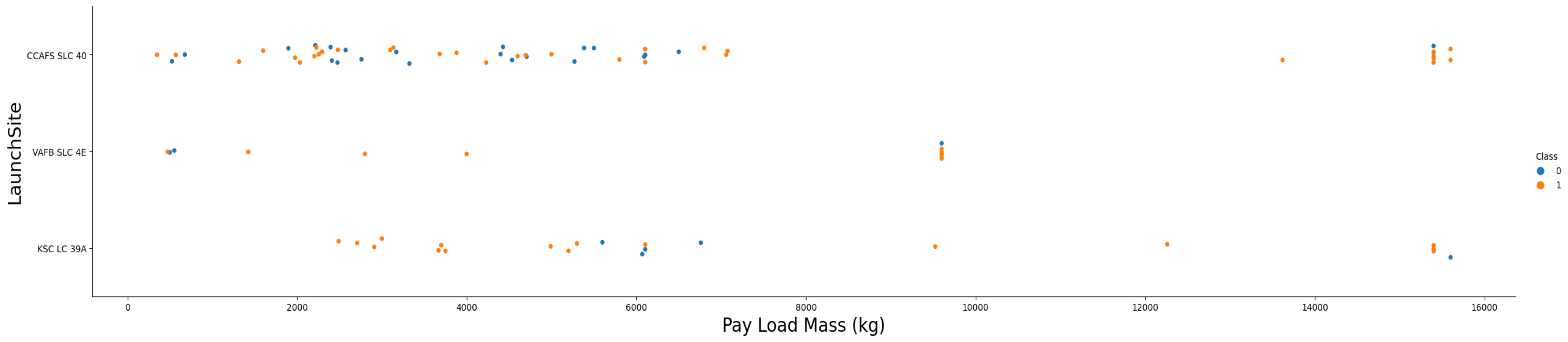
Insights drawn from EDA

Flight Number vs. Launch Site



- It was observed from the plot above that the more the flight number, the more the successful outcome from each launch sites.

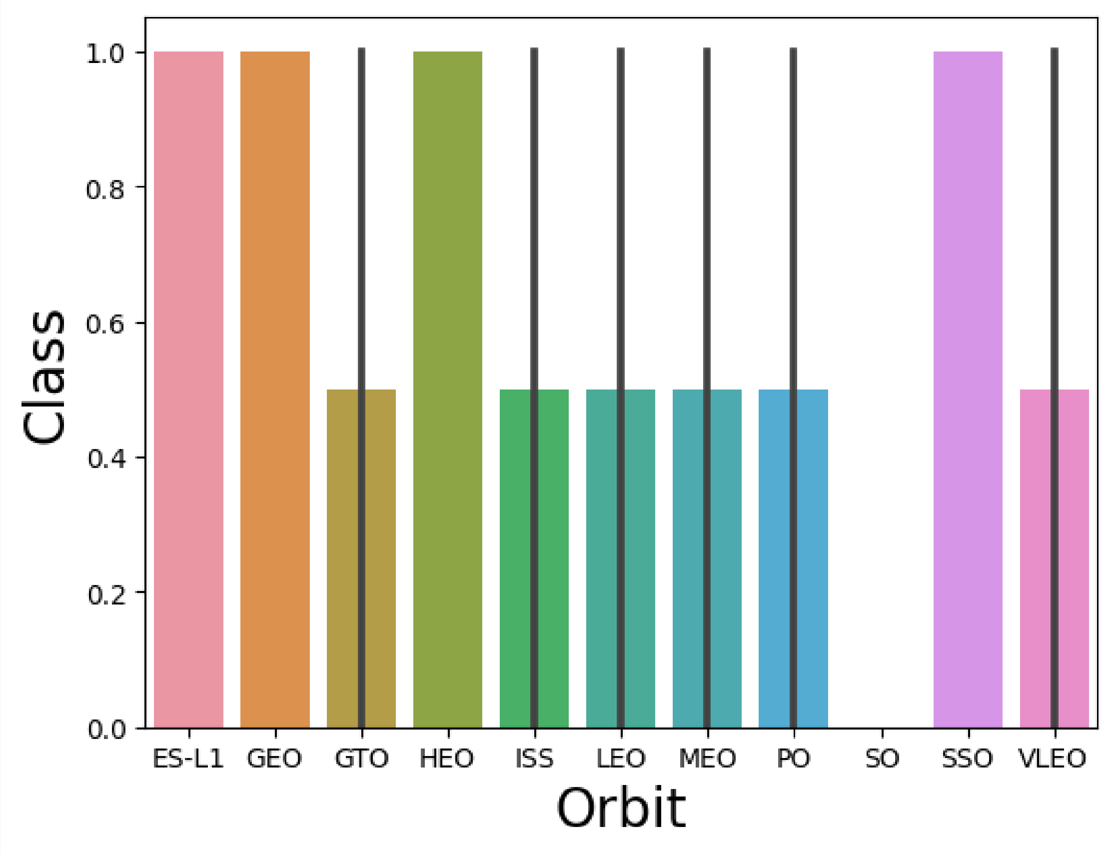
Payload vs. Launch Site



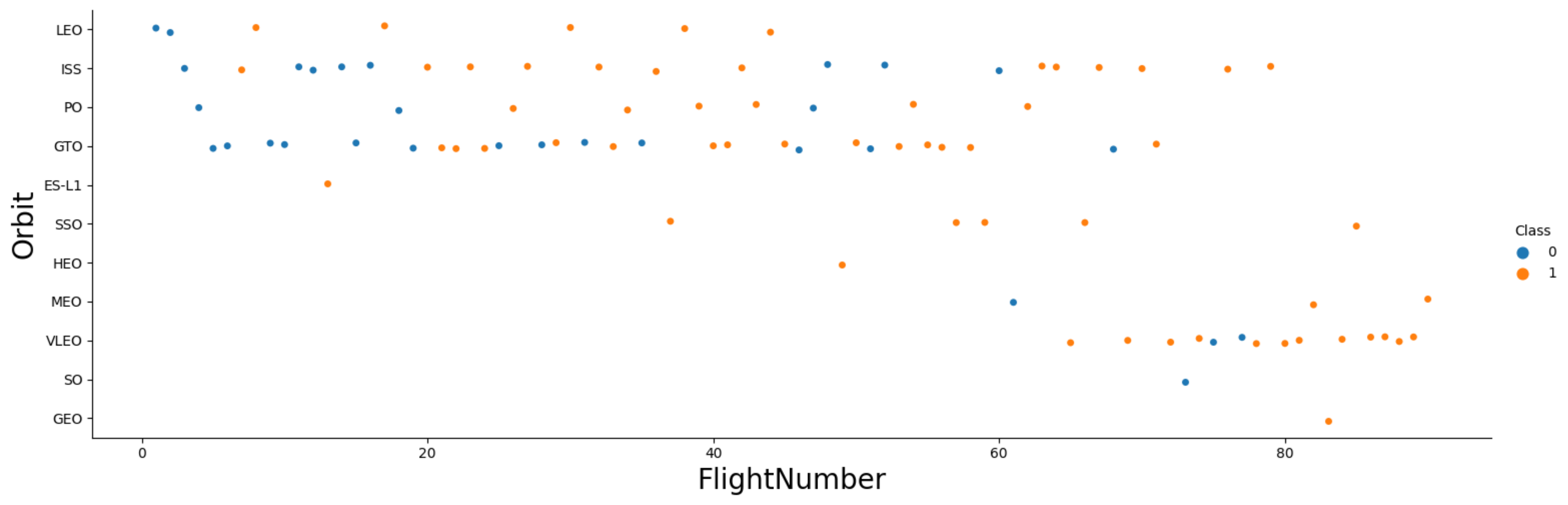
- For LaunchSite CCAFS SLC 40, there is increase in success rate with increase in Payload Mass
- For LaunchSite VAFB-SLC 4F there are no launches for payload greater than 10000

Success Rate vs. Orbit Type

- Orbit ES-L1, GEO, SSO, and HEO has the highest success rate

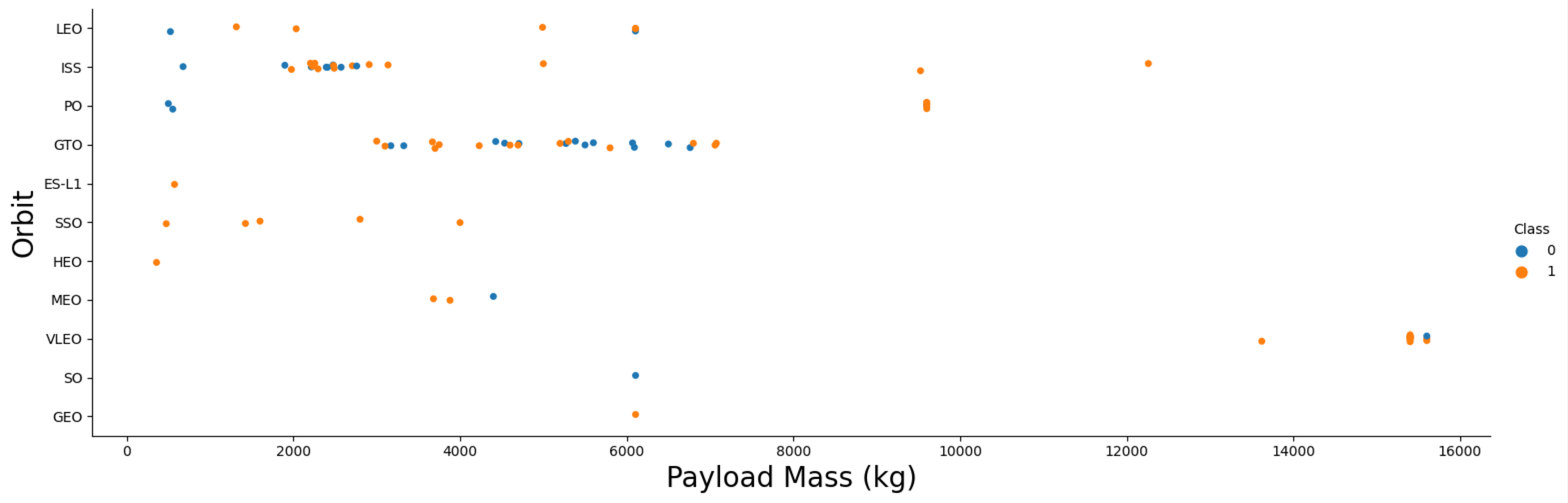


Flight Number vs. Orbit Type



- For LEO Orbit, Success rate appears to increase with flight number
- While for GTO there seems to be no relationship between Orbit and flight number
- Overall Flight number against Orbit is not a good metrics to predict success rate

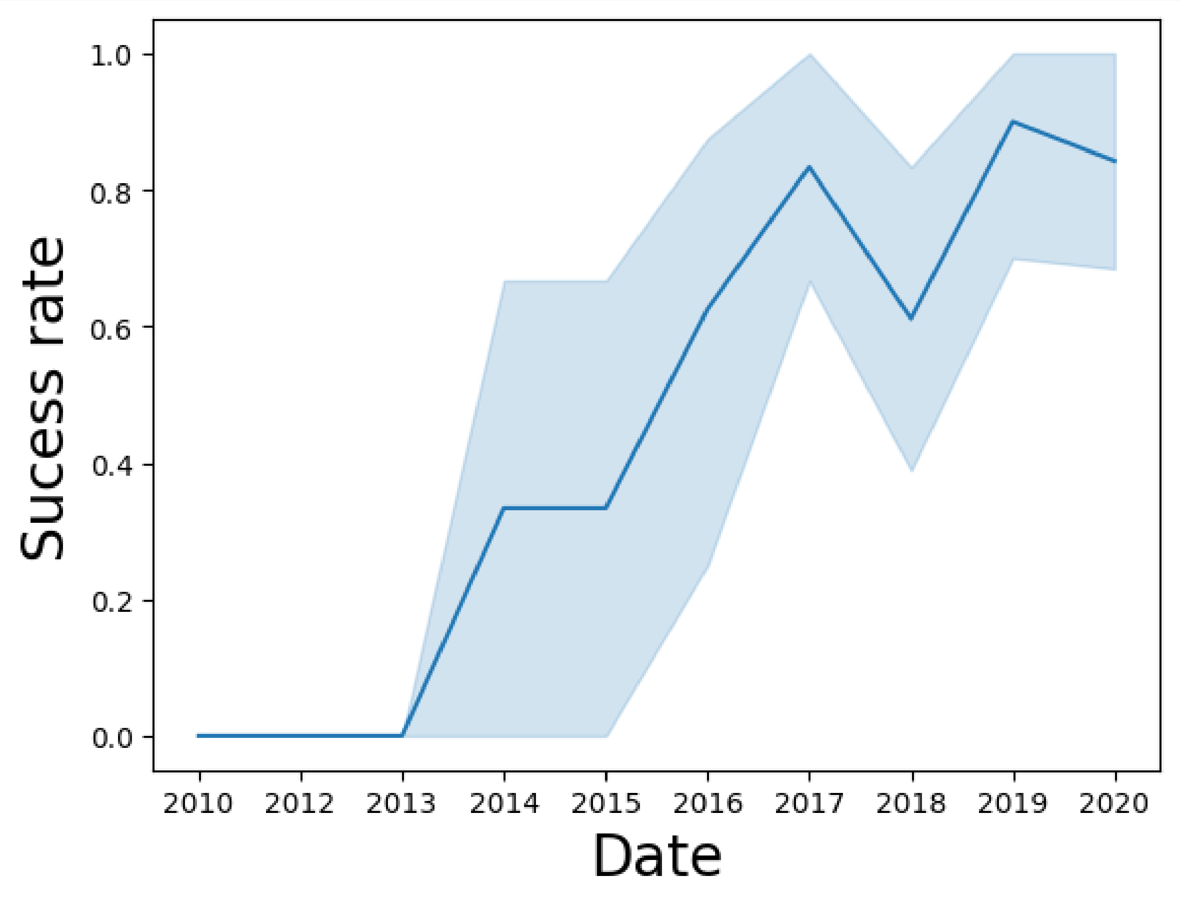
Payload vs. Orbit Type



- There is increase in successful landing rate for PO, LEO, ISS orbit
- There is no distinguishable difference in payload mass and orbit for GTO

Launch Success Yearly Trend

- There is an increase in the yearly success rate till 2020



All Launch Site Names

- We query the database to show all LaunchSites by using DISTINCT

```
[5]: %sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL

 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

```
%sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

[11]

```
* sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- We use the LIKE to get LaunchSite that has CCA
- Then we use the LIMIT to 5

Total Payload Mass

```
[14]  %sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Customer = 'NASA (CRS)';
```

```
 * sqlite:///my_data1.db
Done.
```

SUM(PAYLOAD_MASS__KG_)
45596

- We use SUM to get total payload mass and WHERE to filter for only NASA

Average Payload Mass by F9 v1.1

```
[15]  %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1';

 * sqlite:///my_data1.db
Done.
```

AVG(PAYLOAD_MASS__KG_)
2928.4

- We use AVG to get average payload and WHERE to filter to version F9

First Successful Ground Landing Date

```
%sql SELECT min(Date) FROM SPACEXTBL WHERE TRIM(`Landing _Outcome`) = 'Success (ground pad)';
```

[54]

```
* sqlite:///my_data1.db
Done.
```

min(Date)
01-05-2017

- We use MIN to get the first date WHERE landing outcome is successful

Successful Drone Ship Landing with Payload between 4000 and 6000

```
[53]  %sql SELECT Booster_Version FROM SPACEXTBL WHERE `Landing _Outcome` = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000

 * sqlite:///my_data1.db
Done.
```

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

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

- We use WHERE clause combined with AND to get the results

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT COUNT(Mission_Outcome) FROM SPACEXTBL WHERE Mission_Outcome = 'Success' OR Mission_Outcome = 'Failure (in flight)';
```

```
 * sqlite:///my_data1.db
Done.
```

COUNT(Mission_Outcome)
99

- We use COUNT and the WHERE combined with OR to get our outcome

Boosters Carried Maximum Payload

```
[59]
%sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);

 * sqlite:///my_data1.db
Done.
```

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

- We use a subquery to get the max payload version from which we the SELECT the Booster Version

2015 Launch Records

```
[62] %sql SELECT substr(Date, 4, 2), `Landing _Outcome`, Booster_Version, Launch_Site FROM SPACEXTBL WHERE substr(Date, 7, 4) = '2015' AND `Landing _Outcome` = 'Failure (drone ship)';

 * sqlite:///my_data1.db
Done.
```

substr(Date, 4, 2)	Landing _Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- We use SUBSTR to extract date and use WHERE to filter and get our result

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

```
%%sql SELECT `Landing _Outcome`, COUNT(`Landing _Outcome`)
FROM SPACEXTBL WHERE `Landing _Outcome` like 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017')
ORDER BY COUNT(`Landing _Outcome`) DESC;
```

[22] ✓ 0.5s

```
 * sqlite:///my_data1.db
Done.
```

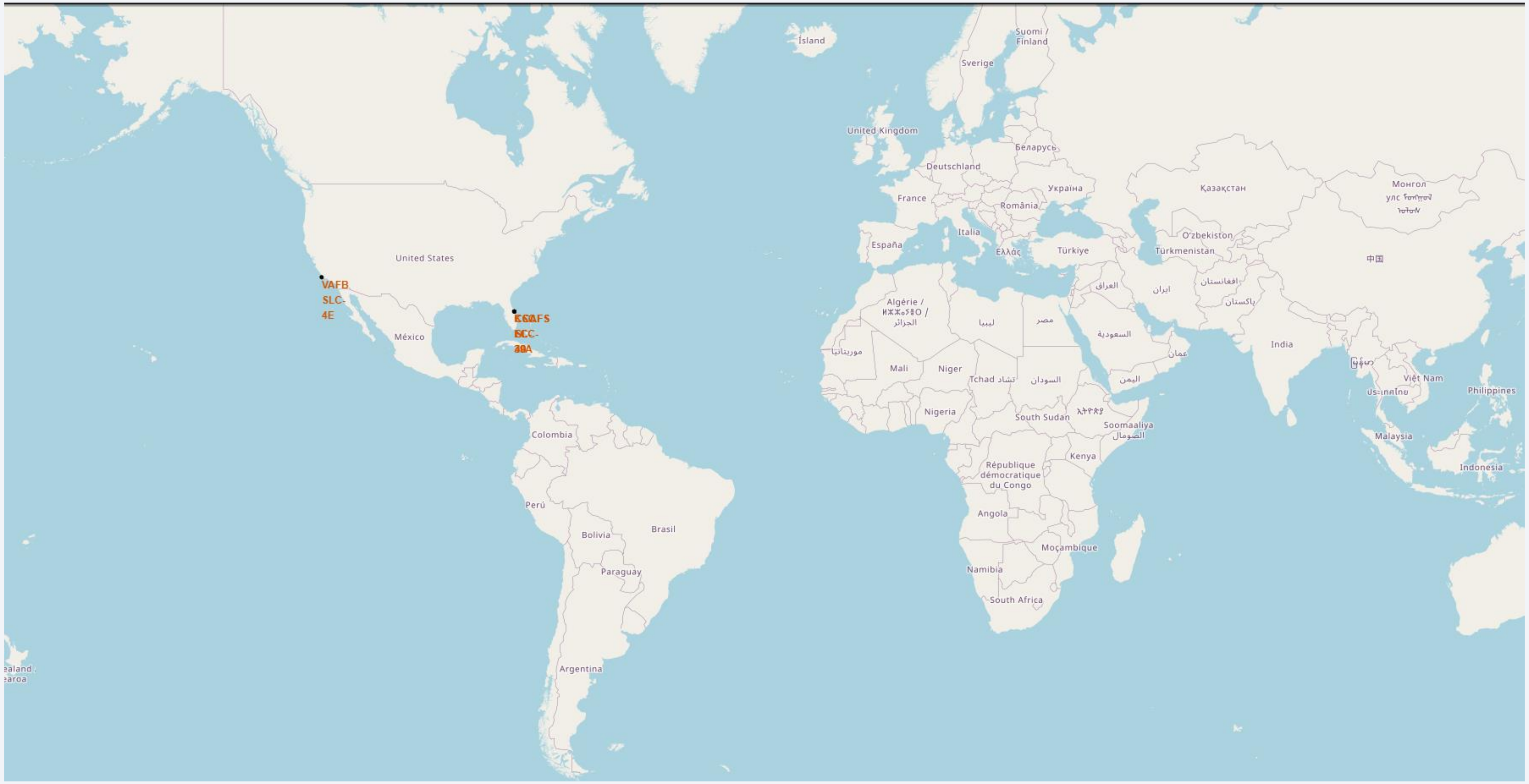
Landing _Outcome	COUNT(`Landing _Outcome`)
Success (drone ship)	34

- We used the COUNT and ORDER BY to get our outcome

Section 3

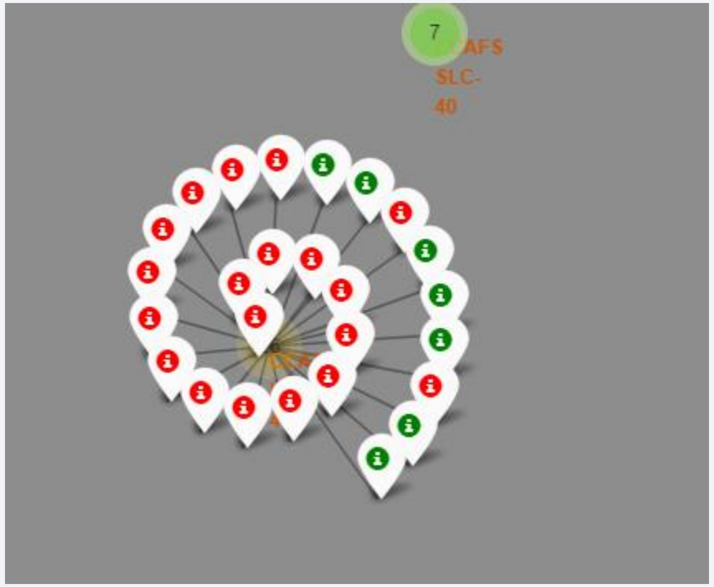
Launch Sites Proximities Analysis

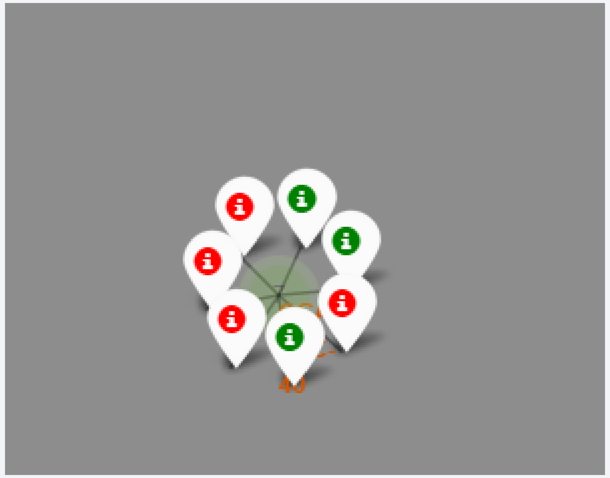
Launch Site Location



- Space X Launch Sites are located in the United States of America

Success and Failed Launches on Map

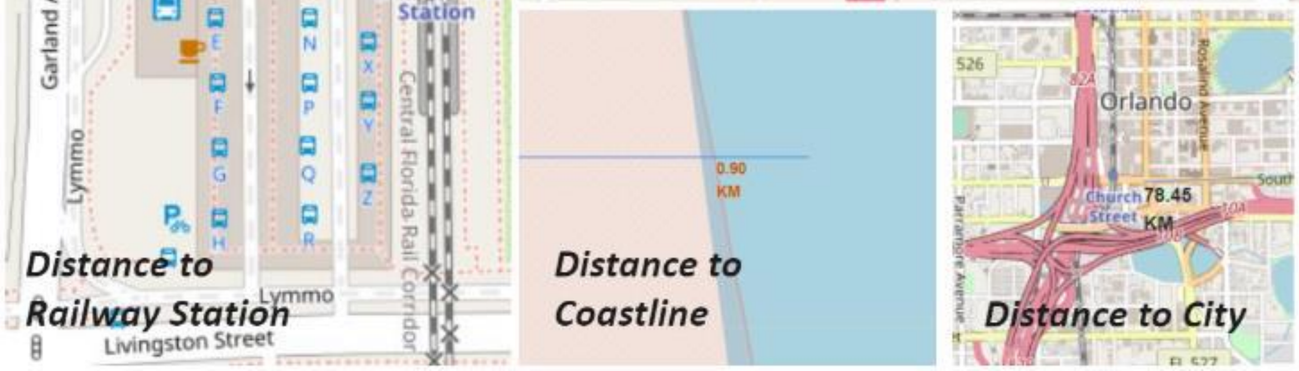




Green – Successful Launches

Red – Failed Launches

Launch Sites Distance from Infrastructure

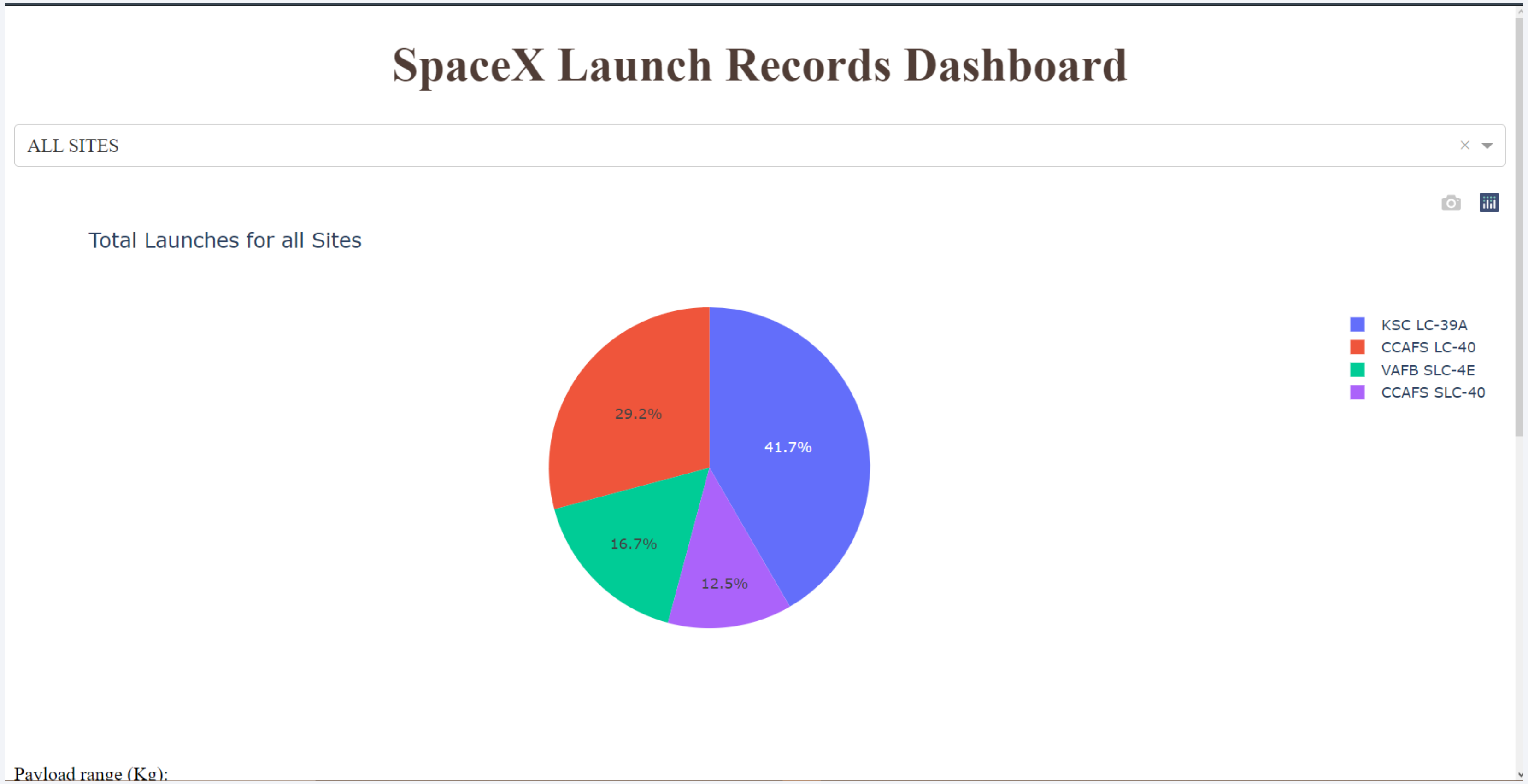


- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Section 4

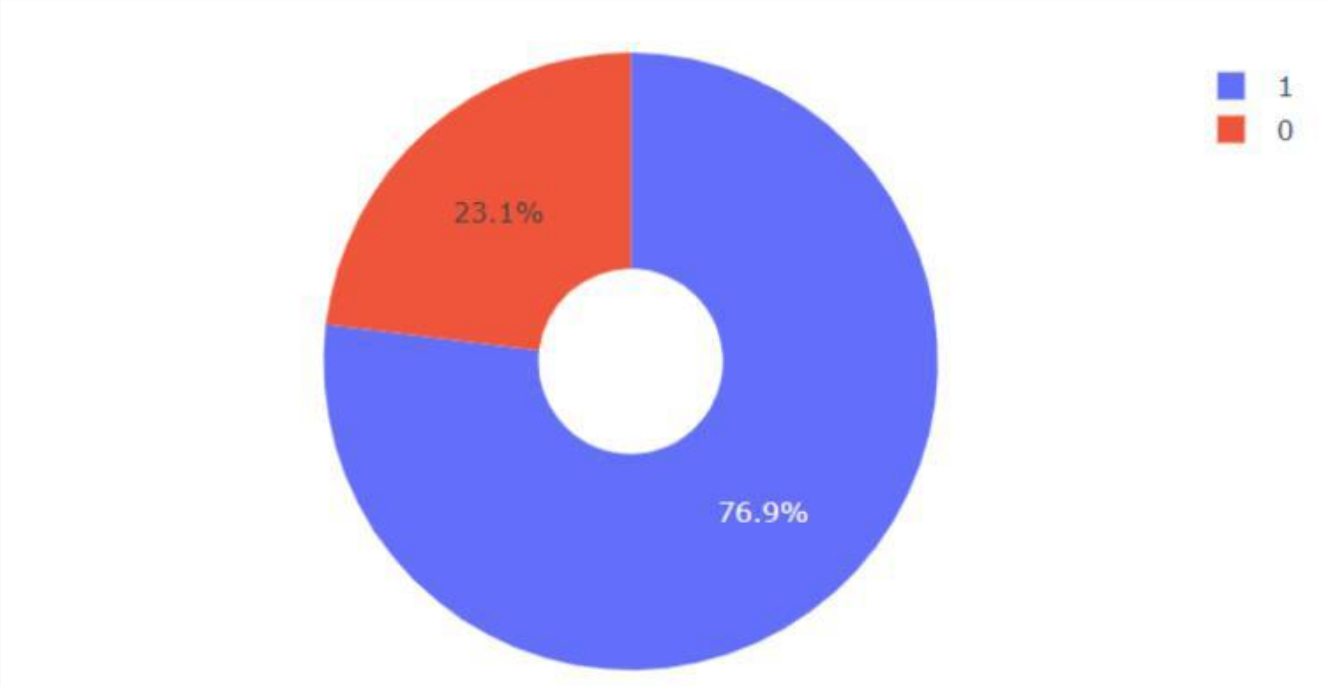
Build a Dashboard with Plotly Dash

Total successful Launches



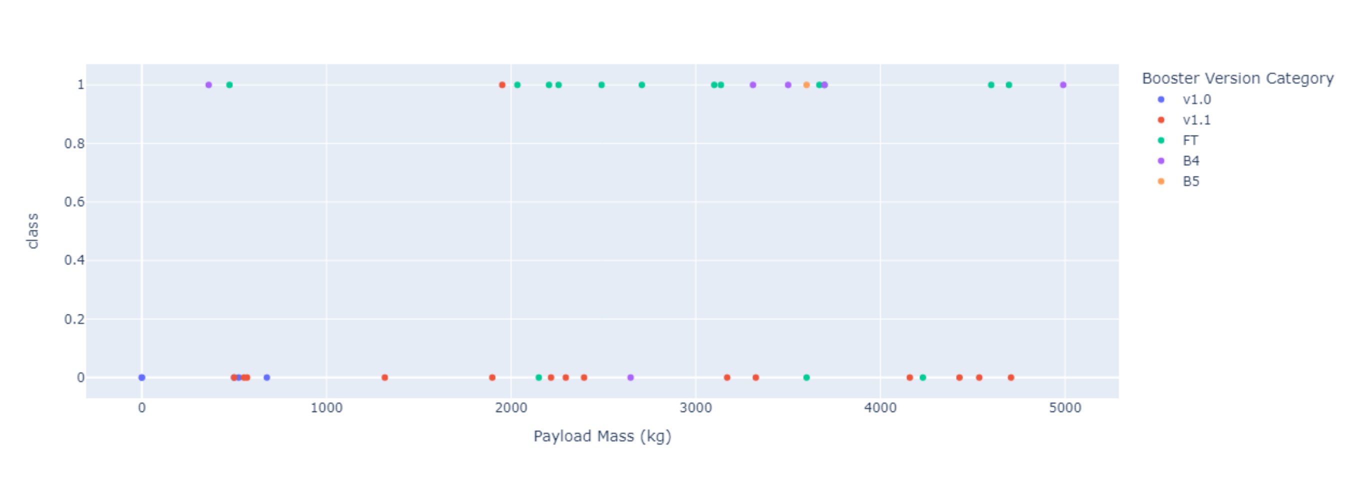
- KSC LC – 39A has the best successful launches with 41.7%

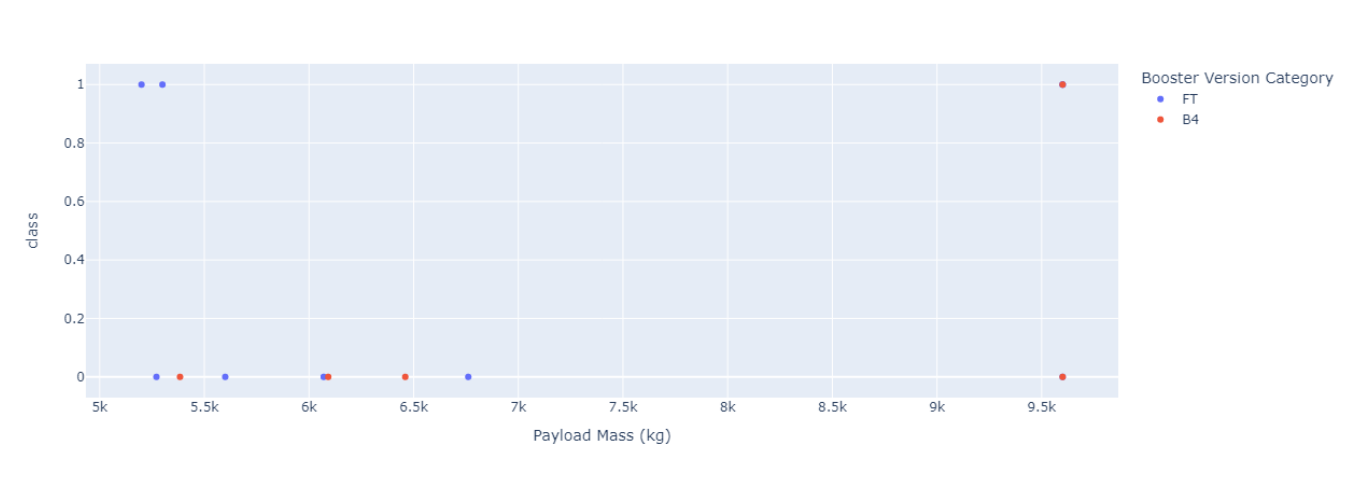
Highest Success Launches



- KSC LC-39A had a success rate of 76.9% and a failure rate of 23.1%

Payload vs Launch Outcome





- Payload range was divided into 2, 0 to 5000kg and 5000kg to 10000kg
- Success rate for low payload mass is greater than for heavy payload mass

Section 5

Predictive Analysis (Classification)

Classification Accuracy

```python
## best model

models = {
    'KNN' : knn_cv.best_score_,
    'DecisionTree' : tree_cv.best_score_,
    'LogisticR' : logreg_cv.best_score_,
    'SVM' : svm_cv.best_score_
}

bestalgo = max(models, key=models.get)

print ('Best Model is', bestalgo, 'with a score of', models[bestalgo])
```

[32] ✓ 0.4s

```
Best Model is DecisionTree with a score of 0.8767857142857143
```

- The DecisionTree models has the highest accuracy

Confusion Matrix

- The confusion matrix for Decision Tree model is shown here. It has 4 wrong predictions for did not land outcome

Conclusions

We can conclude that

- With increasing flight amount, there is increase in success rate at a Launch Site
- Launch success rate started to increase in 2013 till 2020.
- 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

Appendix

Thank you!