IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

KHALIL NAIT BOUZID
29/12/2021

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

Summary of methodologies

- Data collection via API, SQL and Web Scraping
- Data wrangling and Analysis
- EDA with data visualization and EDA with SQL
- Building an Interactive map with folium
- Building a Dashboard with Plotly Dash
- Predictive Analysis - classification

Summary of all results

- Exploratory data analysis results
- Interactive Visualizations in screenshots
- Predictive Analysis results

Introduction

- ***Project background and context***

The aim of this project is to predict if the falcon 9 first stage will land successfully. SpaceX advertises Falcon 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 the launch. This information can used if an alternate company wants to bid against SpaceX for a rocket launch.

- ***Problems we want to find answers***

1. *Which factors influences if the rocket will land successfully?*
2. *What is the effect of each relationship of rockets variables on outcomes?*
3. *What are the Conditions which will help SpaceX to achieve the best results?*

Section 1

Methodology

Methodology

Executive Summary

1. Data collection methodology:

- *SpaceX Rest API*
- *Web Scrapping from wikipedia*

2. Perform data wrangling

- *One Hot Encoding data fields for Machine learning and dropping irrelevant columns*

3. Perform exploratory data analysis (EDA) using visualization and SQL

- *Plotting : Scatter and bar graphs to show relationship between variables and to show patterns of data*

4. Perform interactive visual analytics using Folium and Plotly Dash

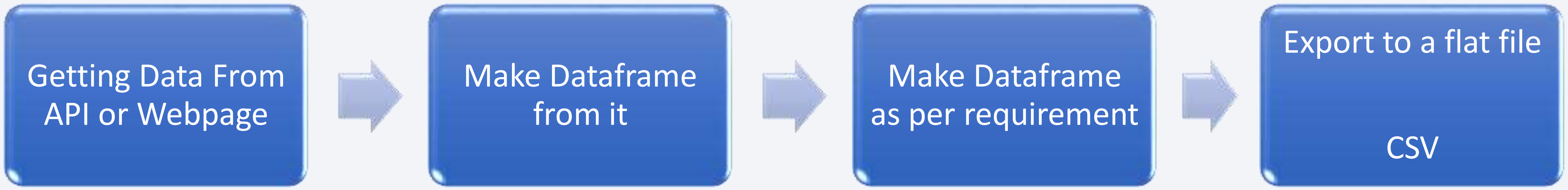
- *Using Folium and Plotly Dash visualization*

5. Perform predictive analysis using classification models

- *Build, tune, evaluate classification models*

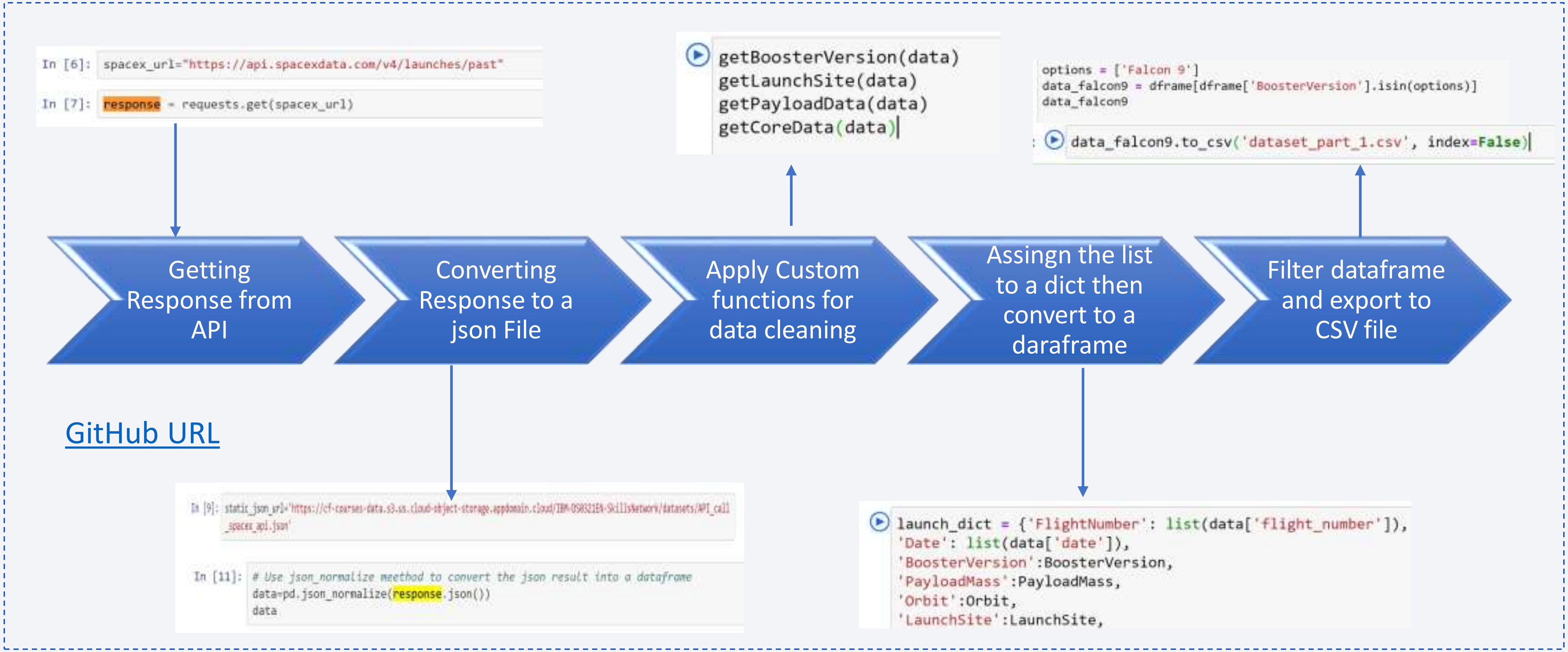
Data Collection

- Datasets Collection Description



[GitHub URL]

Data Collection – SpaceX API



Data Collection - Scraping

Getting Resoponse from HTML

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

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

Creating Beautiful Soup Object

```
soup = BeautifulSoup(data, 'html5lib')
```

Finding Tables

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

Getting Column names

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
ths = first_launch_table.find_all('th')
for th in ths:
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

Creation of dict and appending data to keys

```
launch_dict= dict.fromkeys(column_names)
```

Converting dict to a dataframe

Dataframe to CSV

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

[GitHub URL]

Data Wrangling

- Data wrangling refers to the process of **collecting raw data, cleaning it, mapping it, and storing it in a useful format**.

GitHub URL

Data Wrangling

Calculate the nbre of launches at each site

Calculate the nbre and occurrence of each orbit

Calculate the nbre and occurrence of mission outcpme per orbit type

Creating a landing outcome label from outcome column

Export to a data to a CSV file

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

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
```

```
# Apply value_counts on Orbit colu
df["Orbit"].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
HEO       1
SO        1
GEO       1
ES-L1     1
```

```
landing_outcomes = df["Outcome"].value_counts()
landing_outcomes
```

```
]:  True ASDS      41
    None None      19
    True RTLS      14
    False ASDS      6
    True Ocean      5
    False Ocean     2
    None ASDS       2
    False RTLS      1
```

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

```
df.to_csv("dataset_part_2.csv", index=False)
```

[GitHub URL]()

EDA with Data Visualization

- In data analytics, exploratory data analysis is how we describe the practice of **investigating a dataset and summarizing its main features**. It is a form of descriptive analytics. EDA aims to spot patterns and trends, to identify anomalies, and to test early hypotheses.

GitHub URL

EDA with Data Visualization

1. ***Scatter Graphs:***

Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation.

- ***Flight Number and Payload***
- ***Flight Number and Launch Site***
- ***Payload and Launch Site***
- ***Orbit Type VS. Flight Number***
- ***Payload VS. Orbit Type***

2. Bar Graph :

A bar diagram makes it easy to compare sets of data between different groups at a glance. The graph represents categories on one axis and a discrete value in the other. The goal is to show the relationship between the two axes. Bar charts can also show big changes in data over time.

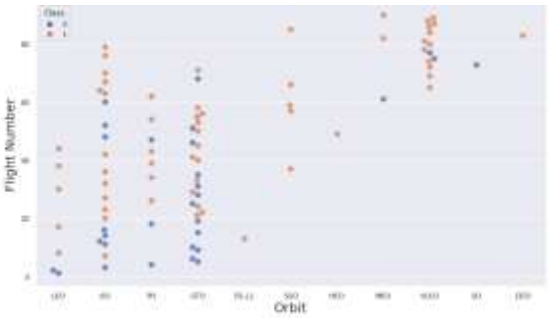
- ***Success Rate and Orbit Type***

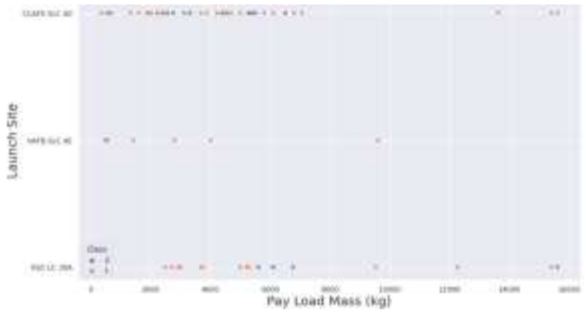
3. Line Graph:

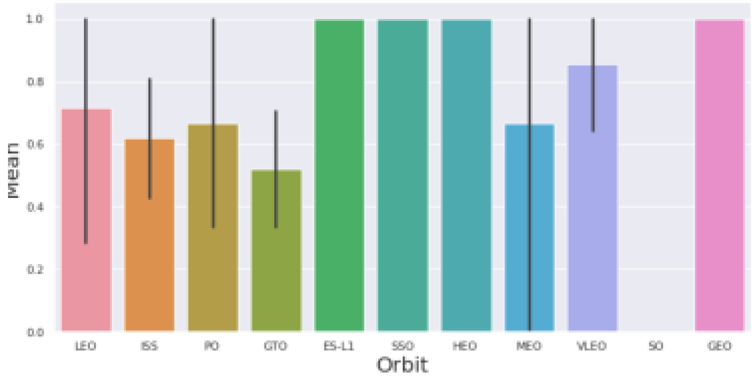
Line graphs are useful in that they show data variables and trends very clearly and can help to make predictions about the results of data not yet recorded –

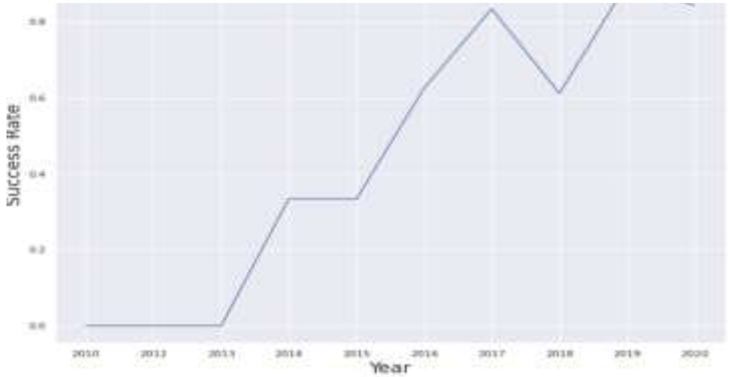
- ***Launch Success Rate and Year***

[GitHub URL]()









EDA with SQL

SQL is a crucial tool for Data Scientist as most of data is stored in databases. It is an incredibly powerful tool for analyzing data and drawing useful insights. For this purpose, we will use the **IBM’s Bb2 for Cloud**.

These are the SQL querries that were performed:

- *Displaying the names of the unique launch sites in the space mission*
- *Displaying 5 records where launch sites begin with the string ‘CCA’*
- *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 successful 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*
- *Listing the records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2015*
- *Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order*

GitHub URL

Build an Interactive Map with Folium

Folium makes it easy to visualize manipulated data in python on an interactive leaflet map. We use the latitude and longitude coordinates for each launch site and added a Circle Marker around each launch site with a label of the name of the launch site. Also, it is easy to visualize the number of success and failure for each launch site with green and Red markers on the map.

GitHub URL

Map Object	Code	Result
Map Marker	**Folium.Marker(**	Map Object to make a mark on map
Icon Marker	**Folium.Icon(**	Create an icon on map
Circle Marker	**Folium.Circle(**	Create a circle where marker is being paced
PolyLine	**Folium.PolyLine(**	Create a line between points
Marker Cluster Object	**Folium.Cluster(**	To simplify a map including many markers having the same coordinates
AnPath	**Folium.plugins.Anpath(**	Create an animated line betwen points

Build a Dashboard with Plotly Dash

Map Objects	Code	Result
Dash and its components	**Import dash Import dash_html_components as html Import dash_core_components as dcc From dash.dependencies import Input, Output**	Plotly python's leading data viz and Ui libraries. With dash open source, Dash apps run on your local laptop or server. The dash core components library contain a set of higher level components like slider, graph, dropdown and tables. Dash provides all html tags.
Pandas	**Import pandas as pd**	Fetching values from CSV and creating dataframe
Plotly	**Import plotly.express as px**	Plot the graoghs with interactive plotly library
Dropdown	**dcc.Dropdown(**	Crate dropdown for lauch sites
Rangeslider	**dcc.RangSlider(**	Create a rangeslider for Payload Mass detection
Pie chart	**Px.pie(**	Creating pie graph for Sucess percentage display
Scatter chart	**Px.scatter(**	Creating scatter grapgh for coorelation display

Predictive Analysis (Classification)

Building Model

- Load feature engineered data into datframe
- Transform it into Numpay Arrays
- Standardize and transform data
- Split data into training and test data sets
- Check how many test samples has been created
- List down ML algo we want to use
- Set parameters and algo to GridSearchCV
- Fit our datasets into GridSearchCV objects and train our model

Model Evaluation

- Check accuracy for each model
- Get best hyperparameters for each type of algo
- Plot Confusion Matrix

Findind Best Performing classifiacttion Model

- The model with the best accuracy score is the best performing model

Best Model

```
y = data['Class'].to_numpy()
transform = preprocessing.StandardScaler()
X = transform.fit_transform(X)
X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size=0.2, random_state=2)
Y_test.shape
```


[GitHub URL]

Results

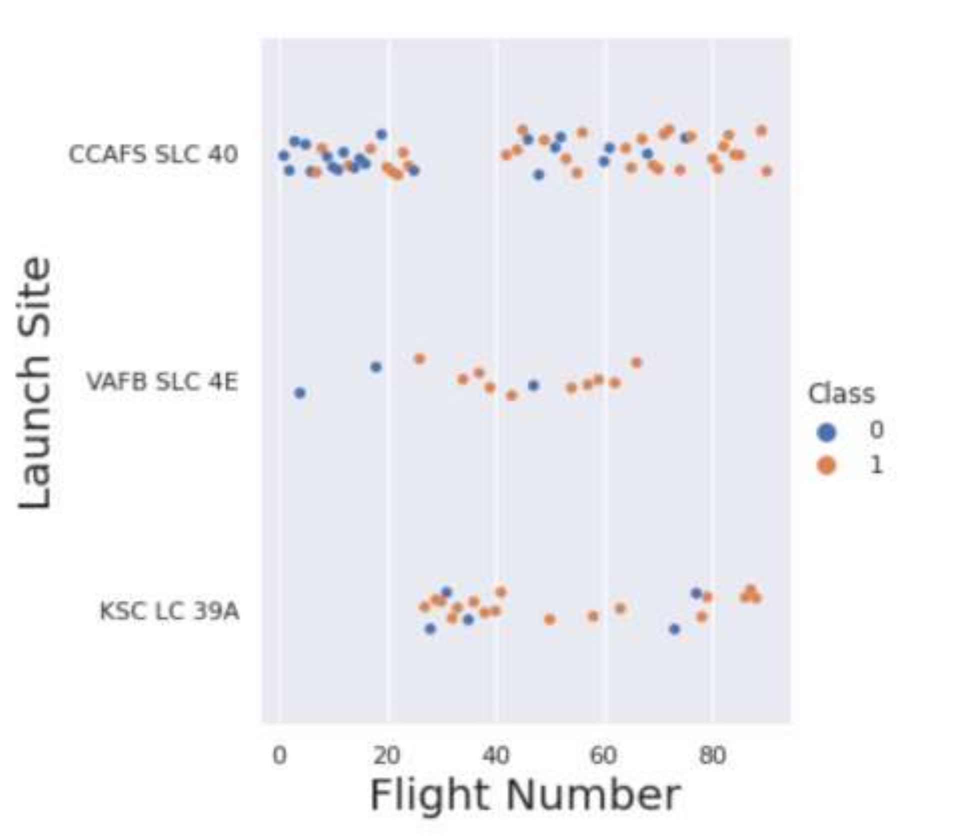
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

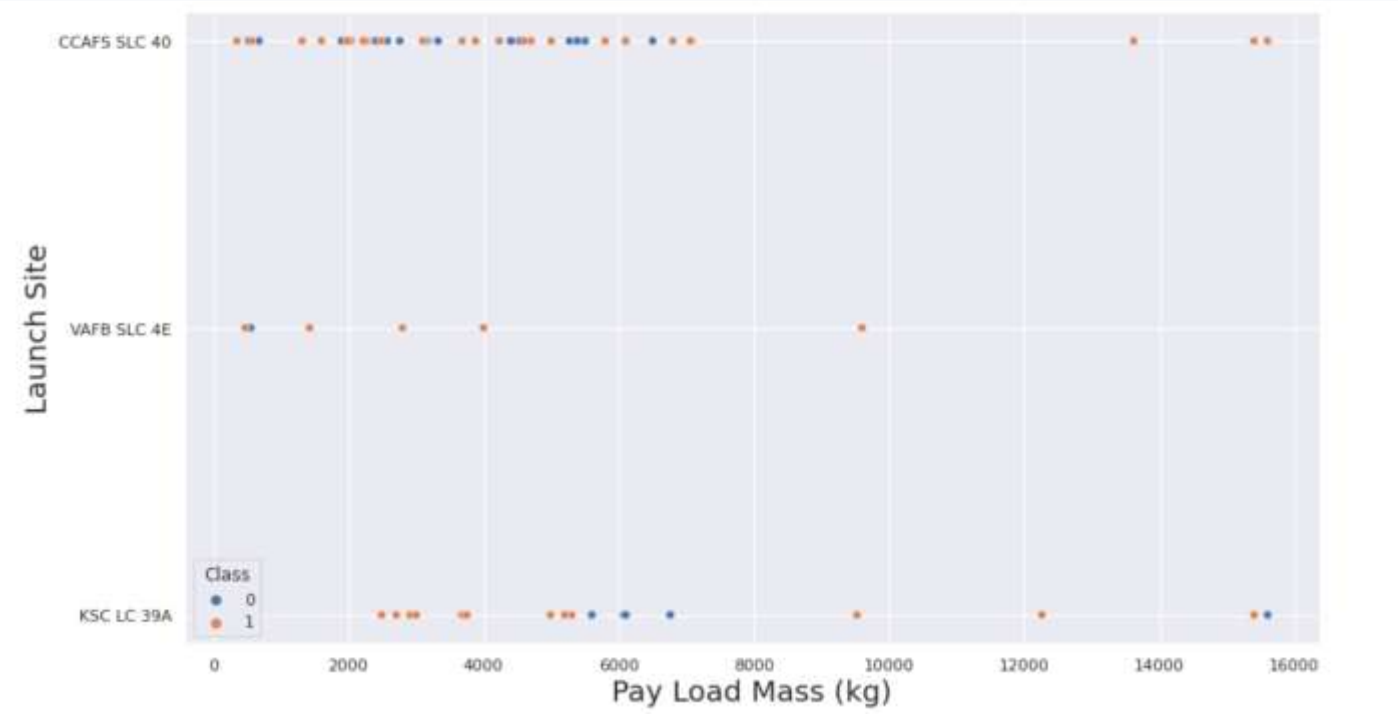
With more flight numbers (after 40) higher the success rate for the Rocket is increasing. However, theres no clear pattern to make a decision if the Flight Number is dependant on Launch Site for a success launch.



[GitHub URL]

Payload vs. Launch Site

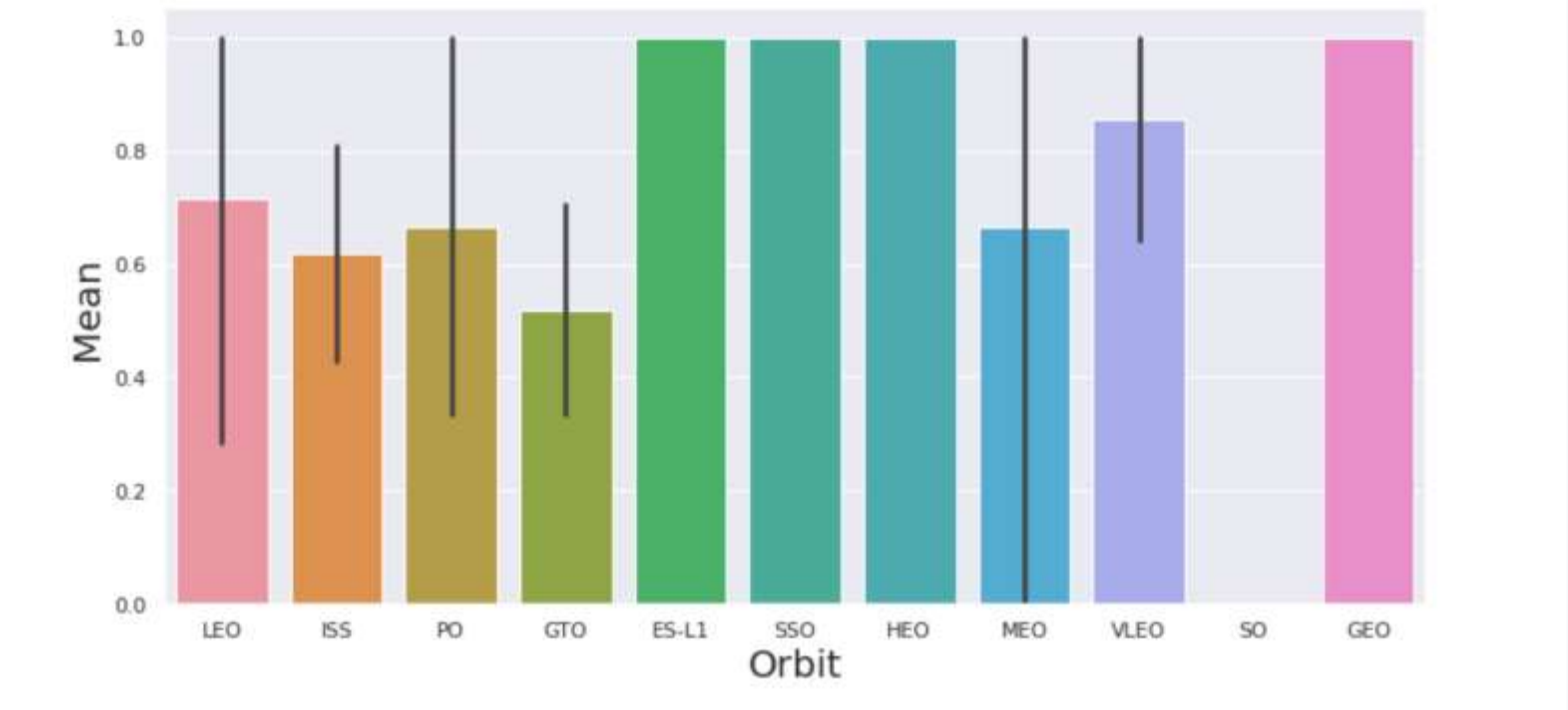
The greater the payload mass (greater than 8000), the higher the success rate for the Rocket. However, there is no clear pattern to make a decision if the Launch Site is dependent on Pay Load Mass for a success launch.



[GitHub URL]

Success Rate vs. Orbit Type

ES-L1, GEO, HEO, SSO has highest Success rates. SO has poorest.



GitHub URL

Flight Number vs. Orbit Type

We see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

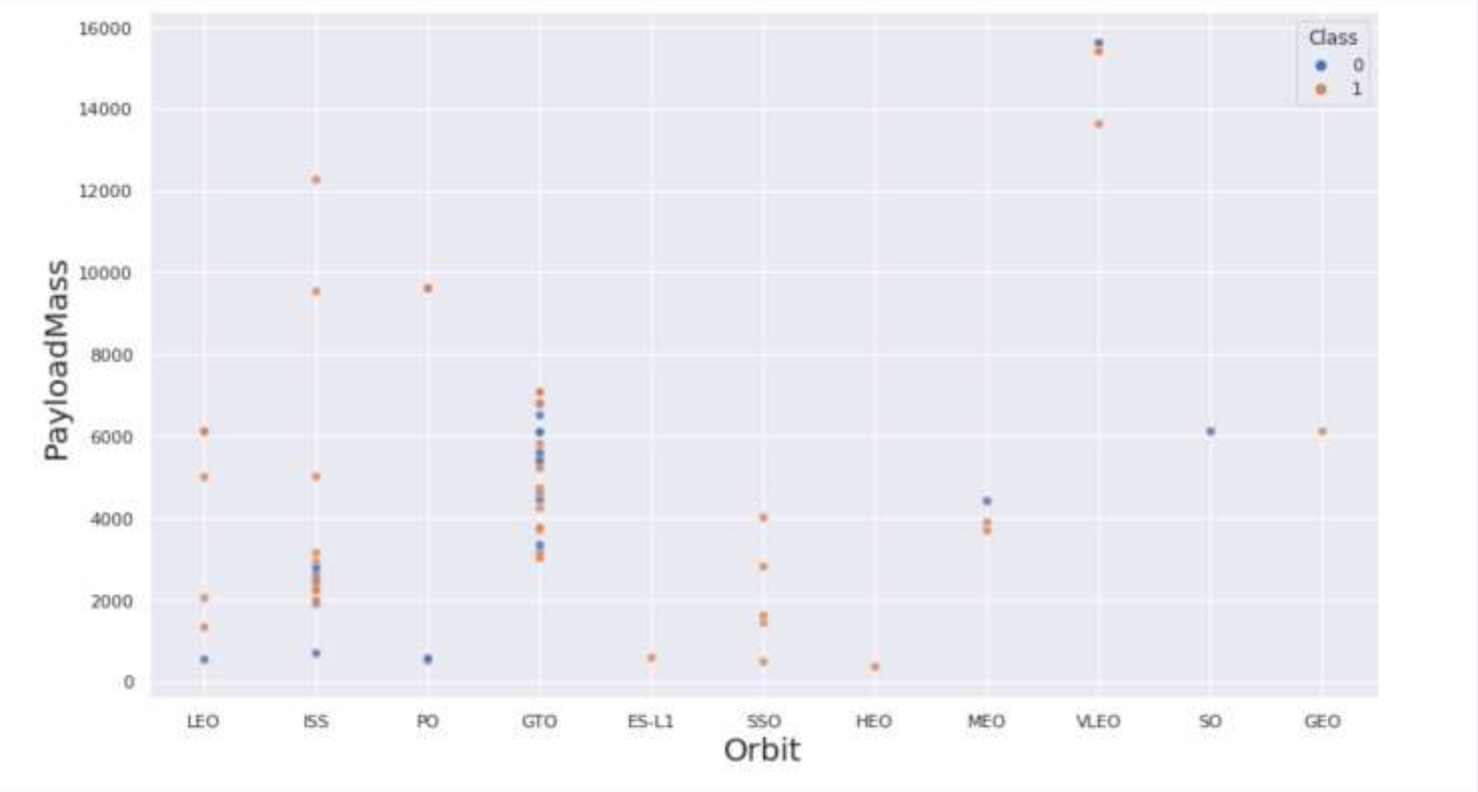
GitHub URL

Payload vs. Orbit Type

With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.

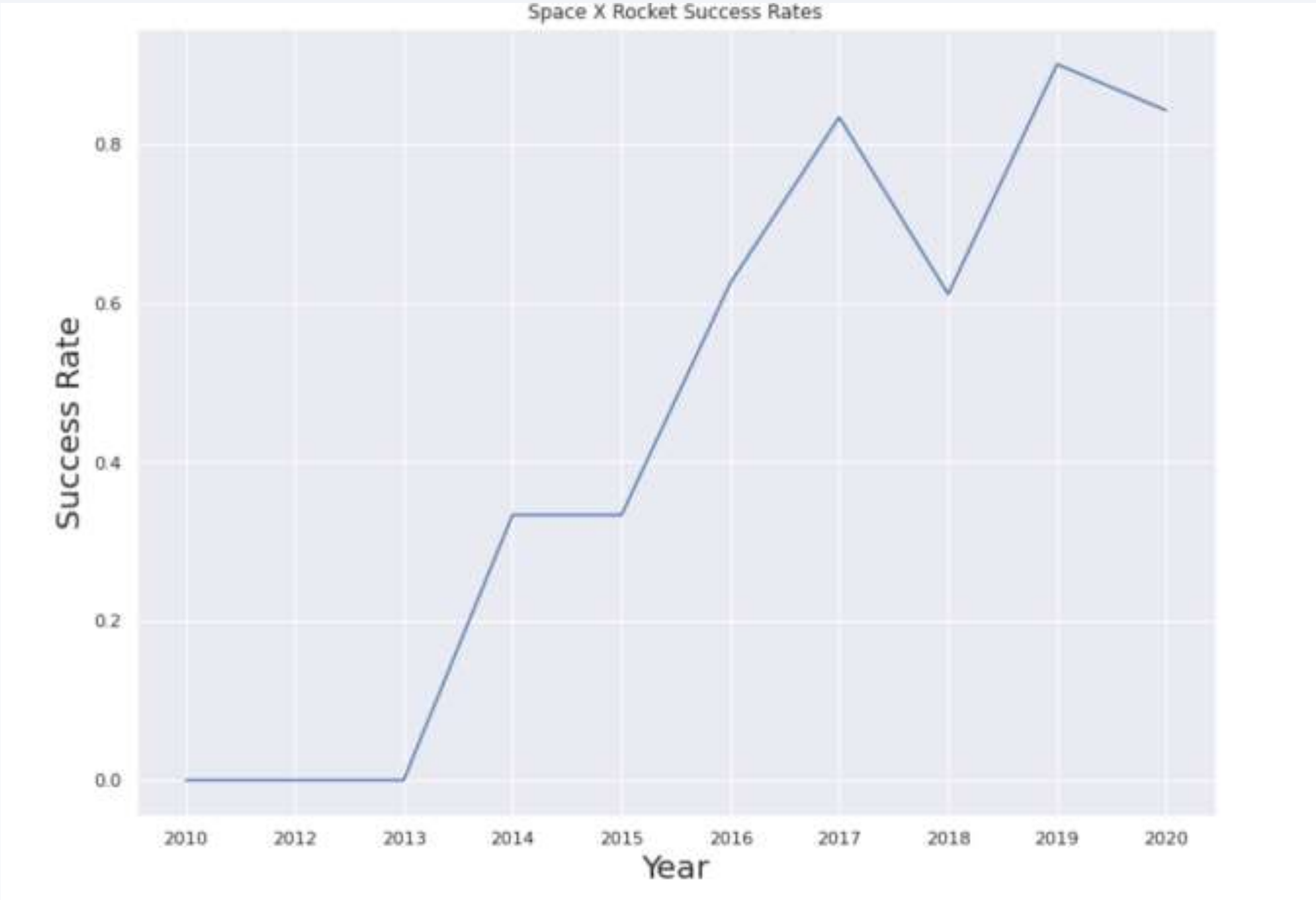
However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

[GitHub URL]



Launch Success Yearly Trend

We can observe that the success rate since 2013 kept increasing till 2020.



GitHub URL

All Launch Site Names

SQL Query

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXDATA;

 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41
Done.
```

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

Description

Using the word ***distinct*** in the query will pull the unique values for the launch Site column from the table SPACEXDATA

GitHub URL

Launch Site Names Begin with 'CCA'

SQL Querry

```
%sql SELECT * FROM SPACEXDATA WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

```
* ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	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)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Description

Using keyword ***"Limit 5"*** in the query will fetch 5 records from table SPACEWDATA, condition ***LIKE*** keyword with wild card ***"CCA%".*** The percentage in the end suggest that the launch_site name must start with CCA.

[GitHub URL]

Total Payload Mass

SQL Query

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)" FROM SPACEXDATA WHERE CUSTOMER = 'NASA (CRS)';
```

```
 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286/blu
Done.
```

Total Payload Mass by NASA (CRS)
45596

Description

Using function ***SUM*** to select the total in the column PAYLOAD_MASS_KG and the ***WHERE*** clause filters the dataset to only perform calculations on customer ***NASA (CRS)***

GitHub URL

Average Payload Mass by F9 v1.1

SQL Query

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster Version F9 v1.1" FROM SPACEXDATA \
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

```
 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:3
Done.
```

Average Payload Mass by Booster Version F9 v1.1
2928

Description

Using the function ***AVG*** works out the average in the column PAYLOAD_MASS_KG The **WHERE** clause filters the dataset to only perform calculations on Booster_version ***f9 v1.1***

GitHub URL

First Successful Ground Landing Date

SQL Query

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad" FROM SPACEXDATA \
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databas
Done.
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

Description

The function ***MIN*** works out the minimum date in the column date while the ***WHERE*** clause filters the dataset to only perform calculations on Landing_outcome **success(ground pad)**

GitHub URL

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL Query

```
%sql SELECT BOOSTER_VERSION FROM SPACEXDATA WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

```
 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.
Done.
```

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

Description

Selecting only Booster_Version*.* ***WHERE*** clause filters ***AND*** clause specifies additional filter.

GitHub URL

Total Number of Successful and Failure Mission Outcomes

SQL Query

```
%sql SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "Successful Mission", \
    sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "Failure Mission" \
FROM SPACEXDATA;
```

```
 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.clouc
Done.
```

Successful Mission	Failure Mission
100	1

Description

Selection ***multiple count*** is regarded as a complex query. We used a case clause within sub query for getting success and failures counts in same query. Case when ***MISSION_OURCOME LIKE '%Success%' then 1 else 0 end"*** returns value which sum to get the result needed.

GitHub URL

Boosters Carried Maximum Payload

SQL Query

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEXDATA \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXDATA);
```

```
 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286
Done.
```

Booster Versions which 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

Description

Using the function ***MAX*** works out the maximum payload in the column PAYLOAD_MASS_KG_ in the sub query and ***WHERE*** clause filters Booster Version which had that maximum payload.

[GitHub URL]

2015 Launch Records

SQL Query

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXDATA WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

```
 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.dat
Done.
```

booster_version	launch_site
F9 v1.1 B1012	CCAFS LC-40
F9 v1.1 B1015	CCAFS LC-40

Description

We use DATE in where clause and like ‘2015-%’ to select all 2015 launch records with a failed landing outcome

GitHub URL

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

SQL Query

```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEXDATA \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING__OUTCOME \
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

```
 * ibm_db_sa://ksk00244:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.clou
Done.
```

Landing Outcome	Total Count
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Description

Selecting only LANDING_OUTCOME, **WHERE** clause filters DATE between ***2010-06-04 and 2017-03-20*** and grouping by LANDING_OUTCOME , **orderd by COUNT(LANDING_OUTCOME)** in Descending order.

GitHub URL

Section 4

Launch Sites Proximities Analysis

All Launch Sites on Folium Map

According to the map: The SpaceX launch sites are in the united states of America coast. Florida and California.

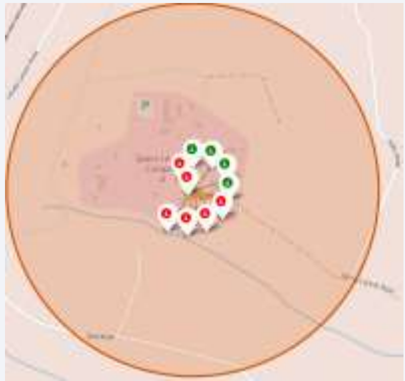
GitHub URL

Color Labeled Launch Record

- Green Marker shows successful launches and Red Marker shows failures. According to the screenshots, we can deduct that ***KSC LC-39A*** has the maximum probability of success.
- 

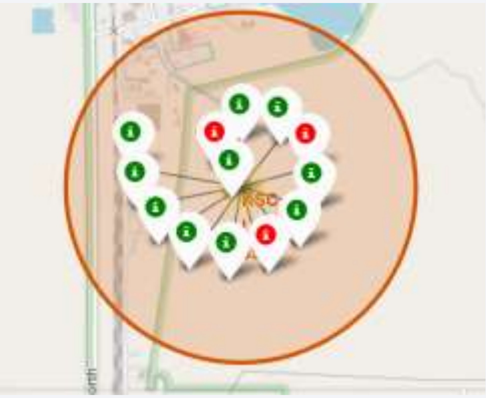
[GitHub URL]

VAFB SLC -4E



CCAFS LC-40

KSC LC-39A



CCAFS SLC-40

Launch Sites Distances from Equator and Railways

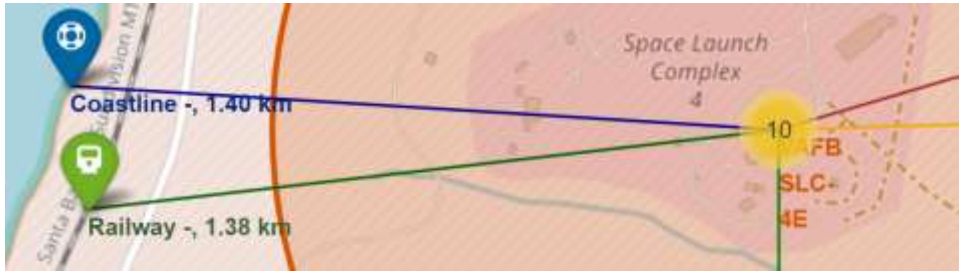
- Distance from Equator is greater than 3000km for all sites

- Distance from Railways for all sites are greater than 0.7 km. Meaning that launch sites are not away from railway tracks.

[GitHub URL]

Launch Sites Distances from Coastlines and Cities

- Distance for all launch sites from coastlines is less than 4 Km.

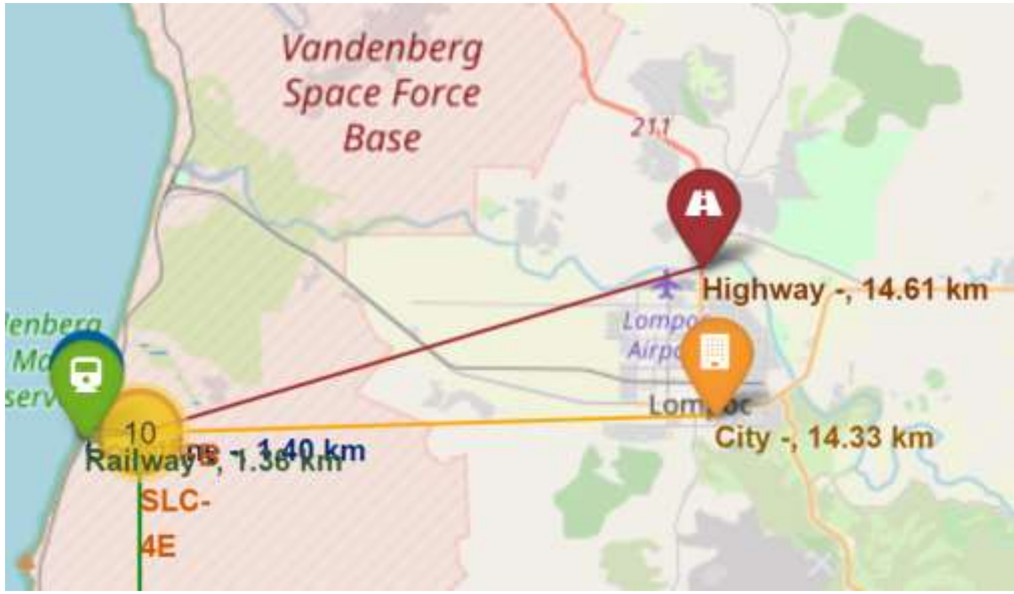


- Distance for all launch sites from cities is greater than 14 km for all sites. So, launch sites are far away from cities.

[GitHub URL]

Launch Sites Distances from Highways

- Launch sites are relatively far from highways.



[GitHub URL]

Conclusion with Folium Results

- ***Are all launche sites in proximity to the Equator line?***

No, (4000 km<distance<3000km)

- ***Are launch sites in close proximity to railways?***

Yes, (0.5km<distance<2km)

- ***Are launch sites in close proximity to highways?***

No, (5km<distance<15km)

- ***Are launch sites in close proximity to coastline?***

Yes, (0.5km<distance<5km)

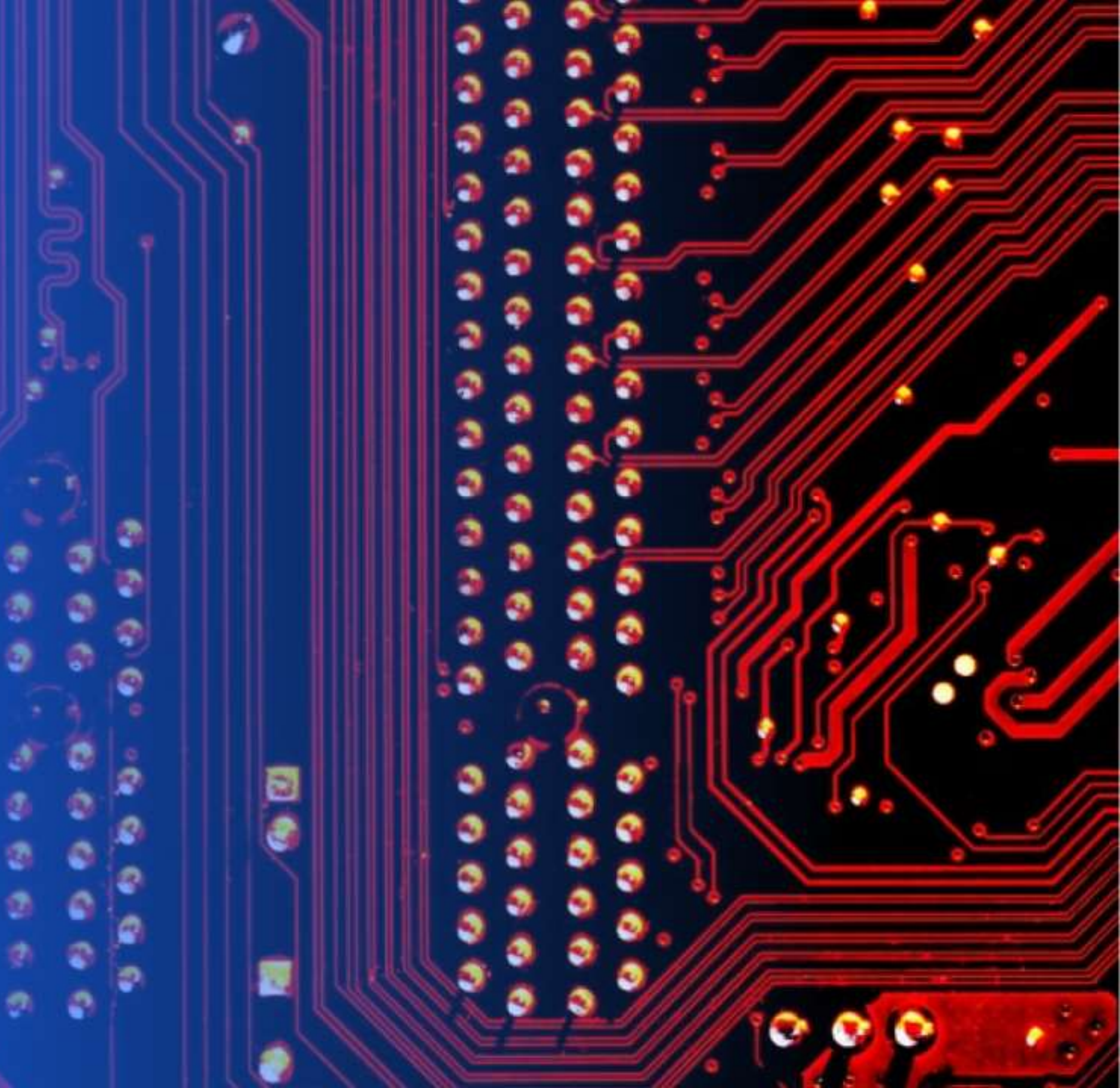
- ***Do launch sites keep certain distance away from cities?***

Yes, (15km<distance<80km)

GitHub URL

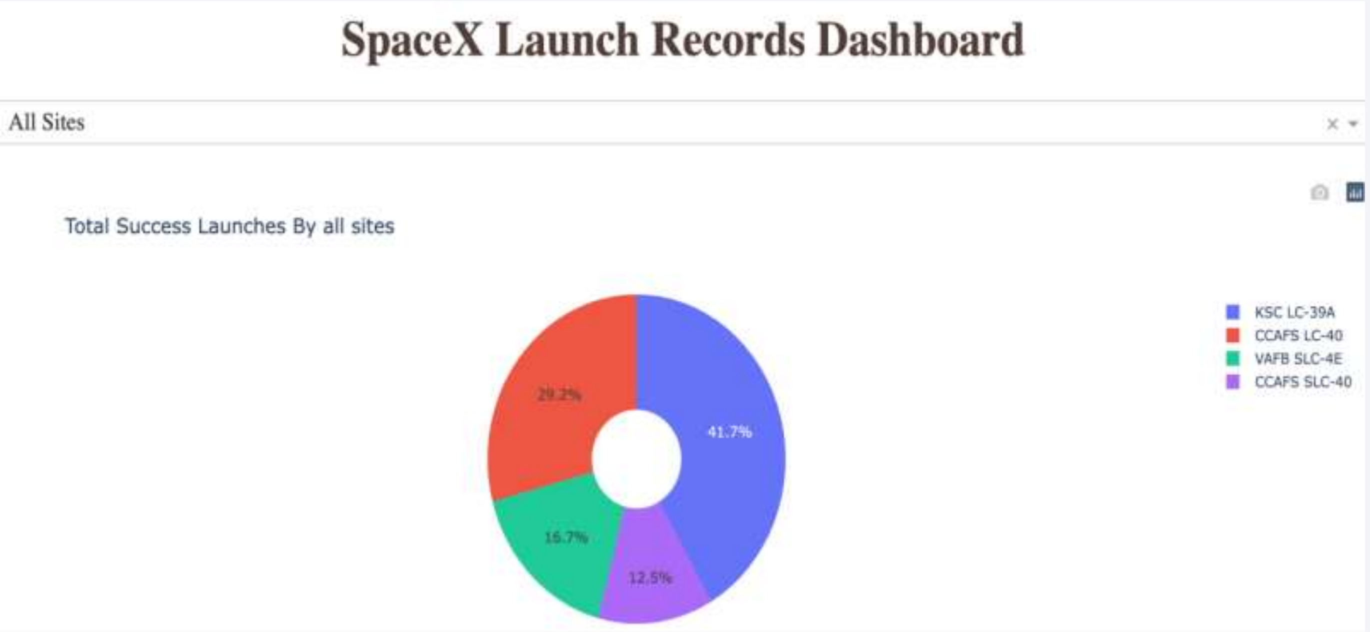
Section 5

Build a Dashboard with Plotly Dash



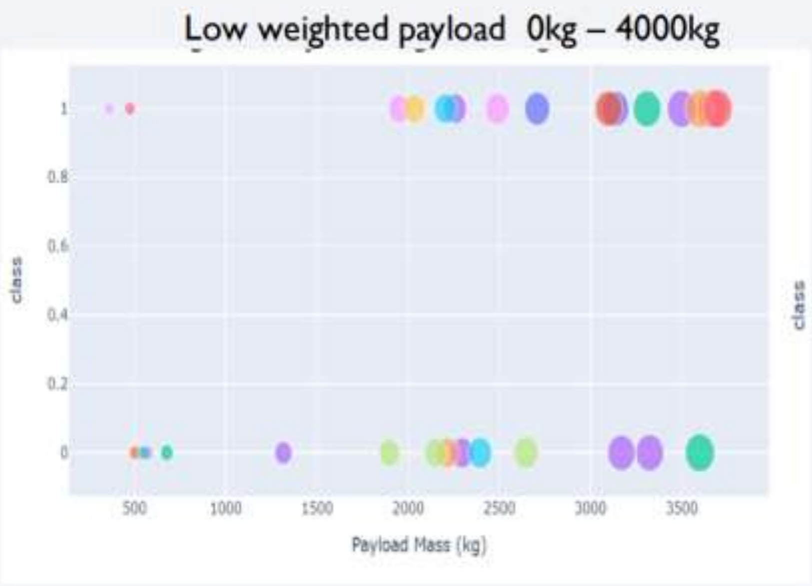
Launch Success Count for All Sites

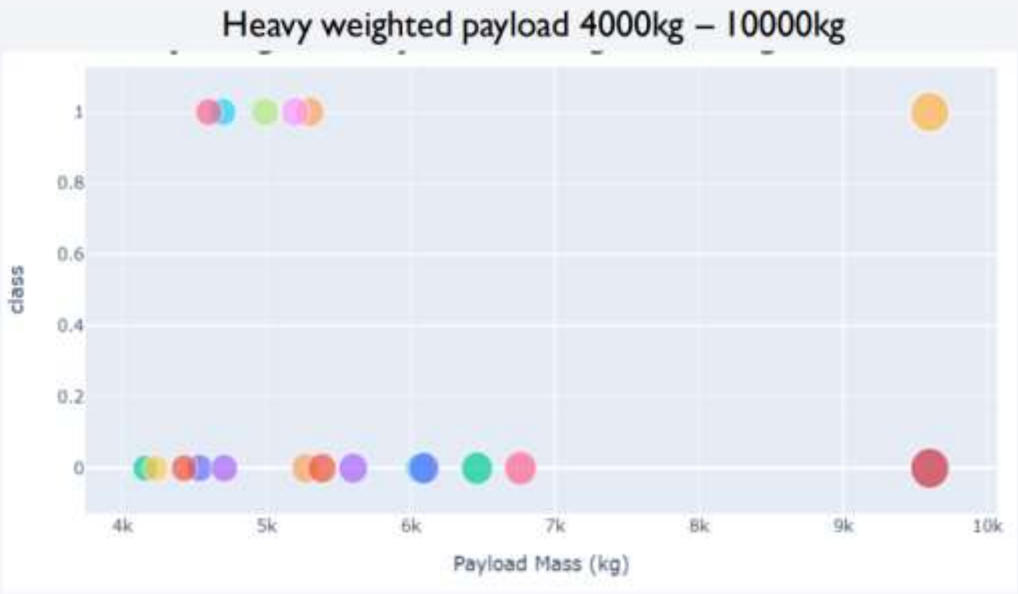
The Pie Charts dhows that KSC LC-39A has the most successful launches from all sites



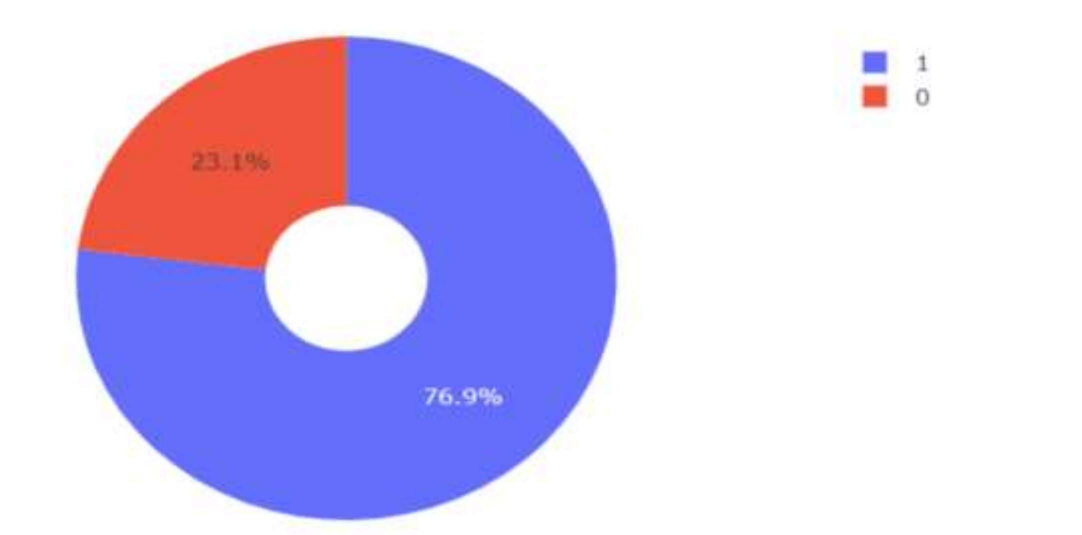
Payload and Launch Outcome Scatter Plot for all sites

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





Launch Site with the Highest Launch Success Ratio



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

- Which is the Highest launch success rate?

KSC LC-39A

- Which payload range has the highest launch success?

2000 kg-10000kg

- Which payload range has the lowest launch success rate?

0kg-1000kg

- Which F9 Booster Version has the highest launch success rate?

FT

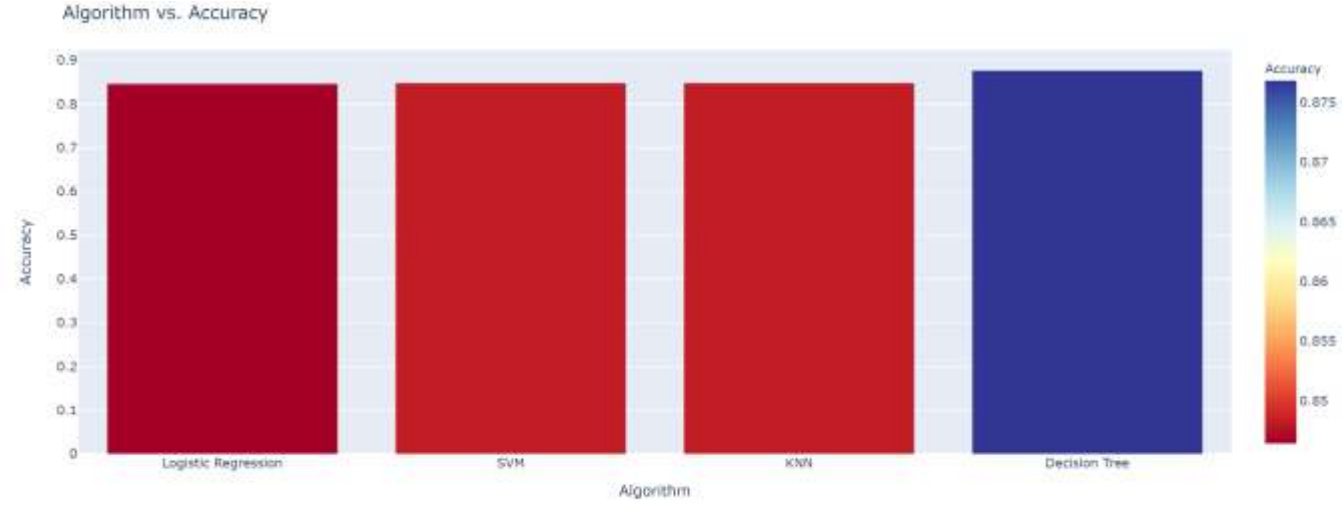
Section 6

Predictive Analysis (Classification)

Classification Accuracy

Decision Tree exhibit the highest accuracy level

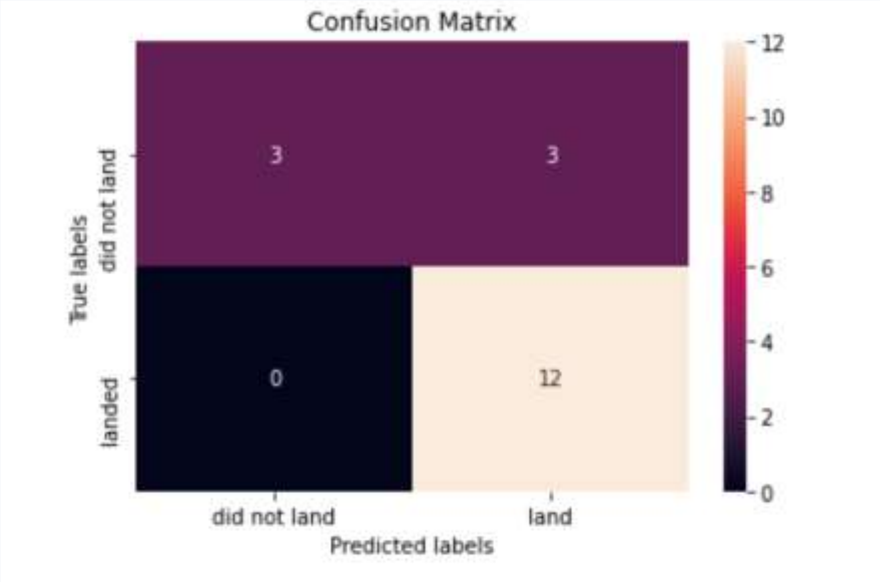
Algorithm	Accuracy	Accuracy on Test data	Tuned Hyperparameters
Logistic Regression	0.846429	0.83333	{'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
SVM	0.848214	0.83333	{'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
KNN	0.848214	0.83333	{'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
Decision Tree	**0.889286**	0.777777	{'criterion': 'entropy', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 10, 'splitter': 'random'}



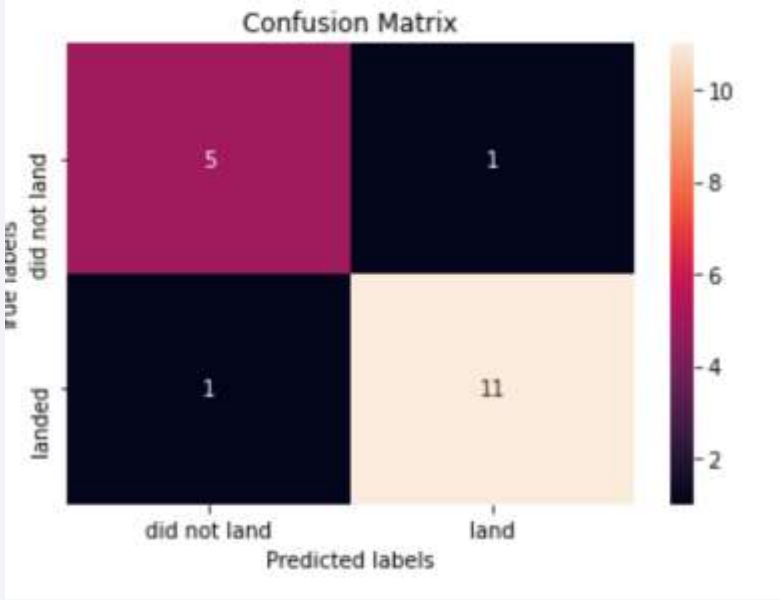
[GitHub URL]

Confusion Matrix

- For Logistic, KNN , SVM, they share same Confusion Matrix.



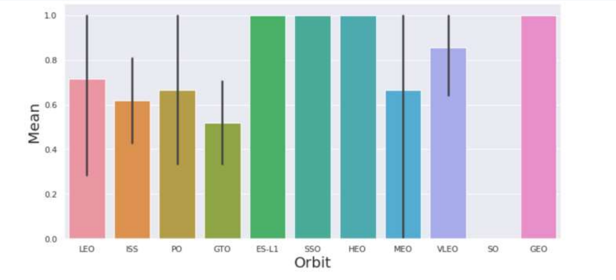
- This is Decision Tree’s Confusion Matrix.

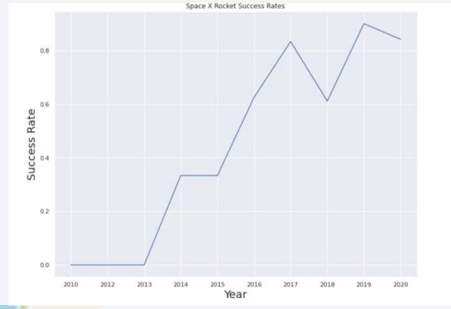


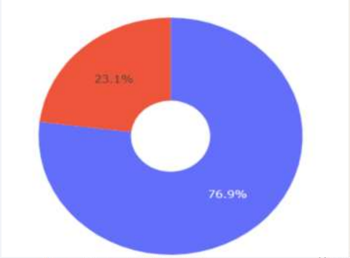
GitHub URL

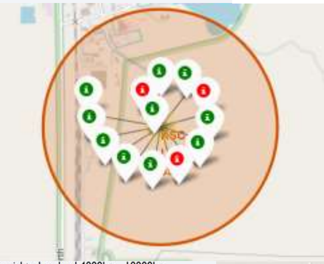
Conclusions

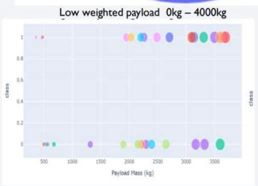
- Orbits ES-L1, GEO, HEO, SSO has highest success rates.
- The Success rates for SpaceX launches has been increasing relatively with time, they will eventually perfect the launches.
- KSC LC_39A had the most successful launches from all the sites.
- Low weighted payloads perform better than the heavier payloads.
- The tree classifier Algorithm is the best machine learning model for this dataset.

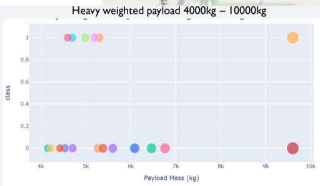




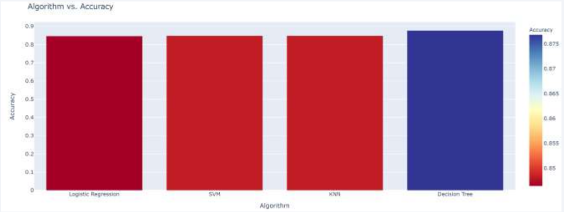








	Accuracy
Logistic Regression	0.846429
SVM	0.848214
KNN	0.848214
Decision Tree	0.889286



Thank you!