IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Rishi
Aug 08 2023

Outline

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

Executive Summary

- Summary of methodologies
  - Collection of Data through API & Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis
  - Visual Analytics
  - Predictions using ML
- Summary of all results
  - KSC LC 39A tops the list in Success rates
  - Higher Payload has more successful landings
  - Decision tree algorithm yielded the best result compared to SVM, Logistics Regression and KNN

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.
- Problems you want to find answers
  - predict if the Falcon 9 first stage will land successfully

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was acquired trough Web scraping and SpaceX API
- Perform data wrangling
  - Applied filters to get required data, checked for missing values; used One-Hot encoding
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Landing outcomes to be predicted by applying different classification Algorithms by fine tuning the hyper parameters.

Data Collection

- Request rocket launch data from SpaceX API
- Filter the dataframe to only include Falcon 9 launches
- Data Wrangling and Handling Missing values(replacing with mean)
- Data exported to CSV file

Data Collection – SpaceX API

- The data is acquired through requests of SpaceX API and normalized through JSON method.

- Github URL: https://github.com/rishikeshRS/DS/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

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

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

In [9]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_
```

We should see that the request was successfull with the 200 status response code

```
In [10]: response.status_code
Out[10]: 200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
In [14]: # Use json_normalize meethod to convert the json result into a dataframe
         data = pd.json_normalize(response.json())
```

Data Collection - Scraping

- Github URL: https://github.com/rishikeshRS/DS/blob/main/jupyter-labs-webscraping.ipynb

```
In [4]: # use requests.get() method with the provided static_url
        # assign the response to a object
        html_data = requests.get(static_url)
        html_data.status_code

Out[4]: 200
```

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

```
In [5]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
        soup = BeautifulSoup(html_data.text)
```

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

```
In [6]: # Use soup.title attribute
        soup.title

Out[6]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

Data Wrangling

- Describe how data were processed
  - Calculated the percentage of the missing values
  - Calculated the number of launches on each site
  - Calculated the number and occurrence of each orbit
  - Calculated the number and occurence of mission outcome of the orbits
  - Created a landing outcome label from Outcome column

In [11]:
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []

for i in df['Outcome']:
    if i in set(bad_outcomes):
        landing_class.append(0)
    else:
        landing_class.append(1)
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, one means the first stage landed Successfully

In [12]:
```
df['Class']=landing_class
df[['Class']].head(8)
```

Out[12]:

	Class
0	0
1	0
2	0
3	0
4	0
5	0

- Github URL: https://github.com/rishikeshRS/DS/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA with Data Visualization

- Scatter plots, Bar Graph and Line chart visuals were used to understand the relationships between various fields
  - **Flight Number and Launch Site**
  - **Payload and Launch Site**
  - **success rate of each orbit type**
  - **FlightNumber and Orbit type**
  - **Payload and Orbit type**
  - **Yearly Trend**

  GitHub URL : https://github.com/rishikeshRS/DS/blob/main/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- Some of the SQL Queries which were performed as part of analysis are:
  - ***unique launch sites names***
  - ***5 records where launch sites begin with the string 'CCA'***
  - ***total payload mass carried by boosters launched by NASA (CRS)***
  - ***average payload mass carried by booster version F9 v1.1***
  - ***total number of successful and failure mission outcomes***
  - ***Github URL: https://github.com/rishikeshRS/DS/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb***

Build an Interactive Map with Folium

- Marked success(1)/failed(0) launches for each site on the map sites on a map
- The distances between a launch site to its proximities was calculated
- Github URL: https://github.com/rishikeshRS/DS/blob/main/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Have added dropdown of launch sites
- Pie chart to visualize successful/ unsuccessful launches
- Scatter plot to understand relationship between Payload Mass and Outcome
- Github URL: https://github.com/rishikeshRS/DS/blob/main/Dashboard.ipynb

Predictive Analysis (Classification)

- Creating a NumPy array from the column Class in data
- Standardizing the data
- Splitting the data into training and test
- Applying Different Algorithms like logistic regression,SVM,KNN, Decesion Trees with GridSearchCV
- Calculating the accuracy
- Find the best classification algorithm
- Github URL: https://github.com/rishikeshRS/DS/blob/main/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite%20(2).ipynb

Results

- Exploratory data analysis results
  - There was an increase in launch success rate across the years
- Predictive analysis results
  - Decision Tree Classifier was giving highest accuracy

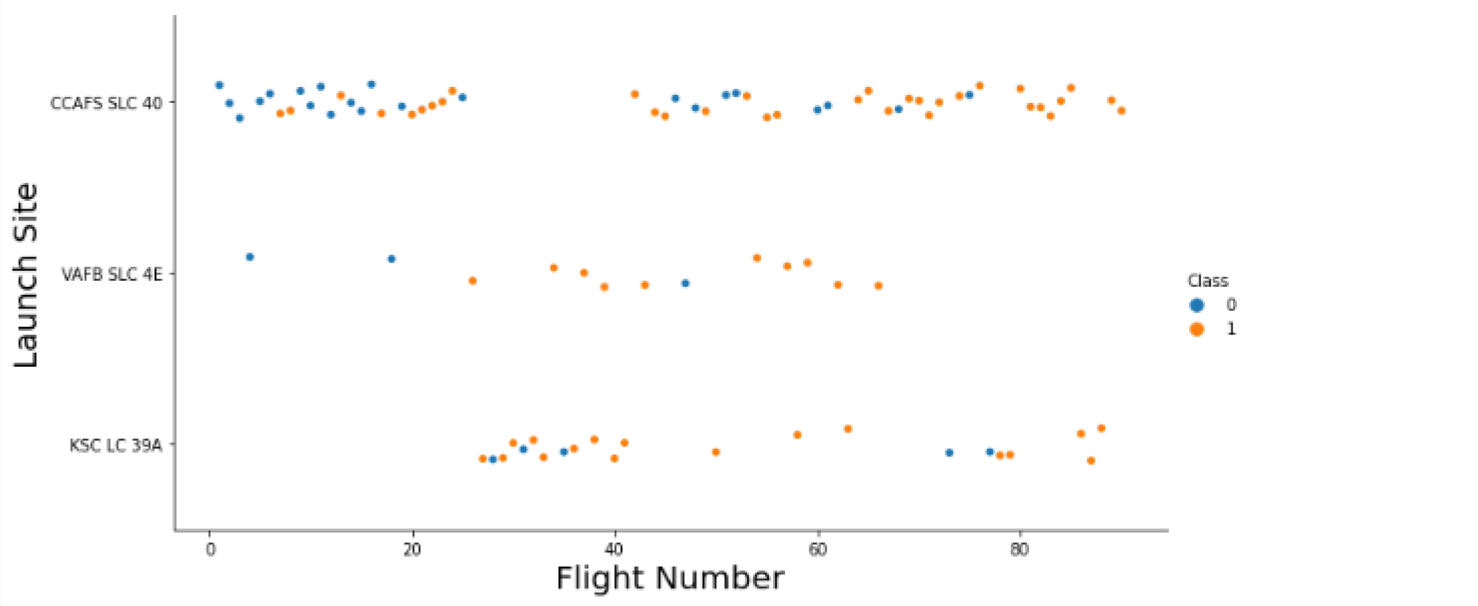
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- Inferences:
  - VAFB SLC 4E and KSC LC 39A have high Success rate
  - CCAFS SLC 40 Has more no of Launches

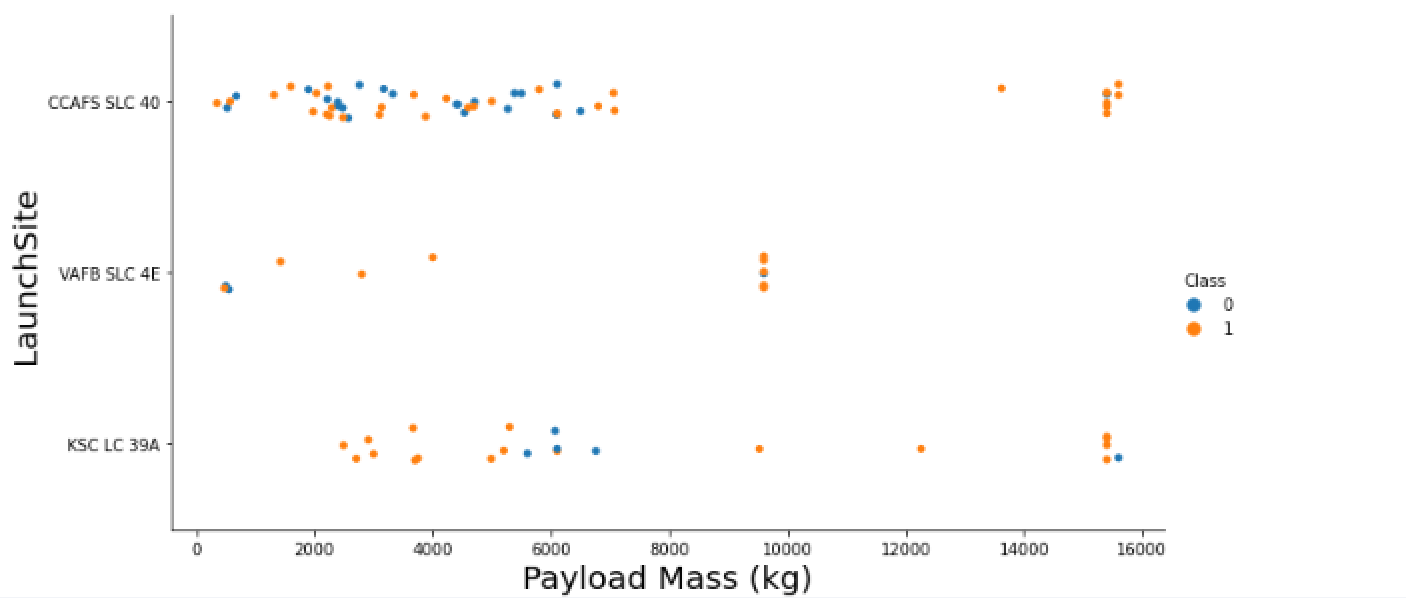
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the cl
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df,aspect=2)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Payload vs. Launch Site

- Inferences:
  - Most launches fall in between 2000 – 7000 Payload Mass Range.
  - Higher the Payload mass Higher is the success rate

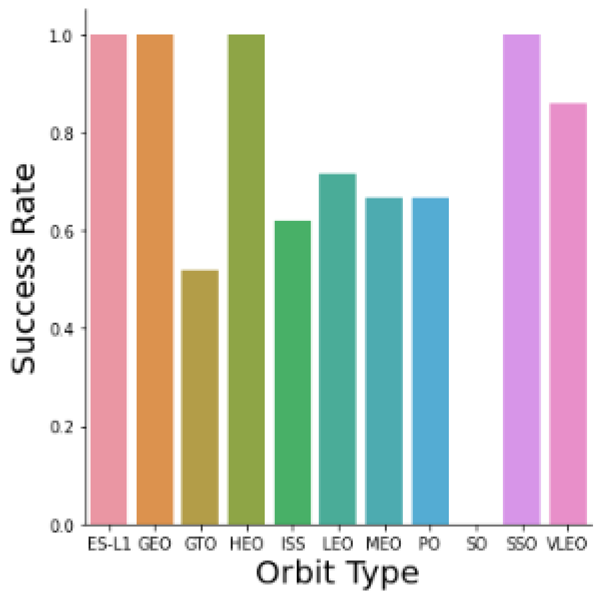
```
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df,aspect=2)
plt.xlabel("Payload Mass (kg)",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



Success Rate vs. Orbit Type

- Inferences:
  - SO – Has 0% success rate
  - GTO,ISS,LEO,PO,MEO have more than 50% success rate
  - ESL1, GEO, HEO,SSO has 100% success rate

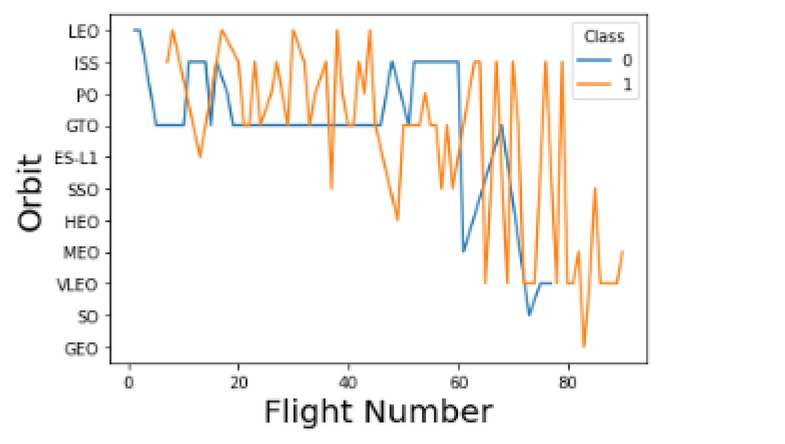
```
# HINT use groupby method on Orbit column and get the mean of Class column
sns.catplot(x= 'Orbit', y = 'Class', data = df.groupby('Orbit')['Class'].mean().reset_index(), kind = 'bar')
plt.xlabel('Orbit Type',fontsize=20)
plt.ylabel('Success Rate',fontsize=20)
plt.show()
```



Flight Number vs. Orbit Type

- Inferences:
  - The higher the flight number higher is the success rate

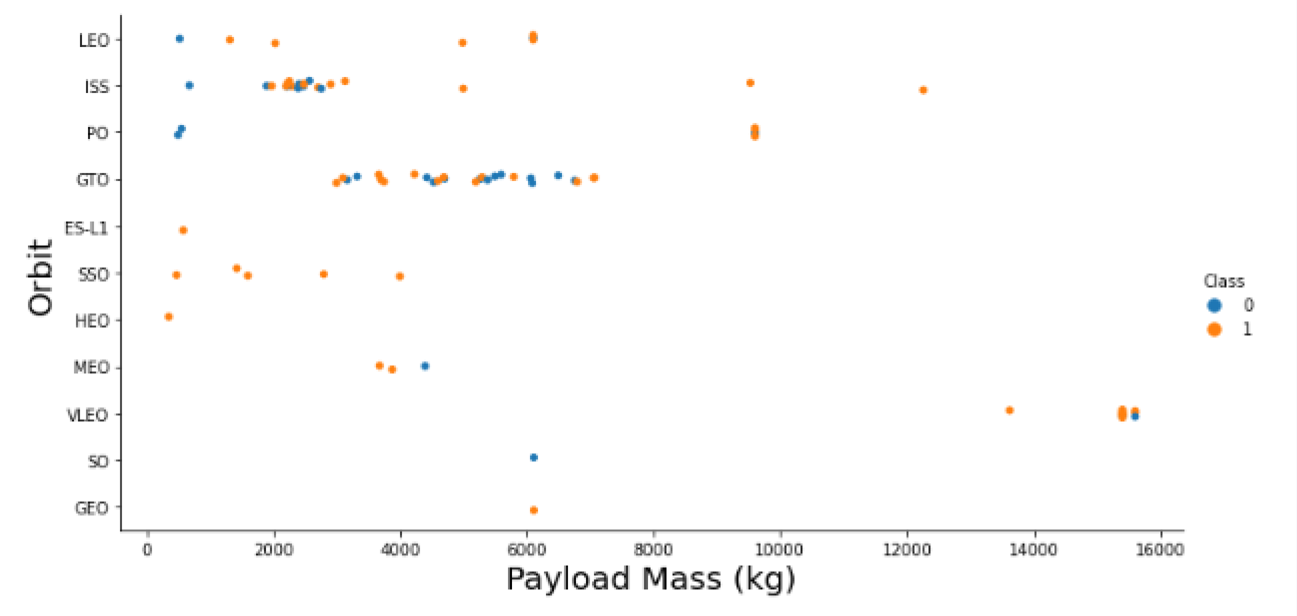
```
# Plot a scatter point chart with x axis to be FlightNumber and
sns.lineplot(y="Orbit", x="FlightNumber",hue="Class",data=df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Payload vs. Orbit Type

- Inferences:
  - Higher Payload has more successful landings

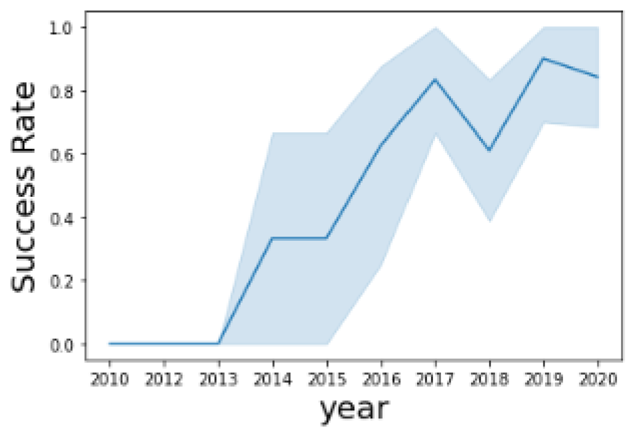
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the cla
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df,aspect=2)
plt.xlabel("Payload Mass (kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend

- Inferences:
  - From 2013 -2020 The success rate was increasing but 2018 had a decline

```
# Plot a line chart with x axis to be the extracted year and y axi
sns.lineplot(data=df, x="Date", y="Class")
plt.xlabel("year",fontsize=20)
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```



All Launch Site Names

- Find the names of the unique launch sites

```
%sql ibm_db_sa://yyy33800:dwNKg8J3L0IBd6CP@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:3
%sql SELECT Unique(LAUNCH_SITE) FROM SPACEXTBL;
```

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

In [13]:
```
%sql SELECT * \
    FROM SPACEXTBL \
    WHERE LAUNCH_SITE LIKE'CCA%' LIMIT 5;
```

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

Out[13]:

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)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	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

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
In [14]: %sql SELECT SUM(PAYLOAD_MASS__KG_) \
             FROM SPACEXTBL \
             WHERE CUSTOMER = 'NASA (CRS)';

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

Out[14]:

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

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

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

 * sqlite:///my_data1.db
Done.

Out[15]:
```

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
In [9]: %sql SELECT MIN(DATE) \
        FROM SPACEXTBL \
        WHERE LANDING_OUTCOME = 'Success (ground pad)'

 * sqlite:///my_data1.db
Done.

Out[9]:
```

MIN(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

In [13]:
```
%sql SELECT Booster_Version \
FROM SPACEXTBL \
WHERE LANDING_OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

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

Out[13]:

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

```
In [17]: %sql SELECT MISSION_OUTCOME, COUNT(*) as total_number \
         FROM SPACEXTBL \
         GROUP BY MISSION_OUTCOME;

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

Out[17]:

Mission_Outcome	total_number
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

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

```
In [18]: %sql SELECT BOOSTER_VERSION \
         FROM SPACEXTBL \
         WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);

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

Out[18]:

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

2015 Launch Records

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

```
%sql SELECT substr(Date,4,2) as month, DATE,BOOSTER_VERSION, LAUNCH_SITE, [Landing_Outcome] \
FROM SPACEXTBL \
where [Landing_Outcome] = 'Failure (drone ship)' and substr(Date,7,4)='2015';
```

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

- 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

```
%sql SELECT [Landing_Outcome], count(*) as Outcomes \
FROM SPACEXTBL \
WHERE DATE between '04-06-2010' and '20-03-2017' group by [Landing_Outcome] order by Outcomes DESC;
```

Section 3

Launch Sites Proximities Analysis

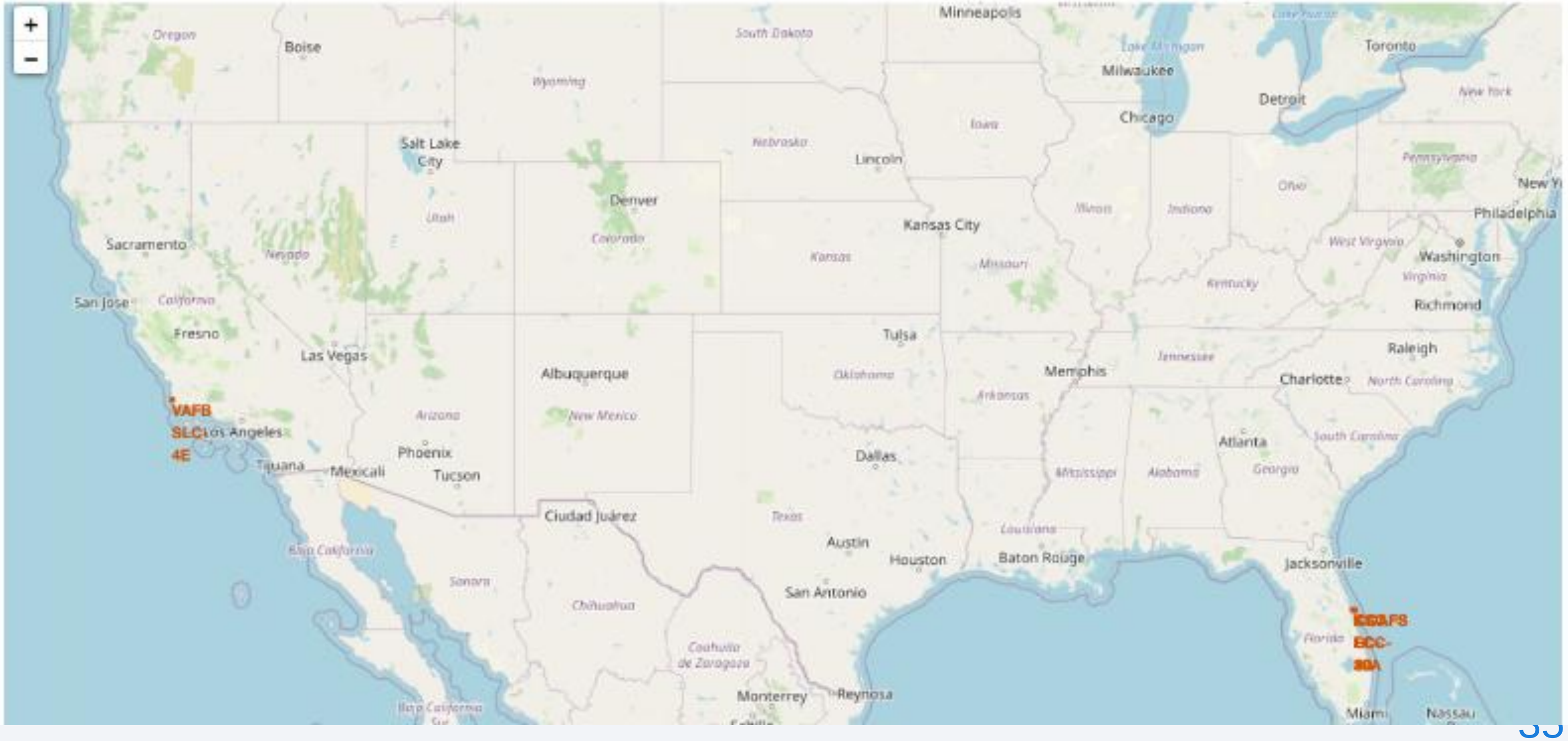
Launch Sites

Inferences:

- The launches were from CA and FL

```
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat, Long) values. In addition, add Launch site name as a po
```

The generated map with marked launch sites should look similar to the following:

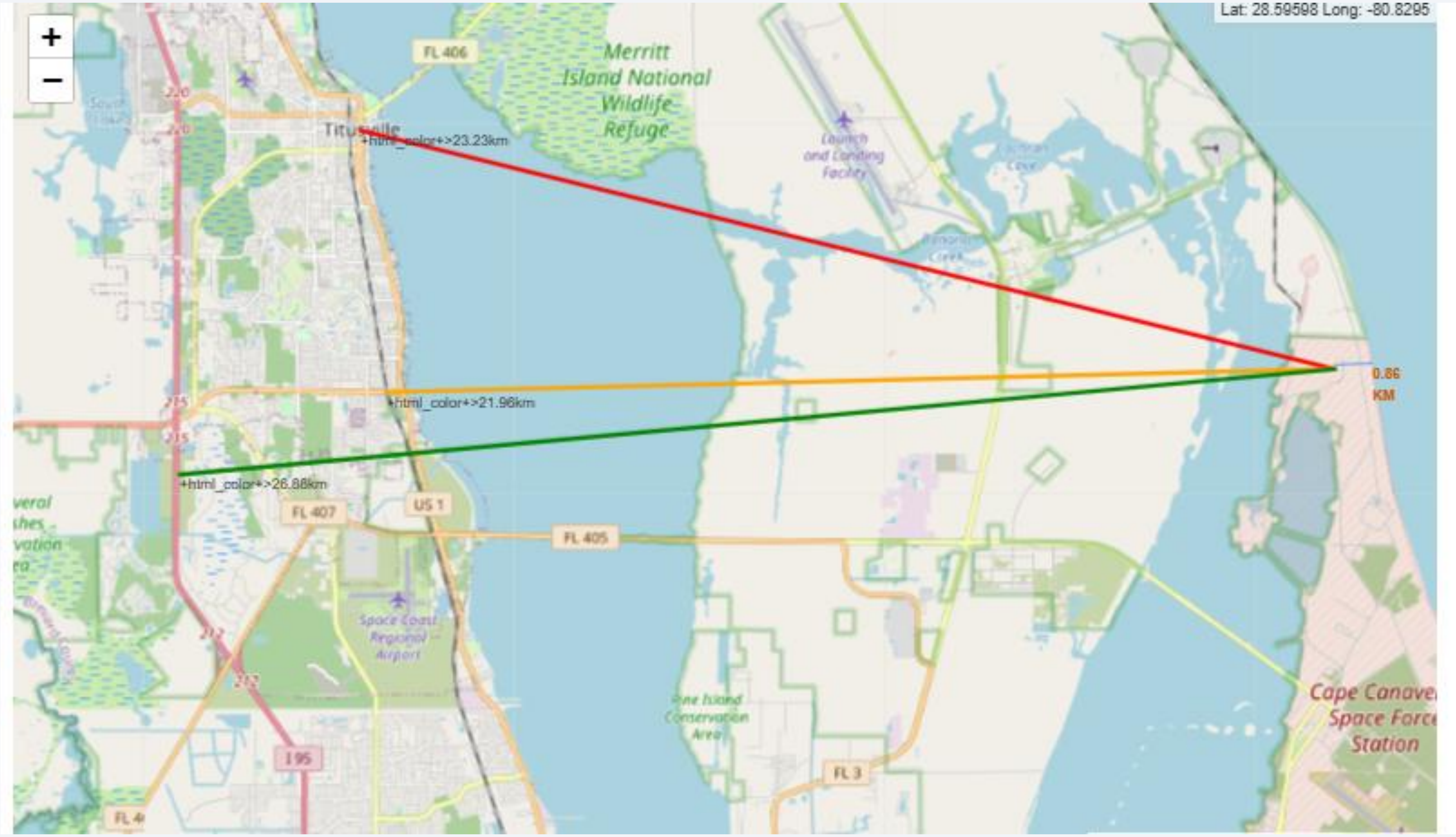


Launches Color-labeled markers

- Red = failed
- Green = Success

Distance to a closest city, railway, highway

- Railways are closer to launch sites

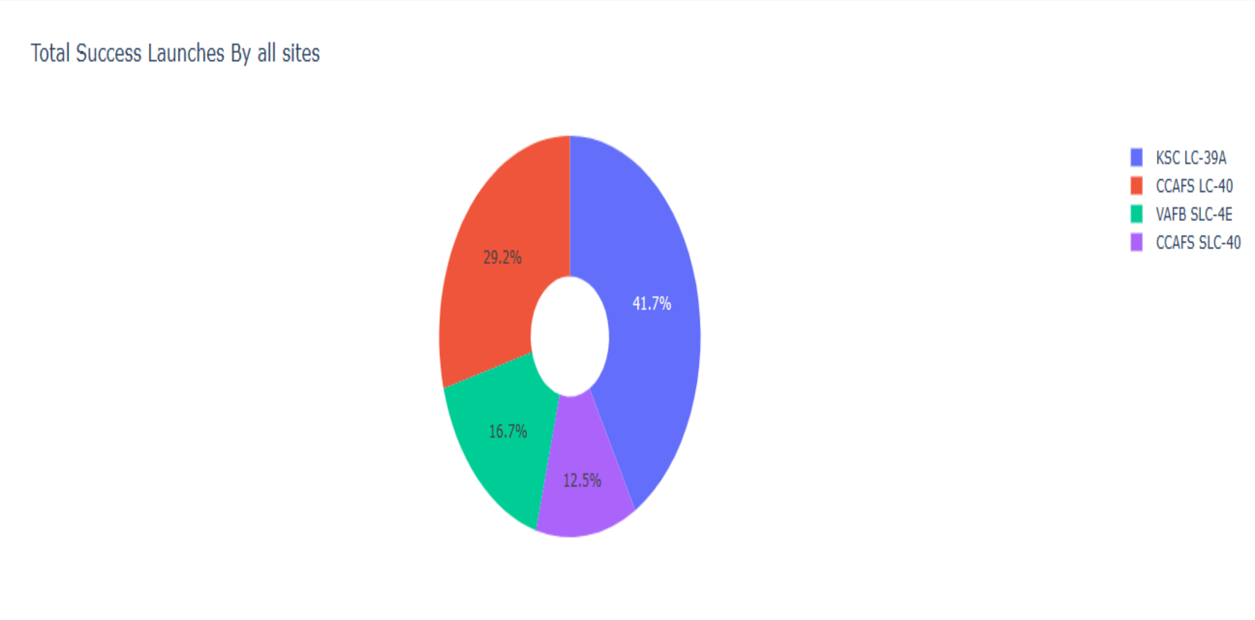


Section 4

Build a Dashboard with Plotly Dash

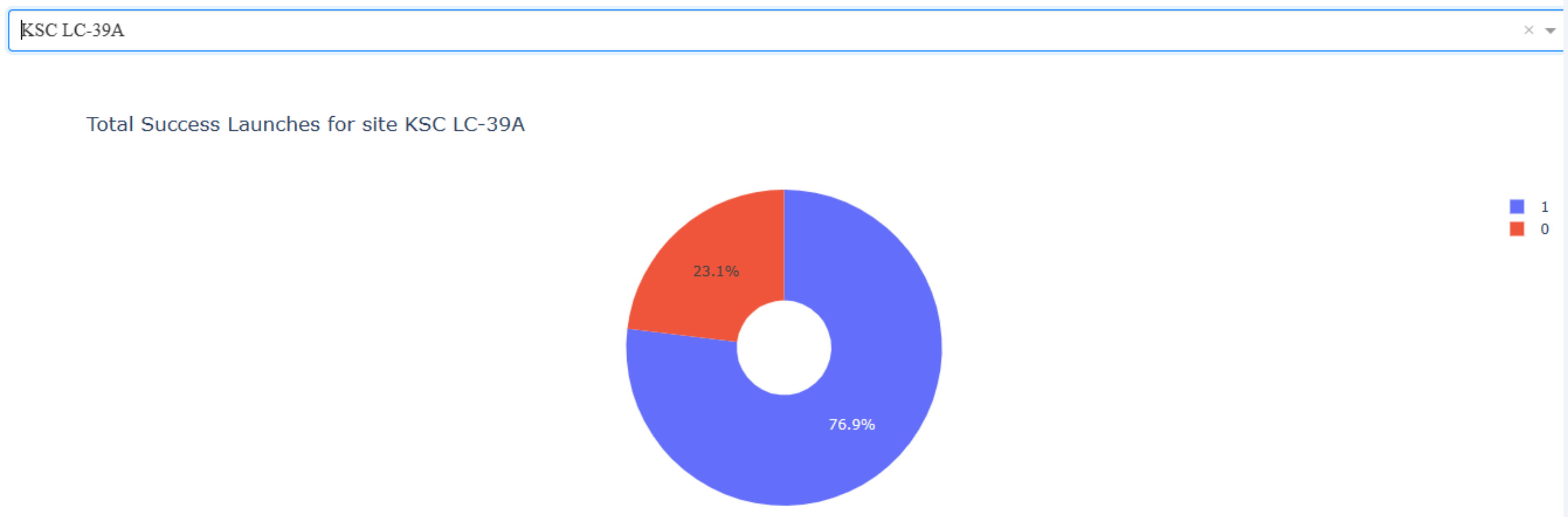
Success Launches For All Sites

- KSC LC 39A has high success rate

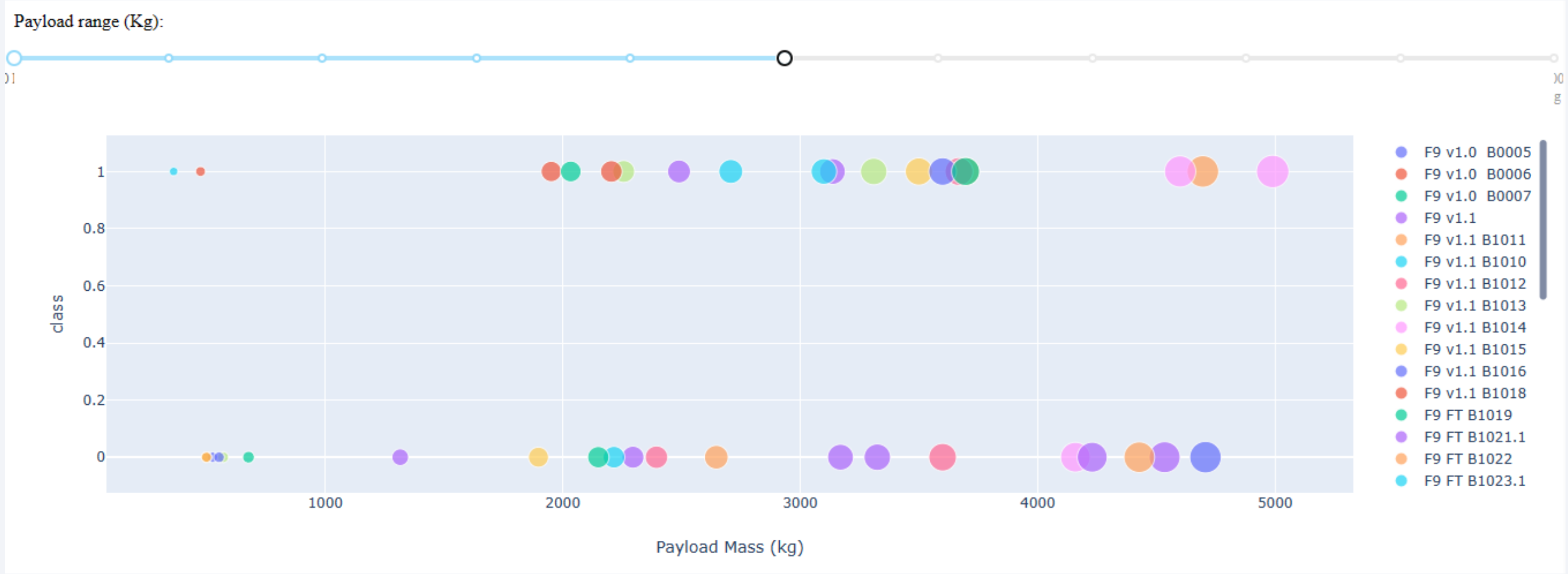


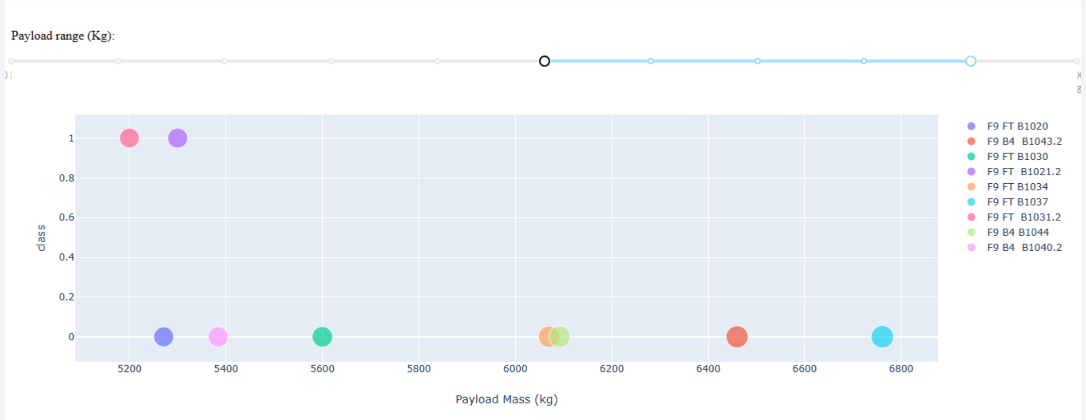
Highest Launch Success Ratio

- KSC LC 39A had 77% success rate



Payload Vs Launch Outcomes





For Higher Payloads Success rate was low

Section 5

Predictive Analysis (Classification)

Classification Accuracy

```
n [40]: accuracy = [svm_score, logscore, knn_cv_score, tree_score]
        accuracy = [i * 100 for i in accuracy]
        method = ['Support Vector Machine', 'Logistic Regression', 'K Nearest Neighbour', 'Decision Tree']
        models = {'ML Method':method, 'Accuracy Score (%)':accuracy}
        ML_df = pd.DataFrame(models)
        ML_df
```

ut[40]:

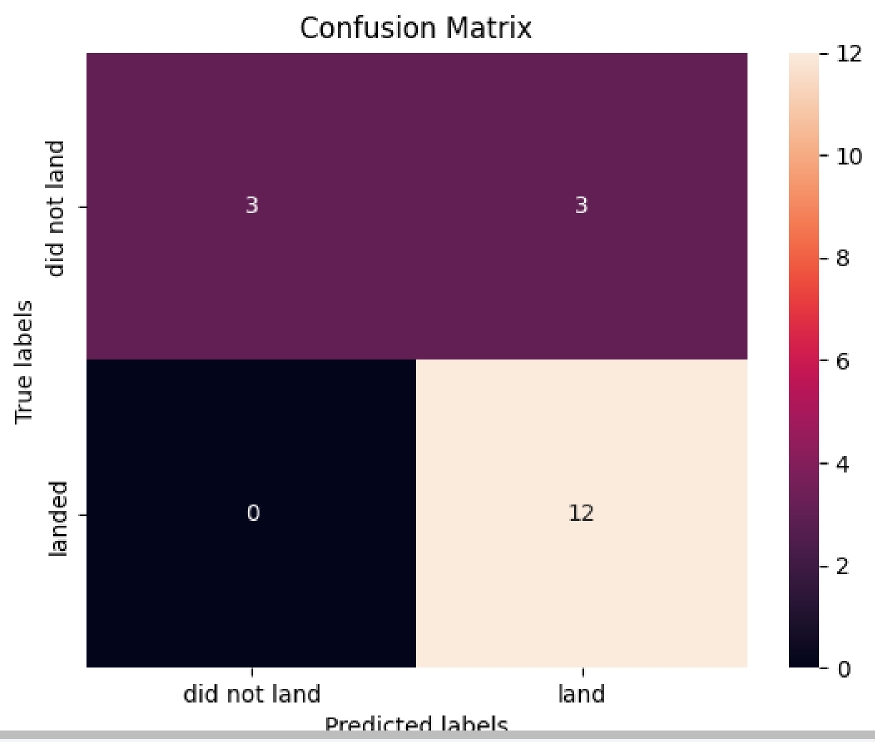
	ML Method	Accuracy Score (%)
0	Support Vector Machine	83.333333
1	Logistic Regression	83.333333
2	K Nearest Neighbour	83.333333
3	Decision Tree	88.888889

```
n [44]: 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(bestalgorithm)
```

```
DecisionTree
```

Confusion Matrix

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



Conclusions

- VAFB SLC 4E and KSC LC 39A have high Success rate
- Most launches fall in between 2000 – 7000 Payload Mass Range (Higher Payload has more successful landings)
- ESL1, GEO, HEO,SSO has 100% success rate
- Launch success rate was increasing across the years
- Launch sites were closer to Railways
- All the classification algorithms which were applied had more than 80% accuracy, but decision tree slightly outperformed.

Thank you!