IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Chiraag P V
06-10-2022

Outline

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

Executive Summary

METHEDOLOGIES

- Collecting the data using API and Web scraping.
- Exploratory Data Analysis (EDA) with Visualization.
- Analysing the locations of the launch sites.
- Using Machine Learning to predict success rate of rocket launch.

RESULTS

- A good amount of data was collected and cleaned.
- Graphs were built for better understanding.
- Determined the launch sites and their success rates of a rocket launch.
- Classification Algorithms were used to best classify the launch.

Introduction

- Our aim is to build a new Space company called **SpaceY.**
- We are using the dataset of SpaceX to get better insights on how the company must function and take decisions in a better way.
- This project helps us to find out best locations for launch site.
- We can estimate cost for launches, required elements for a space project and more.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using API and Web Scarping.
- Perform data wrangling
  - Data was cleaned and processed using normalization and one-hot encoding to get better results
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Using Logistic Regression, SVM's, KNN and Decision tree Algorithms to best classify models using Grid Search.

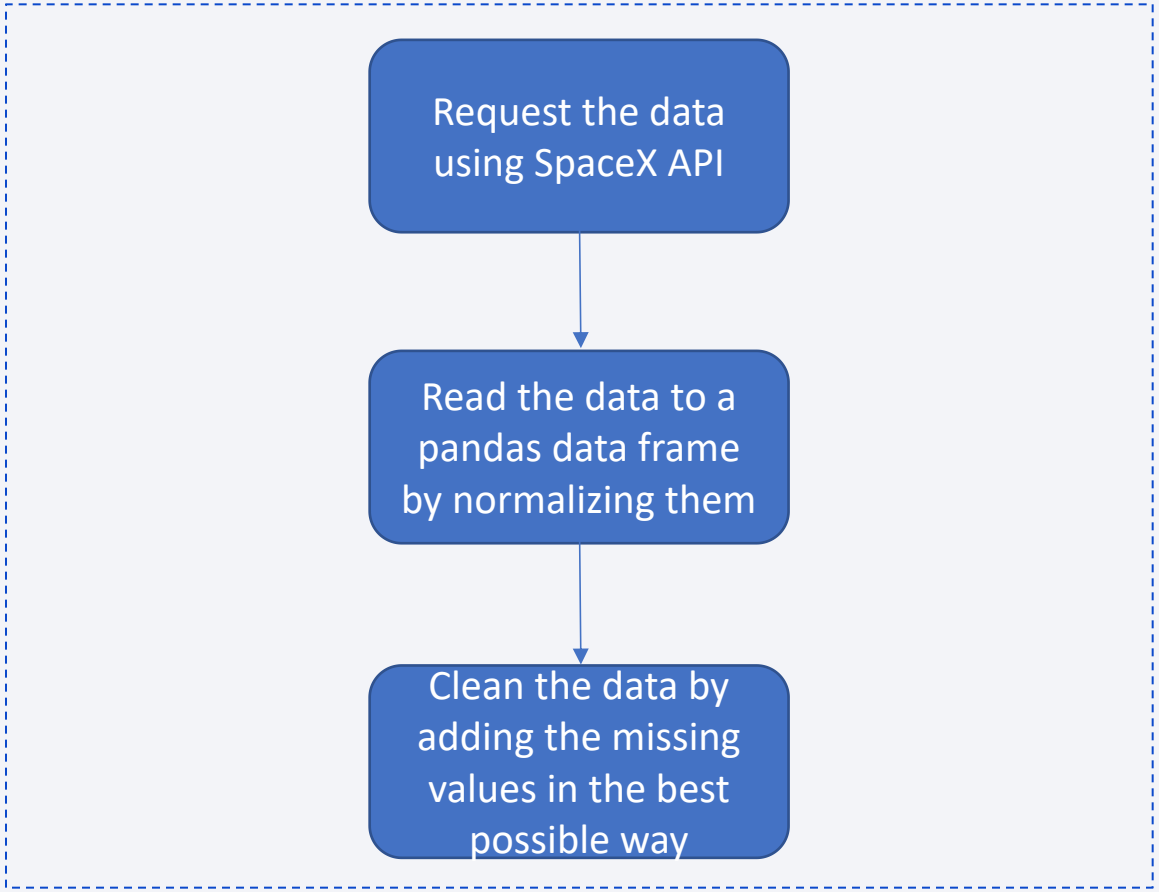
Data Collection

- Data Sets were collected using API's and Web Scraping.
- Data was collected using API's by sending requests to fetch the results of URL and the response was used to fetch the required data.
- Data was also collected by Web Scraping using Beautiful Soup Library.

Data Collection – SpaceX API

- response = requests.get(spacex_url)

  data = pd.json_normalize(response.json())
- An URL was added to a variable spacex_url and the response was fetched using request.get().
- Later the data was normalized and converted to pandas data frame.
- Check out the complete code : CLICK HERE



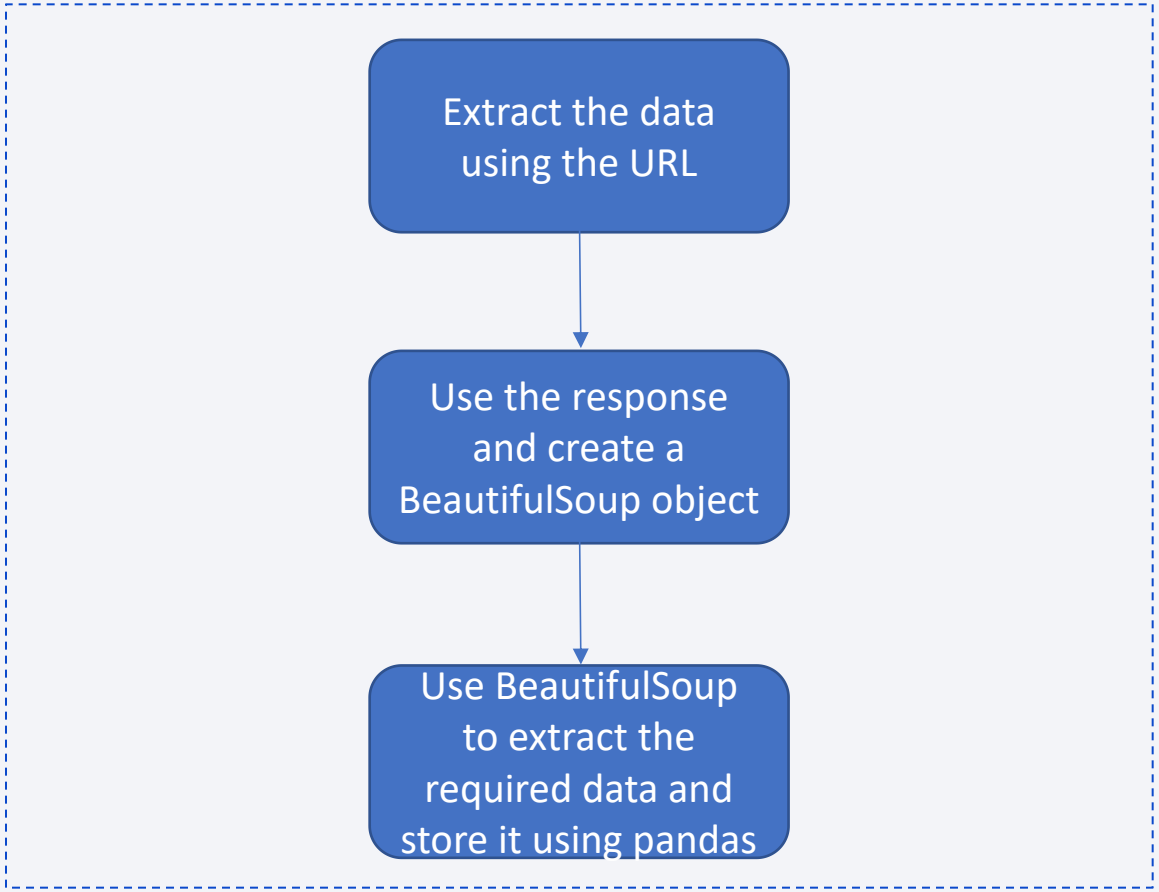
Data Collection – Scraping

- Response = requests.get(static_url)

  soup = BeautifulSoup(response.text,'html5lib')

  html_tables = soup.findAll('table')

- Now, we iterate through the headers(<th>), extract the data and store it in the form of a table.
- Check out the complete code : CLICK HERE



Data Wrangling

- We need to clean the dataset in order to do better data analysis and to obtain better results.
- We use the actual data and determine the best possible ways to look into a data.
- df.isnull().sum()/df.count()*100 : Used to check null values in the data.
- df.dtypes : Used to check the datatypes.
- Check out the complete code : CLICK HERE

EDA with Data Visualization

- Data Visualization is most necessary as a part of Exploratory Data Analysis. This helps in analyzing the data in an easier way using graphs and charts.
- Scatter plots were created between many attributes to obtain their relationship.
- Bar Charts, Line Charts and Cat plots were also created to analyze the data.
- Check out the complete code : CLICK HERE

EDA with SQL

- We will be adding the data to a database where we can query the reuired results using SQL.
- We have displayed the names of unique launch sites, total payload mass carried by boosters launched by NASA (CRS) and many more,
- We can SQL and extract the data for analysis using Pattern Matching, Buil-in Mathematical Functions and many more
- Check out the complete code : CLICK HERE

Build an Interactive Map with Folium

- We can build an interactive map with Folium which can provide us better view of the launch sites and distance from source to destination and many more.
- We have used folium.circle and folium.map.marker and many other object of Folium Library.
- Folium.circle : creates a circle of given radius on the map after providing latitudes and longitudes of the location.
- Folium.map.marker : creates a marker on the specified location.
- Check out the complete code : CLICK HERE

Build a Dashboard with Plotly Dash

- We have created a real-time visual dashboard using Plotly which provides better view of Graphs and Charts.
- The plots were added to the dashboard because it provides real-time experience to the users with a great view of data.
- Added a dropdown to the dashboard to select launch sites and find the EDA of that particular launch site.
- Check out the complete code : CLICK HERE

Predictive Analysis (Classification)

- A Classification model is built in order to predict the success of rocket launch
- There are 4 classification models used :
  - Logistic Regression
  - Support Vector Machines
  - Decision Tree
  - K – Nearest Neighbours
- The accuracy of the model is determined using best_score and confusion matrix.
- Check out the complete code : CLICK HERE

Results

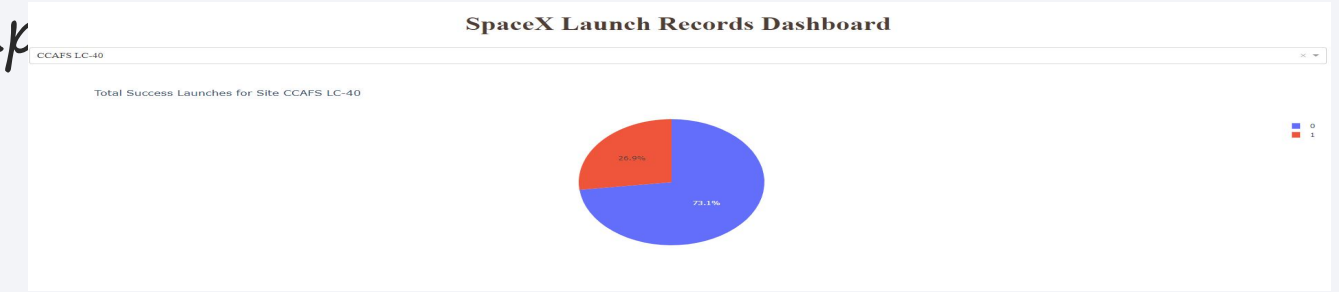
- Exploratory data analysis res

```
%sql select DISTINCT mission_outcome, count(*) as count from XMG48161.SPACEX group by mission_outcome;
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

mission_outcome	COUNT
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- Interactive analytics samp



- Predictive analysis resul

```
Accuracy Score :
Logistic Regression Accuracy Score : 0.8464285714285713
Support Vector Machine Accuracy Score : 0.8482142857142856
Decision Tree Accuracy Score : 0.8892857142857145
KNN Accuracy Score : 0.8482142857142858
```

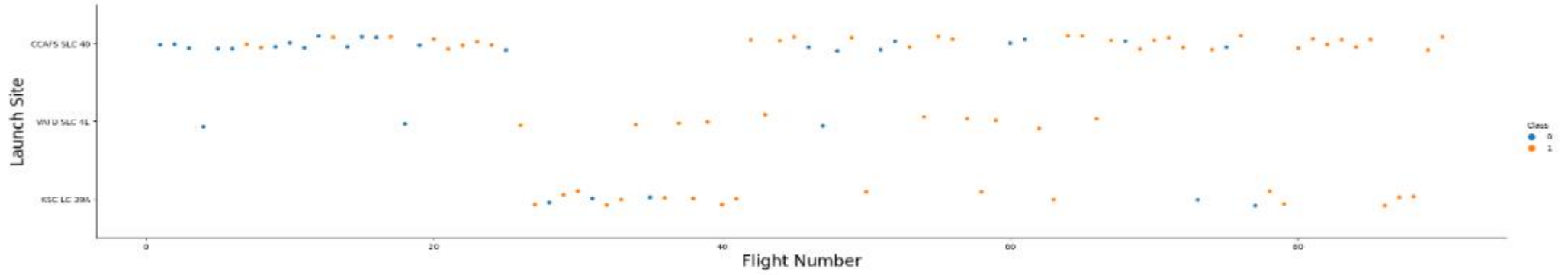
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

- Scatter plot for Flight Number vs. Launch Site :

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

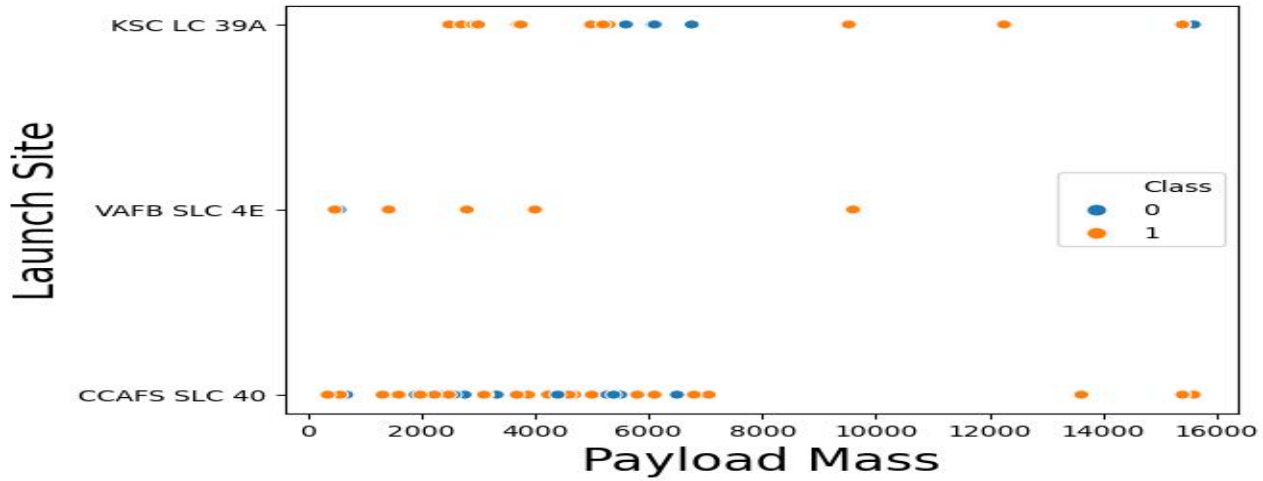


- A scatter plot is built with X-axis as Flight Number and Y-axis as Launch Site. This does not provide us much insights, lets carry on further.

Payload vs. Launch Site

- Scatter Plot for Payload vs. Launch Site

```
# 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 class value
sns.scatterplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df)
plt.xlabel("Payload Mass",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



- We can find that heavier Payload Mass rockets are launched in KSC LC 39A and CCAFS SLC 40 Launch Sites.

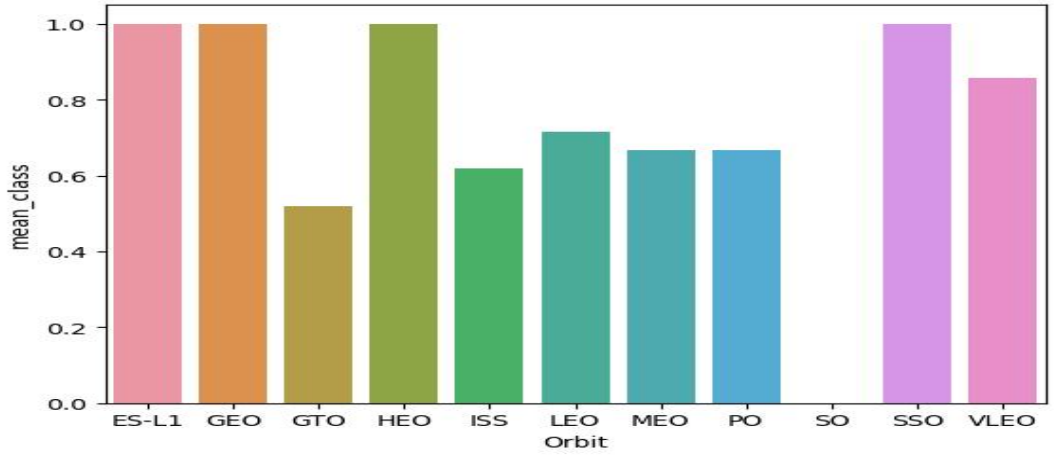
Success Rate vs. Orbit Type

- Success rate of each orbit type using Bar Plot :

```
# HINT use groupby method on Orbit column and get the mean of Class column
A = df.groupby(['Orbit']).agg(mean_class=("Class", 'mean'))
A = A.reset_index()

# plot barplot
sns.barplot(x="Orbit",y="mean_class",data=A)
```

```
<AxesSubplot:xlabel='Orbit', ylabel='mean_class'>
```

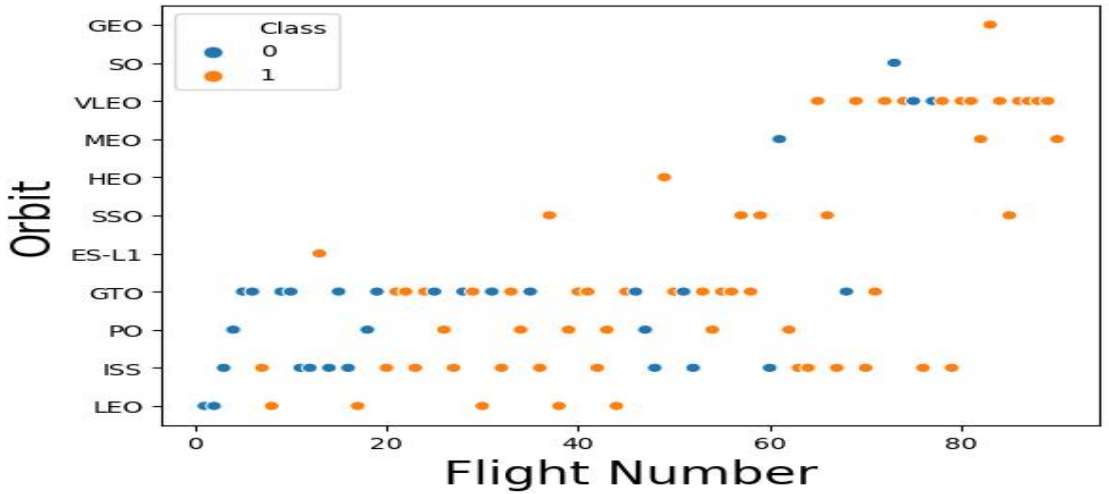


- We can see that rockets launched to ES-L1, GEO, HEO and SSO orbits have higher success rate.

Flight Number vs. Orbit Type

- Scatter Plot of Flight number vs. Orbit type

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

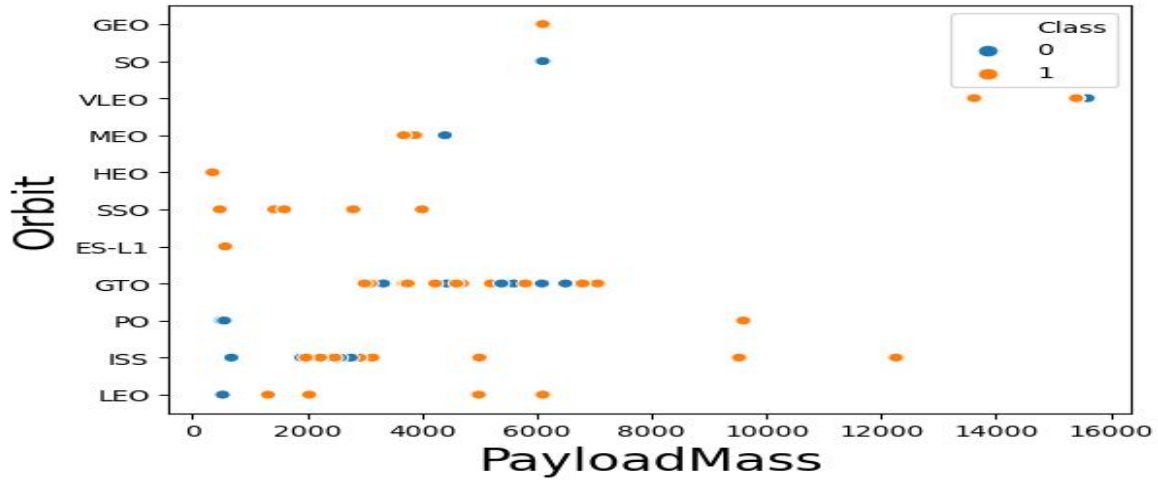


- We can obtain insights by seeing the number of flights launched to specific orbits but cannot fetch better insights when Flight Number is considered.

Payload vs. Orbit Type

- Scatter plot of payload vs. orbit type

```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.scatterplot(y="Orbit", x="PayloadMass", hue="Class", data=df)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



- We can see that with heavy payloads the successful landing rate is more for Polar, VLEO and ISS.

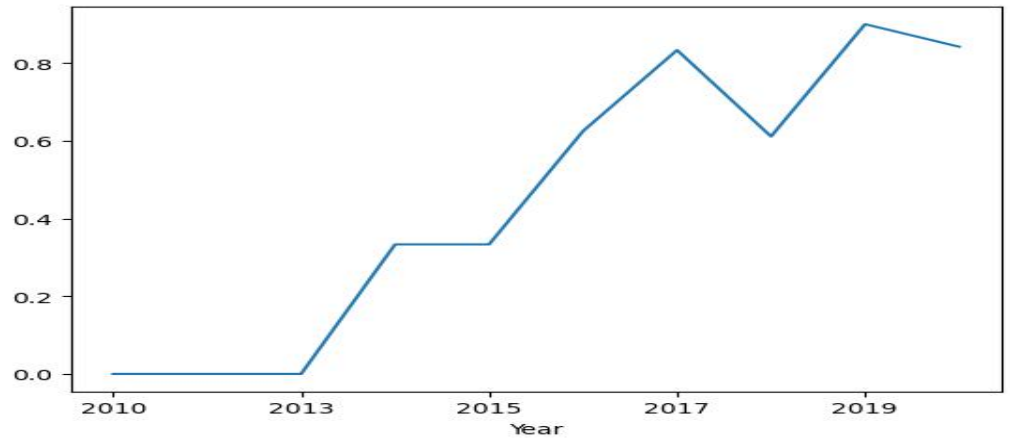
Launch Success Yearly Trend

- Line chart of yearly average success rate

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate

temp_df = df.copy()
temp_df['Year'] = year
temp_df.groupby('Year')['Class'].mean().plot()
```

```
<AxesSubplot:xlabel='Year'>
```



- Success rate after 2013 keeps increasing.

All Launch Site Names

- Unique launch sites :

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

- Query : **%sql** select DISTINCT launch_site from XMG48161.SPACEX;
- This query provides us distinct or unique launch sites present.

Launch Site Names Begin with 'CCA'

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

```
%sql select * from XMG48161.SPACEX where left (launch_site, 3) in ('CCA') LIMIT 5;
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/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

- Query : **%sql** select * from XMG48161.SPACEX where left (launch_site, 3) in ('CCA') LIMIT 5;
- This query provides 5 records of launch sites which begins with 'CCA'.

Total Payload Mass

- Total payload carried by boosters from NASA

```
%sql select SUM(payload_mass__kg_) as Total_payload_mass_by_nasa_crs from XMG48161.SPACEX where right (customer,4) in ('CRS)');
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

total_payload_mass_by_nasa_crs
45596

- Query : **%sql** select SUM(payload_mass__kg_) as Total_payload_mass_by_nasa_crs from XMG48161.SPACEX where right (customer,4) in ('CRS)');
- This query provides us the total payload carried by booster that are launched by NASA.

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.1

```
%sql select AVG(payload_mass__kg_) as Average_payload_mass_by_F9_v1_1 from XMG48161.SPACEX where left (booster_version,7) in ('F9 v1.1');
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

average_payload_mass_by_f9_v1_1
2534

- Query : **%sql** select AVG(payload_mass__kg_) as Average_payload_mass_by_F9_v1_1 from XMG48161.SPACEX where left (booster_version,7) in ('F9 v1.1');

- This query provides us the average of payload mass that was carried by F9 v1.1 booster version.

First Successful Ground Landing Date

- Dates of the first successful landing outcome on ground pad

```
%sql select min(date) as first_landing_outcome_in_ground_pad from XMG48161.SPACEX where left (landing__outcome,15) in ('Success (ground');
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

first_landing_outcome_in_ground_pad
2015-12-22

- Query : **%sql** select min(date) as first_landing_outcome_in_ground_pad from XMG48161.SPACEX where left (landing__outcome,15) in ('Success (ground');
- Using this query we can extract the date of first landing outcome on ground pad.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
%sql select booster_version from XMG48161.SPACEX where payload_mass__kg_ BETWEEN 4000 and 6000 and left (landing__outcome,14) in ('Success (drone');
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

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

- Query : **%sql** select booster_version from XMG48161.SPACEX where payload_mass__kg_ BETWEEN 4000 and 6000 and left (landing__outcome,14) in ('Success (drone');

- This query provides the result of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

- Total number of successful and failure mission outcomes

```
%sql select DISTINCT mission_outcome, count(*) as count from XMG48161.SPACEX group by mission_outcome;
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

mission_outcome	COUNT
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- Query : **%sql** select DISTINCT mission_outcome, count(*) as count from XMG48161.SPACEX group by mission_outcome;
- This query groups the output based on mission_outcomes and provides the count.

Boosters Carried Maximum Payload

- Names of the booster which have carried the maximum payload mass

```
%sql select DISTINCT booster_version,payload_mass__kg_ from (select booster_version,payload_mass__kg_ from XMG48161.SPACEX order by payload_mass__kg_
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600

- Query : **%sql** select DISTINCT booster_version,payload_mass__kg_ from (select booster_version,payload_mass__kg_ from XMG48161.SPACEX order by payload_mass__kg_ DESC) LIMIT 5;
- This query provides the names of the boosters which have carried the maximum payload mass.

2015 Launch Records

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

```
%sql select booster_version, launch_site, landing__outcome from XMG48161.SPACEX where left (date,4) in ('2015') and left (landing__outcome,14) in ('Fa
```

```
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

booster_version	launch_site	landing__outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- Query : **%sql** select booster_version, launch_site, landing__outcome from XMG48161.SPACEX where left (date,4) in ('2015') and left (landing__outcome,14) in ('Failure (drone');
- Failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 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 distinct landing__outcome, count(*) as landing_outcome_count from XMG48161.SPACEX where date between '2010-06-04' and '2017-03-20' group b
 * ibm_db_sa://xmg48161:***@2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32328/bludb
Done.
```

landing__outcome	landing_outcome_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

- Query : **%sql** select distinct landing_outcome, count(*) as landing_outcome_count from XMG48161.SPACEX where date between '2010-06-04' and '2017-03-20' group by landing_outcome order by landing_outcome_count desc ;


Section 3
Launch Sites
Proximities Analysis

All Launch Sites

- All the launch sites are near to the sea.
- The launch should be taken with utmost safety as there is some areas of land nearby.

Launch Outcomes based on Sites

- KSC LC – 39A Launch Site

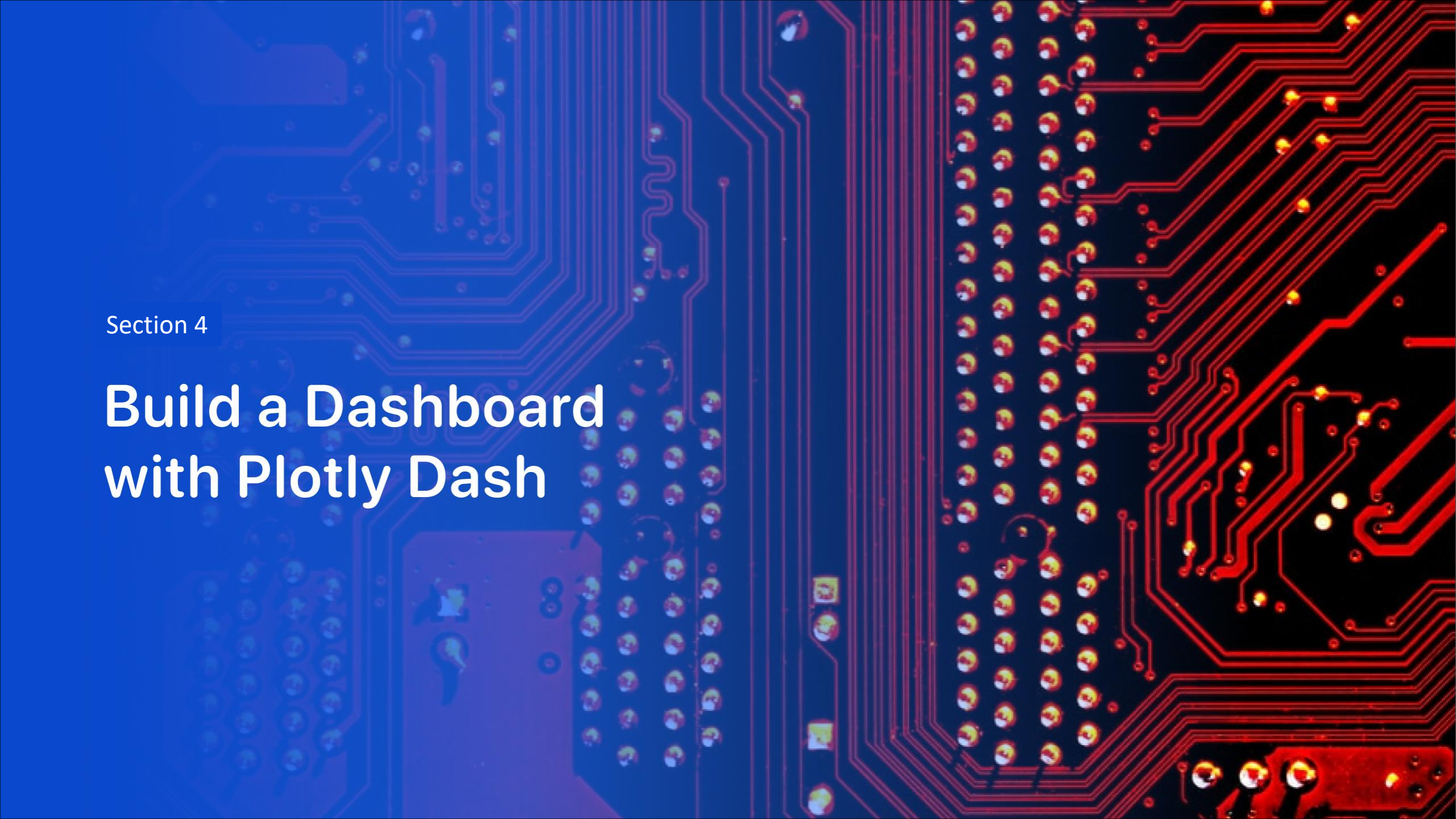
- The green markers in the map gives out successful launch outcomes and red marker gives out failed launch outcomes

Distance between Launch Sites to its proximities

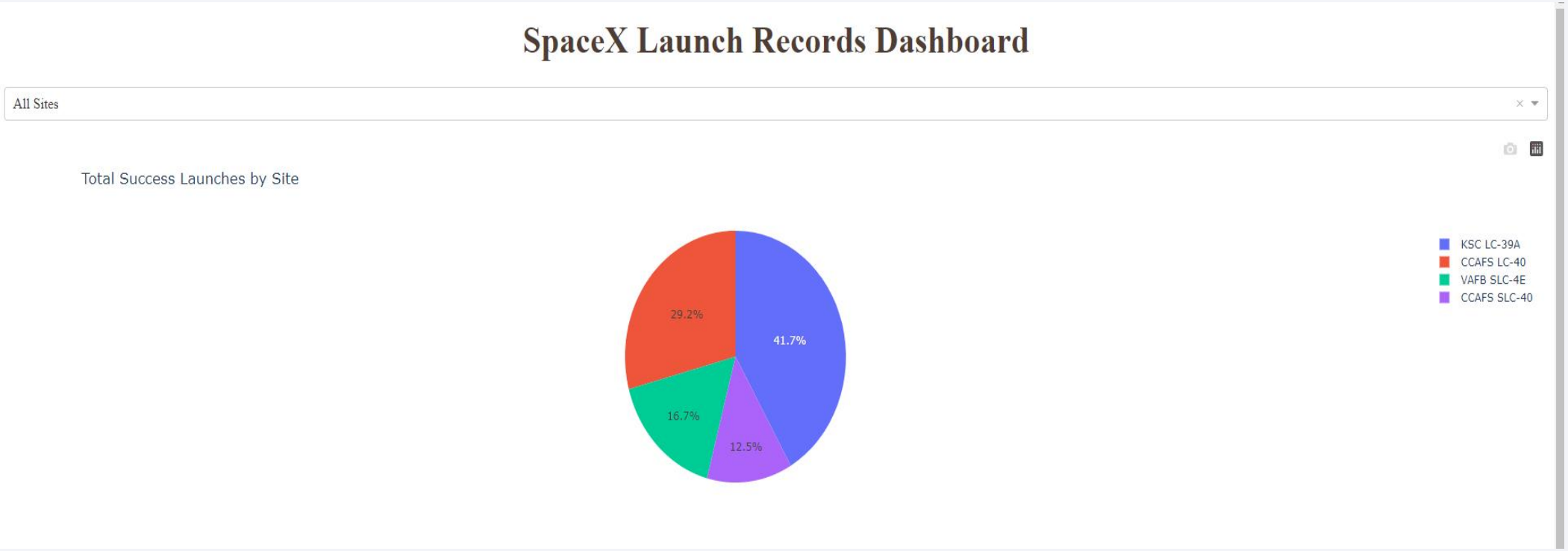
A line is drawn between the launch site and its closest Highway.

Section 4

Build a Dashboard with Plotly Dash

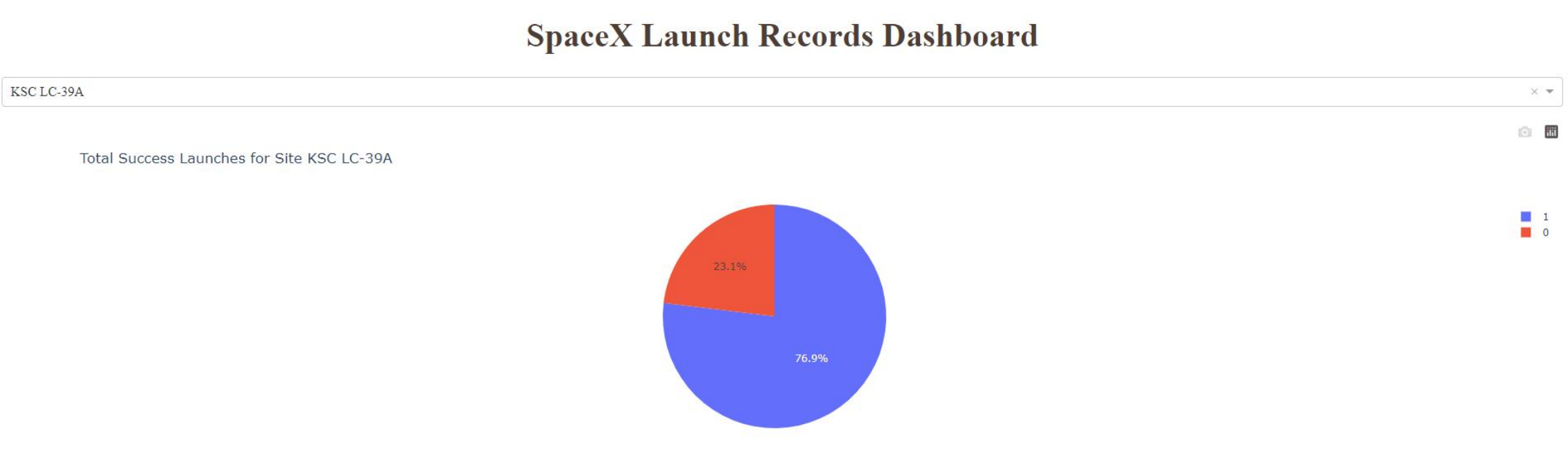


Pie Chart with success rates based on launch sites



This dashboard contains a pie chart with the success rate of rocket launched based on sites.

Pie Chart with Highest Launch Site Success ration



This dashboard contains the pie chart which has the highest successful launches based on Launch Site

Scatter Plot between Class and Payload Mass

This Scatter Plot shows the success rate based on Payload Mass of rockets between 2000 and 6000 KG

Section 5

Predictive Analysis (Classification)

Classification Accuracy

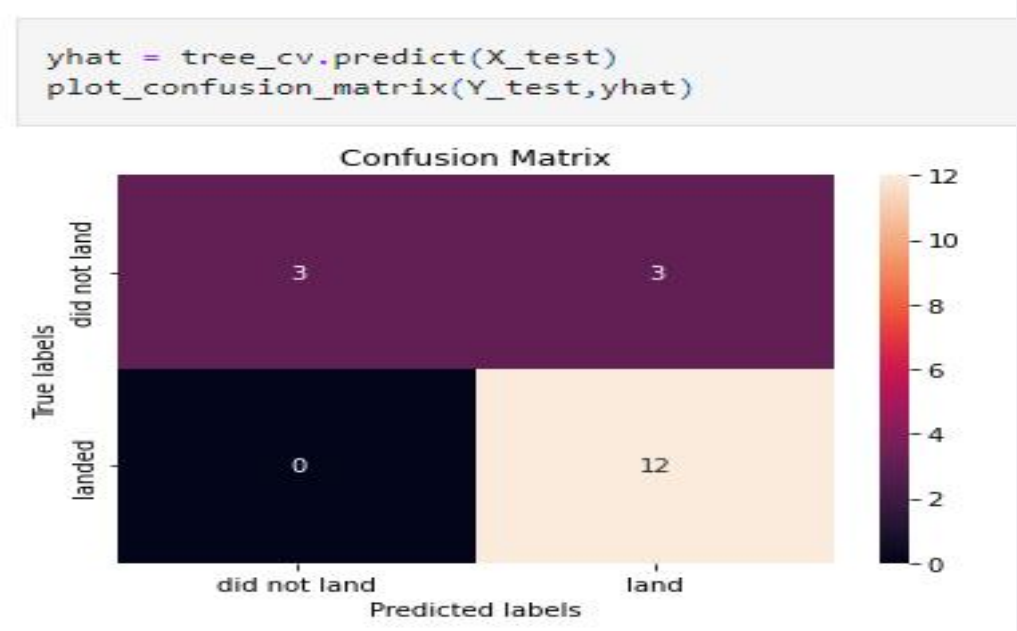
```
print('Accuracy Score : ')
print('Logistic Regression Accuracy Score :',logreg_cv.best_score_)
print('Support Vector Machine Accuracy Score :',svm_cv.best_score_)
print('Decision Tree Accuracy Score :',tree_cv.best_score_)
print('KNN Accuracy Score :',knn_cv.best_score_)
```

```
Accuracy Score :
Logistic Regression Accuracy Score : 0.8464285714285713
Support Vector Machine Accuracy Score : 0.8482142857142856
Decision Tree Accuracy Score : 0.8892857142857145
KNN Accuracy Score : 0.8482142857142858
```

In this, we can see that Decision Tree Algorithm is providing the best accuracy.

Confusion Matrix

```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



- True Positives -12 (landed and land)
- True Negatives - 0 (landed and did not land)
- False Positives – 3 (did not land and did not land)
- False Negatives – 3 (did not land and land)

Conclusions

- This project shows various ways to extract the data using API's and Beautiful Soup. It also shows the usage of pandas and numpy for Data Collection and Analysis.
- Exploratory Data Analysis carried out using Plotly, Seaborn as a part of Data Visualization and data is analyzed using SQL. Both of which has provided good EDA Analysis.
- Getting a better image of the launch sites using the Folium maps. A lot of insights based on locations of Launch Sites on the maps can be seen.
- Used 4 Classification Algorithms using Grid Search which predicts if a rocket launched will be carried out successful or not.
- Decision Tree Algorithm came out with highest accuracy.

Appendix

- Using a lot of in-built libraries like pandas, numpy, seaborn, plotly and many more.
- Used magic sql to execute sql queries in python notebook.

Thank you!