Machine Learning for Rocket Science

Juliano Cunha

31/01/2025

IBM Developer

SKILLS NETWORK

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

EXECUTIVE SUMMARY

- Summary of methodologies
  - Data Collection through API
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization.
  - Exploratory Data Analysis with DASH.
  - Launch Sites Locations Analysis with Folium
  - Predictive Analysis
- Summary of results
  - Exploratory Data Analysis results
  - Dash App and Location Analysis results
  - Predictive Analysis results

INTRODUCTION

- Background and Context
  - SpaceX Falcon 9 rocket launches have a cost of 62 million dollars; while 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
  - How different features influence the landing outcome ?
  - What operational conditions should be in place to ensure a successful landing outcome ?

METHODOLOGY

- Data collection methodology:
  - The data was collected using the SpaceX API.
- Perform data wrangling
  - One-hot encoding was applied to categorical features
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Build, tune and evaluation of the classification models

Data Collection

- The data was collected using various methods
  - The data was done using a get request to the Spacex API.
  - After, 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 in missing values where necessary.

Data Collection – SpaceX API

- We used the get request to the Spacex API to collect data, cleaned the requested data and did some basic data wrangling.

- The link to the notebook is: https://github.com/ultimatejuliano/DataScience_Capstone_SpaceX/blob/8ed85823f71c55063d96e909af3196c39a378506/jupyter-labs-spacex-data-collection-api-v2.ipynb

Data Wrangling

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number of ocurrence of each orbit.
- We created the landing outcome label from outcome column and exported the results as .csv.
- The link to the notebook is: https://github.com/ultimatejuliano/DataScience_Capstone_SpaceX/blob/8ed85823f71c55063d96e909af3196c39a378506/labs-jupyter-spacex-Data%20wrangling-v2.ipynb

Exploratory Data Analysis with SQL

- We loaded the SpaceX dataset into a DB2 database within 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 boostes launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of sucessfull and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names
- The link to the notebook is:

Exploratory Data Analysis with Data Visualization

- We explored the data by visualizing the relationship between payload mass and flight number, flight number and launch site, payload mass and launch site and the relationship between payload mass and orbit for each success rate. We calculated the success rate for each orbit type, the success rate for each orbit type and flight number and also displayed the success rate yearly trend.
- The link to the notebook is:

https://github.com/ultimatejuliano/DataScience_Capstone_SpaceX/blob/8ed85823f71c55063d96e909af3196c39a378506/jupyter-labs-eda-dataviz-v2.ipynb

Plotly Dash App

- We created a pizza plot to show how the succesfull launches are distributed on the launch sites and the success x failure rates for each site.
- We created a scatter plot that shows the correlation between Payload Mass and Sucess for all sites and for each one individually.
- The link to the Dash App notebook is: https://github.com/ultimatejuliano/DataScience_Capstone_SpaceX/blob/1568e2308486b3a96b27f3df75a68fdd4889d33b/dash_app.ipynb

Launch Sites Locations Analysis with Folium

- Using Folium we marked all launch sites on a map. Then we created market clusters for launches within each launch site where green markers indicate a succesfull launch and red markers indicate a failure. We also analysed the proximities of each launch site.
- The link to the notebook is: https://github.com/ultimatejuliano/DataScience_Capstone_SpaceX/blob/8ed85823f71c55063d96e909af3196c39a378506/lab-jupyter-launch-site-location-v2.ipynb

Predictive Analysis

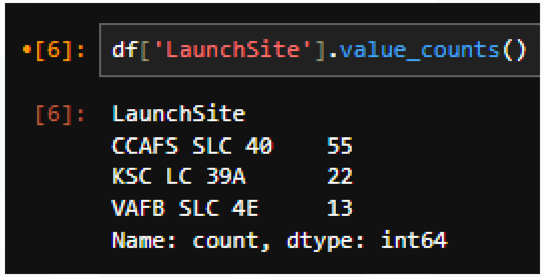
- We loaded the data using numpy and pandas, transformed it and split it into training and testing sets.
- We normalized the data to improve performance of the models.
- After tuning different hyperparameters using GridSearchCV, we chose Decision Trees over Logistic Regression, Suport Vector Machine and K Nearest Neighbors given that it achieved the maximum score in cross validated test data.
- The link to the notebook is: https://github.com/ultimatejuliano/DataScience_Capstone_SpaceX/blob/8ed85823f71c55063d96e909af3196c39a378506/SpaceX-Machine-Learning-Prediction-Part-5-v1.ipynb

Results

- In this section we will present the results from:
  - Exploratory Data Analysis
  - Location Analysis and Dash App
  - Predictive Analysis

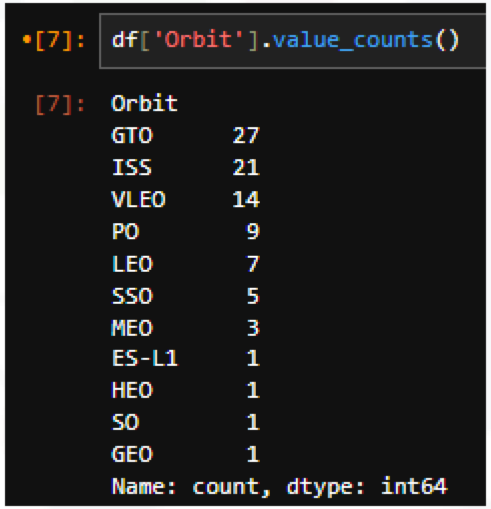
Exploratory Data Analysis With SQL

- We determined the number of launches for each site:



```
•[6]: df['LaunchSite'].value_counts()

 [6]: LaunchSite
      CCAFS SLC 40    55
      KSC LC 39A      22
      VAFB SLC 4E     13
      Name: count, dtype: int64
```

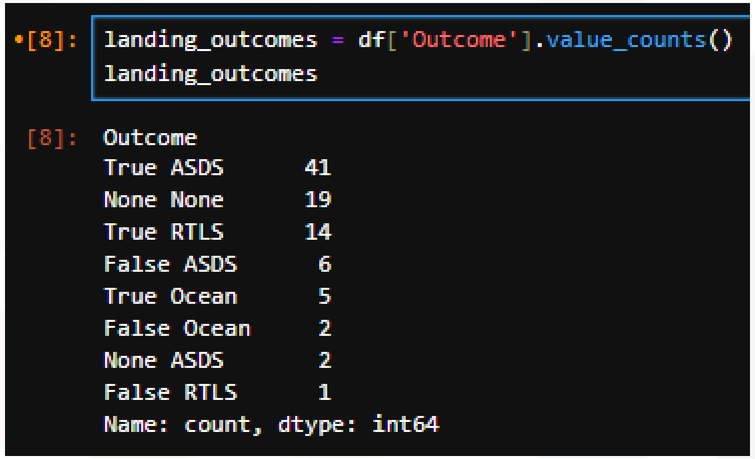
- We determined the number of launches for to each Orbit:



```
•[7]: df['Orbit'].value_counts()

 [7]: Orbit
      GTO      27
      ISS      21
      VLEO     14
      PO        9
      LEO       7
      SSO       5
      MEO       3
      ES-L1     1
      HEO       1
      SO        1
      GEO       1
      Name: count, dtype: int64
```

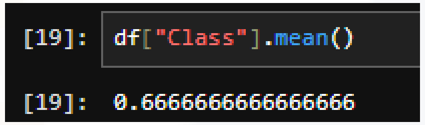
Exploratory Data Analysis With SQL

- We determined the amount of landing outcomes:



```
[8]: landing_outcomes = df['Outcome'].value_counts()
     landing_outcomes

[8]: Outcome
     True ASDS      41
     None None      19
     True RTLS      14
     False ASDS      6
     True Ocean      5
     False Ocean     2
     None ASDS       2
     False RTLS      1
     Name: count, dtype: int64
```

- We determined the general sucess rate within the sample:

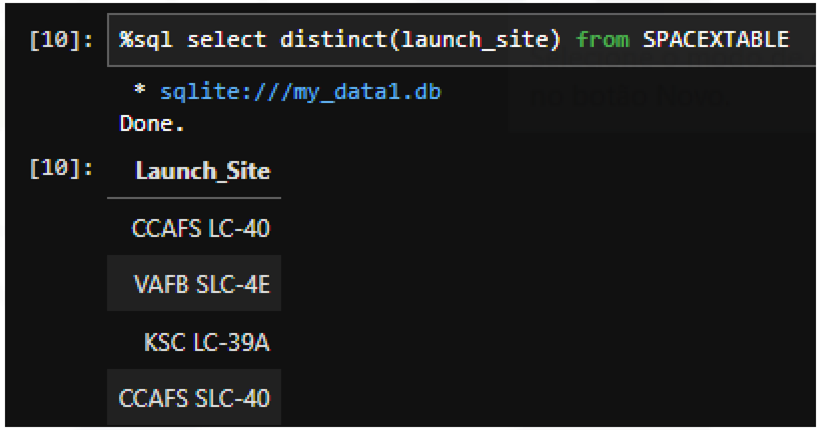


```
[19]: df["Class"].mean()

[19]: 0.6666666666666666
```

Exploratory Data Analysis With SQL

- We uploaded the data to a DB2 database table and then used sql querys to perform an exploratory data analysis.

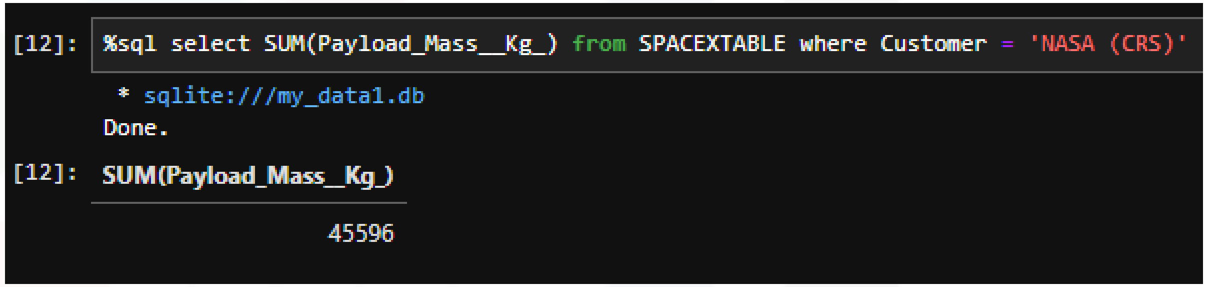
- We created a query to verify what are the launch sites:



```
[10]: %sql select distinct(launch_site) from SPACEXTABLE

       * sqlite:///my_data1.db
      Done.

[10]:  Launch_Site
       CCAFS LC-40
       VAFB SLC-4E
        KSC LC-39A
      CCAFS SLC-40
```

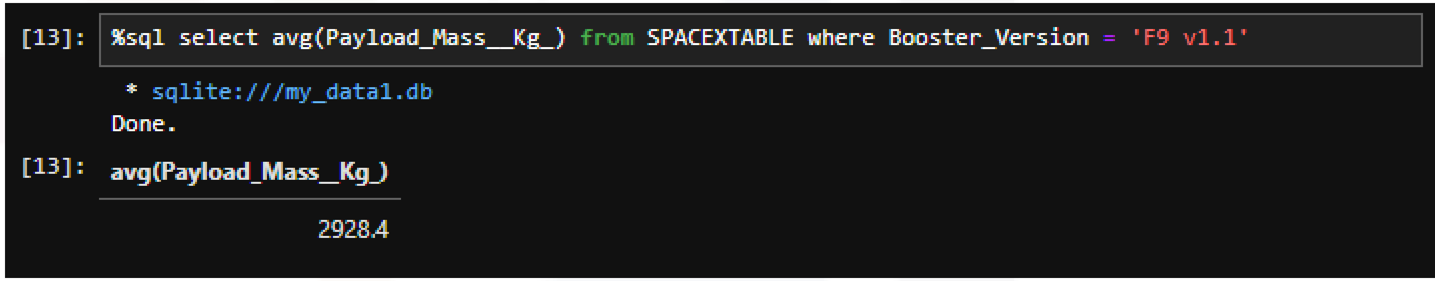
Exploratory Data Analysis With SQL

- We calculated the total payload mass carried by boosters launched by NASA (CRS):



```
[12]: %sql select SUM(Payload_Mass__Kg_) from SPACEXTABLE where Customer = 'NASA (CRS)'

       * sqlite:///my_data1.db
      Done.

[12]: SUM(Payload_Mass__Kg_)
                     45596
```

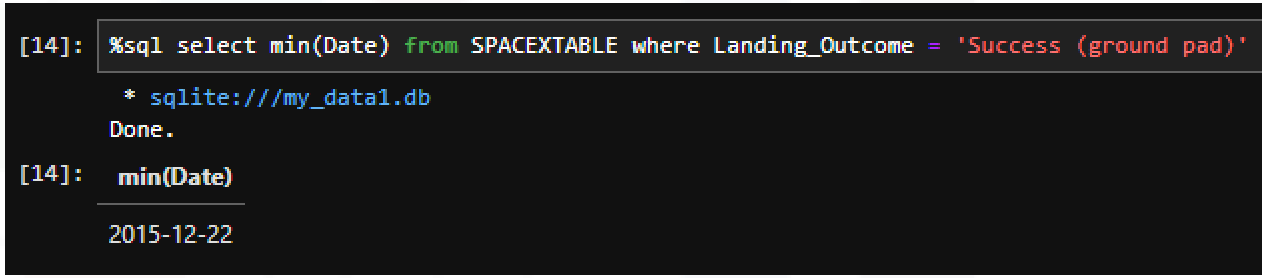
- We calculated the average payload mass carried by booster version F9 v1.1:



```
[13]: %sql select avg(Payload_Mass__Kg_) from SPACEXTABLE where Booster_Version = 'F9 v1.1'

       * sqlite:///my_data1.db
      Done.

[13]: avg(Payload_Mass__Kg_)
                    2928.4
```

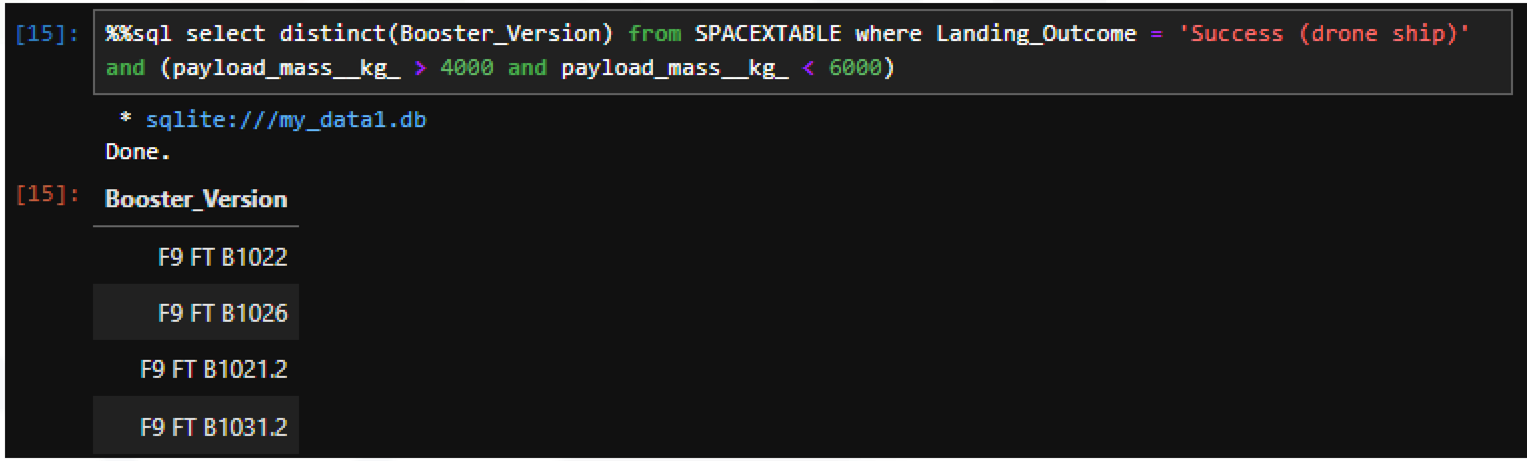
Exploratory Data Analysis With SQL

- We verified the date when the first succesful landing outcome in ground pad was achieved:



```
[14]: %sql select min(Date) from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)'

       * sqlite:///my_data1.db
      Done.

[14]:  min(Date)
      2015-12-22
```

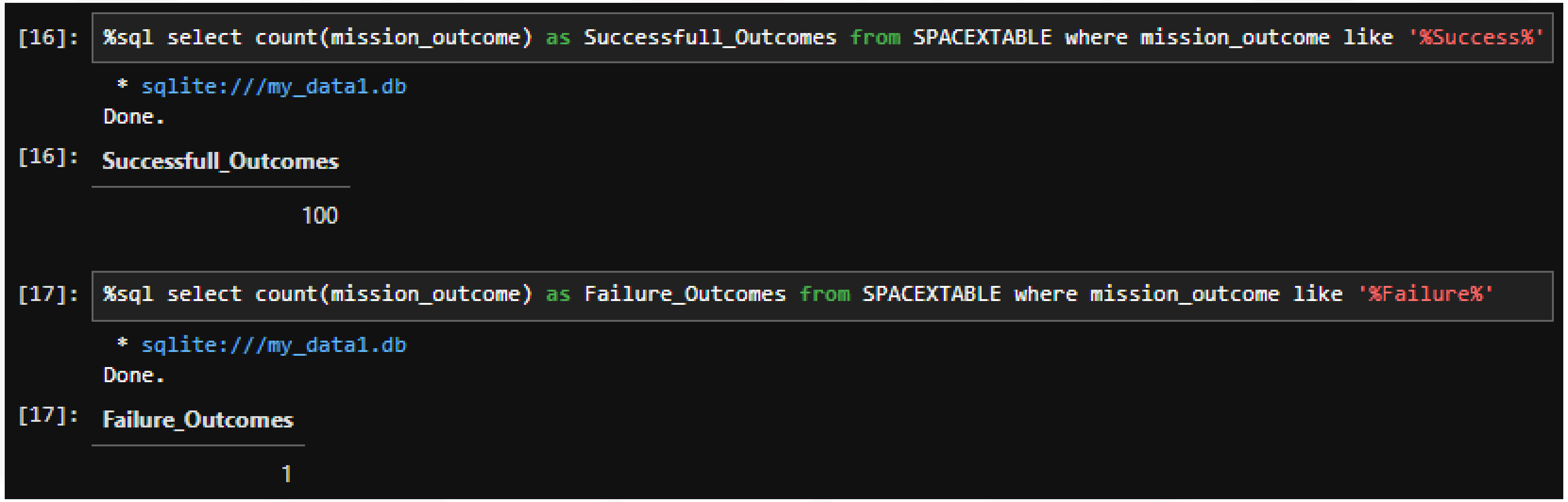
- We verified the boosters which had success with a payload mass between 4000 and 6000:



```
[15]: %%sql select distinct(Booster_Version) from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)'
      and (payload_mass__kg_ > 4000 and payload_mass__kg_ < 6000)

       * sqlite:///my_data1.db
      Done.

[15]: Booster_Version
        F9 FT B1022
        F9 FT B1026
      F9 FT B1021.2
      F9 FT B1031.2
```

Exploratory Data Analysis With SQL

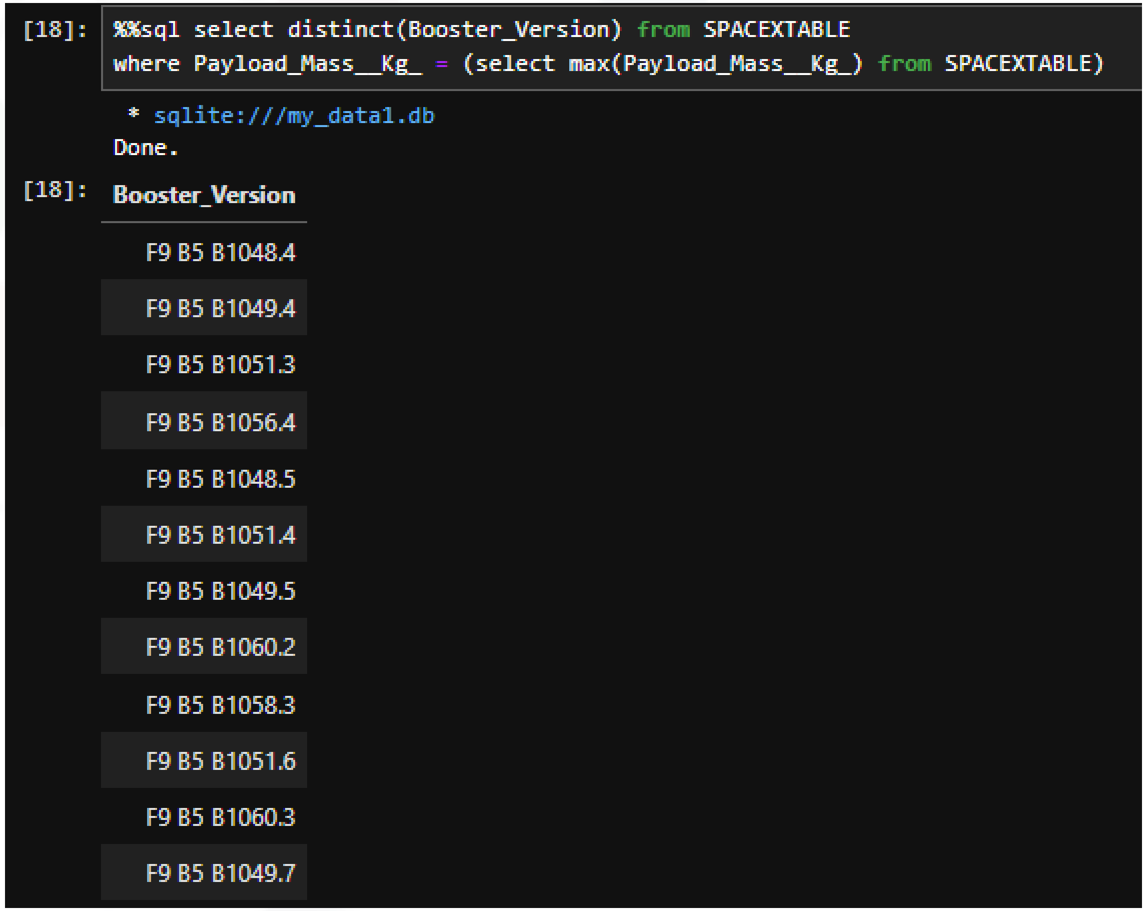
- We verified the amount of succesfull and failure landings:



```
[16]: %sql select count(mission_outcome) as Successfull_Outcomes from SPACEXTABLE where mission_outcome like '%Success%'

       * sqlite:///my_data1.db
      Done.

[16]: Successfull_Outcomes
                       100

[17]: %sql select count(mission_outcome) as Failure_Outcomes from SPACEXTABLE where mission_outcome like '%Failure%'

       * sqlite:///my_data1.db
      Done.

[17]: Failure_Outcomes
                     1
```

Exploratory Data Analysis With SQL

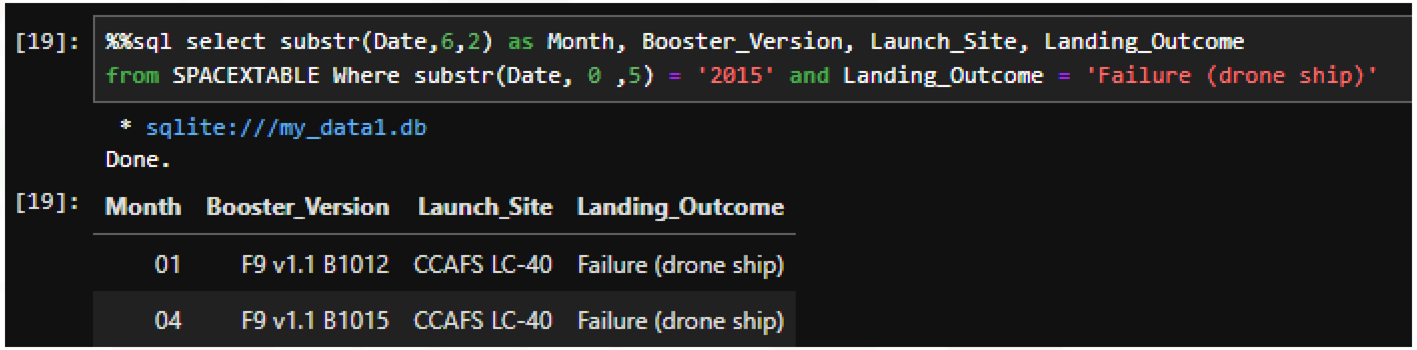
- We listed the booster versions that managed to handle the maximum payload mass:



```
[18]: %%sql select distinct(Booster_Version) from SPACEXTABLE
      where Payload_Mass__Kg_ = (select max(Payload_Mass__Kg_) from SPACEXTABLE)

       * sqlite:///my_data1.db
      Done.

[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
```

Exploratory Data Analysis With SQL

- We listed the records displaying month, failure landing outcomes in drone ship, booster versions and launch site:

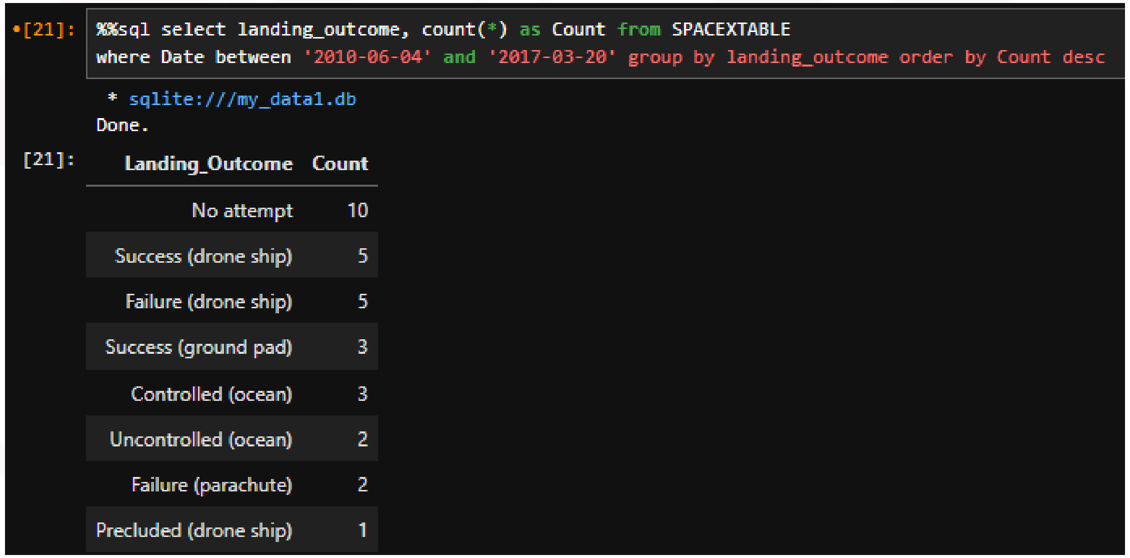


```
[19]: %%sql select substr(Date,6,2) as Month, Booster_Version, Launch_Site, Landing_Outcome
      from SPACEXTABLE Where substr(Date, 0 ,5) = '2015' and Landing_Outcome = 'Failure (drone ship)'

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

[19]:

Month	Booster_Version	Launch_Site	Landing_Outcome
01	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

- We ranked the count of landing outcomes between 2010/06/04 and 2017/03/20 in descending order:



```
•[21]: %%sql select landing_outcome, count(*) as Count from SPACEXTABLE
       where Date between '2010-06-04' and '2017-03-20' group by landing_outcome order by Count desc

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

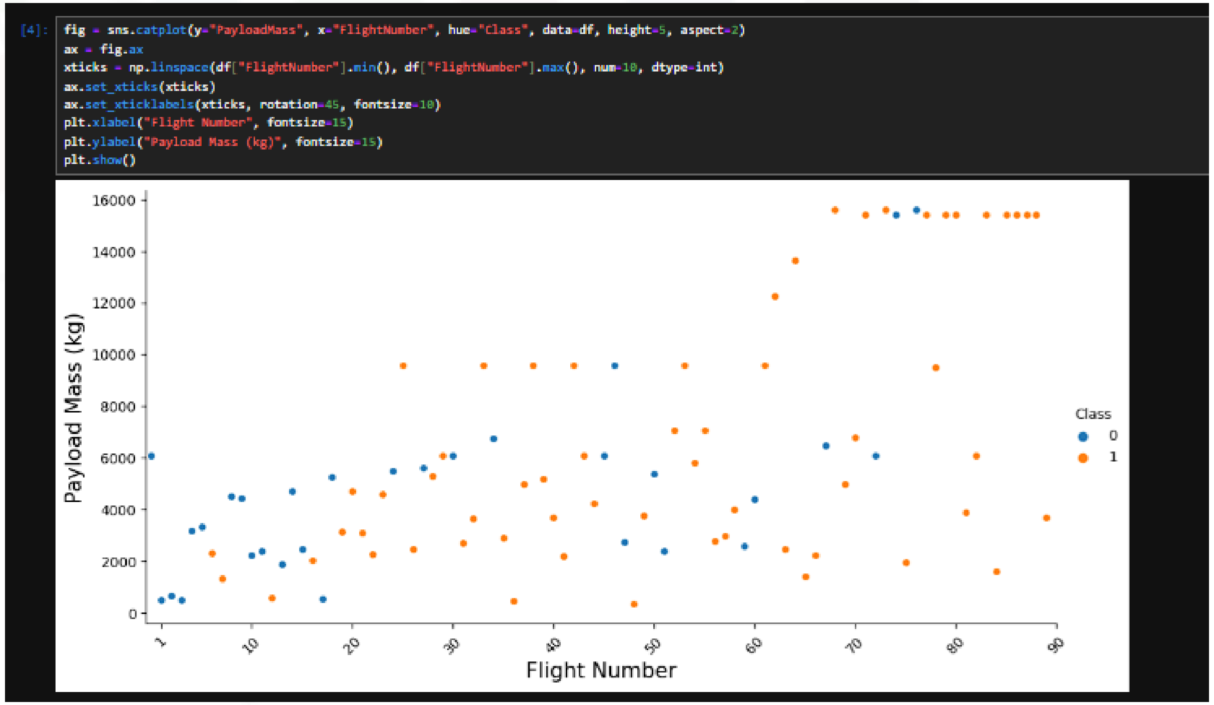
[21]:

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

Exploratory Data Analysis With DataViz

- We made a categorical plot to summarise the relationship between flight number and payload mass:

- We found that the larger the number of flights at a launch site, the bigger the sucess rate.

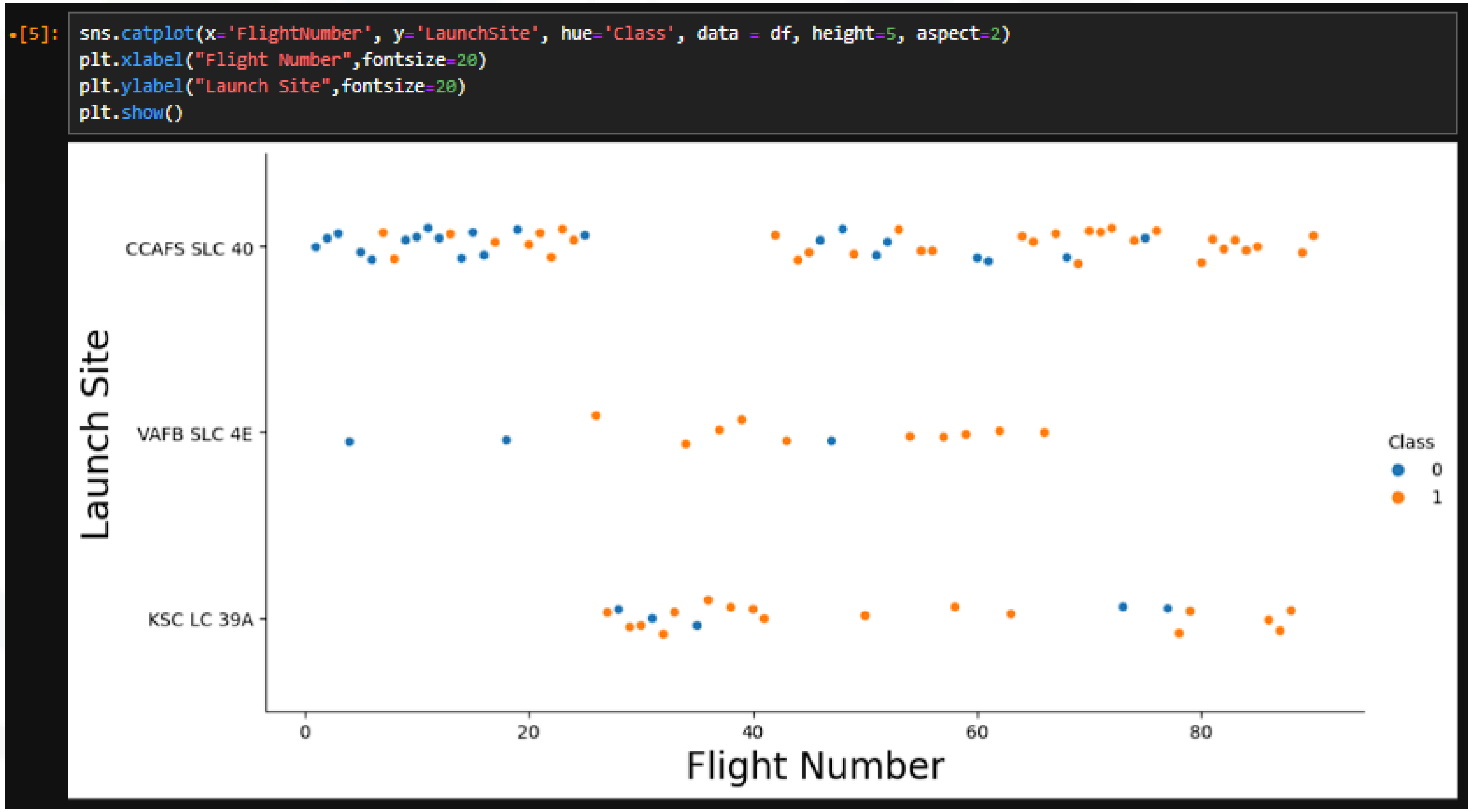


```
[4]: fig = sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, height=5, aspect=2)
ax = fig.ax
xticks = np.linspace(df["FlightNumber"].min(), df["FlightNumber"].max(), num=10, dtype=int)
ax.set_xticks(xticks)
ax.set_xticklabels(xticks, rotation=45, fontsize=10)
plt.xlabel("Flight Number", fontsize=15)
plt.ylabel("Payload Mass (kg)", fontsize=15)
plt.show()
```

Exploratory Data Analysis With DataViz

- We made a categorical plot to summarise the relationship between flight number and launch site:

- For each launch site the success rate is bigger with more flight number.

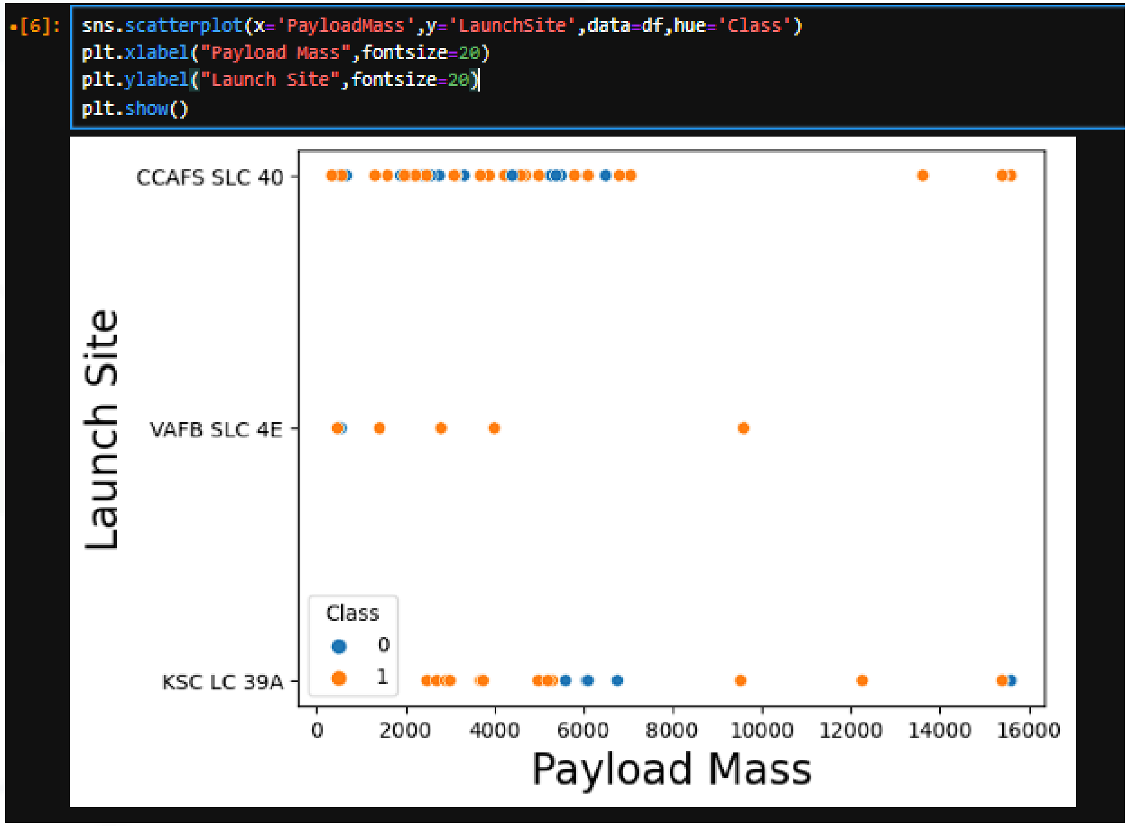


```
[5]: sns.catplot(x='FlightNumber', y='LaunchSite', hue='Class', data = df, height=5, aspect=2)
     plt.xlabel("Flight Number",fontsize=20)
     plt.ylabel("Launch Site",fontsize=20)
     plt.show()
```

Exploratory Data Analysis With DataViz

- We made a categorical plot to summarise the relationship between payload mass and launch site:

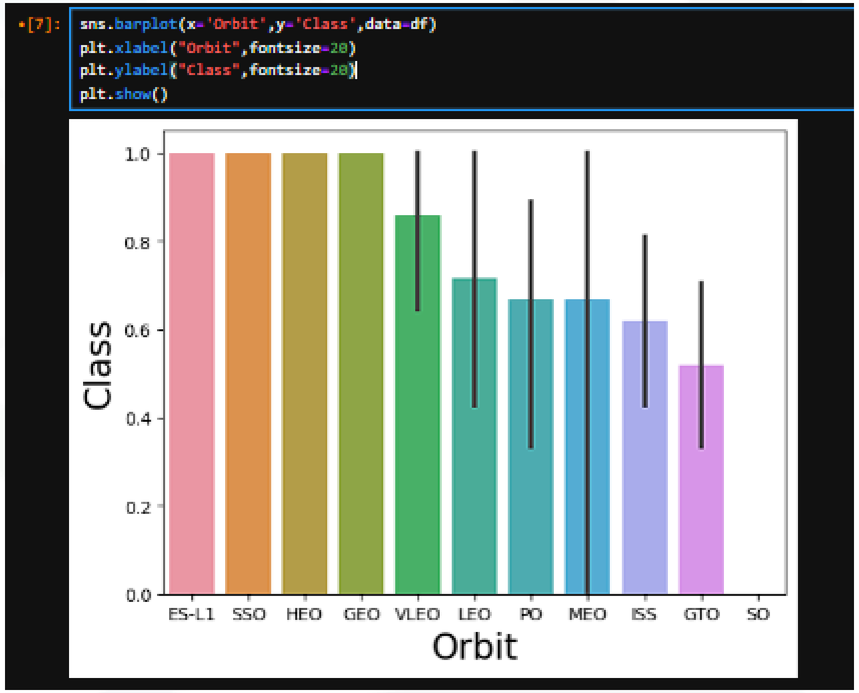
- For launch site CCAFS SLC 40 the success rate is bigger with more payload mass.



```
[6]: sns.scatterplot(x='PayloadMass',y='LaunchSite',data=df,hue='Class')
     plt.xlabel("Payload Mass",fontsize=20)
     plt.ylabel("Launch Site",fontsize=20)
     plt.show()
```

Exploratory Data Analysis With DataViz

- We made a bar plot to show the sucess rate for each Orbit:



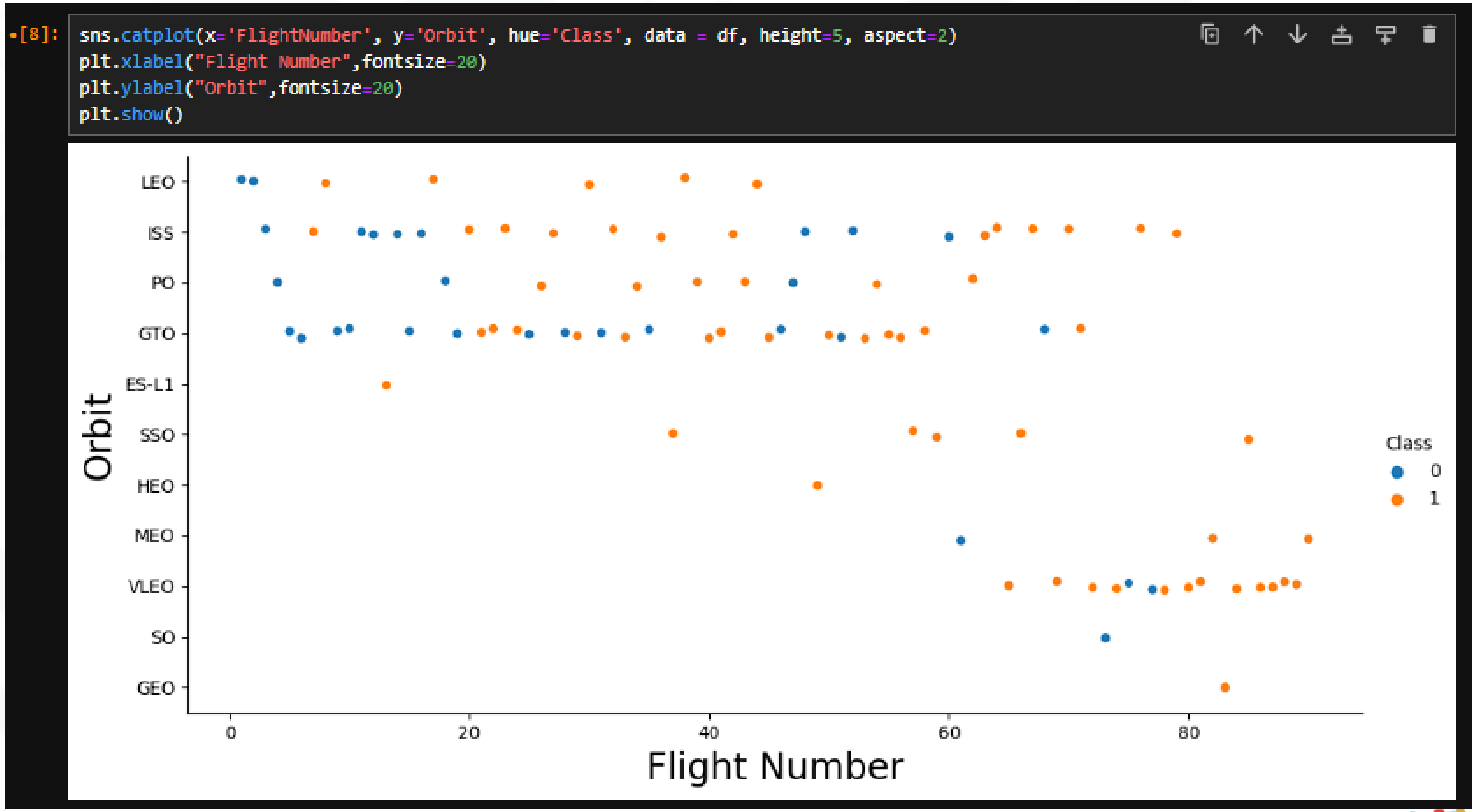
```
•[7]: sns.barplot(x='Orbit',y='Class',data=df)
      plt.xlabel("Orbit",fontsize=20)
      plt.ylabel("Class",fontsize=20)
      plt.show()
```

- ES-L1, SSO, HEO and GEO, are the orbits types with the most succesfull rate.

Exploratory Data Analysis With DataViz

- We made a categorical plot to summarise the relationship between flight number and orbit type:

- The plot suggests that in the LEO orbit, the sucess grows with the number flights, however, in the GTO orbit it suggests no relationship at all.

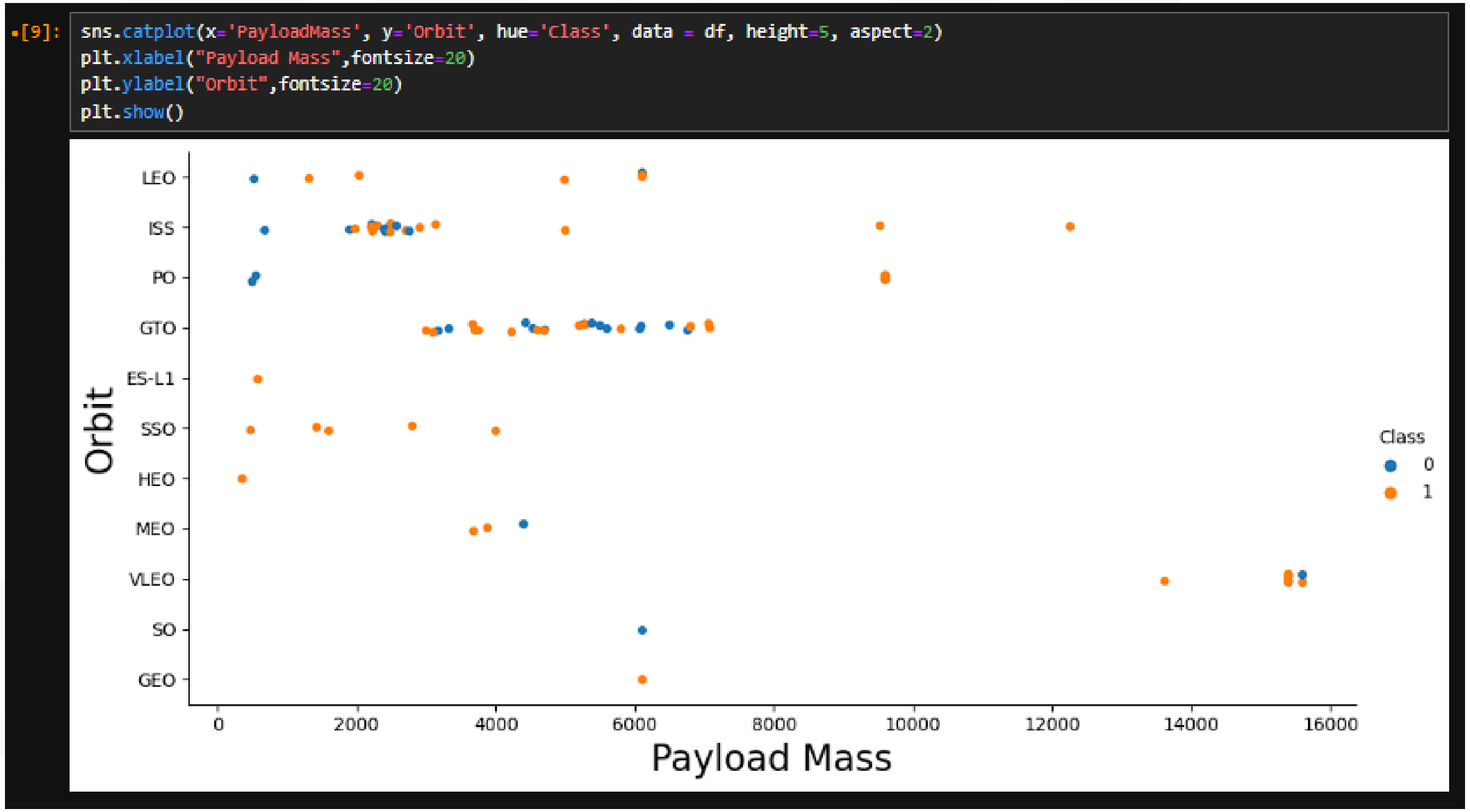


```
[8]:
sns.catplot(x='FlightNumber', y='Orbit', hue='Class', data = df, height=5, aspect=2)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

Exploratory Data Analysis With DataViz

- We made a categorical plot to summarise the relationship between payload mass and orbit type:

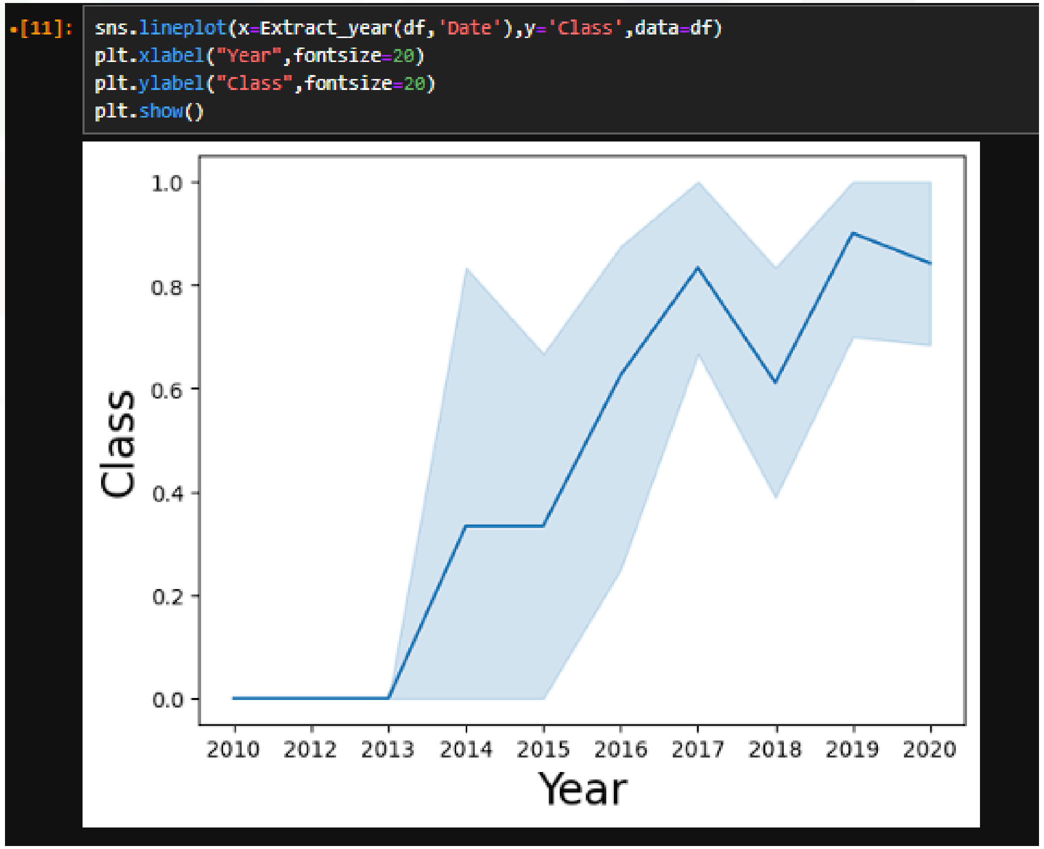
- We can observe that with heavy payloads, the successful landing are more for LEO, ISS and PO orbits.



```
[9]: sns.catplot(x='PayloadMass', y='Orbit', hue='Class', data = df, height=5, aspect=2)
     plt.xlabel("Payload Mass",fontsize=20)
     plt.ylabel("Orbit",fontsize=20)
     plt.show()
```

Exploratory Data Analysis With DataViz

- We made a line plot to show the yearly trend of the success rate:



```
[11]: sns.lineplot(x=Extract_year(df,'Date'),y='Class',data=df)
      plt.xlabel("Year",fontsize=20)
      plt.ylabel("Class",fontsize=20)
      plt.show()
```

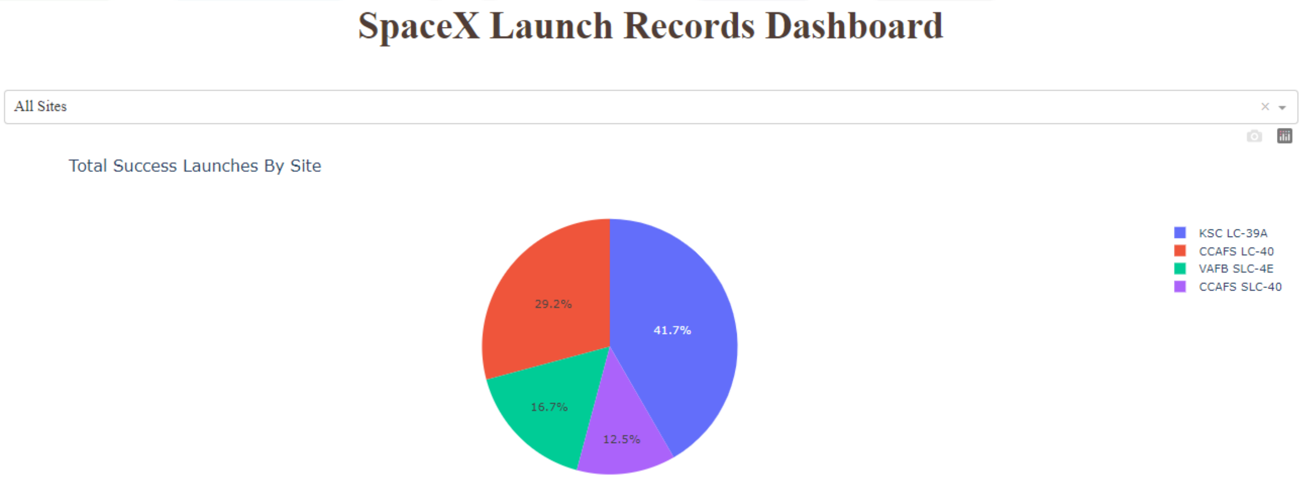
- We can observe that the sucess rate increased considerably from 2013 up to 2020.

Interactive Visual Analytics - DASH

https://github.com/ultimatejuliano/DataScience_Capstone_SpaceX/blob/1568e2308486b3a96b27f3df75a68fdd4889d33b/dash_app.ipynb

DASHBOARD TAB 1

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

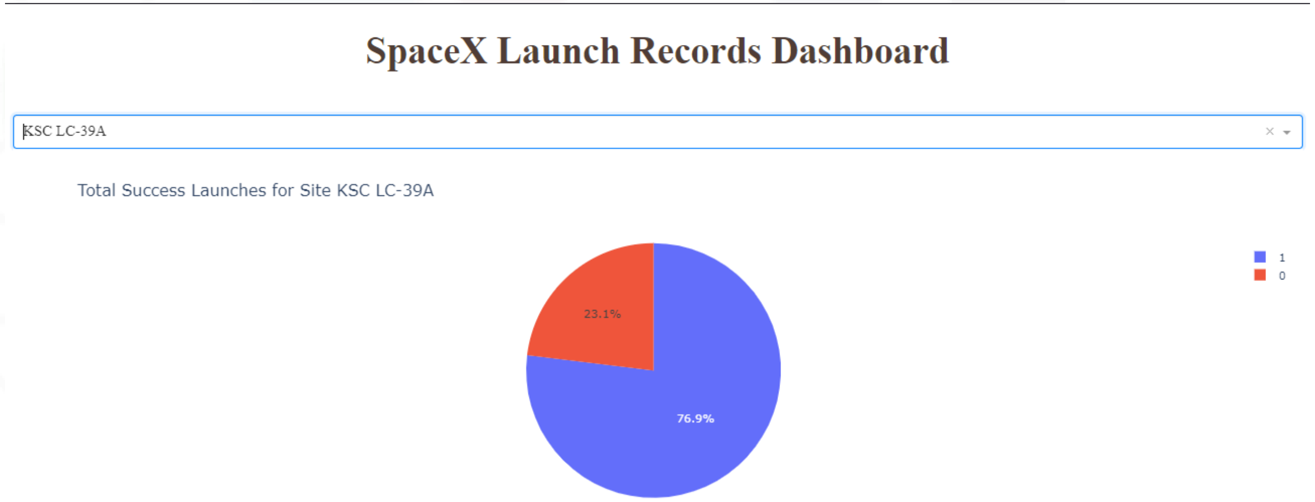


SpaceX Launch Records Dashboard

All Sites

DASHBOARD TAB 2

- KSC LC-39A had a sucess rate of 76.9%



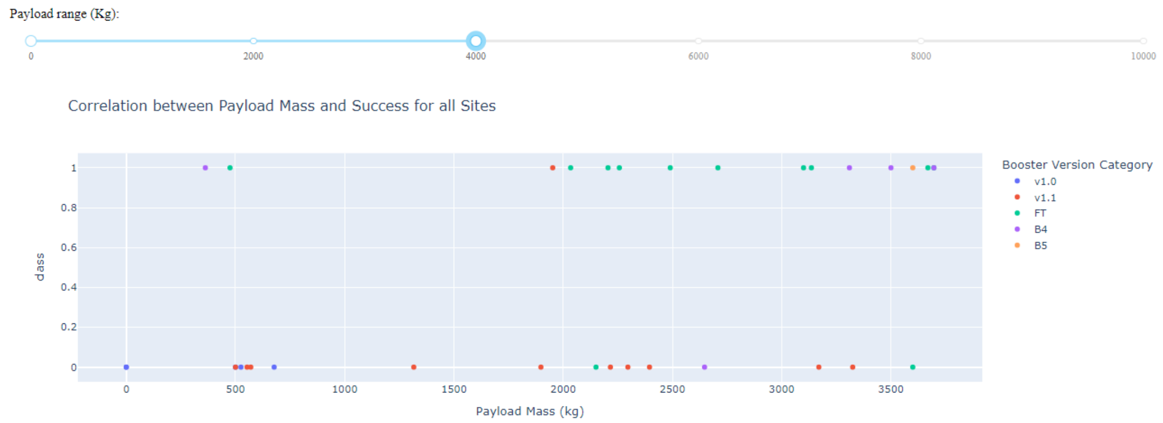
SpaceX Launch Records Dashboard

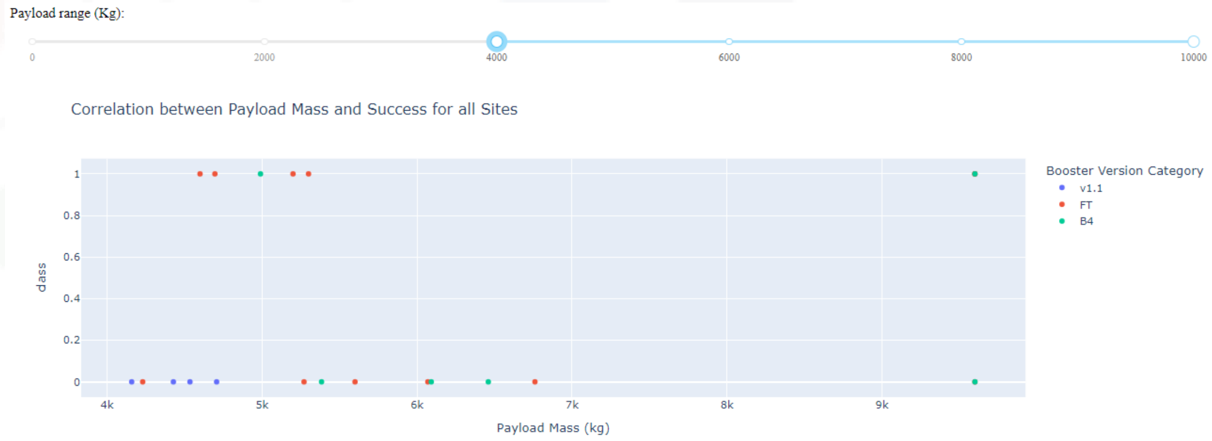
KSC LC-39A

Total Success Launches for Site KSC LC-39A

DASHBOARD TAB 3

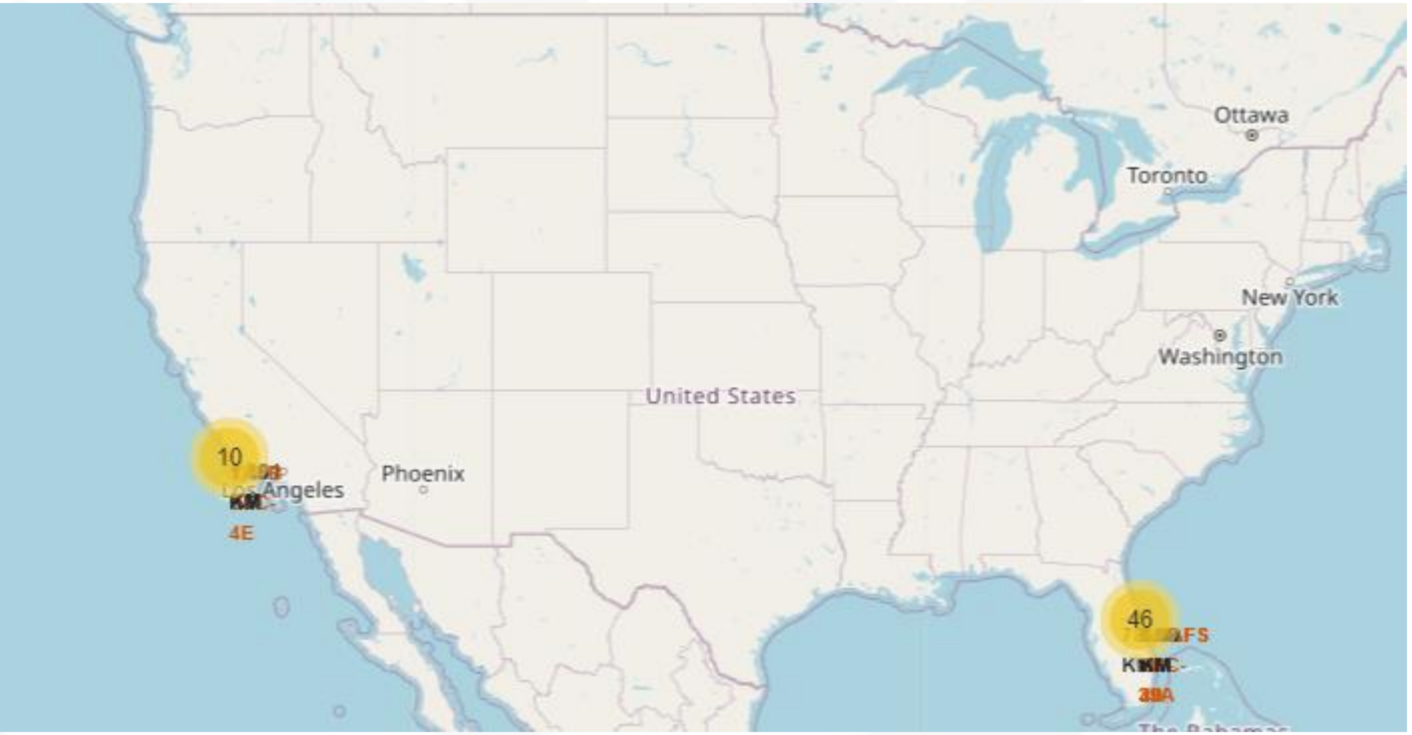
- We can see that the sucess rates from low weighted payloads are higher that those of heavy weighted payloads.





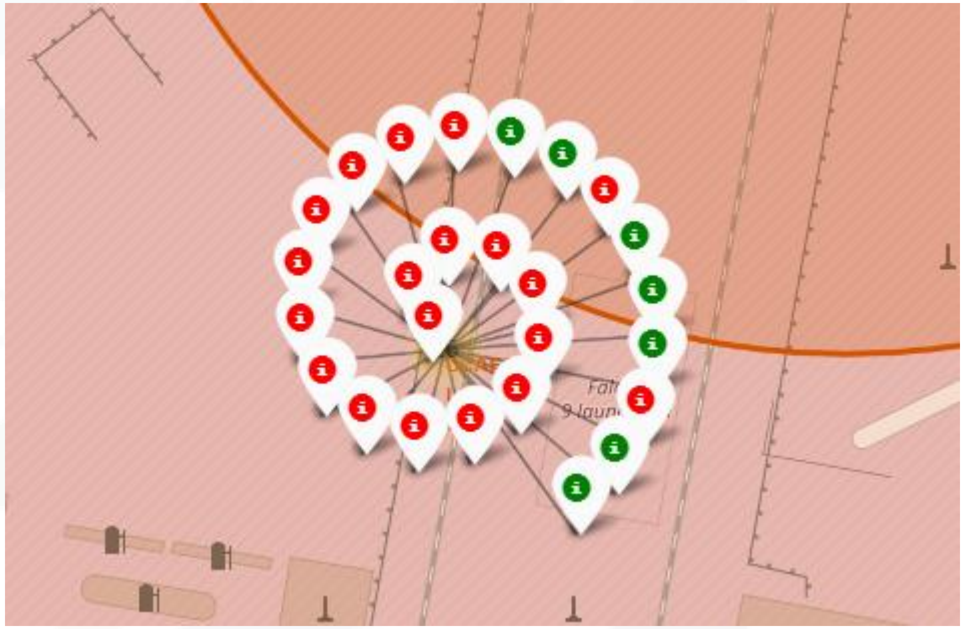
Interactive Visual Analytics - Folium

- We can see that the launch sites are located on the coasts of USA.



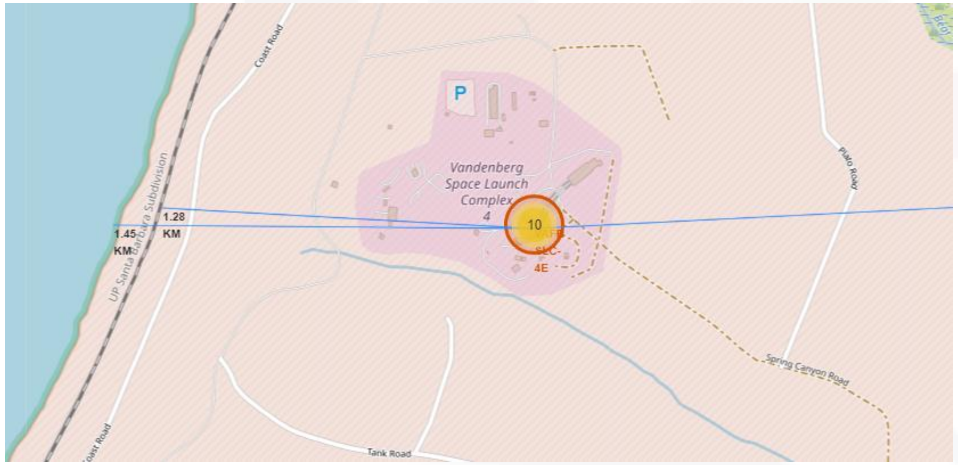
Interactive Visual Analytics - Folium

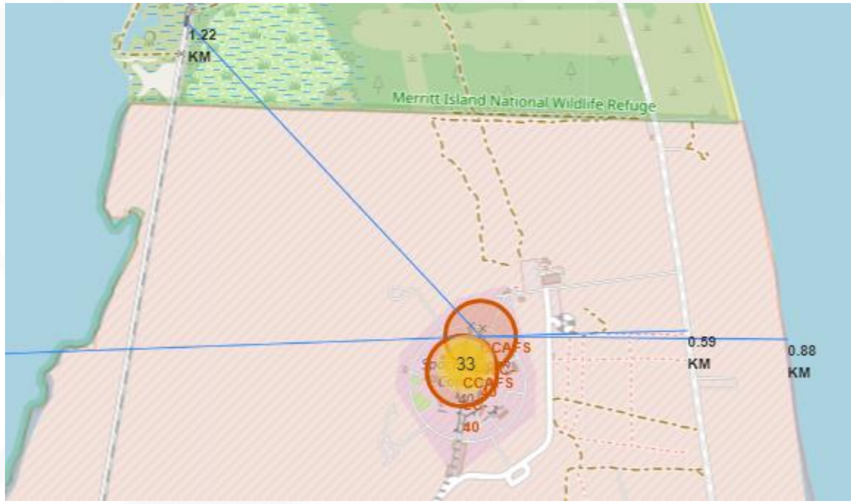
- For a given launch site, a green marks shows a sucessful launch and a red one shows a failure.



Interactive Visual Analytics - Folium

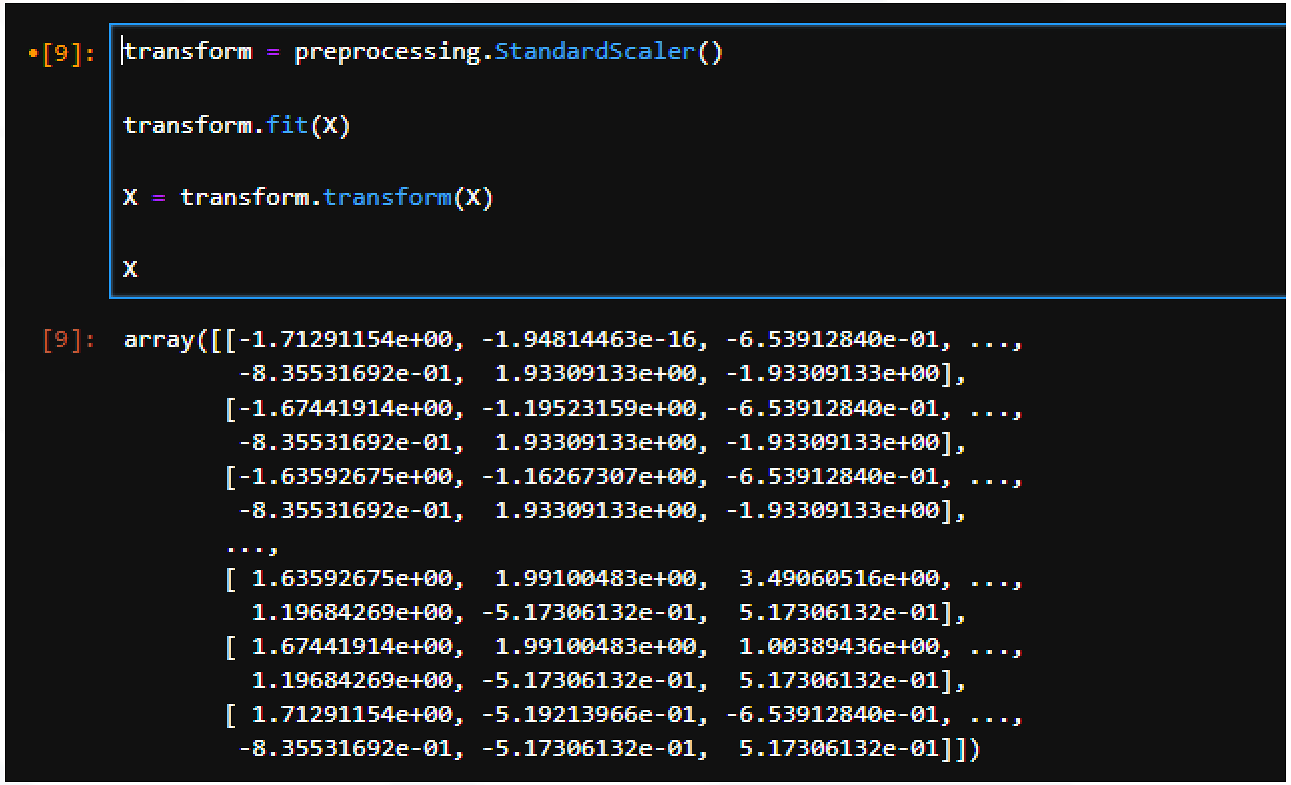
- Are launch sites in close proximity to railways ? No.
- Are launch sites in close proximity to highways ? No.
- Are launch sites in close proximity to coastline ? Yes.
- Do launch sites keep a certain distance away from cities ? Yes.





Predictive Analysis Results

- We started the predictive analysis by standardizing the features to achieve better performance training the models later.

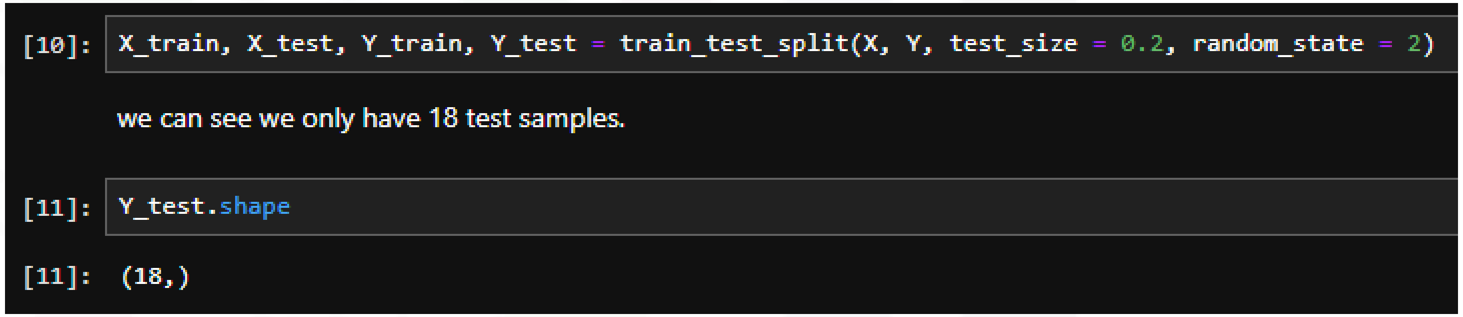


```
•[9]: transform = preprocessing.StandardScaler()

      transform.fit(X)

      X = transform.transform(X)

      X
```

```
[9]:  array([[-1.71291154e+00, -1.94814463e-16, -6.53912840e-01, ...,
              -8.35531692e-01,  1.93309133e+00, -1.93309133e+00],
             [-1.67441914e+00, -1.19523159e+00, -6.53912840e-01, ...,
              -8.35531692e-01,  1.93309133e+00, -1.93309133e+00],
             [-1.63592675e+00, -1.16267307e+00, -6.53912840e-01, ...,
              -8.35531692e-01,  1.93309133e+00, -1.93309133e+00],
             ...,
             [ 1.63592675e+00,  1.99100483e+00,  3.49060516e+00, ...,
               1.19684269e+00, -5.17306132e-01,  5.17306132e-01],
             [ 1.67441914e+00,  1.99100483e+00,  1.00389436e+00, ...,
               1.19684269e+00, -5.17306132e-01,  5.17306132e-01],
             [ 1.71291154e+00, -5.19213966e-01, -6.53912840e-01, ...,
              -8.35531692e-01, -5.17306132e-01,  5.17306132e-01]])
```

Predictive Analysis Results

- We split the data between training and testing sets:



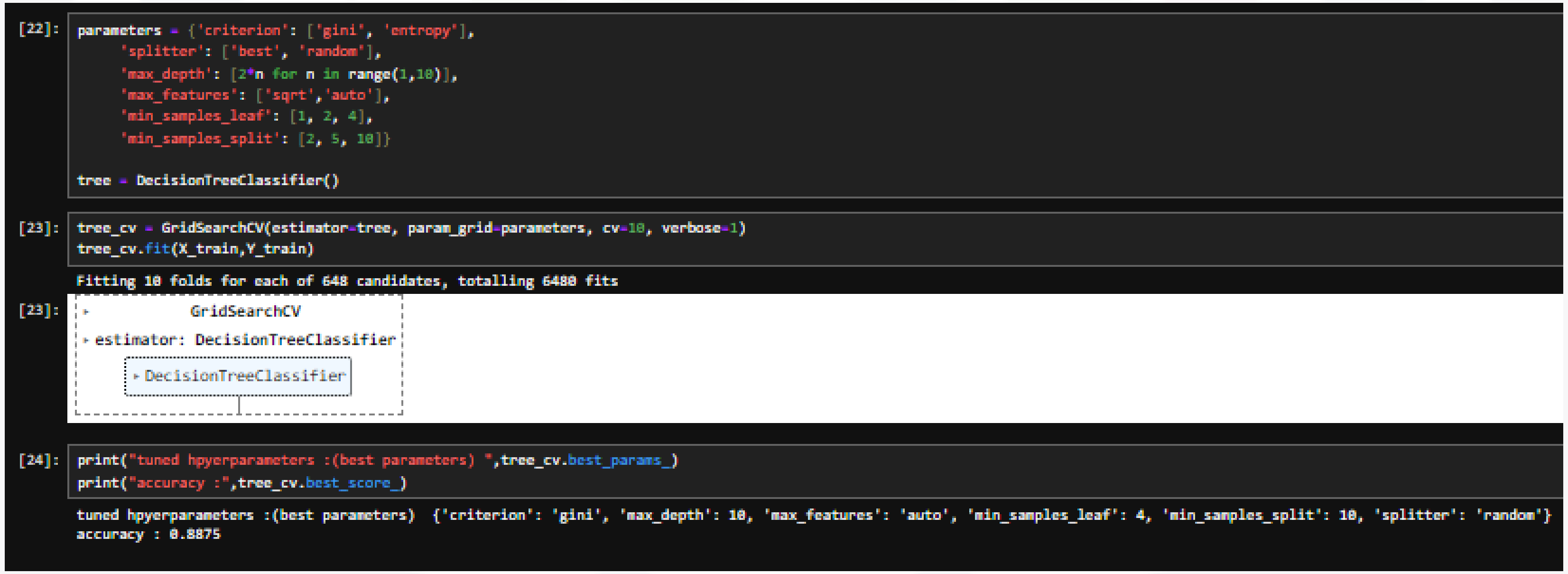
```
[10]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 2)
```

we can see we only have 18 test samples.

```
[11]: Y_test.shape

[11]: (18,)
```

Predictive Analysis Results

- We used GridSearchCV to tune the hyperparameters of Logistic Regression, SVM, Decision Trees and KNN models. Decision Trees was the model that achieved the best score within the cross validated test data:



```
[22]: parameters = {'criterion': ['gini', 'entropy'],
           'splitter': ['best', 'random'],
           'max_depth': [2*n for n in range(1,10)],
           'max_features': ['sqrt','auto'],
           'min_samples_leaf': [1, 2, 4],
           'min_samples_split': [2, 5, 10]}

      tree = DecisionTreeClassifier()
```

```
[23]: tree_cv = GridSearchCV(estimator=tree, param_grid=parameters, cv=10, verbose=1)
      tree_cv.fit(X_train,Y_train)

      Fitting 10 folds for each of 648 candidates, totalling 6480 fits
```

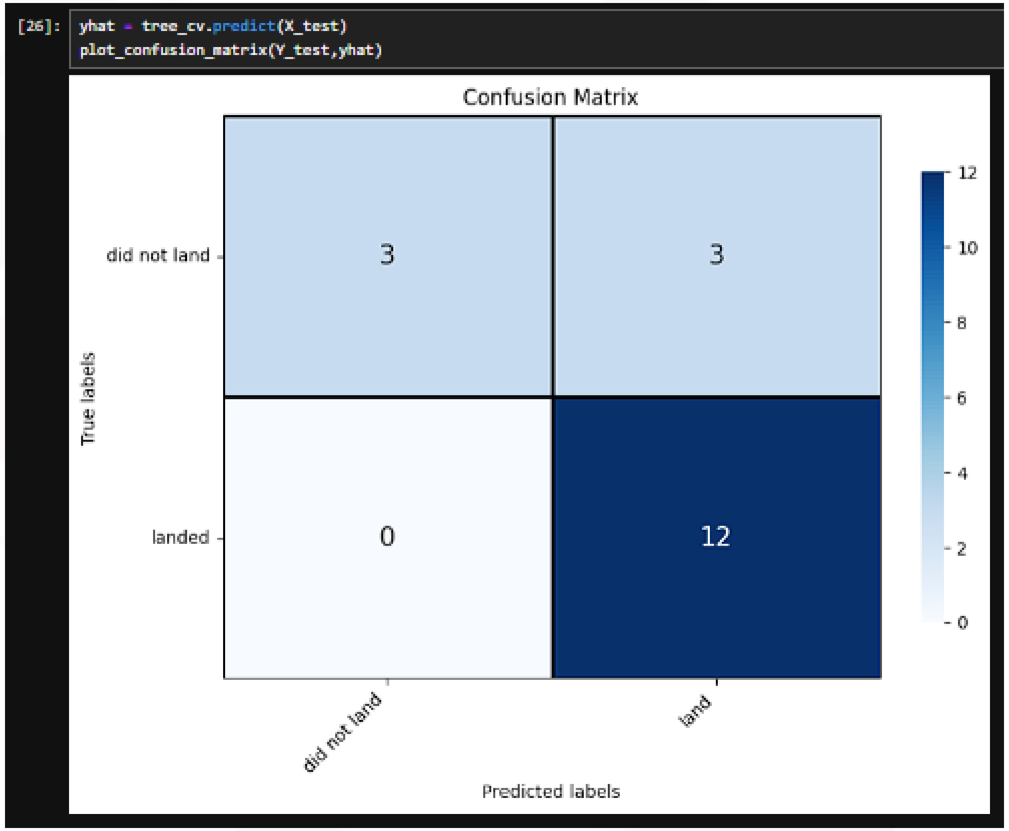
```
[23]: ▸        GridSearchCV
      ▸ estimator: DecisionTreeClassifier
            ▸ DecisionTreeClassifier
```

```
[24]: print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
      print("accuracy :",tree_cv.best_score_)

      tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
      accuracy : 0.8875
```

Predictive Analysis Results

- We made a confusion matrix for the Decision Trees model:

- We were able to realize the model has great accuracy for predicting the launches that landed, however we have a problem with the false positives.



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

CONCLUSION

- We can conclude that:
  - The larger the number of flights, the greater the success rate at a launch site.
  - Launch success rate started to increase in 2013 and went up till 2020.
  - Orbits ES-L1, GEO, GEO, SSO and VLEO had the most success rates.
  - KSC LC-39A had the most successful launches of any sites.
  - The Decision Tree is the best classification model for this challenge.