IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

KDennisL
2023.06.09

Outline

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

Executive Summary

- The following Methodologies were used through this study:
  - Data Collection using APIs
  - Data Collection using Web Scraping Technologies
  - Data Wrangling
  - Exploratory Data Analysis using SQL
  - Exploratory Data Analysis using Data Visualization Technologies
  - Interactive Visual Analytics with Folium
  - Interactive Visual Analytics with Plotly Sash
  - Machine Learning for Predicting Results
- Summary of all results from
  - Exploratory Data Analysis
  - Interactive Visual Analytics
  - Predicting using Machine Learning

Introduction

- Project background and context
  - Commercial Space is evolving and makes travel to space affordable for everyone
  - The costs for launches are relatively expensive for most companies with 165 million dollars
  - SpaceX on the other hand only pays 62 million dollars as they can reuse the first stage

- Problems you want to find answers
  - What factors are important for successfully landing a Falcon 9?
  - How are these connected?
  - Can we predict the outcome of a landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected from two urls:
    - https://api.spacexdata.com/v4/launches/past using a get request &
    - https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches using Webscraping approaches
- Perform data wrangling
  - Data was cleaned and a class label was created:
    - Class = 0 for unsuccessful landing
    - Class = 1 for successful landing
- Perform exploratory data analysis (EDA) using visualization and SQL

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Collected data was scaled and split into Training and Test Dataset
  - Four different ML models were optimized by GridSearch
  - Evaluation was performed using Confusion Matrix and test data accuracy

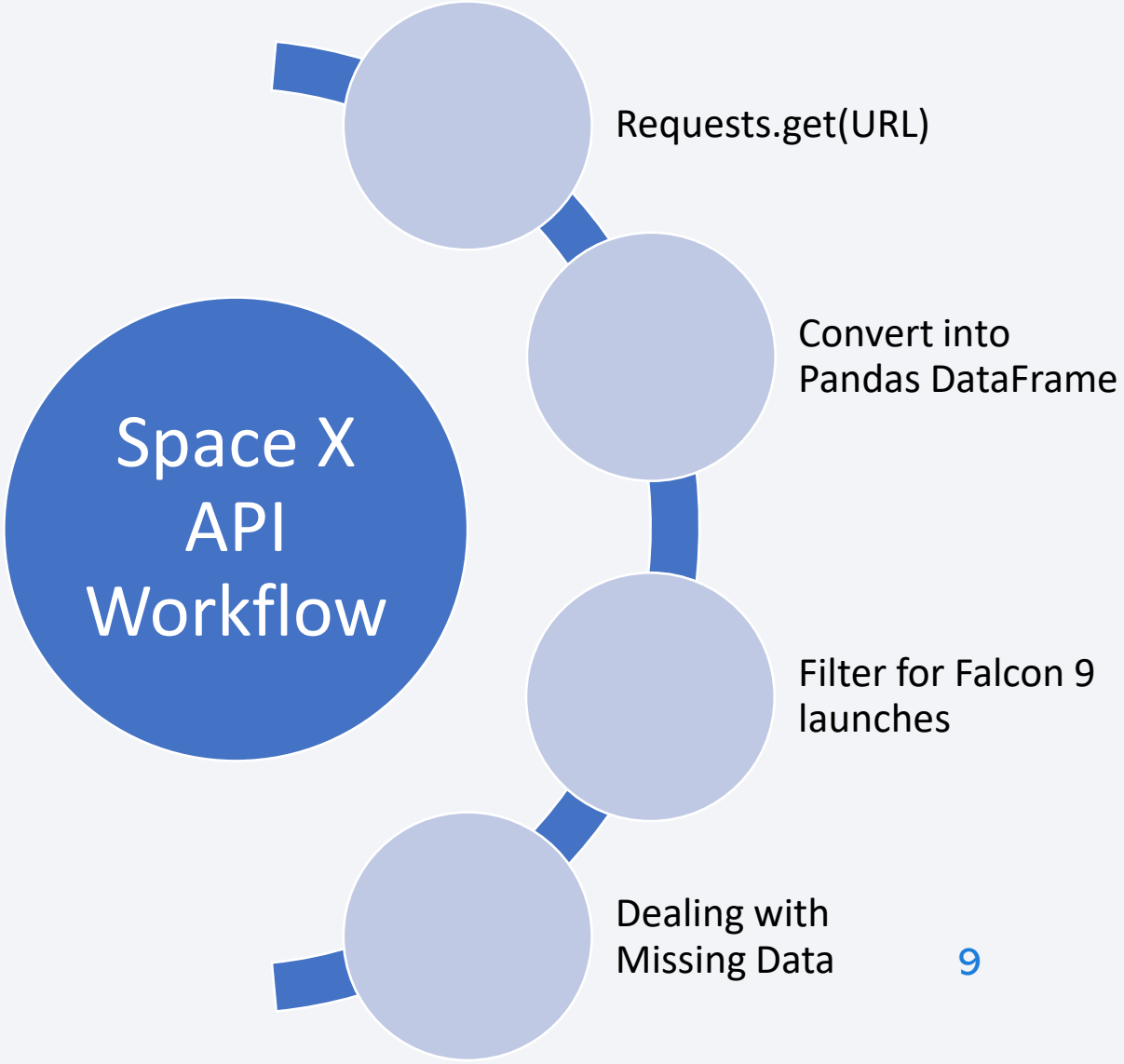
Data Collection

- Describe how data sets were collected.
- Two datasets were collected via:
  - API provided by SpaceX and
  - Websraping from Wikipedia

Data Collection – SpaceX API

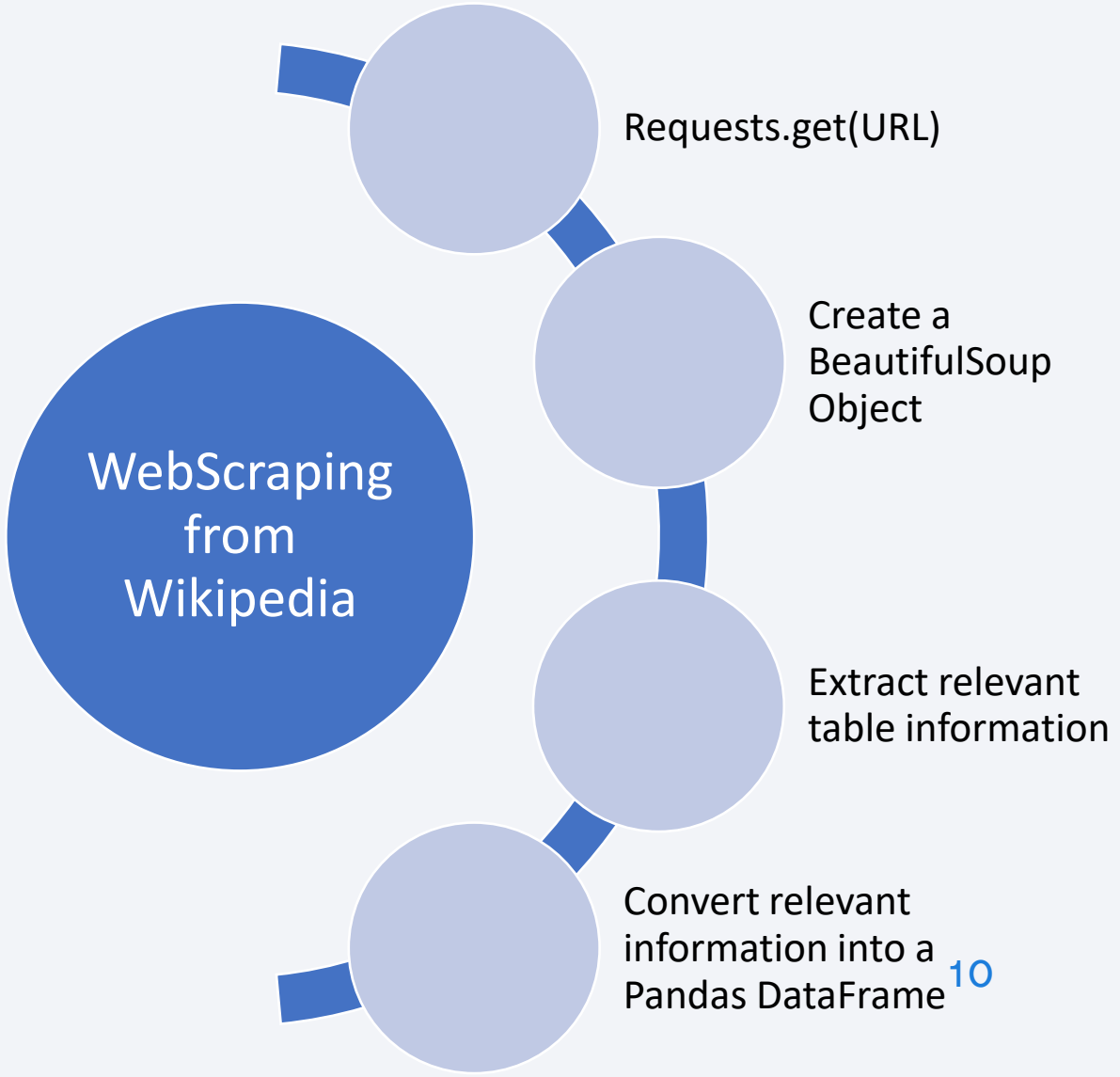
- SpaceX offeres to access past launches via this URL:
  - https://api.spacexdata.com/v4/launches/past

- The Jupyter Notebook can be found here: https://github.com/KDennisL/AppliedDataScienceCapstoneProject/blob/main/J0101-jupyter-labs-spacex-data-collection-api.ipynb



Data Collection Scraping

- Wikipedia contains public available information about Falcon 9 launches from SpaceX
- This data was also parsed using Webscraping processes (see flow chart)
- The Jupyter Notebook can be found here:
  - WebScraping



Data Wrangling

- Previous data was used to:
  - Calculate the number of launches on each site
  - Calculate the number and occurrence of each orbit
  - Calculate the number and occurrence of mission outcome of the orbits
  - Create a landing outcome label from Outcome column
- The Jupyter notebook can be found here:
  - Data Wrangling

EDA with Data Visualization

- The following charts were plotted:
  - relationship between Flight Number and Launch Site as Catplot
  - relationship between Payload and Launch Site as Scatterplot
  - relationship between success rate of each orbit type as Bar Chart
  - relationship between FlightNumber and Orbit type as Scatterplot using Classes as hue
  - relationship between Payload and Orbit type using Scatterplot
  - launch success yearly trend using Lineplot
- The Jupyter Notebook can be found here:
  - EDA Data Visualization

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - %sql SELECT DISTINCT("Launch_Site") FROM SPACEXTBL;
  - %sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE '%CCA%' LIMIT 5
  - %sql PRAGMA table_info(SPACEXTBL);
  - %sql SELECT SUM(PAYLOAD_MASS__KG_)/Count(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1'
  - %sql SELECT MIN(CURRENT_DATE - Date), Date FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)'
  - %sql SELECT Booster_Version, Date,PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000
  - %sql SELECT Mission_Outcome, COUNT(*) FROM SPACEXTBL WHERE Mission_Outcome <> 'None' GROUP BY Mission_Outcome
  - %sql SELECT DISTINCT(Booster_Version) FROM SPACEXTBL where PAYLOAD_MASS__KG_ = (select MAX(PAYLOAD_MASS__KG_) from SPACEXTBL)
  - %sql SELECT DATE, BOOSTER_VERSION, LAUNCH_SITE, Landing_Outcome,substr(Date, 4, 2) as month FROM SPACEXTBL WHERE Landing_Outcome = 'Failure (drone ship)' AND substr(Date,7,4)='2015'
- The Jupyter notebook can be found here:
  - EDA with SQL

Build an Interactive Map with Folium

- Markers, circles, lines and marker clusters added to the sitemap
- Launch sites were marked
- A highlighted circle area with a text label on a specific coordinates
- Marker clusters were used for labeling failed and successful launches
- Lines represent the distance between a launch site to the selected coastline point
- The Jupyter Notebook can be found here:
  - Folium

Build a Dashboard with Plotly Dash

- The Dashboard was created using Plotly Dash
- Interactive Pie Chars and Scatter Plots are shown for
  - Successful launches and
  - The Correlation between Payload and Success
- Interactive graphs easier data analysis than text or tables
- The Jupyter Notebook can be found here:
  - Dash

Predictive Analysis (Classification)

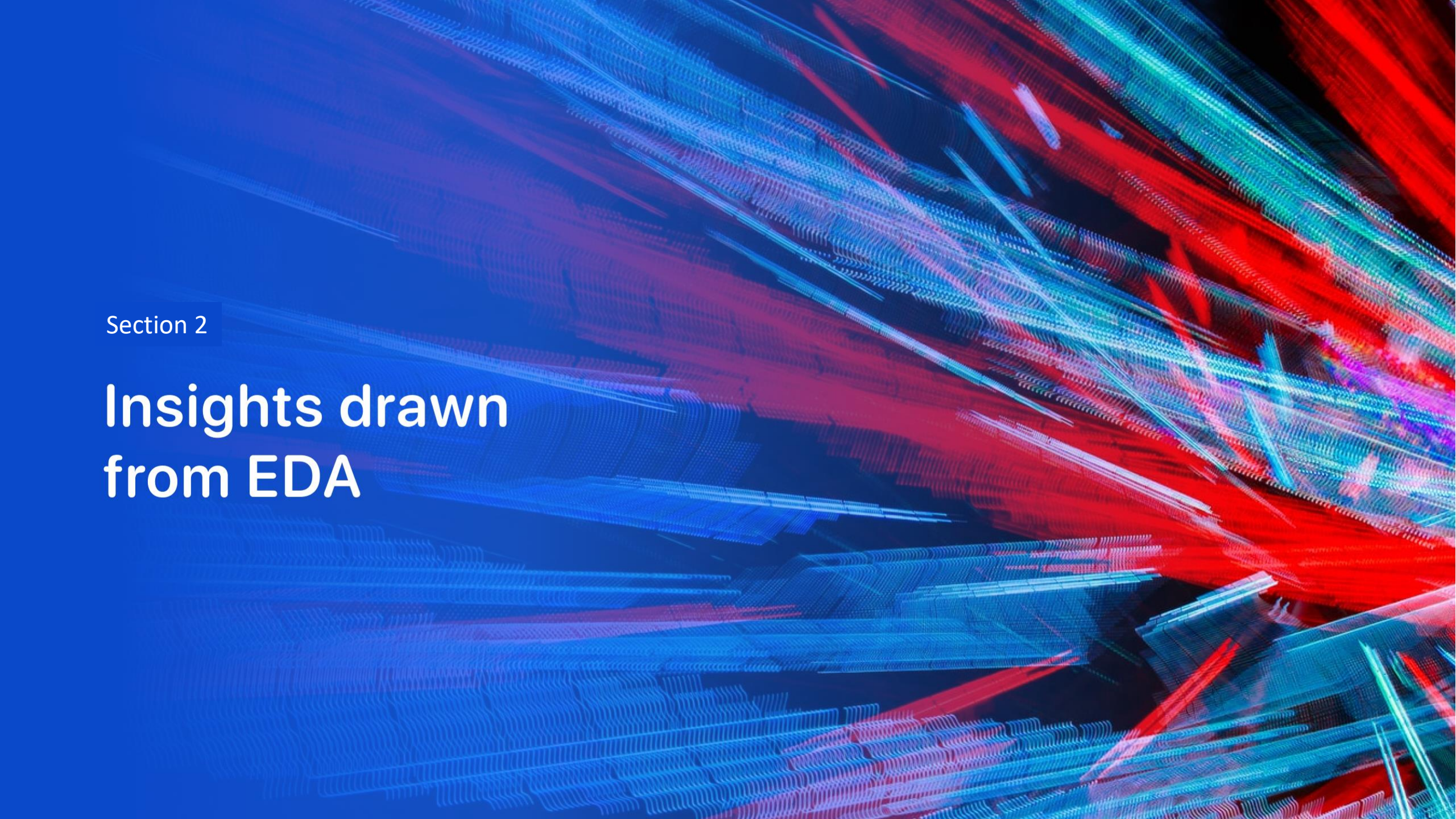
- Data from previous steps was loaded in
- Data was split into Train and Test Datasets
- Four different models were evaluated using GridSearchVG
- All models behaved very similar by accuracy and Confusion plot
- The Jupyter Notebook can be found here:
  - PredictiveAnalysis

Results

- Exploratory data analysis results
  - Distinct Launch Sites: 4
  - The Total Payload was 45596.0 and its mean 2928.4
  - First landing was 2015
  - Four F9 were successful in drone ship and have a mass between 4 and 6k
  - The success of landing increased by year (except latest year)
- Interactive analytics demo in screenshots
- Predictive analysis results

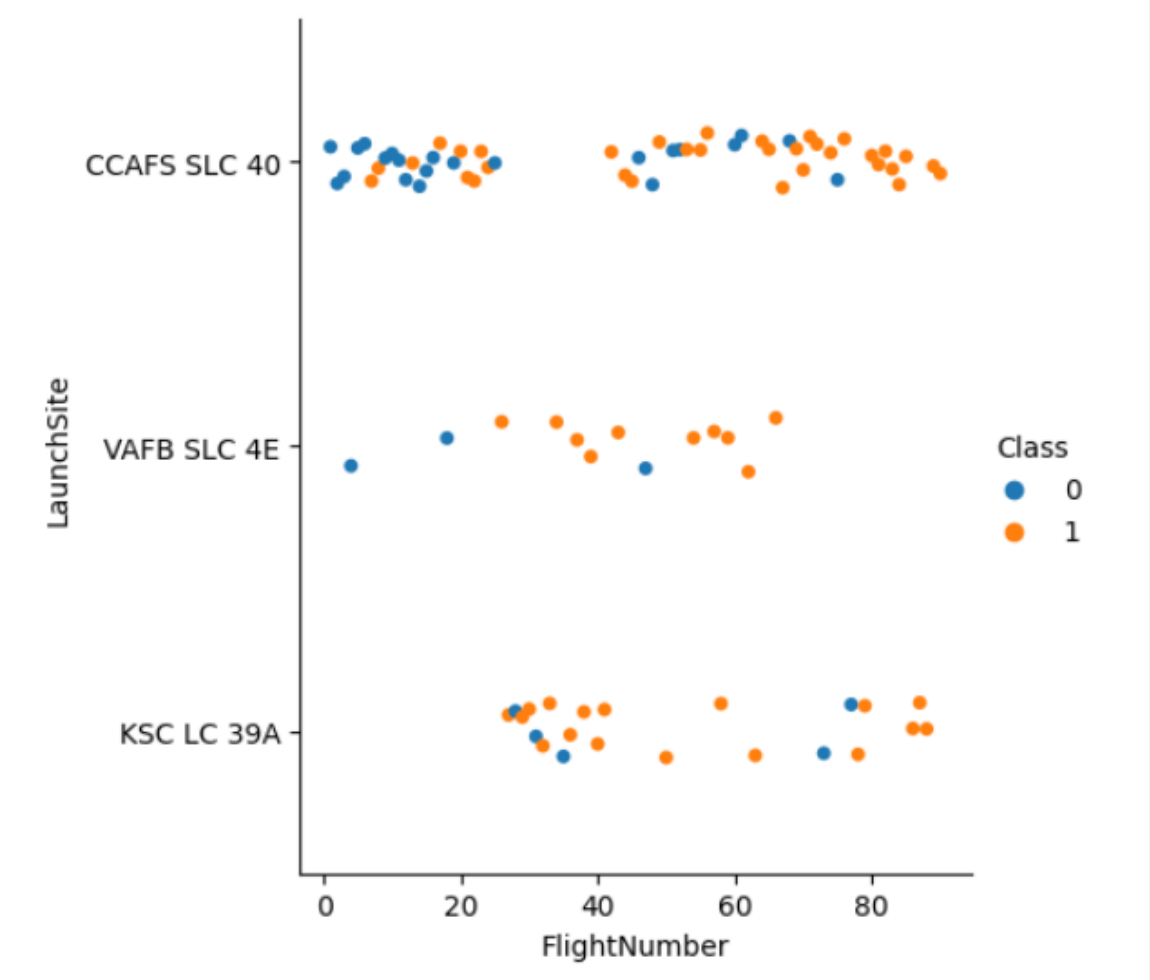
Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
  - Most launches are close to a sea
- Predictive analysis results
  - The accuracy of predicting the result of a landing is ~ 0.8


Section 2
Insights drawn from EDA

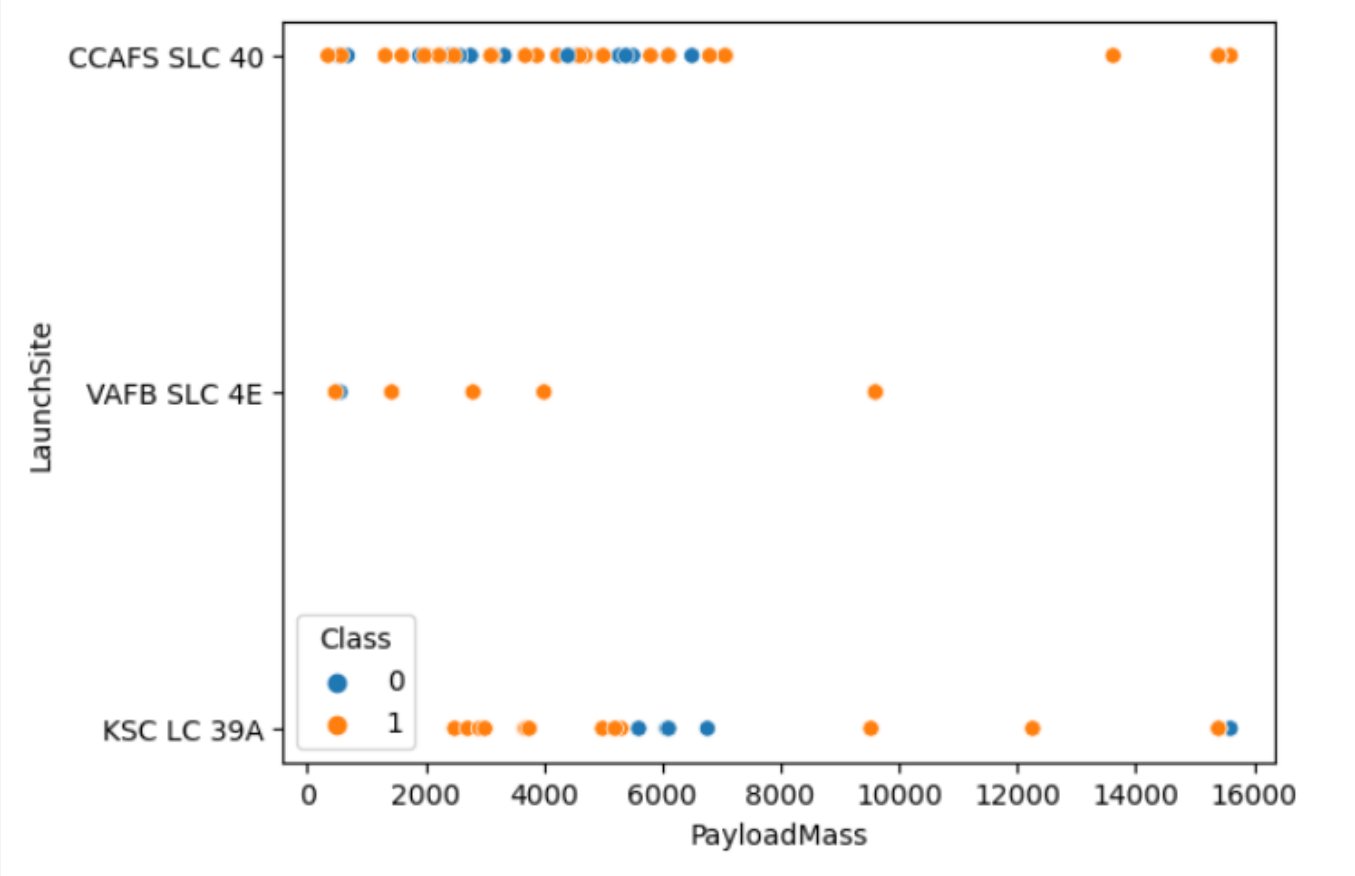
Flight Number vs. Launch Site

- Lowest Flight number is to be found at CCAFS SLC 40
- More Success for Higher Flight numbers



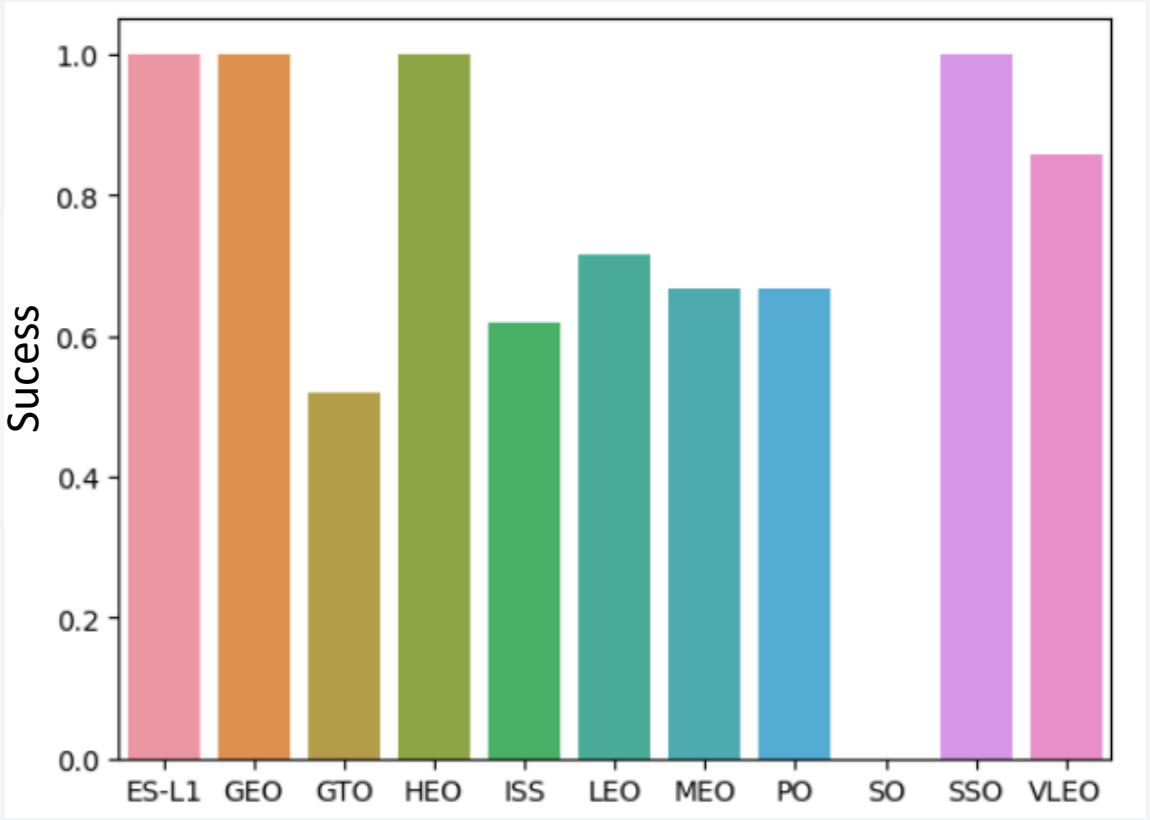
Payload vs. Launch Site

- Lower Payloads seem to have a negative impact on Success



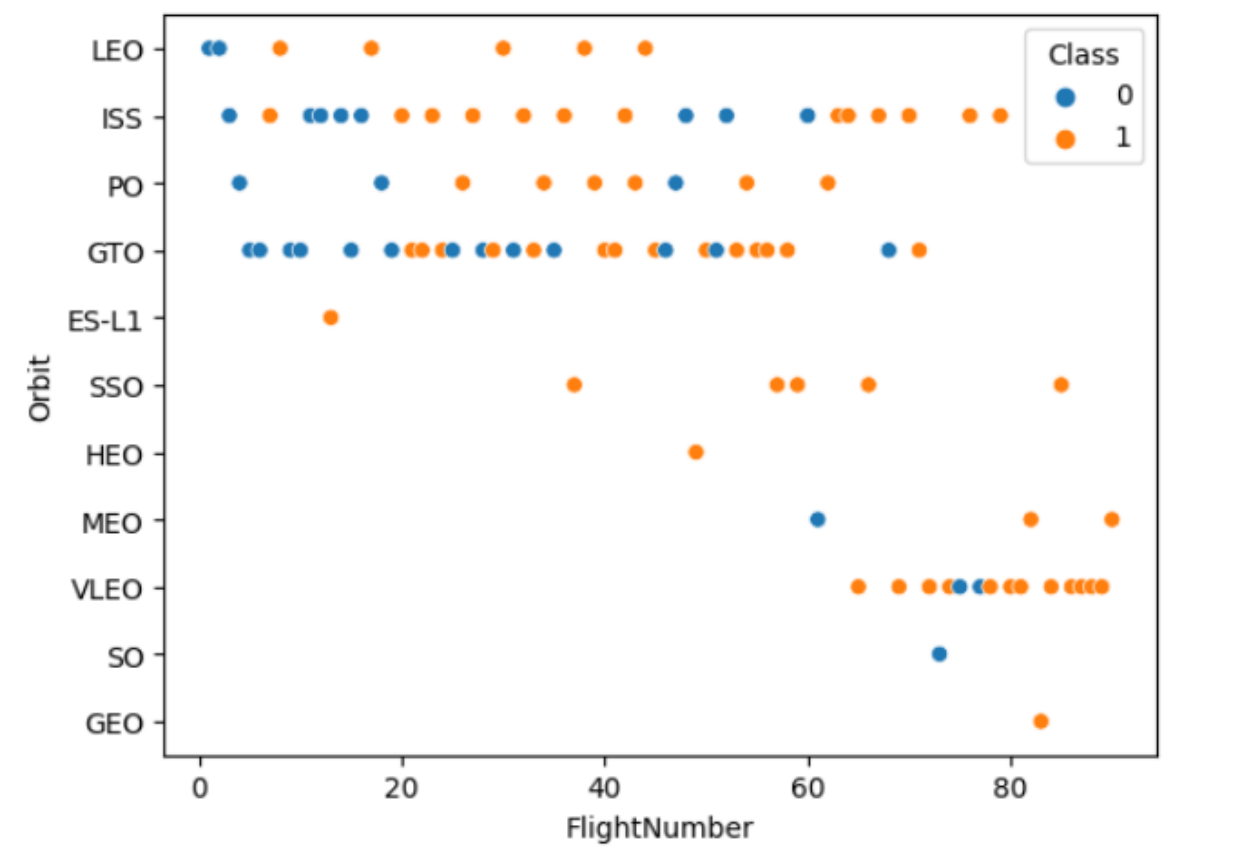
Success Rate vs. Orbit Type

- Highest Success for:
  - EL-L1
  - GEO
  - HEO and SSO



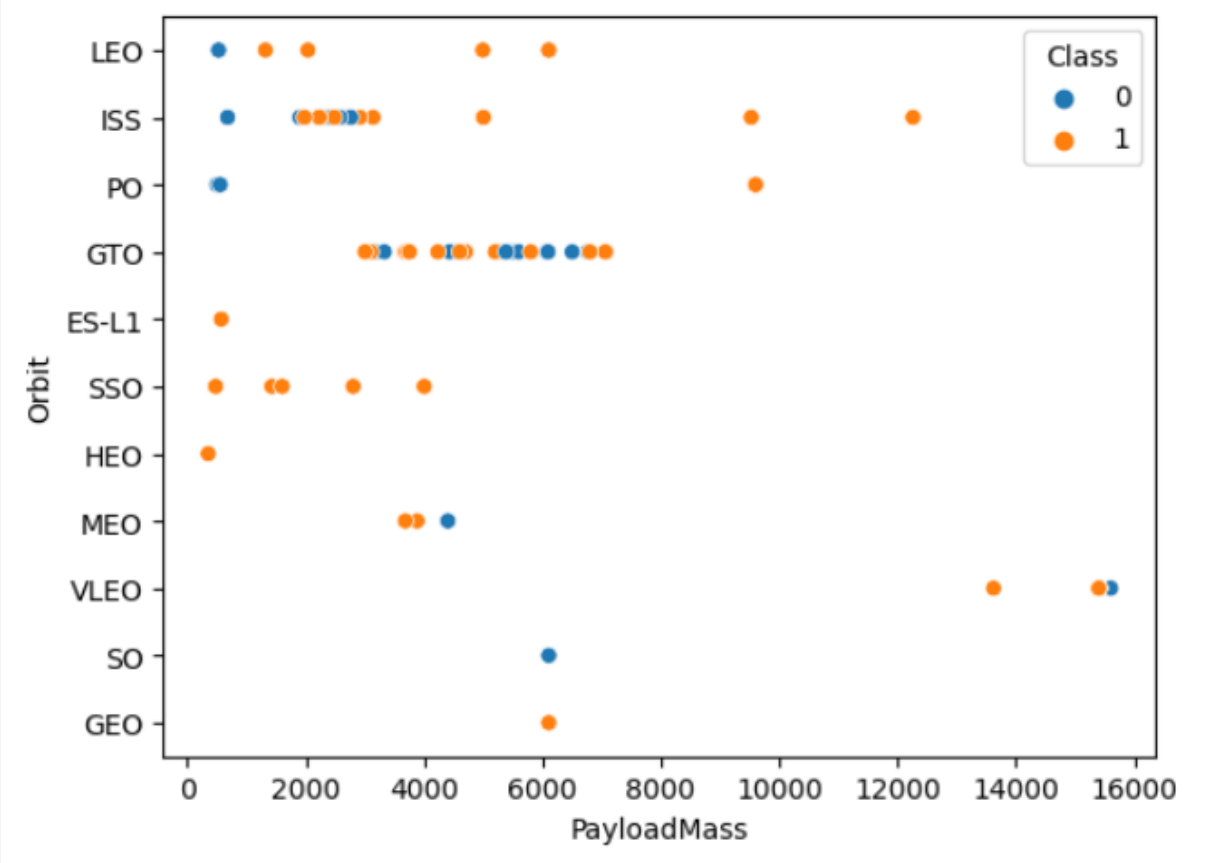
Flight Number vs. Orbit Type

- Highest number of sucess was achieved from VLEO



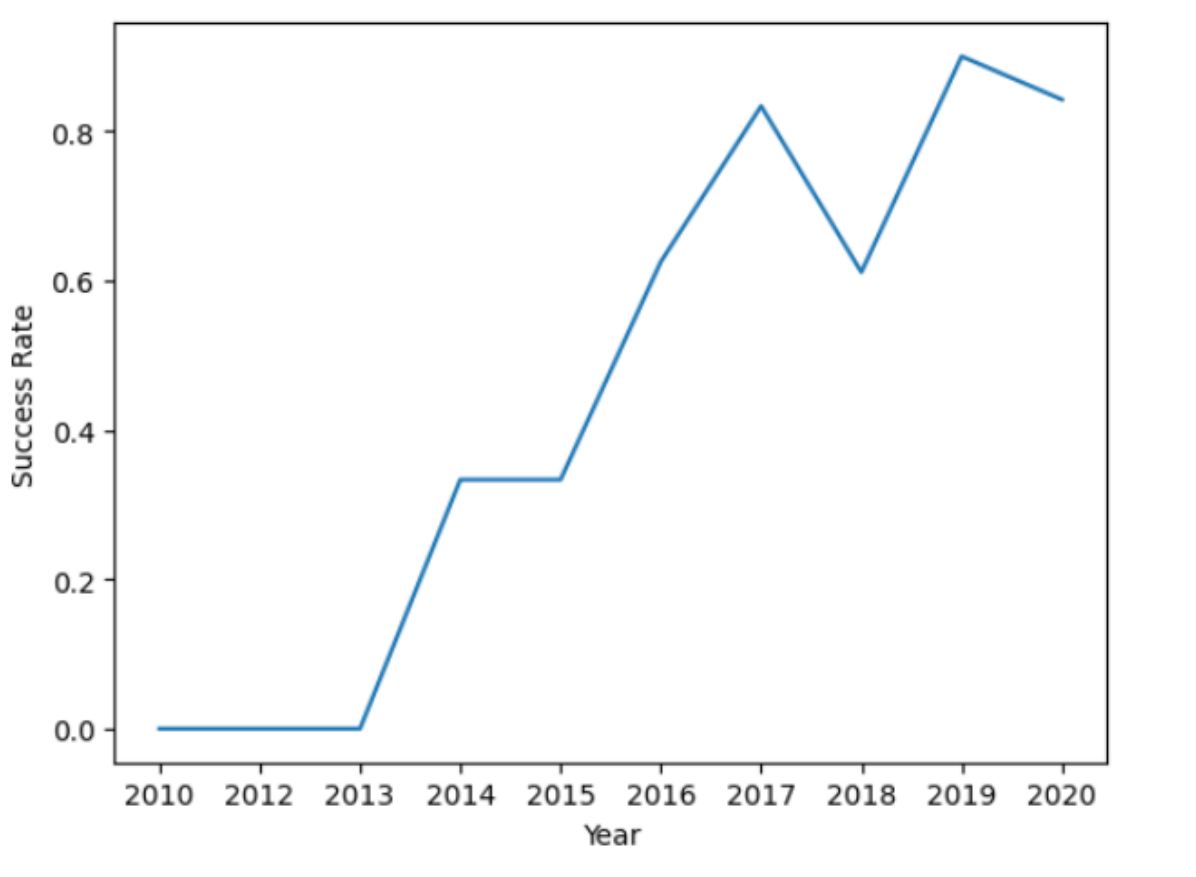
Payload vs. Orbit Type

- Most success in Landing with higher payload for LEO, ISS and Po



Launch Success Yearly Trend

- The success rate increased over the last years
- Potentially, there is a corona dip at 2020



All Launch Site Names

- Using %sql, we could obtain the unique items via DISTINCT from the table SPACEXTBL

Task 1

Display the names of the unique launch sites in the space mission

In [7]:
```
%sql SELECT DISTINCT("Launch_Site") FROM SPACEXTBL;
```

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

Out[7]:

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Task 2

Launch Site Names Begin with 'CCA'

- LIKE operator was used to find items that include CCA
- LIMIT was used to only show 5 items

Task 2

Display 5 records where launch sites begin with the string 'CCA'

In [8]:
```
%sql SELECT * FROM SPACEXTBL WHERE Launch_Site LIKE '%CCA%' LIMIT 5
```

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

Out[8]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outc
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (paracl
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (paracl
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No atte
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No atte
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No atte

Total Payload Mass

- Total Payload was 455969
- Obtained via customer and the sum function

Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

In [9]:
```
%sql PRAGMA table_info(SPACEXTBL);
```

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

Out[9]:

cid	name	type	notnull	dflt_value	pk
0	Date	TEXT	0	None	0
1	Time (UTC)	TEXT	0	None	0
2	Booster_Version	TEXT	0	None	0
3	Launch_Site	TEXT	0	None	0
4	Payload	TEXT	0	None	0
5	PAYLOAD_MASS__KG_	REAL	0	None	0
6	Orbit	TEXT	0	None	0
7	Customer	TEXT	0	None	0
8	Mission_Outcome	TEXT	0	None	0
9	Landing_Outcome	TEXT	0	None	0

In [10]:
```
# NASA contained in column Customer
# payload mass contained in colum PAYLOAD_MASS__KG_ --> with total 3x _
# total == sum of all values in the column

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

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

Out[10]:

SUM(PAYLOAD_MASS__KG_)
45596.0

Average Payload Mass by F9 v1.1

- Average payload was 2928 for F9 v.1.1.

Task 4

Display average payload mass carried by booster version F9 v1.1

In [11]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_)/Count(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Booster_Version = 'F9 v1.1'
```

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

Out[11]:

SUM(PAYLOAD_MASS__KG_)/Count(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- The minimum of current date to the date in the table was obtained
- This was 22/12/2015

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

In [12]:
```
%sql SELECT MIN(CURRENT_DATE - Date), Date FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)'

#SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome = 'Success (ground pad)'
```

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

Out[12]:

MIN(CURRENT_DATE - Date)	Date
2001	22/12/2015

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [13]:
```
%sql SELECT Booster_Version, Date,PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOA
# shwoing date and pyload mass to ensure that the correct results are printed out
```

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

Out[13]:

Booster_Version	Date	PAYLOAD_MASS__KG_
F9 FT B1022	05/06/2016	4696.0
F9 FT B1026	14/08/2016	4600.0
F9 FT B1021.2	30/03/2017	5300.0
F9 FT B1031.2	10/11/2017	5200.0

- WHERE and Payload mass range was used to get the results

Total Number of Successful and Failure Mission Outcomes

- WHERE clause was used with unequal to non and
- Grouped by Mission outcome

Task 7

List the total number of successful and failure mission outcomes

In [14]:
```
%sql SELECT Mission_Outcome, COUNT(*) FROM SPACEXTBL WHERE Mission_Outcome <> 'None' GROUP BY Mission_Outcome
```

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

Out[14]:

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

Boosters Carried Maximum Payload

- Subquerey was used to get the maximum of Payload mass
- Maximum value was used as a filter to get the highest values

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [15]:
```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_, Date FROM SPACEXTBL where PAYLOAD_MASS__KG_ = (select MAX(PAYLOAD_MASS__KG_)
# printing payload and date to check for correctness
# as it is not specified to report unique items; this one is used without distinct
```

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

Out[15]:

Booster_Version	PAYLOAD_MASS__KG_	Date
F9 B5 B1048.4	15600.0	11/11/2019
F9 B5 B1049.4	15600.0	01/07/2020
F9 B5 B1051.3	15600.0	29/01/2020
F9 B5 B1056.4	15600.0	17/02/2020
F9 B5 B1048.5	15600.0	18/03/2020
F9 B5 B1051.4	15600.0	22/04/2020
F9 B5 B1049.5	15600.0	06/04/2020
F9 B5 B1060.2	15600.0	09/03/2020
F9 B5 B1058.3	15600.0	10/06/2020
F9 B5 B1051.6	15600.0	18/10/2020
F9 B5 B1060.3	15600.0	24/10/2020
F9 B5 B1049.7	15600.0	25/11/2020

2015 Launch Records

Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [17]:
```
%sql SELECT DATE, BOOSTER_VERSION, LAUNCH_SITE, Landing_Outcome,substr(Date, 4, 2) as month FROM SPACEXTBL WHERE Landing_Out
```

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

Out[17]:

Date	Booster_Version	Launch_Site	Landing_Outcome	month
01/10/2015	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)	10
14/04/2015	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)	04

Task 10

WHERE Landing_Outcome = 'Failure (drone ship)' AND substr(Date,7,4)='2015' was used to filter the data

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

- Using LIKE operator to get a list of successfully landed items
- Date Between to only contain specific values

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

In [18]:
```
%sql SELECT Landing_Outcome, COUNT(Landing_Outcome)  FROM SPACEXTBL WHERE Landing_Outcome LIKE '%Success%' and Date between
```

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

Out[18]:

Landing_Outcome	COUNT(Landing_Outcome)
Success	20
Success (drone ship)	8
Success (ground pad)	7

Section 3

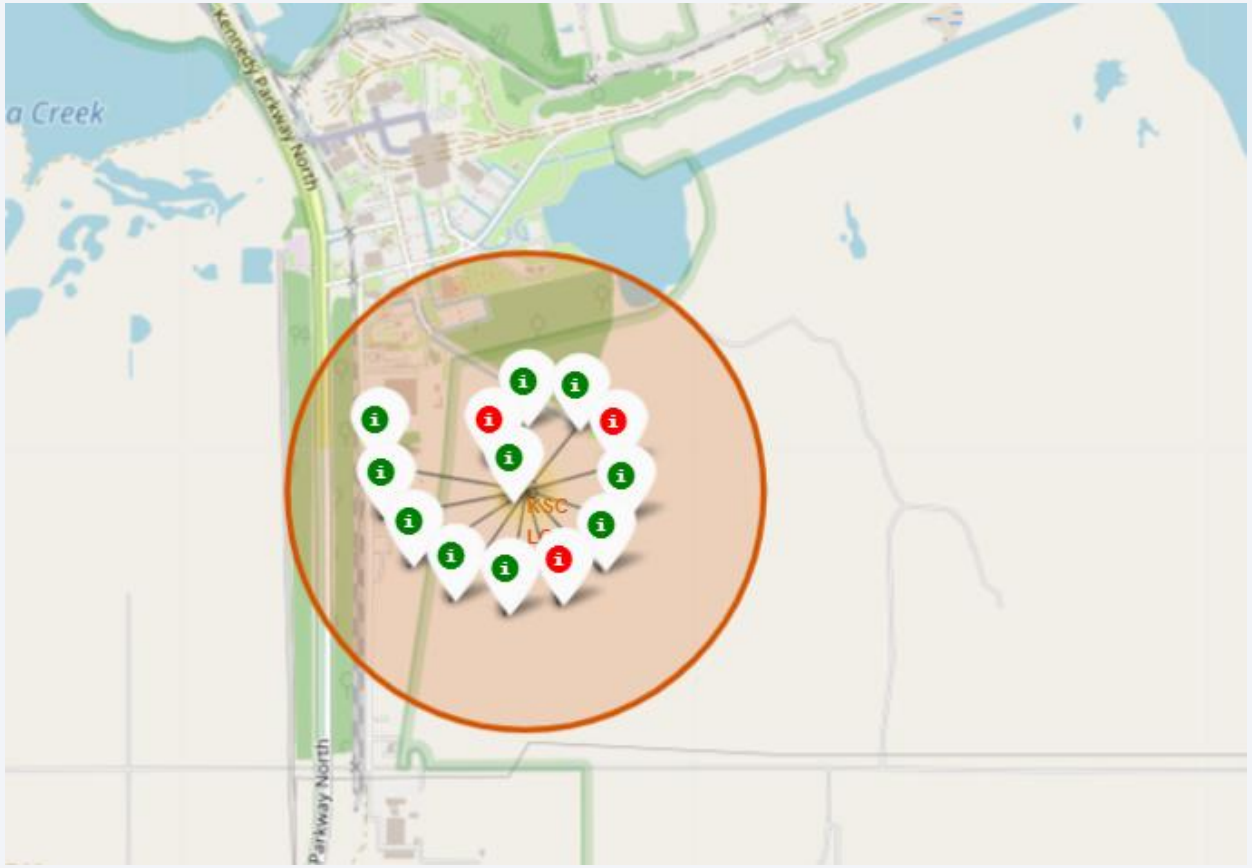
Launch Sites Proximities Analysis

Launch Sites

- Launch Sites were only found in the US
- Either on the east of west cost

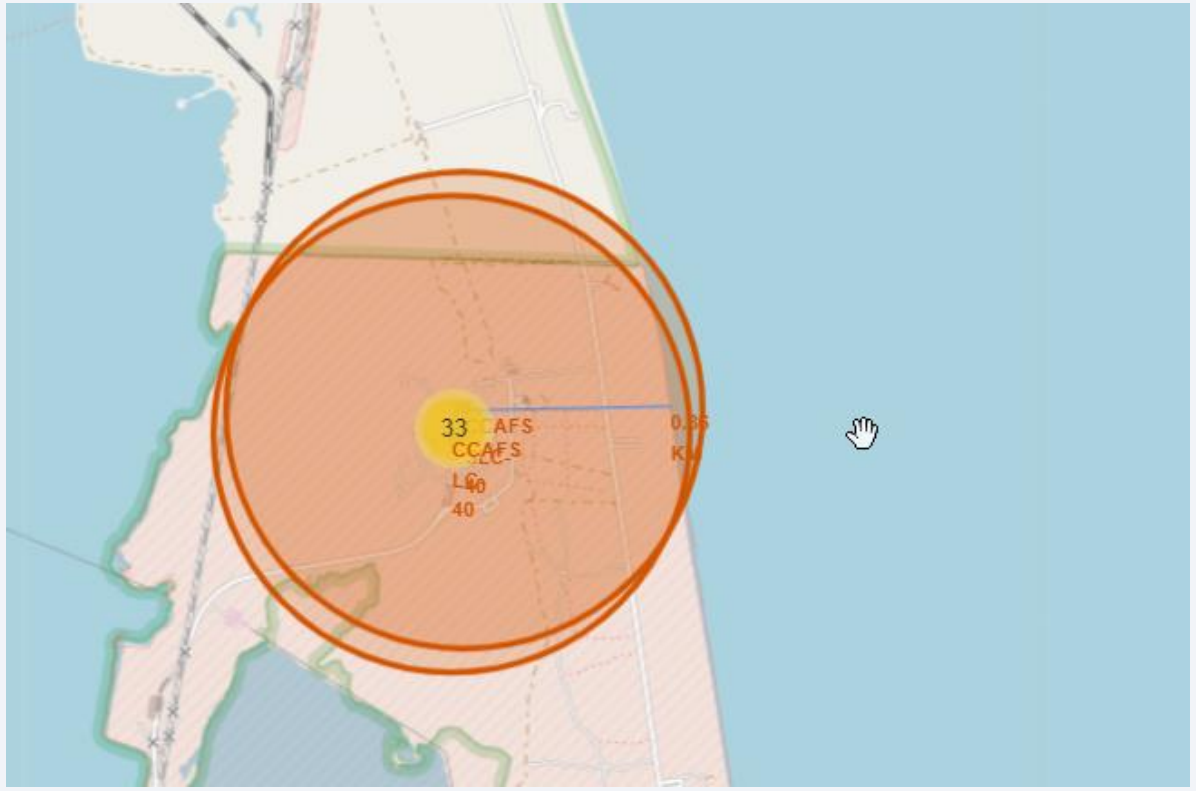
Landing Success

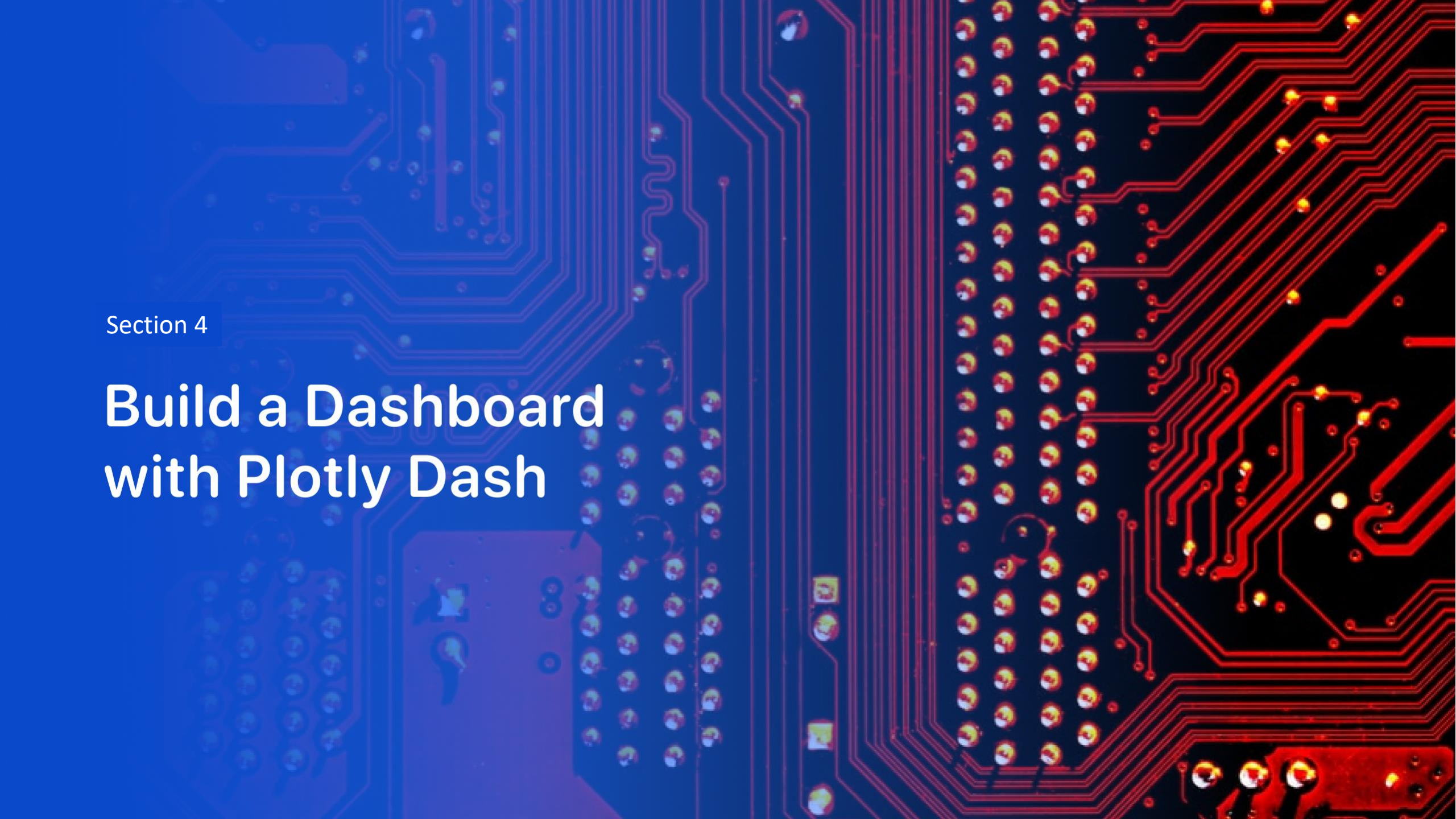
- RSC LC-39A has mostly successful landing (green marks)



Distance to Sea

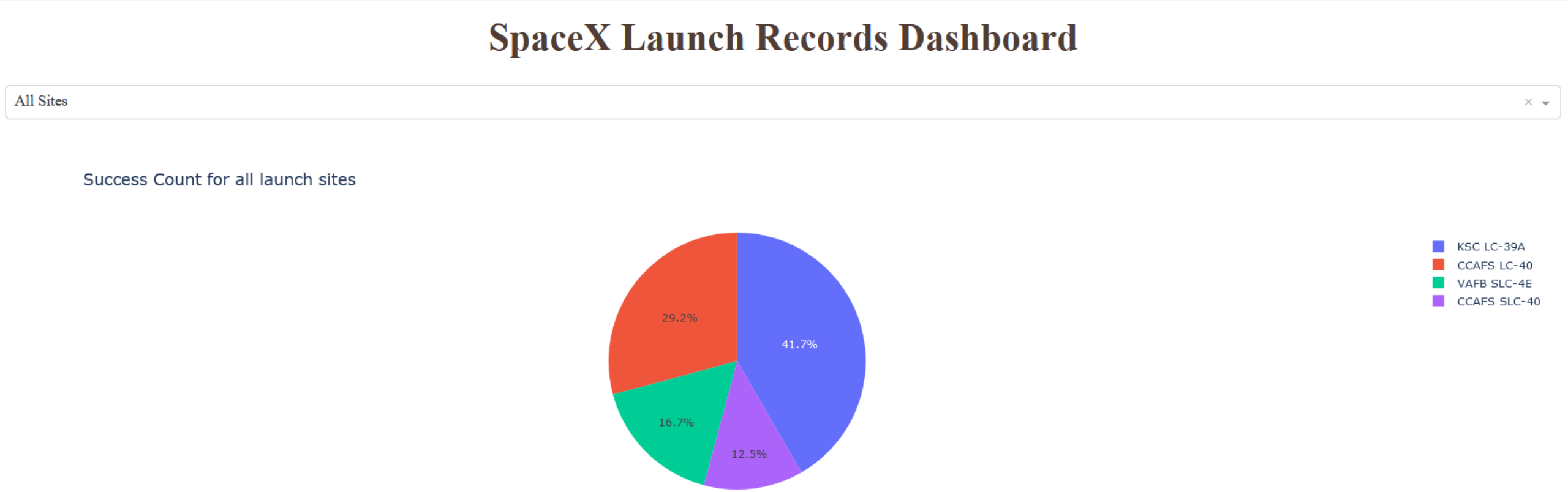
- To eliminate any injury to humans, it is best to have a close location to the sea




Section 4
Build a Dashboard with Plotly Dash

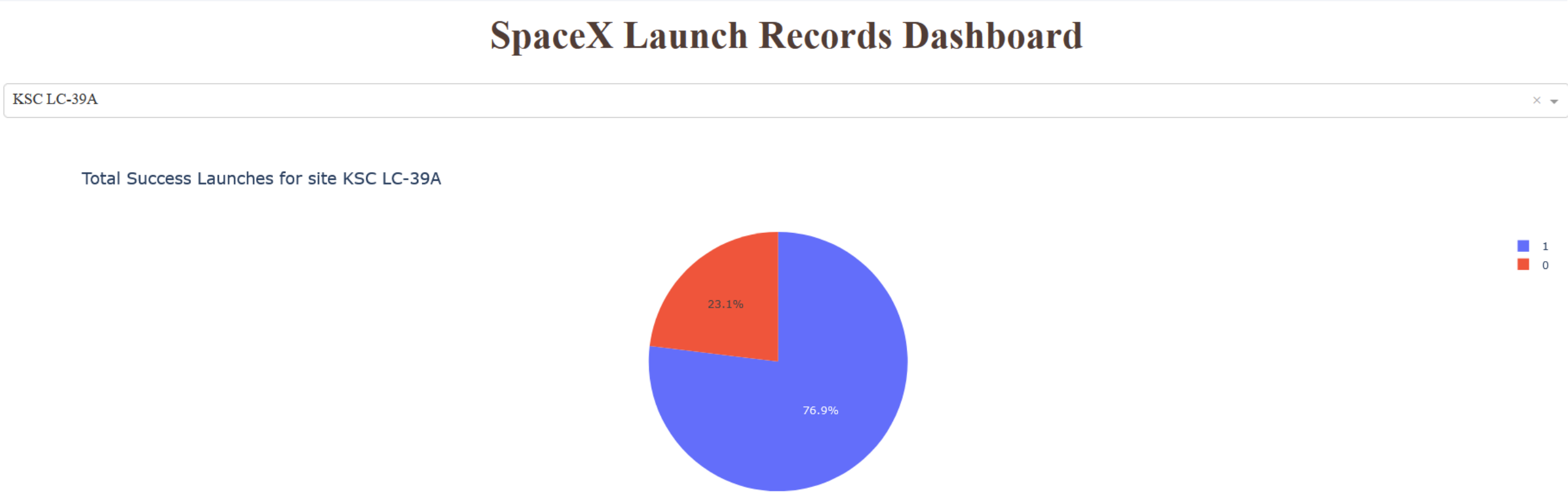
Successful launches at each site

- KSC LC-39A had the most successful launches



Successful launches for KSC LC-39A

- In 77%, the launches were successful



Payloads as a Filter for Success

- The Success is low for extrem high and low payloards

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- I am not going to create another plot here….
- Best model was SM with a score of 0.85

TASK 6

Create a support vector machine object then create a `GridSearchCV` object `svm_cv` with cv - 10. Fit the object to find the best parameters from the dictionary `parameters`.

In [75]:
```
parameters = {'kernel':('linear', 'rbf','poly','rbf', 'sigmoid'),
              'C': np.logspace(-3, 3, 5),
              'gamma':np.logspace(-3, 3, 5)}
svm = SVC()
```

In [76]:
```
svm_cv = GridSearchCV(svm,parameters,cv=10)
# as -10 failed, I continued with cd =10
svm_cv.fit(X_train, Y_train)
```

Out[76]:
```
GridSearchCV(cv=10, estimator=SVC(),
             param_grid={'C': array([1.00000000e-03, 3.16227766e-02, 1.00000000e+00, 3.16227766e+01,
       1.00000000e+03]),
                         'gamma': array([1.00000000e-03, 3.16227766e-02, 1.00000000e+00, 3.16227766e+01,
       1.00000000e+03]),
                         'kernel': ('linear', 'rbf', 'poly', 'rbf', 'sigmoid')})
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [77]:
```
print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
print("accuracy :",svm_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

Confusion Matrix

- There were not type II errors
- The number of type I error was similar to the number of correctly predicted not landed

TASK 7

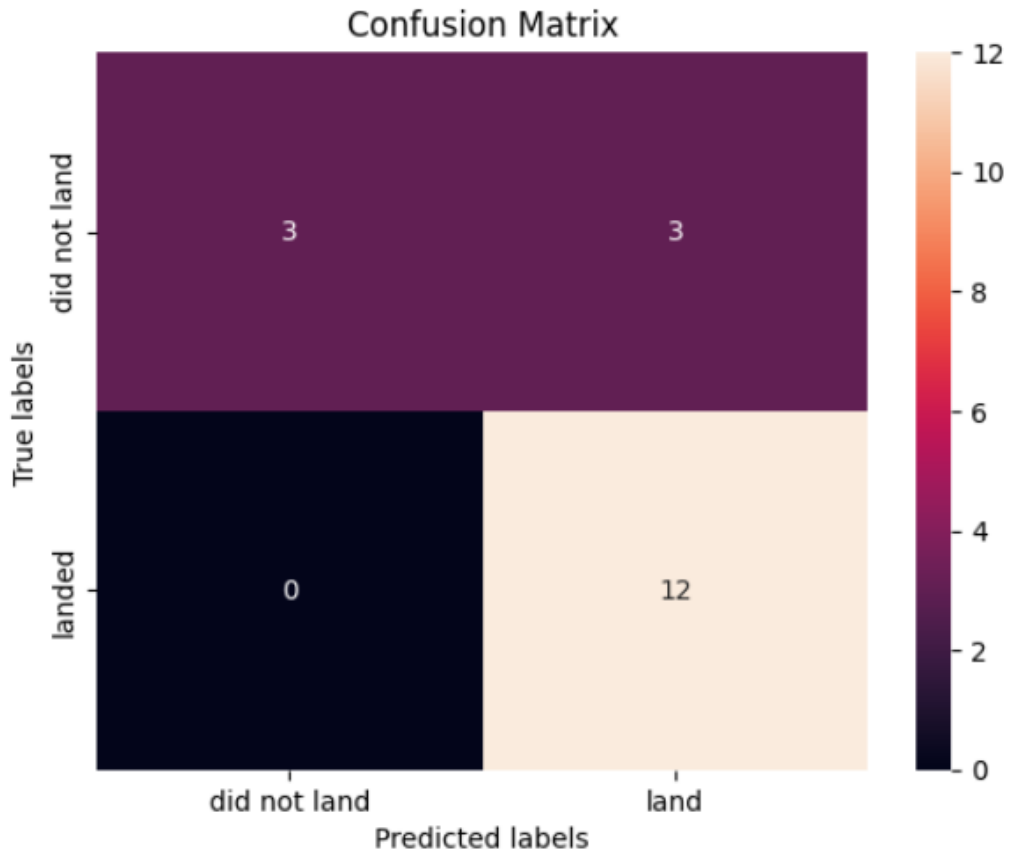
Calculate the accuracy on the test data using the method `score` :

In [78]:
```
print("Accurace of test data set is :",svm_cv.score(X_test, Y_test))
```

```
Accurace of test data set is : 0.8333333333333334
```

We can plot the confusion matrix

In [79]:
```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- Launch success increased from year to year starting with year 2015
- Starting points are close to the sea
- KSC LC-39A had the most successful launches
- Extreme values for Payload had a negative impact
- SVM performed best to predict the outcome

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!