Space Shuttle Landing Prediction

Hafiz Muhammad Kashif

28-Oct-2024

IBM Developer

SKILLS NETWORK

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
  - Visualization – Charts
  - Dashboard
- Discussion
  - Findings & Implications
- Conclusion
- Appendix

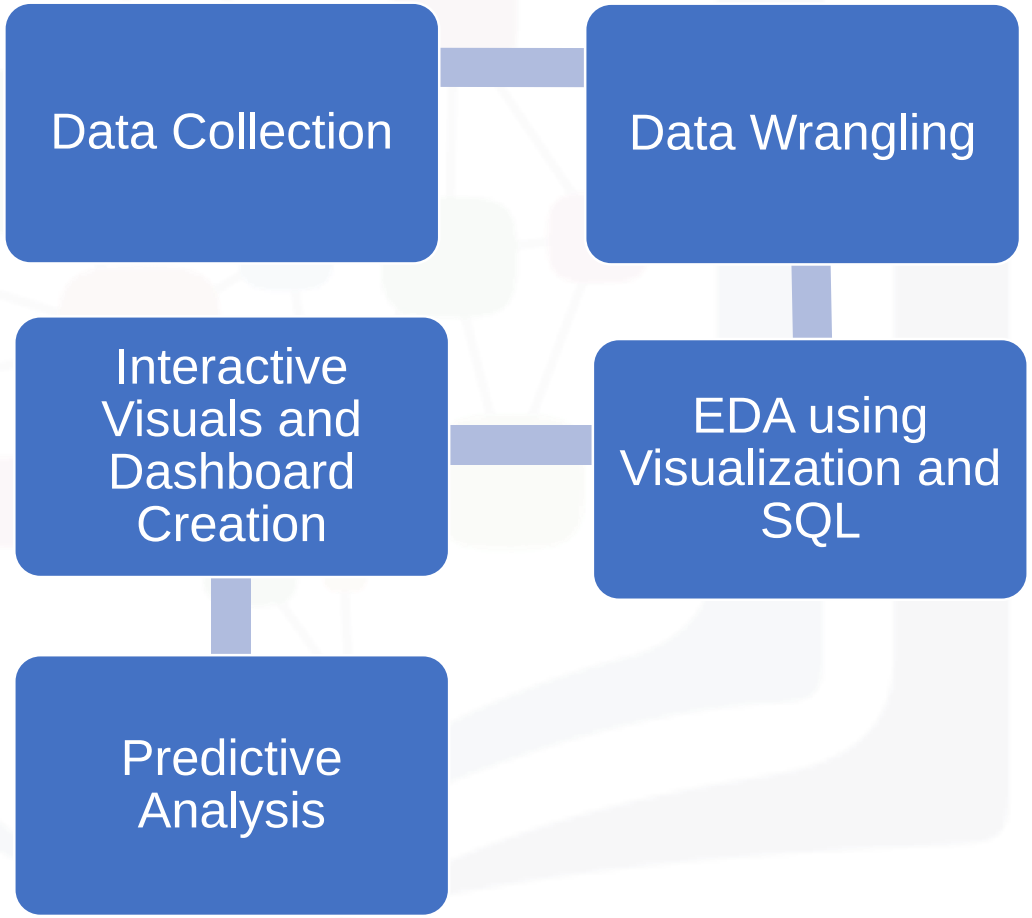
EXECUTIVE SUMMARY

- Data Collection
- Data Wrangling
- EDA with Visualization
- EDA with SQL
- Building an interactive map with folium
- Building a Dashboard with Plotly Dash
- Predictive Analysis
- EDA Results
- Interactive Analytics

INTRODUCTION

- I apply my data science skills as a Data scientist for a private space launch company in this project.
- As a starting point of almost all data science projects, I collected data, as much and relevant as possible.
- After collecting data from various sources. After your raw data has been collected, I improved the quality by performing data wrangling.
- Then I started exploring the processed data.
- I also gain further insights into the data by applying some basic statistical analysis and data visualization,
- Then I drill down into finer levels of detail by splitting the data into groups defined by categorical variables or factors in the data.
- After that, I build, evaluate, and refine predictive models for discovering more exciting insights.
- The final task of this capstone project was to create a presentation that will be developed into stories of all your analysis.

METHODOLOGY



Data Collection – SpaceX API

Task 1: Request and parse the SpaceX launch data using the GET request

To make the requested JSON results more consistent, we will use the following static response object for this project:

```
In [29]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/
```

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

```
In [30]: response.status_code
Out[30]: 200
```

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

```
In [31]: # Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
In [32]: # Get the head of the dataframe
data.head()
Out[32]:   static_fire_date_utc  static_fire_date_unix
```

Requested and parsed the data from SpaceX API using GET request

Finally lets construct our dataset using the data we have obtained. We we combine the columns into a dictionary.

```
In [41]: launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
In [42]: # Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

Show the summary of the dataframe

```
In [43]: # Show the head of the dataframe
df.head()
```

Stored the data and constructed the dataset into a new dictionary with relative columns

Task 2: Filter the dataframe to only include `Falcon 9` launches

Finally we will remove the Falcon 1 launches keeping only the Falcon 9 launches. Filter the data dataframe using the `BoosterVersion` column to only keep the Falcon 9 launches. Save the filtered data to a new dataframe called `data_falcon9`.

```
In [44]: # Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = df[df['BoosterVersion']!='Falcon 1']
```

Now that we have removed some values we should reset the FlgihtNumber column

```
In [45]: data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

Filtered the dataframe to only include Falcon 9 launches needed for the project analysis

Task 3: Dealing with Missing Values

Calculate below the mean for the `PayloadMass` using the `.mean()`. Then use the mean and the `.replace()` function to replace `np.nan` values in the data with the mean you calculated.

```
In [62]: # # Calculate the mean value of PayloadMass column
# # Replace the np.nan values with its mean value
payloadmass_avg = data_falcon9['PayloadMass'].mean()
data_falcon9['PayloadMass'].replace(np.nan, payloadmass_avg, inplace=True)
```

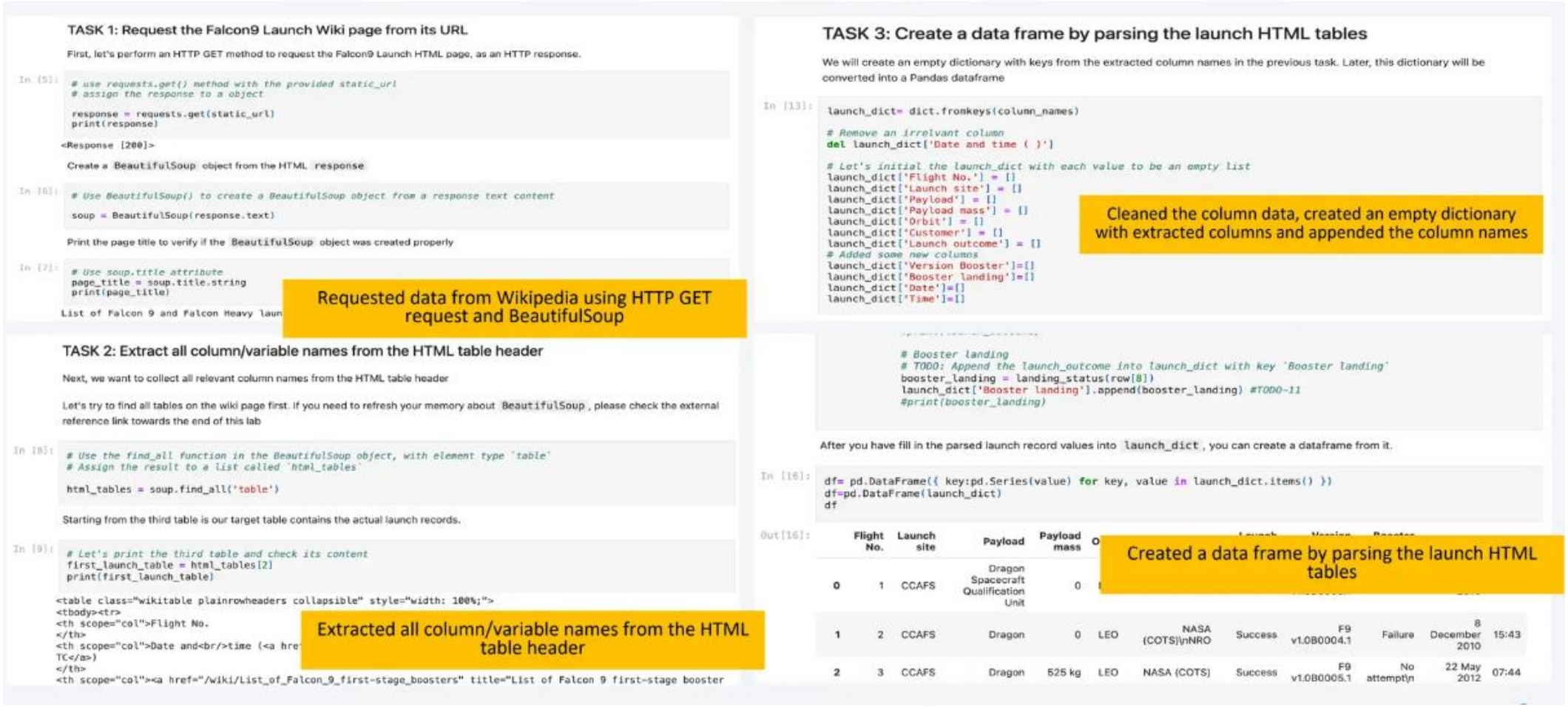
```
In [63]: data_falcon9.isnull().sum()
Out[63]: FlightNumber      0
Date              0
BoosterVersion    0
PayloadMass       0
Orbit             0
LaunchSite        0
Outcome           0
Flights           0
GridFins          0
Reused            0
Legs              0
LandingPad       26
Block             0
ReusedCount       0
Serial            0
Longitude         0
Latitude          0
dtype: int64
```

Cleaned the data and removed missing values o PayLoadMass

You should see the number of missing values of the `PayLoadMass` change to zero.

Now we should have no missing values in our dataset except for in `LandingPad`.

Data Collection – Web Scrapping



Data Wrangling

TASK 1: Calculate the number of launches on each site

The data contains several Space X launch facilities: Cape Canaveral Space Launch Complex 40 **VAFB SLC 4E** , Vandenberg Air Force Base Space Launch Complex 4E **(SLC-4E)**, Kennedy Space Center Launch Complex 39A **KSC LC 39A** .The location of each Launch is placed in the column `LaunchSite`

Next, let's see the number of launches for each site.

Use the method `value_counts()` on the column `LaunchSite` to determine the number of launches on each site:

```
In [5]:
# Apply value_counts() on column LaunchSite
launch_site_count = df['LaunchSite'].value_counts()
print(launch_site_count)
```

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

Calculated the number of launches of each site

TASK 2: Calculate the number and occurrence of each orbit

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

```
In [6]:
orbit_count = df['Orbit'].value_counts()
print(orbit_count)
```

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

TASK 3: Calculate the number and occurence of mission outcome of the orbits

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then assign it to a variable landing_outcomes.

```
In [7]:
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
print(landing_outcomes)
```

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

Calculated the number and occurrence of each orbit and its mission outcome

TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome` , create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome` ; otherwise, it's one. Then assign it to the variable `landing_class` :

```
In [10]:
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for outcome in df['Outcome']:
    if outcome in 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, the first stage did not land successfully; one means the first stage landed Successfully

```
In [11]:
df['Class']=landing_class
df[['Class']].head(8)
```

Out[11]:

	Class
0	0

Created a landing outcome label form Outcome column

```
In [12]:
df.head(5)
```

Out[12]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN

We can use the following line of code to determine the success rate:

```
In [13]:
df["Class"].mean()
```

```
Out[13]: 0.6666666666666666
```

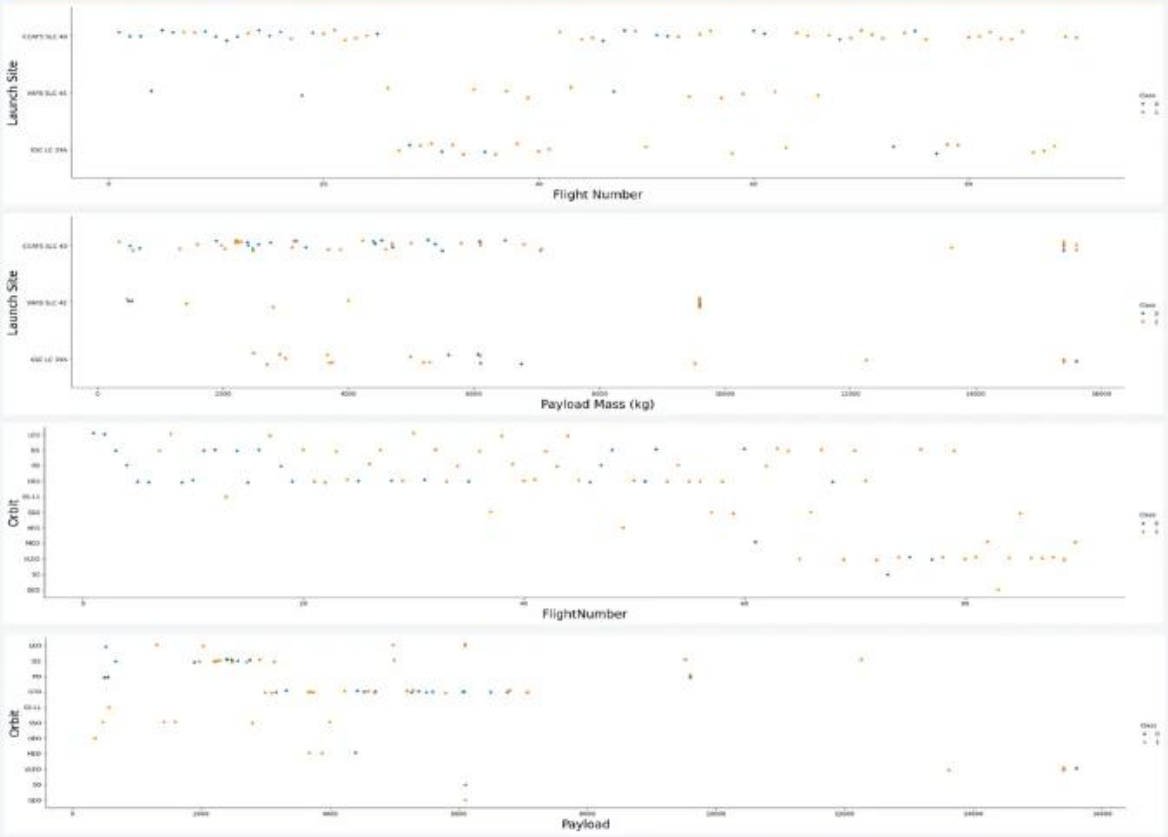
Calculated the average success rate

We can now export it to a CSV for the next section,but to make the answers consistent, in the next lab we will provide data in a pre-selected date range.

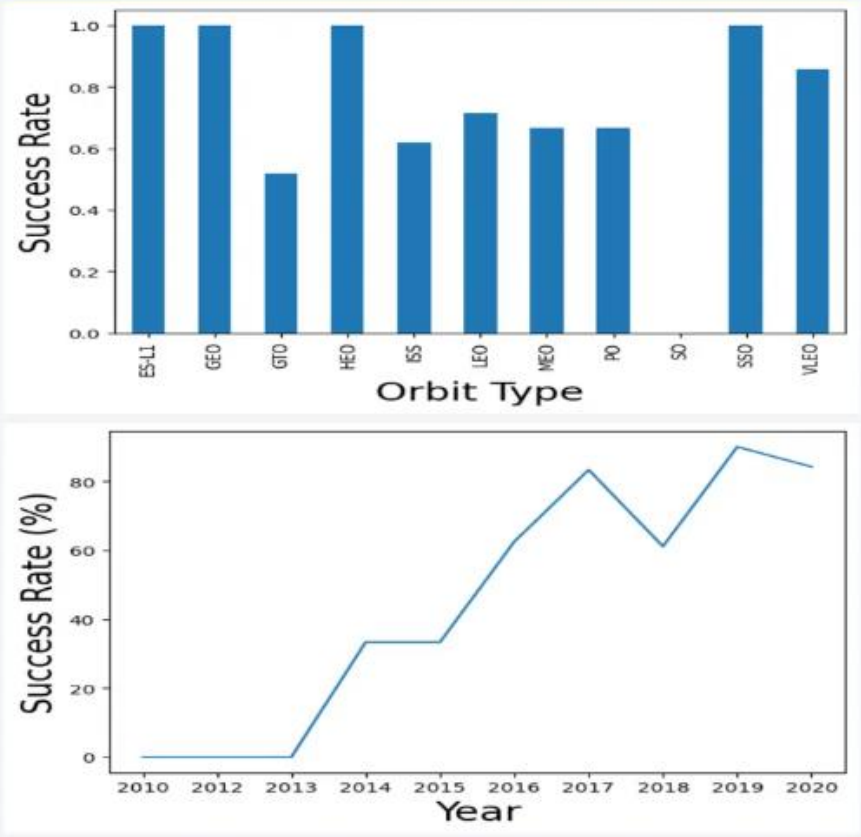
```
df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Visualization

Scatter point charts are used to visualize the relationship between (1) Flight Number and Launch Site, (2) Payload and Launch Site, (3) Flight Number and Orbit type, (4) Payload and Orbit type



Bar chart was used to visualize the relationship between success rate of each orbit type; Line chart was used to visualize the launch success yearly trend



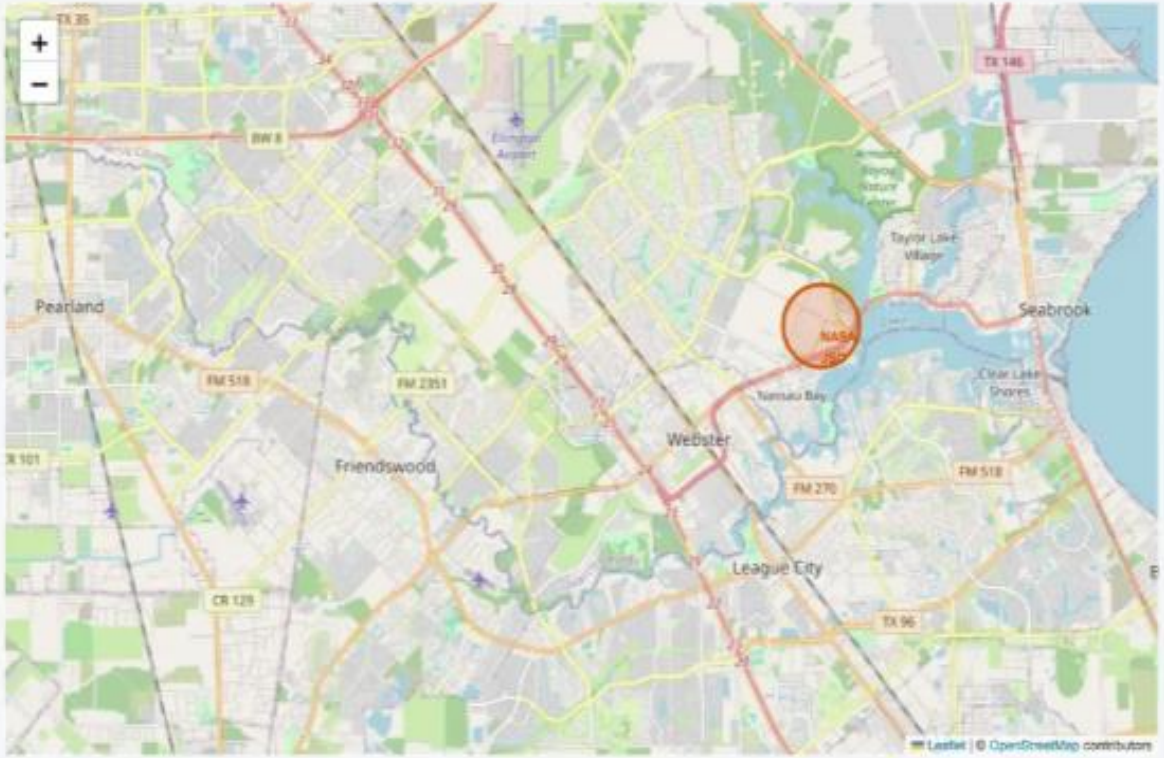
EDA with SQL

SQL queries are performed to:

1. Display the names of the unique launch sites in the space mission
2. Display 5 records where launch sites begin with the string 'CCA'
3. Display the total payload mass carried by boosters launched by NASA (CRS)
4. Display average payload mass carried by booster version F9 v1.1
5. List the date when the first successful landing outcome in ground pad was achieved.
6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7. List the total number of successful and failure mission outcomes
8. List the names of the booster versions which have carried the maximum payload mass using a subquery
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.
10. 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.

Build an Interactive Map with Folium

A circle marker was created to show NASA Johnson Space Center's coordinate

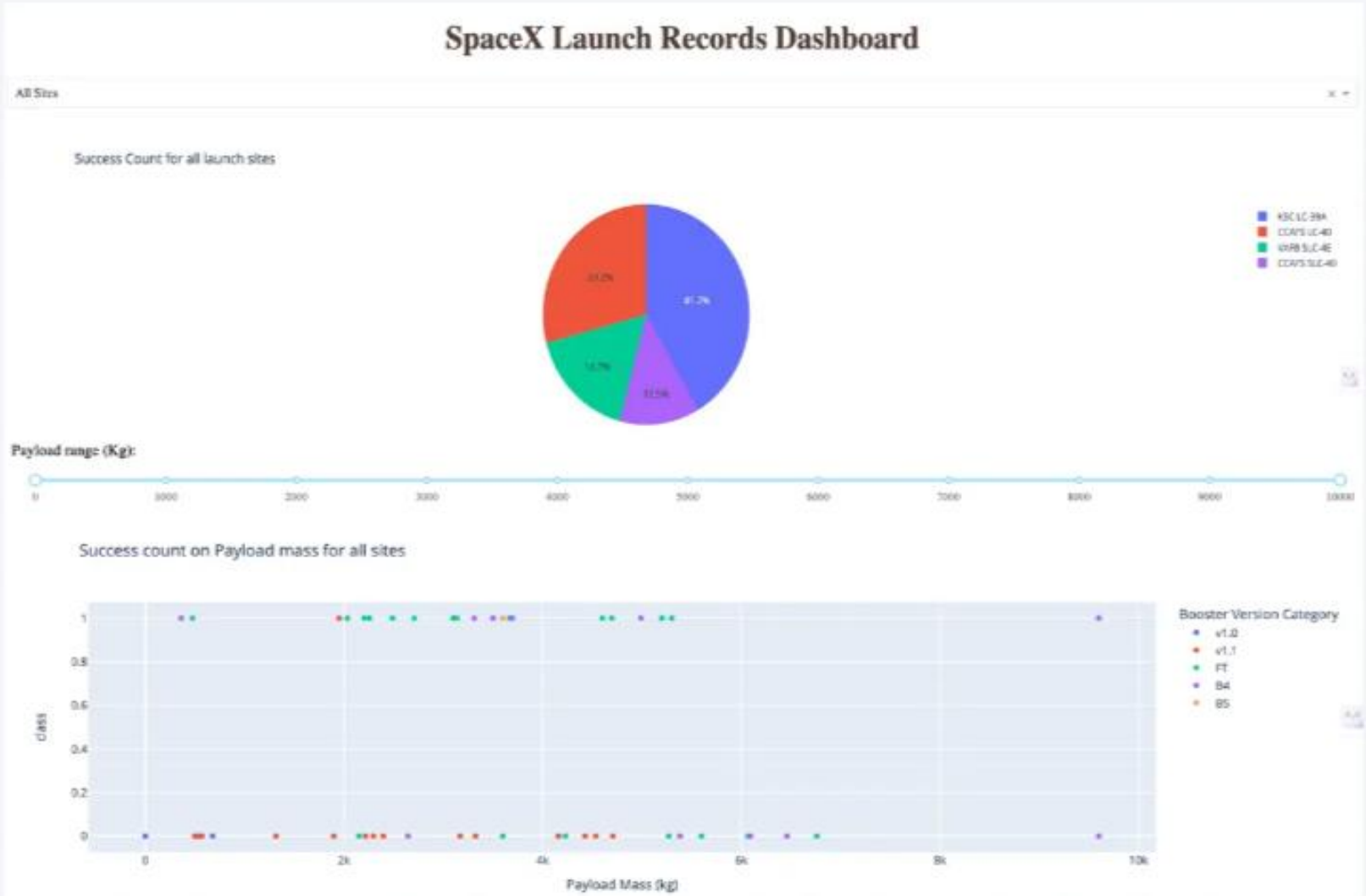


Distance marker was created to show distances between a launch site to its proximities

IBM Developer

SKILLS NETWORK

Build a Dashboard with Plotly Dash



Dropdown option of pie chart created to show the success launches of all / each site

Scatter plot with payload range slider to show the success launches of all / each site by payload mass

Predictive Analysis (Classification)

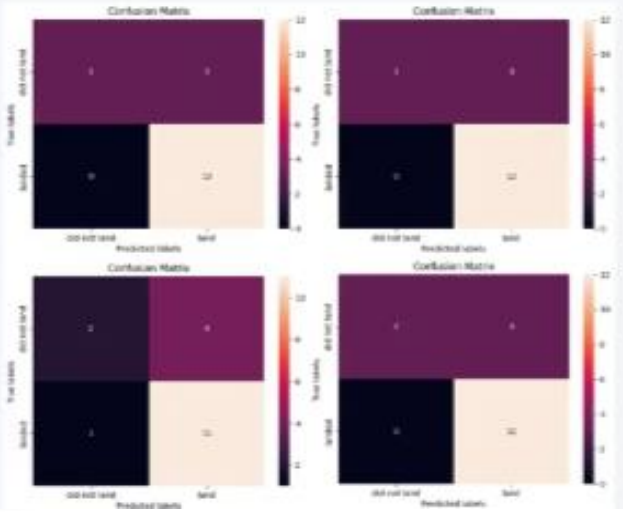
- LR, SVM, Decision Tree and KNN objects are created and fit with GridSearchCV object to find the best parameters, then the models are trained on the training set.
- The accuracy of test data are calculated for each machine learning model. It is found that the methods performed best are LR, SVM, KNN where all 3 achieved the highest accuracy of 83.33%.

TASK 12

Find the method performs best:

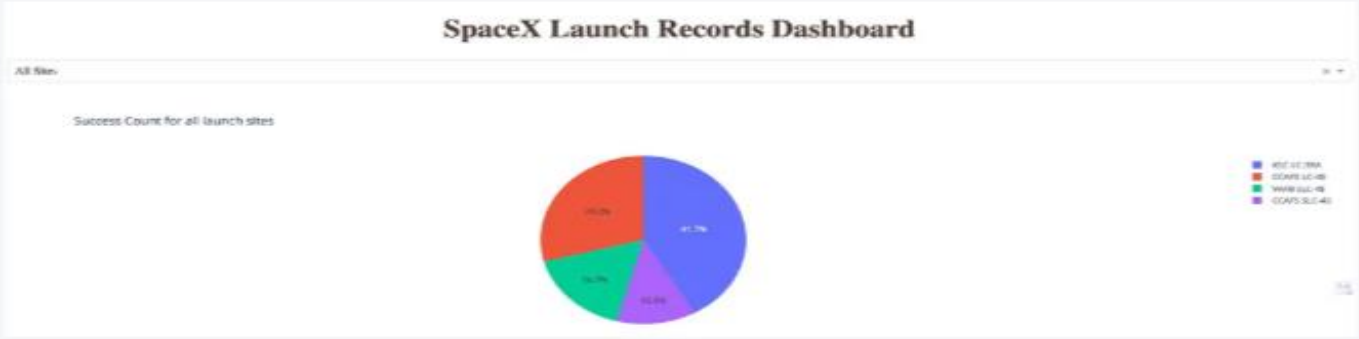
In [58]:
```
print('LR Accuracy:', '{:.2%}'.format(logreg_accuracy))
print( 'SVM Accuracy:', '{:.2%}'.format(svm_accuracy))
print('Decision Tree Accuracy:', '{:.2%}'.format(tree_accuracy))
print('KNN Accuracy:', '{:.2%}'.format(knn_accuracy))
```

```
LR Accuracy: 83.33%
SVM Accuracy: 83.33%
Decision Tree Accuracy: 72.22%
KNN Accuracy: 83.33%
```



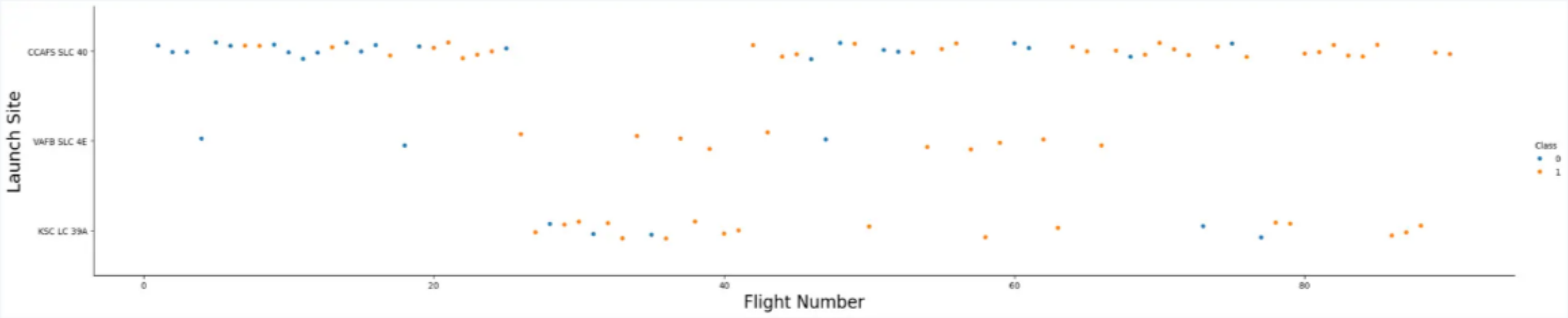
RESULTS

- LR, SVM, KNN are top-performing models for forecasting outcomes in this data.
- Lighter payloads have a higher performance compared to heavier ones.
- The likelihood of a SpaceX launch succeeding increases with the number of years of experience, suggesting a trend towards flawless launches over time.
- Launch Complex 39A at Kennedy Space Center has the highest number of successful launches compared to other launch sites.
- GEO,HEO,SSO,ES L1 orbit types exhibit the highest rates of successful launches.



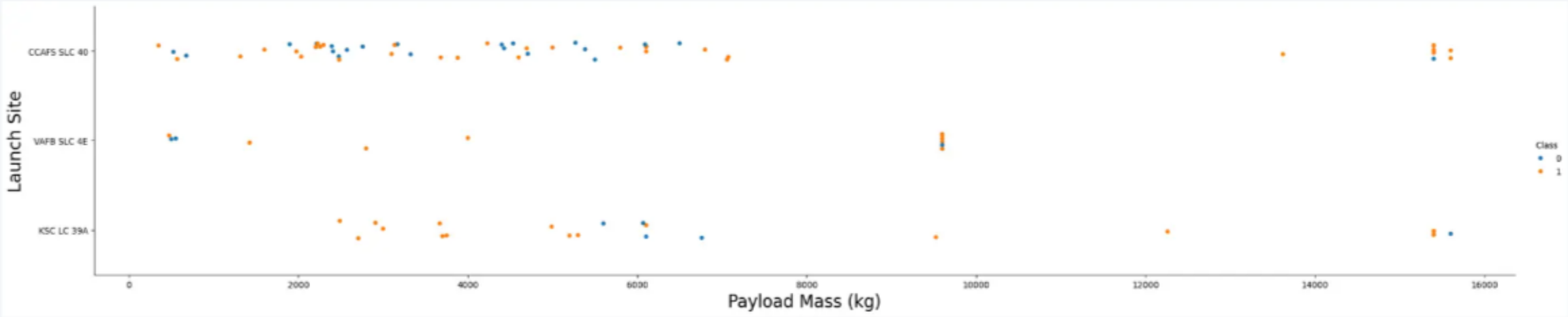
KSC LC-39A has the most successful launches overall

Flight Number vs. Launch Site



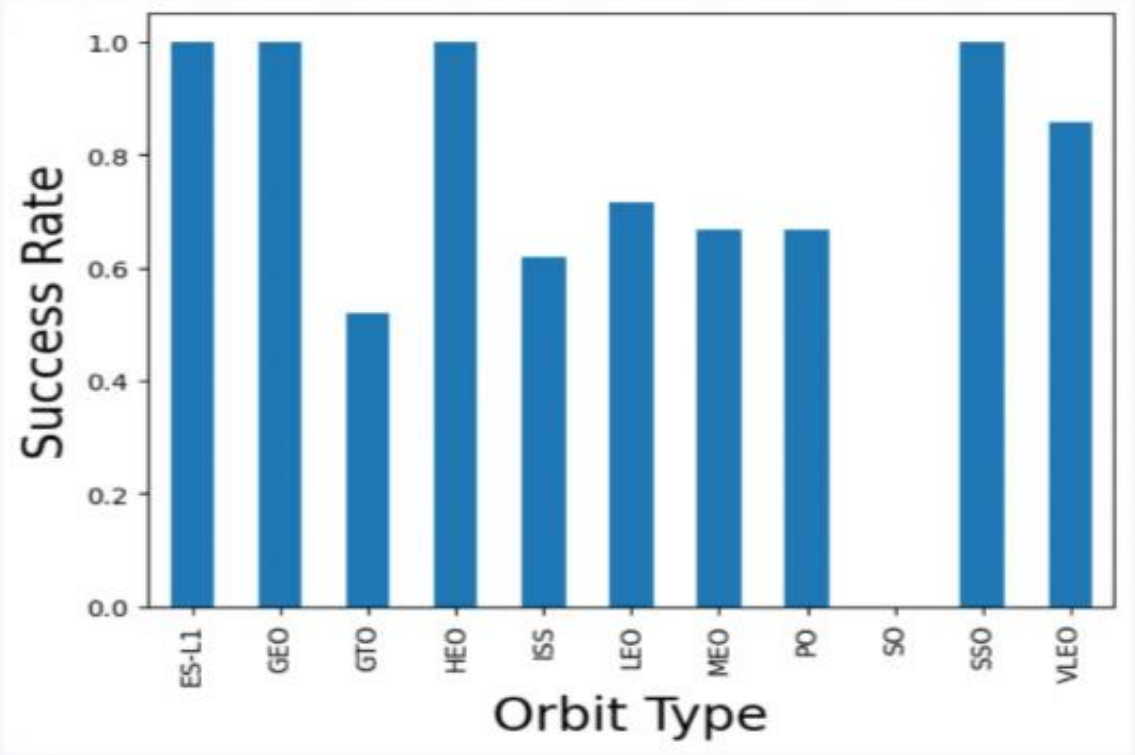
- Total number of launches from launch site CCAFS SLC 40 are significantly higher than the other launch sites.

Payload vs. Launch Site



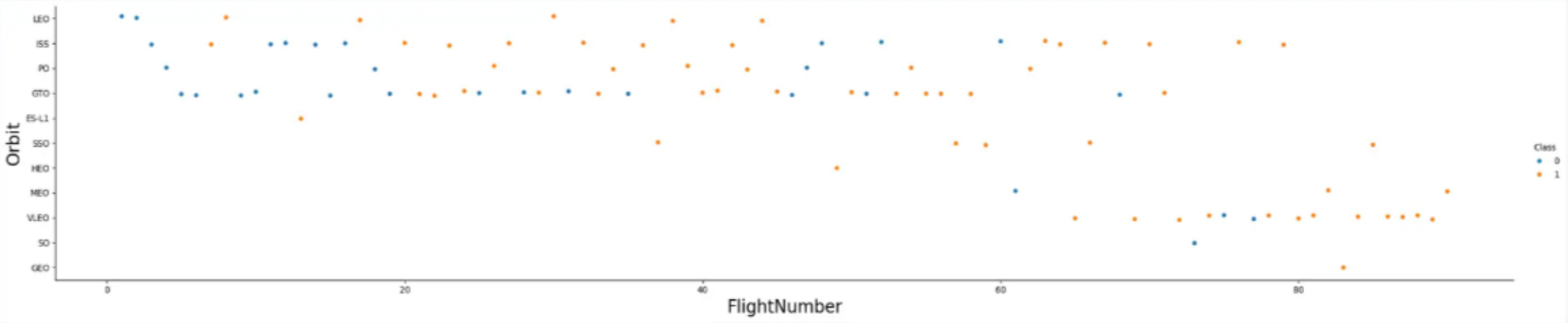
- Payloads with lower mass are have more launches compared to those with higher mass across all three launch sites.

Success Rate vs. Orbit Type



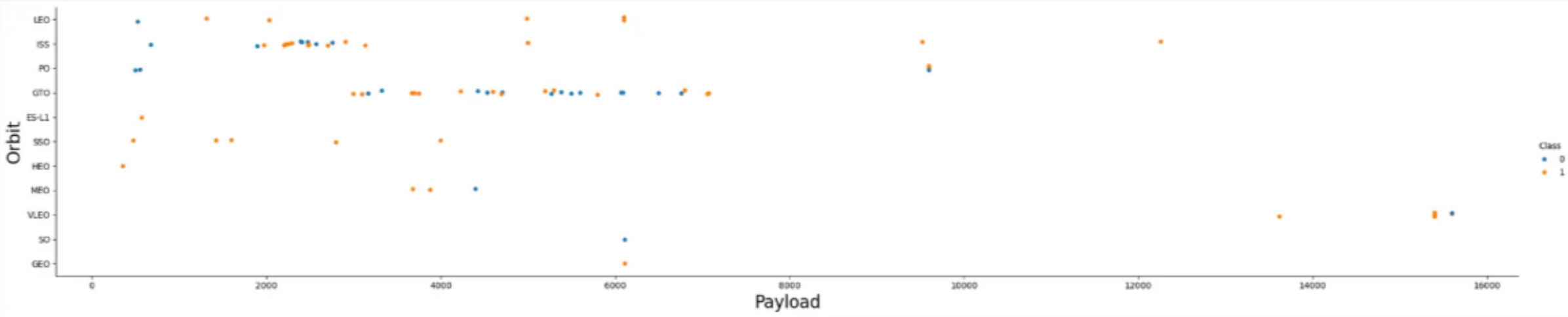
- Orbit types ES-L1, GEO, HEO, SSO have the highest success rate among all.

Flight Number vs. Orbit Type



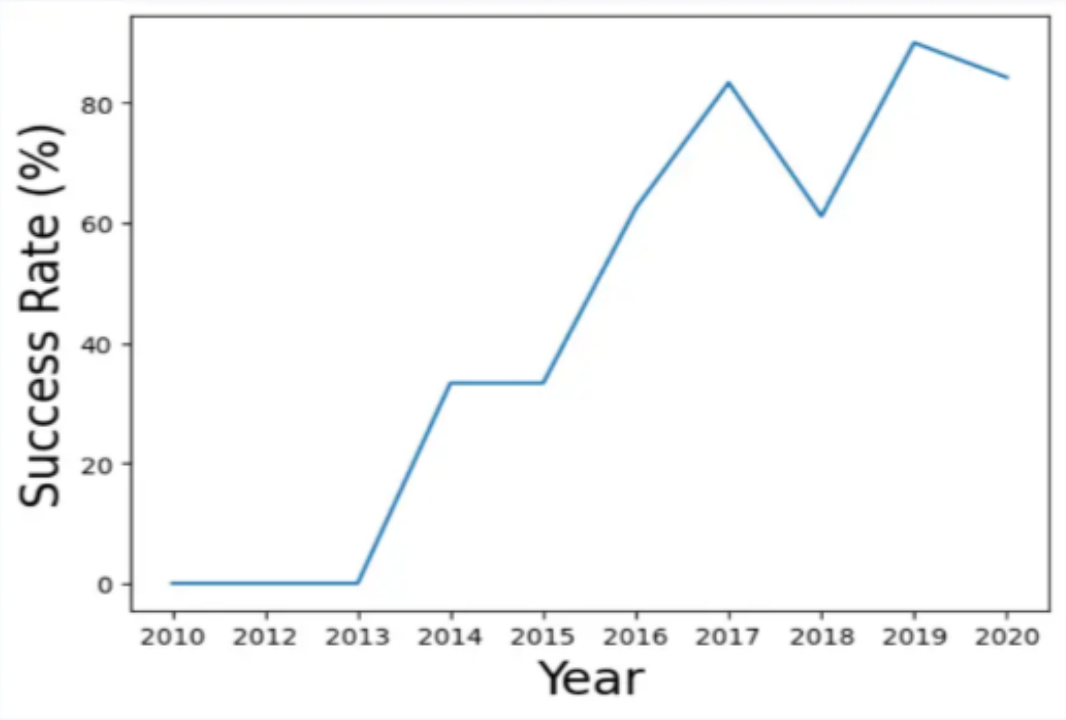
- LEO, ISS, PO, GTO orbits have the most launches in the earlier years, but it slowly shifted to VLEO orbit in the later years.

Payload vs. Orbit Type



- Heavy payloads tend to have higher successful landing rates for PO, LEO, and ISS orbits, but for GTO orbit, success is less predictable with an almost equal mix of successes and failures.

Launch Success Yearly Trend



- The success rate of launches have been increasing since 2013 till 2020, possibly due to technology advancement and experience.

All Launch Site Names

Task 1

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

In [15]:
```
%sql select distinct Launch_Site FROM SPACEXTABLE
```

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

Out[15]:

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

- Performed an SQL query to obtain all launch site names

Launch Site Names Begin with 'CCA'

Task 2

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

In [16]:
```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5
```

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

Out[16]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (p
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 (p
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	N
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	N
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	N

- Performed an SQL query to obtain 5 launch site names that begin with 'CCA'

Total Payload Mass

Task 3

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

```
In [17]: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where customer = 'NASA (CRS)'
 * sqlite:///my_data1.db
Done.
Out[17]: sum(PAYLOAD_MASS__KG_)
         45596
```

- Performed an SQL query to obtain the total payload mass carried by boosters launched by NASA (CRS)

Average Payload Mass by F9 v1.1

Task 4

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

```
In [18]: %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where Booster_Version = 'F9 v1.1'

 * sqlite:///my_data1.db
Done.
Out[18]: avg(PAYLOAD_MASS__KG_)
         2928.4
```

- Performed an SQL query to calculate the average payload mass carried by booster version F9 v1.1

First Successful Ground Landing Date

Task 5

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

Hint:Use min function

In [21]:
```
%%sql
SELECT min(Date)
FROM SPACEXTBL
WHERE Landing_Outcome = 'Success (ground pad)';
```

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

Out[21]:

min(Date)
2015-12-22

- Performed an SQL query to find the dates of the first successful landing outcome on ground pad

2015 Launch Records

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

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, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.

In [26]:
```
%sql select substr(Date, 6,2) as Month, Landing_Outcome, Booster_Version, Launch_Site   from SPACEXTABLE where La
```

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

Out[26]:

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

- Performed an SQL query to list the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015

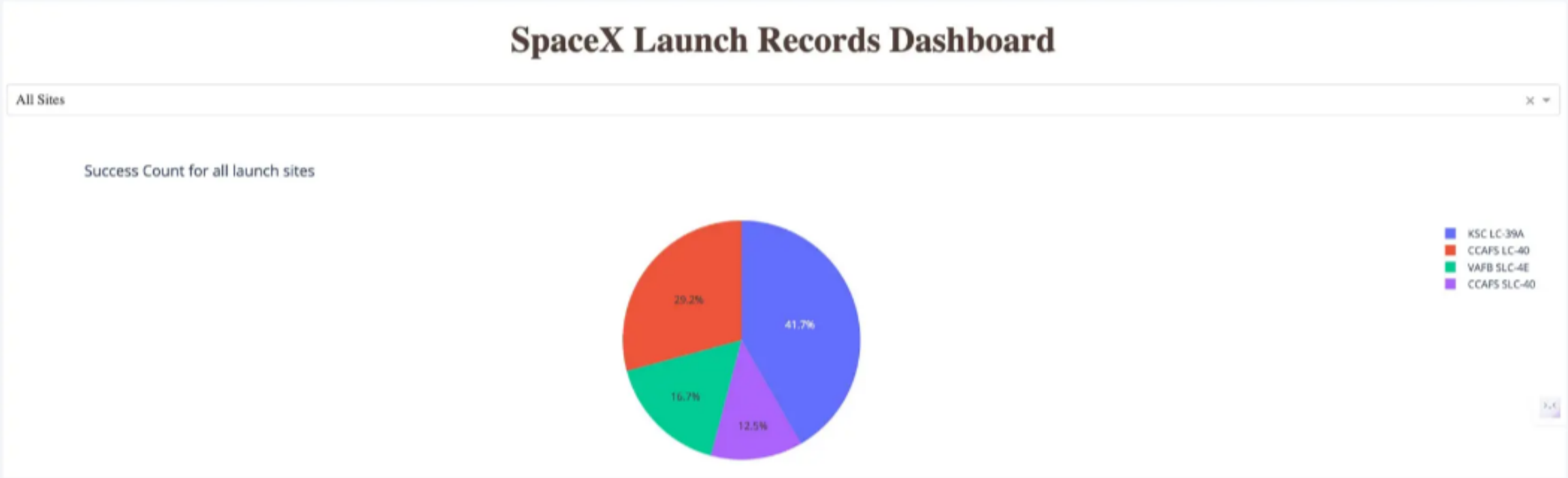
All launch sites on a map

- The launch sites are labelled by a marker with their names on the map.

All success/failed launches for each site on the map

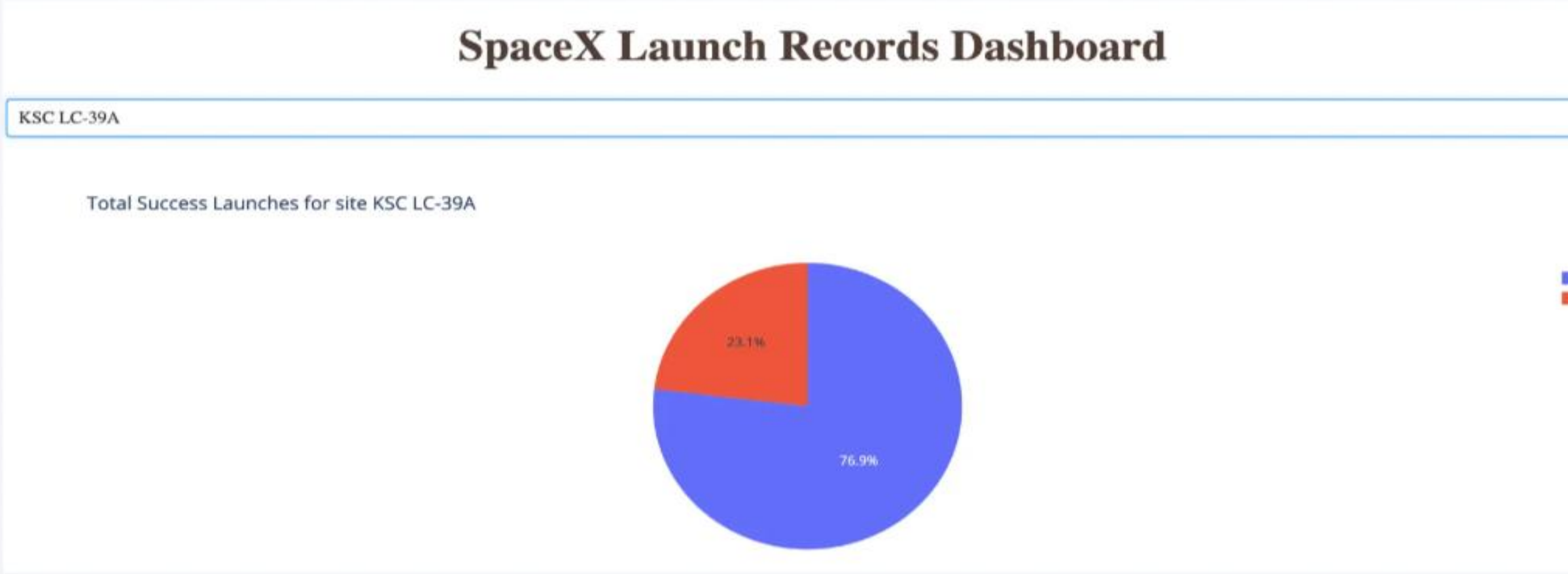
- The launch records are grouped in clusters on the map, then labelled by green markers for successful launches, and red markers for unsuccessful ones.

Total success launches for all sites



- KSC LC-39A has the highest amount of success launches with 41.7% from the entire record, whereas CCAFS SLC-40 has the lowest amount of success launches with only 12.5%.

Success ratio of the launch site with the highest success launches

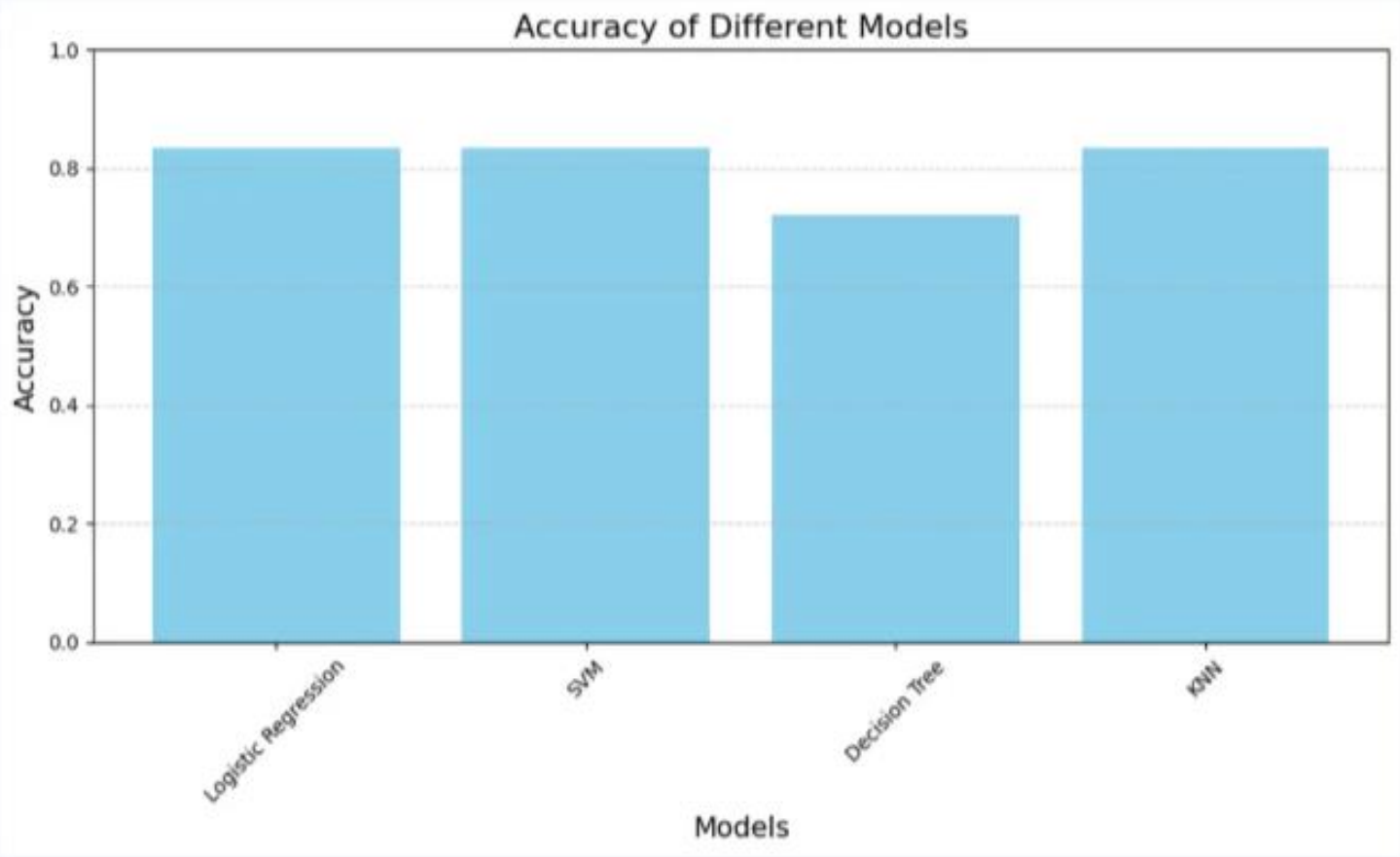


- KSC LC-39A which is the launch site with highest amount of success, has a 76.9% success rate for the launches from its site, and 23.1% failure rate.

Payload vs. launch outcome

- The payload range that has the highest success launches is between 2,000 to 4,000 kg, which can be seen by the most number of plots in that range, followed by the payload range of 4,000 to 6,000 kg, with the second most number of plots.

Classification Accuracy



TASK 12

Find the method performs best:

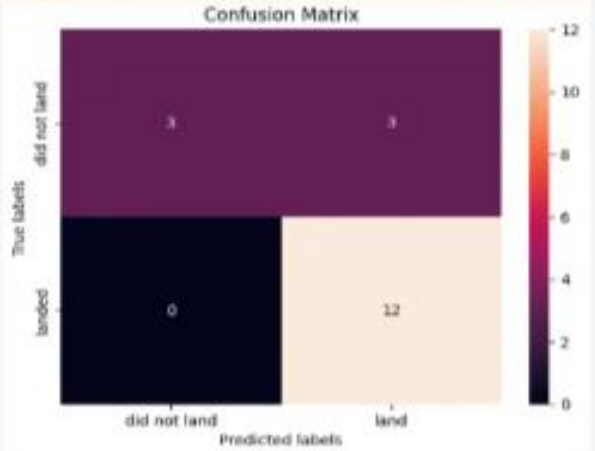
```
print('LR Accuracy:', '{:.2%}'.format(logreg_accuracy))
print( 'SVM Accuracy:', '{:.2%}'.format(svm_accuracy))
print('Decision Tree Accuracy:', '{:.2%}'.format(tree_accuracy))
print('KNN Accuracy:', '{:.2%}'.format(knn_accuracy))
```

```
LR Accuracy: 83.33%
SVM Accuracy: 83.33%
Decision Tree Accuracy: 72.22%
KNN Accuracy: 83.33%
```

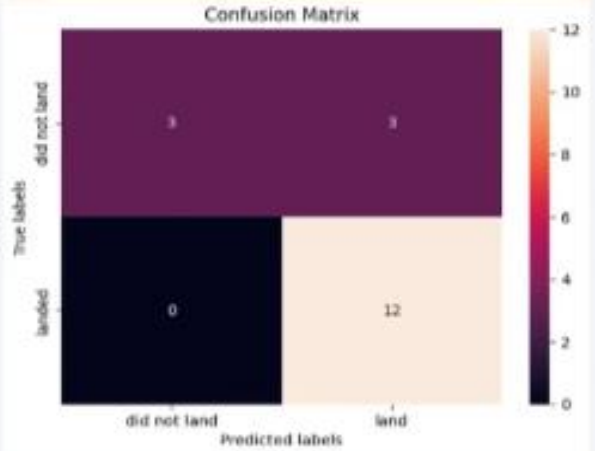
- The model that performed best are LR, SVM, KNN where all 3 achieved the highest accuracy of 83.33%.

Confusion Matrix

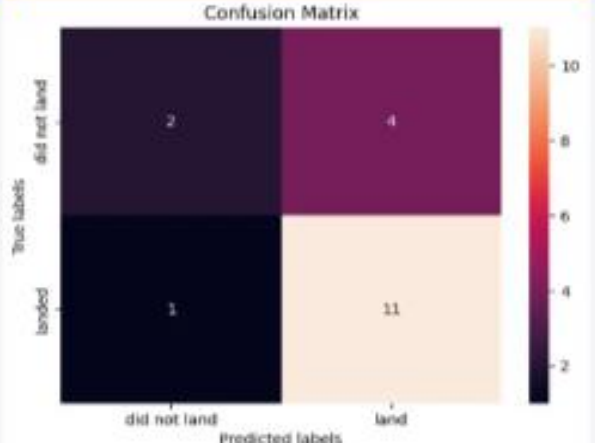
Confusion Matrix of LR



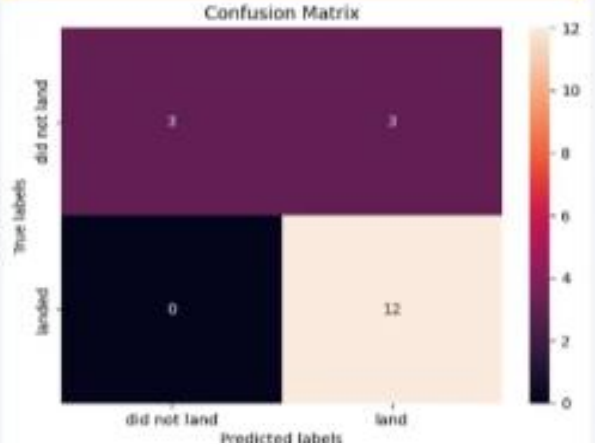
Confusion Matrix of SVM



Confusion Matrix of Decision Tree



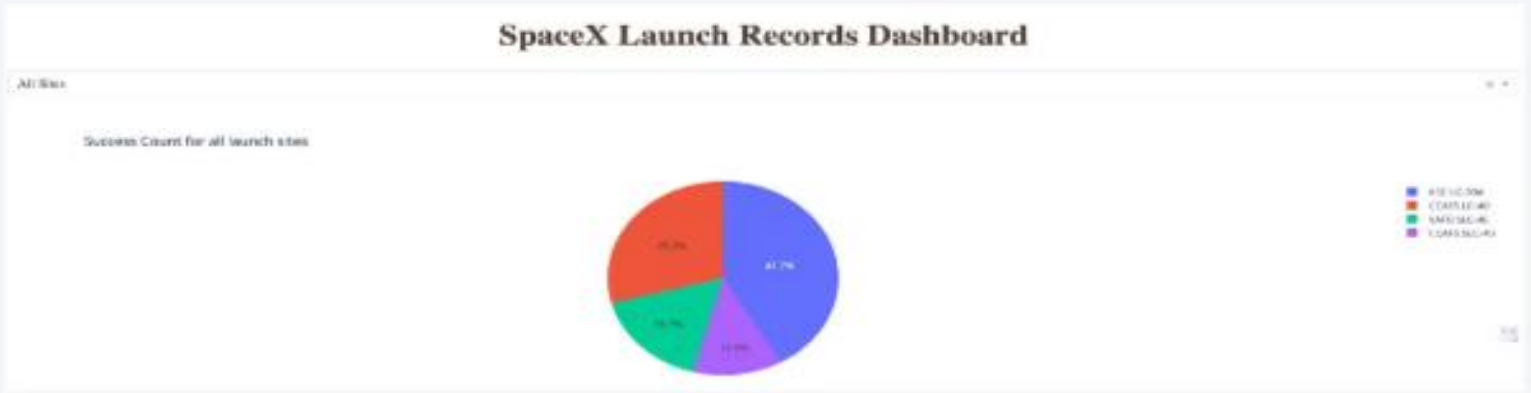
Confusion Matrix of KNN



- LR, SVM, KNN models are good as their confusion matrix show that they predicted all 12 successful landing correctly, with 0 error.
- However, the Decision Tree model only predicted 11 successful landing correctly, with one of them wrongly predicted as a failed / did not land.
- LR, SVM, KNN models have the same accuracy of 83.33% as displayed earlier, hence the same confusion matrix.

Conclusions

- LR, SVM, KNN are top-performing models for forecasting outcomes in this data.
- Lighter payloads have a higher performance compared to heavier ones.
- The likelihood of a SpaceX launch succeeding increases with the number of years of experience, suggesting a trend towards flawless launches over time.
- Launch Complex 39A at Kennedy Space Center has the highest number of successful launches compared to other launch sites.
- GEO,HEO,SSO,ES L1 orbit types exhibit the highest rates of successful launches.



KSC LC-39A has the most successful launches overall

Thanks