IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Leonardo Nakaya
06/19/2023

Outline

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

Executive Summary

- In this project we used the following methodologies:
  - Data collection with web scraping and SpaceX API.
  - Exploratory Data Analysis (EDA)
  - Machine Learning Prediction
- Summary of all results
  - We could get the necessary data to analyze in python.
  - We could find interesting insights such ash: success/failure rate by launch site and orbit.
  - We could find which prediction model is the most suitable for our project.

Introduction

- Since SpaceX is a successful spatial exploration company, we wanted to analyze data to find the viability if a new company can compete with it.
- In the first stage is where SpaceX can reduce costs, so we try to analyze the success rate of landing.
- We also want to find which orbit and launch site have the highest possibility to be successful.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected by SpaceX API (https://api.spacexdata/v4/rockets/)
  - Web scraping (https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches)
- Perform data wrangling
  - We performed EDA to get valuable information by category, modified values to be analyzed correctly and create necessary camps.
- Perform exploratory data analysis (EDA) using visualization and SQL
  - We could find some data using SQL.

Methodology

- Perform interactive visual analytics using Folium and Plotly Dash
  - We could find information about launch sites location and create interactive dashboards to visualize features of every launch site.
- Perform predictive analysis using classification models
  - We could create many predictive models and find the best for our project.

Data Collection

- Data was collected from SpaceX API and Wikipedia.

Data Collection – SpaceX API

- We used SpaceX public API.
- Source: https://github.com/LeoNakaya93/Data_Science_Project_IBM/blob/main/Data_Collection_API.ipynb

- Request API
- Filter Data
- Dealing with missing values

Data Collection - Scraping

- We made webs craping from public information in Wikipedia.

- https://github.com/LeoNakaya93/Data_Science_Project_IBM/blob/main/Data_Webscraping.ipynb

- Request from Wikipedia
- Extract information in HTML format
- Create a data frame

Data Wrangling

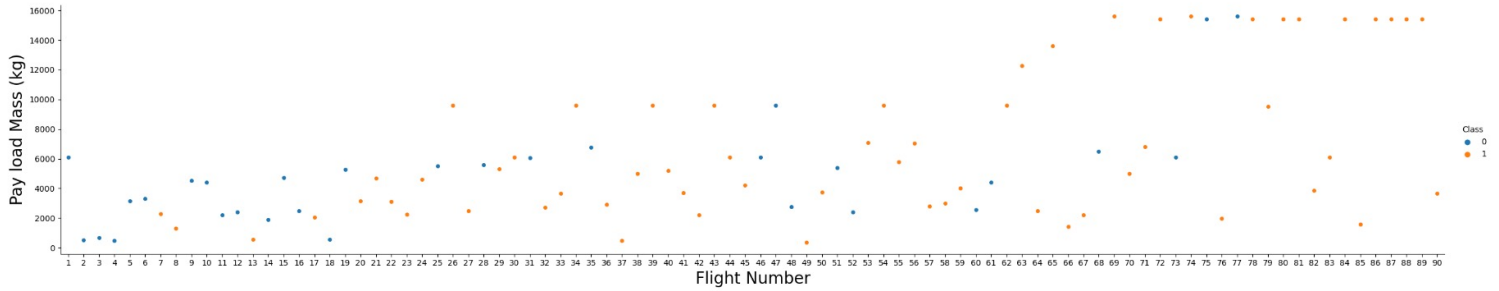
- Exploratory Data Analysis
- SQL
- Interactive map with Folium
- Dashboard with Plotly Dash

EDA with Data Visualization

- Source: https://github.com/LeoNakaya93/Data_Science_Project_IBM/blob/main/EDA_Dataviz.ipynb
- We analyzed correlation between features such as “PayloadMass” and “FlightNumber” by class.

In [5]:

```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



We see that different launch sites have different success rates. `CCAFS LC-40`, has a success rate of 60 %, while `KSC LC-39A` and `VAFB SLC 4E` has a success rate of 77%.

EDA with SQL

- Source: https://github.com/LeoNakaya93/Data_Science_Project_IBM/blob/main/EDA_with_SQL.ipynb
- We could find value information through SQL commands.

Task 3

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

In [18]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) \
  FROM SPACEXTBL \
  WHERE Customer = 'NASA (CRS)';
```

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

Out[18]:

SUM(PAYLOAD_MASS__KG_)
45596.0

Task 4

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

In [19]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) \
  FROM SPACEXTBL \
  WHERE Booster_Version = 'F9 v1.1';
```

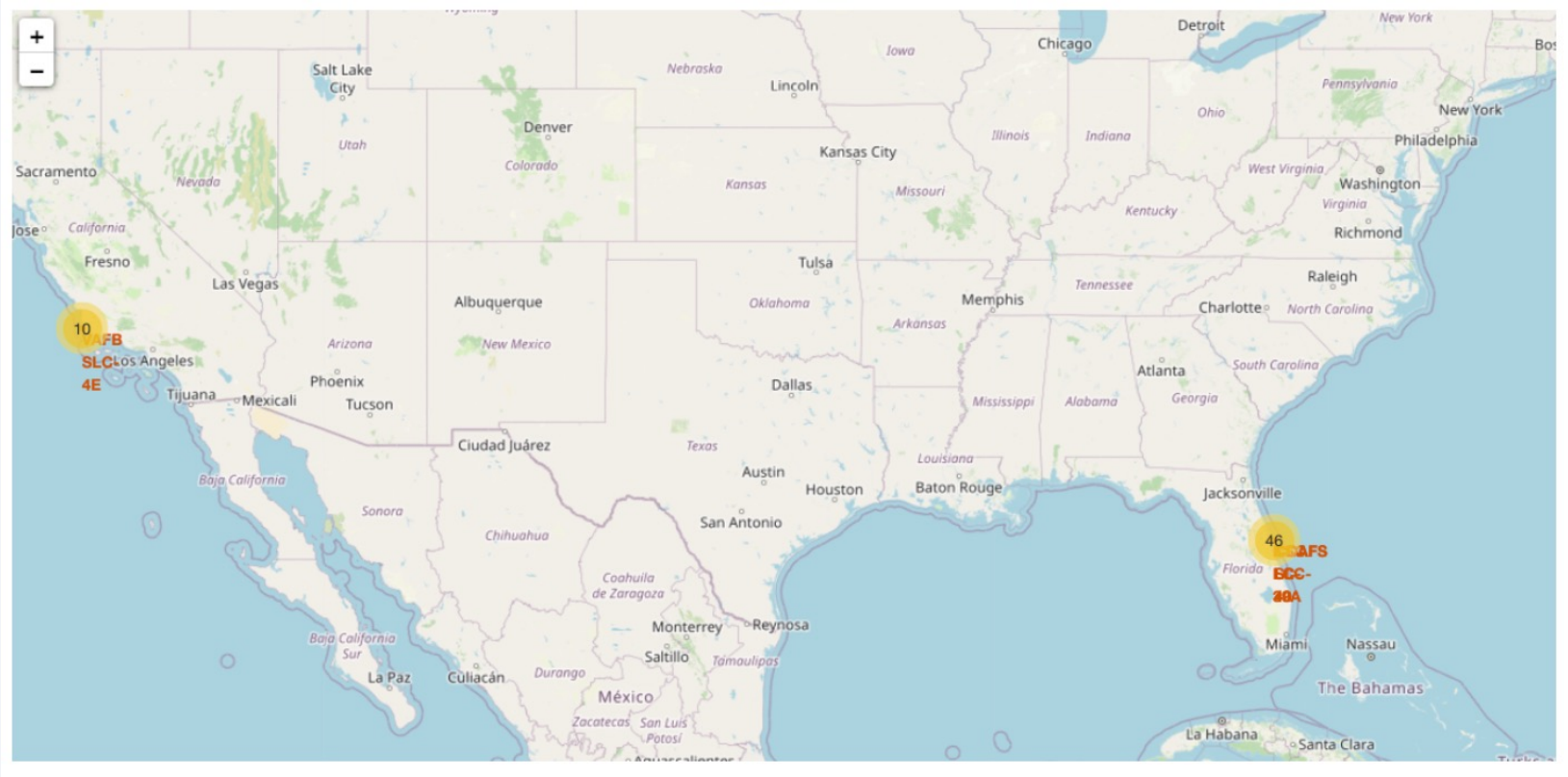
```
* sqlite:///my_data1.db
Done.
```

Out[19]:

AVG(PAYLOAD_MASS__KG_)
2928.4

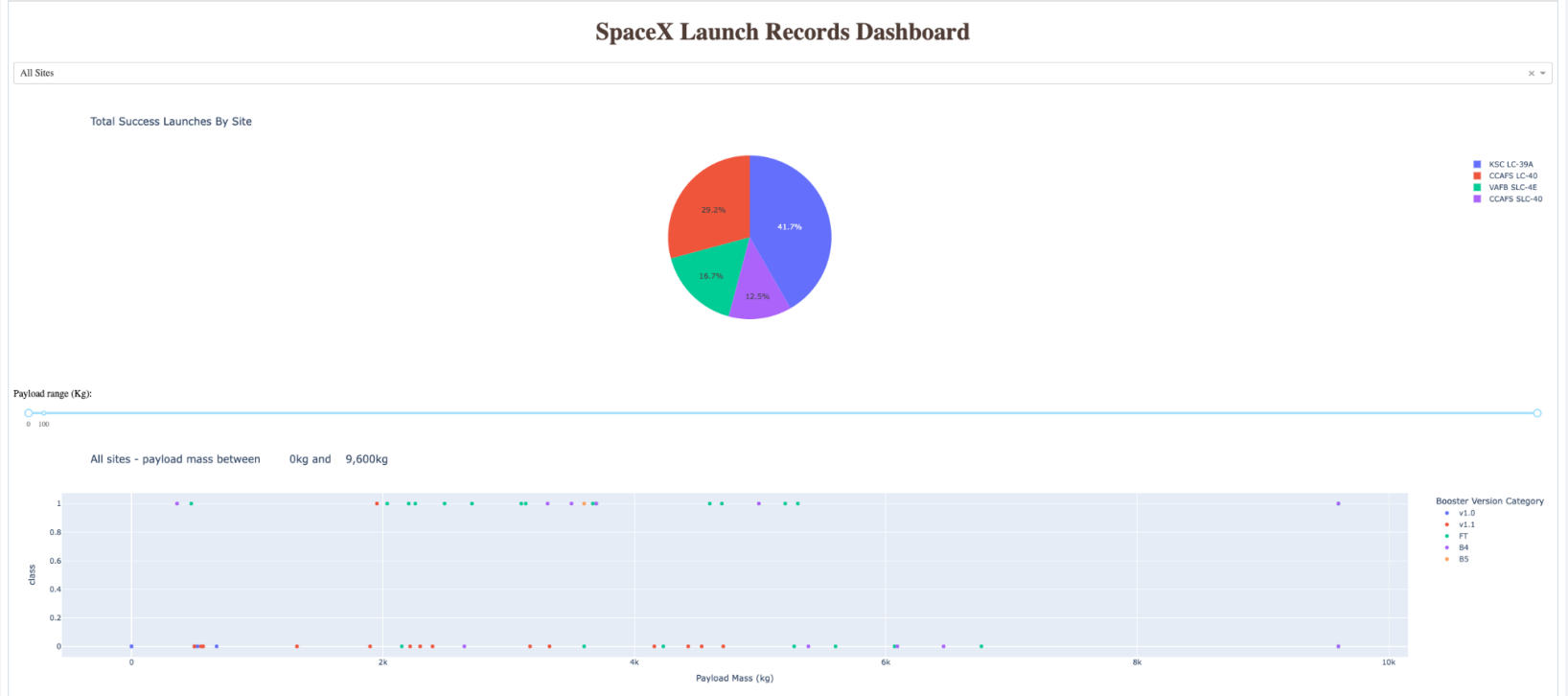
Build an Interactive Map with Folium

- Source: https://github.com/LeoNakaya93/Data_Science_Project_IBM/blob/main/Dataviz_with_Folium.ipynb
- We could analyze launch site locations information with folium library.



Build a Dashboard with Plotly Dash

- Source: https://github.com/LeoNakaya93/Data_Science_Project_IBM/blob/main/Dashboard_with_Ploty_Dash.py
- We could create an interactive dashboard with plotly dash.



Predictive Analysis (Classification)

- Source: https://github.com/LeoNakaya93/Data_Science_Project_IBM/blob/main/Machine_Learning_Prediction.ipynb
- We created many predictive models for machine learning.

TASK 12

Find the method performs best:

In [92]:

```
print("Model\t\tAccuracy\tTestAccuracy")#,logreg_cv.best_score_)
print("LogReg\t\t{}\t\t{}".format((logreg_cv.best_score_).round(5), logreg_cv.score(X_test, Y_test).round(5)))
print("SVM\t\t{}\t\t{}".format((svm_cv.best_score_).round(5), svm_cv.score(X_test, Y_test).round(5)))
print("Tree\t\t{}\t\t{}".format((tree_cv.best_score_).round(5), tree_cv.score(X_test, Y_test).round(5)))
print("KNN\t\t{}\t\t{}".format((knn_cv.best_score_).round(5), knn_cv.score(X_test, Y_test).round(5)))

comparison = {}

comparison['LogReg'] = {'Accuracy': logreg_cv.best_score_.round(5), 'TestAccuracy': logreg_cv.score(X_test, Y_test).round(5)}
comparison['SVM'] = {'Accuracy': svm_cv.best_score_.round(5), 'TestAccuracy': svm_cv.score(X_test, Y_test).round(5)}
comparison['Tree'] = {'Accuracy': tree_cv.best_score_.round(5), 'TestAccuracy': tree_cv.score(X_test, Y_test).round(5)}
comparison['KNN'] = {'Accuracy': knn_cv.best_score_.round(5), 'TestAccuracy': knn_cv.score(X_test, Y_test).round(5)}
```

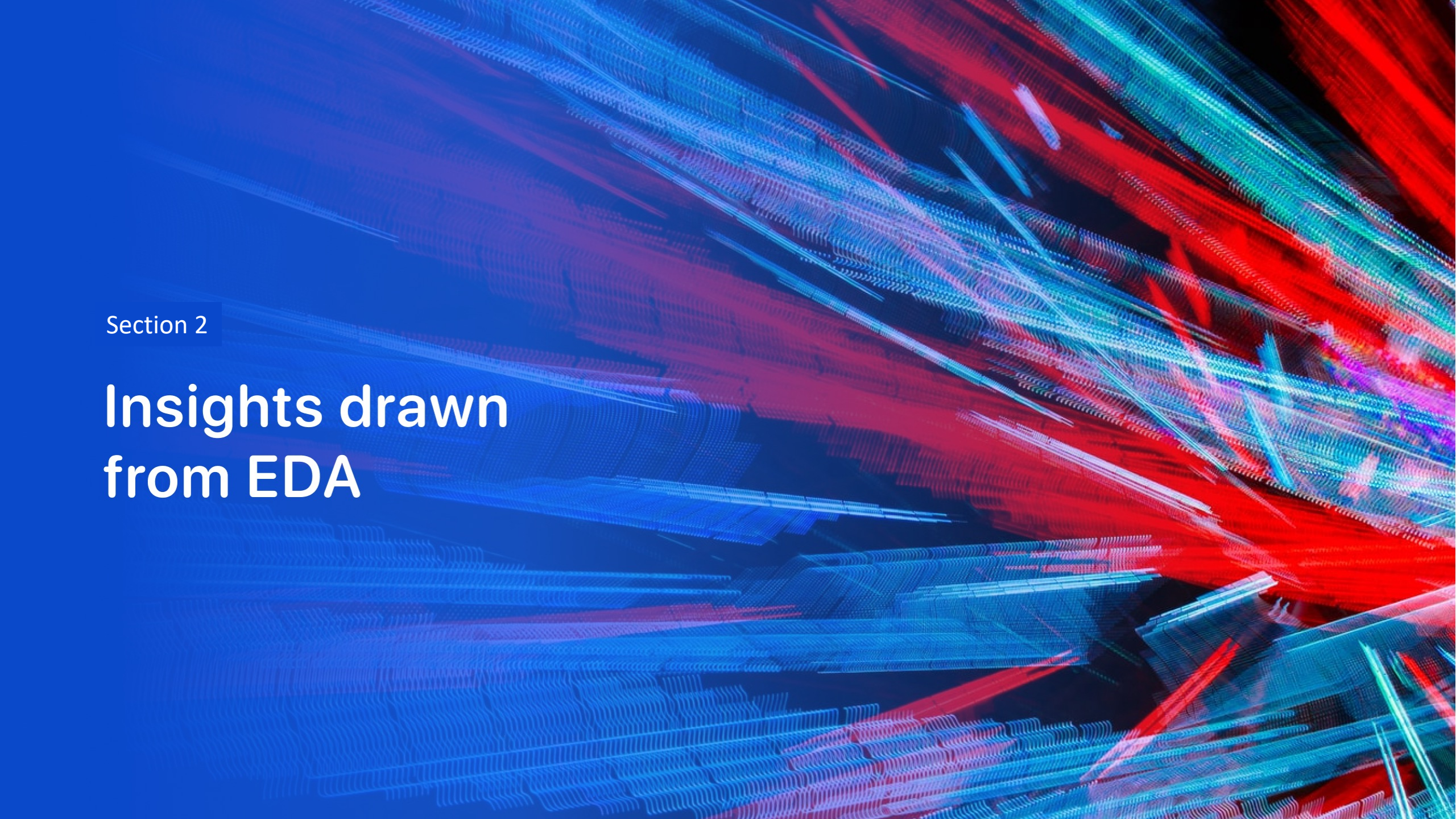
Model	Accuracy	TestAccuracy
LogReg	0.84643	0.83333
SVM	0.84821	0.83333
Tree	0.90179	0.83333
KNN	0.84821	0.83333

In [34]:

```
#We can see that the best method is the "Tree" method.
```

Results

- The first successful landing was in 2015.
- Every year success landing outcomes have been improving.
- SpaceX has three launch sites in Florida (East Coast) and one in California (West Coast).
- Decision Tree Classifier is the best predictive model to successful landings in comparison to LogReg, SVM and KNN.


Section 2
Insights drawn from EDA

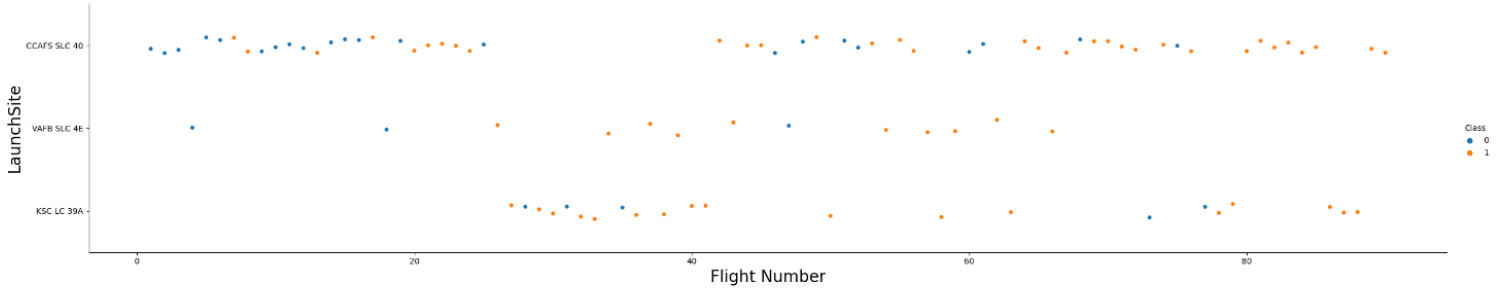
Flight Number vs. Launch Site

- Scatterplot of Flight Number vs Launch Site.

In [7]:

```
### TASK 1: Visualize the relationship between Flight Number and Launch Site

sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

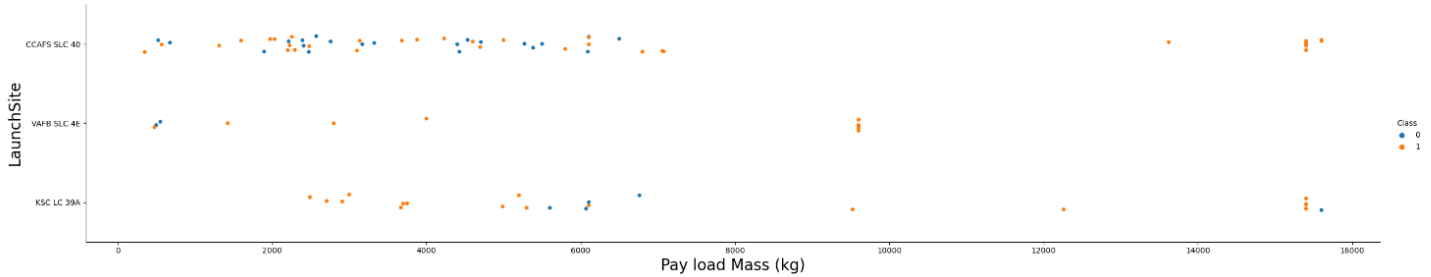
Payload vs. Launch Site

- Scatterplot of Payload vs Launch Site

In [8]:

```
### TASK 2: Visualize the relationship between Payload and Launch Site

sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



We also want to observe if there is any relationship between launch sites and their payload mass.

Success Rate vs. Orbit Type

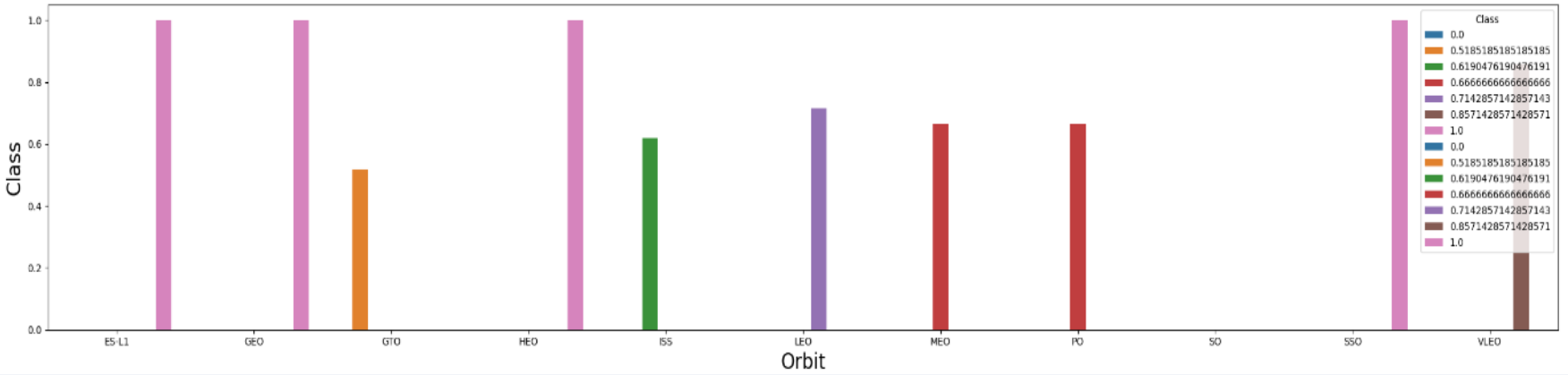
- Success rate by orbit type.

In [18]:

```
### TASK  3: Visualize the relationship between success rate of each orbit type

orbit_success = df.groupby('Orbit').mean()
orbit_success.reset_index(inplace=True)

sns.barplot(y="Class", x="Orbit", hue="Class", data=orbit_success)
plt.xlabel("Orbit",fontsize=20)
plt.ylabel("Class",fontsize=20)
plt.show()
```



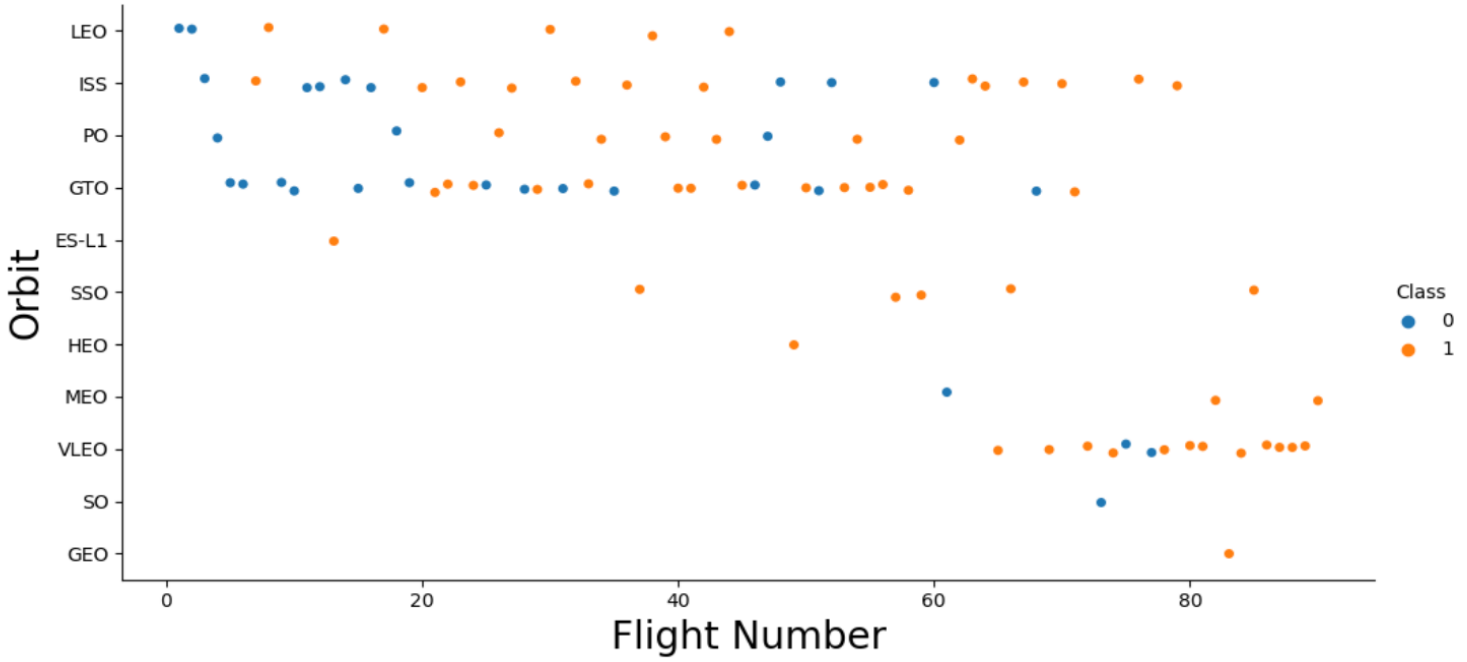
Flight Number vs. Orbit Type

- Scatterplot of Flight Number vs Orbit Type.

In [23]:

```
### TASK  4: Visualize the relationship between FlightNumber and Orbit type

sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 2)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



For each orbit, we want to see if there is any relationship between FlightNumber and Orbit type.

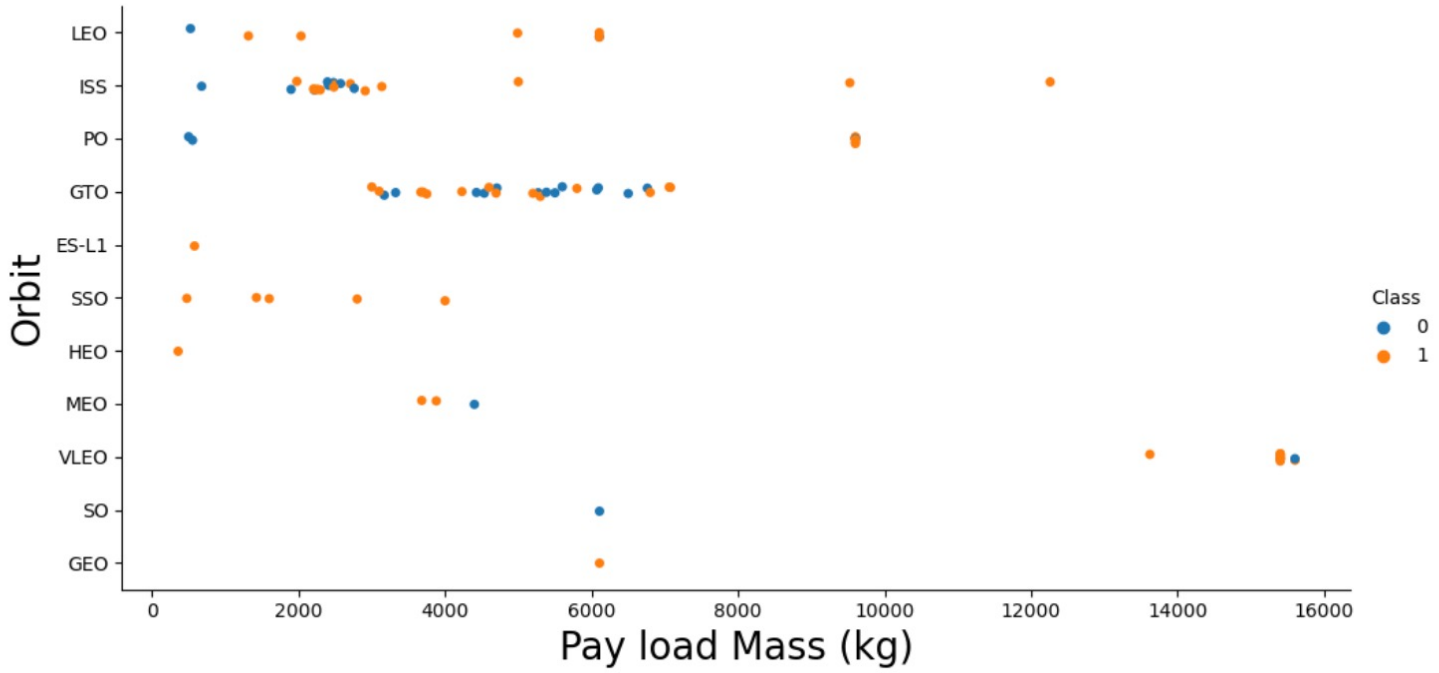
Payload vs. Orbit Type

- Scatterplot of Payload vs Orbit Type.

In [24]:

```
### TASK  5: Visualize the relationship between Payload and Orbit type

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



Similarly, we can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type

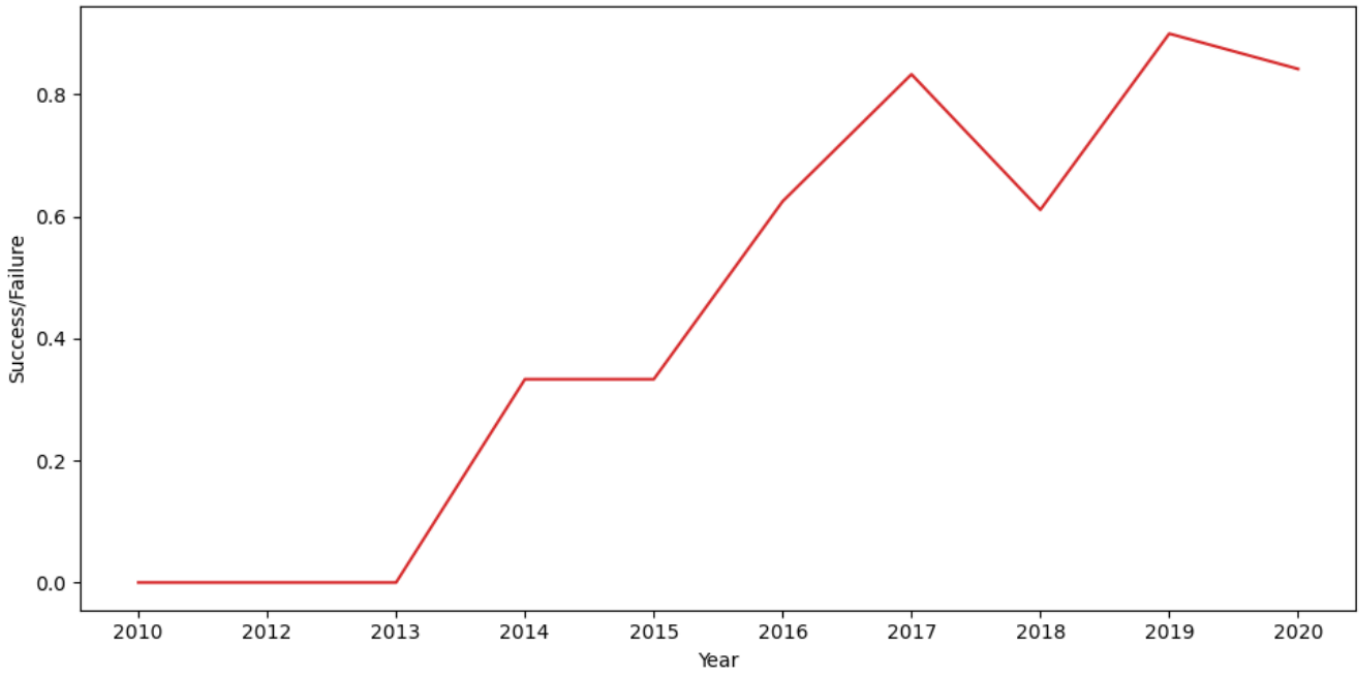
Launch Success Yearly Trend

- Line chart of yearly average success rate

In [36]:

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

plt.plot(average_by_year['Year'],average_by_year['Class'])
plt.xlabel('Year')
plt.ylabel('Success/Failure')
plt.show()
```



All Launch Site Names

- We could find 4 launch sites.

In [41]:

```
# Select relevant sub-columns: `Launch Site`, `Lat(Latitude)`, `Long(Longitude)`, `class`
spacex_df = spacex_df[['Launch Site', 'Lat', 'Long', 'class']]
launch_sites_df = spacex_df.groupby(['Launch Site'], as_index=False).first()
launch_sites_df = launch_sites_df[['Launch Site', 'Lat', 'Long']]
launch_sites_df
```

Out[41]:

	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610745

Launch Site Names Begin with 'CCA'

Task 2

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

In [25]:

```
%sql SELECT * \
   FROM SPACEXTBL \
   WHERE LAUNCH_SITE LIKE'CCA%' LIMIT 5;
```

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

Out[25]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Lan
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Fail
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	Fail
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	

Total Payload Mass

Task 3

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

In [18]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) \
  FROM SPACEXTBL \
  WHERE Customer = 'NASA (CRS)';
```

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

Out[18]:

SUM(PAYLOAD_MASS__KG_)
45596.0

Average Payload Mass by F9 v1.1

Task 4

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

In [19]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) \
  FROM SPACEXTBL \
  WHERE Booster_Version = 'F9 v1.1';
```

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

Out[19]:

AVG(PAYLOAD_MASS__KG_)
2928.4

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 [26]:
```
%sql SELECT MIN(Date) \
   FROM SPACEXTBL \
   WHERE Landing_Outcome = 'Success (ground pad)';
```

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

Out[26]:

MIN(Date)
01/08/2018

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 [29]:

```
%sql SELECT Payload \
   FROM SPACEXTBL \
   WHERE Landing_Outcome = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

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

Out[29]:

Payload
JCSAT-14
JCSAT-16
SES-10
SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

In [31]:
```
%sql SELECT Mission_Outcome, COUNT(*) as total_number \
   FROM SPACEXTBL \
   GROUP BY Mission_Outcome;
```

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

Out[31]:

Mission_Outcome	total_number
None	898
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

Task 8

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

In [34]:

```
%sql SELECT Booster_Version, PAYLOAD_MASS__KG_ \
  FROM SPACEXTBL \
  WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

Out[34]:

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600.0
F9 B5 B1049.4	15600.0
F9 B5 B1051.3	15600.0
F9 B5 B1056.4	15600.0
F9 B5 B1048.5	15600.0
F9 B5 B1051.4	15600.0
F9 B5 B1049.5	15600.0
F9 B5 B1060.2	15600.0
F9 B5 B1058.3	15600.0
F9 B5 B1051.6	15600.0
F9 B5 B1060.3	15600.0
F9 B5 B1049.7	15600.0

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 [37]:
```
%sql SELECT substr(Date,4,2) as month, Date, Booster_Version, Launch_Site, Landing_Outcome \
FROM SPACEXTBL \
where Landing_Outcome = 'Failure (drone ship)' and substr(Date,7,4)='2015';
```

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

Out[37]:

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

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

Task 10

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

In [42]:

```
%sql SELECT Landing_Outcome, COUNT(*) as Count_Outcomes \
FROM SPACEXTBL \
WHERE Date BETWEEN '04-06-2010' AND '20-03-2017' \
GROUP BY Landing_Outcome ORDER BY Count_Outcomes DESC;
```

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

Out[42]:

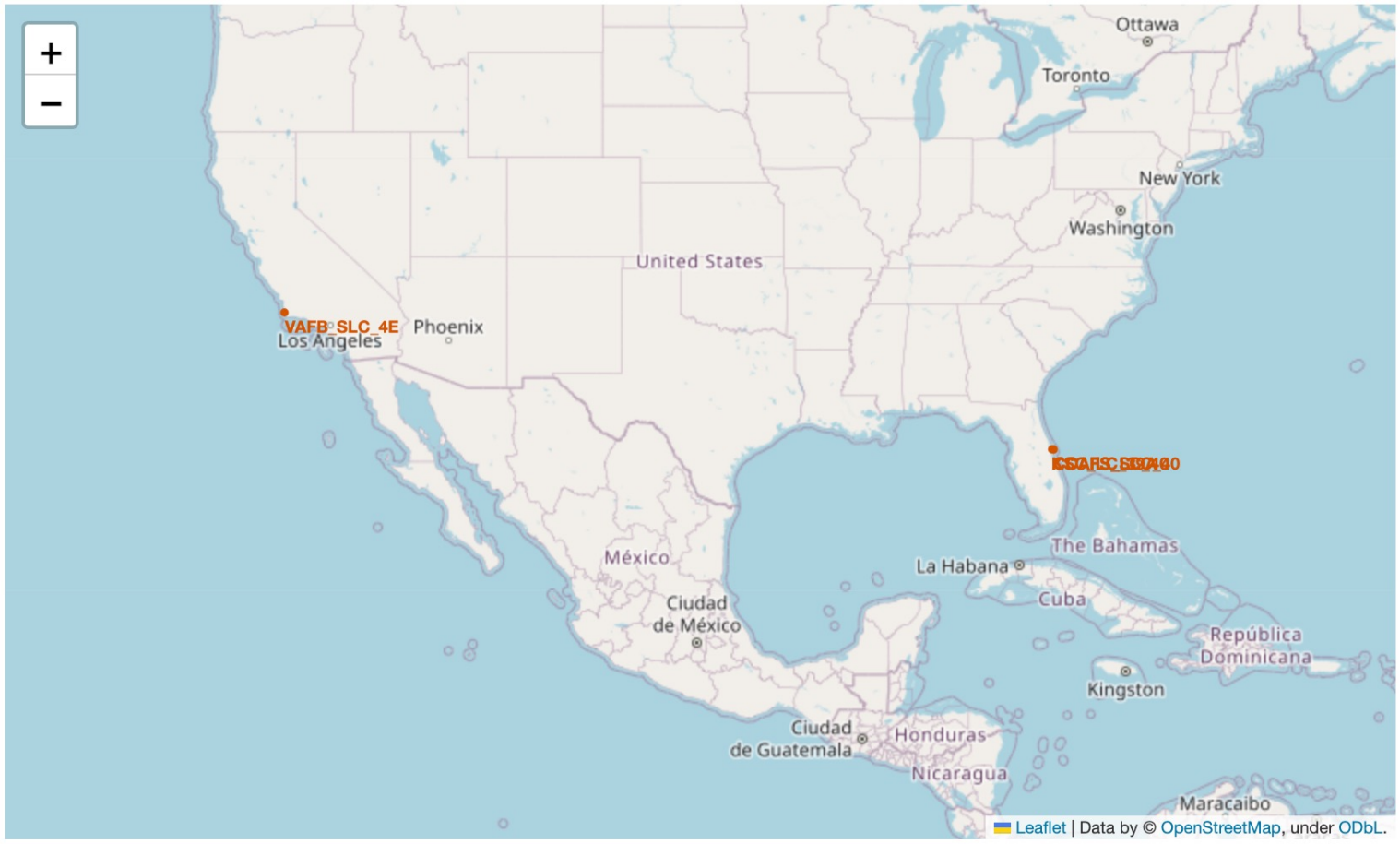
Landing_Outcome	Count_Outcomes
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	7
Failure (drone ship)	3
Failure	3
Failure (parachute)	2
Controlled (ocean)	2
No attempt	1


Section 3
Launch Sites
Proximities Analysis

Launch Sites of SpaceX in USA

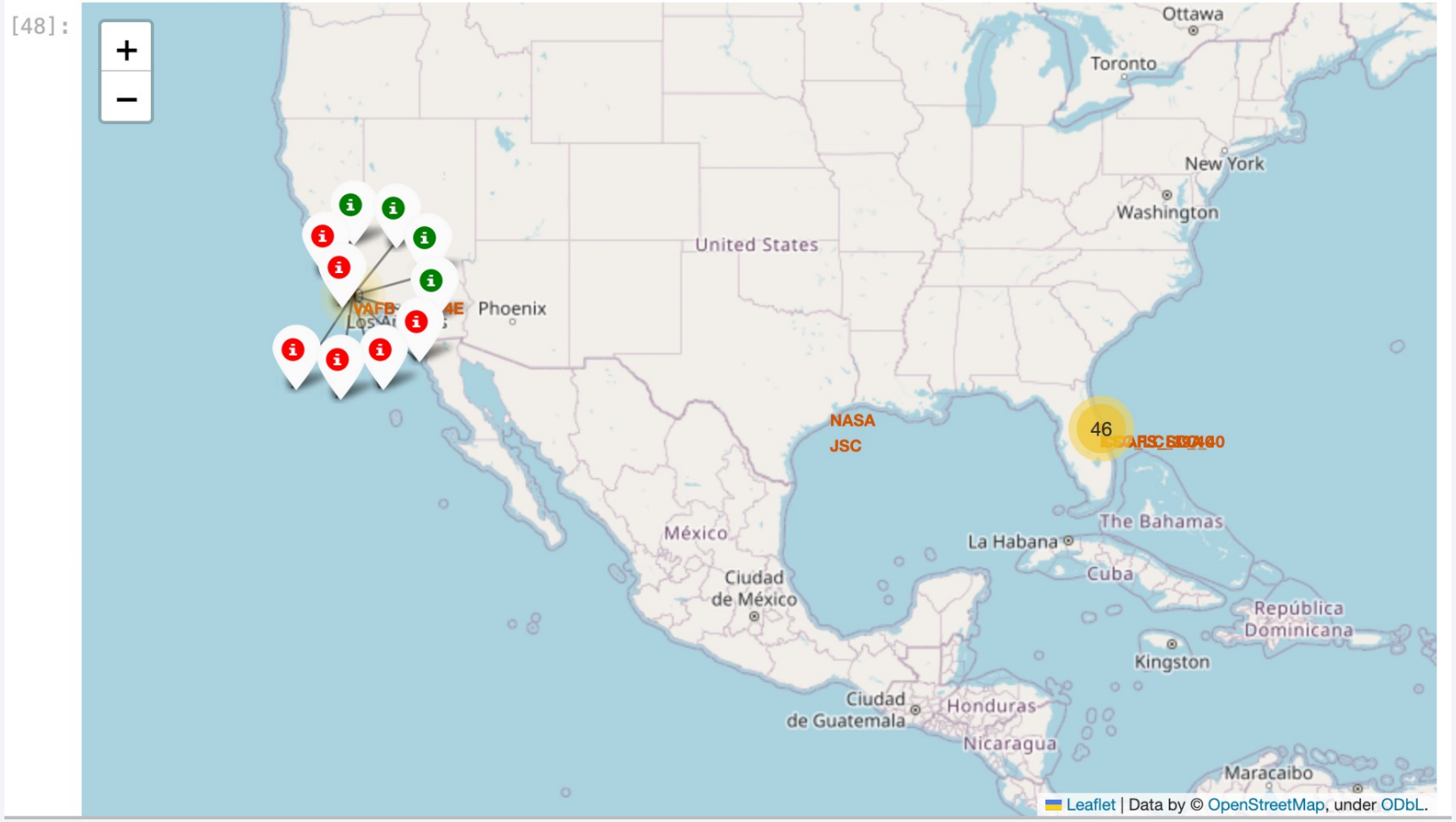
- We could find tree launch sites in east coast and one in west coast.

[44]:



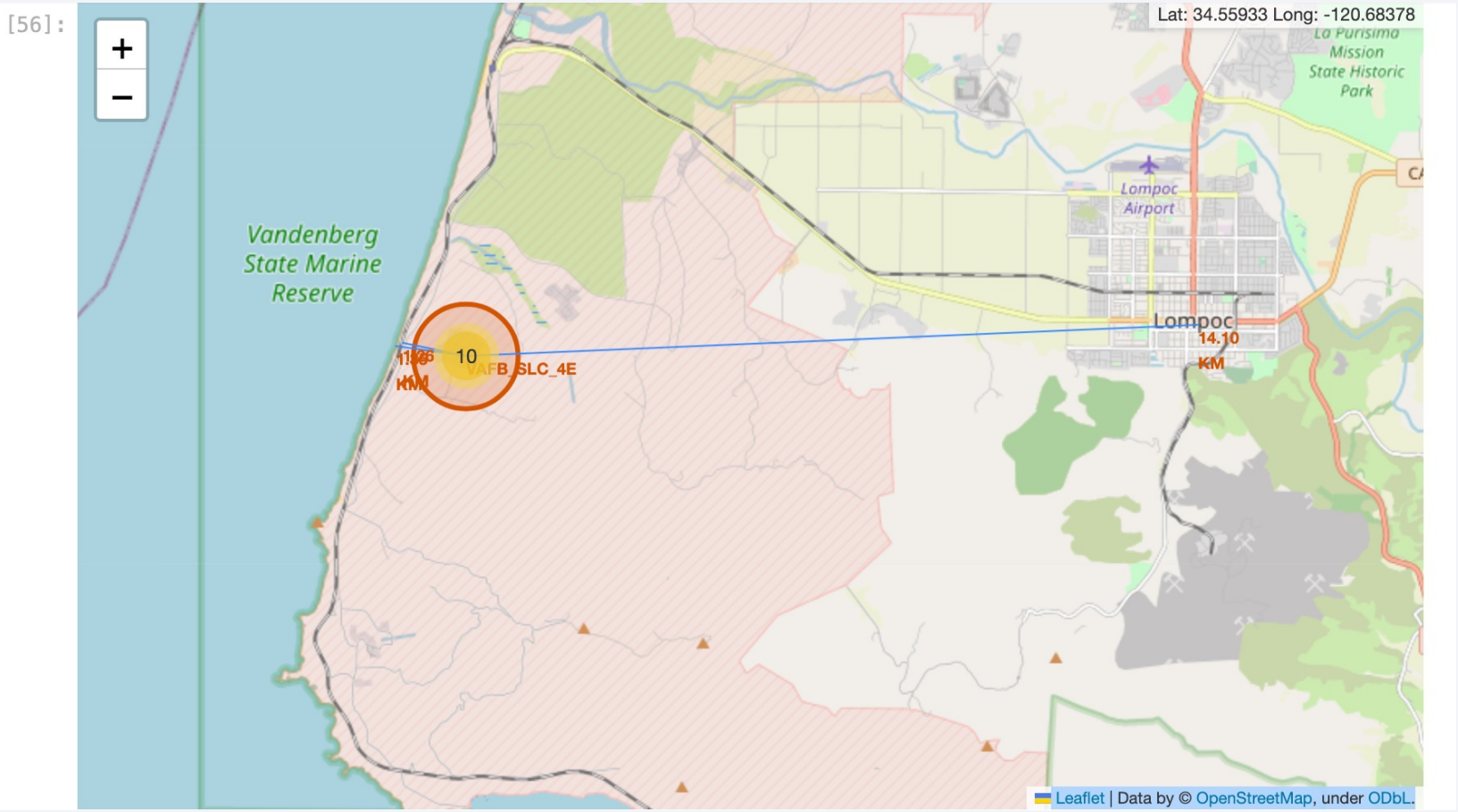
Success/Failure Rate by Launch Site

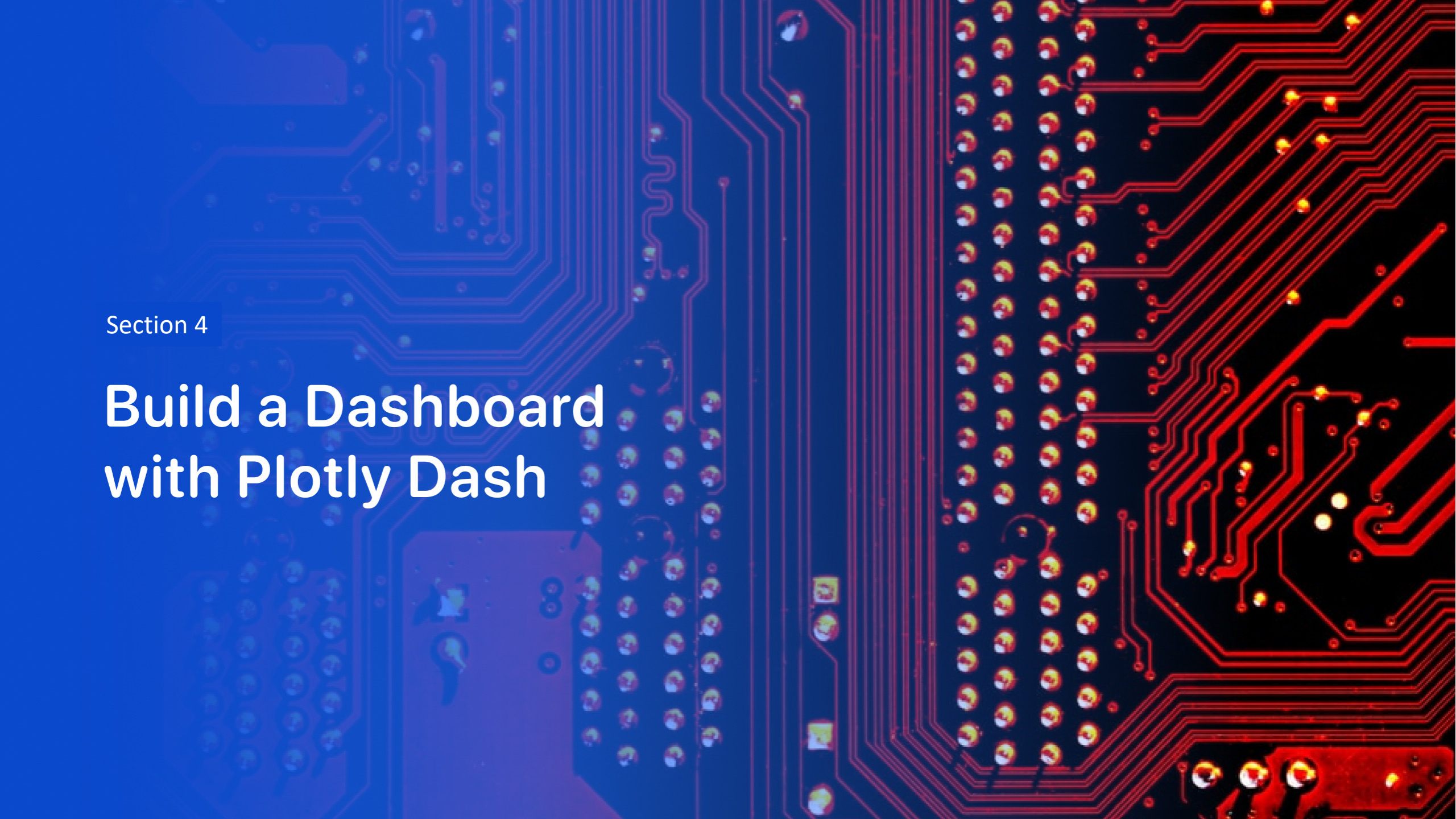
- We could fin success and failure outcomes by launch site, in the case of VAFB SLC-4E in California were 4 success out of 10, and 6 failures.



Nearest Locations from Launch Site

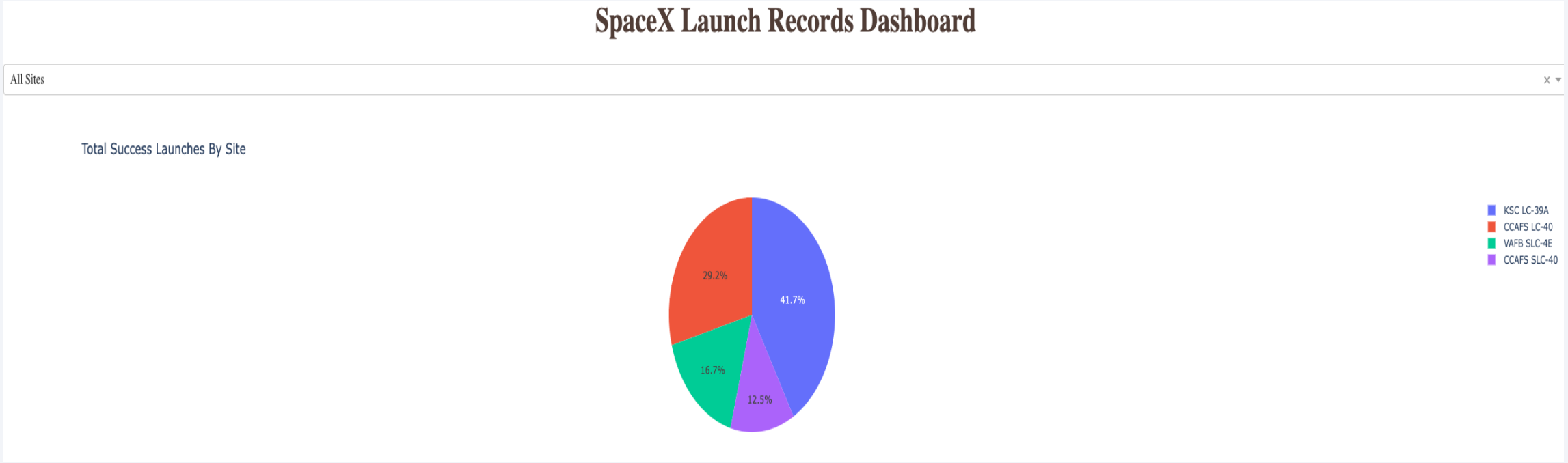
- We could find that nearest city is Lompoc with 14.10km from VAFB SLC-4E, the nearest coastline is 1.35km, the nearest railway is 1.26km and no near highway.




Section 4
Build a Dashboard with Plotly Dash

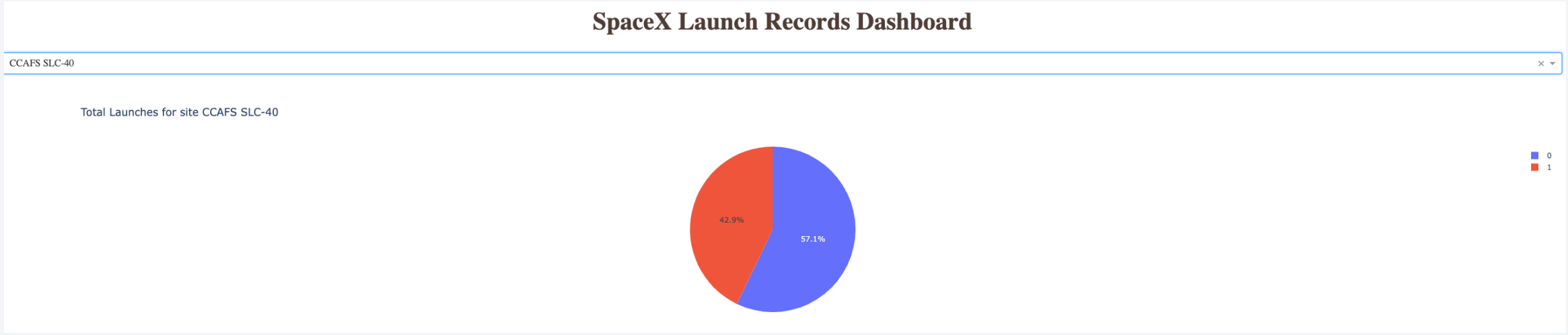
Pie Chart of Success Rate by Launch Site

- CCAFS SLC-40 is the launch site with the highest success rate of 41.7%.



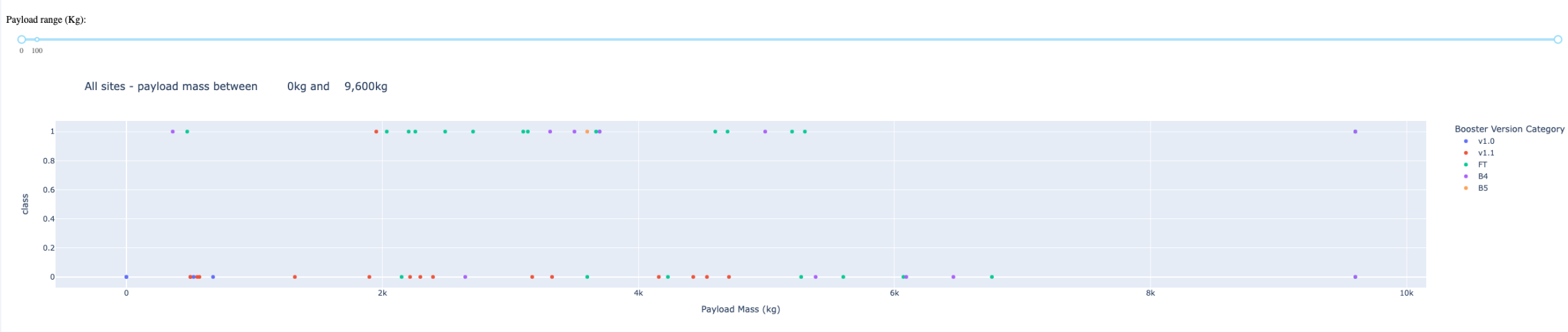
Analyze of The Launch Site with The Highest Rate

- CCAFS SLC-40, the launch site with the highest success rate, 42.9% of the time has been successful and the 57.1% had failures.



Analyze of Success Rate by Booster Version

- FT most of the time has been successful than other Booster Versions independently of its Payload Mass.

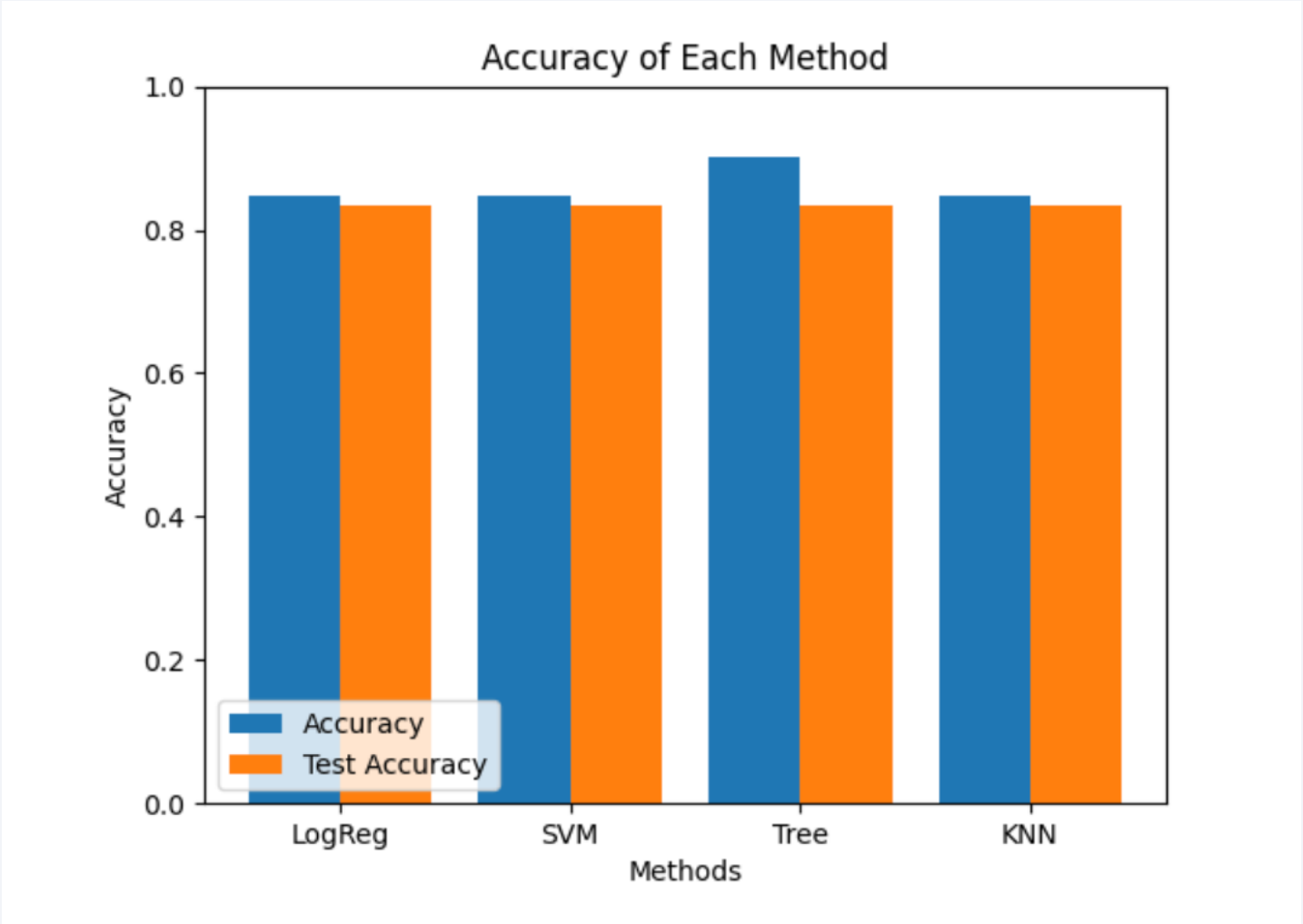


Section 5

Predictive Analysis (Classification)

Classification Accuracy

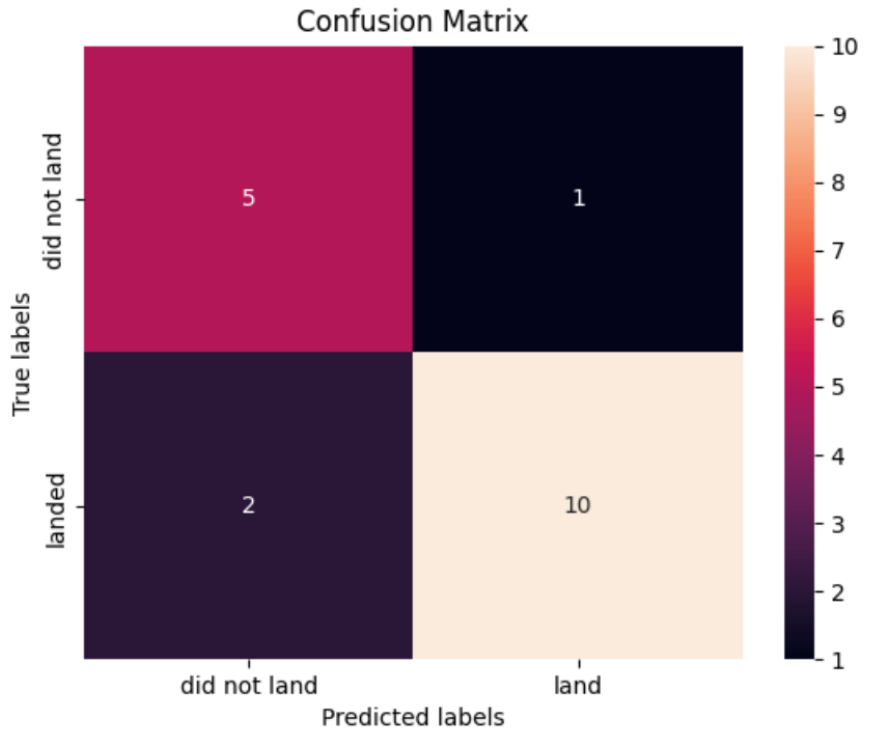
- Tree Predictive Model has the highest accuracy of all.



Confusion Matrix

- This model has the highest TP and TN rate of all while a low FP and FN.

```
In [86]: yhat = tree_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- CCAFS SLC-40 has the highest rate of landing success.
- In order to predict future success landing, Decision Tree Classifier is the best model.
- Trough the years, SpaceX has improved its success rate.
- FT Booster Version most of the time has had the highest success rate.
- SpaceX`s launch sites are near of coastlines.
- Currently, there is no a company with the technology SpaceX has.

Appendix

- When tasks were hard to solve, I looked for information in google, being Stack Overflow a good resource to study some examples similar to the tasks.
- Reviewing past modules videos was useful to refresh concepts.
- Solving all hand-labs, even optional, was the best way to practice for this final project.
- Reading python documentation and blogs about data science was useful to solve and understand some tasks.

Thank you!