IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Teccahoni
2023_Sep_11

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection using SpaceX API
  - EDA, including data wrangling, data visualization and interactive visual analytics
  - Machine Learning Prediction
- Summary of all results
  - able to collect valuable data from open sources.
  - EDA allowed to have a glimpse of the data, and the best approach to extract the expected result
  - Machine Learnng prediction showed the bes model to predict which characteristics are important drive this opportunity by the best way, using all collected data

Introduction

- Project background and context
  - the objective is to evaluated the viability of the new company Space Y to compete with Space X.
- Problems you want to find answers
  - Which characteristics have a major role for the success of reusing the first stage of rockets.
  - what are the best conditions for the first stage to succeed

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Datat from Space X was obtained from 1 sources:
    - Space X API(https://api.spacexdata.com/v4/rockets)
- Perform data wrangling
  - examine the data, which data have null value, and how to manage it.
  - change the dataframe with string, char value to float value, for making prediction model with machine learning technique
- Perform exploratory data analysis (EDA) using visualization
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

Data Collection

- Describe how data sets were collected
  - source API : https://api.spacexdata.com/v4/
  -
- get the json file from the API, and transform to pandas dataframe.

Data Collection – SpaceX API

- The first dataset is collected using the SpaceX API, using get requests in Python. Then we normalize the json contents into a dataframe and then using functions and pandas we extract relevant information, clean the data, and export the cleaned data.

- GitHub URL of the notebook:
  https://github.com/concistency/SpaceX_Data_Analysis_Final_Project/blob/main/collecting%20the%20data.ipynb

Getting the data in a JSON Format using get request call to the API

Normalizing the data into a dataframe

Filtering and Cleaning the data for further analysis

Data Collection - Scraping

- I didnt perform scraping, cause it have the same result with API method

Place your flowchart of web scraping here

Data Wrangling

- Exploratory Data Analysis (EDA) was performed on the dataset before wrangling.
- These tasks were performed
  - replace the null value with mean value
  - Calculated the number of launches at each site.
  - Calculated the number and occurrence of each orbits on which each launch was aimed to.
  - Calculated the number and occurence of mission outcome per orbit type
- the landing outcome label was created from Outcome column, class 0 denoted failure and 1 denoted successful landing.
- Applied One hot Encoding to the features to prepare for data modelling
- GitHub URL of the notebook: https://github.com/concistency/SpaceX_Data_Analysis_Final_Project/blob/main/EDA%20with%20Data%20Visul.ipynb

EDA with Data Visualization

- Visualized the relationship between Flight Number and Launch Site, Payload and Launch Site, FlightNumber and Orbit type, Payload and Orbit type using scatterplots as we need to observe the trend and overall relationship between the variables.
- Visualized the success rate of each orbit type using barchart so that side by side comparison could be done
- Visualizing the launch success yearly trend using a line plot as it best depicts a trend, whether it is increasing or not.

- GitHub URL: https://github.com/concistency/SpaceX_Data_Analysis_Final_Project/blob/main/EDA%20with%20Data%20Visul.ipynb

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

Build an Interactive Map with Folium

- Folium Circles, Markers, Marker clusters, Mouseposition and polyline objects were added.
    - Circles: To add a highlighted circle area with a text label on a specific site.
    - Markers: To mark the site
    - Marker Clusters: To simplify the map as it contained any markers having the same coordinate
    - Mouse Position: to get the coordinates for the position the mouse points on the map.
    - Polyline: to draw a line from a site to the nearest coast, city, highway, etc.
- GitHub URL of the notebook : https://github.com/concistency/SpaceX_Data_Analysis_Final_Project/blob/main/interactive%20visual%20analytics%20with%20folium.ipynb

Build a Dashboard with Plotly Dash

- Built an interactive Dashboard using Plotly Dash having the following features:
- Added a dropdown for the site input, and plotted a pie chart to view the relative success and failures in launches for each site
- Added a Range Slider for the Payload range to view the scatterplot between Payload and Class, according to the input site and within the provided payload range.
- These plots and interactions allowed to quickly visualize and analyze the relation between payloads and launch sites, which helped to find the best site for launching Falcon 9
- GitHub URL of python file:https://github.com/concistency/SpaceX_Data_Analysis_Final_Project/blob/main/interactive%20visual%20analytics%20with%20plotly.ipynb

Predictive Analysis (Classification)

- Four classification models were compared: logistic regression, support vector machine, decision tree and k nearest neighbors.

Data preparation and standardization	Test of each model with combinations of hyperparameters	Comparison of results

- GitHub URL of python file:https://github.com/concistency/SpaceX_Data_Analysis_Final_Project/blob/main/machine%20learning.ipynb

Results

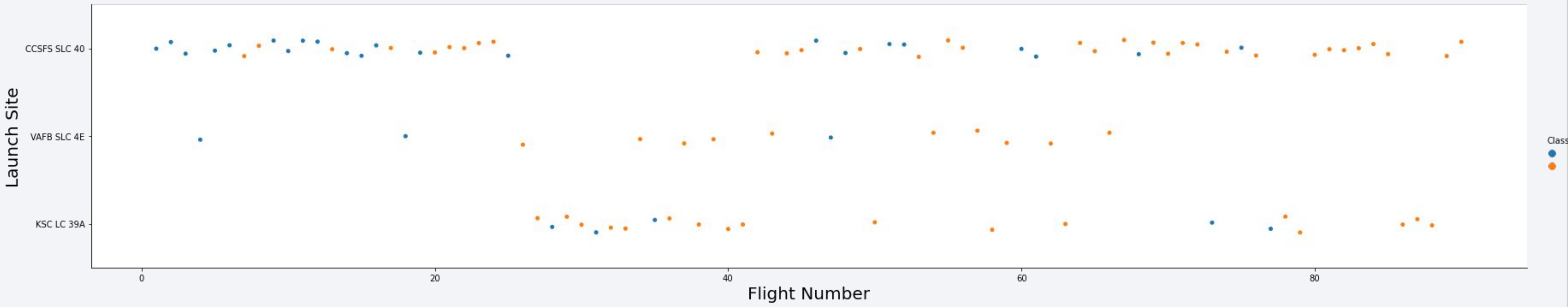
- Exploratory data analysis results
  - Space X uses 4 different launch sites;
  - The first launches were done to Space X itself and NASA;
  - The average payload of F9 v1.1 booster is 2,928 kg;
  - The first success landing outcome happened in 2015 fiver year after the first launch;
  - Almost 100% of mission outcomes were successful;
  - The number of landing outcomes became as better as years passed.

Section 2

Insights drawn from EDA

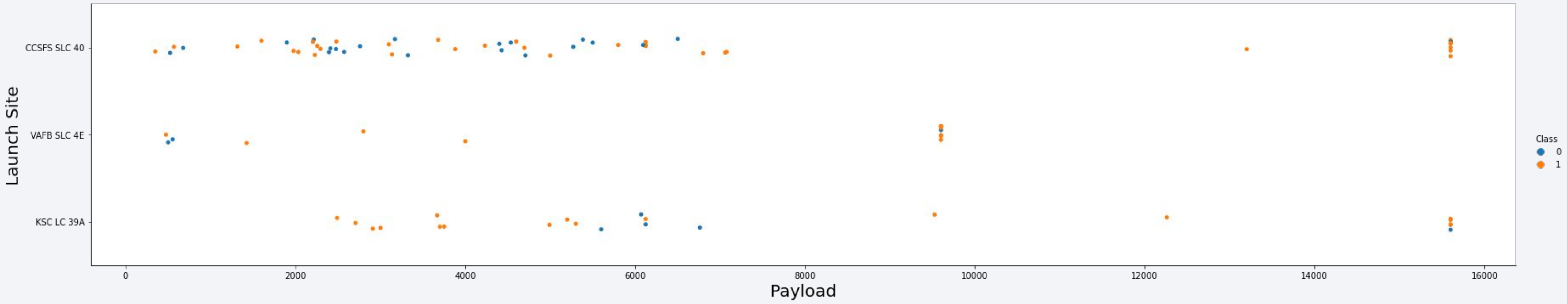
Flight Number vs. Launch Site

- CCSFS was most used for launches
- VAFB was abandoned after certain point
- KSC LC is another option for launches



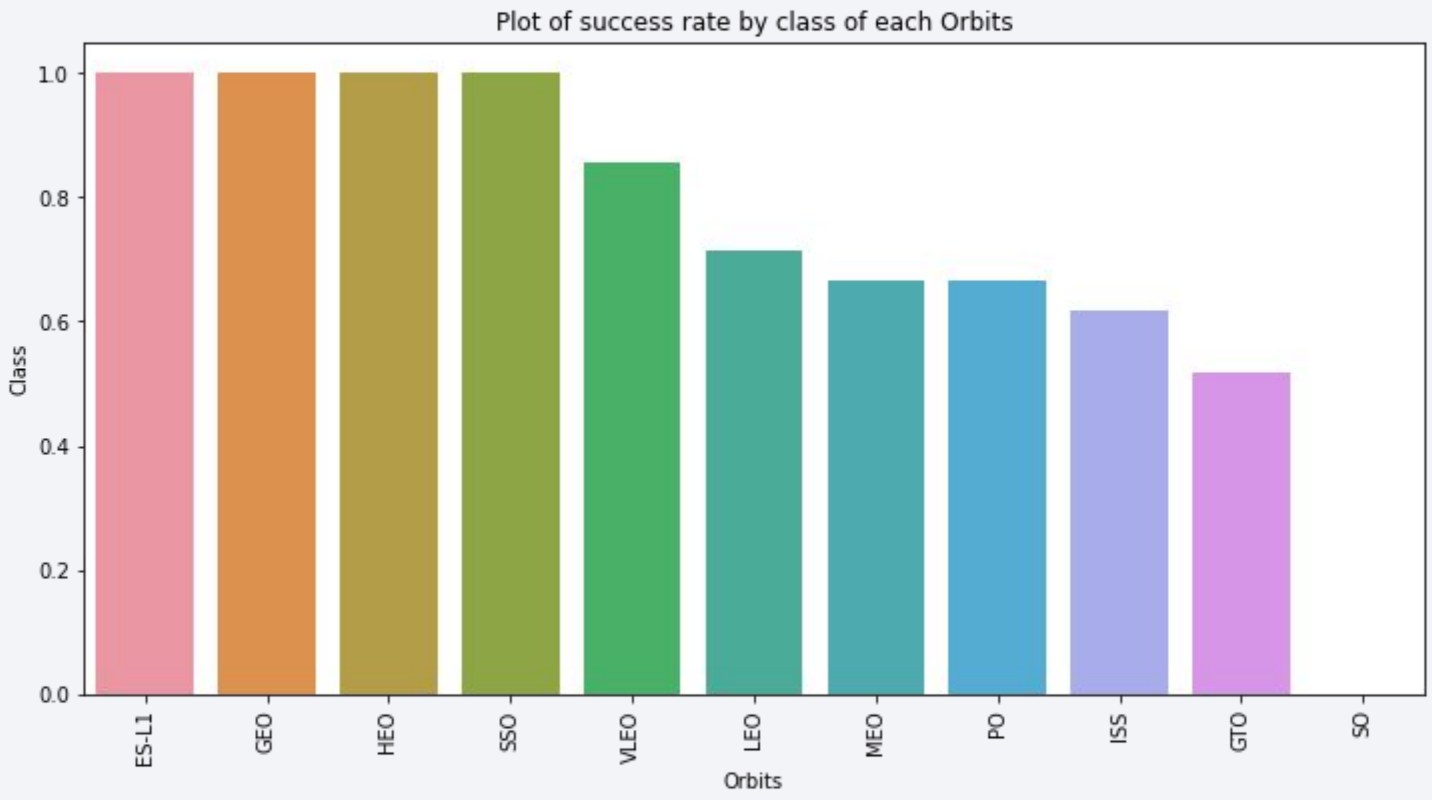
Payload vs. Launch Site

- CCSFS can be used for variable payloads
- KSC can be used for above 2000 payloads



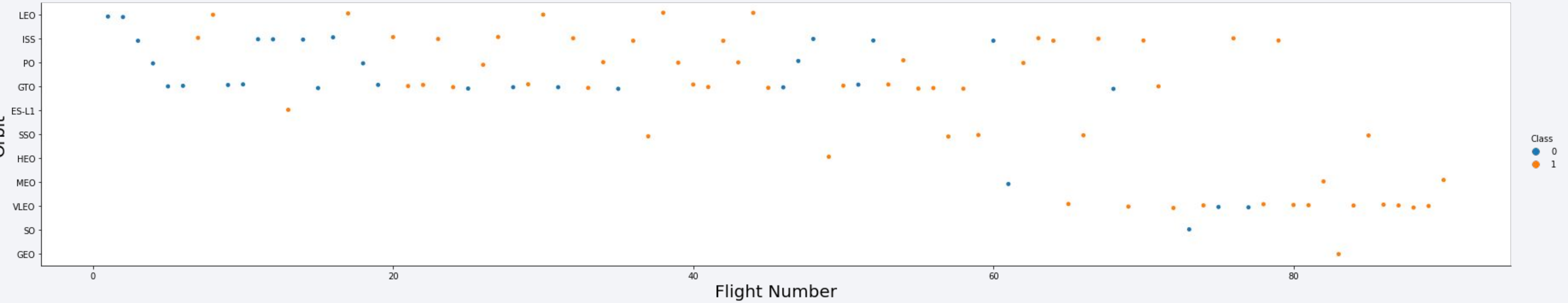
Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO have 100% success rates.
- VLEO has above 80% success rate
- LEO, MEO, PO, ISS have BTW 80% & 60% success rate
- SO never succeeded



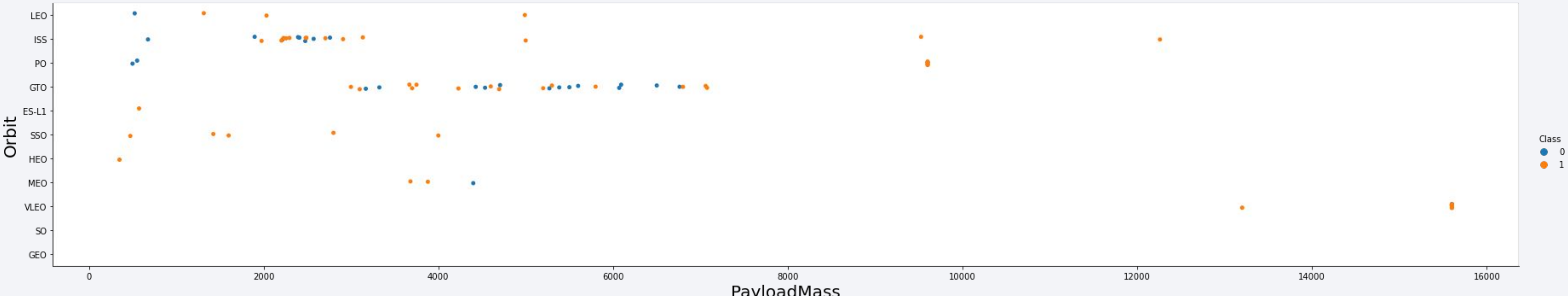
Flight Number vs. Orbit Type

- during early launches, mostly LEO, ISS, PO and GTO mission was executed
- after 30ish launches, the other mission, such as SSO, HEO, VLEO and ETC, were executed.
- recently space X focused on VLEO missions, for space X satellite internet service program



Payload vs. Orbit Type

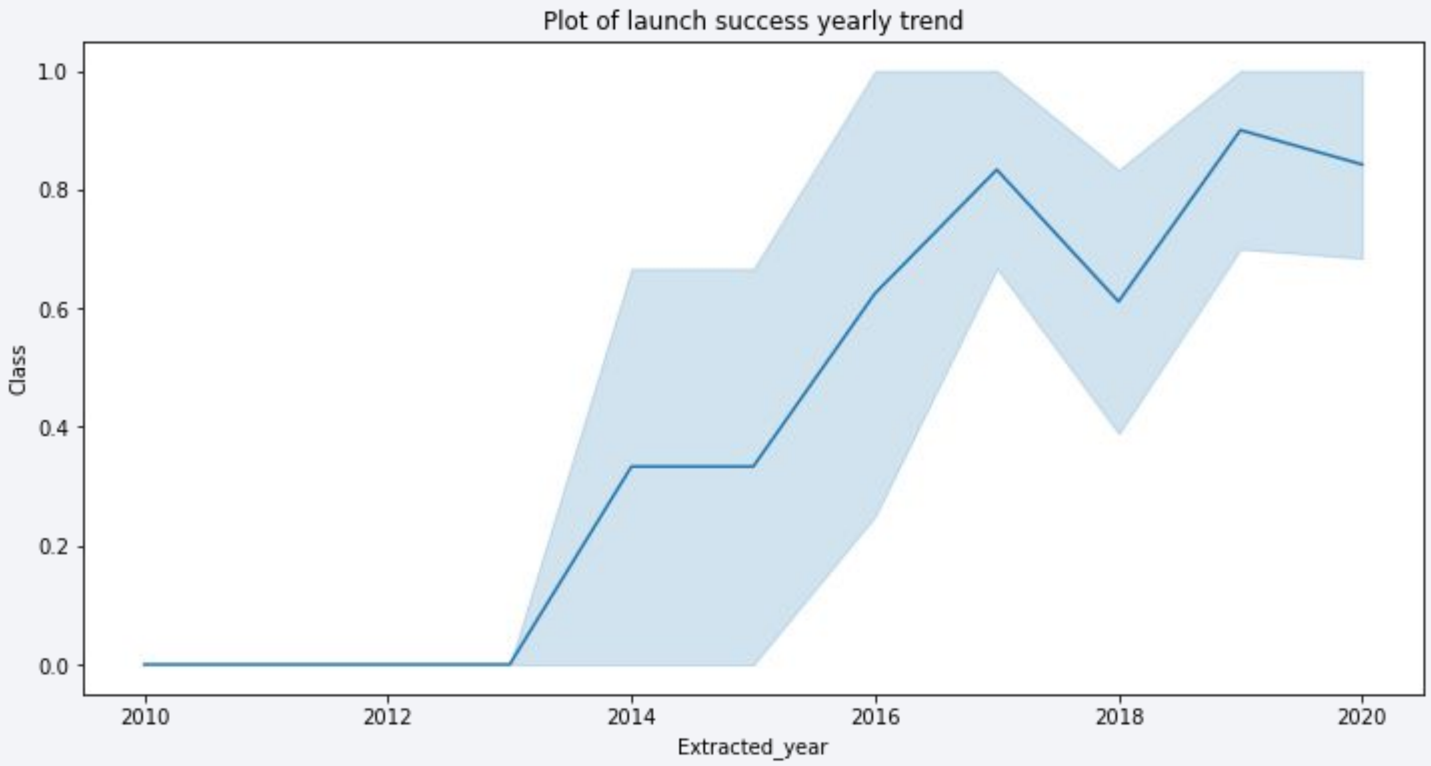
- ISS, PO and VLEO have the heaviest payloads.
- LEO, GTO, SSO have relatively lighter payloads.



Launch Success Yearly Trend

- Beginning from 2013, the success rate started to increace

- Now, success rate is above 80%



All Launch Site Names

- 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
-
-

	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	-120.610746

Launch Site Names Begin with 'CCA'

- spacex_df_duple[spacex_df_duple["Launch Site"].str.contains("CCA")].head(5)

	Flight Number	Date	Time (UTC)	Booster Version	Launch Site	Payload	Payload Mass (kg)	Orbit	Customer	Mission Outcome	Landing Outcome	class	Lat	Long
0	1	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (parachute)	0	28.562302	-80.577356
1	2	2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel o...	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)	0	28.562302	-80.577356
2	3	2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2+	525.0	LEO (ISS)	NASA (COTS)	Success	No attempt	0	28.562302	-80.577356
3	4	2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No attempt	0	28.562302	-80.577356
4	5	2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No attempt	0	28.562302	-80.577356

Total Payload Mass

```
In [37]: spacex_df_NASA = spacex_df_duple[spacex_df_duple["Customer"].str.contains("NASA")]
         spacex_df_NASA["Payload Mass (kg)"].sum()

Out[37]: 39157.0
```

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Present your query result with a short explanation here

```
In [38]: spacex_df_FC1_1 = spacex_df_duple[spacex_df_duple["Booster Version"].str.contains("v1.1")]

In [39]: spacex_df_FC1_1["Payload Mass (kg)"].mean()

Out[39]: 2534.6666666666665
```

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- Present your query result with a short explanation here

In [49]:
```
spacex_df_SG = spacex_df_duple[(spacex_df_duple["class"] == 1) & (spacex_df_duple["Landing Outcome"].str.contains("ground"))]
spacex_df_SG.head(1)
```

Out[49]:

	Flight Number	Date	Time (UTC)	Booster Version	Launch Site	Payload	Payload Mass (kg)	Orbit	Customer	Mission Outcome	Landing Outcome	class	Lat	Long
18	20	2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034.0	LEO	Orbcomm	Success	Success (ground pad)	1	28.562302	-80.577356

Successful Drone Ship Landing with Payload between 4000 and 6000

In [50]:

```
spacex_df_weight = spacex_df_duple[(spacex_df_duple["Payload Mass (kg)"] <6000) &
                                    (spacex_df_duple["Payload Mass (kg)"] >4000)]
```

In [61]:

```
list(set(spacex_df_weight["Booster Version"]))
```

Out[61]:

```
['F9 FT B1020',
 'F9 FT B1030',
 'F9 v1.1',
 'F9 v1.1 B1011',
 'F9 FT B1026',
 'F9 FT  B1032.2',
 'F9 FT B1022',
 'F9 B4  B1040.2',
 'F9 v1.1 B1014',
 'F9 FT  B1021.2',
 'F9 v1.1 B1016',
 'F9 B4 B1040.1',
 'F9 FT  B1031.2']
```

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- Present your query result with a short explanation here

```
In [66]: spacex_df_duple["class"].value_counts()

Out[66]: 0    32
         1    24
         Name: class, dtype: int64
```

Boosters Carried Maximum Payload

```
In [77]: spacex_df_duple[spacex_df_duple["Payload Mass (kg)"] == mini][["Booster Version", "Payload Mass (kg
```

Out[77]:

	Booster Version	Payload Mass (kg)
0	F9 v1.0 B0003	0.0
1	F9 v1.0 B0004	0.0

```
In [78]: spacex_df_duple[spacex_df_duple["Payload Mass (kg)"] == maxi][["Booster Version", "Payload Mass (kg
```

Out[78]:

	Booster Version	Payload Mass (kg)
28	F9 FT B1029.1	9600.0
29	F9 FT B1036.1	9600.0
31	F9 B4 B1041.1	9600.0
32	F9 FT B1036.2	9600.0
34	F9 B4 B1041.2	9600.0

2015 Launch Records

In [92]:
```
spacex_df_duple[(spacex_df_duple["Date"].str.contains("2015")) &
                (spacex_df_duple["Landing Outcome"].str.contains("Failure"))]
```

Out[92]:

	Flight Number	Date	Time (UTC)	Booster Version	Launch Site	Payload	Payload Mass (kg)	Orbit	Customer	Mission Outcome	Landing Outcome	class	Lat	
12	14	2015-01-10	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395.0	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)	0	28.562302	-8
15	17	2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898.0	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)	0	28.562302	-8

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

```
In [100]: spacex_df_duple['Date'] = pd.to_datetime(spacex_df_duple['Date'],format='%Y-%m-%d')
```

```
In [108]: spacex_df_duple[(spacex_df_duple['Date']>='2010-06-04') &
                          (spacex_df_duple['Date']<='2017-03-20')][["Landing Outcome"]].value_counts()
```

```
Out[108]: Landing Outcome
          No attempt              10
          Controlled (ocean)       3
          Failure  (drone ship)    3
          Success  (drone ship)    3
          Failure   (parachute)    2
          Failure (drone ship)     2
          Success (drone ship)     2
          Success (ground pad)     2
          Uncontrolled (ocean)     2
          Precluded  (drone ship)  1
          Success  (ground pad)    1
          dtype: int64
```

Section 3

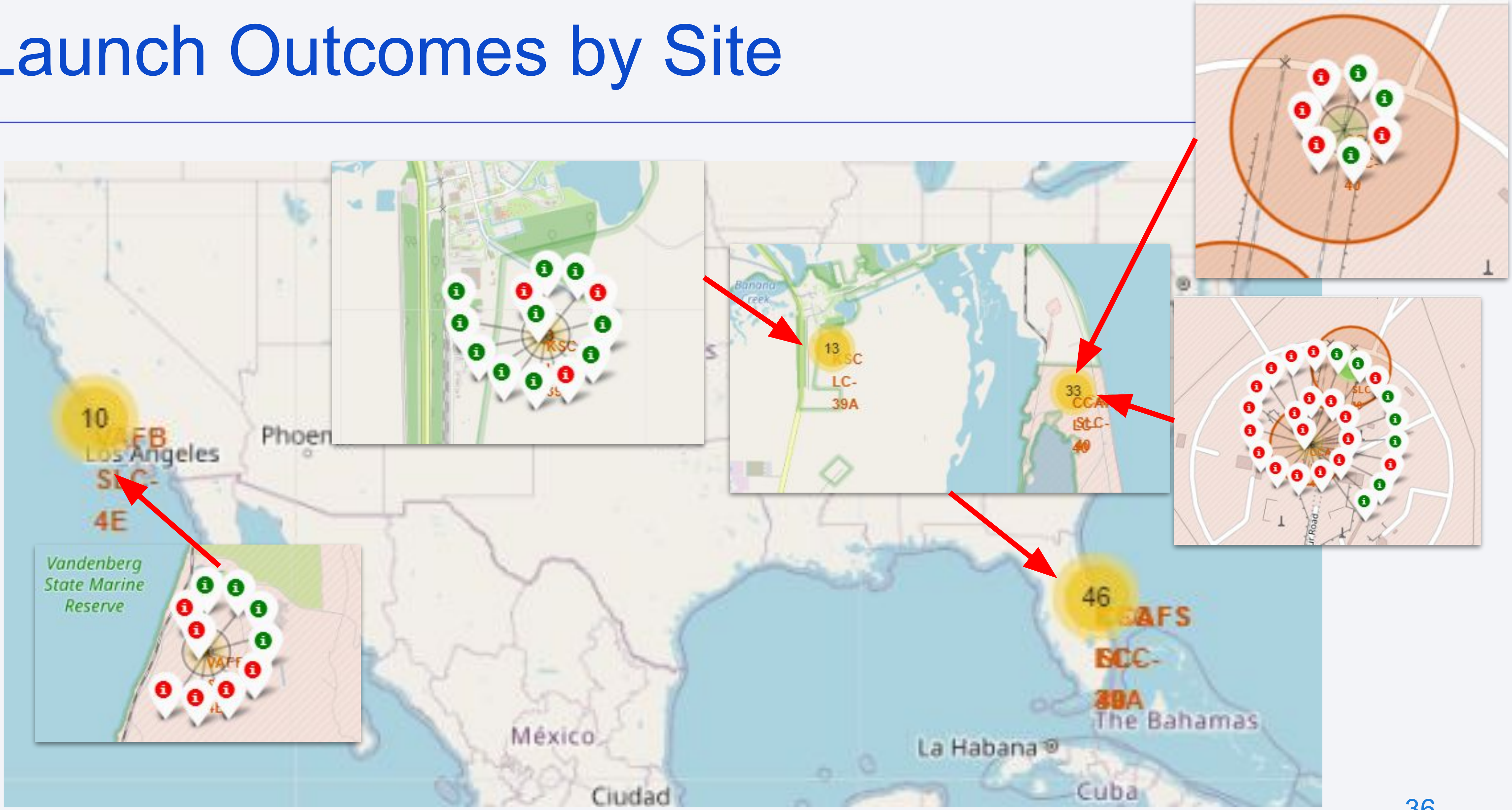
Launch Sites Proximities Analysis

All launch sites

1 site on west coast, 2 sites on east coast

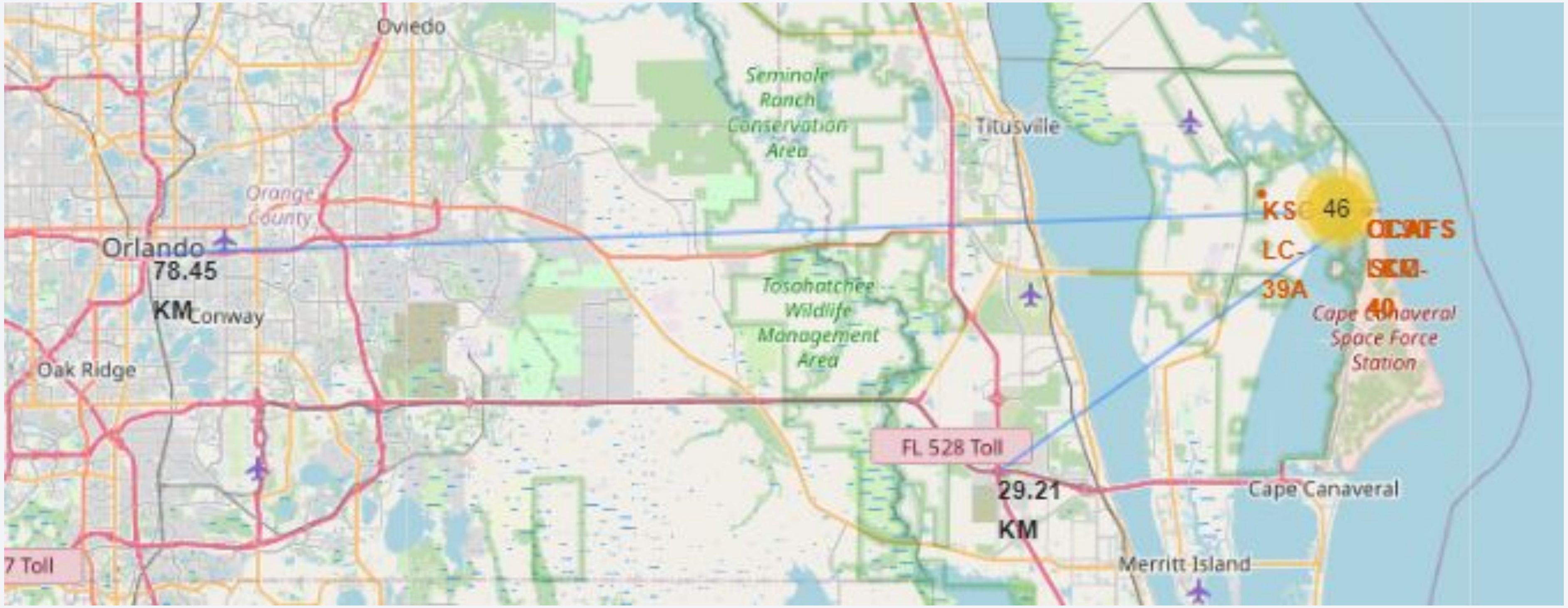
both near the sea, low latitude

Launch Outcomes by Site



Green = Success / Red = Failure

Logistics and Safety

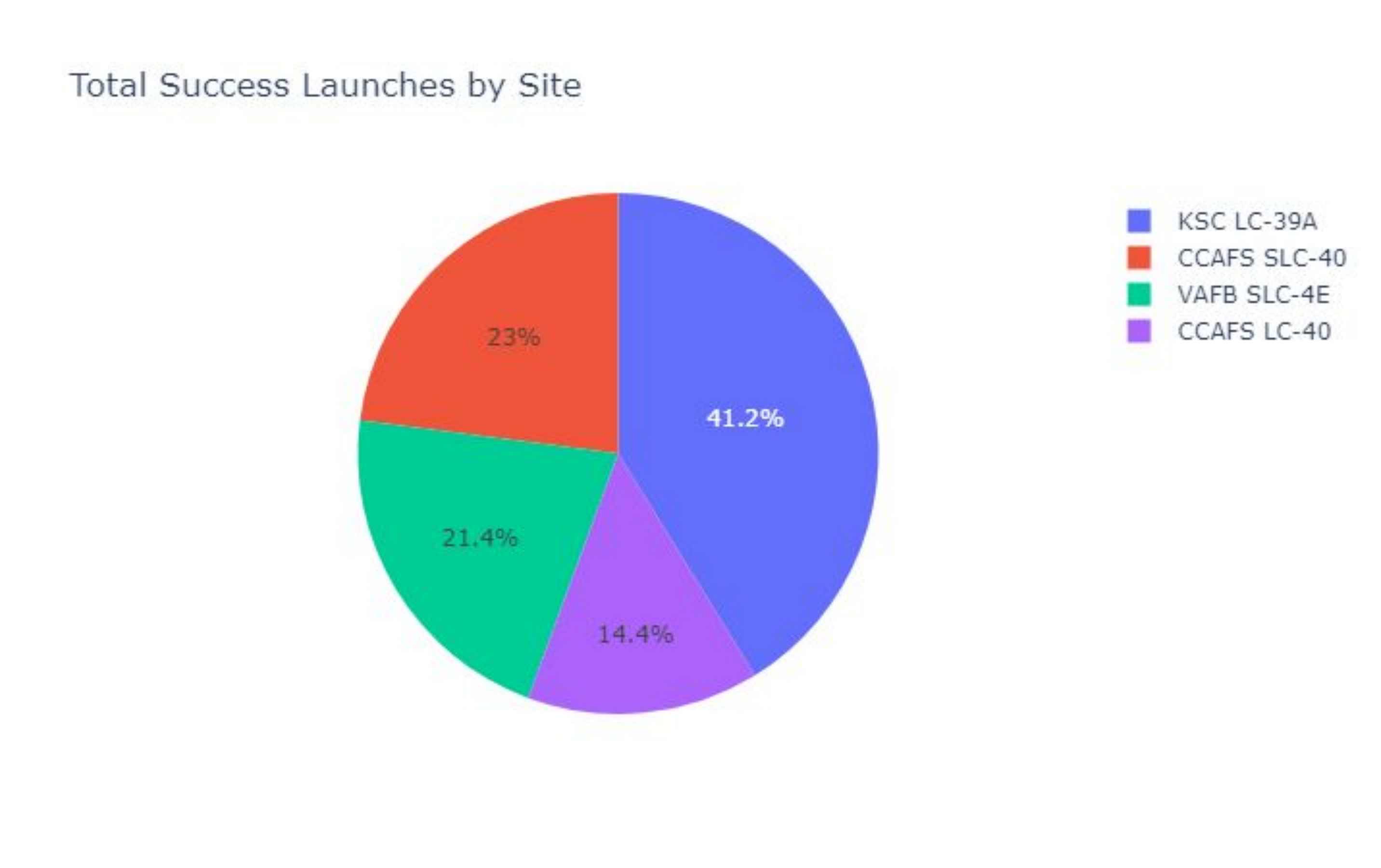


- East Coust Launch sites have good logistics aspects, being near railroad and road and relatively far from inhabited areas

Section 4

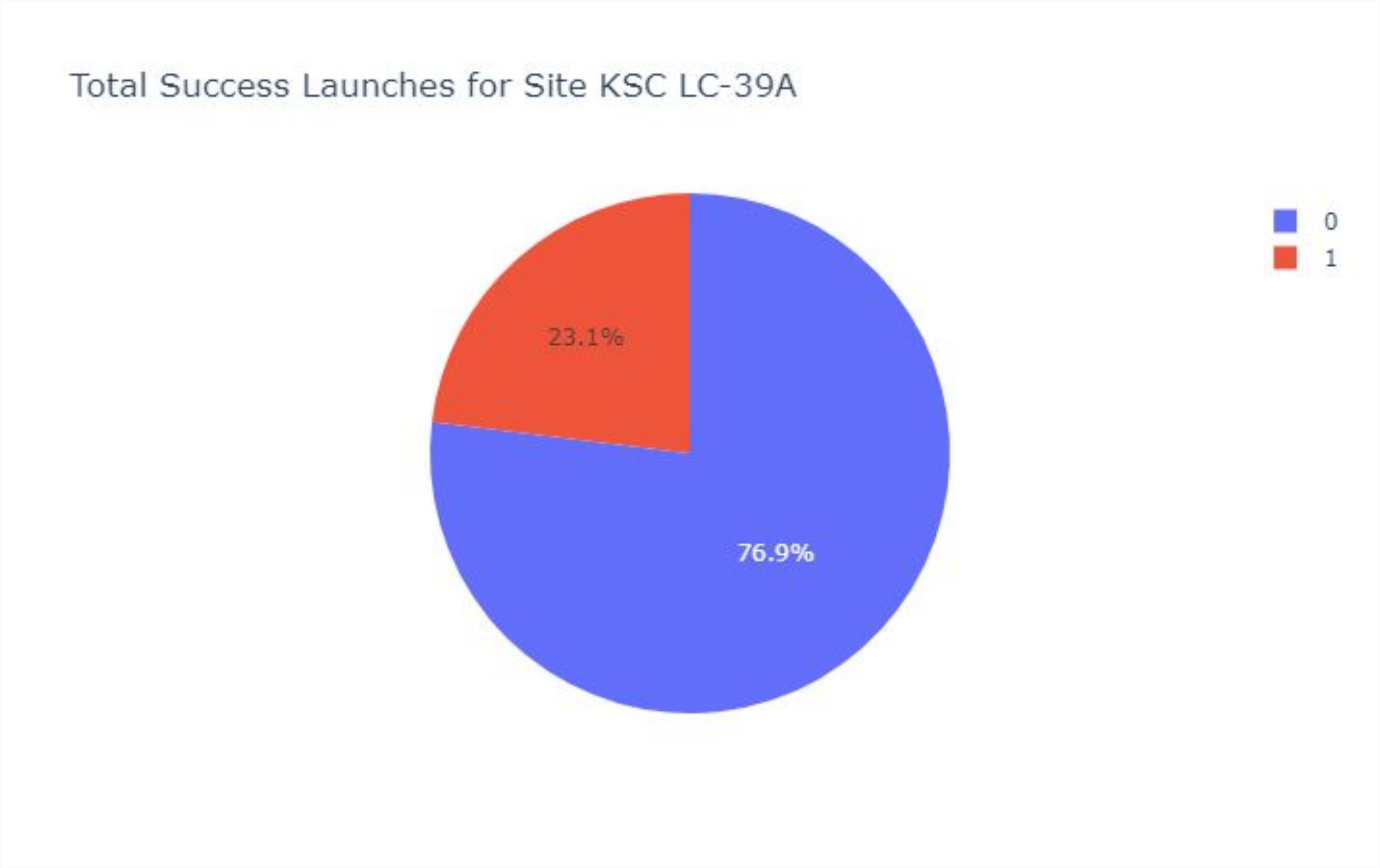
Build a Dashboard with Plotly Dash

Successful Launches by Site



This is not enough info yet, to decide wheter the launch site matters

Launch Success Ratio for KSC LC-39A



- KSC LC-39A has the highest success rate.

Payload vs. Launch Outcome

- Above 6k payload, the success rate decrease significantly.

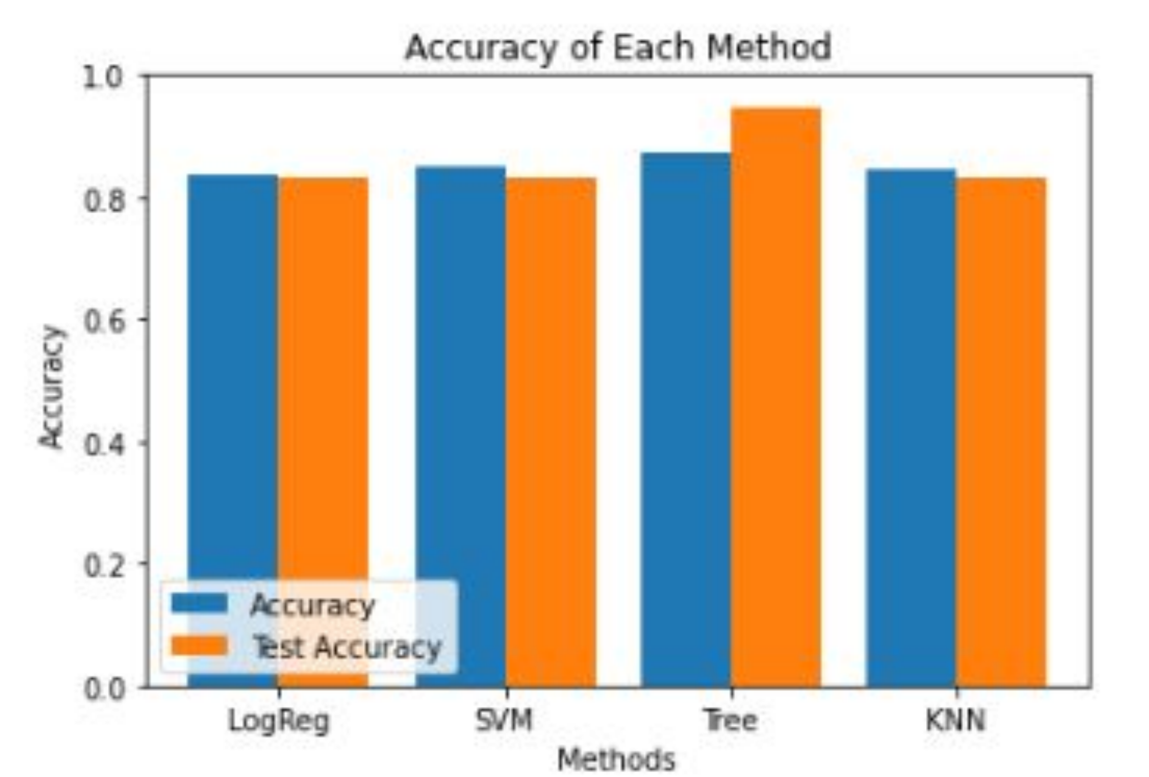
Section 5

Predictive Analysis (Classification)

Classification Accuracy

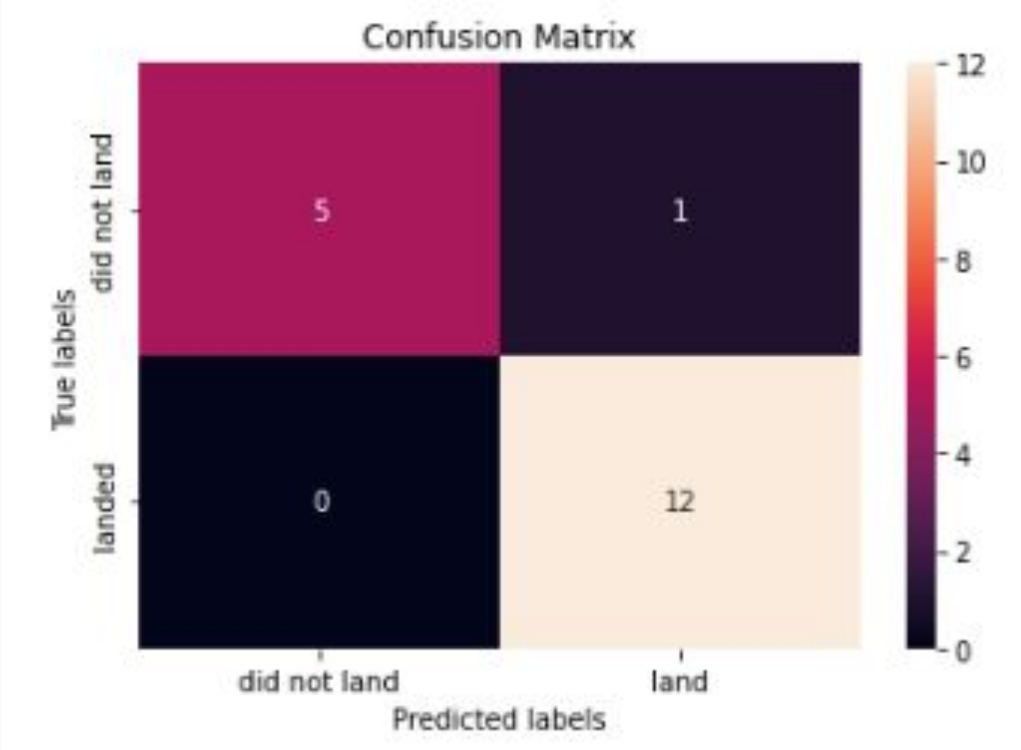
- Decision Tree Classification has the highest Accuracy which over than 87%
-

Model	**Accuracy**	**TestAccuracy**
LogReg	0.83393	0.83333
SVM	0.85	0.83333
Tree	0.87321	0.94444
KNN	0.84643	0.83333



Confusion Matrix of Decision Tree

- True Positive of Decision Tree : 12
- True Negative of Decision Tree : 5



Conclusions

- The best launch site is KSC LC-39A
- Launches under 7,000kg are safer
- Successful landing outcomes seem to improve over time
- Decision Tree Model can be the optimum classification method to predict the success of launches

Appendix

- Nothing.

Thank you!