PREDICTING SUCCESS IN FALCON 9 LANDING

PHUMLANI NGCOBO

IBM Data Science Capstone Project

17 August 2021

OUTLINE

- Executive Summary
- Introduction
- Methodology
  - Data Collection and Data Wrangling (Visualization)
  - EDA and Interactive Visual Analytics (Folium)
  - Machine Learning Predictive Analysis
- Results
  - Data Collection and Data Wrangling results (Visualization)
  - EDA and Interactive Visual Analytics results (Folium)
  - Machine Learning Predictive Analysis results
  - Dashboard with Plotly Dash results
  - EDA with SQL results
- Discussion
  - Findings & Implications
- Conclusion
- Appendix

EXECUTIVE SUMMARY

I have spent a decade working as an Integration Specialist for the IBM BPM, automating and managing business processes. A large chunk of our BPM work was suspended because some of our clients were cutting costs as a result of the pandemic and lockdown.

With some of my free time working from home, I decided to pursue a career on what some call “the sexiest job in the 21st century” – Data Science. In this data-driven world, data scientists have emerged as a hot commodity. So, I enrolled in this IBM Data Science Professional course by Coursera to understand the fundamentals of data science and learn how to practically use various tools and methods such as Jupyter Notebooks, IBM Watson Studio, CRISP-DM and Python with its diverse libraries, such as pandas, numpy, matplatlib, seaborn, folium, scikit learn etc.

With the knowledge acquired in this course, I intend to restart my career as a Data Scientist to help organizations who have a need for professionals who understand a business need, can devise a data-oriented solution, and then implement that solution.

In this presentation, we will go through the many tools and methods learned throughout the course and apply them into the challenge of predicting if Falcon 9 first stage will land successful. 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.

INTRODUCTION

- SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; 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.
- The goal that I want to reach in this presentation is to provide an overview of the problem, method and tools and recommendation to a company that wants to bid against SpaceX

METHODOLOGY

This section describes the data collection, data exploration & visualization and the machine learning algorithms that were conducted and how they relate to what we are trying to achieve which is to predict the Falcon 9 successful landing.

In summary, the following is what was done in this section:

- Data was extracted and cleaned using data wrangling & formatting.
- Performed exploratory data analysis to determine training labels. Feature selection was made
- Maps to mark launch sites and success/failed launches were plotted. Distances between launch sites to their proximities were calculated using Maps
- Interactive dashboards related to launch sites were plotted
- Classification algorithms were used to enhance the performance of machine learning predictions

METHODOLOGY ...continues

Data Collection & Data Wrangling

To carry out this project, we used the following datasets from 3 sources:

- To obtain information about SpaceX launches on SpaceX API:
    - https://api.spacexdata.com/v4/launches/past)
    - https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json

- To extract Falcon 9 specific launch records from Wikipedia:
    - https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

- For Folium maps construction, a file of launch sites was used:
    - https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_geo.csv

METHODOLOGY ...continues

Once the data was arranged, we required a lot of cleaning and wrangling to remove data that's not required and narrowed it down to the specific data then read it through pandas Dataframe that looks like the following:

In [12]:
```
# Get the head of the dataframe
data.head()
```

Out[12]:

	static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	crew	ships	capsules	
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]	[5eb0
1	None	NaN	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine shutdown at T+7 min 30 s, Failed to reach orbit, Failed to recover first stage	[]	[]	[]	[5eb0
2	None	NaN	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Residual stage 1 thrust led to collision between stage 1 and stage 2	[]	[]	[]	[5eb0 5eb0e
								Ratsat was carried to orbit on the first				

METHODOLOGY *...continues*

We cleaned data by using a series of helper functions that helped us use the API to extract information from launch data using the IDs given for each launch, then performed data wrangling to deal with missing values, removed rows with multiple ‘cores’, converted ‘date’, etc.

We also filtered the Dataframe to focus on Falcon 9 launches, checked/dealt with missing values in our dataset. Resulting data looked like below:

Out[32]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedC
4	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0
5	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0
6	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0
7	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0
8	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0

METHODOLOGY ...continues

On Falcon 9 specific launch records from Wikipedia, we found a table of "Past launches" which shows flight numbers, launch sites, payload mass, launch outcomes, etc. then extracted information from Wikipedia and used web scraping:

en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

gcwetiTech SBZA Perso Readir

iii. ^ Controlled descent; soft vertical ocean touchdown; no recovery

Past launches

2010 to 2013

[hide] Flight No.	Date and time (UTC)	Version, Booster [b]	Launch site	Payload[c]	Payload mass	Orbit	Customer	Launch outcome	Booster landing
1	4 June 2010, 18:45	F9 v1.0[7] B0003.1[8]	CCAFS, SLC-40	Dragon Spacecraft Qualification Unit		LEO	SpaceX	Success	Failure[9][10] (parachute)
	First flight of Falcon 9 v1.0.[11] Used a boilerplate version of Dragon capsule which was not designed to separate from the second stage.(more details below) Attempted to recover the first stage by parachuting it into the ocean, but it burned up on reentry, before the parachutes even deployed.[12]								
2	8 December 2010, 15:43[13]	F9 v1.0[7] B0004.1[8]	CCAFS, SLC-40	Dragon demo flight C1 (Dragon C101)		LEO (ISS)	NASA (COTS) NRO	Success[9]	Failure[9][14] (parachute)
	Maiden flight of Dragon capsule, consisting of over 3 hours of testing thruster maneuvering and reentry.[15] Attempted to recover the first stage by parachuting it into the ocean, but it disintegrated upon reentry, before the parachutes were deployed.[12] (more details below) It also included two CubeSats,[16] and a wheel of Brouère cheese.								
3	22 May 2012, 07:44[17]	F9 v1.0[7] B0005.1[8]	CCAFS, SLC-40	Dragon demo flight C2+[18] (Dragon C102)	525 kg (1,157 lb)[19]	LEO (ISS)	NASA (COTS)	Success[20]	No attempt
	Dragon spacecraft demonstrated a series of tests before it was allowed to approach the International Space Station. Two days later, it became the first commercial spacecraft to board the ISS.[17] (more details below)								
4	8 October 2012, 00:35[21]	F9 v1.0[7] B0006.1[8]	CCAFS, SLC-40	SpaceX CRS-1[22] (Dragon C103)	4,700 kg (10,400 lb)	LEO (ISS)	NASA (CRS)	Success	No attempt
				Orbcomm-OG2[23]	172 kg (379 lb)[24]	LEO	Orbcomm	Partial failure[25]	
	CRS-1 was successful, but the secondary payload was inserted into an abnormally low orbit and subsequently lost. This was due to one of the nine Merlin engines shutting down during the launch,								

Out[139]:

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	121	CCSFS	SXM-8	7,000 kg	GTO	SpaceX	Success\n	F9 B5	Success	6 June 2021	04:26
1	121	CCSFS	SXM-8	7,000 kg	GTO	SpaceX	Success\n	F9 B5	Success	6 June 2021	04:26
2	121	CCSFS	SXM-8	7,000 kg	GTO	SpaceX	Success\n	F9 B5	Success	6 June 2021	04:26
3	121	CCSFS	SXM-8	7,000 kg	GTO	SpaceX	Success\n	F9 B5	Success	6 June 2021	04:26
4	121	CCSFS	SXM-8	7,000 kg	GTO	SpaceX	Success\n	F9 B5	Success	6 June 2021	04:26

METHODOLOGY ...continues

EDA

Here, we continue to explore data by performing Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models. In our given dataset, there are several different cases where the booster did not land successfully.

We will mainly convert those outcomes into Training Labels with 1 means the booster successfully landed, 0 means it was unsuccessful.

Firstly, we calculated the number of launches on each site:

```
In [6]: # Apply value_counts() on column LaunchSite
        df['LaunchSite'].value_counts()

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

METHODOLOGY ...continues

Secondly, we calculated the number of occurrence of each Orbit in the dataset:

```
In [7]: # Apply value_counts on Orbit column
        df['Orbit'].value_counts()

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

METHODOLOGY ...continues

Thirdly, we calculated the number of occurrence of mission outcome per orbit type:

```
In [9]: # landing_outcomes = values on Outcome column
        landing_outcomes = df.Outcome.value_counts()
        landing_outcomes

Out[9]: 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: Outcome, dtype: int64
```

METHODOLOGY ...continues

Fourthly, we created a landing outcome label from Outcome column:

In [12]:
```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class=[]
for outcome in enumerate(df['Outcome']):
        if outcome==bad_outcomes:
            landing_class.append(0)
        else:
            landing_class.append(1)
landing_class
```

Out[12]:
```
[1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
 1,
```

gitude	Latitude	Class
7366	28.561857	1
7366	28.561857	1
7366	28.561857	1
10829	34.632093	1
7366	28.561857	1

METHODOLOGY ...continues

EDA and Interactive Visual Analytics (Visualization)

Exploratory data analysis (EDA) is only a key to understand and represent data in a better way which in result helps us to build a powerful and more generalized model. Data visualization is easy to perform EDA which makes it easy to make others understand our analysis.

We used EDA using pandas, matplotlib & seaborn for performing various techniques to explore data using various plots

The results section of EDA and Interactive Visual Analytics display the results of this section.

METHODOLOGY ...continues

EDA and Interactive Visual Analytics (Folium)

As we will see in the results section, exploratory data analysis (EDA) to visualize the SpaceX launch dataset using matplotlib & seaborn were used and discovered some preliminary correlations between the launch site and success rates. We'll see that the launch success rate may depend on many factors such as payload mass, orbit type, and so on.

It may also depend on the location and proximities of a launch site, i.e., the initial position of rocket trajectories. Finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations.

As the results will show us, I will be performing more interactive visual analytics using Folium. **Folium** is a powerful library that uses Python to produce interactive maps using OpenStreetMap technology. I have used the benefits of Folium and our data to plot launch sites on a map using the location data from a file of launch sites that was provided.

METHODOLOGY *...continues*

Machine Learning Predictive Analysis (classification)

Before we start using machine learning algorithms (ML) on our data, we'll need to do some normalizing on it. Normalizing or Standardizing data is the first step that we'll perform before applying ML algorithms. The goal here is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values.

There are a few columns that we will be standardize, so it would not affect negatively our ML algorithms. We start by normalizing the predictor. We also train the model using 80% of the whole data that we have and other reminded 20% for testing.

After training the model, we'll be able to predict the success and failure rates of Falcon 9 landing but first, we'll test the accuracy for our model.

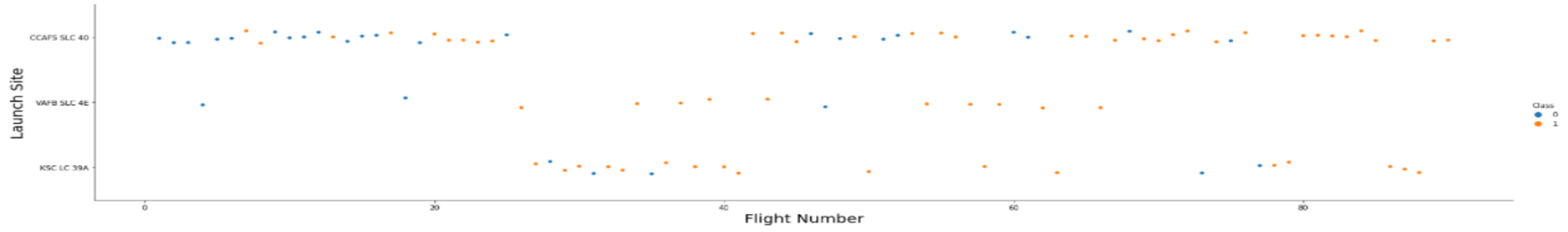
We will look at the 3 models here: **logistic regression, SVM, and Classic Trees**. We will see how the models performed and which one performed better when we go down the results section

RESULTS

EDA and Interactive Visual Analytics (Visualization) results

Here we used EDA using pandas, matplotlib & seaborn for performing various techniques to explore data using various plots. First, we visualized the relationship between Flight Number vs. Launch Site then plotted out the outcome of the launch:

```
In [49]: # Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
         sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
         plt.xlabel("Flight Number",fontsize=20)
         plt.ylabel("Launch Site",fontsize=20)
         plt.show()
```

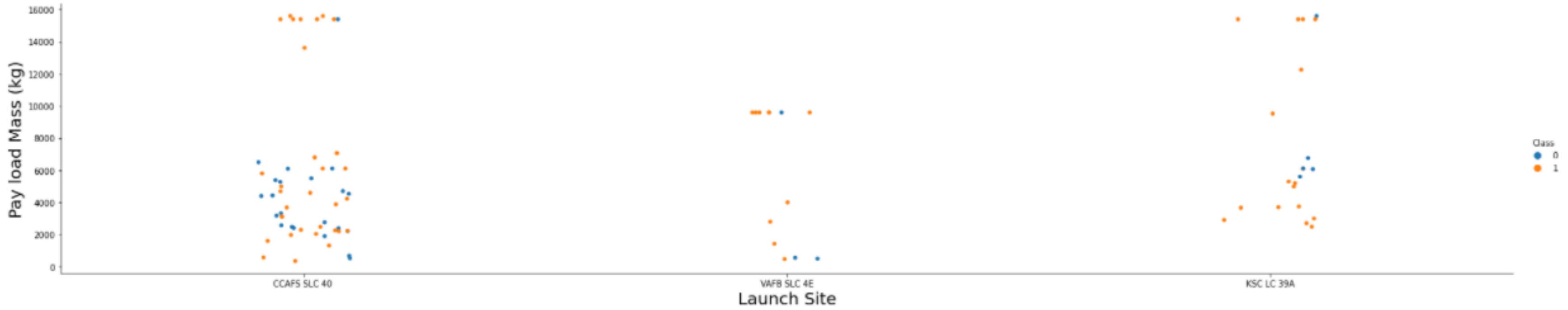


In the plot, we see that different launch sites have different success rates as number of flights increase. CCAFS LC-40, has a success rate of 80 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 30%.

RESULTS

Next, we used visualization to observe if there is any relationship between Launch Site and their Payload Mass:

In [50]:
```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class valu
e
sns.catplot(y="PayloadMass", x="LaunchSite", hue="Class", data=df, aspect = 5)
plt.xlabel("Launch Site",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

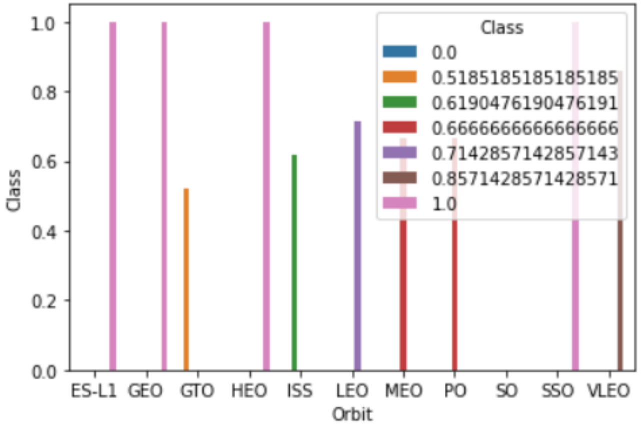


Here we see different launch sites on different payload mass. It seems the less massive the payload, CCAFS LC-40 has a success rate of 80 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 20%.

RESULTS

Next, we wanted to visually check if there are any relationships between success rate and orbit type

```
Out[51]: <AxesSubplot:xlabel='Orbit', ylabel='Class'>
```

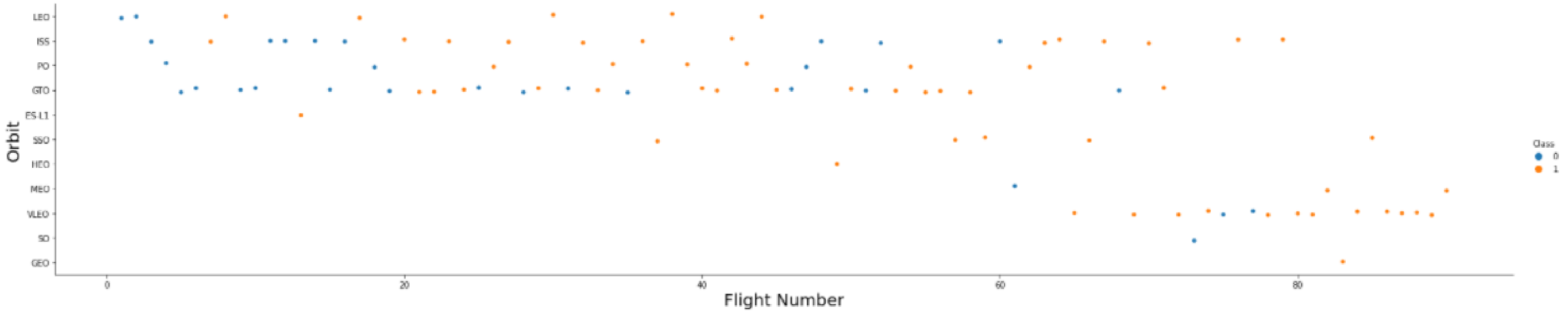


On this bar chart, we see that ES-L1, GOE, HEO, PO, SSO and VLEO orbits have the high success rate

RESULTS

For each orbit, we also wanted to see if there is any relationship between Flight Number and Orbit type:

In [9]:
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

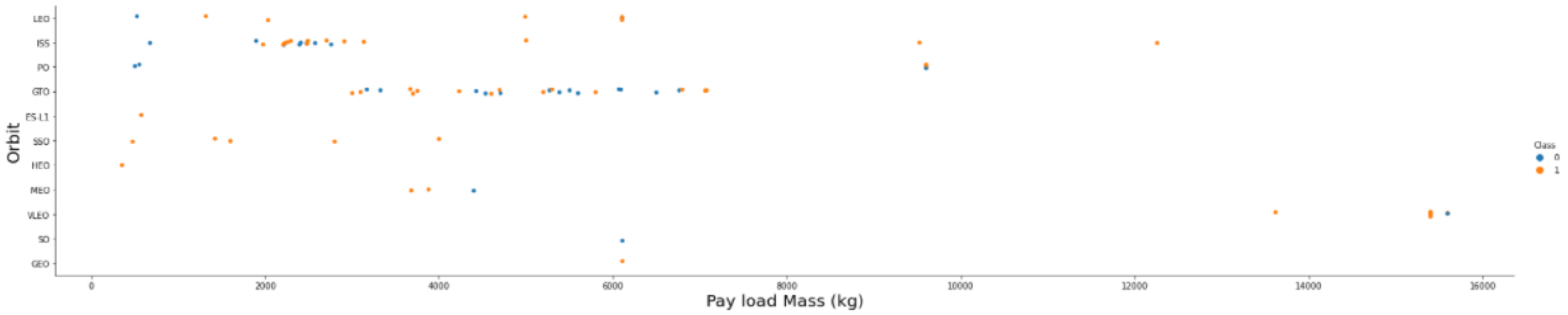


You should see that in the LEO orbit, the success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

RESULTS

Similarly, we plotted the Payload vs Orbit scatter to reveal the relationship between Payload and Orbit type:

In [10]:
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("Pay load Mass (kg)",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```

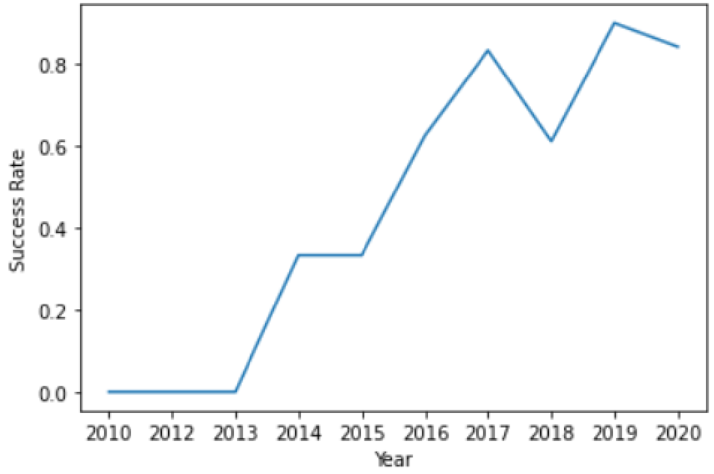


You should observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

RESULTS

We then visualized the launch success yearly trend and plotted a line chart to get the average launch trend:

In [57]:
```
# 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 Rate")
plt.show()
```



you can observe that the sucess rate since 2013 kept increasing till 2020

RESULTS

By now, we have obtained some preliminary insights about how each important variable would affect the success rate, therefore we selected the features that will be used in success prediction.

To start with, we created dummy variables to categorical columns to apply OneHotEncoder to column *Orbit, LaunchSite, LandingPad, Serial*:

In [57]:
```
# HINT: Use get_dummies() function on the categorical columns
features_one_hot = pd.get_dummies(features, columns=['Orbit', 'LaunchSite', 'LandingPad', 'Serial'])
features_one_hot.head()
```

Out[57]:

	FlightNumber	PayloadMass	Flights	GridFins	Reused	Legs	Block	ReusedCount	Orbit_ES-L1	Orbit_GEO	...	Serial_B1048	Serial_B1049	Serial_B1050	Serial_B1051	Serial_B1054	Serial_B1056	Serial_B1058	Ser
0	1	6104.959412	1	False	False	False	1.0	0	0	0	...	0	0	0	0	0	0	0	
1	2	525.000000	1	False	False	False	1.0	0	0	0	...	0	0	0	0	0	0	0	
2	3	677.000000	1	False	False	False	1.0	0	0	0	...	0	0	0	0	0	0	0	
3	4	500.000000	1	False	False	False	1.0	0	0	0	...	0	0	0	0	0	0	0	
4	5	3170.000000	1	False	False	False	1.0	0	0	0	...	0	0	0	0	0	0	0	

5 rows × 80 columns

RESULTS

Then casted all numeric columns to float64

```
Out[59]: FlightNumber    float64
         PayloadMass     float64
         Flights         float64
         GridFins        float64
         Reused          float64
                          ...
         Serial_B1056    float64
         Serial_B1058    float64
         Serial_B1059    float64
         Serial_B1060    float64
         Serial_B1062    float64
         Length: 80, dtype: object
```

RESULTS

EDA and Interactive Visual Analytics (Folium) results

From a downloaded dataset, we read the dateset into a pandas dataframe and yielded the following dataframe:

Out[93]:

	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

RESULTS

We then looked at the coordinates of each site as this is the data that we'll use to mark all launch sites:

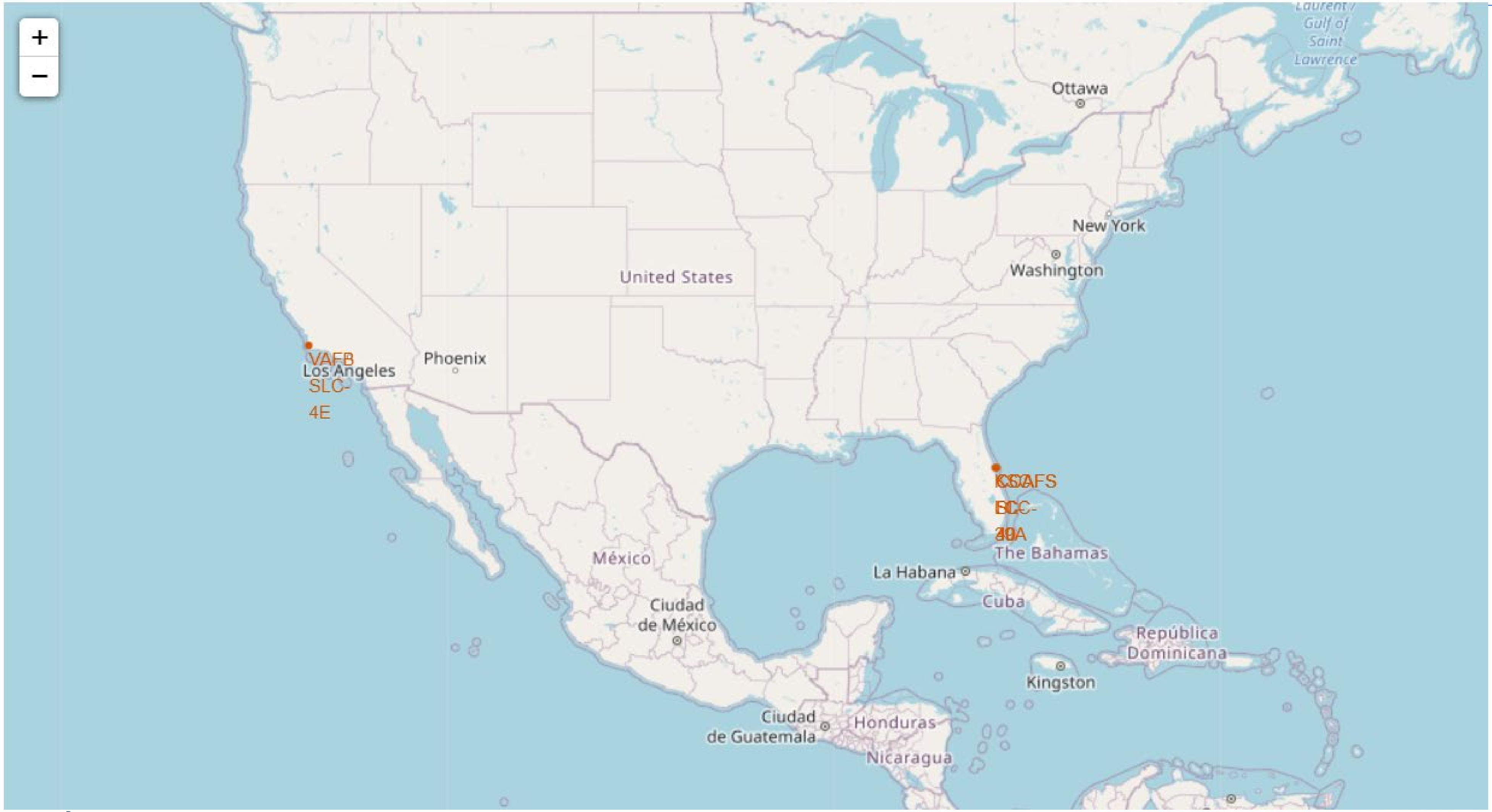
Out[95]:

	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

Then visualized these locations by pinning them on a Folium map as shown below:

RESULTS

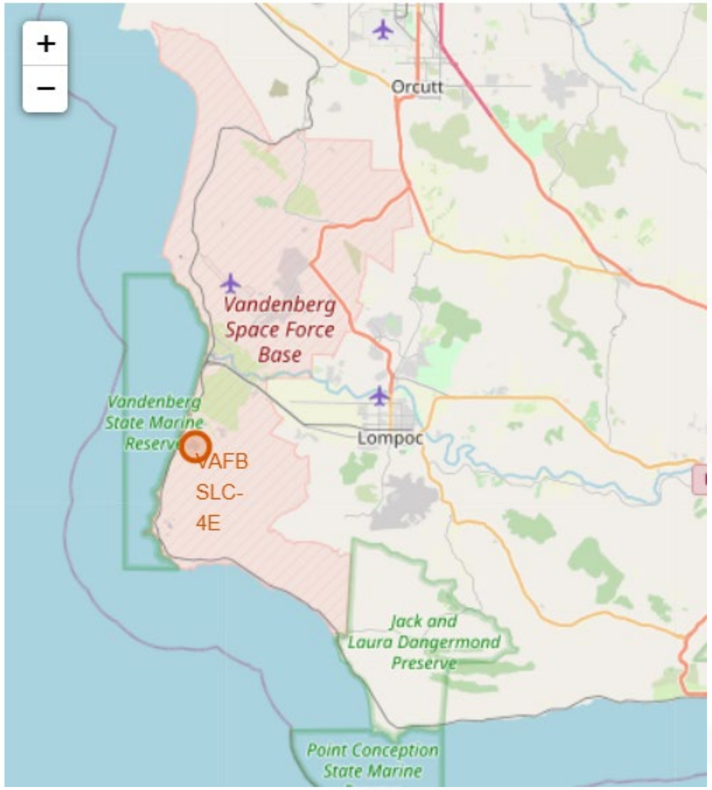
Out[97]:

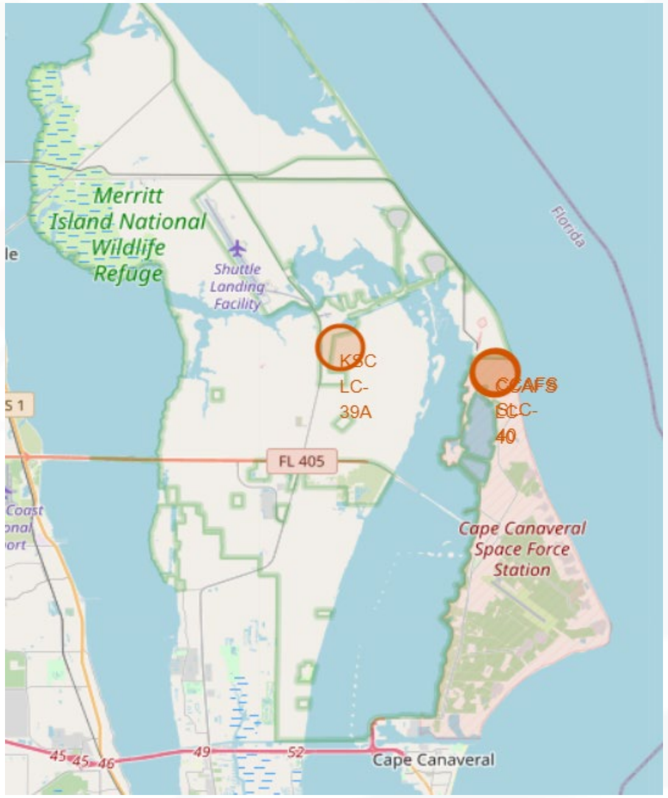


RESULTS

When we zoomed to the marked sites on the above map, we see *VAF SLC-4E* on the left and *KSC LC-39A, CCAFS LC-40 and CCAFS SLC-40* on the far-right sort of close to each other :

Out[97]:





IBM Developer

SKILLS NETWORK

RESULTS

Next, we tried to enhance the above map by adding the launch outcomes for each site, and see which sites have high success rates. Using our dataset as shown below (*class* column indicates if this launch was successful or not):

Out[98]:

	Launch Site	Lat	Long	class
46	KSC LC-39A	28.573255	-80.646895	1
47	KSC LC-39A	28.573255	-80.646895	1
48	KSC LC-39A	28.573255	-80.646895	1
49	CCAFS SLC-40	28.563197	-80.576820	1
50	CCAFS SLC-40	28.563197	-80.576820	1
51	CCAFS SLC-40	28.563197	-80.576820	0
52	CCAFS SLC-40	28.563197	-80.576820	0
53	CCAFS SLC-40	28.563197	-80.576820	0
54	CCAFS SLC-40	28.563197	-80.576820	1
55	CCAFS SLC-40	28.563197	-80.576820	0

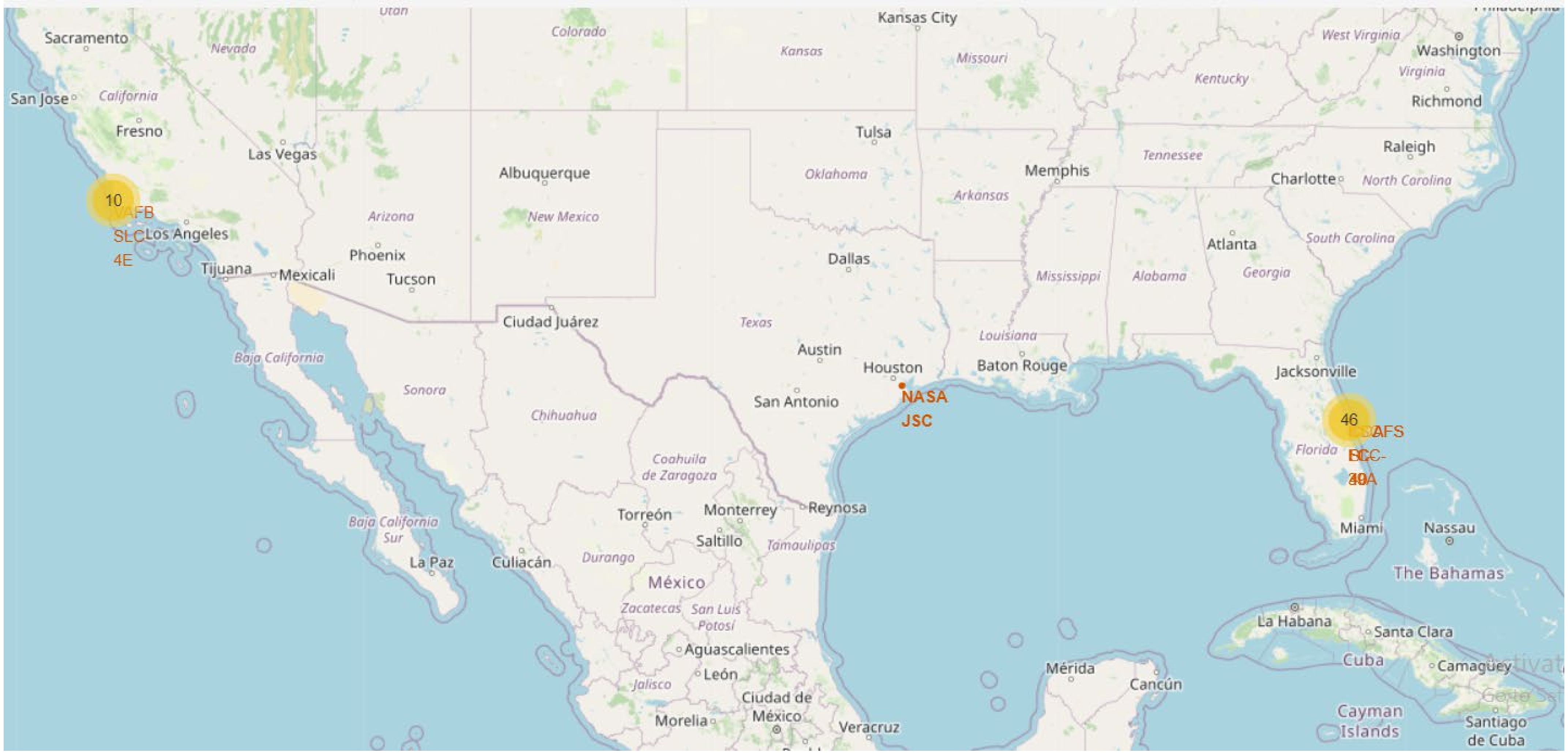
RESULTS

We also created markers for all launch records, i.e if a launch was successful (*class=1*), then we use a green marker and if a launch was failed, we use a red marker (*class=0*). We then added a new column in our dataframe called *marker_color* to store the marker colors based on the class value:

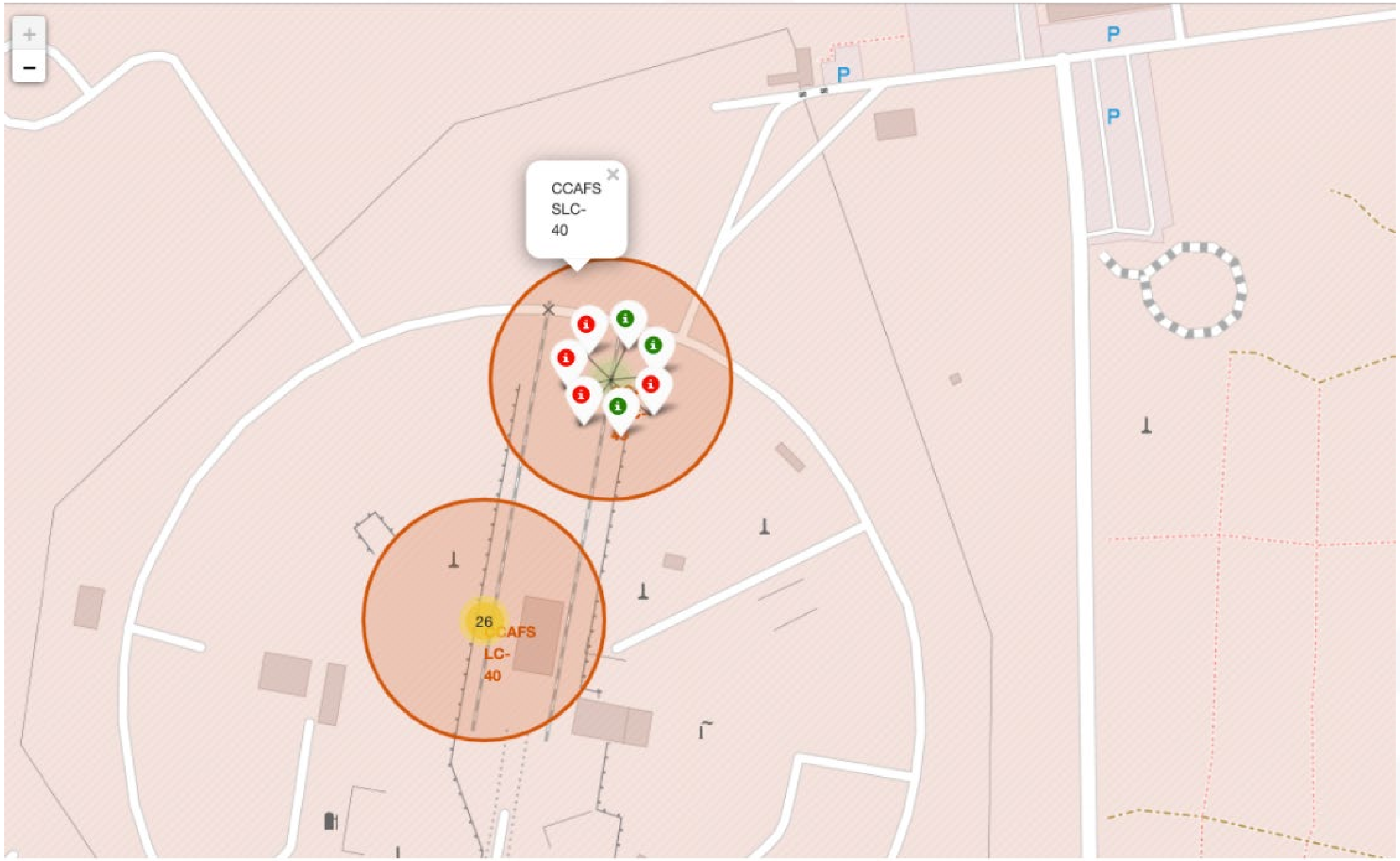
Out[101]:

	Launch Site	Lat	Long	class	marker_color
47	KSC LC-39A	28.573255	-80.646895	1	green
48	KSC LC-39A	28.573255	-80.646895	1	green
49	CCAFS SLC-40	28.563197	-80.576820	1	green
50	CCAFS SLC-40	28.563197	-80.576820	1	green
51	CCAFS SLC-40	28.563197	-80.576820	0	red
52	CCAFS SLC-40	28.563197	-80.576820	0	red
53	CCAFS SLC-40	28.563197	-80.576820	0	red
54	CCAFS SLC-40	28.563197	-80.576820	1	green
55	CCAFS SLC-40	28.563197	-80.576820	0	red

RESULTS



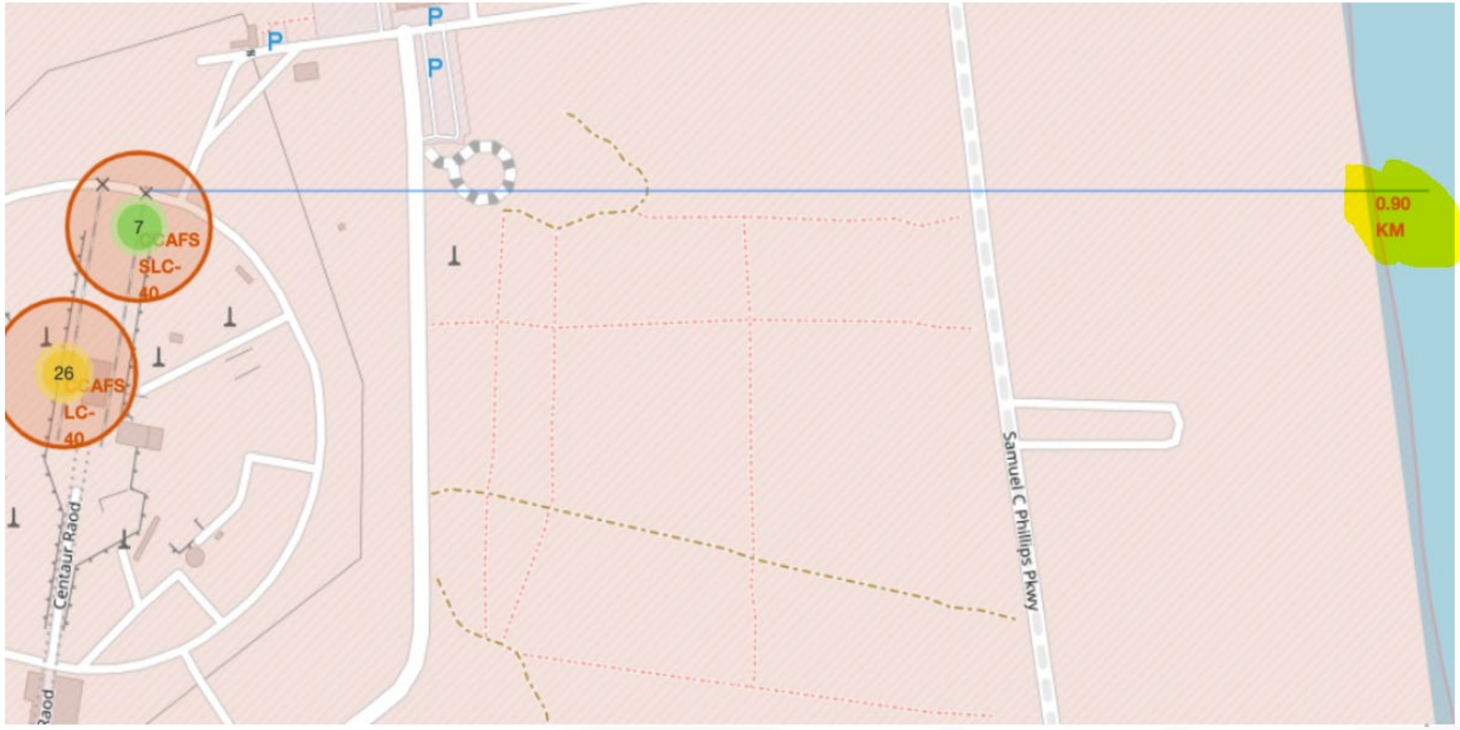
RESULTS



From the color-labeled markers in marker clusters, you should be able to easily identify which launch sites have relatively high success rates.

RESULTS

As I said earlier, finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations. Finally, we calculate the distances between a launch site to it's proximity and a map looks like below:



After plotting distance lines to the proximities, we could see that launch sites are not in close proximity to railways, highways and coastline. We saw that launch sites do keep certain distance away from cities

RESULTS

Machine Learning Predictive Analysis (classification) results

As mentioned in the methodology section of the ML predictive analysis (classification), we started by normalizing the predictor as got the following results:

```
In [6]: X = preprocessing.StandardScaler().fit(X).transform(X)
        X

Out[6]: 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]])
```

RESULTS

For model training, I then splitted data it into training data and test data to find the best hyperparameter for SVM, Classification by Trees, and Logistic Regression:

```
Train set: (72, 83) (72,)
Test set: (18, 83) (18,)
```

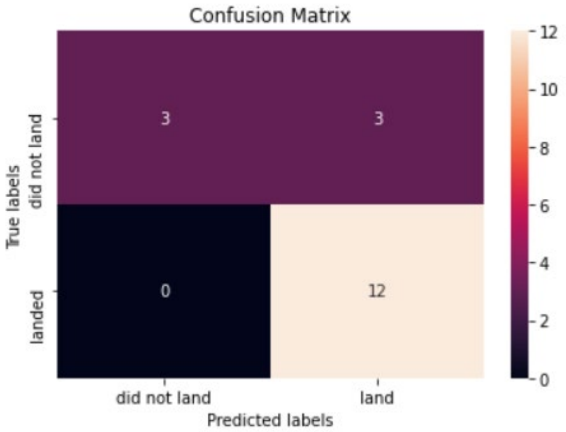
From there, I created ***Logistic Regression*** object using GridSearchCV to get tuned hyperparameters and the accuracy on the validation data:

```
GridSearchCV(cv=10, estimator=LogisticRegression(),
             param_grid={'C': [0.01, 0.1, 1], 'penalty': ['l2'],
                         'solver': ['lbfgs']})
```

```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

```
In [13]: lr_score = logreg_cv.score(X_test, Y_test
         lr_score

Out[13]: 0.8333333333333334
```



RESULTS

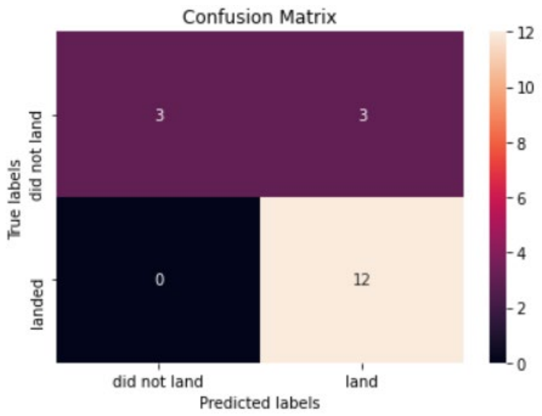
We also calculated the tuned hyperparameters and the accuracy on the validation data for **Support Vector Machine (SVM):**

```
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')})
```

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

```
In [18]: svm_score = svm_cv.score(X_test, Y_test)
         svm_score

Out[18]: 0.8333333333333334
```



RESULTS

Decision Tree Classifier

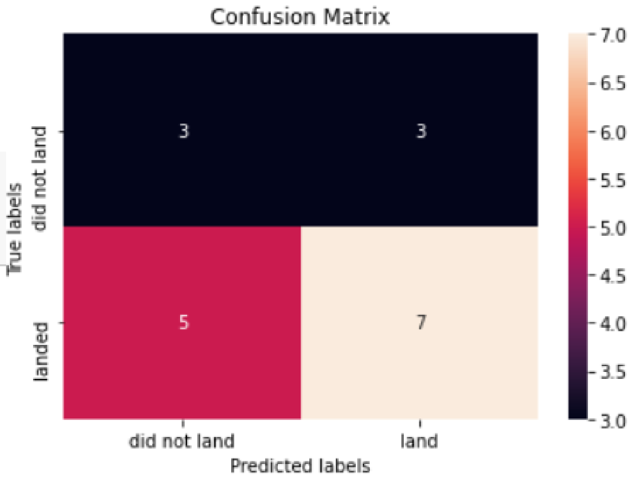
```
GridSearchCV(cv=10, estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [2, 4, 6, 8, 10, 12, 14, 16, 18],
                         'max_features': ['auto', 'sqrt'],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10],
                         'splitter': ['best', 'random']},
             scoring='accuracy')
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 4, 'max_features': 'auto', 'min_samples_leaf': 1,
'min_samples_split': 2, 'splitter': 'random'}
accuracy : 0.9
```

On the test data:

```
In [25]: tree_score = tree_cv.score(X_test,Y_test)
         tree_score

Out[25]: 0.5555555555555556
```



RESULTS

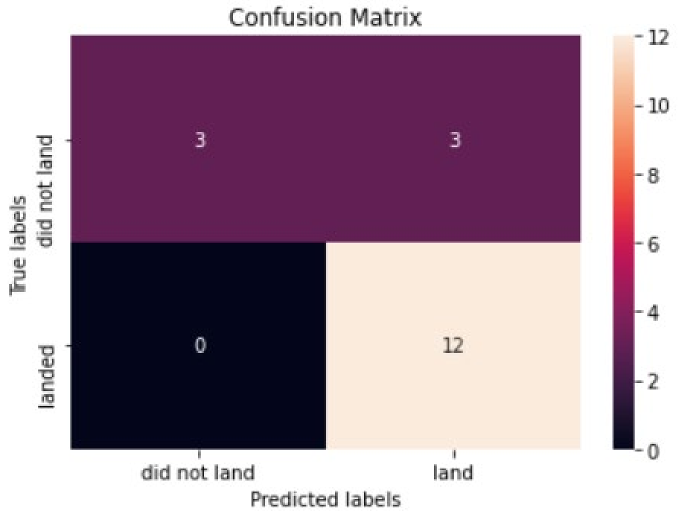
And **k Nearest Neighbors:**

```
GridSearchCV(cv=10, estimator=KNeighborsClassifier(),
             param_grid={'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
                         'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
                         'p': [1, 2]})
```

```
tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```

```
In [29]: knn_score = knn_cv.score(X_test,Y_test)
         knn_score

Out[29]: 0.8333333333333334
```



RESULTS

Finally, I compared the above models to find the one that performed best:

Out[34]:

	Model
Score	
0.833333	Logistic Regression
0.833333	Support Vector Machines
0.833333	Decision Tree
0.555556	KNN

RESULTS

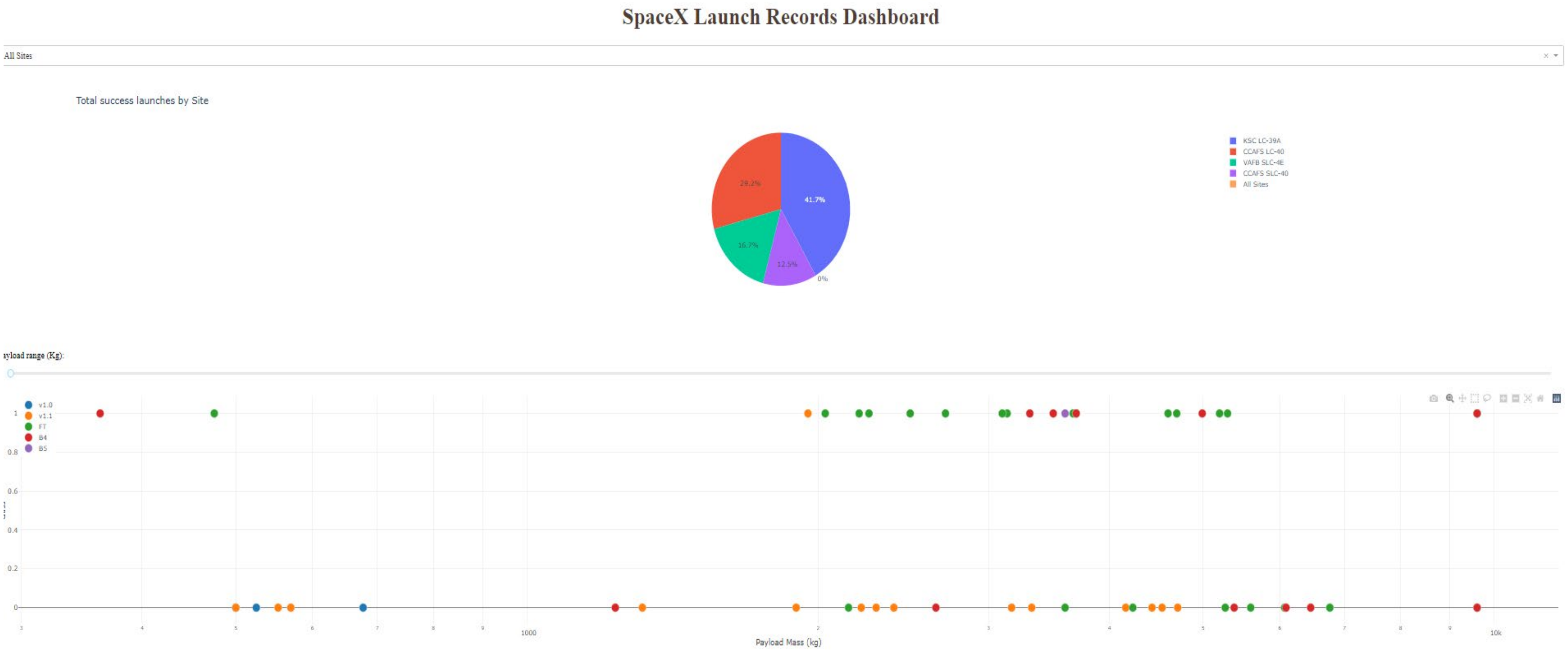
Dashboard with Plotly Dash results

Now that we've discovered many interesting insights related to the launch sites' location using Folium, in a very interactive way, we went on to build a dashboard using Ploty Dash on detailed launch records to perform interactive visual analytics on SpaceX launch data in real-time.

We built Dashboard using this given dataset:

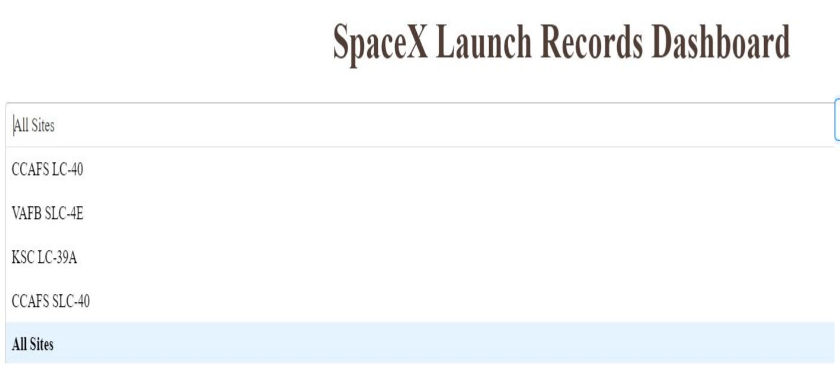
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_dash.csv

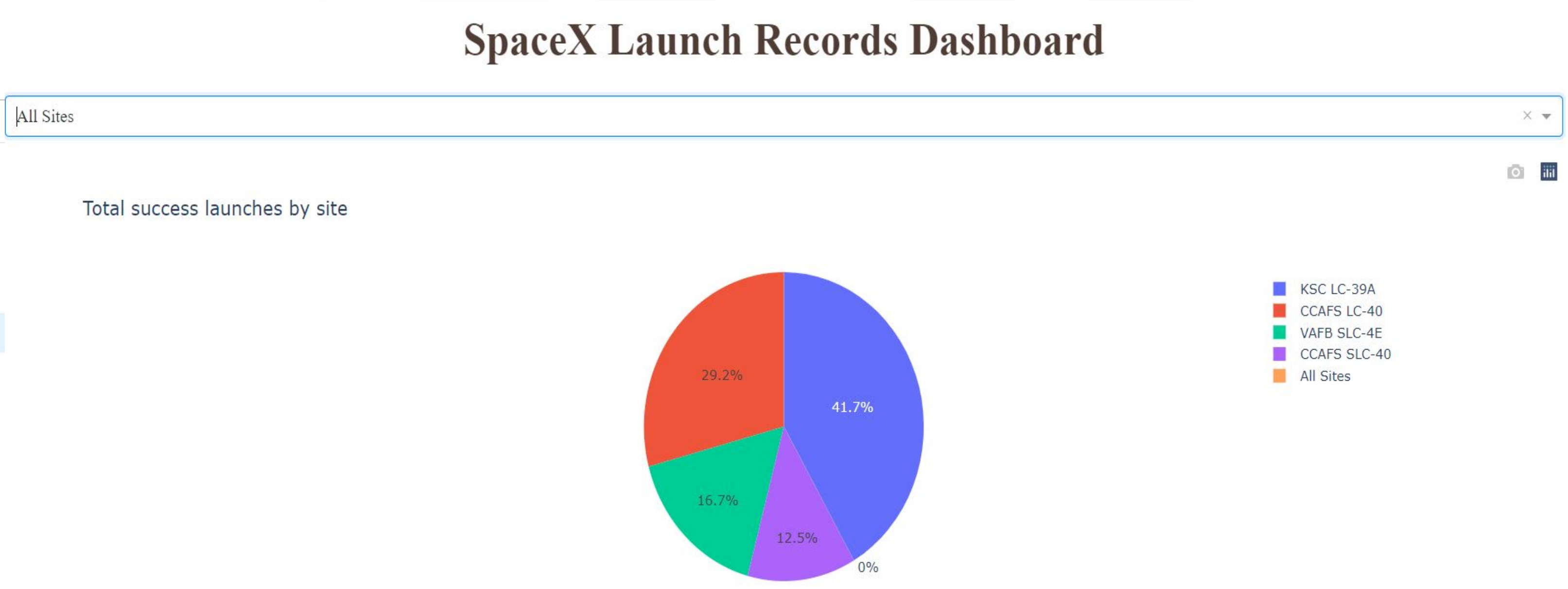
RESULTS



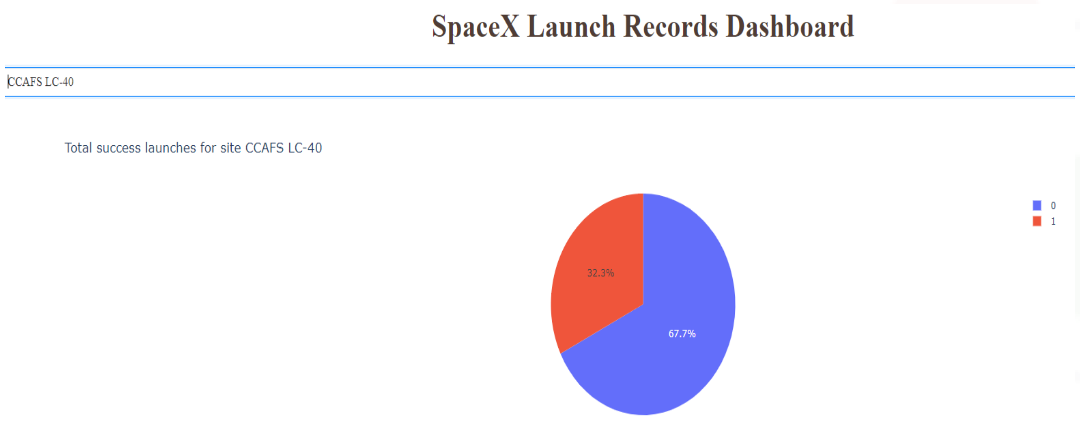
SpaceX Launch Records Dashboard
All Sites
Total success launches by Site
41.7%
29.2%
16.7%
12.5%
0%
KSC LC-39A
CCAFS LC-40
VAFB SLC-4E
CCAFS SLC-40
All Sites
ayload range (Kg):
v1.0
v1.1
FT
B4
B5
1000
10k
Payload Mass (kg)

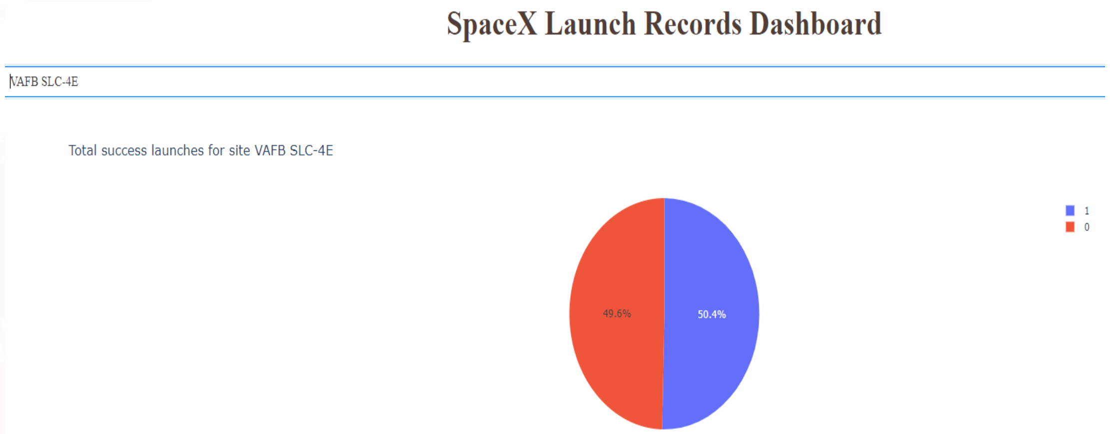
RESULTS

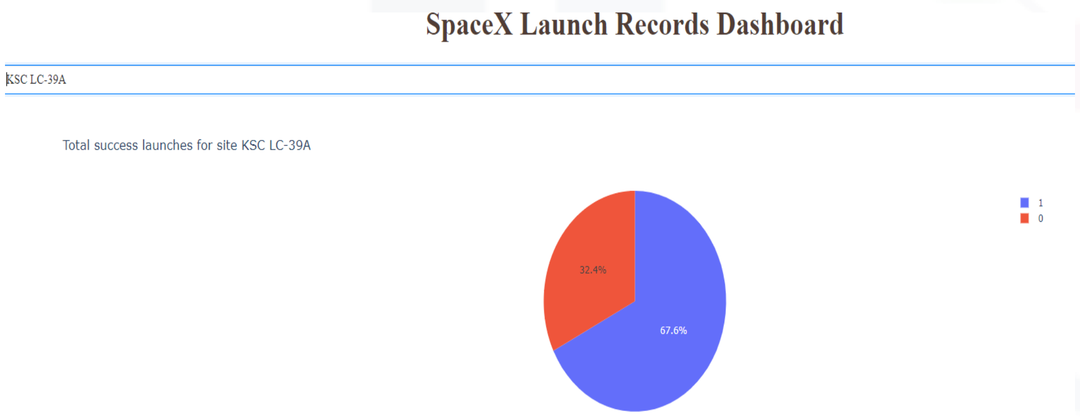

SpaceX Launch Records Dashboard
All Sites
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
All Sites

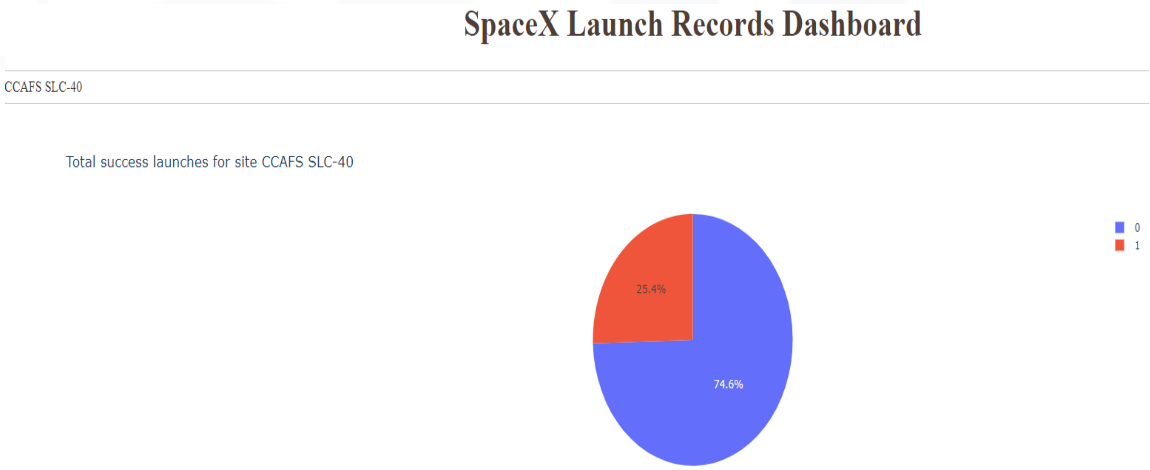

SpaceX Launch Records Dashboard
All Sites
Total success launches by site
41.7%
29.2%
16.7%
12.5%
0%
KSC LC-39A
CCAFS LC-40
VAFB SLC-4E
CCAFS SLC-40
All Sites

RESULTS

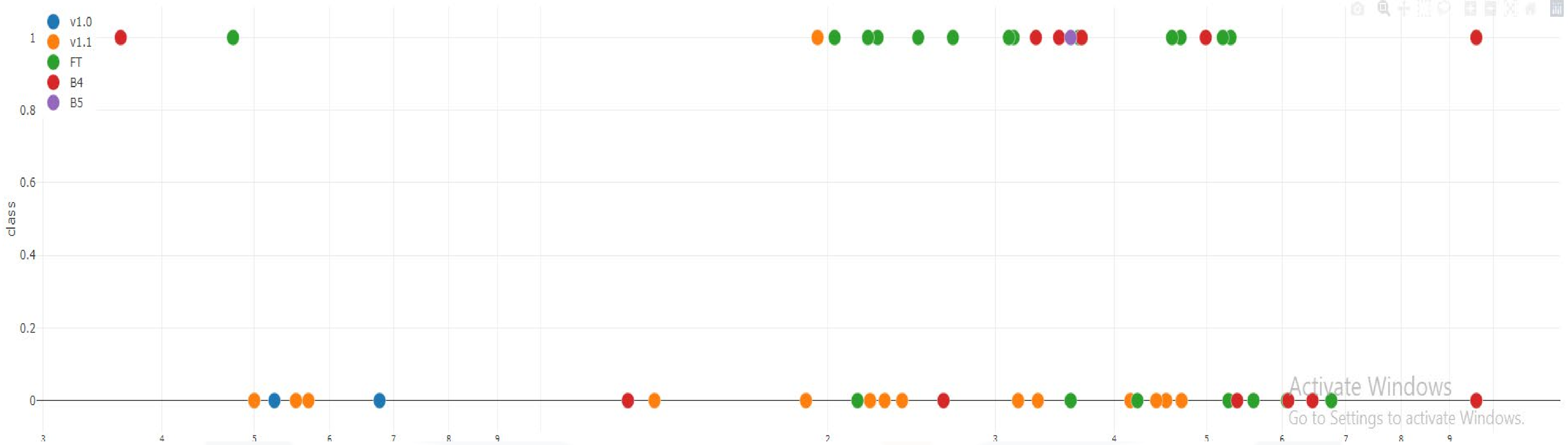

SpaceX Launch Records Dashboard
CCAFS LC-40
Total success launches for site CCAFS LC-40
0
1
32.3%
67.7%


SpaceX Launch Records Dashboard
VAFB SLC-4E
Total success launches for site VAFB SLC-4E
1
0
49.6%
50.4%


SpaceX Launch Records Dashboard
KSC LC-39A
Total success launches for site KSC LC-39A
1
0
32.4%
67.6%


SpaceX Launch Records Dashboard
CCAFS SLC-40
Total success launches for site CCAFS SLC-40
0
1
25.4%
74.6%

RESULTS



v1.0
v1.1
FT
B4
B5
class
1
0.8
0.6
0.4
0.2
0
3
4
5
6
7
8
9
2
3
4
5
6
7
8
9
Activate Windows
Go to Settings to activate Windows.

RESULTS

After visual analysis using the the above dashboard, we were able to obtain some insights to answer the following five questions:

1. KSC LC-39A is the site that has the largest successful launches at **41.7%**.
2.KSC LC-39A is site that has the highest launch success rate at **67.6%** successes over **32.4%** failures.
3.Payload range(s) between **2k** and **4k** has the highest launch success rate
4.Payload range(s) between **8k** and **10k** has the lowest launch success rate
5. **FT** is the F9 Booster version with the highest launch success rate

RESULTS

EDA with SQL results

In many cases, SQL is the "meat and potatoes" of data analysis—it's used for accessing, cleaning, and analyzing data that's stored in databases., Here, we show the results of SQL queries that we executed more data analysis for SpaceX dataset.

After loading the given dataset into the table in a Db2 database, we executed the following SQL queries to answer the assignment questions. The results are as follows:

Task 1

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

In [28]:
```
%sql select distinct(launch_site) from SPACEXTBL
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[28]:

launch_site
CCAFS LC-40
CCAFS SLC-40
CCAFSSLC-40
KSC LC-39A
VAFB SLC-4E

RESULTS

Task 2

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

In [12]:
```
%sql select launch_site from SPACEXTBL where launch_site like 'CCA%' limit 5
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[12]:

launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

RESULTS

Task 3

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

In [13]:
```
%sql select sum(payload_mass__kg_) as payloadmass from SPACEXTBL
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[13]:

payloadmass
619967

Task 4 ¶

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

In [17]:
```
%sql select avg(payload_mass__kg_) AS AVERAGE_PAYLOAD_MASS from SPACEXTBL where booster_version = 'F9 v1.1'
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[17]:

average_payload_mass
2928

RESULTS

Task 5

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

Hint:Use min function

In [19]:
```
%sql select min(Date) from SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[19]:

1
2015-12-22

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 [20]:
```
%sql select booster_version from SPACEXTBL where Landing_Outcome = 'Success (drone ship)' and Payload_Mass__Kg_ between 4000 and
6000
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[20]:

booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

RESULTS

Task 7

List the total number of successful and failure mission outcomes

In [21]:
```
%sql select count(MISSION_OUTCOME) as missionoutcomes from SPACEXTBL GROUP BY MISSION_OUTCOME;
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[21]:

missionoutcomes
1
99
1

RESULTS

Task 8

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

In [22]:
```
%sql select booster_version, payload_mass__kg_ from SPACEXTBL where payload_mass__kg_ = (select max(payload_mass__kg_) from SPAC
EXTBL)
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[22]:

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

Activat

RESULTS

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

In [23]:
```
#%sql select MONTH(Date) AS month_of_date, booster_version, launch_site, landing_outcome from SPACEXTBL where landing_outcome in
(select landing_outcome from SPACEXTBL where landing_outcome = 'Failure (drone ship)') and MONTH(Date) in (select MONTH(Date) fr
om SPACEXTBL where YEAR(Date) = '2015')

%sql select MONTH(Date), MISSION_OUTCOME, BOOSTER_VERSION, LAUNCH_SITE from SPACEXTBL where EXTRACT(YEAR FROM DATE) = '2015';
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[23]:

1	mission_outcome	booster_version	launch_site
1	Success	F9 v1.1 B1012	CCAFS LC-40
2	Success	F9 v1.1 B1013	CCAFS LC-40
3	Success	F9 v1.1 B1014	CCAFS LC-40
4	Success	F9 v1.1 B1015	CCAFS LC-40
4	Success	F9 v1.1 B1016	CCAFS LC-40
6	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	CCAFS LC-40

RESULTS

Task 10

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

In [27]:
```
%sql select landing_outcome, Rank() OVER(ORDER BY landing_outcome DESC) Rank from SPACEXTBL where Date between '2010-06-04' and
'2017-03-20' ORDER BY Date Desc
```

```
 * ibm_db_sa://mrr81681:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[27]:

landing_outcome	RANK
No attempt	12
Success (ground pad)	3
Success (drone ship)	6
Success (drone ship)	6
Success (ground pad)	3
Failure (drone ship)	24
Success (drone ship)	6
Success (drone ship)	6
Success (drone ship)	6
Failure (drone ship)	24
Failure (drone ship)	24
Precluded (drone ship)	11
No attempt	12
Failure (drone ship)	24
No attempt	12
Controlled (ocean)	29
Failure (drone ship)	24
Uncontrolled (ocean)	1
No attempt	12
No attempt	12
Controlled (ocean)	29
Controlled (ocean)	29
No attempt	12
No attempt	12
Uncontrolled (ocean)	1
No attempt	12
No attempt	12
No attempt	12
Failure (parachute)	22
Failure (parachute)	22

DISCUSSION

If I reflect the work necessary to create these results, what comes to my mind is that for typical ways of scraping, cleaning, handling, transforming and visualizing data, all the tools are simply there.

We started with the data exploration where we got a feeling for the dataset, checked missing data and learned which features are important. During the data preprocessing part, we computed missing values, converted features into numeric ones, grouped values into categories and created a few new features.

Afterwards, we started training 3 different machine learning models, picked one of them that performed best. We looked at different features and tuned its performance through optimizing it's hyperparameter values, then looked at their accuracy and confusion matrix.

I ended the study by visualizing the data and classifying information on the NASA Johnson Space Centre map and calculated launch sites proximities using maps.

CONCLUSION

During this project, I have used the SpaceX API and different methods of data science and machine learning to obtain a final recommendation as to where to find launch site, what's in their proximity, etc. The results of this study may present useful information that can be used by the alternate company who want to bid against SpaceX for a rocket launch.

In conclusion, this project has been a wonderful opportunity for me to harness data technologies in order to find a solution to a real-world problem.

REFERENCES

- SpaceX API
- https://www.spacex.com/vehicles/falcon-9/
- https://towardsdatascience.com/data-science-skills-web-scraping-using-python-d1a85ef607ed
- https://towardsdatascience.com/predicting-the-survival-of-titanic-passengers-30870ccc7e8
- https://www.analyticsvidhya.com/blog/2020/06/guide-geospatial-analysis-folium-python/