The Shortcut to Space with Data

Haydar Uçar

19.9.22

Outline

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

Executive Summary

-Introduction

-Methodology

- -Data Collection Methodology
- -Data Wrangling Methodology
- -EDA and Interactive Visual Analytics Methodology
- -Predictive Analysis Methodology

-Results

- -EDA with Visualization
- -EDA with SQL
- -Interactive Map with Folium
- -Plotly Dash Dashboard
- -Predictive Analysis

-Conclusion

Introduction

- Project Background

  - I gathered information about Space X and creating dashboards for determine if SpaceX will reuse the first stage. I trained a machine learning model and use public information to predict if SpaceX will reuse the first stage.

- Promlems I Want to Answer

  - What factors determine whether the rocket will land successfully or not?

  -How accurately can I predict whether the rocket will land successfully with the data I have?

METHODOLOGY

- Data Collection Methodology
- Data Wrangling Methodology
- EDA and Interactive Visual Analytics Methodology
- Predictive Analysis Methodology

Data Collection Methodology – SpaceX API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url).json()
```

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

```
mean_pl = data_falcon9["PayloadMass"].mean()
data_falcon9["PayloadMass"].replace(np.nan, mean_pl, inplace = True)
data_falcon9["PayloadMass"].isnull().sum()
```

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

- Collected data with using get request to the SpaceX API
- Decoded the response content with using json().
- Cleaned the data, checked missing values and fill in missing values where necessary .
- Filter Dataframe and export to (.csv)

https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/data_collection_api.ipynb

Data Collection - Webscraping

- Using GET method to request the HTML page, as an HTTP response and create a `BeautifulSoup` object from the HTTP response.
- Finding tables and getting column names.
- Create a dictionary and append data.
- Export to (.csv)

- https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/webscraping.ipynb

```
response = requests.get(static_url)
soup = BeautifulSoup(response.content, "html.parser")
```

```
html_tables = soup.find_all("table")
```

```
first_launch_table = html_tables[2]
```

```
for column in first_launch_table.find_all('th'):
    column_name = extract_column_from_header(column)
    if column_name is not None and len(column_name)>0:
        column_names.append(column_name)
```

```
launch_dict= dict.fromkeys(column_

# Remove an irrelvant column
del launch_dict['Date and time ( )

# Let's initial the launch_dict wi
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table'
    # get table row
    for rows in table.find_all("tr"):
```

In [27]:

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling Methodology

- Perform EDA
- Calculate the number of launches on each site and the number and occurrence of each orbit
- Create a landing outcome label from Outcome column
- Export it to a CSV

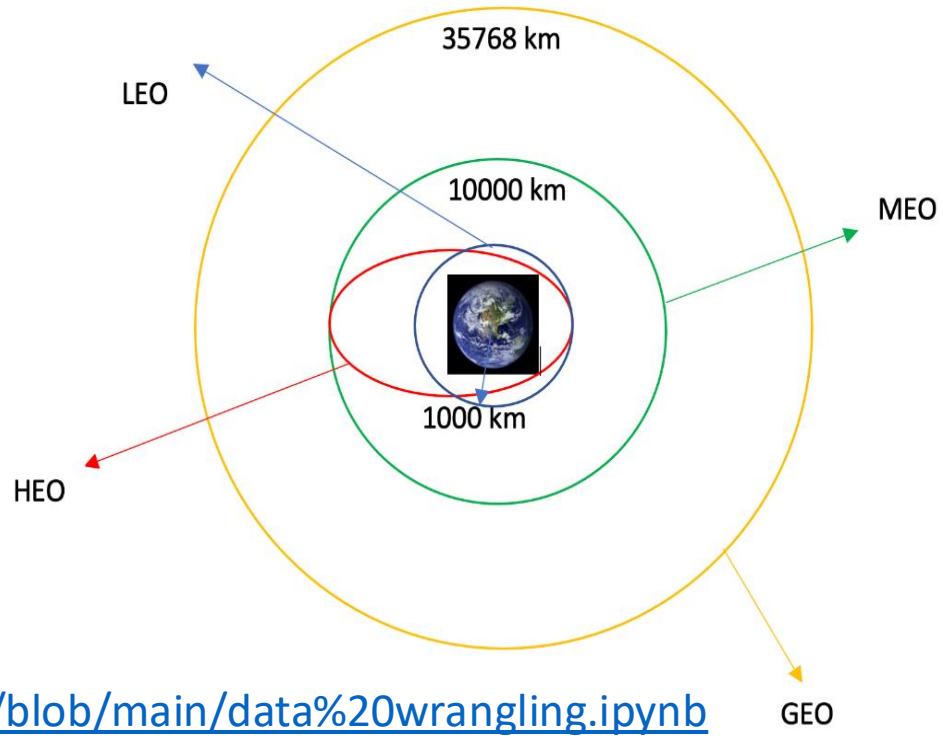
```
In [3]: df.isnull().sum()/df.count()*100
```

```
df["LaunchSite"].value_counts()
```

```
df["Orbit"].value_counts()
```

```
for key,value in df["Outcome"].items(
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

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



https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/data%20wrangling.ipynb

EDA with SQL Methodology

- Write and execute SQL queries to find out for instance.

-Display the names of the unique launch sites

-Display the total payload mass carried by boosters launched by NASA

-Display average payload mass carried by booster version F9

-List the total number of successful and failure mission outcomes

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL
```

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)" FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';
```

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster Version F9 v1.1" FROM SPACEXTBL \
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/eda_sql.ipynb

EDA with Visualization Methodology

- Visualize the relationship between Flight Number and Launch Site
- Visualize the relationship between Payload and Launch Site
- Visualize the relationship between success rate of each orbit type
- Visualize the relationship between Payload and Orbit type
- Visualize the launch success yearly trend
- Features Engineering
- Create dummy variables to categorical columns
- Export it to a **CSV**

```
plt.figure(figsize=(14,8))
sns.scatterplot(x="FlightNumber", y="LaunchSite", hue="Class", data = df)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

```
plt.figure(figsize=(14,8))
sns.scatterplot(x="PayloadMass", y="LaunchSite", hue="Class", data = df)
plt.xlabel("Pay Load Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```

https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/eda_dataviz.ipynb

EDA with Visualization Methodology

```
xh = df.groupby('Orbit')['Class'].mean()
ax = xh.plot(kind='bar', figsize=(8, 7), color='#86bf91', zorder=2, width=0.9)
ax.set_xlabel("Orbit", labelpad=20, weight='bold', size=14)
ax.set_ylabel("Sucess rate of each orbit", labelpad=20, weight='bold', size=14)
```

```
df['year']=Extract_year(df["Date"])
df_groupby_year=df.groupby("year",as_index=False)["Class"].mean()
sns.set(rc={'figure.figsize':(12,9)})
sns.lineplot(data=df_groupby_year, x="year", y="Class" )
plt.xlabel("Year",fontsize=20)
plt.title('Space X Rocket Success Rates')
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```

```
features = df[['FlightNumber', 'PayloadMass', 'Orbit', 'LaunchSite', 'Flights', '(
features.head()
```

```
features_one_hot.to_csv('dataset_part_3.csv', index=False)
```

https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/eda_dataviz.ipynb

Map with Folium Methodology

- Mark all launch sites on a map
- Mark the success/failed launches for each site on the map
- Calculate the distances between a launch site to its proximities

```
for index, site in launch_sites_df.iterrows():
    circle = folium.Circle([site['Lat'], site['Long']], color='#d35400', radius=50, fill=True).ad
    marker = folium.Marker(
    [site['Lat'], site['Long']],
    # Create an icon as a text label
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % site['Launch Site'],
```

```
marker_cluster = MarkerCluster()

for index, row in spacex_df.iterrows():

    folium.map.Marker((row['Lat'], row['Long']), icon=folium.Icon(color='white', icon_color=row['marker_color'])).add_to(marker_cluster)

site_map.add_child(marker_cluster)
```

https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/launch_site_location.ipynb

Dashboard Methodology

- Build an interactive dashboard.
- Plot piecharts showing total launches by a certain sites.
- Plot scatter plot showing the relationship with Outcome and Payloadmass for the different booster version.

- https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/spacex_dash_app.py
- Dashboars site
- https://haydarucar1-8050.theiadocker-2-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai

ML Prediction Methodology

- Create a NumPy array from the column `Class`
- Standardize the data
- `Train, test spilting`
- Create a ML Classification algorithm(Logistic Regression, SVM, Decision Tree, KNN) object then create a GridSearchCV object algorithm_cv with cv = 10. Fit the object to find the best parameters.

-Calculate the accuracy of algorithm_cv on the test data using the method

-Plot the confusion matrix

https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/ml_prediction.ipynb

```
Y = data["Class"].to_numpy()
Y
```

```
transform = preprocessing.StandardScaler()
X = transform.fit_transform(X)
X
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.20, random_state = 2)
```

ML Prediction Methodology

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()
logreg_cv = GridSearchCV(lr,parameters,cv =10)
logreg_cv.fit(X_train, Y_train)
```

```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

```
logreg_cv.score(X_test, Y_test)
```

```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

```
algorithms = [logreg_cv, svm_cv, tree_cv, knn_cv]
results = []
for model in algorithms:
    model_dictionary = {'Model': str(model.estimator), 'Accuracy': str(model.best_score_), 'test score': str(model.score(X_test, Y_test))}
    results.append(model_dictionary)
results_df = pd.DataFrame(results)

results_df
```

https://github.com/icayir/IBM-Applied-Data-Science-Capstone-Project/blob/main/ml_prediction.ipynb

RESULTS

EDA results with SQL

EDA with Visualization

Predictive Analisys Results

EDA with SQL

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

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL
```

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

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

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

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

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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 (parachute)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)" FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)';
```

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

Total Payload Mass by NASA (CRS)
45596

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster Version F9 v1.1" FROM SPACEXTBL \
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

Average Payload Mass by Booster Version F9 v1.1
2928.4

- List the total number of successful and failure mission outcomes

```
%sql SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "Successful Mission", \
    sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "Failure Mission" \
FROM SPACEXTBL;
```

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

Successful Mission	Failure Mission
100	1

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

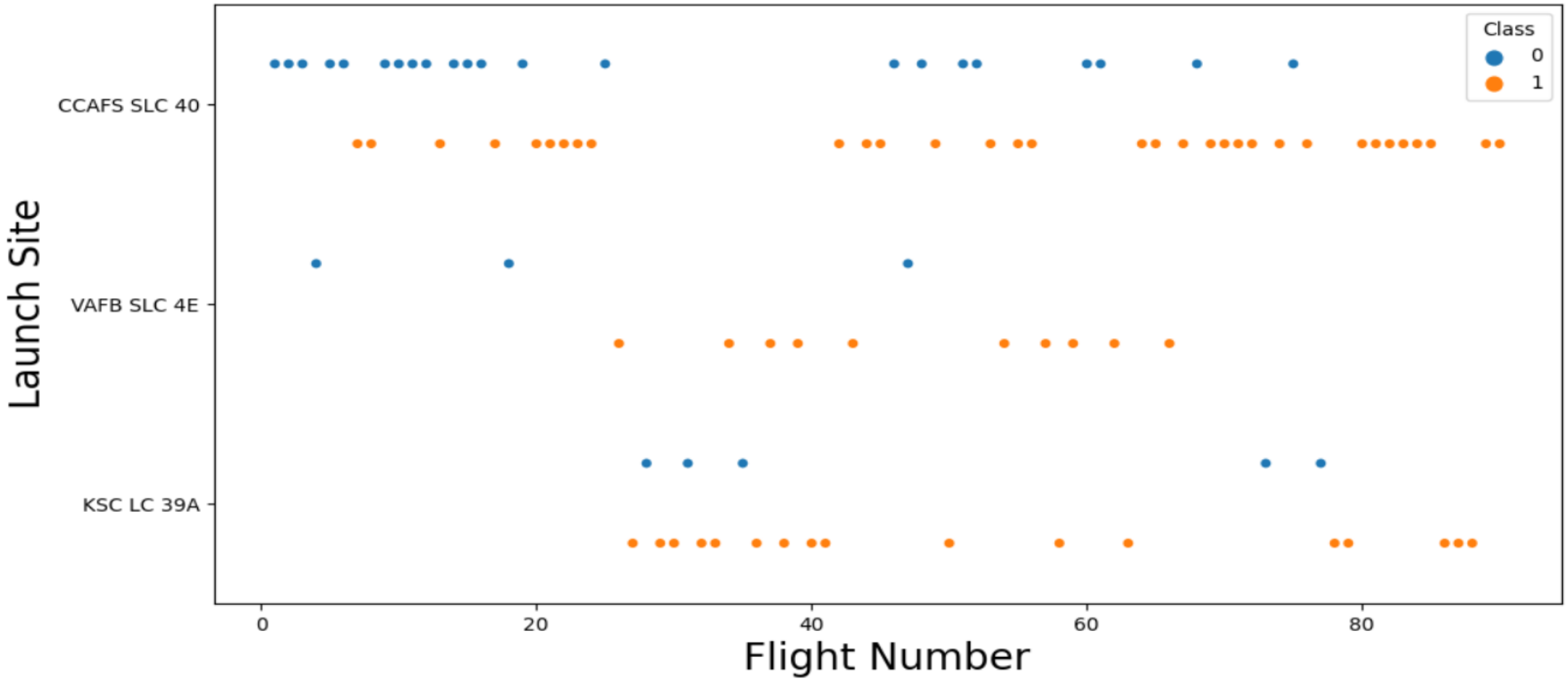
	firstsuccessfull_landing_date
0	2015-12-22

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

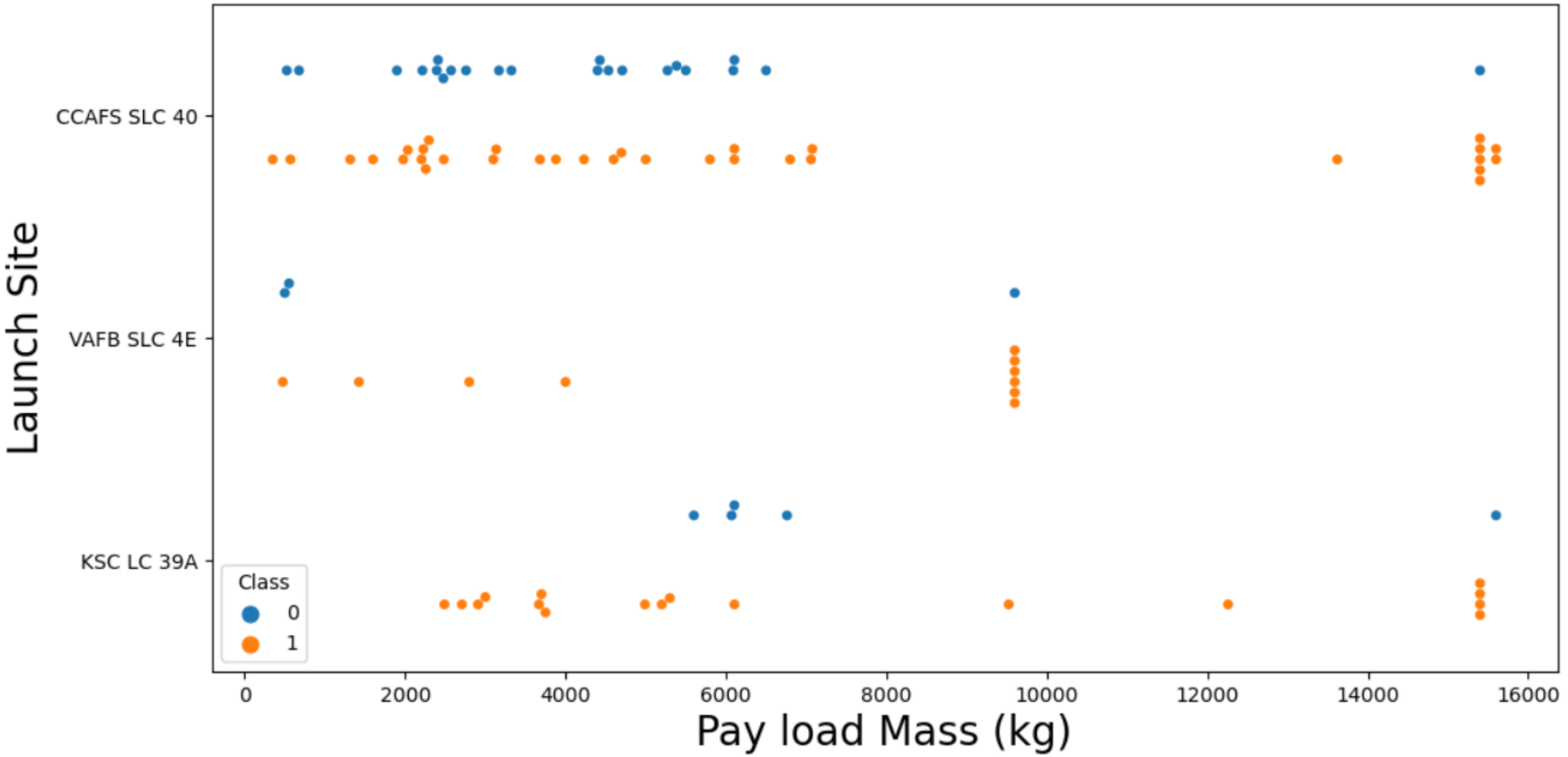
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

EDA with Visualization

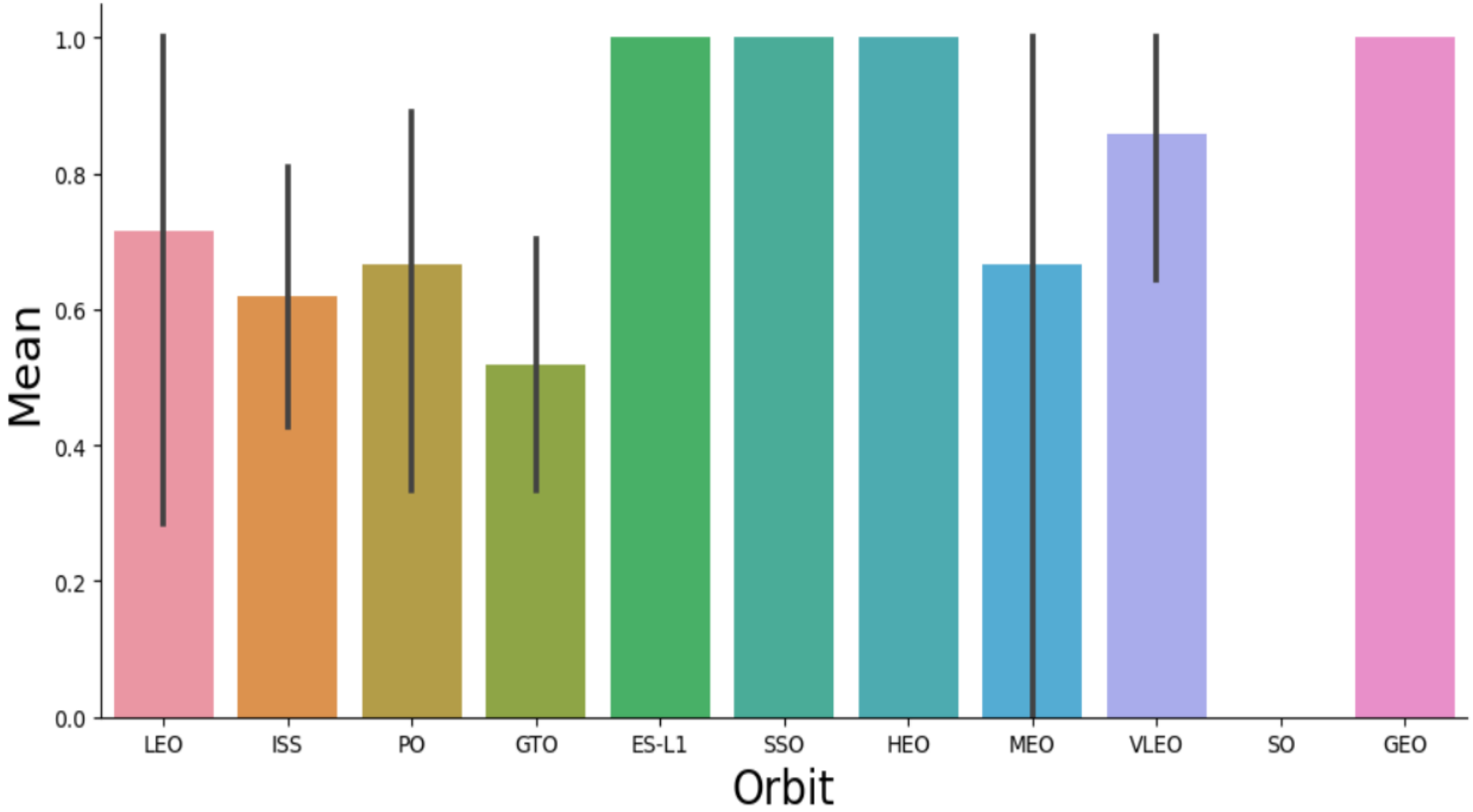
- Visualize the relationship between Flight Number and Launch Site



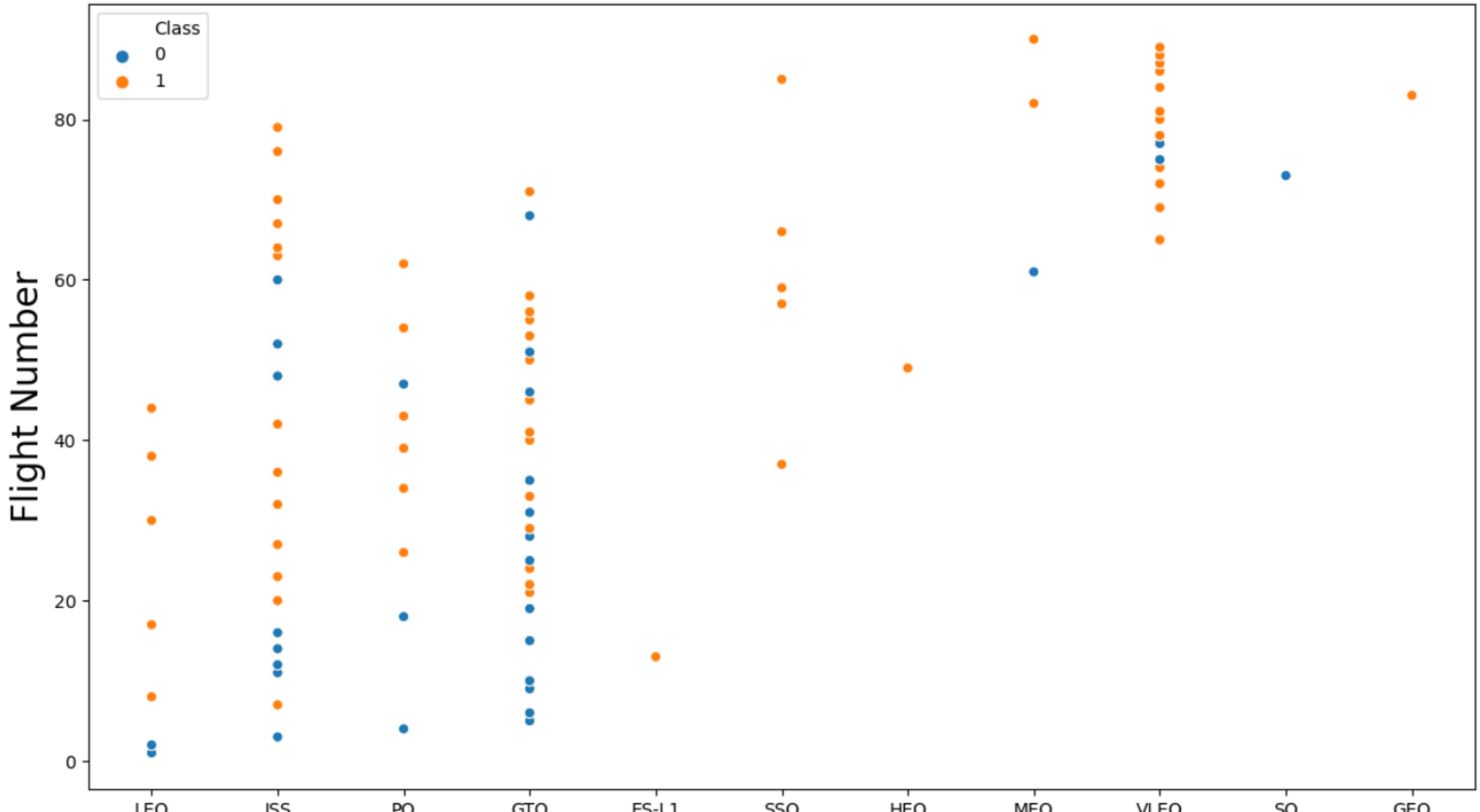
- Visualize the relationship between Payload and Launch Site



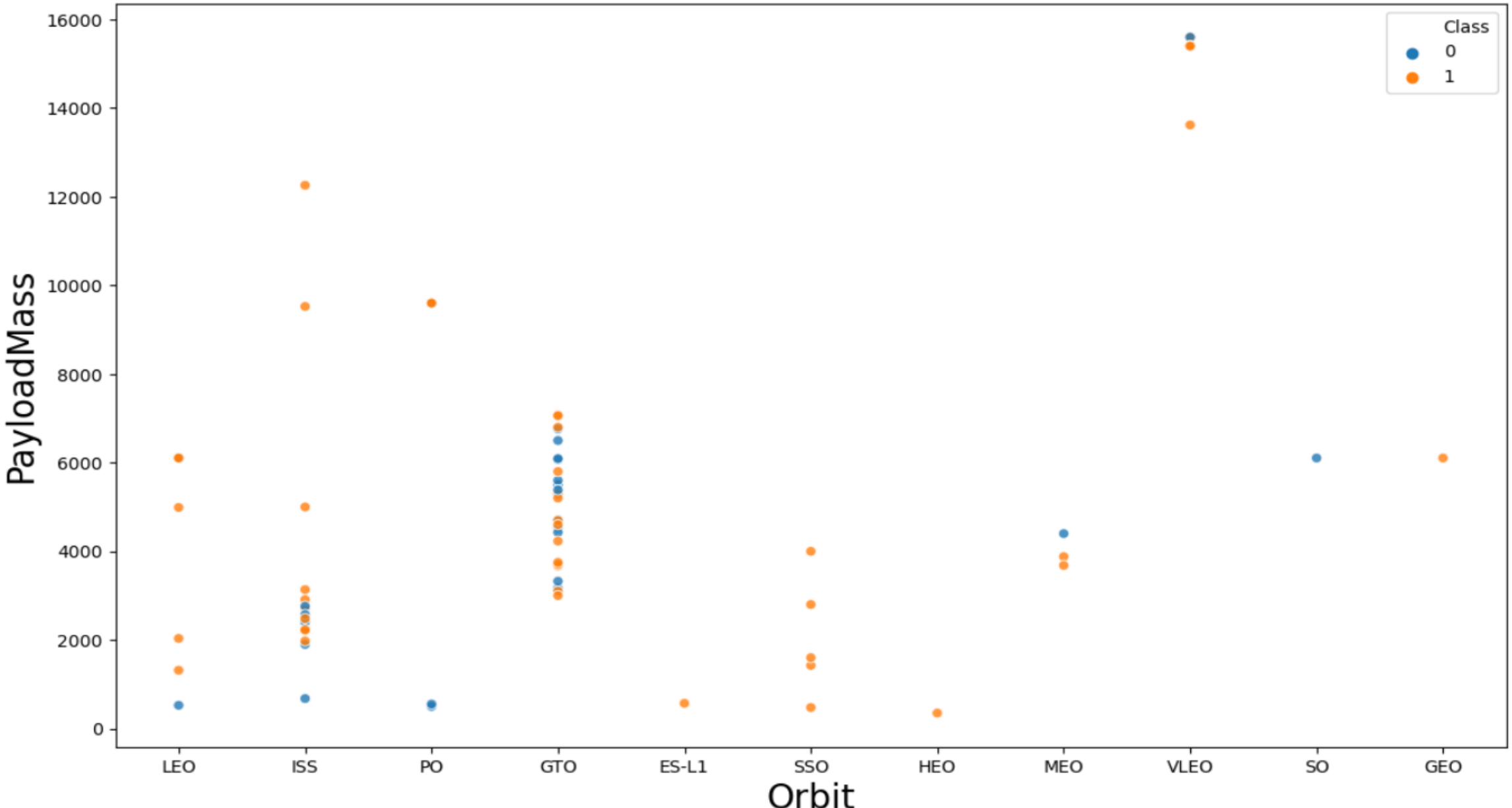
- Visualize the relationship between success rate of each orbit type



- Visualize the relationship between FlightNumber and Orbit type



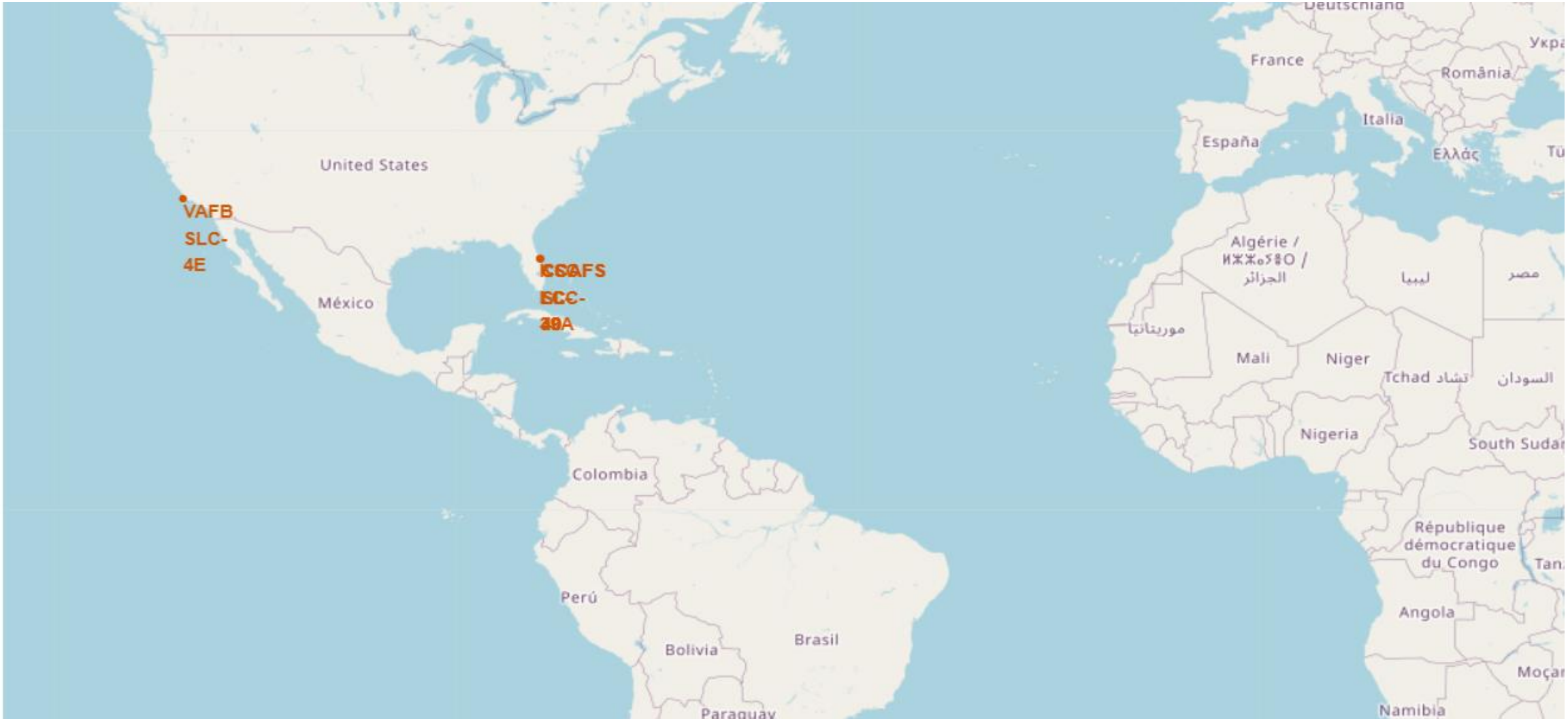
- Visualize the relationship between Payload and Orbit type



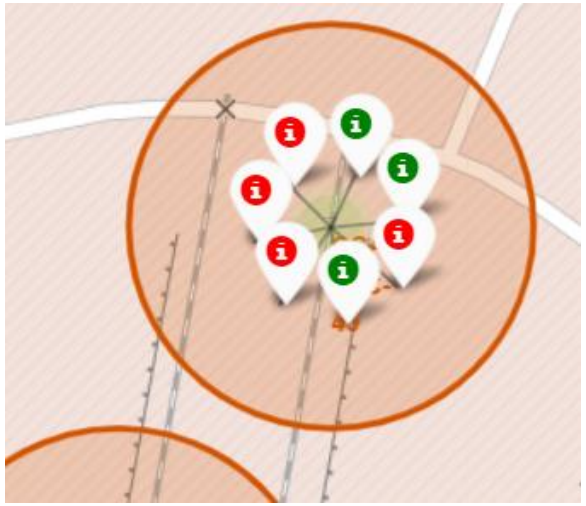
- Visualize the launch success yearly trend

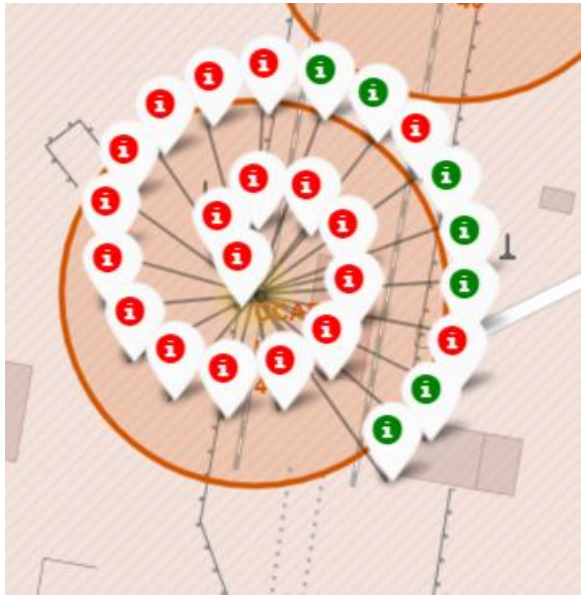
Interactive map with Follium

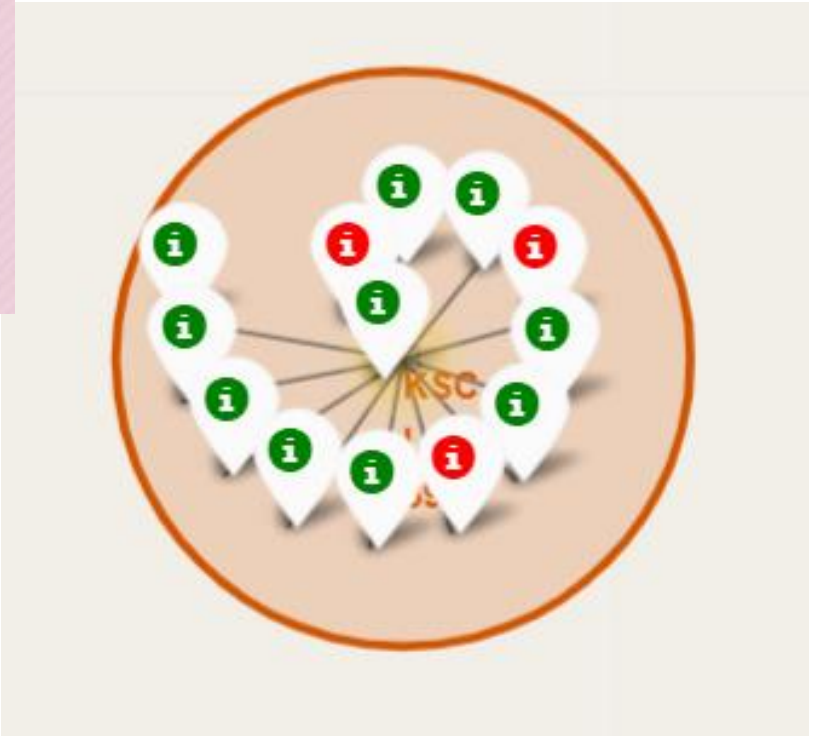
- Mark all launch sites on a map



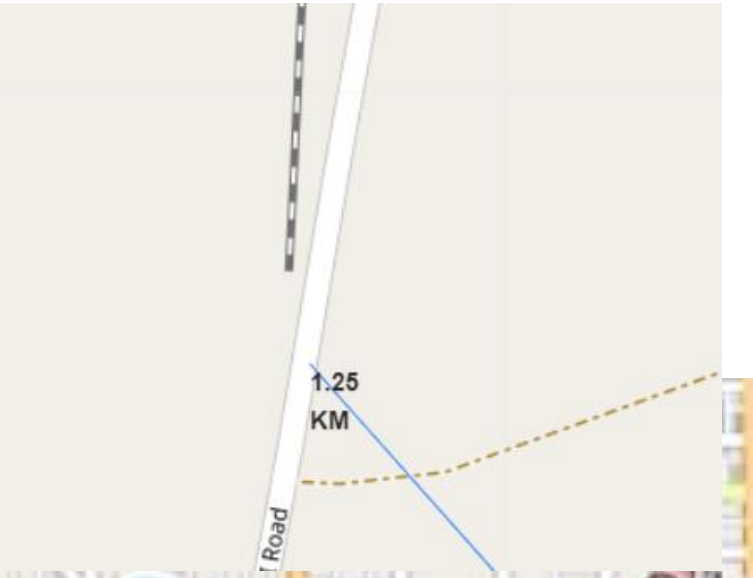
- Mark the success/failed launches for each site on the map

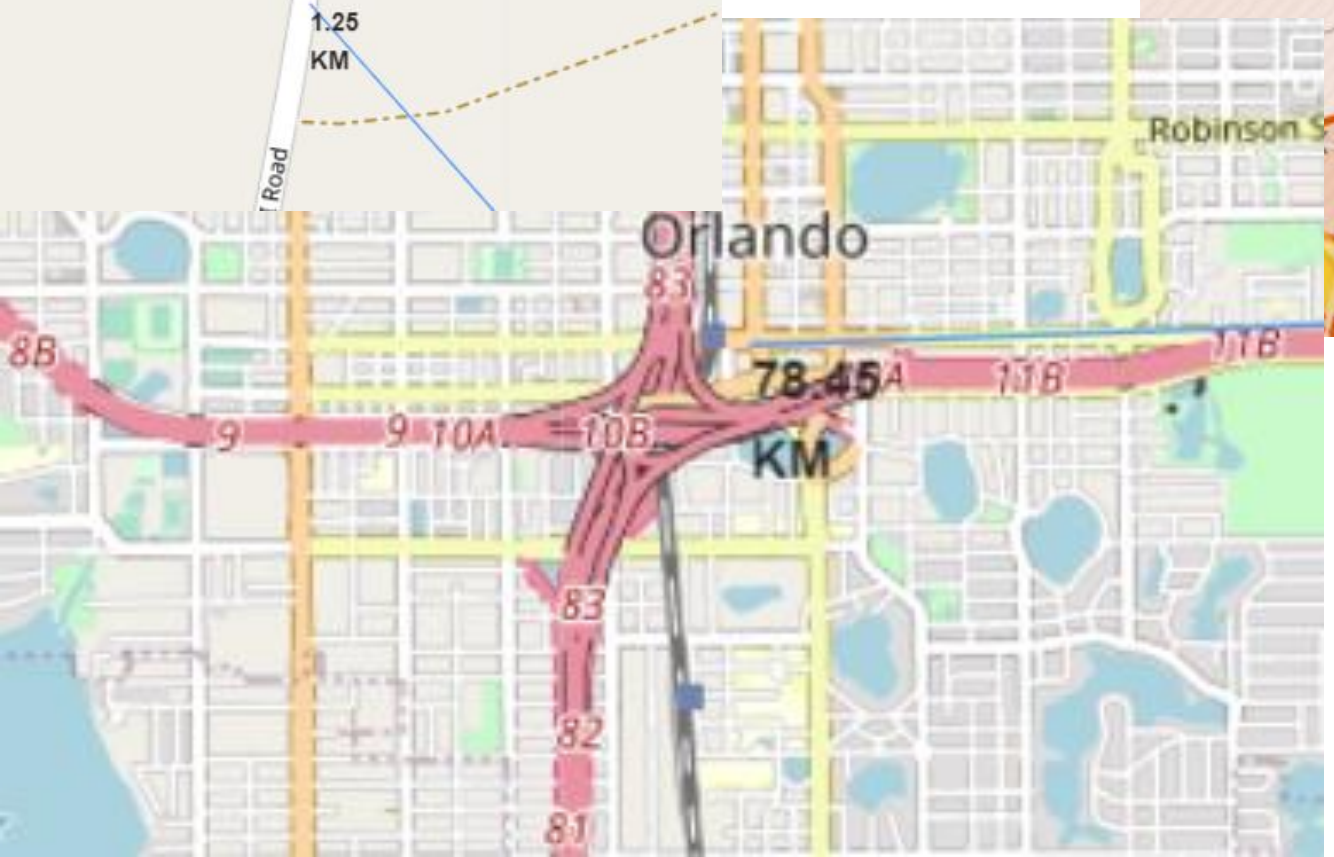






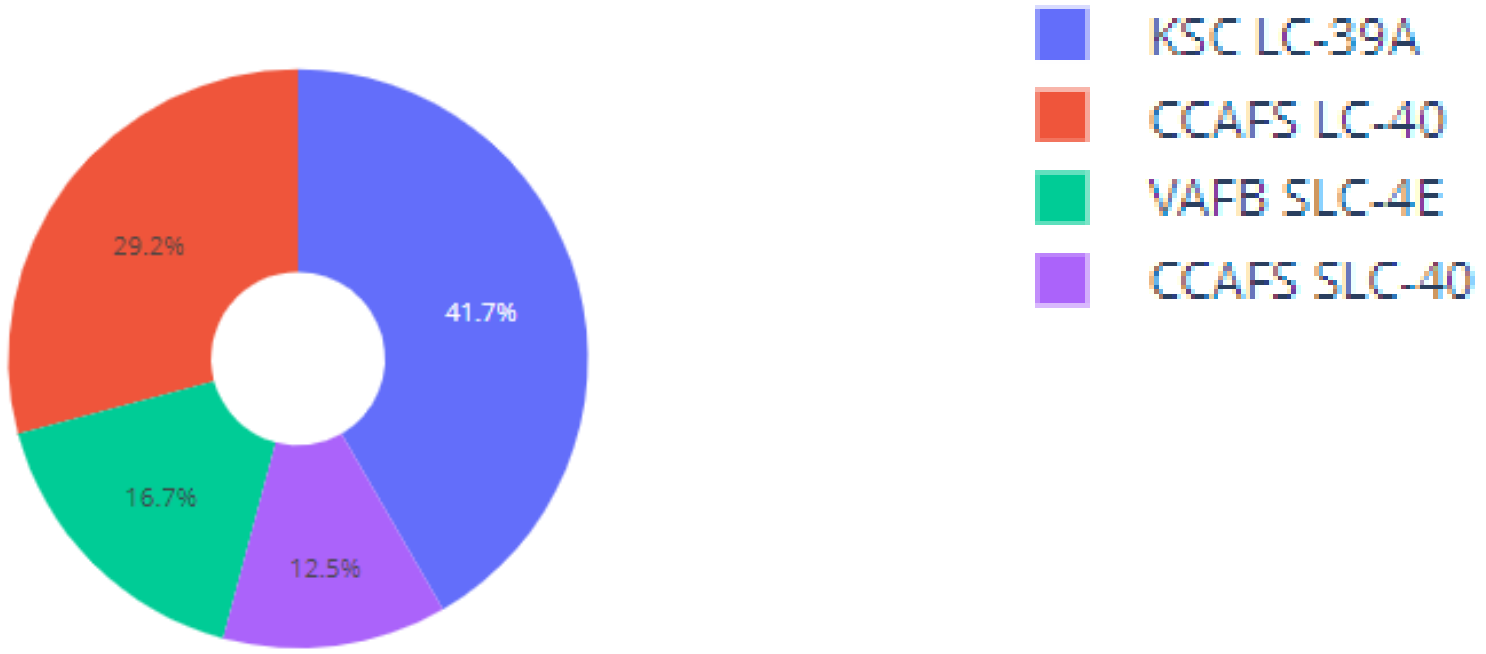
- Calculate the distances between a launch site to its proximities for CCAFS SLC-40





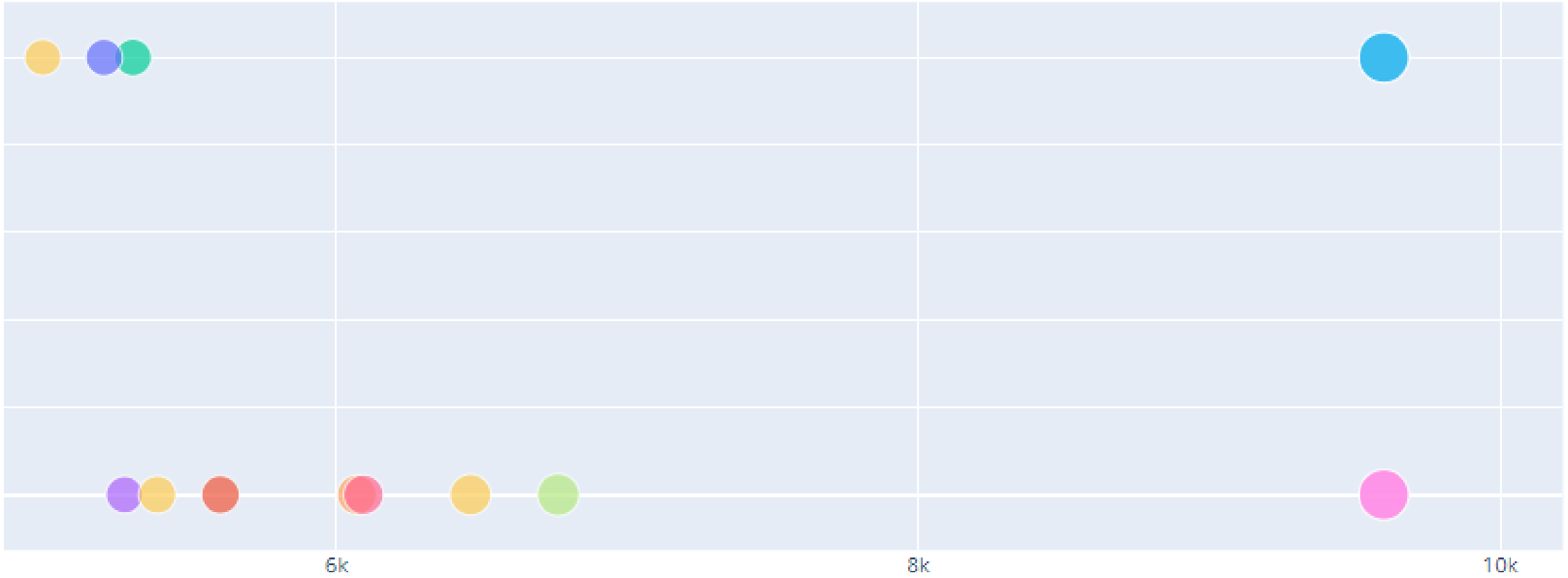
Dashboard

- Total success launches by all sites



- KLC LC-39A has bes success rate

- Payloadmass vs Launch Success



Low weighted payloads success rate higher than heavy weighted payloads

ML Prediction Results

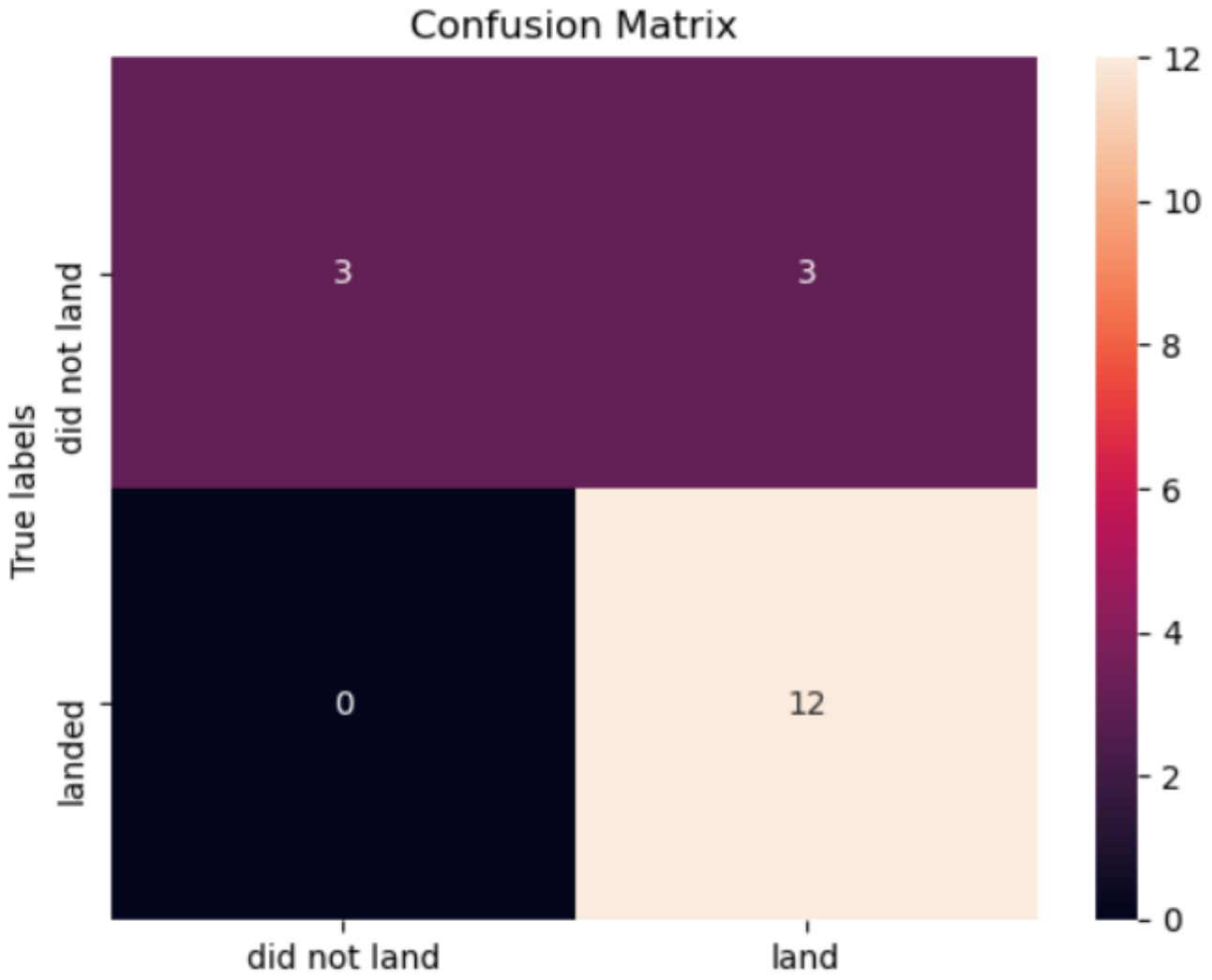
- Finding witch method perrforms best

[45]:

	Model	Accuracy	test score
0	LogisticRegression(C=1.0, class_weight=None, d...	0.8472222222222222	0.8333333333333334
1	SVC(C=1.0, cache_size=200, class_weight=None, ...	0.8472222222222222	0.8333333333333334
2	DecisionTreeClassifier(class_weight=None, crit...	0.875	0.8333333333333334
3	KNeighborsClassifier(algorithm='auto', leaf_si...	0.8472222222222222	0.8333333333333334

- With cv=10 all methods test scores are same(0.83) but Decision Tree's train accuracy score is the best.

Confision Matrix



Conclusion

All algorithms are same accuracy rate(0.83) for this dataset .

Low weighted payloads perform better than the heavier payloads.

• The success rates for SpaceX launches success rate geting higher year by year.

KSC LC-39A had the most successful launches from all the sites

• Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Appendix

- IBM Skills Network Labs