SpaceX Falcon 9 First Stage Landing Prediction

Applied Data Science Capstone Project

Shobeir Pirayeh Gar, Phd, PE

Technical Advisor, Mechanical Engineering

Halliburton Technology Center

October 2021

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
  - EDA and Visualization
  - SQL
  - Interactive Map with Folium
  - Predictive Analysis
- Conclusion
- Discussion and Recommendations
- Appendix

EXECUTIVE SUMMARY

- In this applied data science project we predicted if the Falcon 9 first stage would land successfully. This is critical as SpaceX reuses the first stage to bring down the cost of rocket launches from 165 million dollars to 62 million dollars.
- Raw data was collected using SpaceX API with booster version selected as Falcon 9. Dataframe included 90 samples with 17 different features. The following main analyses were performed:
    - Eploratory Data Analysis (EDA) to find patterns, and lable data
    - Visualization to understand the relation between features
    - SQL analysis to better manage and understand the dataset
    - Features Engineering
    - Machine Learning techniques to make predictions
- Initial EDA showed the rate of success as 67%
- The luanch site CCAFS LC-40 had a succes rate of 60% while KSC LC-39A and VAFB SLC 4E had a success rate of 77%
- Orbits ES-L1, SSO, HEO, and GEO had the highest success rate
- The success rate since 2013 kept increasing till 2020
- Logistic regression and classification tree algorithms resulted in highest accuray of predictions (83%) while the K nearest neighbor (KNN) method resulted in the lowest prediction accuracy of 66%.

INTRODUCTION

- The goal is to predict if the SpaceX Falcon 9 first stage landing is successsful
- The main analysis steps include: data collection and data wrangling, data management, exploratory data analysis, data visualization, and predictive analysis using machine learning techniques
- The nature of the predictive analysis is supervised learning
- The main questions to answer are
  - What are the main features govenring the success or failure?
  - What are the relationships between the main features and the outcome (success/failure)?
  - What are the best predictive machine learning techniques?

METHODOLOGY

- Data Collection and Data Wrangling:
  - Raw data is collected using SpaceX API
  - Data is cleaned and missing values are addressed
  - Data is flitered to only include the Falcon 9 as the booster version
  - Dataset management is done through SQL

METHODOLOGY

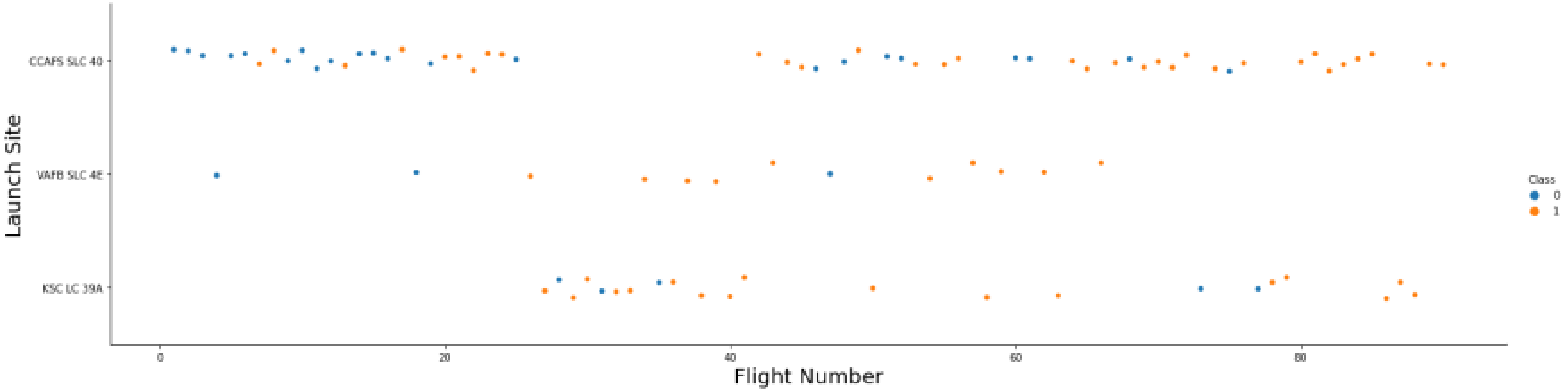
- EDA and Visual Analytics:
  - Visulaize the relationship between the main features
  - Visualize the resltionship between the main features and the success rate
  - Perform features engineering
  - Creat dummy variables to categorial features
  - Determine the training lables
  - Finalize the dataframe will all numeric column features

METHODOLOGY

- Predictive Analysis:
    - Create a column for the class in the datframe
    - Standardize (normalize) the data
    - Split the data set into training and test data
    - Find the best Hyperparameters for SVM, Classification Trees, and Logistic Regression
    - Evaluate the accuracy of each method using cross validation technique
    - Plot the confusion matrix to gain insight
    - Select the best predictive model

RESULTS - EDA and Visualization

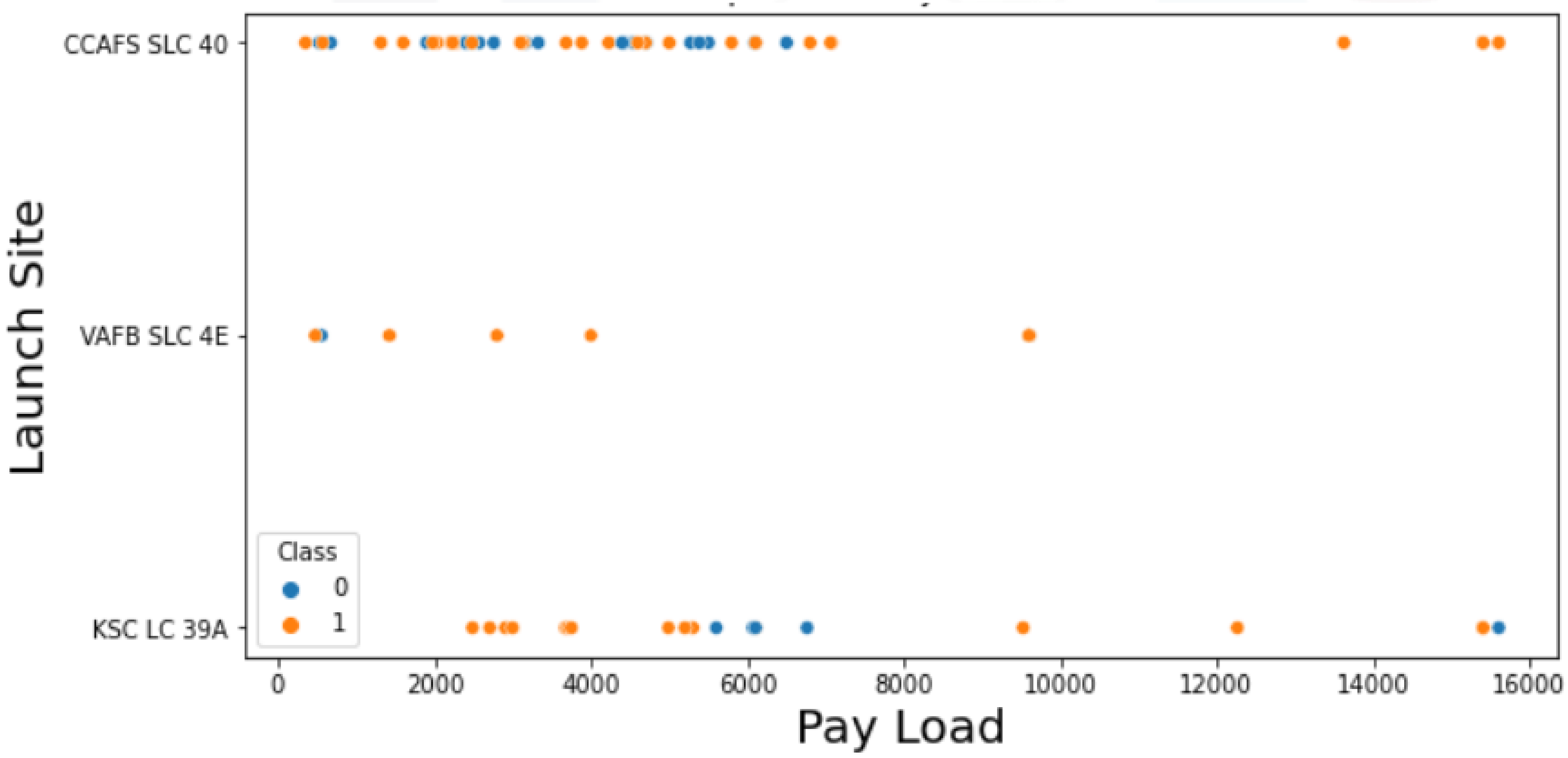
The relationship between Flight Number and Launch Site



➢ *CCAFS LC-40 , has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.*

RESULTS - EDA and Visualization

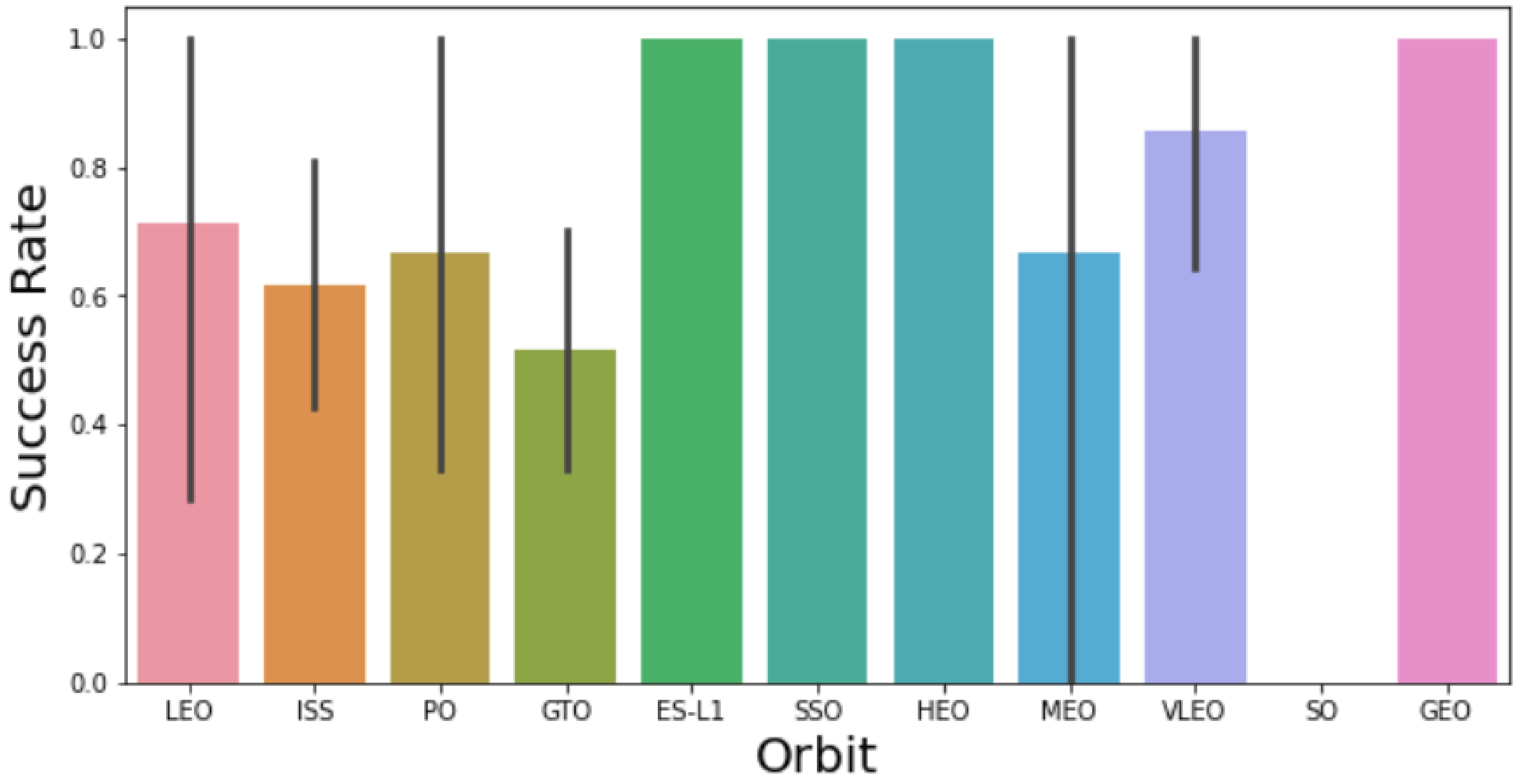
The relationship between Payload and Launch Site



- *CCAFS LC-40 is more successful with higher payloads*
- *VAFB SLC 4E is successful with average payloads*
- *KSC LC 39A is more successful with lower payloads*

RESULTS - EDA and Visualization

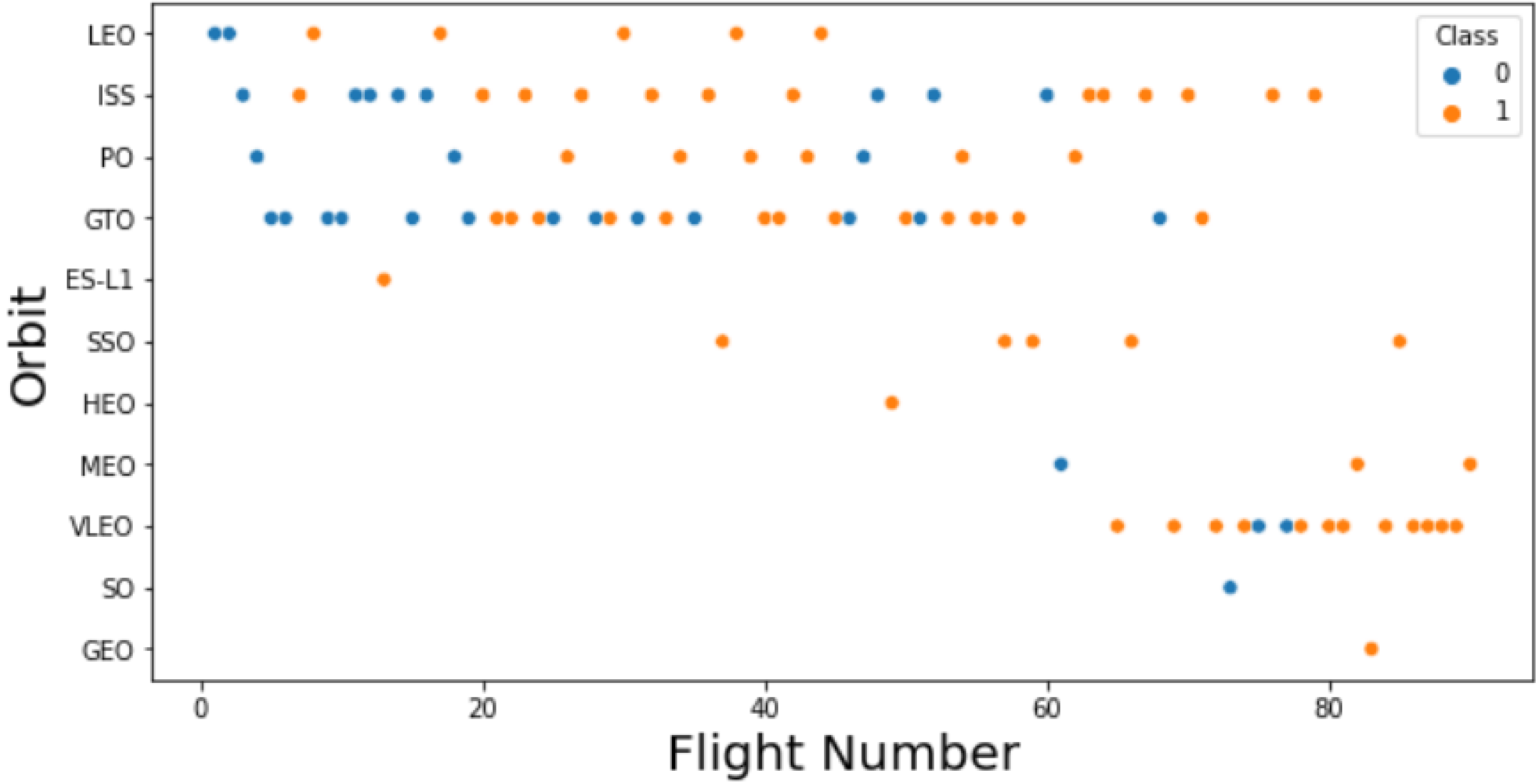
The relationship between Orbit and Success Rate



- Orbits ES-L1, SSO, HEO, and GEO has the highest success rate

RESULTS - EDA and Visualization

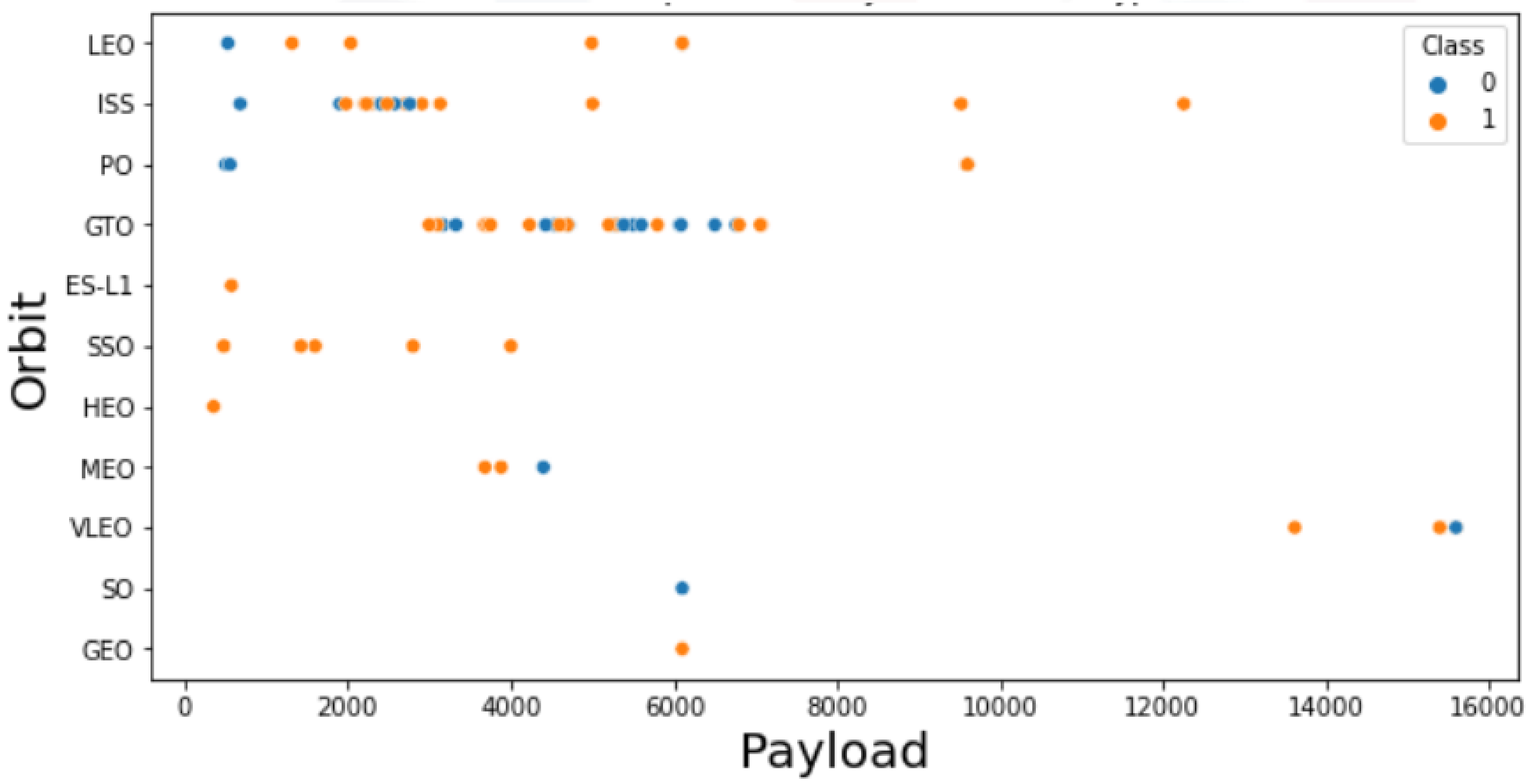
The relationship between Flight Number and Orbit type



- 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 - EDA and Visualization

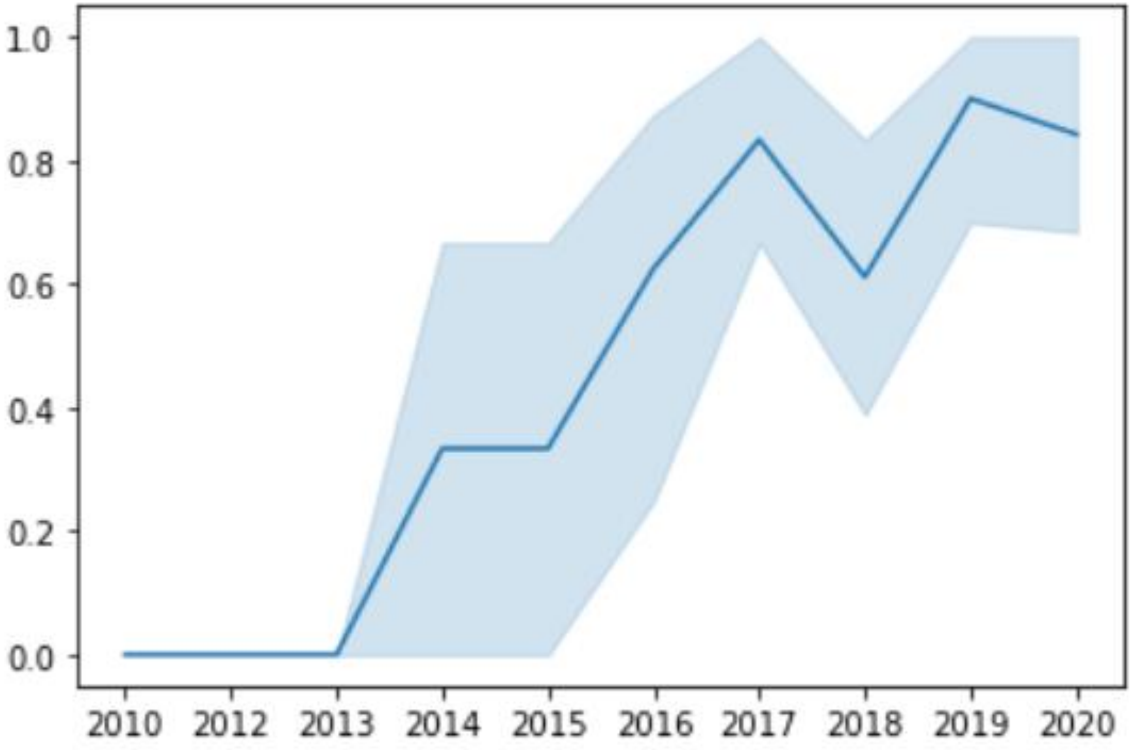
The relationship between Payload and Orbit type



- Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

RESULTS - EDA and Visualization

Launch Success Yearly Trend



- the sucess rate since 2013 kept increasing till 2020

RESULTS - SQL

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

In [7]:

```
%sql select DISTINCT(UCASE(LAUNCH_SITE)) from SPACEXTBL
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[7]:

1
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

RESULTS - SQL

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

In [18]:

```
%sql select LAUNCH_SITE from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[18]:

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

RESULTS - SQL

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

In [24]:

```
%sql select SUM(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER='NASA (CRS)'
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.net:50000/BLUDB
Done.
```

Out[24]:

1
45596

RESULTS - SQL

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

In [25]:

```
%sql select AVG(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION='F9 v1.1'
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[25]:

1
2928.400000

RESULTS - SQL

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

Hint:Use min function

In [46]:

```
%sql select MIN (DATE) from SPACEXTBL where LANDING__OUTCOME = 'Success (ground pad)'
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[46]:

1
2015-12-22

RESULTS - SQL

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [103]:

```
%sql select BOOSTER_VERSION, PAYLOAD_MASS__KG_ from SPACEXTBL where LANDING__OUTCOME = 'Su
ccess (drone ship)' and PAYLOAD_MASS__KG_ \
BETWEEN 4000 and 6000
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[103]:

booster_version	payload_mass__kg_
F9 FT B1022	4696
F9 FT B1026	4600
F9 FT B1021.2	5300
F9 FT B1031.2	5200

RESULTS - SQL

List the total number of successful and failure mission outcomes

In [58]:

```
%sql select COUNT(*) from SPACEXTBL where (MISSION_OUTCOME) LIKE 'Success%'
# %sql select COUNT(*) from SPACEXTBL where (MISSION_OUTCOME) LIKE 'Failure%'
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[58]:

1
100

RESULTS - SQL

In [61]:

```
%sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select Max(PAYLOAD_M
ASS__KG_) from SPACEXTBL)
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[61]:

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

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

RESULTS - SQL

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [70]:

```
%sql select BOOSTER_VERSION, LAUNCH_SITE, LANDING__OUTCOME, DATE from SPACEXTBL where LAND
ING__OUTCOME ='Failure (drone ship)' and YEAR(DATE) < 2016
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[70]:

booster_version	launch_site	landing__outcome	DATE
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)	2015-01-10
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)	2015-04-14

RESULTS - SQL

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20 in descending order

In [102]:

```
%sql select count(LANDING__OUTCOME), LANDING__OUTCOME from SPACEXTBL where DATE BETWEEN '2
010-06-04' and '2017-03-20' GROUP BY LANDING__OUTCOME \
ORDER BY Count DESC
```

```
 * ibm_db_sa://jgl54010:***@dashdb-txn-sbox-yp-dal09-08.services.dal.bluemix.
net:50000/BLUDB
Done.
```

Out[102]:

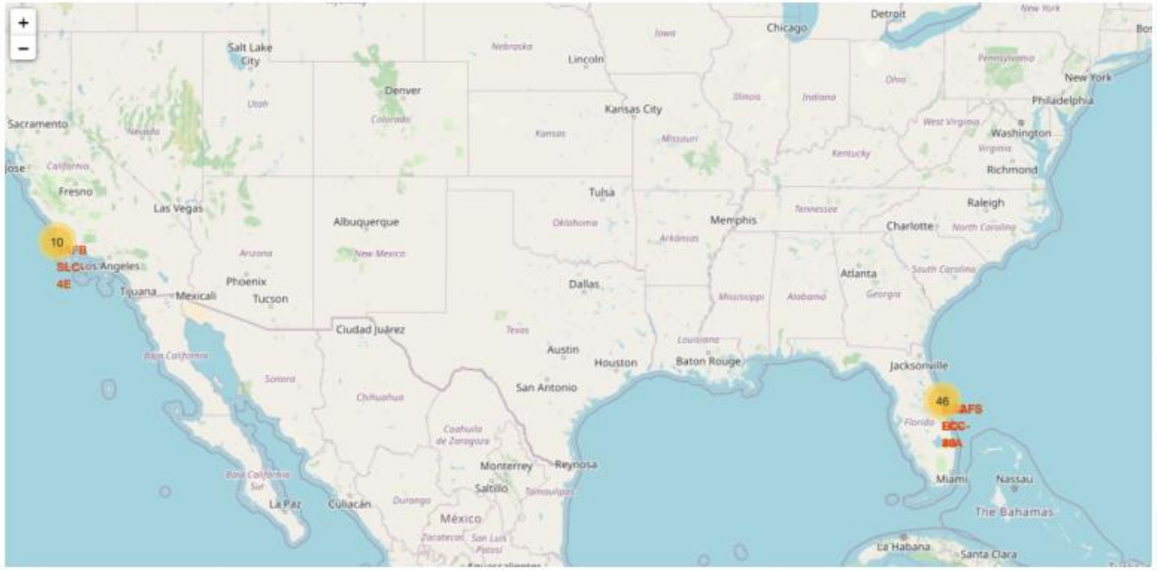
1	landing__outcome
10	No attempt
5	Failure (drone ship)
5	Success (drone ship)
3	Controlled (ocean)
3	Success (ground pad)
2	Failure (parachute)
2	Uncontrolled (ocean)
1	Precluded (drone ship)

RESULTS - Interactive Map with Folium

Mark all launch sites on a map

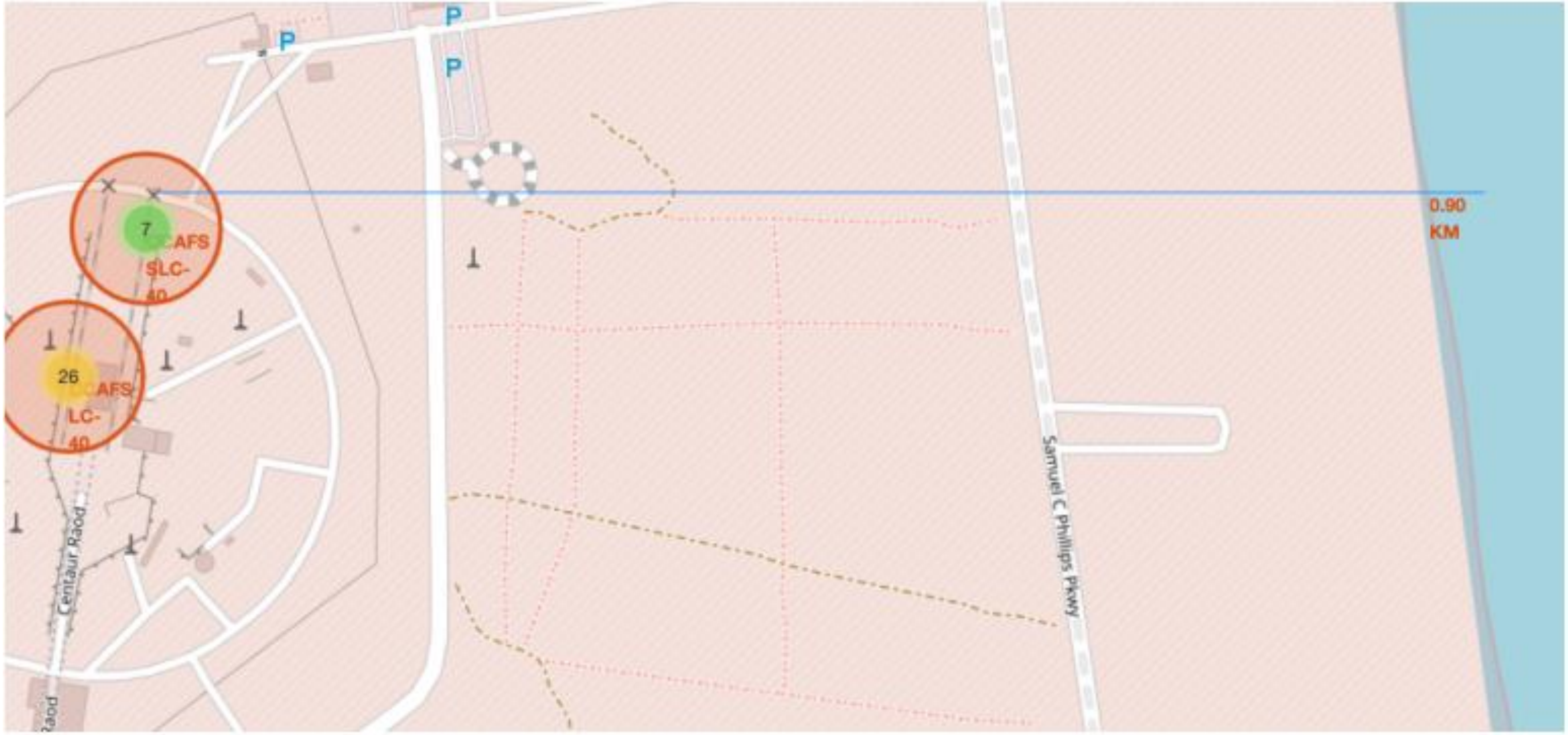
RESULTS - Interactive Map with Folium

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



RESULTS - Interactive Map with Folium

Calculate the distances between a launch site to its proximities



RESULTS - Predictive Analysis

Class Label

In [6]:
```
Y=data['Class'].to_numpy()
#Y=pd.Series(Y)
Y
```

Out[6]:
```
array([0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1,
       1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1])
```

RESULTS - Predictive Analysis

Train and Test Data Sets

X_train, X_test, Y_train, Y_test

```
In [9]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, rando
        m_state=2)
        #print('Train Set:', X_train.shape, Y_train.shape)
        #print('Test Set:', X_test.shape, Y_test.shape)
```

```
In [10]: Y_test.shape
Out[10]: (18,)
```

RESULTS - Predictive Analysis

Logistic Regression

In [11]:
```
parameters ={'C':[0.01,0.1,1],
             'penalty':['l2'],
             'solver':['lbfgs']}
```

In [12]:
```
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)
```

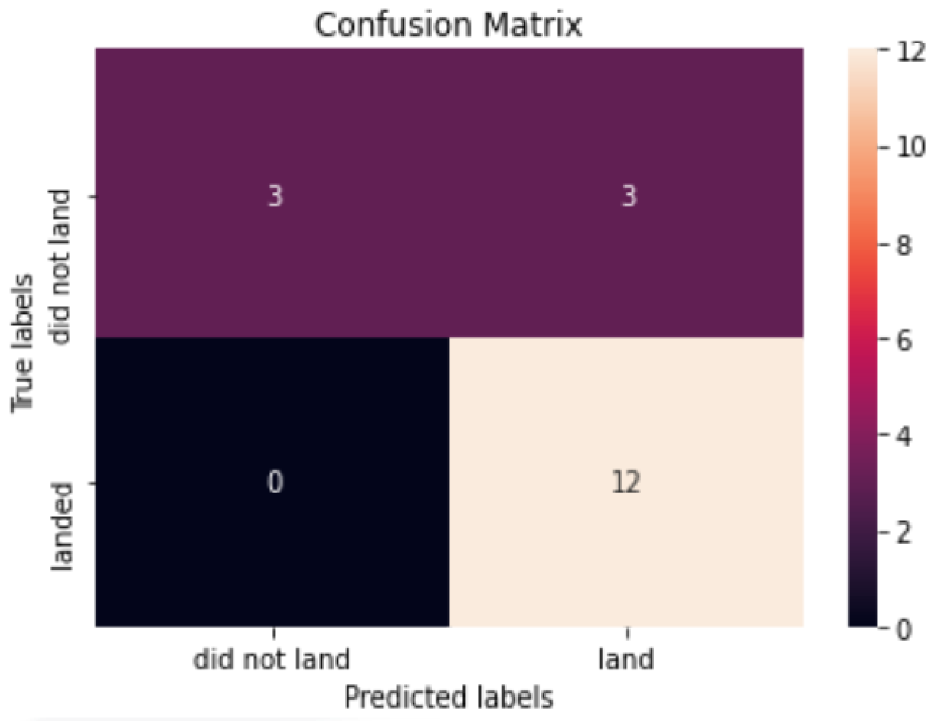
In [13]:
```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 0.1, 'penalty': 'l2', 'solve
r': 'lbfgs'}
accuracy : 0.8196428571428571
```

In [14]:
```
logreg_cv.score(X_test,Y_test)
```

Out[14]: `0.8333333333333334`

In [15]:
```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



RESULTS - Predictive Analysis

Decision Tree

In [16]:
```
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier()
```

In [17]:
```
tree_cv=GridSearchCV(tree,parameters,cv=10)
tree_cv.fit(X_train,Y_train)
```

Out[17]:
```
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']})
```

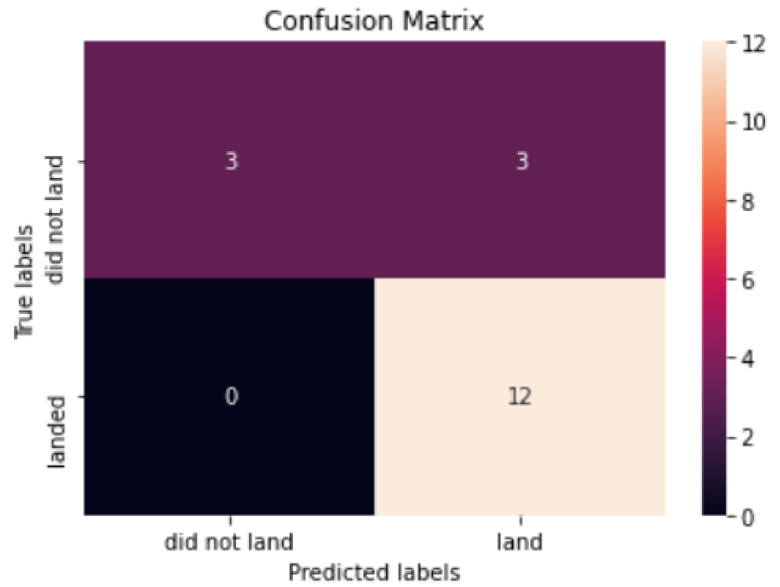
In [18]:
```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10, 'splitter': 'best'}
accuracy : 0.8892857142857145
```

In [20]:
```
tree_cv.score(X_test,Y_test)
```

Out[20]:
```
0.8333333333333334
```

In [22]:
```
yhat = tree_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



RESULTS - Predictive Analysis

K Nearest Neighbor (KNN)

```
In [23]: parameters = {'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
                       'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
                       'p': [1,2]}

         KNN = KNeighborsClassifier()
```

```
In [24]: knn_cv=GridSearchCV(KNN,parameters,cv=10)
         knn_cv.fit(X_train,Y_train)

Out[24]: GridSearchCV(cv=10, estimator=KNeighborsClassifier(),
                      param_grid={'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brut
         e'],
                                  'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
                                  'p': [1, 2]})
```

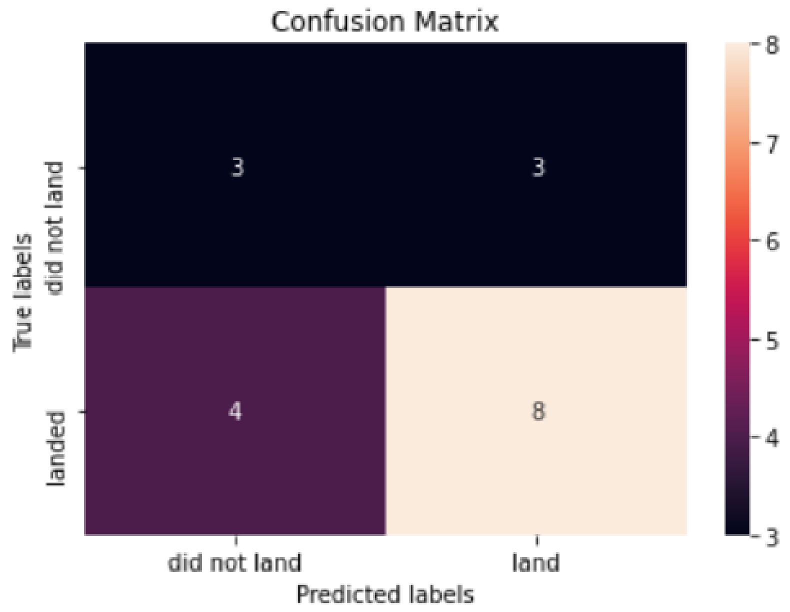
```
In [25]: print("tuned hpyerparameters :(best parameters) ",knn_cv.best_params_)
         print("accuracy :",knn_cv.best_score_)

         tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbor
         s': 3, 'p': 1}
         accuracy : 0.6642857142857143
```

```
In [26]: knn_cv.score(X_test,Y_test)

Out[26]: 0.6111111111111112
```

```
In [27]: yhat = knn_cv.predict(X_test)
         plot_confusion_matrix(Y_test,yhat)
```



RESULTS - Predictive Analysis

Best Method (?)

Logistic Regression

```
In [14]: logreg_cv.score(X_test,Y_test)
Out[14]: 0.8333333333333334
```

Decision Tree

```
In [20]: tree_cv.score(X_test,Y_test)
Out[20]: 0.8333333333333334
```

K Nearest Neighbor

```
In [26]: knn_cv.score(X_test,Y_test)
Out[26]: 0.6111111111111112
```

The results indicate that the Logistic regression and Descision Tree have the highest prediction accuracies

CONCLUSIONS

- Initial EDA showed the rate of success as 67%
- The launch site CCAFS LC-40 had a success rate of 60% while KSC LC-39A and VAFB SLC 4E had a success rate of 77%
- Orbits ES-L1, SSO, HEO, and GEO had the highest success rate
- The success rate since 2013 kept increasing till 2020
- Payload, Orbit, and Launch Sites were the main features controlling the success or failure flight.
- Logistic regression and classification trees machine learning algorithms resulted in highest accuracy of predictions (83%) while the K neighbor method resulted in the lowest prediction accuracy of 66%.

Discussion and Recommendations

- Feature engineering was performed based on EDA and engineering judgment. It is recommended to apply mathematical and modern techniques like Genetic Algorithm (GA) for feature selection.

- Although prediction accuracy of each machine learning algorithm was evaluated, further analysis on confusion matrix could be performed to provide insight on *sensitivity, recall, specificity,* and *precision* of each method.

- Further analysis and discussion is required to determine the feature(s) with the highest impact on the success of the flight. This feedback is critical for the engineering team as it help them understand what to be focused on for improvement and enhancing the rate of success.

- Although the project was about Falcon 9, but comparing the results with the companion booster like Falcon 1 can provide excellent engineering insight on the important parameters and features impacting the rate of success.

- An optimization analysis can be performed to suggest the best combination of features that provide the highest likelihood of success.

APPENDIX

number of launches on each site

```
In [9]:  # Apply value_counts() on column LaunchSite
         df['LaunchSite'].value_counts()

Out[9]:  CCAFS SLC 40    55
         KSC LC 39A      22
         VAFB SLC 4E     13
         Name: LaunchSite, dtype: int64
```

number and occurrence of each orbit

```
In [10]:  # Apply value_counts on Orbit column
          df['Orbit'].value_counts()

Out[10]:  GTO      27
          ISS      21
          VLEO     14
          PO        9
          LEO       7
          SSO       5
          MEO       3
          SO        1
          GEO       1
          HEO       1
          ES-L1     1
          Name: Orbit, dtype: int64
```

number and occurence of mission outcome per orbit type

```
In [11]:  # landing_outcomes = values on Outcome column
          landing_outcomes = df['Outcome'].value_counts()
          landing_outcomes

Out[11]:  True ASDS      41
          None None      19
          True RTLS      14
          False ASDS      6
          True Ocean      5
          None ASDS       2
          False Ocean     2
          False RTLS      1
          Name: Outcome, dtype: int64
```

GITHUB Repository Link

Please see all the completed Notebooks and Python files in my GitHub repository:

https://github.com/Shobeir-Gar/Applied-Data-Science-Capstone-Project