Capstone Project – SpaceX

Venu Chaikam
09/24/2021

OUTLINE

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

EXECUTIVE SUMMARY

- **Summary of methodologies**
  - ❖ Data collection
  - ❖ Data Wrangling
  - ❖ EDA with data visualization
  - ❖ EDA with SQL
  - ❖ Building with an interactive map with folium
  - ❖ Building Dashboard with Plotly Dashboard
  - ❖ Predictive Analysis
- **Summary of all results**
  - ❖ Exploratory data analysis results
  - ❖ Interactive analytics demo in screen shots
  - ❖ Predictive analysis results

INTRODUCTION

- **Project background and content**

The aim this project is to verify Falcon 9 first stage will land successfully. Space X advertises Falcon 9 rocket launches in its website, with cost of 62 million dollars; other space companies launch cost is 165 million. The Space X has less cost due to reusable first stage. In conclusion, if we can determine, if the first stage landing is success and its costs of launching. This information can be used by an alternating company wants to bid against Space X for a rocket launch.

- **Common problems we need to address:**

❖ What factors will influence for successful landing?

❖ The effect of relationship with certain rocket variables will impact in determining success rate of landing

❖ What conditions does Space X have to achieve to get the best results and best rocket success landing

METHODOLOGY

- **Data Collection methodology**
  - SpaceX Rest API
  - Web scrapping from Wikipedia
- **Performed data wrangling (Transforming data)**
  - One hot encoding data fields for Machine Learning and dropping irrelevant columns
- **Performed Exploratory Data Analysis (EDA) using visualization and SQL**
  - Plotting: Scatter Graphs, Bar Graphs to show relationship between variables to show patterns of data
- **Performed Interactive visual analytics using Folium and Plotly Dash**
- **Performed predictive analysis using classification models**
  - How to build, tune, evaluate classification Models

Data Collection - API

- Data is collected using Space X REST API URL “spacex_url=https://api.spacexdata.com/v4/launches/past
- This API will give information about launches, rocket used payload delivered, launch specifications, and landing outcome
- After filtering the data stored only Falcon 9 rocket information into the Data Frame data_faclcon9
- Data is missed in some of the rows, to fill the missed I have captured mean() of PayloadMass and filled with missing data for PayloadMass column
- GitHub link for Data collection

  https://github.com/vchaikam1/Capstone_SpaceX

Data Collection – Web Scrapping

- Falcon 9 Data is collected from Wikipedia HTML file https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
- Extracted the data using Beautiful Soup
- Parsed the HTML data into table List using only Table Header “TH” fields. From the List created Data Dictionary
- Finally data stored into Data Frame and .CSV file
- GitHub link for Web Scrapping

  https://github.com/vchaikam1/Capstone_SpaceX

Data Wrangling

In the dataset, there are several different cases where booster didn't land successfully, sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to specific region of the ocean. True RTLS means the mission outcome was successfully landed to ground pad, False RTLS means the mission outcome was unsuccessfully landed to ground pad. True ASDS means the mission outcome was successfully landed on drone ship, False ASDS means the mission outcome was unsuccessfully landed on drone ship.

Here I have converted the landing outcomes into Training labels '1' is successful landing; '0' is unsuccessful landing

Stored the landing outcome into .CSV file

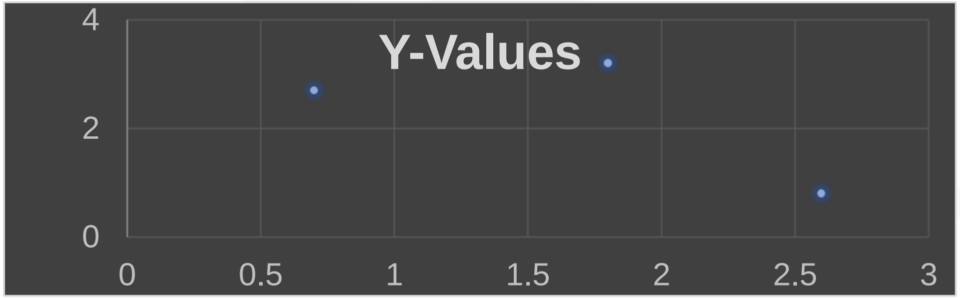
GitHub link:

https://github.com/vchaikam1/Capstone_SpaceX

EDA – Data Visualization

The following Scatter Graphs were drawn to establish the correlation between two variables. How much one variable is effected by the other.

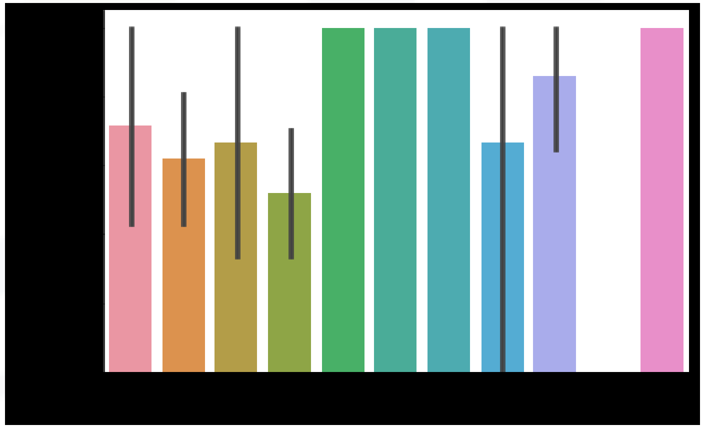
- Flight Number vs. PayloadMass
- Flight Number vs. Launch Site
- Pay Load vs. Launch Site
- Orbit vs. Flight Number
- Payload vs. Orbit Type
- Orbit Vs. PayloadMass



Plotted Bar Graph:

Mean Vs. Orbit

A bar diagram is easy to compare sets of data between different groups at a glance. The graph represents categories on one axis and discrete value on the other axis. This graph also shows big changes in the data overtime.

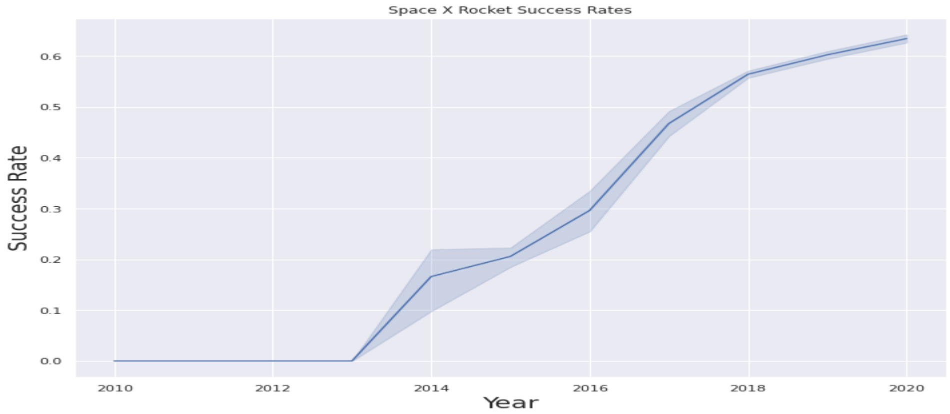


EDA – Data Visualization

Line Graphs:

Success Rate Vs. Year

Line graphs are useful in way to show data variables and trends very clearly and can help to make predictions about results not yet recorded.



GitHub Repository for Data Visualization:

https://github.com/vchaikam1/Capstone_SpaceX

EDA – With SQL

Performed SQL to gather information about SpaceX TBL dataset

The following SQL queries have executed with SpaceX table dataset

1) **Display the names of the unique launch sites in the space mission**
2) **Display 5 records where launch sites begin with the string 'CCA'**
3) **Display the total payload mass carried by boosters launched by NASA (CRS)**
4) **Display average payload mass carried by booster version F9 v1.1**
5) **List the date when the first successful landing outcome in ground pad was achieved.**
6) **List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000**
7) **List the total number of successful and failure mission outcomes**

EDA – With SQL

Performed SQL to gather information about SpaceX TBL dataset

The following SQL queries have executed with SpaceX table dataset

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

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

10) 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

GitHub Repository for Data Visualization:

https://github.com/vchaikam1/Capstone_SpaceX

Building an Interactive map with Folium

To Visualize the Lunch data into interactive map. I have used the Latitude and Longitude coordinates at each launch site and added a circle marker around each lunch site with a label of name of the launch site.

Assigned the data frame launch outcomes to classes 0 and 1 with Green and Red markers on the map in a MarkerCluster()

Using Haversine's formula to calculate the distance from the launch site to various landmarks to find various trends about what is around launch site to measure patterns. Lines are drawn on the map to measure distance to land marks

- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

GitHub Repository for Launch site analysis with Folium:

https://github.com/vchaikam1/Capstone_SpaceX

Predictive Analysis (Classification)

Building Model

- Load Dataset into NumPy and Pandas
- Transform Data
- Split data into Training and Test Datasets
- Check how many test samples we have
- Decide which type of Machine Learning algorithms to use
- Set parameters and algorithms to GridSearchCV
- Fit datasets into GridSearchCV objects and train dataset

Evaluating Model

- Check accuracy of each model
- Get tuned hyper parameters for each type of algorithms
- Plot confusion matrix

Predictive Analysis (Classification)

Improving Model

- Feature Engineering
- Algorithm Tuning

Finding the best performing classification model

- The model with the best accuracy score wins the best performing model

GitHub Repository for Machine Learning Predication:

https://github.com/vchaikam1/Capstone_SpaceX

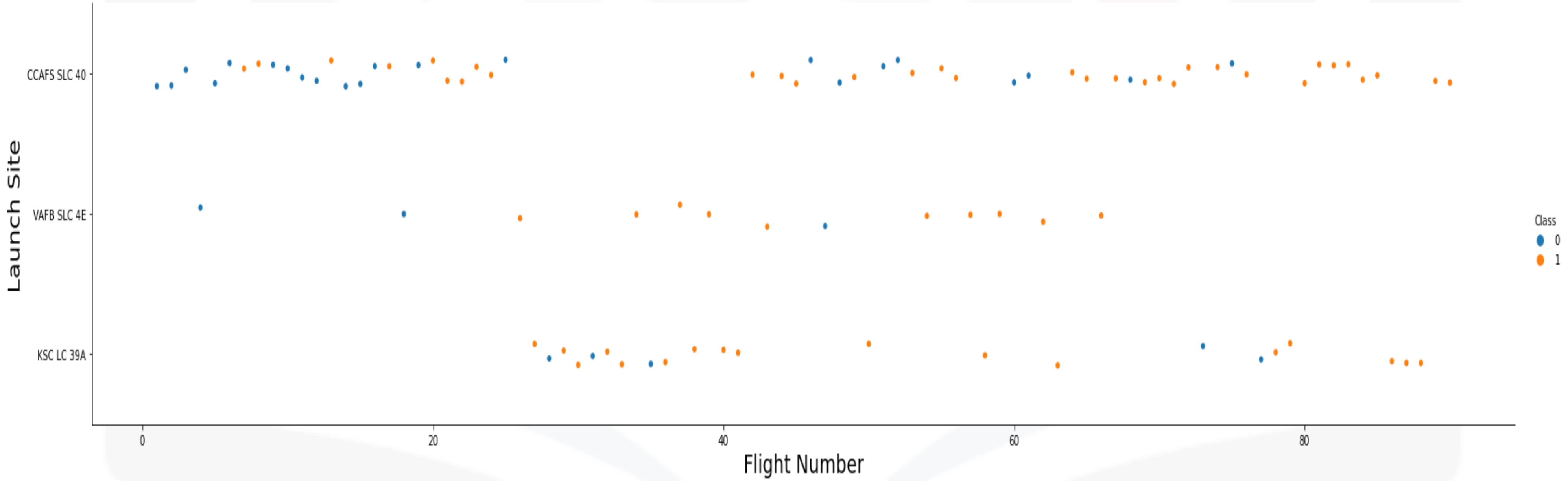
Results

- Exploratory Data Analysis with Visualization Results
- Exploratory Data Analysis with SQL Results
- Interactive Analytics Demo Screen Shots
- Predictive Analysis Results

EDA Visualization

Flight Number Vs. Launch Site

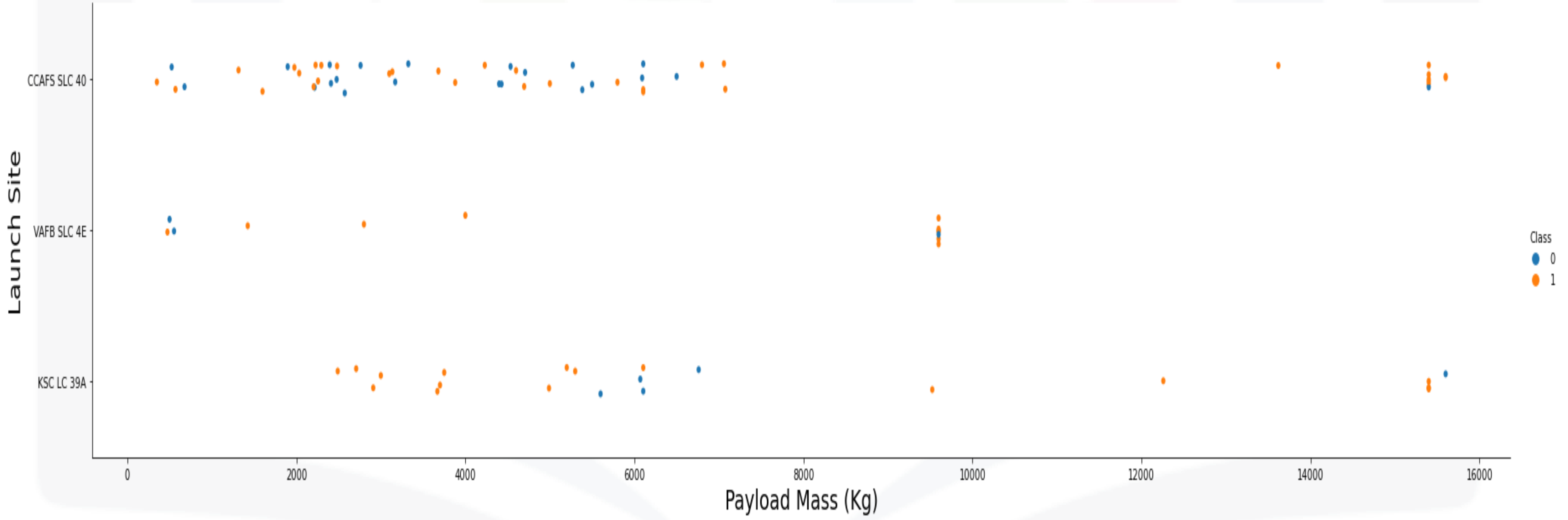
The more amount of flight at a Launch site the greater the success rate at Launch site



EDA Visualization

Payload Mass Vs. Launch Site

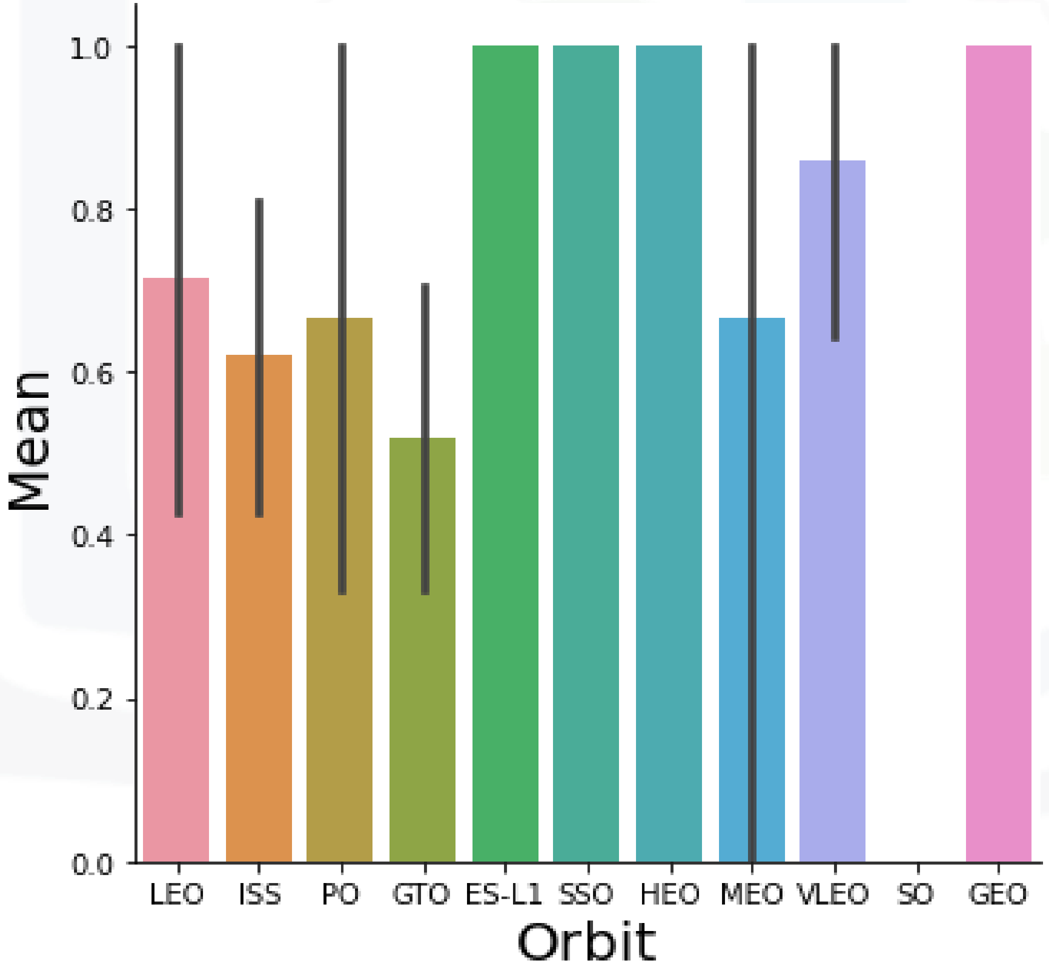
The greater the payload mass for launch site CCAFS SLC 40 the higher the success rate for the Rocket.



EDA Visualization

Success Rate Vs. Orbit

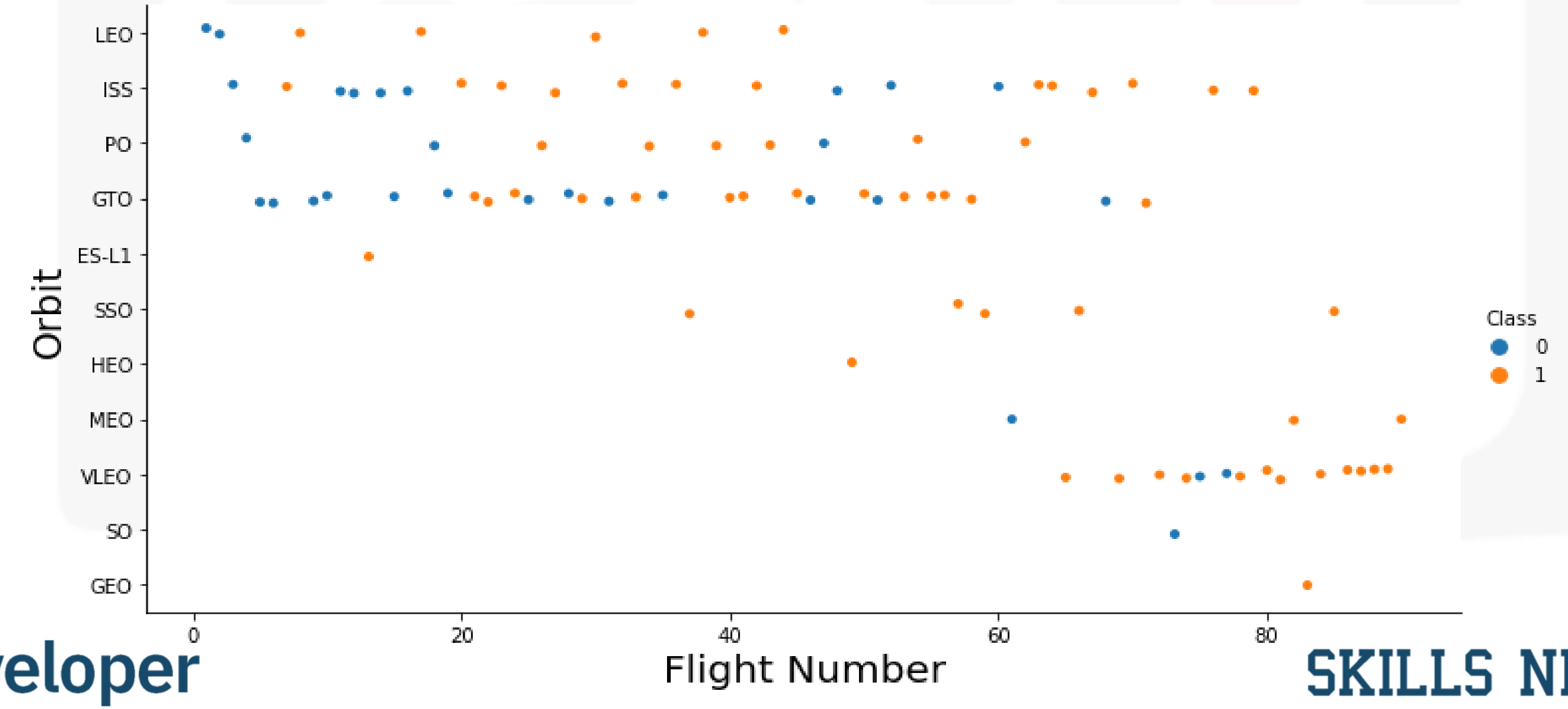
Orbit GEO, HEO, SEO ES-L1 has the best success rate



EDA Visualization

Flight Number Vs. Orbit Type

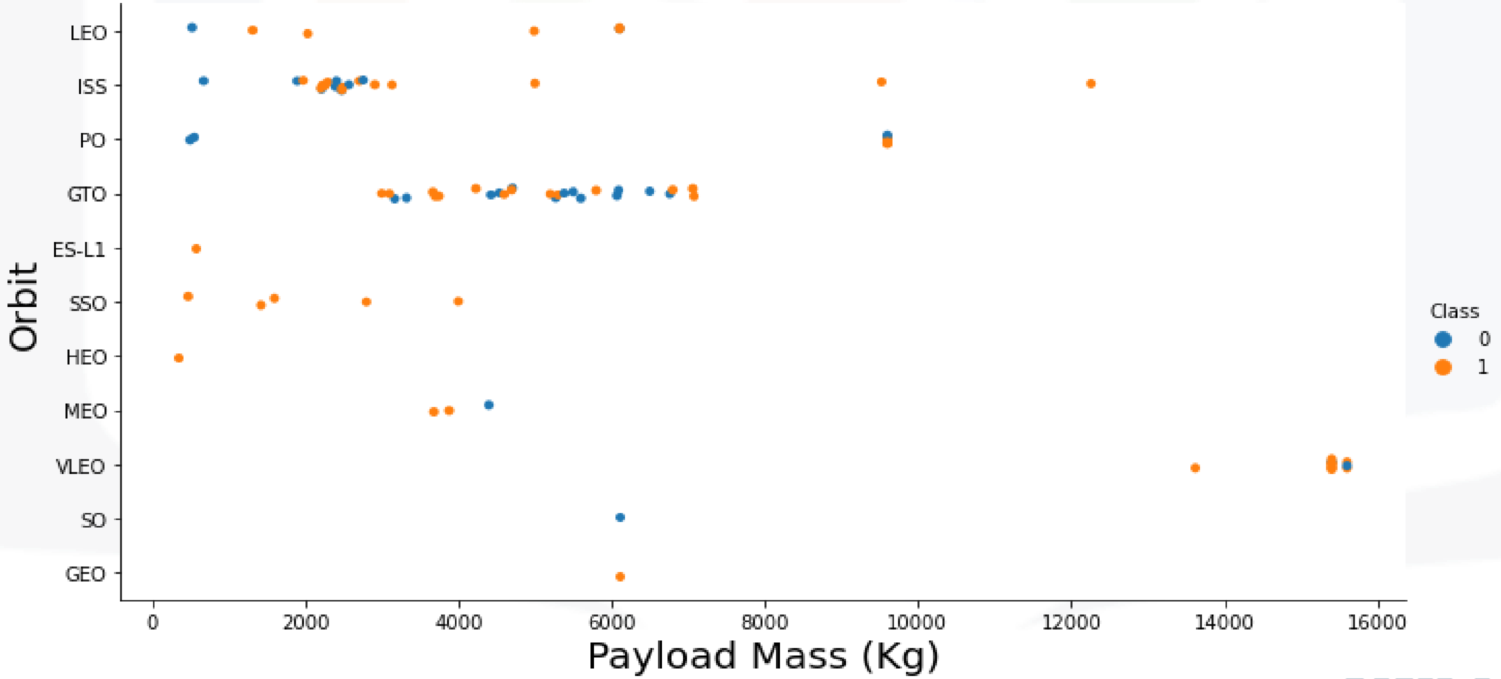
In the LEO orbit the success appears related to the number of flights; on the other hand there is no relationship between Flight Number when in GTO orbit



EDA Visualization

Payload Mass Vs. Orbit Type

Heavy payloads have negative influence on GTO orbit and positive on GTO and LEO and Polar ISS orbit

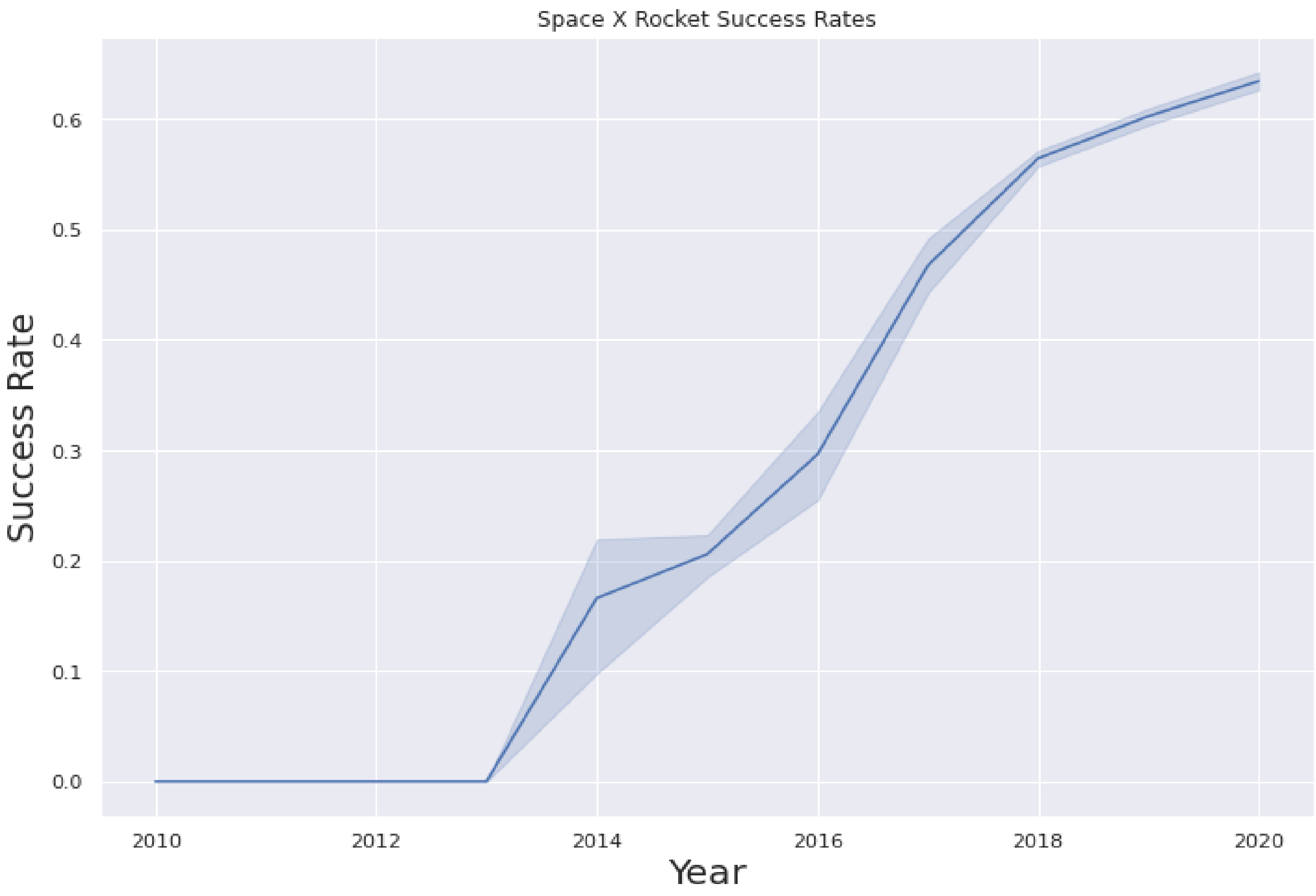


EDA Visualization

Launch success yearly trend

The success rate is increasing

Since 2013 onwards



EDA SQL

Unique Launch Sites

Query: SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL;

	LAUNCH_SITE
0	CCAFS LC-40
1	CCAFS SLC-40
2	KSC LC-39A
3	VAFB SLC-4E

Using DISTINCT in the query, it will list only unique Launch Sites

EDA SQL

Top 5 rows where Launch site begin with 'CCA'

Query: SELECT * FROM SPACEXTBL where LAUNCH_SITE like 'CCA%' LIMIT 5;

	DATE	TIME__UTC_	BOOSTER_VERSION	LAUNCH_SITE	PAYLOAD	PAYLOAD_MASS__KG_	ORBIT	CUSTOMER	MISSION_OUTCOME	LANDING__OUTCOME
0	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
1	2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of...	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2	2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
3	2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
4	2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

LIKE, 'CCA%' with LIMIT option will retrieve Launch site names begin with 'CCA'

EDA SQL

Total Payload Mass carried by boosters launched by NASA (CRS)

Query: SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL
WHERE CUSTOMER = 'NASA (CRS)‘;

	1
0	45596

Using SUM() and Where clause we can get the Total Payload Mass carried by NASA (CRS)

EDA SQL

Average payload mass carried by booster version F9 v1.1

Query: SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL
WHERE BOOSTER_VERSION = 'F9 v1.1';

	1
0	2928.4

Using AVG() and Where clause we can get the Average Payload Mass carried by Booster Version ‘F9 v1.1’

EDA SQL

Date when the first successful landing outcome in ground pad is achieved

Query: SELECT MIN (DATE) FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Success (ground pad)';

	1
0	2015-12-22

Using MIN() and Where clause we can get the FIRST DATE of success landing on ground pad

EDA SQL

Names of the Boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
Query: SELECT  BOOSTER_VERSION FROM SPACEXTBL
               WHERE LANDING__OUTCOME = 'Success (drone ship)' AND
               PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

	BOOSTER_VERSION
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Using WHERE AND > , < we can get the all booster versions having Payload Mass between 4000 and 6000 KGs and successful landing on drone ship

EDA SQL

Total number of successful and failure mission outcomes

```
Query: SELECT  COUNT(*) as SUCCSFUL FROM SPACEXTBL
               WHERE MISSION_OUTCOME  LIKE  '%Success%‘;
       SELECT  COUNT(*) AS FAILURE FROM SPACEXTBL
               WHERE MISSION_OUTCOME  LIKE  'Fail%';
```

	SUCCSFUL
0	100.0

	FAILURE
0	1.0

Using COUNT(*) and LIKE ‘%’ we can get the total number success and failures

EDA SQL

Names of the booster versions which have carried the maximum Payload Mass

Query: SELECT BOOSTER_VERSION, PAYLOAD_MASS__KG_ FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);

Using WHERE clause and Subquery with MAX()

We can get Booster Version and It's Maximum Payload Mass

	BOOSTER_VERSION	PAYLOAD_MASS__KG_
0	F9 B5 B1048.4	15600
1	F9 B5 B1049.4	15600
2	F9 B5 B1051.3	15600
3	F9 B5 B1056.4	15600
4	F9 B5 B1048.5	15600
5	F9 B5 B1051.4	15600
6	F9 B5 B1049.5	15600
7	F9 B5 B1060.2	15600
8	F9 B5 B1058.3	15600
9	F9 B5 B1051.6	15600
10	F9 B5 B1060.3	15600
11	F9 B5 B1049.7	15600

EDA SQL

Failed landing outcomes in drone ship, their Booster Versions and Launch Site names for in year 2015

Query: SELECT BOOSTER_VERSION, LAUNCH_SITE, LANDING__OUTCOME,

YEAR(DATE) AS YR FROM SPACEXTBL

WHERE LANDING__OUTCOME = 'Failure (drone ship)'

AND YEAR(DATE) = '2015';

	BOOSTER_VERSION	LAUNCH_SITE	LANDING__OUTCOME	YR
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)	2015
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)	2015

Using Where clause AND YEAR() we can get the Booster Version, Launch site, Failure Landing on drone ship

EDA SQL

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

```
Query: SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) AS LAND_CNT
       FROM SPACEXTBL WHERE DATE > '2010-06-04' AND DATE < '2017-03-20'
       GROUP BY(LANDING__OUTCOME) ORDER BY LAND_CNT DESC;"
```

Using Count(), WHERE Clause, GROUP BY() ORDER BY DESC we can get the Count of Landing outcome between '2010-06-04' AND '2017-03-20' in Descending Order

	LANDING__OUTCOME	LAND_CNT
0	No attempt	10.0
1	Failure (drone ship)	5.0
2	Success (drone ship)	5.0
3	Controlled (ocean)	3.0
4	Success (ground pad)	3.0
5	Uncontrolled (ocean)	2.0
6	Failure (parachute)	1.0
7	Precluded (drone ship)	1.0

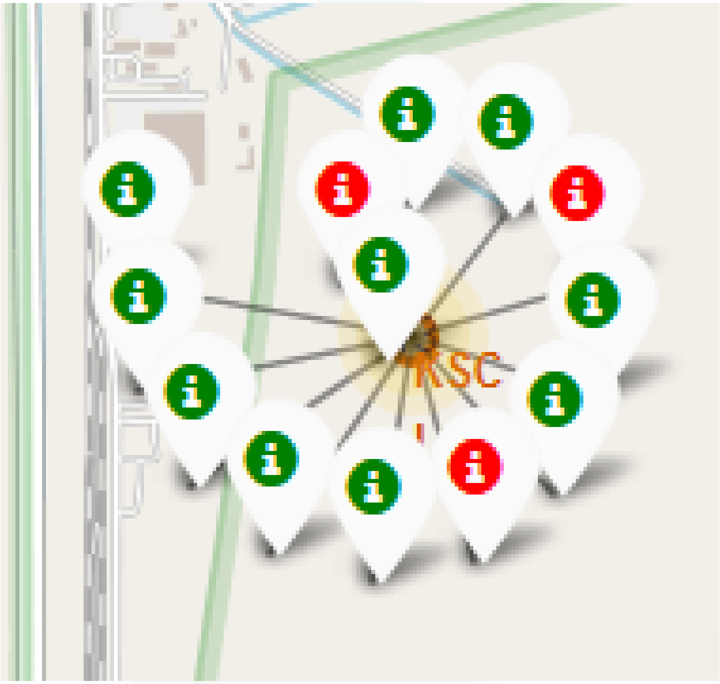
Interactive Map with Folium

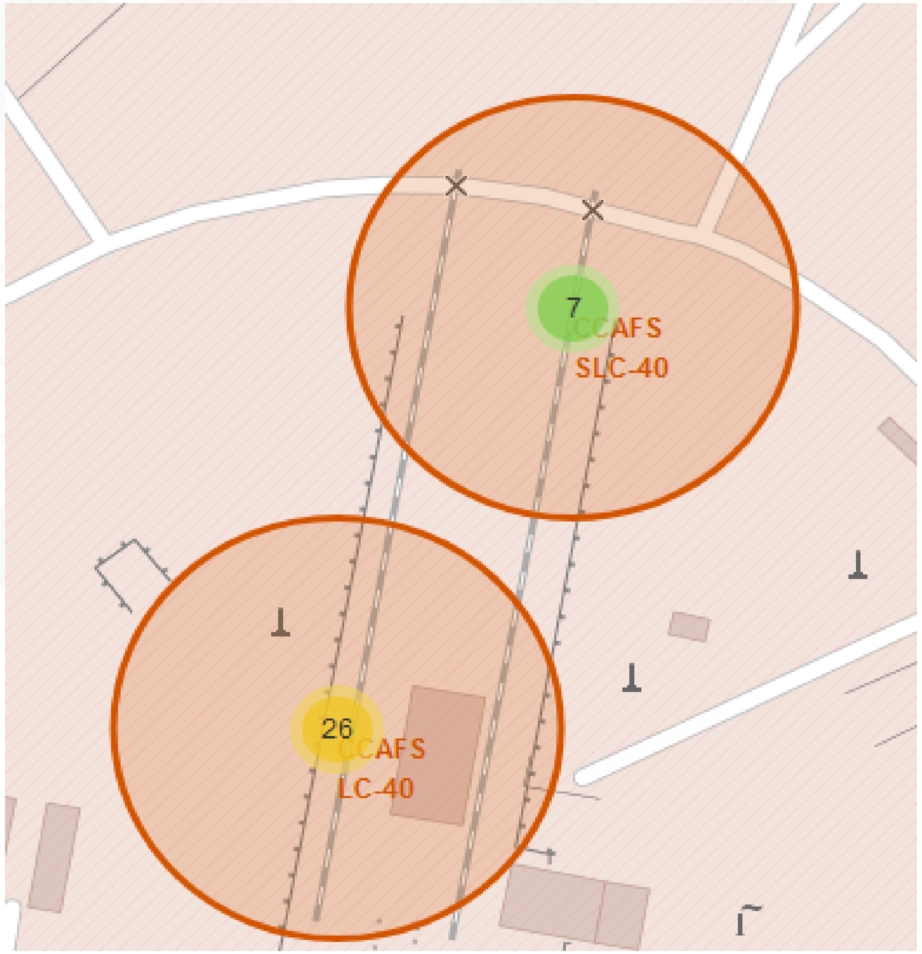
All Launch sites in global maps: The Launch sites are in USA costs on Florida and California

Interactive Map with Folium

Colored Labelled Markers : Florida Launch sites: Green Success, Red Failure

KSC LC-39A, CCAFS SLC-40, CCAFS LC-40

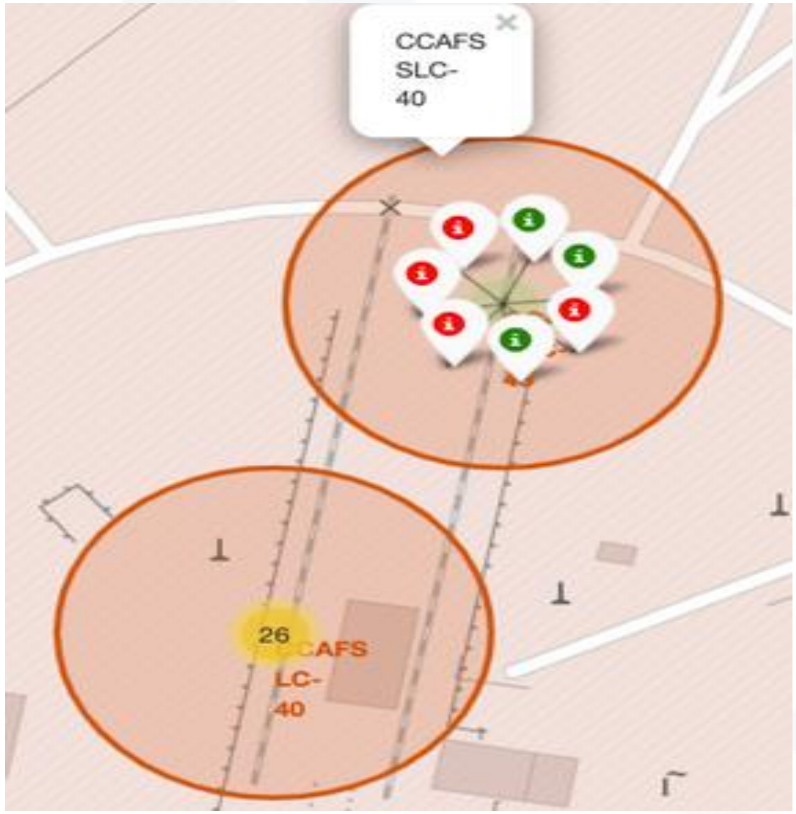




Interactive Map with Folium

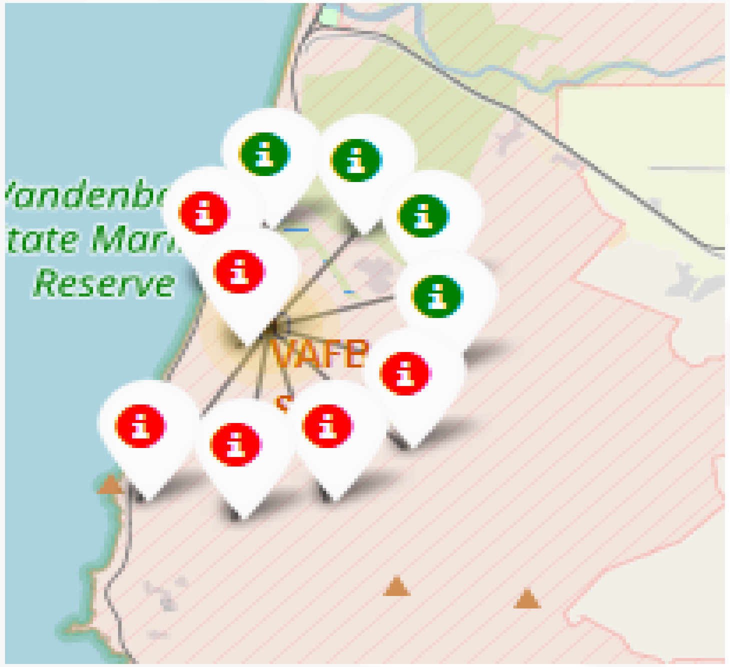
Colored Labelled Markers : Florida Launch sites: Green Success, Red Failure

KSC LC-39A, CCAFS SLC-40, CCAFS LC-40



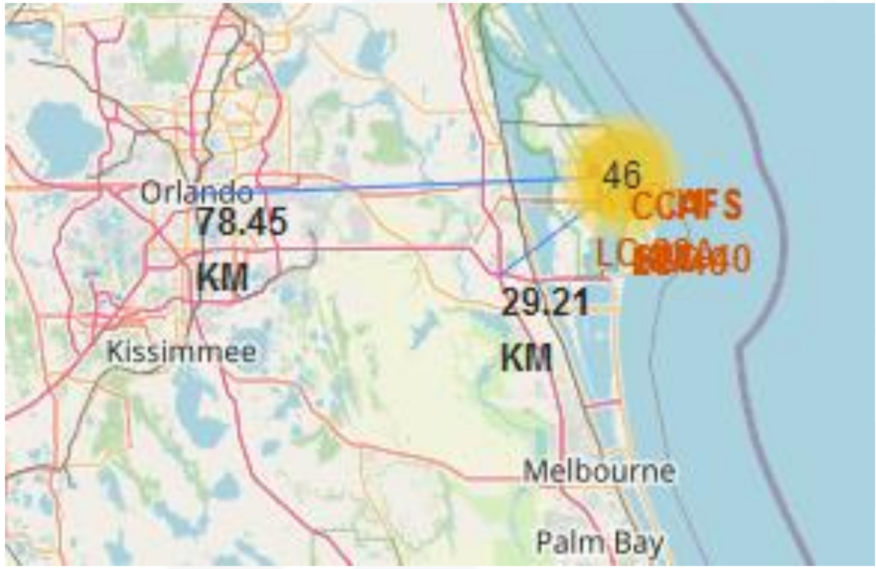
Interactive Map with Folium

Colored Labelled Markers : California Launch sites: Green Success, Red Failure VAFB SLC-4E



Interactive Map with Folium

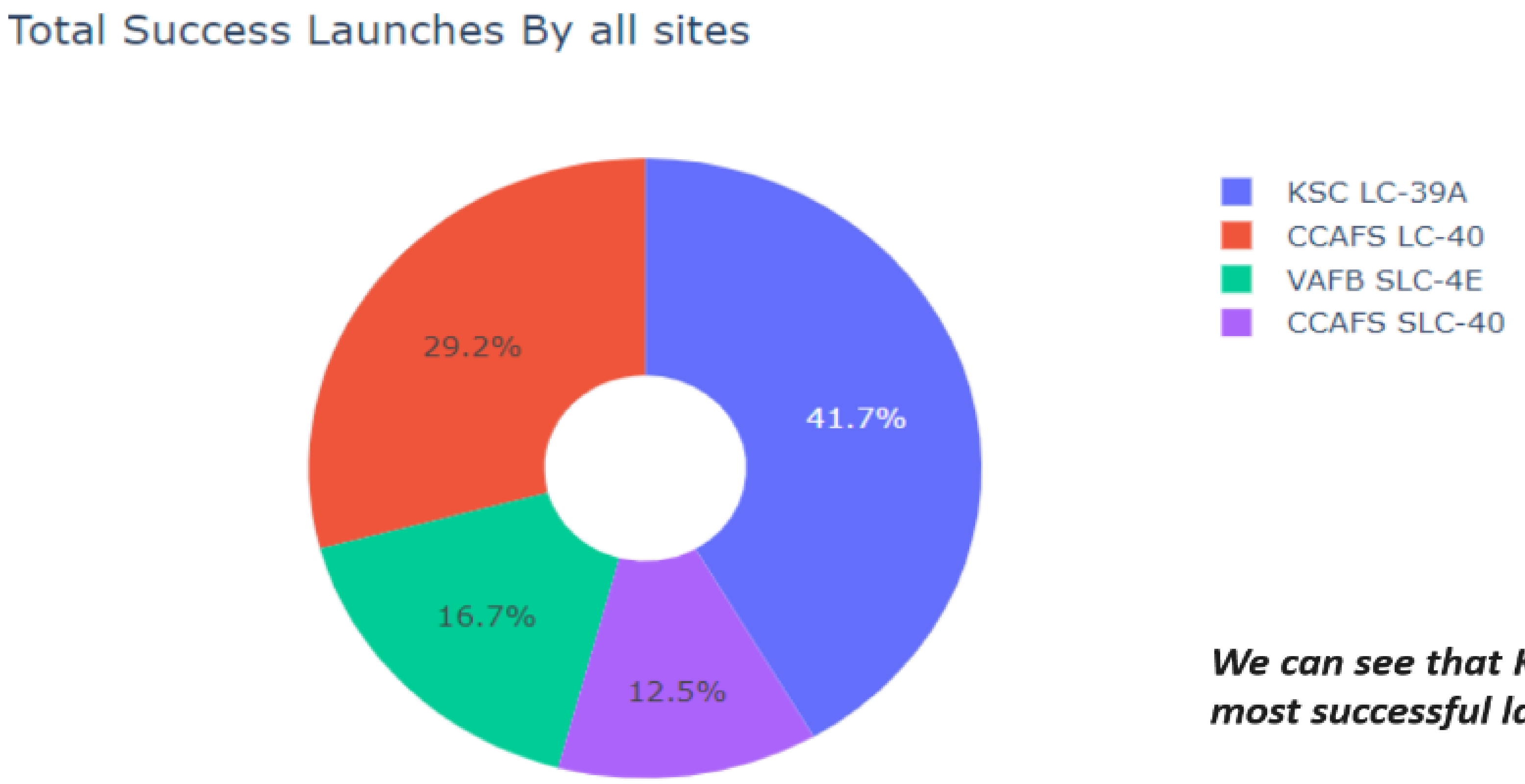
Distance to Various Places



- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Dashboard with Plotly Dash

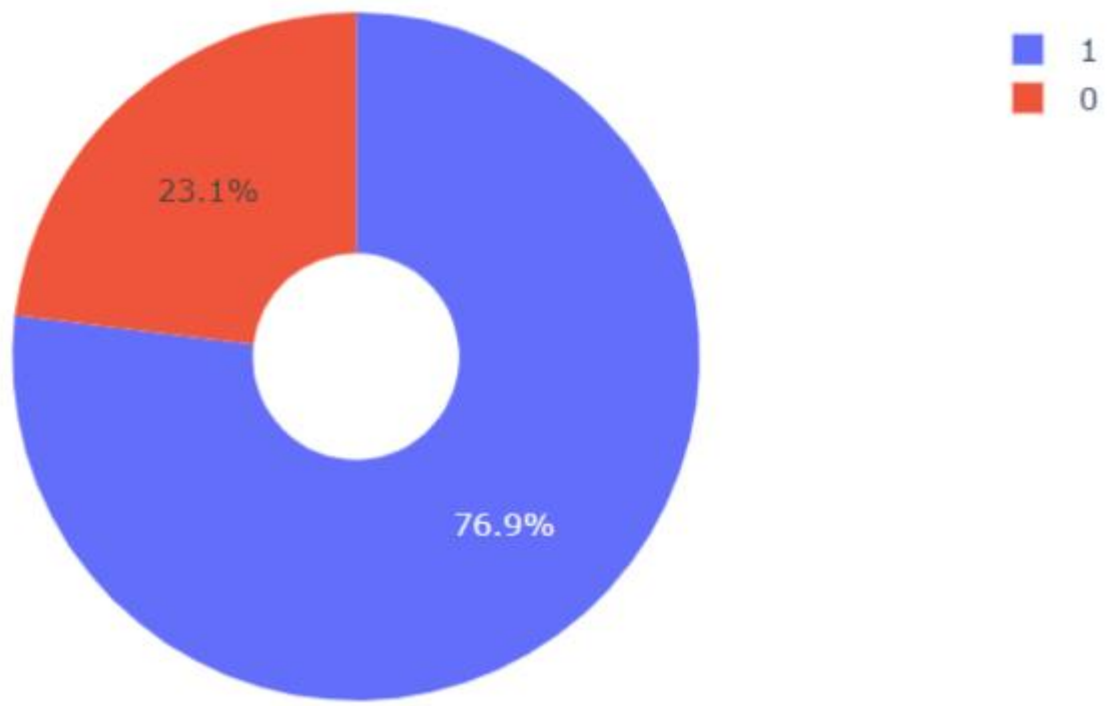
Dash Board – Pie Chart showing the success percentage achieved by each launch site



We can see that KSC LC-39A had the most successful launches from all the sites

Dashboard with Plotly Dash

Dash Board – Pie Chart for the launch site with highest launch success ratio



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Predictive Analysis (Classification)

Logistic regression object and then create a GridSearchCV object

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

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

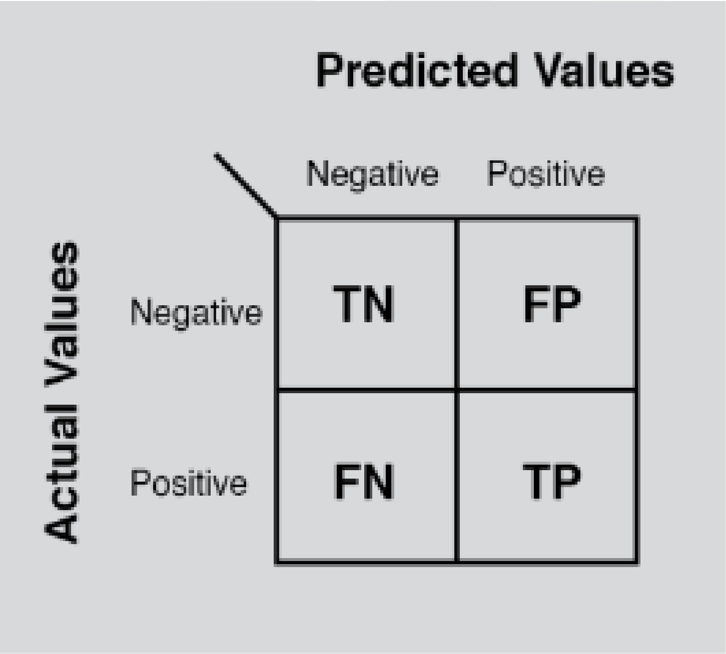
```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

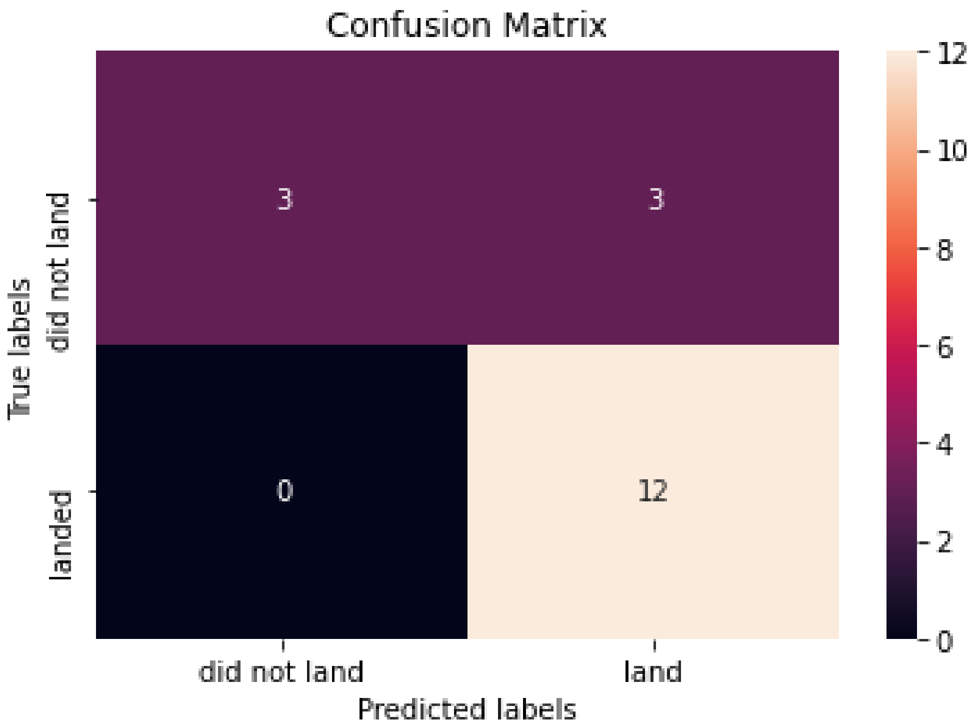
```
print('Accuracy=  ',logreg_cv.score(X_test,Y_test))
```

```
Accuracy=   0.8333333333333334
```

Predictive Analysis (Classification)

Logistic regression confusion matrix





logistic regression can distinguish between the different classes. We see that the major problem is false positives.

Predictive Analysis (Classification)

k nearest neighbors object then create a GridSearchCV

```
gscv = GridSearchCV(svm,parameters,scoring='accuracy',cv=10)
svm_cv = gscv.fit(X_train,Y_train)
```

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

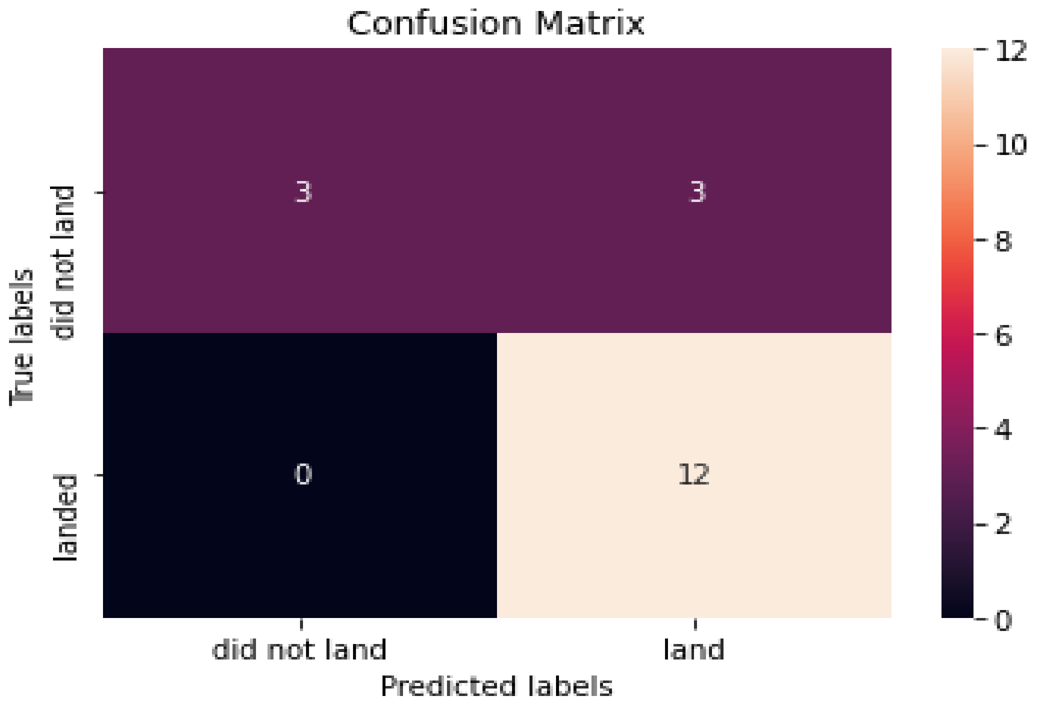
```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

```
print("accuracy: ",svm_cv.score(X_test,Y_test))
```

```
accuracy:  0.8333333333333334
```

Predictive Analysis (Classification)

[Support Vector Machine confusion matrix]



Predictive Analysis (Classification)

Decision Tree classifier object then create a GridSearchCV object

```
gscv = GridSearchCV(tree,parameters,scoring='accuracy',cv=10)
tree_cv = gscv.fit(X_train,Y_train)
```

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

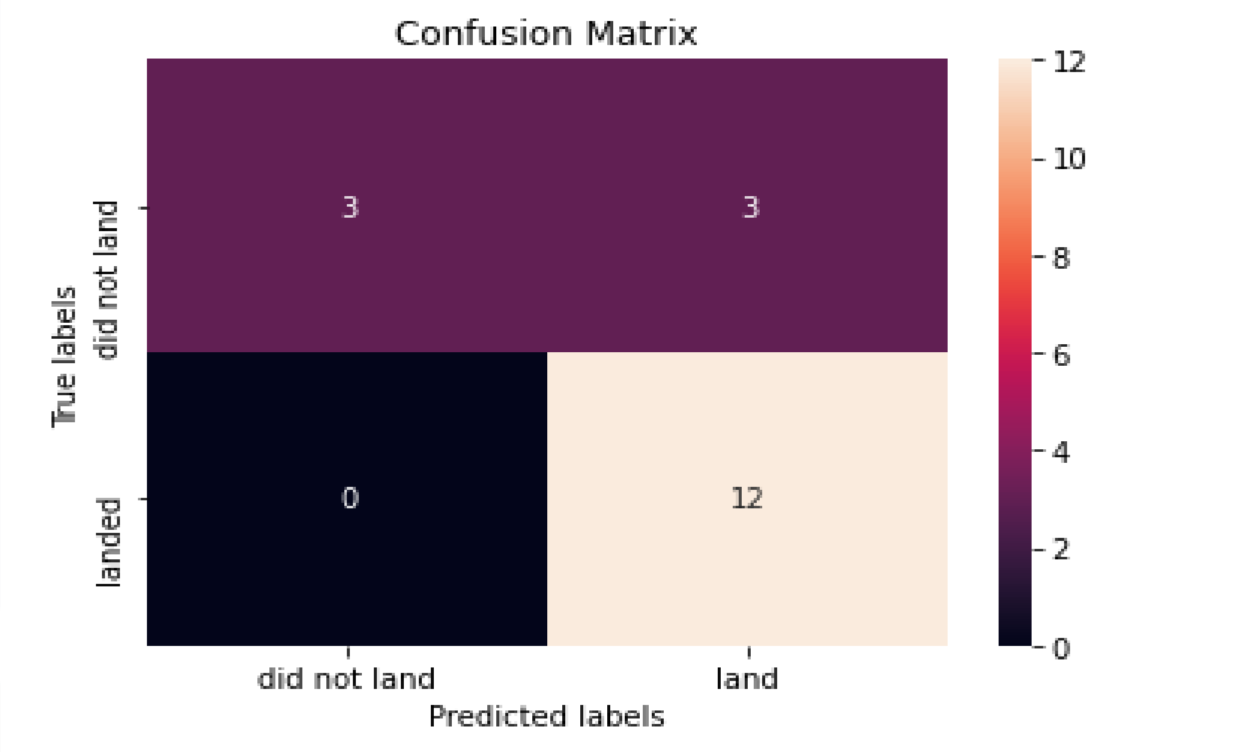
```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'random'}
accuracy : 0.8875
```

```
print("accuracy: ",tree_cv.score(X_test,Y_test))
```

```
accuracy:  0.8333333333333334
```

Predictive Analysis (Classification)

Decision tree confusion matrix



Predictive Analysis (Classification)

k nearest neighbors object then create a GridSearchCV

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

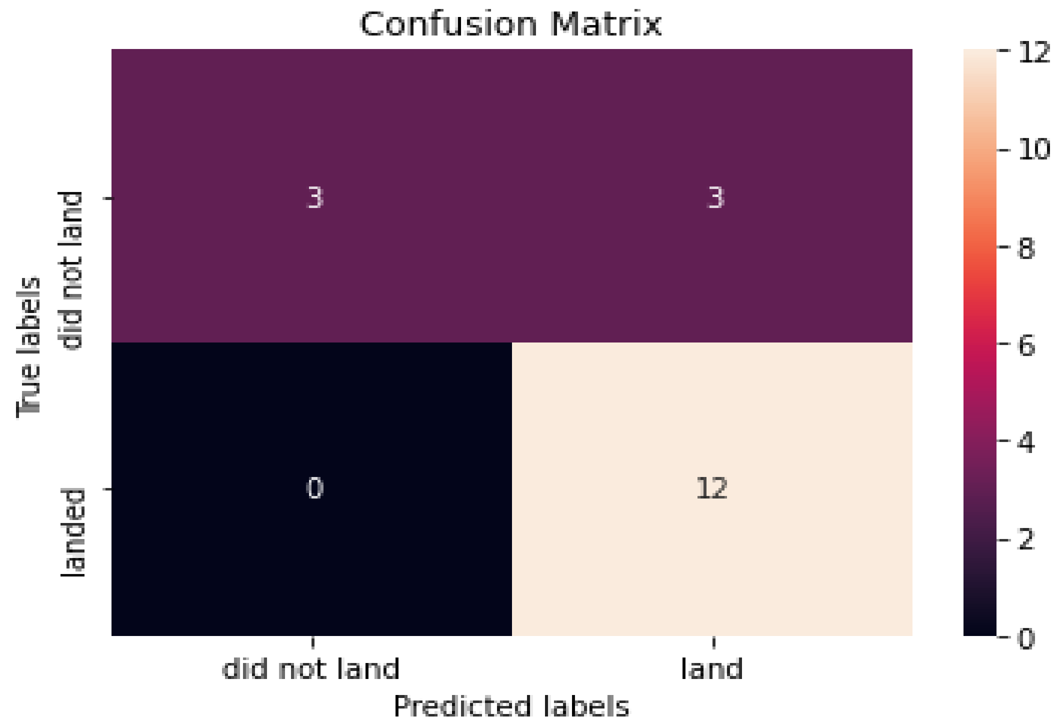
```
tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
accuracy : 0.8482142857142858
```

```
print("accuracy: ",knn_cv.score(X_test,Y_test))
```

```
accuracy:  0.8333333333333334
```

Predictive Analysis (Classification)

[K nearest confusion matrix]



Best Algorithm is Tree with a score of 0.8875

Predictive Analysis (Classification)

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.8875
Best Params is : {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'split
ter': 'random'}
```

Best Algorithm is Tree with a score of 0.8875

After selecting the best parameters for decision tree classifier using the validation data, we achieved 83.33% accuracy on the test data

DISCUSSION

- We have tested only SpaceX Falcon 9 Rocket. There might be other rockets which have launched by Space X.
- These Rockets will be evaluated and tested in the future projects

CONCLUSION

- Low weighted payloads performed better than heavier payloads
- The success rates for SpaceX launch is directly proportional time in years they will perfect the launches
- We can note that KSC LC-39A had the most successful launches from all the sites
- Orbit GEO, HEO, SSO, ES-L1 has the best success rate
- The Tree Classifier Algorithm is the best for Machine Learning for given dataset

APPENDIX

- Beautiful Soap for Web Scrapping
- Haversine Formula for distance calculation
- Folium for interactive maps