Jorge Castro M

Data Science and Machine Learning Capstone Project

Content

- Summary
- Introduction
- Methodology
- Results
- Conclusion
- References

Summary

- In this project, data has been collected from the most successful company of commercial flights to space "spaceX". Knowledge is obtained through the application of different analysis laboratories, with an emphasis exploratory on statistical analysis and data visualization. In short, the data was organized and cleaned. Furthermore, the available data was interrelated, queried and compiled using SQL. Subsequently, some preliminary correlations between launch site and success rates were discovered, with interactive visual analyses. Finally, the data is disaggregated by dividing it into groups defined by categorical variables or factors. Different predictive models are evaluated and refined to uncover more insights.

Introduction

- Different laboratories are carried out and it is sought to determine if by using techniques such as data analysis, data visualization and machine learning with Python. It is possible to increase the success of the company's Falcon 9 first stage launches, with the reuse of SpaceX launches.

The notebooks covered are:

- Data collection
- Data wrangling
- Exploratory Data Analisys and interactive visual analytics methodology related slides
- Exploratory Data Analisys with visualization results
- Exploratory Data Analisys with SQL
- Interactive map with Follium
- Predictive analysis (classification)

Methodology

- Data collection: RESTful API and web scraping are used to get the data columns.
- Spacex API: Different methods are used like: .json(), .json_normalize(), getLaunchSite(),getPayloadData(),getCoreData(), create a data frame, filter missing data and other values to export to csv.
- Web scraping: Create a BeatifulSoup from the html response, all columns are extracted from the html table header, data is collected by parsing, create a dataframe by parsing the starting HTML tables, and finally a frame is created data, to export it to csv .

Data wrangling: data is cleaned and relevant information is obtained.

- The value_counts() method is used on the LaunchSite column to determine the number of launches on each site.
- The .value_counts() method is used to determine the number and occurrence of each orbit in the Orbit column.
- The .value_counts() method is used on the result column to determine the number of landing results.
- It is then assigned to a landing_outcomes variable. Using the result, a list is created bad_outcome and landing_class are assigned.
- Finally, a landing result tag is created from the Result column.

EDA with visualization: Exploration and data preparation the data were graphed y compared.

- The relationship between the flight number and the launch site is displayed.
- The relationship between the payload and the launch site is displayed.
- The relationship between the success rate of each type of orbit is displayed.
- The relationship between the flight number and the orbit type is displayed.
- The relationship between the payload and the type of orbit is displayed. The annual trend of launch success is displayed.

EDA with SQL: Different queries are made showing names, lists and ranking

- Queries are run for names of launch sites unique to the space mission, records where launch sites begin with the string 'KSC‘.
- Showing the total and average payload mass carried by boosters launched by NASA (CRS).
- Indicating the date on which the first successful landing in a drone was achieved.
- Listing the names of thrusters that are successful on the ground platform and have a charge mass greater than 4,000 but less than 6,000.

- Listing the total number of successful and failed mission results.
- Listing the records that will show the names of the months, successful landing results on the ground platform, booster versions, launch site for the months of the year 2017.
- Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

Map with the Folium results/Launch Sites Locations Analysis with Folium

- You will perform a more interactive visual analysis using `Folium`.
- Plot all the launch sites on a map.
- Create a Folio Map object, with an initial center location to be NASA's Johnson Space Center in Houston, Texas. Use folium-circle to add a highlighted circle area with a text label at a specific coordinate.
- Mark successful/unsuccessful launches for each site on the map.
- Calculate the distances between a launch site and its surroundings.

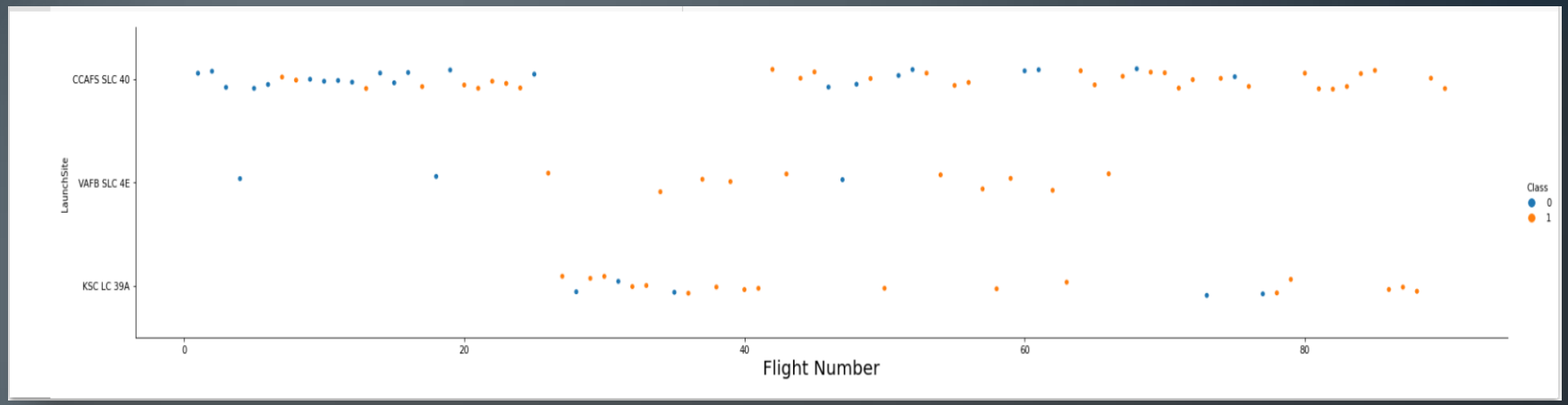
Predictive analysis /Machine Learning Prediction (classification

- Perform exploratory data analysis and determine training labels.
- Create a column for the class.
- Standardize the data.
- Split into training data and test data.
- Find the best hyperparameter for SVM, classification trees, logistic regression, and k nearest neighbor objects.
- Find the method that works best using test data

Results

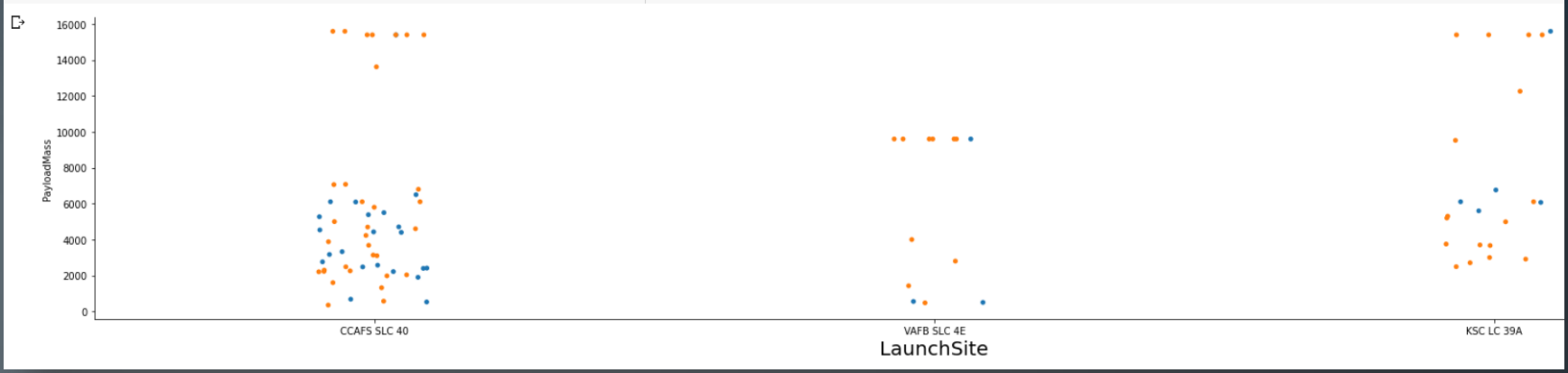
Exploratory Data Analysis

Flight Number and Launch Site



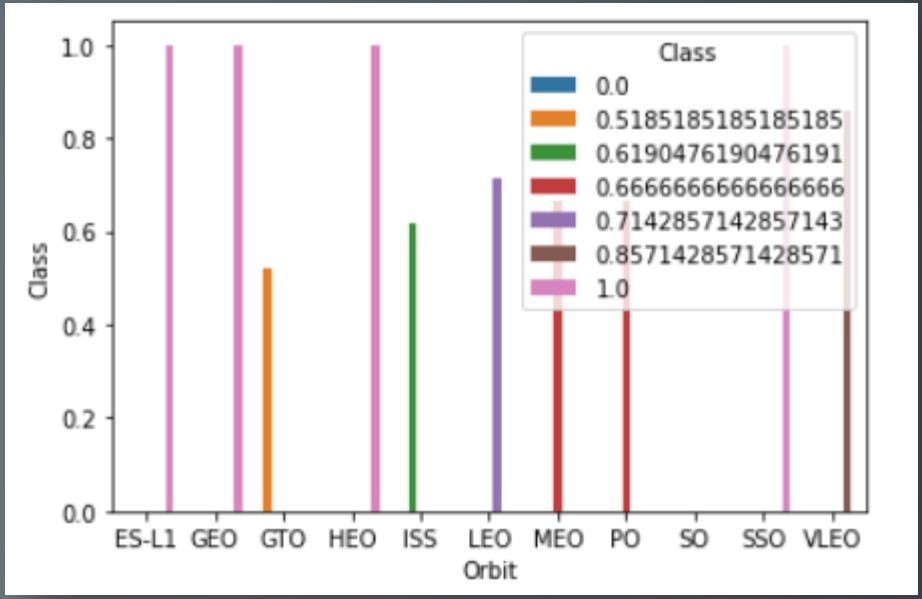
- On the left end there are failed flights and on the right there are successful flights.
- Most flight sites are from CCAFS SLC 40
- In general, successful returns increased

Payload and Launch Site



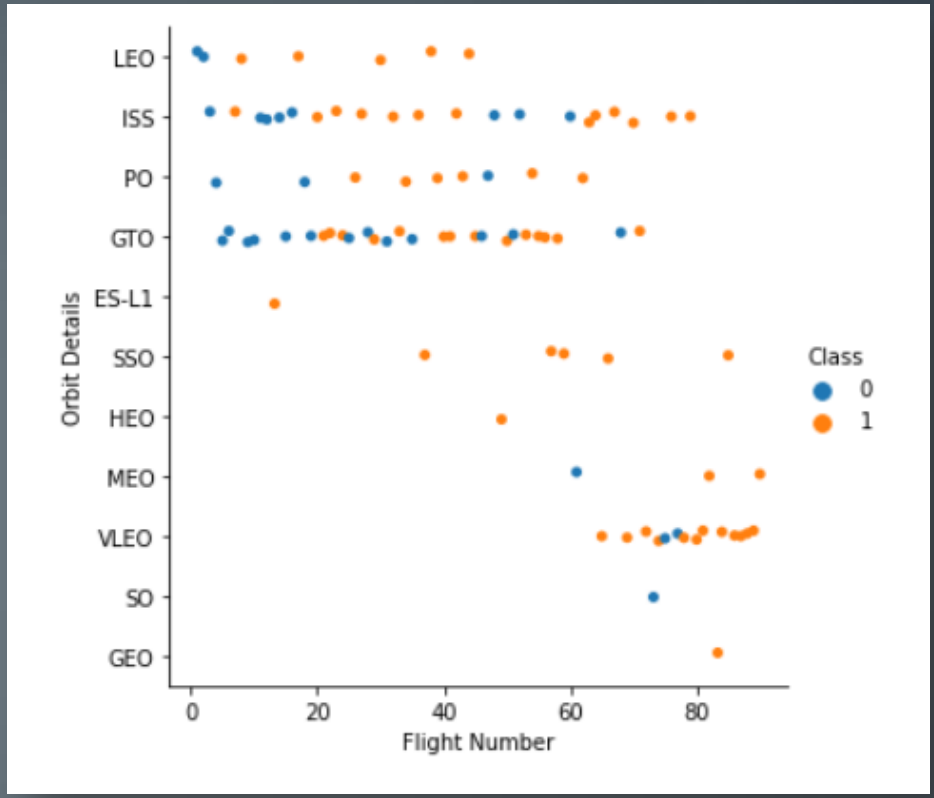
- No rockets are launched for heavy payload mass (greater than 10000)
- Flights with the largest loads come from CCAFS SLC 40 and kSC LC 39A sites

Success rate of each orbit type



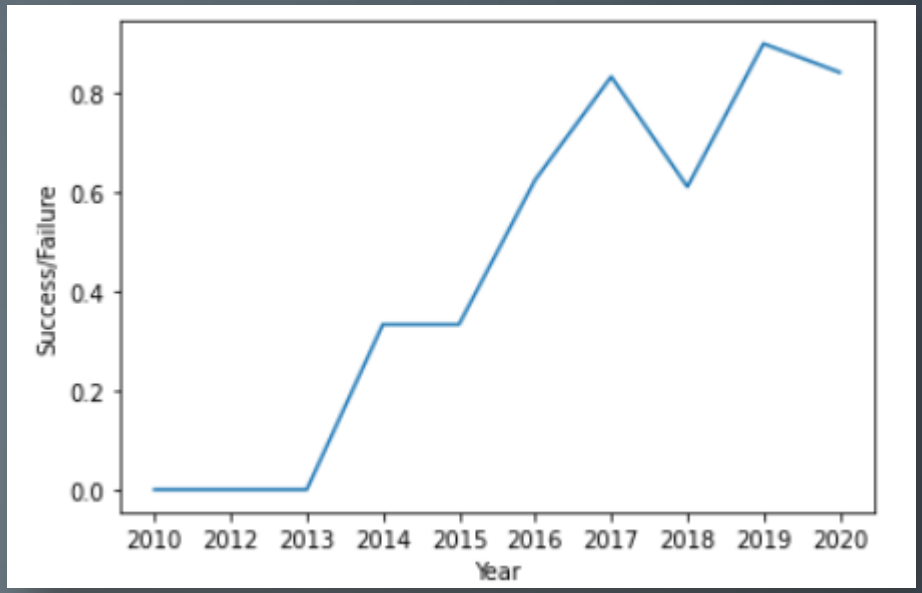
- The orbits with the highest success rate are ES-L1, GEO, SSO and HEO.
- The orbits with the lowest success rate are ISS, GTO and SO with a value of 0.

FlightNumber and Orbit type



- In the LEO orbit, 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.

Launch success yearly trend



- The graph shows a marked trend of successful launches since 2013, only reduced in 2018.

SQL Notebook for Peer Assignment

Task 1

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

```
[4]  %sql select distinct(LAUNCH_SITE) from SPACEXTBL
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 2

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

```
[5]  %sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' limit 5
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 3

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

```
[6]  %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 4

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

```
[7]  %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 5

List the date where the first succesful landing outcome in drone ship was acheived.

Hint:Use min function

```
%sql select min(DATE) from SPACEXTBL where Landing__Outcome = 'Success (ground pad)'
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 6

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

```
[9] %sql select BOOSTER_VERSION from SPACEXTBL where Landing__Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 7

List the total number of successful and failure mission outcomes

```
[10] %sql select count(MISSION_OUTCOME) from SPACEXTBL where MISSION_OUTCOME = 'Success' or MISSION_OUTCOME = 'Failure (in flight)'
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 8

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

```
[11] %sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 9

List the records which will display the month names, succesful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017

```
[12] %sql SELECT EXTRACT(MONTH, select min(DATE) from SPACEXTBL where Landing__Outcome = 'Success (ground pad)')
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

Task 10

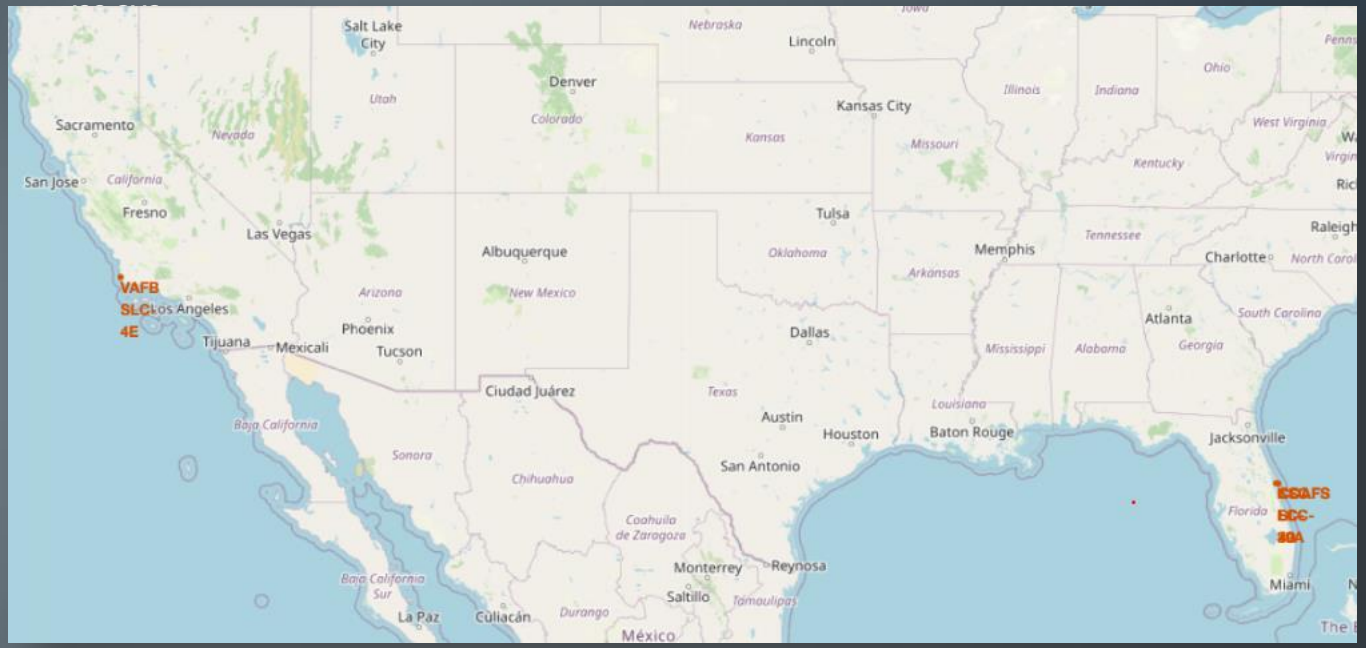
Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

```
[13] %sql select * from SPACEXTBL where Landing__Outcome like 'Success%' and (DATE between '2010-06-04' and '2017-03-20') order by date desc
```

```
Environment variable $DATABASE_URL not set, and no connect string given.
Connection info needed in SQLAlchemy format, example:
               postgresql://username:password@hostname/dbname
               or an existing connection: dict_keys([])
```

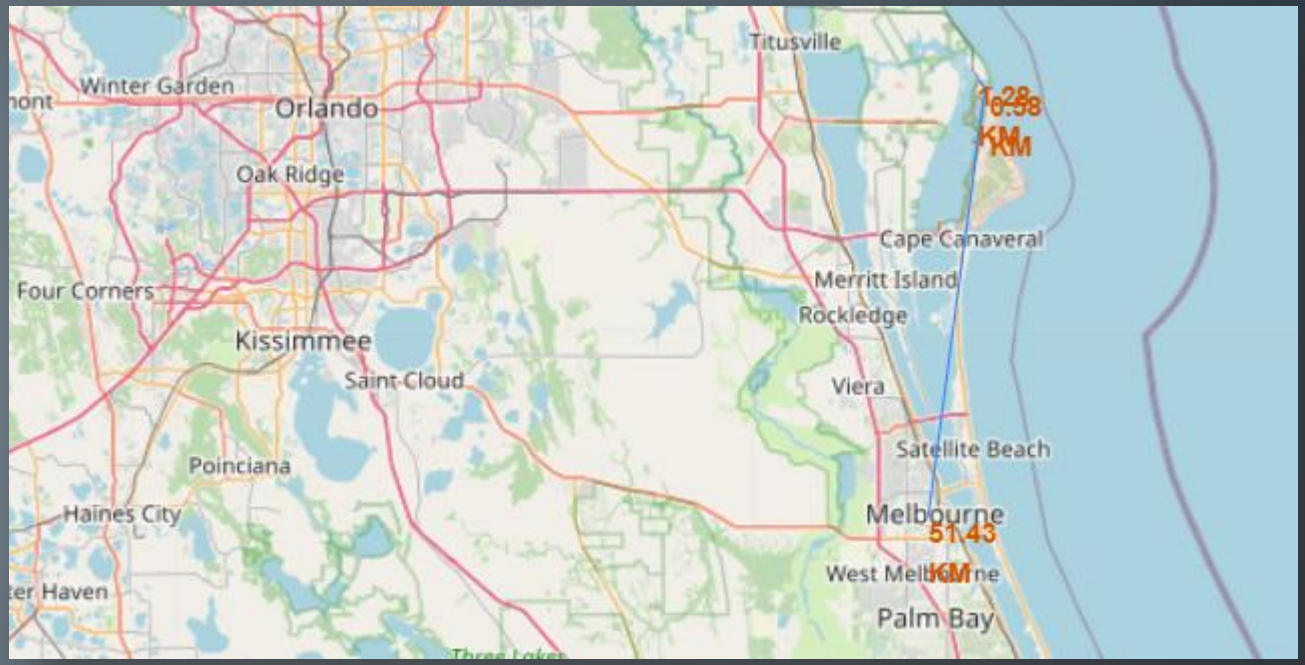
Launch Sites Locations Analysis with Folium

Mark all launch sites on a



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

Calculate the distances between a launch site to its proximities



Machine Learning

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

```
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])
```

```
[ ]  # students get this
     transform = preprocessing.StandardScaler()
```

```
[ ]  X[0:5]
```

	FlightNumber	PayloadMass	Flights	Block	ReusedCount	Orbit_ES-L1	Orbit_GEO
0	1.0	6104.959412	1.0	1.0	0.0	0.0	0.0
1	2.0	525.000000	1.0	1.0	0.0	0.0	0.0
2	3.0	677.000000	1.0	1.0	0.0	0.0	0.0
3	4.0	500.000000	1.0	1.0	0.0	0.0	0.0
4	5.0	3170.000000	1.0	1.0	0.0	0.0	0.0

5 rows × 83 columns

```
[ ]  X_train, X_test, Y_train, Y_test = train_test_split( X, Y, test_size=0.2, random_state=2)
     print ('Train set:', X_train.shape,  Y_train.shape)
     print ('Test set:', X_test.shape,  Y_test)
     Y_test.shape
```

```
Train set: (72, 83) (72,)
Test set: (18, 83) [1 1 1 1 1 1 1 0 0 0 0 0 1 1 1 1 0 1]
(18,)
```

we can see we only have 18 test samples.

```
[ ]  Y_test.shape
```

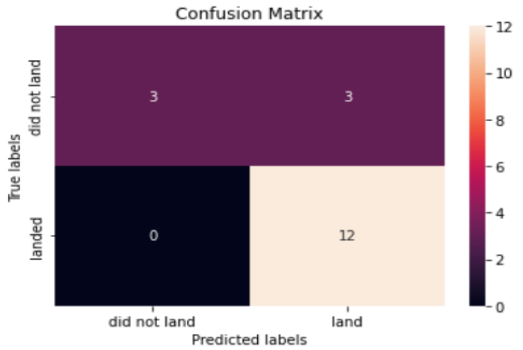
```
(18,)
```

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

```
0.8333333333333334
```

Lets look at the confusion matrix:

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

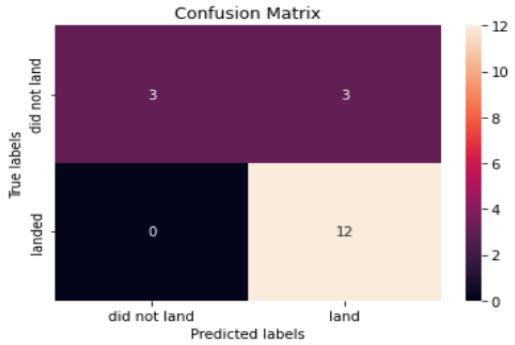



```
[25] tree_cv.score(X_test, Y_test)
```

```
0.6111111111111112
```

We can plot the confusion matrix

```
[26] yhat = svm_cv.predict(X_test)
     plot_confusion_matrix(Y_test,yhat)
```

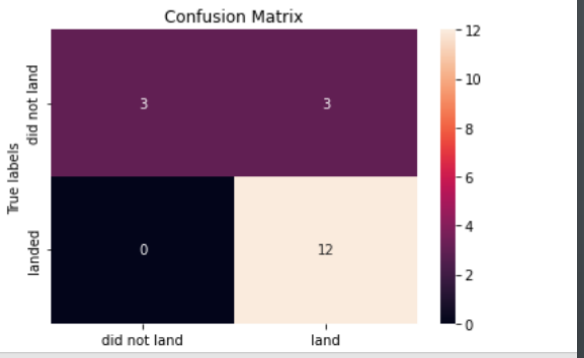



```
[20] svm_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```

We can plot the confusion matrix

```
[21] yhat=svm_cv.predict(X_test)
     plot_confusion_matrix(Y_test,yhat)
```

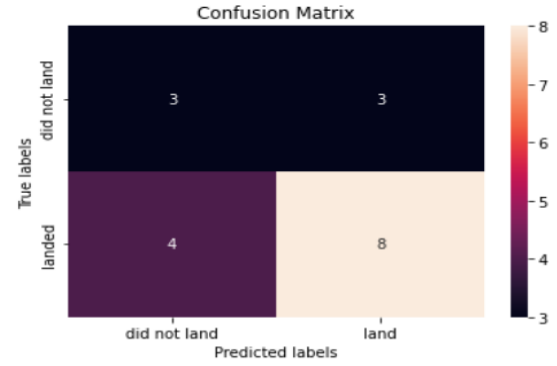



```
[30] knn_cv.score(X_test, Y_test)
```

```
0.6111111111111112
```

We can plot the confusion matrix

```
[31] yhat = knn_cv.predict(X_test)
     plot_confusion_matrix(Y_test,yhat)
```



Method perfomance

```
[32] print('Accuracy for Logistics Regression method:', logreg_cv.score(X_test, Y_test))
     print( 'Accuracy for Support Vector Machine method:', svm_cv.score(X_test, Y_test))
     print('Accuracy for Decision tree method:', tree_cv.score(X_test, Y_test))
     print('Accuracy for K nearsdt neighbors method:', knn_cv.score(X_test, Y_test))
```

```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.6111111111111112
Accuracy for K nearsdt neighbors method: 0.6111111111111112
```

Conclusion

- Decision Tree Model is the best algorithm for this data set.
- Launches with a low payload mass show better results than launches with a higher payload mass.
- KSC LC-39 has the highest launch success rate of all sites.
- Orbits ES-Li, Geo, HEO and SSO present success rate Rockets with heavy payload mass (greater than 10000) are not launched, also launches with low payload mass show better results than launches with higher payload mass.
- In general, there has been a marked trend of successful launches since 2013, only reduced in 2018.
- Spacex requires analyzing data (collecting, transforming, cleaning and modeling this data) to discover useful information for decision making.

References

- Github: https://github.com/JorgeTRY

- https://www.edx.org/es/professional-certificate/ibm-data-science

THANK YOU, FOR YOUR ATTENTION!!