DATA SCIENCE CAPSTONE PROJECT

VIDYA PRAKASH KHAROTE

11 JUNE 2025

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusions
- Appendix

EXECUTIVE SUMMARY

- Summary of methodologies
  - Data collection
  - Data wrangling
  - Exploratory Data Analysis with Data Visualization
  - Exploratory Data Analysis with SQL
  - Building an interactive map with Folium
  - Predictive analysis(classification)
- Summary of all results
  - Exploratory Data Analysis(EDA) results
  - Geospatial analytics
  - Interactive dashboard
  - Predictive analysis of classification models

INTRODUCTION

Project background and Context

SpaceX is the most successful company of the commercial space age, making space travel affordable. The company advertises Falcon 9 rocket launches on its website, with a cost of 62 millions dollars, other providers cost upward of 165 million dollars each, much of the saving 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. Based on public information and machine learning models, we are going to predict if SpaceX will reuse the first stage.

Questions to be answered

- How do variables such as payload mass, launch site, number of flights and orbits affect the success of the stage landing?
- Does the rate of successful landings increase over the years?
- What is the best algorithm that can be used for binary classification in this case?

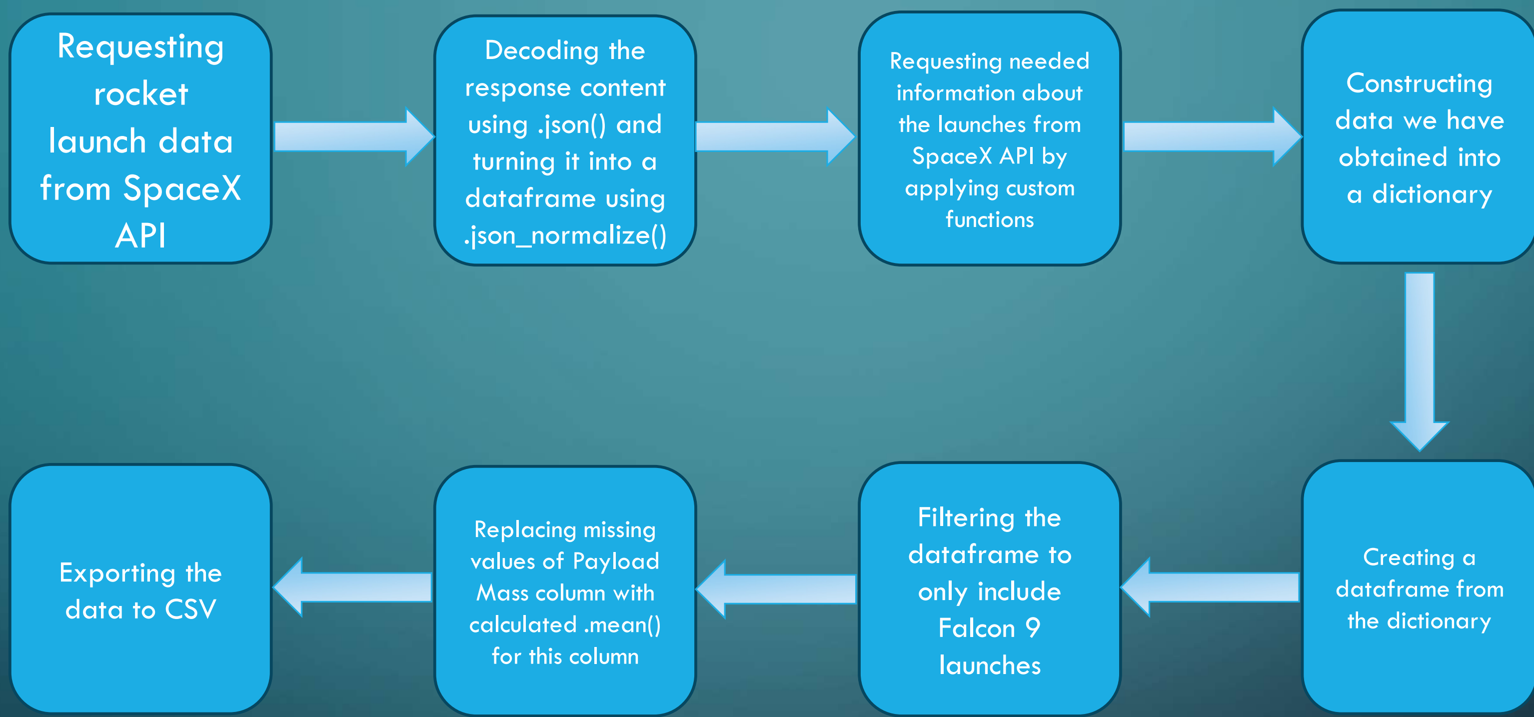
METHODOLOGY

1. Data Collection
   - Making GET requests to the SpaceX REST API
   - Web Scraping
2. Data Wrangling
   - Using the .fillna() method to remove NaN values
   - Using the .value_counts() method to determine the following:
     - Number of launches on each site
     - Number and occurrence of each orbit
     - Number and occurrence of mission outcome per orbit type
   - Creating a landing outcome label that shows the following:
     - 0 when the booster did not land successfully
     - 1 when the booster did land successfully
3. Exploratory Data Analysis
   - Using SQL queries to manipulate and evaluate the SpaceX dataset
   - Using Pandas and Matplotlib to visualize relationships between variables, and determine patterns
4. Interactive Visual Analytics
   - Geospatial analytics using Folium
   - Creating an interactive dashboard using Plotly Dash
5. Data Modelling and Evaluation
   - Using Scikit-Learn to:
   - Pre-process (standardize) the data
   - Split the data into training and testing data using train_test_split
   - Train different classification models
   - Find hyperparameters using GridSearchCV
   - Plotting confusion matrices for each classification model
   - Assessing the accuracy of each classification model

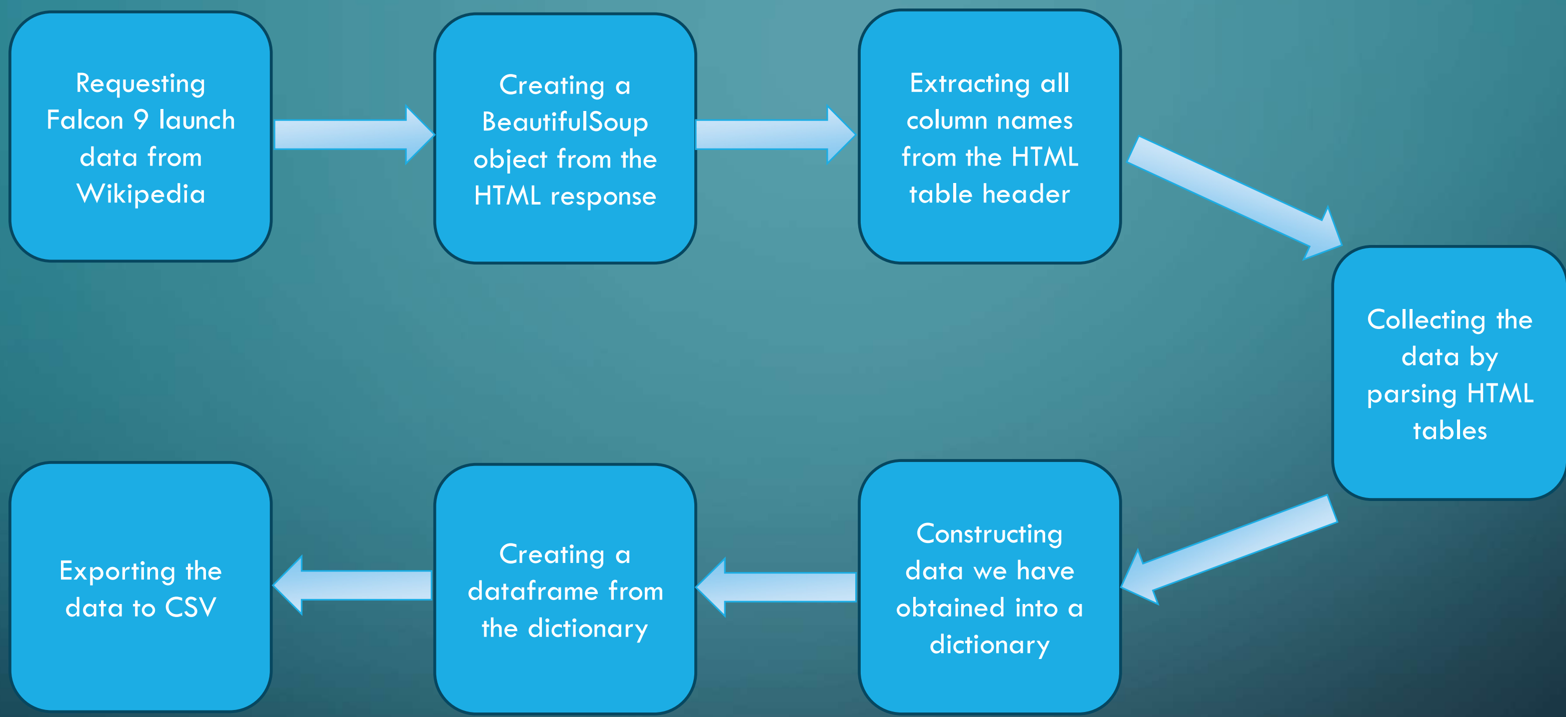
DATA COLLECTION – SPACE X REST API

- Using the SpaceX API to retrieve data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
- Data collection process involved a combination of API requests from SpaceX REST API and Web Scraping data from a table in SpaceX's Wikipedia entry. We had to use both of these data collection methods in order to get complete information about the launches for a more detailed analysis.
- Data Columns are obtained by using SpaceX REST API: FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude
- Data Columns are obtained by using Wikipedia Web Scraping: Flight No., Launch site, Payload, PayloadMass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time

DATA COLLECTION – SPACEX API



DATA COLLECTION – WEB SCRAPING



EDA WITH DATA VISUALIZATION

Charts were plotted:
Flight Number vs Payload Mass
Flight Number vs. Orbit type,
Payload Mass vs Orbit Type
Success Rate Yearly Trend
Flight Number vs. Launch Site
Orbit Type vs. Success Rate

Scatter plots shows the relationship between variables. If a relationship exists, they could be used in machine learning model.

Bar charts show comparisons among discrete categories. The goal is to show the relationship between the specific categories bing compared and a measured value.

Line charts show trends in data over time (time series)

EDA WITH SQL

Performed SQL queries:

- Displaying the names of the unique launch sites in the space mission
- Displaying 5 records where launch sites begin with the string ‘CCA’
- Displaying the total payload mass carried by boosters launched by NASA (CRS)
- Displaying average payload mass carried by booster version F9 v1.1
- Listing the date when the first successful landing outcome in ground pad was achieved
- Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- Listing the total number of successful and failure mission outcomes
- Listing the names of the booster versions which have carried the maximum payload mass
- Listing the failed landing outcomes in drone ship, their booster versions and launch site names for the months in year 2015
- Ranking 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

BUILD AN INTERACTIVE MAP WITH FOLIUM

Markers of all launch sites:

- Added Marker with Circle, Popup label and Text Label of NASA Johnson Space Center using its latitude and Longitude coordinates as a start location.
- Added Markers with Circle, Popup Label and Text Label of all Launch Sites using their latitude and longitude coordinates to show their geographical locations and proximity to Equator and Coasts.

Colored Markers of the launch outcomes for each Launch Site:

- Added colored Markers of success (Green) and failed(Red) launches using Marker Cluster to identify which launch sites have relatively high success rates.

Distance between a Launch Site to its proximities:

- Added colored Lines to show distance between the Launch Site KSC LC-39A (as an example) and its proximities like Railway, Highway, Coastline and Closest City.

BUILD A DASHBOARD WITH PLOTLY DASH

Launch Sites DropDown List:

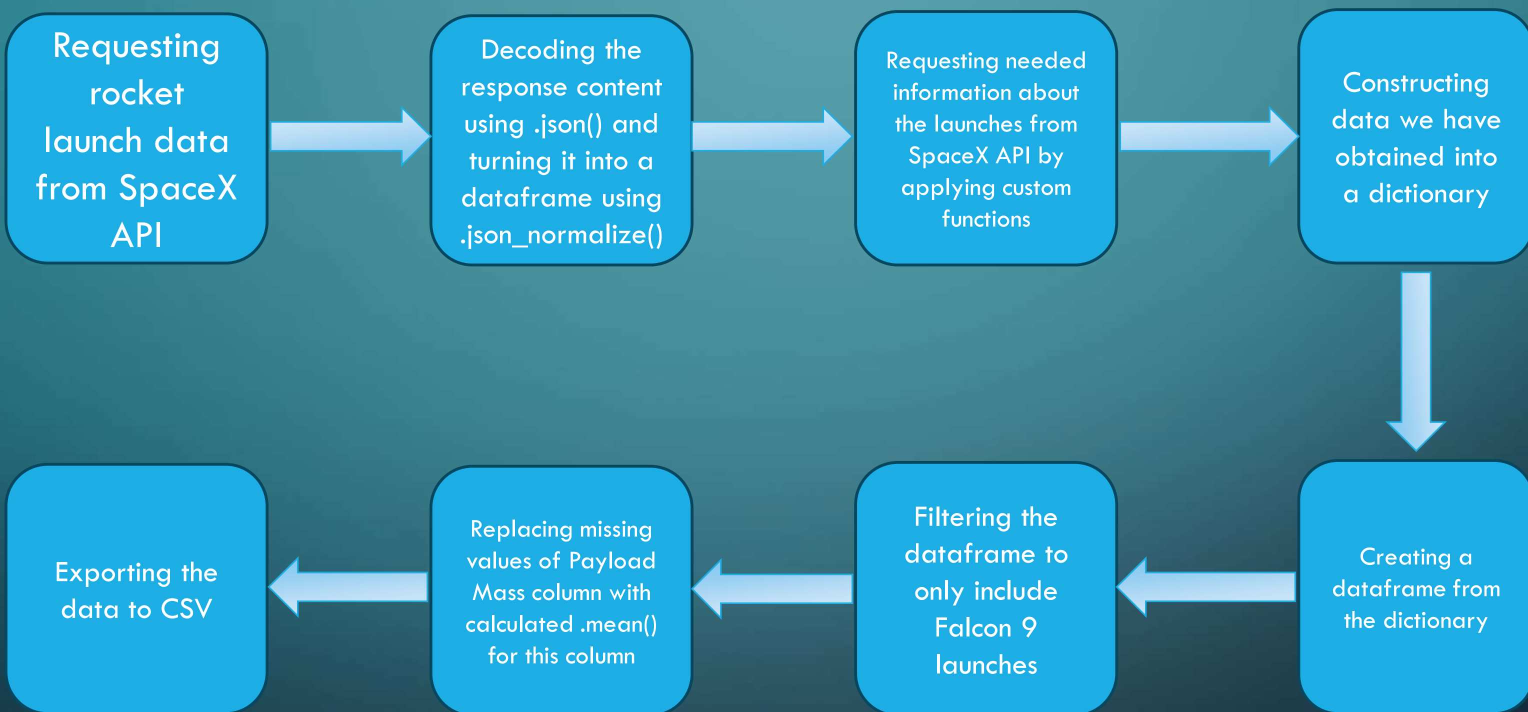
- Added a dropdown list to enable launch site selection

Pie Chart showing Success Launches (All Sites/Certain Site):
-Added a pie chart to show the total successful launches count for all sites and the Success vs.Failed counts for the site, if a specific Launch Site was selected.

Slider of Payload Mass Range:
-Added a slider to select Payload range.

Scatter Chart of Payload Mass vs. Success Rate for the different Booster Versions:
-Added a scatter chart to show the correlation between Payload and Launch Success.

PREDICTIVE ANALYSIS (CLASSIFICATION)



RESULTS

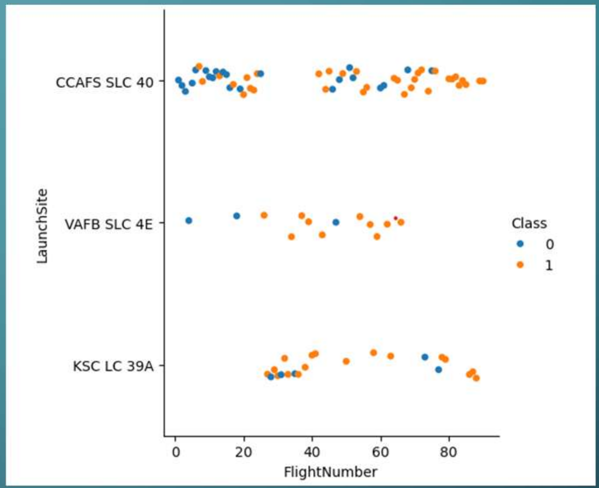
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

EDA WITH VISUALIZATION

FLIGHT NUMBER VS. LAUNCH SITE

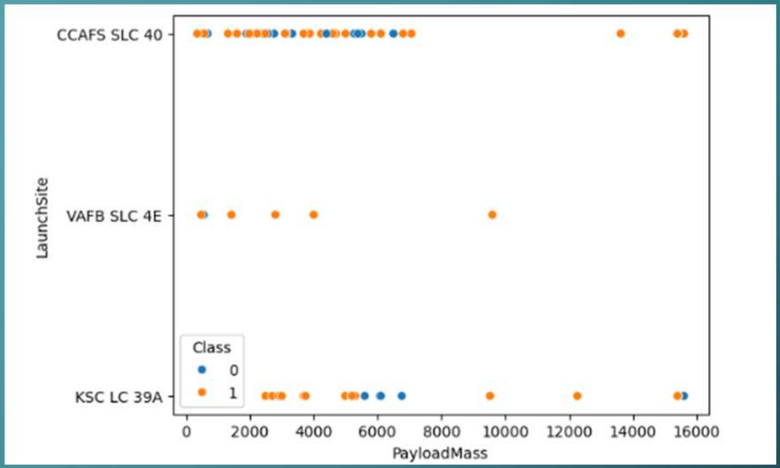
Explanation:

- The earliest flights all failed while the latest flights all succeeded.
- The CCFAS SLC 40 launch site has about a half of all launches
- VAFB SLC 4E and KSC LC 39A have higher success rates
- It can be assumed that each new launch has a higher rate of success.

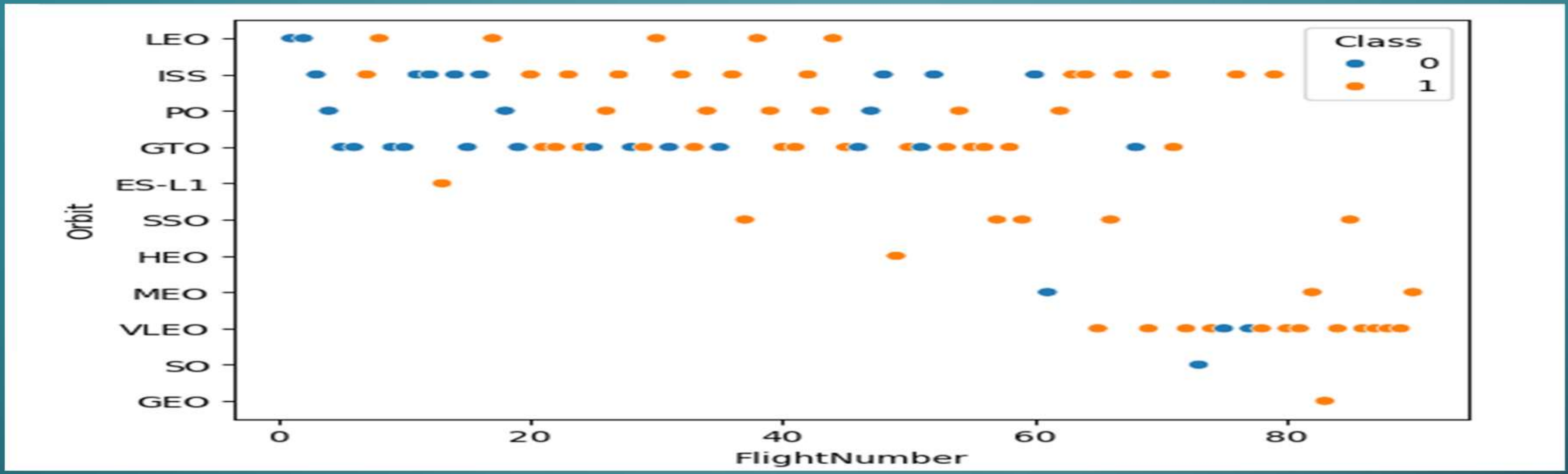


PAYLOAD MASS VS. LAUNCH SITE

- VAFB SLC 4E launch site has no rockets launched for greater than 10000kg
- CCAFS SLC 40 has 100% success rate.
- KSC LC 39A has 100% success rate for launches with payload masses less than 5000kg.

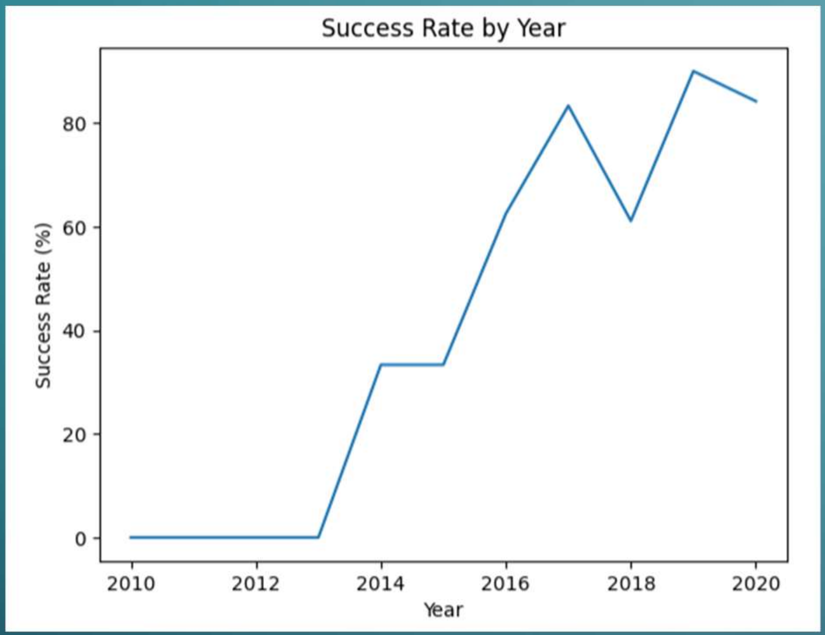


SUCCESS RATE VS. ORBIT TYPE



- ES-L1, SSO, HEO has 100% success rate.
- SO has 100% has failure rate
- LEO orbit has relationship between success rate and flight number
- GTO has not relation between success rate and flight number

LAUNCH SUCCESS YEARLY TREND



- The success rate since 2013 kept increasing till 2017
- It again gain success between 2018-2019
- And then it again started falling.

EDA WITH SQL

ALL LAUNCH SITE NAMES

Task 1

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

```python
# Task 1: Display the names of the unique launch sites
query = 'SELECT DISTINCT "Launch_Site" FROM SPACEXTBL'
cur.execute(query)
# Fetch the results
unique_launch_sites = cur.fetchall()

# Print the results
print("Unique Launch Sites:")
for site in unique_launch_sites:
    print(site[0])
```

```
Unique Launch Sites:
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
```

Explanation:

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

LAUNCH SITE NAMES BEGIN WITH 'CCA'

Task 2

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

[12]:

```
# Display 5 records where launch sites begin with the string 'CCA'
query = 'SELECT * FROM SPACEXTBL WHERE "Launch_Site" LIKE "CCA%" LIMIT 5'
cur.execute(query)

# Fetch the results
records = cur.fetchall()

# Print the results
print("Records where Launch Sites begin with 'CCA':")
for record in records:
    print(record)
```

```
Records where Launch Sites begin with 'CCA':
('2010-06-04', '18:45:00', 'F9 v1.0  B0003', 'CCAFS LC-40', 'Dragon Spacecraft Qualification Unit', 0, 'LEO', 'SpaceX', 'Success', 'Failure (parachute)')
('2010-12-08', '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)')
('2012-05-22', '7:44:00', 'F9 v1.0  B0005', 'CCAFS LC-40', 'Dragon demo flight C2', 525, 'LEO (ISS)', 'NASA (COTS)', 'Success', 'No attempt')
('2012-10-08', '0:35:00', 'F9 v1.0  B0006', 'CCAFS LC-40', 'SpaceX CRS-1', 500, 'LEO (ISS)', 'NASA (CRS)', 'Success', 'No attempt')
('2013-03-01', '15:10:00', 'F9 v1.0  B0007', 'CCAFS LC-40', 'SpaceX CRS-2', 677, 'LEO (ISS)', 'NASA (CRS)', 'Success', 'No attempt')
```

Displaying 5 records where launch sites begin with string 'CCA'

TOTAL PAYLOAD MASS

Task 3

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

```
# Specify the table name
table_name = 'SPACEXTBL'  # Replace with your actual table name

query = f'PRAGMA table_info("{table_name}")'
cur.execute(query)

# Fetch all results
columns_info = cur.fetchall()

# Extract column names
column_names = [column[1] for column in columns_info]  # column[1] contains the column name

print("Column names in the table:", column_names)
```

```
Column names in the table: ['Date', 'Time (UTC)', 'Booster_Version', 'Launch_Site', 'Payload', 'PAYLOAD_MASS__KG_', 'Orbit', 'Customer', 'Mission_Outcome', 'Landing_Outcome']
```

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

AVERAGE PAYLOAD MASS

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

```
# Display the  average payload mass carried by booster version F9 v1.1
query = '''
SELECT AVG("PAYLOAD_MASS__KG_")
FROM SPACEXTBL
WHERE "Booster_Version" = "F9 v1.1"
'''
cur.execute(query)

# Fetch the result
avg_payload_mass = cur.fetchone()[0]

# Print the total payload mass
print("Average payload mass carried by Booster Version F9 v1.1:", avg_payload_mass)
```

```
Average payload mass carried by Booster Version F9 v1.1: 2928.4
```

Average payload mass carried by Booster Version F9 v1.1 : 2928.4

FIRST SUCCESSFUL LANDING

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

Hint:Use min function

```
# SQL query to get the date of the first successful landing outcome in ground pad
query = '''
SELECT MIN("Date") AS "First_Successful_Landing_Date"
FROM SPACEXTBL
WHERE "Mission_Outcome" = 'Success' AND "Landing_Outcome" = 'Success (ground pad)'
'''
cur.execute(query)

# Fetch the result
first_successful_landing_date = cur.fetchone()

# Print the result
print("Date of the first successful landing outcome in ground pad:", first_successful_landing_date[0])
```

```
Date of the first successful landing outcome in ground pad: 2015-12-22
```

On 22nd dec 2015, first successful landing outcome in ground

SUCCESSFUL DRONE WITHIN 4000 AND 6000 RANGE

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

```
# SQL query to get the names of boosters with specified conditions
query = '''
SELECT DISTINCT "Booster_Version"
FROM SPACEXTBL
WHERE "Landing_Outcome" = 'Success (drone ship)'
  AND "PAYLOAD_MASS__KG_" > 4000
  AND "PAYLOAD_MASS__KG_" < 6000
'''

cur.execute(query)

# Fetch all results
boosters = cur.fetchall()

# Print the results
print("Boosters with successful landing in drone ship and payload mass between 4000 and 6000:")
for booster in boosters:
    print(booster[0])  # Access the first column of each row
```

```
Boosters with successful landing in drone ship and payload mass between 4000 and 6000:
F9 FT B1022
F9 FT B1026
F9 FT  B1021.2
F9 FT  B1031.2
```

COUNT OF SUCCESS/FAILURE MISSION

List the total number of successful and failure mission outcomes

```
# SQL query to count successful and failed mission outcomes
query = '''
SELECT "Mission_Outcome", COUNT(*) AS "Total"
FROM SPACEXTBL
GROUP BY "Mission_Outcome"
'''

cur.execute(query)

# Fetch all results
outcomes = cur.fetchall()

# Print the results
print("Total number of successful and failed mission outcomes:")
for outcome in outcomes:
    print(f"{outcome[0]}: {outcome[1]}")
```

```
Total number of successful and failed mission outcomes:
Failure (in flight): 1
Success: 98
Success : 1
Success (payload status unclear): 1
```

List of total number of successful and failure mission outcomes

LIST OF BOOSTERS CARRIED MAXIMUM VERSION

List all the booster_versions that have carried the maximum payload mass. Use a subquery.

```
# SQL query to find booster_versions that have carried the maximum payload mass
query = '''
SELECT "Booster_Version"
FROM SPACEXTBL
WHERE "PAYLOAD_MASS__KG_" = (
    SELECT MAX("PAYLOAD_MASS__KG_")
    FROM SPACEXTBL
)
'''

cur.execute(query)

# Fetch all results
max_payload_boosters = cur.fetchall()

# Print the results
print("Booster versions that have carried the maximum payload mass:")
for booster in max_payload_boosters:
    print(booster[0])
```

```
Booster versions that have carried the maximum payload mass:
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
```

Booster versions that have carried the maximum payload mass

LIST OF FAILURE LANDING

```
# SQL query to get the required records for failed landings on a drone ship in 2015

query = '''
SELECT
    CASE substr("Date", 6, 2)
        WHEN '01' THEN 'January'
        WHEN '02' THEN 'February'
        WHEN '03' THEN 'March'
        WHEN '04' THEN 'April'
        WHEN '05' THEN 'May'
        WHEN '06' THEN 'June'
        WHEN '07' THEN 'July'
        WHEN '08' THEN 'August'
        WHEN '09' THEN 'September'
        WHEN '10' THEN 'October'
        WHEN '11' THEN 'November'
        WHEN '12' THEN 'December'
    END AS month_name,
    "Landing_Outcome",
    "Booster_Version",
    "Launch_Site"
FROM SPACEXTBL
WHERE "Landing_Outcome" = 'Failure (drone ship)'
    AND "Date" LIKE "2015%"


'''
# or you can use this SQLite, string indexing starts at 1- 1-4 extract first 4 characters
#AND substr("Date", 1, 4) = '2015'

cur.execute(query)

# Fetch all results
failed_landings_2015 = cur.fetchall()

# Print the results
print("Month Name | Landing Outcome | Booster Version | Launch Site")
print("------------------------------------------------------------")
for record in failed_landings_2015:
    print(f"{record[0]} | {record[1]} | {record[2]} | {record[3]}")
```

```
Month Name | Landing Outcome | Booster Version | Launch Site
------------------------------------------------------------
January | Failure (drone ship) | F9 v1.1 B1012 | CCAFS LC-40
April | Failure (drone ship) | F9 v1.1 B1015 | CCAFS LC-40
```

RANKING

Task 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.

```
# SQL query to rank the count of landing outcomes
query = '''
SELECT "Landing_Outcome", COUNT(*) AS outcome_count
FROM SPACEXTBL
WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing_Outcome"
ORDER BY outcome_count DESC;
'''

# Execute the query
cur.execute(query)

# Fetch all results
results = cur.fetchall()

# Display the results
for row in results:
    print(row)
```

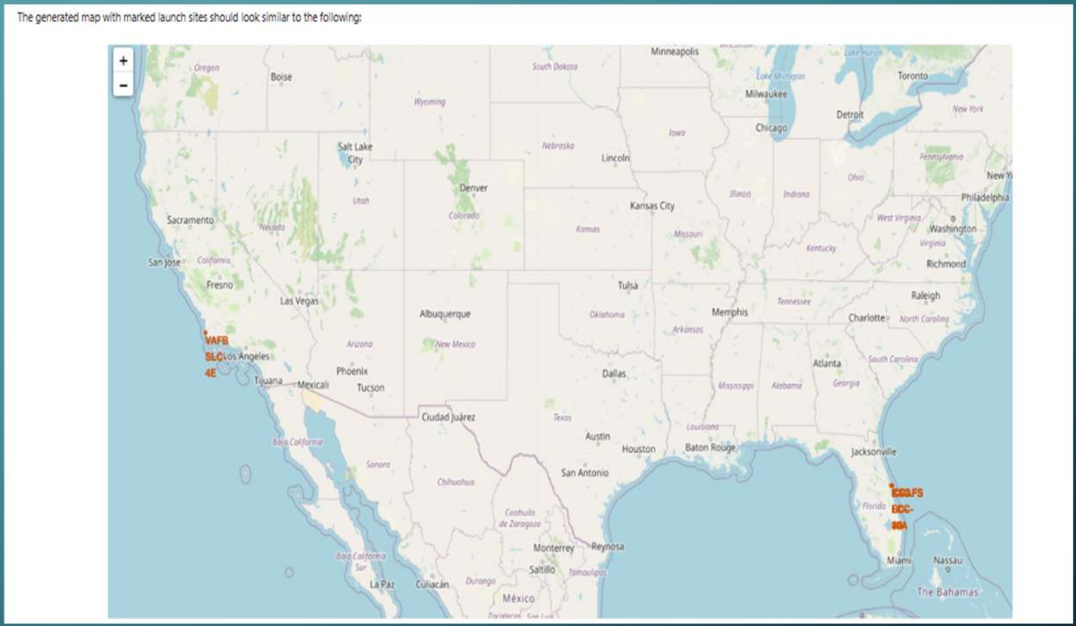
```
('No attempt', 10)
('Success (drone ship)', 5)
('Failure (drone ship)', 5)
('Success (ground pad)', 3)
('Controlled (ocean)', 3)
('Uncontrolled (ocean)', 2)
('Failure (parachute)', 2)
('Precluded (drone ship)', 1)
```

Listing the failed landing outcomes in drone ship, their booster versions and launch site names for the months in year 2015

INTERACTIVE MAP WITH FOLIUM

All Launch site’s location markers on a global map

Explanation:
Most of launch sites are in proximity to the Equator line, the land is moving dast at the equator than any other place on the surface of the Eart. Anything on the surface of the Earth at the equator is already moving at 1670 km/hour. If a ship is launched from equator it goes up into space, and it is also moving around the earth at the same speed it was moving before launching. This is because of the inertia. This speed will help the spacecraft keep up a good enough speed to stay in orbit
All launch sites are in very close proximity to the coast; while launching rockets towards the ocean it minimizes the risk of having any debris dropping or exploding near people.

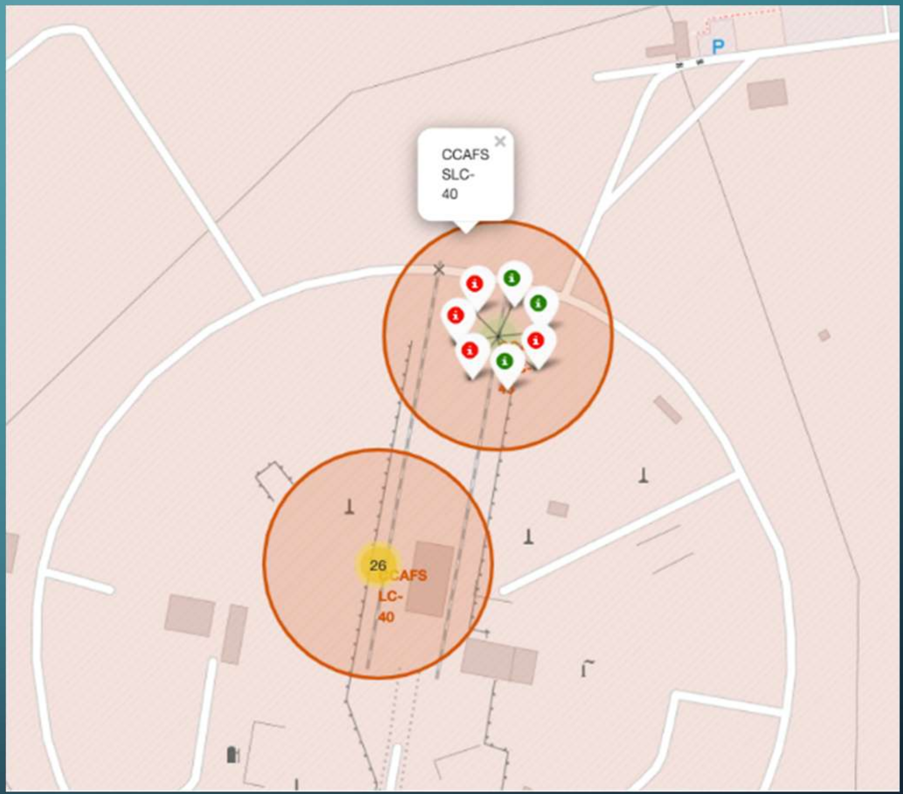


Color – labeled launch records on the map

Explanation:
From the color labeled markers we should be able to easily identify which launch sites have relatively high success rates.

Green Marker – Successful launch
Red Marker – Failed Launch

Launch site KSC LC-39A has a very high success rate



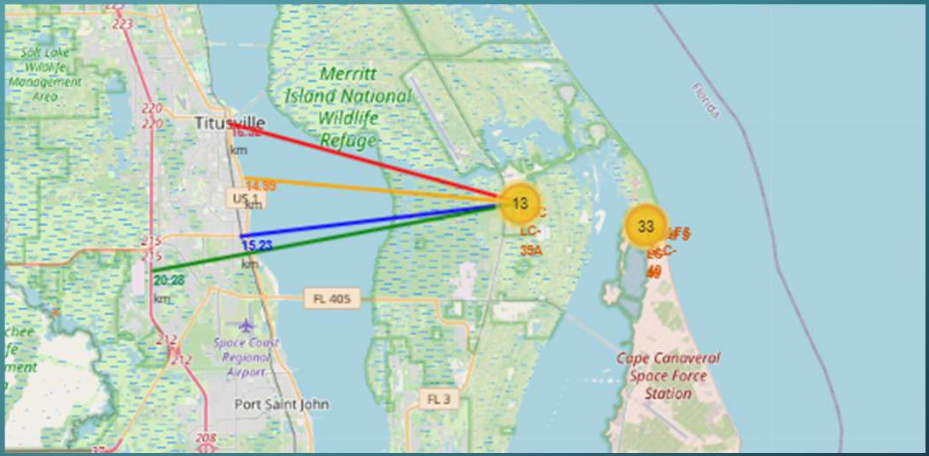
Distance from the launch site KSC LC-39A to its proximities

Explanation: • From the visual analysis of the launch site KSC LC-39A we can clearly see that it is:

- relative close to railway (15.23 km)
- relative close to highway (20.28 km)
- relative close to coastline (14.99 km)

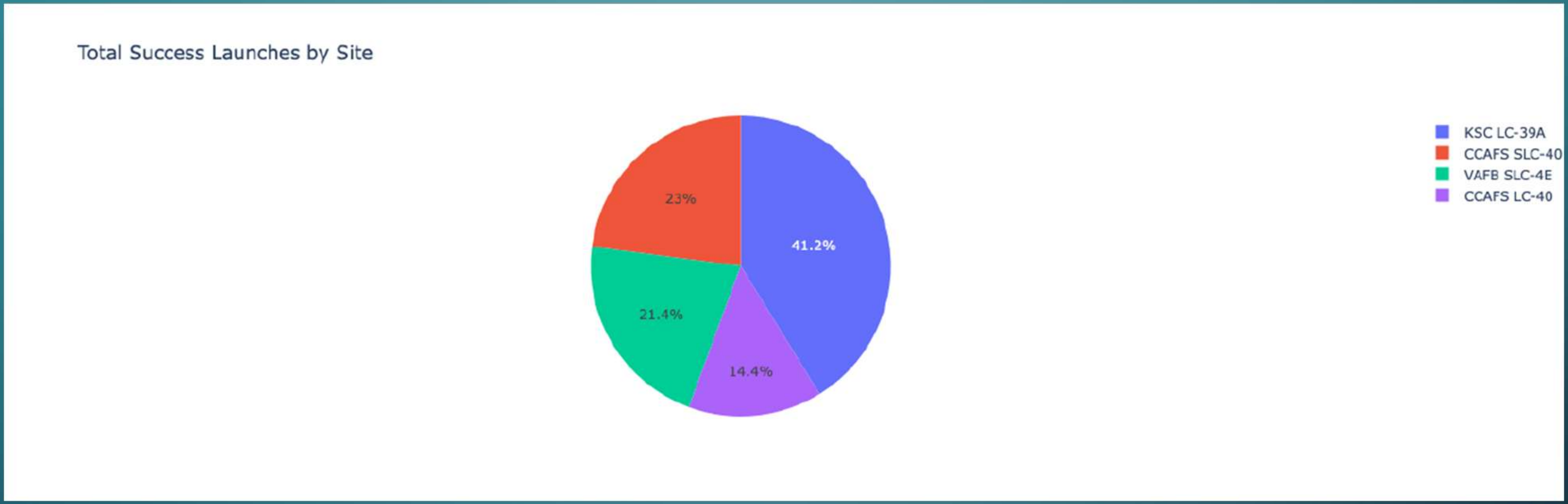
• Also the launch site KSC LC-39A is relative close to its closest city Titusville (16.32 km).

• Failed rocket with its high speed can cover distances like 15-20 km in few seconds. It could be potentially dangerous to populated areas.



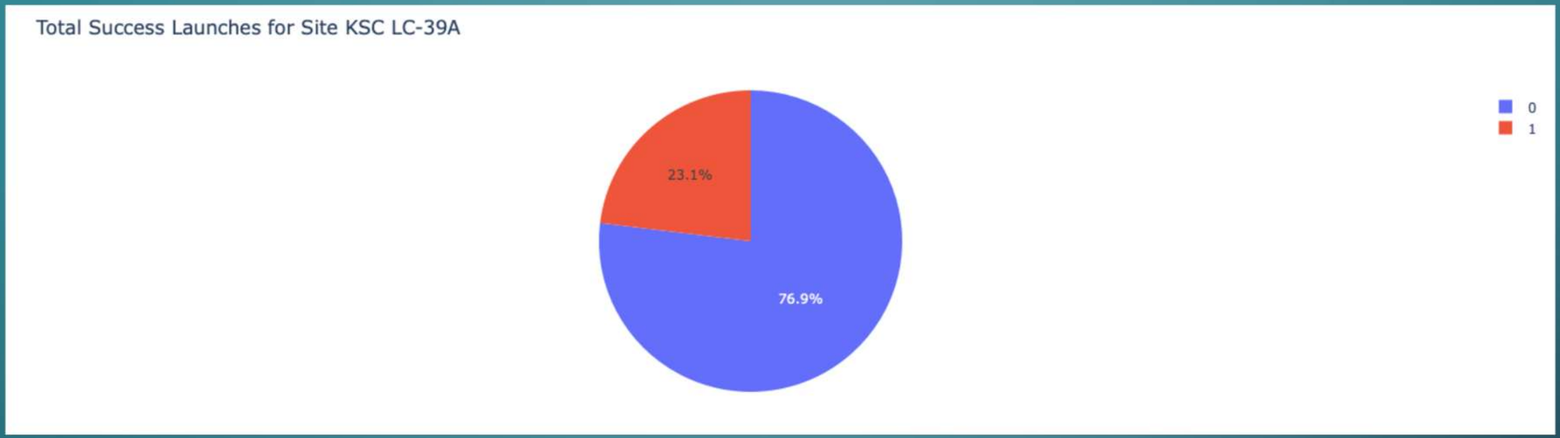
BUILD DASGBOARD WITH PLOTLY DASH

LAUNCH OF SUCCESS COUNT FOR ALL SITES



The chart clearly shows that from all the sites, KSC LC-39A has the most successful launches.

LAUNCH SITE WITH HIGHEST LAUNCH SUCCESS RATIO



• KSC LC-39A has the highest launch success rate (76.9%) with 10 successful and only 3 failed landings.

PAYLOAD MASS VS. LAUNCH OUTCOME FOR ALL SITES

The charts show that payloads between 2000 and 5500 kg have the highest success rate.

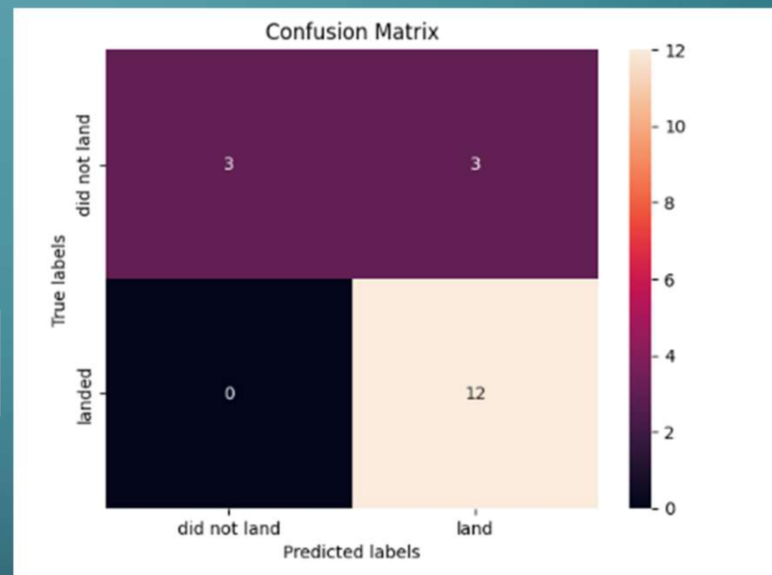
PREDICTIVE ANALYSIS (CLASSIFICATION)

LOGISTIC REGRESSION

Logistic regression

- GridSearchCV best score: 0.8464285714285713
- Accuracy score on test set: 0.8333333333333334

```
[19]: print("Logistic Regression test data accuracy :",logreg_cv.score(X_test, Y_test))
      Logistic Regression test data accuracy : 0.8333333333333334
```



SVM

Support vector machine (SVM)

- GridSearchCV best score: 0.8482142857142856
- Accuracy score on test set: 0.8333333333333334

TASK 7

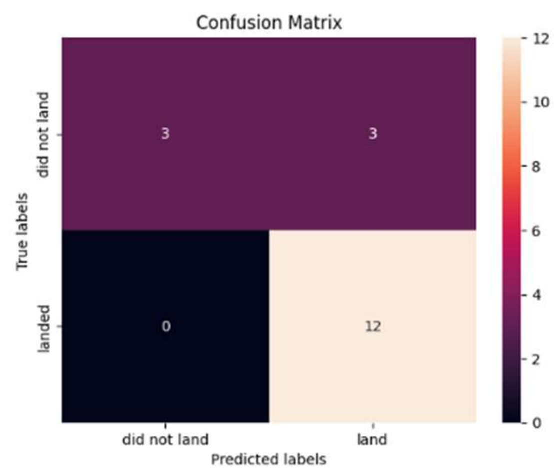
Calculate the accuracy on the test data using the method `score` :

```
[24]: print("SVM test data accuracy :",svm_cv.score(X_test, Y_test))
```

```
SVM test data accuracy : 0.8333333333333334
```

We can plot the confusion matrix

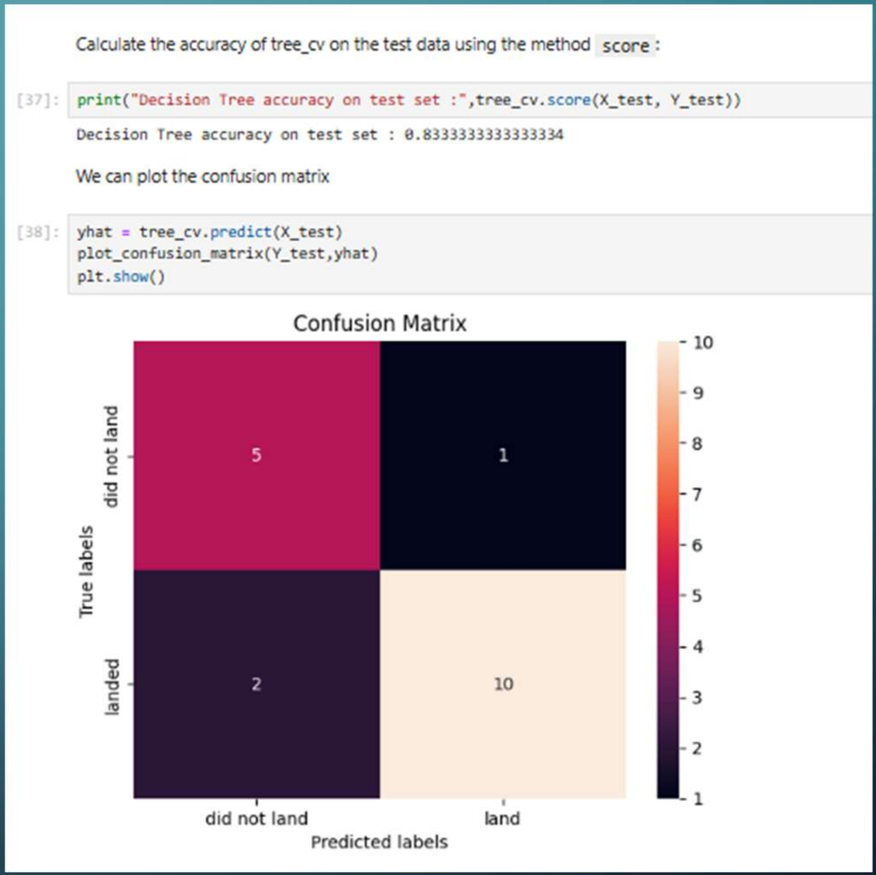
```
[25]: yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
plt.show()
```



DECISION TREE

Decision Tree

- GridSearchCV best score: 0.8482142857142856
- Accuracy score on test set: 0.8333333333333334



KNN

KNN Classifier

- GridSearchCV best score: 0.8482142857142856
- Accuracy score on test set: 0.8333333333333334

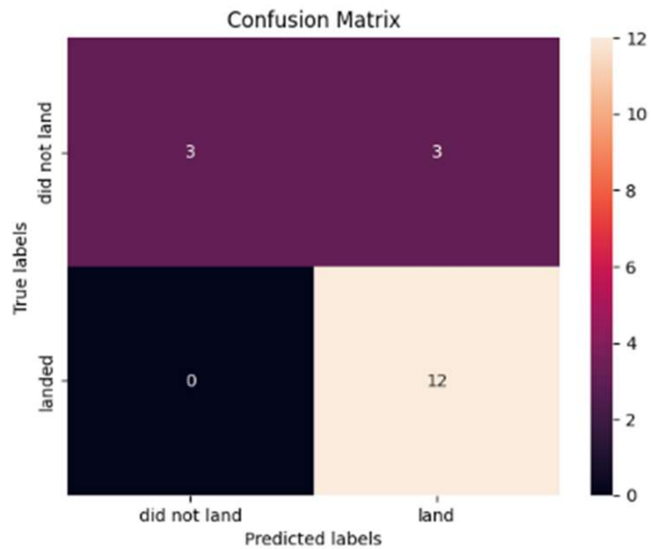
Calculate the accuracy of knn_cv on the test data using the method `score` :

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

```
[42]: 0.8333333333333334
```

We can plot the confusion matrix

```
[43]: yhat = knn_cv.predict(X_test)
      plot_confusion_matrix(Y_test,yhat)
      plt.show()
```



FIND BEST METHOD

Bases on Test Data Accuracy we can not predict Which method performs best.

Find the method performs best:

```
[44]: Report = pd.DataFrame({'Method' : ['Test Data Accuracy']})

knn_accuracy=knn_cv.score(X_test, Y_test)
Decision_tree_accuracy=tree_cv.score(X_test, Y_test)
SVM_accuracy=svm_cv.score(X_test, Y_test)
Logistic_Regression=logreg_cv.score(X_test, Y_test)

Report['Logistic_Reg'] = [Logistic_Regression]
Report['SVM'] = [SVM_accuracy]
Report['Decision Tree'] = [Decision_tree_accuracy]
Report['KNN'] = [knn_accuracy]



Report.transpose()
```

[44]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

INSIGHTS

- From the data visualization section, we can see that some features may have correlation with the mission outcome in several ways. For example, with heavy payloads the successful landing or positive landing rate are more for orbit types Polar, LEO and ISS. However, for GTO, we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.
- Therefore, each feature may have a certain impact on the final mission outcome. The exact ways of how each of these features impact the mission outcome are difficult to decipher. However, we can use some machine learning algorithms to learn the pattern of the past data and predict whether a mission will be successful or not based on the given features.
- In this project, we try to predict if the first stage of a given Falcon 9 launch will land in order to determine the cost of a launch.

- Each feature of a Falcon 9 launch, such as its payload mass or orbit type, may affect the mission outcome in a certain way.
- Several machine learning algorithms are employed to learn the patterns of past Falcon 9 launch data to produce predictive models that can be used to predict the outcome of a Falcon 9 launch.
- The predictive model produced by decision tree algorithm performed the best among the 4 machine learning algorithms employed.