IBM Developer
SKILLS NETWORK

IBM Data Science Capstone Project - Space X

Hitesh Mistry
November 14, 2021

Outline

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

Executive Summary

Summary of Methodologies:

- Data Collection
- Data Wrangling
- Exploratory Data Analysis (EDA)
  - with Data Visualisation
  - with Sql
- Building Interactive Map with Folium
- Building Plotly Dashboard
- Classification

Summary of Results:

- Screenshots of Visual Results
- Predictive Analysis Results
- EDA Results

Introduction

Project Background/Context:

The main objective of this project was to predict of the Falcon 9 SpaceX First Stage Rocket will land successfully. This investigation will allow for a subsequent analysis into calculating the cost of a launch in order for these findings to be used by an alternate company to support in a bid against SpaceX for the next rocket launch.

Problems you want to find answers for:

- Variables that can affect the rocket landing successfully
- Calculating the strength of each relationship between these independent variables and the result of a rocket landing
- What are the best conditions required for an optimal launch result

Methodology

Methodology

Data collection methodology:

- Describe how data was collected

Perform Data Wrangling

- Describe how data was processed

Perform exploratory data analysis (EDA) using visualization and SQL
Perform interactive visual analytics using Folium and Plotly Dash
Perform predictive analysis using classification models

- How to build, tune, evaluate classification models

Methodology

A breakdown of the datasets that were collected and utilised in this investigation:

1. SpaceX Launch Data - Obtained via SpaceX Rest API
   - Data set includes key variables such as Launches, Rocket Type, Outcome, Payload Weight, Landing Pad
2. Falcon 9 Launch Data - Obtained via Web Scraping
   - Leveraging the BeautifulSoup library we can scrape wikipedia for key launch data to augment our data investigation

Data Collection using SpaceX API

GitHub URL

Part 1 - Getting a response from the API endpoint

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

Part 2 - Converting response to a .json file by using .json_normalize

```
# Use json_normalize method to convert the json result into a dataframe
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

Part 3 - Apply data cleansing methods/functions

```
# Call getBoosterVersion
getBoosterVersion(data)
```

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

Part 4 - Combine columns into a dictionary and then turn into a Pandas Dataframe

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

Part 5 - Filter Dataframe for only the Falcon 9 launches and then update dataframe before exporting as .csv

```
data_falcon9 = df[df['BoosterVersion']!='Falcon 1']
```

```
data_falcon9.to_csv('dataset_part\_1.csv', index=False)
```

Use the SpaceX Rest API

API call returns data in .json response

↓

Normalize, flatten, and save as .csv

Data Collection using Web Scraping

GitHub URL

Part 1 - Getting a response from the HTML URL

```
response = requests.get(static_url)
```

Part 2 - Create a BeautifulSoup object from the HTML response

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
html_file = BeautifulSoup(response.text, "html.parser")
```

Part 3 - Find the tables

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = html_file.find_all('table')
```

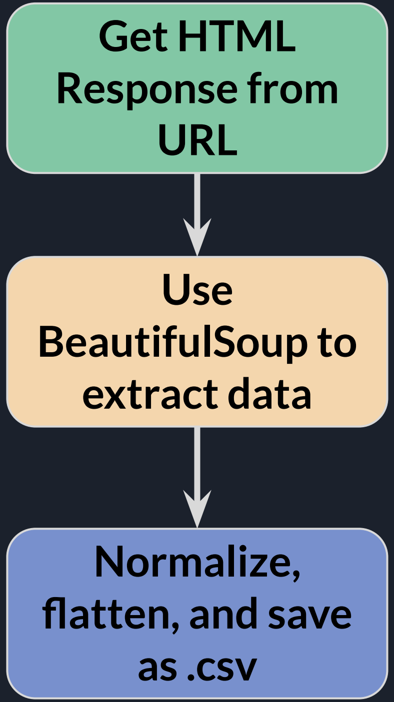
Part 4 - Retrieve the column names

```
column_names = []
# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if (name != None and len(name) > 0):
        column_names.append(name)
```

Part 5 - Create dictionary with keys from the extracted column names convert into a Pandas dataframe. Save the dataframe and export as .csv

```
df=pd.DataFrame(launch_dict)
```

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



Data Wrangling

GitHub URL

In the data set, there are several different cases where the booster did not 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 a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad.True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.

Perform EDA on Dataset

Calculate the number of Launches at Each Site

Calculate the number of occurrence of each orbit

- **Number of missing outcomes per orbit type**
- **Export as .csv**
- **Create landing outcomes labels**
- **Work out success rate**

EDA with Data Visualization

GitHub URL

Bar Graphs

Bar graphs allow for a visual representation of categorical data and the overall value associated with each

- Success Rate of Orbit Type

Line Graphs

Line graphs enable for the visualisation of data over time to understand movement and trends easily in addition to helping with predictions

- Launch Success Yearly Trend

Scatter Graphs

These relationships were graphed using a scatter plot in order to easily identify how much one variable is being affected by the secondary variable. Also, scatter plots are useful when plotting large number of data points

- Flight vs Launch Site
- Payload vs. Launch Site
- Flight vs. Orbit Type
- Payload vs. Orbit Type

EDA with SQL

GitHub URL

- Displaying the names of the unique launch sites in the space mission
- Displaying 5 records where launch sites begin with the string 'KSC’
- 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 where the successful landing outcome in drone ship was achieved.
- Listing the names of the boosters which have success in ground pad 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 records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017
- Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

Interactive Map using Folium

GitHub URL

Mark all launch sites on a map

- Using the lat long coordinates at each launch site, a circle marker was added around each launch site with a label displaying the launch site name

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

- Using the data frame launch_outcomes, we were able to assign a green/red marker on the map in a markercluster displaying the launch outcomes

Calculate the distances between a launch site to its proximities

- Using the coordinates of various landmarks and the haversine formula, we calculated the distance from the launch site to key locations to understand the surrounding area around a launch site

Interactive Dashboard using Dash

GitHub URL

Using Dash web framework, the following components were added into the Dashboard:

1. Pie Chart - displaying the total number of launches by each launch sites

Pie charts are great for displaying proportions of a specific variable

2. Scatter Graphs - representing the relationship between the Landing Outcome and the Payload (KG) for each of the booster versions

Scatter graphs are great for representing the relationship between 2 variables enabling for clear observations and trend analysis

Predictive Analysis - Classification

GitHub URL

Part 1 - Building the Model

- Create a NumPy array from the column Class in data
- We split the data into training and testing data using the function train_test_split
- Models are trained and hyperparameters are selected using the function GridSearchCV

Part 2 - Evaluating the Model

- We display the best parameters using the data attribute best_params_ and the accuracy on the validation data using the data attribute best_score_
- Plot Confusion Matrix

Part 3 - Improving the Model

- Feature Engineering
- Algorithm Tuning

Part 4 - Identifying the best performing Classification Model

- The best performing Model is determined by the best accuracy score

Results

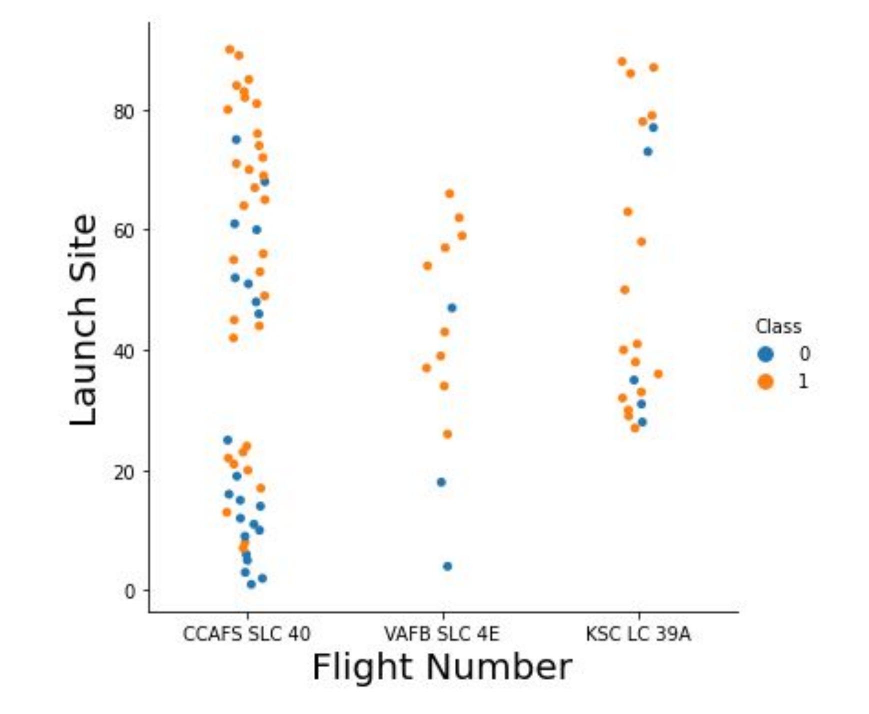
1. Exploratory data analysis results
2. Interactive analytics demo in screenshots
3. Predictive analysis results

Insights drawn from EDA

Flight Number vs. Launch Site

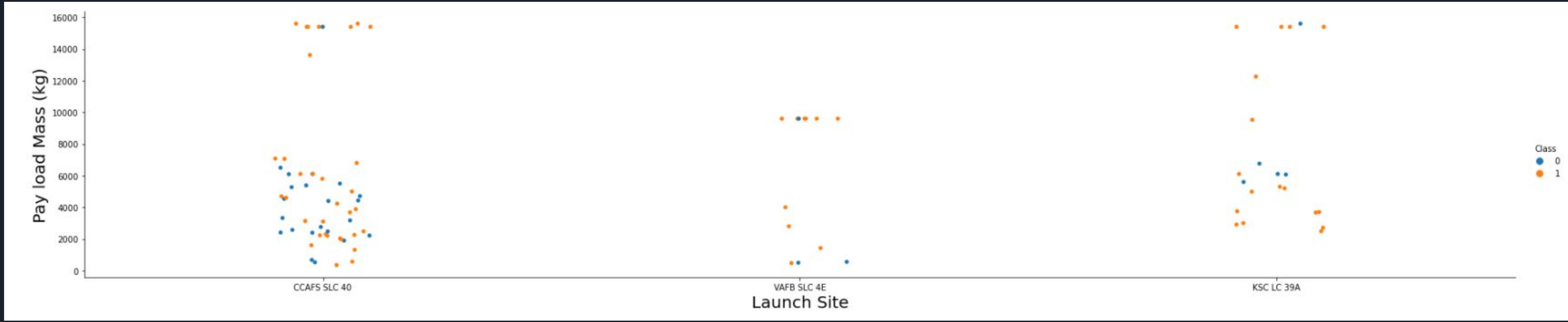
Using the function catplot to plot Flight Number vs LaunchSite, set the parameter x parameter to Flight Number,set the y to Launch Site and set the parameter hue to 'class'.

There seems to be a relationship where if the number of flights increases, there is a greater success rate



Payload vs. Launch Site

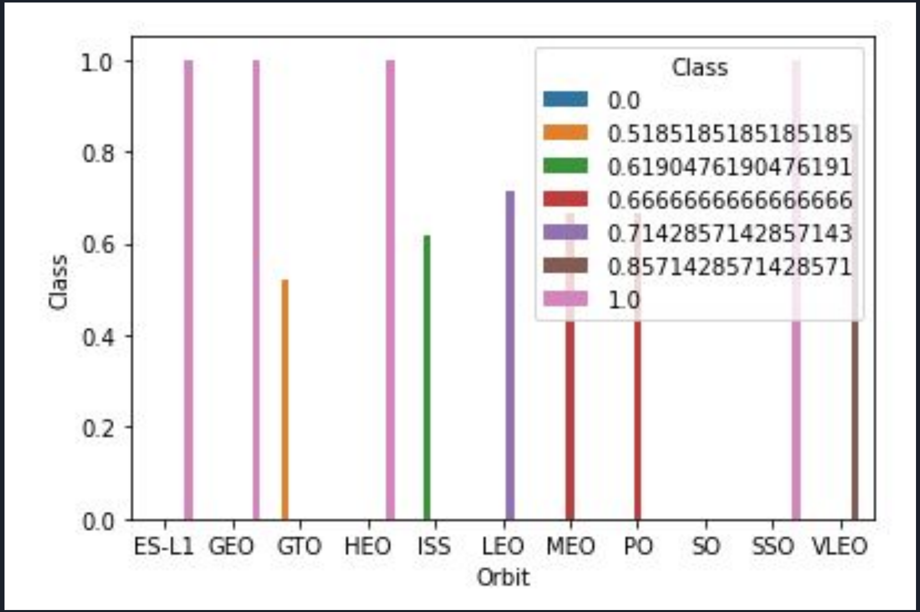
The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket. Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000)



Success Rate of each Orbit Type

Next, we want to visually check if there are any relationship between success rate and orbit type.

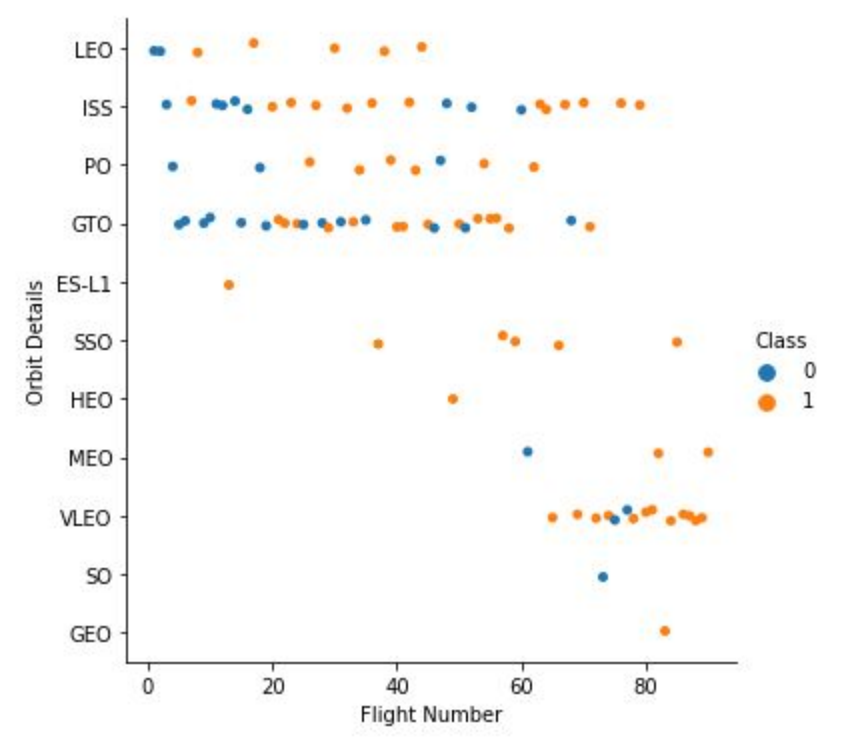
Orbit GEO,HEO,SSO,ES-L1 has the best SuccessRate



Orbit vs. Flight Number

We want to see if there is any relationship between FlightNumber and Orbit type.

LEO orbit success rate is related to the number of flights. No relationship between flight number within GTO orbit.

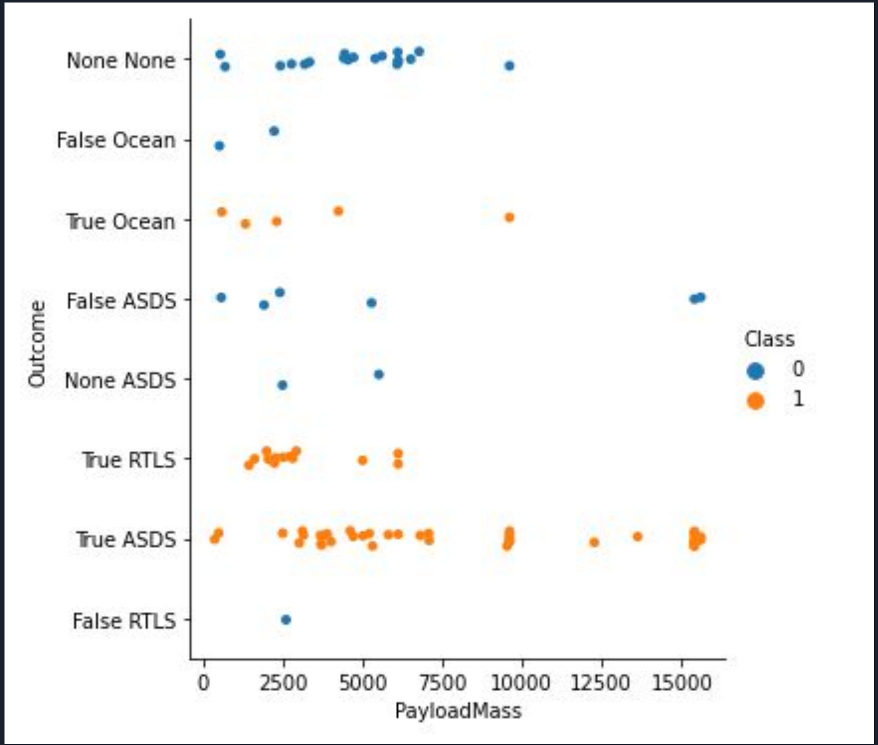


Payload vs. Orbit

We can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type.

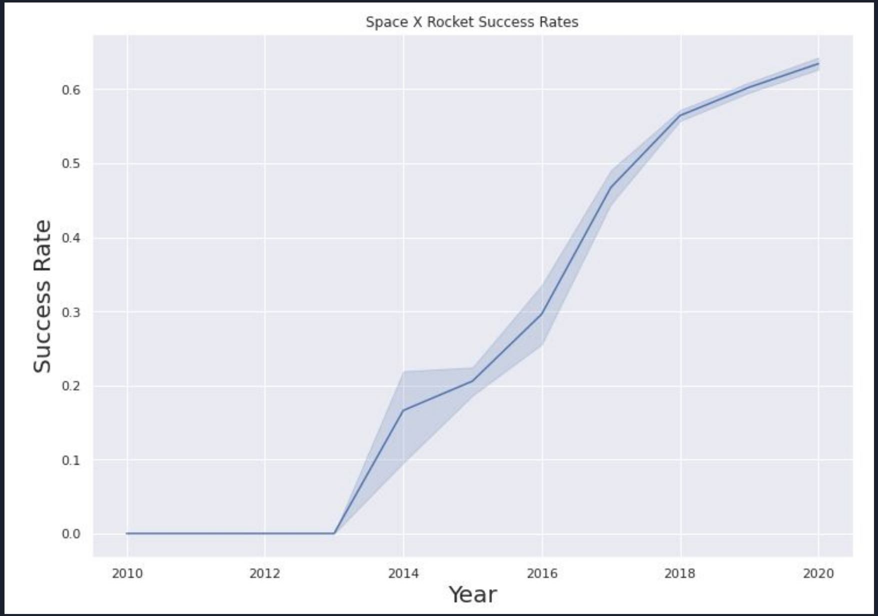
With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.

However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.



Launch Success Yearly Trend

You can observe that the success rate since 2013 kept increasing till 2020



EDA with SQL Queries

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

Query Explanation:

The syntax distinct allows for the unique list of launch sites to be returned

```
%sql select count(*),LAUNCH_SITE from SPACEXDATASET GROUP BY LAUNCH_SITE
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

1	launch_site
26	CCAFS LC-40
34	CCAFS SLC-40
25	KSC LC-39A
16	VAFB SLC-4E

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

Query Explanation:

The top 5 syntax allows for the top 5 records to be returned in the query where launch site contains the string ‘CCA’

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

```
]: %sql select * from SPACEXDATASET WHERE LAUNCH_SITE like 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

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

Query Explanation:

The sum function allows for the total value of payload to be returned based on the where clause for boosters launched by NASA (CRS)

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXDATASET WHERE customer = 'NASA (CRS)'
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

1
45596

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

Query Explanation:

The average function allows for the average value of payload to be returned based on the where clause for booster version F9 1.1

```
%sql SELECT avg(PAYLOAD_MASS__KG_) FROM SPACEXDATASET where booster_version = 'F9 v1.1'
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

1
2928

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

Query Explanation:

The min date function allows for the query to return the first successful landing date (first date) where ground pad was achieved

```
%sql SELECT min(DATE) from SPACEXDATASET WHERE landing__outcome = 'Success (ground pad)'
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

1
2015-12-22

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

Query Explanation:

Using 2 rules in the where clause we can filter the results for all boosters that have a success in drone ship in addition to payload mass between 4000-6000

```
%sql select BOOSTER_VERSION from SPACEXDATASET where LANDING__OUTCOME='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

List the total number of successful and failure mission outcomes

Query Explanation:

The group by function allows us to aggregate the total number of missions by the mission outcomes

```
%sql select MISSION_OUTCOME,count(MISSION_OUTCOME) as missionoutcomes from SPACEXDATASET GROUP BY MISSION_OUTCOME
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

mission_outcome	missionoutcomes
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

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

Query Explanation:

Using a subquery we can determine the maximum payload mass carried by each booster, which is then used in the main query to list the names of the booster versions which carried the max

```
%sql select BOOSTER_VERSION as boosterversion from SPACEXDATASET where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXDATASE
T)
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

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

Query Explanation:

Extracting the year from the date field we can use that in the where clause to easily fitler the data for all results in 2015

```
%sql SELECT MONTH(DATE),MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXDATASET where EXTRACT(YEAR FROM DATE)='2015' AND landing__
outcome = 'Failure (drone ship)'
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

1	mission_outcome	booster_version	launch_site
1	Success	F9 v1.1 B1012	CCAFS LC-40
4	Success	F9 v1.1 B1015	CCAFS LC-40

Rank the count of landing outcomes between the date 2010-06-04 and 2017-03-20

```
%sql SELECT LANDING__OUTCOME FROM SPACEXDATASET WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' ORDER BY DATE DESC
```

```
 * ibm_db_sa://lbp26139:***@ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31505/bludb
Done.
```

landing__outcome
No attempt
Success (ground pad)
Success (drone ship)
Success (drone ship)
Success (ground pad)
Failure (drone ship)
Success (drone ship)
Success (drone ship)
Success (drone ship)
Failure (drone ship)
Failure (drone ship)
Success (ground pad)
Precluded (drone ship)
No attempt
Failure (drone ship)
No attempt
Controlled (ocean)
Failure (drone ship)
Uncontrolled (ocean)
No attempt
No attempt
Controlled (ocean)
Controlled (ocean)
No attempt
No attempt
Uncontrolled (ocean)
No attempt
No attempt
No attempt
Failure (parachute)
Failure (parachute)

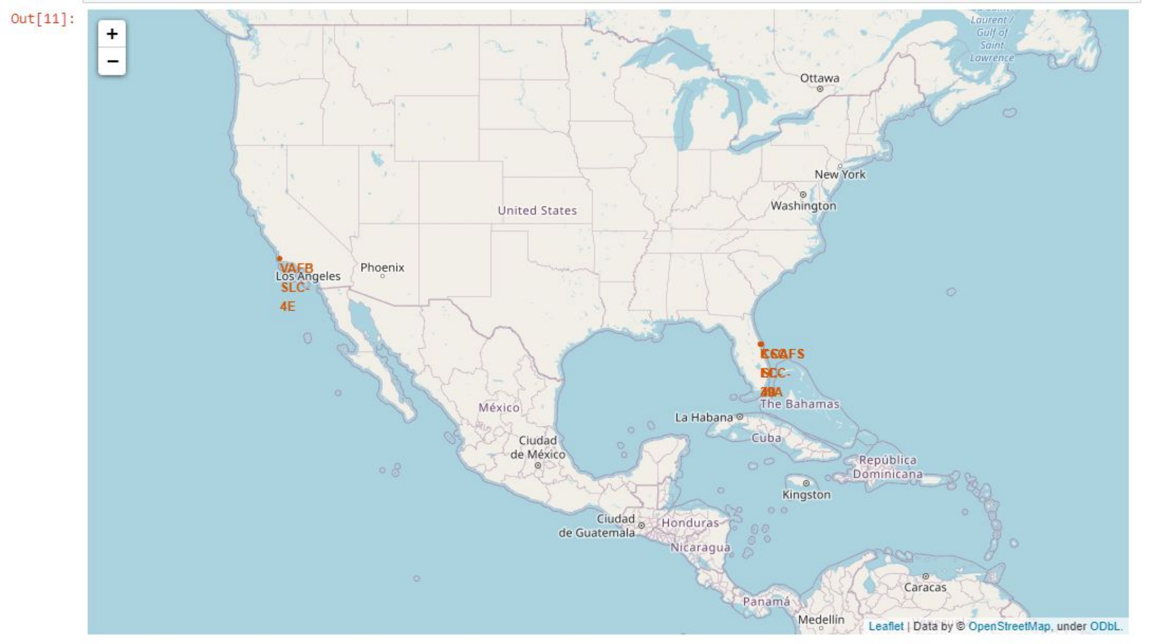
Query Explanation:

Using the order by function on the date field we can chronologically rank the results between the 2 dates specified in the where clause

Launch Sites Proximities Analysis - Interactive Maps with Folium

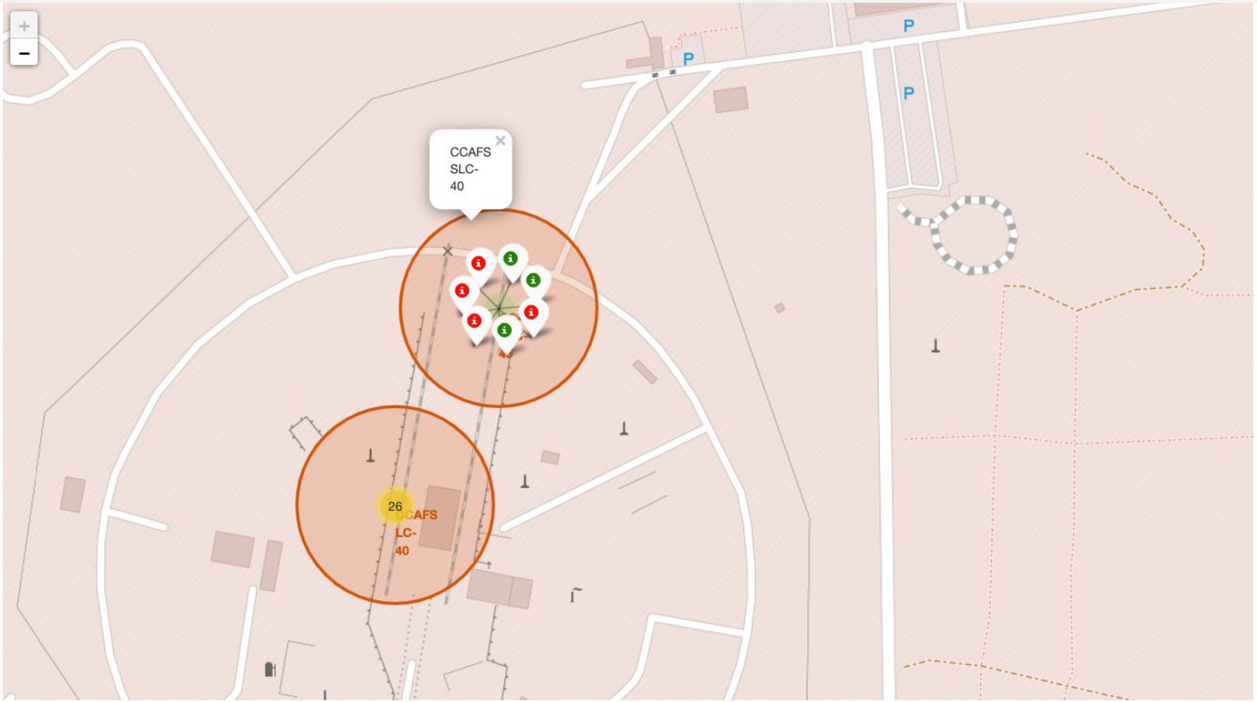
Launch Sites using Markers

We can see that the SpaceX launch sites are in the United States



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

We can see that the SpaceX launch sites are in the United States

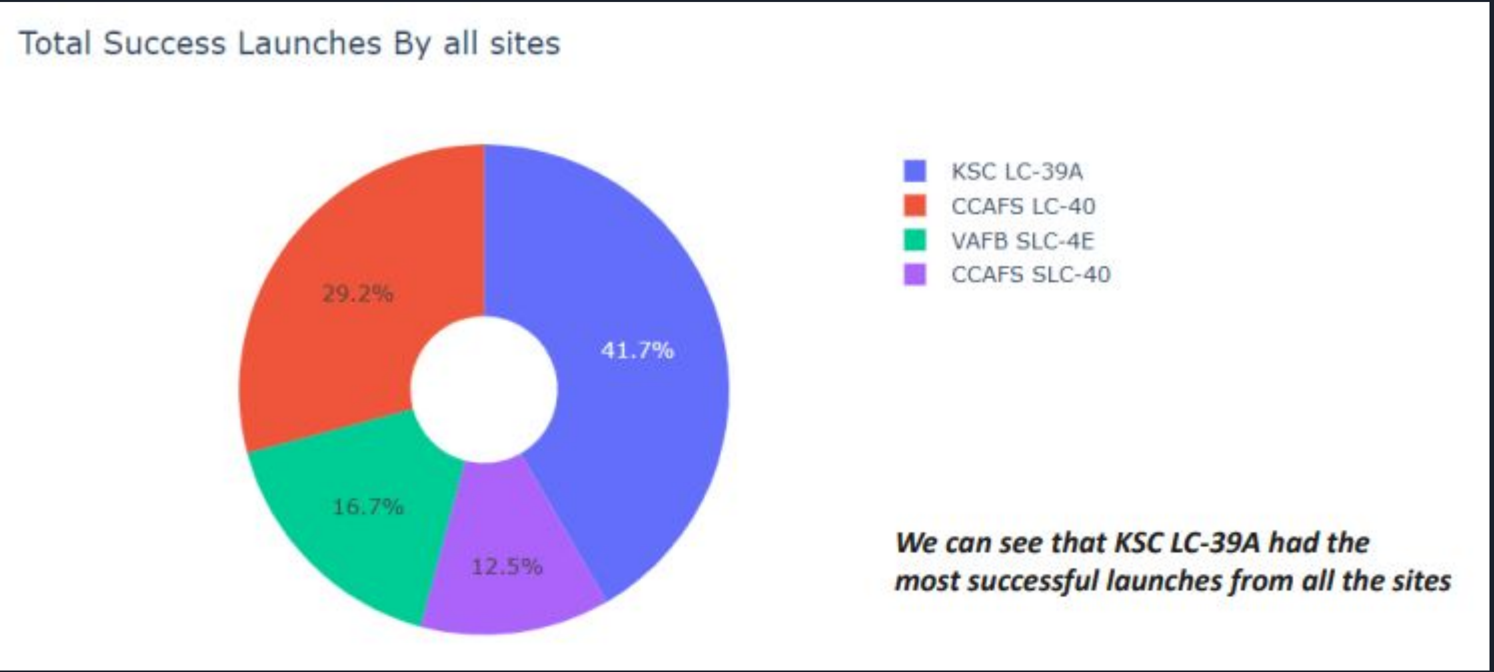


Calculate the distances between a launch site to its proximities

Build a Dashboard with Plotly Dash

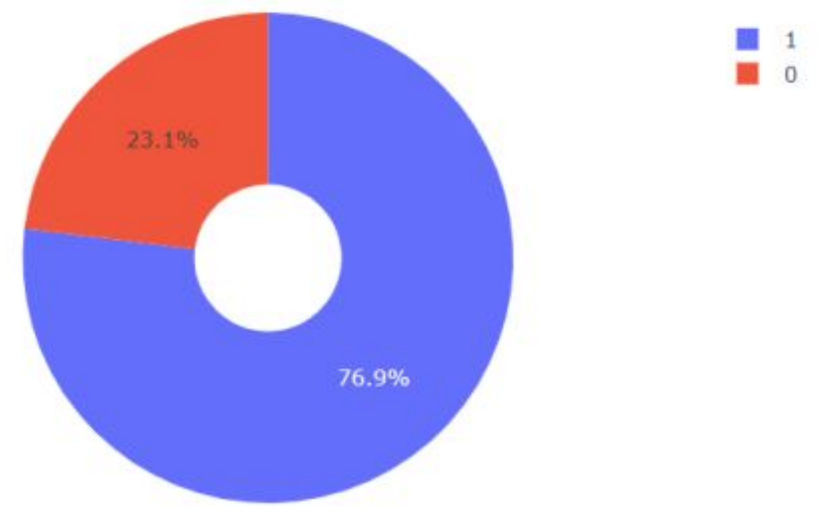
Launch Sites using Markers

We can see that the SpaceX launch sites are in the United States



Launch Sites using Markers

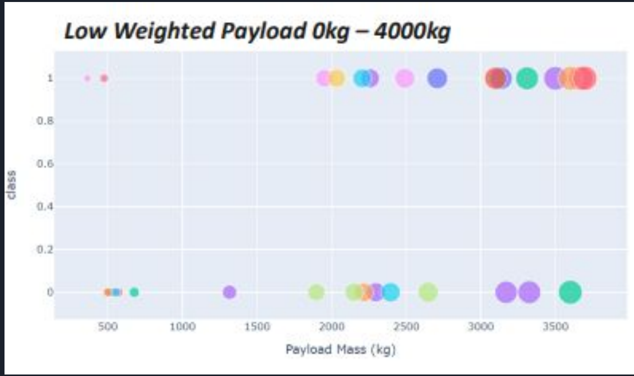
We can see that the SpaceX launch sites are in the United States

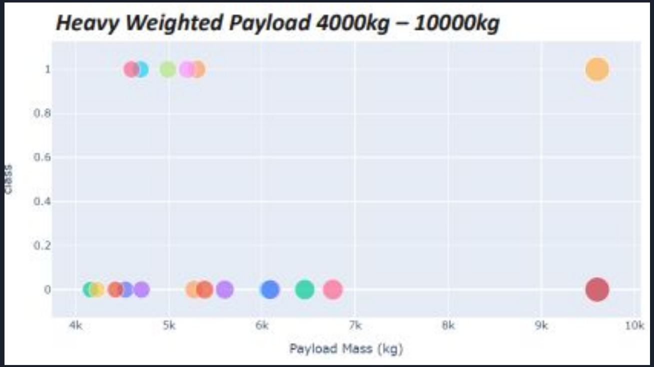


KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Launch Sites using Markers

We can see that the SpaceX launch sites are in the United States

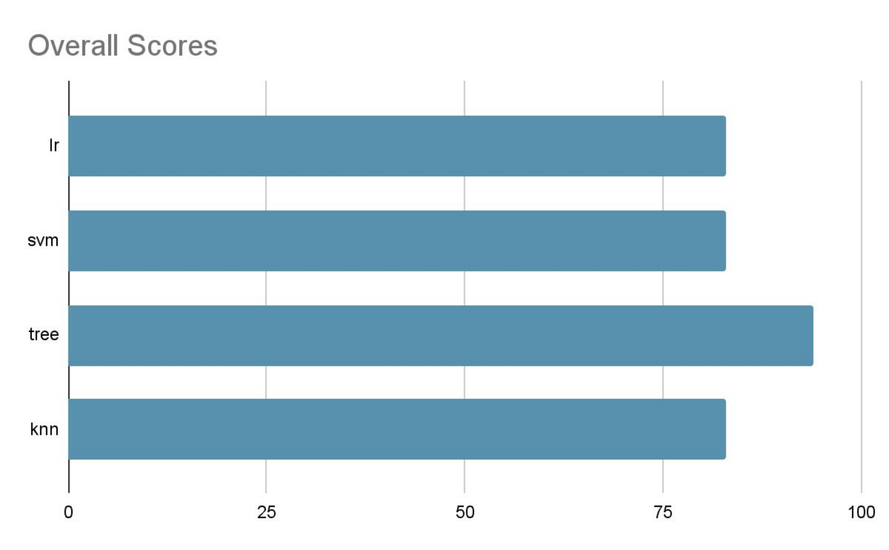




Predictive Analysis - Classification

Classification Accuracy

Based on the function below to determine the best method, we can see that the tree classification performed the best at 94%



TASK 12

Find the method performs best:

In [32]:
```
scores = [lr_score,svm_score,tree_score,knn_score]
print(scores)
print(scores.index(max(scores)))
```

```
[0.8333333333333334, 0.8333333333333334, 0.9444444444444444, 0.8333333333333334]
2
```

Confusion Matrix

Based on the confusion matrix, we can see the main area for concern is the top 2 sections which equates to False Positive.

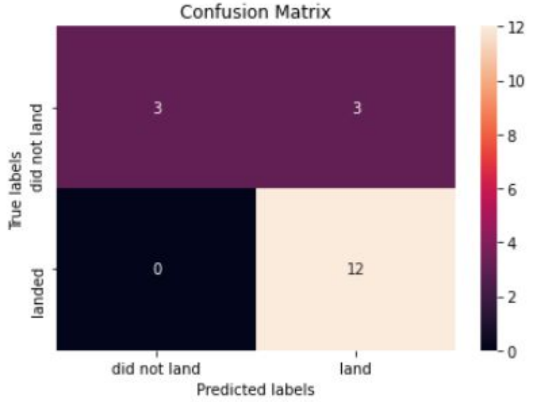
Calculate the accuracy of tree_cv on the test data using the method `score`:

```
In [30]: knn_score = knn_cv.score(X_test,Y_test)
         knn_score
```

```
Out[30]: 0.8333333333333334
```

We can plot the confusion matrix

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



Conclusion

- The Tree Classifier Algorithm proves to be the best Machine Learning algorithm for this dataset
- Low weighted payloads perform better
- The success rates for SpaceX launches is directly proportional time in years
- We can see that KSC LC-39A had the most successful launches from all the sites
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

Appendix

Github Library

https://github.com/hvmistry/IBMWatsonStudio/tree/master