Winning Space Race with Data Science

CHHAVI

22 APRIL 2023

Let’s Dive In

Executive Sumary

Summary of Methodologies:

The course covers a range of methodologies used in data science, including:

- Data collection and data cleaning
- Data exploration and visualization
- Data analysis and statistical inference
- Machine learning algorithms and models
- Data storytelling and communication

Later in the upcoming slides you'll understand every point of this sub-topic.

Summary of All Results:

This Project covers almost every part of the entire IBM course on DATA SCIENCE certifiction.

Here is what comes next :

1. *Exploratory Data Analysis and feature engineering*
   (Here , we will predict if the Falcon 9 first stage will land successfully.)

2. *Data wrangling*
   (Space X Falcon 9 First Stage Landing Prediction)

3. *Launch Sites Locations Analysis with Folium*
   (analysis & visualization using Python's library)

4. *Machine Learning Prediction*
   (Space X Falcon 9 First Stage Landing Prediction)

Introduction

Project background and context

The commercial space industry is experiencing a boom with companies like Virgin Galactic, Rocket Lab, Blue Origin, and SpaceX offering space travel and satellite services to the masses. Among these companies, SpaceX has been particularly successful in its endeavors, with accomplishments such as sending spacecraft to the International Space Station, launching manned missions, and creating a satellite internet constellation called Starlink. One of the reasons for SpaceX's success is the relatively low cost of their Falcon 9 rocket launches, which can be attributed to the reuse of the first stage of the rocket.

Problems you want to find answers

1. We will predict if the Falcon 9 first stage will land successfully
2. we will perform some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
3. If we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch.
4. Finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations.

METHODOLOGY

Executive Summary

Data collection methodology:

1. **Describe how data was collected**
2. **Perform data wrangling**
3. **Describe how data was processed**
4. **Perform exploratory data analysis using visualization &SQL**
5. **Perform interactive visual analytics using Folium & Plotly Dash**
6. **Perform predictive analysis using classification models**
7. **How to build, tune, evaluate classification models**

Data Collection

DATA COLLECTION USING 'SPACEX API'

Here we will collect te data using an API, specifically the SpaceX REST API, to gather information about launches, including rocket, payload, launch and landing specifications, and landing outcome. The endpoint used is api.spacexdata.com/v4/launches/past, and the JSON response obtained can be converted into a dataframe using the json_normalize function.

DATA COLLECTION USING WEB SCRAPING

Web scraping is also mentioned as another source of data, using Python BeautifulSoup to extract Falcon 9 launch records from HTML tables and convert them into a Pandas dataframe. The collected data needs to be cleaned, including filtering out Falcon 1 launches and dealing with null values in the PayloadMass column. The column LandingPad with null values is left as is, and will be addressed later with one hot encoding.

DATA COLLECTION USING DATA WRANGLING

Your text has been concise and beautiful, but tHere we'll analyse attributes involved in data wrangling. These attributes include Flight Number, Date, Booster version, Payload mass, Orbit, Launch Site, Outcome, Flights, Grid Fins, Reused, Legs, Landing pad, Block, Reused count, Serial, Longitude, and latitude of launch.The Outcome column indicates whether the first stage of the launch was successful or nothe information is inextricably inextricable and needs to be expressed in more words

EDA WITH DATA VISUALIZATION

STEP 1

Analyzing and visualizing data to gain insights and understand the underlying patterns and relationships. In the context of SpaceX's Falcon 9 launches, EDA can help determine if the first stage can be reused and predict if it will land successfully.

Some of the attributes that can be used for EDA include Flight Number, Date, Booster version, Payload mass Orbit, Launch Site, Outcome, Grid Fins, Reused, Legs, Landing pad, Block, Reused count, Serial, Longitude and latitude of launch.

STEP 2

By analyzing the data, we can see that the success rate of Falcon 9 launches has improved since 2013. Launch Number can also be used as a feature to predict successful landings. Additionally, different launch sites have different success rates, which can also be used as a feature.

When we combine multiple features and overlay the landing outcomes as a color, we can see even more patterns. For example, CCAFS LC-40 has a success rate of 60%, but if the payload mass is above 10,000 kg, the success rate is 100%.

STEP 3

To prepare the data for machine learning, categorical variables will be converted using one hot encoding. The ultimate goal is to develop a model that can predict if the first stage will land successfully based on various attributes.

DATA DATA COLLECTION USING 'SPACEX API'

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

response = requests.get(spacex_url)

# Use json_normalize meethod to convert the json
data = pd.json_normalize(response.json())
```

API GET REQUEST & NORMALIZING THE DATA

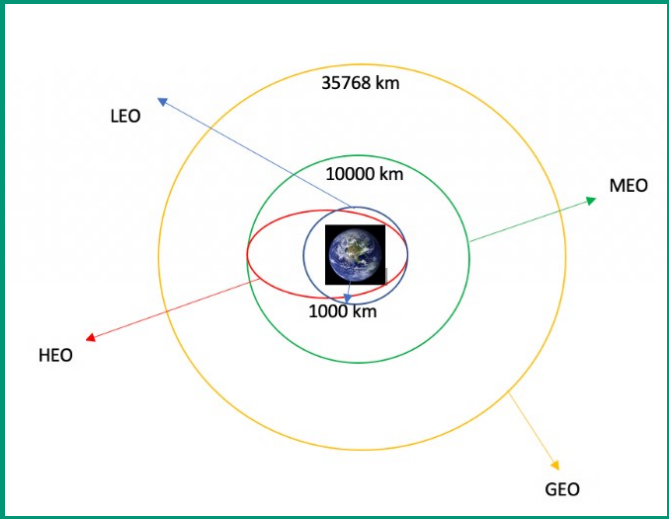
```
#Global variables
BoosterVersion = []
PayloadMass = []
Orbit = []
LaunchSite = []
Outcome = []
Flights = []
GridFins = []
Reused = []
Legs = []
LandingPad = []
Block = []
ReusedCount = []
Serial = []
Longitude = []
Latitude = []
```

Global variables that stores global data

```
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}
```

Combining column & data gathered into dictionary

DATA DATA COLLECTION USING 'SPACEX API'



TASK 4: Create a landing outcome label from Outcome column

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
# Create a set of bad outcomes
bad_outcomes = set(landing_outcomes.keys()[[1, 3, 5, 6, 7]])

# Create a list of landing outcomes
landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome']]

# Print the first 10 elements of the landing_class list
print(landing_class[:10])
```

```
[0, 0, 0, 0, 0, 0, 1, 1, 0, 0]
```

This variable will represent the classification variable that represents the outcome of each launch.
If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

```
df["Class"].mean()
```

```
0.6666666666666666
```

We can now export it to a CSV for the next section,
but to make the answers consistent, in the next lab we
will provide data in a pre-selected date range.

```
df.to_csv("dataset_part_2.csv", index=False)
```

Each launch aims to an dedicated orbit, and here are some common orbit types

If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

We can use the above line of code to determine the success rate

SCRAPING

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

html_data = requests.get(static_url).text
```

Falcon 9 home page request

```
html_tables = soup.find_all ('table')

column_names = []

# Apply find_all() function with `th` element on firs
# Iterate each th element and apply the provided extr
# Append the Non-empty column name (`if name is not N
colnames = soup.find_all('th')
for x in range (len(colnames)):
    name2 = extract_column_from_header(colnames[x])
    if (name2 is not None and len(name2) > 3):
        column_names.append(name2)
```

Found tables from wiki & made columns for our data

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with eac
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

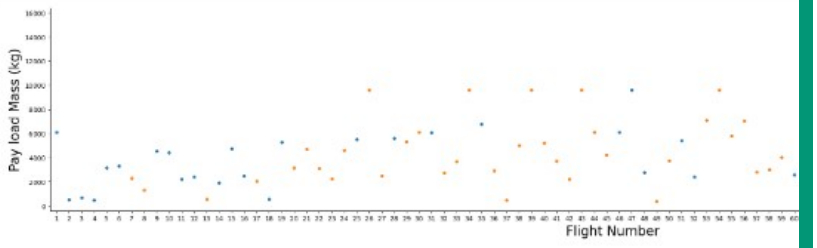
Create empty dictionary from keys

```
def date_time(table_cells):

def booster_version(table_cells):

def landing_status(table_cells):

def get_mass(table_cells):
```

Fill data dictionary from launch data

Exploring and Preparing Data

```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



We see that different launch sites have different success rates. `CCAFS LC-40` , has a success rate of 60 %, while `KSC LC-39A` and `VAFB SLC 4E` has a success rate of 77%.

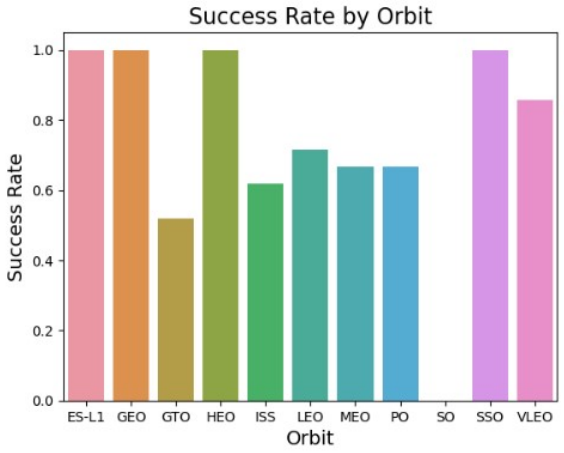
We see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.

TASK 3: Visualize the relationship between success rate of each orbit type

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

Let's create a `bar chart` for the sucess rate of each orbit

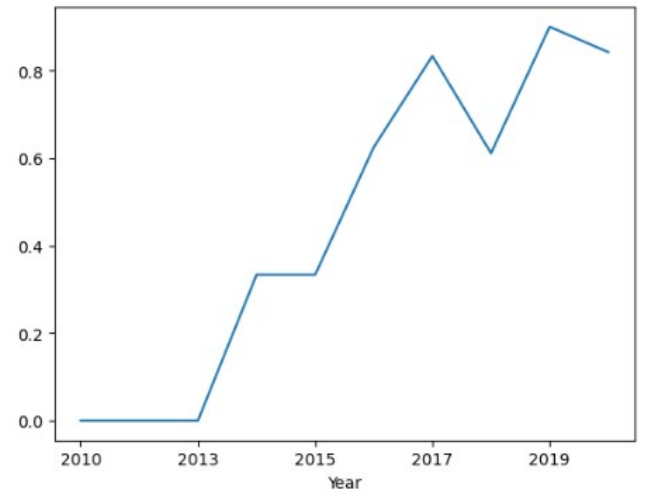
```
n [6]: # HINT use groupby method on Orbit column and get the mean of Class column
       # Group the data by Orbit and calculate the mean of Class to get the success rate f
       orbit_success_rate = df.groupby('Orbit')['Class'].mean()

       # Plot a bar chart of the success rate for each orbit
       sns.barplot(x=orbit_success_rate.index, y=orbit_success_rate.values)
       plt.xlabel('Orbit', fontsize=14)
       plt.ylabel('Success Rate', fontsize=14)
       plt.title('Success Rate by Orbit', fontsize=16)
       plt.show()
```



Visualize the relationship between success rate of each orbit type

```
temp_df = df.copy()
temp_df['Year'] = year
temp_df.groupby('Year')['Class'].mean().plot()
```

```
<Axes: xlabel='Year'>
```



you can observe that the sucess rate since 2013 kept increasing till 2020

Final results: you can observe that the sucess rate since 2013 kept increasing till 2020.
GOT A PATTERN

Data Wrangling

```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.a
art_1.csv")
```

Load SpaceX

```
df['Orbit'].value_counts()
```

GTO	27
ISS	21
VLEO	14
PO	9
LEO	7
SSO	5
MEO	3
GEO	1
HEO	1
SO	1
ES-L1	1

```
landing_class = []
for i in df['Outcome']:
    if i in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

```
df['LaunchSite'].value_counts()
```

```
landing_outcomes = df['Outcome'].value_counts()
```

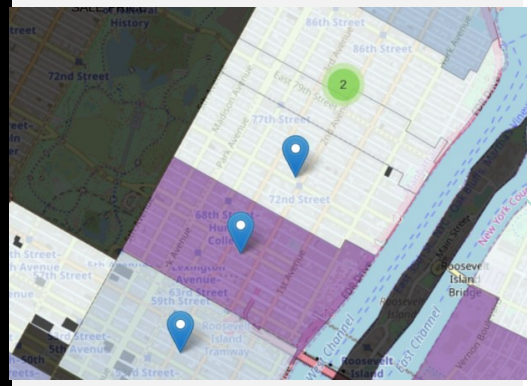
Find Data Pattern

```
df['Class']=landing_class
df[['Class']].head(8)
```

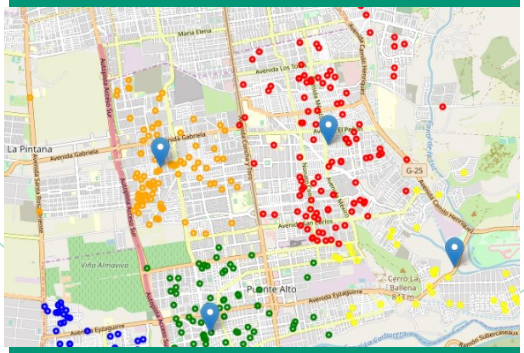
Create landing Outcome

INTERACTIVE VISUAL ANALYTICS & DASHBOARDING

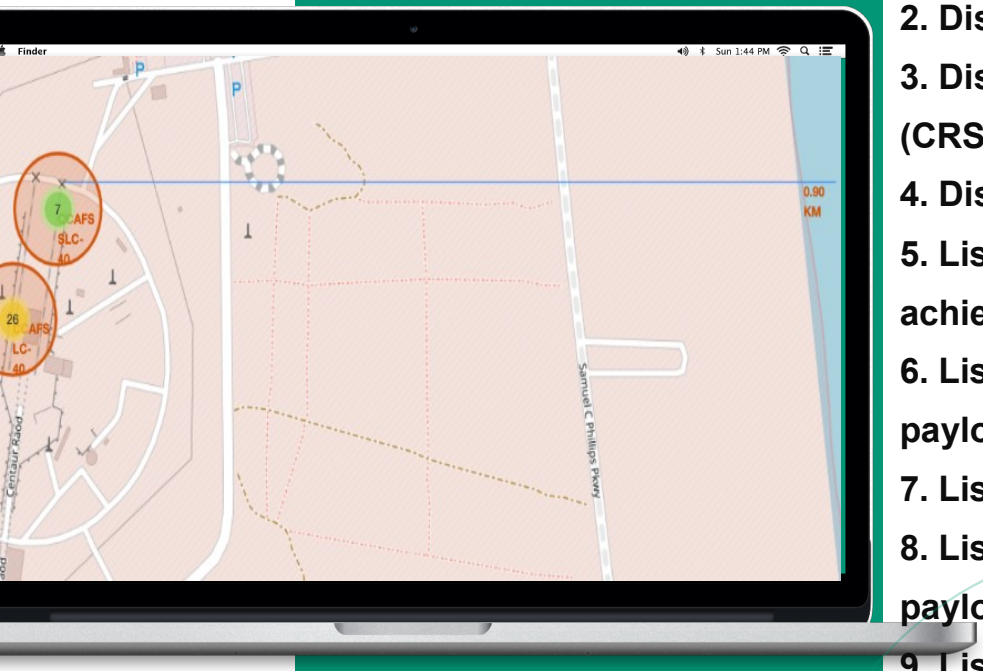
This approach allows for exploring and manipulating data in real-time through common interactions such as zooming, filtering, searching, and linking. You will use tools like Folium and Plotly Dash to build interactive maps and dashboards. The first part of the module focuses on analyzing launch site geos and proximities with Folium,
While the second part is about building a dashboard application with Plotly Dash that contains input components to interact with charts.
The goal is to use interactive visual analytics to discover patterns and insights in the SpaceX dataset more effectively than with static graphs.



We'll do something like this



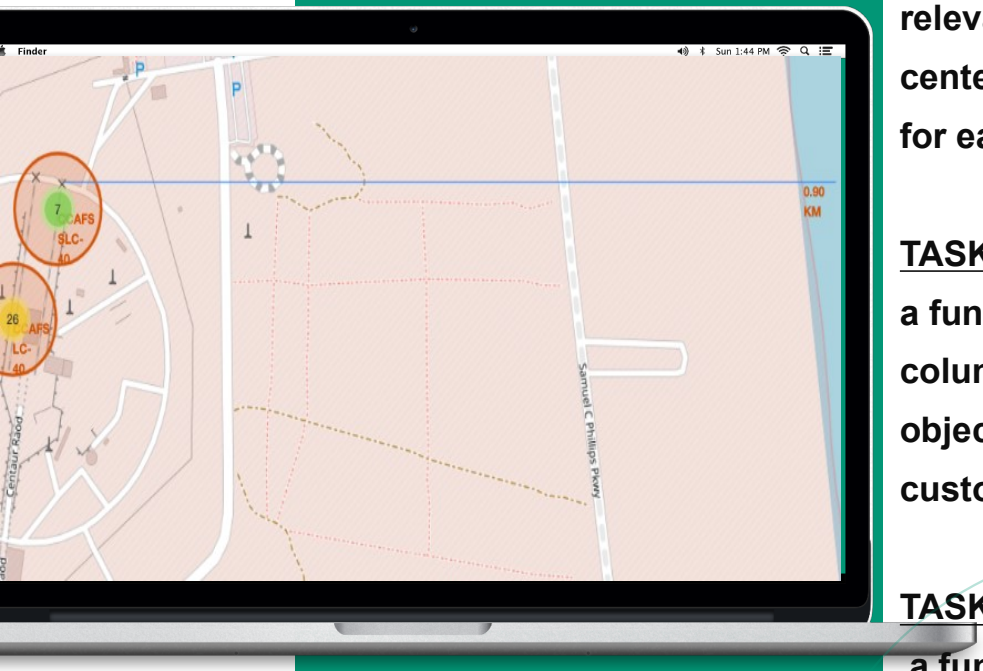
ie. We'll create markers of this sort using folium



EDA WITH SQL

To understand SpaceX data set, following SQL queries were performed on the jupyter lab notebook:

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
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 or Success between the date 2010-06-04 and 2017-03-20, in descending order



Build an Interactive Map with Folium

TASK 1: Mark all launch sites on a map

The lab gives dataset spacex_launch_geo.csv, which is an augmented dataset with latitude and longitude. We oad dataset using Pandas and select relevant columns.A circle and marker are added to the map to represent the center. Next, a marker cluster is added to the map, and
for each launch site, a Circle object is created based on its coordinate.

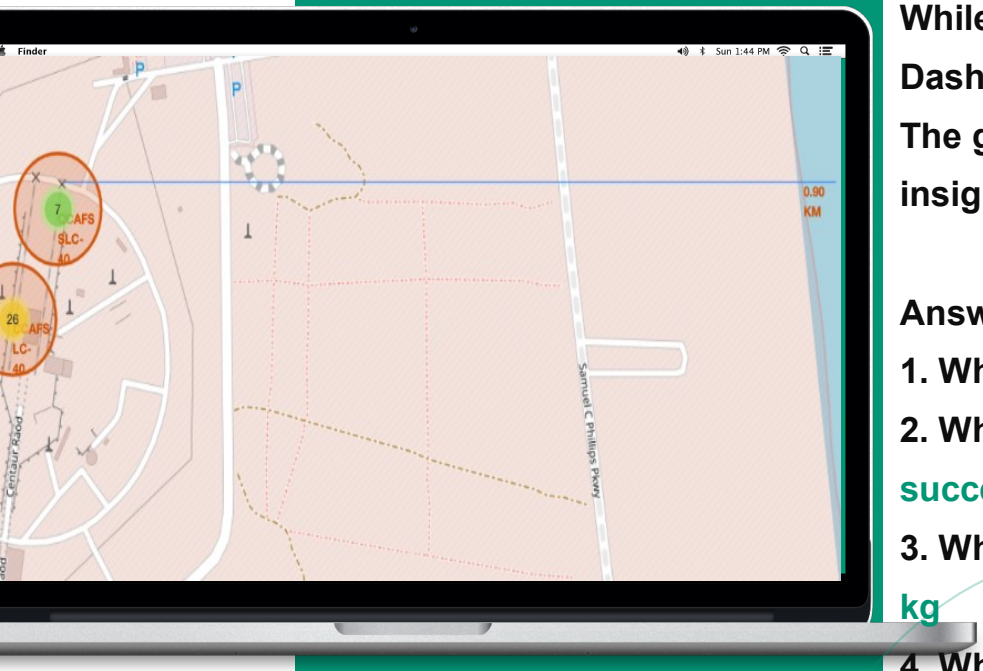
TASK 2: Mark the success/failed launches for each site on the map

a function is created to assign a color to the launch outcome. Then, a new column marker_color is added to the dataframe. For each row , a Marker object is created with its coordinate, and the Marker's icon property is customized to indicate if the launch was successful or failed.

TASK 3: Calculate the distances between a launch site to its proximities

a function is created to calculate the distance between two points on the earth's surface. The function takes four arguments,
the latitude and longitude values for the two points. Then, the distance is calculated using the Haversine formula.

Build a Dashboard with Plotly Dash



We used tools like Folium & Plotly Dash to build interactive maps and dashboards. In the 1st part we analyzing launch site with Folium,

While the second part is about building a dashboard application with Plotly Dash that contains input components to interact with charts.
The goal is to use interactive visual analytics to discover patterns and insights in the SpaceX dataset more effectively than with static graphs.

Answer to following questions using Dashboard:

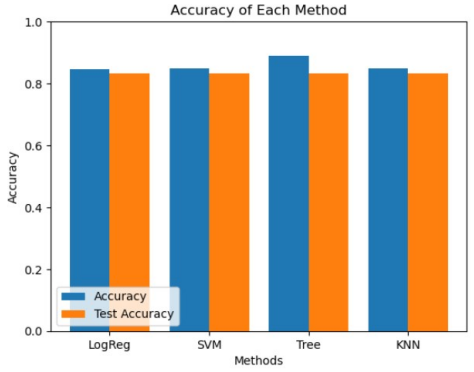
1. Which site has the largest successful launches? KSCLC-39A with 10
2. Which site has the highest launch success rate? KSCLC-39A with 76.9% success
3. Which payload range(s) has the highest launch success rate? 2000 – 5000 kg
4. Which payload range(s) has the lowest launch success rate? 0-2000 and 5500 - 7000
5. Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest launch success rate? FT

Predictive Analysis and Building a Machine Learning Pipeline

```
data = pd.read_csv("https://cf-courses-data.s3.us.cloud-object
et_part_2.csv")
```

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

```
X_train, X_test, Y_train, Y_test =
train_test_split(X, Y, test_size=0.2, random_state=2)

Y_test.shape
parameters ={'C':[0.01,0.1,1],'penalty':['l2'],
'solver':['lbfgs']}
parameters ={"C":[0.01,0.1,1],'penalty':['l2'],
'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()
```

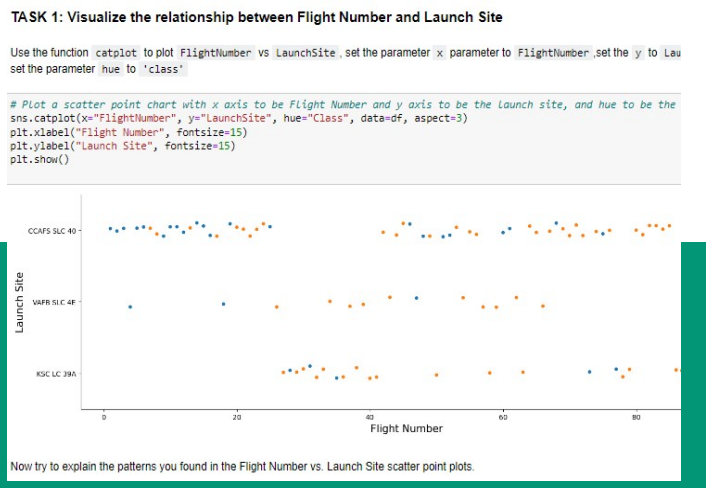


Process of building a machine learning pipeline to predict the successful landing of the first stage of the Falcon 9

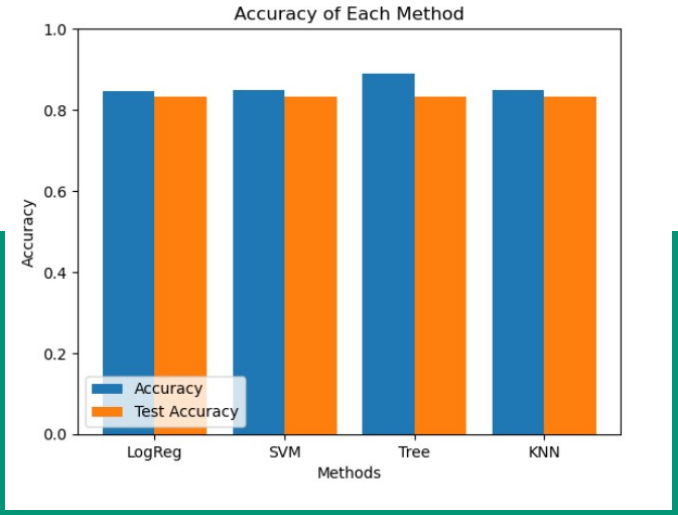
The process will include preprocessing and train_test_split to standardize and split the data, respectively. The model will be trained and Grid Search will be performed to find the best hyperparameters.

Different algorithms will be tested, including Logistic Regression, Support Vector Machines, Decision Tree Classifier, and K-nearest neighbors. Finally, the confusion matrix will be generated to evaluate the model's accuracy.

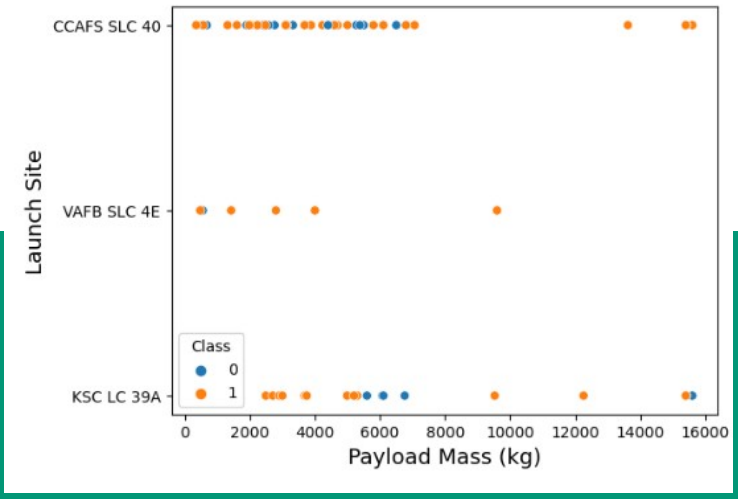
RESULTS



EDA analysis results



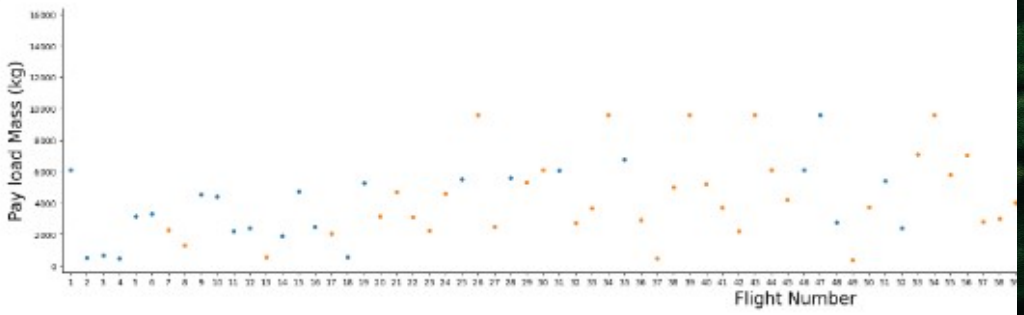
Predictive analysis results



Analytics demo

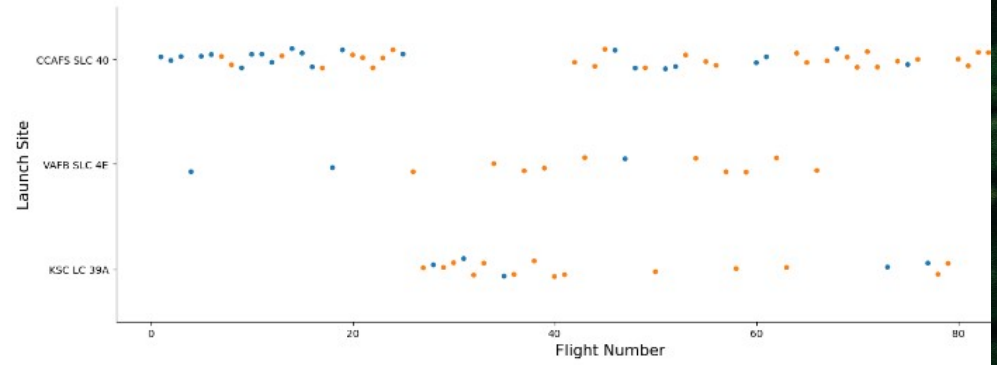
EDA FINDINGS

```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



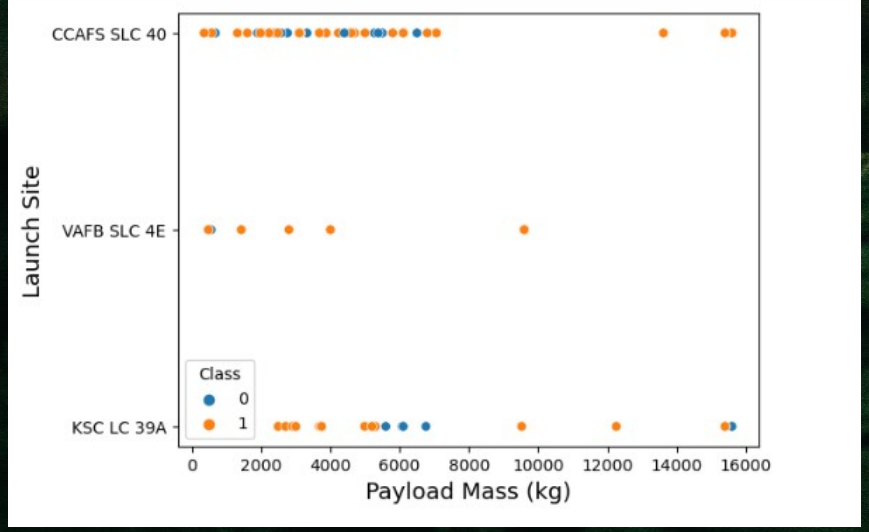
Flight no & payload

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be
sns.catplot(x="FlightNumber", y="LaunchSite", hue="Class", data=df, aspect=3)
plt.xlabel("Flight Number", fontsize=15)
plt.ylabel("Launch Site", fontsize=15)
plt.show()
```



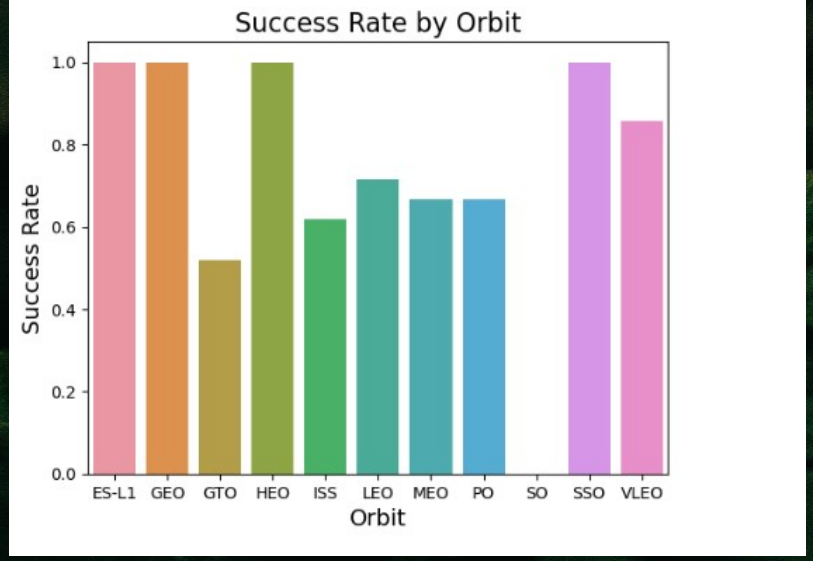
Visualize the relationship between Flight Number and Launch Site

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch s
sns.scatterplot(x="PayloadMass", y="LaunchSite", hue="Class", data=df)
plt.xlabel("Payload Mass (kg)", fontsize=14)
plt.ylabel("Launch Site", fontsize=14)
plt.show()
```



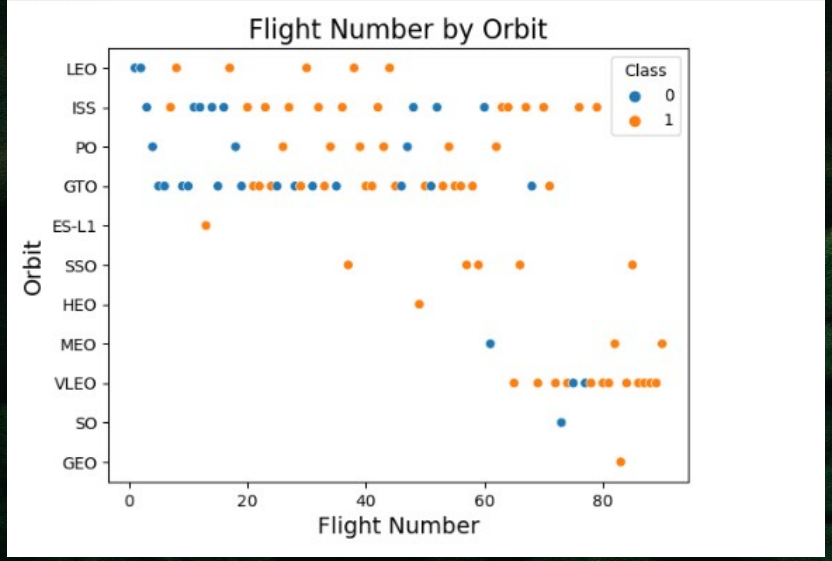
Visualize the relationship between Payload and Launch Site

```
# HINT use groupby method on Orbit column and get the mean of Class column
# Group the data by Orbit and calculate the mean of Class to get the success rate for eac
orbit_success_rate = df.groupby('Orbit')['Class'].mean()

# Plot a bar chart of the success rate for each orbit
sns.barplot(x=orbit_success_rate.index, y=orbit_success_rate.values)
plt.xlabel('Orbit', fontsize=14)
plt.ylabel('Success Rate', fontsize=14)
plt.title('Success Rate by Orbit', fontsize=16)
plt.show()
```



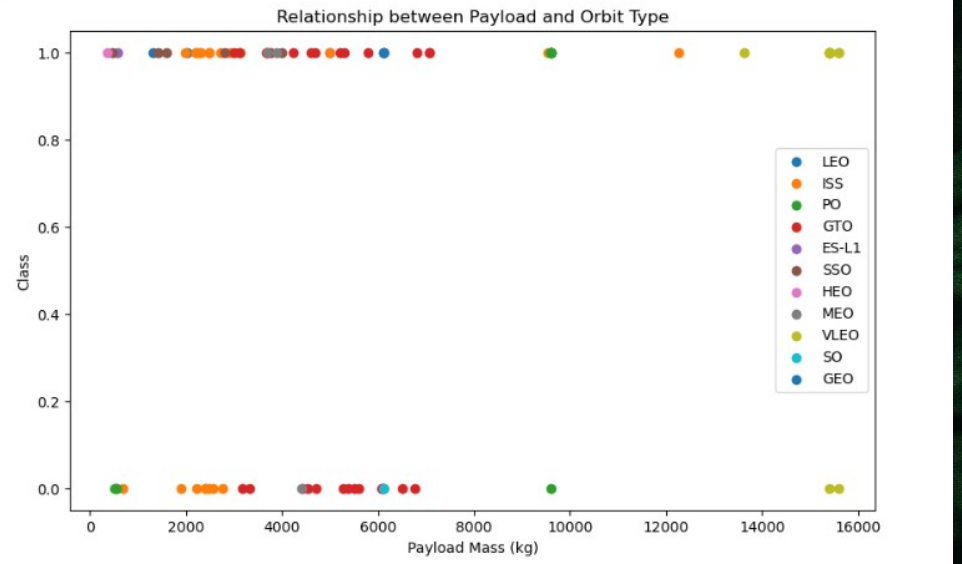
Visualize the relationship between success rate of each orbit type

```
import seaborn as sns

sns.scatterplot(data=df, x='FlightNumber', y='Orbit', hue='Class')
plt.xlabel('Flight Number', fontsize=14)
plt.ylabel('Orbit', fontsize=14)
plt.title('Flight Number by Orbit', fontsize=16)
plt.show()
```



Visualize the relationship between FlightNumber and Orbit type

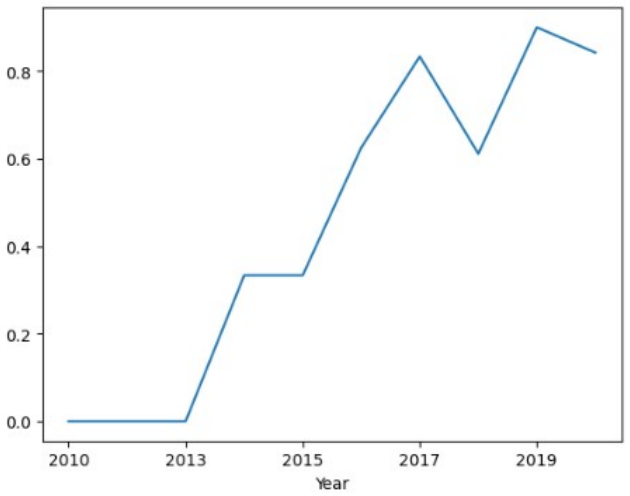
```
# Create subplots
fig, ax = plt.subplots(figsize=(10, 6))
# Get unique orbit types
orbit_types = df["Orbit"].unique()
# Create scatter plots for each orbit type
for orbit in orbit_types:
    df_orbit = df[df["Orbit"] == orbit]
    ax.scatter(df_orbit["PayloadMass"], df_orbit["Class"], label=orbit)
# Add axis labels
ax.set_xlabel("Payload Mass (kg)")
ax.set_ylabel("Class")
# Add title and legend
ax.set_title("Relationship between Payload and Orbit Type")
ax.legend()
plt.show()
```



Visualize the relationship between Payload and Orbit type

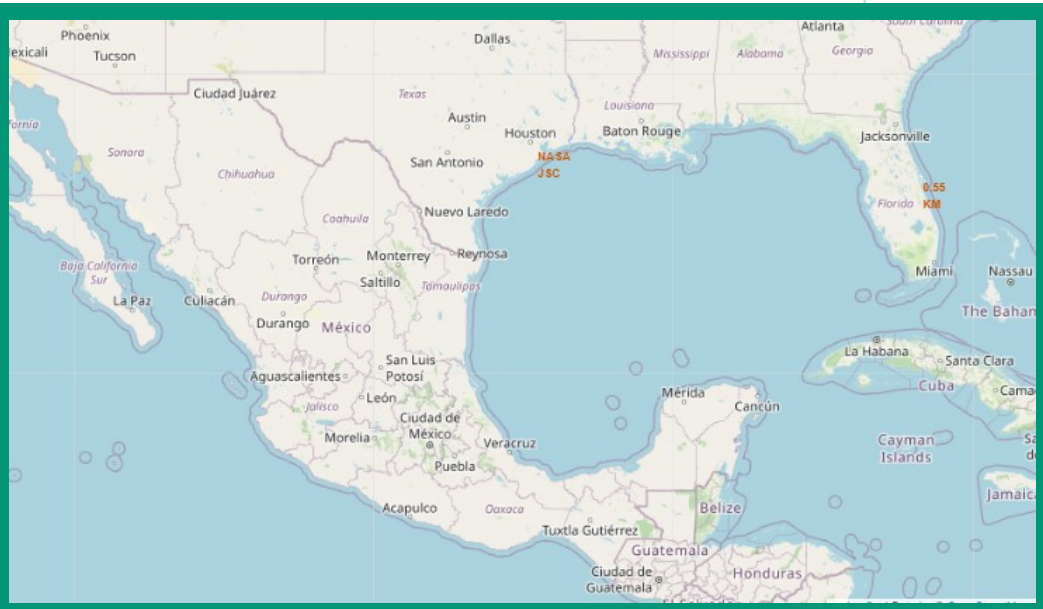
```
temp_df = df.copy()
temp_df['Year'] = year
temp_df.groupby('Year')['Class'].mean().plot()
```

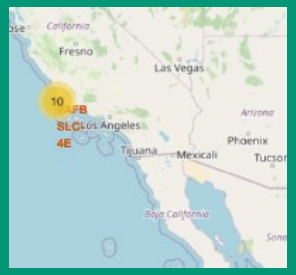
```
<Axes: xlabel='Year'>
```



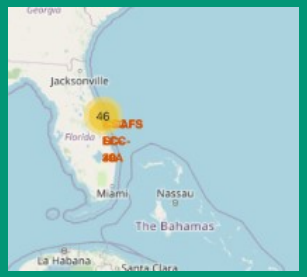
Visualize the launch success yearly trend

SpaceX Falcon9 Launch Sites Map



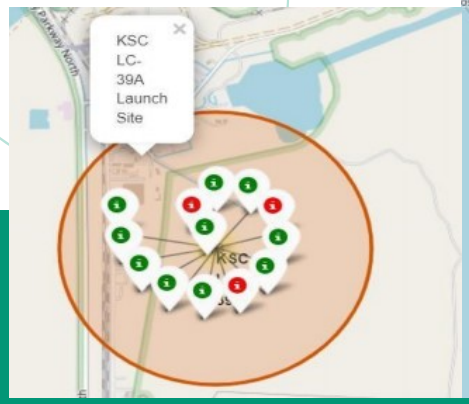


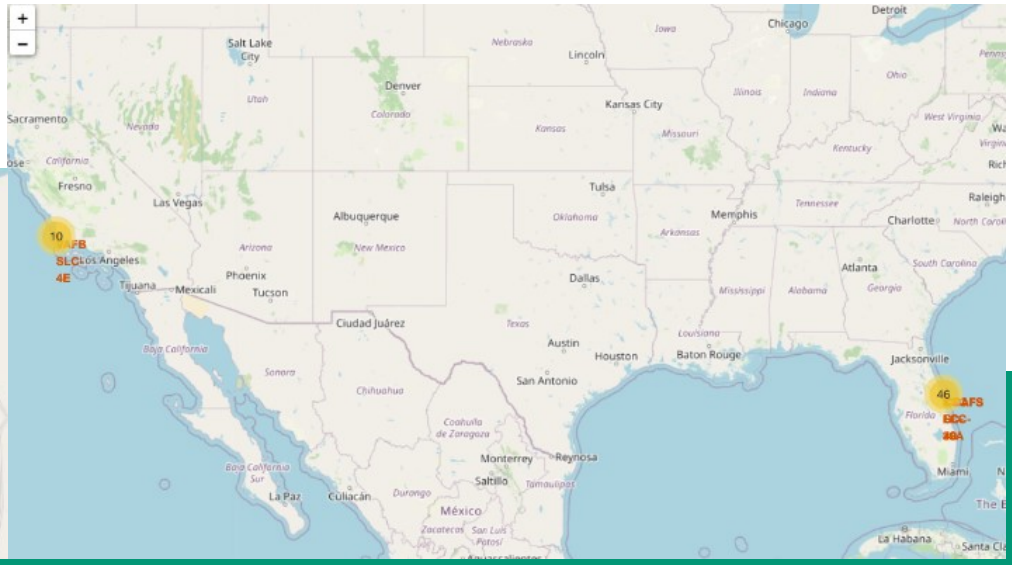
ZOOM 1

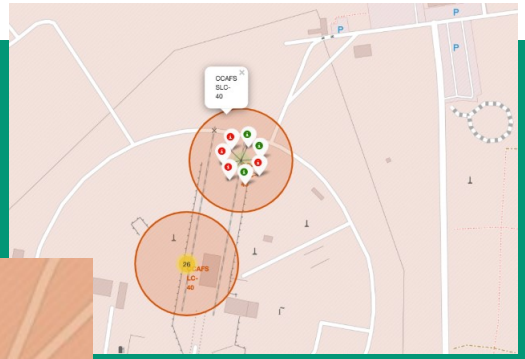


ZOOM 2

SpaceX Falon9 – Success/Failed Launch Map for all Launch Sites







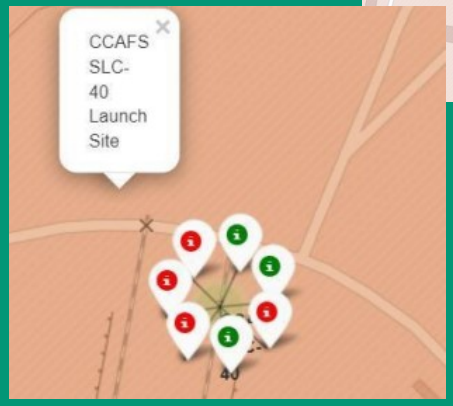
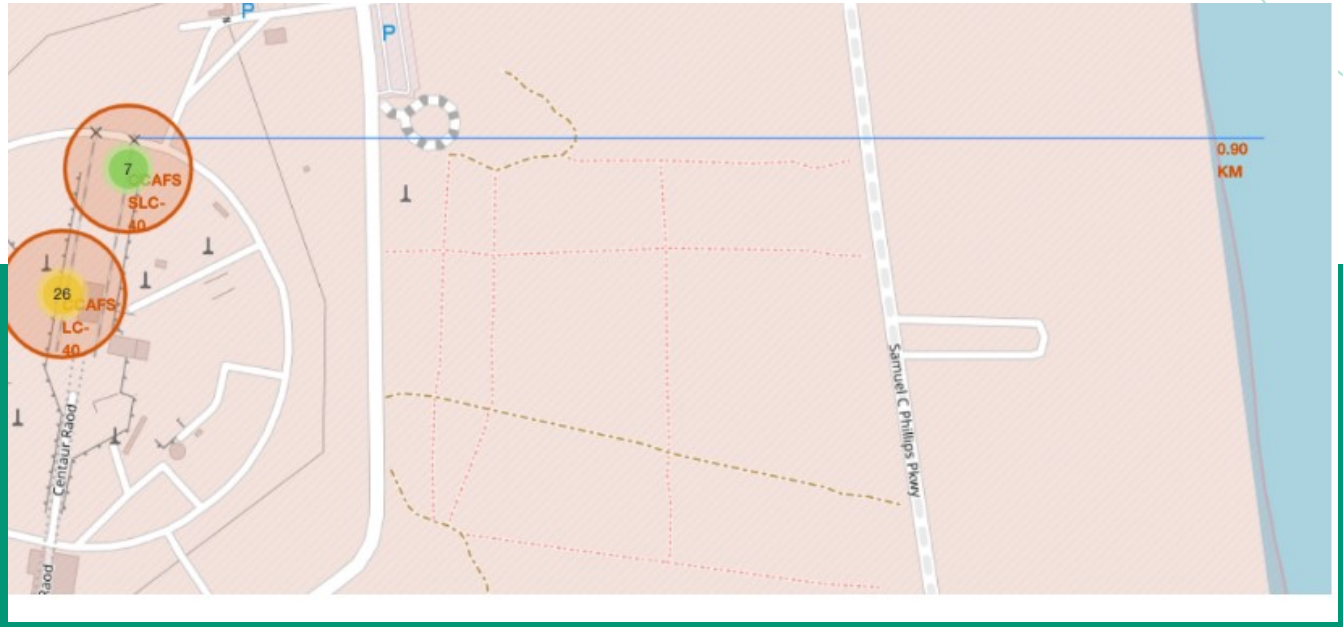


Figure 1 is the US map with all the Launch Sites. The numbers on each site depict the total number of successful and failed launches

• Figure 2, 3, 4, and 5 zoom in to each site and displays the success/fail markers with green as
success and red as failed

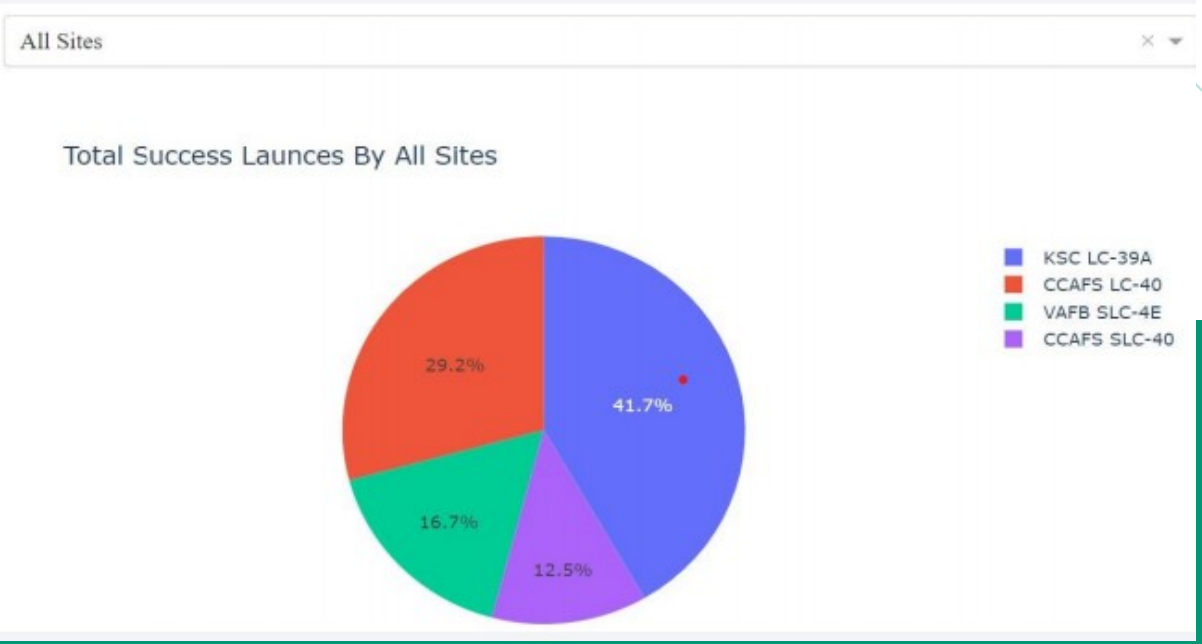
• By looking at each site map, KSC LC-39A Launch Site has the greatest number of successful launches

SpaceX Falcon9 – Launch Site to proximity Distance Map



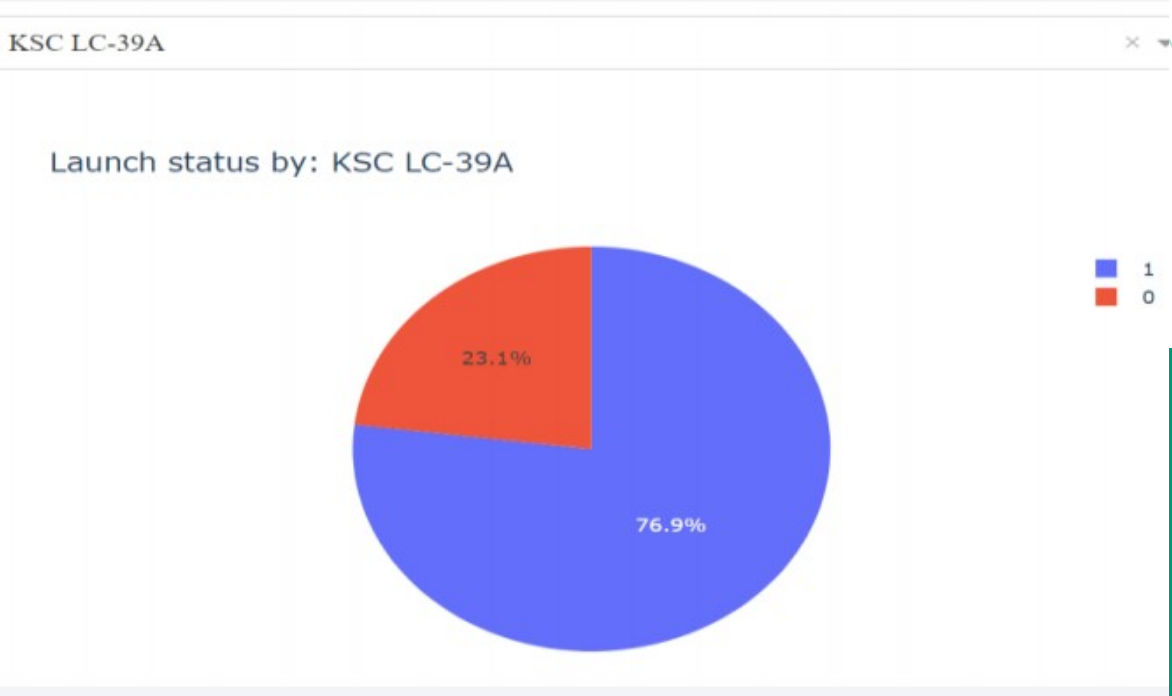
Calculating the distances between a launch site to its proximities gives this as a result

Launch Success Counts For All Sites



1. Launch Site ‘KSC LC-39A’ hasthe highest launch success rate
2. LaunchSite ‘CCAFSSLC- 40’ has the lowest launch success rate

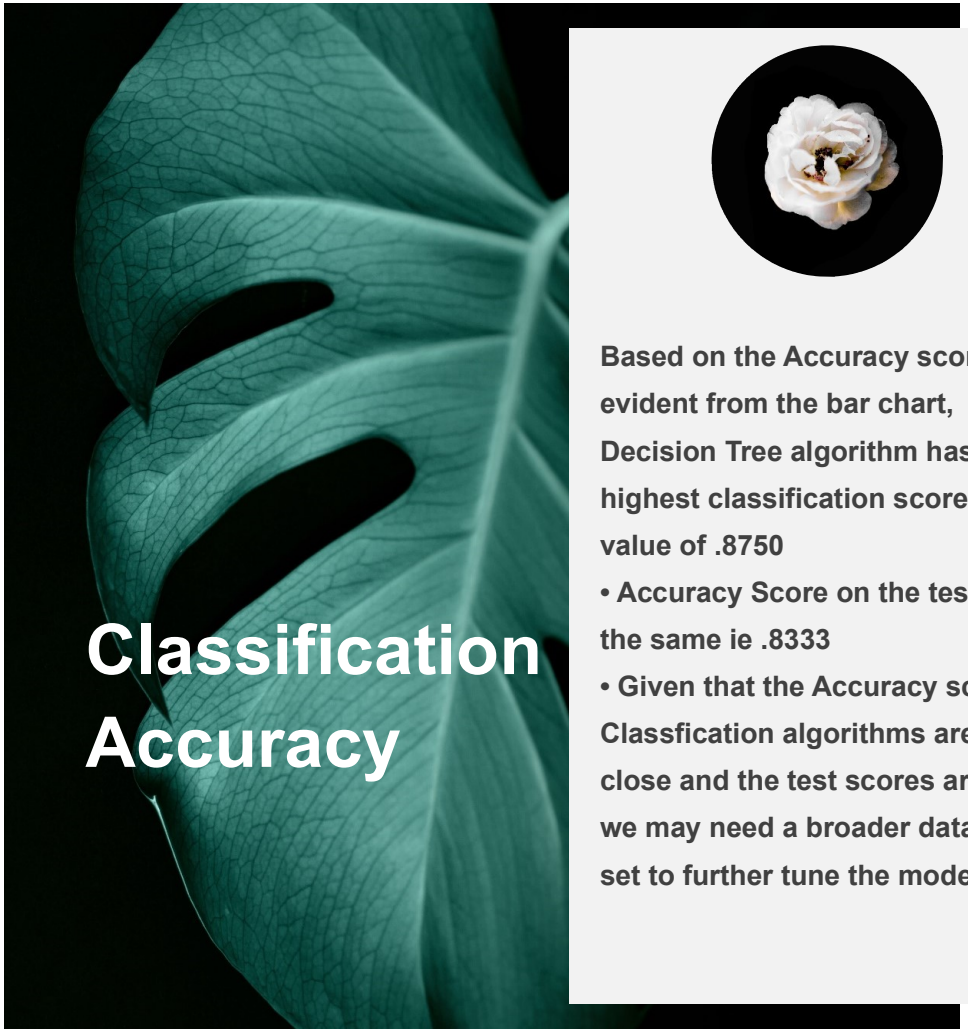
Launch Site with Highest Launch Success Ratio



1. KSC LC-39A Launch Site has the highest launch success rate and count
2. Launch success rate is 76.9%
3. Launch success failure rate is 23.1%

Payload vs. Launch Outcome Scatter Plot for All Sites

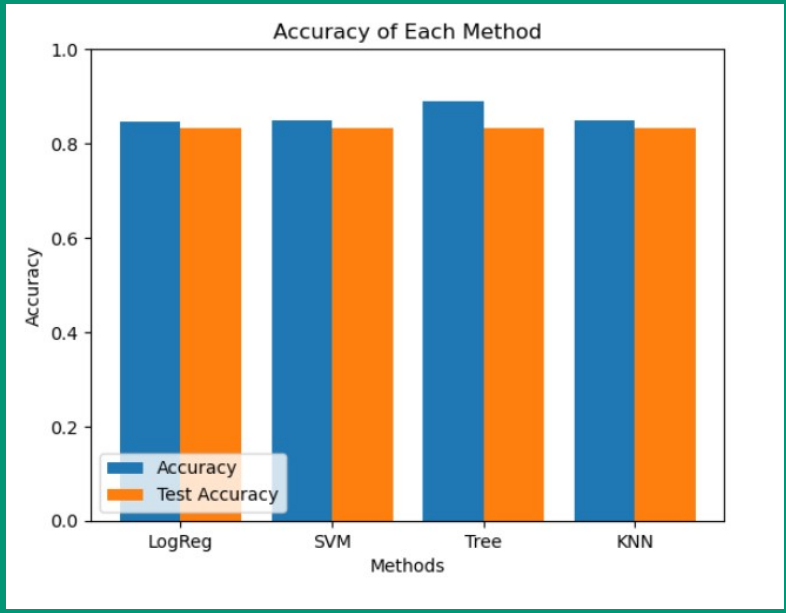
- Most successful launches are in the payload range from 2000 to about 5500
- Booster version category ‘FT’ has the most successful launches
- Only booster with a success launch when payload is greater than 6k is ‘B4’



Classification Accuracy

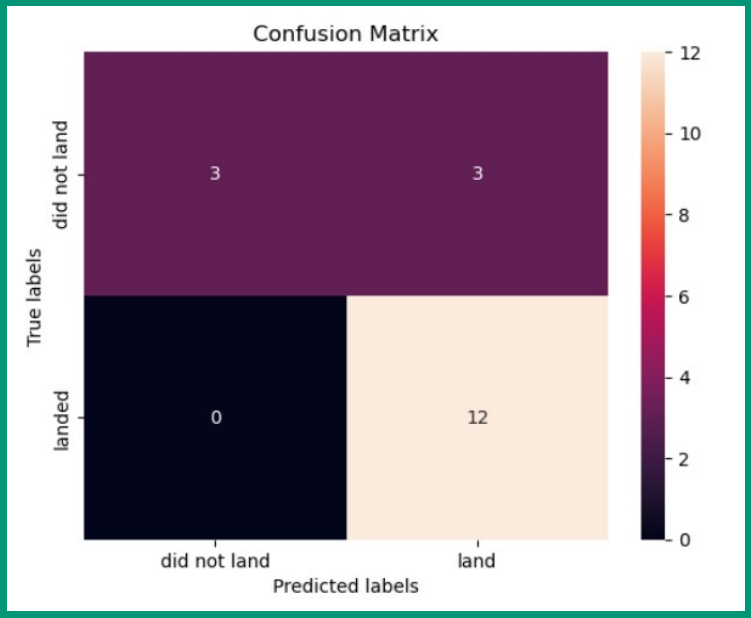
Based on the Accuracy scores it is evident from the bar chart, Decision Tree algorithm has the highest classification score with a value of .8750

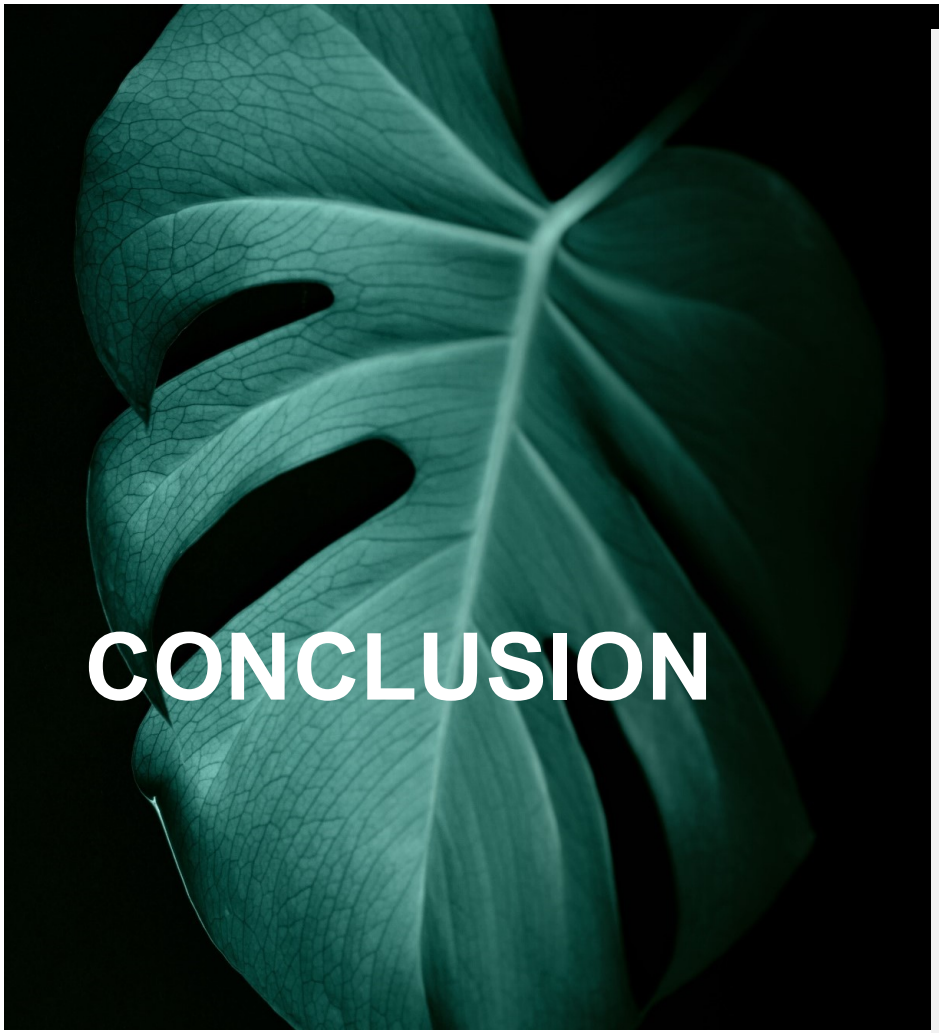
- Accuracy Score on the test data is the same ie .8333
- Given that the Accuracy scores for Classfication algorithms are very close and the test scores are the same, we may need a broader data set to further tune the models



Confusion Matrix

- The confusion matrix is same for all the models
- Per the confusion matrix, the classifier made 18 predictions
- 12 scenarios were predicted Yes for landing,and they did land successfully (True positive)
- 3 scenarios (top left) were predicted No for landing, and they did not land (True negative)
- 3 scenarios (top right) were predicted Yes for landing, but they did not land successfully (False positive)
- Overall, the classifier is correct about 83% of the time ((TP+ TN)/ Total) with a misclassification or error rate ((FP + FN) / Total) of about 16.5%





CONCLUSION

The aviation industry has seen a surge in flight numbers, and as a result, the initial stage of flight is more likely to land successfully. Interestingly, while success rates tend to increase with a higher payload, there isn't a clear correlation between payload mass and success rates.

However, one thing that's definitely improving is the launch success rate - it's increased by an impressive 80% from 2013 to 2020! In terms of specific launch sites, 'KSC LC-39A' has the highest launch success rate, while 'CCAFS SLC-40' has the lowest.

When it comes to orbits, ES-L1, GEO, HEO, and SSO have the highest success rates, with GTO being the lowest. To help ensure safety and efficiency, launch sites are located strategically away from cities and closer to coastlines, railroads, and highways.

Finally, the Decision Tree ML classification model is currently the top performer, with an accuracy of around 87.5%. However, there is still room for improvement, as when the models were tested on new data, the accuracy score dropped slightly to around 83%. It may be necessary to gather more data to further refine the models and achieve an even better fit.

THANK - YOU !!