IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Mohammed Aiman Khan
14-Jan-2024

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/tree/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone

Outline

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

Executive Summary

Summary of methodologies

1. **Gathered data** from the public using **SpaceX API** and from SpaceX Wikipedia page using **Data Scarpping**.
2. **Cleanse and Prepare Workable Data** Perform **Data wrangling** and Introduced a 'class' column to categorize successful landings.
3. **Visualize data by** SQL output, **visualization methods**, **Folium maps**, and **dashboards**.

1. Selected relevant data as **features for machine learning.**
2. Converted categorical variables into binary format using **one-hot encoding**.
3. **Standardized the dataset**
4. Applied GridSearchCV to identify optimal parameters for **machine learning models**.
5. Implemented four machine learning models**: Logistic Regression, Support Vector Machine, Decision Tree Classifier, and K Nearest Neighbors**.
6. **Visualize** accuracy scores for all models.

Summary of results

(Predication for SpaceY)

1. Obtained consistent results with an accuracy rate of between 83.33% to 89% across all models.
2. Identified a common tendency among models to over-predict successful landings.
3. Acknowledged the necessity for additional data to refine models and enhance accuracy.
4. Concluded that acquiring more data is crucial for improving model determination and achieving more accurate predictions.

Introduction

Project background and context

To predict if the Falcon 9 first stage will land successfully.

SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

Therefore if we can determine if the first stage will land, in turn we can determine the cost of a launch.

This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Problems you want to find answers

Space Y that would like to compete with SpaceX founded by Billionaire industrialist Allon Musk. Your job is to determine the price of each launch. We will do this by gathering information about Space X and creating dashboards for our team.

We will also determine if SpaceX will reuse the first stage.

Instead of using rocket science to determine if the first stage will land successfully, we will train a machine learning model and use public information to predict if SpaceX will reuse the first stage.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected from the SpaceX public API along with the SpaceX Wikipedia page using web-scraping.
- Data wrangling
  - Data of landings was labelled as successful or unsuccessful landing.
- Exploratory data analysis (EDA) using visualization and SQL
- Interactive visual analytics using Folium and Plotly Dash
- Predictive analysis using classification models
  - Built, tuned, evaluated classification models using GridSearchCV

Data Collection - Space X public API

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/blob/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone/Week%201%20Introduction/Data%20Collection/jupyter-labs-spacex-data-collection-api.ipynb

Data collection process to gather data this is done by using

- Get request to Space X public API
- Clean the Data

<u>Pre-Req for Get Request:</u>

import libraries ***requests; panda; numpy; datetime***

Define helper functions

Request for data from the url "***spacex_url="https://api.spacexdata.com/v4/launches/past***"

Validate by printing the response and also by checking the ***response.status_code***

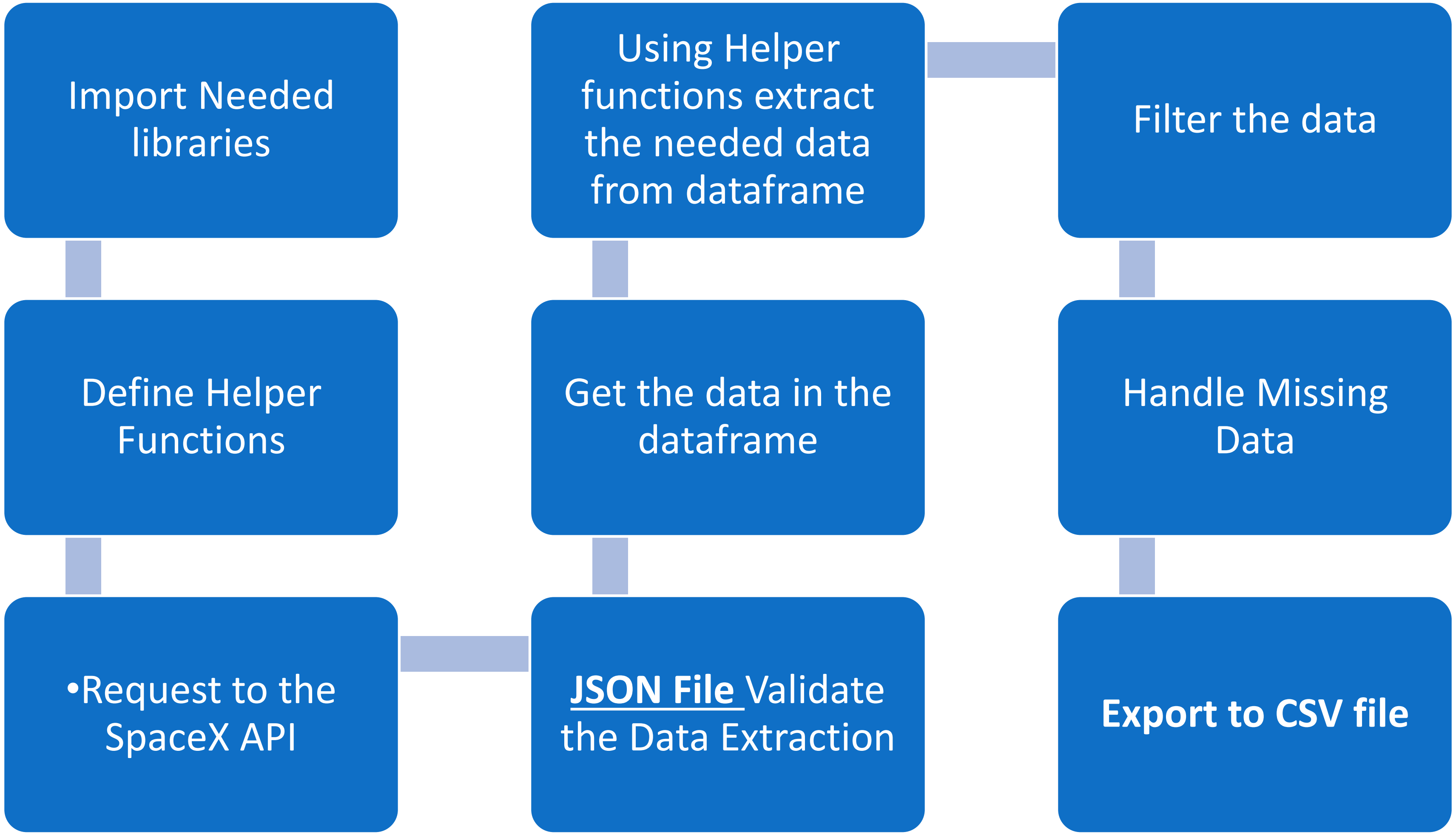
Turn the response data into panda dataframe named "***data***"

Now we see the data with lot of id's (like in rocket column we have some id)
Next step we use the API to get the data for the columns which were utilized in the helper functions to extract appropriate data
Ex: in payload column we wanted to see the mass of payload and the orbit

<u>Space X Data Columns collected are:</u>
FlightNumber;Date;BoosterVersion; PayloadMass; Orbit; LaunchSite; Outcome; Flights; GridFins;
Reused; Legs; LandingPad; Block; ReusedCount; Serial; Longitude; Latitude

Filter and handle missing data prior to importing into .CSV file

Data Collection - Space X public API Flow chart



Data Collection – WEB SCRAPING

Web scraping is done by accessing the site(the data is stored in the HTML table)
https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

- used “beautifulSoup” for extracting the data z
- Parse and convert to dataframe using “Panda”

Pre-Req for Scraping:

Import the needed libraries and build the needed help functions

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/blob/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone/Week%201%20Introduction/Data%20Collection/jupyter-labs-webscraping.ipynb

Perform the HTTP GET request to the Falcon9 Launch HTML page and as an HTTP response create a BeauitfulSoup object.

Find the tables in the BeautifulSoup object and assign to List.

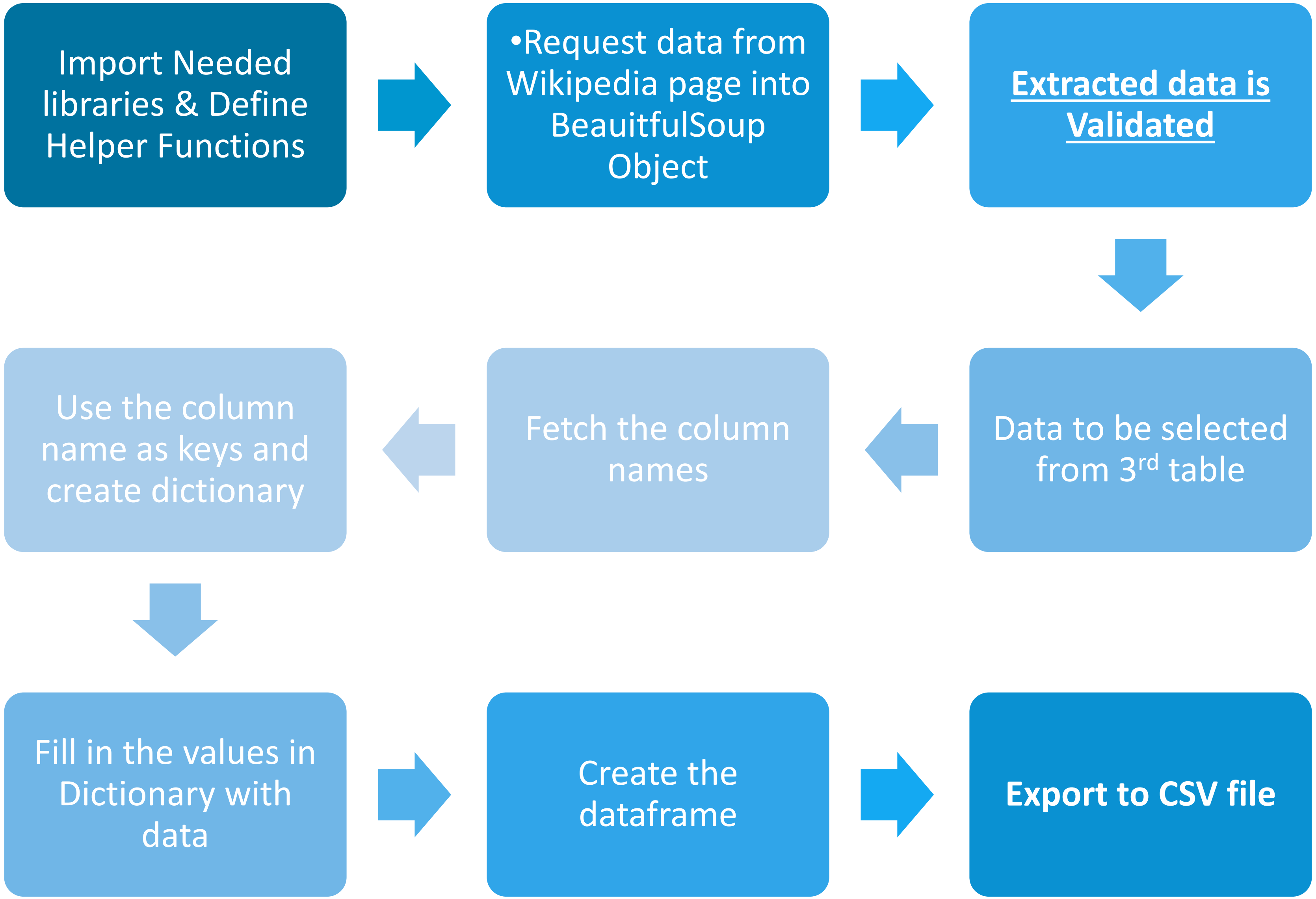
Our data is needed from the 3rd table. Check by printing

Get the Column names

Create Dictionary with keys=Column name and fill-in the values using the data.

Create a Dataframe and Export the data into .CSV

Data Collection – WEB SCRAPING Flow Chart



Data Wrangling

The process of transforming and organizing raw data into a more usable format.

Fetch the data of SpaceX extracted , Fill in missing data if any, and perform analysis for the below instances

Import Libraries → Load and Read SpaceX dataset → Fill in missing values → Have dataset ready for working

Task1: Calculate the number of launches performed from each ‘site’ – using method value_counts()

Task2: Calculate number of occurrence at each Orbit – using method value_counts()

Task3: Calculate the number of occurrence and its outcome at each Orbit – using method value_counts() and enumerates

Task 4: Creating a Custom Outcome label with the data values in the outcome column (0-failure and 1-successful launch)

Based on the above labels Calculate the success rate using the mean function on the custom outcome label.

Export the data into a .CSV for next steps……

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/blob/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone/Week%201%20Introduction/Date%20Wrangling/labs-jupyter-spacex-Data%20wrangling.ipynb

EDA - Visualization

Import the libraires
Panda, Numpy
Matplotlib ,seaborn

Load the SpaceX data from .CSV
into the database

Perform SQL queries to visualize data in tabular format.

Task No.	Activities Performed
1	Visualize between flight number and launch sites -- ***using scatter point chart***
2	Visualize between payload-mass and launch sites -- ***using scatter point chart***
3	Visualize success-rate for each orbit -- ***using bar chart***
4	Visualize flight number and orbit type -- ***using scatter point chart***
5	Visualize between payload and orbit type -- ***using scatter point chart***
6	Visualize launch success Trend -- ***using line chart***
7	Create dummy variables to categorical – NON chart activity (data categorizing task)
8	Create all Numeric columns of data set to float size 64 – NON chart activity (data conversion task)

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/blob/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone/Week%202%20EDA/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA - SQL

Install Database and needed extensions → Load the SpaceX data from .CSV into the database. Remove empty records → Perform SQL queries to visualize data in tabular format.

Task No.	Activities Performed
1	View the SpaceX Launch Site's -- *using UNIQUE keyword*
2	View only first 5 records of the SpaceX dataset where Launch Site's name begins with 'CCA' *using filters , limits*
3	Display total payload mass launched 'NASA (CRS)' *– using aggregate function SUM*
4	Display total payload mass by booster beginning with 'F9 v1.1'*– using aggregate function AVG and %wildcard*
5	Display the date when the 1st successful landing took place – *using min function*
6	Display booster successfully landed and having payload mass between 4000 and 6000 – *using LIKE and AND operators in where clause*
7	Display total number of successful and failure outcome – *using And operator where clause*
8	Display booster_version with maximum payload mass*– using subquery in where clause*
9	Display the month name and other data for failure drone ship in year 2015– *using LIKE in where clause*
10	Rank Landing outcomes in descending order– *using Group by and Order by clause*

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/blob/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone/Week%202%20EDA/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

Import and Install folium and panda

Use the dataset

Sapcex launch geo

For launch site location mapping

folium.Circle and Marker objects were used to identify launch sites, Transportation proximity using coordinates

For Successful and Failure Launch's for a specific site

Class is created to represent success by 1 and failure by 0

Maker color is set to green for class 1 and red for class 0

folium.Circle and Marker objects were used to identify success and failure launch's at the launching site

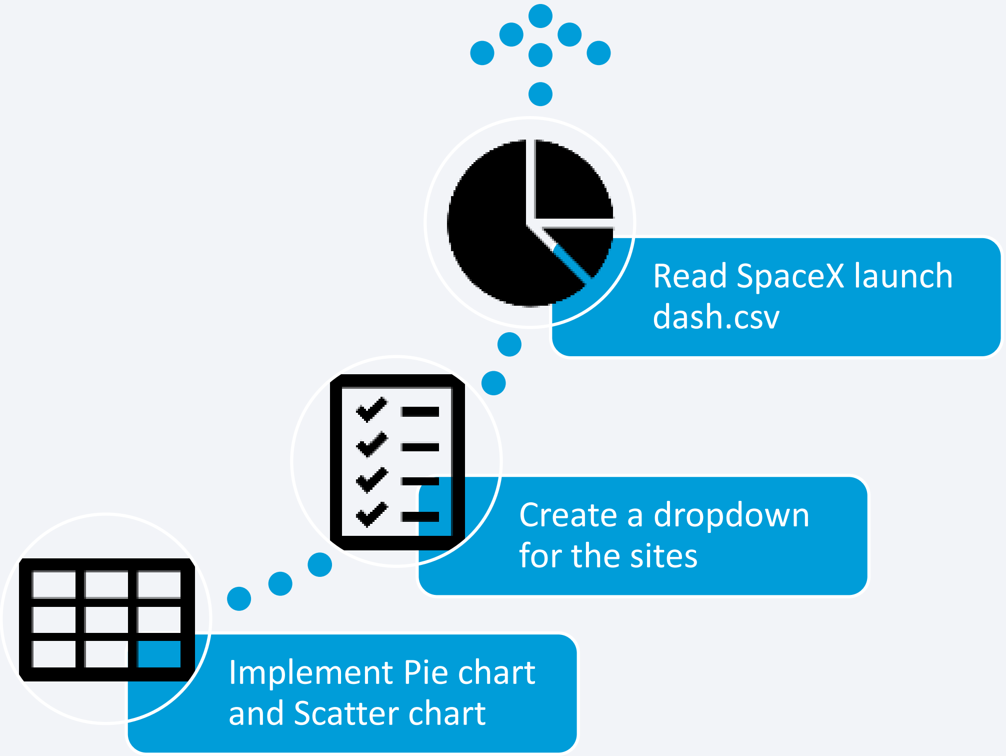
For proximity of Transportation

Use the mouse position to locate the proximity of transportation like railway lines, highways , coasts and cities

Place the line marker from the launch site to each of the proximities and get their appropriate distance

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/blob/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone/Week%203%20Interactive%20Visual%20Analytics%20and%20Dashboard/lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash



Pie Chart is created for Percentage share of Successful launching at various launch sites

One of the site is picked from drop down to see Success rate against the failure rate for all its Launch’s

Scatter Chart is generated to display success Across the various payload masses

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/tree/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone/Week%203%20Interactive%20Visual%20Analytics%20and%20Dashboard/Dashboards

Predictive Analysis (Classification)

1. Install seaborn, panda
2. Create a function for confusion matrix
3. Load the data set
4. Create a array from column class
5. Standardize the data
6. Split X and Y data from training and testing
7. Create a logistic regression object
   Create a GridSearchCV a and numpy
8. Use the above and generate confusion matrix for each model
9. Plot the bar chart based on the data of each model

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/blob/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone/Week%204%20Predictive%20Analysis/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

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

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

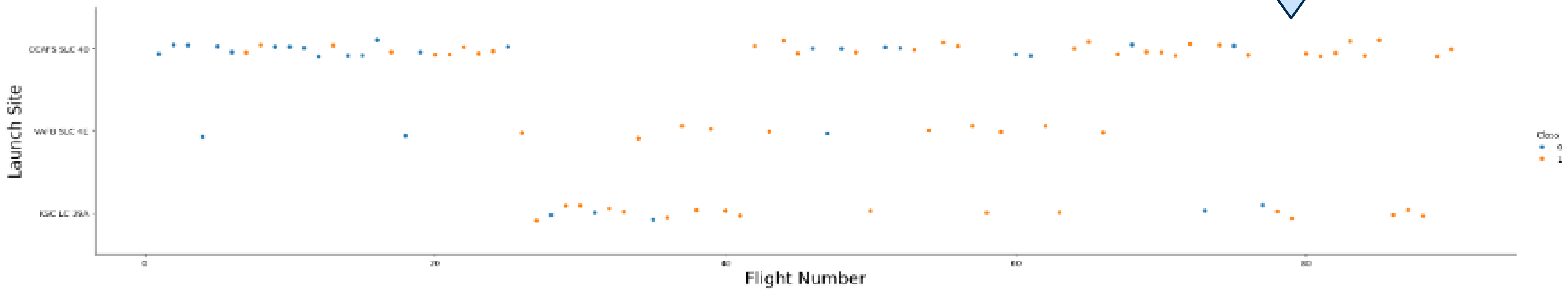
Orange is Success Blue the Failures

Success Rate increases after flight 20

CCSA has major launch’s hence we assume it as primary launching site

In [6]:

```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the L
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



In here… Scatter plot is performed between flight number and launch site and setting the x and y coordinates to the plot along with appropriate labels

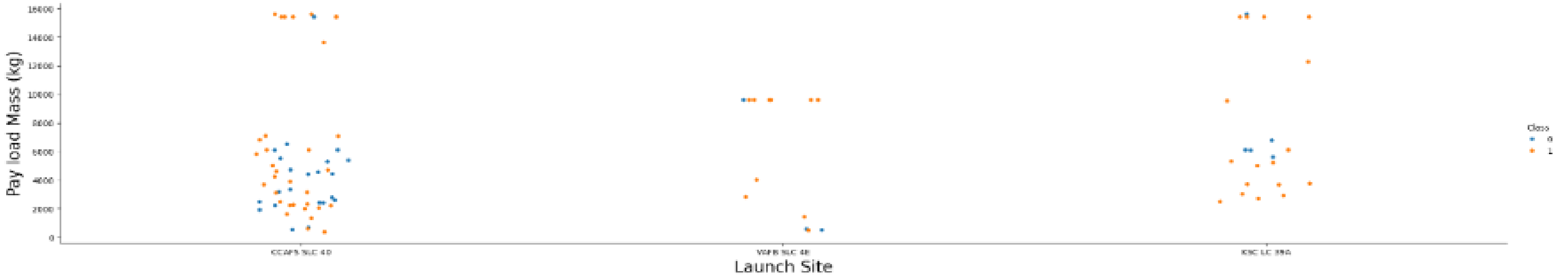
Payload vs. Launch Site

Orange is Success Blue the Failures

Payload mass majorly at launch site CCAPS
Less payload mass at WFB

CCPAS site is using payload mass from 0 to 6000

```
# Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis
sns.catplot(y="PayloadMass", x="LaunchSite", hue="Class", data=df, aspect = 5
plt.xlabel("Launch Site",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



In here… Scatter plot is performed between payload and launch site and setting the x and y coordinates to the plot along with appropriate labels

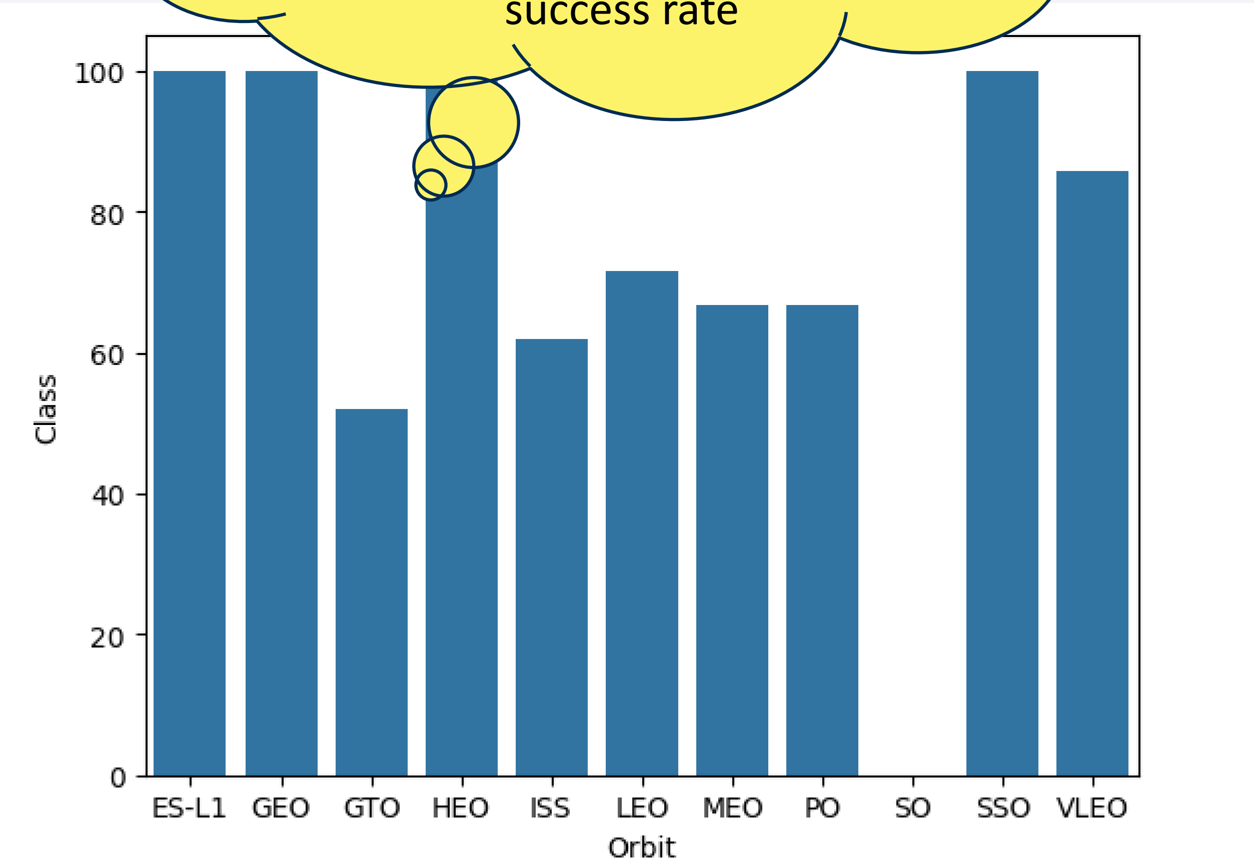
Success Rate vs. Orbit Type

100% success rate at ES-L1,GEO,HEO,SSO

SO has the least success rate

VLEO,LEO,ISS,MEO,PO have 60% success rate

```python
# HINT use groupby method on Orbit column and get the mean of Class column
temp = df.groupby(["Orbit"]).mean().reset_index()
temp2 = temp[["Orbit", "Class"]]
temp2["Class"] = temp2["Class"]*100
sns.barplot(x = "Orbit", y = "Class", data = temp2)
```



In here… BAR chart is plotted for each orbit type on x-axis and success rate count on y-axis

setting the x and y coordinates to the plot along with appropriate labels

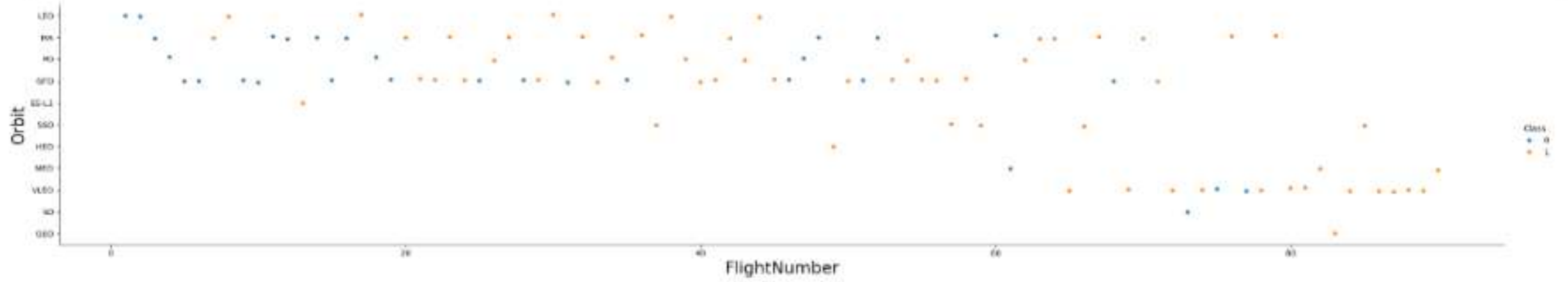
Flight Number vs. Orbit Type

Orange is Success Blue the Failures

Initial launch’s had failure rate but later we see the increase in success rate

VLEO has good success rate

```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



In here… Scatter plot is performed between flight numbers and orbit type setting the x and y coordinates to the plot along with appropriate labels

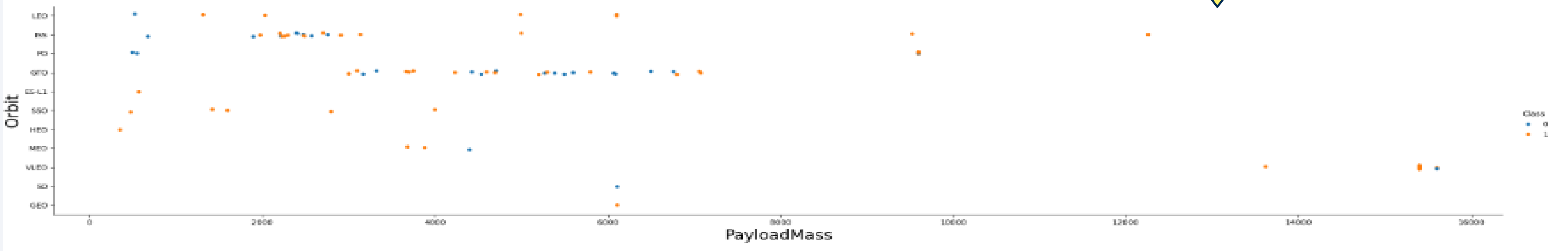
Payload vs. Orbit Type

LEO, SO and GEO has less success rates

Payloads more at range below 6000kg

VLEO is other successss rate orbit

```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orb
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



In here… Scatter plot is performed between payload and orbit type setting the x and y coordinates to the plot along with appropriate labels

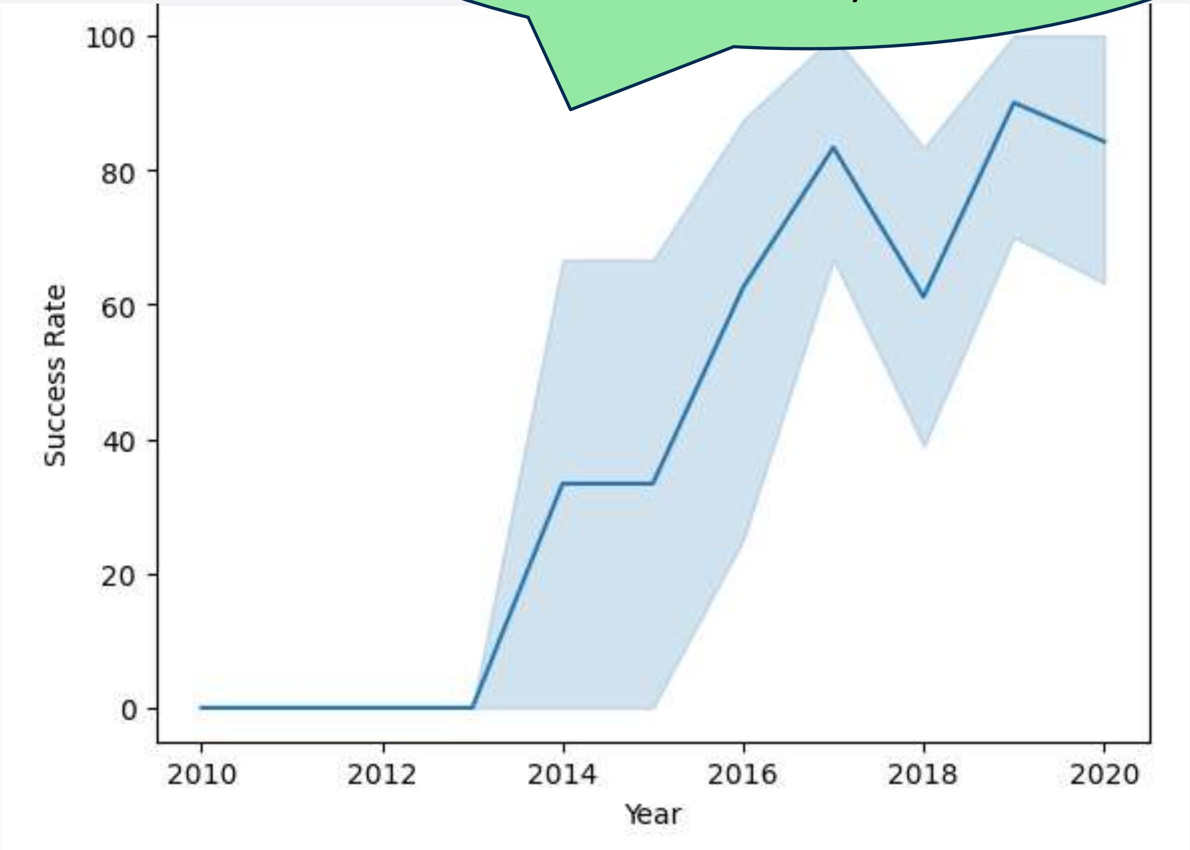
Launch Success Yearly Trend

95% confidence level is shaded around
80% is the success rate
Success rate trend steep from 2013 to2017
2017 to 2020 success trend increases slowly

```
# A function to Extract years from the date
year=[]
def Extract_year():
    for i in df["Date"]:
        year.append(i.split("-")[0])
    return year
Extract_year()
df['Date'] = year
df.head()
```

```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
df["Date"] = pd.to_datetime(df["Date"])
df["Year"] = df["Date"].dt.year
df["Success Rate"] = df["Class"] * 100

sns.lineplot(data=df, x="Year", y="Success Rate")
```



All Launch Site Names

TASK: **Find the names of the unique launch sites**

QUERY: SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL ORDER BY 1;

Distinct clause fetch's unique launch_site, If Distinct was not used then duplicate launch_site sill be displayed

OUTPUT

Out[8]:

Launch_Site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Launch Site Names Begin with 'CCA'

TASK: Find 5 records where launch sites begin with `CCA`

QUERY: select * from SPACEXTBL where launch_site like 'CCA%' limit 5;

In Where clause we use like operator to filter records whose launch_site begins with CCA , And we use limit option to restrict to 5 records

OUTPUT

Out[11]:

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	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

Total Payload Mass

TASK: **Calculate the total payload carried by boosters from NASA(CRS)**

QUERY: select sum(payload_mass__kg_) as sum from SPACEXTBL where customer like 'NASA (CRS)’;

In Where clause we use like operator to filter records having customers like NASA(CRS)
We also use SUM aggregate function to get total payload mass .

OUTPUT

Out[12]:

sum
45596

Average Payload Mass by F9 v1.1

TASK: **Calculate the average payload mass carried by booster version F9 v1.1**

QUERY: select avg(payload_mass__kg_) as Average from SPACEXTBL where booster_version like 'F9 v1.1%' ;

In Where clause we use like operator to filter records having booster_version starting with F9 v1.1
We also use AVG aggregate function to average total payload mass .

OUTPUT

Out[13]:

Average
2534.6666666666665

First Successful Ground Landing Date

TASK: **Find the dates of the first successful landing outcome on ground pad**

QUERY: select min(date) as Date from SPACEXTBL where mission_outcome like 'Success' ;

In Where clause we use like operator to filter records whose outcome is Success
We also use MIN function to get the minimum date of landing

OUTPUT

Out[14]:

Date
2010-06-04

Successful Drone Ship Landing with Payload between 4000 and 6000

TASK: **List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000**

QUERY: SELECT Booster_Version FROM SPACEXTBL WHERE Mission_Outcome LIKE 'Success' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;

In Where clause we use like operator to filter records whose outcome is Success AND relation operator to also have additional filter condition for fetching records whose payload mass weighs between 4000 and 6000kgs

OUTPUT

Out[28]:

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

Total Number of Successful and Failure Mission Outcomes

TASK: Calculate the total number of successful and failure mission outcomes

QUERY: SELECT mission_outcome, count(*) as Count FROM SPACEXTBL GROUP by mission_outcome ORDER BY mission_outcome;

We use the aggregate function count(*) along with GROUP clause for fetching total records for each mission_outcome
ORDER clause is also used to fetch the data alphabetic sorted by mission_outcome values

OUTPUT

Out[29]:

Mission_Outcome	Count
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

TASK: List the names of the booster which have carried the maximum payload mass

CODE:

```
maxm = %sql SELECT MAX(payload_mass__kg_) FROM SPACEXTBL
maxv = maxm[0][0]
%sql SELECT booster_version FROM SPACEXTBL
WHERE payload_mass__kg_ = :maxv;
```

Select the maximum payload mass using MAX aggregate function and store in array maxm

Fetch the 1st value from maxm and store in variable maxv

Fetch the data where payload mass is equal to maxv value

OUTPUT

Out[31]:

Booster_Version
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

2015 Launch Records

CODE:

```
SELECT
    CASE
        WHEN strftime('%m', Date) = '01' THEN 'January'
        WHEN strftime('%m', Date) = '02' THEN 'February'
        WHEN strftime('%m', Date) = '03' THEN 'March'
        WHEN strftime('%m', Date) = '04' THEN 'April'
        WHEN strftime('%m', Date) = '05' THEN 'May'
        WHEN strftime('%m', Date) = '06' THEN 'June'
        WHEN strftime('%m', Date) = '07' THEN 'July'
        WHEN strftime('%m', Date) = '08' THEN 'August'
        WHEN strftime('%m', Date) = '09' THEN 'September'
        WHEN strftime('%m', Date) = '10' THEN 'October'
        WHEN strftime('%m', Date) = '11' THEN 'November'
        WHEN strftime('%m', Date) = '12' THEN 'December'
    END AS Month,
    landing_outcome,
    booster_version,
    launch_site
FROM SPACEXTBL
WHERE substr(Date, 0, 5) = '2015'
    AND landing_outcome LIKE 'Failure (drone ship)';
```

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

Use Switch Case to convert the month number to month name

Then Fetch the records where year is 2015 and also having the outcome as Failure (drone ship)

OUTPUT

Month	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

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

TASK: 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

QUERY: SELECT landing_outcome, COUNT(*) AS count FROM SPACEXTBL WHERE "Date" BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY landing_outcome ORDER BY date DESC;

We use the aggregate function count(*) along with GROUP clause for fetching total records for each landing outcome between the dates ORDER clause by date in descending order

OUTPUT

Landing_Outcome	count
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

Section 3

Launch Sites Proximities Analysis

Launch Site Location Map

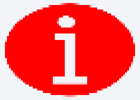
<u>**Launch site location as we see they satisfy the needed criteria**</u>

- **built as far as possible away from major population centers for safety and securi**
- **Nearing the equator to launch in equatorial orbit**
- **Near to transportation infrastructure for easy of resource movement.**
- **By the Ocean for clear sky and weather**

Color Coded Launch’s Markers for VAFB SLC 4E site

VAFB SLC-4E SITE View

Successful Landings ➔ 4



Failure Landings ➔ 6

Proximities to Transportation from KSC LC-39A – Rail ways

- **Nearest line is less than 1km**
- **Multiple Railway lines in various direction**
- **We observe at least 4 lines availability within proximity**

Proximities to Transportation from KSC LC-39A - Waterways

- Nearest coast in 7km range
- Multiple access in various direction
- We observe at least 2 waterways within proximity

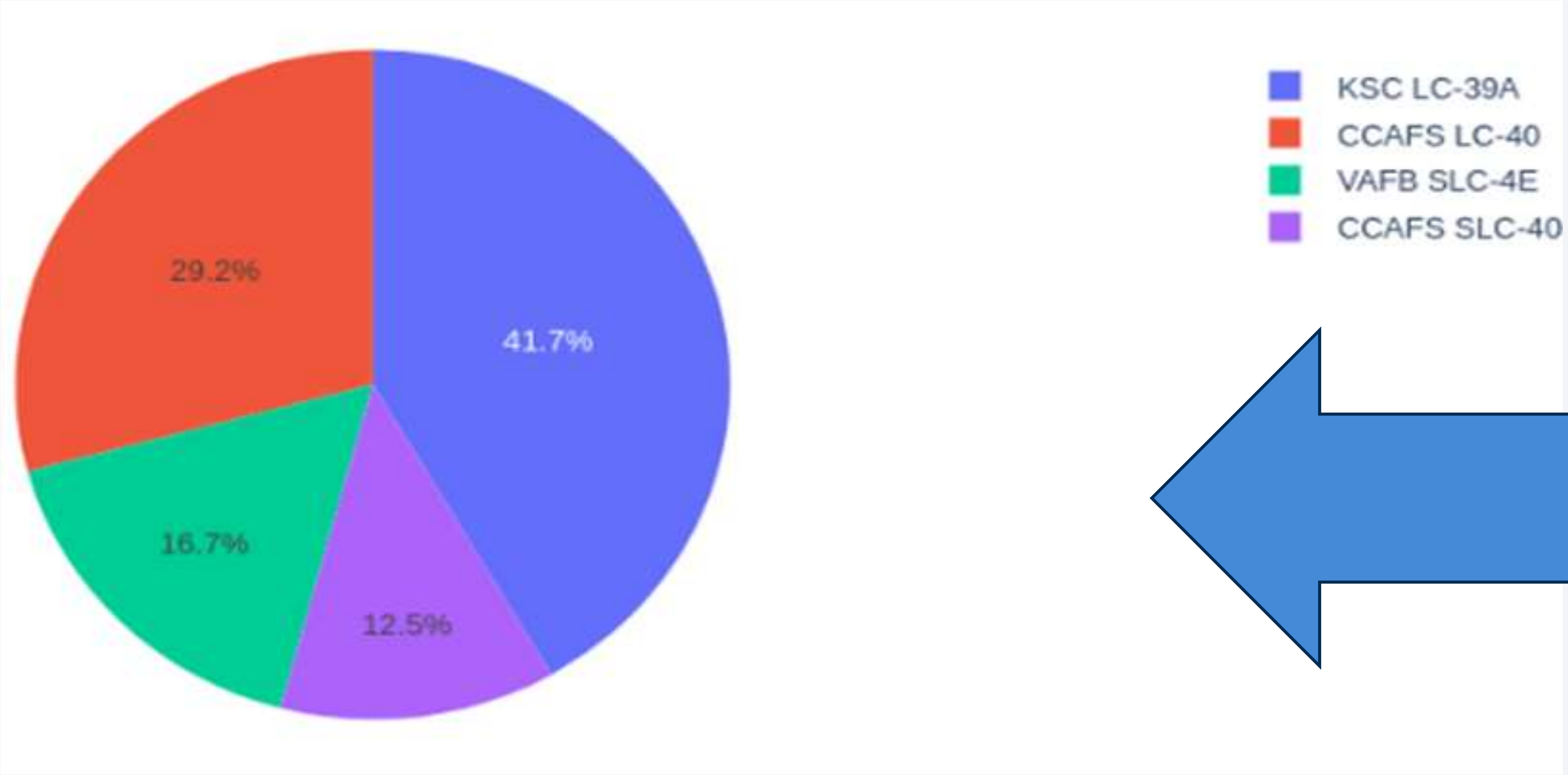
Proximities to Transportation from KSC LC-39A - Roadways

- **Highway FL405 in proximity 5.5km**
- **Nearest City around 15km**

Section 4

Build a Dashboard with Plotly Dash

SpaceX Success Launch Dashboard – Location wise



Major success rate% is seen at KSC LC-39A

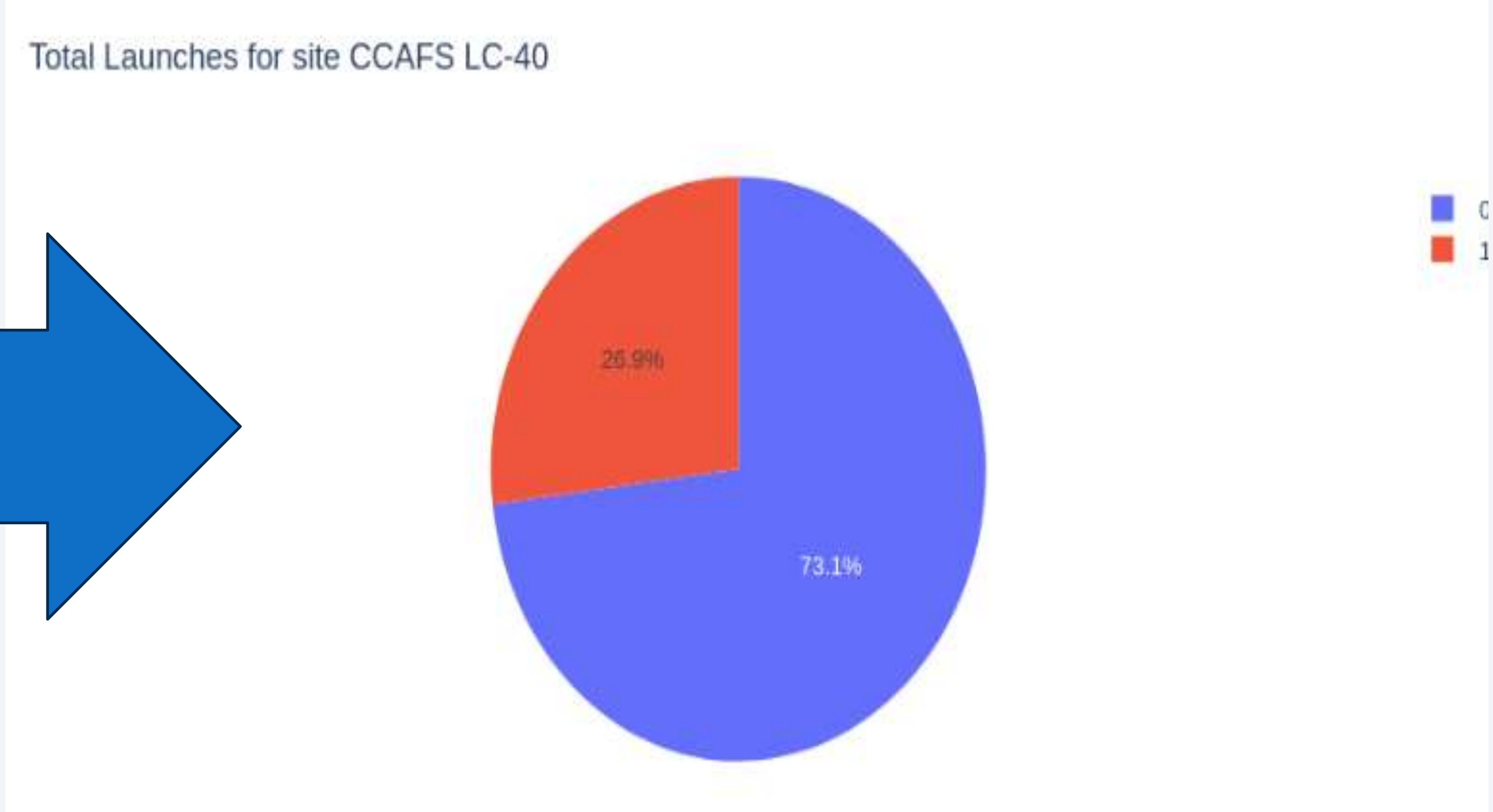
Least at CCAFS SLC-40 decision can be taken to continue or shift the load to a new location or existing launch site.

Location site CCAFS can be improved to get higher success rate

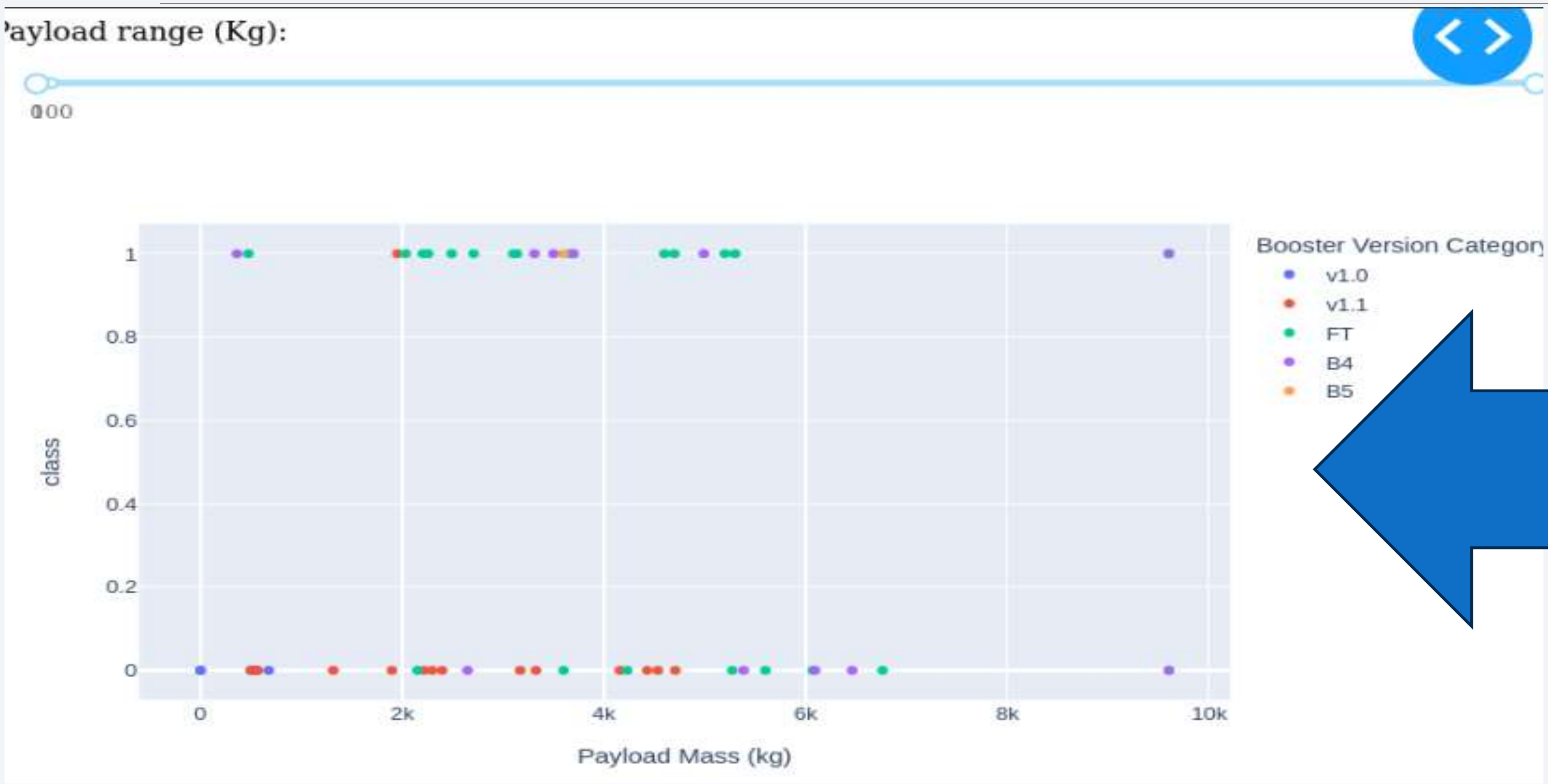
SpaceX Highest Success Rate launch site Dashboard (Success and Failure breakup)

73% Success rate site CCAFS LC-40

With 26.9% failure rate



SpaceX Highest Success Rate launch site Dashboard (Success and Failure breakup)



At Booster version v1.0 we see a failed zero payload mass

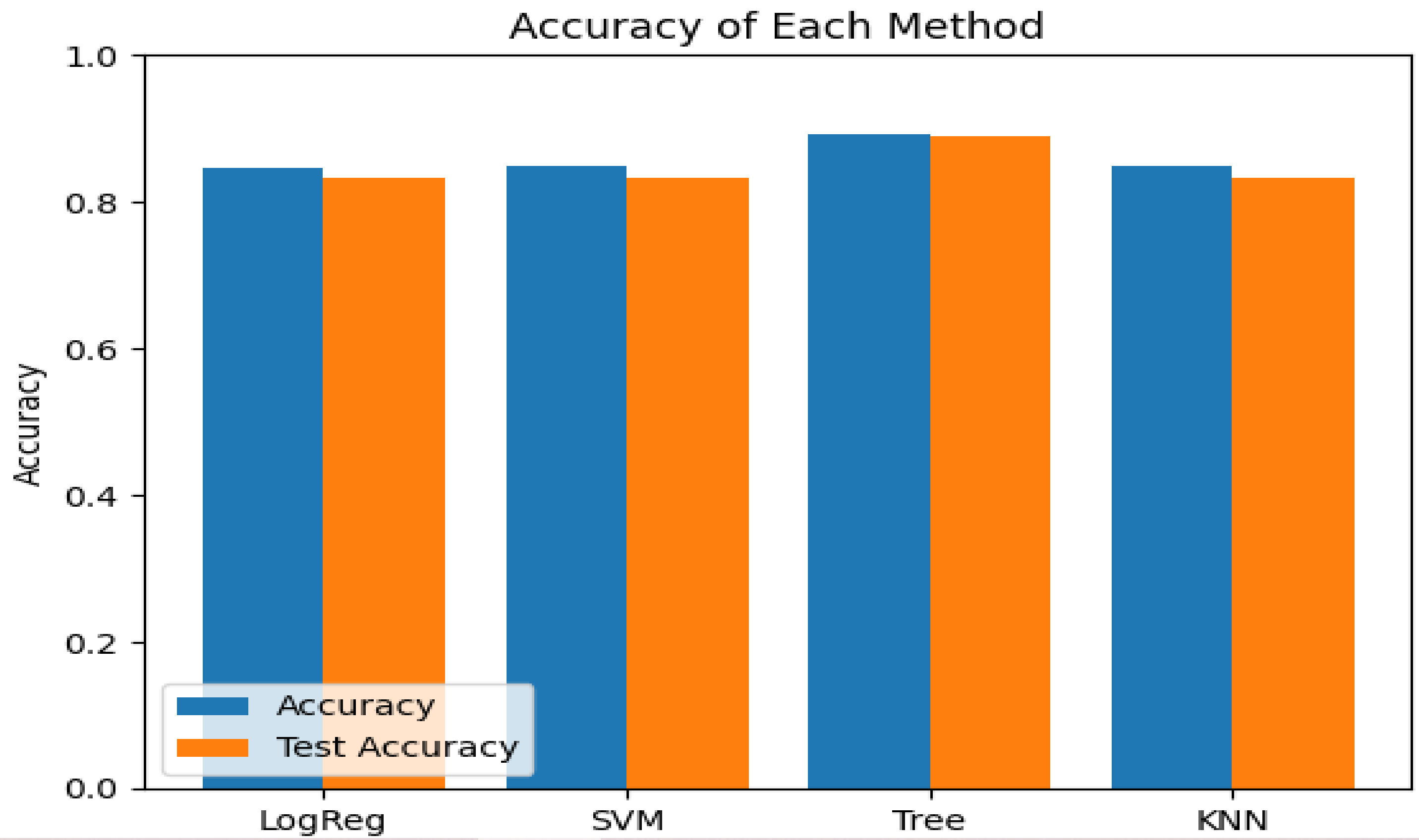
Lot of success rate is visible between payload mass of 2k and 4k
Do not see even one success between payload range 6k to 8k

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Model	Accuracy	TestAccuracy
LogReg	0.84643	0.83333
SVM	0.84821	0.83333
Tree	0.89107	0.88889
KNN	0.84821	0.83333



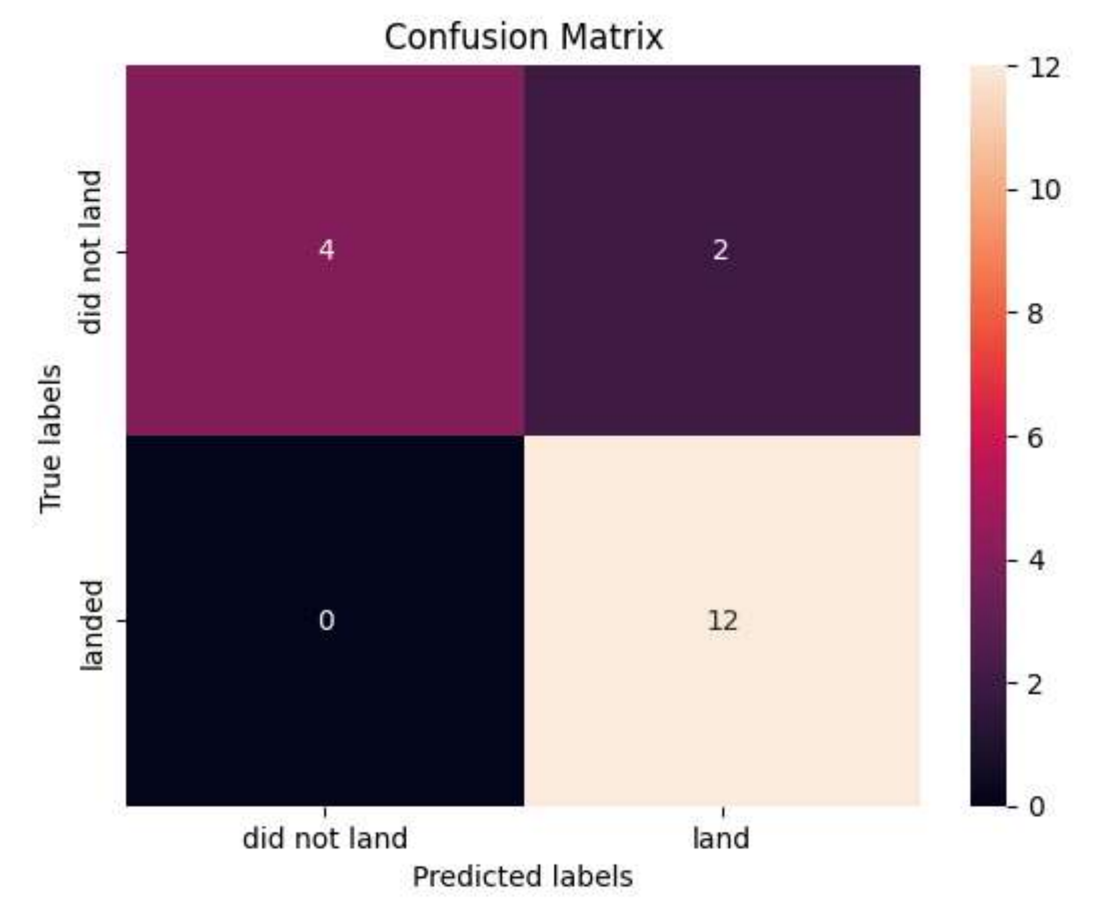
89% → Best Accuracy is seen in the model Tree

Other Model are in the range of 83% to 84%

Confusion Matrix

For model Tree

- ✓ **predicted 12 successful landings when true labels was successful**
- ✓ **predicted 2 successful landings when true labels was unsuccessful**
- ✓ **predicted 0 unsuccessful landings when true labels was successful**
- ✓ **predicted 4 unsuccessful landings when true labels was unsuccessful**



Conclusions

Point 1: A fair Machine Learning Model was created for Space Y to compete with Space X

Point 2: Various models were executed to get a fair idea of predication

Point 3: Models showed accuracy varying from 83.3% to 89%

Point 4: Visualization of SpaceX data has provided very good insight of data related to launch sites, payloads, proximity of transportation, critical success factor and risks of failures.

Point 5: More accurate predication could have been made with more data and further development of other models as well.

Point 6:   Space Y can use this model to predict with relatively high accuracy of launch.

Appendix

Coursera material

https://github.com/aimanmrkhan/IBM_Data_Science_Professional_Certification/tree/b020d44a4fe2e12115331c8fa17eb098022fd26e/10.Applied_Data_Science_Capstone

Thank you!