IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Name: Alok Ranjan
Date: 8 Nov 2023

Outline

Executive Summary

- Methodologies:-
  - Data Collection through API & Web Scraping
  - Data Wrangling
  - EDA with SQL, pandas & Matplotlib
  - Interactive visual Analytics with Folium Lab
  - Interactive Dashboard with Plotly Dash
  - Machine Learning Landing Prediction
- Summary of all results:-
  - Launch success rates improve with more launches.
  - Certain orbit types (ES-L1, GEO, HEO, SSO, TU) have a 100% success rate.
  - LEO and VLEO success correlates with the number of flights.
  - Launch sites are strategically located away from cities.
  - KSC LC-39A is the most successful launch site.
  - Falcon 9 booster has a slightly higher success rate.
  - Support Vector Machine (SVM) is the best ML model for the dataset.

Introduction

- **Project background and context**

SpaceX is most successful space company. One reason is inexpensive rocket launches, by reusing the first stage of their rockets.

Unlike other rocket providers, SpaceX's Falcon can recover & reuse the first stage.

By predicting whether the first stage will land successfully, we can determine the cost of a launch.

Another company, Space Y wants to compete with SpaceX in the space industry.

- **Problems we want to find answers**

1) Determine the price of each launch by collecting information about SpaceX and creating dashboards for our team.
2) Predict whether SpaceX will reuse the first stage.

To achieve this, Instead of using rocket, we plan to train a machine learning model and use publicly available information to predict if SpaceX reuse will the first stage.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

  ~ SpaceX Rest API ~ Web Scraping from Wikipedia

- Perform data wrangling

  ~ Converting landing outcome to classes (either 0 or 1)

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

  ~ Utilize multiple classification models, optimize hyperparameters and evaluate model performance on test data

Data Collection

- Data collection involved using both SpaceX Rest API and web scraping from SpaceX Wikipedia page.
- The SpaceX API provided various data columns related to launches, including flight number, date, payload details, outcomes, and more.
- Web scraping was performed using BeautifulSoup to extract Falcon 9 launch records from Wikipedia.
- The goal of data collection was to gather information for predicting SpaceX's first stage successful landing.

Data Collection – SpaceX API

Notebook link

1. Requesting response from SpaceX API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

2. Convert json to pandas dataframe

```
data=pd.json_normalize(response.json())
```

3. Custom functions uses API to extract additional information

```
getBoosterVersion(data) getLaunchSite(data)
getPayloadData(data) getCoreData(data)
```

4. Construct dataset using the data we have obtained

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

5. Filter the data & export it to CSV file

```
data_falcon9=df[df['BoosterVersion']=='Falcon 9']
data_falcon9.to_csv('dataset_part_1.csv',index=False)
```

Data Collection - Scraping

GitHub lab
Notebook link

1. Create soup object of url response

```
static_url = '''https://en.wikipedia.org/w/index.php?title=
List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922'''
response=requests.get(static_url)
soup=BeautifulSoup(response.content)
```

2. Extract column names using custom function

```
first_launch_table = soup.find_all('table')[2]
column_names = []
for x in first_launch_table.find_all('th'):
    name=extract_column_from_header(x)
    if (name !=None) and (len(name)>0):
        column_names.append(name)
```

3. Create an empty dict with keys from the extracted column names

```
launch_dict= dict.fromkeys(column_names)
del launch_dict['Date and time ( )']
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']=[]
```

4. Fill up the launch_dict with launch records extracted from table rows

Refer notebook

5. Create a dataframe from it & export it to a CSV

```
df= pd.DataFrame({ key:pd.Series(value)
                   for key,value in launch_dict.items()})
df.to_csv('spacex_web_scraped.csv',index=False)
```

Data Wrangling

Notebook link

- Perform some EDA to find some patterns in the data & determine what would be the label for training supervised models.

- We mainly convert landing outcomes into Training Labels :-
  - ~ 1 Booster successfully landed
  - ~ 0 Unsuccessful

Removing unnecessary features

```
data = data[['rocket', 'payloads', 'launchpad',
             'cores', 'flight_number', 'date_utc']]
```

Remove rows with multiple payloads and cores

```
data=data[data.payloads.map(len)==1]
data=data[data.cores.map(len)==1]
```

Extract payloads and cores value from list

```
data.payloads=data.payloads.map(lambda x:x[0])
data.cores=data.cores.map(lambda x:x[0])
```

Convert date_utc to datetime datatype and extracting only date

```
data['date']=pd.to_datetime(data.date_utc).dt.date
```

Restrict date of launch

```
data=data[data['date']<=datetime.date(2020,12,31)]
```

Replace missing PayloadMass with mean value

```
pl_mass_mean=data_falcon9.PayloadMass.mean()
data_falcon9.PayloadMass.replace(np.nan,
                                 pl_mass_mean,inplace=True)
```

```
Calculate the no. of launches on each site
Calculate the no. and occurrence of each orbit
Calculate the no. and occurence of mission outcome of the orbits
```

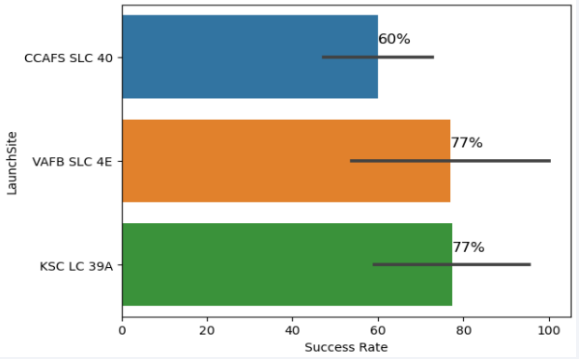
Create a landing outcome label from Outcome column

```
bad_outcomes=df.Outcome.value_counts().keys()[[2,3,5,6,7]]
landing_class=[0 if outcome in bad_outcomes else 1
               for outcome in df.Outcome]
df['Class']=landing_class
```

EDA with Data Visualization

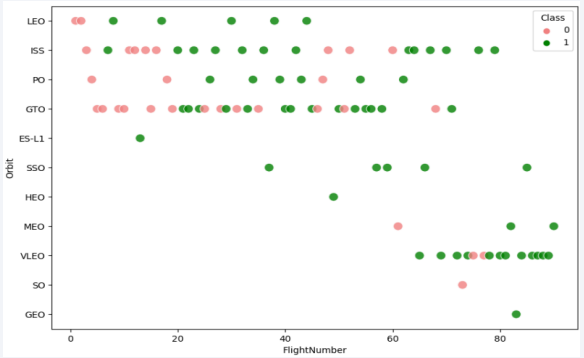
Notebook link

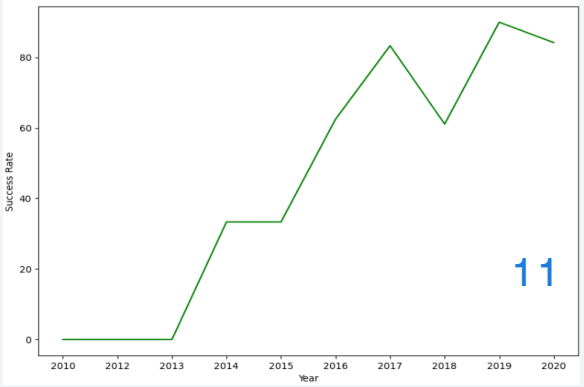
- **Bar graph** is plotted to get Success Rate with Launch Sites and Orbit type.



- **Line chart** is plotted to get average launch success trend.

- **Scatter plot** is plotted to get relationship with Launch Outcome :-
  - Flight no. vs Payload Mass
  - Flight no. vs Launch Site
  - Flight no. vs Orbit Type
  - Payload Mass vs Launch Site
  - Payload Mass vs Orbit Type





EDA with SQL

Notebook link

- Loaded the SQL extension, established a database connection.
- Displayed unique launch sites' name.
- Retrieved records where launch sites start with 'CCA'
- Calculated the total payload mass for NASA's CRS boosters.
- Obtained the average payload mass for booster version F9 v1.1
- Identified the date of the first successful ground pad landing.
- List the total counts of successful and failed mission outcomes.
- Rank the count of landing outcomes in date range.
- Listed boosters that successfully landed on a drone ship with a payload mass between 4000 and 6000.
- Listed the records displaying the month, failure landing outcomes on a drone ship, booster versions, and launch sites in year 2015.

```
%sql \
select distinct(launch_site) \
from spacextable
```

```
 * sqlite:///my_data1.db
Done.
```

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

Build an Interactive Map with Folium

GitHub lab
Notebook link

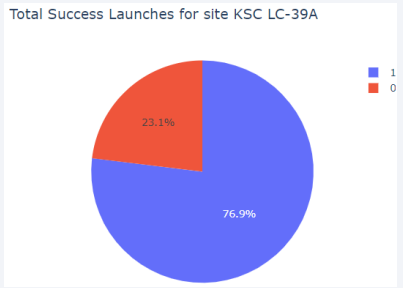
- Created a Folium map centered at NASA JSC with highlighted circles and text labels for launch sites using folium.Circle and folium.Marker.

- Added equator line with folium.PolyLine to see distance with launch sites.
- Added launch outcomes to each site to find high success rates and used green markers for successful launches (class=1) and red markers for failed launches (class=0), while simplifying with marker clusters.
- Calculated distances from launch sites to key locations, displayed them on a map using Folium with lines and markers.

Build a Dashboard with Plotly Dash

GitHub lab
Notebook link

- Added a **dropdown list** to filter data by launch site.



- Added a **pie chart** for an overview of success rates, showing:-
  - Total successful launches for all sites when "All Sites" are selected
  - Success vs. failure counts when a specific launch site is chosen

- Added a **range slider** for users to choose the payload weight range they want to look at.

- Added a **scatter chart** to display how payload mass affects launch outcomes. It considers the chosen launch site and payload range from the slider.

Predictive Analysis (Classification)

Notebook link

1. Data Standardization

```
transform = preprocessing.StandardScaler()
X=transform.fit_transform(X)
```

2. Data Splitting

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

3. Model Training with Hyperparameters Tuning and Selection

```
lr=LogisticRegression(random_state=2)
parameters ={'C':[0.01,0.1,1],'penalty':['l2'],
             'solver':['lbfgs']}
logreg_cv=GridSearchCV(lr,param_grid=parameters,
                       cv=10)
logreg_cv.fit(X_train,Y_train)
```

4. Model Evaluation

```
logreg_score=logreg_cv.score(X_test,Y_test)
yhat=logreg_cv.predict(X_test)
logreg_f1=f1_score(Y_test,yhat)
logreg_proba=logreg_cv.predict_proba(X_test)[:,1]
logreg_auc=roc_auc_score(Y_test,logreg_proba)
plot_confusion_matrix(Y_test,yhat)
```

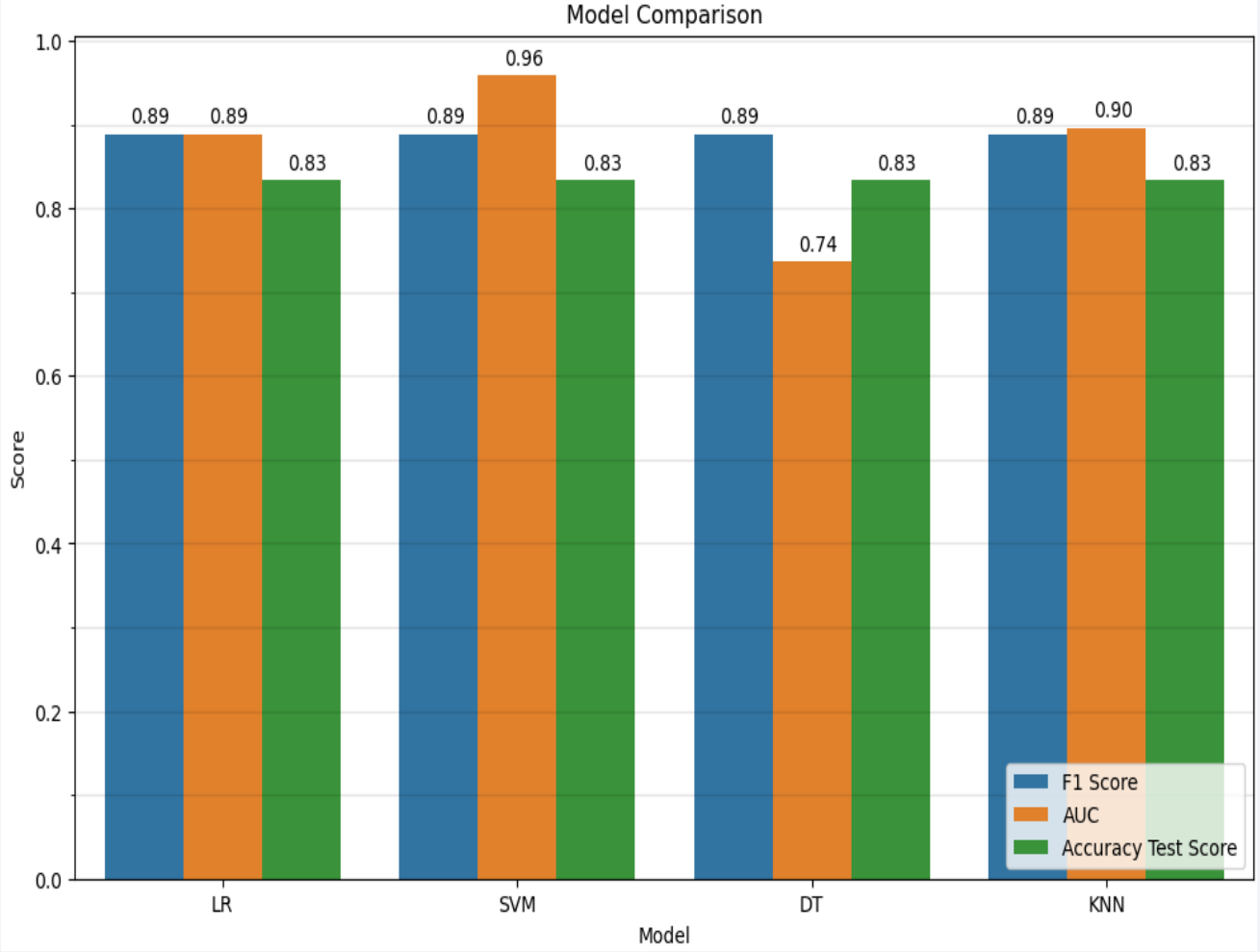
5. Determine Best Method

```
pd.DataFrame({'Model':['LR','SVM','DT','KNN'],
              'F1 Score':[logreg_f1,svm_f1,tree_f1,knn_f1],
              'AUC':[logreg_auc,svm_auc,tree_auc,knn_auc],
              'Accuracy Test Score':[logreg_score,svm_score,
                                     tree_score,knn_score]}
             ).set_index('Model')
```

Results

- Exploratory data analysis results
  - More flights, better first stage landings
  - Heavier payloads, worse first stage returns.
  - Positive trend in success rate with no. of launches.

- Predictive analysis results
  - All models have similar accuracy and F1 Score.
  - Here, SVM is the best ML model.

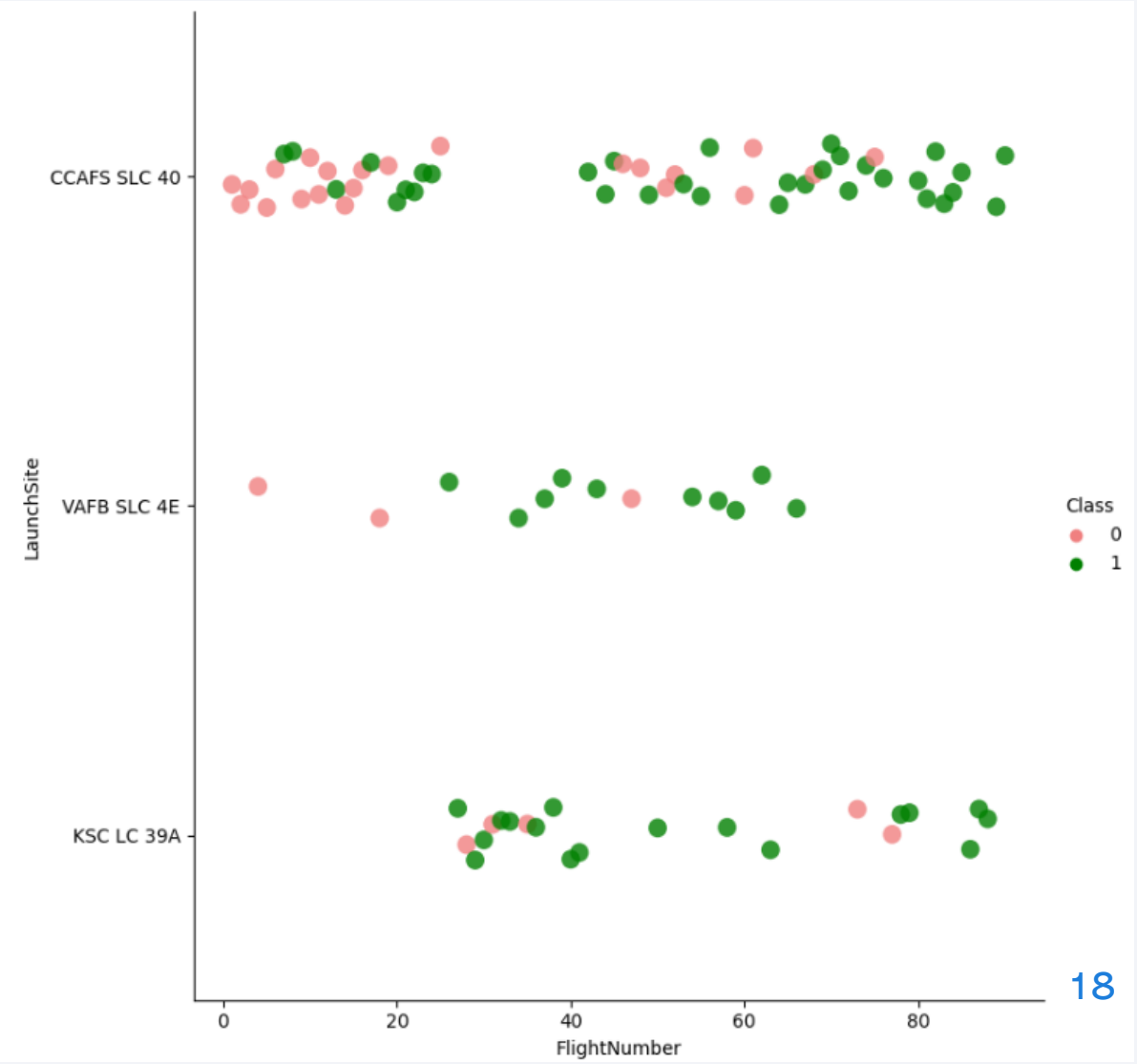


Section 2

Insights drawn from EDA

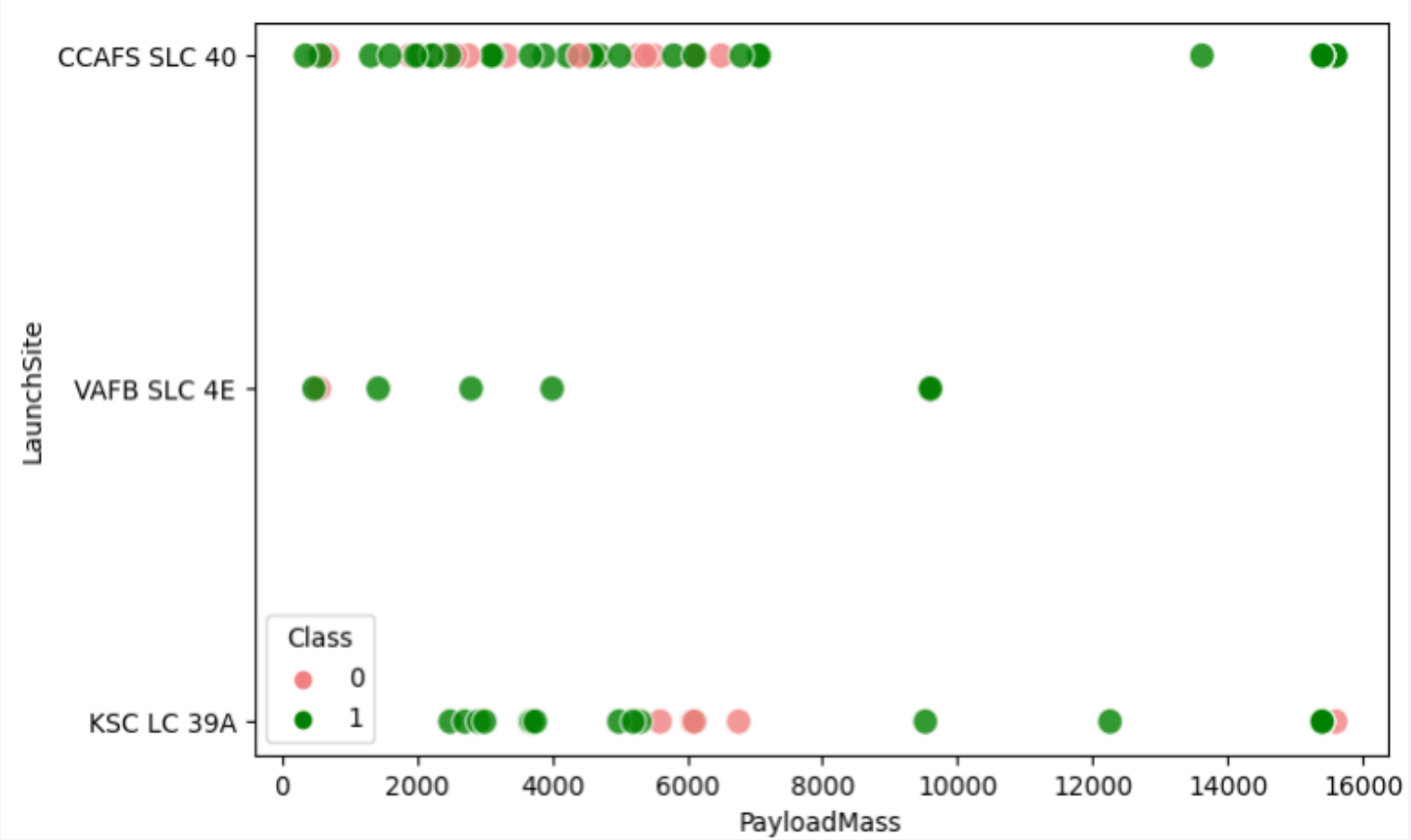
Flight Number vs. Launch Site

- The launch site `**KSCKC 39A**` was first used after the 25th flight.
- There is a higher launch frequency
  - For `**CCAFS SLC-40**` with flight numbers between 1-25 and above 40
  - For `**KSCKC 39A**` between flight numbers 25-40.
- There is a positive trend in the success rate with increasing flight numbers.



Payload vs. Launch Site

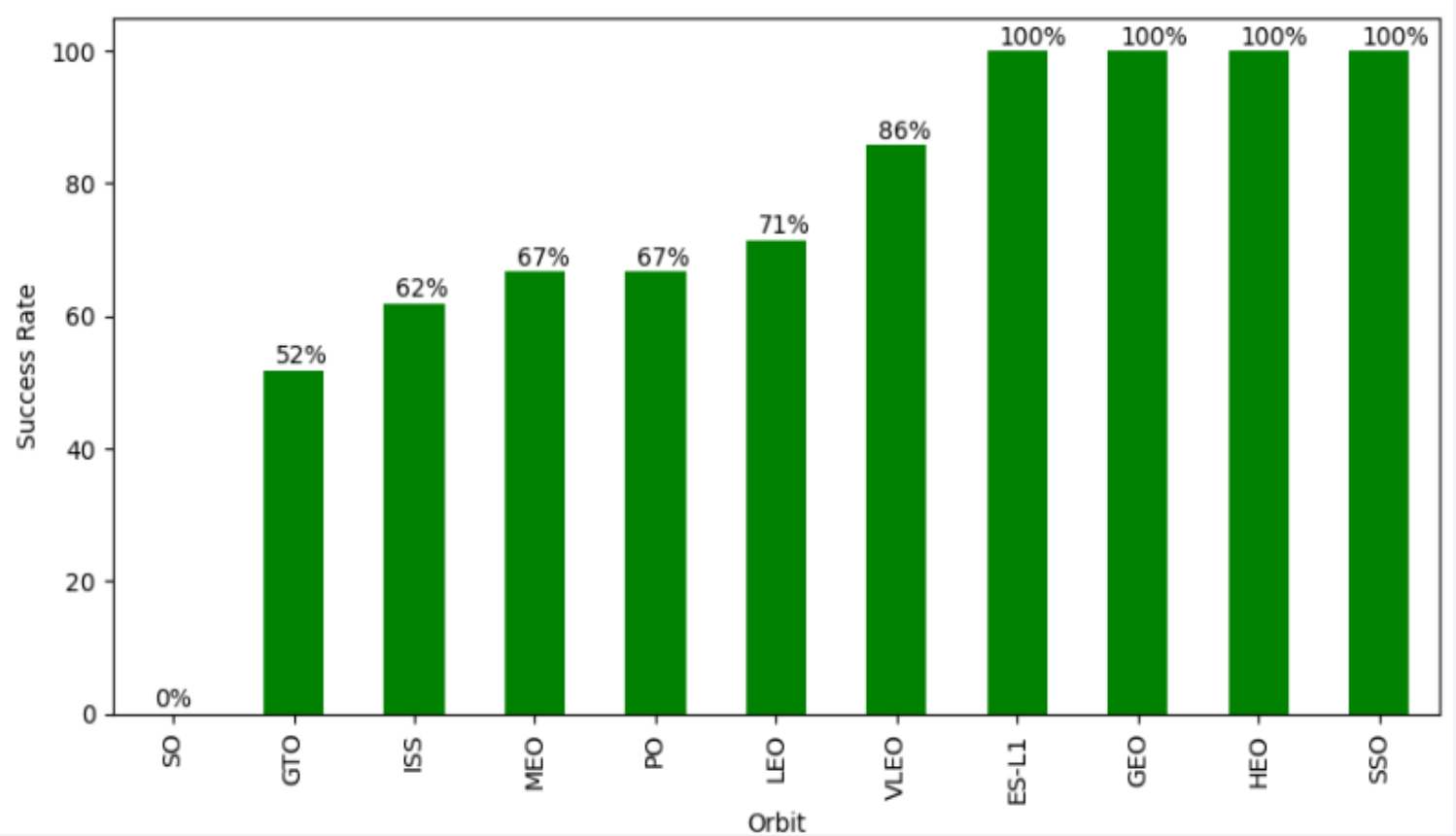
- Only few rockets launched for heavy payload mass (greater than 10000 Kg).
- **VAFB-SLC** launch site is least used.



Success Rate vs. Orbit Type

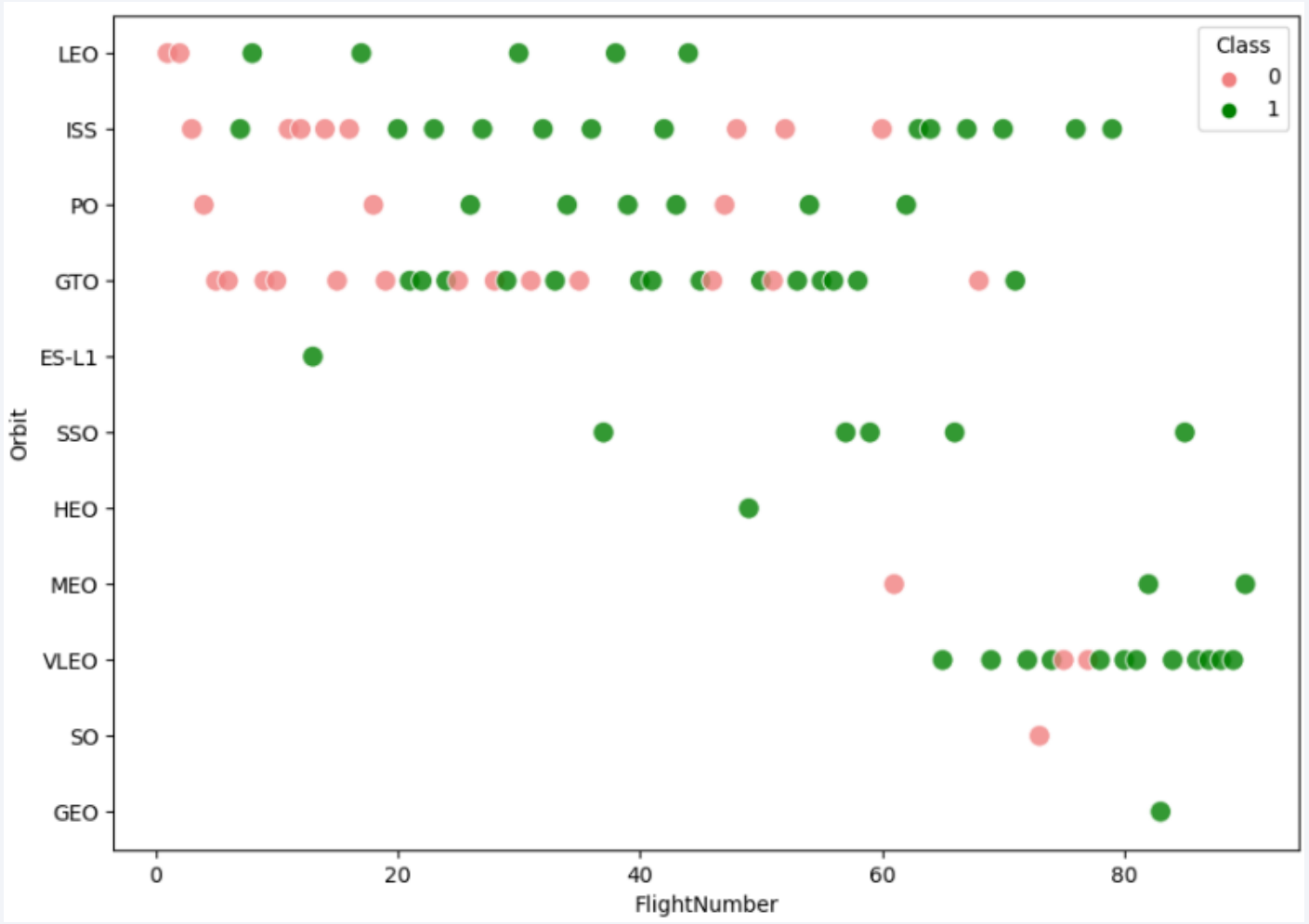
- Flights went to orbit type **ES-L1**, **GEO**, **HEO** , **SSO** & **TU** have highest success rate with **100%**

- Orbit type **SO** has least success rate with **0%**.



Flight Number vs. Orbit Type

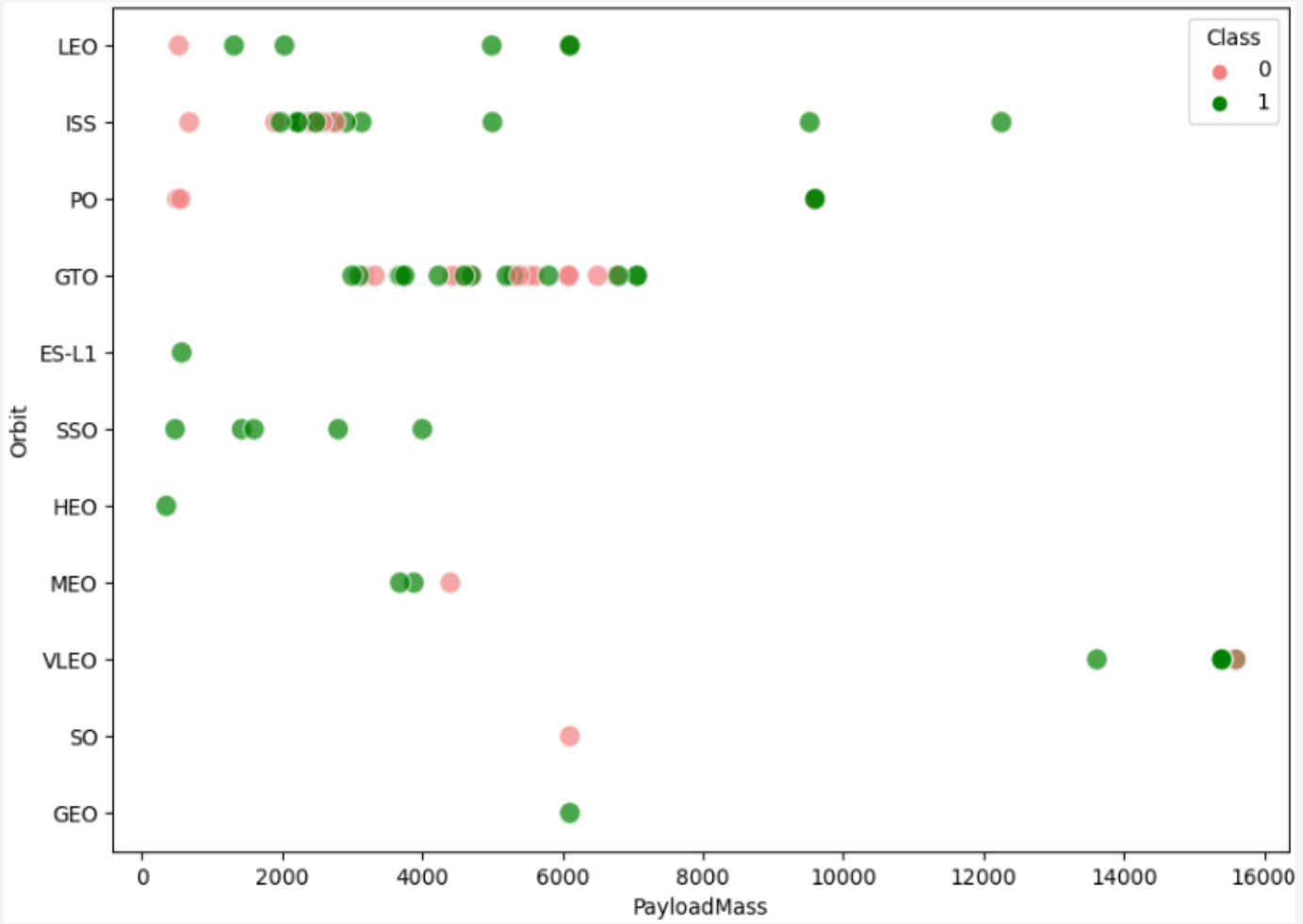
- For **LEO** & **VLEO** orbit, the Success appears related to the number of flights.
- And, there seems to be no relationship between flight number when in **GTO** orbit.



Payload vs. Orbit Type

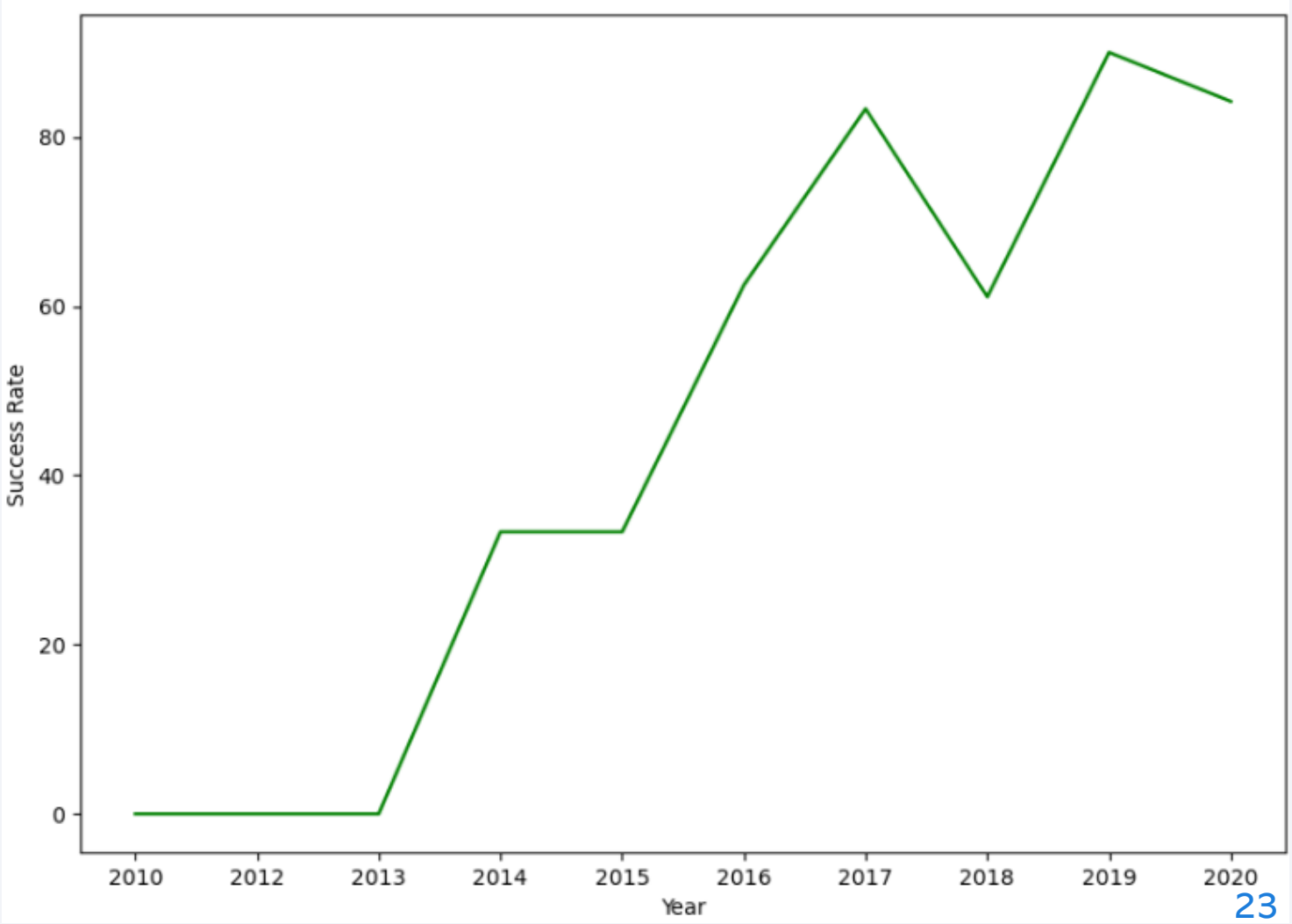
- Wit heavy payloads (above 6000kg) the successful landing or positive landing rate are more for Polar, LEO and ISS.

- 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

- The success rate since 2013 kept increasing till 2017 (stable in 2014) and after 2015 it started increasing.



All Launch Site Names

- The query result displays the distinct launch sites from the "spacextable" in the database.

- There are four unique launch sites:
  - CCAFS LC-40
  - VAFB SLC-4E
  - KSC LC-39A
  - CCAFS SLC-40

```
%sql \
select distinct(launch_site) \
from spacextable
```

```
 * sqlite:///my_data1.db
Done.
```

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

Launch Site Names Begin with 'CCA'

- The query selects the :-
  - first 5 records
  - from the "spacextable"
  - where the launch site starts with "CCA"

```
%%sql
select * from spacextable
where launch_site like 'CCA%'
limit 5;
```

```
 * sqlite:///my_data1.db
Done.
```

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- This query provides the total payload mass for missions conducted for NASA's Commercial Resupply Services (CRS) program, which can be useful for understanding the payload requirements for these missions.

```
%sql select sum(PAYLOAD_MASS__KG_) \
from spacextable \
where customer = 'NASA (CRS)'
```

```
 * sqlite:///my_data1.db
Done.
```

sum(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- This query calculates and displays the average payload mass specifically for the 'F9 v1.1' booster version from the 'spacextable' database.

```
%sql select avg(payload_mass__kg_) \
from spacextable \
where booster_version = 'F9 v1.1'
```

```
 * sqlite:///my_data1.db
Done.
```

avg(payload_mass__kg_)
2928.4

First Successful Ground Landing Date

- This query extracts the date when the first successful landing outcome in ground pad was achieved.

```
%sql \
select min(date) from spacextable \
where landing_outcome = 'Success (ground pad)'
```

```
 * sqlite:///my_data1.db
Done.
```

min(date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- This query helps identify specific booster versions used in successful drone ship landings for payloads within the specified range

```
%sql \
select booster_version from spacextable \
where landing_outcome = 'Success (drone ship)' \
and payload_mass__kg_ > 4000 \
and payload_mass__kg_ < 6000
```

```
 * sqlite:///my_data1.db
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

- This query groups and counts the occurrences of different mission outcomes in the "spacextable" dataset, providing a summary of the number of successful and failed missions.

```
%sql \
select  mission_outcome,count(*) 'Total_No.' \
from spacextable \
group by mission_outcome
```

```
 * sqlite:///my_data1.db
Done.
```

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

Boosters Carried Maximum Payload

- This query identifies the booster versions associated with the heaviest payloads.

```
%sql \
select distinct booster_version \
from spacextable \
where payload_mass__kg_= \
(select max(payload_mass__kg_) \
 from spacextable)
```

```
 * sqlite:///my_data1.db
Done.
```

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

- The query extracts data from the table - 'spacextable' for the year 2015 having failure landing outcomes in drone ship

```
%sql \
select substr(date,6,2) 'Month',date, \
landing_outcome,booster_version,launch_site \
from spacextable \
where substr(date,1,4)='2015' \
and landing_outcome='Failure (drone ship)'
```

```
 * sqlite:///my_data1.db
Done.
```

Month	Date	Landing_Outcome	Booster_Version	Launch_Site
10	2015-10-01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- This query summarizes the count of different SpaceX rocket landing outcomes within the specified date range, displaying the results in descending order of count.

- It provides insight into the success and failure of landing attempts during this period.

```
%sql \
select landing_outcome, \
count(landing_outcome) 'Total Count' \
from spacextable \
where date between '2010-04-06' and '2017-03-20' \
group by landing_outcome \
order by 2 desc
```

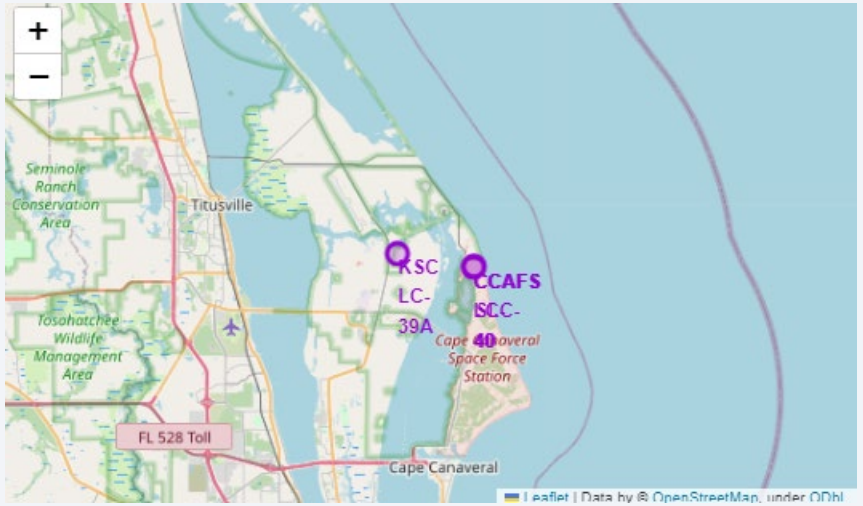
```
 * sqlite:///my_data1.db
Done.
```

Landing_Outcome	Total Count
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1


Section 3
Launch Sites
Proximities Analysis

All Launch Sites on Map

- For each launch site,
  - A circle at the site's coordinates, representing the launch area
  - A icon of the launch site's name is present at its coordinates



- The visualization shows that all launch sites are located in close proximity to the coast.

This strategic placement ensures safety, convenience for transportation, and reduced environmental impact.

Color Code Launch Outcomes

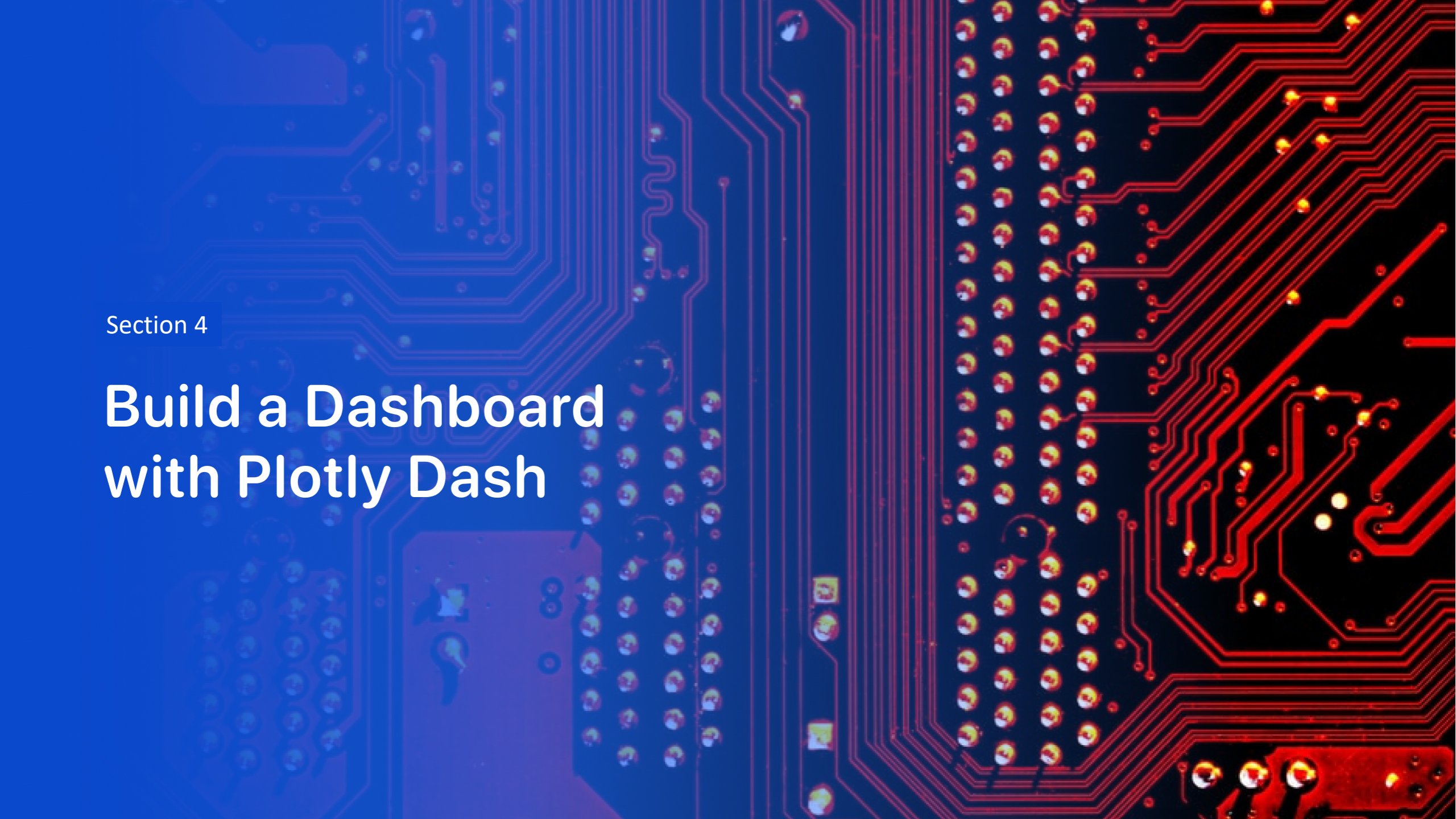
- We can easily identify which launch sites have relatively high success rates (green markers) and which ones have a higher number of failures (red markers).

- This visualization helps in assessing launch success at different sites.

Key Location Proximities

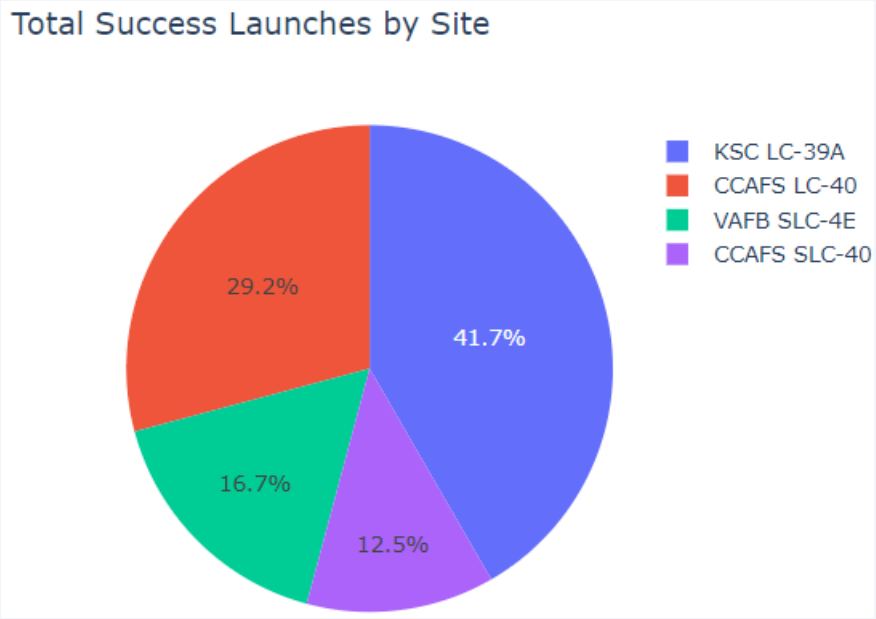
- Displaying the distances on a map using markers and lines, making it visually informative.

- Findings:
  - Launch sites near railways and highways simplify transportation of equipment and personnel.
  - Launch sites near coastlines provide a safe area for rocket stages to land after separation.
  - Keeping launch sites at a distance from cities minimizes risks to people and property, especially in the event of launch failures or falling debris.


Section 4
Build a Dashboard with Plotly Dash

Total Successful Launches by Site

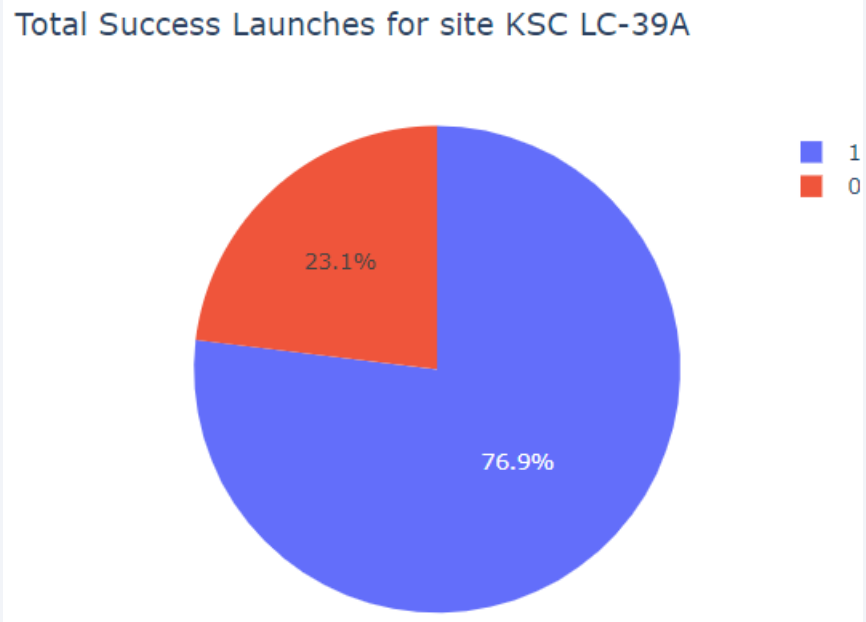
- The pie chart shows the total success launches by site.
- According to the chart, the launch sites **KSC LC-39A** has the highest number of successful launches.



Launch Site with highest Launch Success Ratio

- The pie chart showing the highest success ratio of 76.9% at site KSC LC-39A, with 10 successful landings and 3 failed landings.

- This data indicates that the launch site is highly reliable.



Payload vs Success with Booster Version

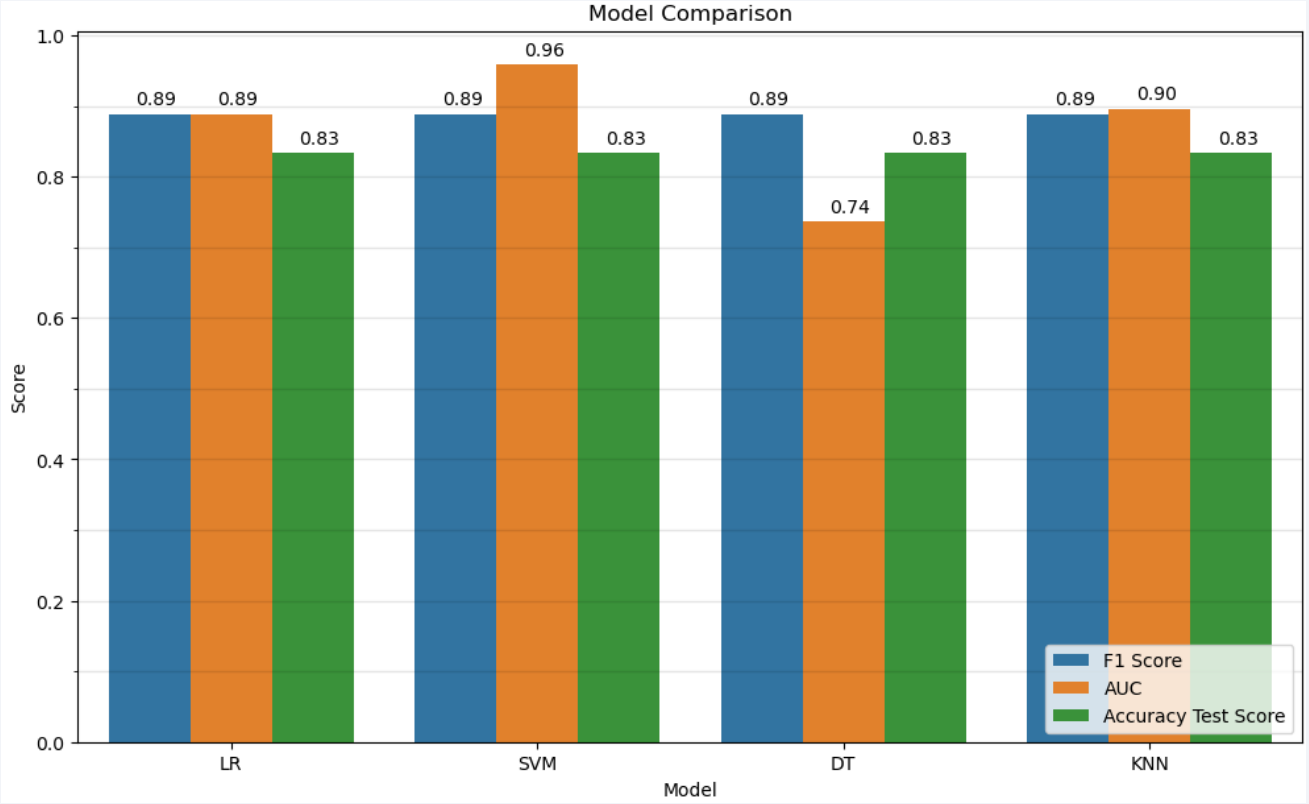
- Y-axis represents the launch outcome (1 for success, 0 for failure).
- There is a weak positive correlation between payload mass and launch outcome.
- The Falcon 9 booster has a slightly higher success rate as it has a more powerful and reliable rocket.

Section 5

Predictive Analysis (Classification)

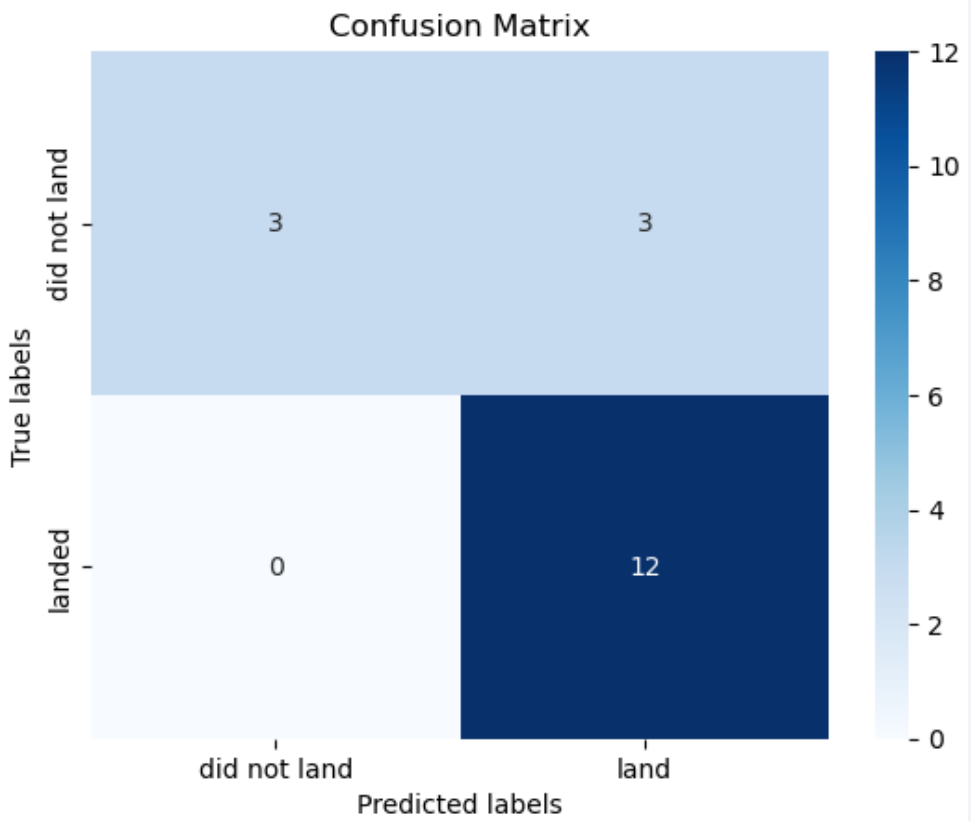
Classification Accuracy

- All models have similar accuracy and F1 Score.
- The SVM model appears to have an advantage in terms of its ability to discriminate between classes, as indicated by the higher AUC score.



Confusion Matrix

- All models perform identically on the test set, so confusion matrix is also same.
- The models predict successful landings well when the true label is a successful landing, but they tend to over-predict successful landings, leading to false positives.
- The models are more likely to classify a landing as successful even when it's not.



Conclusions

- There is a positive trend in the success rate as more launches occur.
- Flights went to orbit type ES-L1, GEO, HEO , SSO & TU have 100% success rate.
- For LEO & VLEO orbit, the Success appears related to the number of flights.
- All launch sites are located near the coast, highways, railways, and far from cities.
- Launch sites KSC LC-39A has the highest number of successful launches.
- The Falcon 9 booster has a slightly higher success rate compared to others.
- The Support Vector Machine is the best Machine Learning model for this dataset.

Appendix- Haversine Formula

- The Haversine formula used to calculate the distance between two points on the surface of a sphere. It's particularly useful for determining the great-circle distance (the shortest distance over the Earth's surface) between two coordinates specified in terms of latitude and longitude.

- The formula is named after the haversine function, which is used in the calculation.

Haversine Formula

1. Calculate a:

- $a = \sqrt{hav(\Delta\phi) + cos\phi_1 * cos\phi_2 * hav(\Delta\lambda)}$

where

- $hav(x) =$ haversine of $x = \sin^2(\frac{x}{2})$
- ϕ_1, ϕ_2 and $\lambda_1, \lambda_2 = latitude_1, latitude_2$ and $longitude_1, longitude_2$
- $\Delta\phi$ and $\Delta\lambda = latitude_1 - latitude_2$ and $longitude_1 - longitude_2$

2. Calculate θ:

- $\theta = 2 * \sin^{-1}(\sqrt{a})$ - $\theta = 2 * \tan^{-1}(\frac{\sqrt{a}}{\sqrt{1-a}})$

$l = r * \theta$

Appendix- Hyperlinks

GitHub Repo Link

IBM Course Link

- Special Thanks to All Instructors:

Thank you!