IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Vishal Srivastava

Outline

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

Executive Summary

- Summary of methodologies
  - Data were collected with several ways
  - Machine learning models were built
  - Data visualizations were created
- Summary of all results
  - The optimal model was acquired
  - Visualizations were great for decision making

Introduction

- Project background and context

  In this project I will work in SpaceX company and try to predict the Falcon 9 first stage. It's important to know if the rockets will land successfully or not because the failure will cost the company much resources.

- Problems that need answers

  1. Which factors are behind the failure of landing?
  2. Will the rockets land successfully?
  3. What the accuracy of a successful landing?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - With Rest API and Web Scrapping
- Perform data wrangling
  - Data were transformed and one hot encoded to be apply later on the Machine Learning models
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Discovering new patterns in the data with visualization techniques such as scatter plots
- Perform interactive visual analytics using Folium and Plotly Dash
  - Dash and Folium were used to achieve this goal
- Perform predictive analysis using classification models
  - Classification machine learning models were built to achieve this goal

Data Collection

Data sets were collected using the API call from several websites, I collected rocket, launchpad, payloads, and cores data from https://api.spacexdata.com/v4 website.

1. Collecting the data with API call
2. Converting to data frame with help of JSON
3. Updating columns and rows (pre-processing)
4. Filtering the data to keep only Falcon 9 launches
5. Convert the data to csv file with name 'dataset_part_1.csv'

Data Collection – SpaceX API

1. Collecting the data with API call

2. Converting to data frame with help of JSON

3. Updating columns and rows (pre-processing)

4. Filtering the data to keep only Falcon 9 launches

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9.drop(data_falcon9[data_falcon9['BoosterVersion']!='Falcon 9'].index, inplace = True)
```

5. Convert the data to csv file with name 'dataset_part_1.csv'

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

GitHub repo: https://github.com/yousefhosam-b/IBM_SpaceX_Capstone_Project/tree/main/1.%20spacex-data-collection-api

Data Collection - Scraping

GitHub repo: https://github.com/yousefhosam-b/IBM_SpaceX_Capstone_Project/tree/main/2.%20webscraping

1. Creating the BeautifulSoup object

```python
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data, 'html5lib')
```

2. Getting column names

```python
ths = first_launch_table.find_all('th')
for th in ths:
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

3. Creating the launch_dict

```python
# Customer
# TODO: Append the customer into launch_dict with key 'Customer'
if (row[6].a is not None):
    customer=row[6].a.string
else:
    customer=row[6].string
launch_dict['Customer'].append(customer)

# Launch outcome
# TODO: Append the launch_outcome into launch_dict with key 'Launch outcome'
launch_outcome = list(row[7].strings)[0]
launch_dict['Launch outcome'].append(launch_outcome)

# Booster landing
# TODO: Append the launch_outcome into launch_dict with key 'Booster landing'
booster_landing = landing_status(row[8])
launch_dict['Booster landing'].append(booster_landing)
```

4. Converting to final data frame

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

5. Convert the data to csv file with name 'spacex_web_scraped.csv'

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

Data Wrangling

GitHub repo: https://github.com/yousefhosam-b/IBM_SpaceX_Capstone_Project/tree/main/3.%20spacex-data-wrangling

1. Loading the data set

```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/d
art_1.csv")
df.head(10)
```

```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

2. Creating landing outcomes

3. Finding the bad outcomes

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

4. Presenting outcomes as 0 and 1

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

5. Determining the success outcome

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

```
0.6666666666666666
```

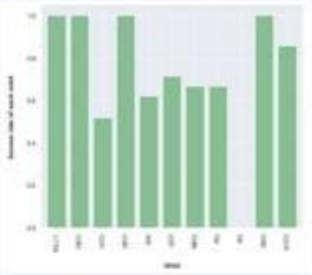
6. Convert the data to csv file with name 'dataset_part_2.csv'

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

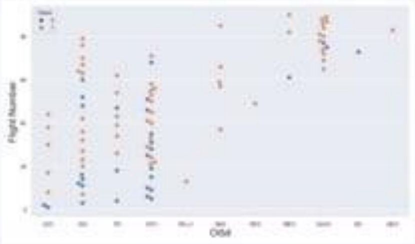
EDA with Data Visualization

GitHub repo: https://github.com/yousefhosam-h/IBM_SpaceX_Capstone_Project/tree/main/5.%20eda-datavi

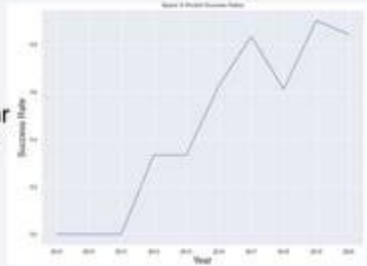
Categorial plot between Flight number and Pay load mass (kg)



Bar chart between Orbit and Success rate of each orbit



Scatter plot between Orbit and Flight number



Line chart between Year and Success rate

EDA with SQL

GitHub repo: https://github.com/yousefhosam-b/IBM_SpaceX_Capstone_Project/tree/main/4.%20eda-sql

I used SQL queries to answer the following questions:

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

Build an Interactive Map with Folium

- folium.Marker() was used to create marks on the maps.
- folium.Circle() was used to create a circles above markers on the map.
- folium.Icon() was used to create an icon on the map.
- folium.PolyLine() was used to create polynomial line between the points.
- folium.plugins.AntPath() was used to create animated line between the points.
- markerCluster() was used to simplify the maps which contain several markers with identical coordination.

GitHub repo: https://github.com/yousefhosam-b/IBM_SpaceX_Capstone_Project/tree/main/6.%20launch-site-location

Build a Dashboard with Plotly Dash

- Dash and html components were used as they are the most important thing and almost everything depends on them, such as graphs, tables, dropdowns, etc.
- Pandas was used to simplifying the work by creating dataframe.
- Plotly was used to plot the graphs.
- Pie chart and scatter chart were used to for plotting purposes.
- Rangeslider was used for payload mass range selection.
- Dropdown was used for launch sites.

GitHub repo: https://github.com/yousefhosam-b/IBM_SpaceX_Capstone_Project/tree/main/7.%20dashboard

Predictive Analysis (Classification)

1. Building the model

Create column for the class

Standardize the data

Split the data info train and test sets

Build GridSearchCV model and fit the data

2. Evaluating the model

Calculating the accuracies

Calculating the confusion matrixes

Plot the results

3. Finding the optimal model

Find the best hyperparameters for the models

Find the best model with highest accuracy

Confirm the optimal model

GitHub repo: https://github.com/yousefhosam-b/IBM_SpaceX_Capstone_Project/tree/main/8.%20machine-learning-prediction

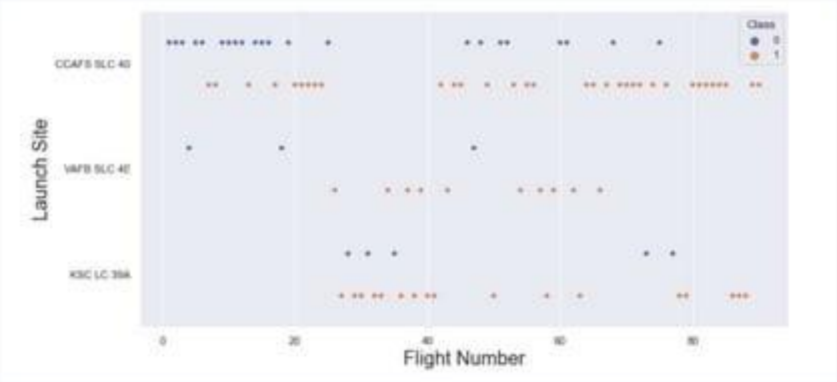
Results

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

Section 2

Insights drawn from EDA

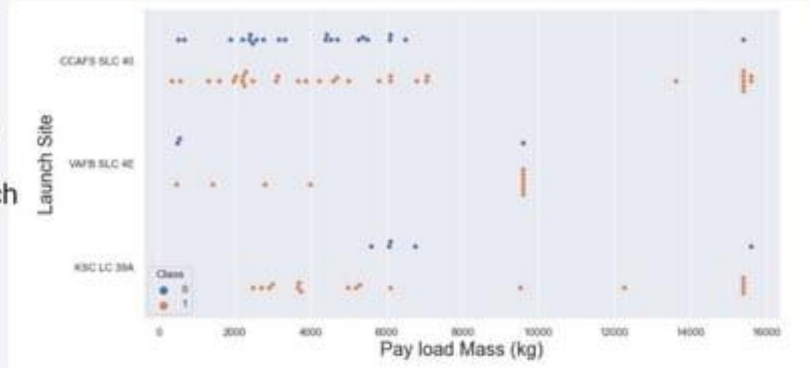
Flight Number vs. Launch Site



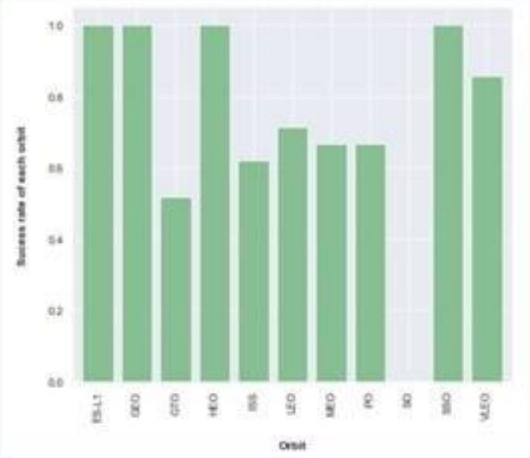
With the increase of flight number, the success rate is increasing as well in the launch sites

Payload vs. Launch Site

With the increase of Pay load Mass, the success rate is increasing as well in the launch sites



Success Rate vs. Orbit Type

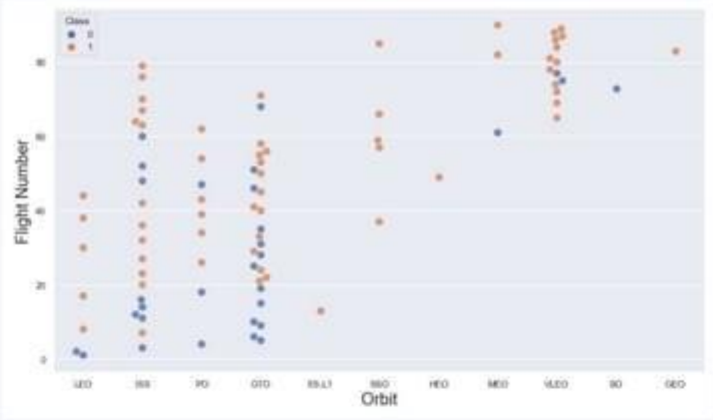


ES-L1, GEO, HEO, and SSO have a success rate of 100%

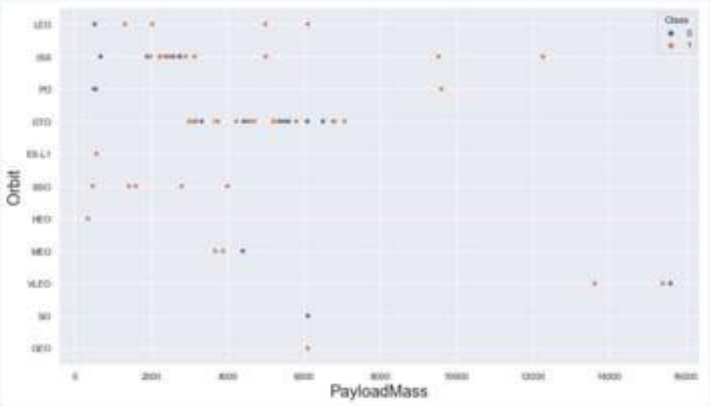
SO has a success rate of 0%

Flight Number vs. Orbit Type

It's hard to tell anything here, but we can say there is no actual relationship between flight number and GTO.



Payload vs. Orbit Type



First thing to see is how the Pay load Mass between 2000 and 3000 is affecting ISS.

Similarly, Pay load Mass between 3000 and 7000 is affecting GTO.

Launch Success Yearly Trend

Since the year 2013, there was a massive increase in success rate. However, it dropped little in 2018 but later it got stronger than before.

All Launch Site Names

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;
```

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

We can get the unique values by using "DISTINCT"

Launch Site Names Begin with 'CCA'

```
%sql SELECT * FROM SPACEX WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

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

We can get only 5 rows by using "LIMIT"

Total Payload Mass

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)" FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)';
```

Total Payload Mass by NASA (CRS)
45596

We can get the sum of all values by using "SUM"

Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) AS "Average Payload Mass by Booster Version F9 v1.1" FROM SPACEX \
WHERE BOOSTER_VERSION = 'F9 v1.1';
```

Average Payload Mass by Booster Version F9 v1.1
2928

We can get the average of all values by using “AVG”

First Successful Ground Landing Date

```
%sql SELECT MIN(DATE) AS "First Succesful Landing Outcome in Ground Pad" FROM SPACEX \
WHERE LANDING__OUTCOME = 'Success (ground pad)';
```

First Succesful Landing Outcome in Ground Pad
2015-12-22

We can get the first successful data by using "MIN", because first date is same with the minimum date

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

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

The payload mass data was taken between 4000 and 6000 only, and the landing outcome was determined to be "success drone ship"

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Successful Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Success%';
```

Successful Mission
100

We can get the number of all the successful mission by using "COUNT" and LIKE "Success%"

```
%sql SELECT COUNT(MISSION_OUTCOME) AS "Failure Mission" FROM SPACEX WHERE MISSION_OUTCOME LIKE 'Failure%';
```

Failure Mission
1

We can get the number of all the failure mission by using "COUNT" and LIKE "Failure%"

Boosters Carried Maximum Payload

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions which carried the Maximum Payload Mass" FROM SPACEX \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX);
```

Booster Versions which carried the Maximum Payload Mass
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

We can get the maximum payload masses by using "MAX"

2015 Launch Records

```
%sql SELECT month(DATE) as Month, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE year(DATE) = '2015' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

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

We can get the months by using month(DATE) and in the WHERE function we assigned the year value to "2015"

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

```
%sql SELECT LANDING__OUTCOME as "Landing Outcome", COUNT(LANDING__OUTCOME) AS "Total Count" FROM SPACEX \
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' \
GROUP BY  LANDING__OUTCOME \
ORDER BY COUNT(LANDING__OUTCOME) DESC ;
```

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

By using "ORDER" we can order the values in descending order, and with "COUNT" we can count all numbers as we did previously

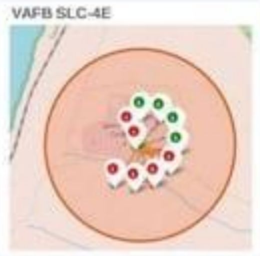
Section 4

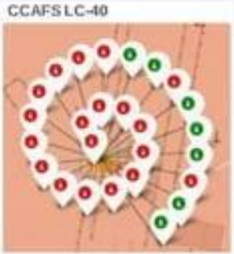
Launch Sites Proximities Analysis

All Launch Sites' Location Markers

All the launches are near USA, Florida, and California

Color-labeled Launch Outcomes





Green means successful

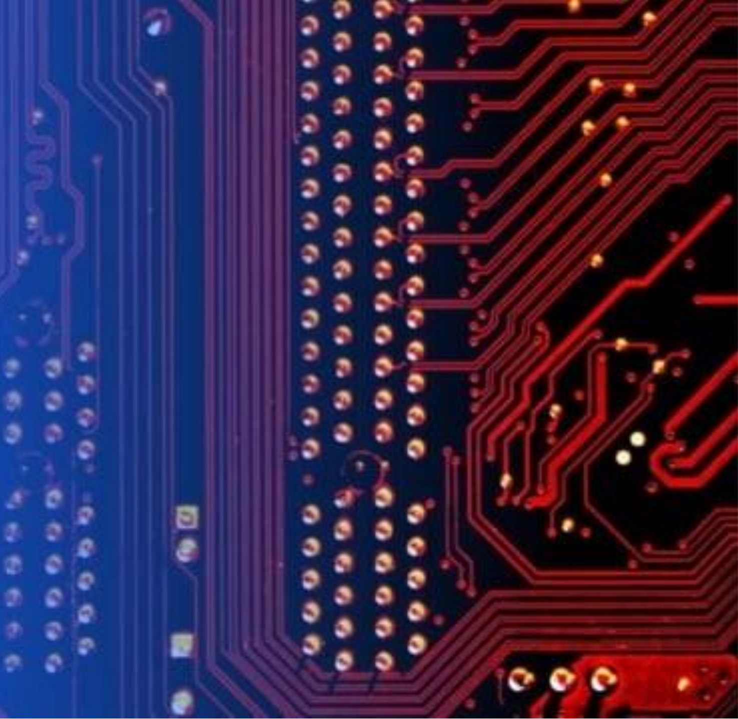
Red means Failure

Launch Sites to its Proximities

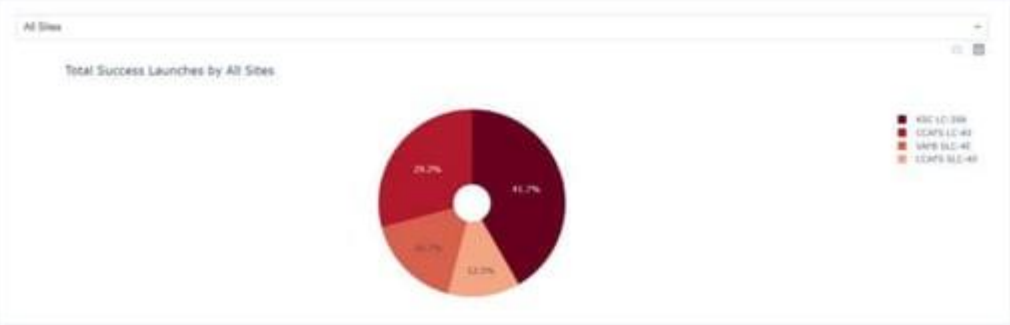
All distances from launch sites to its proximities, they weren't far from railway tracks.

Section 5

Build a Dashboard with Plotly Dash



Launch Success Count



KSC LC-39A has the highest success score with 41.7%

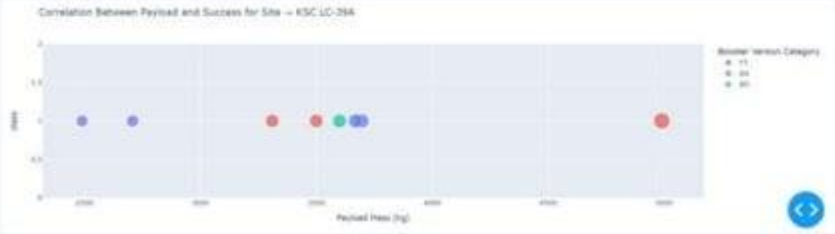
CCAFS LC-40 comes next with 29.2%

Finally, VAFB SLC-4E and CCAFS SLC-40 with 16.7% and 12.5% respectively

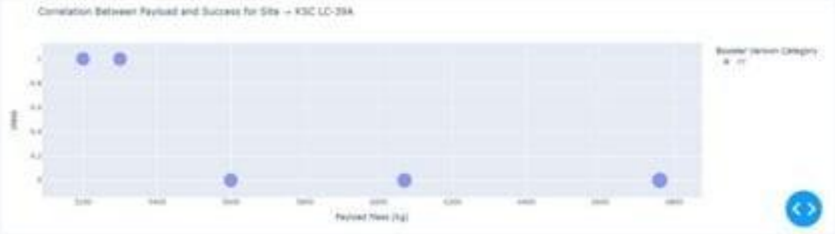
Launch Site with Highest Score

KSC LC-39A has the highest score with 76.9% with payload range of 2000 kg - 10000 kg, and FT booster version has the highest score

Payload vs. Launch Outcome



Payload 0 kg - 5000 kg
(first half)

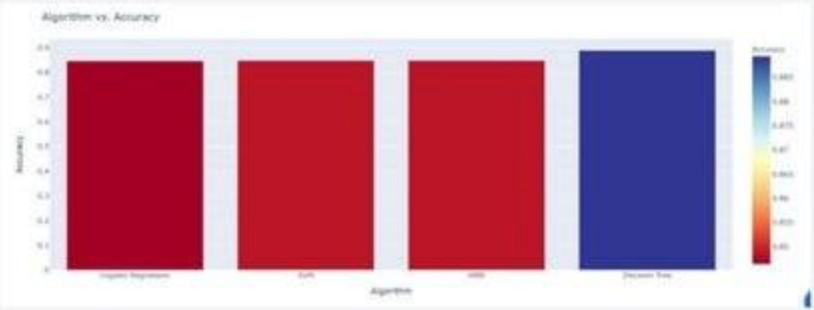


Payload 6000 kg - 10000 kg
(second half)

Section 6

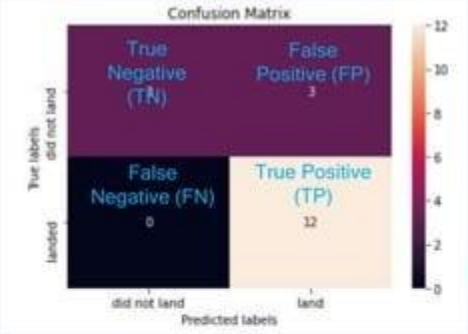
Predictive Analysis (Classification)

Classification Accuracy



Decision Tree has the highest accuracy with almost 0.89, then comes the remaining models with almost same accuracy of 0.84

Confusion Matrix



Sensitivity = 1.00, formula: TPR = TP / (TP + FN)

Specificity = 0.50, formula: SPC = TN / (FP + TN)

Precision = 0.80, formula: PPV = TP / (TP + FP)

Accuracy = 0.83, formula: ACC = (TP + TN) / (P + N)

F1 Score = 0.89, formula: F1 = 2TP / (2TP + FP + FN)

False Positive Rate = 0.50, formula: FPR = FP / (FP + TN)

False Discovery Rate = 0.20, formula: FDR = FP / (FP + TP)

Conclusions

- We found the site with highest score which was KSC LC-39A
- The payload of 0 kg to 5000 kg was more diverse than 6000 kg to 10000 kg
- Decision Tree was the optimal model with accuracy of almost 0.89
- We calculated the launch sites distance to its proximities

Thank you!