
IBM Developer
SKILLS NETWORK
DATA SCIENCE
CAPSTONE PROJECT
(Analysis on SpaceX launches)
RAVISH KUMAR VERMA
SPACEX

OUTLINE

EXECUTIVE SUMMARY

- **Summary of Methodologies -**
  - Data Collection via API, SQL and Web Scraping
  - Data Wrangling and Analysis
  - EDA with Data Visualization and SQL
  - Building interactive maps with Folium
  - Building dashboard with Plotly Dash
  - Predictive Analysis for different classification models

- **Summary of all results -**
  - Data Analysis along with Interactive Visualization
  - Most suitable Model for Predictive Analysis

INTRODUCTION

Project Background and context:

The primary goal of this project is to predict the landing outcome of the Falcon-9 rocket of SpaceX, using the best practices of Data Science.

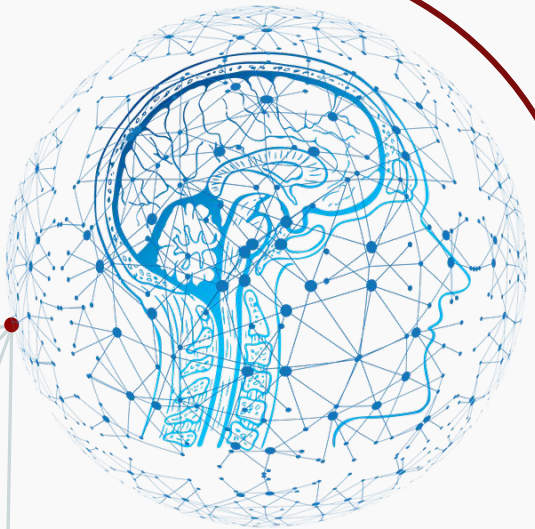
SpaceX advertises its Falcon-9 rocket on its website, stating that it costs only 62 million dollars per launch, whereas other providers cost approx. 165 million dollars each. Much of the saving of SpaceX comes from the reuse of its First Stage. Therefore, most valuable information is to predict what are chances for a Falcon-9 FIRST STAGE to land successfully. This valuable insight can be used by potential investors predict future success and growth of SpaceX and to ensure their profit, if they would bid for SpaceX.

Problems that will be answered:

- ➔ What are the factors that results in successful landing of rocket
- ➔ Relationship among different variables which determines outcome
- ➔ Factors which results into positive results i.e. successful landing


METHODOLOGY
RAVISH KUMAR VERMA
ravish1208

METHODOLOGY



- **Data Collection methodology:**
  - Using SpaceX Rest API
  - Web Scraping of Wikipedia
- **Data Wrangling:**
  - One hot encoding data fields and dropping irrelevant columns, i.e. transforming data suitable for Machine Learning
- **Exploratory Data Analysis (EDA) via Visualization and SQL:**
  - Scatter and Bar graph to show pattern and relation between different data fields.
- **Perform interactive visual analytics:**
  - Using Folium and Plotly Dash Visualization
- **Predictive analysis using Classification Models:**
  - Building and evaluation of Classification Models

Data Collection - via SpaceX API

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

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

```
data_falcon9 = df[df['BoosterVersion']!='Falcon 1']
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Getting response from API

Converting response to a .json file

Apply custom functions to clean data

Apply list to dictionary then create data frame

Getting response from API

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

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
```


	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial
4	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003
5	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005
6	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007
7	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003
8	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004

Data Collection - via Web Scraping (Wikipedia)

1. Getting response from HTML

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

2. Creating BeautifulSoup Object

```
soup = BeautifulSoup(data, 'html5lib')
```

3. Finding Tables

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

4. Getting column names

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

5. Creation of Dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
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']=[]
```

6. Appending Data to keys

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table'
   # get table row
    for rows in table.find_all("tr"):
```

7. Converting Dictionary to Data Frames

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

8. Dataframe to .csv

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

Data Wrangling

The process for cleaning and unifying messy and complex datasets for easy access and analysis

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

```
]: CCAFS SLC 40    55
   KSC LC 39A      22
   VAFB SLC 4E     13
   Name: LaunchSite, dtype: int64
```

Calculate number of launches at each Site

```
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
```

Calculate number and occurrence of mission outcome per orbit type

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

Export dataset as .csv

Calculate number and occurrence of each orbit

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

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

Create landing outcome label from outcome column

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


ravish1208

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

EDA with Data Visualization

Approach of analyzing data sets to summarize their main characteristics, using statistical graphical and other data visualization methods

Scatter Graphs

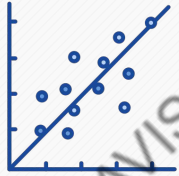
Payload and Flight number

Flight number and Launch site

Payload and Launch site

Flight number and Orbit type

Payload and orbit type



Scatter plot show dependency of attributes on each other. Once a pattern is determined from the graphs, it is very easy to predict which factor will lead to maximum probability of of success

Bar Graph

Success rate vs Orbit Type

Scatter plot show dependency of attributes on each other. Once a pattern is determined from the graphs, it is very easy to predict which factor will lead to maximum probability of of success

Line Graph

Launch Success Yearly Trend

Line Graphs are useful as they show trends clearly and thus helpful in prediction of the future

EDA with SQL

ravish1208

SQL is an indispensable tool for Data Scientists and Analysts as most of the real-world data is stored in databases. It is not only the standard language for Relational Database operations, but also an incredibly powerful tool for analyzing data and drawing useful insights from it. We are using IBM's Db2 Cloud, which is fully managed SQL Database provide as a service

SQL

We performed SQL queries to gather information on below question from the available dataset-

- The names of the unique launch sites in the space mission.
- 5 records where launch sites begin with the string 'KSC'.
- Total payload mass carried by boosters launched by NASA (CRS).
- Average payload mass carried by booster version F9 v1.1
- The list of date where the successful landing outcome in drone ship was achieved.
- The names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000
- Total number of successful and failure mission outcomes.
- The names of the booster_versions which have carried the maximum payload mass.
- Records which will display the month names, successful landing_outcomes in ground pad ,booster versions, launch_site for the months in year 2017
- Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

+

Interactive maps

with Folium

ravish1208

Building an Interactive Map with Folium

Visualization of the Launch Data into an interactive map. We took the Latitude and Longitude Coordinates at each launch site and added a *Circle Marker around each launch site with a label of the name of the launch site.*

We assigned the dataframe launch_outcomes(failures, successes) to classes 0 and 1 with Green and Red markers on the map in a MarkerCluster()

Using Haversine formula we calculated the distance from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns. Lines are drawn on the map to measure distance to landmarks

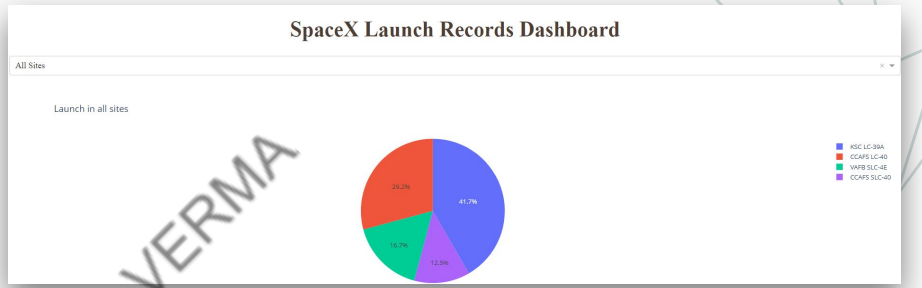
Example of some trends in which the Launch Site is situated in.

- Are launch sites in close proximity to railways? No
- Are launch sites in close proximity to highways? No
- Are launch sites in close proximity to coastline? Yes
- Do launch sites keep certain distance away from cities? Yes

Building a Dashboard with Plotly Dash

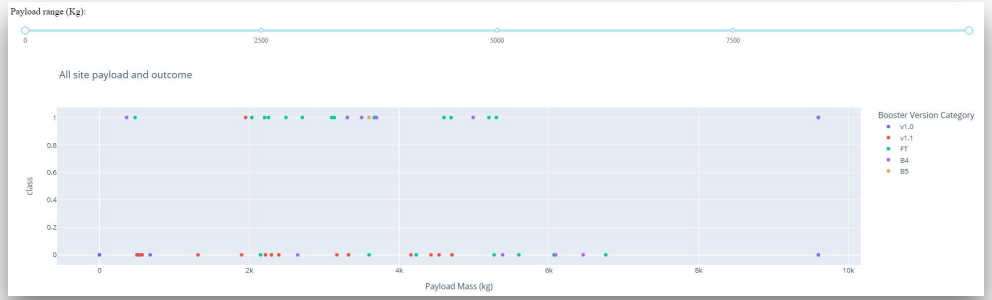
Graphs -

- **Pie Chart showing the total launches by a certain site/all sites**
- display relative proportions of multiple classes of data.
- size of the circle can be made proportional to the total quantity it represents



Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions -

- It shows the relationship between two variables.
- It is the best method to show you a non-linear pattern.
- The range of data flow, i.e. maximum and minimum value, can be determined.
- Observation and reading are straightforward.



Predictive Analysis (classification)

- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

The model with the best accuracy score wins the best performing model

BUILDING MODEL → **EVALUATING MODEL** → **FINDING BEST MODEL**

- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion MatriX

```
Y = data['Class'].to_numpy()
X= preprocessing.StandardScaler().fit(X).transform(X)
X_train, X_test, Y_train, Y_test = train_test_split( X, Y, test_size=0.2, random_state=2)
Y_test.shape
```

```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorith
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

RESULTS

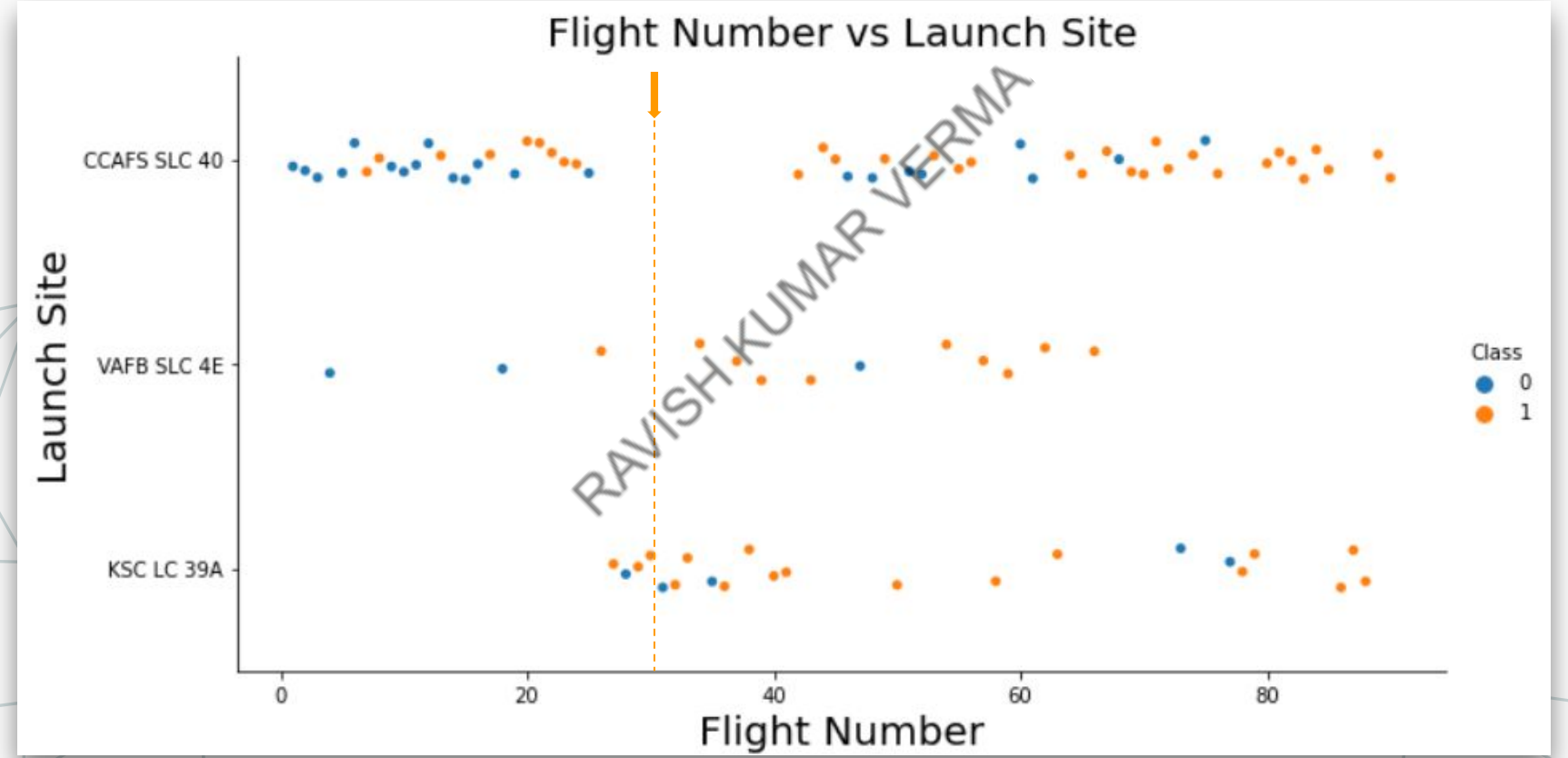
- Exploratory data analysis results
- Interactive analytics demo in screensho
- Predictive analysis results


ravish1208
RESULTS
EDA WITH VISUALISATION
RAVISH KUMAR VERMA

FLIGHT NUMBER vs LAUNCH SITE

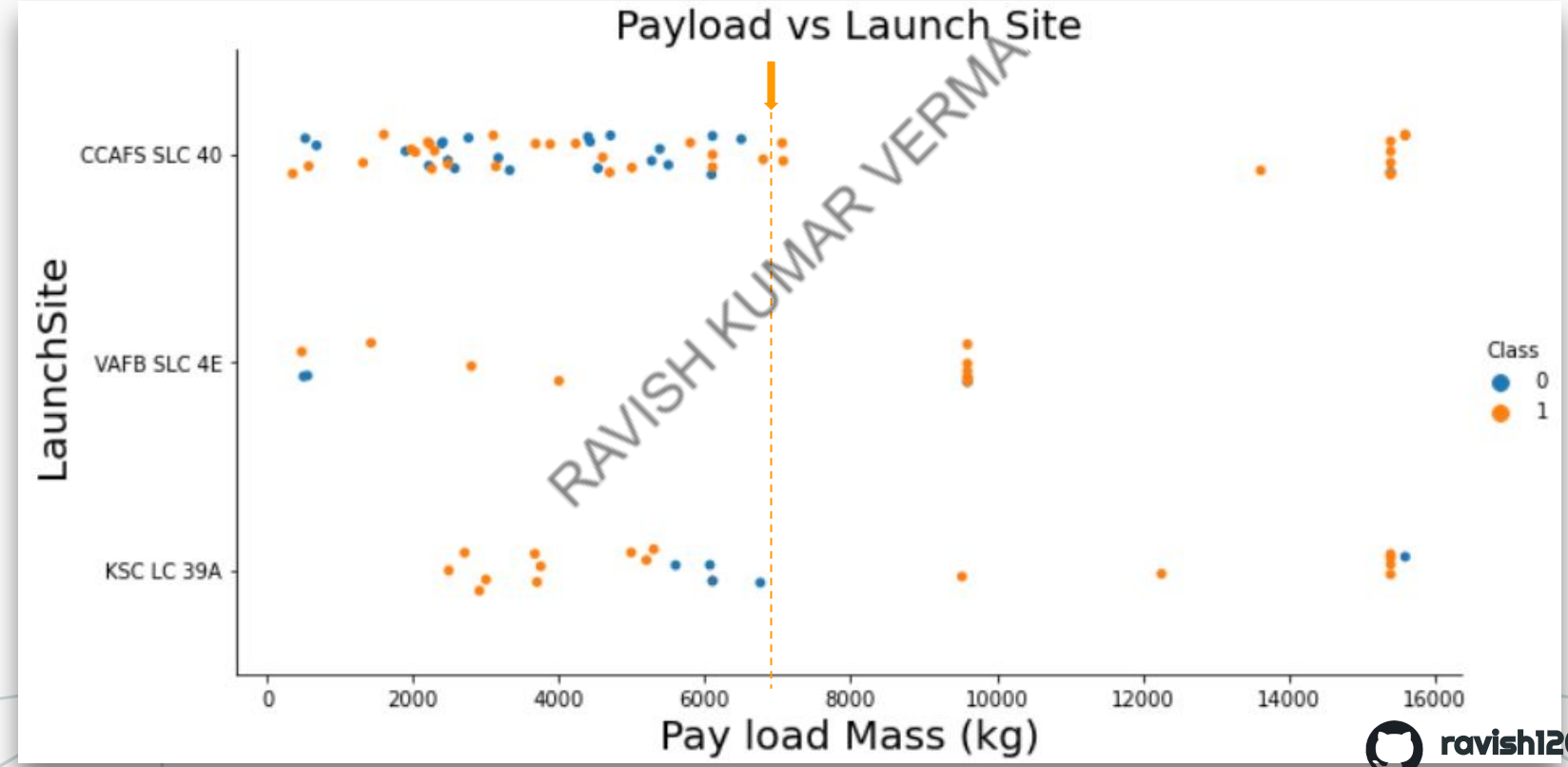
ravish1208

- With higher flight number (greater than 30) the success rate for the Rocket is increasing



PAYLOAD vs LAUNCH SITE

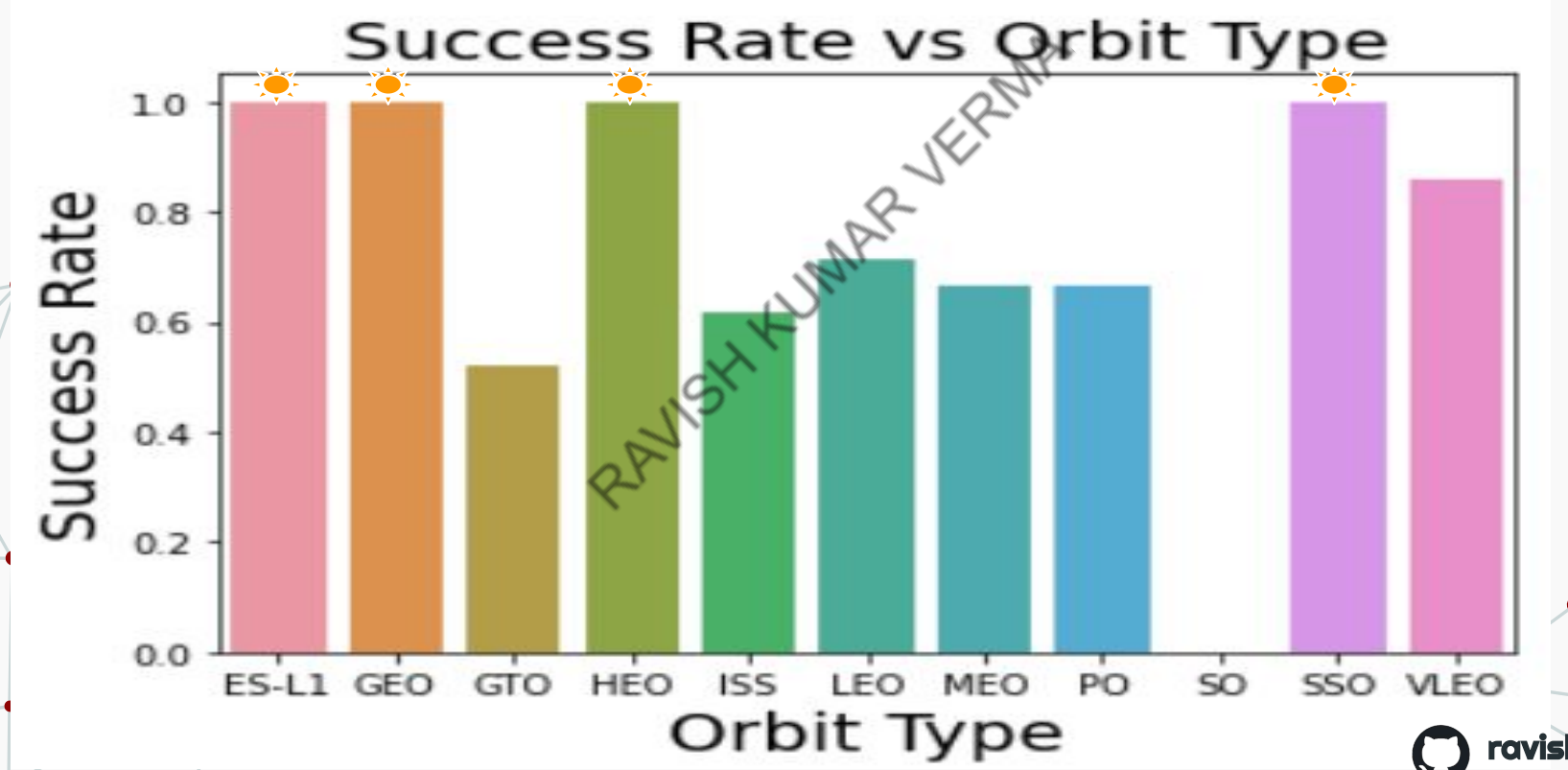
- The greater the payload mass (greater than 7000 KG) higher the success rate for the Rocket. But there is clear pattern to confirm, if the Launch Site is dependant on PayLoad Mass for a success launch



ravish1208

SUCCESS RATE vs ORBIT TYPE

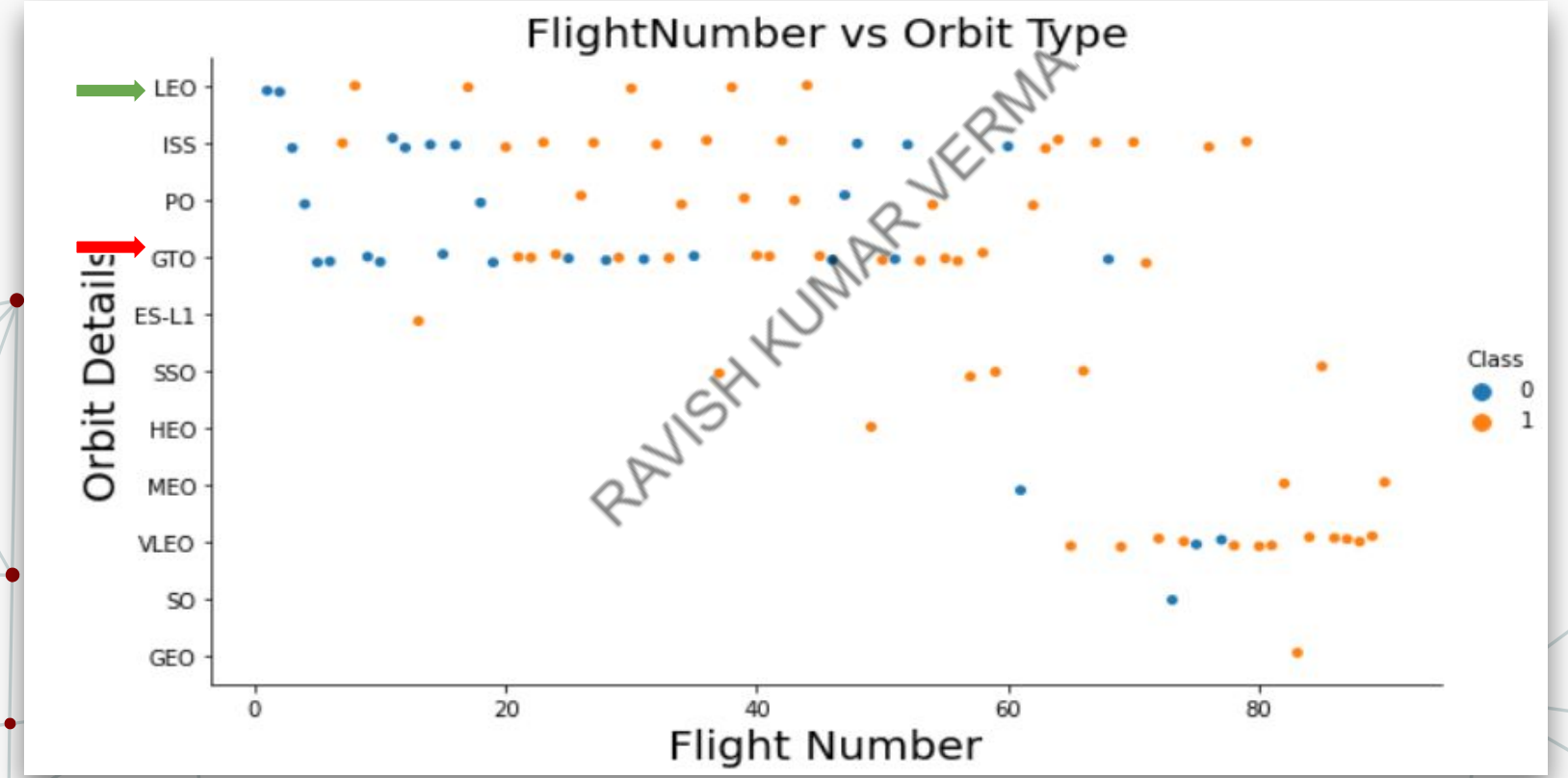
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate



FLIGHT NUMBER vs ORBIT TYPE

ravish1208

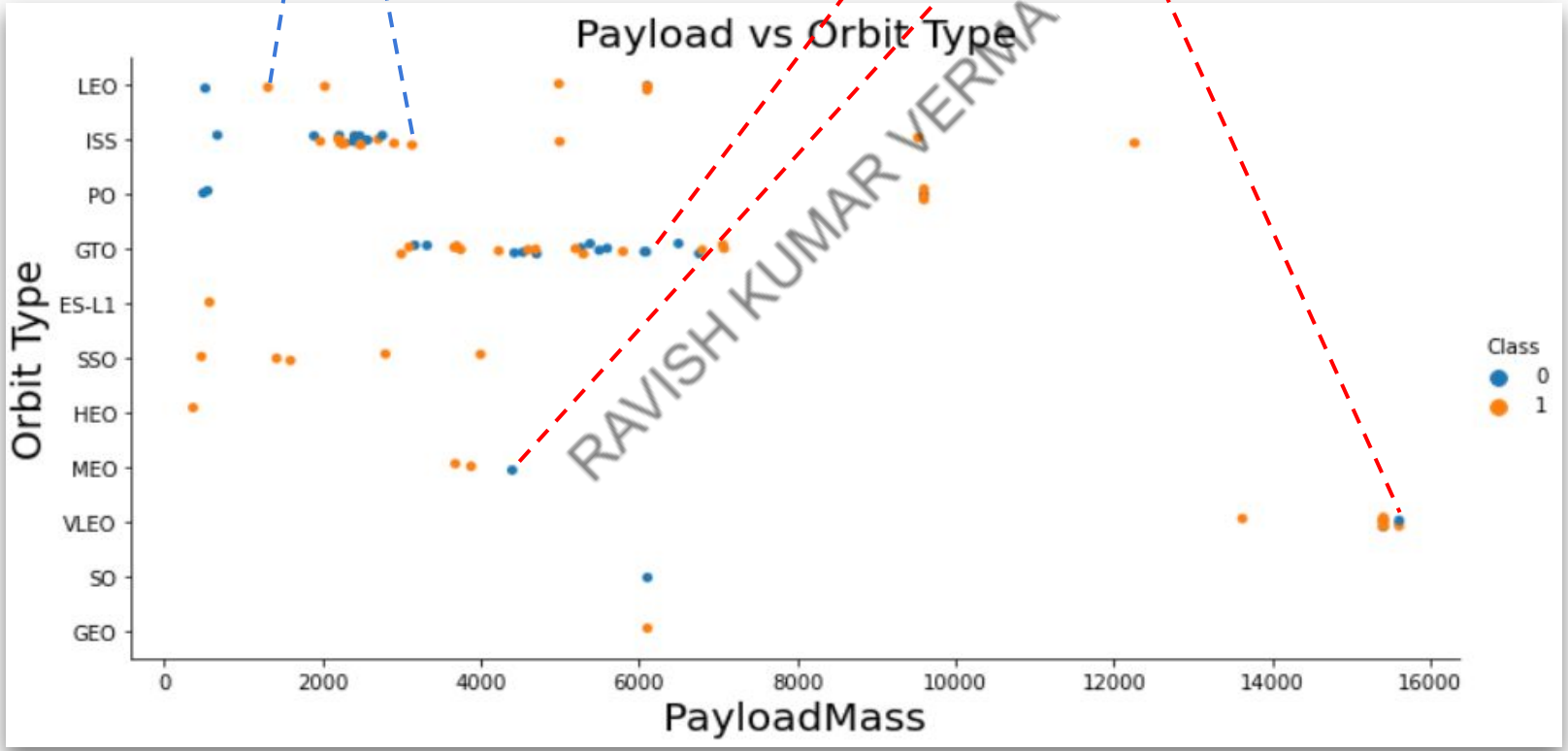
- It is observed that in the LEO orbit the Success appears related to the number of flights;
- On the other hand, there seems to be no relationship between flight number when in GTO orbit



PAYLOAD vs ORBIT TYPE

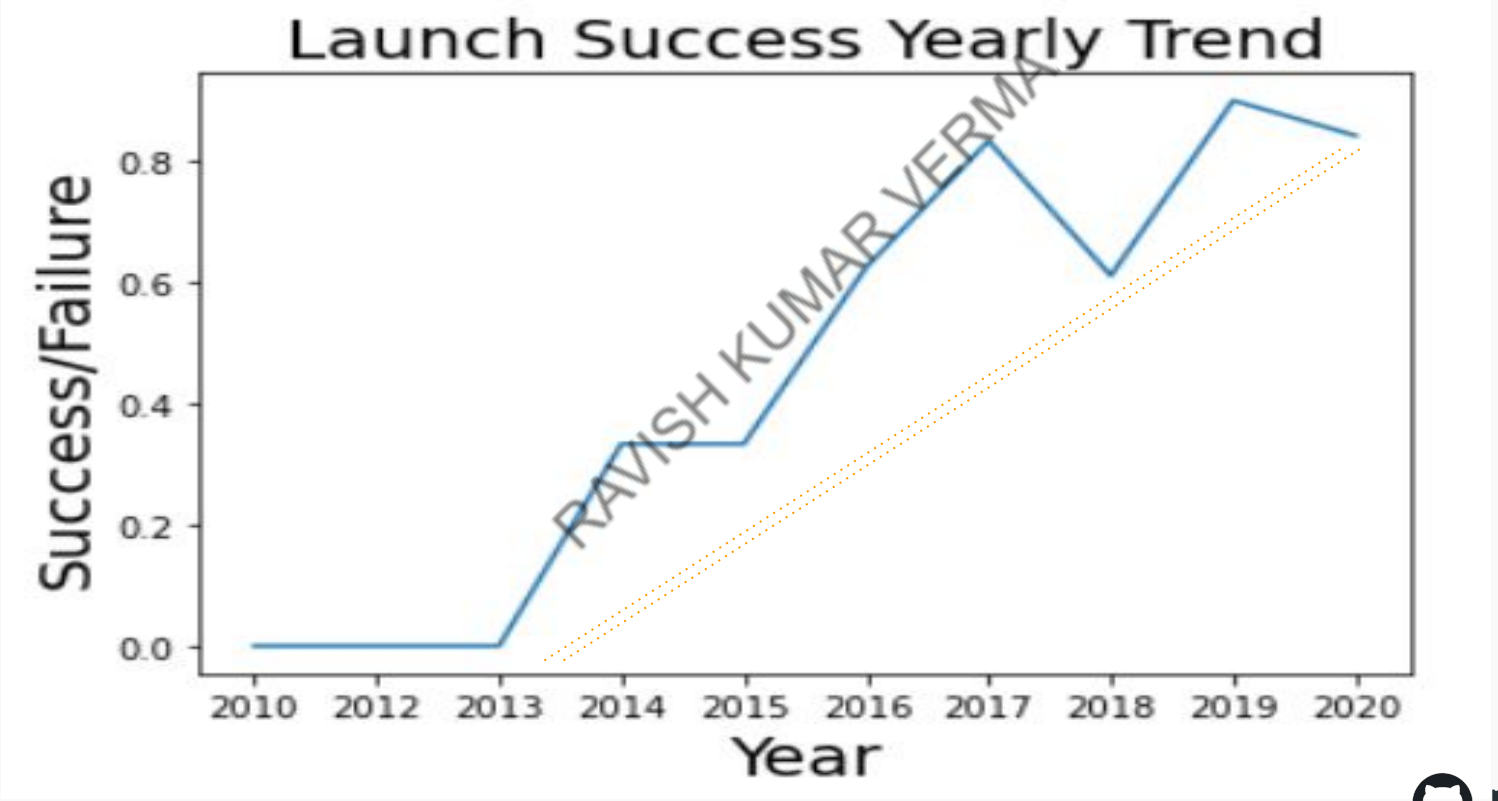
ravish1208

- It is observed that heavy payload have a negative impact on GTO, MEO, VLEO orbits
- Positive impact on LEO, ISS orbits



LAUNCH SUCCESS YEARLY TREND

- We can observe that Launch Success Rate is increasing since 2013, with a dip from 2017-2018 and 2019-2020



RESULTS

EXPLORATORY DATA SQL ANALYSIS

UNIQUE LAUNCH SITE NAMES

SQL Query

```
%sql select Unique(LAUNCH_SITE) from SPACEXTBL;
```

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

QUERY EXPLANATION

Using the word DISTINCT in the query means that it will only show Unique values in the Launch_Site column from SPACEXTBL

LAUNCH SITE NAMES THAT BEGINS WITH 'CCA'

SQL Query

```
%sql SELECT * from SPACEXTBL where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

QUERY EXPLANATION

Displaying 5 records where launch sites begin with the string 'CCA'. Using *'LIMIT 5'* in the query will display only top 5 records from *SPACEXTBL*. Using *'CCA%'* in query retrieves the record where *LAUNCH SITE* name begins with CCA.

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

TOTAL PAYLOAD MASS BY *CUSTOMER* 'NASA(CRS)'

SQL Query

```
%sql SELECT sum(payload_mass__kg_) as sum_payload from SPACEXTBL where (customer) = 'NASA (CRS)'
```

QUERY EXPLANATION

- Using the function SUM summates the total in the column PAYLOAD_MASS_KG_
- The WHERE clause filters the dataset to only perform calculations on Customer NASA (CRS)

sum_payload
45596

Average Payload Mass carried by booster version F9 v1.1

SQL Query

```
%%sql
select BOOSTER_VERSION, avg(PAYLOAD_MASS__KG_) as avgpayloadmass_F9
from SPACEXTBL
where BOOSTER_VERSION = 'F9 v1.1'
group by BOOSTER_VERSION;
```


QUERY EXPLANATION

- Using the function ***AVG*** works out the average in the column ***PAYLOAD_MASS_KG_***
- The ***WHERE*** clause filters the dataset to only perform calculations on ***Booster_version F9 v1.1***

booster_version	avgpayloadmass_f9
F9 v1.1	2928

The date where the successful landing outcome in GROUND PAD was achieved

SQL Query

```
%sql SELECT MIN(Date) from SPACEXTBL where Landing_Outcome = "Success (ground pad)"
```


QUERY EXPLANATION

- Using the function ***MIN*** works out the minimum date in the column Date
- The ***WHERE*** clause filters the dataset to only perform calculations on ***Landing_Outcome Success (drone ship)***

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

Successful drone ship landing with payload between 4000 and 6000

SQL Query

```
%%sql
select BOOSTER_VERSION from SPACEXTBL
where LANDING_OUTCOME ='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000
```


QUERY EXPLANATION

- Selecting only *Booster_Version*
- The *WHERE* clause filters the dataset to *Landing_Outcome = Success (drone ship)*
-
- The AND clause specifies additional filter conditions *Payload_MASS_KG_ BETWEEN 4000 AND 6000*

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

Total Number of Successful and Failure Mission Outcomes

SQL Query

```
%%sql
SELECT(SELECT Count(Mission_Outcome) from SPACEXTBL where Mission_Outcome
LIKE '%Success%') as Successful Mission,
(SELECT Count(Mission_Outcome) from tblSpaceX where Mission_Outcome
LIKE '%Failure%') as Failure Mission
```


QUERY EXPLANATION

- We used subqueries here to produce the results. The LIKE '%Success%' & '%Failure%' wildcard shows that in the record the Success and Failure phrase is in any part of the string in the records

- for example.
  PHRASE "(Drone Ship was a Success)" LIKE '%Success%' Word 'Success' is in the phrase the filter will include it in the dataset

Successful Mission	Failure Mission
100	1

Boosters carried maximum payload

ravish1208

- SQL Query

```
%%sql
SELECT BOOSTER_VERSION FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

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

QUERY EXPLANATION

- Using the function MAX works out the maximum payload in the column PAYLOAD_MASS_KG_ subquery.
- WHERE clause filters Booster Version which had that maximum payload.

List the failed landing_outcomes in drone ship- year 2015

SQL Query

```
%sql SELECT DATE, BOOSTER_VERSION, LAUNCH_SITE, LANDING__OUTCOME FROM SPACEXTBL
WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND YEAR(DATE) = 2015
```


QUERY EXPLANATION

- We need to list the records where Landing_outcome was a 'Failure (drone ship)
- Via *year(Date) = '2015'* function and WHERE clause on 'Failure (drone ship)' fetches our required values.

	boosterversion	launchsite	landingoutcome
0	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
1	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

Rank of Landing Outcomes between 2010-06-04 and 2017-03-20

SQL Query

```
%%sql
SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY LANDING__OUTCOME ORDER BY COUNT(LANDING__OUTCOME) DESC;
```

QUERY EXPLANATION

- Selecting LANDING_OUTCOME applying WHERE clause on DATE - *2010-06-04 & 2017-03-20*
- Displaying records GROUP BY Landing Outcomes and ordering records in DESCENDING as per COUNT of LANDING_OUTCME

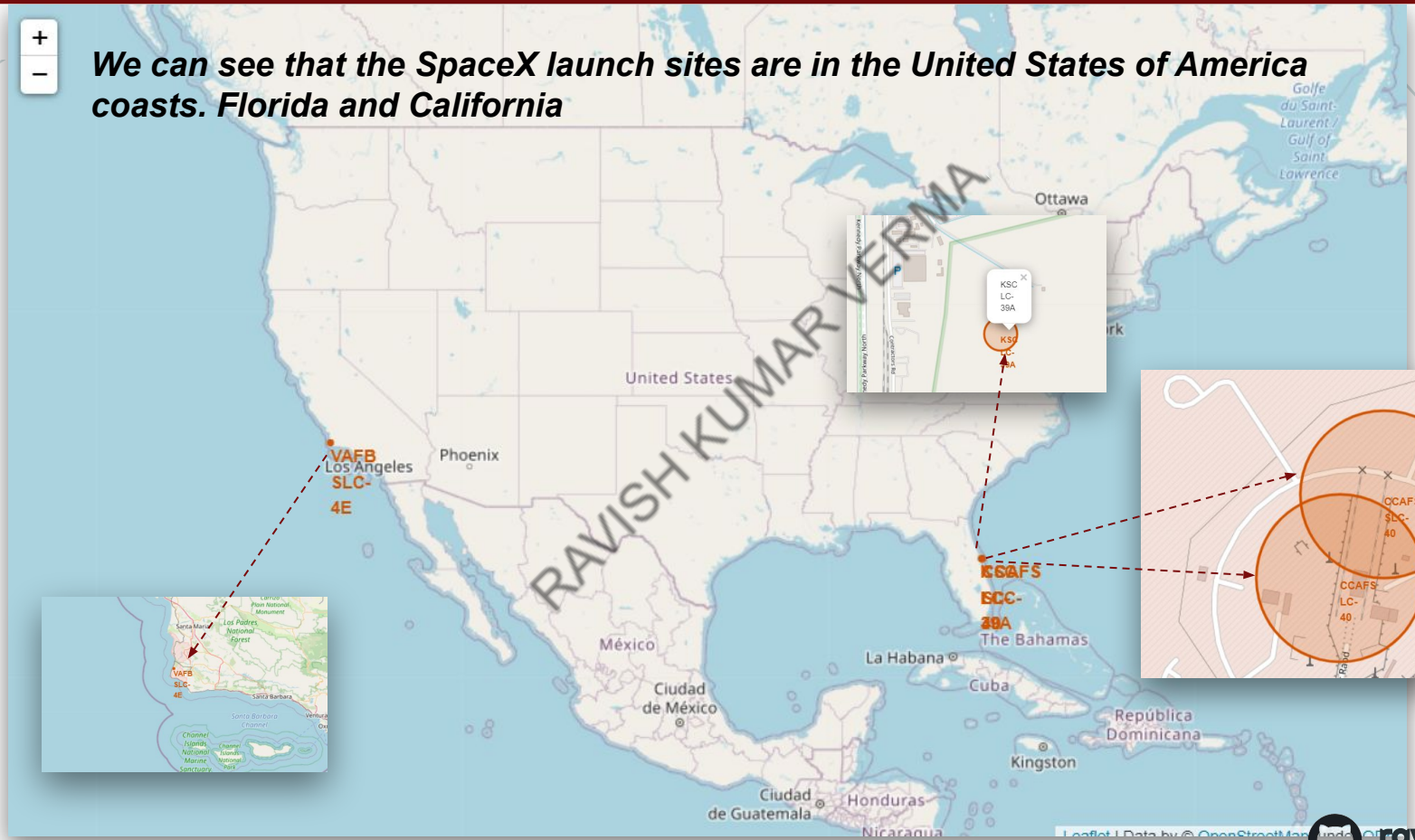
	landingoutcome	count
0	No attempt	10
1	Success (drone ship)	6
2	Failure (drone ship)	5
3	Success (ground pad)	5
4	Controlled (ocean)	3
5	Uncontrolled (ocean)	2
6	Precluded (drone ship)	1
7	Failure (parachute)	1

RESULTS / FINDINGS

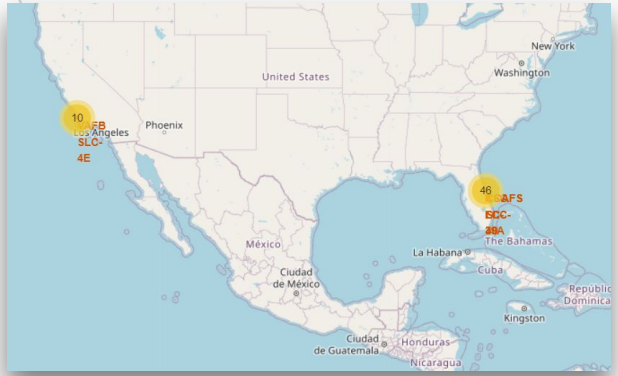
Interactive Maps Using Folium [Python]

All launch sites global map markers

We can see that the SpaceX launch sites are in the United States of America coasts. Florida and California



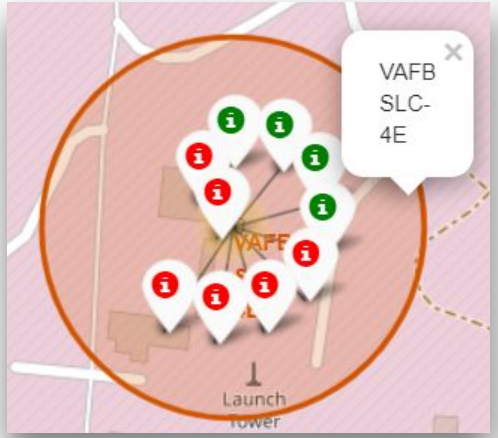
Launch Site locations - color coded Fail and Success launches

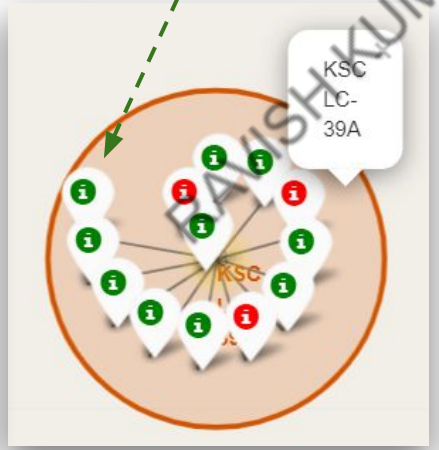


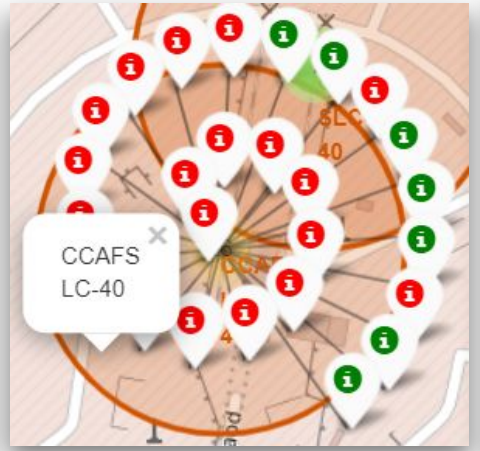
Green Marker Shows successful launches and **Red Marker** shows failed launches.

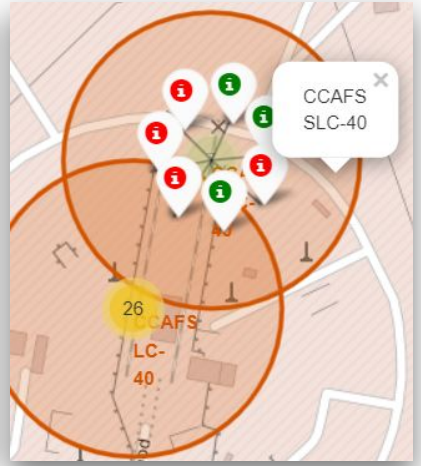
Finding:

KSC LC-39A has the maximum probability of launch success





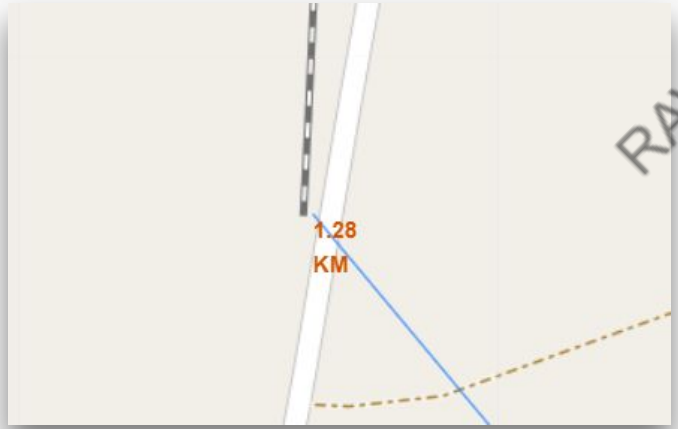




Launch Sites distance to closest landmarks: Coast & Railway

(CCAFS-SLC-40 as a reference)

Closest Coast : 0.86 KM

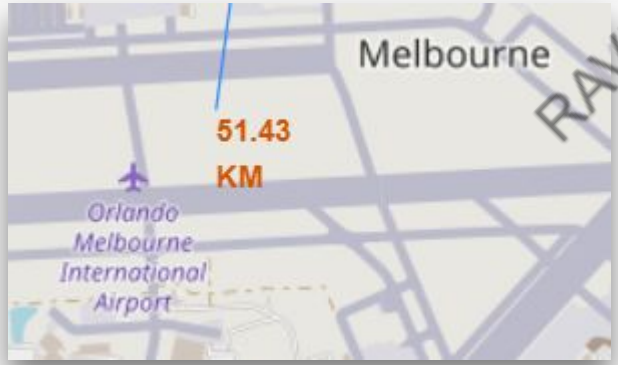


Closest Railway: 1.28 KM

Launch Sites distance to closest landmarks: Highway & City

(CCAFS-SLC-40 as a reference)

Closest Highway : 0.58 KM



Closest City : 51.43 KM

Launch Sites distance : FINDINGS

Closest Coast
0.86 KM

Launch sites are in close proximity to coastline so they can fly over the ocean during launch, for at least two safety reasons-

a. Launch Team will have an option to abort mission and attempt landing rockets in ocean.
b. To minimize risk to people and property from explosion of rocket.

Closest Highway
0.58 KM

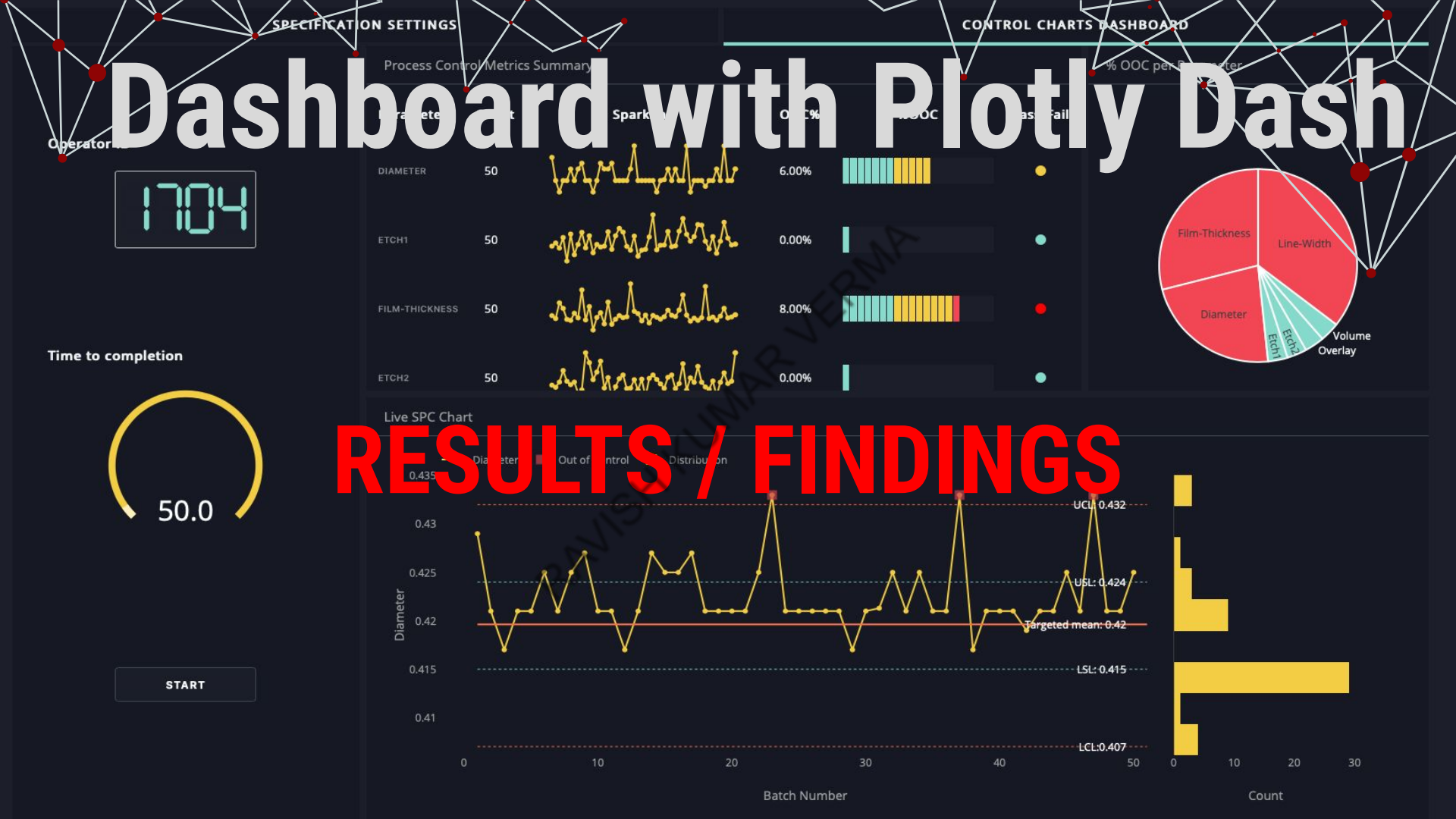
Launch sites are in close proximity to highways, which allows easy transportation of manpower, workforce and other necessary items.

Closest Railway
1.28 KM

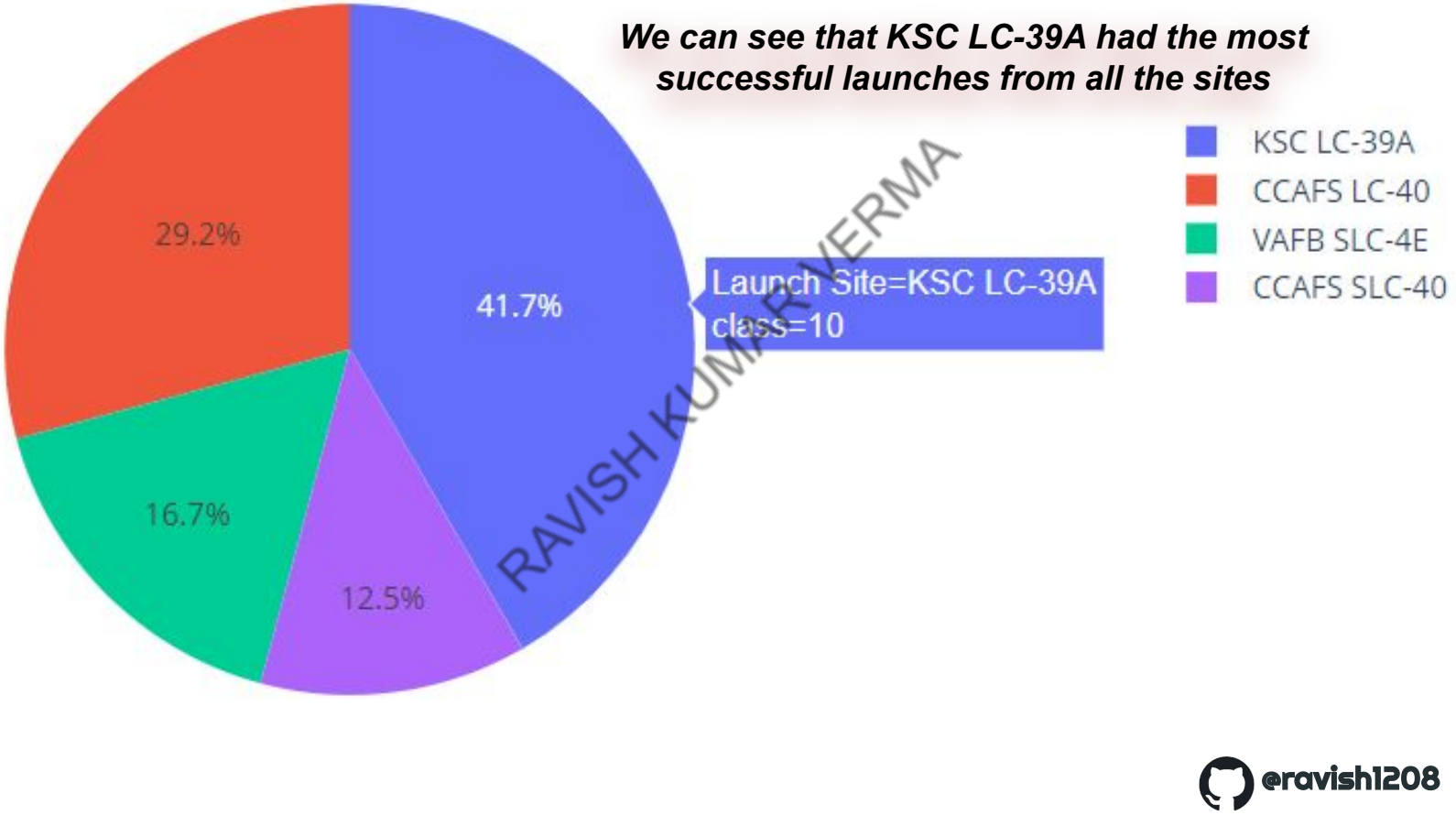
Launch sites are in close proximity to railways, which allows convenient transportation of heavy equipments and other cargo.

Closest City
51.43 KM

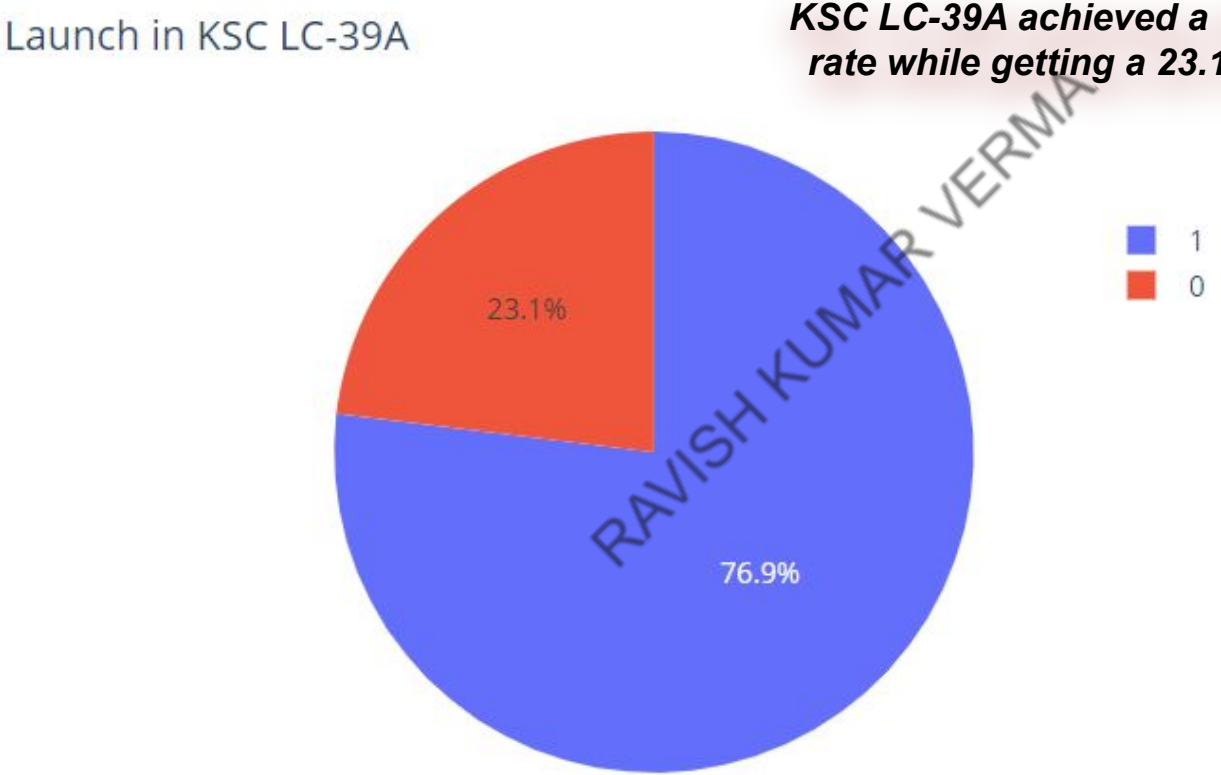
Launch sites are not in close proximity to cities, which minimizes danger to Human Life and property dense areas.


SPECIFICATION SETTINGS
CONTROL CHARTS DASHBOARD
Process Control Metrics Summary
Dashboard with Plotly Dash
Operator
1704
DIAMETER 50 6.00%
ETCH1 50 0.00%
FILM-THICKNESS 50 8.00%
ETCH2 50 0.00%
Film-Thickness
Line-Width
Diameter
Etch2
Etch1
Volume Overlay
Time to completion
50.0
START
Live SPC Chart
RESULTS / FINDINGS
UCL: 0.432
USL: 0.424
Targeted mean: 0.42
LSL: 0.415
LCL:0.407
Diameter
Batch Number
Count

Launch Success percentage of all Sites



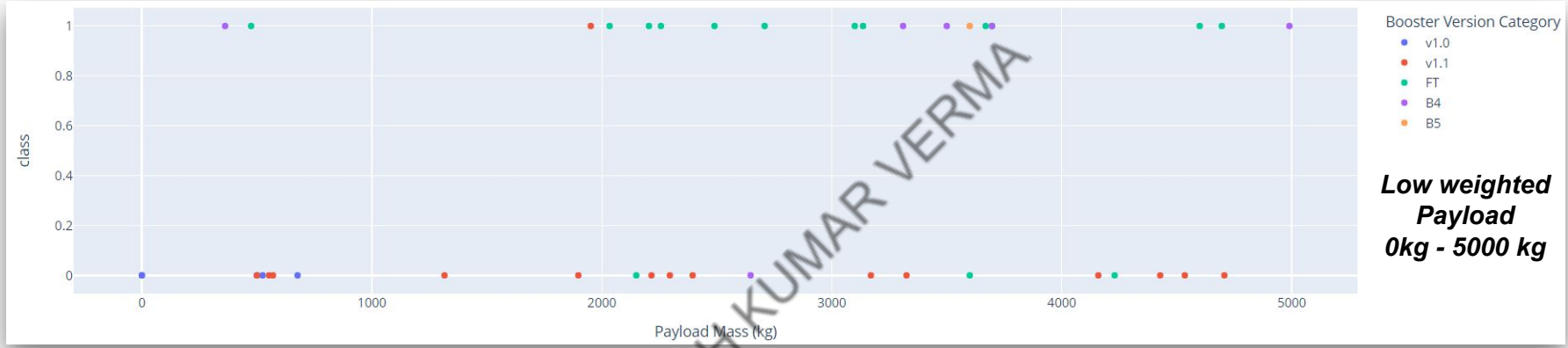
Launch site with highest launch success ratio



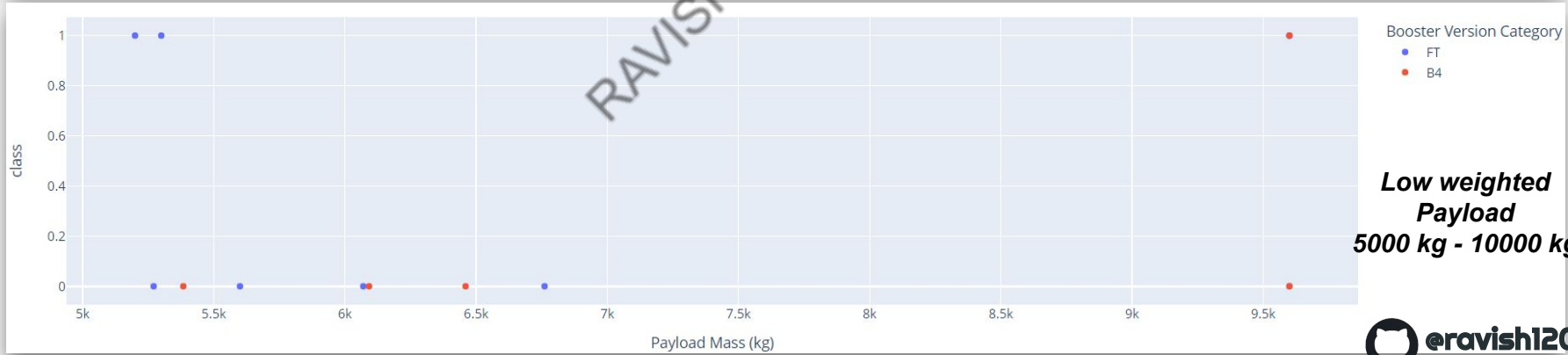
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs. Launch Outcome scatter plot for all sites

We can see the success rates for low weighted payloads is higher than the heavy weighted payloads



Low weighted Payload 0kg - 5000 kg



Low weighted Payload 5000 kg - 10000 kg


PREDICTIVE ANALYSIS
(CLASSIFICATION)
RESULTS / FINDINGS

Confusion Matrix

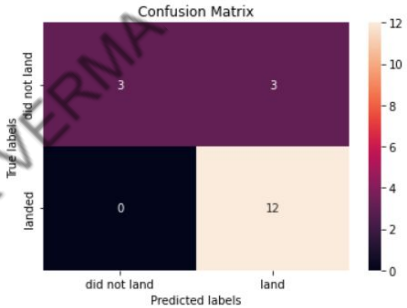
@ravish1208

As we can see, for all models we have same confusion matrix

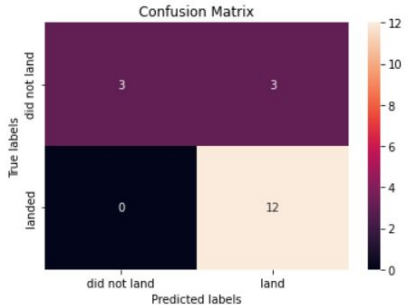
Accuracy: (TP+TN)/total = (12+3)/18 = 0.83333
Misclassification Rate: (FP+FN)/total = (3+0)/18 = 0.1667
True Positive Rate: TP/Actual yes = 12/12 = 1
False Positive Rate: FP/actual no = 3/6 = 0.5
True Negative Rate: TN/actual no = 3/6 = 0.5
Precision: TP/predicted yes = 12/15 = 0.8
Prevalence: Actual yes/total = 12/18 = 0.6667

	Predicted Negative	Predicted Positive	
Actual Negative	True *Negative* ✓ **3**	False Positive ✗ **3**	6
Actual Positive	False *Negative* ✗ **0**	True Positive ✓ **12**	12
	3	15	**Total Cases 18**

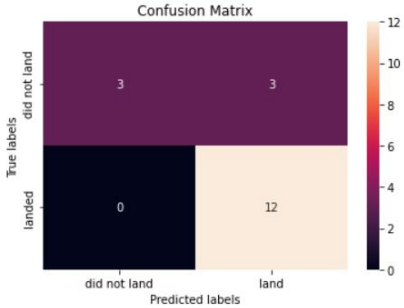
Logistic Regression



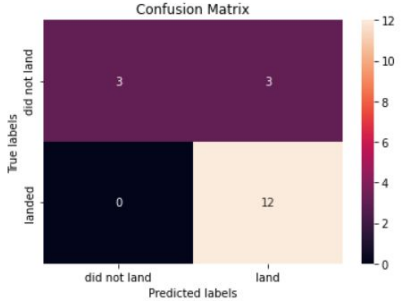
SVM



Decision Tree



KNN



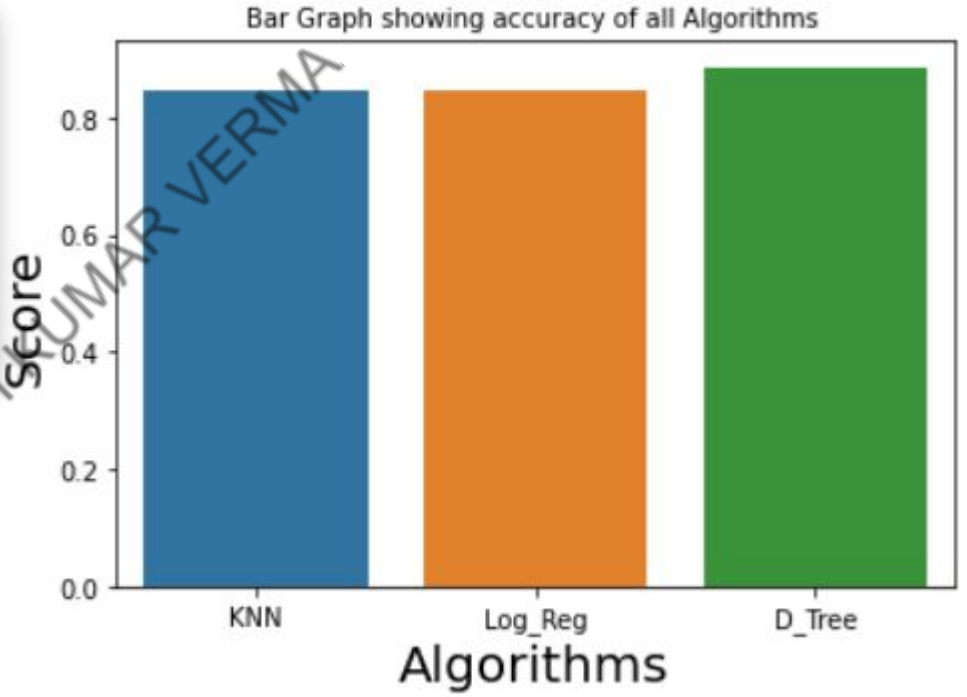
Classification Accuracy

@ravish1208

As you can see our accuracy is extremely close but we do have a winner! using this function

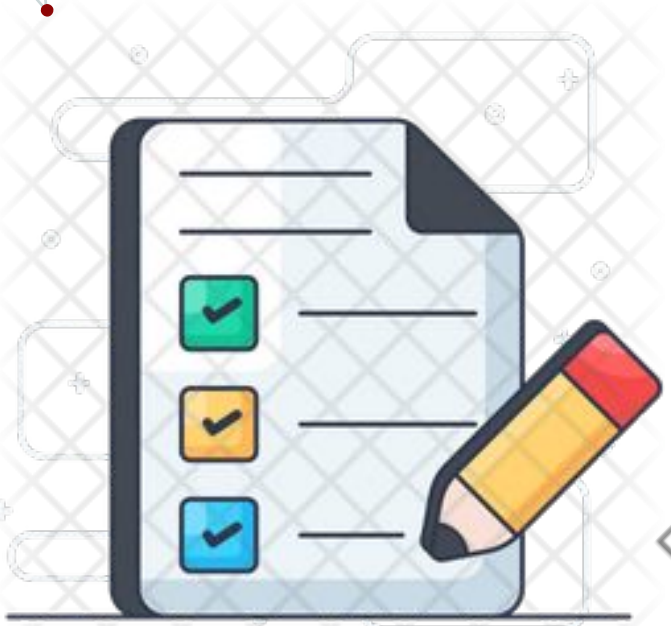
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorith
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)

Accuracy = pd.DataFrame(algorithms,index=[0])
print("Best Accuracy Score is:")
Accuracy
```

	Algorithm	Accuracy_Score
0	KNN	0.848214
1	Log_Reg	0.846429
2	D_Tree	0.885714




```
Best Algorithm is Tree with a score of 0.8857142857142858
```

Classification Accuracy



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

@ravish1208

THANKS

@ravish1208