IBM Developer
SKILLS NETWORK

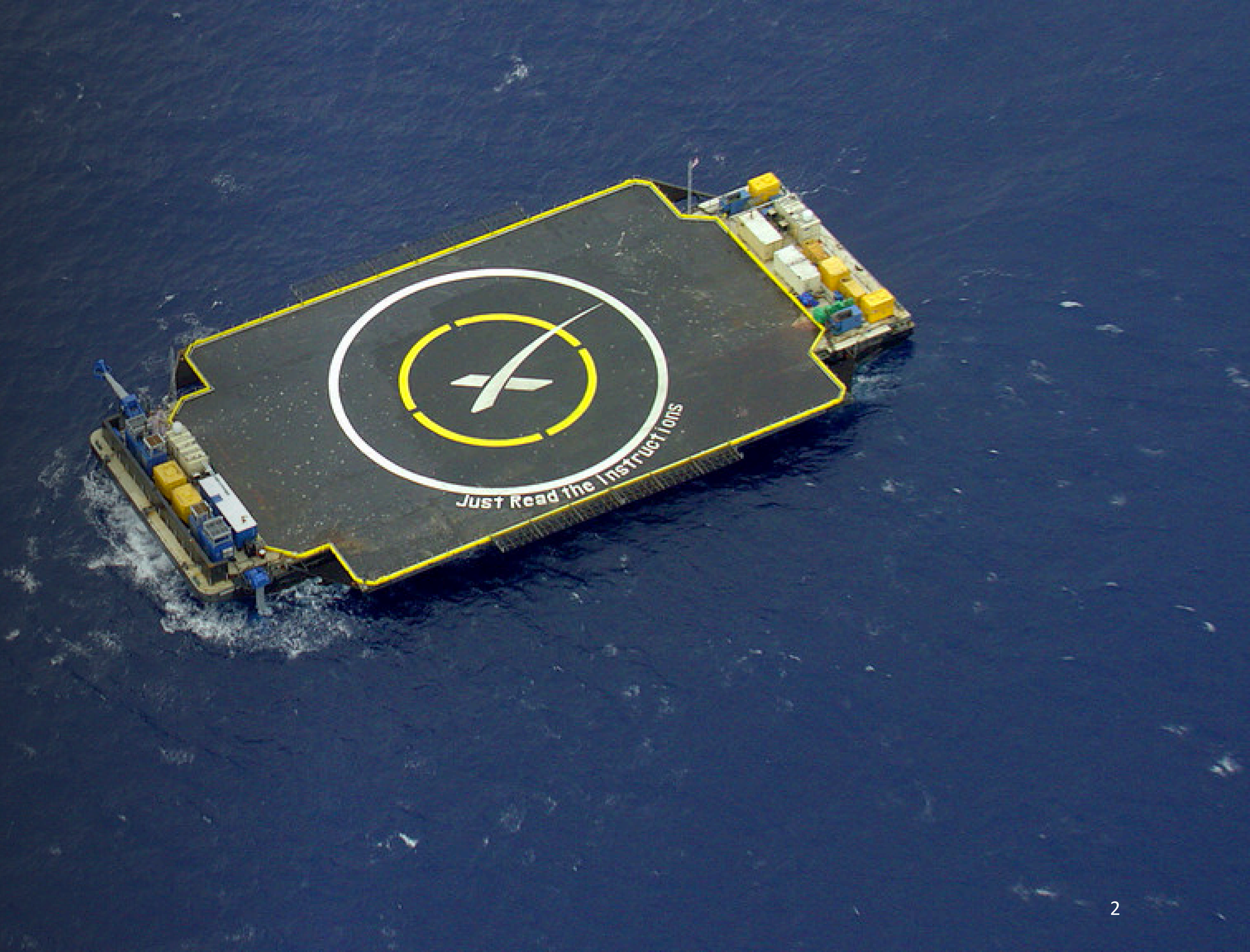
Winning Space Race with Data Science

Felipe Maia

06-Apr-2022

Outline

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



Executive Summary

The following methodologies were used to analyze data:

- Data Collection using Web Scraping from Wikipedia, SpaceX REST API and SQL;
- Exploratory Data Analysis (EDA), including data wrangling, data visualization and interactive visual analytics like Dash, Folium and Seaborn Plots ;
- Machine Learning Prediction.

Summary of all results

- EDA allowed to identify which features are the best to predict success of launchings.
- Interactive Analysis
- Best Model for Predictive Analysis

Introduction

Project background and context:

The commercial space age is here, companies are making space travel affordable for everyone. There are major companies in segment, like Blue Origin, Virgin Galactic, and Space X. The current leader in this race seems to be the last one, and the reason behind that **is the reusability of their first stage**. Which reduces the cost of launch from a minimum of 165 million to around 62 million per launch.

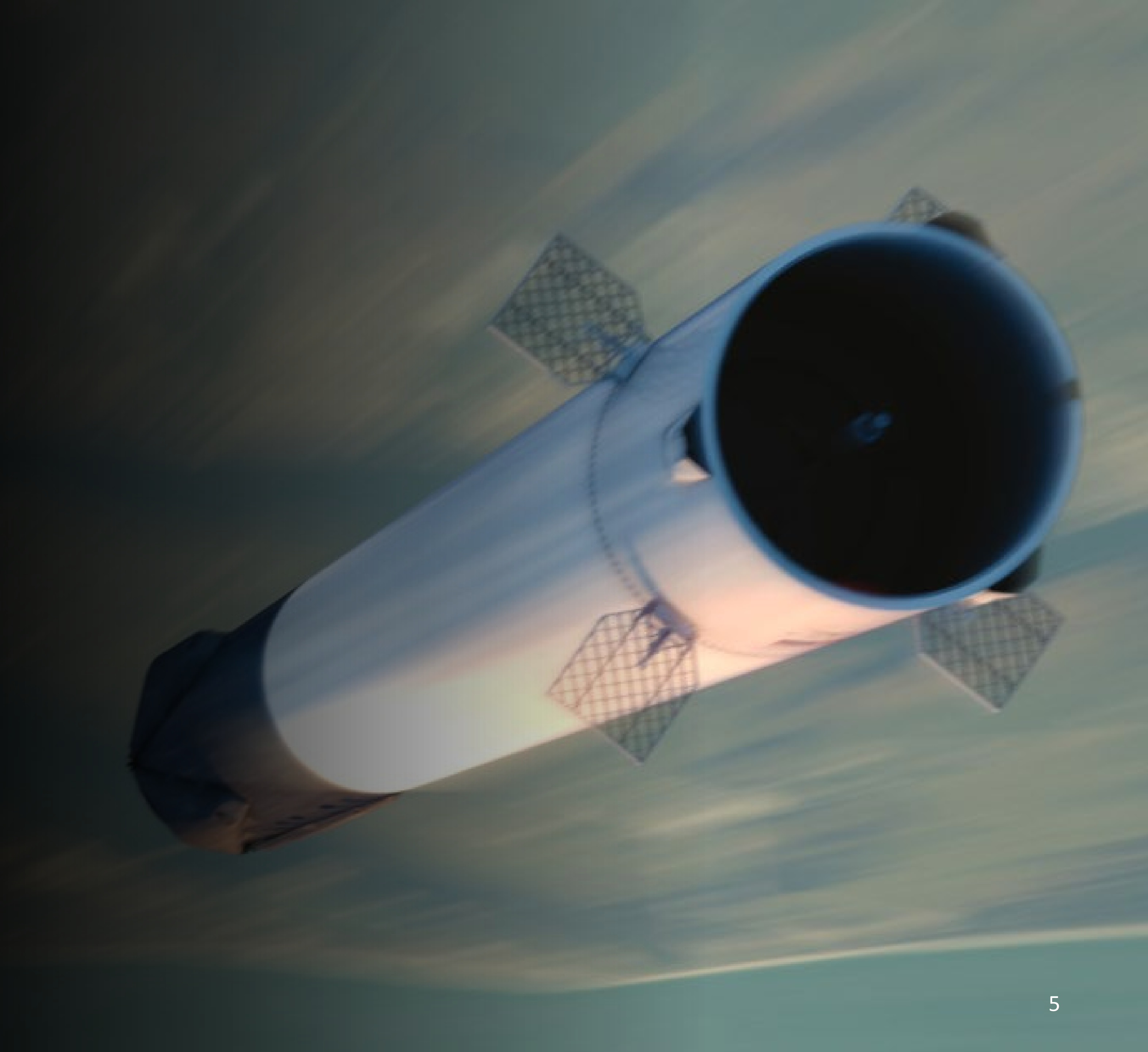
The objective of this Project is to evaluate the viability of a new company to compete with Space X. So, we need to find the best way to estimate the total cost for launches, by predicting successful landings of the first stage of rockets and where is the best place to make launches.

So, the questions are:

- What factors determine if the rocket will land successfully?
- What is the interaction amongst various features that determine the rate of a successful landing?
- Conditions which will aid SpaceX have to achieve the best results.

Section 1

Methodology



Methodology 1/2

Executive Summary

- Data collection methodology:
  - SpaceX Rest API
  - Web Scrapping from Wikipedia

- Perform data wrangling
  - Data cleaning of null values and irrelevant columns. One Hot Encoding for ML.
  - The Landing Pad column will retain None values to represent when landing pads were not used.
  - The mean and the replace() function was used to replace np.nan values in the data with the mean calculated.

Methodology 2/2

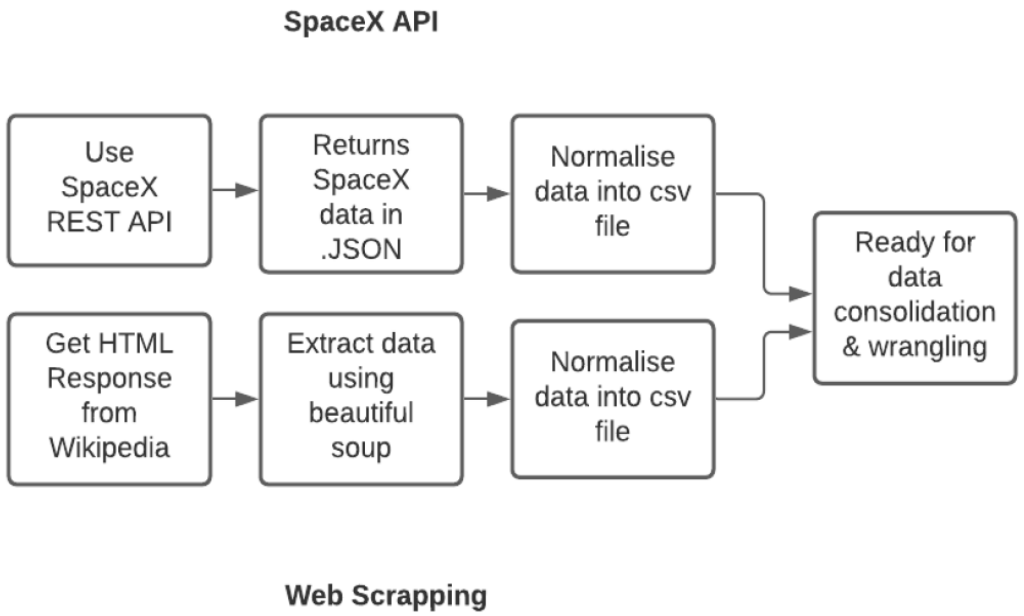
Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
- Data was normalized, splitted in Training and Test data sets, performed a GridSearchCV to find the best parameters and evaluated by four different classification models.
- The models used was LR, KNN, SVM and Decision Tree to find the best classifier and the best accuracy.

Data Collection

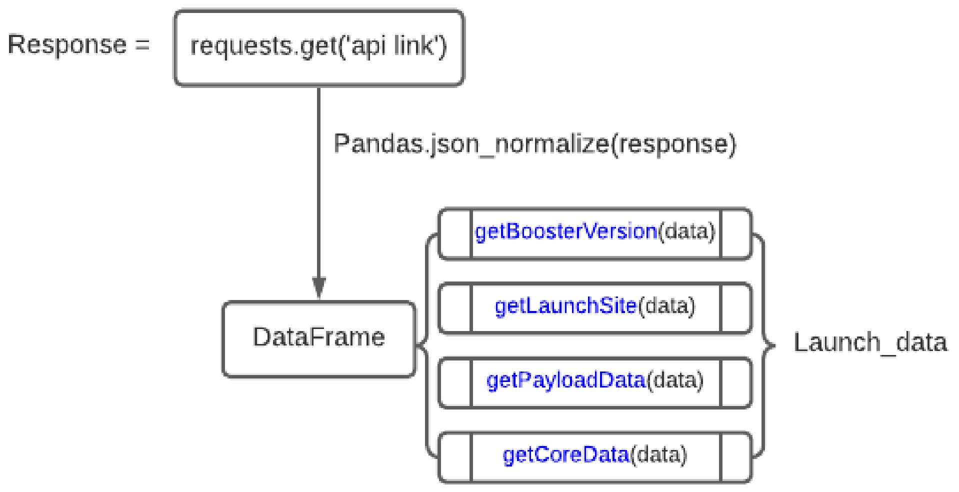
The following datasets was collected:

- SpaceX Launch Data was obtained from the SpaceX REST API.
- This API gave us data about the Launches, including information about the Rocket used, Payloads, Orbit, Core, launch specifications, landing specifications, and landing outcome.
- The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/
- And data collected by Web Scraping method from Wikipedia using BeautifulSoup.



Data Collection – SpaceX API

- Data collection with SpaceX REST calls using key phrases:
- Request and parse the SpaceX launch data using the GET request.
- Response content as a Json using .json() and turn it into a Pandas dataframe using json_normalize()
- Using Dataframe into the created functions.



[GitHub - Lab 1: Collecting the data]

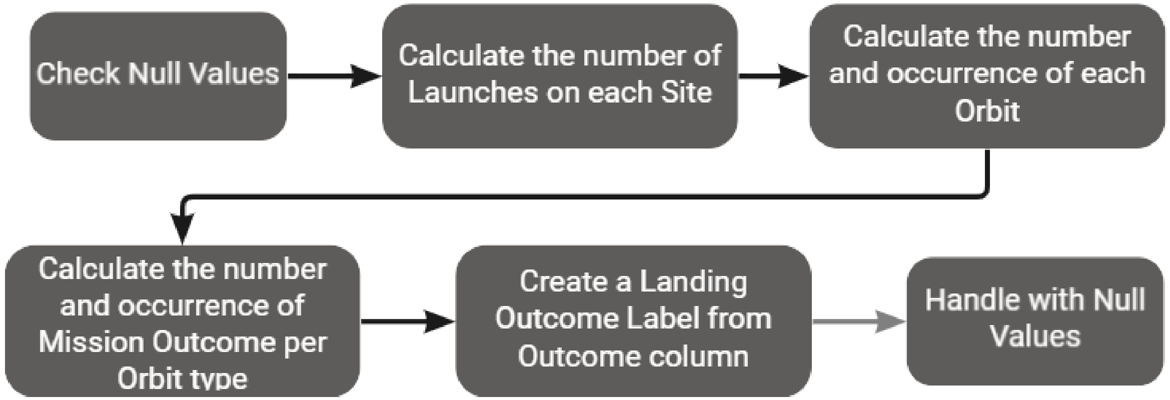
Data Collection – Web Scraping Method

Getting Response from HTML
↓
Creating Beautiful Soup Object
↓
Finding Tables
↓
Getting Column Names
↓
Creation of Dictionary and Appending Data to Keys
↓
Converting the Dictionary to Dataframe
↓
Dataframe to CSV

```
!pip3 install beautifulsoup4
!pip3 install requests

import sys
import requests
from bs4 import BeautifulSoup
import re
import unicodedata
import pandas as pd

soup = BeautifulSoup(response.text, 'html')

html_tables = soup.find_all('table')

column_names = []

launch_dict= dict.fromkeys(column_names)

df=pd.DataFrame(launch_dict)
df

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

[GitHub - Lab 2: Web Scraping]

Data Wrangling



Initially, the data were filtered to contain **only Falcon 9 records**, then it was found that there were some **null values** in the dataset. The **Landing Pad column will retain None** values to represent when landing pads were not used. Null values in the column PayloadMass **was replaced by the mean value**. And **One hot encoding** was done for some categorical variables in order to feed them to the ML algorithm after performing feature scaling.

GitHub - Data Wrangling Lab

EDA with Data Visualization

Scatter Graphs

Payload and Flight Number
Flight Number and Launch Site
Payload and Launch Site
Flight Number and Orbit Type
Payload and Orbit Type

A scatter plot uses dots to represent values for two different numeric variables. The position of each dot on the horizontal and vertical axis indicates values for an individual data point. Scatter plots are used to observe relationships between variables.

Line Graph

Launch Success Yearly Trend

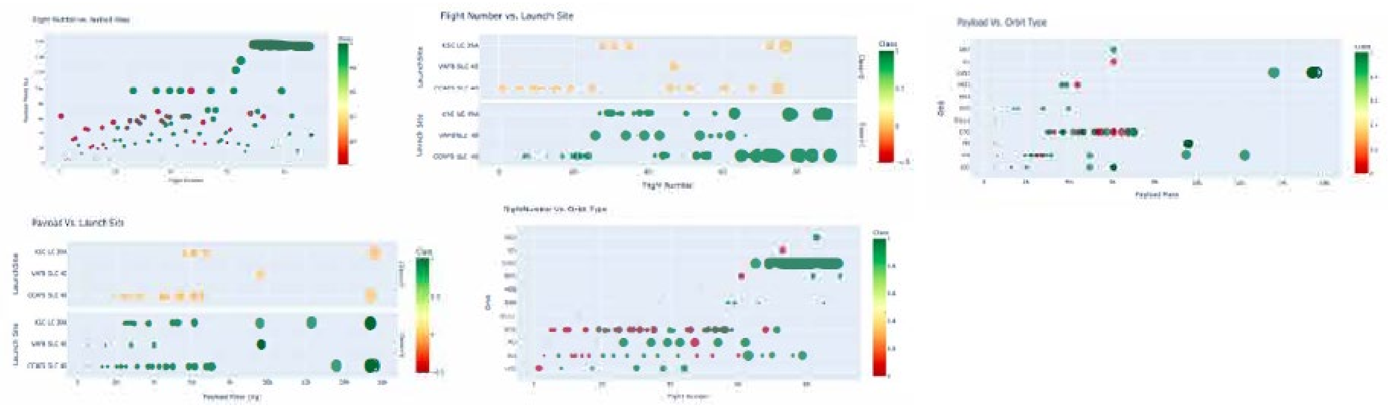
A line chart is a way of plotting data points on a line. Often, it is used to show trend data, or the comparison of two data sets.

Bar Graph

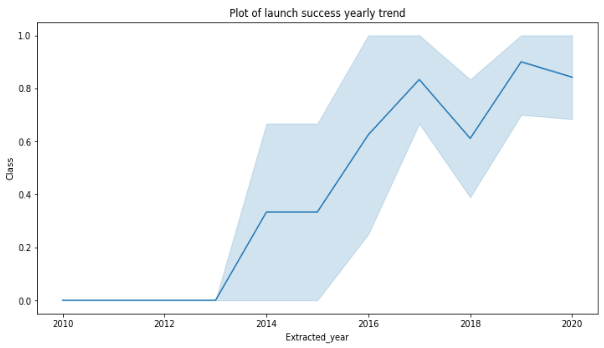
Success Rate VS. Orbit Type

A bar chart provides a way of showing data values represented as vertical bars. It is sometimes used to show trend data, and the comparison of multiple data sets side by side.

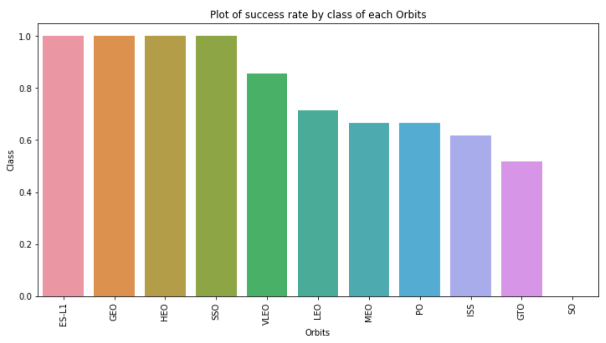
Scatter



Line



Bar



[GitHub - Visualization Lab]

EDA with SQL

SQL queries to get the Data:

1. Display the names of the unique launch sites in the space mission.
2. Display 5 records where launch sites begin with the string 'CCA’
3. Display the total payload mass carried by boosters launched by NASA (CRS)
4. Display average payload mass carried by booster version F9 v1.1
5. List the date when the first successful landing outcome in ground pad was achieved.
6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7. List the total number of successful and failure mission outcomes
8. List the names of the booster_versions which have carried the maximum payload mass.
9. List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
10. Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

[GitHub - SQL Lab]()

Build an Interactive Map with Folium

Folium makes it easy to visualize data that's been manipulated in Python on an interactive leaflet map. It enables both the binding of data to a map for choropleth visualizations as well as passing rich vector/raster/HTML visualizations as markers on the map.

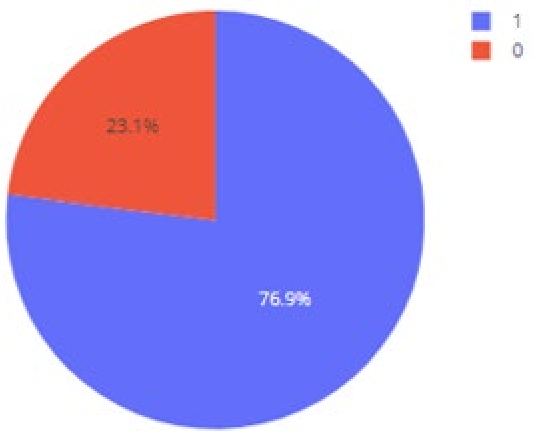
We use the latitude and longitude coordinates for each launch site and added a circle marker with a label around each launch site.

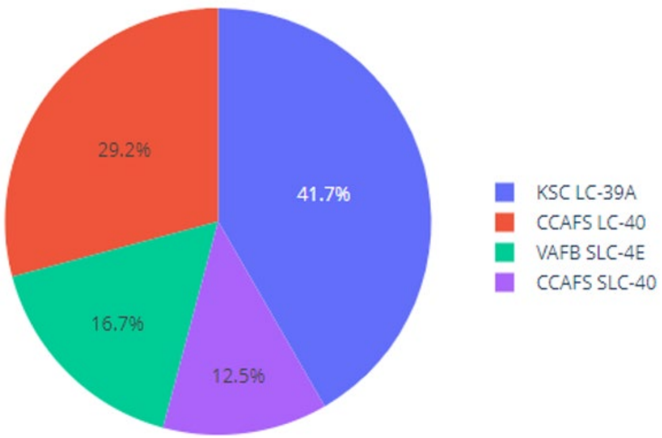
Map Objects	Code	Result
Map Marker	folium.Marker()	Map object to make a mark on map.
Icon Marker	folium.Icon()	Create an icon on map.
Circle Marker	folium.Circle()	Create a circle where Marker is being placed.
Poly Line	folium.PolyLine()	Create a line between points.
Marker Cluster Object	folium.MarkerCluster()	Good way to simplify a map containing many markers having the same coordinate.
AntPath	folium.plugins.AntPath()	Create an animated line between points.

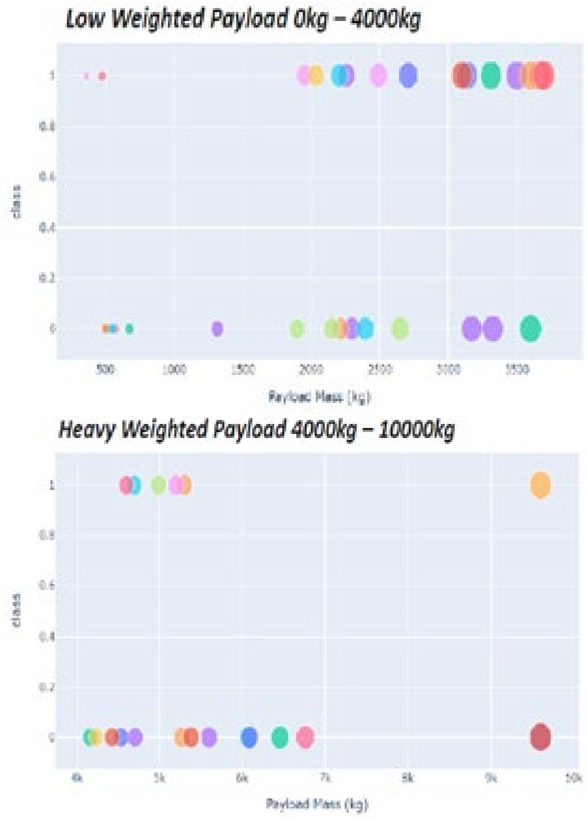
GitHub - FOLIUM Lab

Build a Dashboard with Plotly Dash

Dash give a point-&-click interface to models written in Python, vastly expanding the notion of what's possible in a traditional "dashboard." With Dash we can cross data and visualize instantly and find the success of landings.







KSC LC-39A achieved a 76.9% Success Rate (1) while getting a 23.1% Failure Rate (0).

[GitHub - DASH PLOTLY Lab]

Predictive Analysis (Classification)

Building Model

1) Load our feature engineered data into Dataframe
2) Transform it into NumPy arrays
3) Standardize and transform data
4) Split data into training and test data sets
5) Check how many test samples has been created
6) List down machine learning algorithms we want to use
7) Set our parameters and algorithms to GridSearchCV
8) Fit our datasets into the GridSearchCV objects and train our model

Train and Test

```
Y = data["Class"].to_numpy()
transform = preprocessing.StandardScaler()
X = transform.fit_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
```

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

Evaluating Model

Check accuracy for each model
Get best hyper parameters for each type of algorithms
Plot Confusion Matrix

Find the Best Score

```
logreg_cv.score(X_test, Y_test)
svm_cv.score(X_test, Y_test)
tree_cv.score(X_test, Y_test)
knn_cv.score(X_test, Y_test)
```

Finding Best Performing Classification Model

The model with best accuracy score wins the best performing model

Best Model

[GitHub - ML Lab]

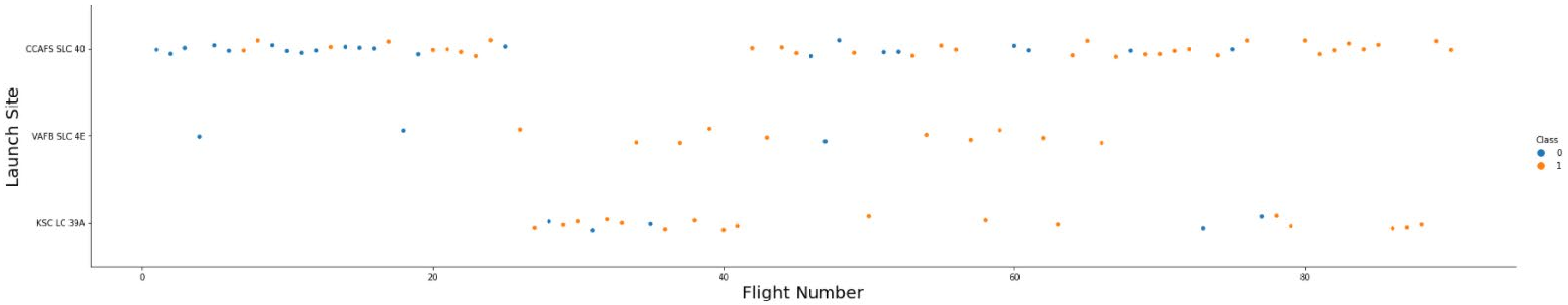
Results

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

Section 2

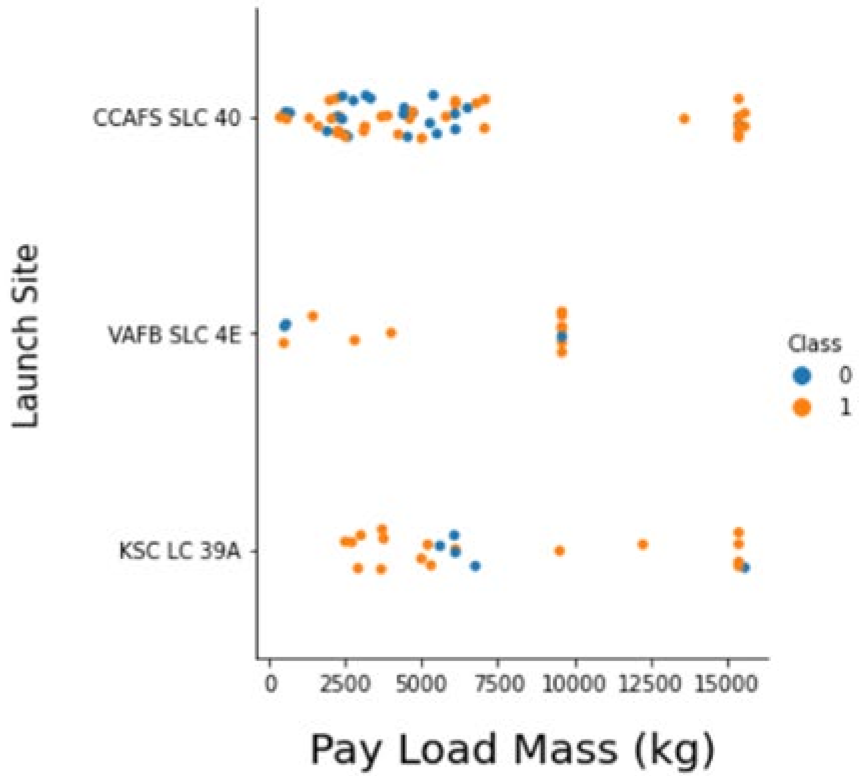
Insights drawn from EDA

Flight Number vs. Launch Site



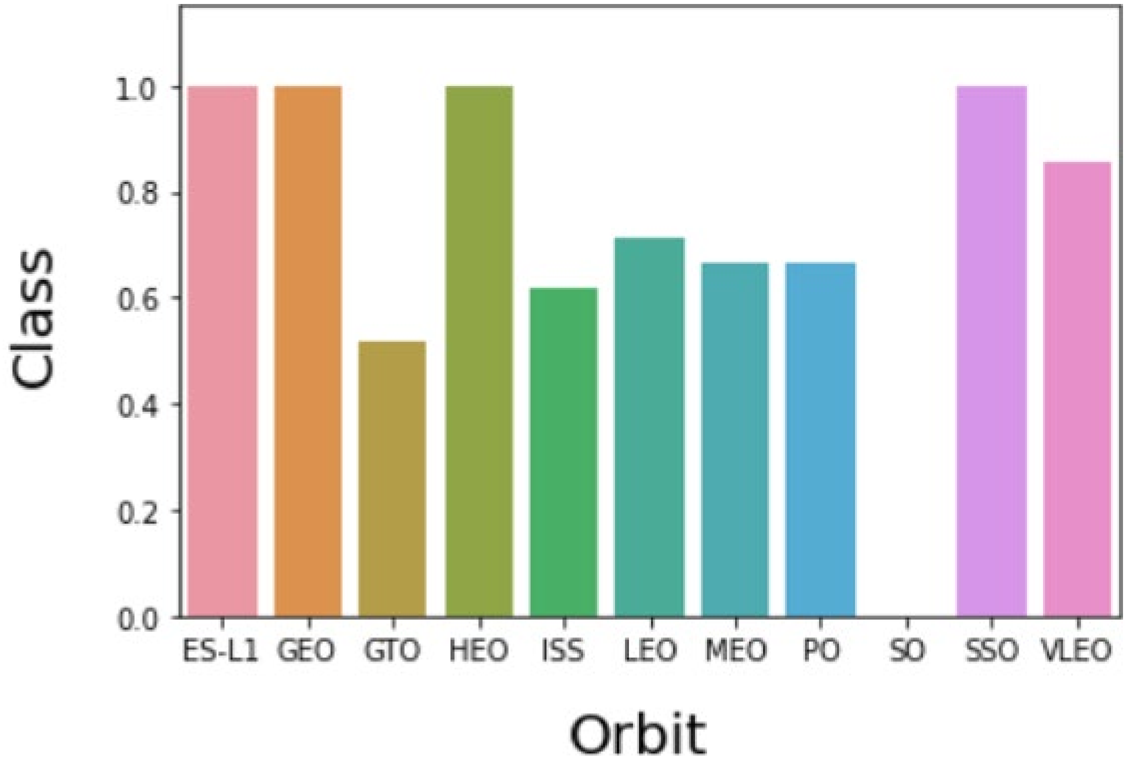
With Flight Numbers more than 30, the success rate for the Rocket is increasing.

Payload vs. Launch Site



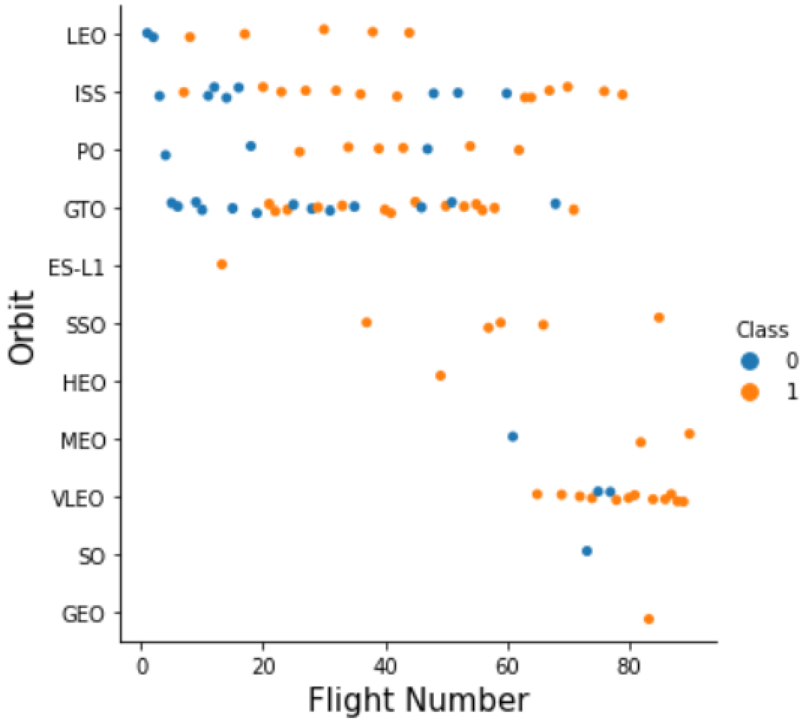
Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000). And the payload mass greater than 7000Kg have the higher success rate for the Rocket.

Success Rate vs. Orbit Type



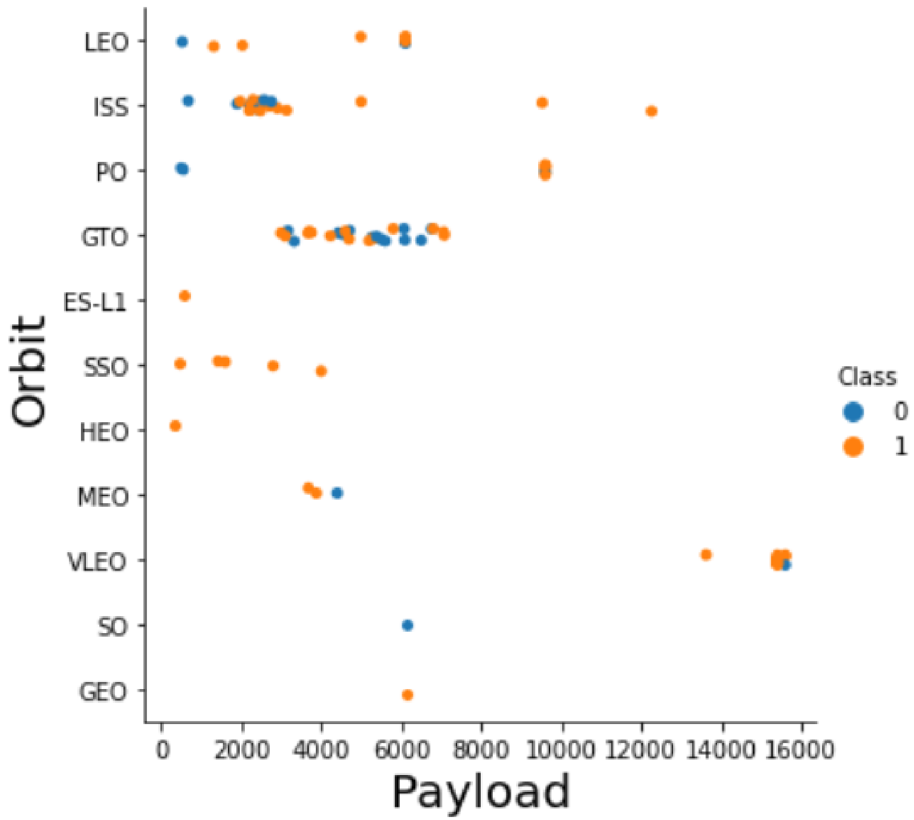
ES-L1, GEO, HEO, SSO has highest Sucess rates.

Flight Number vs. Orbit Type



You should see 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.

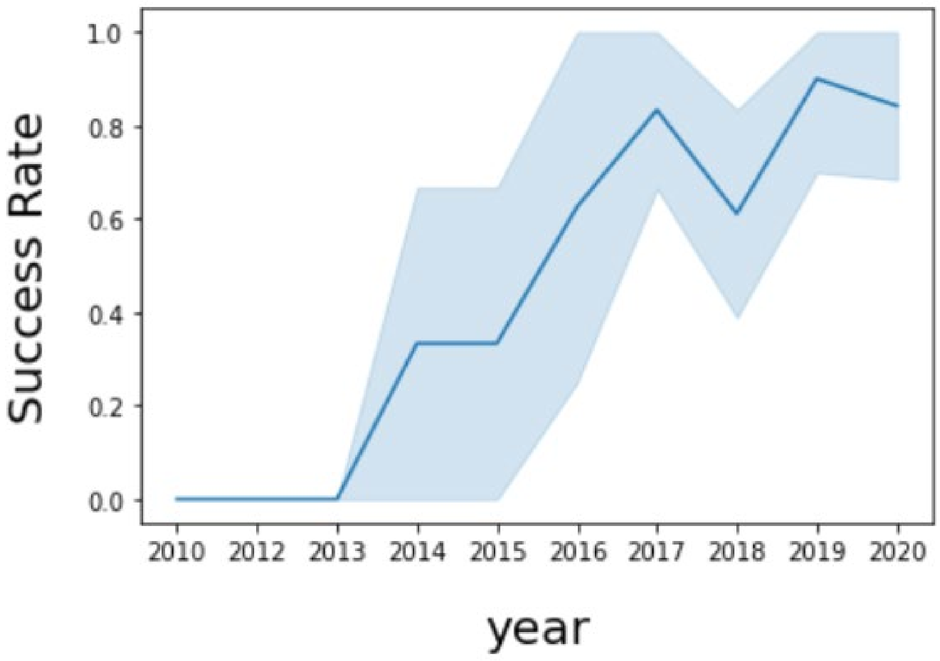
Payload vs. Orbit Type



With heavy payloads 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 increase since 2013 till 2020.

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

The DISTINT statement is used to return only distinct (different) values. Inside the SpaceX table, the column Launch_Sites contained many duplicate values; and in our case we just want to list the different (distinct) values.

Launch Site Names Begin with 'CCA'

```
%sql SELECT LAUNCH_SITE from SPACEXTBL 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

The SQL SELECT LIMIT statement is used to retrieve records from one or more tables in a database and limit the number of records returned based on a limit value. The LIKE command is used in a WHERE clause to search for a specified pattern in a column.

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

The SUM() function returns the total sum of a numeric column.

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXTBL;
```

payloadmass
2928

The AVG statement calculates Average in the PAYLOAD_MASS_KG_ column.

First Successful Ground Landing Date

```
%sql select min(DATE) from SPACEXTBL;
```

2015-12-22

The MIN() function returns the smallest value of the selected column. In our case, the DATE column.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

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

Using WHERE with AND statement we can expand the power of the filter associating with a condition. Using the BETWEEN statement we can indicate a specific range to the filter.

Total Number of Successful and Failure Mission Outcomes

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

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

Successful Mission
100

Failure Mission
1

The COUNT() function returns the number of rows that matches a specified criterion. Using the LIKE statement, we can filter every letters in quotes " " .

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 B1049.7	F9 B5 B1056.4
F9 B5 B1048.5	F9 B5 B1051.3	F9 B5 B1058.3
F9 B5 B1049.4	F9 B5 B1051.4	F9 B5 B1060.2
F9 B5 B1049.5	F9 B5 B1051.6	F9 B5 B1060.3

The DISTINT statement is used to return only distinct (different) values (BOOSTER VERSION column).
We use the WHERE statement associating with MAX statement to figure out the maximum payload value in the PAYLOAD_MASS_KG_ column. In this case we had used a subquery to improve the filter.

2015 Launch Records

```
%sql SELECT BOOSTER_VERSION, LAUNCH_SITE FROM SPACEX WHERE DATE LIKE '2015-%' AND \
LANDING__OUTCOME = 'Failure (drone ship)';
```

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

We used a combinations of the WHERE clause, LIKE + AND conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 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 ;
```

We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2010-03-20.

We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order.

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

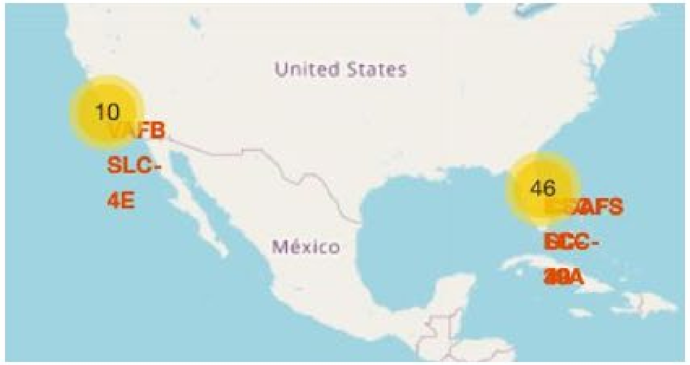
Section 3

Launch Sites Proximities Analysys

All Launch Sites on Folium Map

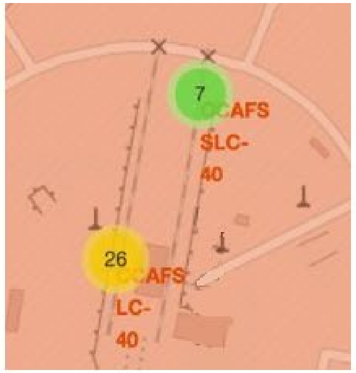
We can see that the SpaceX launch sites are near to the United States of America coasts i.e., Florida and California Regions.

Color Labeled Launch Records

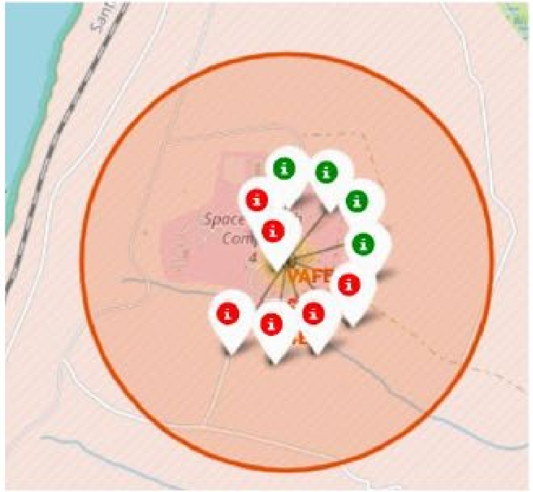


Green Marker shows successful launches and **Red Marker** shows failures.

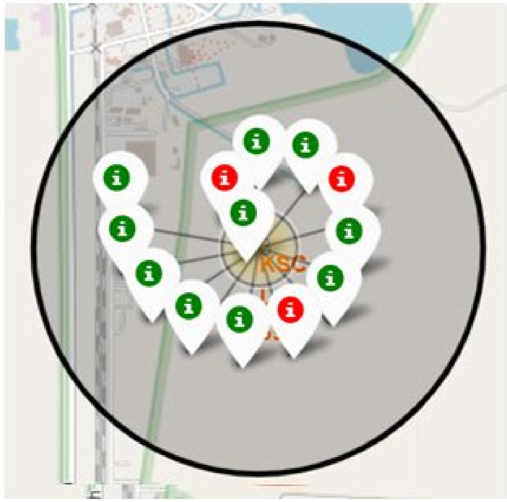
From these screenshots **its easily understandable that KSC LC-39A has the maximum probability of success.**



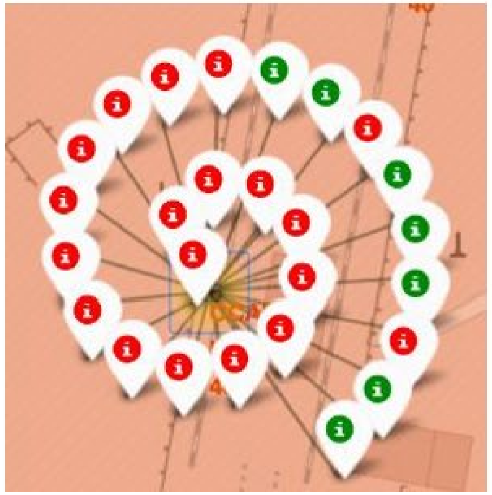
VAFB SLC-4E



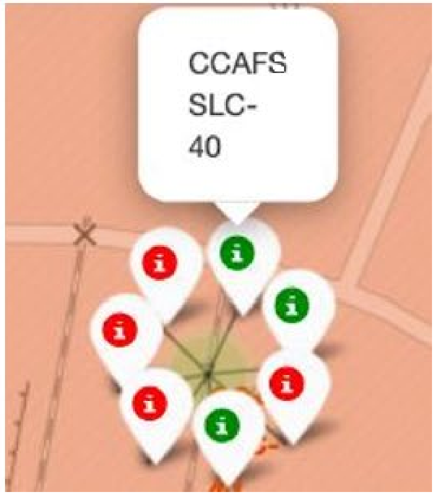
KSC LC-39A



CCAFS LC-40



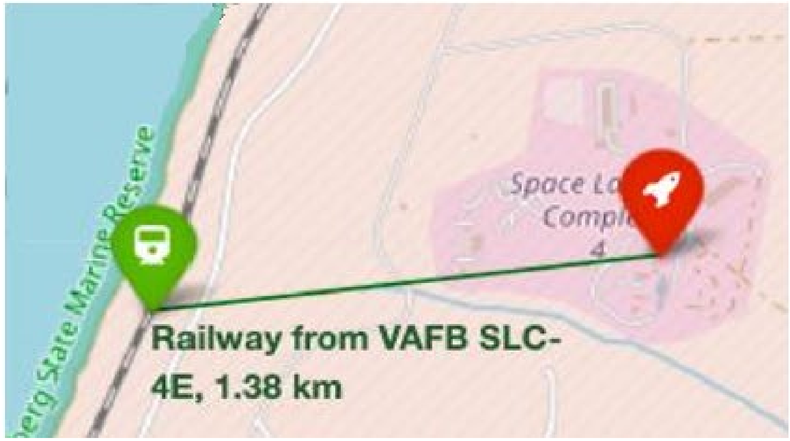
CCAFS SLC-40

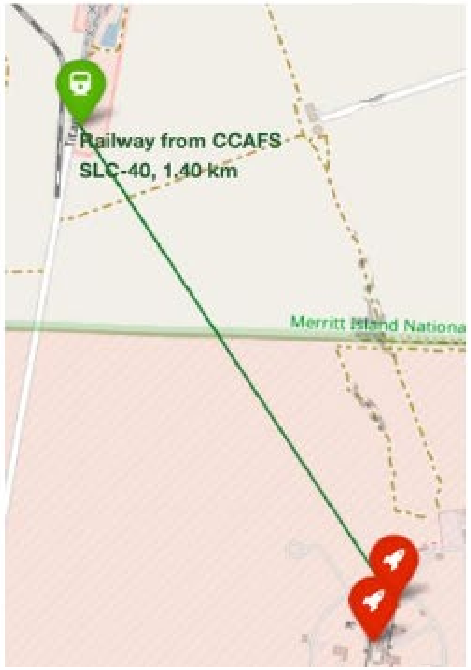


Launch Site Distances from Equator & Railways

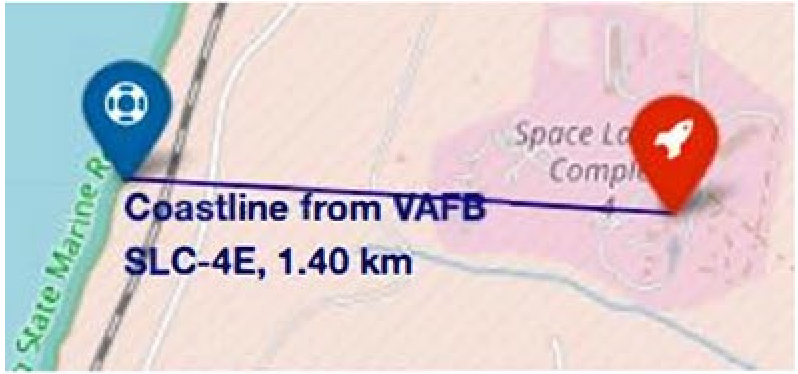
Distance from Equator is greater than 3000 Km for all sites.

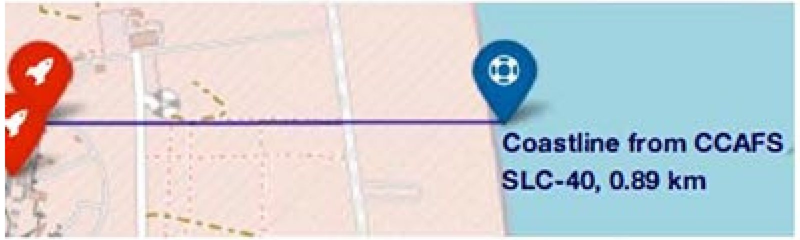
Distance for all launch sites from railway tracks are greater than .7 Km for all sites. So, launch sites are not so far away from railway tracks.





Launch Site Distances from Coastlines & Cities

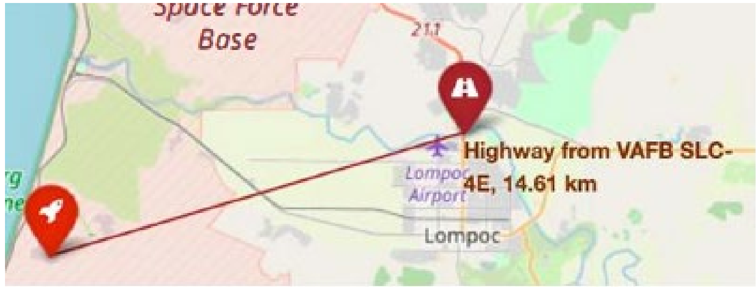




Distance for all launch sites from coastline is less than 4 Km.

Distance for all launch sites from cities is greater than 14 Km for all sites. So, launch sites are far away from cities.

Launch Site Distances from Highways



Distance for all launch sites from highways is greater than 5 Km for all sites. So, launch sites are relatively far away from highways.

Launch Site Distances Resume

Distance for all launch sites from highways is greater than 5 Km for all sites. So, launch sites are relatively far away from highways.

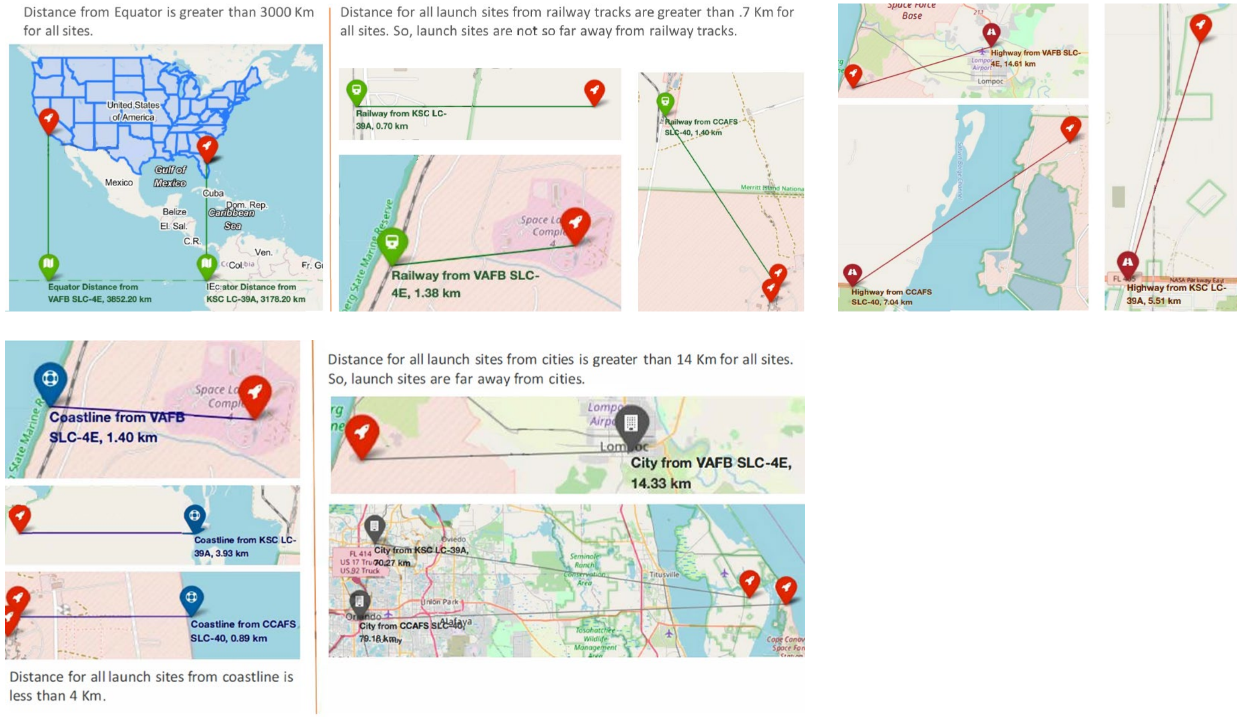
The launch sites keep certain distance away from Cities (15 Km> distance > 80 Km).

There is no launch sites in proximity to the Equator Line. (4000 Km> distance > 3000 Km).

There is no launch sites in close proximity to Highways (15 Km> distance > 5 Km).

There is launch sites in close proximity to Coastline (5 Km> distance > 5 Km).

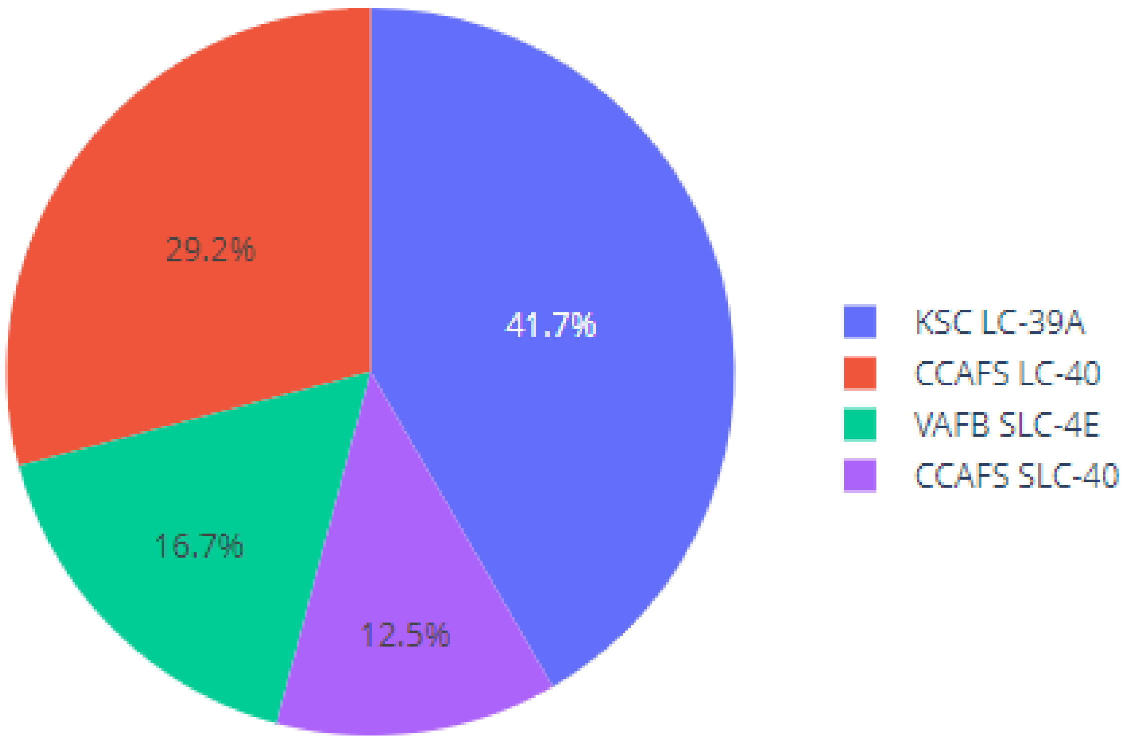
There is launch sites in close proximity to Railways. (2 Km> distance >5 Km).



Section 4

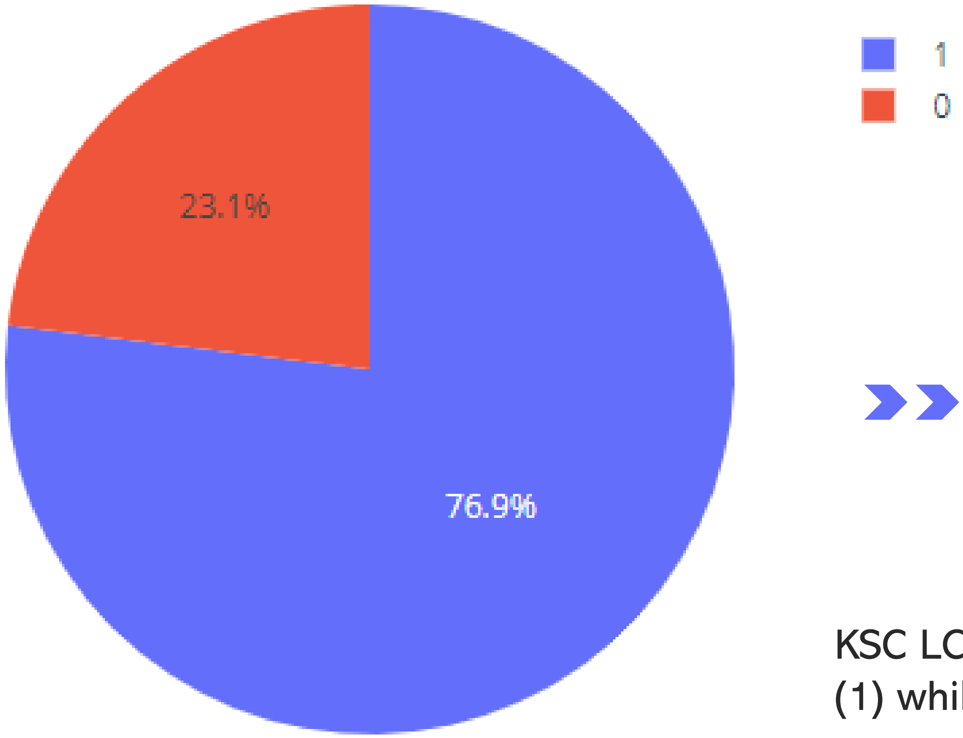
Building a Dashboard with Plotly Dash

Launch Success for All Sites



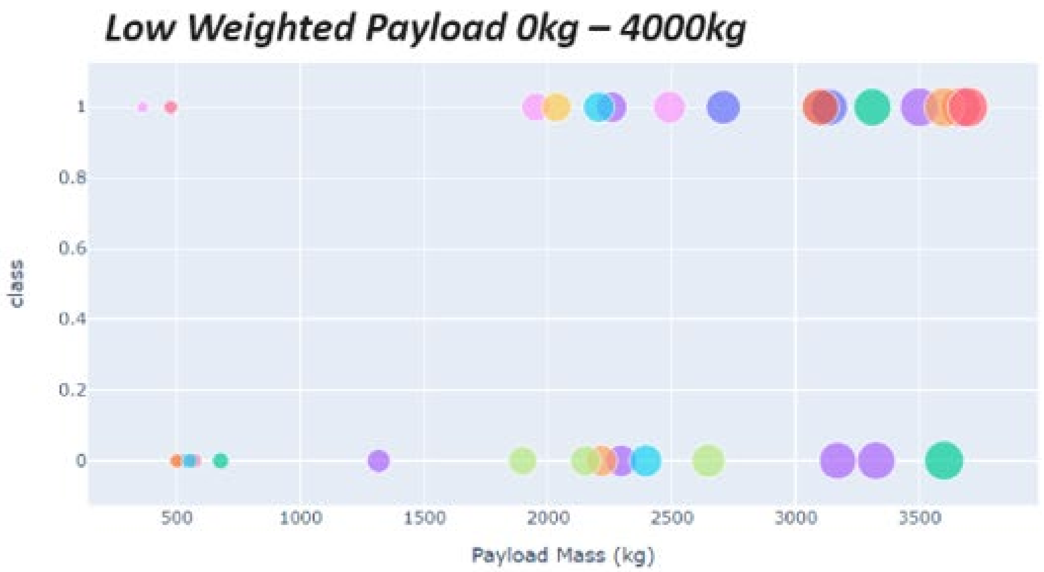
We can see that KSC LC-39A had the most successful launches from all the sites.

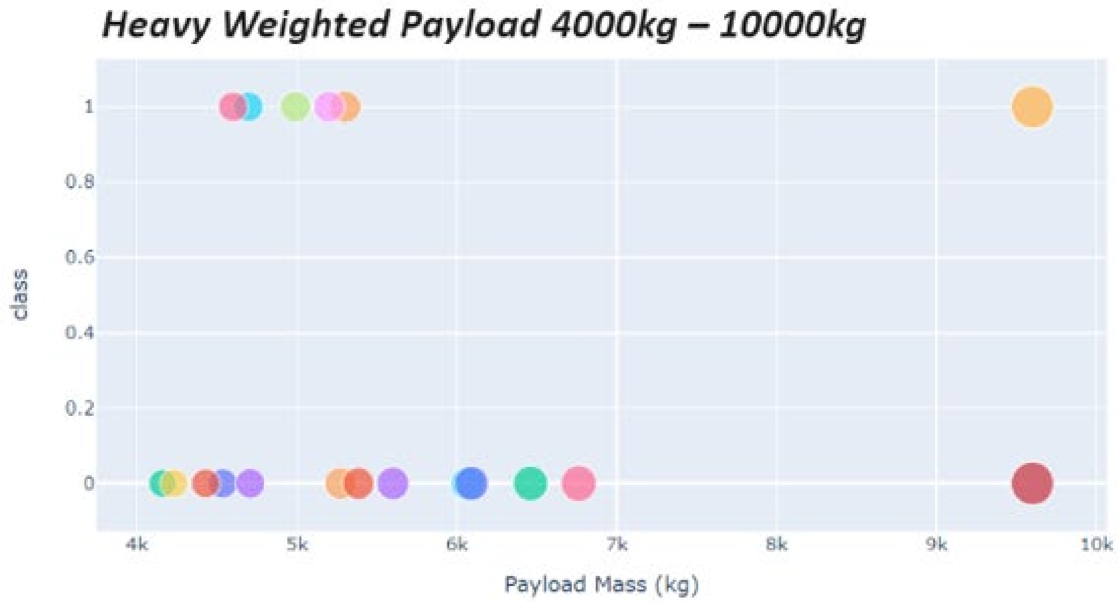
Launch Site with Highest Launch Success Ratio

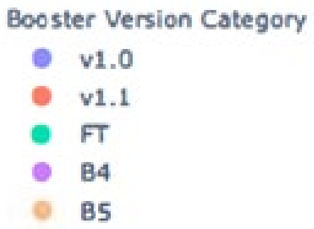


KSC LC-39A achieved a 76.9% success rate (1) while getting a 23.1% failure rate (0).

Payload vs. Launch Outcome Scatter Plot for All Sites





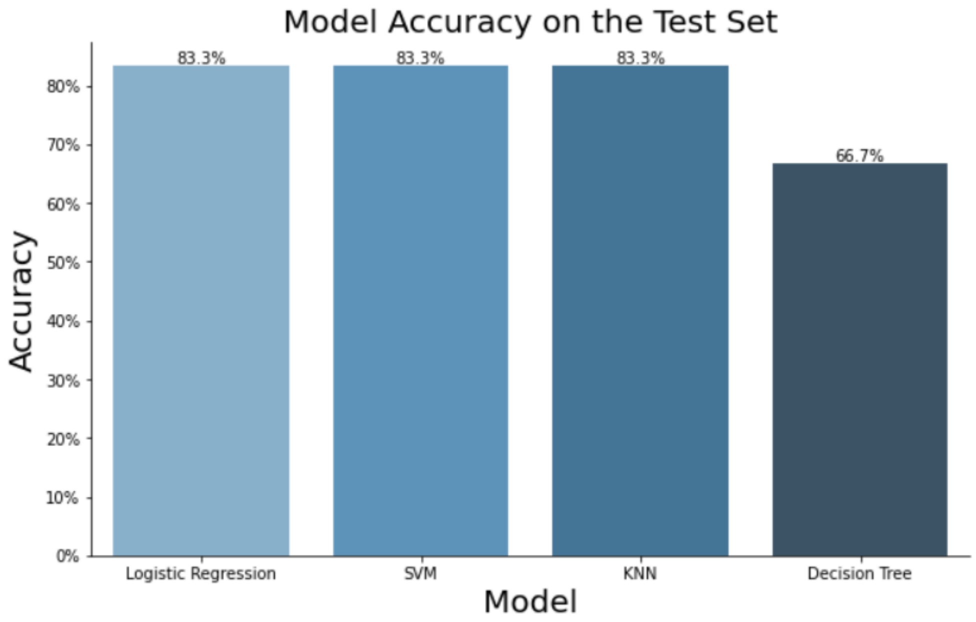


The success rates for **low weighted payloads** is higher than the heavy weighted payloads.

Section 5

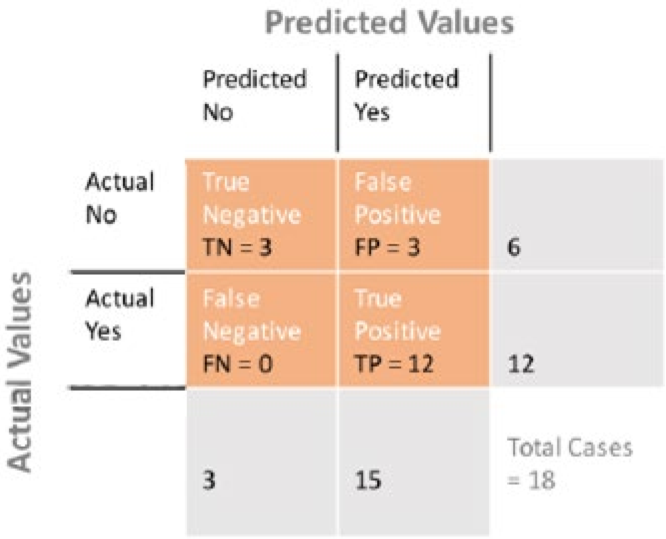
Predictive Analysis (Classification)

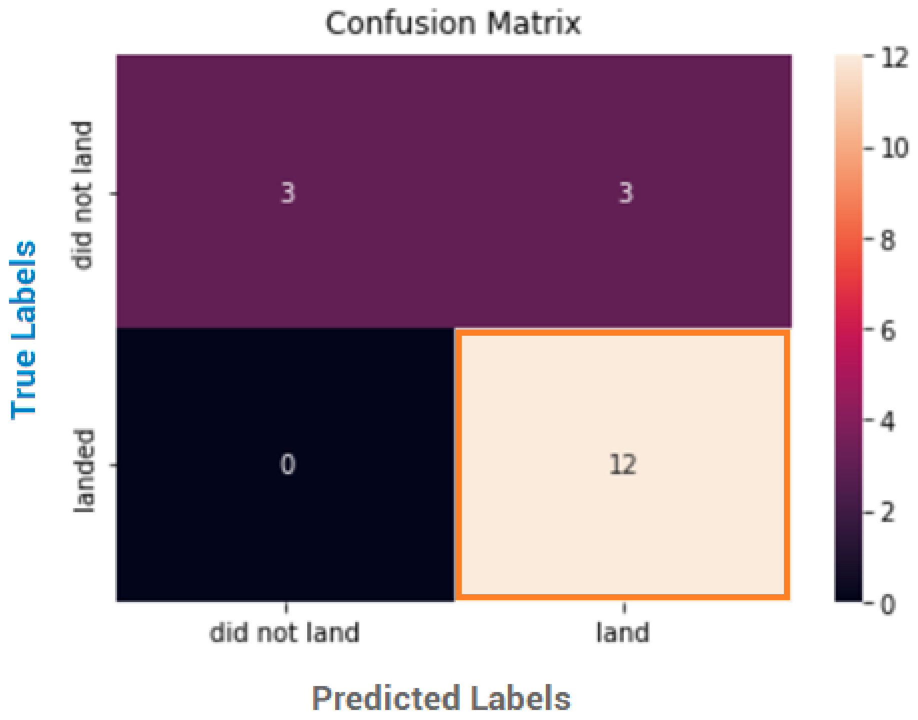
Classification Accuracy



Four machine learning models were used. The Logistic Regression, SVM and KNN models has the highest classification accuracy, 83.3%. The Decision Tree model has the lower, 66,7%.

Confusion Matrix





The Logistic Regression, SVM and KNN models has the same Confusion Matrix. The “landed” x “land” block (True Positive =12) shows the landings success.

Conclusions

We had pursuit what factors determine if the rocket will land successfully, what features impact the rate of successful landing and the conditions which will aid SpaceX have to achieve the best results. So, the data show us that:

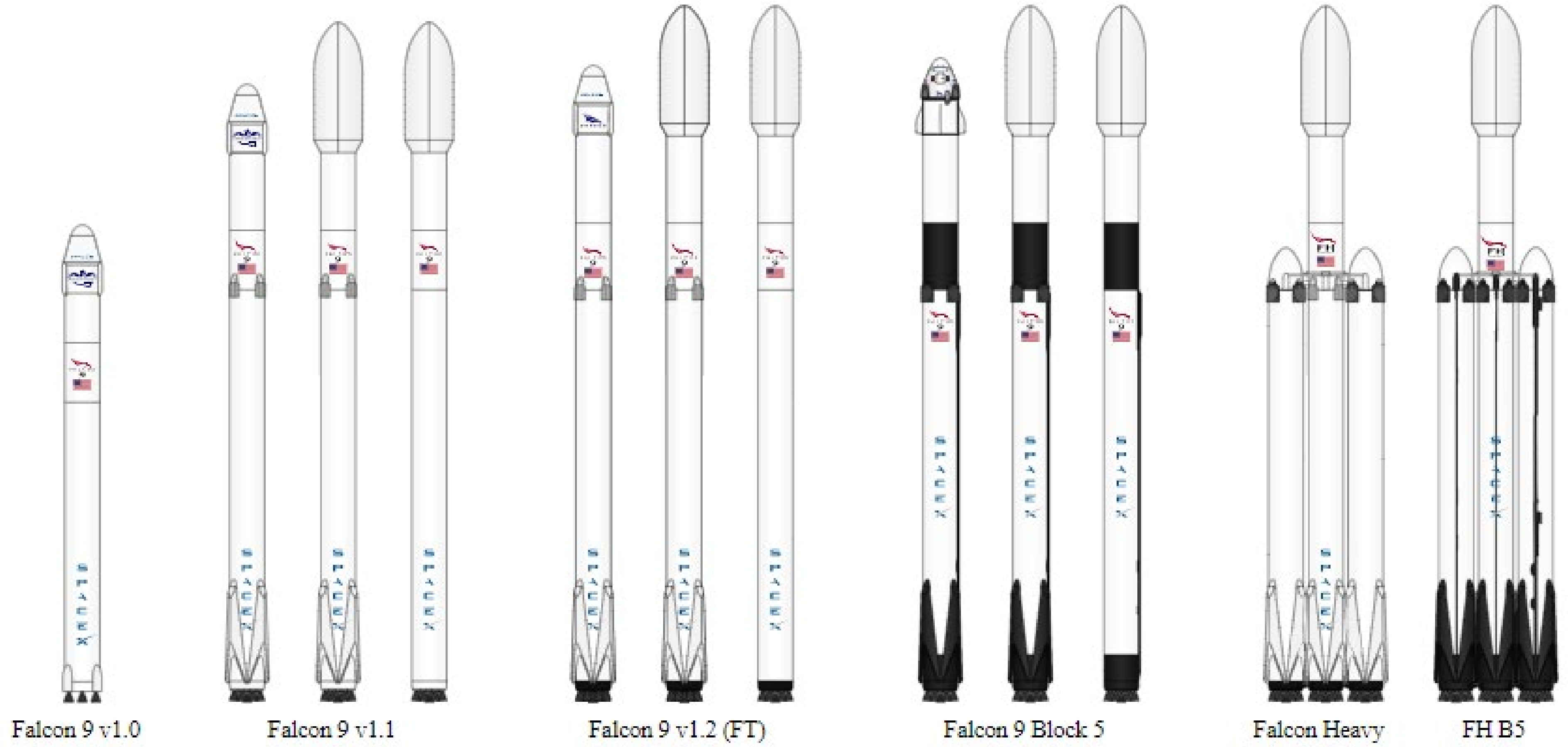
- The best Orbits are: ES-LI, GEO, HEO, SSO.
- KSC LC-39A is best place to make launches, with 76.9% success rate .
- Low weighted payloads perform better than the heavier payloads. However, in the CCAFS SLC 40 Launch Site the payload mass greater than 7000Kg have the higher success rate for the Rocket.
- LR, SVM and KNN Algorithms are the best Machine Learning Models for provided Dataset.

Success rates for SpaceX launches has been increasing with Time. Considering that there are variables that significantly alter the landing success rate, it is possible to say that it is a matter of time for Space X to reach an incredible level of performance and precision.

Appendix

Appendix

Falcon Models



Appendix

[hide] Flight No.	Date and time (UTC)	Version, Booster [b]	Launch site	Payload[c]	Payload mass	Orbit	Customer	Launch outcome	Booster landing
1	4 June 2010, 18:45	F9 v1.0[7] B0003[8]	CCAFS, SLC-40	Dragon Spacecraft Qualification Unit	No payload (excl. Dragon Mass)	LEO	SpaceX	Success	Failure[9][10] (parachute)
	First flight of Falcon 9 v1.0.[11] Used a boilerplate version of Dragon capsule which was not designed to separate from the second stage.(more details below) Attempted to recover the first stage by parachuting it into the ocean, but it burned up on reentry, before the parachutes even got to deploy.[12]								
2	8 December 2010, 15:43[13]	F9 v1.0[7] B0004[8]	CCAFS, SLC-40	Dragon demo flight C1 (Dragon C101)	Classified (excl. Dragon Mass)	LEO (ISS)	NASA (COTS) NRO	Success[9]	Failure[9][14] (parachute)
	Maiden flight of SpaceX's Dragon capsule, consisting of over 3 hours of testing thruster maneuvering and then reentry.[15] Attempted to recover the first stage by parachuting it into the ocean, but it disintegrated upon reentry, again before the parachutes were deployed.[12] (more details below) It also included two CubeSats,[16] and a wheel of Brouère cheese. Before the launch, SpaceX discovered that there was a crack in the nozzle of the 2nd stage's Merlin vacuum engine. So Elon just had them cut off the end of the nozzle with a pair of shears and launched the rocket a few days later. After SpaceX had trimmed the nozzle, NASA was notified of the change and they agreed to it.[17]								
3	22 May 2012, 07:44[18]	F9 v1.0[7] B0005[8]	CCAFS, SLC-40	Dragon demo flight C2+[19] (Dragon C102)	525 kg (1,157 lb)[20] (excl. Dragon mass)	LEO (ISS)	NASA (COTS)	Success[21]	No attempt
	The Dragon spacecraft demonstrated a series of tests before it was allowed to approach the International Space Station. Two days later, it became the first commercial spacecraft to board the ISS.[18] (more details below)								
4	8 October 2012, 00:35[22]	F9 v1.0[7] B0006[8]	CCAFS, SLC-40	SpaceX CRS-1[23] (Dragon C103)	4,700 kg (10,400 lb) (excl. Dragon mass)	LEO (ISS)	NASA (CRS)	Success	No attempt
				Orbcomm-OG2[24]	172 kg (379 lb)[25]	LEO	Orbcomm	Partial failure[26]	
	CRS-1 was successful, but the secondary payload was inserted into an abnormally low orbit and subsequently lost. This was due to one of the nine Merlin engines shutting down during the launch, and NASA declining a second reignition, as per ISS visiting vehicle safety rules, the primary payload owner is contractually allowed to decline a second reignition. NASA stated that this was because SpaceX could not guarantee a high enough likelihood of the second stage completing the second burn successfully which was required to avoid any risk of secondary payload's collision with the ISS.[27][28][29]								

Wikipedia Web Scraping

Appendix

Data Wrangling Session

```
import pandas as pd
import numpy as np
```

```
df.isnull().sum()/df.count()*100
```

```
FlightNumber      0.000
Date              0.000
BoosterVersion    0.000
PayloadMass       0.000
Orbit             0.000
LaunchSite        0.000
Outcome           0.000
```

```
df.dtypes
```

```
FlightNumber        int64
Date               object
BoosterVersion     object
PayloadMass       float64
```

```
# Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

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

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

```
True ASDS      41
None None      19
True RTLS      14
```

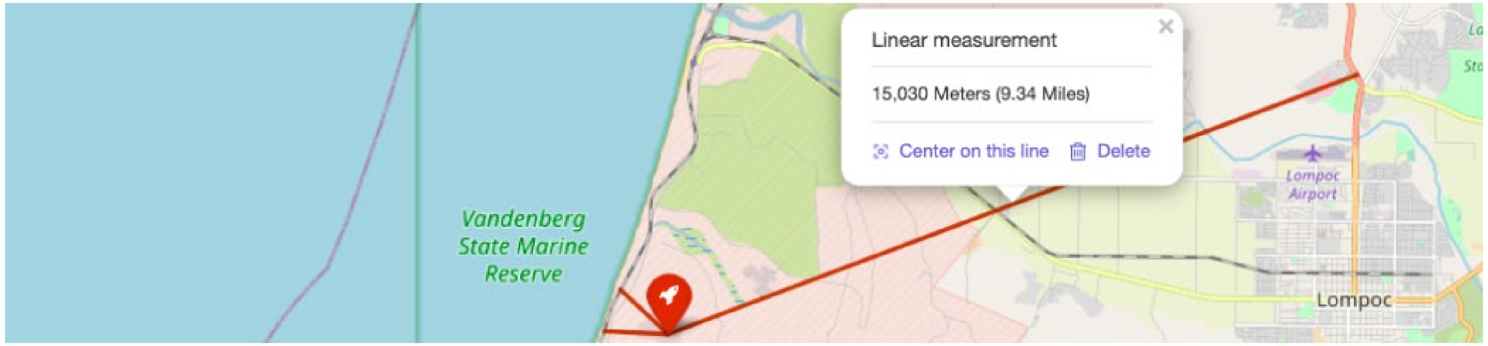
```
df["Class"].mean()*100
```

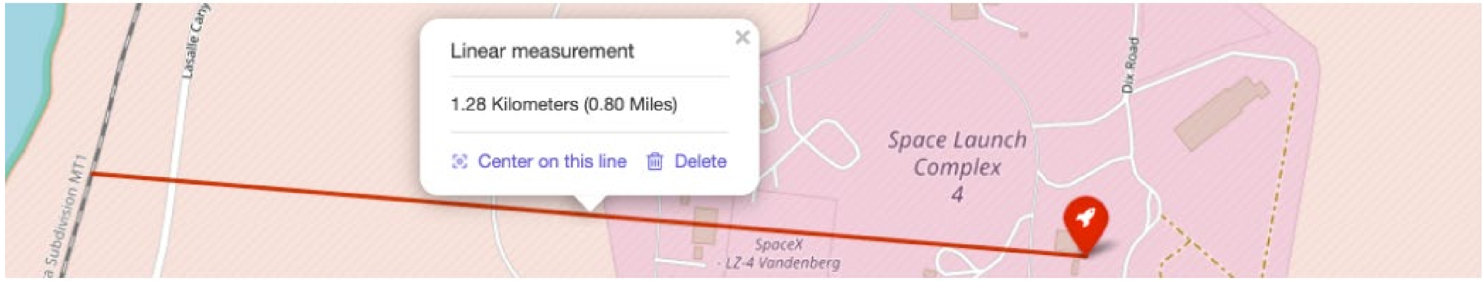
```
66.66666666666666
```

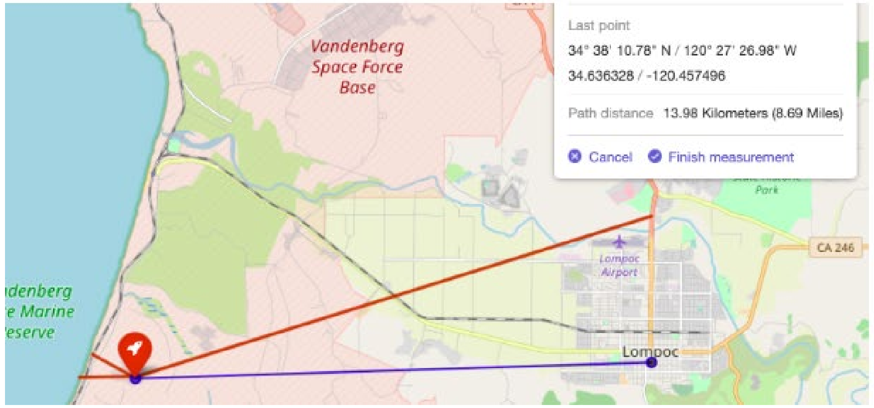
```
df.to_csv("dataset_part_2.csv", index=False)
```

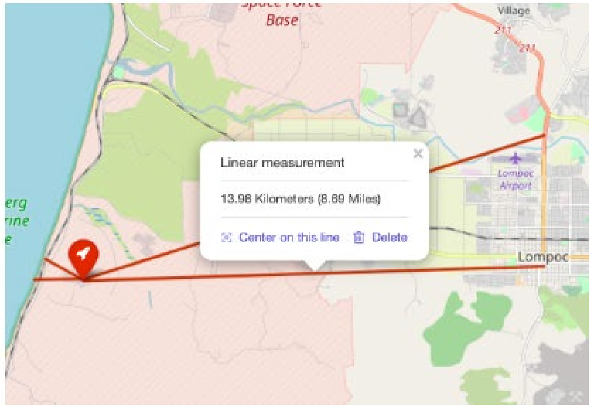
Appendix

Proximities



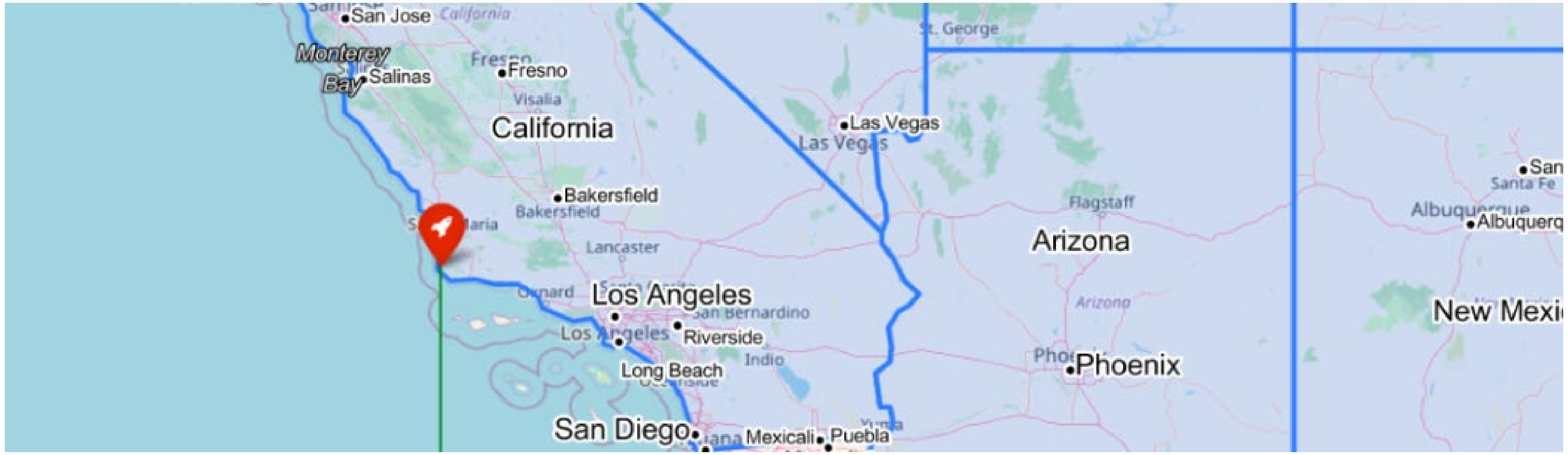






Appendix

Proximities



Appendix

SQL query successful and failure mission outcomes.

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

Successful Mission
100

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

Failure Mission
1


Thank you!