IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Rajesh Prashanth A
10/February/2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results

Introduction

- Project Background

Space X on its website advertises that their Falcon 9 rockets can launch a payload with a cost of 62 million dollars, while other providers cost upward of 165 million dollars.

Space X makes this huge difference with the re-use of the first stage of Falcon 9. Therefore, if we have to determine, the cost of a Space X launch, it would be suffice to track, whether the carriers from their previous launches had successfully landed. This information can be used by an alternate company that wants to compete with Space X.

The goal of this project is to predict if the first stage of the Falcon 9 carrier can land successfully with State of art Machine Learning Algorithms.

Introduction

- Problems to address

1. What are the variables that influence the landing of first stage of Falcon 9?

2. Inter-dependencies between the variables that determine the success rate of successful landing of Falcon 9.

3. Determining favorable conditions at which Space X could achieve best results and ensure that Falcon 9 will land successfully.

Section 1

Methodology

Methodology

Executive Summary

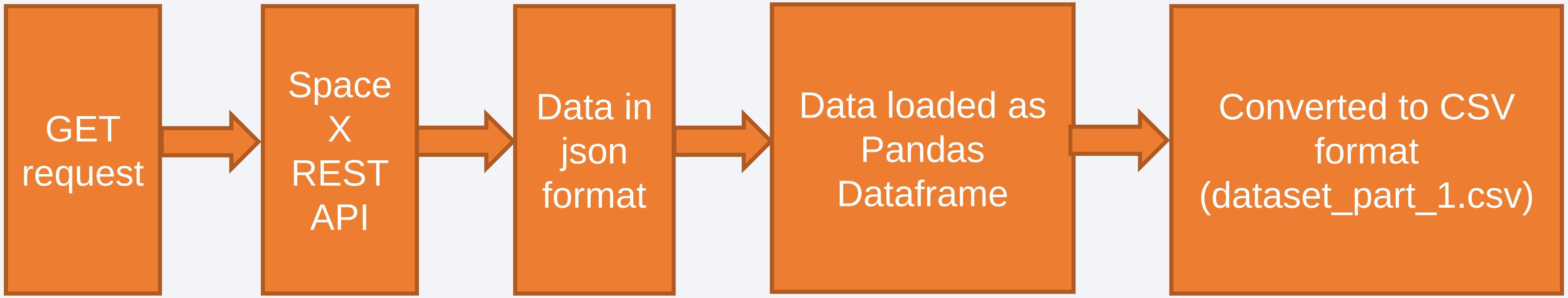
- Data collection methodology:
    1. Space X REST API
    2. Web scrapping Space X Wikipedia page
- Perform data wrangling
    1. One Hot Encoding data fields for Machine Learning and dropping irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    1. Build, tune, evaluate Machine Learning classification models

Data Collection

Dataset collection were made by

- Space X REST API (https://api.spacexdata.com/v4/launches/past)
- Web scrapping from Wikipedia Page of Space X project (https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922")

Space X REST API data collection protocol

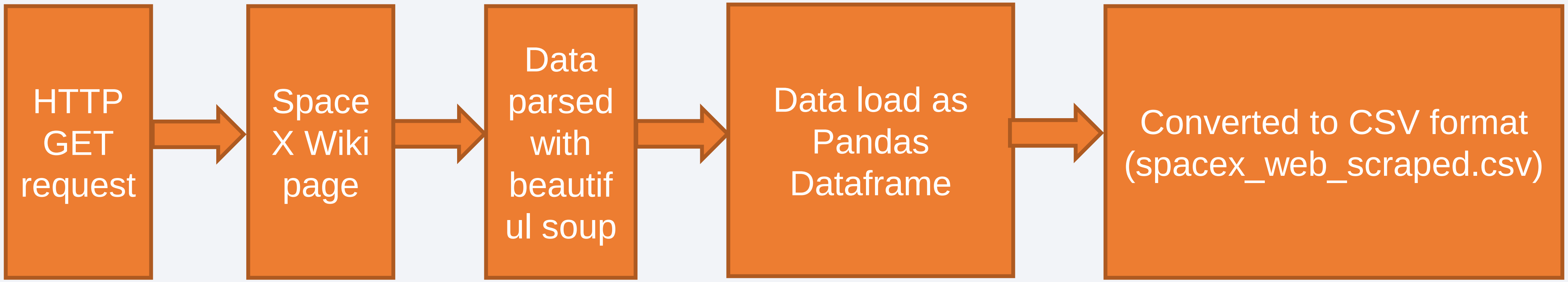


Data Collection

Dataset collection were made by

- Space X REST API (https://api.spacexdata.com/v4/launches/past)
- Web scrapping from Wikipedia Page of Space X project (https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922")

Space X Webscrapping data collection protocol



Data Collection – SpaceX API

1. Getting response from Space X REST API
2. Convert API response into json file
3. Normalize json results to Pandas Raw Dataframe
4. Parse Raw Dataframe for more information through custom functions and convert them to list and assign to python dictionary and then to Pandas Dataframe
5. Cleanup Data points in new Dataframe
6. Convert Dataframe to CSV file for portability and further re-use.

[Github Jupyter notebook URL]

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

```
response = requests.get(static_json_url).json()
```

```
# Use json_normalize method to convert the json result into a dataframe
data = pd.json_normalize(response)
```

```
# Call getBoosterVersion
getBoosterVersion(data)
```

```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

```
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}
```

```
# Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
data_falcon9
```

```
# Calculate the mean value of PayloadMass column
Mean_PayloadMass = data_falcon9.PayloadMass.mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan,Mean_PayloadMass)
```

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

Data Collection - Scraping

1. Getting response from Space X Wikipedia page
2. Create Beautifulsoup object
3. Find table data from raw webpage
4. Get the column names and create dictionary
5. Append data to keys
6. Convert dictionary to Pandas Dataframe
7. Convert Dataframe to CSV file for portability and further re-use.

[Github Jupyter notebook URL]

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

```
# use requests.get() method with the provided static_url
# assign the response to a object
data  = requests.get(static_url).text
```

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data,"html.parser")
```

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

```
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
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']=[]
```

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a n
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            #print(flight_number)
            launch_dict['Flight No.'].append(flight_number)
            datatimelist=date_time(row[0])
```

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

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

Data Wrangling

Exploratory Data Analysis (EDA)

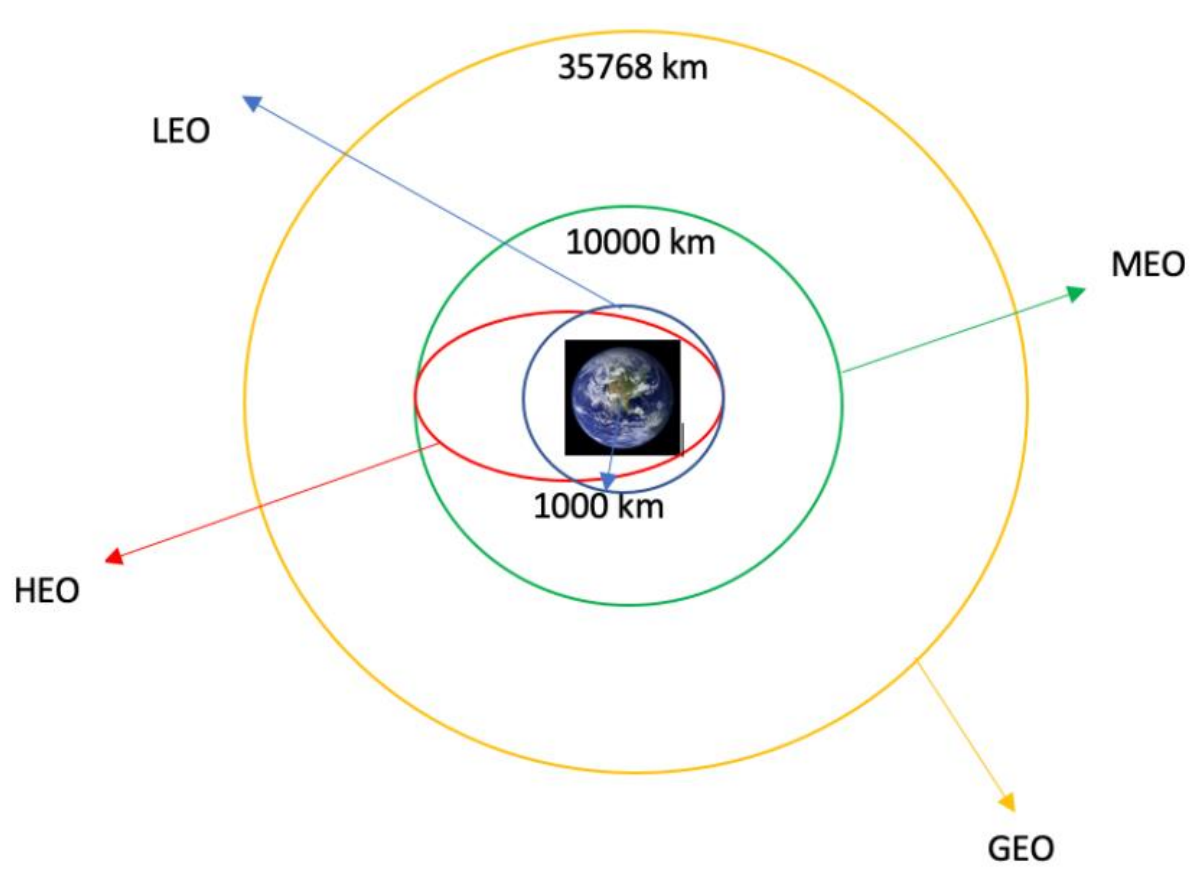
- Calculate the number of launches on each site

CCAFS SLC 40	55
KSC LC 39A	22
VAFB SLC 4E	13

- Calculate the number and occurrence of each orbit

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

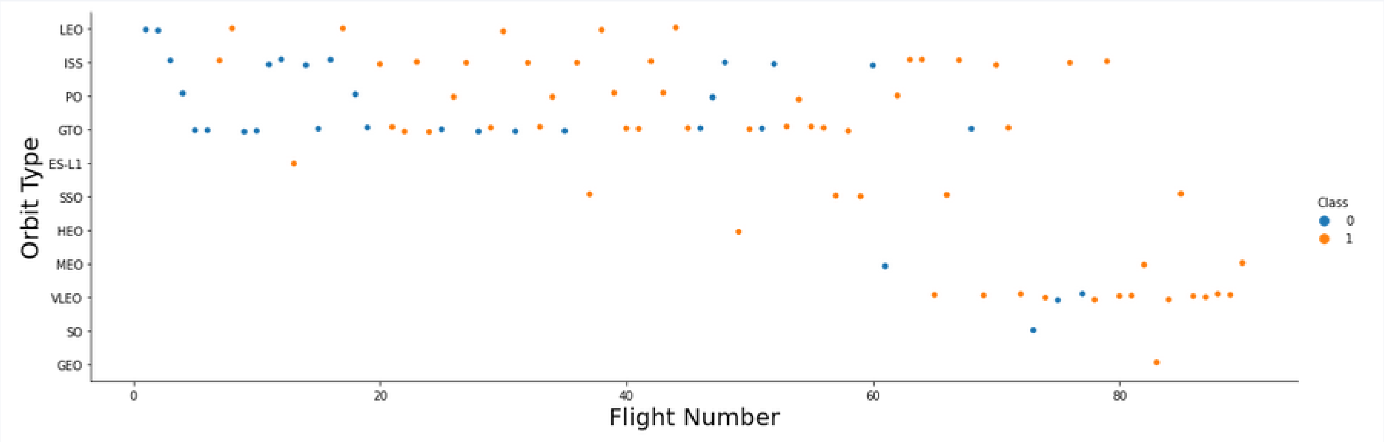
- Calculate the number and occurrence of mission outcome per orbit type
- Create a landing outcome label from Outcome column
- Label Success rate of each launch
- Convert Dataframe to CSV file for portability and further re-use



[Github Jupyter notebook URL]

EDA with Data Visualization

- Scatter Charts
  - ✓ To show relationship between independent and dependent variables (correlation).
    1. Flight Number vs Launch Site
    2. Payload vs Launch Site
    3. FlightNumber vs Orbit type
    4. Payload vs Orbit type
    5. FlightNumber vs PayloadMass

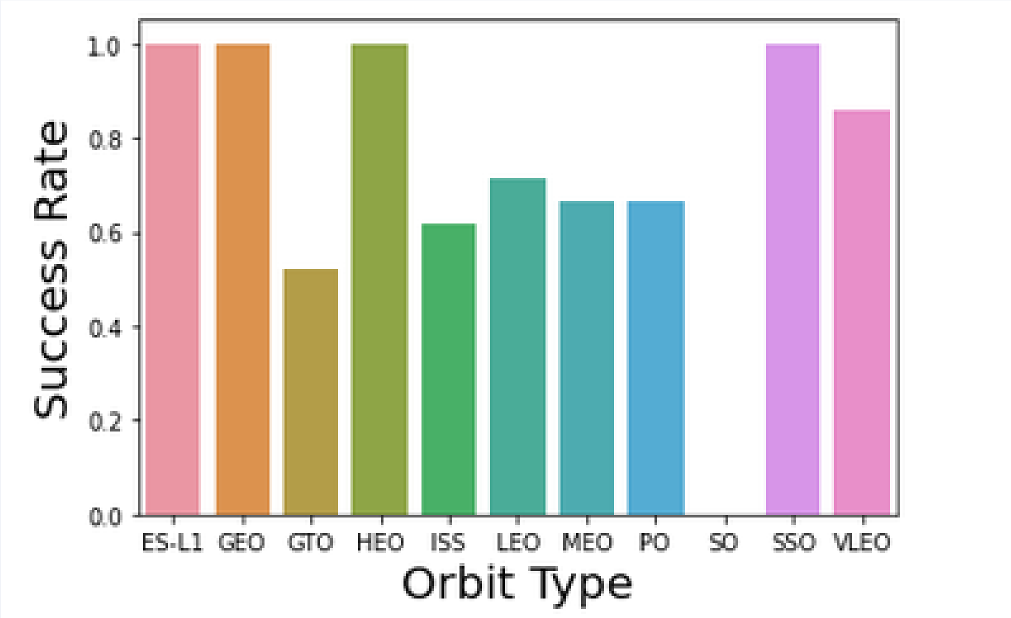


[Github Jupyter notebook URL]

EDA with Data Visualization

- Bar Charts
  - ✓ To compare sets of data between different groups at an instance

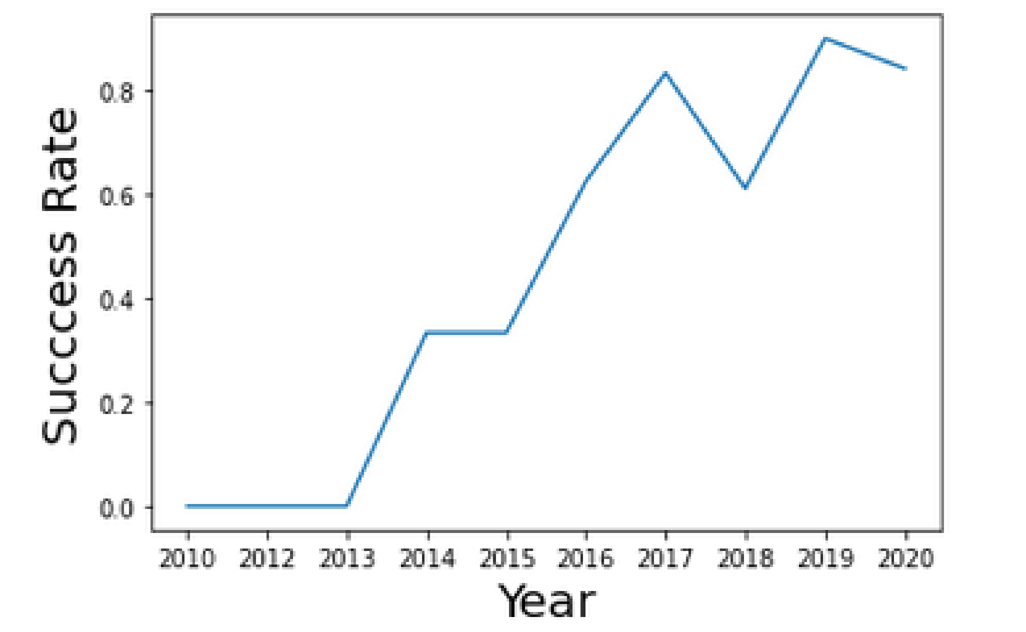
[Github Jupyter notebook URL]



EDA with Data Visualization

- Line Charts
  - ✓ To visualize the trend in the variation of dependent variable

[Github Jupyter notebook URL]



EDA with SQL

- Technique

    1. Load SpaceX Dataset to IBM DB2 Cloud hosted Database with Tablename as SPACEXTBL
    2. Connectivity to cloud instance of IBM DB2 is made with ibm_db_sa and ipython-sql python modules
    3. SPACEXTBL has been queried from jupyter notebook though in-line SQL queries supported by SQLAlchemy module

[Github Jupyter notebook URL]

EDA with SQL

- Parameters to query

1. Names of the unique launch sites in the space mission
2. The Launch sites begin with the string 'CCA'
3. The Total payload mass carried by boosters launched by NASA (CRS)
4. The Average payload mass carried by booster version F9 v1.1
5. The date when the first successful landing outcome in ground pad was acheived
6. The names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7. The total number of successful and failure mission outcomes
8. The names of the booster versions which have carried the maximum payload mass
9. 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 between the given dates

Github Jupyter notebook URL

Build an Interactive Map with Folium

- Visualize the Launch site location in an interactive map.
  - ✓ Latitude and Longitude at each launch site were added as Circle Marker with a label containing the name of launch sites
- Indication successful and failed launches in an interactive
  - ✓ Launch_outcomes(successes, failures) were added to class 1 (Green) and 0 (Red) on the map.
- Proximity Analysis with distance
  - ✓ Distance calculation made with Haversine's formula.
  - ✓ Lines were used to mark the distance between launch sites and different landmarks(Eg. Railways, highways, costal Lines, cities)

[Github Jupyter notebook URL]

Build a Dashboard with Plotly Dash

- Interactive with Flask and Dash web framework
  - Pie Chart
    - ✓ Total launches by a specific/all sites
      - ❖ Shows relative proportions of multiple classes of data
      - ❖ Shows contribution of each classes of date to the whole

  - Scatter plot
    - ✓ Outcome and Payload Mass (Kg) for the different Booster Versions
      - ❖ Shows the relationship between two variables.

[Github Python Dashboard]()

Predictive Analysis (Classification)

- Loaded the data using numpy and pandas module, transformed and split the data into training and testing test.
- Different machine learning models (Classification problem) and hyperparameters were tuned using GridSearchCV
- Accuracy was used as metric for the above model.
- Improved the model using feature engineering and algorithm tuning.
- Best performing classification model was chosen.

Github Jupyter notebook URL

Results

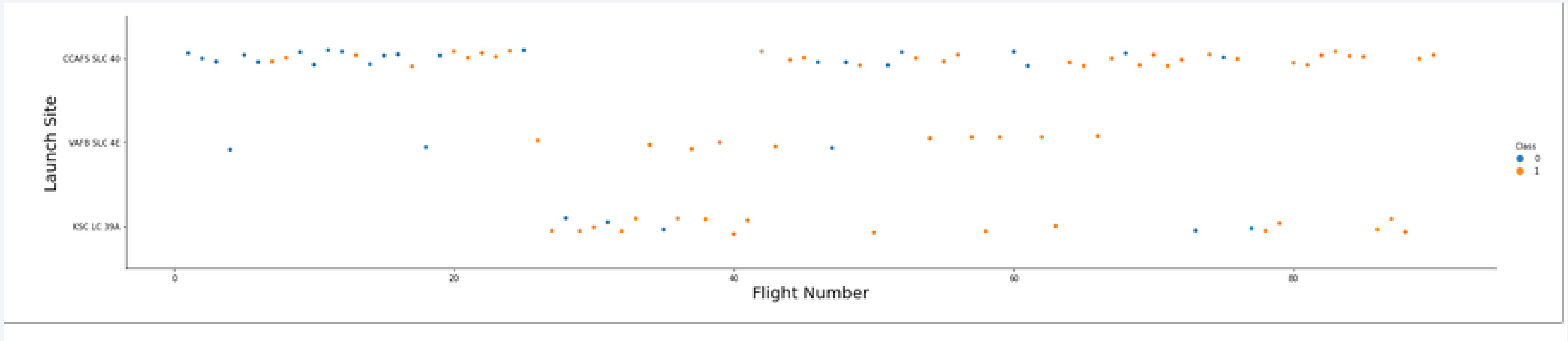
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

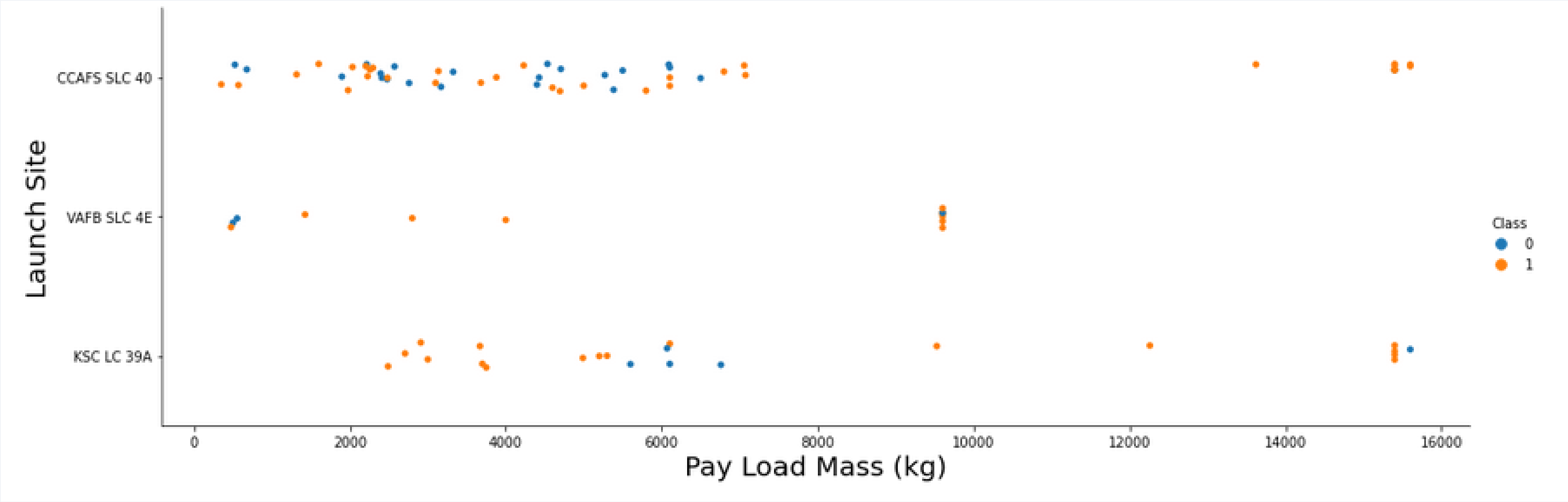
Flight Number vs. Launch Site

- Larger the flight number at a launch site resulted in the higher the success rate at that launch site.



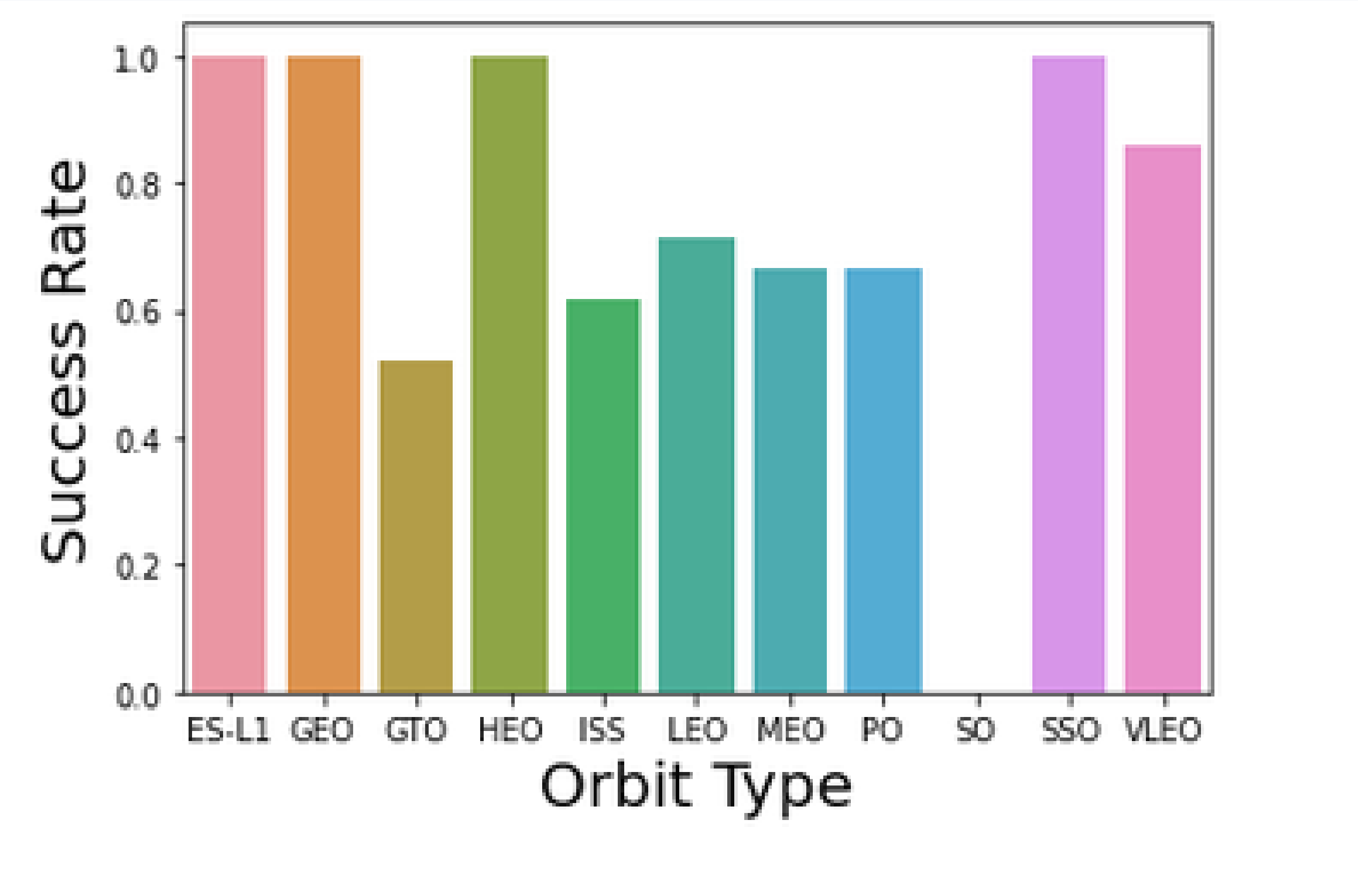
Payload vs. Launch Site

- Greater the payload mass at CCAFS SLC 40 resulted in higher success rate.



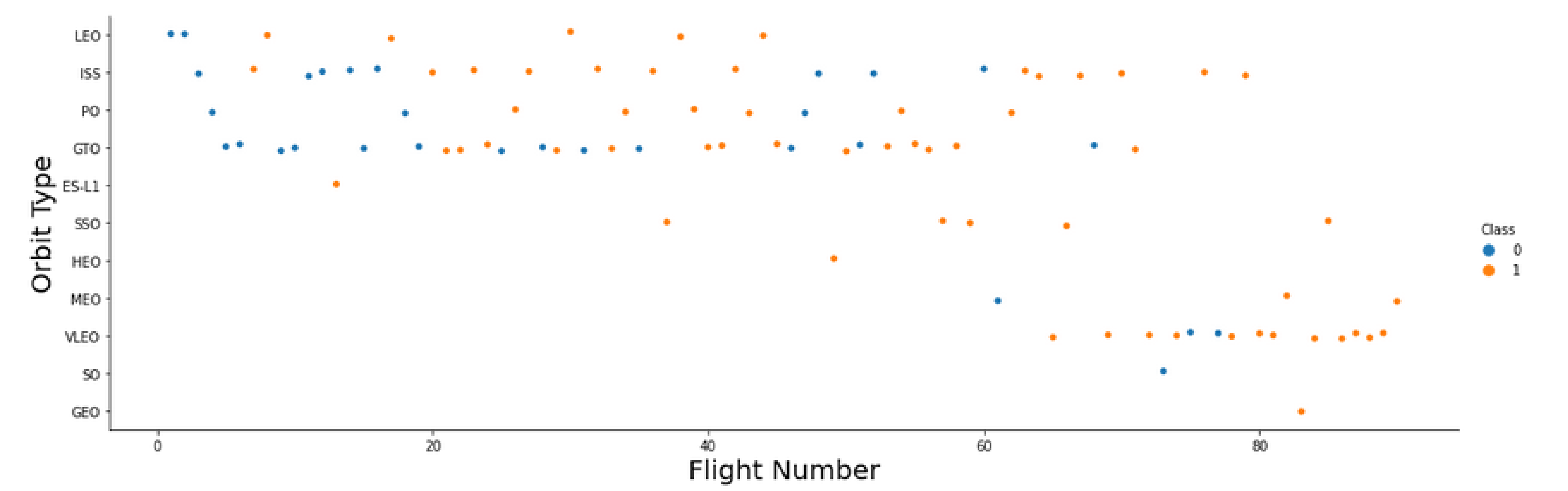
Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



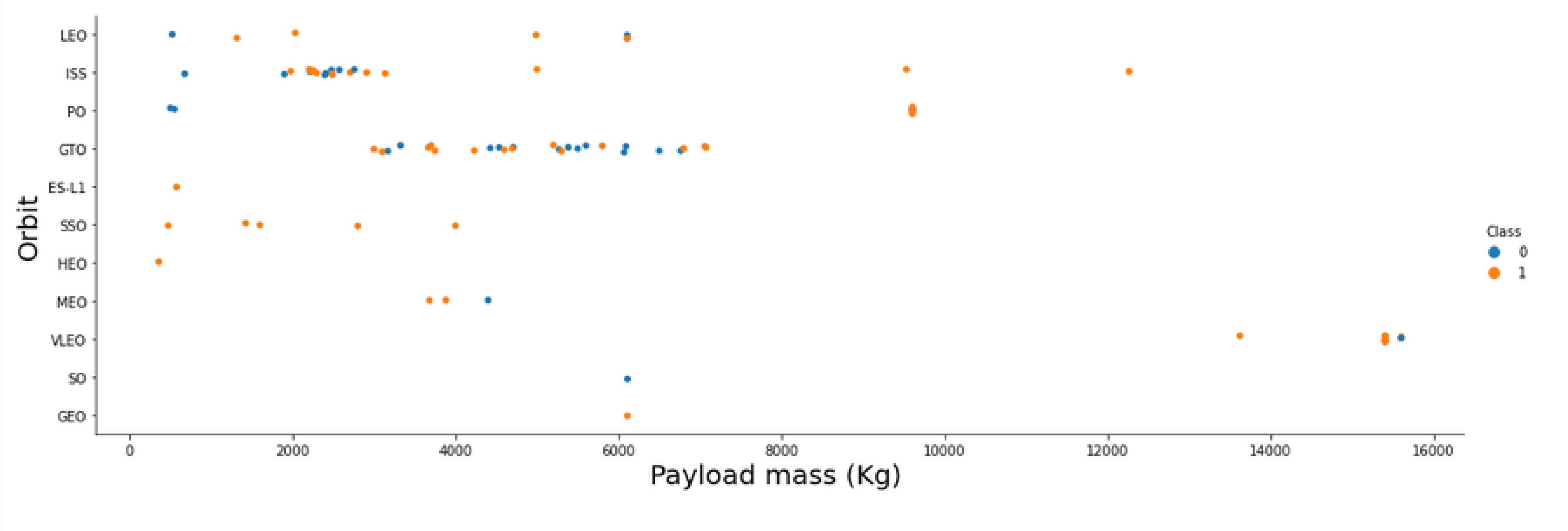
Flight Number vs. Orbit Type

- Vehicles launched in LEO and VLEO orbit have higher success is related to the number of flights



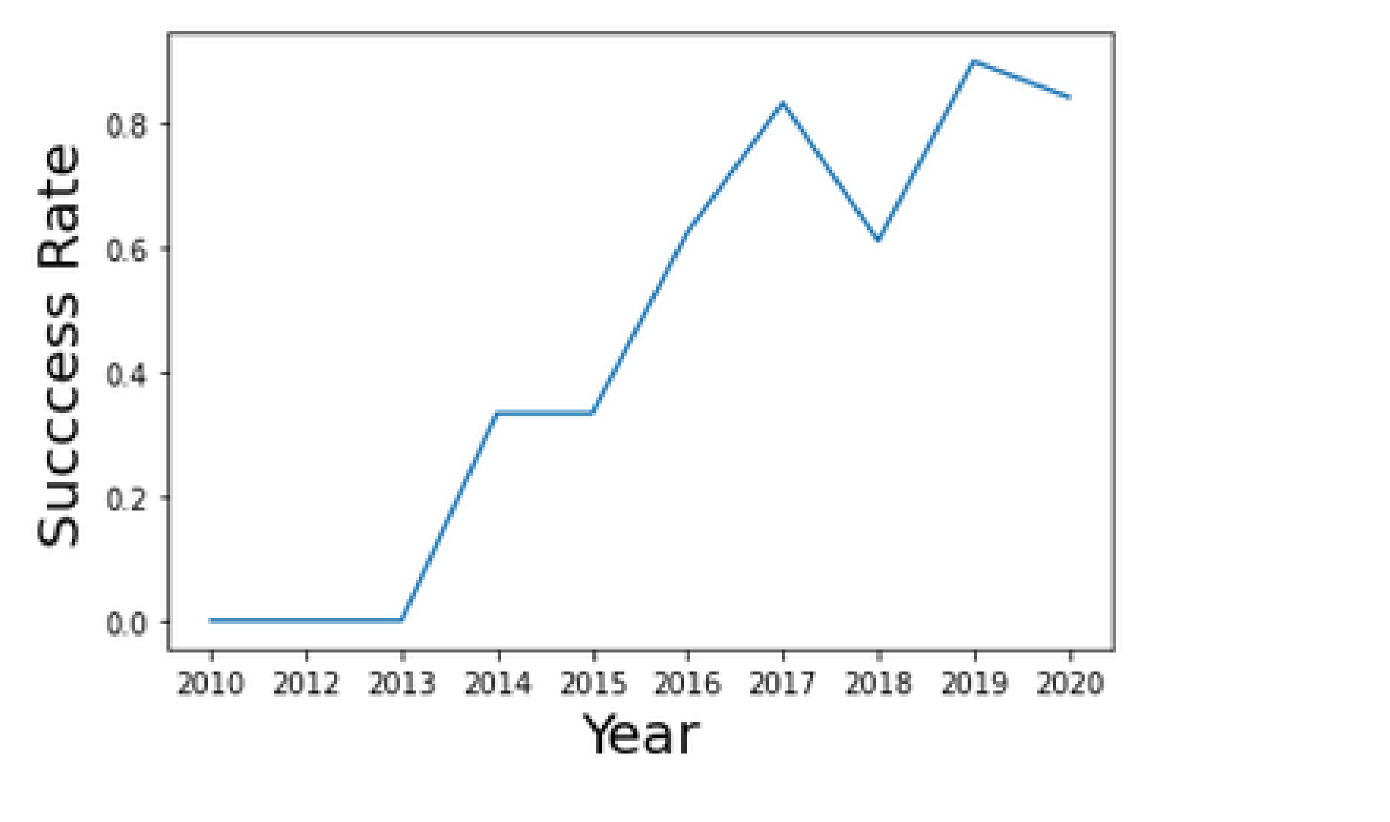
Payload vs. Orbit Type

- Heavy payloads have the successful landing for PO, LEO and ISS orbits



Launch Success Yearly Trend

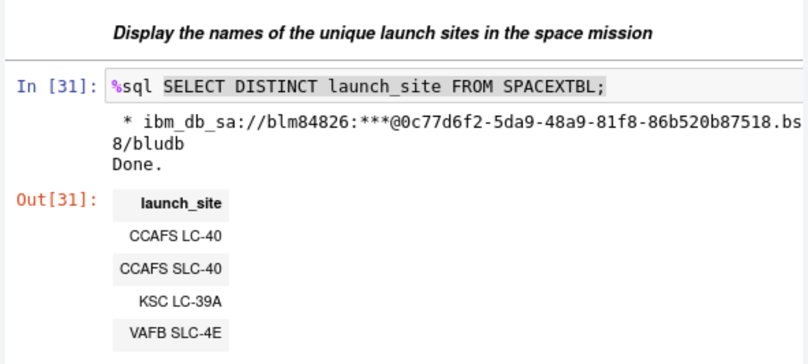
- Success rate since 2013 kept till 2020 is in increasing trend.



All Launch Site Names

SQL Query: SELECT DISTINCT launch_site FROM SPACEXTBL;

Launch sites
CCAFS LC-40 (Cape Canaveral Space Launch Complex 40)
CCAFS SLC-40 (Cape Canaveral Space Launch Complex 40)
KSC LC-39A (Kennedy Space Center Launch Complex 39)
VAFB SLC-4E (Vandenberg Space Launch Complex 4)



Keyword: DISTINCT shows only unique records from the Table SPACEXTBL

Launch Site Names Begin with 'CCA'

SQL Query: SELECT * FROM SPACEXTBL WHERE launch_site LIKE 'CCA%' LIMIT 5

```
In [32]: %sql SELECT * FROM SPACEXTBL WHERE launch_site LIKE 'CCA%' LIMIT 5

 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:3119
8/bludb
Done.
```

Out[32]:

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

Keyword: LIMIT restricts the record to given number. WHERE clause combined with like and the “%” can be used for pattern search

Total Payload Mass

SQL Query: SELECT SUM (payload_mass__kg_) FROM SPACEXTBL WHERE CUSTOMER ='NASA (CRS)'

```
%sql SELECT SUM (payload_mass__kg_) FROM SPACEXTBL WHERE CUSTOMER ='NASA (CRS)'
 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

1
45596

The total payload carried by boosters from NASA is 45596

Average Payload Mass by F9 v1.1

SQL Query: SELECT AVG (payload_mass__kg_) FROM SPACEXTBL WHERE booster_version ='F9 v1.1'

```
%sql SELECT AVG (payload_mass__kg_) FROM SPACEXTBL WHERE booster_version ='F9 v1.1'
```

```
 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:3119
8/bludb
Done.
```

1
2928

Keyword: AVG computes average for the given records

First Successful Ground Landing Date

SQL Query: SELECT MIN(DATE) FROM SPACEXTBL WHERE landing__outcome = 'Success (ground pad)'

```
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE landing__outcome = 'Success (ground pad)'

 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:3119
8/bludb
Done.
```

1
2015-12-22

Keyword: MIN(DATE) displays Oldest date from the record which is formatted in date and time stamped

Successful Drone Ship Landing with Payload between 4000 and 6000

SQL Query: SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_ BETWEEN 4000 AND 6000 AND landing__outcome = 'Success (drone ship)'

```
%sql SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_ BETWEEN 4000 AND 6000 AND landing__outcome = 'Succ
```

```
 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:3119
8/bludb
Done.
```

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

Keyword: BETWEEN can be used when a specified range or record is required.

Total Number of Successful and Failure Mission Outcomes

SQL Query: SELECT COUNT(*) FROM SPACEXTBL WHERE mission_outcome LIKE '%Success%' OR mission_outcome LIKE '%Failure%’

```
%sql SELECT COUNT(*) FROM SPACEXTBL WHERE mission_outcome LIKE '%Success%' OR mission_outcome LIKE '%Failure%'

 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:3119
8/bludb
Done.
```

1
101

Keyword: WHERE clause can be used in conjunction with LIKE and with wildcard “%” and other logical statements like “AND”, “OR”, “NOT”

Boosters Carried Maximum Payload

SQL Query: SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL)

```
%sql SELECT booster_version FROM SPACEXTBL WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL)
```

```
 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
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

Keyword: MAX() can be used to find the maximum value in the record. This can be used in subqueries too.

2015 Launch Records

SQL Query:

SELECT TO_CHAR(TO_DATE(MONTH("DATE"), 'MM'), 'MONTH') AS MONTH_NAME, \
LANDING__OUTCOME AS landing__outcome, \
booster_version AS booster_version, \
launch_site AS launch_site \
FROM SPACEXTBL WHERE landing__outcome = 'Failure (drone ship)' AND "DATE" LIKE '%2015%'

```
%sql SELECT TO_CHAR(TO_DATE(MONTH("DATE"), 'MM'), 'MONTH') AS MONTH_NAME, \
    LANDING__OUTCOME AS landing__outcome, \
    booster_version AS booster_version, \
    launch_site AS launch_site \
    FROM SPACEXTBL WHERE landing__outcome = 'Failure (drone ship)' AND "DATE" LIKE '%2015%'
```

```
 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

month_name	landing__outcome	booster_version	launch_site
JANUARY	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
APRIL	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

WHERE clause, LIKE, AND, and BETWEEN 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 Query:

SELECT "DATE", COUNT(landing__outcome) as COUNT FROM SPACEXTBL \
WHERE "DATE" BETWEEN '2010-06-04' and '2017-03-20' AND landing__outcome LIKE '%Success%' \
GROUP BY "DATE" \
ORDER BY COUNT(landing__outcome) DESC

```
%sql SELECT "DATE", COUNT(landing__outcome) as COUNT FROM SPACEXTBL \
    WHERE "DATE" BETWEEN '2010-06-04' and '2017-03-20' AND landing__outcome LIKE '%Success%' \
    GROUP BY "DATE" \
    ORDER BY COUNT(landing__outcome) DESC
```

```
 * ibm_db_sa://blm84826:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqblod8lcg.databases.appdomain.cloud:3119
8/bludb
Done.
```

DATE	COUNT
2015-12-22	1
2016-04-08	1
2016-05-06	1
2016-05-27	1
2016-07-18	1
2016-08-14	1
2017-01-14	1
2017-02-19	1

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

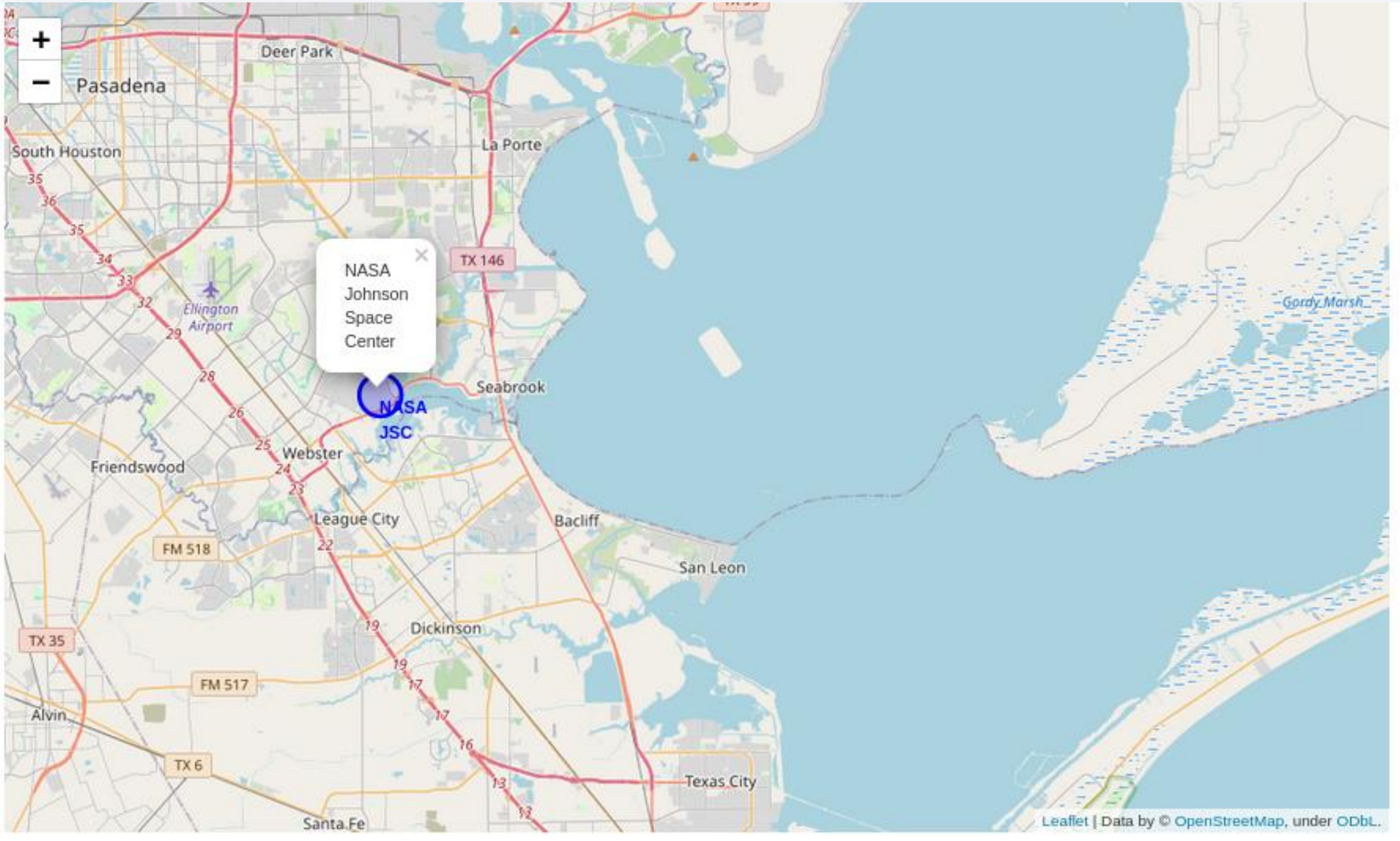
GROUP BY clause to group the landing outcomes

ORDER BY clause to order the grouped landing outcome in descending order.

Section 3

Launch Sites Proximities Analysis

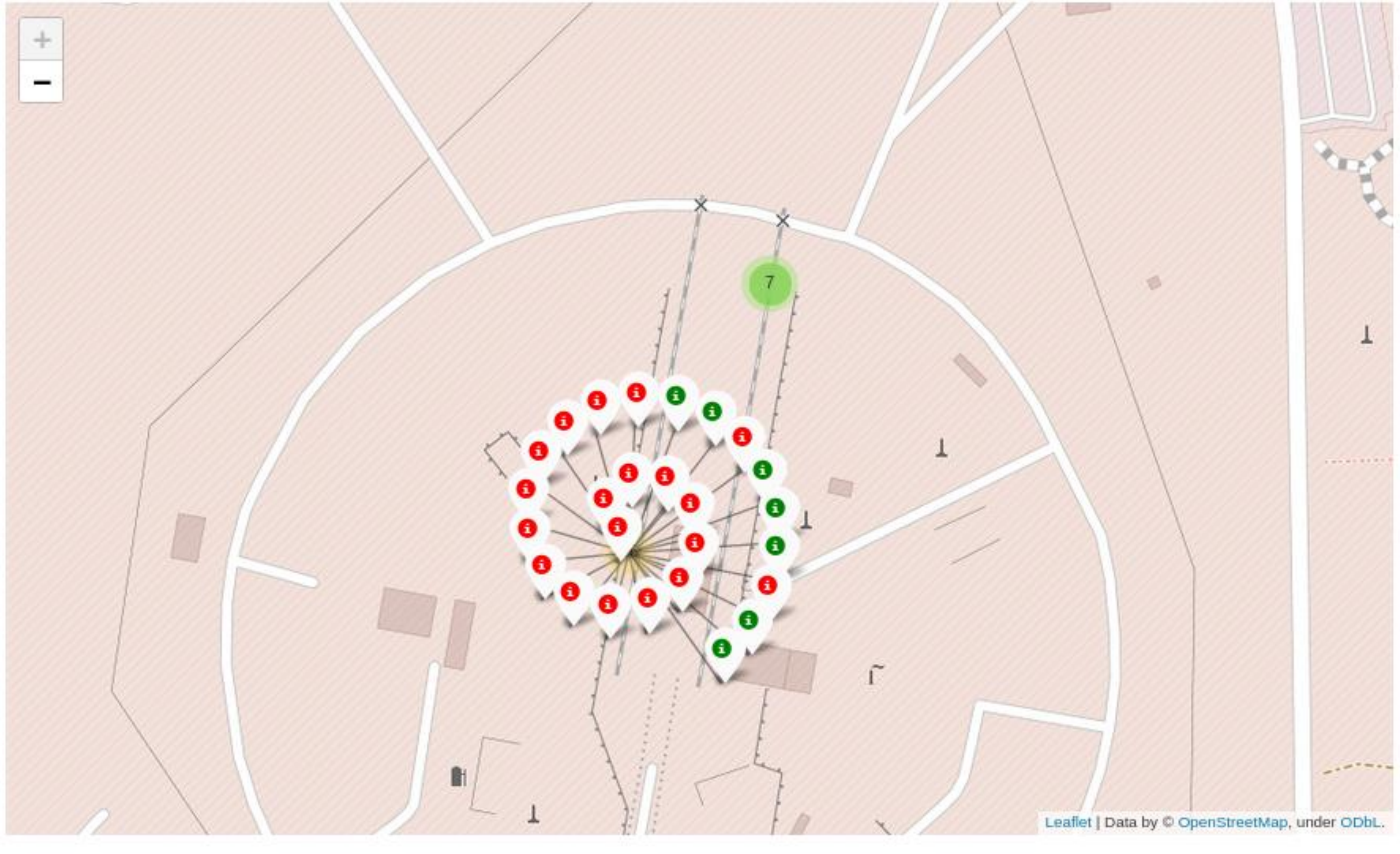
Location of NASA Johnson Space Center



All launch sites

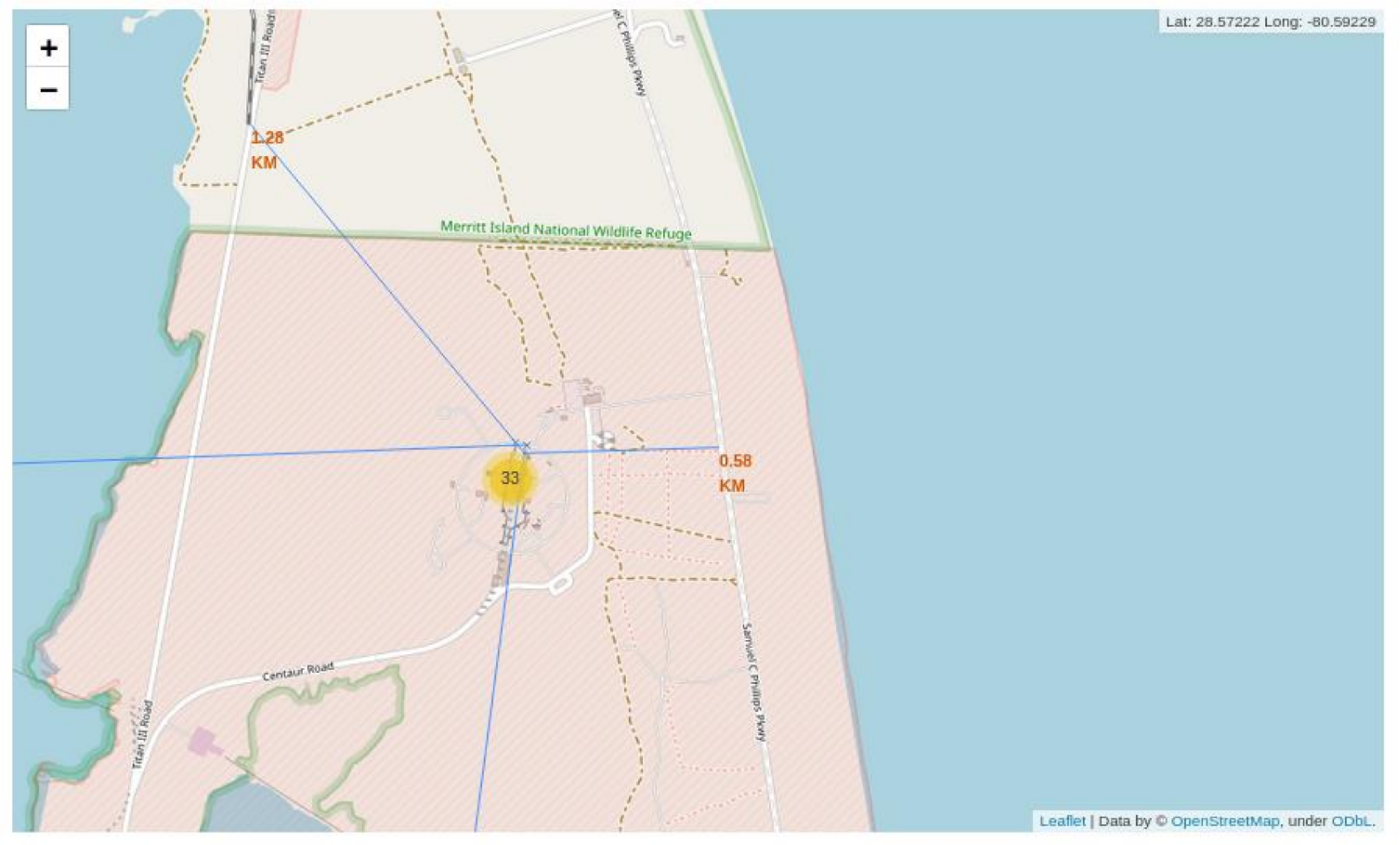
- Circle added for each launch site in data frame launch_sites.
- folium.Circle and folium.Marker were added for each launch site on the site map

Marker for successful and failed launches



- Markers for all launch records.
- Green Marker for successful launch (class=1)
- Red Marker for successful launch (class=0)

Proximity analysis

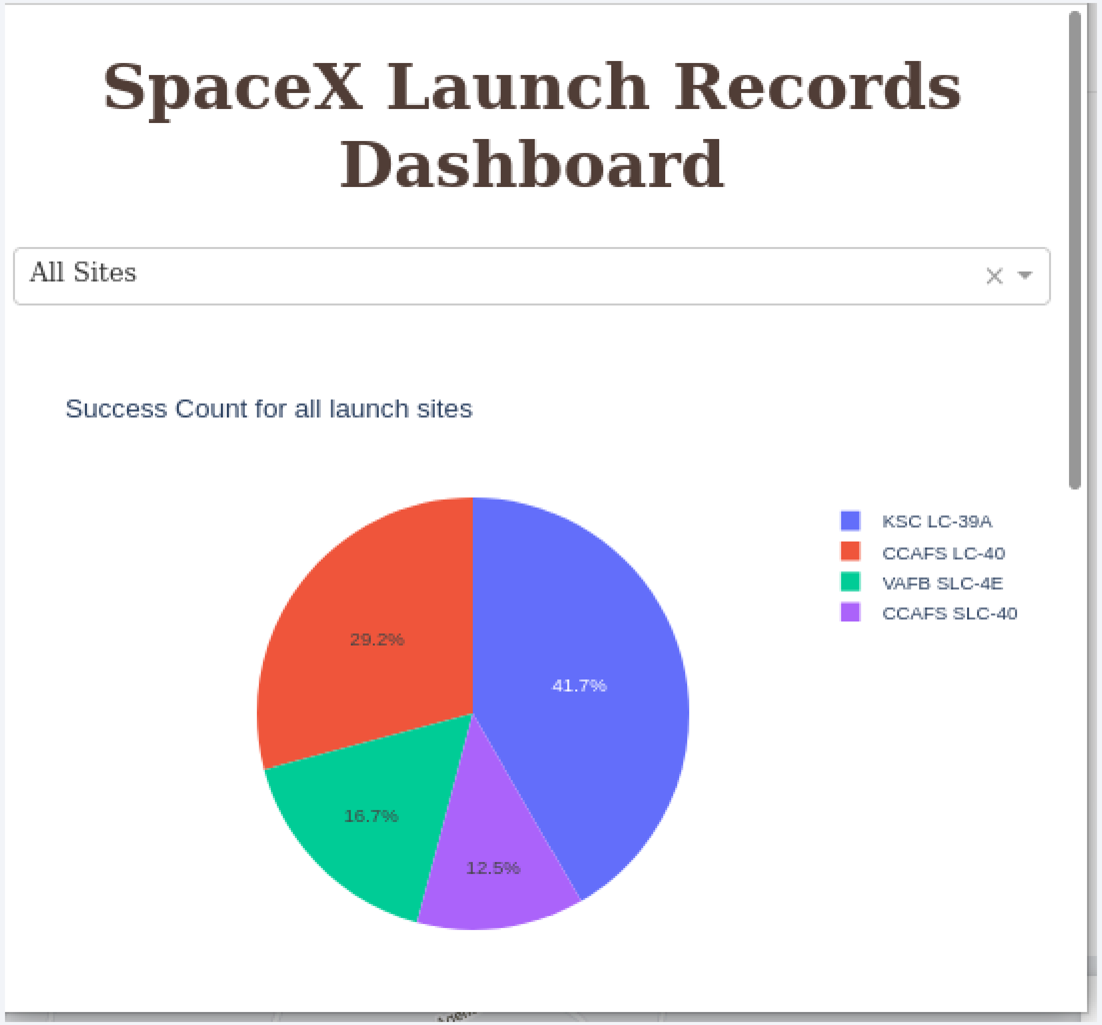


distance_highway = 0.58 km
distance_railroad = 1.28 km
distance_city = 51.43 km

Section 4

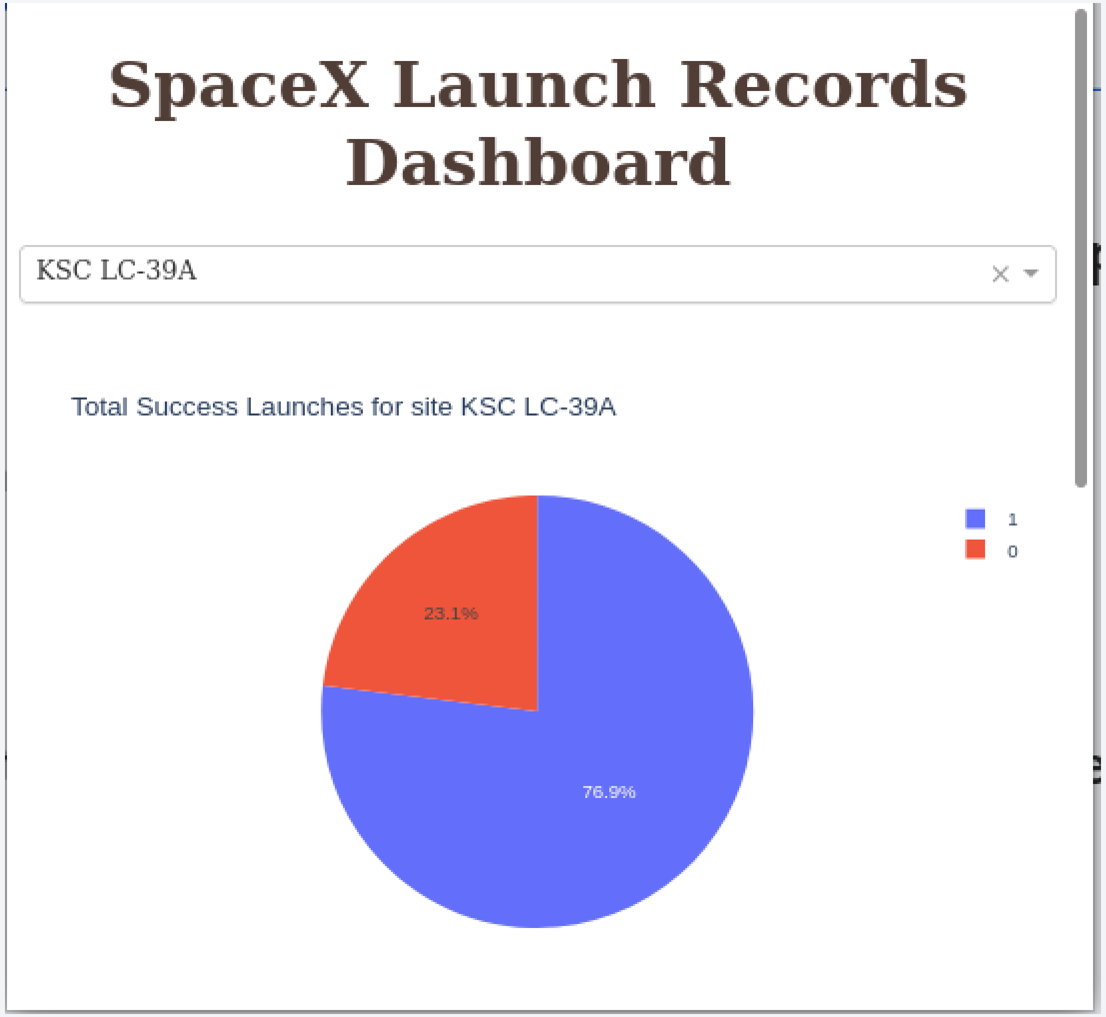
Build a Dashboard with Plotly Dash

Space X Launch Records Dashboard (All sites)



Launch sites	% success
CCAFS LC-40 (Cape Canaveral Space Launch Complex 40)	29.2
CCAFS SLC-40 (Cape Canaveral Space Launch Complex 40)	12.5
KSC LC-39A (Kennedy Space Center Launch Complex 39)	41.7
VAFB SLC-4E (Vandenberg Space Launch Complex 4)	16.7

Space X Launch Records Dashboard (Highest success ratio)



KSC LC-39A (Kennedy Space Center Launch Complex 39) have highest success rate of launching

Payload Mass vs Success rate

Low Weighted Payload 0kg – 4000kg

Higher Weighted Payload 4000kg and above

Higher success rate for the low weighed payloads than the heavier ones.

Section 5

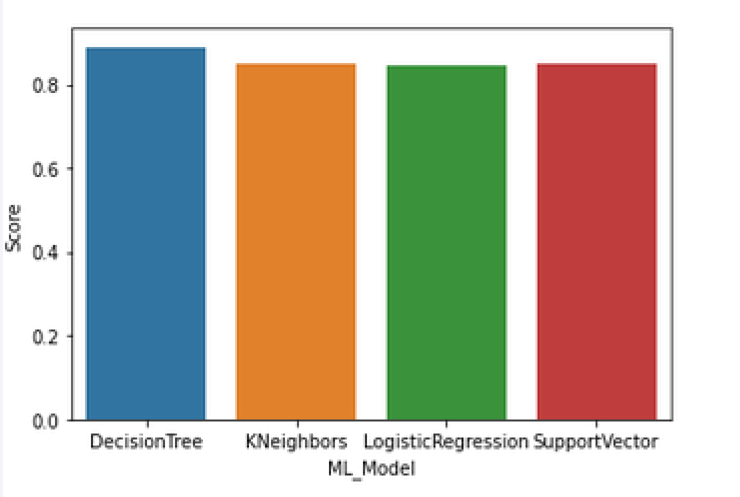
Predictive Analysis (Classification)

Classification Accuracy

```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

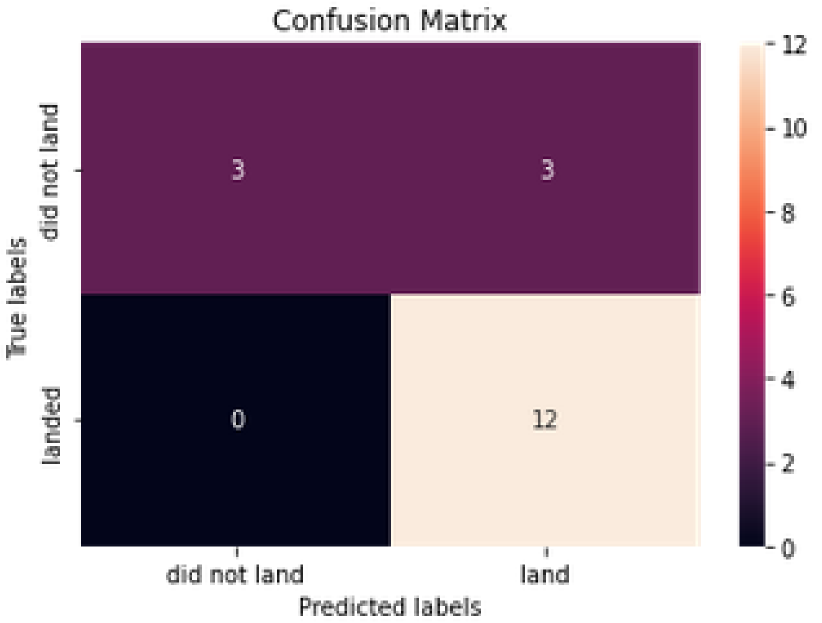
```
Best model is DecisionTree with a score of 0.8892857142857145
Best params is : {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_s
plit': 10, 'splitter': 'random'}
```

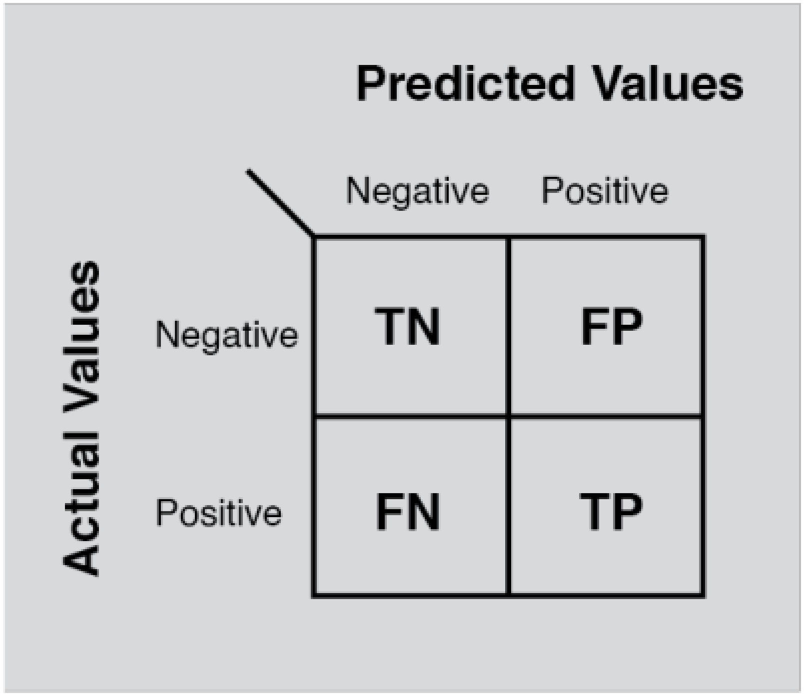


The decision tree classifier is the model with the highest classification accuracy

Confusion Matrix

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





From Confusion Matrix, The ML model correctly predicted the successful landing which is True Positive.

Conclusions

- ES-L1, GEO, HEO, SSO, VLEO had the most success rate
- Vehicles launched in LEO and VLEO orbit have higher success is related to the number of flights
- Heavy payloads have the successful landing for PO, LEO and ISS orbits
- Success rate since 2013 kept till 2020 is in increasing trend
- KSC LC-39A (Kennedy Space Center Launch Complex 39) have highest success rate than the other launch sites
- The decision tree classifier is the model with the highest classification accuracy with score of 0.889285

Appendix

- Github reposistory for SpaceX Capstone Project
  - ✓ https://github.com/rajeshprasanth/Applied_Data_Science_Capstone

- SpaceX REST API
  - ✓ https://api.spacexdata.com/v4/launches/past

- SpaceX Wikipedia page
  - ✓ https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"

Thank you!