IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Elena Marukhlenko
05/25/2023

Outline

Executive Summary

Overall, the project provides a practical and real-world example of using data science to solve a business problem in the commercial space industry.

To analyze the data used the predictive analysis based on the classification models cluster:

- used the categorical variable as the prediction target to assign it to a specific category based on a set of features;
- used the visualization, SQL queries and Folium map to determine the effect of the features on the prediction target;
- used the numerical measures to evaluate different models.

Based on the above processes performed, have the results:

- all classification methods using test data are equal;
- the accuracy measure is realistic, so we can use our models for prediction.

Introduction

- Project background and context

  We have the following problem. Our young rocket company Space Y, which would like to compete with the successful company SpaceX, is faced with the task of calculating the cost of launching rockets. It is known, that the launch of the SpaceX Falcon 9 rocket can cost 62 million dollars. While other suppliers cost more than $165 million each to launch rockets. Much of the savings is because SpaceX can reuse the first stage, which is the most costly of all rocket launch costs.

- Problems we want to find answers

  Landing the first stage of the Falcon 9 is not always successful. Therefore, if we can determine if the first stage will land, we can determine the launch cost. So we’re going predict, whether the first stage of the Falcon 9 will land successfully.

Successful landing Falcon9

Section 1

Methodology

Methodology

Executive Summary

- Data collection:
  1. Using a GET request method to the SpaceX API to extract information about the launch data.
  2. Using Web scraping method to extract records from Wikipedia.
- Data wrangling:

  Dealing with missing values and categorical variables, performing Exploratory Data Analysis (EDA) and determining Training Labels.
- EDA using visualization and SQL:
  1. Created and analyzed the plots of the relationship between the different variables and goal.
  2. Loading the dataset from .csv file into IBM Db2 database and executing SQL queries for analysis.
- Interactive visual analytics:

  Building map and dashboard using Folium and Plotly Dash.
- Predictive analysis:

  Building, tuning and evaluating classification models using a machine learning pipeline.

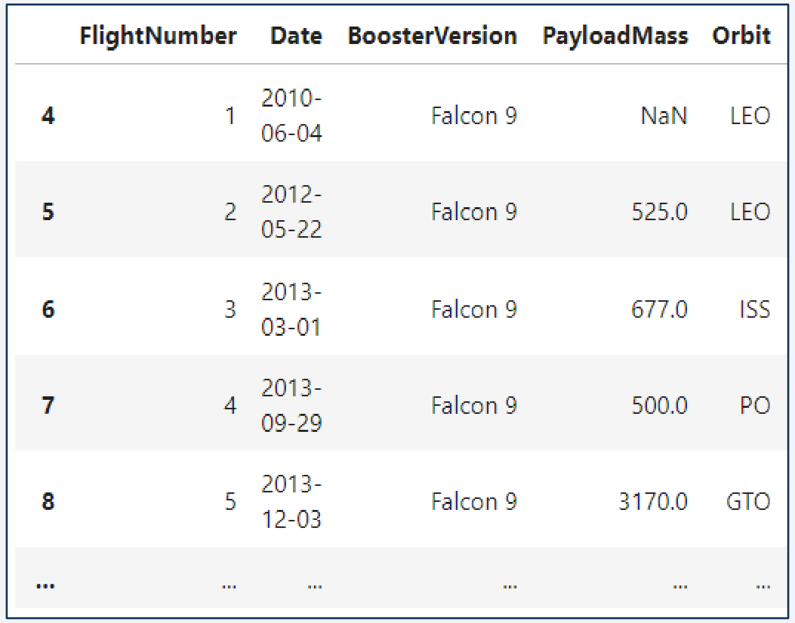
Data Collection

1. Used a GET request to the SpaceX API to extract information in the launch data: requests.get(https://api.spacexdata.com/v4/...)

2. To get more features about the launches opened source REST API for launch, rocket core, capsule, starlink, launchpad and landing pad data.

3. Decoded a response content as a JSON using .json() and converted this JSON to a Pandas dataframe using the json_normalize() function.

4. Filtered the dataframe to only include Falcon 9 launches.



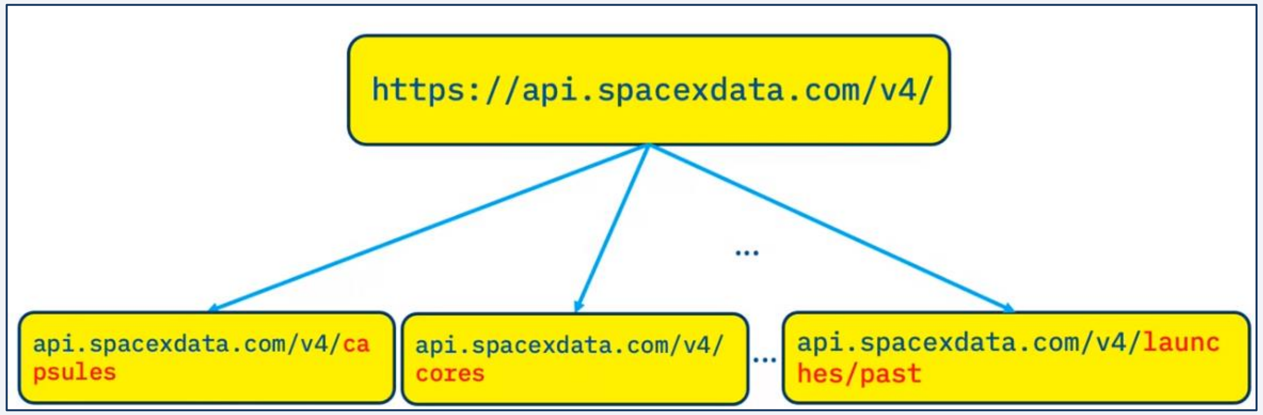
	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit
4	1	2010-06-04	Falcon 9	NaN	LEO
5	2	2012-05-22	Falcon 9	525.0	LEO
6	3	2013-03-01	Falcon 9	677.0	ISS
7	4	2013-09-29	Falcon 9	500.0	PO
8	5	2013-12-03	Falcon 9	3170.0	GTO
...	...	...	...	...	...

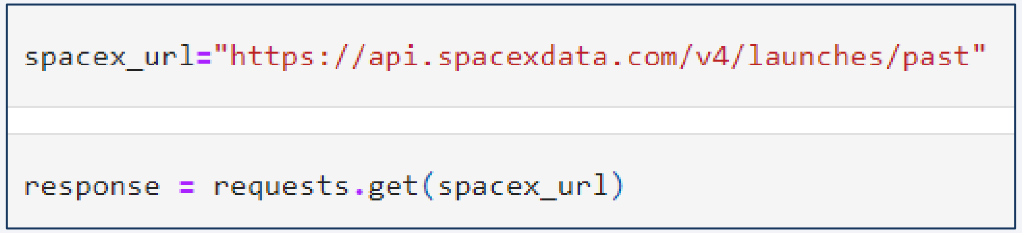
The part of the filtered dataframe

Data Collection – SpaceX API

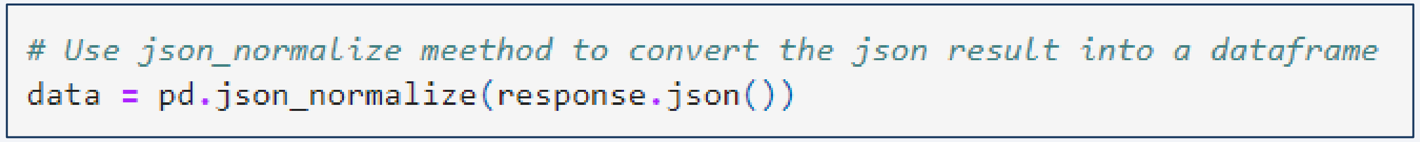
Lab1_Data_Collection

(the link to Jupyter Notebook from my GitHub repository)


https://api.spacexdata.com/v4/
...
api.spacexdata.com/v4/capsules
api.spacexdata.com/v4/cores
...
api.spacexdata.com/v4/launches/past



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

response = requests.get(spacex_url)
```



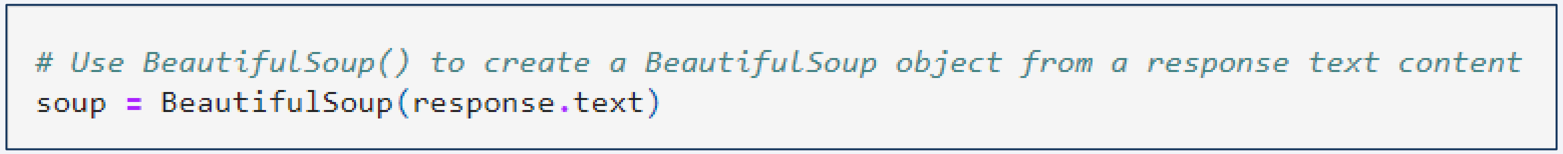
```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Data Collection - Scraping

Lab2_Web_Scraping

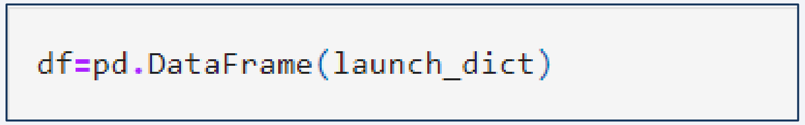
(the link to Jupyter Notebook from my GitHub repository)

1. Used a GET request and created a BeautifulSoup object to extract a Falcon 9 launch records HTML table from Wikipedia: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches



```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.text)
```

2. Extracted all column names from the HTML table, did the parsing process and created the dataframe from the extracted data list using .find_all(), .extract_column_from_header(), pd.DataFrame() functions.



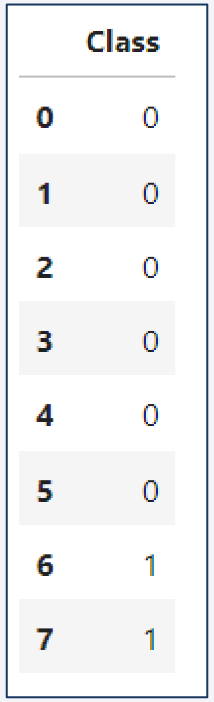
```
df=pd.DataFrame(launch_dict)
```

Data Wrangling

[Lab3_Data_Wrangling]

(the link to Jupyter Notebook from my GitHub repository)

1. Missing values of "PayLoadMass" were replaced by the mean value using:
   - .isnull().sum(), .mean() and .replace(np.nan, mean, inplace=True) functions.
2. Calculated and analyzed data using value_counts():
   - the number of launches on each "LaunchSite" (the location of launch);
   - the number of each orbit "Orbit" (each launch aims to an dedicated orbit);
   - the number of mission outcome per orbit type "Outcome" (to determine if it was successfully landed or not);
3. Created a landing outcome label (the classification variable) for further prediction:
   - value: 0 – failed outcome, 1 – successful outcome;
   - name: "Class".



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

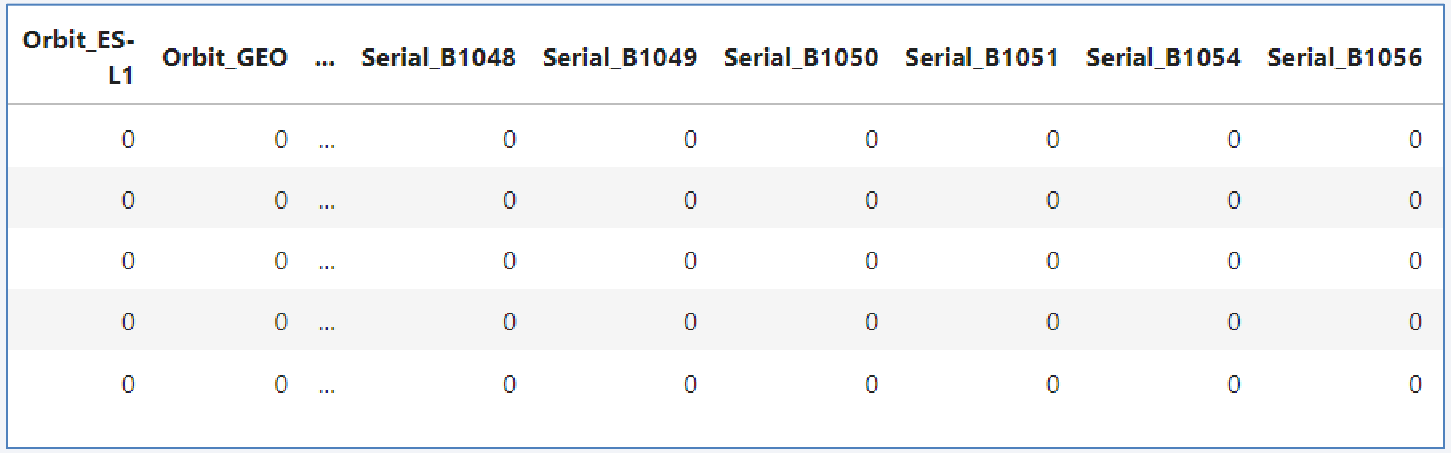
Classification variable

Data Wrangling

Lab3_Data_Wrangling

(the link to Jupyter Notebook from my GitHub repository)

4. Created dummy variables to categorical columns and casted to type "float64" using .get_dummies() and .astype(float) functions for:
   - "Orbits", "LaunchSite", "LandingPad", and "Serial".



Orbit_ES-L1	Orbit_GEO	...	Serial_B1048	Serial_B1049	Serial_B1050	Serial_B1051	Serial_B1054	Serial_B1056
0	0	...	0	0	0	0	0	0
0	0	...	0	0	0	0	0	0
0	0	...	0	0	0	0	0	0
0	0	...	0	0	0	0	0	0
0	0	...	0	0	0	0	0	0

New categorical columns

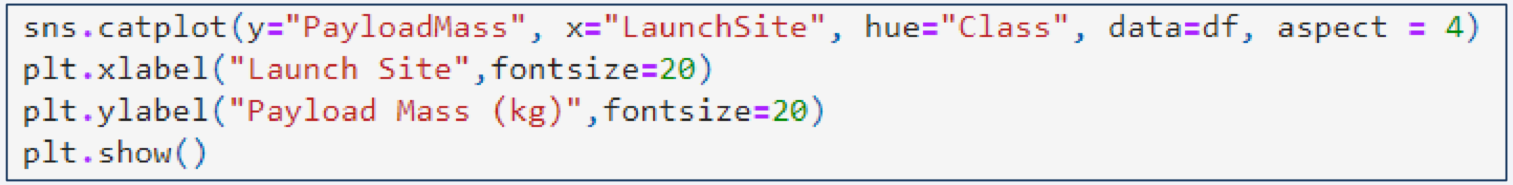
EDA with Data Visualization

Lab5_Data_Visualization

(the link to Jupyter Notebook from my GitHub repository)

Created and analyzed the plots using Pandas, Seaborn and Matplotlib libraries:

- a scatter point chart using .catplot() to show, how the different variables would affect the launch outcome (FlightNumber, PayloadMass, LaunchSite). For example:



```
sns.catplot(y="PayloadMass", x="LaunchSite", hue="Class", data=df, aspect = 4)
plt.xlabel("Launch Site",fontsize=20)
plt.ylabel("Payload Mass (kg)",fontsize=20)
plt.show()
```

- a bar chart using .plot(kind='bar') to show the relationship between success rate and orbit type;
- a line chart using .lineplot() to show the launch success yearly trend.

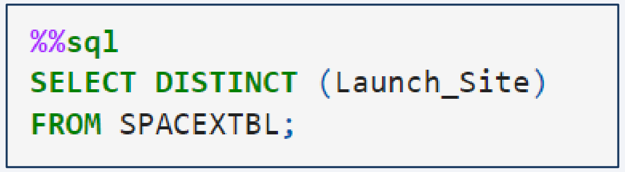
EDA with SQL

Lab4_SQL

(the link to Jupyter Notebook from my GitHub repository)

1. Loaded the dataset from .csv file into the corresponding table in a IBM Db2 database using .connect(), .cursor() and .read_csv() functions.

2. Executed SQL queries for analysis. For example obtained:
   - list of the names of the unique launch sites in the space mission;
   - list the date when the first successful landing outcome in ground pad was achieved;
   - list the total number of successful and failure mission outcomes;
   - list the names of the booster_versions which have carried the maximum payload mass;
   - ranked list of successful landing_outcomes and so on…



```
%%sql
SELECT DISTINCT (Launch_Site)
FROM SPACEXTBL;
```

Build an Interactive Map with Folium

Lab6_Interactive_map

(the link to Jupyter Notebook from my GitHub repository)

1. Created a Folium map object to find some geographical patterns about launch sites using Folium library and .map() function.
2. Marked all launch sites on the map using geographic coordinates, Markers and Circles objects with .circle(), .marker() functions:
   - they were used to add a highlighted circle area with a text label.
3. Marked the success/failed launches for each site on the map using MarkerCluster object with .markerCluster() function:
   - it was used to simplify a map containing many markers having the same coordinate.
4. Calculated and displayed the distance between a launch site to its proximities (the closest coastline, city and e.g.) with .MousePosition() and .PolyLine() functions.

Build a Dashboard with Plotly Dash

Lab7_Dashboard

(the link to Jupyter Notebook from my GitHub repository)

Built a web-based dashboard application for users to perform interactive visual analytics in real-time using Cloud IDE, Plotly Dash (Python framework) and its functions. Added:

- a Launch Site Drop-down Input Component to select one specific site or All sites using .Dropdown() function;
- a Pie chart visualizing launch success counts based on selected Site Drop-down using .Graph() function and @app.callback() function decorator;
- a Range Slider to Select Payload using .RangeSlider() function;
- a Scatter chart showing the correlation between payload and launch success based on selected Site Drop-down using .Graph() function and @app.callback() function decorator.

Predictive Analysis (Classification)

Lab8_Machine_Learning_Prediction

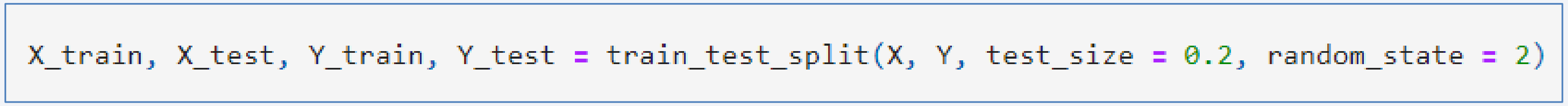
(the link to Jupyter Notebook from my GitHub repository)

Created a machine learning pipeline for our prediction:

1. Created a NumPy array from the column “Class” using .to_numpy() method and assigned it to the variable Y (the goal for prediction).

2. Standardized the data without the column “Class” in variable X using preprocessing.StandardScaler(), .fit_transform() from Sklearn library.

3. Split the data X and Y into training and test data using .train_test_split function:



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

Predictive Analysis (Classification)

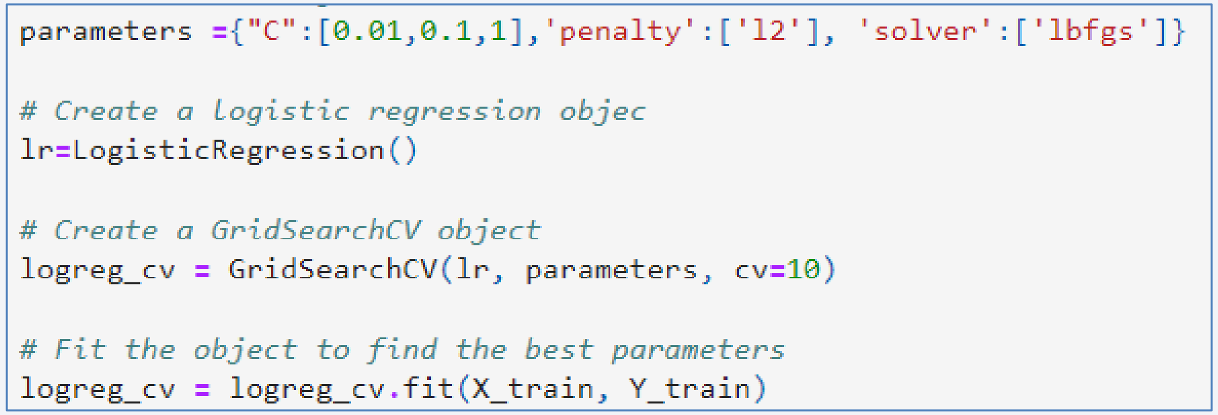
Lab8_Machine_Learning_Prediction

(the link to Jupyter Notebook from my GitHub repository)

Created and evaluated Classification models for our prediction:

SVM (support vector machine), Decision Tree Classifier, KNN (k nearest neighbors) and Logistic Regression.

1. Created an Object for each method using Sklearn library.
2. Created a GridSearchCV object and fitted it. Example for Logistic Regression:



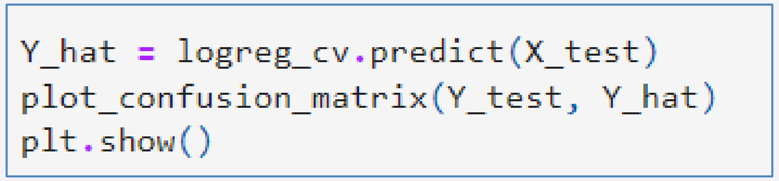
```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}

# Create a logistic regression objec
lr=LogisticRegression()

# Create a GridSearchCV object
logreg_cv = GridSearchCV(lr, parameters, cv=10)

# Fit the object to find the best parameters
logreg_cv = logreg_cv.fit(X_train, Y_train)
```

Predictive Analysis (Classification)

Lab8_Machine_Learning_Prediction

(the link to Jupyter Notebook from my GitHub repository)

3. Calculated for GridSearchCV object the best parameters and the accuracy on the validation data using .best_params_ and .best_score_ the data attributes.
4. Calculated the Test Data Accuracy using .score() method.
5. Plotted and examined the Confusion Matrix for each method:



```
Y_hat = logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test, Y_hat)
plt.show()
```

6. Found a method that performs best by comparing the Test Data Accuracies.

Results

Exploratory data analysis (EDA) results:

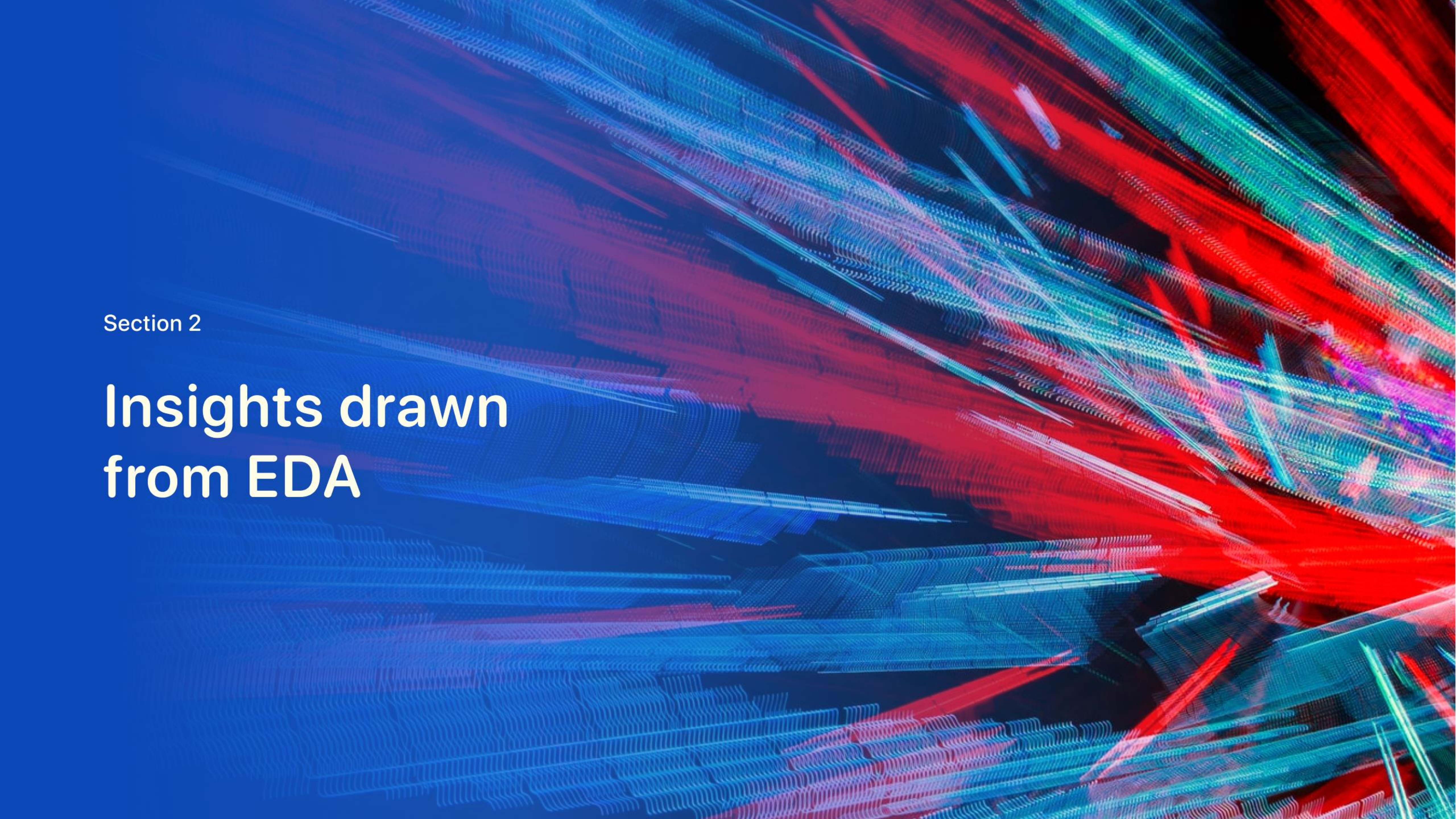
- Based on EDA with Data Visualization and SQL queries we have the selected variables, that affect the launch outcome, which is the prediction target. So we can use them as the test data for the classification models.

Data analysis with Dashboard and Interactive Map:

- Based on Data analysis with Dashboard and Interactive Map we have, that a launch site (location) depends on its proximity to the coastline, city, railway, etc. We learned, how the choice of the launch site affects the launch outcome.

Predictive analysis results:

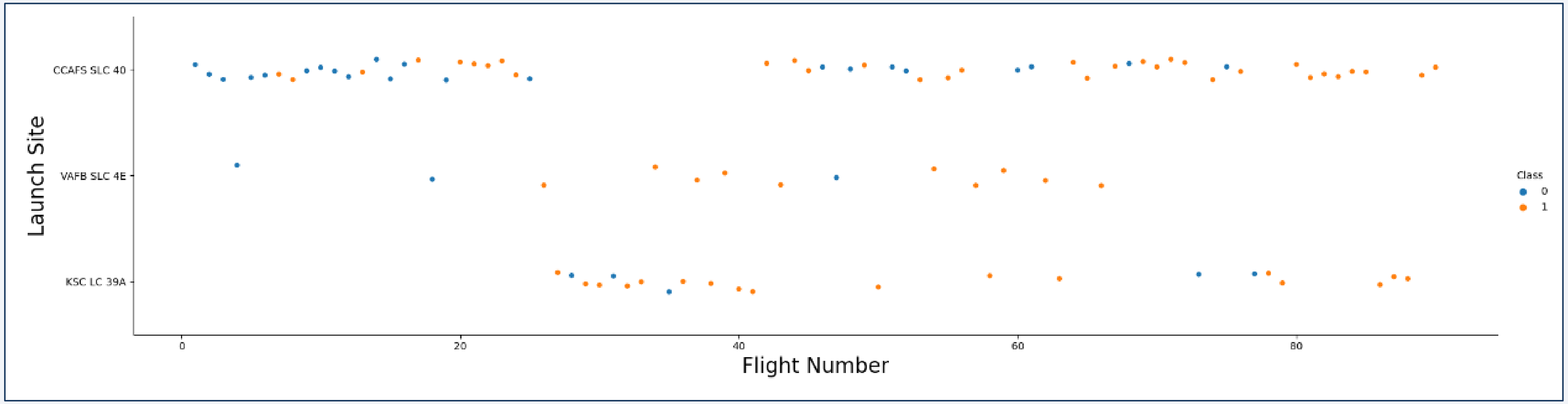
- Based on the Test Data Accuracies we have, that all classification methods using test data are equal: SVM (support vector machine), Decision Tree Classifier, KNN (k nearest neighbors) and Logistic Regression.
- The accuracy measure is realistic, so we can use any model to predict, if the first stage will land.



Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

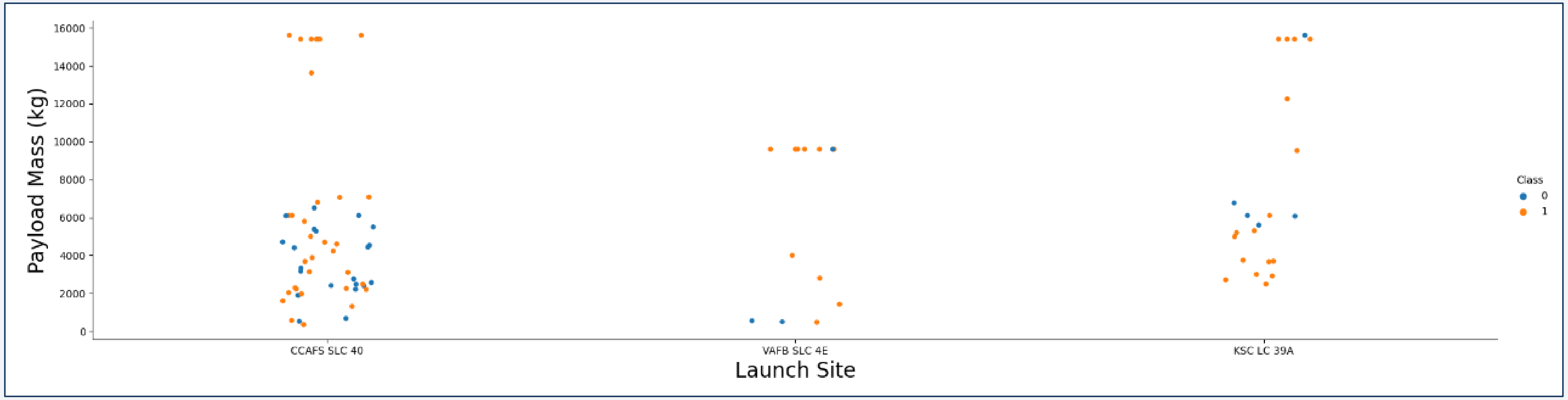


Result:

Launch site CCAFS SLC-40 has the highest number of launches. KSC LC-39A and VAFB SLC 4E have a less number. Therefore, CCAFS SLC-40 has more failures, especially during the first launches.

KSC LC-39A began to be used around the 25th launch. Nowadays, after 70th launch VAFB SLC 4E isn't used at all.

Payload vs. Launch Site



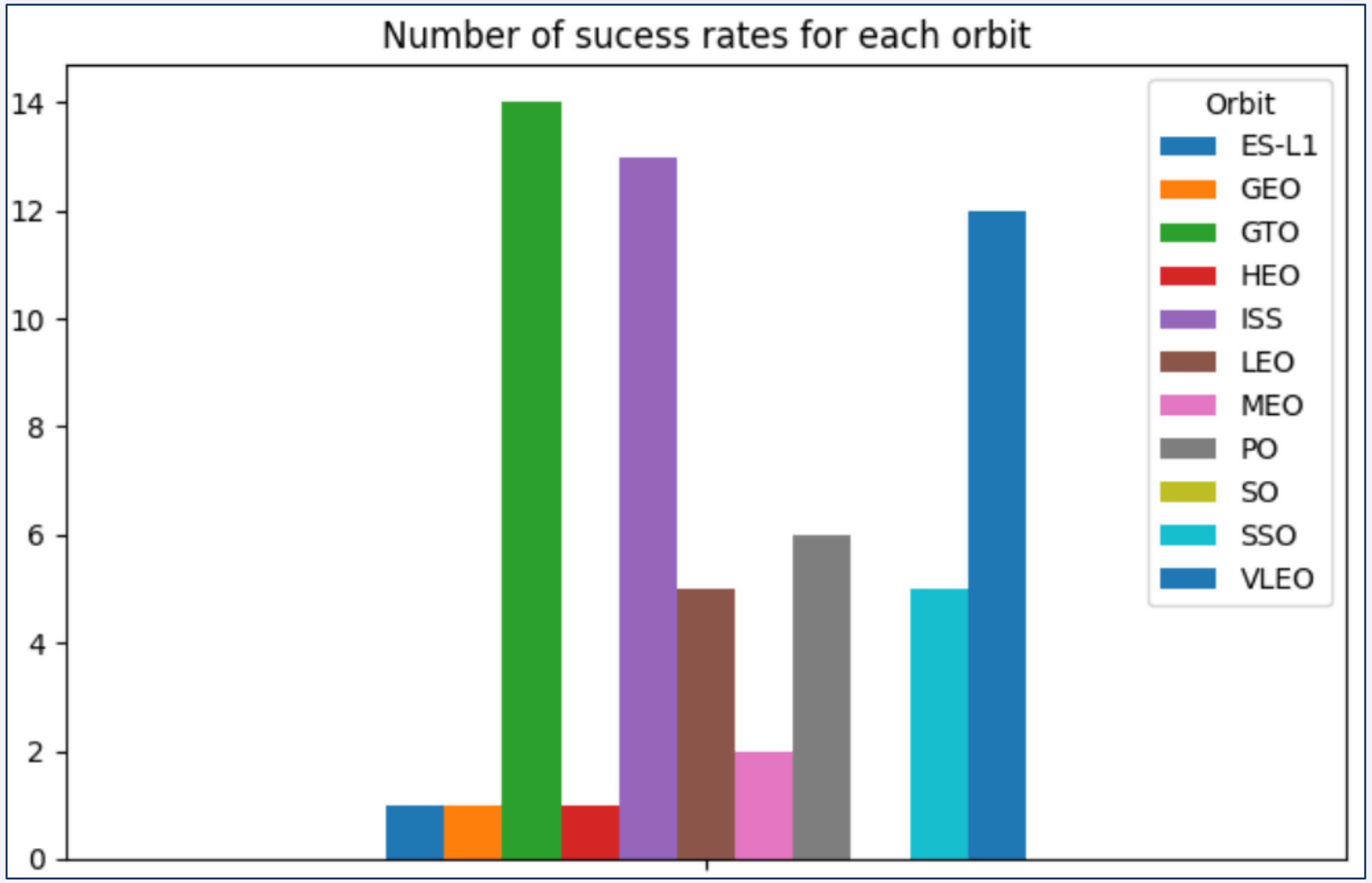
Result:

For the VAFB-SLC 4E launch site there are no rockets launched for heavy payload mass (greater than 10 000).

For the KSC LC 39A launch site there are no rockets launched for light payload mass (less than 2 000).

The rockets with payload mass greater than 8 000 has less failures launches.

Success Rate vs. Orbit Type

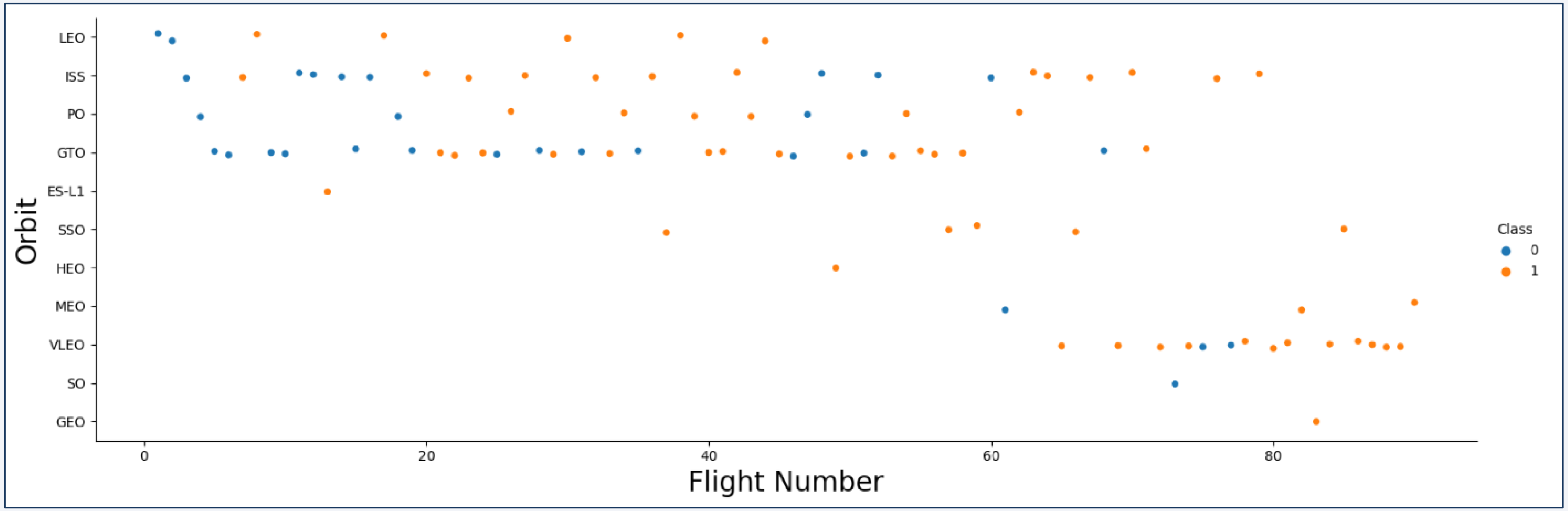


<u>Result:</u>

The orbit like GTO, ISS and VLEO have more success rates, accordin to:

14, 13 and 12 successful launches.

Flight Number vs. Orbit Type

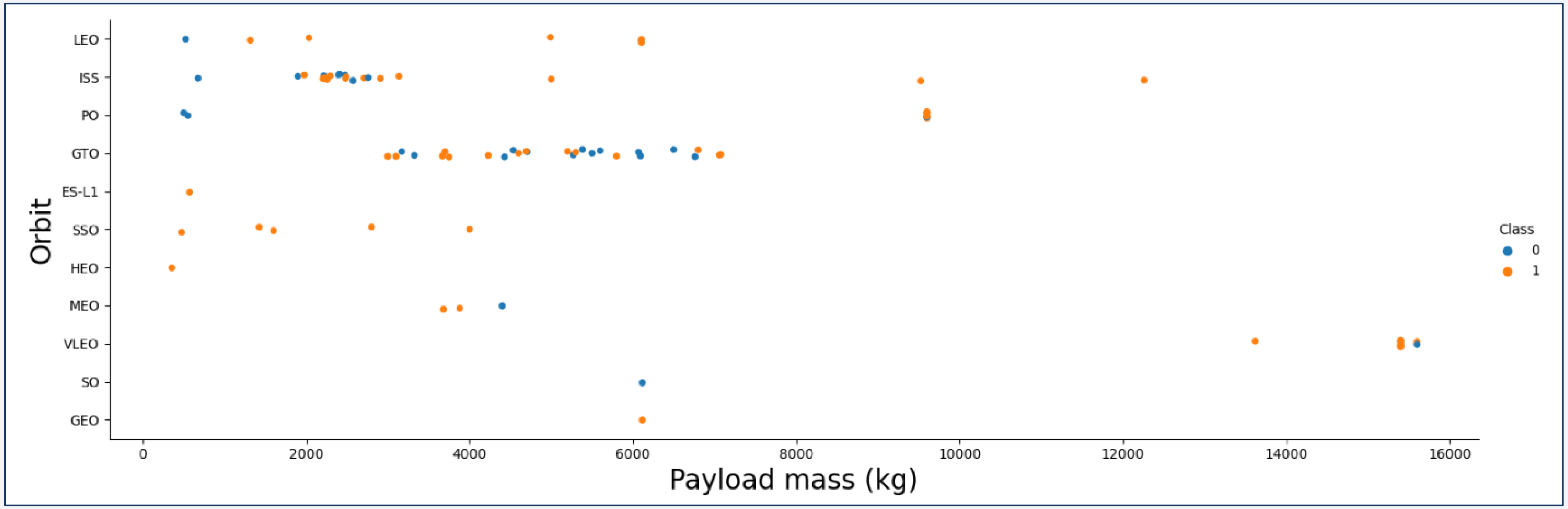


<u>Result:</u>

In LEO orbit the success appears related to the number of flights; on the other hand, in GTO and ISS orbit the success doesn't depend on the number of flights.

VLEO is the popular orbit nowadays, having been used since about the 65th launch. VLEO has the success rate.

Payload vs. Orbit Type

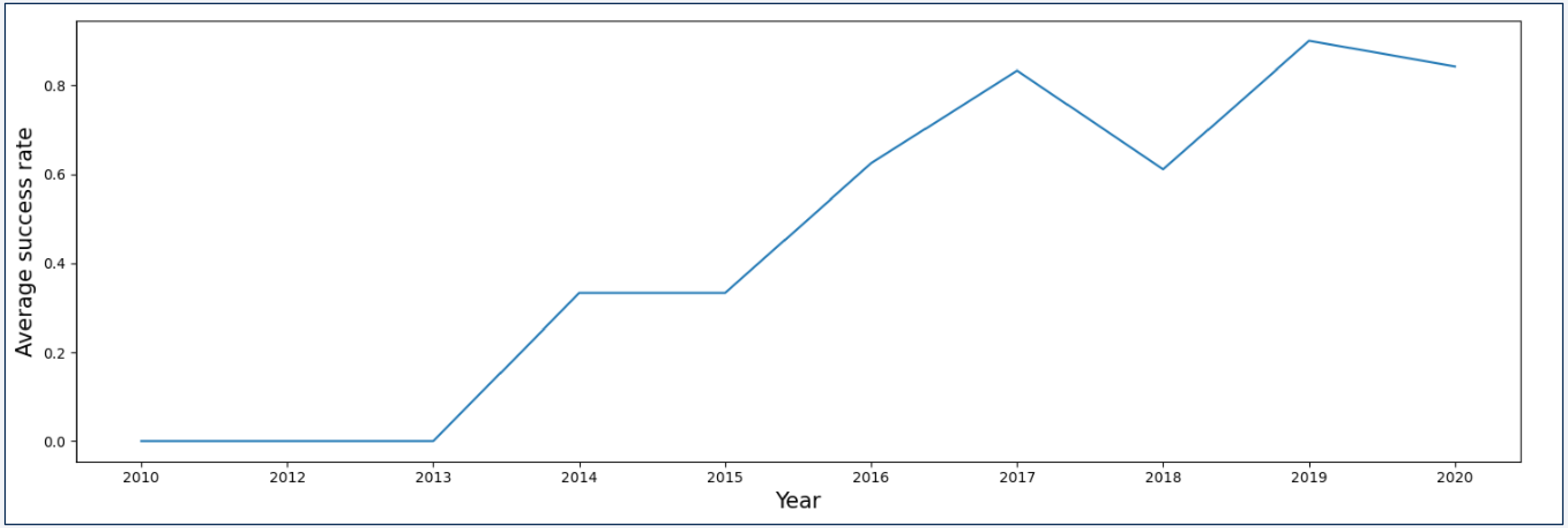


Result:

With heavy Payloads the successful landing are more for PO, LEO and ISS.

However for GTO and VLEO we cannot distinguish this well, because the successful landing and unsuccessful mission are both there here.

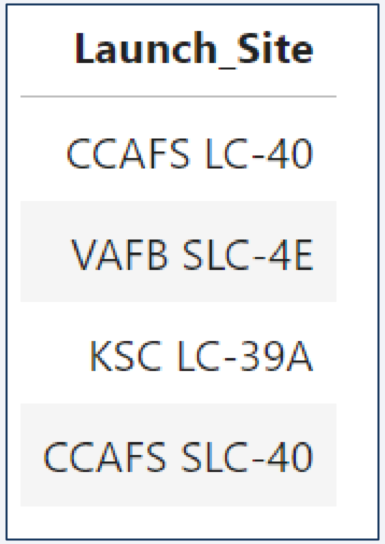
Launch Success Yearly Trend



Result:

The success rate since 2013 year kept increasing till 2020 year.

All Launch Site Names



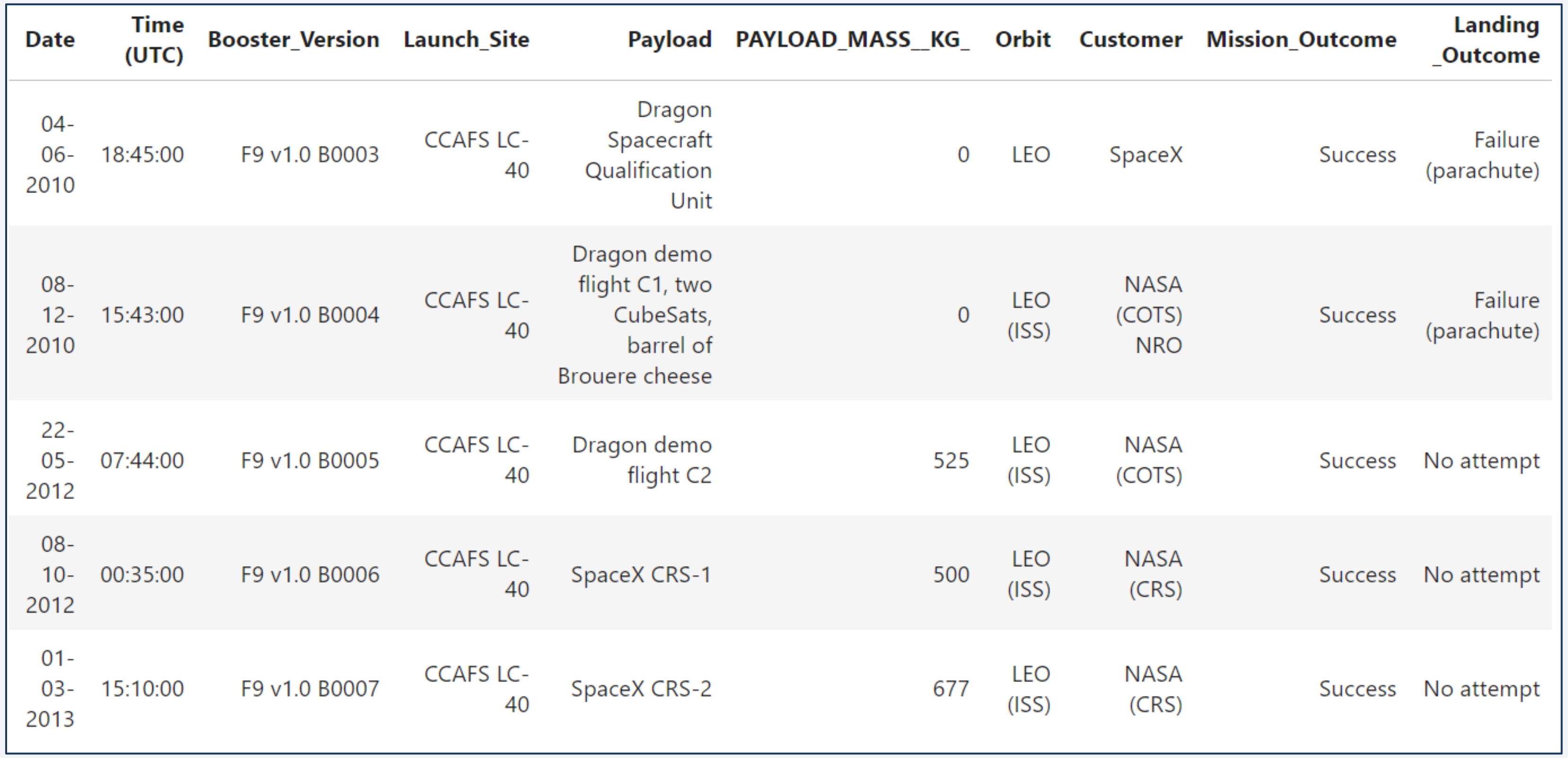
Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Result:

For the future analysis with Interactive Map we need understand, how many launch sites are used in the space mission.

So we have 4 launch sites.

Launch Site Names Begin with 'CCA'

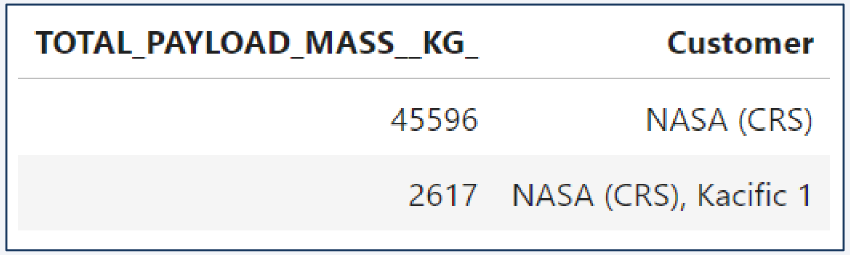


Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

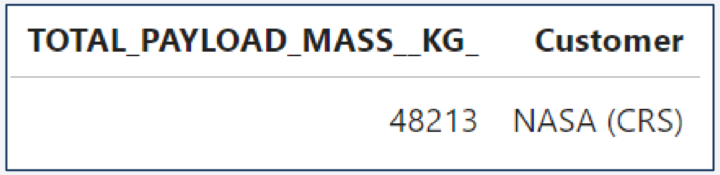
Result:

Checked then records where launch sites begin with the string “CCA”.

Total Payload Mass



TOTAL_PAYLOAD_MASS__KG_	Customer
45596	NASA (CRS)
2617	NASA (CRS), Kacific 1

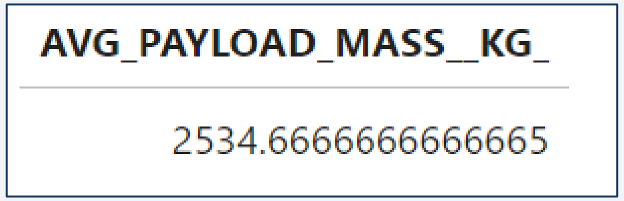


TOTAL_PAYLOAD_MASS__KG_	Customer
48213	NASA (CRS)

Result:

1. The total Payload mass of all boosters by only NASA (CRS) and by NASA with Kacific.
2. The total Payload mass of all boosters by NASA (CRS): single or with Kacific.

Average Payload Mass by F9 v1.1



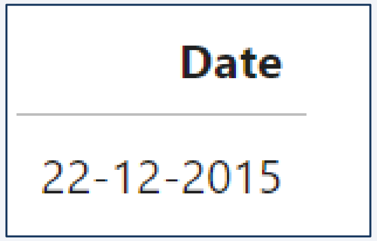
AVG_PAYLOAD_MASS__KG_
2534.6666666666665

<u>Result:</u>

We need to know the average payload mass carried by booster version F9 v1.1.

This information will also be useful for future rocket cost calculations.

First Successful Ground Landing Date



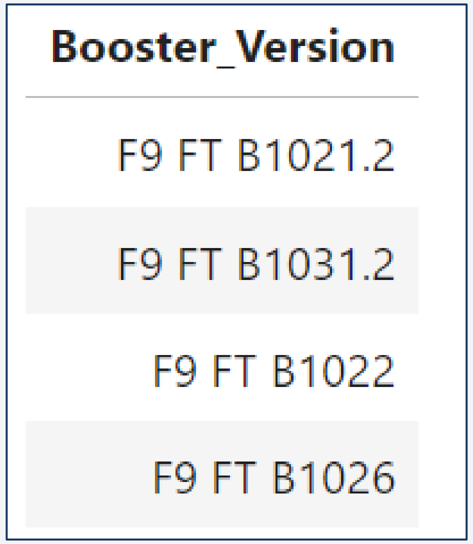
Date
22-12-2015

Result:

We have a data base since 2010 year. We need to know the date, when the first successful landing outcome in ground pad was achieved.

In other words, from what point did successful ground landing begin and what changes were made to this.

Successful Drone Ship Landing with Payload between 4000 and 6000



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

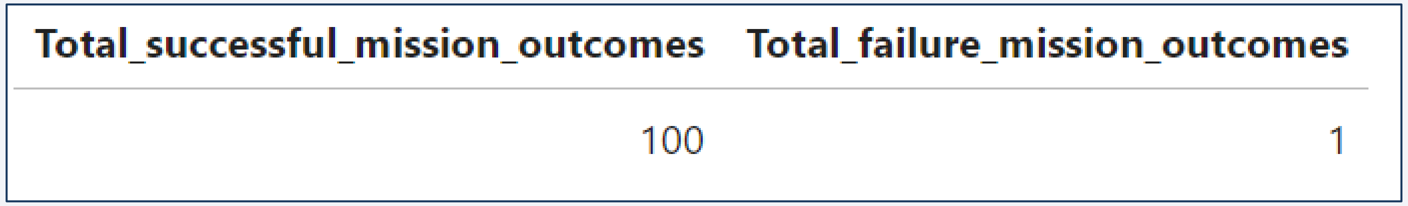
Result:

We are interesting in the boosters, which payload mass greater than 4000 but less than 6000 and have success in drone ship.

So we have 4 booster versions.

This information will also be useful for future rocket cost calculations.

Total Number of Successful and Failure Mission Outcomes



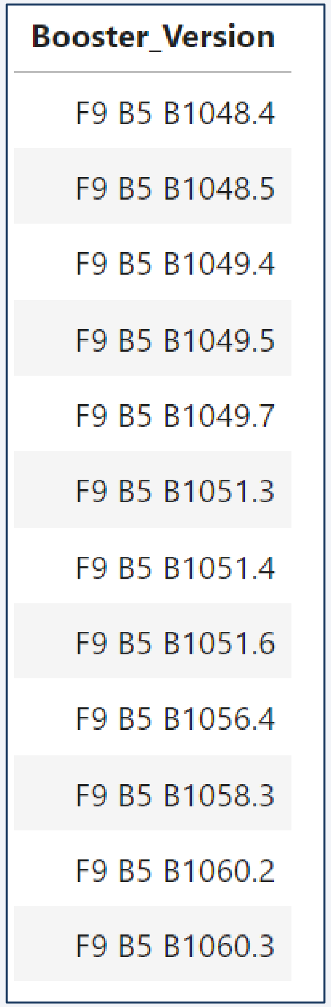
Total_successful_mission_outcomes	Total_failure_mission_outcomes
100	1

Result:

From all data we have the next total number of successful and failure mission outcomes: 100 and 1. It's a great result.

That's why SpaceX is the successful company, which our company would like to compete with.

Boosters Carried Maximum Payload



Booster_Version
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

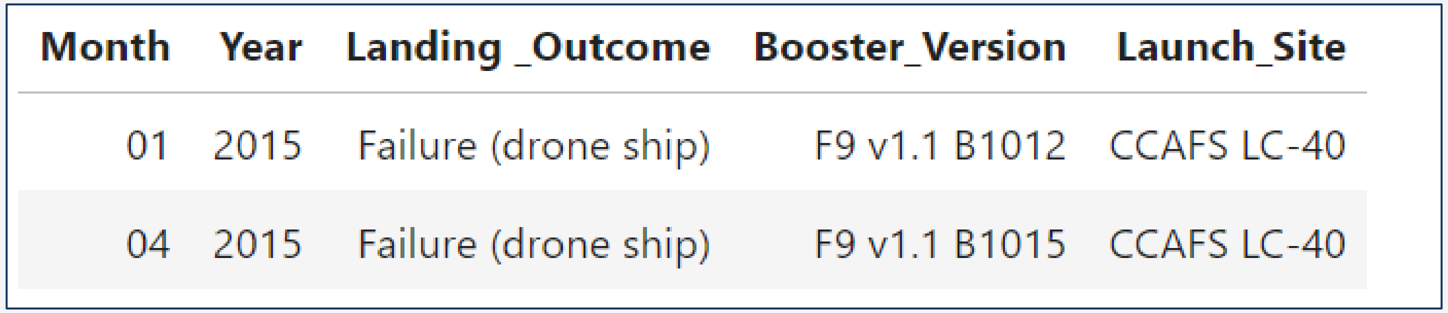
Result:

We are interesting in the boosters, which have carried the maximum payload mass.

So we have many booster versions.

This information will also be useful for future rocket cost calculations.

2015 Launch Records

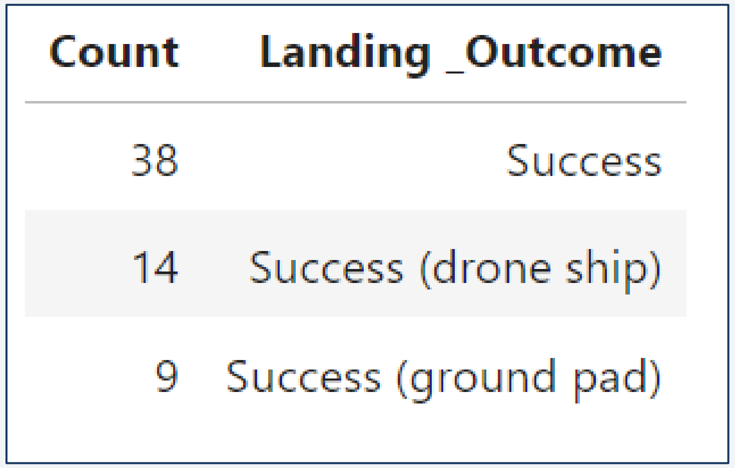


Month	Year	Landing _Outcome	Booster_Version	Launch_Site
01	2015	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	2015	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

Result:

We obtained the list of the records, which display the month names, failure landing outcomes in drone ship, booster versions, launch site for the months in year 2015.

Rank Landing Outcomes



Count	Landing _Outcome
38	Success
14	Success (drone ship)
9	Success (ground pad)

Result:

We are interesting in the rank of the successful landing outcomes.

This information will also be useful for future choice the type of the landing outcome for the rocket launches by our company.

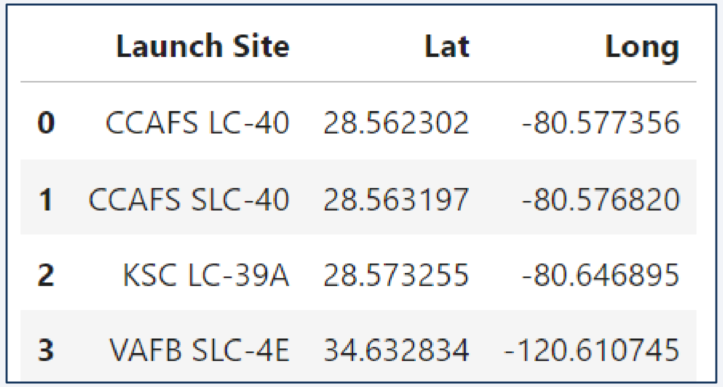
In the first line we have the result “success” without the type of landing outcome. It’s not a full value.

But we couldn't to update the information of the feature “Landing_Outcome”, it’s missing in the original sources.

Section 4

Launch Sites Proximities Analysis

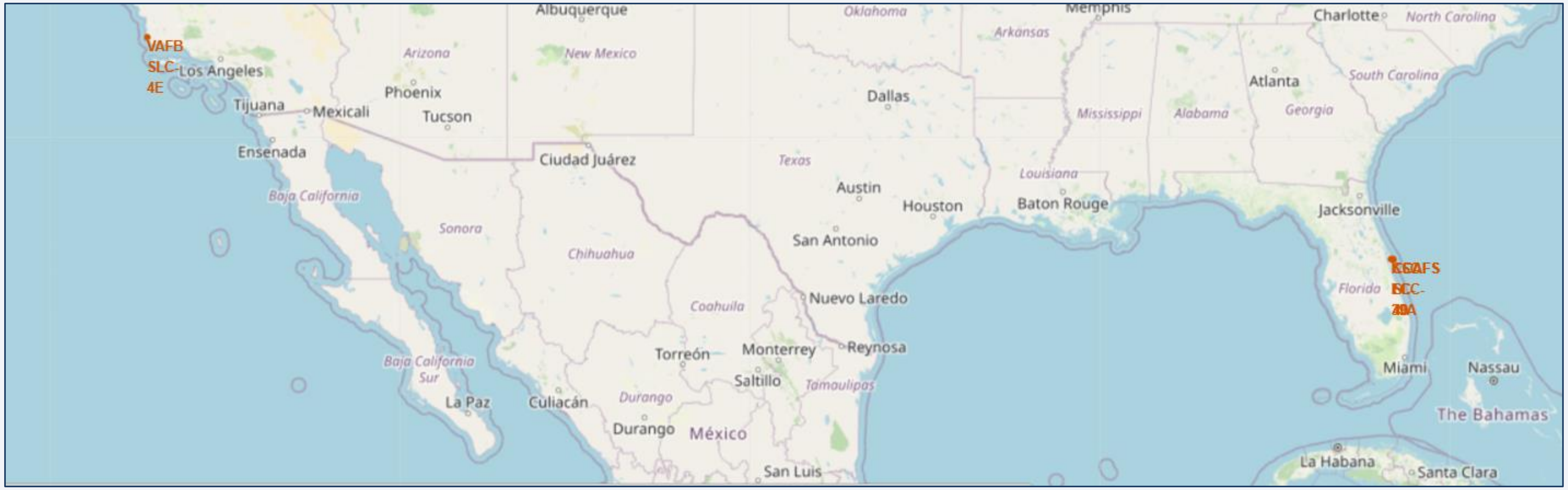
All launch sites locations on the Folium map



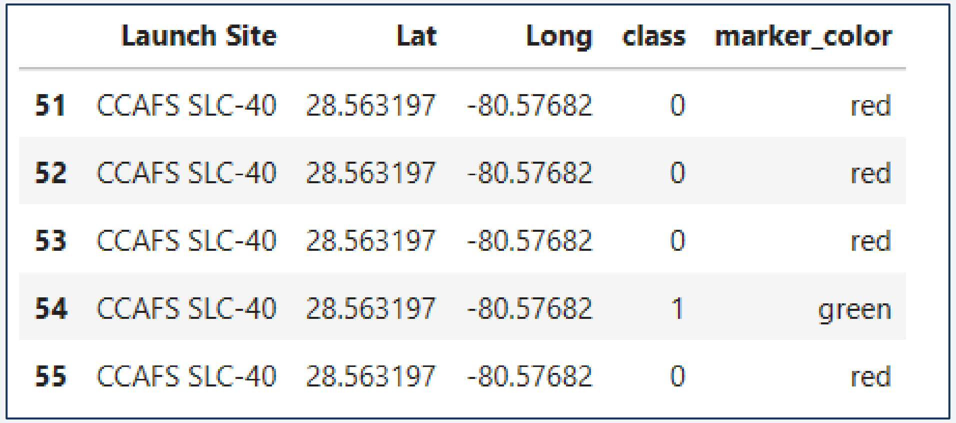
	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610745

<u>Result:</u>

- CCAFS LC-40, CCAFS SLC-40, KSC LC-39A have the close location coordinates in the state Florida, USA.
- VAFB SLC-4E is located in the state California, USA.
- All launch sites are located close to the coastline.



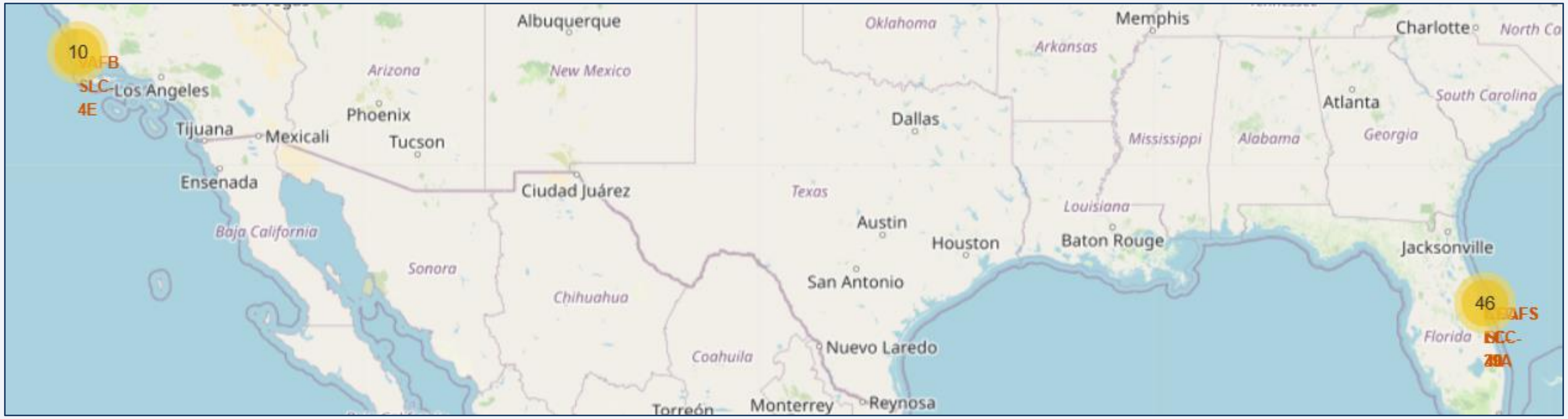
All launch outcomes on the Folium map



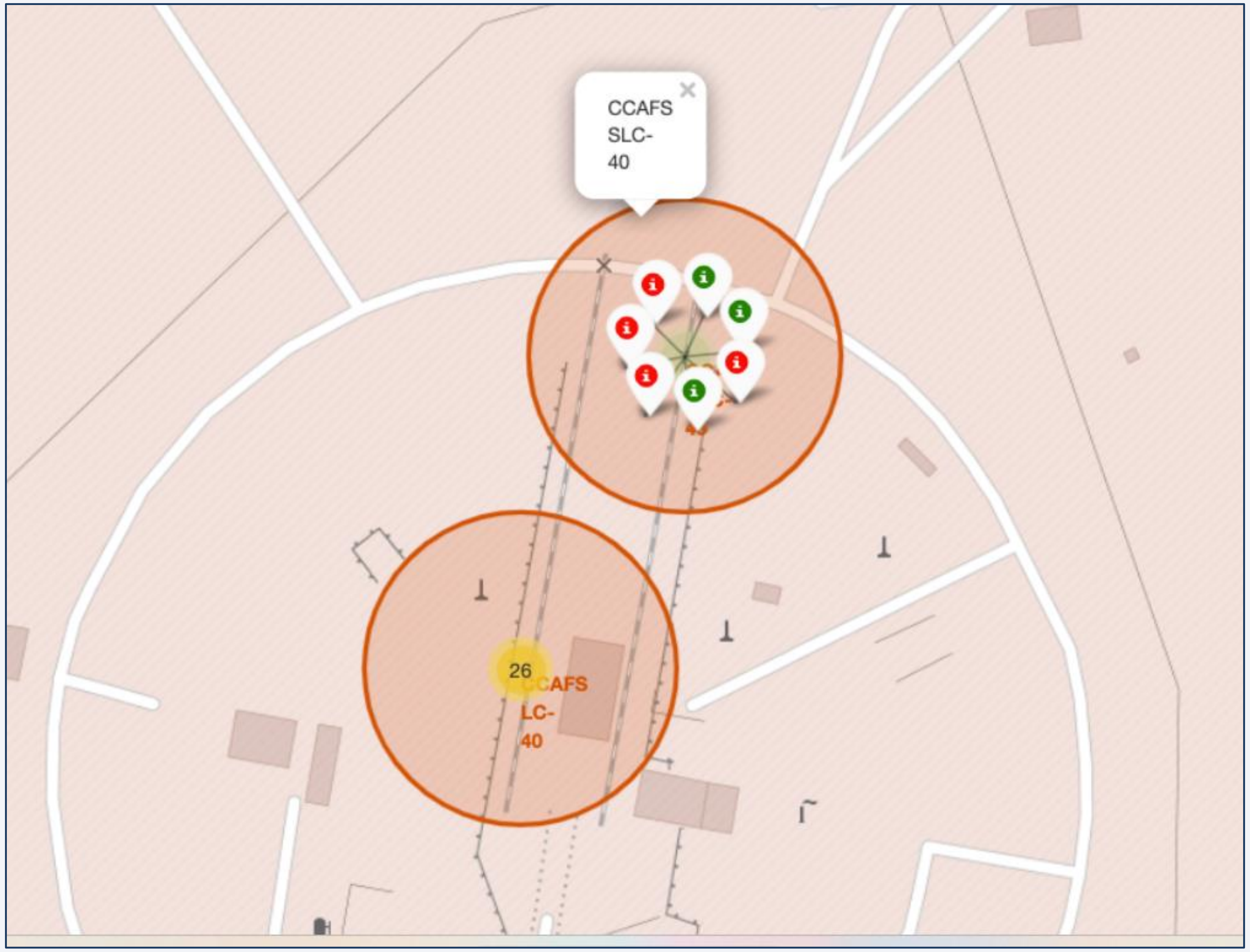
	Launch Site	Lat	Long	class	marker_color
51	CCAFS SLC-40	28.563197	-80.57682	0	red
52	CCAFS SLC-40	28.563197	-80.57682	0	red
53	CCAFS SLC-40	28.563197	-80.57682	0	red
54	CCAFS SLC-40	28.563197	-80.57682	1	green
55	CCAFS SLC-40	28.563197	-80.57682	0	red

Result:

- A launch only happens in one of the four launch sites, which means many launch records have the exact same coordinate. That's why we used the marker clusters to simplify a map.
- Let's zoom the location of the any launch site.



The launch outcomes for CCAFS SLC-40 on the Folium map



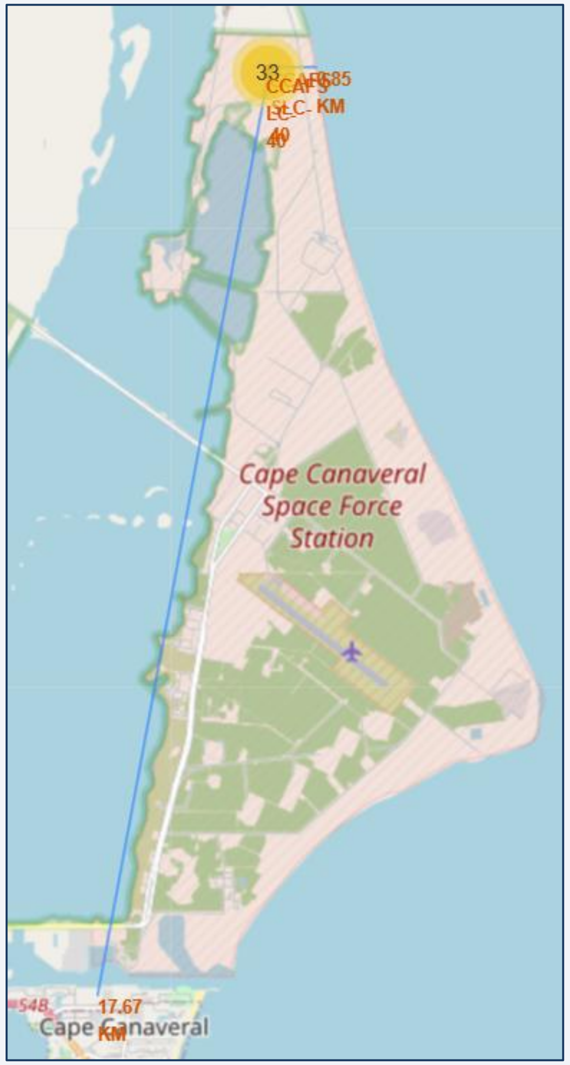
Result:

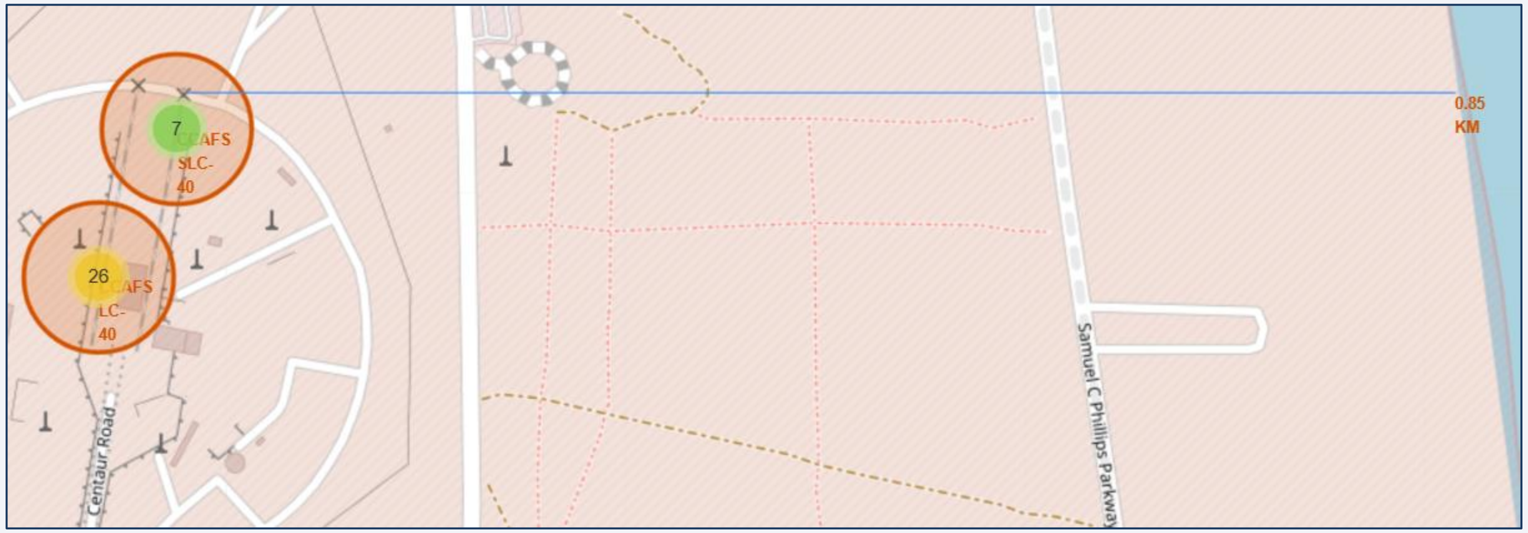
From the color-labeled markers in marker clusters we should be able to easily identify, which launch sites have relatively high success rates:

- Green (=1) – a launch was successful;
- Red (=0) – a launch was failed.

In the screenshot is a example of CCAFS SLC-40 launch site.

CCAFS SLC-40 launch site and its proximities on the Folium map





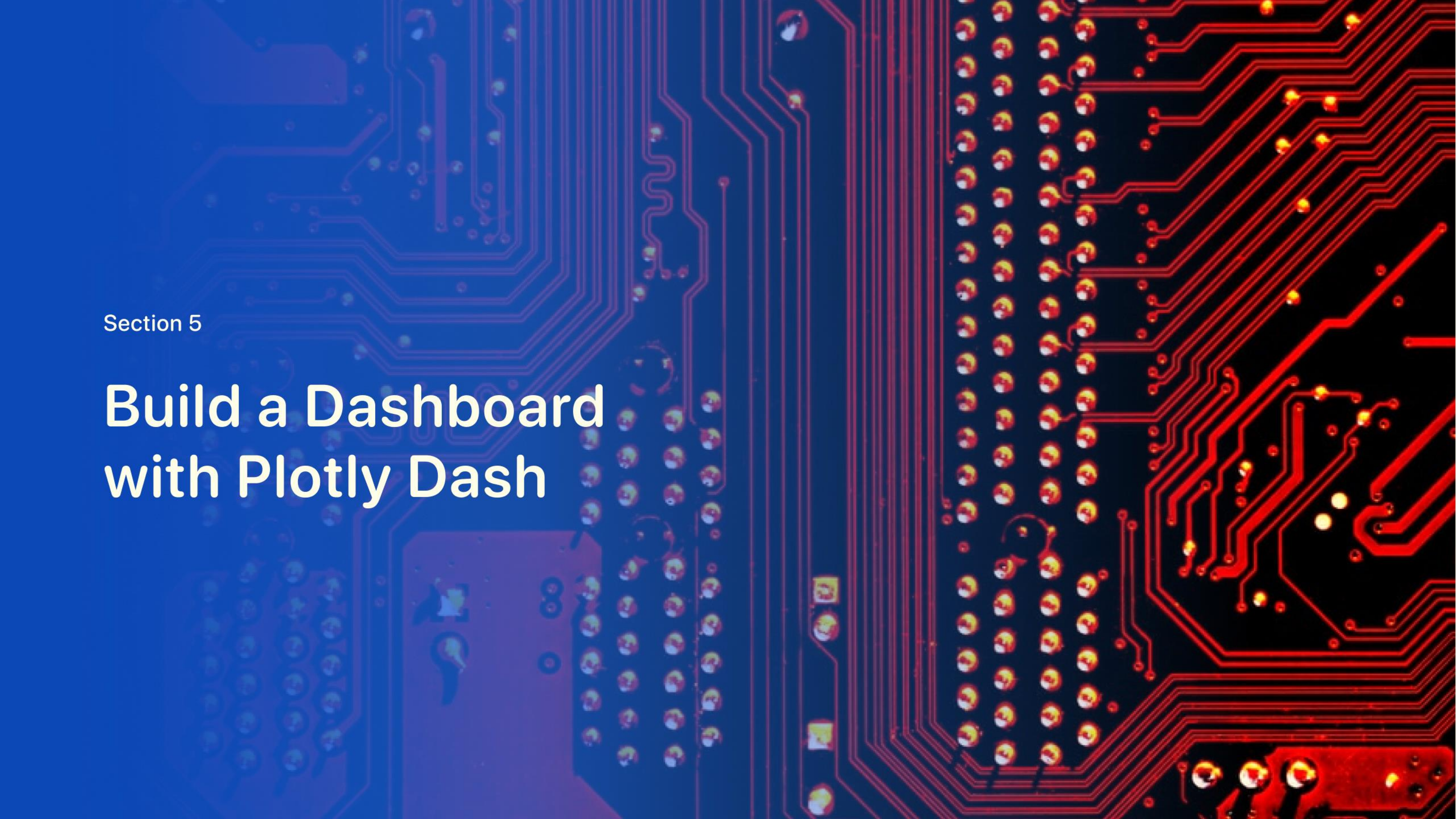
Result:

The launch site CCAFS SLC-40 is located close to:

- railways and roads - about a few meters;
- coastline - about 1 km;

and away from:

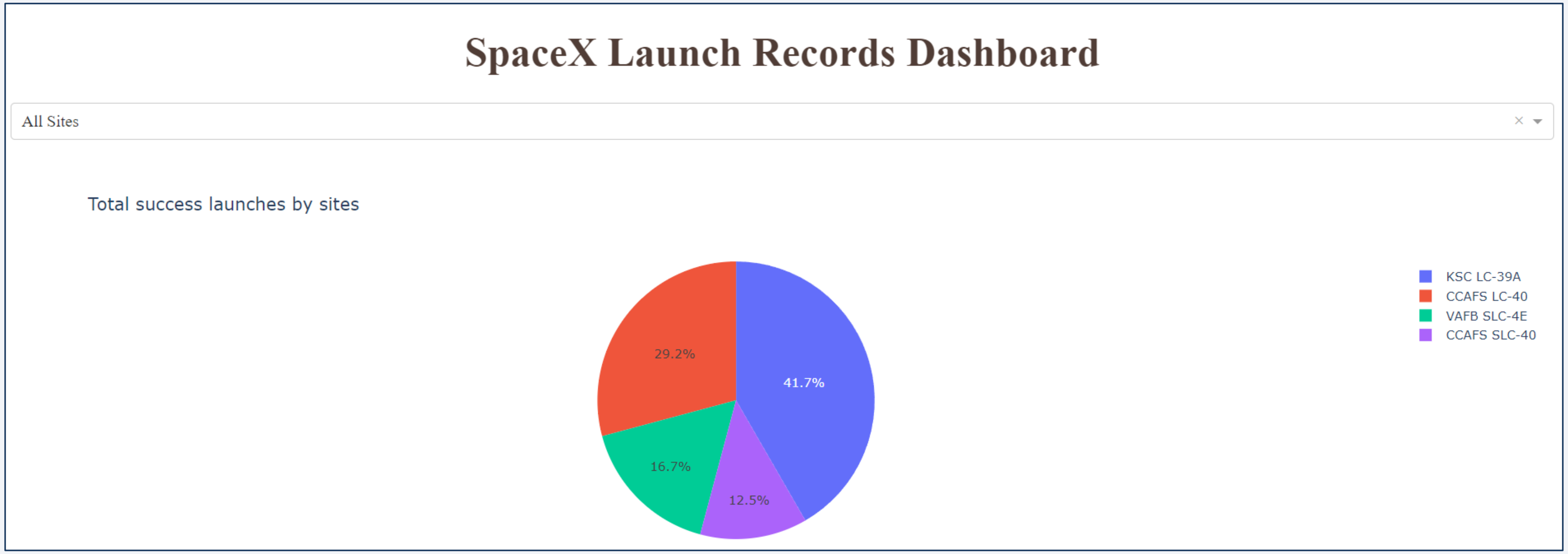
- city - about 18 km.



Section 5

Build a Dashboard with Plotly Dash

All success launches in a piechart in Dashboard



SpaceX Launch Records Dashboard

All Sites

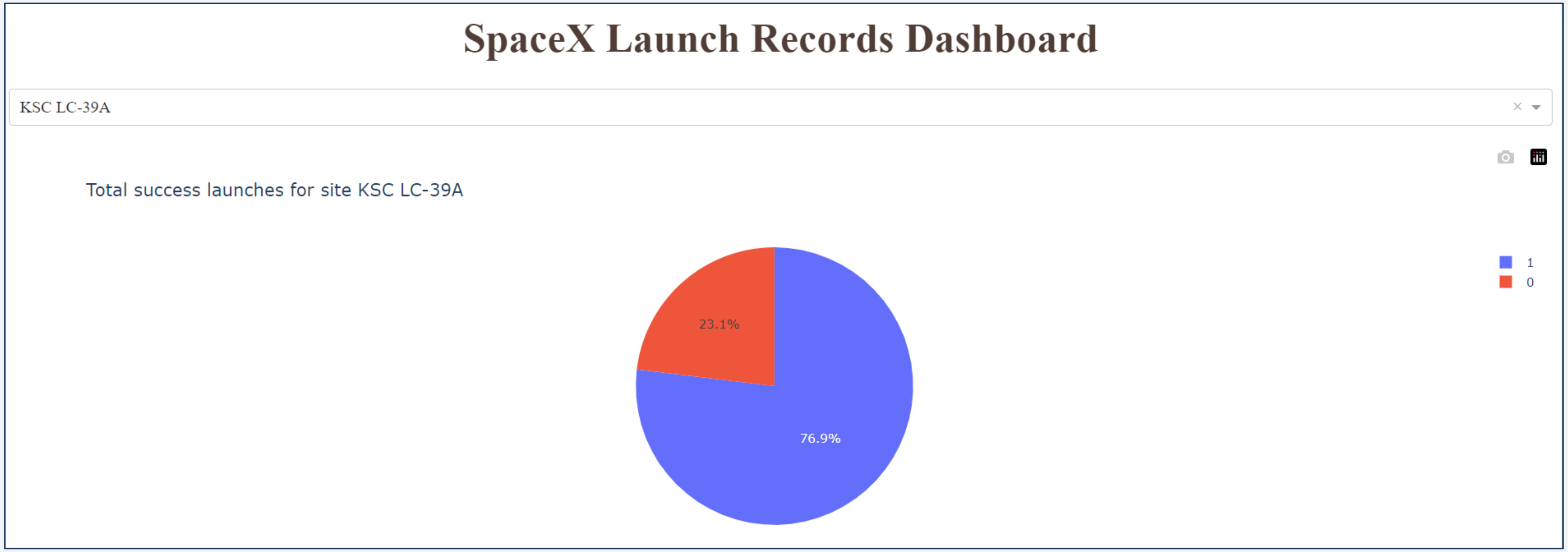
Total success launches by sites

<u>Result:</u>

The launch site KSC LC-39A has the largest successful launches (41,7%).

The launch site CCAFS SLC-40 has the lowest successful launches (12,5%).

The launch site with highest launch success ratio in a piechart in Dashboard


SpaceX Launch Records Dashboard

KSC LC-39A

Total success launches for site KSC LC-39A

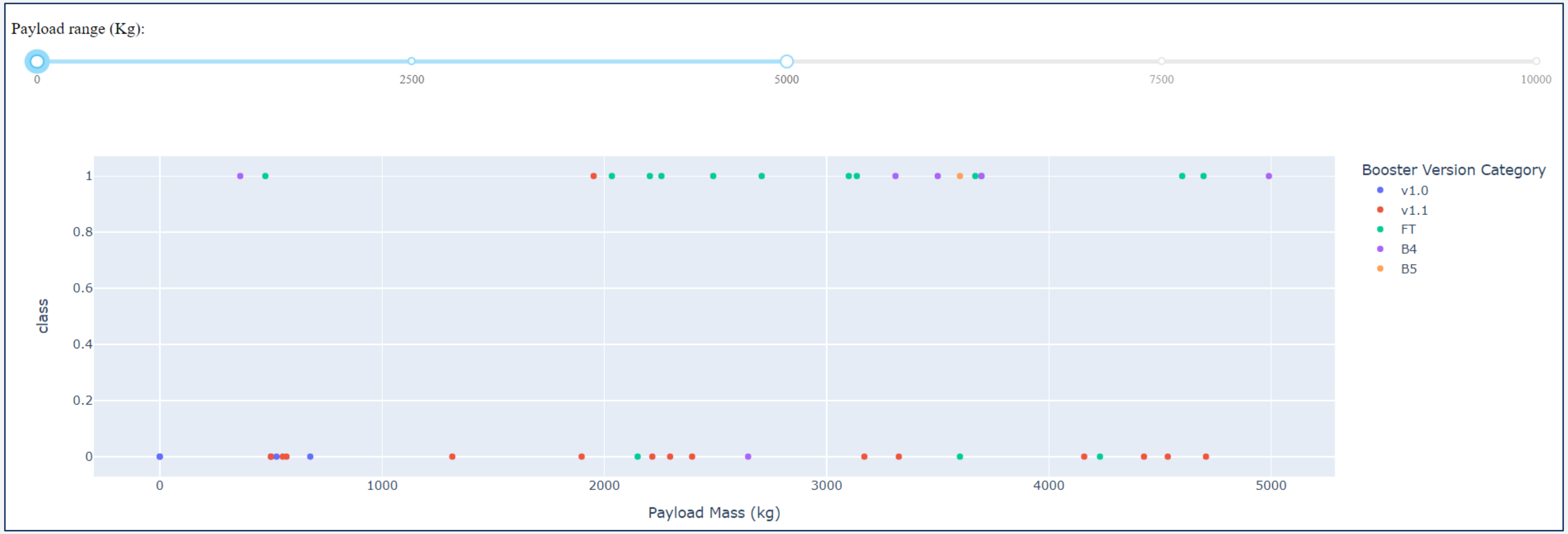
Result:

The launch site KSC LC-39A has the successful launches 76,9% of the total.

The launch site CCAFS SLC-40 has the failed launches 23,1% of the total.

This is a good result.

Payload vs. Launch Outcome scatter plot for all sites in Dashboard



<u>Result:</u>

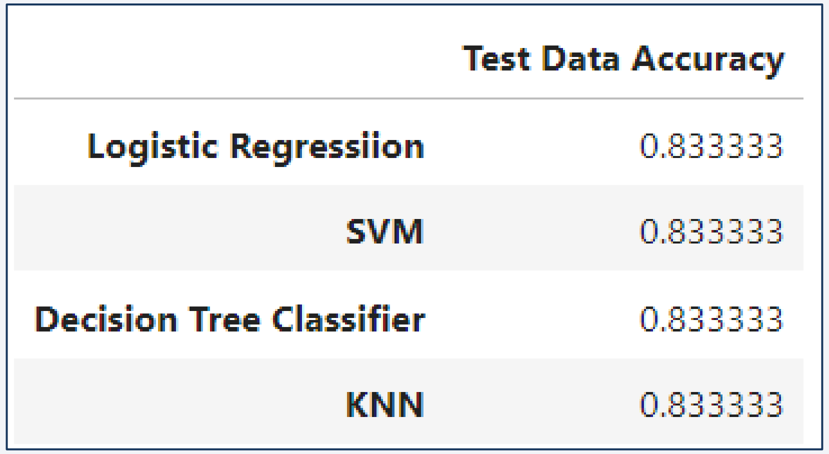
The Payload range from 2000 to 4000 (kg) has the largest success rate.

The Booster version FT has the largest success rate.

Section 6

Predictive Analysis (Classification)

Classification Accuracy

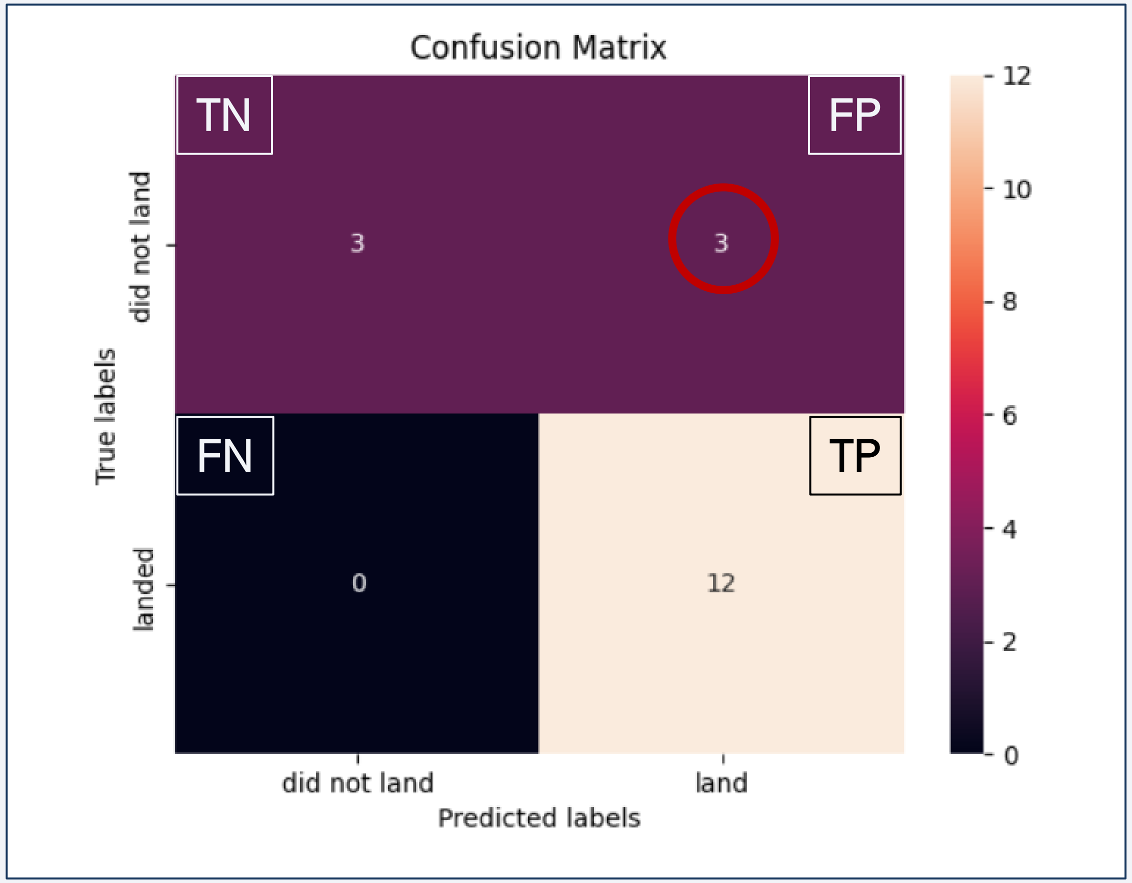


	Test Data Accuracy
Logistic Regressiion	0.833333
SVM	0.833333
Decision Tree Classifier	0.833333
KNN	0.833333

- All Classification methods using test data are equal.
- The accuracy measure is 83%, it's realistic.
- We can use any model to predict, if the first stage will land.

The same Accuracy for all Classification models: SVM, Decision Tree Classifier, KNN, Logistic Regression

Confusion Matrix



- This matrix shows the corrected and wrong predicted labels, in comparison with the actual (true) labels.
- This method can distinguish between the different classes.
- The major problem of this method is false positives (FP) = 3.

The same confusion matrix for all Classification models: SVM, Decision Tree Classifier, KNN, Logistic Regression

Conclusions

- To analyze the data used the predictive analysis based on the classification models cluster.
- Used the visualization, SQL queries and Folium map to determine the effect of the features on the prediction target.
- Used the numerical measures to evaluate different models.
- The accuracy measure is 83%, that is equal for all classification models.
- The accuracy measure is realistic. This is also consistent with industry standards.
- So any classification model can be used for prediction, if the first stage of the rocket Falcon 9 will land successfully: SVM, Decision Tree Classifier, KNN, Logistic Regression.
- In the future based on the results of this predictive analysis our company Space Y can determine the launch cost of the rocket.

Appendix

The references:

- SpaceX API to extract information:

  https://api.spacexdata.com/v4/rockets/

  https://api.spacexdata.com/v4/launchpads/

  https://api.spacexdata.com/v4/payloads/

  https://api.spacexdata.com/v4/cores/

  https://api.spacexdata.com/v4/launches/past

  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json
- Wikipedia to extract a Falcon 9 launch records:

  https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
- The dataset .csv file to download and implement the SQL queries:

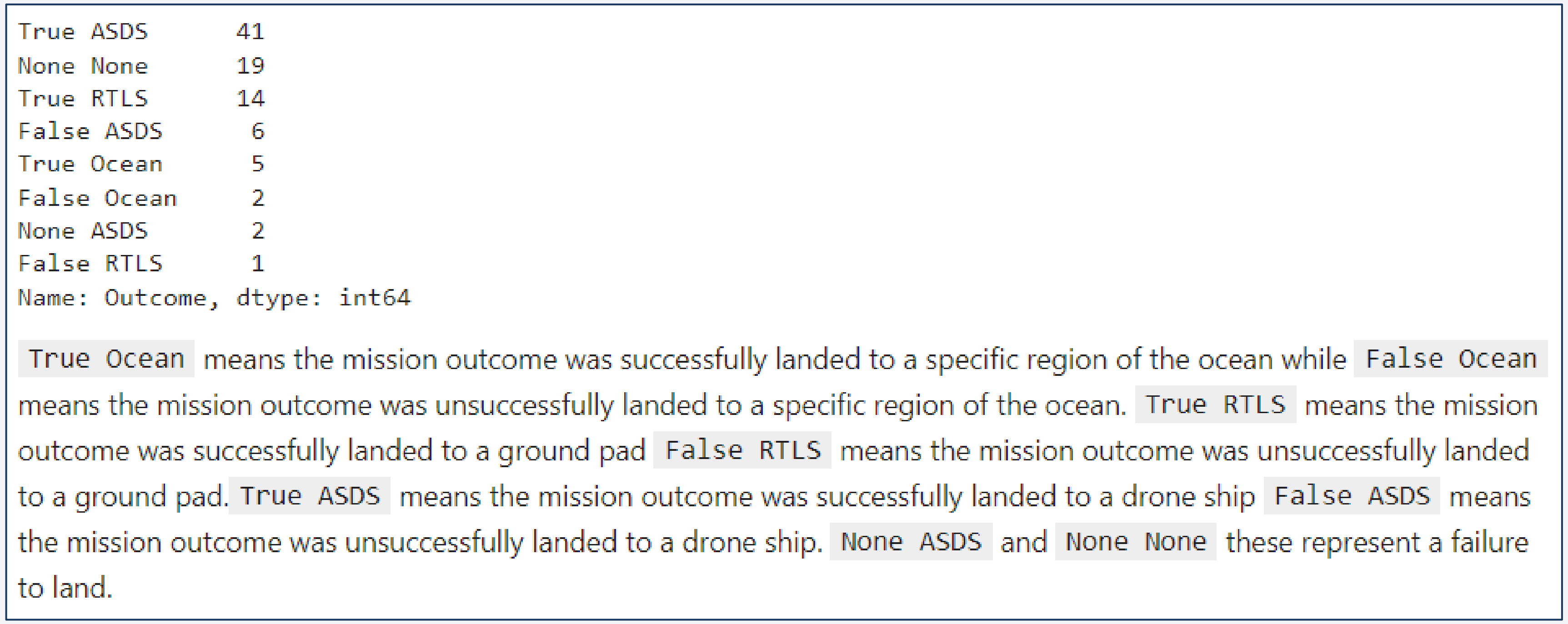
  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/labs/module_2/data/Spacex.csv
- The dataset .csv to download for dashboard:

  https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/spacex_launch_dash.csv

Appendix

Additional information about the number and occurrence of mission outcome:

Used the method .value_counts() on the column “Outcome”.



```
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
```

`True Ocean` means the mission outcome was successfully landed to a specific region of the ocean while `False Ocean` means the mission outcome was unsuccessfully landed to a specific region of the ocean. `True RTLS` means the mission outcome was successfully landed to a ground pad `False RTLS` means the mission outcome was unsuccessfully landed to a ground pad. `True ASDS` means the mission outcome was successfully landed to a drone ship `False ASDS` means the mission outcome was unsuccessfully landed to a drone ship. `None ASDS` and `None None` these represent a failure to land.

Result:

The drone ship has the largest successful mission outcome (41 from 90).

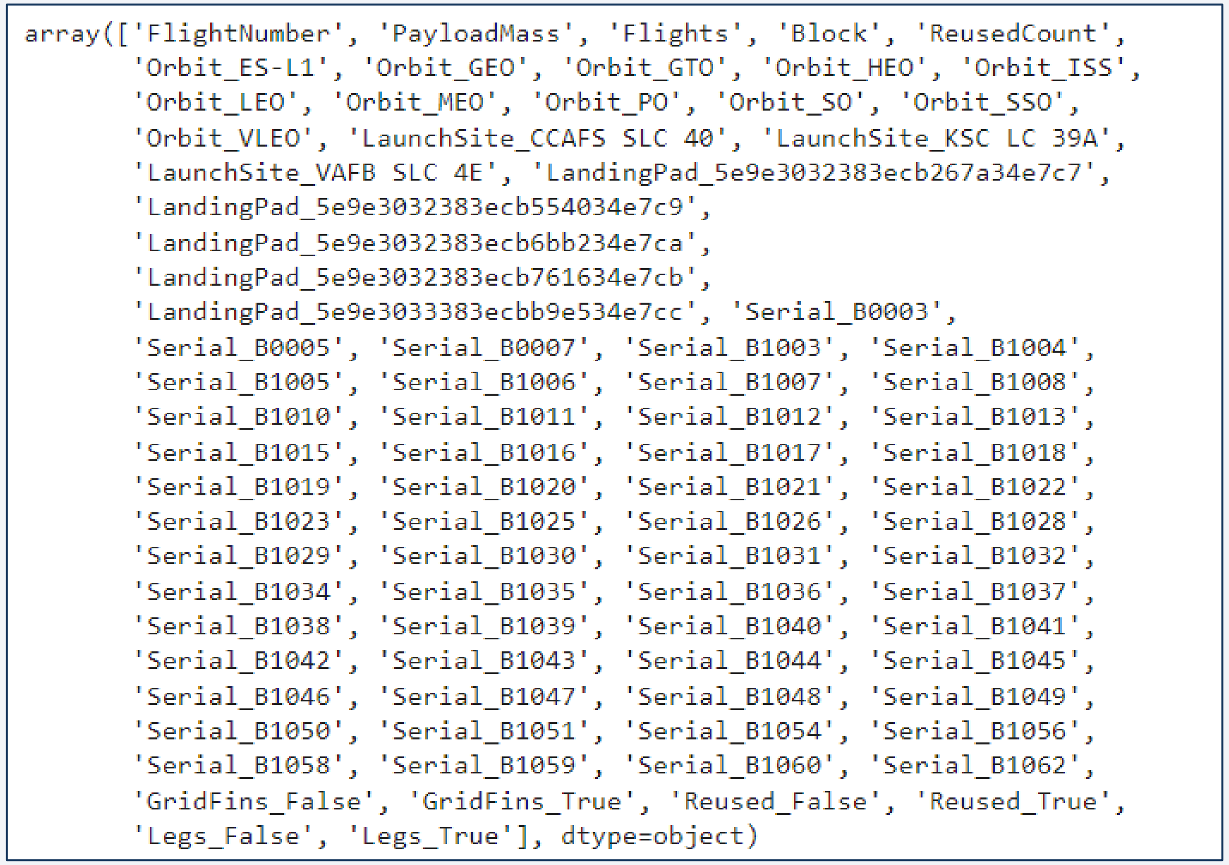
Total: 60 – successful, 30 – failed mission outcome.

Appendix

Additional information about Machine Learning Prediction:

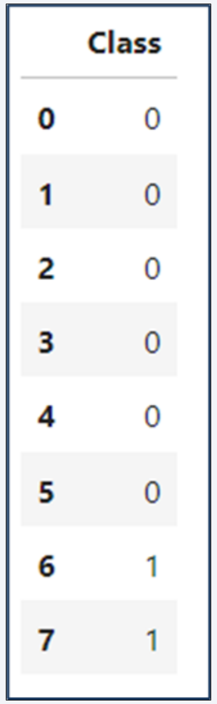
The variables from data frame, that we used for classification models into training and test data.

X =



```
array(['FlightNumber', 'PayloadMass', 'Flights', 'Block', 'ReusedCount',
       'Orbit_ES-L1', 'Orbit_GEO', 'Orbit_GTO', 'Orbit_HEO', 'Orbit_ISS',
       'Orbit_LEO', 'Orbit_MEO', 'Orbit_PO', 'Orbit_SO', 'Orbit_SSO',
       'Orbit_VLEO', 'LaunchSite_CCAFS SLC 40', 'LaunchSite_KSC LC 39A',
       'LaunchSite_VAFB SLC 4E', 'LandingPad_5e9e3032383ecb267a34e7c7',
       'LandingPad_5e9e3032383ecb554034e7c9',
       'LandingPad_5e9e3032383ecb6bb234e7ca',
       'LandingPad_5e9e3032383ecb761634e7cb',
       'LandingPad_5e9e3033383ecbb9e534e7cc', 'Serial_B0003',
       'Serial_B0005', 'Serial_B0007', 'Serial_B1003', 'Serial_B1004',
       'Serial_B1005', 'Serial_B1006', 'Serial_B1007', 'Serial_B1008',
       'Serial_B1010', 'Serial_B1011', 'Serial_B1012', 'Serial_B1013',
       'Serial_B1015', 'Serial_B1016', 'Serial_B1017', 'Serial_B1018',
       'Serial_B1019', 'Serial_B1020', 'Serial_B1021', 'Serial_B1022',
       'Serial_B1023', 'Serial_B1025', 'Serial_B1026', 'Serial_B1028',
       'Serial_B1029', 'Serial_B1030', 'Serial_B1031', 'Serial_B1032',
       'Serial_B1034', 'Serial_B1035', 'Serial_B1036', 'Serial_B1037',
       'Serial_B1038', 'Serial_B1039', 'Serial_B1040', 'Serial_B1041',
       'Serial_B1042', 'Serial_B1043', 'Serial_B1044', 'Serial_B1045',
       'Serial_B1046', 'Serial_B1047', 'Serial_B1048', 'Serial_B1049',
       'Serial_B1050', 'Serial_B1051', 'Serial_B1054', 'Serial_B1056',
       'Serial_B1058', 'Serial_B1059', 'Serial_B1060', 'Serial_B1062',
       'GridFins_False', 'GridFins_True', 'Reused_False', 'Reused_True',
       'Legs_False', 'Legs_True'], dtype=object)
```

Y =



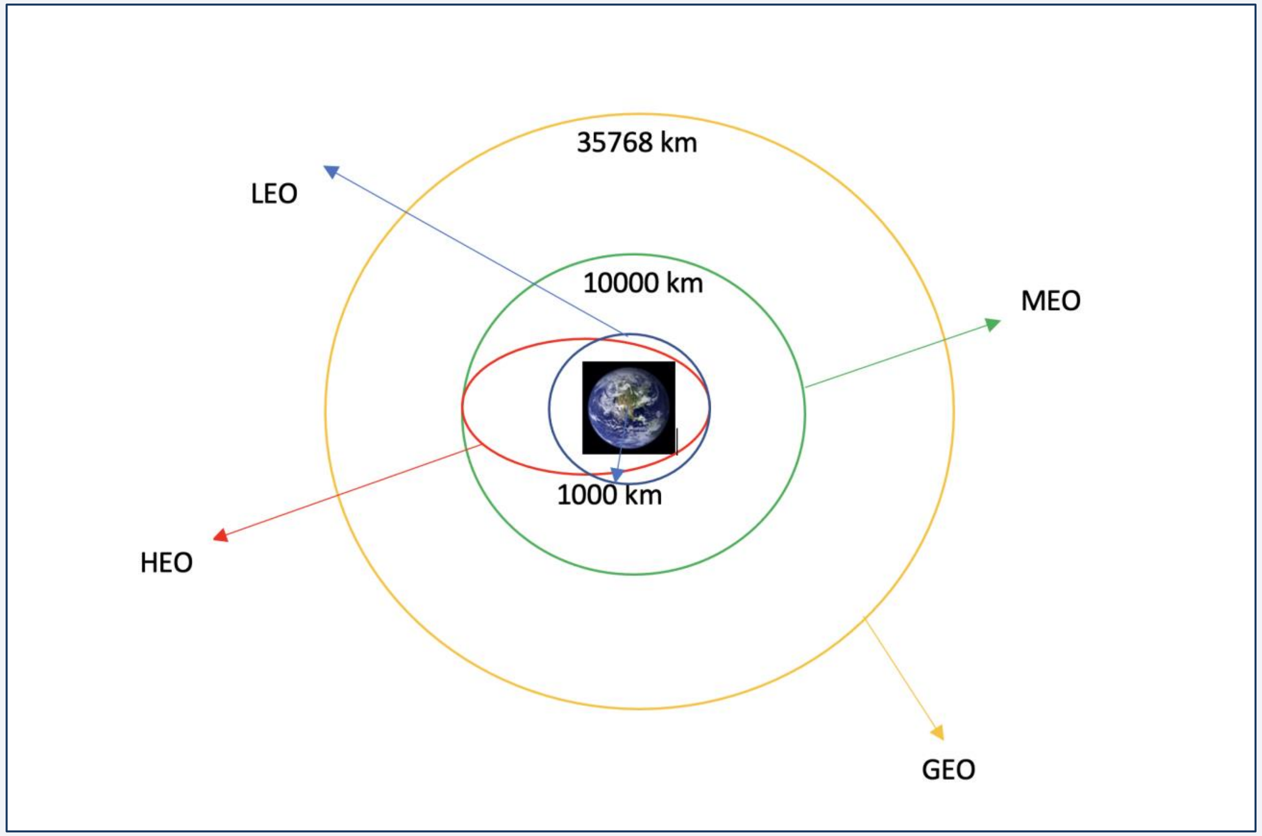
	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

prediction variable

Appendix

Additional information about Orbits:

https://en.wikipedia.org/wiki/Geocentric_orbit



Thank you!