IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Philippe LF
2024 August 11

Outline

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

Executive Summary

- In this project we have used data from Space X API and complementary data from Wikipedia
- We made Exploratory Data Analysis on that data using Pandas and SQL queries
- We built an interactive map with Folium to study the Launch Sites proximities and we built an interactive dashboard with Plotly/Dash to realize correlations between payloads, boosters and success rate
- Finally, we built a Classifier to predict the landing outcome of the 1st stage depending on several Features Engineering
- Our main result is that GTO orbit and payloads between 4000-10000 kg are critical factors.
- We may be competitive for this type of mission with payloads up to 10 tons.
- We should use the Classifier to make a better decision.

Introduction

Project background and context

- Our company SPACE Y aims to place satellites on orbit at the best price in order to meet our customers requirements
- There are several competitors in the world e.g. Space X, Lockheed Martin, Boeing, NASA in the US and also ArianeGroup in Europe and other companies in Russia, China and India in the near future
- Space X advertises Falcon 9 rocket launches can cost 62 million dollars each whereas other contenders announce price between 100 million and 165 millions each launch
- Space X attractive price mostly relies on a “recoverable” first stage (that is the more expensive part of the rocket system) but success is not guaranteed – there maybe some failures depending on several features

Problems you want to find answers

- Based on Data Science tools, we want to analyze the success rate of 1st stage landing and how it is correlated to Launch Site, Payload Mass, Booster Version…
- Then we can decide if we can win the competition against Space X with our own rocket launcher.

Example : Success vs Failure

Succesful 1st stage Landing on ground

1st stage Landing failed

Section 2

Methodology

Data Collection – SpaceX API

- We got launch data from Space X API using GET request function and we used helper functions provided by Coursera to get clear data replacing IDs
- We put all the needed data in a dictionary objet
- We built our database data_falcon9 (a Python dataframe) from the dictionary and filtered only Falcon 9 launches resulting in a dataframe of 90 rows and 18 columns
- We conducted some basic data wrangling and formatting
  - Remove unnecessary rows
  - Convert date-utc to datetype dataframe
  - Change type of data with astype() when needed
  - Detect missing data with isnull() function and replace missing data with mean value in column
- Please refer to GitHub URL of the completed notebook : https://github.com/PhilippeLF/Data_Science_Capstone_Project.git File : jupyter-labs-spacex-data-collection-api.ipynb

SPACEX

Request rocket launch data from url « https://api.spacexdata.com/v4/launches/past »

↓

Get booster version from launch data and url « https://api.spacexdata.com/v4/rockets »

Get Launch site from launch data and url « https://api.spacexdata.com/v4/launchpads »

Get Payload data from launch data and url « https://api.spacexdata.com/v4/payloads »

Get Cores data from launch data and url « https://api.spacexdata.com/v4/cores »

Put data in a newly created dictionary

↓

Construct database data_falcon9 with filter(s)

↓

Basic data wrangling and formatting

↓

Export database to a CSV or Excel file for next use

Data Collection - Scraping

- In order to get complementary data regarding Customers and Booster Landing success/fail, we performed Web Scrapping from Wikipedia (HTML code)
- We used GET request function to connect and helper functions provided by Coursera to parse values
- We used the BeautifulSoup package : see process on right side
- After we fill in the parsed launch record values into a dictionary, we created a dataframe and saved it in csv format
- Please refer to GitHub URL of the completed notebook : https://github.com/PhilippeLF/Data_Science_Capstone_Project.git File : jupyter-labs-webscraping.ipynb

Request Launch data from url « https://en.wikipedia.org/wiki/list_of_Falcon_9_and_Falcon_Heavy_launches »

↓

Create a BeautifulSoup object

↓

Find all HTML Tables header with find_all('table') function and choose the 3rd one

↓

Extract column names with find_all('th') function

↓

Create a dictionary with keys as the colum headers and fill it up with launch records extracted from table row with find_all('tr') function and helper functions provided

↓

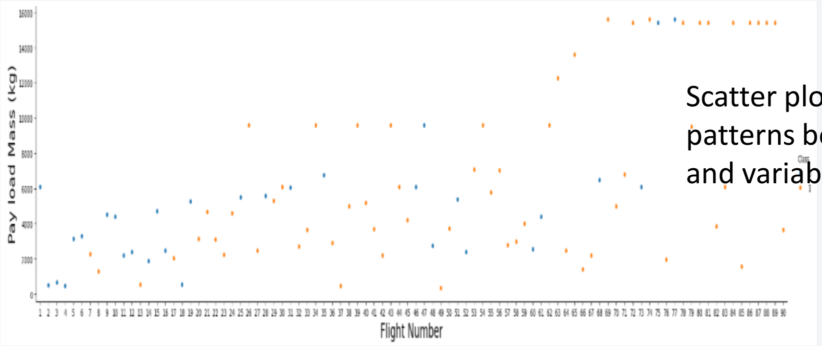
Create a dataframe from the dictionary and export it in a CSV or Excel file for next use

Data Wrangling

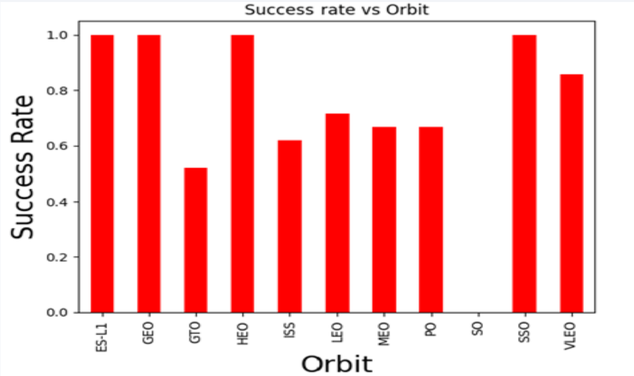
- Objective was to convert Launch outcomes data into Training Labels with 1 means booster successfully landed and 0 when it fails
- We made preliminary EDA to identify:
  - Number of launches on each site
  - Number and occurrence of each orbit (list of 11)
- We created a new column called "Class" and computed the mean of success : 0,67
- Please refer to this GitHub URL of the completed Data Wrangling notebook : https://github.com/PhilippeLF/Data_Science_Capstone_Project.git File : labs-jupyter-spacex-Data wrangling.ipynb

EDA with Data Visualization

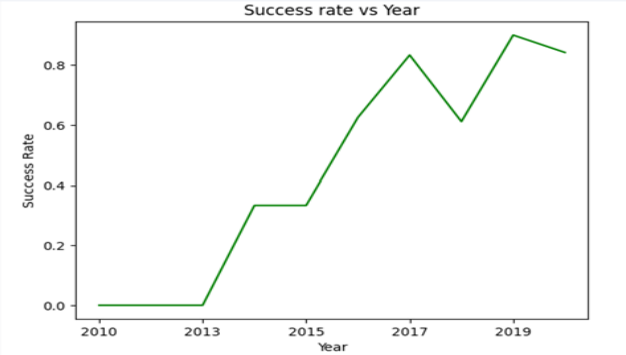
- Objective of EDA is to predict if the Falcon 9 first stage will land successfully
- We have looked for any correlation/patterns between success and Flight Numbers, Launch Site, Payload Mass, Orbit type, Yearly trend with various plots
- We also have created dummy variables to categorical columns to comply with Machine Learning models (next section)
- Please refer to GitHub URL of the completed notebook : https://github.com/PhilippeLF/Data_Science_Capstone_Project.git File : edadataviz.ipynb



Scatter plots used to identify patterns between variable x and variable y



Bar plots were used to compare categorical variables



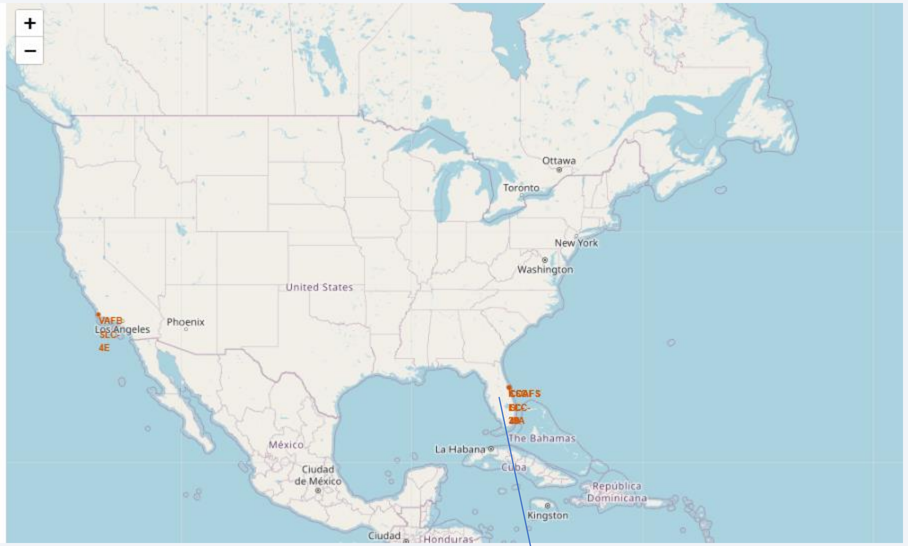
Line plot was used see the Yearly trend of one variable

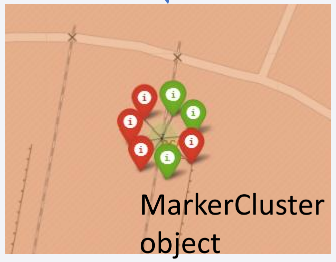
EDA with SQL

- Complementary Exploratory Data Analysis of Space X database using SQL queries
- Got more information on Customers, Booster Version, Payload Mass, type of landing outcomes
- We used sqlite3 package and %sql Magic commands
- We used several type of SQL Commands such as:
  - Simple command e.g. %sql SELECT * from SPACEXTABLE
  - Conditional commands e.g. %sql ….. Where {condition}
  - Aggregate function commands e.g. %sql SELECT sum("column name") from SPACEXTABLE
  - Grouping and sorting commands e.g. %sql SELECT "col name" from SPACEXTABLE GROUP BY DESC
- All Command lines used are listed in Appendix 1
- Please refer to GitHub URL of the completed notebook:
  https://github.com/PhilippeLF/Data_Science_Capstone_Project.git File : jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

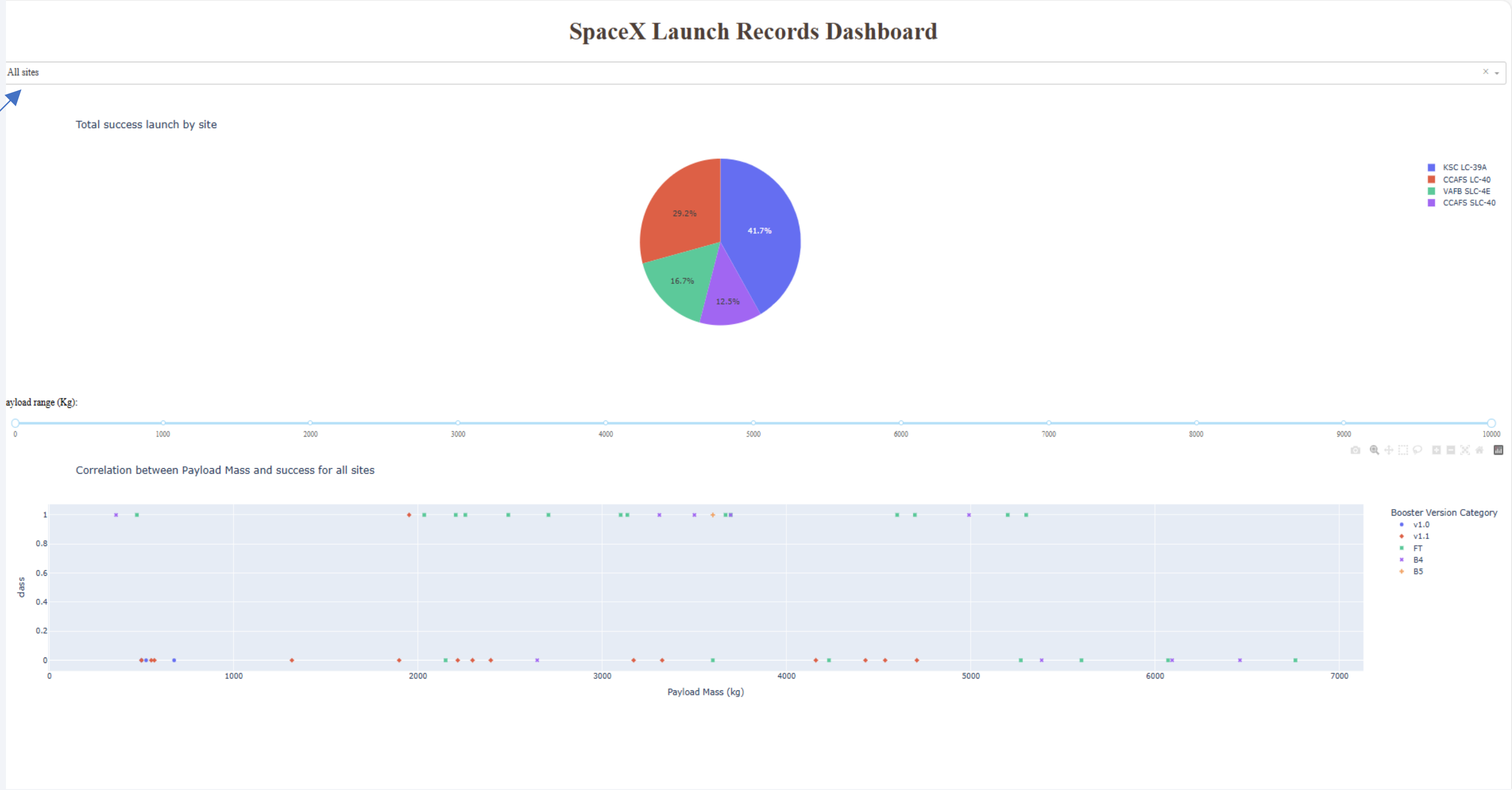
- Objective was to locate Launch Site and their proximities such as railway, highways, city on a map to be used to assess success rate and economic benefit
- We used Space X database and Folium packages and sub-packages
- We have marked the Launch Sites on the map to see their distance from the Equator line using folium.Map function centered on NASA Johnson Space Centers's coordinates
- We used folium.Circle and folium.Marker fucntions to show all 4 Launch Sites on the map with their names
- We have used **MarkerCluster objects** to show all launch outcomes (success or not) having the same coordinate using folium.Marker and folium.Icon functions and add.child() method to aggregate
- We have calculated the distances between a Launch Site and its proximities and we have drawn a line in between with folium.PolyLine function
- Please refer to GitHub URL of the completed notebook : https://github.com/PhilippeLF/Data_Science_Capstone_Project.git File : lab_jupyter_launch_site_location.ipynb





Build a Dashboard with Plotly Dash

- An interative Dasboad was created to get quick access to Success rate vs Launch Site and Payload Mass
- Upper part uses a **Dropdown menu** and displays a **Pie plot** that shows:
  - The total successful launch for all sites (in %)
  - Success vs Fail Counts if one Luanch site is chosen,
- Lower part uses a **Range Slider** and displays a **Scatter plot** that shows:
  - The correlation between payload and launch success for all sites or selected one
  - Booster Version category is also indicated by different colors
- Please refer to GitHub URL of the completed notebook : https://github.com/PhilippeLF/Data_Science_Capstone_Project.git File : spacex_dash_app.ipynb



Predictive Analysis (Classification)

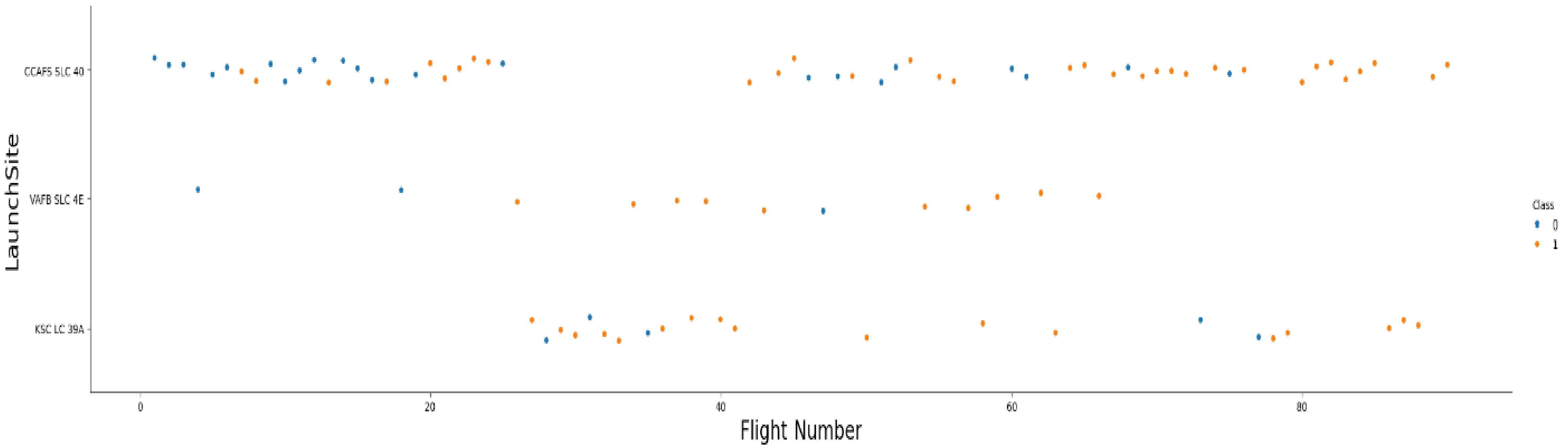
- Features Engineering:
    - from Space X database we have the following features for each Flight: 'FlightNumber', 'PayloadMass', 'Orbit', 'LaunchSite', 'Flights', 'GridFins', 'Reused', 'Legs', 'LandingPad', 'Block', 'ReusedCount', 'Serial'.
    - We created dummy variables to categorical columns using pd.get_dummies Python function and we casted all numeric columns to float64
- Machine Learning Models been tested:
    - Logistic regression
    - Support Vector Machine
    - Decision Tree Classifier
    - K-nearest Neighbors
- For hyperparamertes analysis of each model, we used the GridSearchCV package. It helps selecting the best model by evaluating multiple combinations of hyperparameters
- Evaluation criteria been used:
    - Best_score in between 0 and 1 – close to 1 is better
    - Confusion matrix showing False Negative and False positive
- Please refer to GitHub URL of the completed notebook : https://github.com/PhilippeLF/Data_Science_Capstone_Project.git File SpaceX_Machine Learning Prediction_Part_5.ipynb

Section 3

Insights drawn from EDA

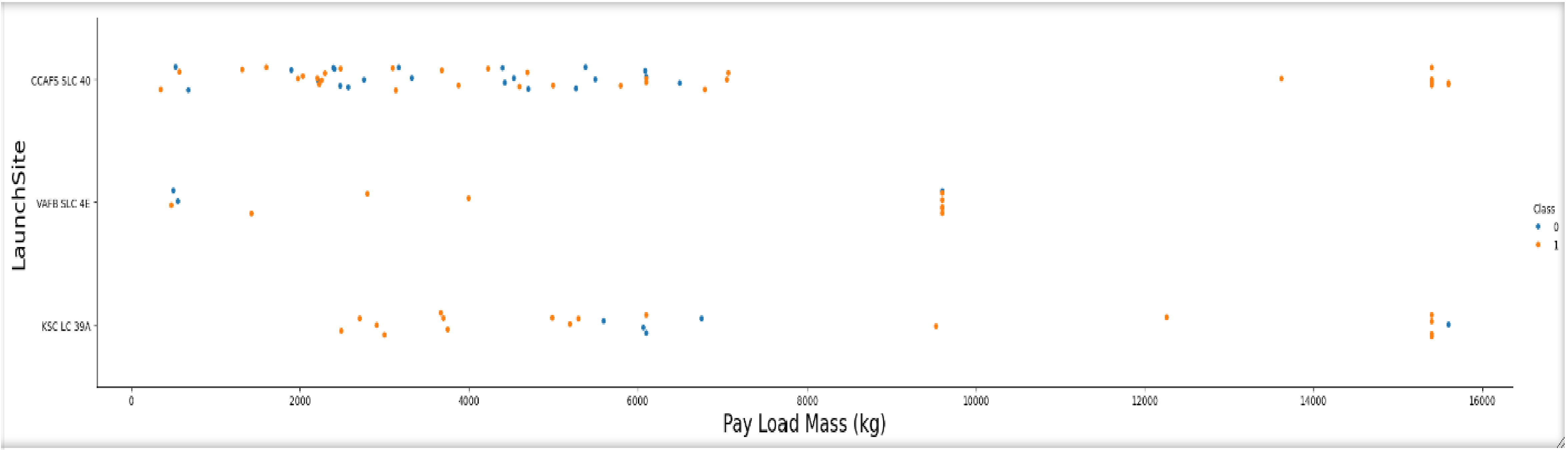
Flight Number vs. Launch Site

- Most of flight numbers started in the CCAFS SLC 40 Launch Site with many failures (class 1 blue mark). Starting FN23 Space X decided for some reason (unavailability of the Launch Site ?) to use the other two launch Sites
- They came back to the original Launch Site after Flight Number 41
- Starting from Flight Number 60, the outcome drastically improved (most are class 1).



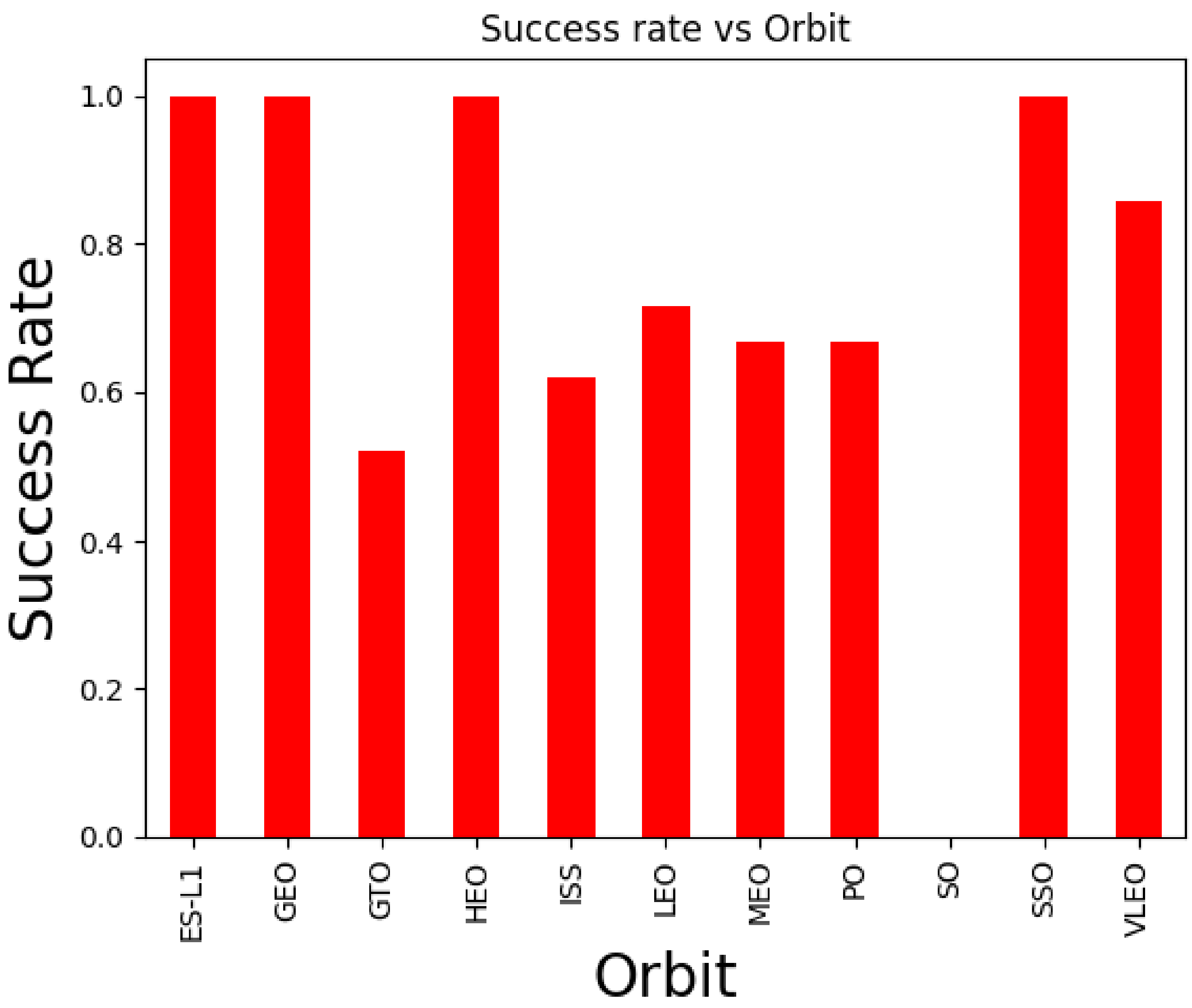
Payload vs. Launch Site

- For the VAFB-SLC launch site there are no rockets launched for heavy payload mass (greater than 10000)
- It seems that payload range 4000-6000 kg is critical since there are many failures



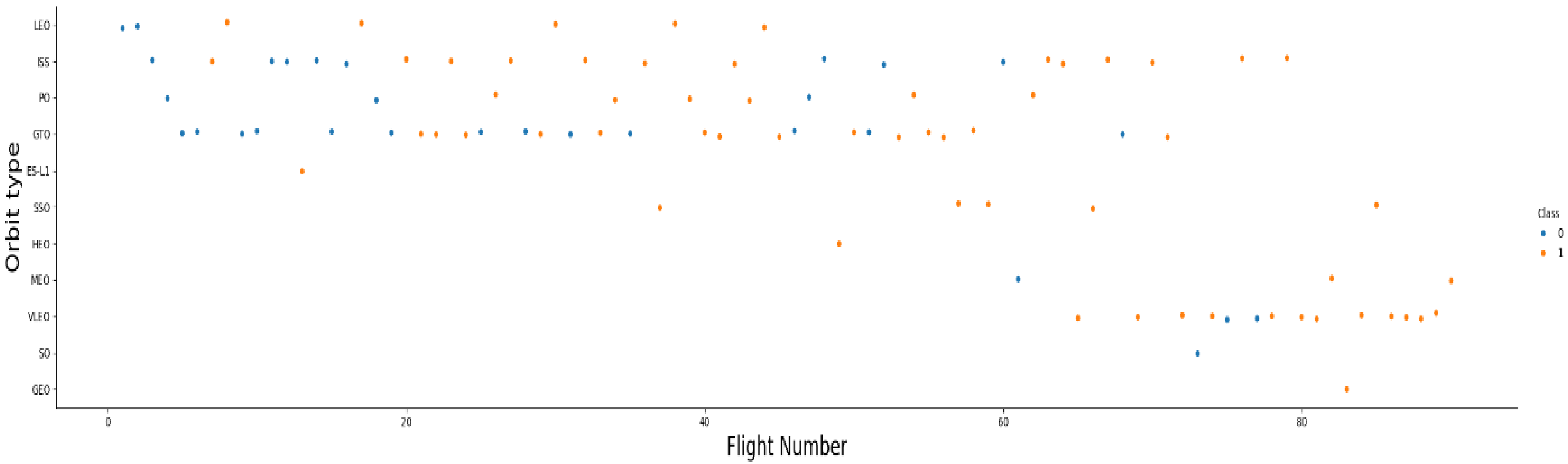
Success Rate vs. Orbit Type

- Orbit with 100% success are ES-L1, GEO, HEO, SSO however there has been only a few launches (see next slide)
- LEO and VLEO orbits had relatively good success because they are low altitude
- The success rate is smaller for high altitude orbit such as GTO



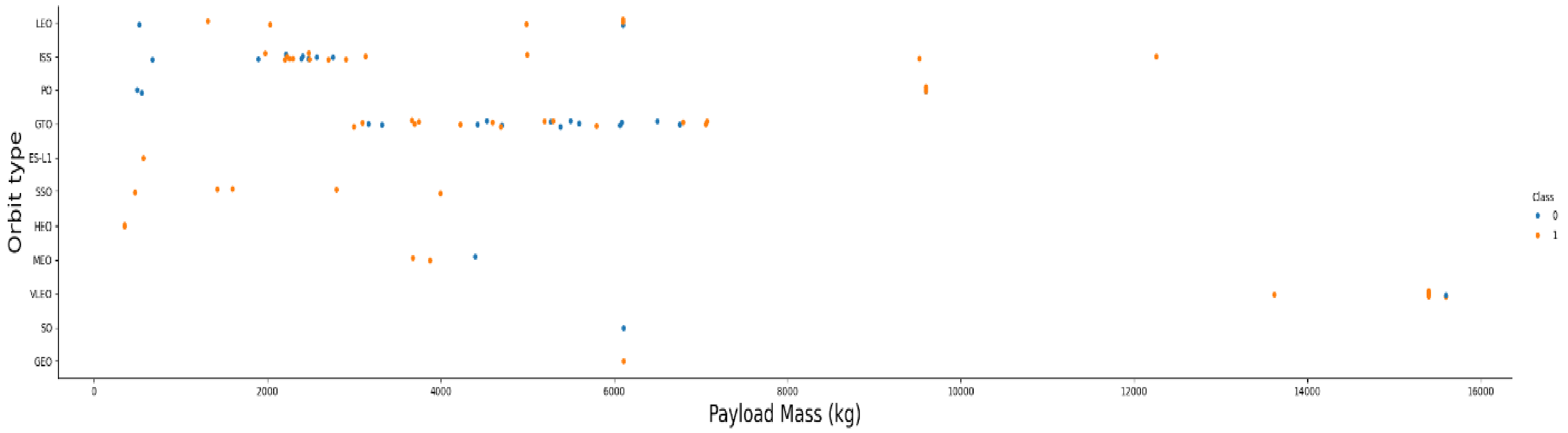
Flight Number vs. Orbit Type

- GTO, ISS,VLEO are the most used orbits especially VLEO since Flight Number 60
- Success rate has improved regularly



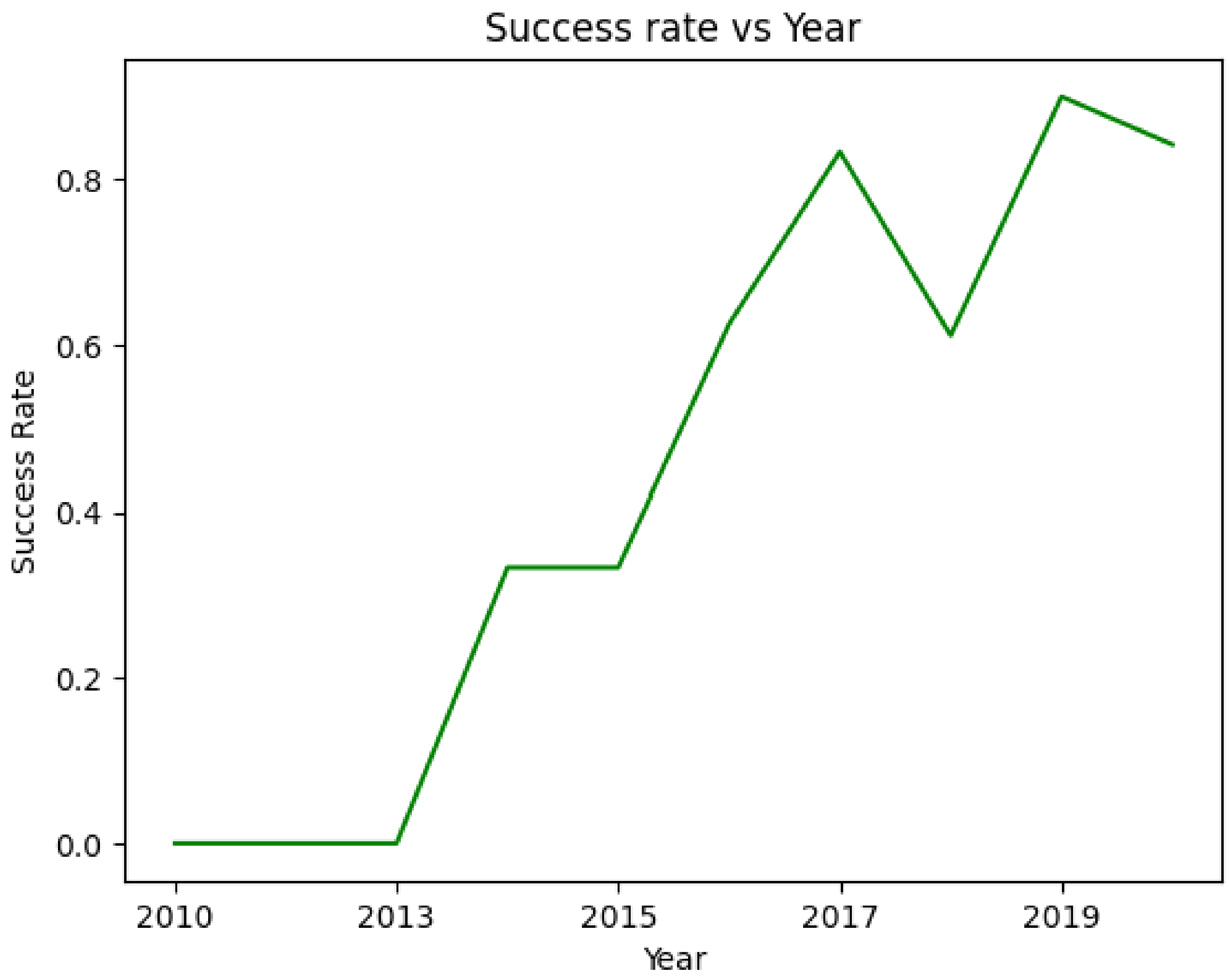
Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS
- However, for GTO, it's difficult to distinguish between successful and unsuccessful landings as both outcomes are present.



Launch Success Yearly Trend

We can observe that the sucess rate since 2013 kept increasing till 2020



All Launch Site Names

- 4 Unique launch sites obtained with SQL queries:

- VAFB SLC-4E is situated on the West Coast in California (Vandenberg Space Force Base, owner US Space Force)

- The three other ones are situated in Florida at Cape Canaveral and own either to the US Air Force or to NASA (KSC LC-39A)

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

Launch Site Names Begin with 'CCA'

- Here are 5 records where launch sites begin with `CCA`
- First test of Falcon 9 started in 2010 from CCAFS LC-40 Launch site then from VAFB SLC-4E in 2013
- Small Payload mass and short orbit (LEO) to start with
- "landing outcome" is about landing the first stage which started in 2012

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- NASA is the main Customer of Space X
- Total payload carried by boosters from NASA ≈ 100 tons,
- About half (45 tons) is for NASA (Commercial Resupply services) with the aim of supplying ISS

- We used the following query : %sql select SUM(PAYLOAD_MASS__KG_) as PayLoad_Total_Mass_All_NASA from SPACEXTABLE where Customer like 'NASA%'

PayLoad_Total_Mass_All_NASA
99980

Customer	PayLoad_Total_Mass
NASA (CRS)	45596

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.1 is 2534 kg
- It varies between 500 kg and 4700 kg

Average_Payload_Mass
2534.6666666666665

- We used the following SQL command : %sql SELECT "Booster_Version", AVG("PAYLOAD_MASS_KG") as Payload_Mass_Avg from SPACEXTABLE GROUP BY "Booster_Version“

First Successful Ground Landing Date

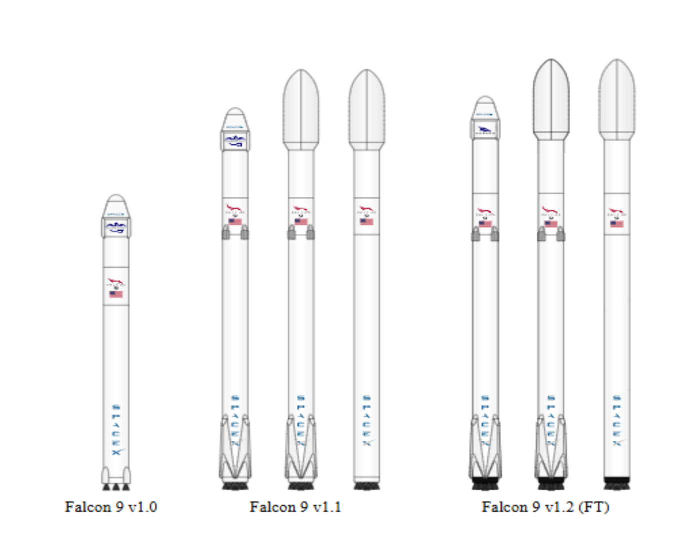
- Table on the right shows the earlier dates of Landing Outcome types
- It can be seen from the table that the first successful landing outcome in ground was on **22 December 2015**

Date_Minimum	Landing_Outcome
2014-04-18	Controlled (ocean)
2018-12-05	Failure
2015-01-10	Failure (drone ship)
2010-06-04	Failure (parachute)
2012-05-22	No attempt
2019-08-06	No attempt
2015-06-28	Precluded (drone ship)
2018-07-22	Success
2016-04-08	Success (drone ship)
2015-12-22	Success (ground pad)
2013-09-29	Uncontrolled (ocean)

Successful Drone Ship Landing with Payload between 4000 and 6000

- Names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

Booster_Version	Landing_Outcome
F9 FT B1022	Success (drone ship)
F9 FT B1026	Success (drone ship)
F9 FT B1021.2	Success (drone ship)
F9 FT B1031.2	Success (drone ship)



- We used the following Query : %sql SELECT "Booster_Version","Landing_Outcome“ from SPACEXTABLE WHERE "Landing_Outcome"='Success (drone ship)' AND "PAYLOAD_MASS__KG_" BETWEEN 4000 AND 6000

Total Number of Successful and Failure Mission Outcomes

- Total number of successful and failure mission outcomes
- Mission outcome was very good 99%

Mission_Outcome	count("Mission_Outcome")
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- Names of the booster which have carried the maximum payload mass (15,6 tons) : see table on the right
- %sql SELECT "Booster_Version","PAYLOAD_MASS__KG_" from SPACEXTABLE WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") from SPACEXTABLE); (we used a subquery for this complex query)

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

- List of failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

Month_Name	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- Present your query result with a short explanation here

Landing_Outcome	Landing_Outcome_Count
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

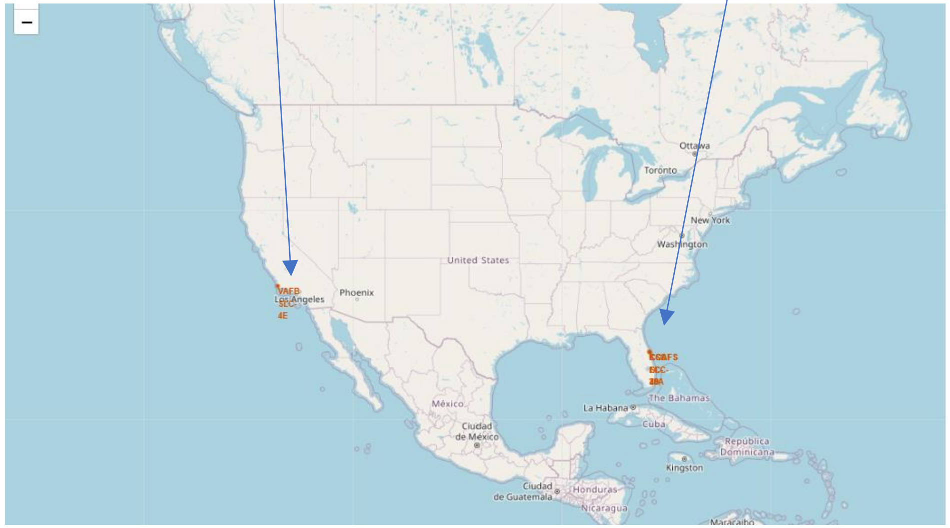
Space X Launch Sites

They have 4 Launch sites : one in the West Coast and the 3 others in Florida

All 4 are close to the coastline for safety reasons

They are all in proximity to the Equator line which is better for rocket trajectory with a small advantages to the sites in Florida

	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



Launch outcome for each site

We realize that the largest number of launches took place in CCAFS LC-40 but with limited success (26%)

KSC LC-39A demonstrated the best success rate (76%)

VAFB SLC-4E

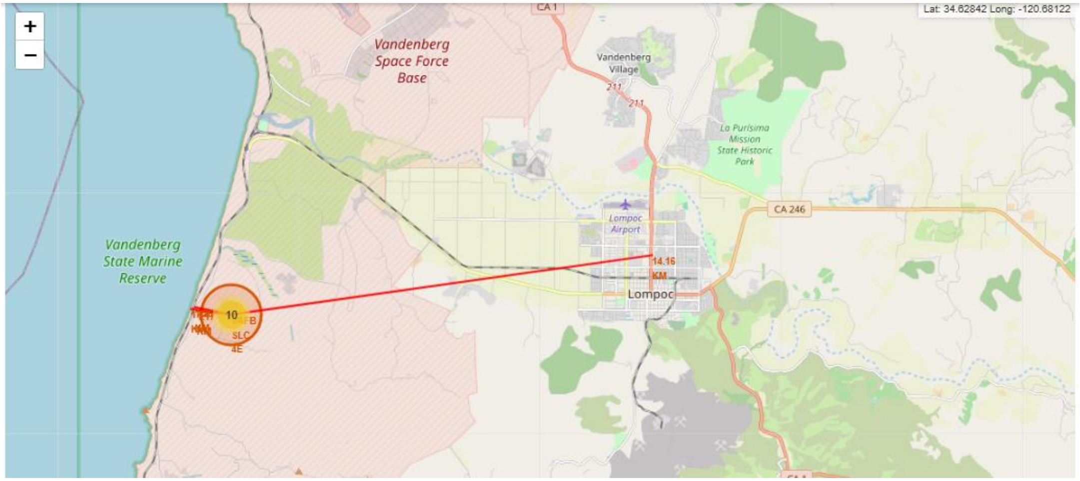
KSC LC-39A

CCAFS SLC-40

CCAFS LC-40

Launch sites proximities

- VAFB SLC-4E on the West Coast :
  - The site is situated about 1 km from the Coast, the Rail Way and the closest road
  - The nearest city is Lompoc at 14 km
- CCAFS SLC-40 and CCAFS LC-40 are more interesting on those criteria since Railway and Road are at less than 1 km and the nearest city Titusville is at more than 25 km
- Conclusion : all 4 sites are sufficiently close to Highways/Railways to facilitate the transportation of the rocket on ground and sufficiently far from the nearest city for safety reason

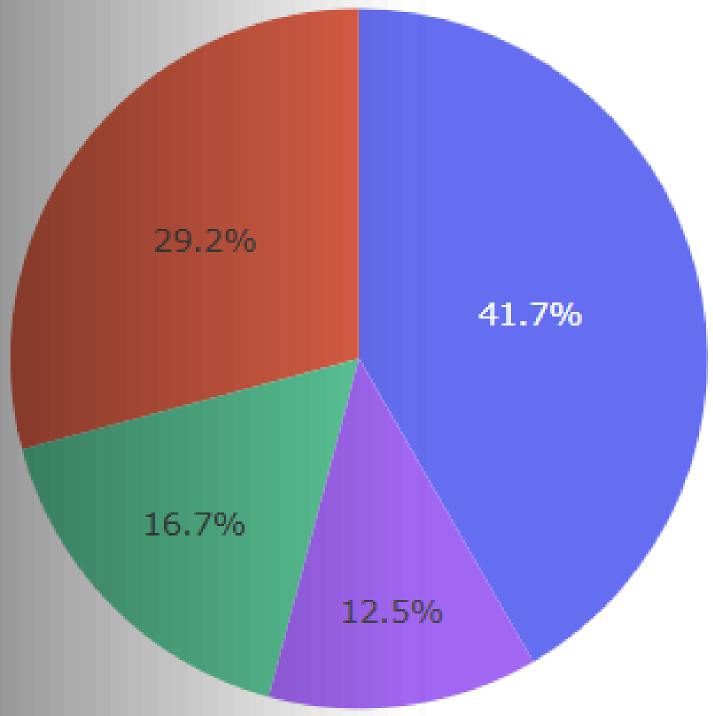


Section 4

Build a Dashboard with Plotly Dash

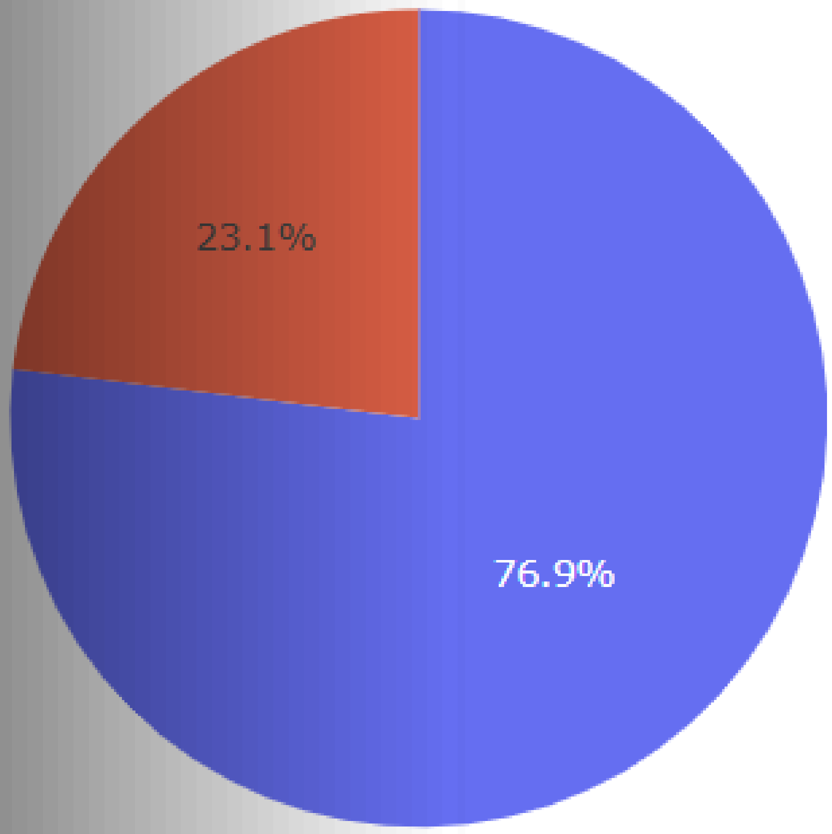
Launch success for all sites

KSC LC-39A Launch site has the best record for success landing

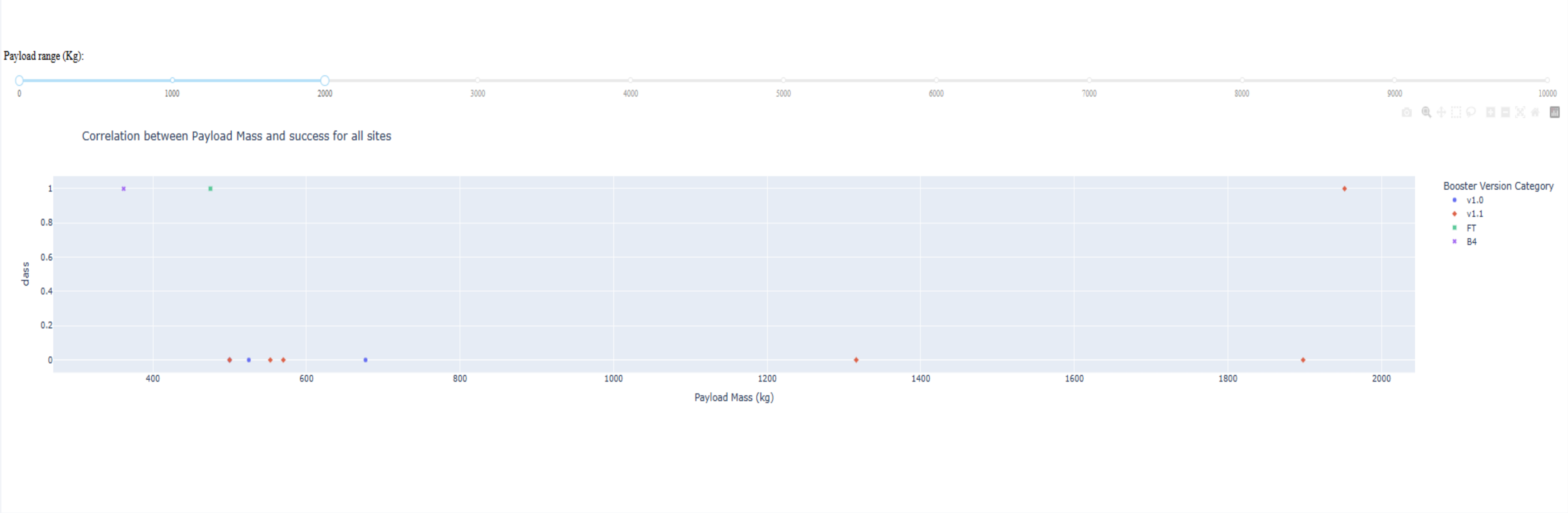


Success rate on the best site

KSC LC-39A Launch site has a good succes rate approaching 77%

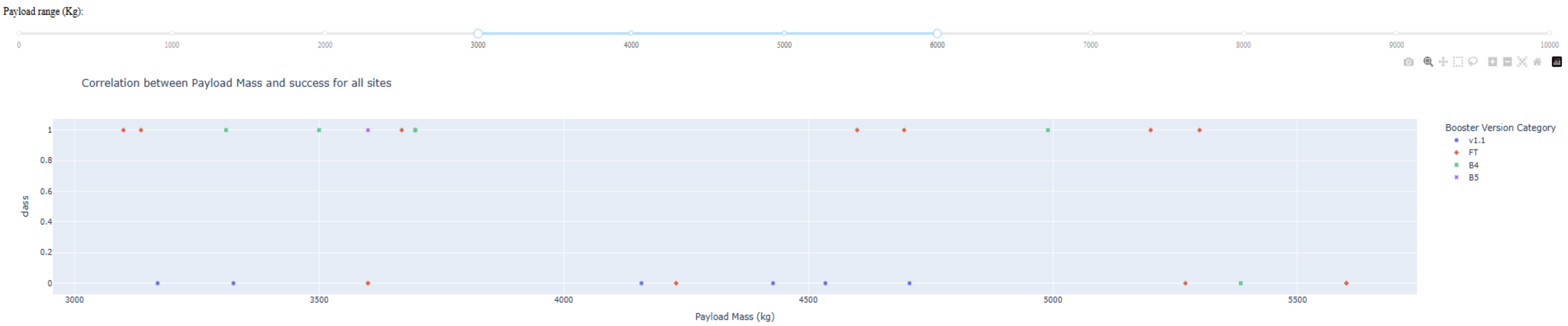


Correlation between Payload and Launch outcome



- For Smal Payloads, in the range 0-2000kg, we can see that V1.0 and v1.1 boosters perform poorly

Correlation between Payload and Launch outcome



- For Mid-range Payloads, in the range 3000-6000kg, which is the most common, we can see that V1.0 and v1.1 boosters perform poorly
- FT booster and B4 booster perform relatively well in this range with some failures as it is a critical range

Correlation between Payload and Launch outcome

- For Heavy Payloads, in the range 6000-10000kg, we can see that the failure rate is rather high for FT and B4 booster
- For Very Heavy payload mass (e.g. 15 tons), only B5 booster is used with good success (refer to slide 17 and 29)

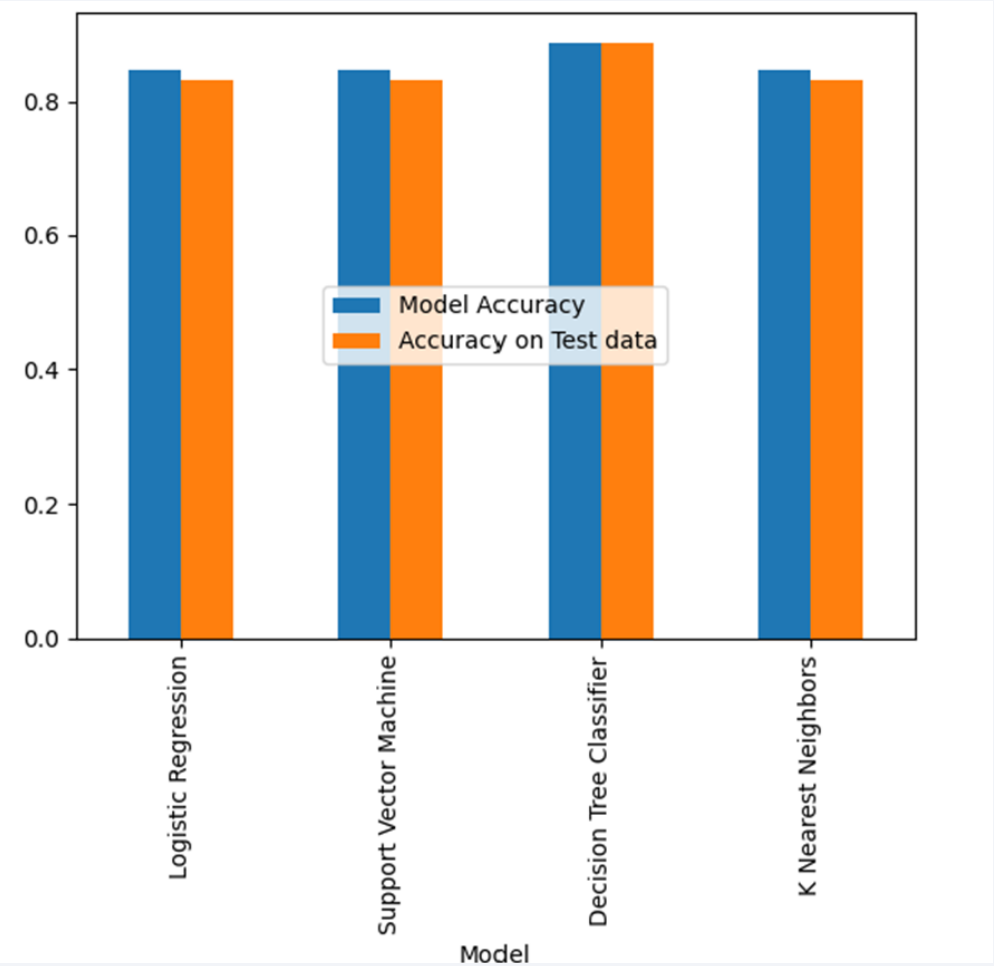
Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Model accuracy and accuracy on test data is shown on the bar graph for each Model that has been tested

- Decision Tree Classifier is the best model with Model accuracy of 0,8767 and Accuracy on Test data of 0,8889



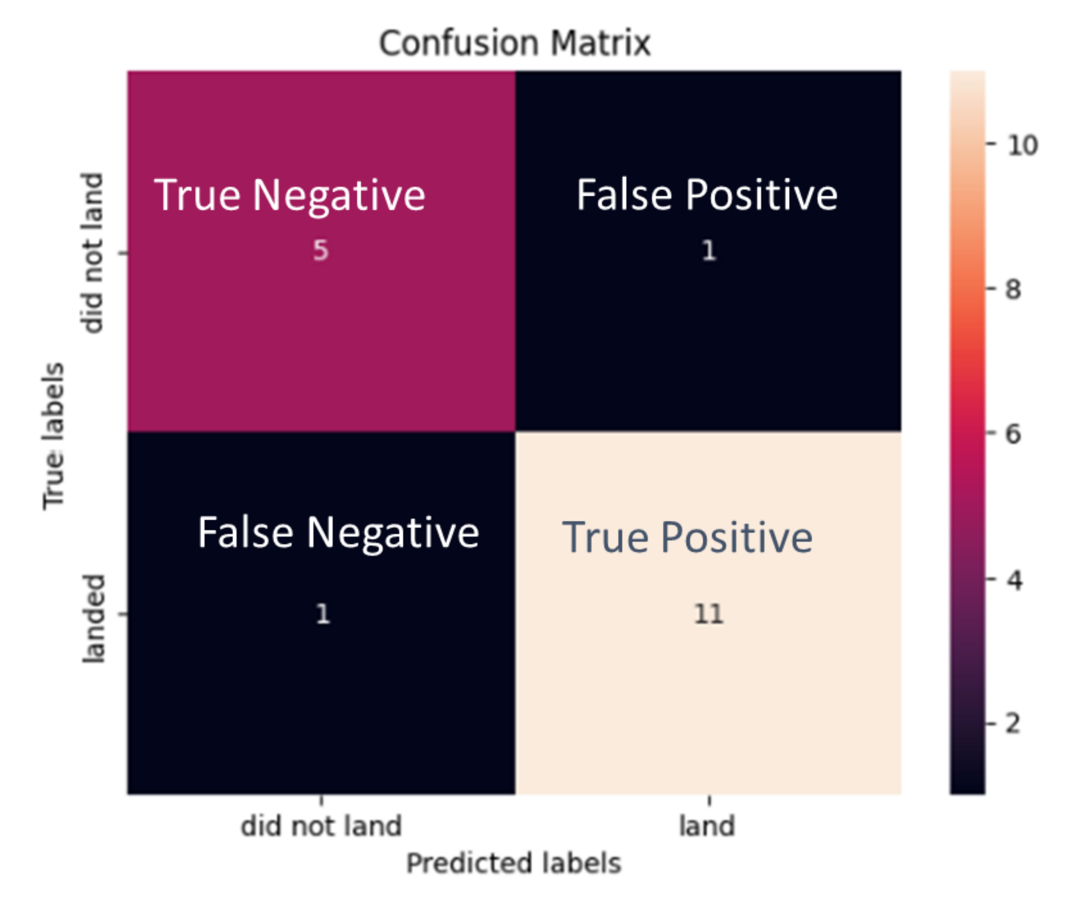
Confusion matrix

- Confusion matrix of the best performing model Decision Tree Classifier

- Parameters of the Best Classifier are:

```
Best estimator found: DecisionTreeClassifier(criterion='entropy', max_depth=6, max_features='sqrt',
                       min_samples_split=10, splitter='random')
```

- It can distinguish between different classes quite well. False Positive and Fase Negative are only 1 so it is a good predictor for the 1st stage landing success



Conclusions

- Space X uses 4 Launch sites. The 3 ones in Florida are closer to Equator Line and have better proximities that the one in Califormia
- CCAFS SLC 40 was the first Launch test at the beginning of the Space X history so the success rate is rather small. VABF SLC-4E in California does not operate payload above 9t because it is farther from the Equator line. KSC LC-39A is the best Launch site with 77% of 1st stage landing success.
- High altitude orbit (e.g. GTO) and payload in the range 4000-6000 kg have not so good success rate
- V1.0 and v1.1 booster version did not perform well enough whereas FT booster and B4 booster show better performance especially starting from 2017/Flight Number 60. Booster B5 is reserved for heavy payload (15 tons) and performs well.
- With all the available data, we have built a Decision Tree Classifier that predicts the landing outcome with good confidence. It can be used by Space Y to decide on a bid against SpaceX.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Appendix 1 : SQL queries used – Full list

%sql select distinct "Launch_Site" from SPACEXTABLE

%sql SELECT * from SPACEXTABLE WHERE "Launch_Site" like 'CCA%' LIMIT 5

%sql SELECT "Customer", (SELECT SUM("PAYLOAD_MASS_KG_") from SPACEXTABLE) as PayLoad_Total_Mass from SPACEXTABLE; (subquery)

%sql select "Customer", sum("PAYLOAD_MASS_KG_") as PayLoad_Total_Mass from SPACEXTABLE group by "Customer";

%sql SELECT "Booster_Version", AVG("PAYLOAD_MASS_KG") as Payload_Mass_Avg from SPACEXTABLE GROUP BY "Booster_Version“

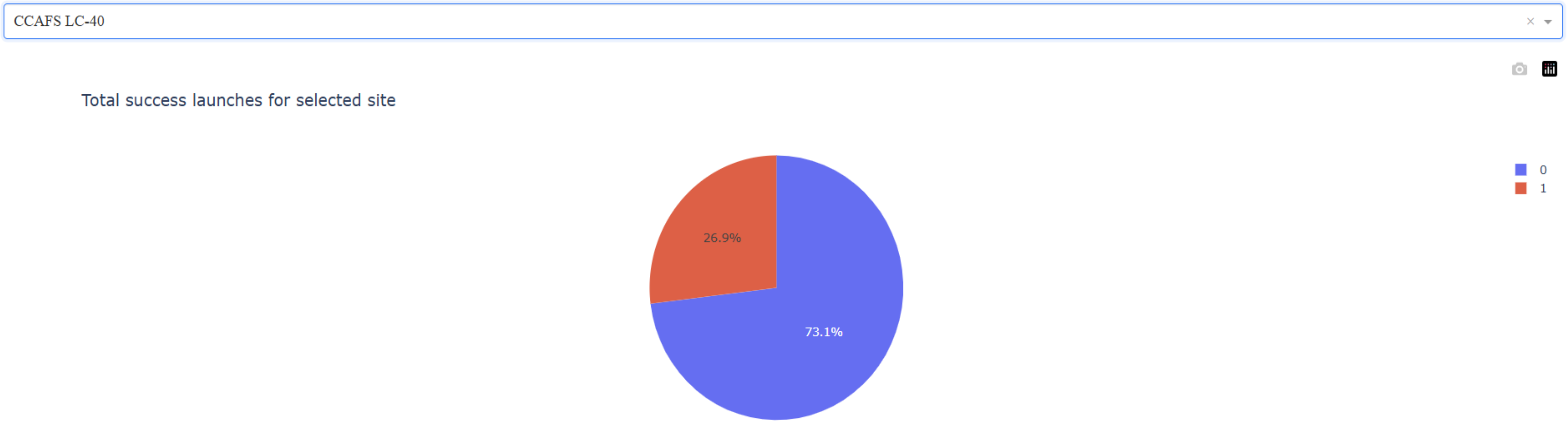
%sql SELECT min(date) as Minimum_Date, Landing_Outcome from SPACEXTABLE GROUP BY "Landing_Outcome“

%sql SELECT "Booster_Version","Landing_Outcome","PAYLOAD_MASS_KG_" WHERE "Landing_Outcome"='Success (drone ship)' AND "PAYLOAD_MASS_KG_" BETWEEN 4000 AND 6000;

%sql SELECT substr(Date, 6,2) as "Month_Name","Landing_Outcome","Booster_Version","Launch_Site" from SPACEXTABLE WHERE "Landing_Outcome"='Failure (drone ship)' and substr(Date,0,5)='2015';

%sql SELECT "Landing_Outcome", count("Landing_Outcome") as Landing_Outcome_Count from SPACEXTABLE WHERE "Date" between '2010-06-04' and '2017-03-20' GROUP BY "Landing_Outcome" ORDER BY Landing_Outcome_Count DESC;

Appendix 2 : Success rate of sites

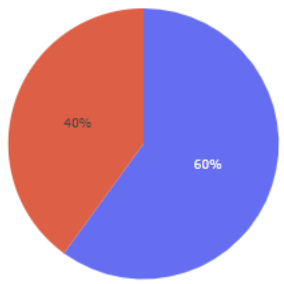


Appendix 2 : Success rate of sites

SpaceX Launch Records Dashboard

VAFB SLC-4E

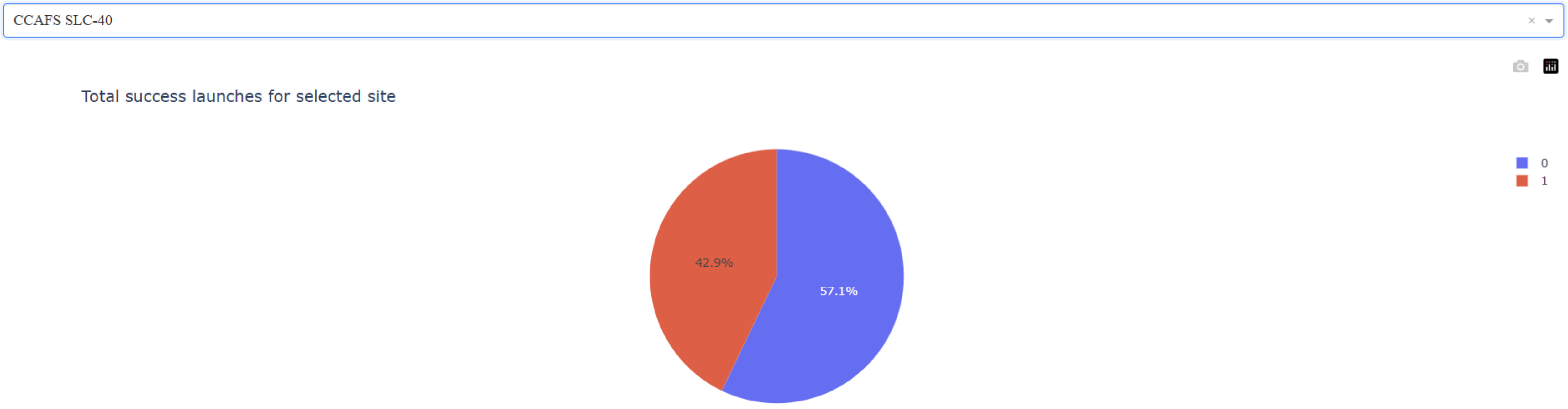
Total success launches for selected site



0
1

Payload range (Kg):

Appendix 2 : Success rate of sites



Appendix 3 : Machine Learning Model Comparison

Logistic Regression

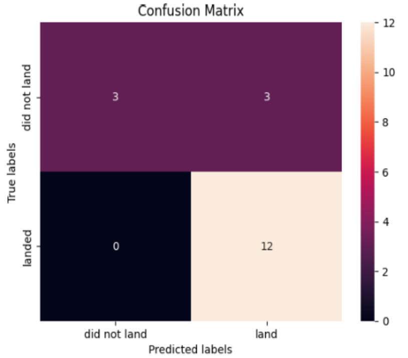
```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("Best estimator found:", logreg_cv.best_estimator_)
print("accuracy :",logreg_cv.best_score_)

tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
Best estimator found: LogisticRegression(C=0.01)
accuracy : 0.8464285714285713
```

```
Yhat=logreg_cv.best_estimator_.predict(X_test)
[19]: logreg_cv.best_estimator_.score(X_test,Y_test)
[19]: 0.8333333333333334
[22]: from sklearn.metrics import classification_report
[23]: # let us look at the classification report which provides precision, recall, F1 score and support for
print(classification_report(Y_test,Yhat))

              precision    recall  f1-score   support

           0       1.00      0.50      0.67         6
           1       0.80      1.00      0.89        12

    accuracy                           0.83        18
   macro avg       0.90      0.75      0.78        18
weighted avg       0.87      0.83      0.81        18
```

Lets look at the confusion matrix:

```
[24]: plot_confusion_matrix(Y_test,Yhat)
```

12 TP but 3 FP



Support Vector Machine

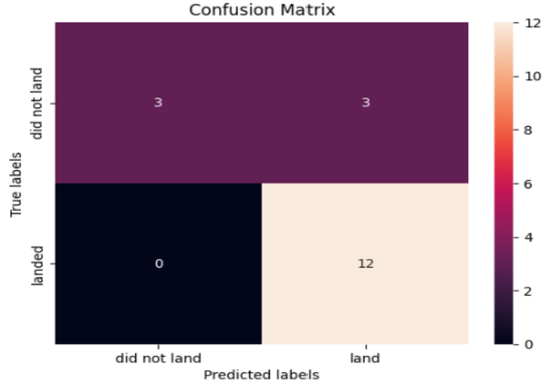
```
print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
print("Best estimator found:", svm_cv.best_estimator_)
print("accuracy :", svm_cv.best_score_)

tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
Best estimator found: SVC(gamma=0.03162277660168379, kernel='sigmoid')
accuracy : 0.8482142857142856
```

```
svm_cv.best_estimator_.score(X_test,Y_test)

0.8333333333333334
```

```
# let us look at the classification report which provides precision, recall, F1 sc
Yhat=svm_cv.best_estimator_.predict(X_test)
print(classification_report(Y_test,Yhat))

              precision    recall  f1-score   support

           0       1.00      0.50      0.67         6
           1       0.80      1.00      0.89        12

    accuracy                           0.83        18
   macro avg       0.90      0.75      0.78        18
weighted avg       0.87      0.83      0.81        18
```

We can plot the confusion matrix

```
Yhat=svm_cv.best_estimator_.predict(X_test)
plot_confusion_matrix(Y_test,Yhat)
```



KNN model

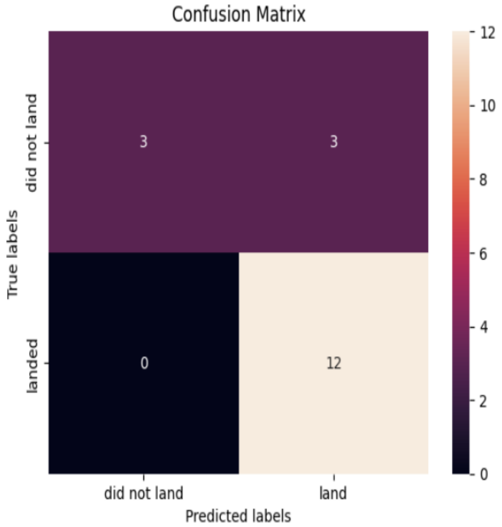
```
70]: print("tuned hpyerparameters :(best parameters) ",knn_cv.best_params_)
print("Best estimator found:", knn_cv.best_estimator_)
print("accuracy :",knn_cv.best_score_)

tuned hpyerparameters :(best parameters)  {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
Best estimator found: KNeighborsClassifier(n_neighbors=10, p=1)
accuracy : 0.8482142857142858
```

Calculate the accuracy of knn_cv on the test data using the method `score`:

```
knn_cv.best_estimator_.score(X_test,Y_test)

0.8333333333333334
```

We can plot the confusion matrix

```
yhat = knn_cv.best_estimator_.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Thank you!