Data Science Capstone Project

Víctor Gutiérrez Castillo

02/09/2021

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- Summary of methodologies.
  - Throughout the development of this course the main methodologies developed to collect data have been the use of an API or with the Beautiful Soup package. In order to make an Exploratory Data Analysis (EDA) we have made use of SQL, using Pyplot and Seaborn, and regarding the interactive analysis we have used Folium and Plotly. Finally, we have concluded with several techniques for Predictive Analysis, as Logistic Regression, Support Vector Machine, Decision Tree Classifier and K-Nearest Neighbours.
- Summary of all results.
  - Concerning the results achieved, we have been able to implement techniques to predict the success of reusing the first stage of the Falcon 9 rocket. The conclusions have been a good accuracy for the ML-techniques applied and the assumption of been able to use these models to predict the landing of the rocket.

Introduction

- Project background and context.
  - The main reason to develop this study is to predict the chances of been able to reuse the first stage of the rocket Falcon 9. After too many tries it has not always been possible to recover this part of the rocket and having the chance to know the future of this rocket will allow the company Space X to save a lot of money.
- Problems you want to find answers.
  - Could be possible to implement a Machine Learning algorithm to predict the likelihood of recovering the first stage of the rocket Falcon 9?

Methodology

- Data collection methodology:
  - Data has been collected using two different ways. On the one hand, we have made use of an API and requested the data with the SpaceX API. On the other hand, we have used the information that appears in the Wikipedia and extracted it with the package Beautiful Soup.
- Perform data wrangling:
  - First of all, we have reviewed the fields of the data and achieved some basics information about them. Then, we have created a new field called ‘Outcome’ and classified each fight with 1 if the landing outcome was a success or 0 if the landing outcome was not.
- Perform exploratory data analysis (EDA) using visualization and SQL:
  - Once loaded the dataset in the Db2 database, using the SQL magic package we have implemented several queries in order to visualize the dataset. After that, Matplotlib, Seaborn and Plotly have been the packages that we have used to represent graphically the data.
- Perform interactive visual analytics using Folium and Plotly Dash:
  - Making use of maps and interactive visualization we have been able to analyze the results of rockets launched in several occasions.
- Perform predictive analysis using classification models
  - Thanks to several techniques of classification, as Logistic Regression, Support Vector Machine, Decision Tree Classifier and K-Nearest Neighbors, we have tried to predict the future of the rocket Falcon 9.

Methodology

Data collection

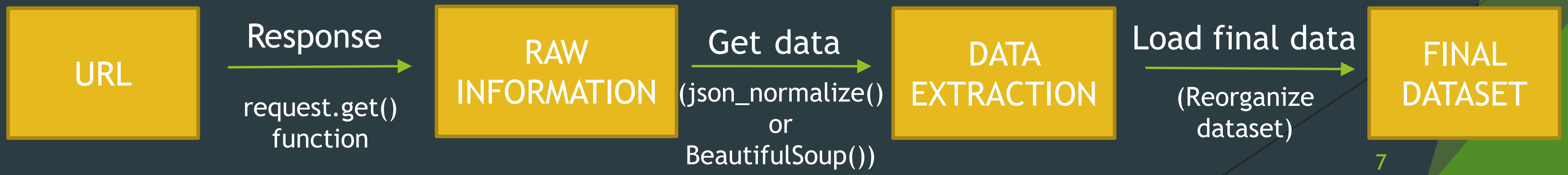
- The data sets used have been collected in two different ways:

SpaceX API

1. Making a request with the *request* package we have collected data from the SpaceX API.
2. The function *.get()* lets us to obtain the information.
3. The functions *.json()* and *.json_normalize()* allowed us to obtain a dataframe.
4. We used several functions to add the information to the dataframe.

Beautiful Soup Package

1. Making use of an URL of the Wikipedia Web Page we have applied the function *.get()*.
2. The function BeautifulSoup() allowed us to obtain the BeautifulSoup object.
3. After finding the table with *.find_all()* we used a For loop to read the BeautifulSoup object.
4. We achieved the dataframe object with all the information.



URL

Response

request.get() function

RAW INFORMATION

Get data

(json_normalize() or BeautifulSoup())

DATA EXTRACTION

Load final data

(Reorganize dataset)

FINAL DATASET

Data collection – SpaceX API

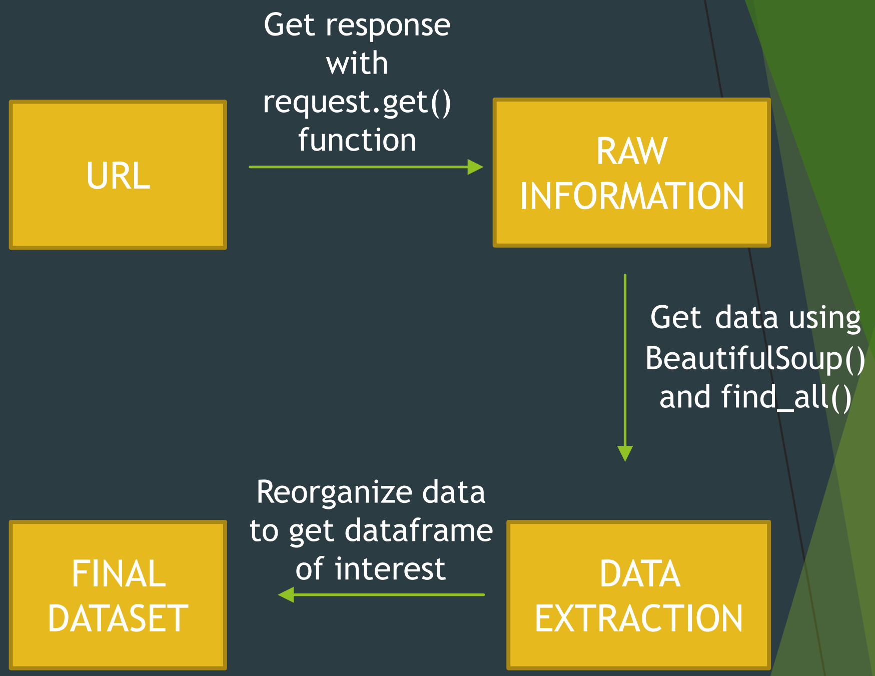
The main functions used has been:

1. requests.get()
2. .json()
3. .json_formalize()
4. Functions to fill the global variables.

The URL where the notebook can be found is:

https://github.com/vicguti/Data-Science-Capstone-Project/blob/master/Data%20Collection%20API%20Lab.ipynb

Analogous to the flowchart from previous slide, in this case we have:


URL
Get response with request.get() function
RAW INFORMATION
Get data using BeautifulSoup() and find_all()
DATA EXTRACTION
Reorganize data to get dataframe of interest
FINAL DATASET

Data collection – Web scraping

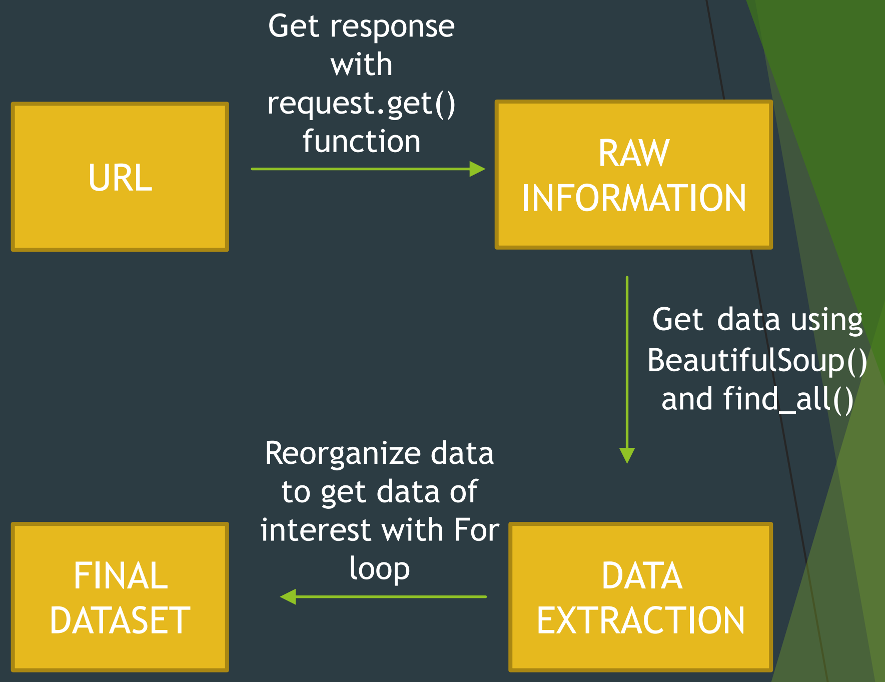
The main functions used has been:

1. requests.get()
2. BeautifulSoup()
3. .find_all()
4. For loop to retrieve data of interest.

The URL where the notebook can be found is:

https://github.com/vicguti/Data-Science-Capstone-Project/blob/master/Data%20Collection%20with%20Web%20Scraping.ipynb

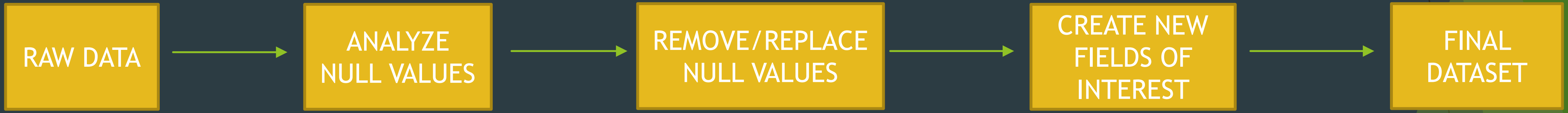
Analogous to the flowchart from previous slide, in this case we have:



URL
Get response with request.get() function
RAW INFORMATION
Get data using BeautifulSoup() and find_all()
DATA EXTRACTION
Reorganize data to get data of interest with For loop
FINAL DATASET

Data wrangling

- The first step taken was to analyze the missing values and have a simple idea of the amounts of data in the different fields. After that we wanted to create a new field in order to classify each flight as success (1) or failure (0). The logic uses to this was based on the field *Outcome*, where we could watch the result of each flight.
- The main packages used were Pandas and Numpy, with functions as *.isnull()*, *.count()* and *.value_counts()*.



RAW DATA → ANALYZE NULL VALUES → REMOVE/REPLACE NULL VALUES → CREATE NEW FIELDS OF INTEREST → FINAL DATASET

- The URL to watch the notebooks about Data wrangling is:

https://github.com/vicguti/Data-Science-Capstone-Project/blob/master/Data%20Wrangling.ipynb

EDA with data visualization

- The main charts used were:
  - Scatter plots: It allows us to visualize field where one axis is a categorical variable, as *Class*.
  - Pie plots: It allows us to visualize from a glance the percentage of success and failures.
  - Categorical plots: Using two nominal variables, this plot let us to introduce a categorical variable to analyze the success of each flight depending of the two nominal variables.
  - Bar plots: It allows us to study the relationship between two different variables, in this case success rate an orbit type.
  - Linear plots: The most common use is to recognize trends and to study correlation between variables.

- The URL to watch the notebooks about Eda with data visualization is:

https://github.com/vicguti/Data-Science-Capstone-Project/blob/master/Complete%20the%20EDA%20with%20visualization.ipynb

EDA with SQL

- The main queries used had been:
  - %sql SELECT DISTINCT launch_site FROM SPACEXDATASET;
  - %sql SELECT * FROM SPACEXDATASET WHERE LAUNCH_SITE LIKE 'KSC%' LIMIT 5;
  - %sql SELECT CUSTOMER, SUM(PAYLOAD_MASS__KG_) AS "TOTAL PAYLOAD" FROM SPACEXDATASET WHERE CUSTOMER = 'NASA (CRS)' GROUP BY CUSTOMER ;
  - %sql SELECT BOOSTER_VERSION, AVG(PAYLOAD_MASS__KG_) AS "AVERAGE PAYLOAD" FROM SPACEXDATASET WHERE BOOSTER_VERSION = 'F9 v1.1' GROUP BY BOOSTER_VERSION;
  - %sql SELECT min(DATE) as MIN_DATE, LANDING__OUTCOME FROM SPACEXDATASET WHERE LANDING__OUTCOME = 'Success (drone ship)' GROUP BY LANDING__OUTCOME;
  - %sql SELECT booster_version FROM SPACEXDATASET WHERE LANDING__OUTCOME = 'Success (ground pad)' AND PAYLOAD_MASS__KG_>4000 AND PAYLOAD_MASS__KG_<6000;
  - %sql SELECT MISSION_OUTCOME, COUNT(*) AS CUANTITY FROM SPACEXDATASET GROUP BY MISSION_OUTCOME;
  - %sql SELECT BOOSTER_VERSION FROM SPACEXDATASET WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXDATASET);
  - %sql SELECT MONTHNAME(DATE), LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXDATASET WHERE LANDING__OUTCOME ='Success (ground pad)' AND YEAR(DATE)=2017;
  - %sql SELECT * FROM SPACEXDATASET WHERE LANDING__OUTCOME LIKE 'Success%' AND DATE>=TO_DATE('2010-06-04','YYYY-MM-DD') AND DATE<=TO_DATE('2017-03-20','YYYY-MM-DD') ORDER BY DATE DESC ;

- The URL to watch the notebooks about EDA with SQL is:

https://github.com/vicguti/Data-Science-Capstone-Project/blob/master/Complete%20the%20EDA%20with%20SQL.ipynb

Build an interactive map with Folium

- The objects added to a folium map were:
  - **Circles:** We used it to locate in the map the locations from where the rockets were launched.
  - **Markers Cluster:** Due to the similarity of the coordinates between different launch we used this kind of Markers to group similar locations.
  - **Markers:** Single markers in order to create point to calculate the distance from launch locations.
  - **PolyLine:** We added lines to connect the launch locations with the markers that we selected in the map.

- The URL to watch the notebooks about the interactive map with Folium is:

https://github.com/vicguti/Data-Science-Capstone-Project/blob/master/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

- The plots and interactions added to the dashboard were:
  - **Drop-down:** The objective of this item was to allow the user to select a location among the 4 possibilities, apart from a new option that collected the 4 options *(All)*.
  - **Pie-chart:** It has a logic implemented that showed the amount of launched in each location if the Drop-down input is *All* or the percentage of success/failure if it was selected just one location.
  - **Range Slider:** It lets the user to introduce the range of *Payload Mass (kg)* to be taken into consideration by the Dashboard. The options were between 0 and 10000 with a step of 1000.
  - **Scatter plot:** It compare the *Payload Mass (kg)* field with the *Class*. It can be filtered by the location and furthermore to be enclosed by the Range Slider.

- The URL to watch the notebooks about the Dashborad with Plotly Dash is:

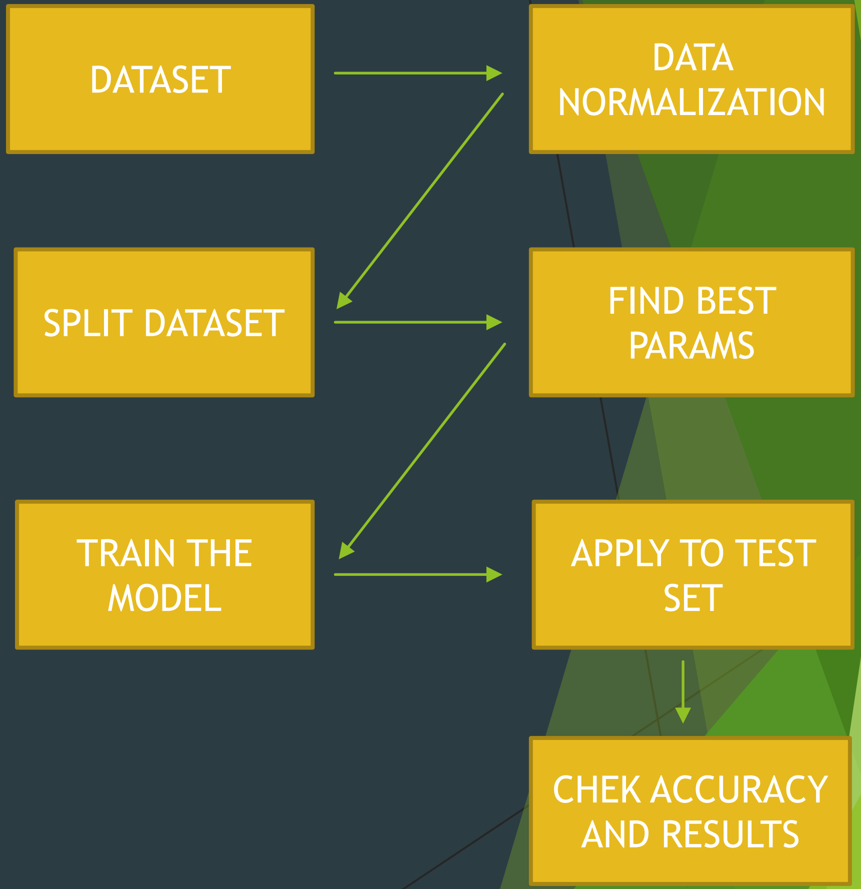
https://github.com/vicguti/Data-Science-Capstone-Project/blob/master/Dashboard%20with%20Plotly%20Dash.py

Predictive analysis (Classification)

- The first step after importing the different packages to implement the models is normalize the dataset in order to avoid scale or measure problems. Then we have to divide the data set into two subsets, one for training and other for testing. The function used is *trains_test_split()*.
- Once we have chosen the machine learning model we define a dictionary with several parameters. Using the functions *GridSearchCV()* the program choose the best parameters. Applying the *.fit()* function we train the model. To visualize the best parameters chosen from the dictionary there is the extension *.best_params_*.
- The last thing is to calculate the accuracy of the test subset with *.score()* and to visualize the estimation results with a confusion matrix.

- The URL to watch the notebooks about the Machine Learning Prediction is:

https://github.com/vicguti/Data-Science-Capstone-Project/blob/master/Complete%20the%20Machine%20Learning%20Prediction%20lab.ipynb



DATASET

DATA NORMALIZATION

SPLIT DATASET

FIND BEST PARAMS

TRAIN THE MODEL

APPLY TO TEST SET

CHEK ACCURACY AND RESULTS

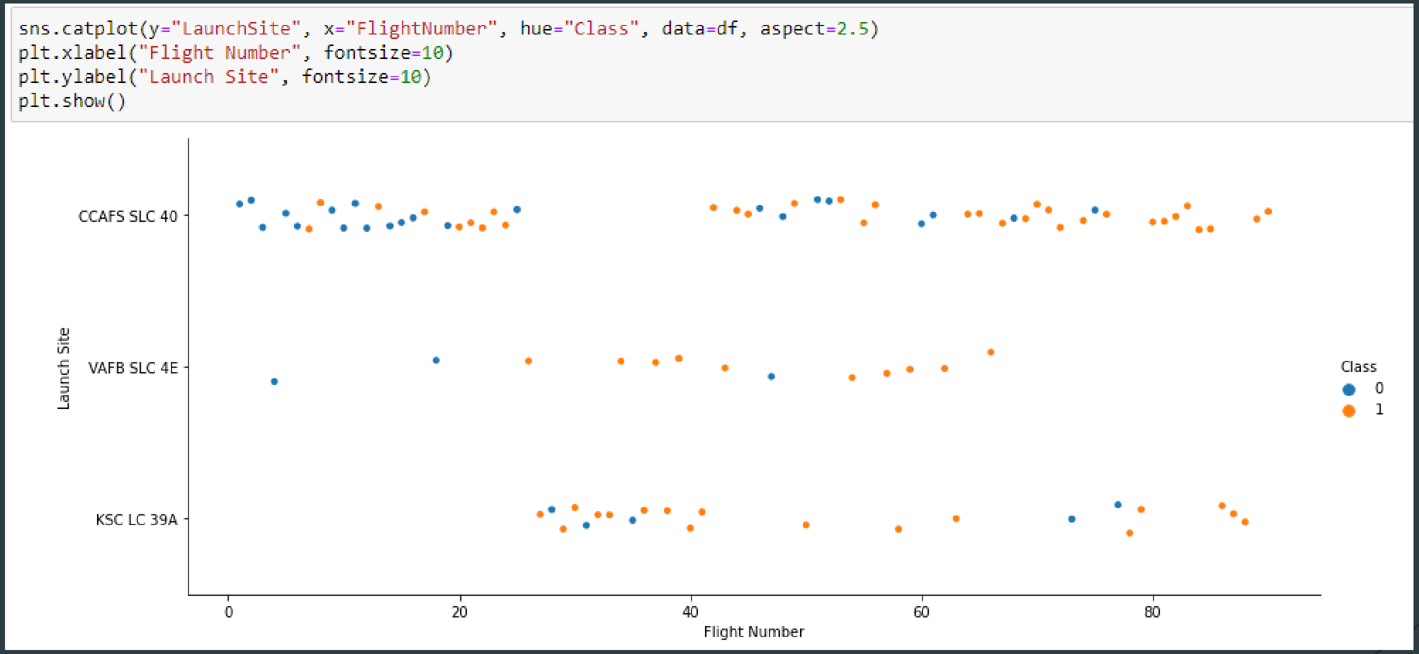
Results

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

EDA with Visualization

Flight Number vs. Launch Site

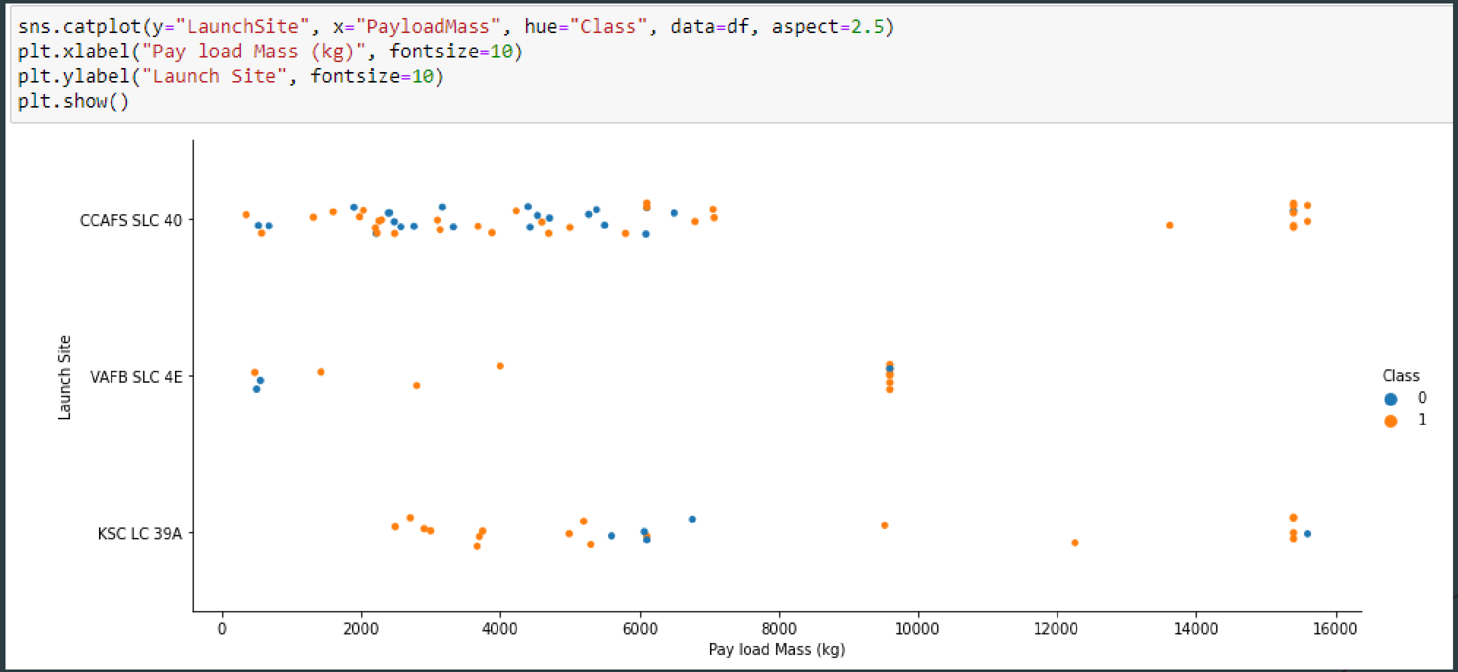
In this plot we can watch how the location CCAFS SLC 40 is the most used. In general it is possible to recognize how as bigger become the Flight Number, more launches has landed successfully. Furthermore, the location less common has ben VAFB SLC 4E.



```
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect=2.5)
plt.xlabel("Flight Number", fontsize=10)
plt.ylabel("Launch Site", fontsize=10)
plt.show()
```

Payload vs. Launch Site

This plot shows how the number of launches with a high Pay Load Mass (kg) is small. However, almost every launch with a high mass has landed successfully. If we focus on the location CCAFS SLC 40, we can watch that with a low mass there is not any trend but inthis case of VAFB SLC 4E almost every launches have landed successfully, especially around 10000 Kg.

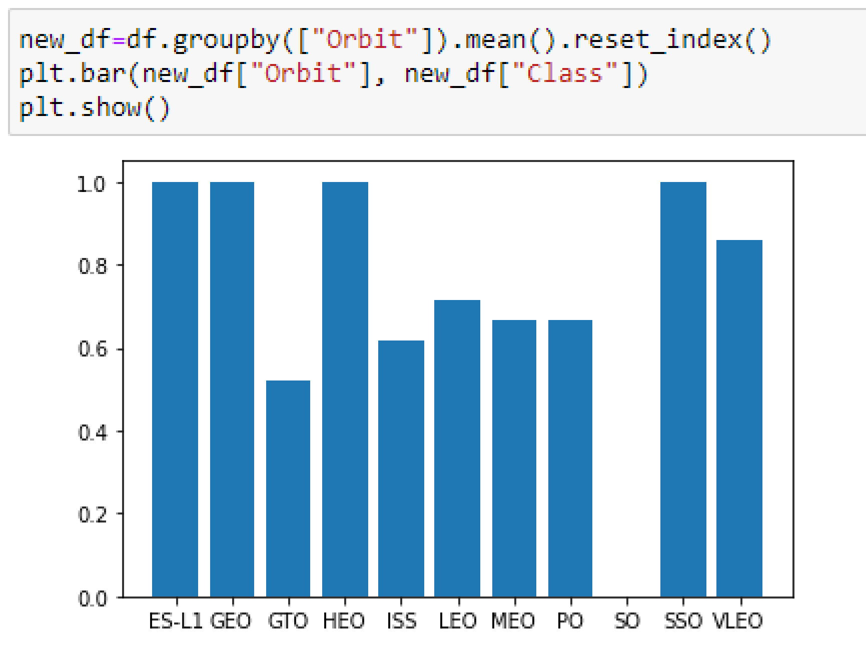


```
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect=2.5)
plt.xlabel("Pay load Mass (kg)", fontsize=10)
plt.ylabel("Launch Site", fontsize=10)
plt.show()
```

Success rate vs. Orbit type

It is clear that the orbit ES-L1, GEO HEO and SSO has the highest likelihood to land successfully. However, SO is the lowest one with a value of 0.

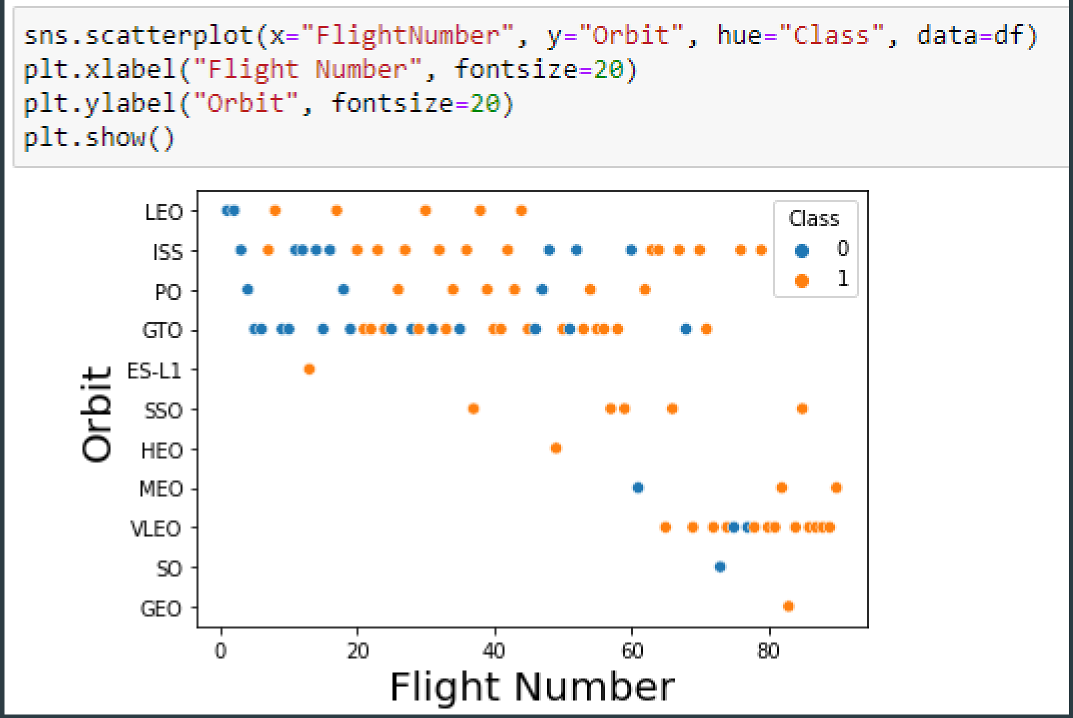
Other orbits like GTO, ISS, LEO, MEO, PO and even VLEO has a medium probability to land successfully. It is not very high but almost everyone surpass the 50%.



```
new_df=df.groupby(["Orbit"]).mean().reset_index()
plt.bar(new_df["Orbit"], new_df["Class"])
plt.show()
```

Flight Number vs. Orbit type

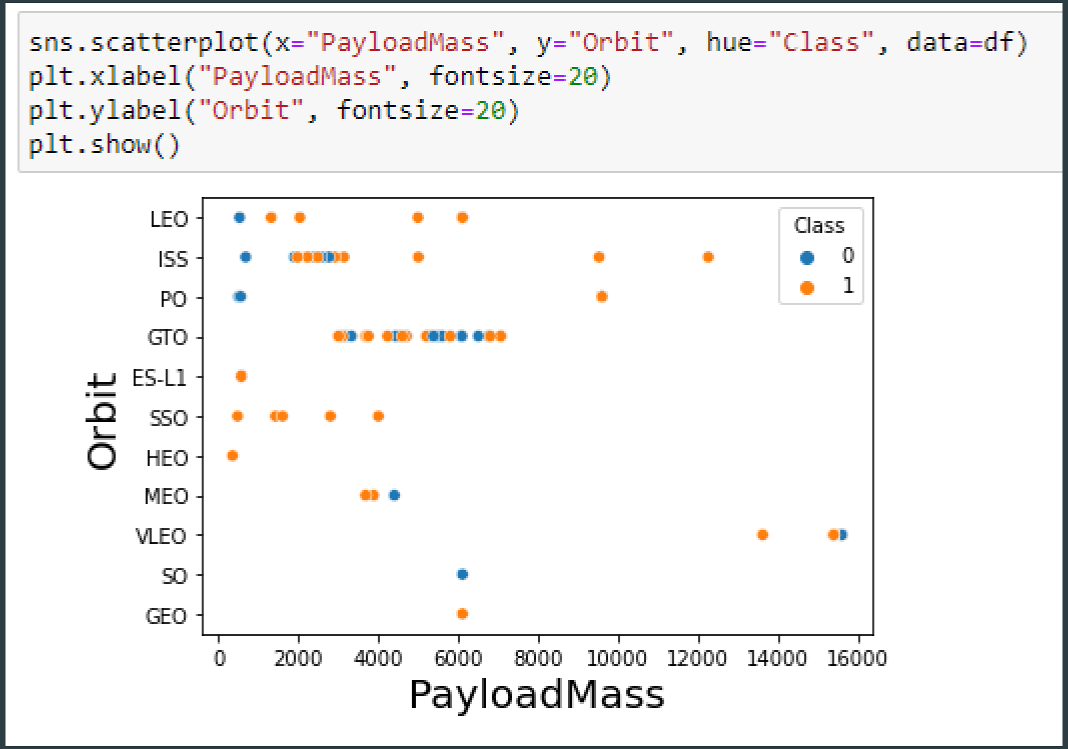
We can watch that in some orbits there are only successful lands, like ES-L1, SSO, HEO or GEO. There are other orbits like ISS, LEO or PO that have a positive trend to land successfully and there is other like GTO which has not any trend. However, we can appreciate a general positive trend.



```
sns.scatterplot(x="FlightNumber", y="Orbit", hue="Class", data=df)
plt.xlabel("Flight Number", fontsize=20)
plt.ylabel("Orbit", fontsize=20)
plt.show()
```

Payload vs. Orbit type

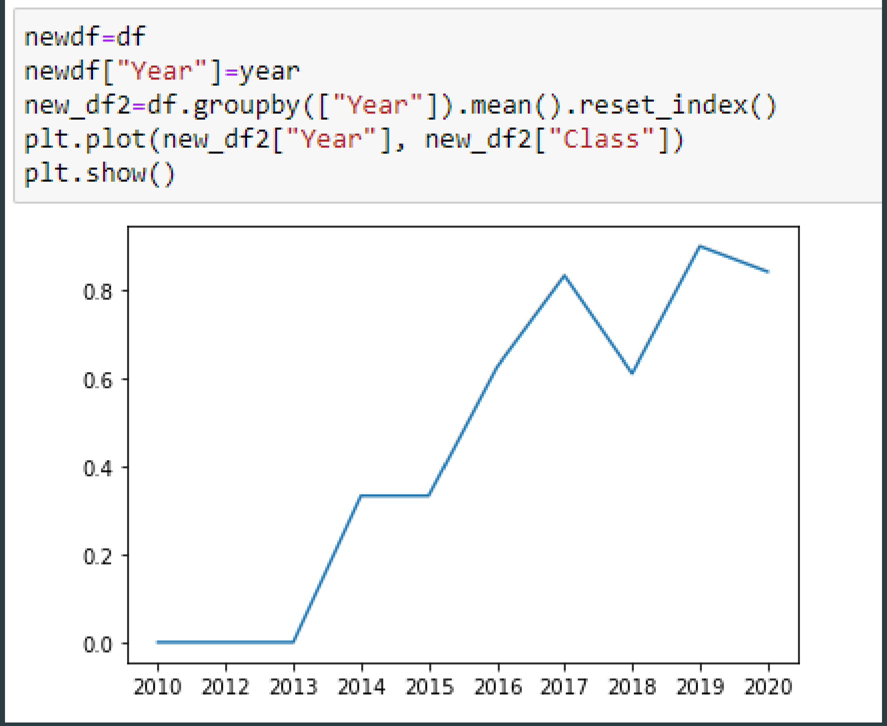
We can watch that heavy payloads have a negative influence on GTO orbits and positive on PO, LEO and ISS orbits. Furthermore, there are orbits that with a low payload have always landed successfully, like ES-L1, SSO and HEO.



```
sns.scatterplot(x="PayloadMass", y="Orbit", hue="Class", data=df)
plt.xlabel("PayloadMass", fontsize=20)
plt.ylabel("Orbit", fontsize=20)
plt.show()
```

Launch success yearly trend

We can watch how in 2013 happened the first successful launch. From that moment the likelihood started to increase, reaching a peak in 2017. Unfortunately, it decreased in 2018 but after that it began to increase again until reach its maximum in 2019.

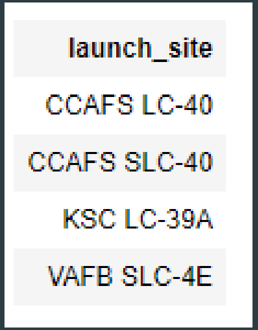


```
newdf=df
newdf["Year"]=year
new_df2=df.groupby(["Year"]).mean().reset_index()
plt.plot(new_df2["Year"], new_df2["Class"])
plt.show()
```

EDA with SQL

All launch site names

- The distinct places are:

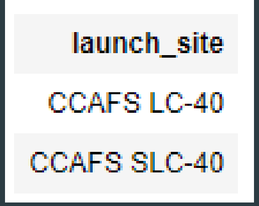


launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

- %sql SELECT DISTINCT launch_site FROM SPACEXDATASET;
  - This query select the unique values from the variable launch_site.

Launch site names begin with `CCA`

- The launch sites names are:

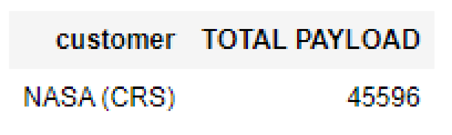


launch_site
CCAFS LC-40
CCAFS SLC-40

- %sql SELECT DISTINCT launch_site FROM SPACEXDATASET WHERE LAUNCH_SITE LIKE 'CCA%’;
  - Similar to the previous query in this case we have added a *like* condition to select only those launch sites that begin with ‘CCA’.

Total payload mass

- The total payload carried by boosters from NASA is 45596 Kg.

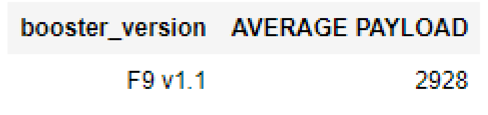


customer	TOTAL PAYLOAD
NASA (CRS)	45596

- %sql SELECT CUSTOMER, SUM(PAYLOAD_MASS__KG_) AS "TOTAL PAYLOAD" FROM SPACEXDATASET WHERE CUSTOMER = 'NASA (CRS)' GROUP BY CUSTOMER;
  - The purpose of this query is to select first every costumer called *NASA (CRS)* and then group by costumer, making an addition in the filed PAYLOAD_MASS__KG_.

Average payload mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 is 2928 Kg.

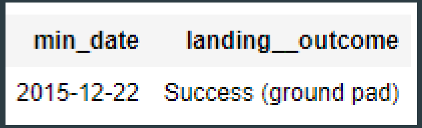


booster_version	AVERAGE PAYLOAD
F9 v1.1	2928

- %sql SELECT BOOSTER_VERSION, AVG(PAYLOAD_MASS__KG_) AS "AVERAGE PAYLOAD" FROM SPACEXDATASET WHERE BOOSTER_VERSION = 'F9 v1.1' GROUP BY BOOSTER_VERSION;
    - Repeating the process of the previous query, in this one we have filtered by BOOSTER_VERSION = 'F9 v1.1’ and, after grouping, we have calculated the average with the *avg()* function.

First successful ground landing date

- The date when the first successful landing outcome in ground pad occurred on 2015-12-22.

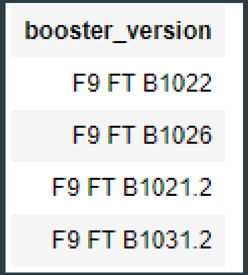


min_date	landing__outcome
2015-12-22	Success (ground pad)

- %sql SELECT min(DATE), LANDING__OUTCOME FROM SPACEXDATASET WHERE LANDING__OUTCOME = 'Success (ground pad)' GROUP BY LANDING__OUTCOME;
  - Repeating the same process, first of all we have filter by LANDING__OUTCOME = 'Success (ground pad)' and, after grouping, we have selected the minimum date with the *min()* function.

Successful drone ship landing with payload between 4000 and 6000

- The names of boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000 are:

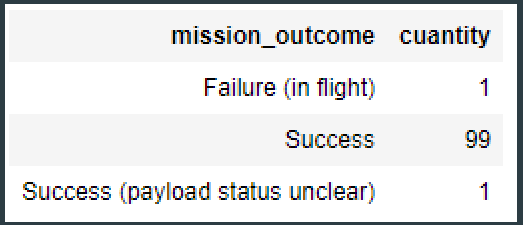


booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

- %sql SELECT booster_version FROM SPACEXDATASET WHERE LANDING__OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_>4000 AND PAYLOAD_MASS__KG_<6000;
  - In this case we have filtered the table SPACEXDATASET by LANDING__OUTCOME = 'Success (drone ship)’ and PAYLOAD_MASS__KG_>4000 and PAYLOAD_MASS__KG_<6000. Then, we have selected the field booster_version.

Total number of successful and failure mission outcomes

- The total number of successful and failure mission outcomes is:

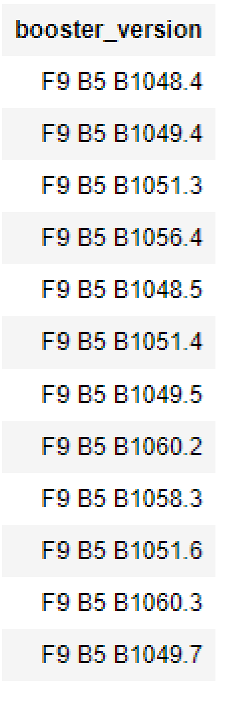


mission_outcome	cuantity
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

- %sql SELECT MISSION_OUTCOME, COUNT(*) AS CUANTITY FROM SPACEXDATASET GROUP BY MISSION_OUTCOME;
  - First of all we have grouped by mission_outcome and then we have used the *count()* function to calculated the number of successful and failure missions outcomes.

Boosters carried maximum payload

- The names of the booster which have carried the maximum payload mass are:

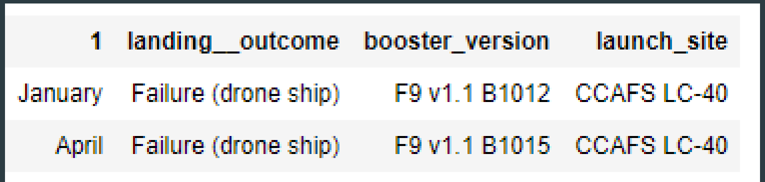


booster_version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

- %sql SELECT BOOSTER_VERSION FROM SPACEXDATASET WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXDATASET);
  - In this query we have used a subquery to calculate the maximum of the PAYLOAS_MASS__KG_ and then to filter the table to get the booster_version with the highest mass.

2015 launch records

- The list of records is:

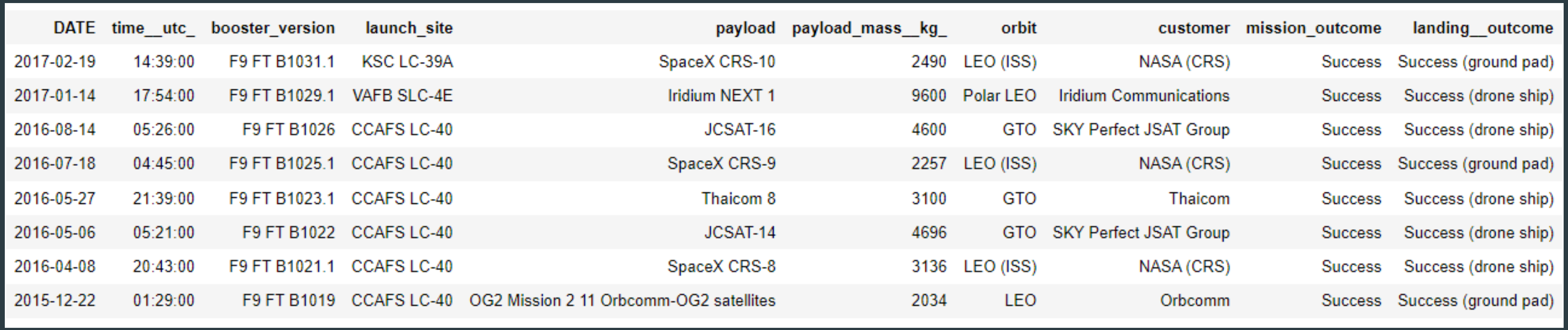


1	landing__outcome	booster_version	launch_site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- %sql SELECT MONTHNAME(DATE), LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXDATASET WHERE LANDING__OUTCOME ='Failure (drone ship)' AND YEAR(DATE)=2015;
  - What we have done is to filter the table by LANDING__OUTCOME ='Failure (drone ship)’ and YEAR(DATE)=2015 (Highlight the use of the *year()* function). Then, we have selected the columns of interest, using in this case the *monthname()* function.

Rank success count between 2010-06-04 and 2017-03-20

- The list of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.



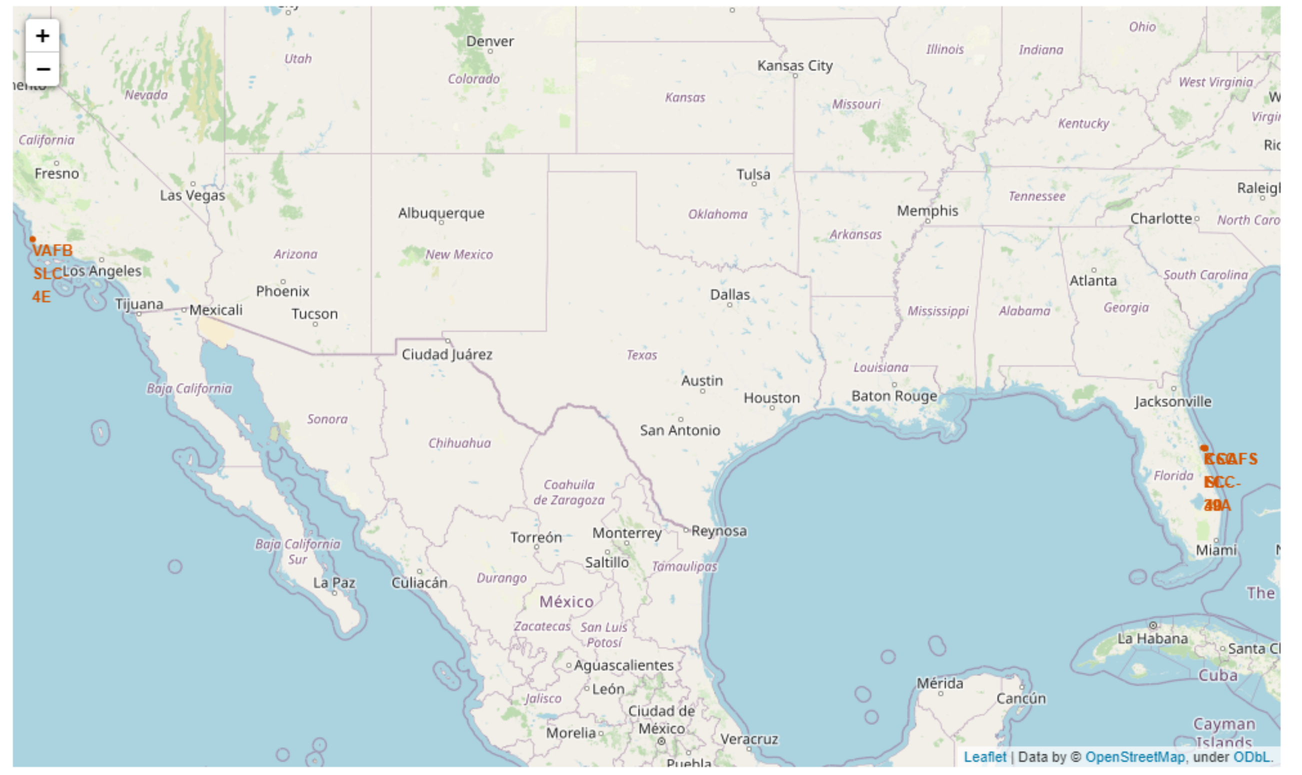
DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Success (drone ship)
2016-08-14	05:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-07-18	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Success (drone ship)
2016-05-06	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-04-08	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (drone ship)
2015-12-22	01:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

- %sql SELECT * FROM SPACEXDATASET WHERE LANDING__OUTCOME LIKE 'Success%' AND DATE>=TO_DATE('2010-06-04','YYYY-MM-DD') AND DATE<=TO_DATE('2017-03-20','YYYY-MM-DD') ORDER BY DATE DESC ;
  - We have just selected all the columns and filtered the table by the conditions shown in the query. We have used the *like* condition and *to_date()* function to control the date interval.

Interactive map with Folium

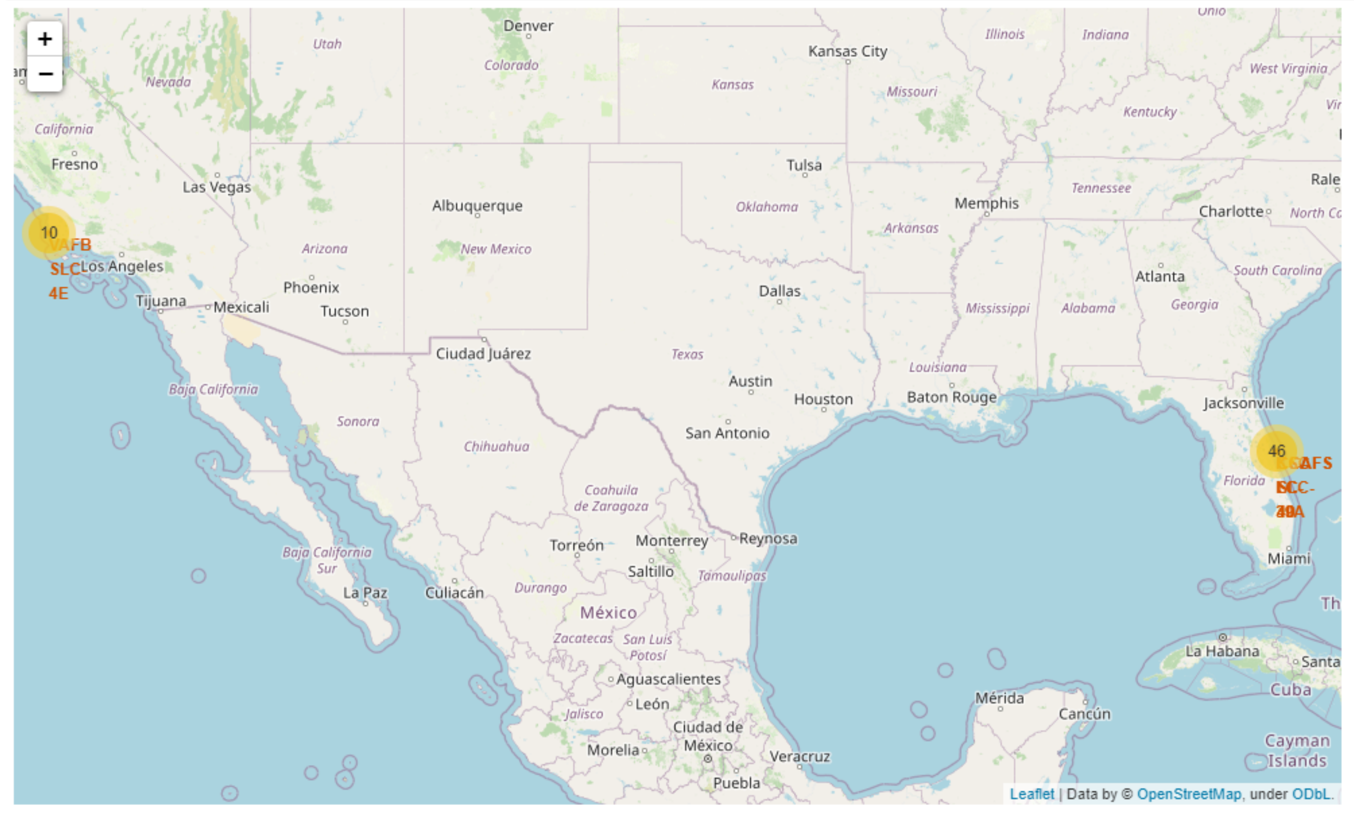
Launch sites’ locations

- The relevant elements used in this map from the folium package have been circles and marker.
- The results have been two different places located in opposite places of the USA. It could be due to the weather conditions, because it should be places with low wind speed and good climatological conditions.



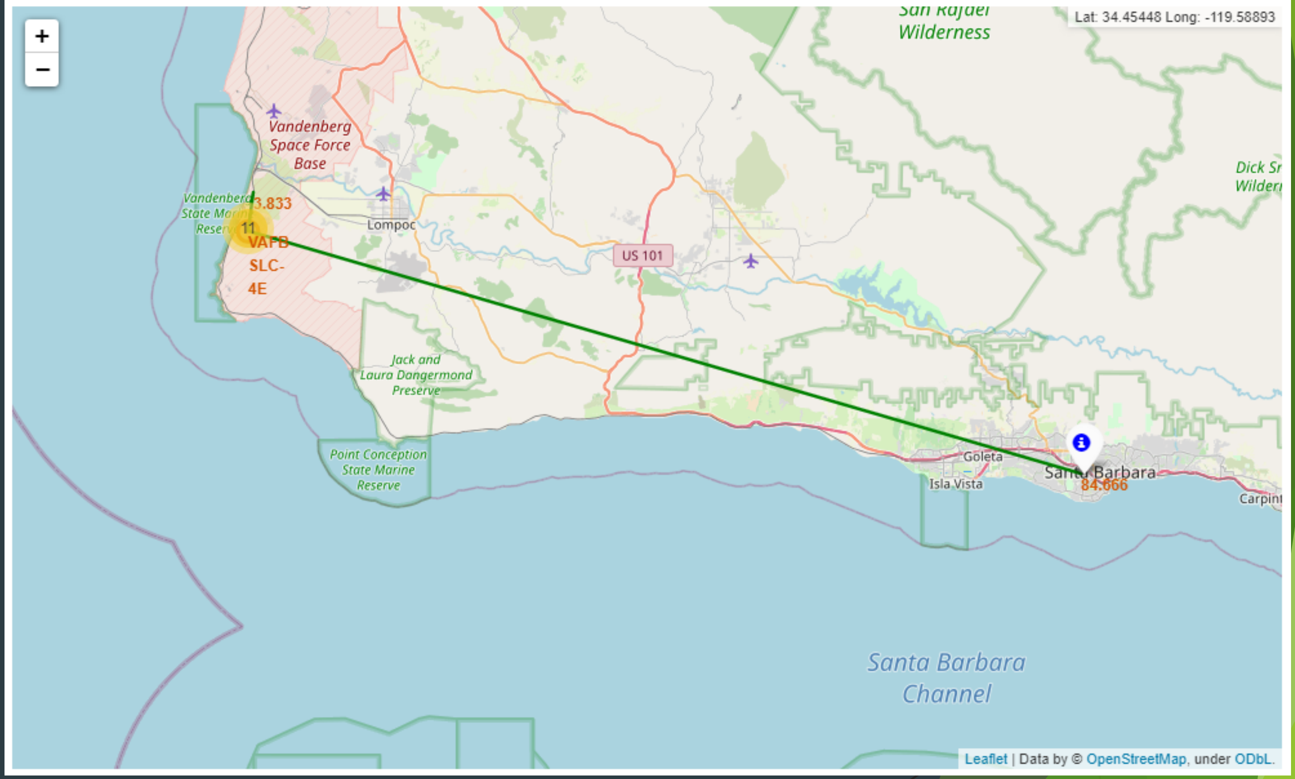
Labeled launch records

- The object applied from the folium package has been marker but defining first a maker_cluster to include all the launches with similar coordinates in the same cluster.
- In the image we can watch that there are 10 launches in California and 46 in Florida.



Distance to different locations

- To get both points we have used markers and then PolyLine to draw the lines. The places taken into account have been one near the launch site, in the railway, and the other one in the middle of Santa Barbara.
- The results can be watched in the image. We have achieved that the distance to the railway is 3,832 Km and to Santa Barbara is 84,666 Km.

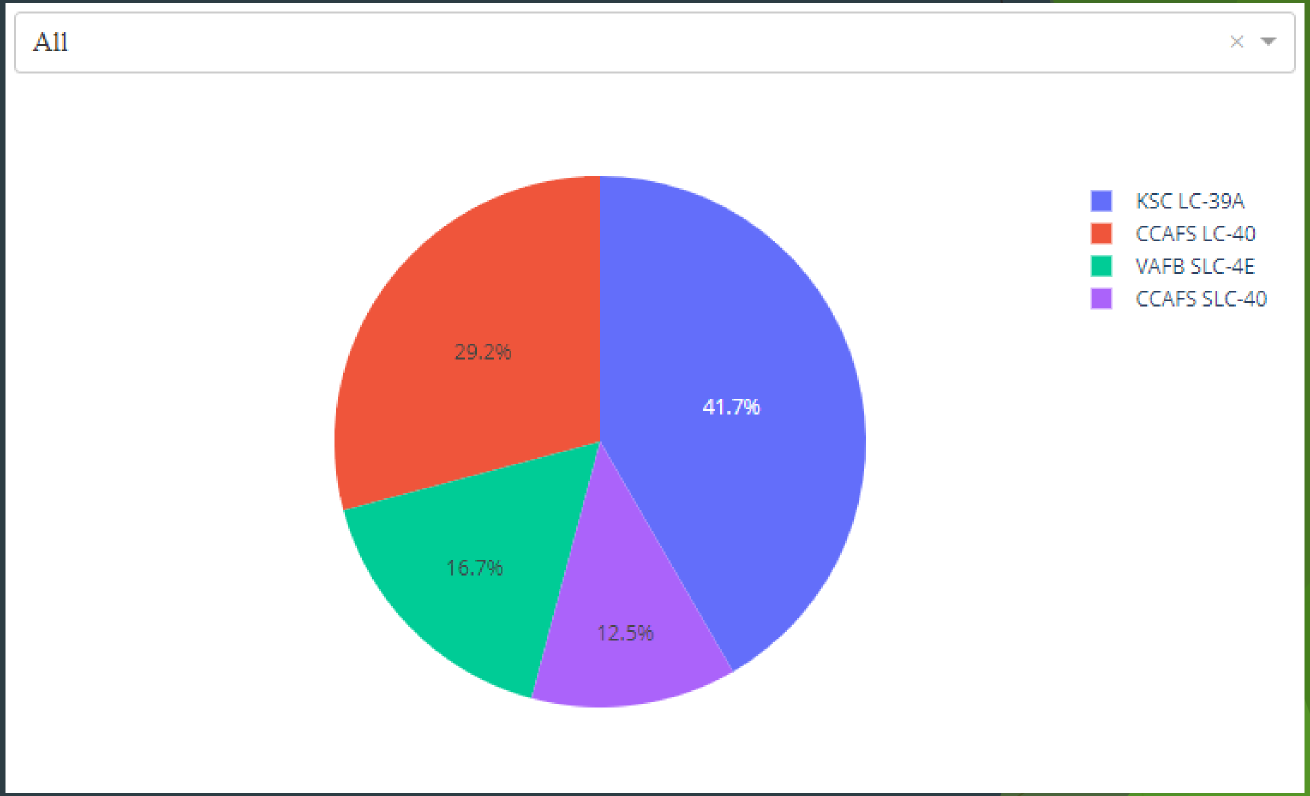


+
−
San Rafael Wilderness
Lat: 34.45448 Long: -119.58893
Vandenberg Space Force Base
Vandenberg State Marine Reserve
3.833
11
VAFB SLC-4E
Lompoc
US 101
Dick Smith Wilderness
Jack and Laura Dangermond Preserve
Point Conception State Marine Reserve
Goleta
Isla Vista
Santa Barbara
84.666
Carpinteria
Santa Barbara Channel
Leaflet | Data by © OpenStreetMap, under ODbL.

Build a Dashboard with Plotly Dash

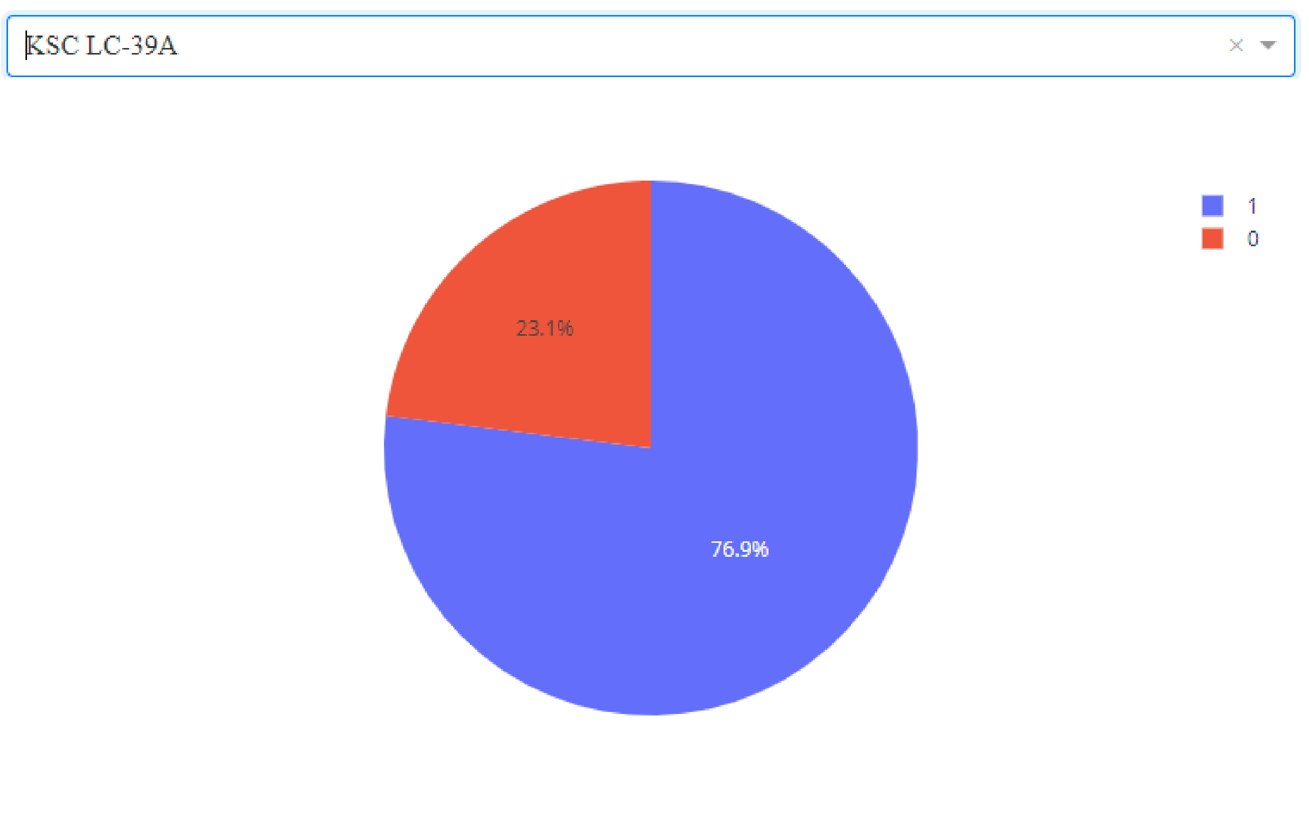
Pie chart with Drop-down

- The elements used in this section have been, in first place, a Dropdown to filter the pie chart and then a pie chart with the package Plotly. Furthermore, we have used an Input and Output conditions to make the dropdown useful.
- Regarding the results, we have obtained that the highest success rate is for KSC LC-39A and the second one is CCAFS LC-40.



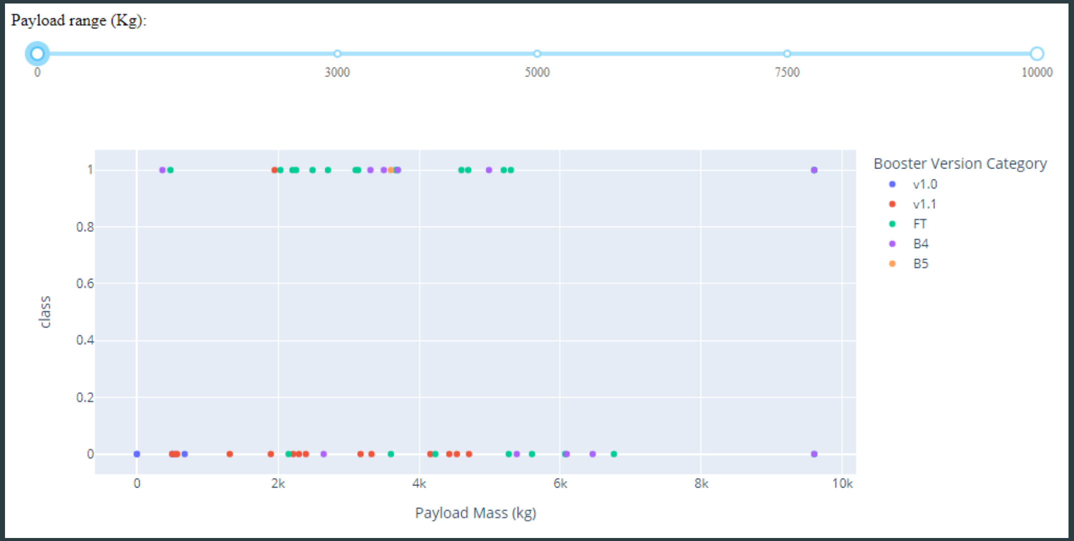
Pie chart for a specific location

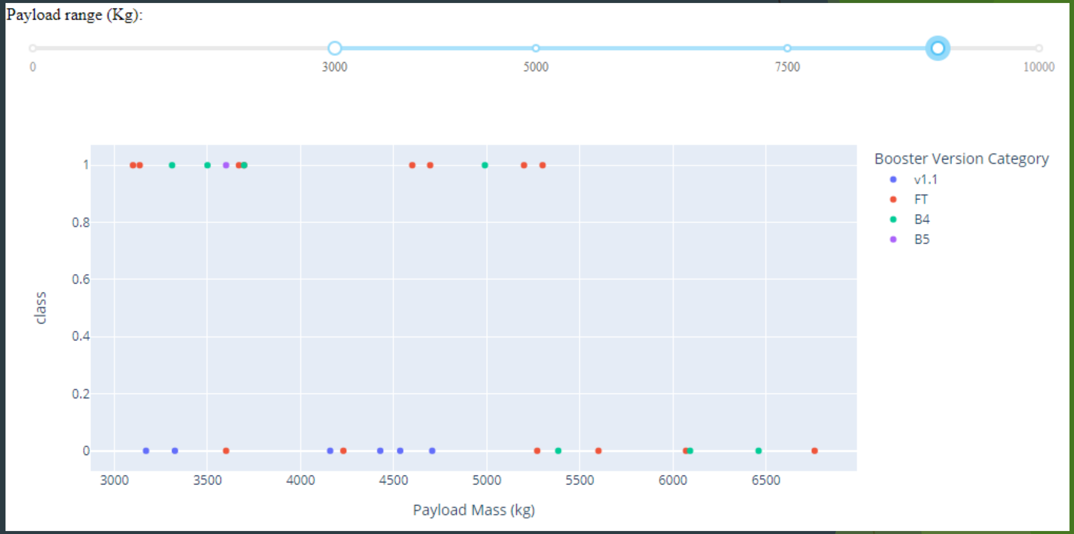
- To get this pie chart we have included an If-else condition to show the labels of the field *Class*. In the case of filter by a specific location the pie chart uses the field *Class* as labels.
- The results are a 76,9% of success and a 23,1% of failure.



KSC LC-39A

Scatter plot with range slider



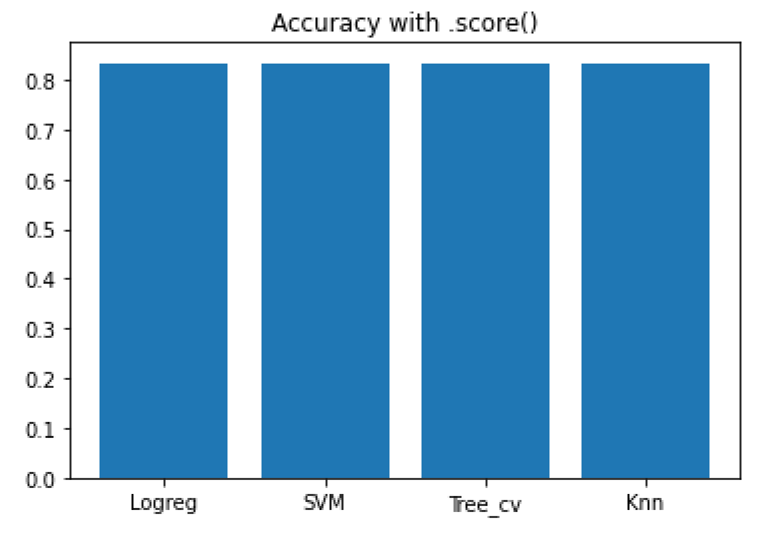


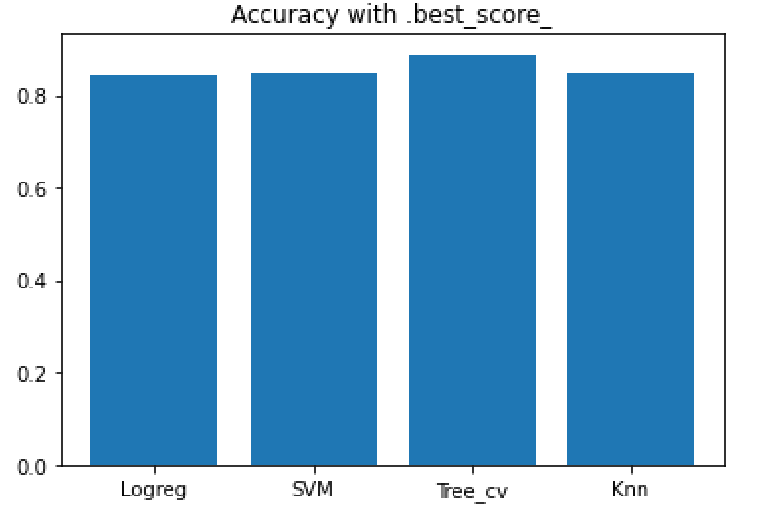
- In order to achieve this plot we have added a range slider with the Payload Mass field. It lets the user to introduce a range for the value of this variable. A part from be able to filter by the location selected in the previous filter.
- The results are colored points which depend of the Booster Version Category. In this case the y axis only can show 1 or 0 points, due to the fact that it refers to the field Class.

Predictive analysis (Classification)

Classification Accuracy

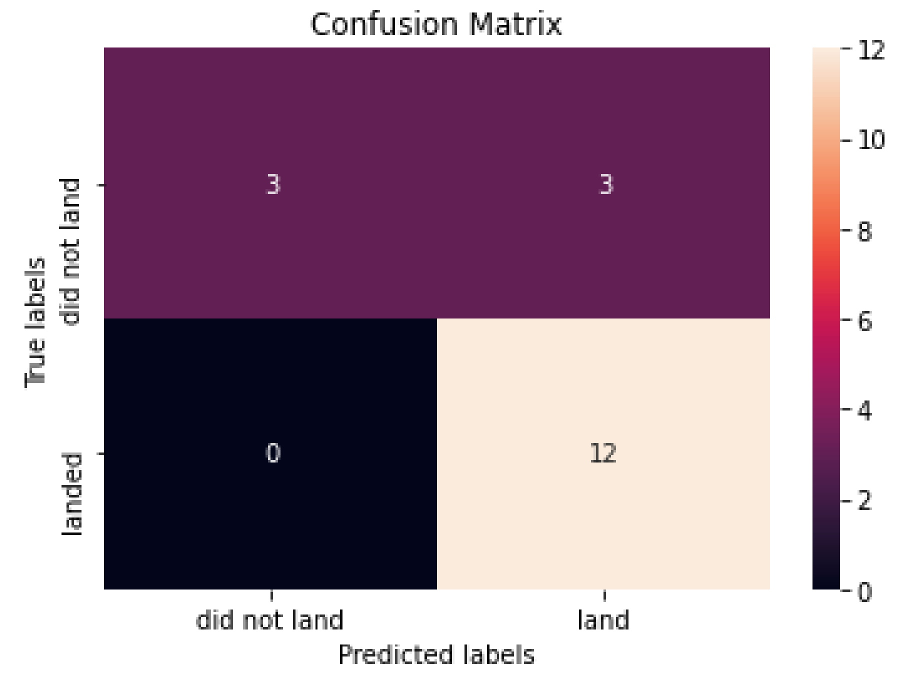
- In order to achieve this plot we have taken into account two different functions. On the one hand, we have applied the .score() function and, on the other hand, the .best_score_ function.
- The results for the first bar plot is the same for all the models. However, in the second plot we can watch that the Decision Tree model has achieved the highest accuracy. Consequently, in the next slide we will consider this model.





Confusion Matrix

- Taking into account that the best model is the Decision Tree, the confusion matrix can be watched in the image.
- The results are the following:
  - From 6 launches that did not land, 3 has been classified correctly and 3 incorrectly.
  - From 12 launches that landed, 12 has been classified correctly and 0 incorrectly.



CONCLUSION

- The most common launch site is CCAFS SLC 40.
- There is a positive trend to land successfully in regard to the flight number.
- Increment in the success rate over time.
- The first successful landing outcome in ground pad occurred on 2015-12-22.
- KSC LC-39A has the highest success rate.
- Florida is most common than California in launching rockets.
- The most accurate machine learning model is the Decision Tree.