Applied Data Science Capstone by IBM

In this project I apply data science skill to the private space launch company called Space X. First, we start our project with collecting data, as much and relevant as possible. I have two resources where I can collect data and after collecting raw data, I will apply data wrangling to improve the quality of data. Then I start exploring the processed data. I will explore some really interesting real-world information.

I will also apply some SQL to gather some insights. For finding further insight from the data, I apply some basic statistical analysis and data visualization using Python and able to see directly how variables might be related to each other. I will also drill down into finer levels of detail by splitting the into group defined by categorical variables or factors in your data. Then I will build, evaluate, and refine predictive models for discovering more exciting insights.

The final task for this capstone project is to create a presentation that will be developed into stories of all your analysis.

Project Scenario and Overview

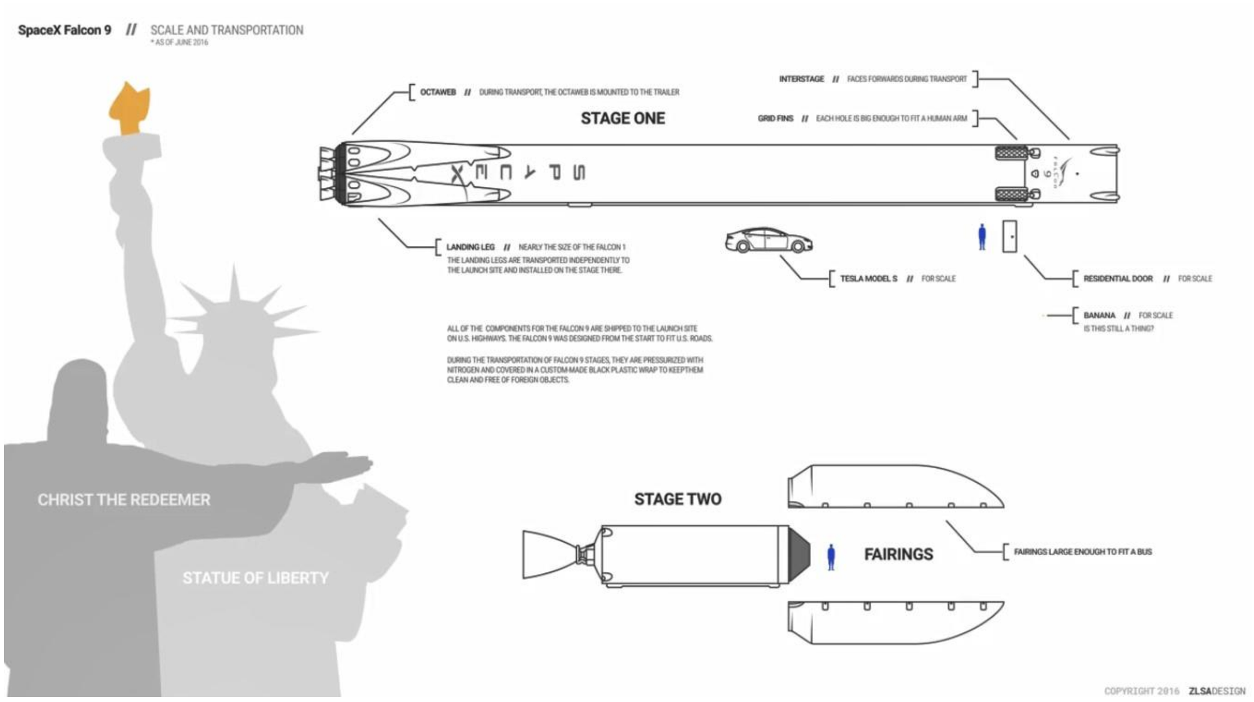
The commercial space age is here, companies are making space travel affordable for everyone. Virgin Galactic is providing suborbital spaceflights. Rocket Lab is a small satellite provider. Blue Origin manufactures sub-orbital and orbital reusable rockets. Perhaps the most successful is SpaceX.

SpaceX's accomplishments include:

Sending spacecraft to the International Space Station. Star link, a satellite internet constellation providing satellite Internet access. Sending manned missions to Space. One reason SpaceX can do this is the rocket launches are relatively inexpensive. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. Spaces X's Falcon 9 launch like regular rockets. To help us understand the scale of the Falcon 9, we are going to use these diagrams from Forest Katsch, at zlsadesign.com.

The payload is enclosed in the fairings. Stage two, or the second stage, helps bring the payload to orbit, but most of the work is done by the first stage. The first stage is shown here. This stage does most of the work and is much larger than the second stage. Here we see the first stage next to a person and several other landmarks. This stage is quite large and expensive. Unlike other rocket providers, SpaceX's Falcon 9 Can recover the first stage. Sometimes the first stage does not land. Sometimes it will crash as shown in this clip. Other times, Space X will sacrifice the first stage due to the mission parameters like payload, orbit, and customer.


SpaceX Falcon 9 // SCALE AND TRANSPORTATION
* AS OF JUNE 2016
OCTAWEB // DURING TRANSPORT, THE OCTAWEB IS MOUNTED TO THE TRAILER
STAGE ONE
INTERSTAGE // FACES FORWARDS DURING TRANSPORT
GRID FINS // EACH HOLE IS BIG ENOUGH TO FIT A HUMAN ARM
LANDING LEG // NEARLY THE SIZE OF THE FALCON 1
THE LANDING LEGS ARE TRANSPORTED INDEPENDENTLY TO THE LAUNCH SITE AND INSTALLED ON THE STAGE THERE.
TESLA MODEL S // FOR SCALE
RESIDENTIAL DOOR // FOR SCALE
BANANA // FOR SCALE
IS THIS STILL A THING?
ALL OF THE COMPONENTS FOR THE FALCON 9 ARE SHIPPED TO THE LAUNCH SITE ON U.S. HIGHWAYS. THE FALCON 9 WAS DESIGNED FROM THE START TO FIT U.S. ROADS.
DURING THE TRANSPORTATION OF FALCON 9 STAGES, THEY ARE PRESSURIZED WITH NITROGEN AND COVERED IN A CUSTOM-MADE BLACK PLASTIC WRAP TO KEEPTHEM CLEAN AND FREE OF FOREIGN OBJECTS.
CHRIST THE REDEEMER
STATUE OF LIBERTY
STAGE TWO
FAIRINGS
FAIRINGS LARGE ENOUGH TO FIT A BUS
COPYRIGHT 2016 ZLSADESIGN

In this capstone, I will take the role of a data scientist working for a new rocket company. Space Y that would like to compete with SpaceX founded by Billionaire industrialist Allon Musk. My job is to determine the price of each launch. I will do this by gathering information about Space X and creating dashboards. I will also determine if SpaceX will reuse the first stage. Instead of using rocket science to determine if the first stage will land successfully, I will train a machine learning model and use public information to predict if SpaceX will reuse the first stage.

Data Collection:

As I said I have two resources where I can get the data one is SpaceX rest API and second is Wikipedia website. Let's see which source will give me a better response.

SpaceX Rest API:

I will be working with SpaceX launch data that is gathered from an API, specifically the SpaceX REST API. This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome. My goal is to use this data to predict whether SpaceX will attempt to land a rocket or not. The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/.

I have the different end points, for example: /capsules and /cores I will be working with the endpoint api.spacexdata.com/v4/launches/past. Let's see how the API works. I will use this URL to target a specific endpoint of the API to get past launch data. I will perform a get request using the requests library to obtain the launch data, which we will use to get the data from the API. This result can be viewed by calling the `.json()` method. Our response will be in the form of a JSON, specifically a list of JSON objects. Since I am using an API, I will notice that when I get a response it is in the form of a JSON. Specifically, I have a list of JSON objects which each represent a launch. To convert this JSON to a DataFrame, we can use the json_normalize function. This function will allow us to “normalize” the structured JSON data into a flat table. This is what your JSON will look like in a table form.

SpaceX Wikipedia:

Another popular data source for obtaining Falcon 9 Launch data is web scraping related Wiki pages. I will be using the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records. Then I need to parse the data from those tables and convert them into a Pandas data frame for further visualization and analysis. We want to transform this raw data into a clean dataset which provides meaningful data on the situation we are trying to address: Wrangling Data using an API, Sampling Data, and Dealing with Nulls.

In some of the columns, like rocket, we have an identification number, not actual data. This means I will need to use the API again targeting another endpoint to gather specific data for each ID number. These functions are already created for you, and will use the following:

- Booster,
- Launchpad,
- payload, and
- core.

The data will be stored in lists and will be used to create our dataset. Another issue I have is that the launch data we have includes data for the Falcon 1 booster whereas we only want falcon 9. I will need to figure out how to filter/sample the data to remove Falcon 1 launches.

Finally, not all gathered data is perfect. We may end up with data that contains NULL values. We must sometimes deal with these null values in order to make the dataset viable for analysis. I will deal with the NULL values inside the PayloadMass. I must figure out a way to calculate the mean of the PayloadMass data and then replace the null values in PayloadMass with the mean. I will leave the column LandingPad with NULL values, as it is represented when a landing pad is not used. This will be dealt with using one hot encoding later on.

Data Wrangling:

After getting data, I will perform some Data Wrangling with that data which I collect from the help of SpaceX Rest Api to find some patterns in the data and determine what would be the label for training supervised models.

Let’s review some of the attributes: Flight Number, Date, Booster version, Payload mass Orbit, Launch Site, Outcome: this is the status of the first stage Flights, Grid Fins: these help with landing Reused, Legs: used in landing Landing pad, Block, Reused count, Serial, Longitude and latitude of launch

Let’s take a look at some of these attributes: The column “LaunchSite” contains the different launch sites, including: Vandenberg AFB Space Launch Kennedy Space Center CCAFS SLC 40

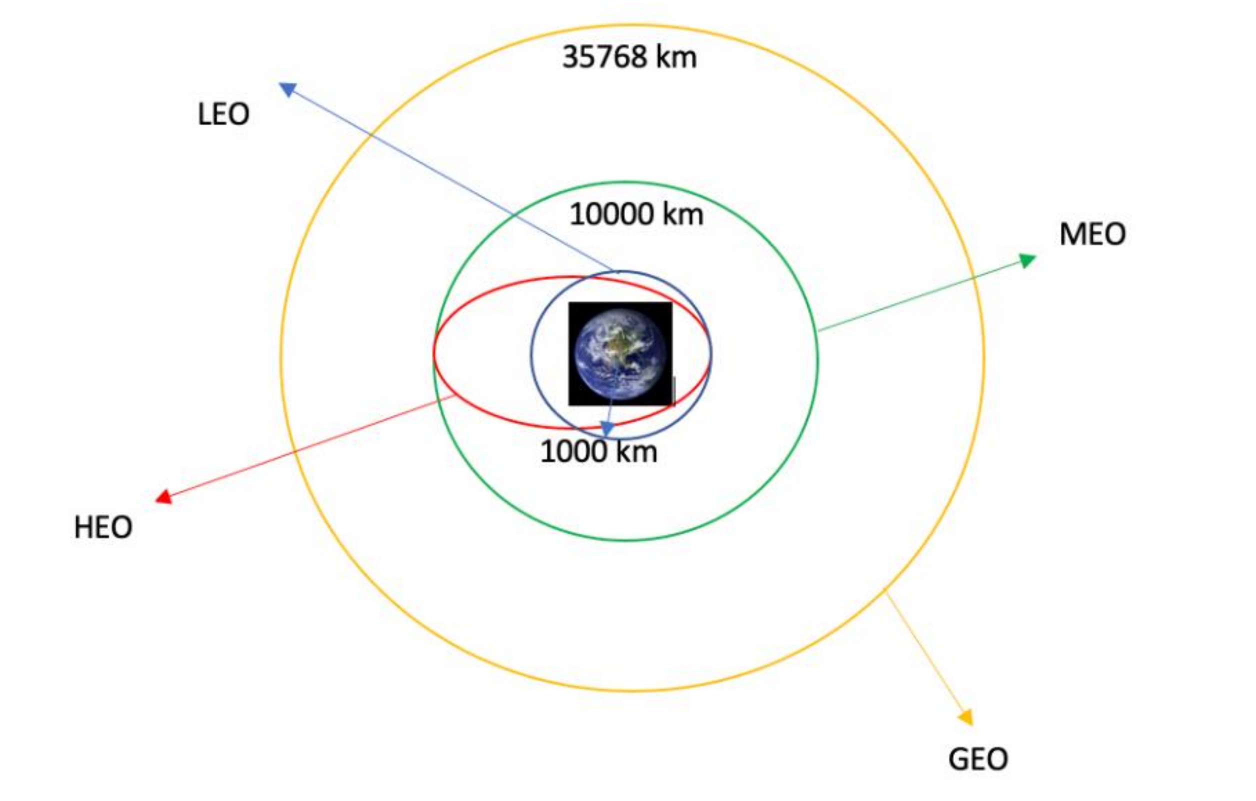
The column orbits are the different orbits of the payload. For Example: LEO: Low Earth orbit (LEO)is an Earth-centered orbit with an altitude of 2,000 km, GTO A geosynchronous orbit is a high Earth orbit that allows satellites to match Earth's rotation. It is located at 22,236 miles (35,786 kilometers) above Earth's equator. The column Outcome indicates if the first stage successfully landed.

We would like landing outcomes to be converted to Classes y. y. (either 0 or 1). 0 is a bad outcome, that is, the booster did not land. 1 is a good outcome, that is, the booster did land. The variable Y will represent the classification variable that represents the outcome of each launch.

Data Analysis & Imputation:

I identify and calculate the percentage of the missing values in each attribute and find that only Landing Pad have 40% missing values. After that I check is column is numerical or categorical and the results are positive.

Now First I collect the number of launches on each site. The data contains several Space X launch facilities: Cape Canaveral Space Launch Complex 40 VAFB SLC 4e, Vandenberg Air Force Base Space Launch Complex 4E (SLC-4E), Kennedy Space Center Launch Complex 39A KSC LC 39A. The location of each Launch is placed in the column Launch-Site.



Data Imputation:

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.

So, I create a set of outcomes where the second stage did not land successfully and converted it into 0 which represent bad outcome and other convert it into 1 which represent good outcome.

Exploratory Data Analysis (EDA):

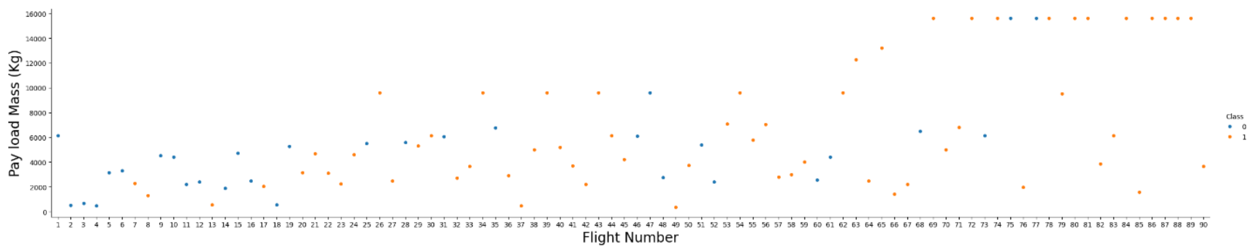
In this section, I will see if the data can be used to automatically determine if the Falcon 9's first stage will land. Some attributes can be used to determine if the first stage can be reused. We can then use these features with machine learning to automatically predict if the first stage can land successfully.

I can observe that the success rate since 2013 has improved. We can incorporate this as a feature via launch Number. We see that different launch sites have different success rates. As a result, they can be used to help determine if the first stage will land successfully. CCAFS LC-40 has a success rate of 60%, while KSC LC-39A and VAFB SLC 4E have a success rate of around 77%.

Combining attributes also gives us more information. If we overlay the result of the landing outcomes as a color, we see that CCAFS LC-40, has a success rate of 60%, but if the mass is above 10,000 kg the success rate is 100%. Therefore, we will combine multiple features. The categorical variables will be converted using one hot encoding, preparing the data for a machine learning model that will predict if the first stage will successfully land.

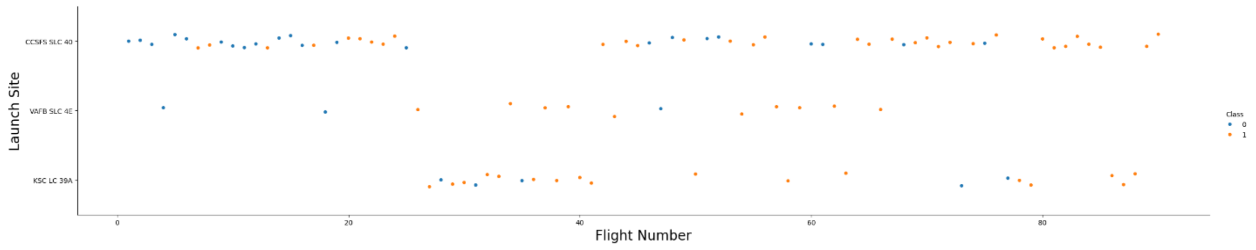
First, let's try to see how the Flight-Number (indicating the continuous launch attempts) and Payload variables would affect the launch outcome.

We can plot out the FlightNumber vs. PayloadMass and overlay the outcome of the launch. We see the as the flight number increases, the first stage is more likely to land successfully. The payload mass is also important, it seems the more massive the payload, the less likely the first stage will return.

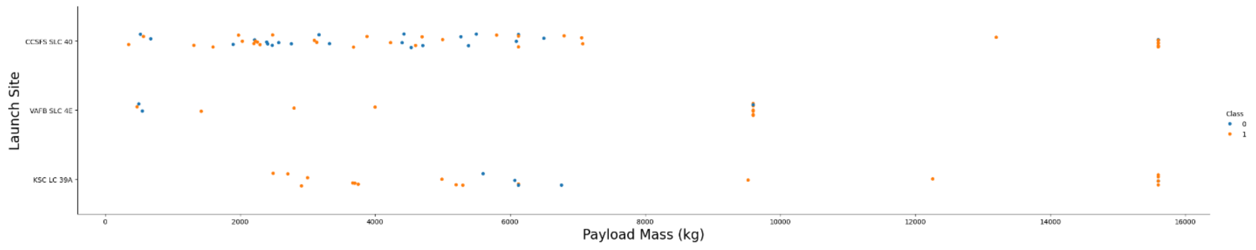


We see that different launch site have different success rates. CCAFS LC-40, has a success rate of 60%, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.

Now plot the FlightNumber vs. LaunchSite.

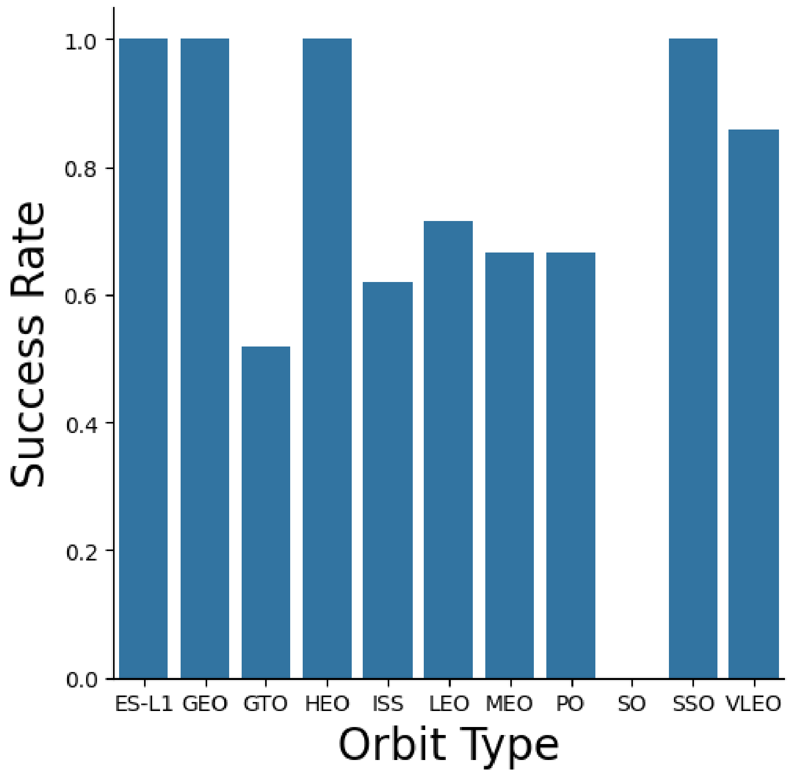


I also want the relationship between launch sites and their payload mass.

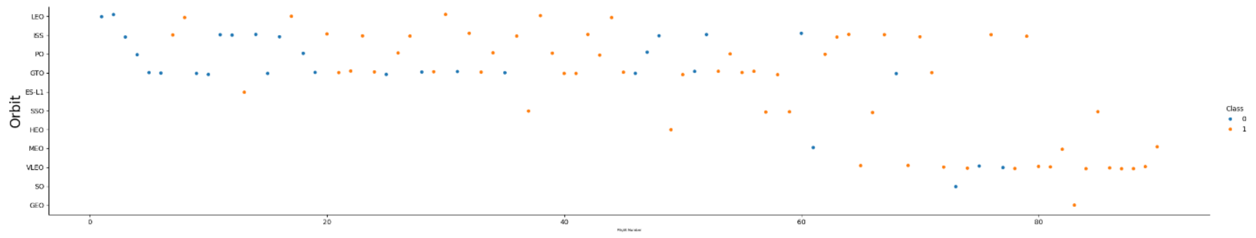


I observe in Payload Vs. Launch Site scatter point chart that the VAFB-SLC launchsite has no rockets launched for heavypayload mass (greater then 10000).

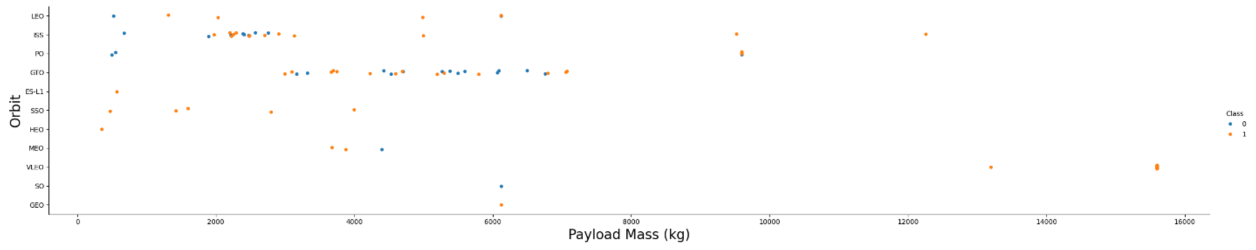
Now check the relationship between success rate and orbit type.



For each orbit, I want to see if there is any relationship between FlightNumber and Orbit type.



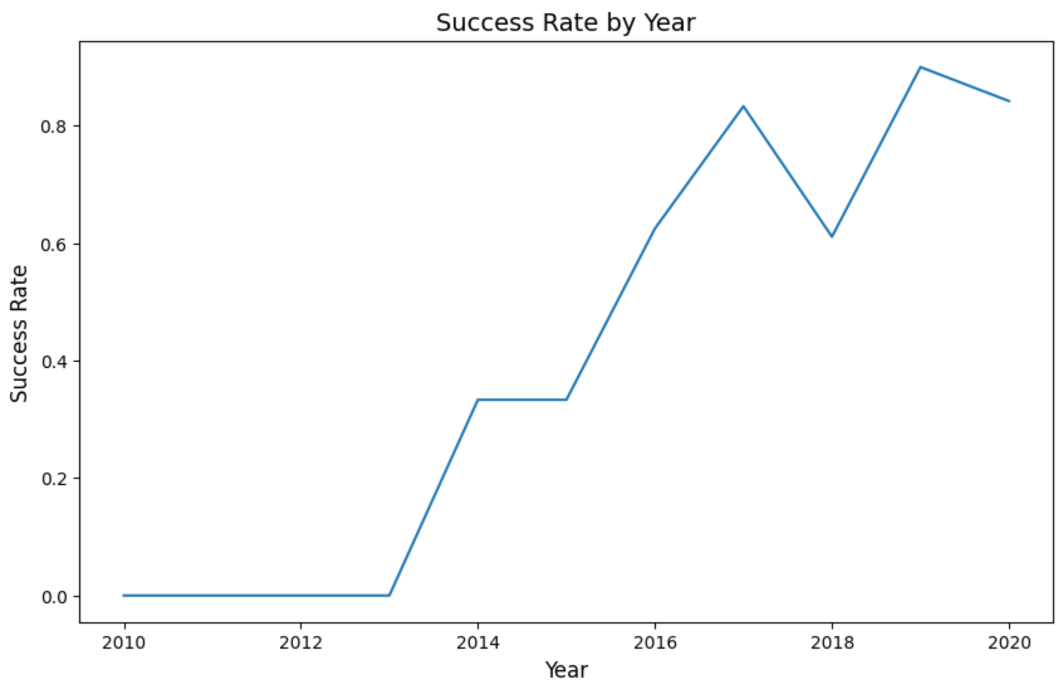
I see that in the LEO orbit the Success appears related to the number of flights, on the other hand, there seems to be no relationship between flight number when in GTO orbit.



With heavy payloads the successful landing or positive landing rate are more for Polar, LEO, ISS.

However, the GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both here.

Now visualize the launch success yearly trend



we can observe that the success rate since 2013 kept increasing till 2020.

Feature Engineering:

By now, I obtain some preliminary insights about how each important variable would affect the success rate, I will select the features that will be used in success prediction in the future module.

Now I'm using get_dummies and features function of the dataframe to apply OneHotEncoder to the categorical columns like Orbit, LaunchSite, LandingPad, and Serial. Then I am assigning the value to the variable features_one_hot and also convert into type float64.

Machine Learning Prediction:

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. In this section, I will create a machine learning pipeline to predict if the first stage will land given the data from the preceding notebooks.

Perform EDA and determine Training Labels

- create a column for the class
- standardize the data
- split into training data and test data

Find best Hyperparameter for SVM, Classification Trees and Logistic Regression

- Find the method performs best using test data

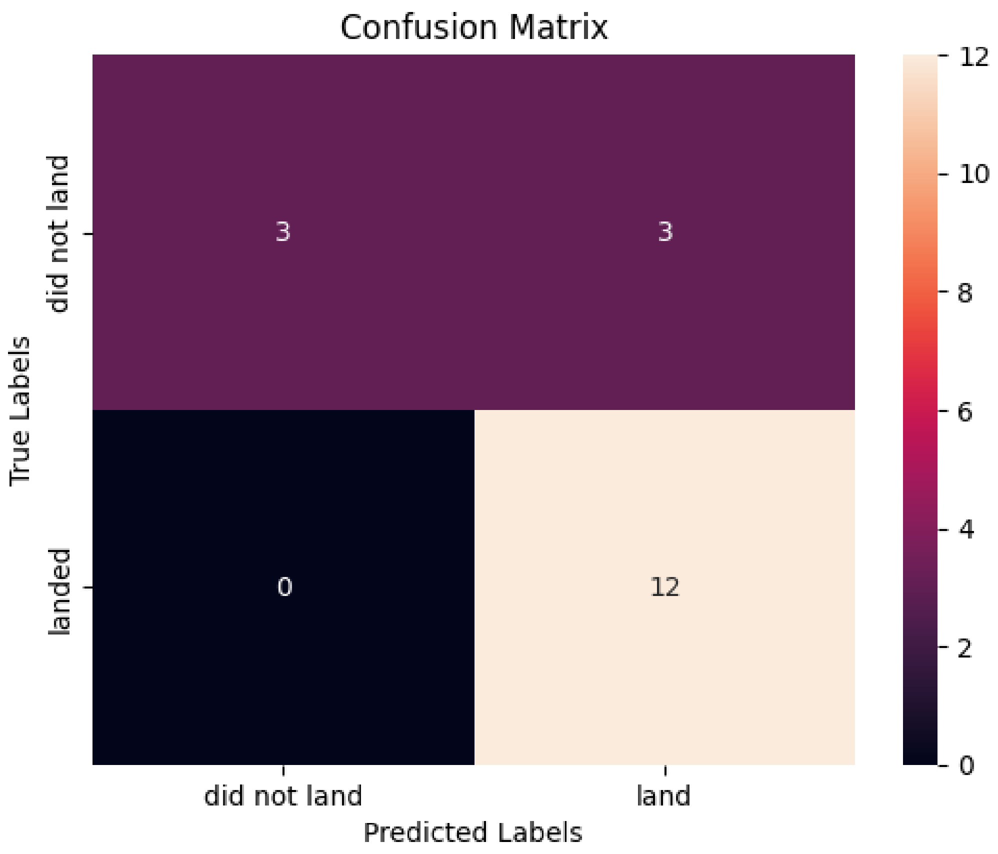
I split the data into training and testing data using the function train_test_split. The training data is divided into validation data, a second set used for training data then the models are trained and hyperparameters are selected using the function GridSearchCV.

Logistic Regression:

I create a logistic regression object then create a GridSearchCV object logreg_cv with cv=10. Fit the object to find the best parameters from the dictionary parameters and then output the GridSearchCV object for logistic regression. After that I display the best parameters using the data attribute best_params_ and the accuracy on the validation data using the data attribute best_score_.

- Tuned hyperparameters: (best parameters) {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
- Accuracy: 0.8464285714285713

Now let's look at the confusion matrix:



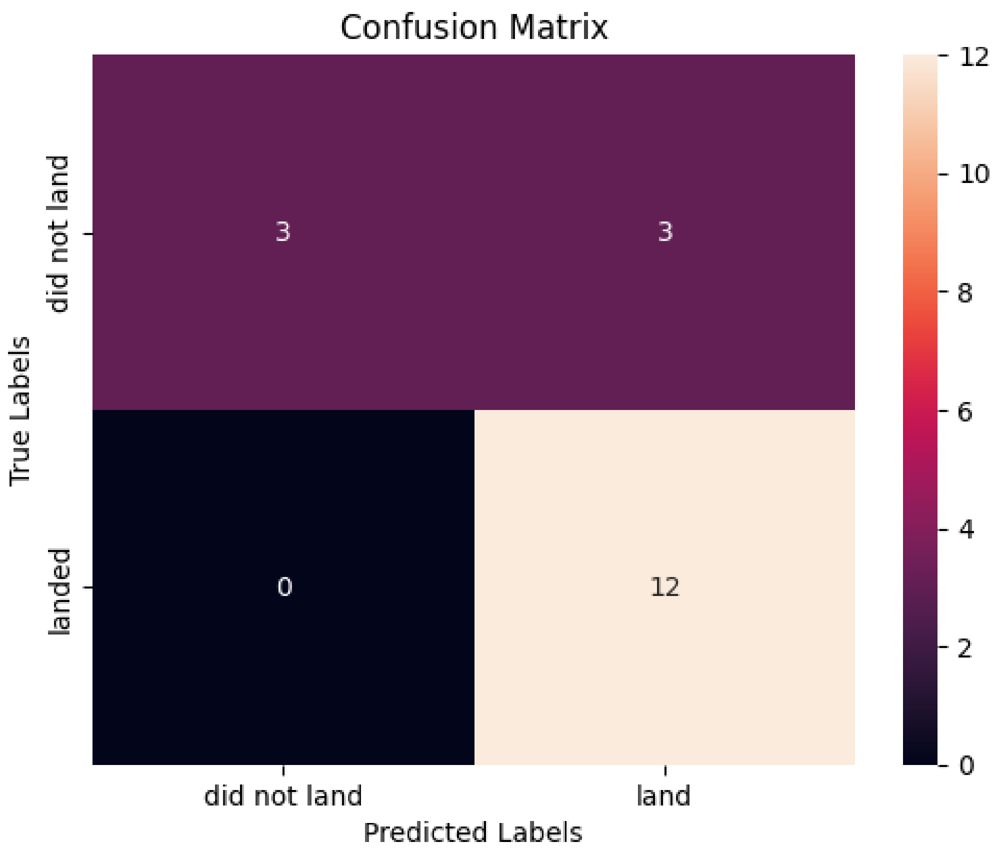
Examining the confusion matrix, we see that logistic Regression can distinguish between the different classes. We see that the major problem is false positive.

Support Vector Machine (SVM):

I create a support vector machine object then create a GridSearchCV object svm_cv with cv=10. Fit the object to find the best parameters from the dictionary parameters and the results are given below:

- Tuned hyperparameter: (best parameters) {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
- Accuracy: 0.8482142857142856

Let's look at the confusion matrix of SVM:

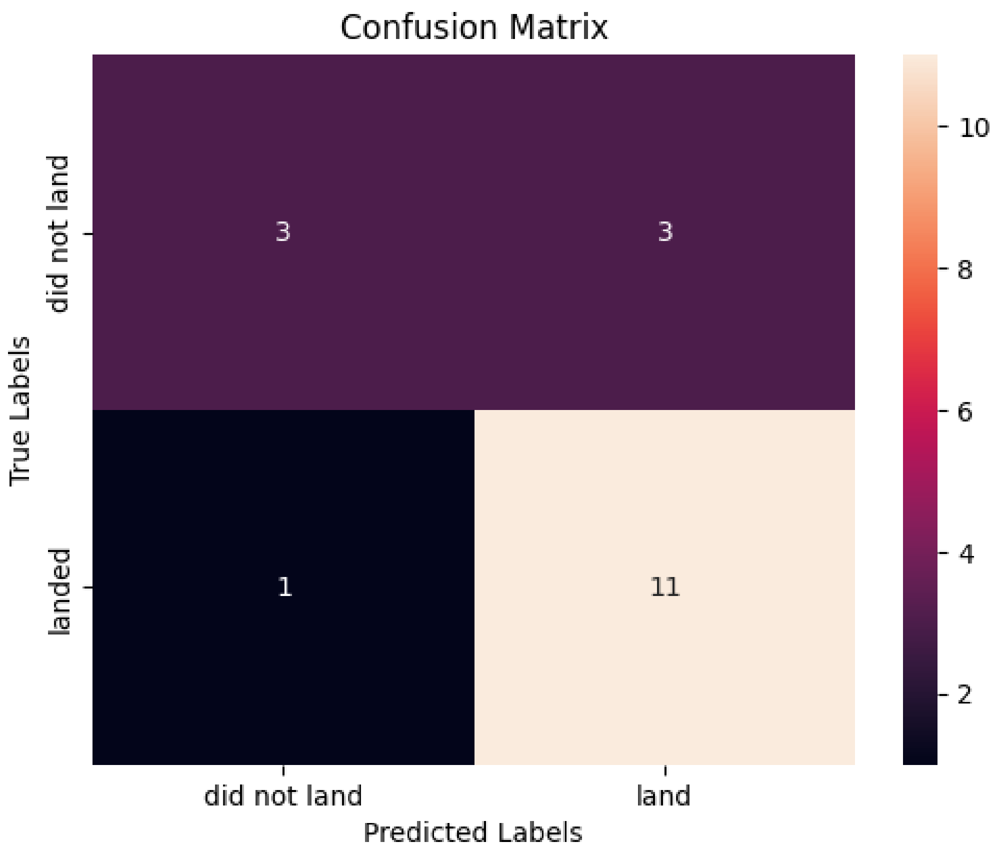


Decision Tree Classifier:

I create a decision tree classifier object then create a GridSearchCV object tree_cv with cv=10. Fit the object to find the best parameters from dictionary parameters and the results are:

- tuned hpyerparameters :(best parameters) {'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'best'}
- accuracy: 0.8892857142857142

Let’s see the confusion matrix:

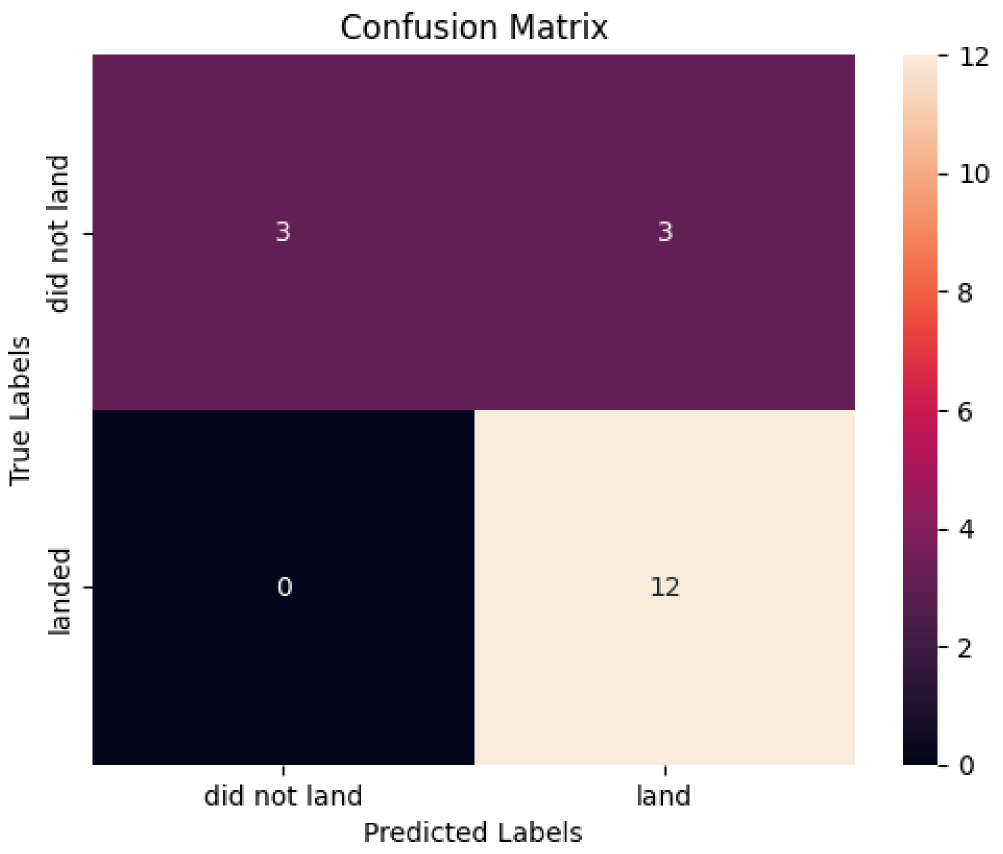


K-Nearest Neighbors:

Create a k nearest neighbor’s object then create a GridSearchCV object knn_cv with cv=10. Fit the object to find the best parameters from the dictionary parameters and here are the results:

- tuned hpyerparameters :(best parameters) {'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
- accuracy: 0.8482142857142858

Let’s look at the confusion matrix:



Finding Best Model:

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.833333	0.845070	0.774648	0.819444
F1_Score	0.909091	0.916031	0.873016	0.900763
Accuracy	0.866667	0.877778	0.822222	0.855556

Conclusion:

- Based on the scores of the Test Set, I cannot confirm which method performs best.
- Same Test Set scores may be due to the small test sample size (18 samples). Therefore, we tested all methods based on the whole Dataset.
- The scores of the whole Dataset confirm that the best model is the Support Vector Machine. This model has not only higher scores, but also the highest accuracy.