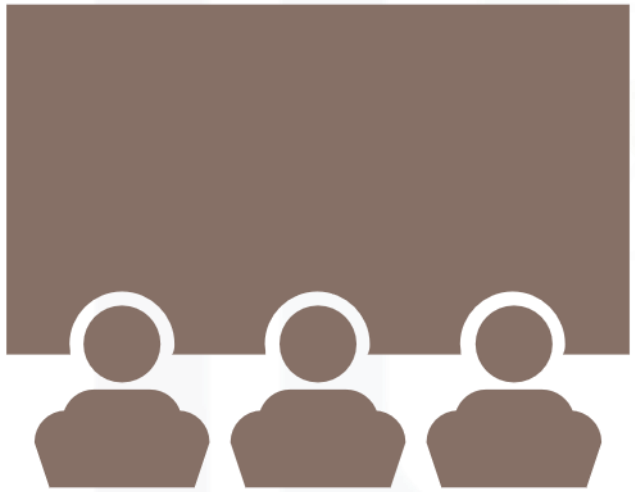
# ASSESSING THE LANDING SUCCESS OF THE FALCON9 LAUNCH VEHICLE


Work performed by:
Taras Mikhailovich Tiunov
05/03/2024

English version

# Report structure:



- Summary
- Introduction
- Methodology
- Results
- Discussion
- Conclusion
- Application

# Summary:

- Methodology:
  - Collecting data using the API
  - Collecting missing data using web scraping
  - Data Preprocessing
  - Analysis with SQL
  - Analysis using data visualization
  - Predicting results using machine learning
- Results
  - Data analysis results
  - Charts and graphs
  - Results of predictive analysis

# Introduction

Human space expansion is a big, complex and expensive problem. The main consumable item is launch vehicles, which are usually used once and are not suitable for reuse. SpaceX has learned to solve this problem and has been able to reduce the cost of suborbital and orbital flights by more than three times. The Falcon 9 launch vehicle, suitable for repeated use, became the “magic pill”. The cost of a space flight now depends not so much on the high cost of the launch vehicle (it has not become cheaper, but quite the opposite), but on how many times it can be used. In my work, I will estimate the probability of successfully landing the Falcon 9 rocket, which will help estimate the costs of space flight.

# Introduction

The goal of the project is to create a machine learning pipeline to predict whether the Falcon 9's first stage will be saved for next use.

Questions we are looking for answers to:

- What factors determine the success of a launch vehicle landing?
- What conditions must be created for a successful landing of a launch vehicle?

# Methodology

- Data was obtained from open sources:
  - Request to SpaceX data REST API
  - Web scraping of Wikipedia
- Data preprocessing:
  - One-hot encoding is a way to move from categorical variables to logical or numeric ones
- Exploratory data analysis (EDA) using SQL and visualization tools
- Predictive analysis using machine learning algorithms

# Data collection

- Some of the data for analysis was obtained from the SpaceX API.
- Click here for the link to the notebook for this part of the work!

[5]:
```
# Takes the dataset and uses the cores column to call the API and append the data to the lists
def getCoreData(data):
    for core in data['cores']:
            if core['core'] != None:
                response = requests.get("https://api.spacexdata.com/v4/cores/"+core['core']).json()
                Block.append(response['block'])
                ReusedCount.append(response['reuse_count'])
                Serial.append(response['serial'])
            else:
                Block.append(None)
                ReusedCount.append(None)
                Serial.append(None)
            Outcome.append(str(core['landing_success'])+' '+str(core['landing_type']))
            Flights.append(core['flight'])
            GridFins.append(core['gridfins'])
            Reused.append(core['reused'])
            Legs.append(core['legs'])
            LandingPad.append(core['landpad'])
```

Now let's start requesting rocket launch data from SpaceX API with the following URL:

[6]:
```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

Check the content of the response

[7]:
```
print(response.content)
```

# Data collection

- Part of the data for analysis was obtained by web scraping the English-language Wikipedia.
- Click here for the link to the notebook for this part of the work!

- The result of data collection was a Pandas dataframe with information about all launches of the Falcon 9 rocket

To keep the lab tasks consistent, you will be asked to scrape the data from a snapshot of the `List of Falcon 9 and Falcon Heavy launches` Wikipage updated on `9th June 2021`

[4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=
```

Next, request the HTML page from the above URL and get a `response` object

### TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

[5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

[6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
bs_0=BeautifulSoup(response.text,"lxml")
```

# Preliminary processing

- Before moving directly to analysis, you need to make the data more convenient.
  - If there is not enough data, then, if possible, supplement it from another source.
  - If replenishment is not possible, replace the missing values in the observations with the average or most probable values.
  - If replacement is not possible, delete the observations.
- The next step is to replace the categorical variables with "pseudo-numeric" ones (dummies).
- Our task is to evaluate the success of the launch vehicle landing. Therefore, you need to introduce a variable that will take the value 1 if the landing is successful, and 0 in other cases (when the landing is unsuccessful or the result is unknown).
- Link to notebook with this task (click me!)

**TASK 4: Create a landing outcome label from Outcome column**

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
print(landing_class)
```

```
[0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1]
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

# Exploratory analysis

Exploratory analysis allows you to quickly find out answers to questions such as:

- Were launches from any station more successful than from all others?
- What is the total and average payload for all launches launched by Falcon 9 rockets into near-Earth space?
- When did rocket launches with this type of accelerator begin?
- And the like.

For exploratory analysis, we used Jupyter notebook, MySQL, Matplotlib, Seaborn, and interactive charting tools such as Folium to create terrain maps and Plotly Dash to create custom graphs.

# Exploratory analysis – relational database

During exploratory analysis, we converted a dataframe with data into a relational database (more precisely, into a table for such a database). Next, this table was examined with MySQL queries. Link to lab notebook on this subject (click me!)

```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5
```

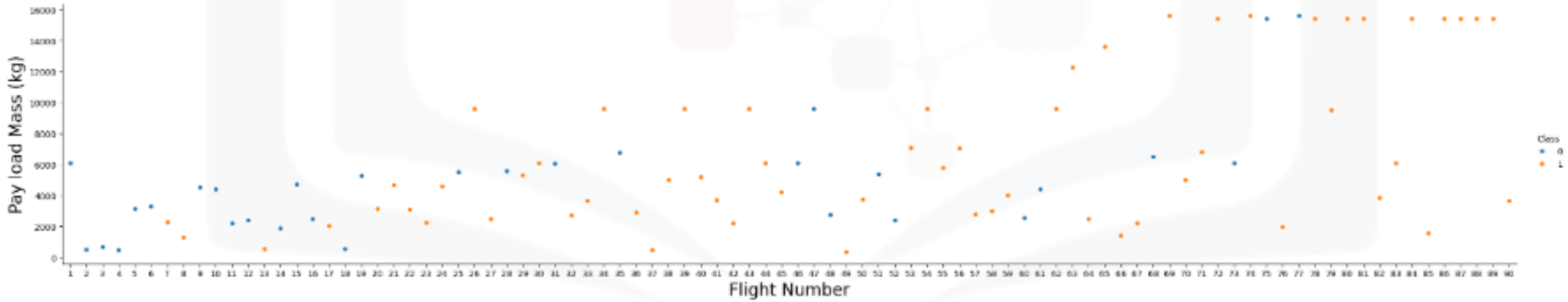
```
 * sqlite:///my_data1.db
Done.
```

| Date | Time (UTC) | Booster_Version | Launch_Site | Payload | PAYLOAD_MASS__KG_ | Orbit | Customer | Mission_Outcome |
|---|---|---|---|---|---|---|---|---|
| 2010-06-04 | 18:45:00 | F9 v1.0 B0003 | CCAFS LC-40 | Dragon Spacecraft Qualification Unit | 0 | LEO | SpaceX | Success |
| 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 |
| 2012-05-22 | 7:44:00 | F9 v1.0 B0005 | CCAFS LC-40 | Dragon demo flight C2 | 525 | LEO (ISS) | NASA (COTS) | Success |
| 2012-10-08 | 0:35:00 | F9 v1.0 B0006 | CCAFS LC-40 | SpaceX CRS-1 | 500 | LEO (ISS) | NASA (CRS) | Success |
| 2013-03-01 | 15:10:00 | F9 v1.0 B0007 | CCAFS LC-40 | SpaceX CRS-2 | 677 | LEO | NASA | Success |

# Explorator analysis - diagrams and graphs

- Using Matplotlib and Seaborn, we have created several plots that can help answer exploratory analysis questions.
- Link to the notebook with this part of the work (click me!)

```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```

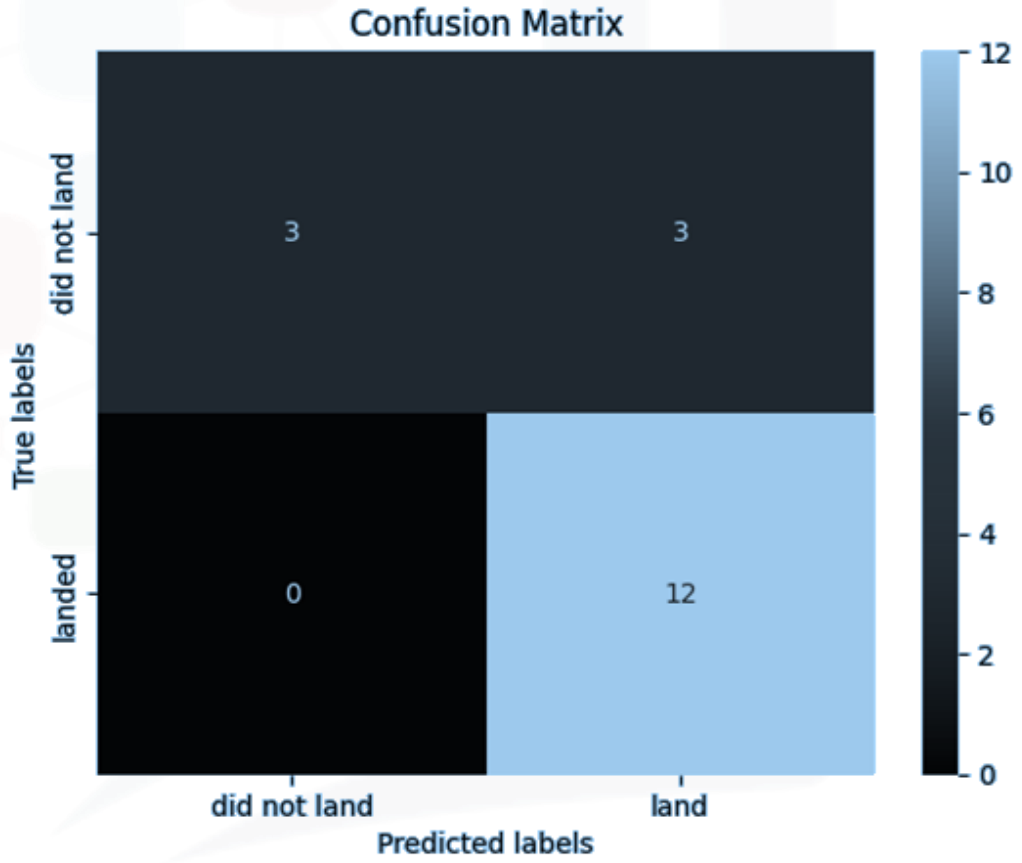


# Machine learning

Machine learning allows you to predict the future behavior of the system based on available data. In our case, it is to predict the successful landing of a launch vehicle, knowing where it starts from and with what accelerator, what payload it carries, and to what orbit it flies.

In this work, we used support vector machine, logistic regression, nearest k-neighbor and decision tree. We compared these algorithms and determined the best one.

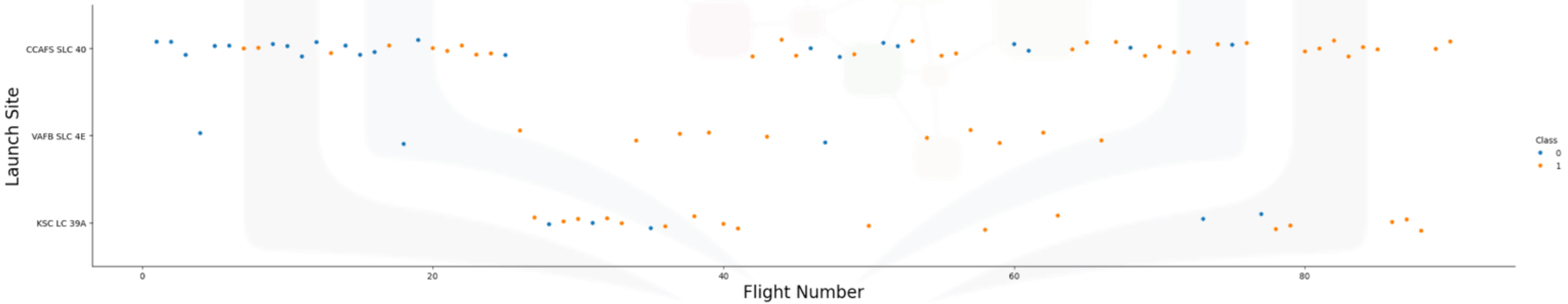
Link to lab notebook on this subject (click me!)



# Results

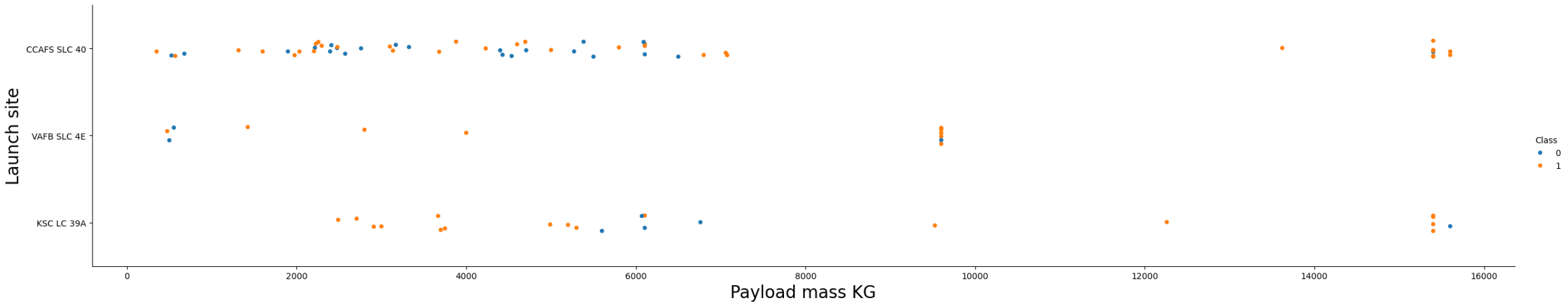
## Flights from each launch site

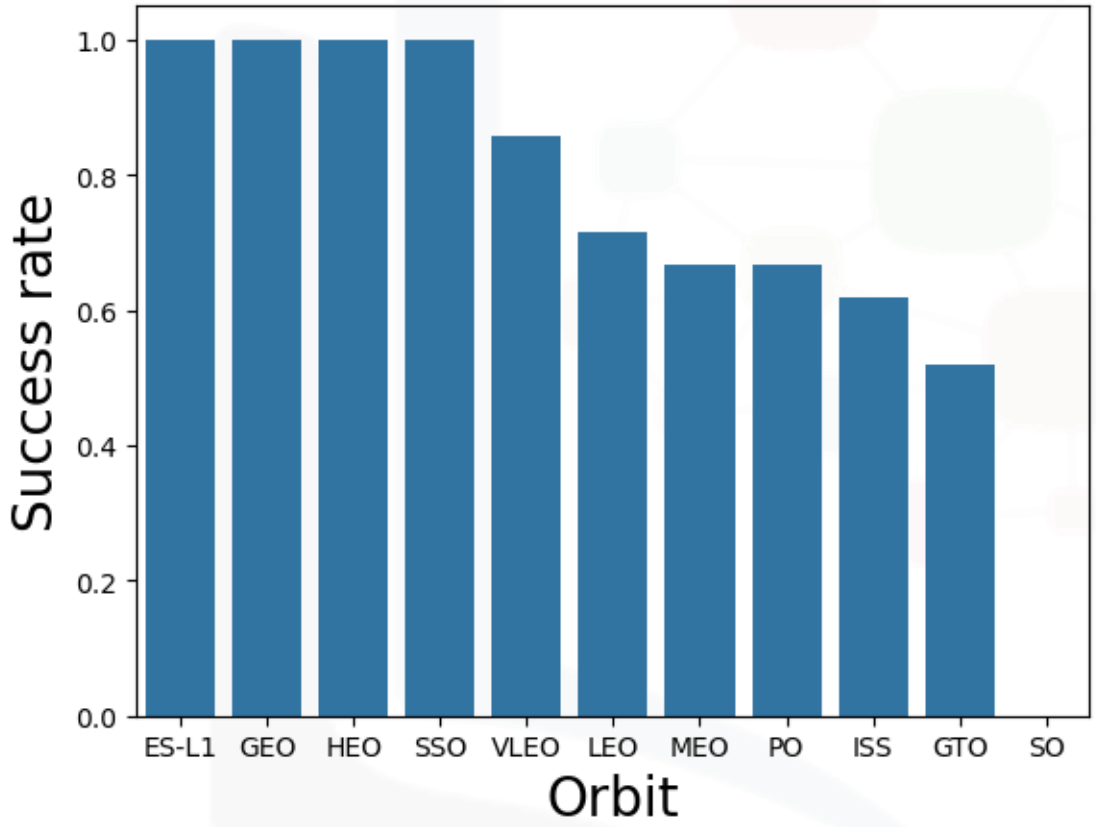
From this graph we can see that the more launches were made from each launch site, the more successful the launches were.



## Payload mass vs. launch site

1) The greater the load, the more successful the flights.
2) No rockets with a load of more than 10 tons were launched from the VAFB SLC 4E launch site. Perhaps it is not suitable for such launches.





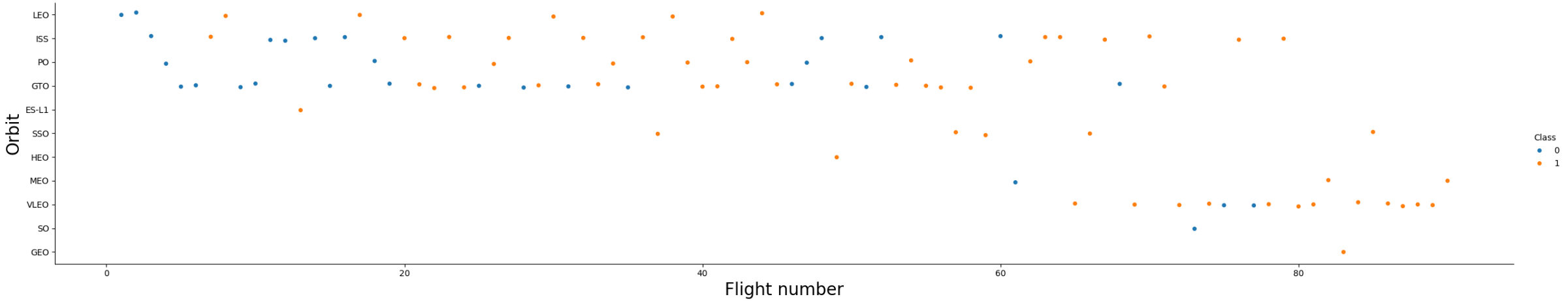

## Orbit type vs. Success rate

The diagram shows which orbits were the most successful launches. The appendix contains a table describing all orbits (in Russian).

## Flights to the different orbit types

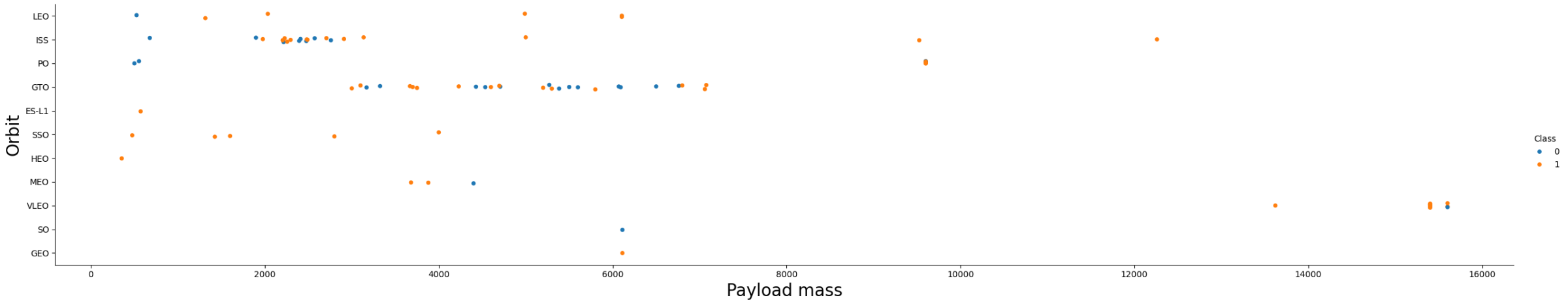
SpaceX mastered orbits one by one, following the previous one, and learned to successfully land rockets from each orbit. This can be seen from the fact that the dots moved from the upper left to the lower right, and the color of the dots steadily changed from blue (landing failed) to orange (landing successful). Of course, the chance of landing the rocket did not become 100% (we see that between the 60th and 80th launch, after four consecutive successful launches to VLEO orbit, we see two blue dots). Let us remember that landing a launch vehicle is something that no one has done before SpaceX (or has not done successfully, Buran and Challenger do not count, these are devices of a different class), and this is a very difficult task, depending on many conditions (including weather and human factor). I would call such a success incredible, and it speaks of the vast experience accumulated by the company and the highly qualified specialists working there.

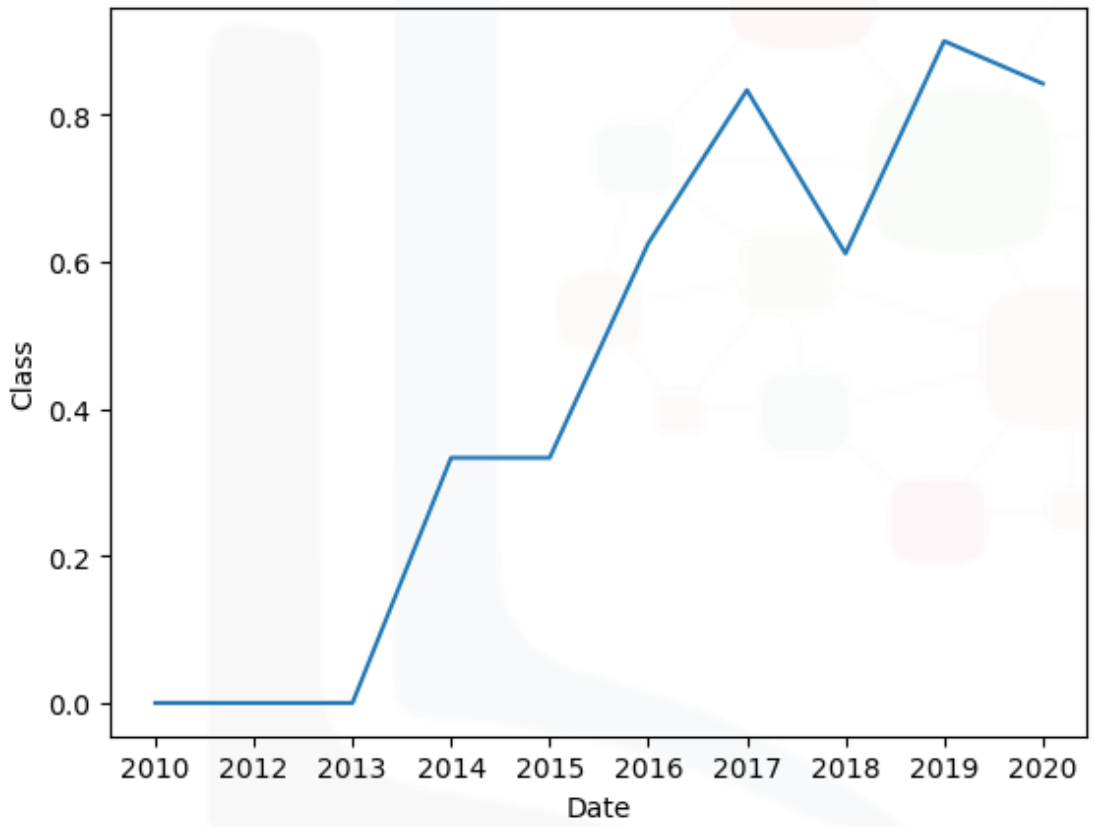
There is no direct connection between the flight number and the type of orbit.



## Payload mass vs. Orbit type

We can see that with heavy payload, successful landing is more observed in PO, LEO and ISS orbits






## Yearly trend of success rate

From the graph we see that the probability of success increased in 2013 and increases with varying success until 2020

## List of all spaceports from which the Falcon 9 heavy launch vehicle launched

Accessing a MySQL database with the Distinct keyword

```
%%sql
select distinct(Launch_Site) from SPACEXTABLE
```

```
 * sqlite:///my_data1.db
Done.
```

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

Display 5 records where launch sites begin with the string 'CCA'

```
%sql select * from SPACEXTABLE where Launch_Site like 'CCA%' limit 5
```

```
 * sqlite:///my_data1.db
Done.
```

| 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 |

## Trainee! Show me the first five launches from the cosmodrome... I don't remember what it is called, the name starts with CCA

Having SQL at hand, you can search for information using such a query.

## Total payload mass launched by NASA (CRS)

Using SQL built-in functions

Display the total payload mass carried by boosters launched by NASA (CRS)

```
%sql select sum(PAYLOAD_MASS__KG_) as 'Total payload mass' from SPACEXTABLE where Customer='NASA (CRS)'
```

```
 * sqlite:///my_data1.db
Done.
```

| Total payload mass |
| --- |
| 45596 |

## Average payload mass carried by F9 v.1.1 for a launch

Using SQL built-in functions

Display average payload mass carried by booster version F9 v1.1

```
]: %sql select avg(PAYLOAD_MASS__KG_) as 'Average mass' from SPACEXTABLE where Booster_Version like 'F9 v1.1%'
```

```
 * sqlite:///my_data1.db
Done.
```

]:

| Average mass |
| --- |
| 2534.6666666666665 |

List the date when the first succesful landing outcome in ground pad was acheived.

*Hint:Use min function*

```
%sql select distinct Landing_Outcome from SPACEXTABLE
```

```
 * sqlite:///my_data1.db
Done.
```

| Landing_Outcome |
|---|
| Failure (parachute) |
| No attempt |
| Uncontrolled (ocean) |
| Controlled (ocean) |
| Failure (drone ship) |
| Precluded (drone ship) |
| Success (ground pad) |
| Success (drone ship) |
| Success |
| Failure |
| No attempt |

```
%sql select min(Date) from SPACEXTABLE where Landing_Outcome = 'Success (ground pad)'
```

```
 * sqlite:///my_data1.db
Done.
```

| min(Date) |
|---|
| 2015-12-22 |

## Date of first successful landing on ground pad

Using built-in SQL functions.

I realize that I could have just as easily used the UCASE or LCASE functions, but I wanted to do it that way. I first displayed all the landing outcomes, found what I needed there, Ctrl-C, Ctrl-V - and no errors in the register.

## List of Falcon 9 boosters

Using these boosters, the rocket successfully lifted a payload of between 4 and 6 tons into orbit, and then successfully landed on an drone ship.

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql select Booster_Version from SPACEXTABLE where Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4(
```

```
 * sqlite:///my_data1.db
Done.
```

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

## Total number of Falcon 9 missions with known landing outcome

ALL

List the total number of successful and failure mission outcomes

```
%sql select count (Landing_Outcome) as 'COUNT' from SPACEXTABLE
```

```
 * sqlite:///my_data1.db
Done.
```

| COUNT |
|---|
| 101 |

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
%sql select Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTABLE where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) fr
```

```
 * sqlite:///my_data1.db
Done.
```

| 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 |

## List of Falcon 9 boosters

These boosters lifted a record weight into orbit

## Unsuccessful landings on an drone ship in 2015

Using the built-in substr function to extract a date substring

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

**Note: SQLLite does not support monthnames. So you need to use substr(Date, 6,2) as month to get the months and substr(Date,0,5)='2015' for year.**

```
%sql select substr(Date, 6,2) as month, Booster_Version, Launch_site, Landing_Outcome from SPACEXTABLE where substr(Date,0,
```

```
 * sqlite:///my_data1.db
Done.
```

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

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.

```
%sql select Landing_outcome as 'Outcome', count (*) as 'COUNT' from SPACEXTABLE where Date between '2010-06-04' and '2017-0
```

```
 * sqlite:///my_data1.db
Done.
```

| 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 |

Counting the number of successful and unsuccessful landings of all types over a period

# Predictive analysis:

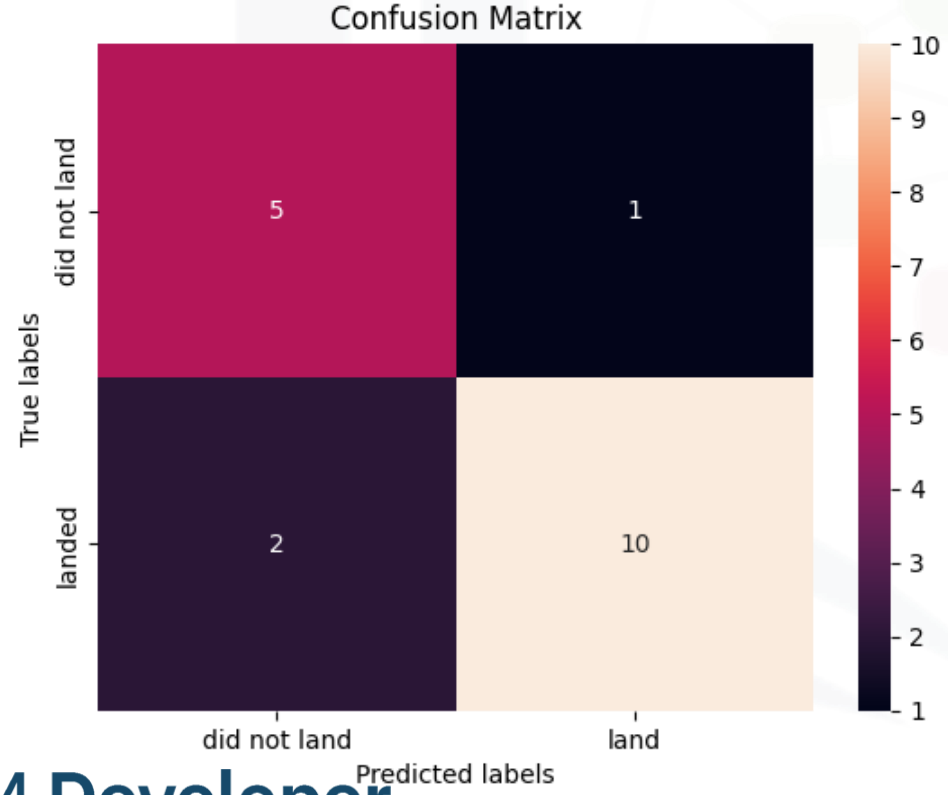
When evaluating the performance of machine learning algorithms, I proceeded from the following:

- A true positive is preferable to a false positive
- False negatives are preferable to false positives

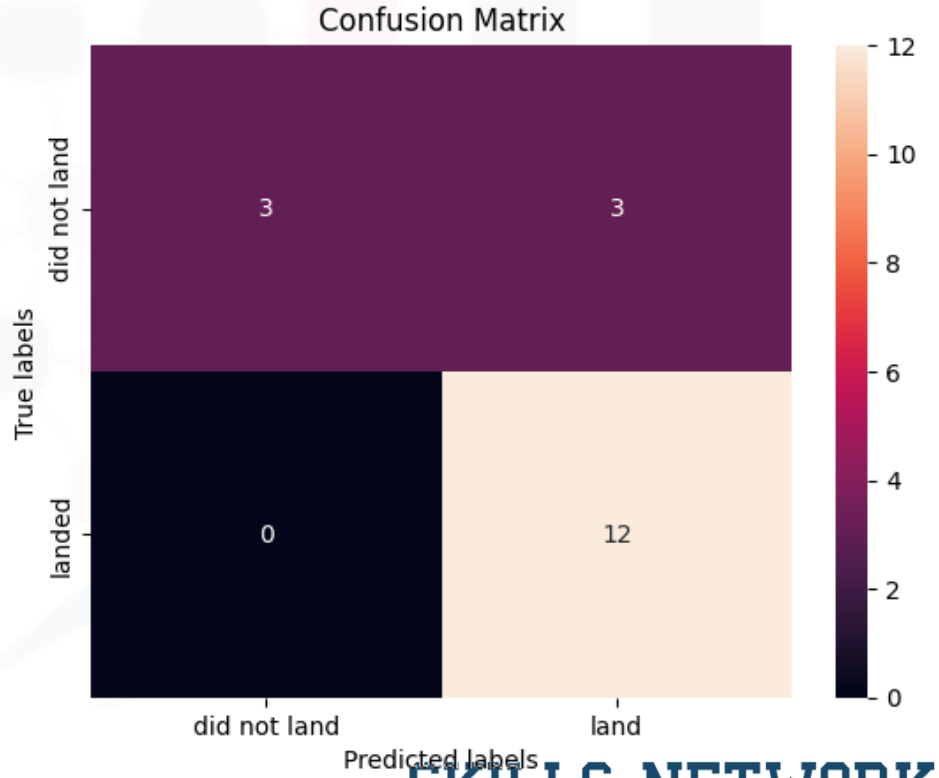
Based on this premise, the most preferred algorithm for evaluation is the decision tree algorithm.

# Comparison of algorithms

**Decision tree**




**Another alorythms**



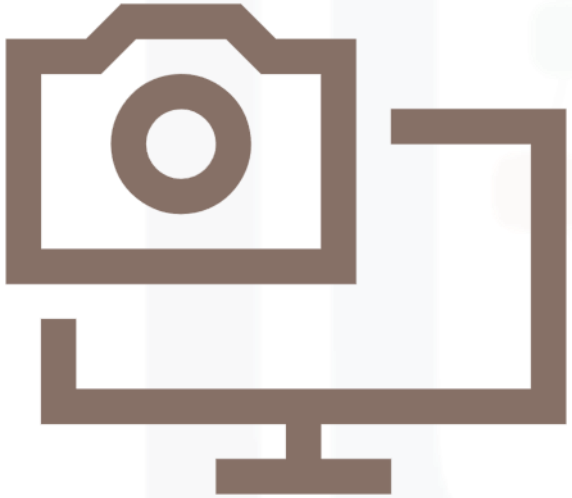
# Comparison of algorithms

Assessing the prediction accuracy using the standard method (algorithm.score(x_test, y_test)) returns the same number for all algorithms. Comparing the confusion matrices (which, by the way, are identical for all algorithms except the decision tree) tells us that the decision tree makes fewer false positives and more false negatives. Applying this approach in real-world settings eliminates positive bias and improves experimental results.

# Conclusion

- More launches made - a higher probability of success (both for an individual launch site and in general)
- The likelihood of a successful landing increased in 2013 and continues to increase through 2020
- Launches from the KSC LC-39A site were more successful than others.
- The probability of a successful flight is higher for orbits ES-L1, GEO, HEO, SSO, VLEO
- The best algorithm for assessing flight success is a decision tree.
- The main factor for a successful landing is accumulated experience.

# Application



| Орбита | Описание орбиты |
|---|---|
| LEO | Low Earth orbit – низкая околоземная орбита, высотой до 2000 км. Орбитальный период 128 минут. |
| VLEO | Very low Earth orbit – сверхнизкая околоземная орбита (до 450 км) |
| GTO | Геосинхронная орбита. 35786 км над экватором. |
| SSO или SO | Гелиосинхронная (или солнечно-синхронная) орбита. Почти полярная орбита, на которой спутник проходит над любой заданной точкой планеты в одно и то же солнечное время. |
| ES-L1 | Первая точка Лагранжа. Силы гравитации двух больших тел (в данном случае Земли и Солнца) в этой точке уравновешены. |
| HEO | Высокая эллиптическая околоземная орбита. |
| ISS | Низкая околоземная орбита, на которой расположена МКС |
| MEO | Геоцентрические орбиты в диапазоне от 2000 до 35786 км (между низкой околоземной и геосинхронной) |
| HEO | Геоцентрические орбиты выше геосинхронной |
| GEO | Круговая геосинхронная орбита 35786 км |
| PO | Полярная орбита |

# Application

- I express my gratitude to the educational portal Coursera (https://coursera.org) for the opportunity to take this course and try my skills in conditions close to “combat”.
- Due to a technical glitch, I was unable to save my work in Folium and Plotly Dash, and was unable to redo it before the deadline.