IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Carolin John
30th of March 2025

Outline

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

Executive Summary

- Summary of methodologies:
- First the data concerning SpaceX launches was obtained doing webscraping on the SpaceX Wikipedia-page and by doing a HTTP-request to the SpaceX-API
- After data wrangling with python and exploratory data analysis with SQL and Data Visualization, an interactive map was built with Folium and a dashboard was built with python dash
- Lastly predictive analysis was performed using the GridSearchCV classification with different estimators and the confusion-matrices were computed

Executive Summary

- Summary of results:
- Using a GridSearchCV-algorith with a DecisionTreeClassifier as estimator, one can accurately predict in 89% of cases whether a launch will be successful
- one of the most striking features of the data is the nearly continuous improvement in the yearly success rate of the launches from 0% to more than 80% (2019)
- There is great variation in success rates for launch-sites, orbits, payloads etc.
- the data shows great evolution of the launch-sites, booster-types, orbits etc. during the years
- The most prominent trend is the change in yearly success rates, which is why I would propose to use a classification algorithm in such a way, that the later launches are given more weight in predicting the outcome of a launch

Introduction

- The aim of this project is to predict whether the first stage of the Falcon 9 will land successfully and therefore can be reused.
- Since a reused Falcon 9 first stage saves the space company a lot of costs, it is necessary to give a good prediction in order to have competitive pricing
- It is the aim of this report to find a good predictive method and draw conclusions about the influence of certain variables

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data Collection by doing a HTTPS-request of the SpaceX-API
  - Data Collection by doing Webscraping with BeautifulSoup on the SpaceX Wikipedia page
- Perform data wrangling
  - Using Python's Pandas package the data was transformed into a dataframe
  - With the function value_counts() and is_null() the distribution of the data and missing values were checked
  - A new dataframe column with training outcomes (0 or 1) was added
  - At last the dataframe was saved as a csv-File

Methodology

Executive Summary

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    - read the data into a dataframe, split the result-data from the dataframe
    - Standardize the data using StandardScaler
    - Use train-test-split() in order to create training and testing data
    - Do a GridSearchCV with different estimators (LogisticRegression, KNN, DecisionTreeClassifier, SVM)
    - For each estimator plot the confusion_matrix and compute the accuracy score

Data Collection

- Data sets were collected in two ways:
  - Request to the SpaceX-API
  - Webscraping of the Wikipedia-Site concerning SpaceX Launches

Data Collection – SpaceX API

- Make HTTP-Request to SpaceX-API
- Convert Json-Response to Pandas dataframe (df)
- Extract relevant information from dataframe using pandas and python helper-functions
- Clean up the data (extract relevant information from strings)
- Change data-types if necessary (convert date to date-format)
- Deal with missing values (replace with mean)
- Export final dataframe to .csv-file

- **https://github.com/C-D-John/Datascience_capstone/blob/main/jupyter-labs-spacex-data-collection-api.ipynb**

HTTP-Request to SpaceX-API

- Request and Parse the SpaceX launch data using the Get-request
- Get a Json-File

Convert Json-Response

- Convert Json-Response to Pandas Dataframe

Clean up Dataframe

- Extract only the relevant information using python
- Clean up the data
- Change data-types if necessary
- Deal with missing values
- Export data-frame to csv

Data Collection - Scraping

- Webscrape Falcon 9 Launch-records using BeautifulSoup
- Use HTTP Get-Method to request Falcon 9 Launch HTML-page as HTTP-response
- Turn HTTP-reponse into BeautifulSoup-object
- Extract all tables from object using find_all-method
- Get third table and iterate through all table-elements in order to extract column-names and column-data
- Create dictionary with column-names and column-data and turn it into dataframe
- Save dataframe as .csv-file using Pandas
- https://github.com/C-D-John/Datascience_capstone/blob/main/jupyter-labs-webscraping.ipynb

Webscrape Falcon 9 launch records

- Use BeautifulSoup package from python
- Webscrape Wikipedia-Page of SpaceX-Launches
- Use HTTP Get-Method to request Falcon 9 Launch HTML-page as Http-response

Create a BeautifulSoup-object from HTTP-response

- Extract text from HTTP-response
- Turn it into BeautifulSoup object

Extract relevant information from BeautifulSoup object

- Extract all tables from BeautifulSoup object
- Extract the third table
- Iterate through all table elements and extract columns-names

Turn information into Dataframe

- Create empty dictionary with column names as keys
- Iterate through the table and extract rows into lists with dictionary-values
- Turn Dictionary into Dataframe using pandas
- Save Dataframe as .csv-File using Pandas

Data Wrangling

- Read the csv-File into a pandas dataframe
- Check missing values with function is_null()
- Check distribution of values with pandas function value_counts()
- Check datatype of columns with function dtypes, change if possible to float/integer
- Create a new columns and assign the outcomes to the training labels 1 (outcome successful) and 0 (outcome unsuccessful)
- Save Dataframe as csv-File
- **https://github.com/C-D-John/Datascience_capstone/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb**

Load Data into Dataframe	Explore data in Dataframe with pandas	Label outcomes with numerical values	Save Dataframe as .csv-File
• Read the csv-File into a dataframe with the function pd.read_csv()	• Check missing values with panda function is_null() • Check the distribution of values with panda function value_counts() for different columns • Check which outcomes are numerical or categorical. Change the data-type from object to float/int if possible with function astype()	• Create a new column and assign the outcomes to the training-labels 1 (outcome successful) and 0 (outcome unsuccessful)	

EDA with Data Visualization

- Since the aim is to notice a clue, whether a landing will be successful or not, I have used mostly scatter plots, where the points have different colors based on the success (scatter-plots, since each flight is a distinct event)
- In a lot of plots I have used the flight-number as x-axis in order to see the development of the success-rate over time (Payload-class, Launch-Site or Orbit as y-axis), I have also done a line-plot with the success-rate over the years
- Another interesting aspect was the relationship between the payload-mass and either the orbit or the launch-site, which were other scatterplots
- **https://github.com/C-D-John/Datascience_capstone/blob/main/edadataviz(2).ipynb**

EDA with SQL

Summary of SQL-queries:

- Unique launch sites
- Records where launch-site begins with certain string and limited numbers
- Total/average payload mass carried by certain boosters or for certain customers
- Number of successful and unsuccessful missions
- Limit the query to certain payload-ranges, certain dates, launch-sites or outcomes or a combination of multiple of those issues
- Rank the count of landing outcomes restricted to certain dates, et.

- **https://github.com/C-D-John/Datascience_capstone/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb**

Build an Interactive Map with Folium

- I added a circle and marker for each launch-site on the map to see the geographical distribution of the launch-sites
- I also added a folium.marker for each launch with a color indicating whether it was successful or not, in order to show the success rate for each launch site
- Finally, I picked a launch site and added markers to the nearest sites (coast, road, city etc.) and drew a line to the launch site in order to have an overview of the infrastructure

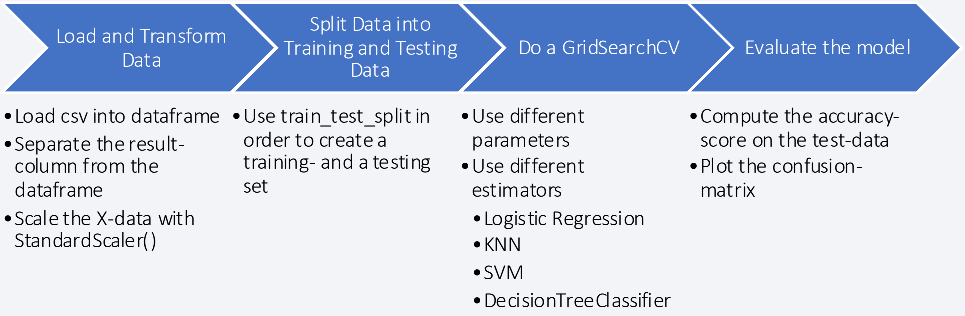
- **https://github.com/C-D-John/Datascience_capstone/blob/main/lab_jupyter_launch_site_location.ipynb**

Build a Dashboard with Plotly Dash

- I have added to plots two the dashboard:
  - A pie-chart that displays the total percentage of successful launches per site, when no site is chosen, or the percentage of successful launches per site, when a specific launch-site is chosen
  - A scatter-plot, that displays the success of a launch (class 0 or 1) over the payload, with shades for the different booster types
- I have added these plots, because one now can easily compare the success-rates for different launch-sites and see the influence of the payload and the booster
- **https://github.com/C-D-John/Datascience_capstone/blob/main/spacex_dash_app.py**

Predictive Analysis (Classification)

- Load data into dataframe
- Separate the result-column, transform the other data with StandardScaler()
- Create training and testing-data with train_test_split()
- Do a GridSearch with different parameters and estimators (LogisticRegression, KNN, SVM, DecisionTreeClassifier)
- Compute the accuracy-score on the test-data and plot the confusion-matrix
- **https://github.com/C-D-John/Datascience_capstone/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5.ipynb**



Results

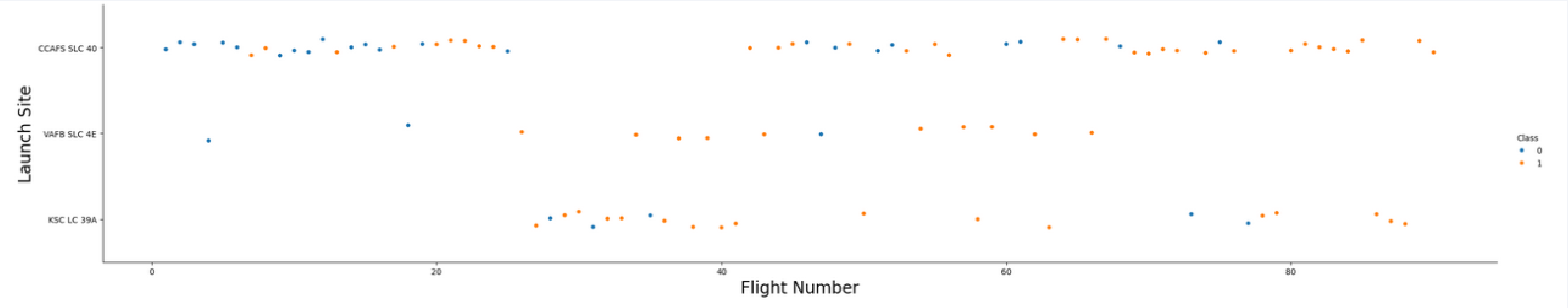
- Exploratory data analysis results
  - Launch success rate varies widely with regards to launch-site, year, orbit, payload etc. With a great increase in the success-rate over the years
- Interactive analytics demo in screenshots
- Predictive analysis results
  - Grid SearchCV algorithm with DecisionTreeClassifier estimator can with a percentage of 89% accuracy predict, whether a launch will be successful

Section 2

Insights drawn from EDA

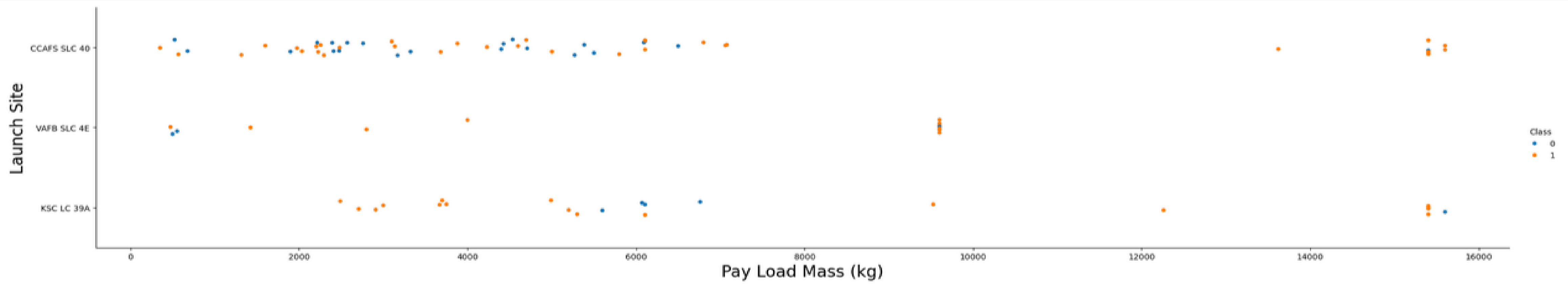
Flight Number vs. Launch Site

- Here one can see a scatter-plot of Flight Number vs. Launch-site, where successful landings are marked with a yellow dot
- One can see, that the success-rate improves with a growing number of flights and that additional launch-sites are added for later flight-numbers



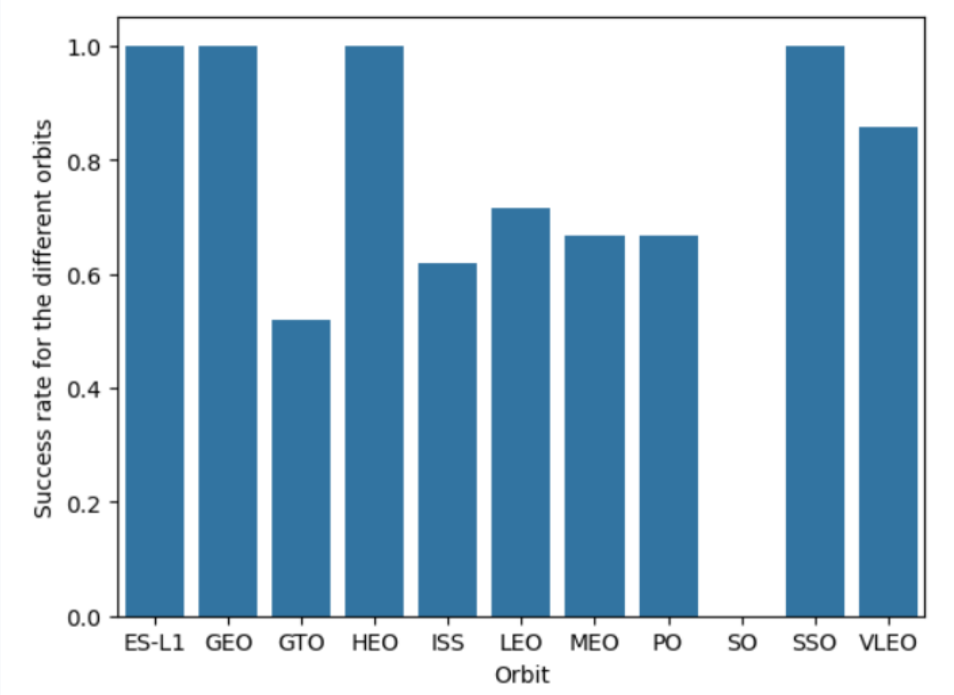
Payload vs. Launch Site

- Here one can see a scatter-plot of Payload vs. Launch Site with successful landing marked with a yellow dot
- One can see, that the distribution of payloads varies across the launch-sites, with the launch-site CCAFS SLC 40 having the greatest variation
- One can also see, that the percentage of successful landings varies by payload and launch-site



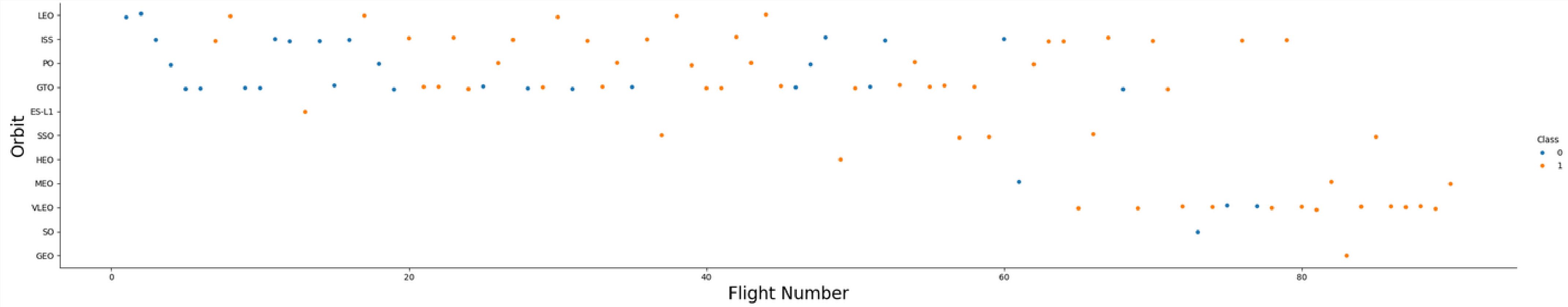
Success Rate vs. Orbit Type

- Here one can see a bar-chart of the relationship between orbit and success rate
- One can see, that the success rate varies widely between 0% (SO) and 100% (ES-L1, GEO, HEO, SSO)



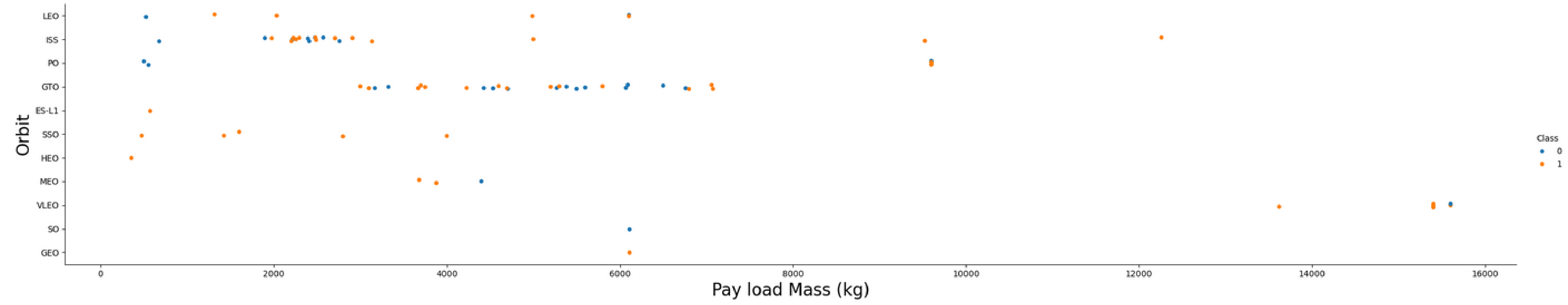
Flight Number vs. Orbit Type

- Here one can see a scatter plot of Flight number vs. Orbit type with successful landings marked with a yellow dot
- One can see, that the orbit varies widely over time



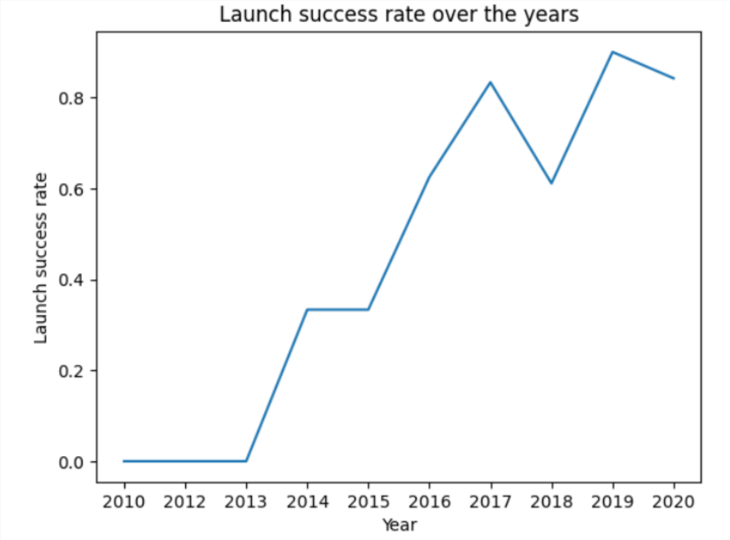
Payload vs. Orbit Type

- Here one can see a scatter plot of payload vs. orbit type with successful landings marked with a yellow dot
- One can see, that different payloads are associated with different orbits and that the variation of payloads also varies between the orbits
- For one orbit, the success-rate also varies with the payload



Launch Success Yearly Trend

- Here one can see a line chart of yearly average success rate
- One can see, that the yearly success rate is sharply rising from 0% in 2010 to more than 80% in 2019



All Launch Site Names

- Find the names of the unique launch sites:
- SQL: `SELECT DISTINCT(Launch_Site) FROM SPACEXTABLE`
- `The command DISTINCT queries for unique entries, so DISTINCT (Launch_Site) gets the unique Launch Sites from SPACEXTABLE`

```
%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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`
- `%sql SELECT * FROM SPACEXTABLE WHERE Launch_Site LIKE 'CCA%' LIMIT 5`
- The command 'LIKE'checks for a similar string and 'LIMIT' limits the output to the number called

```
%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_Outc
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (paracl
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 (paracl
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No att
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No att
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No att

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
- `SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer LIKE 'NASA (CRS)'`
- `Order 'SUM(PAYLOAD_MASS__KG_)' computes the total payload mass`

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer LIKE 'NASA (CRS)'
```

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version LIKE 'F9 v1.1'
- Order AVG(PAYLOAD_MASS_KG_) computes the average payload mass

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Booster_Version LIKE 'F9 v1.1'
```

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

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- `%sql SELECT MIN(Date) FROM SPACEXTABLE WHERE Landing_Outcome LIKE 'Success (ground pad)'`
- `MIN(Date)` is the first date
-

```
]: %sql SELECT MIN(Date) FROM SPACEXTABLE WHERE Landing_Outcome LIKE 'Success (ground pad)'

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

]:

MIN(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- `SELECT Booster_Version FROM SPACEXTABLE  WHERE Landing_Outcome LIKE 'Success (drone ship)' AND (PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ <6000)`
- `'AND' connects different queries`

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 LIKE 'Success (drone ship)' AND (PAYLOAD_MASS__KG_ > 40
```

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

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

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
- 'COUNT' calculates the number of outcomes
- `SELECT COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE '%Success%'`
- `SELECT COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE '%Failure%'`

```
%sql SELECT COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE '%Success%'
 * sqlite:///my_data1.db
Done.
```

COUNT(*)
100

```
%sql SELECT COUNT(*) FROM SPACEXTABLE WHERE Mission_Outcome LIKE 'Failure%'
 * sqlite:///my_data1.db
Done.
```

COUNT(*)
1

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- `SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)`
- This query needs a nested query

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

2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- `SELECT substr(Date, 6,2) AS MONTH, Booster_Version, Launch_Site, Landing_Outcome FROM SPACEXTABLE WHERE Date LIKE '%2015%' AND Landing_Outcome LIKE '%Failure (drone ship)%'`
- `Substr()` takes a substring out of a string, in this case the month

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

- 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
- `SELECT Landing_Outcome, COUNT(*) FROM SPACEXTABLE WHERE (Date BETWEEN '2010-06-04' AND '2017-03-21' ) GROUP BY Landing_Outcome ORDER BY COUNT(*) DESC`
- `'GROUP BY' groups data according to data in 1 column`

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

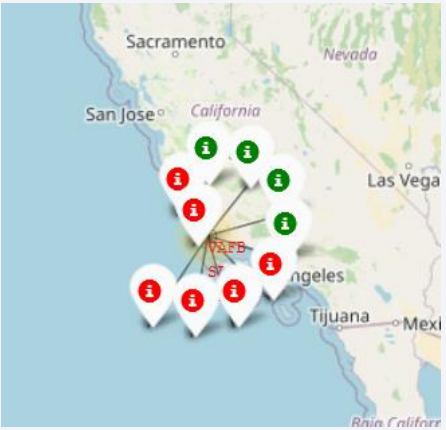
Launch Sites Proximities Analysis

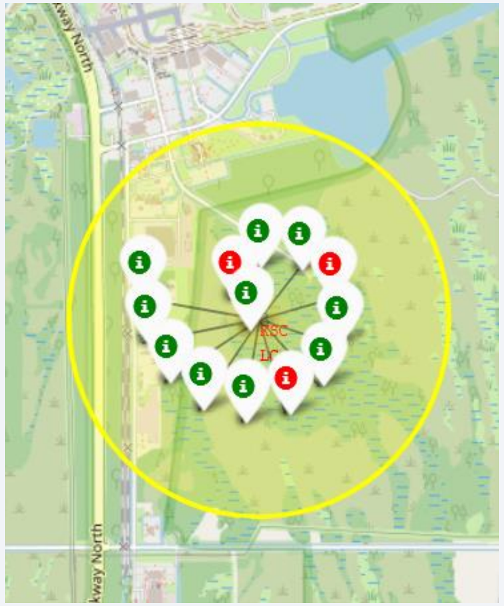
Distribution of Launch Sites

- Each launch site is marked by a yellow dot and the name of it is written next to it with a red marker.
- Every launch site is in the most Southern part of the US, very close to the coast, with 3 launch sites right next to each other

Distribution of successful launches

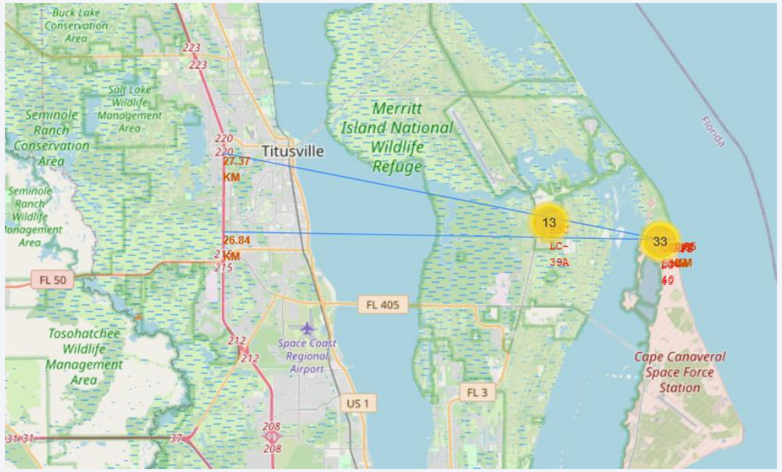
- If one puts a colored marker on the map for each launch, one can see, that the vast majority of launches were done in Florida
- One can see that the percentage of successful landings varies widely among the launch-sites, ranging from 27% success to 77% success

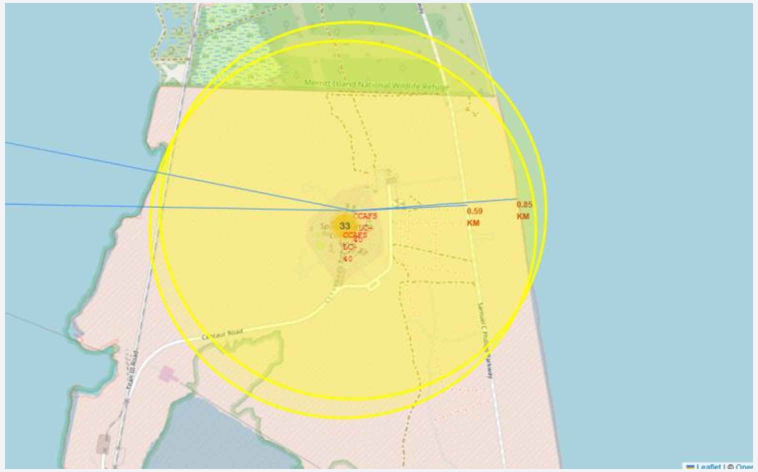




Distance of Launch Sites to Infrastructure

- Here is the proximity of the launch-site CCAFS SLC-40:
  - Distance to coast: 0.85km --> the most prominent feature of ALL site is proximity to coast
  - Distance to nearest street: 0.59km
  - Distance to nearest interstate: 26.84km
  - Distance to nearest city: 27.37 km



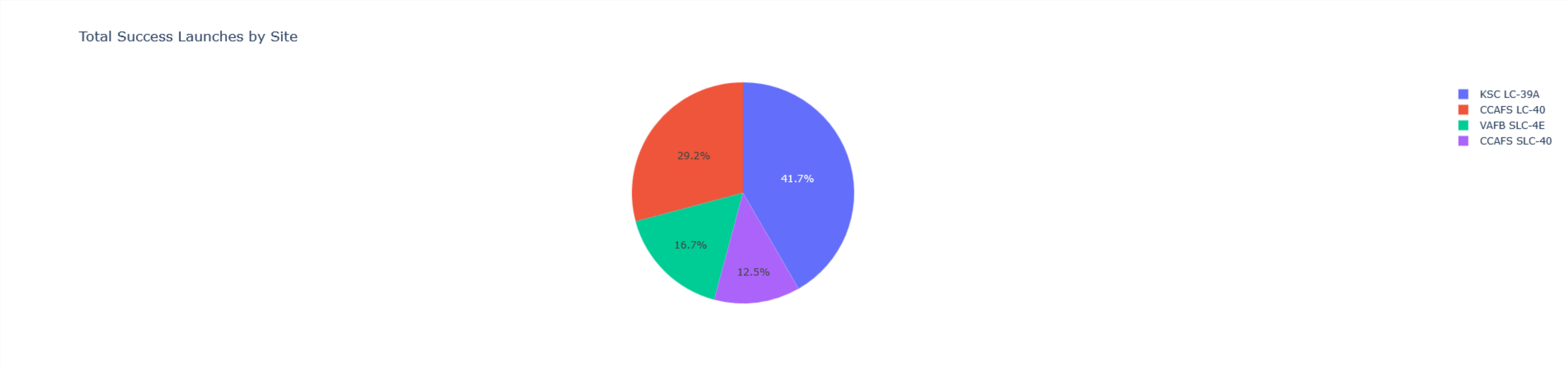


Section 4

Build a Dashboard with Plotly Dash

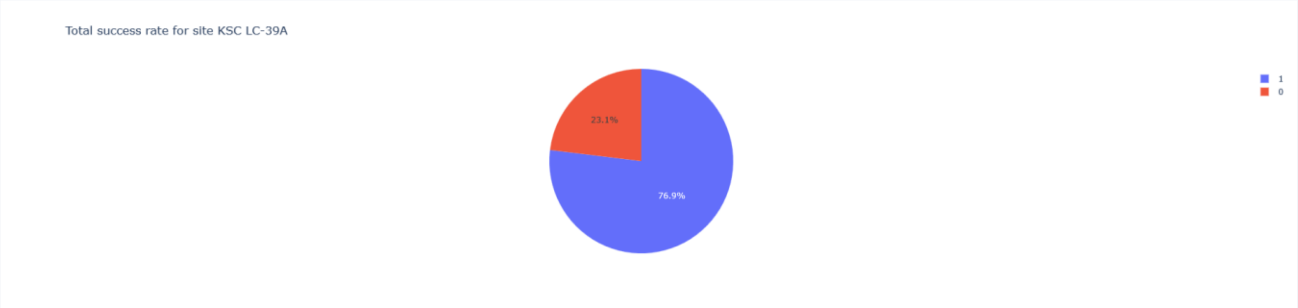
Overview Launch success count

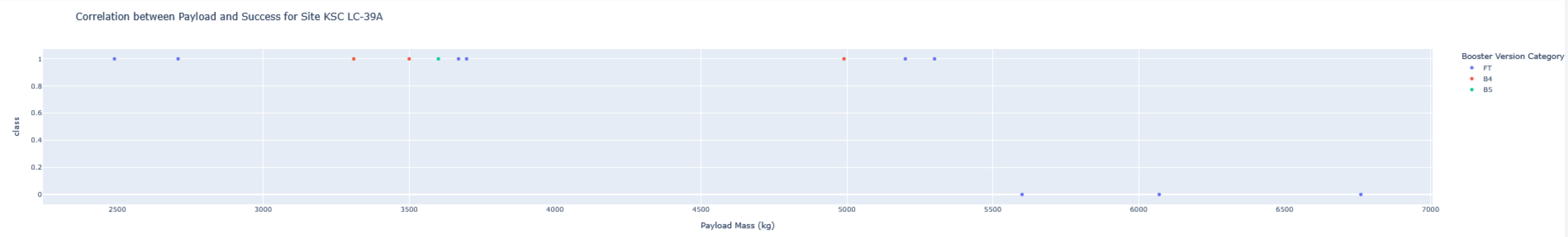
- The launch success count is not distributed evenly across the 4 launch-sites
  - The site KSC LC 39A contributes nearly 42% to all successful launches, whereas the site CC AFS SLC-40 contributes only 12.5% to all successful launches
  - Two sites (KSC LC-39A and CCAFS LC 40) contribute more than 70% of all successful launches



Highest Launch success ratio

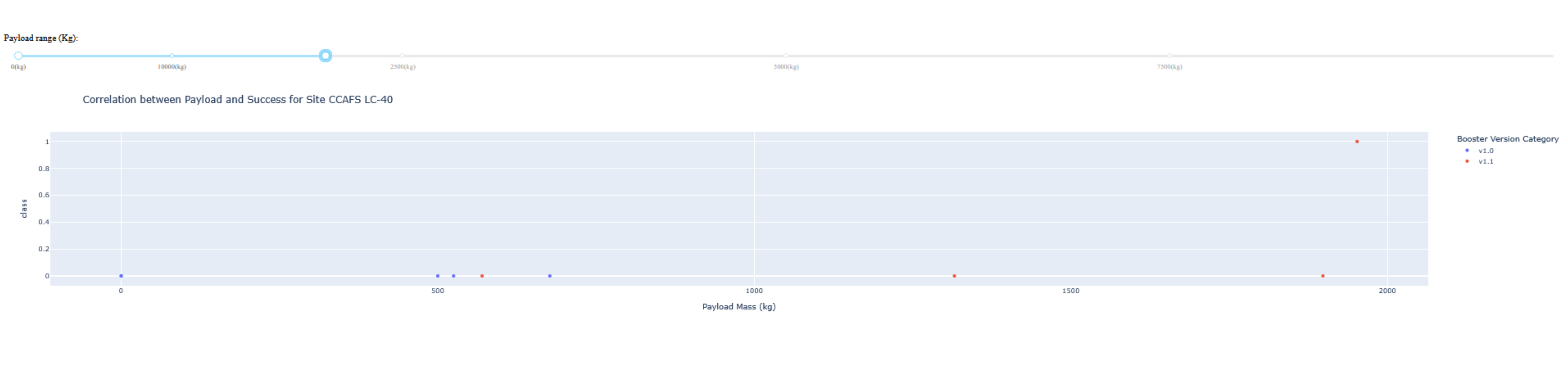
- the highest launch success ratio has the site KSC LC 39A with a success ratio of 76.9%
- When looking at the payload one can see, that ALL launches on this site with a payload <5500kg were successful, while ALL launches >5500kg were failures





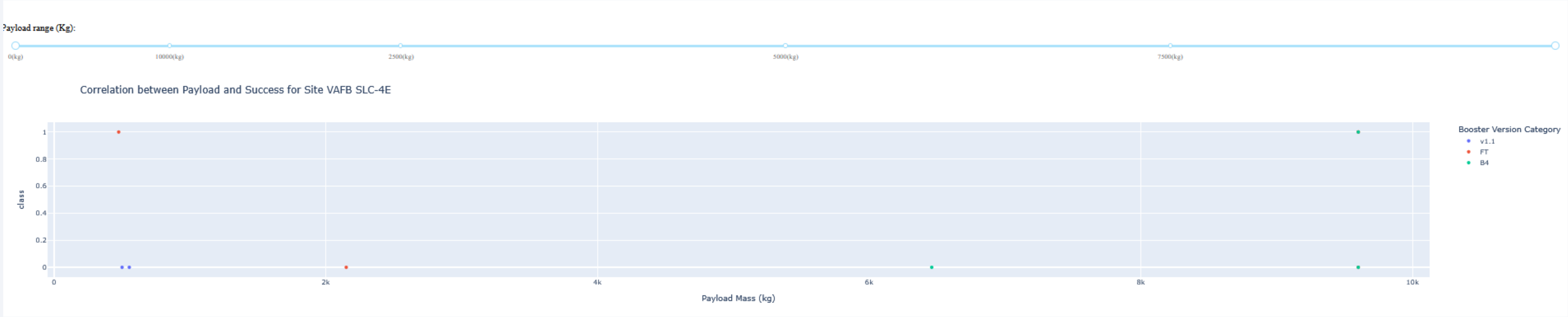
Influence Payload/Booster to Launch Outcome

- CCAFS LC 40
  - Every launch with a payload <1900 kg is a failure
  - There are only two booster versions v1.1 and v1.0 in this payload-range



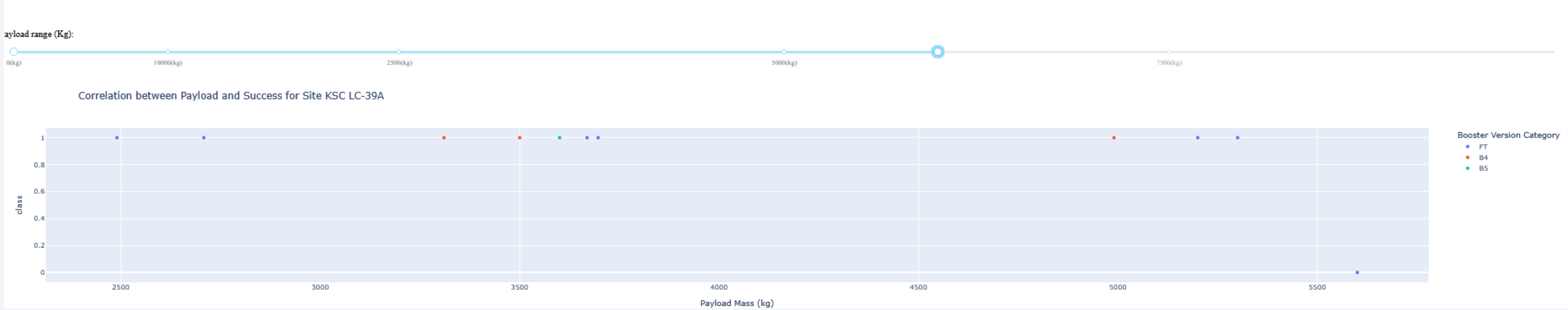
Influence Payload/Booster to Launch Outcome

- VAFB SLC 4E:
    - There are not many launches overall, so it is not easy to see a pattern
    - All the launches with a payload between 500kg and 9500kg failed



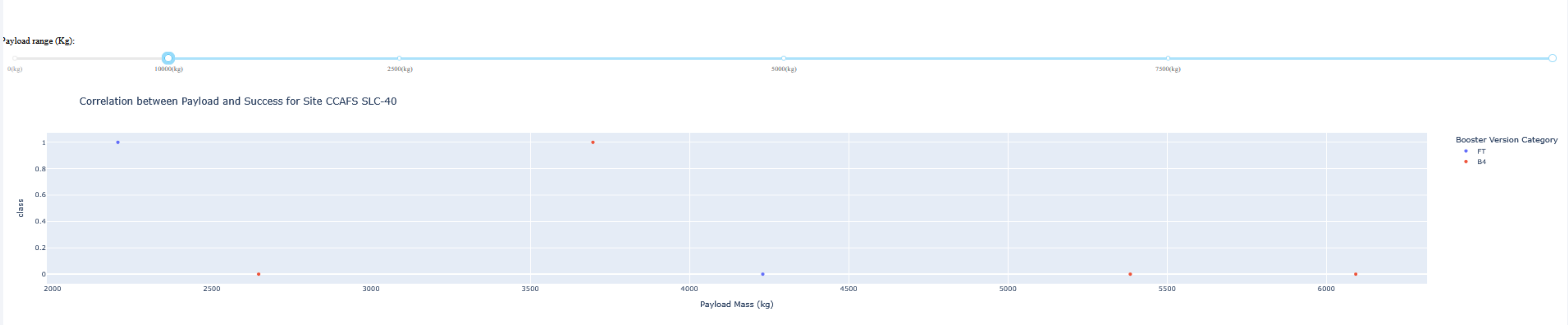
Influence Payload/Booster to Launch Outcome

- KSC LC-39A:
  - The minimum payload is around 2490kg
  - ALL launches under 5500kg are successful



Influence Payload/Booster to Launch Outcome

- CCAFS SLC-40:
  - There are only two booster versions (FT and B4) used on this site
  - No launch with a payload >3700 kg was successful

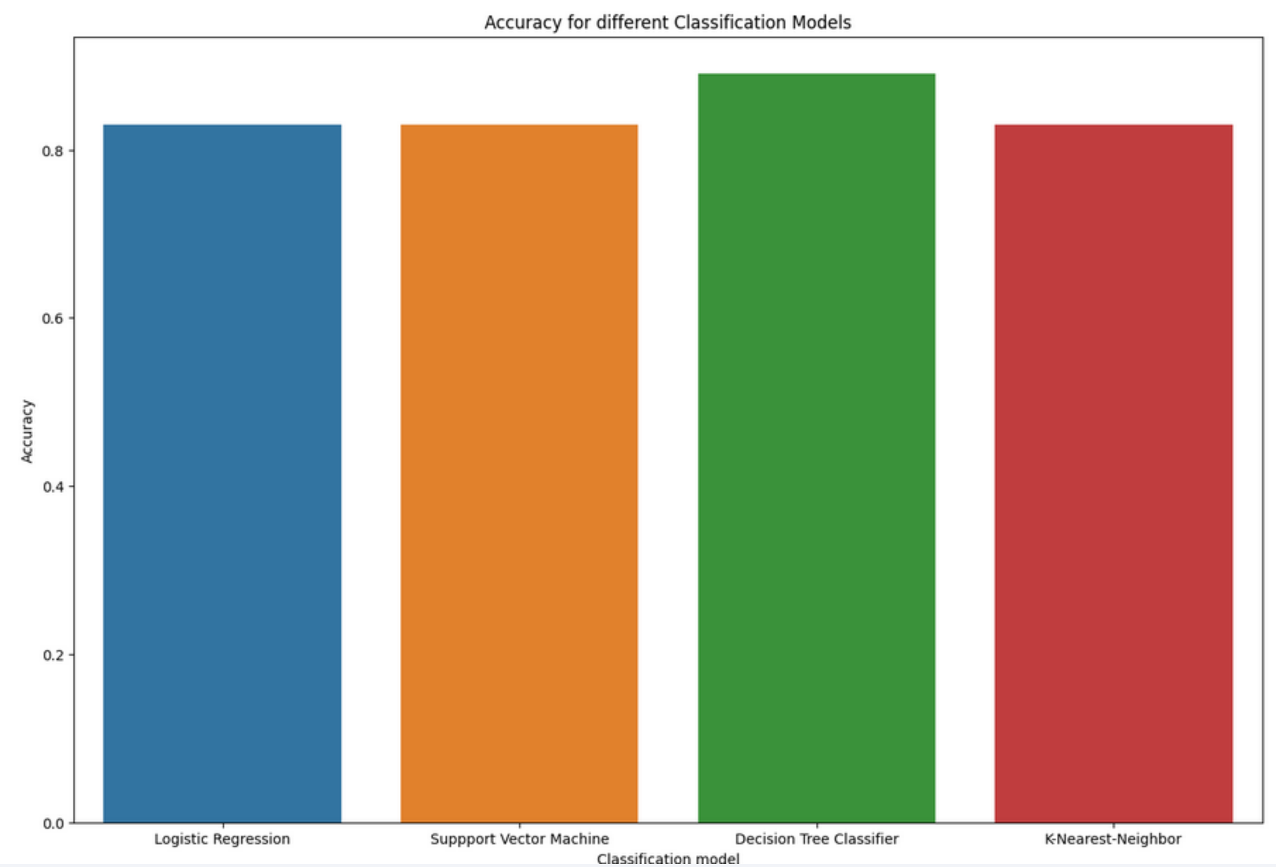


Section 5

Predictive Analysis (Classification)

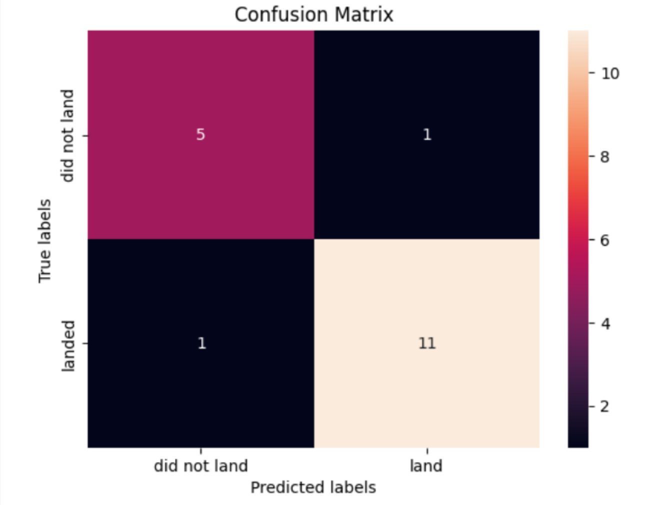
Classification Accuracy

- The highest accuracy has the Decision Tree Classifier Model with 89%



Confusion Matrix

- This is the confusion matrix of the Decision-Tree-Classifier model
- The overall accuracy is 89%
- It has
  - 11 True Positives (the model successfully predicted it would land)
  - 5 True Negatives (the model successfully predicted it would not land)
  - 1 False Positive (the model falsely predicted a successful landing)
  - 1 False Negative (the model falsely predicted an unsuccessful landing)



Conclusions

- There is a significant change in the use of certain launch-sites over time
- For each launch-site a significant increase in the success-rate over time was found
    - For each launch-site the last 5 launches were successful
    - For site CCAFS-SLC 40 the FIRST 5 launches were failures, for site VAFB SCL 4E the first two were
- There is a wide spread in payloads for each launch-site
- The success rate varies greatly by orbit type with some orbits have a 0% success rate (SO) and some having a 100% success rate (ES-L1, GEO HEO, SSO)
- There is a change in the preferred orbits over time from LEO, ISS, PO, GTO to GEO and VLEO

Conclusions

- The success rate has increased for each orbit over time
- The orbit used varies by the used payload with some orbits for smaller payloads (ISS) and some for larger (VLEO)
- There is a nearly continuous improvement for the yearly rate for successful launches from 0% (year 2010) to over 80% (2019) with only dips in 2018 and 2020
- A GridSearchCV-algorithm with a DecisisonTreeClassifier estimator predicts whether a launch will be successful with a 89% accuracy, with the same number of False Positives and False Negatives
- The most striking feature of the data was dependance of the success-rate on the year of the launch with a marked increase in success towards the later years. It is therefore advisable to use the classification in such a way, that the later launches have more weight in the prediction of future launches

Appendix

- All of the code used for this presentation can be found on my github page
- **https://github.com/C-D-John/Datascience_capstone**

Thank you!