APPLIED DATA SCIENCE CAPSTONE PROJECT

Quy Duong Nguyen

21st May 2024

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
  - Visualization – Charts
  - Dashboard
- Conclusion
- Appendix

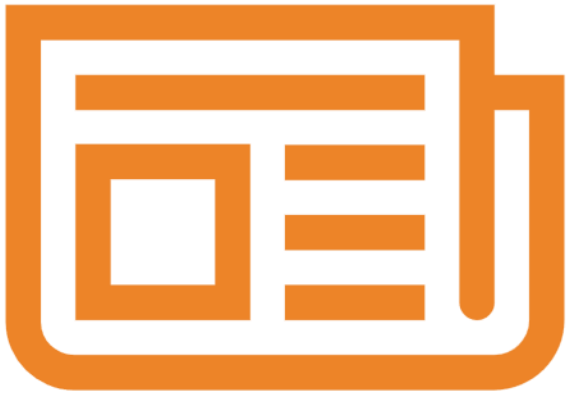
EXECUTIVE SUMMARY

- Collected data from SpaceX API and SpaceX Wikipedia
- Exploratory Data Analysis (EDA) including
  - SQL
  - Folium maps
  - Dashboards
- Gathered relevant columns as features and changed all categorical variables to binary using one hot coding
- Standardized data and applied GridSearchCV to find best parameters for machine learning models (Logistic Regression, Support Vector Machine, Decision Tree, KNN Classification)
- Visualized performances for all models

INTRODUCTION

- Background
  - Commercial Space Age
  - SpaceX has best pricing (62 million USD) compared to all other competitors (165 million USD)
  - More affordable prices thanks to ability to recover parts of rocket (Stage 1)
  - SpaceY wants to compete with SpaceX
- Problem: SpaceY tasks me to train a machine learning model to predict successful Stage 1 recovery

METHODOLOGY



- Data collection: combined data from SpaceX API and SpaceX Wikipedia page
- Data wrangling: classifying true landings as successful and unsuccessful otherwise
- Exploratory Data Analysis (EDA) with visualization and SQL
- Interactive Visual Analytics with Folium and Plotly Dash
- Predictive Analysis with models: tuned models with GridSearchCV

OVERVIEW OF DATA COLLECTION

- The data collection process included both API requests from SpaceX API and web scraping data from a table in Space X's Wikipedia Entry
- **SpaceX API Columns**: FlightNumber, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude
- Wikipedia Webscraping Columns: Flight No., Launch Site, Payload, PayloadMass, Orbit, Customer, Launch Outcome, Version Booster, Booster Landing, Date, Time

DATA COLLECTION: SPACEX API

- Request (SpaceX APIs)
- .Json file and Lists (Launch Site, Booster Version, Payload)
- Apply Json_normalize to the DataFrame from JSON
- Create a dictionary with relevant data
- Cast dictionary to a DataFrame
- Filter data to only include Falcon 9 launches
- Impute missing PayloadMass values with mean
- Github URL: IBM-Data-Science-Professional-/jupyter-labs-spacex-data-collection-api.ipynb at main · UchihaDuongbito/IBM-Data-Science-Professional- (github.com)

DATA COLLECTION: WEB SCRAPING

- Request (Wikipedia HTML)
- Apply BeautifulSoup html parser
- Find launch info in html table
- Create a dictionary with relevant data
- Iterate through table cells to extract data to dictionary
- Cast dictionary to a DataFrame
- Github URL: IBM-Data-Science-Professional-/jupyter-labs-webscraping.ipynb at main · UchihaDuongbito/IBM-Data-Science-Professional- (github.com)

DATA WRANGLING

- Create a training table label with landing outcomes where successful is 1 and failure is 0
- Outcome column has 2 components: "Mission Outcome" and "Landing Location"
- New training label column "class" has a value of 1 if "Mission Outcome" is True and 0 otherwise
- True ASDS, True RTLS and True Ocean: 1
- None None, False ASDS, None ASDS, False Ocean, False RTLS: 0
- Github URL: IBM-Data-Science-Professional-/labs-jupyter-spacex-Data wrangling.ipynb at main · UchihaDuongbito/IBM-Data-Science-Professional-(github.com)

EDA USING SQL

- Loaded dataset into db2 database
- Queried using SQL Python
- Queries were made to get a better understanding of the dataset
- Queried information about launch site names, mission outcomes, various pay load sizes of customers and booster versions, landing outcomes
- Github URL: IBM-Data-Science-Professional-/jupyter-labs-eda-sql-coursera_sqllite.ipynb at main · UchihaDuongbito/IBM-Data-Science-Professional- (github.com)

EDA USING PANDAS AND MATPLOTLIB

- Visualization were performed on variables Flight Number, Payload Mass, Launch Site, Orbit, Class and Year
- Flight Number vs Payload Mass, Flight Number vs Launch Site, Payload Mass vs Launch Site, Orbit vs Success Rate, Flight Number vs Orbit, Payload vs Orbit, and Success Yearly Trend
- Scatter, line, and bar plots were used to compare the relationship between variables to decide if a relationship exists so that they could be trained in the machine learning model later on
- Github URL: IBM-Data-Science-Professional-/edadataviz.ipynb at main · UchihaDuongbito/IBM-Data-Science-Professional- (github.com)

INTERACTIVE VISUAL ANALYTICS WITH FOLIUM

- Folium maps mark Launch Sites, successful and unsuccessful landings, and a proximity example to key locations: Railway, Highway, Coast and City
- From this, we can understand how to choose the appropriate launch sites. Successful landings and their locations are also visualized.
- Github URL: IBM-Data-Science-Professional-/lab_jupyter_launch_site_location.ipynb at main · UchihaDuongbito/IBM-Data-Science-Professional- (github.com)

INTERACTIVE VISUAL ANALYTICS WITH PLOTLY DASH

- Folium maps mark Launch Sites, successful and unsuccessful landings, and a proximity example to key locations: Railway, Highway, Coast and City
- From this, we can understand how to choose the appropriate launch sites. Successful landings and their locations are also visualized.
- Github URL: IBM-Data-Science-Professional-/spacex_dash_app.py at main · UchihaDuongbito/IBM-Data-Science-Professional- (github.com)

PREDICTIVE ANALYSIS

- Overall procedure:
- Split label column "class" from the dataset
- Fit and transform features using StandardScaler
- Split the data into training and testing sets
- Apply GridSearchCV (cv equal to 10) to find optimal parameters
- Apply the score method on the testing sets
- Apply Confusion Matrix
- Apply barplot to compare performances of all models
- Github URL: IBM-Data-Science-Professional-/SpaceX_Machine Learning Prediction_Part_5.ipynb at main · UchihaDuongbito/IBM-Data-Science-Professional- (github.com)

RESULTS

DATA COLLECTION: SPACEX API

Only include Falcon 9 Launches

Dealing with missing values in Payload

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
89	86	2020-09-03	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3032383ecb6bb234e7ca	5.0
90	87	2020-10-06	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	3	True	True	True	5e9e3032383ecb6bb234e7ca	5.0
91	88	2020-10-18	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	6	True	True	True	5e9e3032383ecb6bb234e7ca	5.0
92	89	2020-10-24	Falcon 9	15600.0	VLEO	CCSFS SLC 40	True ASDS	3	True	True	True	5e9e3033383ecbb9e534e7cc	5.0
93	90	2020-11-05	Falcon 9	3681.0	MEO	CCSFS SLC 40	True ASDS	1	True	False	True	5e9e3032383ecb6bb234e7ca	5.0

90 rows × 17 columns

```
FlightNumber      0
Date              0
BoosterVersion    0
PayloadMass       0
Orbit             0
LaunchSite        0
Outcome           0
Flights           0
GridFins          0
Reused            0
Legs              0
LandingPad       26
Block             0
ReusedCount       0
Serial            0
Longitude         0
Latitude          0
dtype: int64
```

You should see the number of missing values of the `PayLoadMass` change to zero.

DATA COLLECTION: WEB SCRAPING

- The desired dataframe

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
df.head()
```

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
2	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	F9 v1.0B0003.1	No attempt\n	4 June 2010	18:45
3	1	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0003.1	No attempt	4 June 2010	18:45
4	2	CCAFS	Dragon	525 kg	LEO	NASA	Success\n	F9 v1.0B0004.1	No attempt\n	8 December 2010	15:43

DATA WRANGLING

- The dataframe with new "class" column

```
df.head(5)
```

ersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
alcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
alcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
alcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
alcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
alcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

EDA WITH SQL

- Names of unique launch sites in the space mission

Task 1

Display the names of the unique launch sites in the space mission

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

EDA WITH SQL

Launch sites begin with the string "CCA"

EDA WITH SQL

- Total paymass load carried by boosters launched by NASA (CRS)
- CRS stands for Commercial Resupply Services, indicating that these payloads were sent to the International Space Station (ISS)

Task 3

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTABLE WHERE Customer like "NASA (CRS)"
```

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

SUM(PAYLOAD_MASS__KG_)
45596

EDA WITH SQL

- Average payload mass carried by booster version F9 v1.1

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_)
2534.6666666666665

EDA WITH SQL

- The date of the 1st successful landing outcome in ground pad
- We can see that before ground pad landing was unachievable until the end of 2015

Task 5

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

Hint:Use min function

```
%sql SELECT MIN(Date) FROM SPACEXTABLE WHERE Landing_Outcome = "Success (ground pad)"
```

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

MIN(Date)
2015-12-22

EDA WITH SQL

- Names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than six thousand

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 (Mission_Outcome like "Success")
AND (Landing_Outcome like "Success (drone ship)")
AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000)
```

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

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

EDA WITH SQL

- Total number of succeesful and failůe mission outcomes
- It seems that SpaceX achieves its mission outcome 99 percent of all times, meaning the majority of the landing failures are calculated
- Surprisingly, there turns out to be 1 launch with an unknown payload status and unfortunately, 1 launch failed in flight

Task 7

List the total number of successful and failure mission outcomes

```
%sql SELECT mission_outcome, count(*) as Count FROM SPACEXTABLE GROUP by mission_outcome ORDER BY mission_outcome
```

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

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

EDA WITH SQL

- All booster verions having carried the maximum payload mass
- It is quite obvious that these versions are quite similar and all of them are of the F9 B5 B10xxx
- There's likely to be a correlation between payload mass and the booster version

Task 8

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

```
%sql SELECT Booster_Version FROM SPACEXTABLE WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
```

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

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

EDA WITH SQL

- All records displaying month names, failure landing outcomes in drone ship, booster versions and launch sites in 2015

```
%sql SELECT SUBSTR(DATE, 6, 2), Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTABLE WHERE DATE like "2015%" AND Landing_Outcome like "Failure (drone_ship)"
```

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

SUBSTR(DATE, 6, 2)	Landing_Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

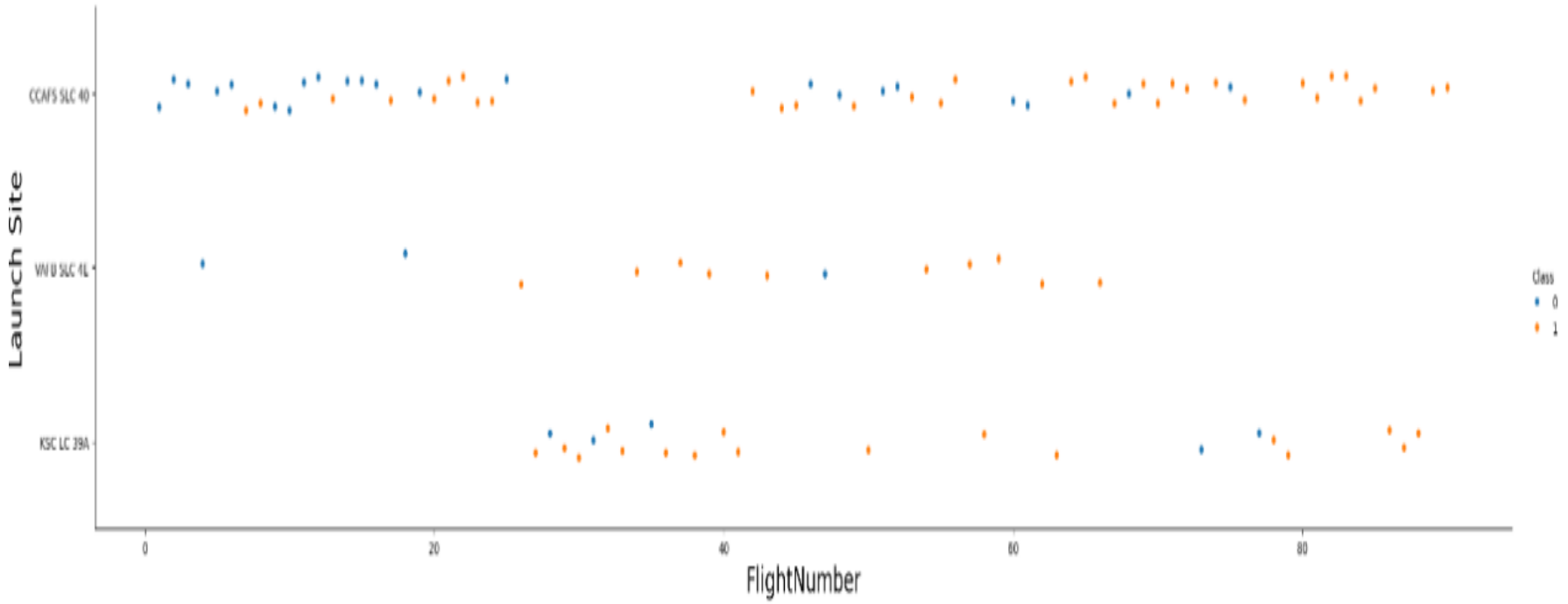
EDA WITH SQL

Ranking the count of landing outcomes between the given date (seen in the picture) by descending order

```
%sql SELECT Landing_Outcome, count(*) as TOTAL FROM SPACEXTABLE
WHERE DATE BETWEEN "2010-06-04" AND "2017-03-20" GROUP BY Landing_Outcome ORDER BY TOTAL DESC
```

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

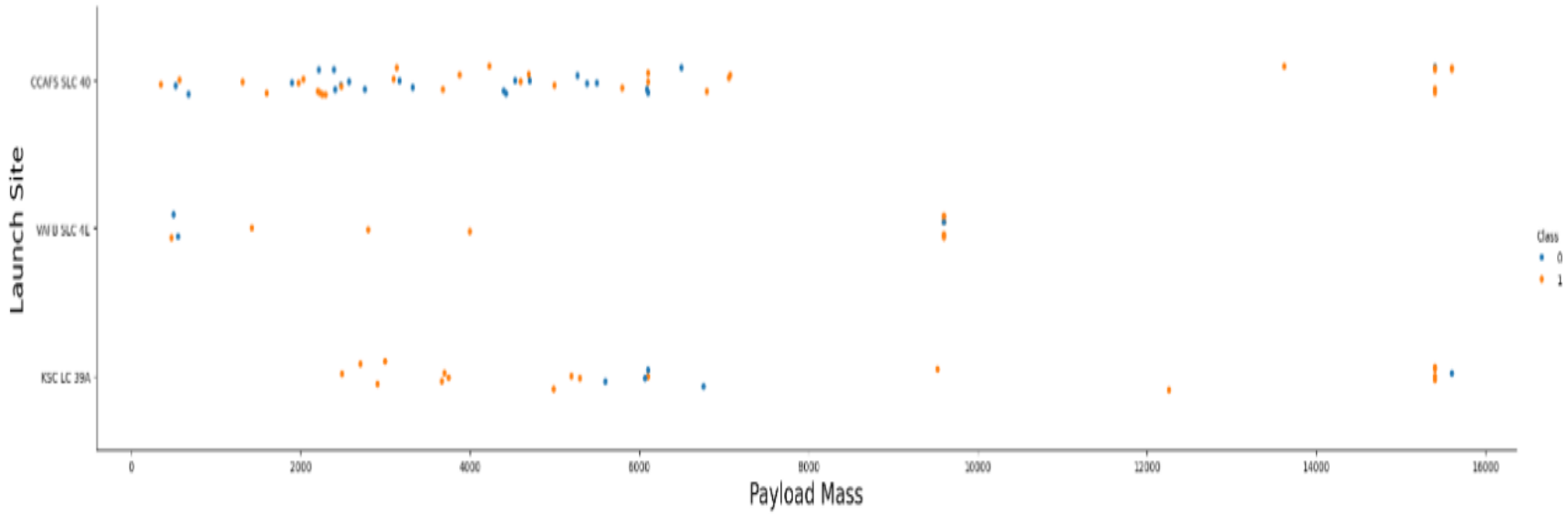
Landing_Outcome	TOTAL
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1



EDA WITH PANDAS AND MATPLOTLIB

Relationship between Flight Number and Launch Site: orange indicates successful, blue indicates failure

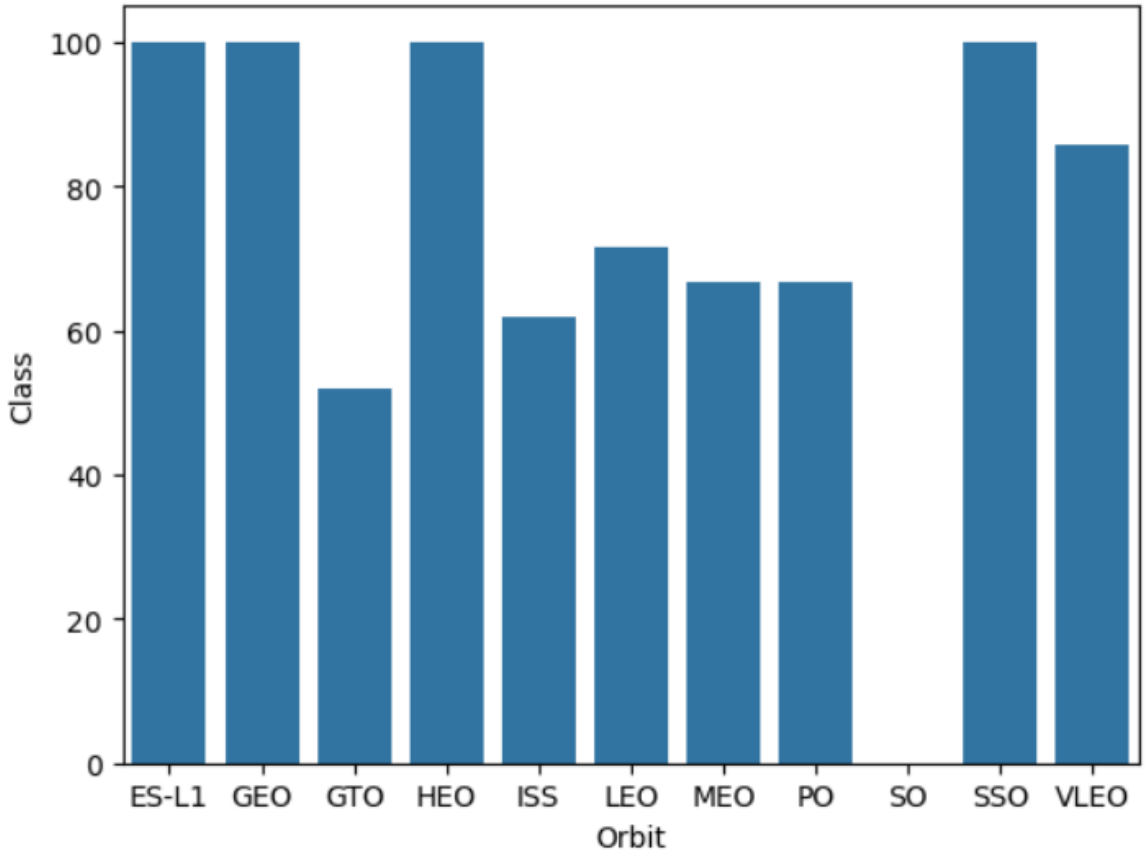
The scatter plot suggests a possible increase in success rate over time (Flight Number). From flight 20, the number of blue points, indicating 0 or failure missions, decreases significantly for all launch sites, hinting a breakthrough at the event of flight 20. Additionally, CCAFS can be considered the most reliable launch site as it has the most flights



EDA WITH PANDAS AND MATPLOTLIB

Relationship between Payload and Launch Site: orange indicates successful, blue indicates failed

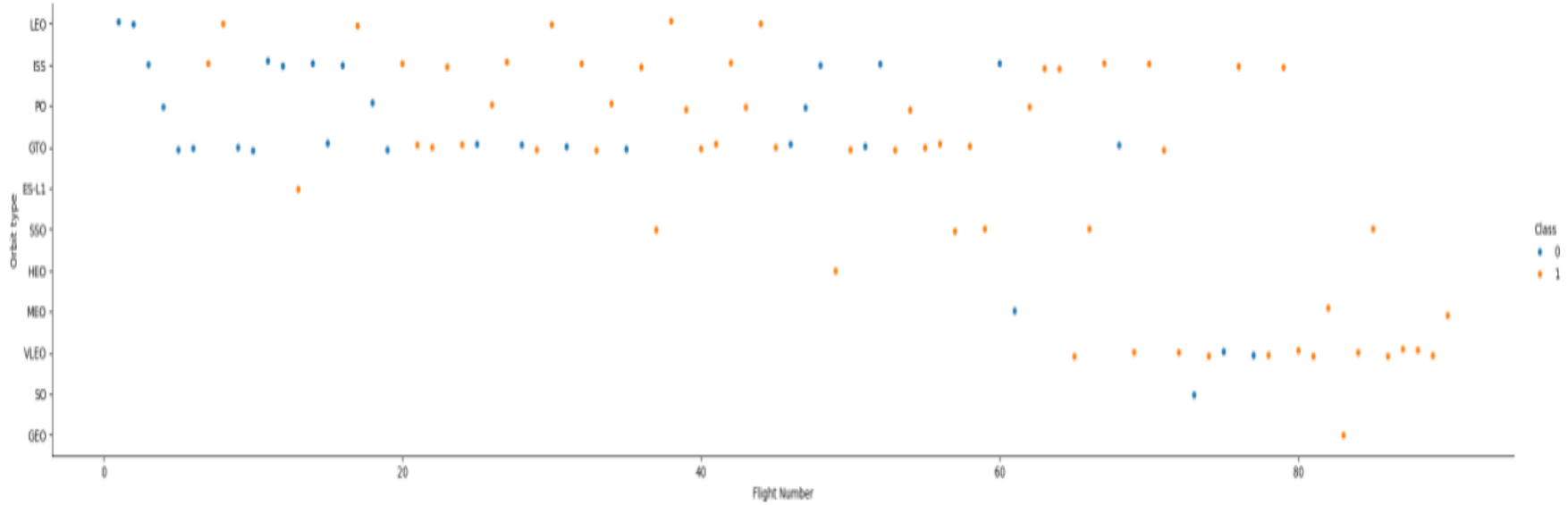
The majority of launches seem to have payload mass under six thousand kg, and there does not seem to be quite a difference in payload mass between launch sites



EDA WITH PANDAS AND MATPLOTLIB

Relationship between Payload and Launch Site:

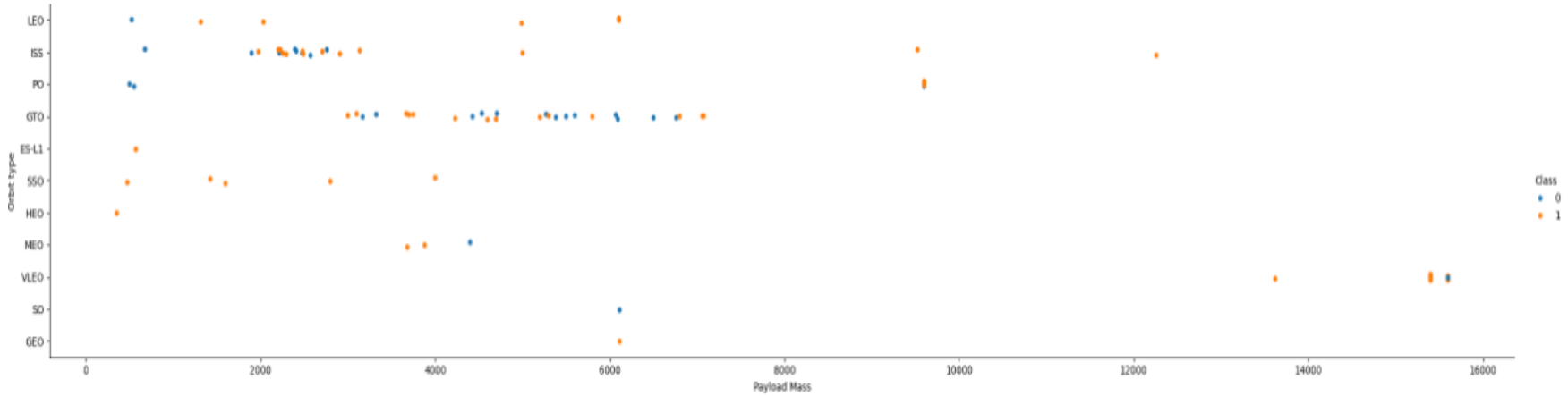
ES L1, GEO, HEO, and SSO have the success rate of 100 percent. ISS, LEO, MEO, PO, and VLEO have acceptable success rates, while GTO have the success rate just above fifty percent and SO has the lowest, 0 percent, success rate



EDA WITH PANDAS AND MATPLOTLIB

Relationship between Flight Number and Orbit: orange indicates successful, blue indicates failed

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. SpaceX started with LEO, which had moderate success rate, and turned to VLEO in recent launches

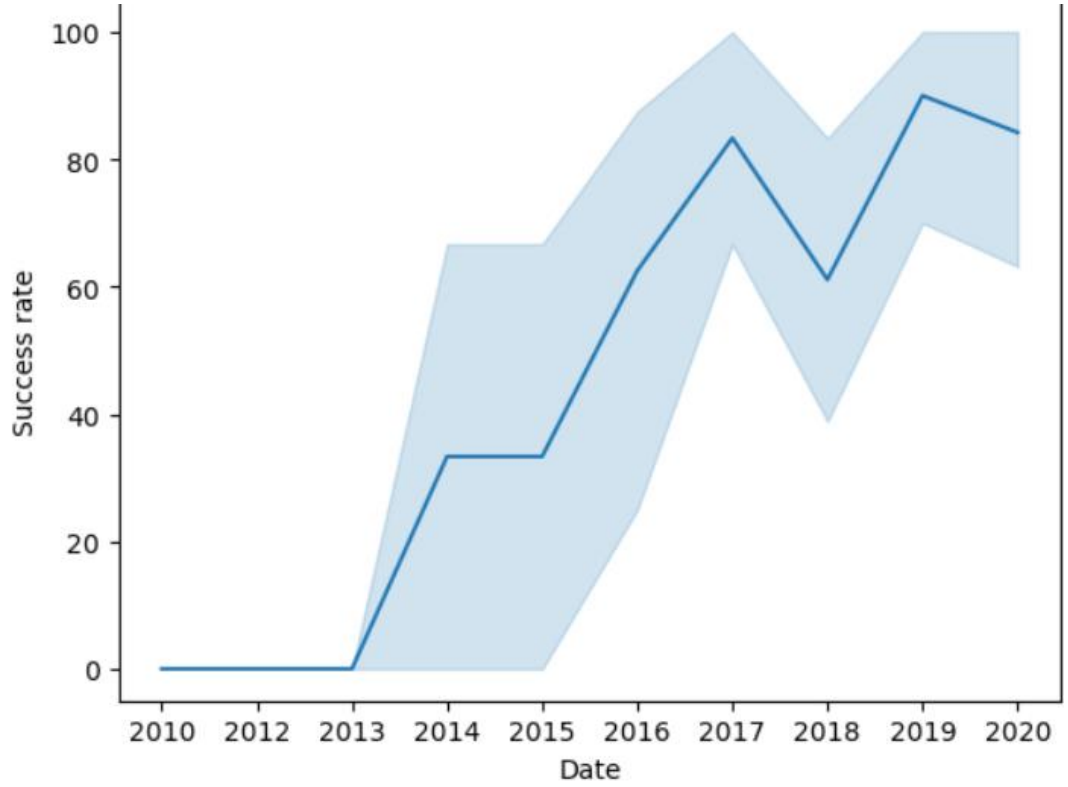


EDA WITH PANDAS AND MATPLOTLIB

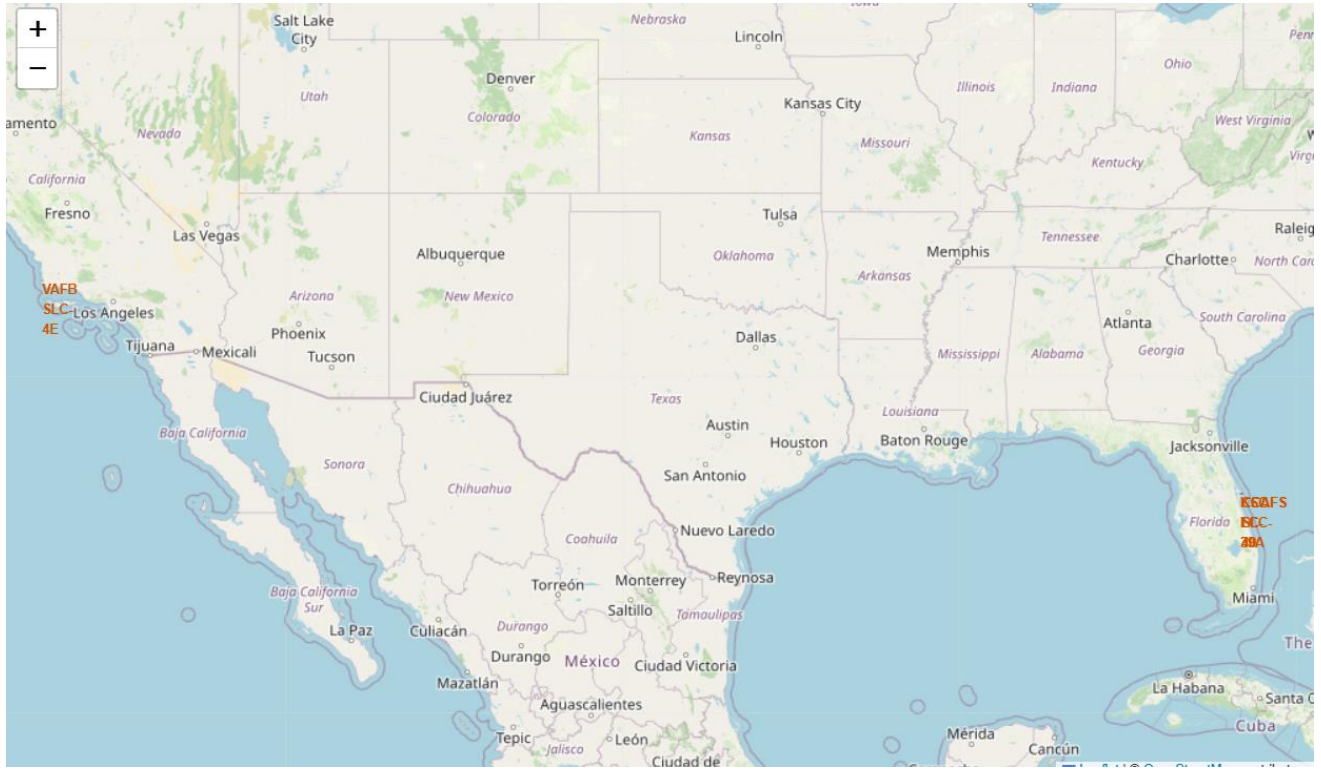
Relationship between Payload Mass and Orbit: orange indicates successful, blue indicates failed

The most successful orbits seem to have low payload mass. Only VLEO has some payload mass values in the upper end of the range

EDA WITH PANDAS AND MATPLOTLIB

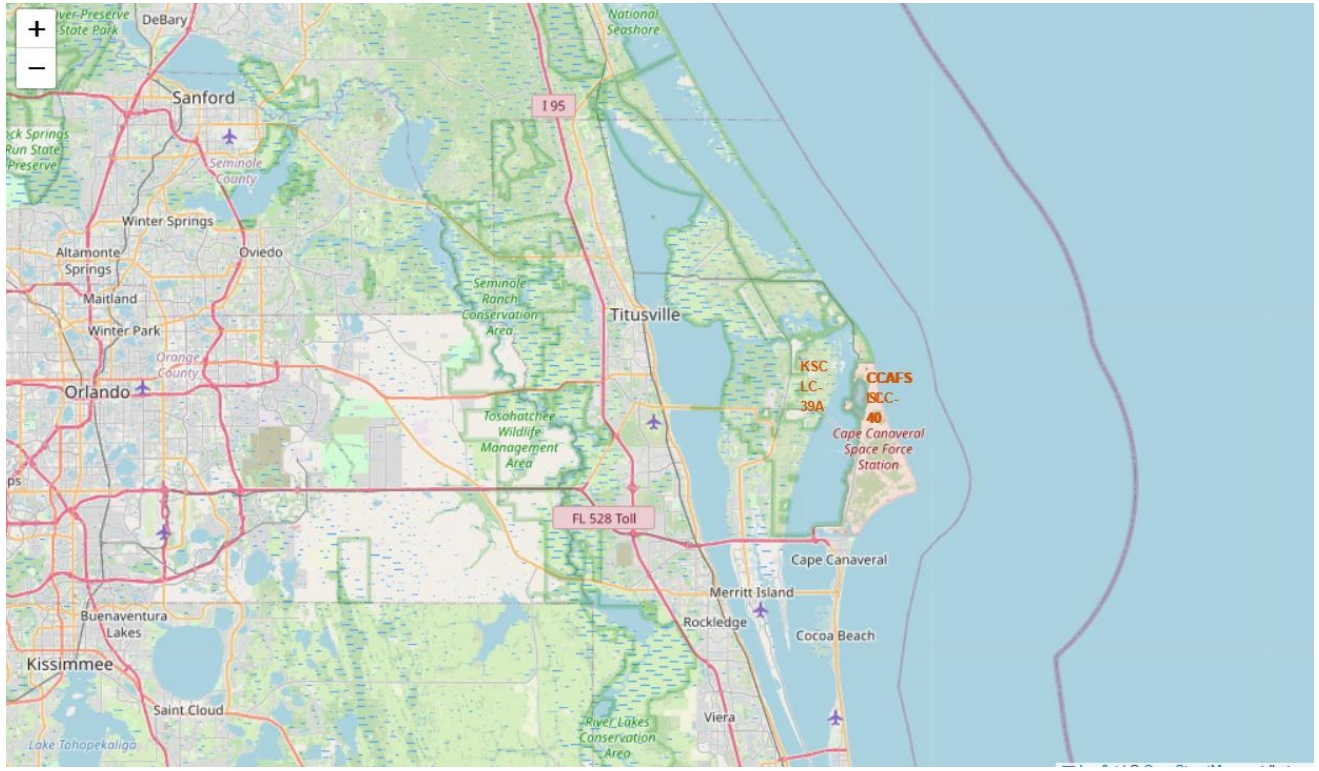


Success yearly trend: The success rate in 2013 kept rising until near 2020



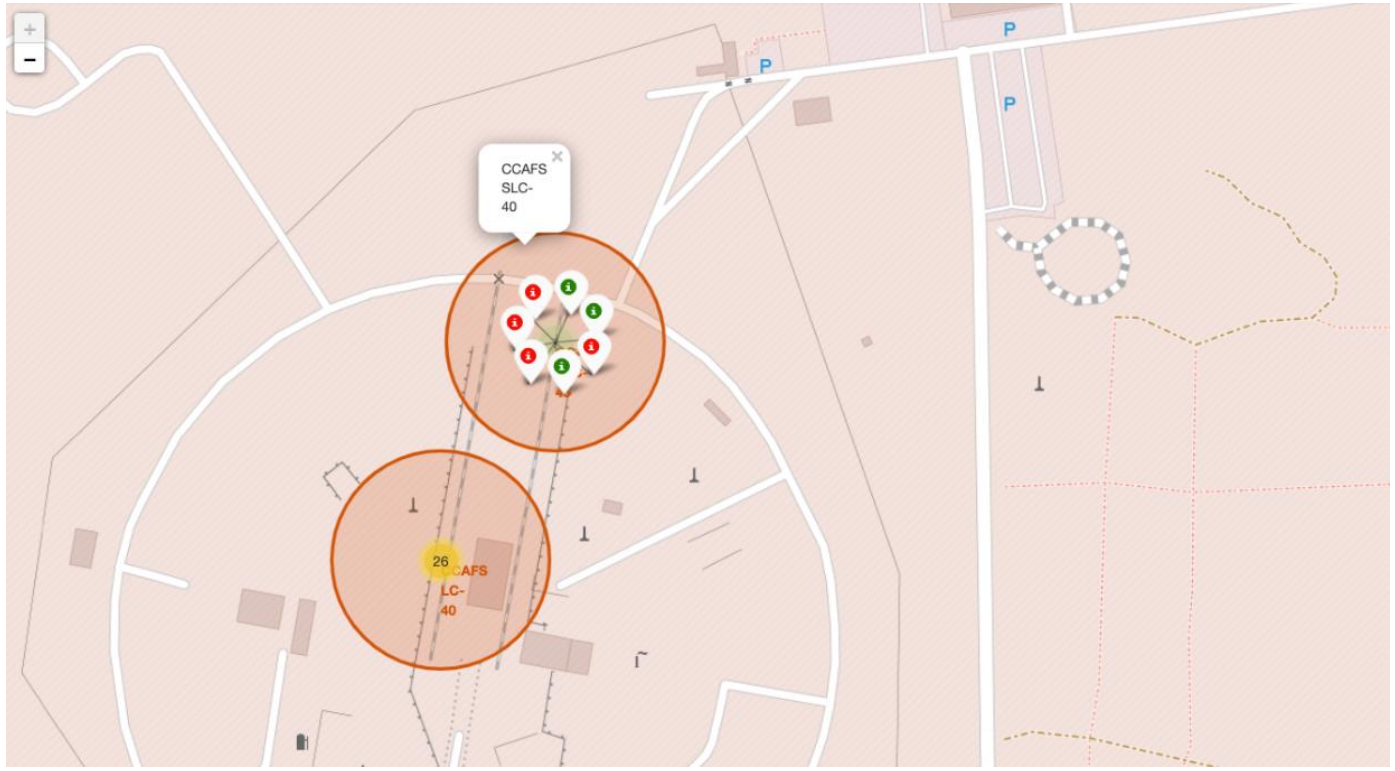
INTERACTIVE VISUAL ANALYTICS WITH FOLIUM

The map shows all launch sites in the US map



INTERACTIVE VISUAL ANALYTICS WITH FOLIUM

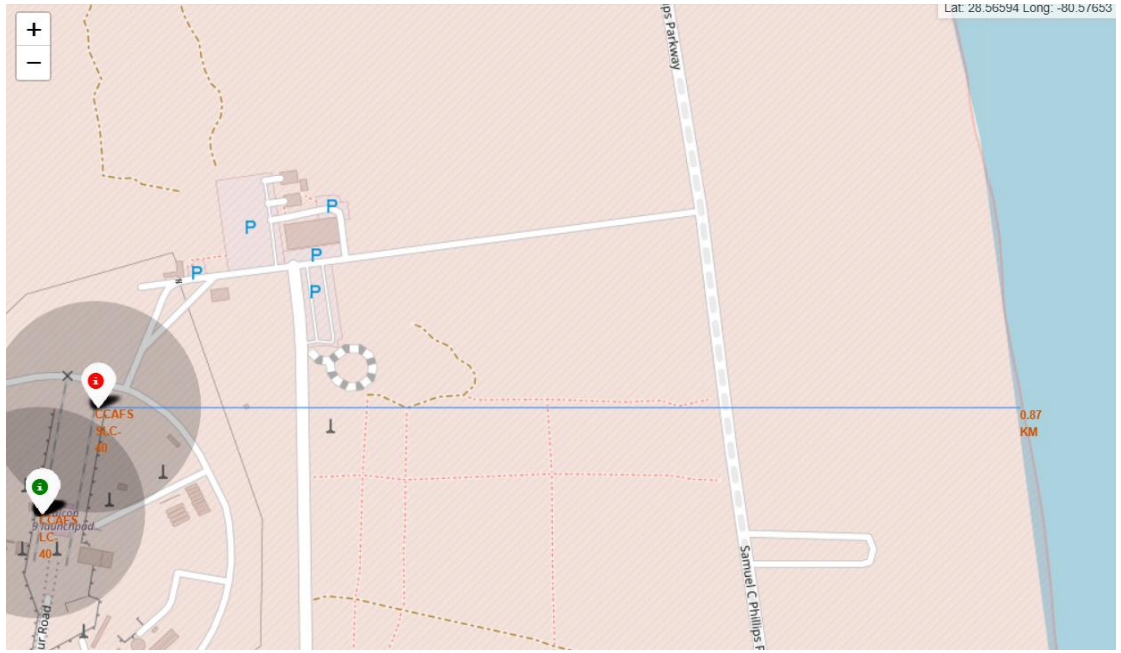
The map shows 2 close launch sites



INTERACTIVE VISUAL ANALYTICS WITH FOLIUM

Clusters on Folium map can be clicked on to display each successful landing (green icon) and failed landing (red icon)

INTERACTIVE VISUAL ANALYTICS WITH FOLIUM

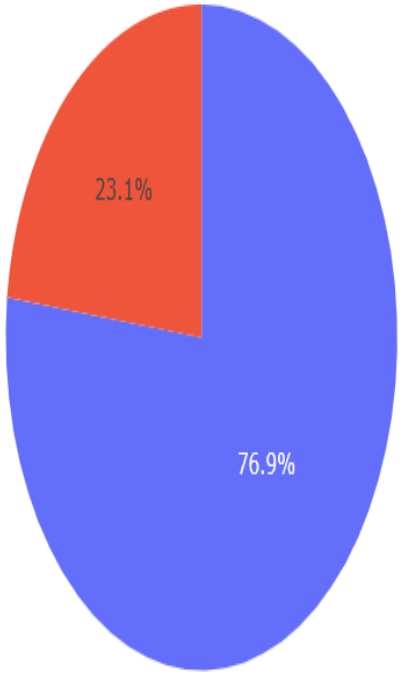


- Looking at CCAFS SLC 40, we can see its proximity to a highway and a railway for human and supply support

INTERACTIVE VISUAL ANALYTICS WITH PLOTLY DASH

This pie chart shows the distribution of successful landings across all sites. We can see that CCAFS LC 40 and CCAFS SLC 40 (possibly the same site) have a total of 41.7 percent of successful landings, equal to KSC LC 39A. If CCAFS SLC 40 and CCAFS LC 40 is the same, then VAFB SLC 4E has the lowest percent of successful landings

INTERACTIVE VISUAL ANALYTICS WITH PLOTLY DASH

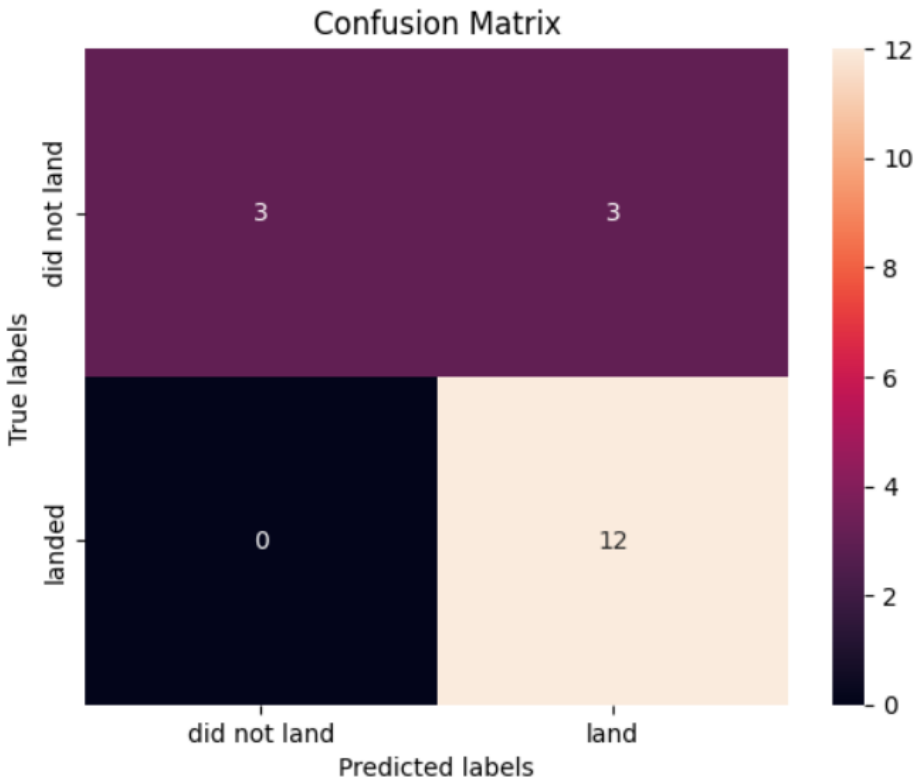


This pie chart shows the distribution of successful landings of the site KSC LC 39A. With nearly 77 percent of successful landing, this is the site with the highest rate of successful landing

INTERACTIVE VISUAL ANALYTICS WITH PLOTLY DASH

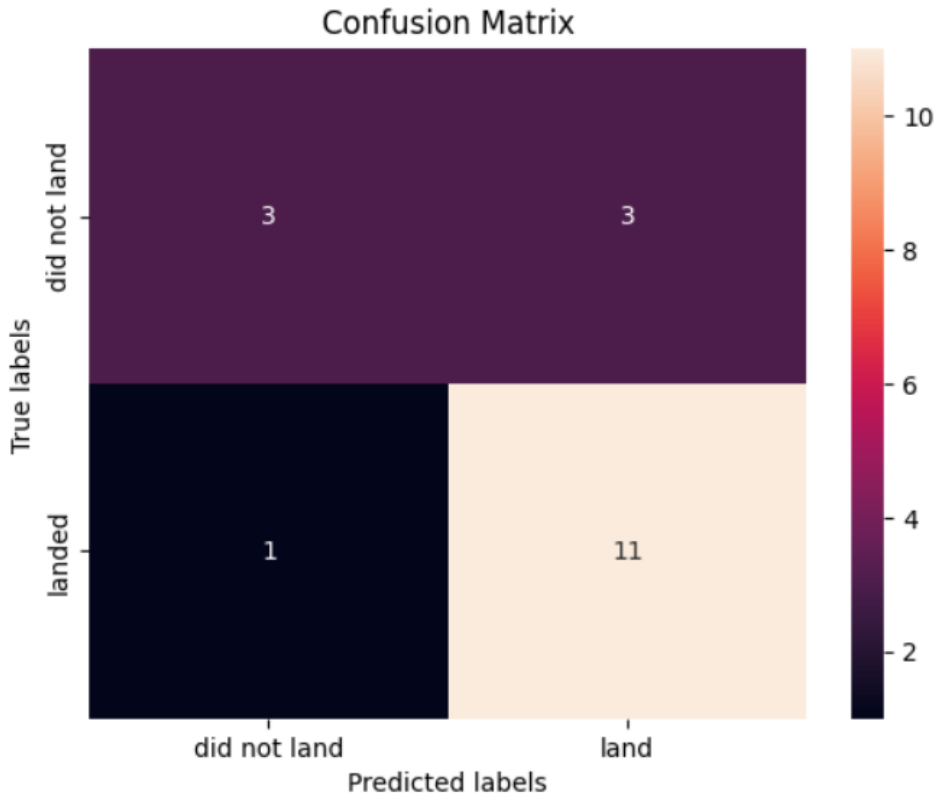
- There is a Payload range in the Plotly Dashboard. However, it is important to note that the range is from 0 to 10000 kg, while the maximum payload mass is fifteen thousands and six hundreds.
- Class 1 means successful landing, while class 0 means failure. The color of points is based on the booster version.

PREDICTIVE ANALYSIS



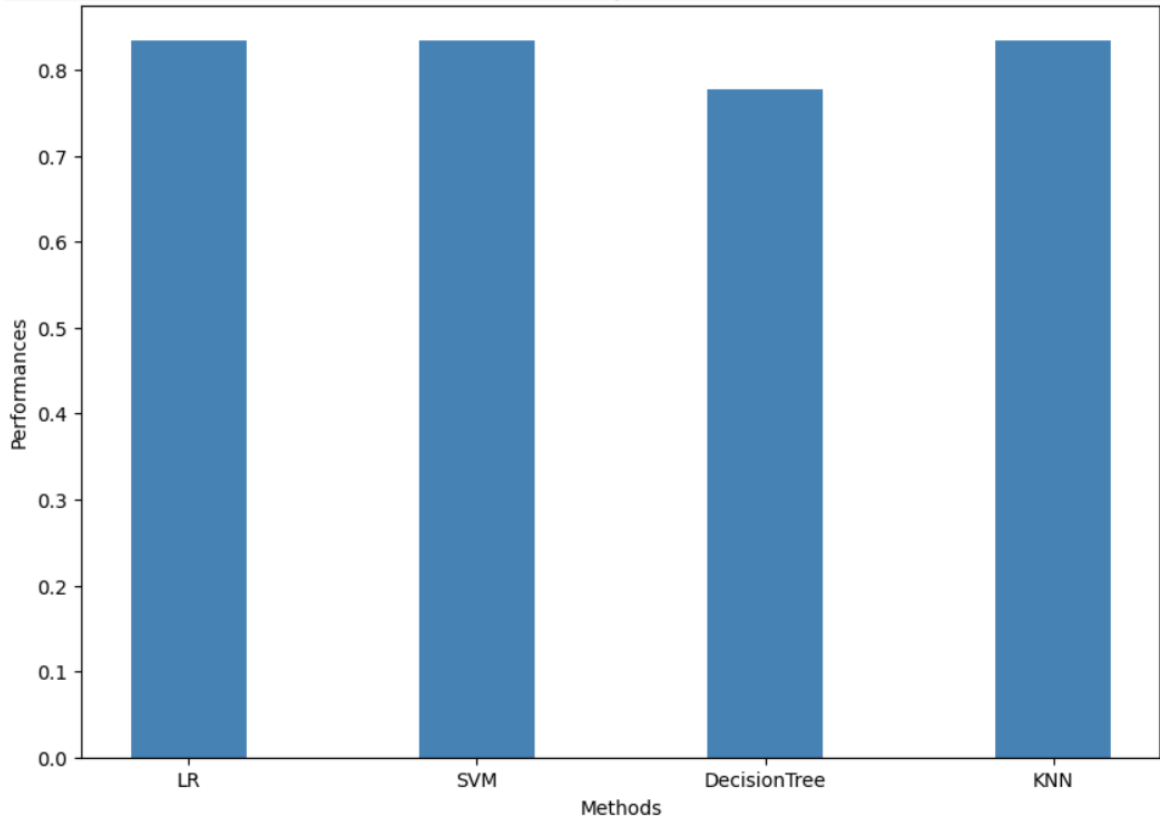
- This is the confusion matrix for the Logistic Regression, Support Vector Machine (SVM) and KNN model. According to the matrix, there were fifteen correctly labeled launches, with 3 one of them being predicted as "did not land" and 12 others being classified as "landed". Only 3 cases were predicted as "landed" but in fact, they did not

PREDICTIVE ANALYSIS



- This is the confusion matrix for the remaining model, Decision Tree. The only difference is that, there were 11 launches correctly labeled as "landed" instead of 12 like we have seen in the previous confusion matrix. Here, the 12th case was initially thought to be "did not land", but it turned out to land successfully.

PREDICTIVE ANALYSIS

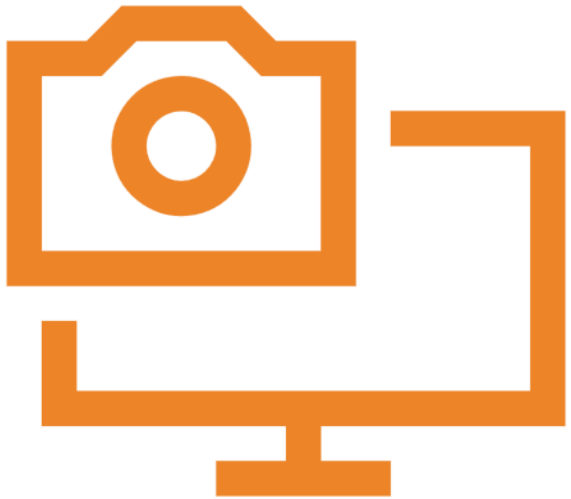


- This bar plot shows the performance of all models trained. We can see that Logistic Regression, SVM and KNN all yield an accuracy score of 83.3, while the Decision Tree only yields an accuracy of 77.7. Although this is such a promising result, we must keep in mind that the sample for the model is just 18, and the model might not perform well if there are more data. To determine the best model, we should train them with a large volume of data

CONCLUSION

- After carrying out all the necessary steps, from loading in the data from SpaceX API and web scraping SpaceX Wikipedia page, creating labels and storing data in a SQL database, creating visualizations to training predictive models, we have a machine learning model yielding an accuracy score of 83 percent.
- SpaceY can use this model to predict with relatively high accuracy whether a launch will have a successful Stage 1 landing before launch to determine whether the launch should be continued or not
- More data should be collected to improve the reliability and applicability of the machine learning model

APPENDIX



- Github URL: UchihaDuongbito/IBM-Data-Science-Professional- (github.com)