IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Igor Shirokov
30 December 2021
https://github.com/SanYattsu/Data_Science_Projects

Outline

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

Executive Summary

During the work we:

- Perform Web scraping and made a get request to the SpaceX API to collect Falcon 9 data.
- Perform Exploratory Data Analysis (EDA), Feature Engineering and determine Training Labels.
- Build an Interactive Map with Folium and Dashboard with Plotly Dash.

The main results:

- The best Hyperparameter for SVM, Classification Trees and Logistic Regression is found.
- The method that performs best in the first stage rocket successful landing prediction using test data is found.

Introduction

In the project, we use a variety of machine learning models to predict the successful landing of Falcon 9 rocket, the optimum method to predict if the first stage will land successfully was founded.

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each. As we know, much of the savings that helps to evaluate the cost comes from the first stage rocket reuse. That’s why the cost of a launch can also be determined based on its successful landing.

The results of the work can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Section 1

Methodology

Methodology

- **Data collection methodology:**

We made a get request to the SpaceX API and performed web scraping to collect Falcon 9 historical launch records from a Wikipedia page.

- **Perform data wrangling**

We downloaded the dataframe, found bad outcomes, and picked the classification variable that represents the outcome of each launch.

Methodology

- **Perform exploratory data analysis (EDA) using visualization and SQL**

We listed the total number of successful and failure mission outcomes and ranked the count of landing outcomes. We visualized and plotted the relationship between different parameters. We observed that the success rate since 2013 kept increasing till 2020.

Methodology

- **Perform interactive visual analytics using Folium and Plotly Dash**

We determined that the launch sites keep certain distance from cities, railways, highways and coastline.

- **Perform predictive analysis using classification models**

We standardized the data, split it into training and test, applied GridSearchCV to found the best parameters for classification models based on calculated accuracy.

Data Collection

01

Request and parse the SpaceX launch data using the GET request

02

Decode the response content as a Json and turn it into a dataframe

03

Use the API again to get information about the launches using the IDs

04

Request Wiki HTML page and extract its content

05

Create a data frame by parsing the HTML with BeautifulSoup

Data Collection – SpaceX API

- Import Libraries and Define Auxiliary Functions
- Request and parse the SpaceX launch data using the GET request
- Do Data Wrangling
- Dealing with Missing Values
- Export dataframe to a CSV

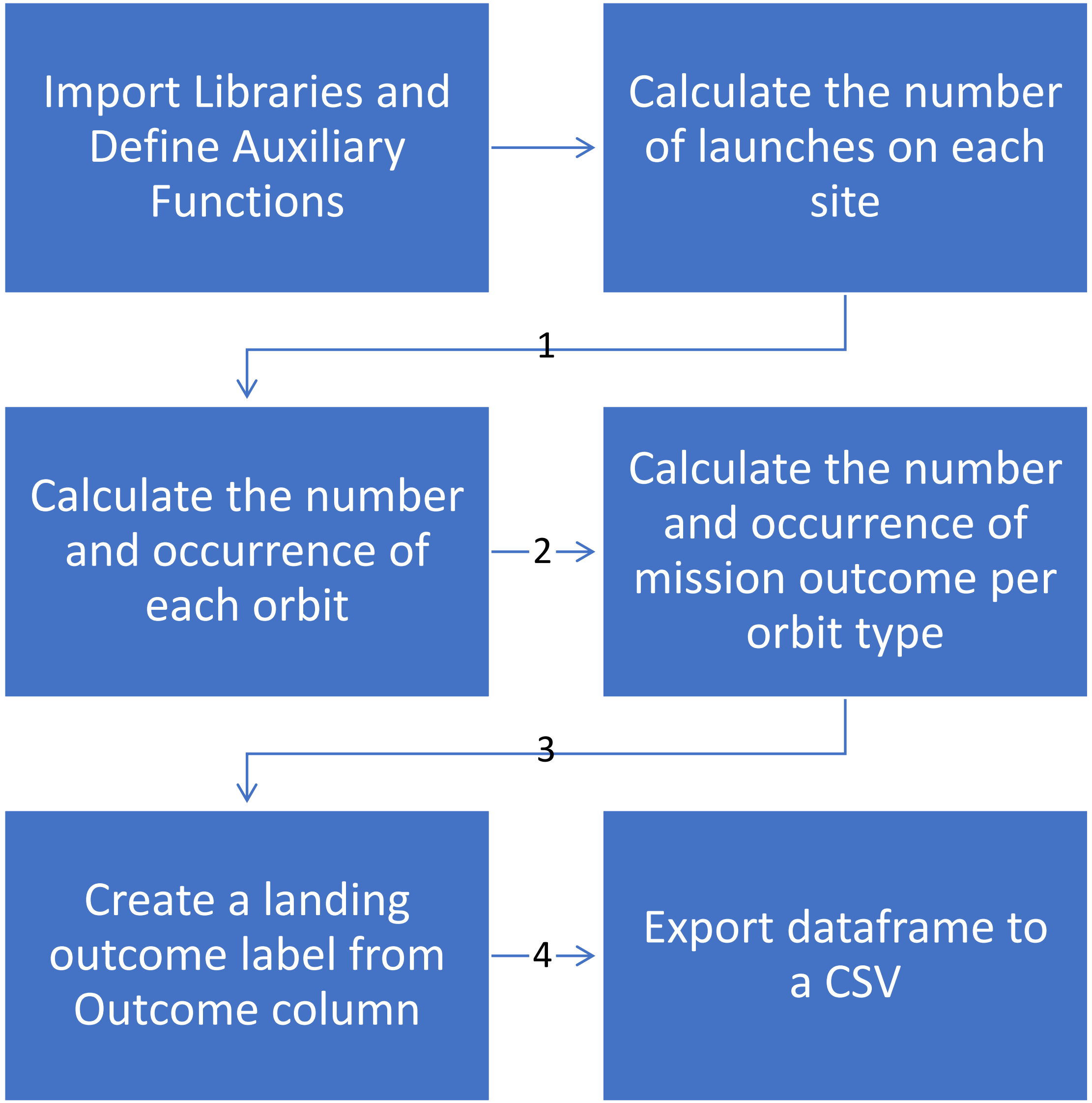
https://github.com/SanYattsu/coursera_projects/blob/master/1%20-%20Collecting%20the%20data.ipynb

Data Collection - Scraping

Request	Request the Falcon9 Launch Wiki page from its URL
Extract	Extract all column/variable names from the HTML table header
Create	Create a data frame by parsing the launch HTML tables
Export	Export dataframe to a CSV

https://github.com/SanYattsu/coursera_projects/blob/master/1.1%20-%20Web%20scraping.ipynb

Data Wrangling



https://github.com/SanYattsu/coursera_projects/blob/master/2%20-%20Data%20wrangling.ipynb

EDA with Data Visualization

In the work were plotted comparison charts to visualize the relationship between parameters: FlightNumber vs. PayloadMass, Flight Number and Launch Site, Payload and Launch Site and etc.

We found relationship between success rate of each orbit type and observe that the success rate since 2013 kept increasing till 2020.

I also took extra work to find out should we use all features or the ones that highly corelate with the "Class". The acquired data used in modeling. I found that its useless to minimalize the number of futures (accuracy stay the same).

EDA with SQL

- Display the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA’
- Display the total payload mass carried by boosters launched by NASA (CRS)
- Display average payload mass carried by booster version F9 v1.1
- List the date when the first successful landing outcome in ground pad was achieved.
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass.
- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
- 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

Build an Interactive Map with Folium

- The launch success rate may depend on the location and proximities of a launch site, i.e., the initial position of rocket trajectories. In the work, we discover some of the factors by analyzing the existing launch site locations.
- Objects as a MarkerCluster, markers, circles and lines was created. The MarkerCluster helped to group markers when we zoom out the map. And other objects helped to pinpoint sites and proximity locations.
- We also added Lat. and Lon. to the map’s top to found out proximity locations.

Build a Dashboard with Plotly Dash

We added:

- the dropdown list to choose the Launch site,
- the pie chart to see Total Success Launches by or for the Site,
- the slider to filtered payload range,
- the scatter plot to show the Correlation between Payload and Success for different Booster Versions.

Predictive Analysis (Classification)

Load	Load the dataframe
Standardize	Standardize the data in X
Split	Split into training data and test data
Create	Create model object
Use	Use GridSearchCV to find the best parameters for the model
Calculate	Calculate the accuracy on the test data
Plot	Plot the confusion matrix
Repeat	Repeat step 4-7 for 3 other models
Find	Find the method performs the best and have the highest score

In the project, we used 4 different classification methods:

- SVM,
- k nearest neighbors,
- Classification Trees,
- Logistic Regression.

Results

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

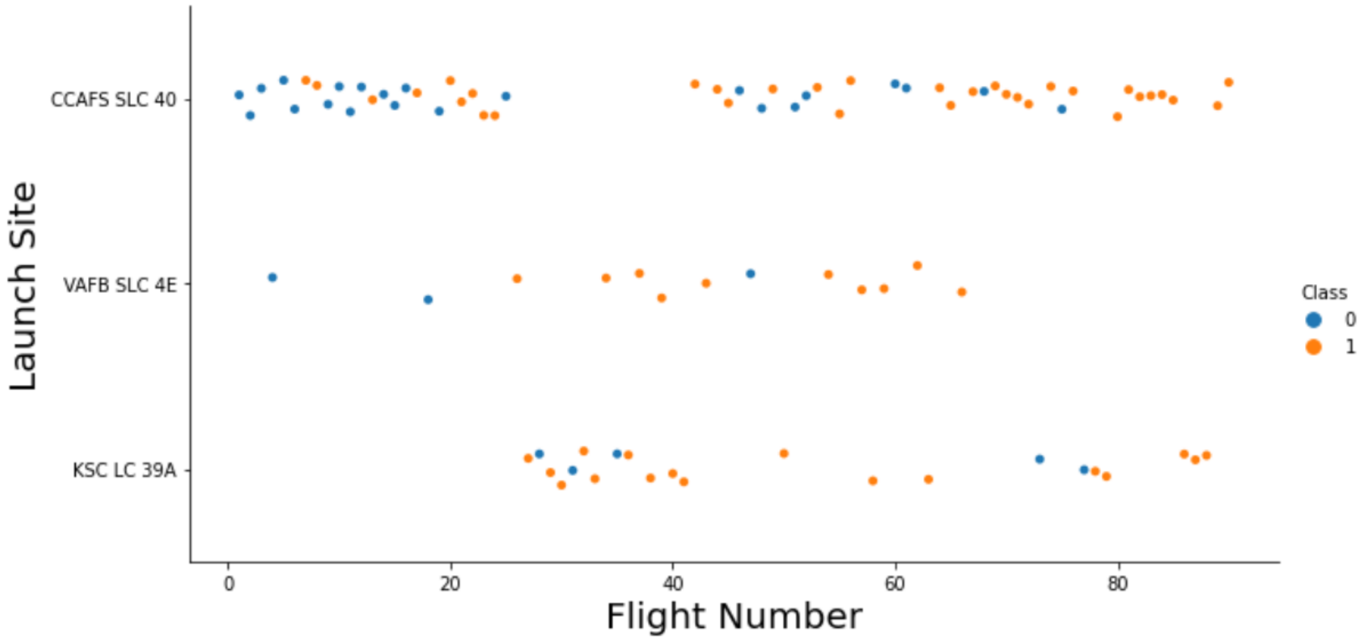
Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

From the fig. is seen that:

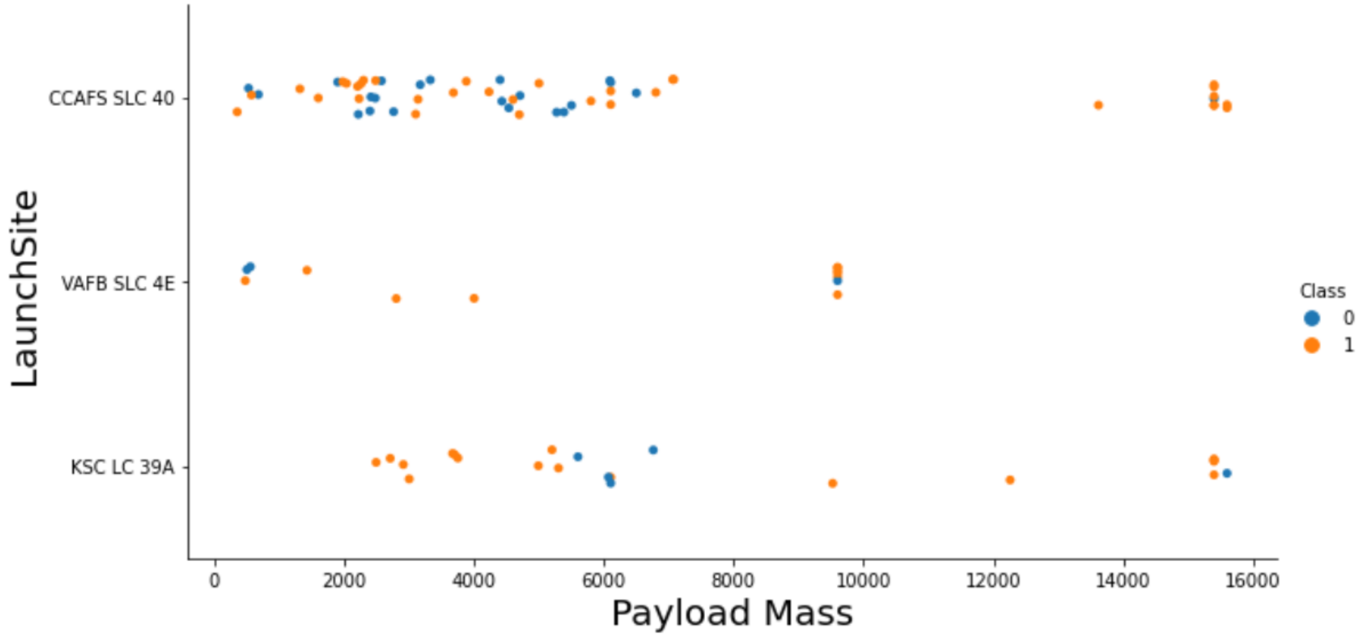
- As the flight number increases, the first stage is more likely to land successfully.
- CCAFS SLC 40 have more lunches, but worser success ratio compare to 2 other sites.
- After flight number ~64 there wasn't any new lunches from VAFB SLC 4E.



Payload vs. Launch Site

From the fig. is seen that:

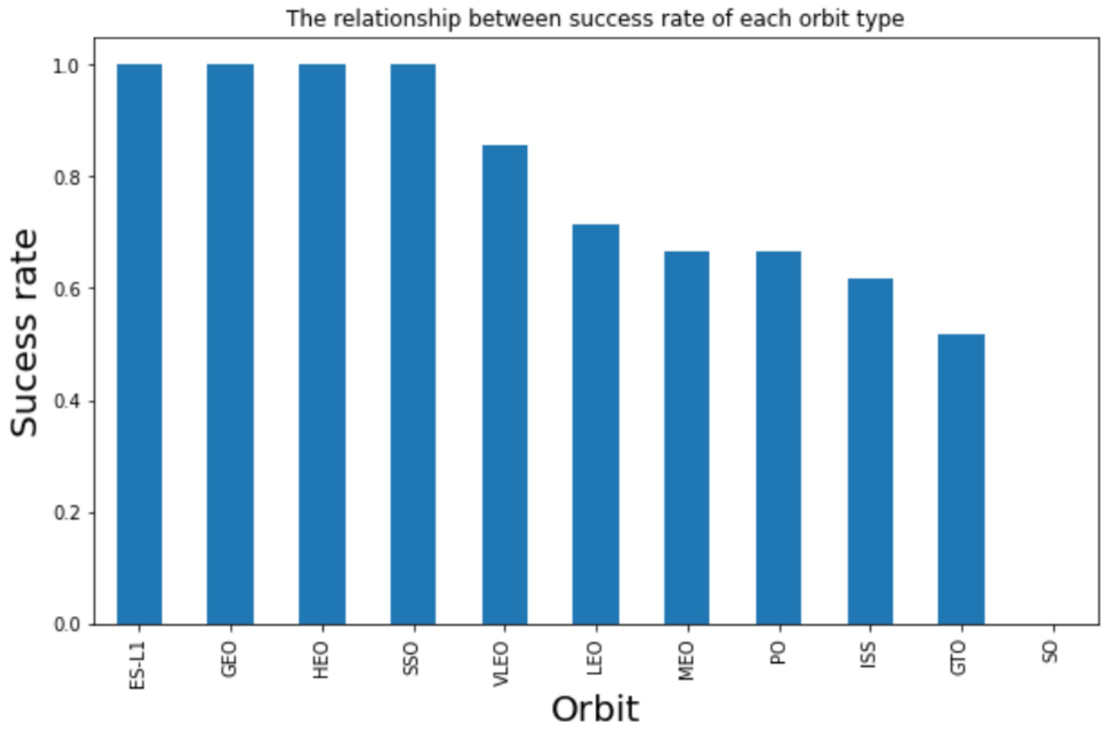
- VAFB-SLC launch site there are no rockets launched for heavy payload mass.
- Success ratio is higher if payload greater than 7000.
- KSC LC 39A have more successful launches.



Success Rate vs. Orbit Type

From the fig. is seen that:

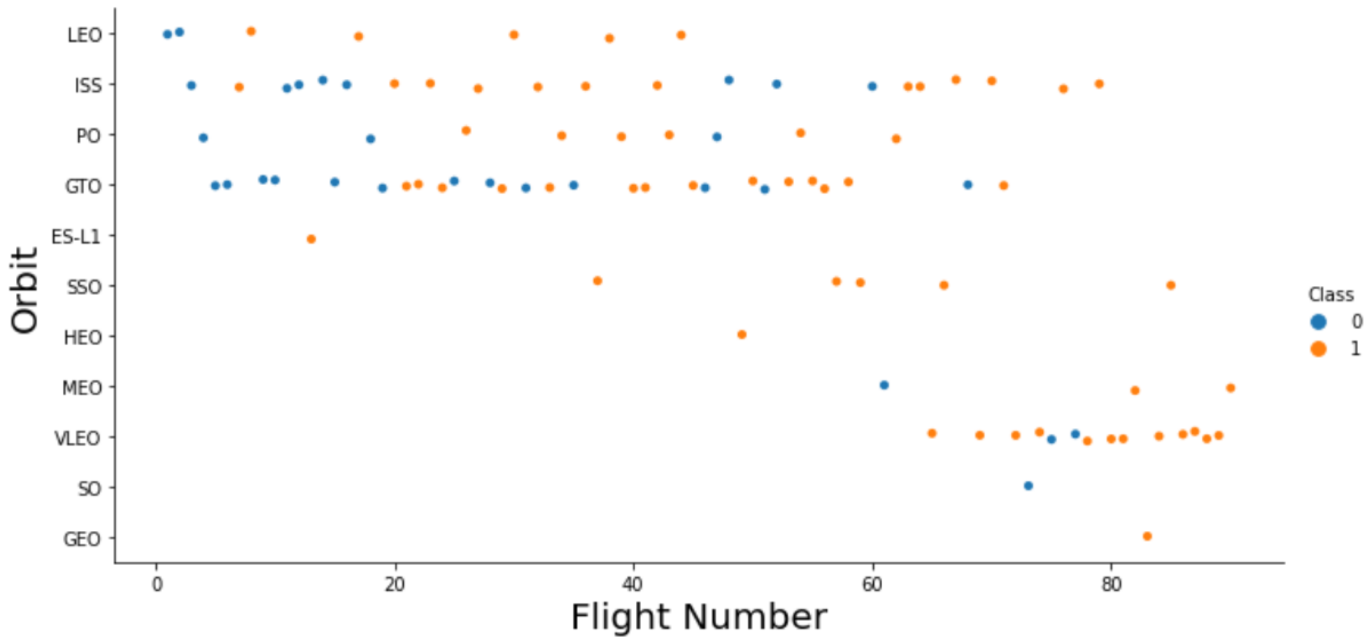
- No success launches for SO.
- Success ratio for orbit types ES-L1, GEO, HEO and SSO are 100%.



Flight Number vs. Orbit Type

From the fig. is seen that:

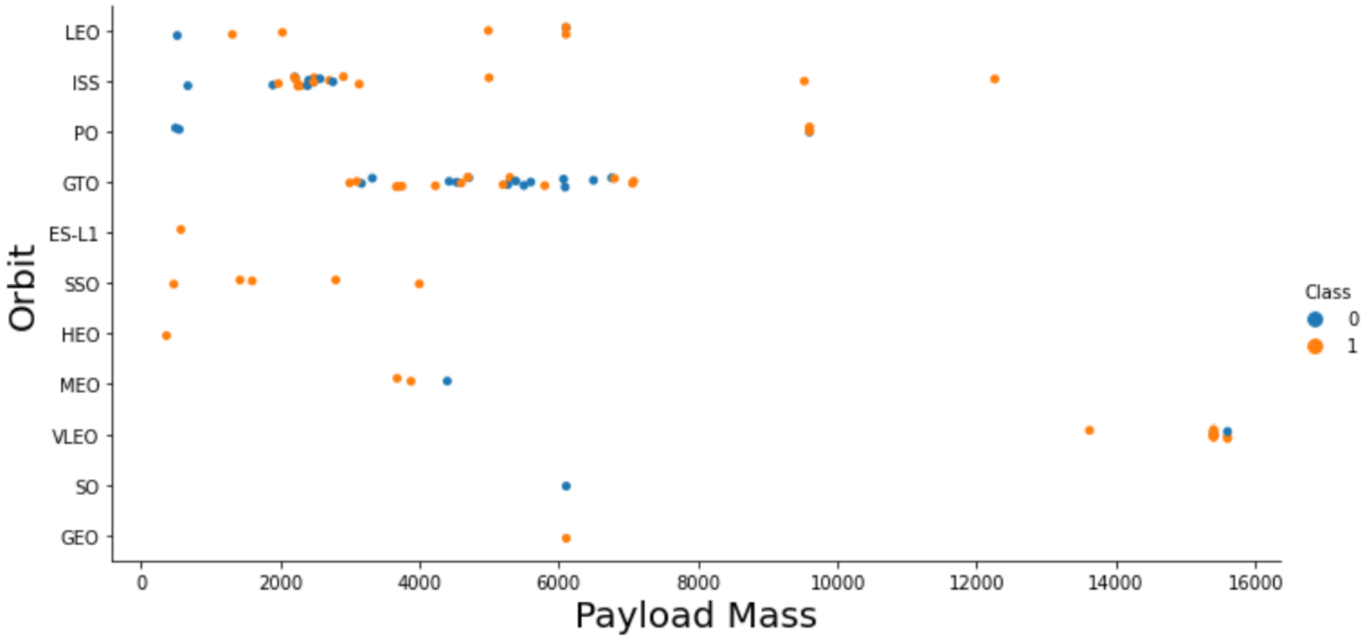
- More than half of orbit types wasn’t used till Flight Number become 60. **And it’s probably related to increase in success ratio**.



Payload vs. Orbit Type

From the fig. is seen that:

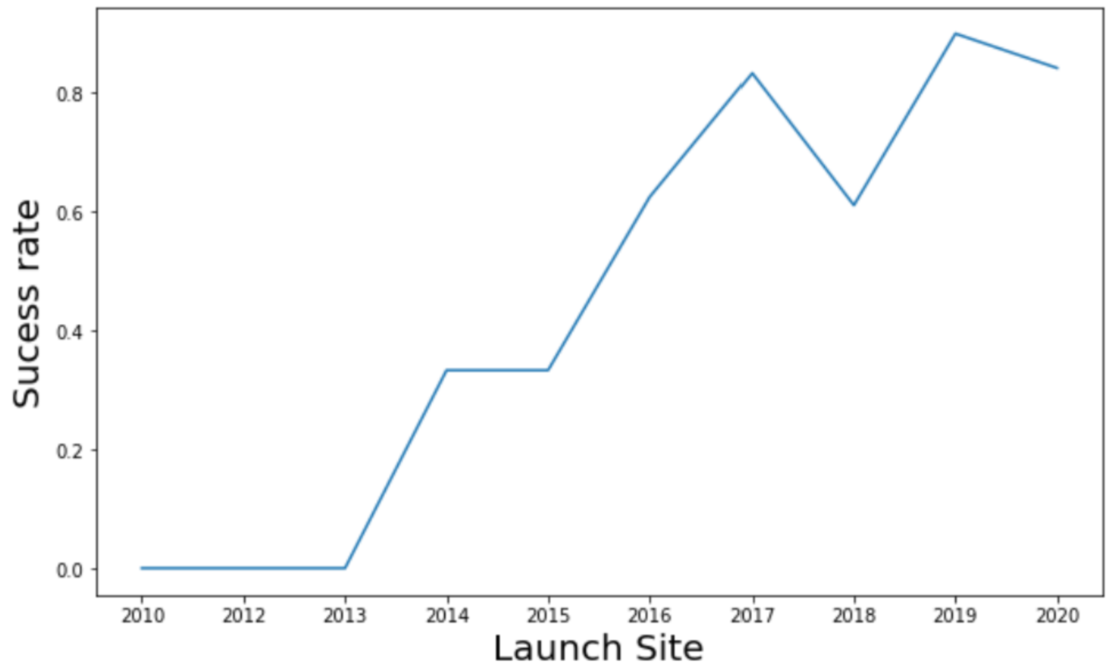
- With heavy payloads the successful landing rate are more for PO, LEO and ISS.
- ISS is mostly used for payload in range 2000-3000.
- GTO is mostly used for payload in range 3000-7000.
- SSO had the highest success ratio.



Launch Success Yearly Trend

From the fig. is seen that:

- the success rate was 0 till 2013.
- the success rate kept increasing since 2013 till 2020.
- 1 out of 10 launches would always end in failure.



All Launch Site Names

Used "distinct" to select unique values.

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

[4]:
```
%sql select distinct LAUNCH_SITE from SPACEX
```

```
 * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999ed
SSL
Done.
```

[4]:

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

Launch Site Names Begin with 'CCA'

Used “limit” and “like” with %.

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

```
[30]: %sql select * from SPACEX where upper(LAUNCH_SITE) LIKE 'CCA%' limit 5

 * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cm
SSL
Done.
```

[30]:

DATE	time__utc_	booster_version	launch_site	payload	payloa
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	

Total Payload Mass

No comments ☺

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

```
[6]: %sql select sum(payload_mass__kg_) as """Total Payload Mass""" from SPACEX \
     where SPACEX.customer = 'NASA (CRS)'

      * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqn
     SSL
     Done.
```

[6]:

Total Payload Mass
45596

Total Payload Mass

Used sum() method.

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

```
[6]: %sql select sum(payload_mass__kg_) as """Total Payload Mass""" from SPACEX \
     where SPACEX.customer = 'NASA (CRS)'

      * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqn
     SSL
     Done.
```

[6]:

Total Payload Mass
45596

Average Payload Mass by F9 v1.1

Used avg() method.

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

```
[23]: %sql select avg(payload_mass__kg_) as """Average Payload Mass""" from SPACEX \
      where SPACEX.booster_version like 'F9 v1.1%'

       * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk
      SSL
      Done.
```

[23]:

Average Payload Mass
2534

First Successful Ground Landing Date

Used the Hint ☺

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

Hint:Use min function

```
[8]: %sql select min(date) as """Date""" from SPACEX \
     where SPACEX.landing__outcome = 'Success (ground pad)'
```

```
 * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nq
SSL
Done.
```

[8]:

Date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Used "between … and …" for a range.

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

```
[9]: %sql select distinct booster_version as """Boosters """ from SPACEX \
where SPACEX.landing__outcome = 'Success (drone ship)' and SPACEX.payload_mass__kg_ between 4000 and 6000
```

```
 * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:
SSL
Done.
```

[9]:

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

Total Number of Successful and Failure Mission Outcomes

Used Group by.

List the total number of successful and failure mission outcomes

```
[10]: %sql select mission_outcome, count(mission_outcome) as count from SPACEX \
      group by mission_outcome

 * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0n
SSL
Done.
```

[10]:

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

Boosters Carried Maximum Payload

Used subquery.

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

[11]:
```
%sql select distinct booster_version from SPACEX \
where SPACEX.payload_mass__kg_  = (select max(payload_mass__kg_) from SPACEX)
```

```
 * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databa
SSL
Done.
```

[11]:

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

2015 Launch Records

Used year() function.

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
[12]: %sql select landing__outcome, booster_version, launch_site from SPACEX \
      where SPACEX.landing__outcome = 'Failure (drone ship)' and year (SPACEX.DATE) = 2015

 * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.a
SSL
Done.
```

[12]:

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

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

Used order by desc for descending order.

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

[13]:
```
%%sql
select landing__outcome, count(landing__outcome) as count from SPACEX
where date(SPACEX.DATE) between '2010-06-04' and '2017-03-20'
group by landing__outcome
order by count desc
```

```
 * ibm_db_sa://qsn83869:***@b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:31249/BLUDB;s
SSL
Done.
```

[13]:

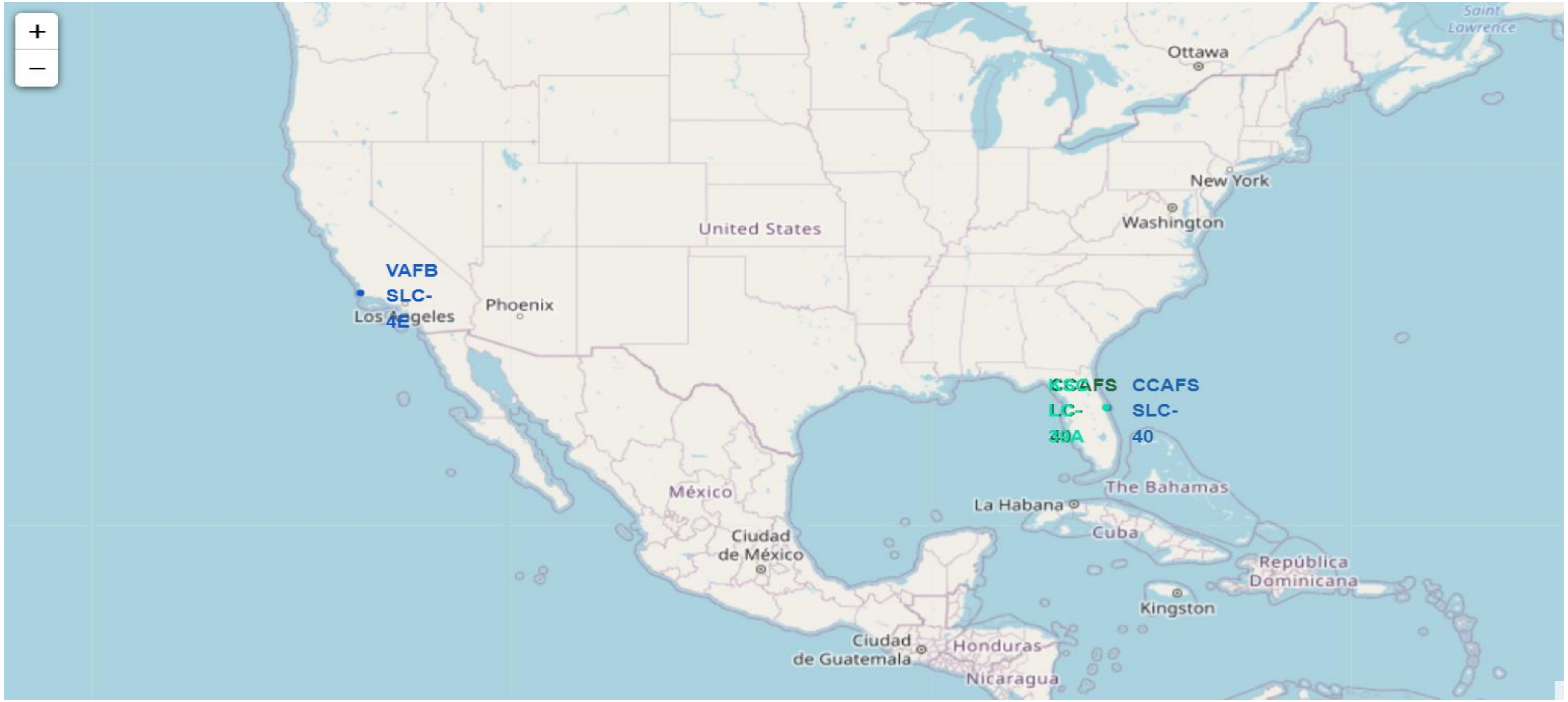
landing__outcome	COUNT
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Section 4

Launch Sites Proximities Analysis

Launch sites locations

Here we can see 4 launch sites. They all are very close to the coastline.

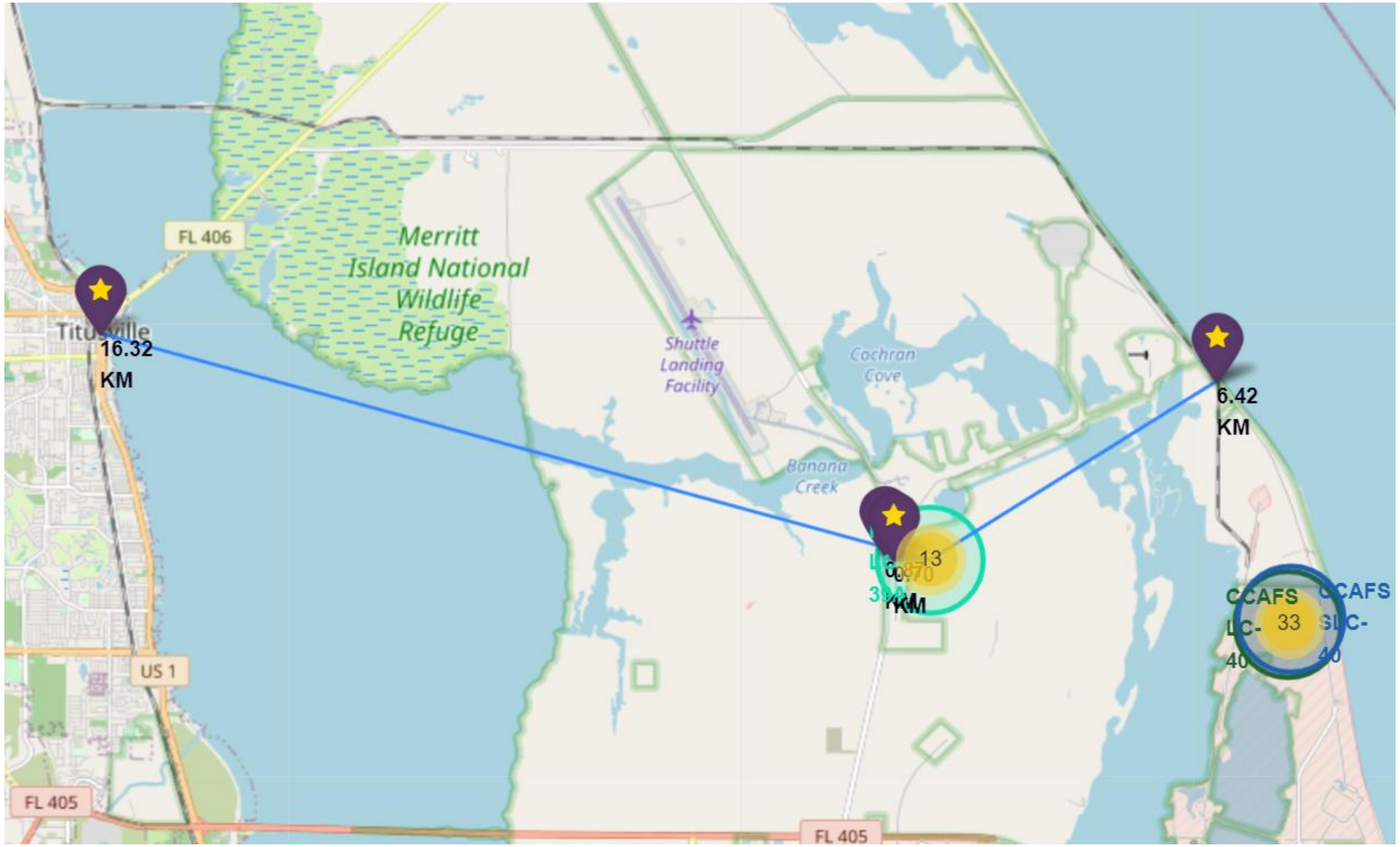


Success/failed launches for each site

CCAFS LC-40 have more launches, but KSC LC-39F have more successful launches.

The distance to the nearest sights

All sites located very close to the coastline, railways and highways, while the distance to the nearest town exceed 15 km.



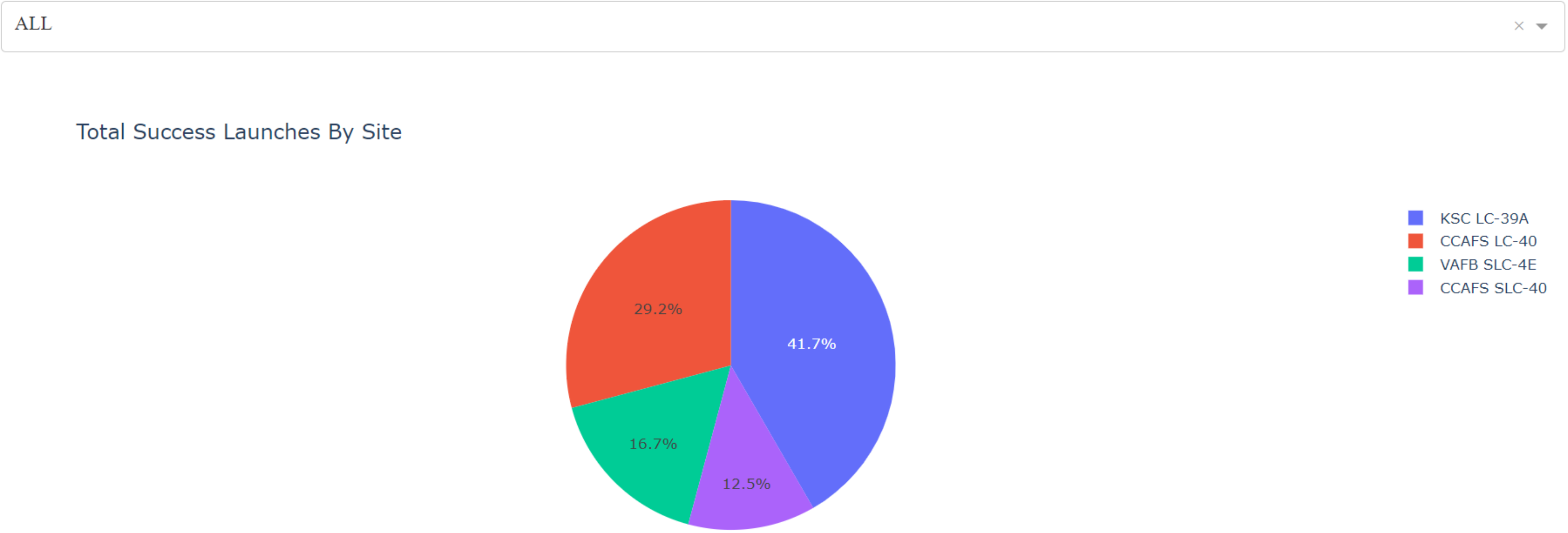
Section 5

Build a Dashboard with Plotly Dash

Total Success Launches By Site

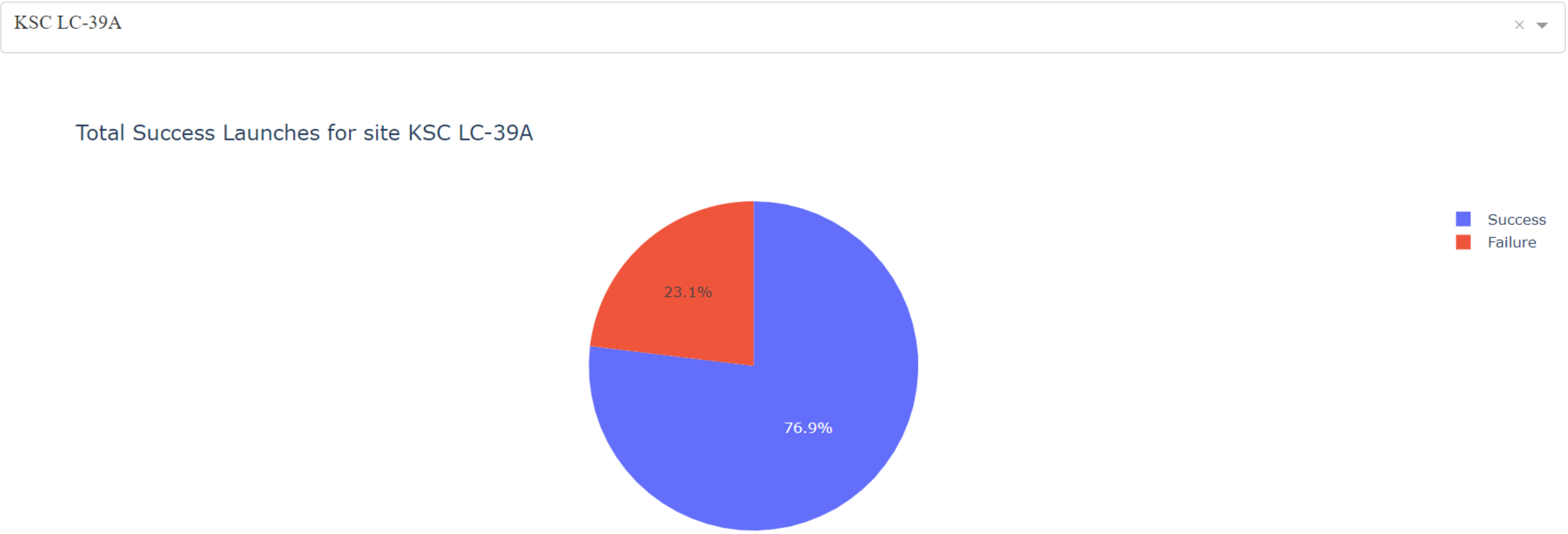
KSC LC-39A have more successful launches than any other site.

CCAFS SLC-40 – the lest successful.



Total Success Launches for site KSC LC-39A

¾ launches are successful.

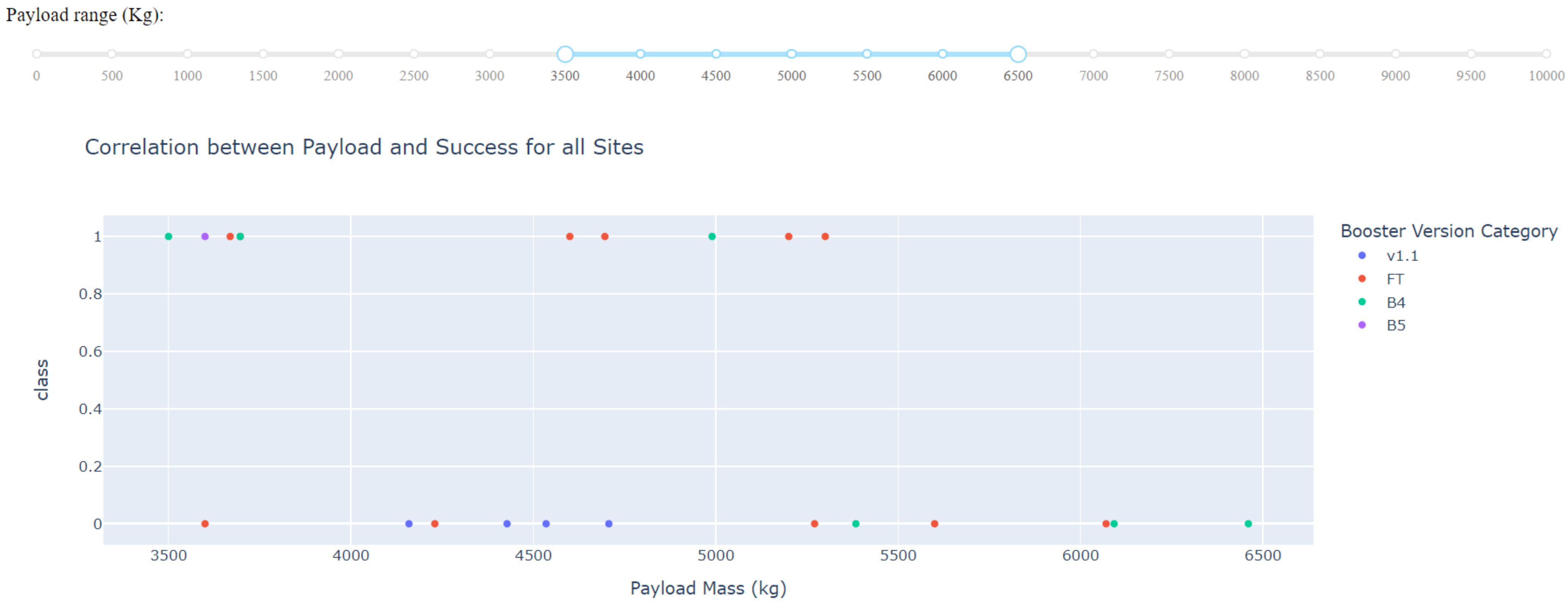


Correlation between Payload and Success for all Sites

Booster Version B5 and FT have the highest success rate.

Correlation between Payload and Success for all Sites

Boosters v.1.0 don’t used in this Payload range.

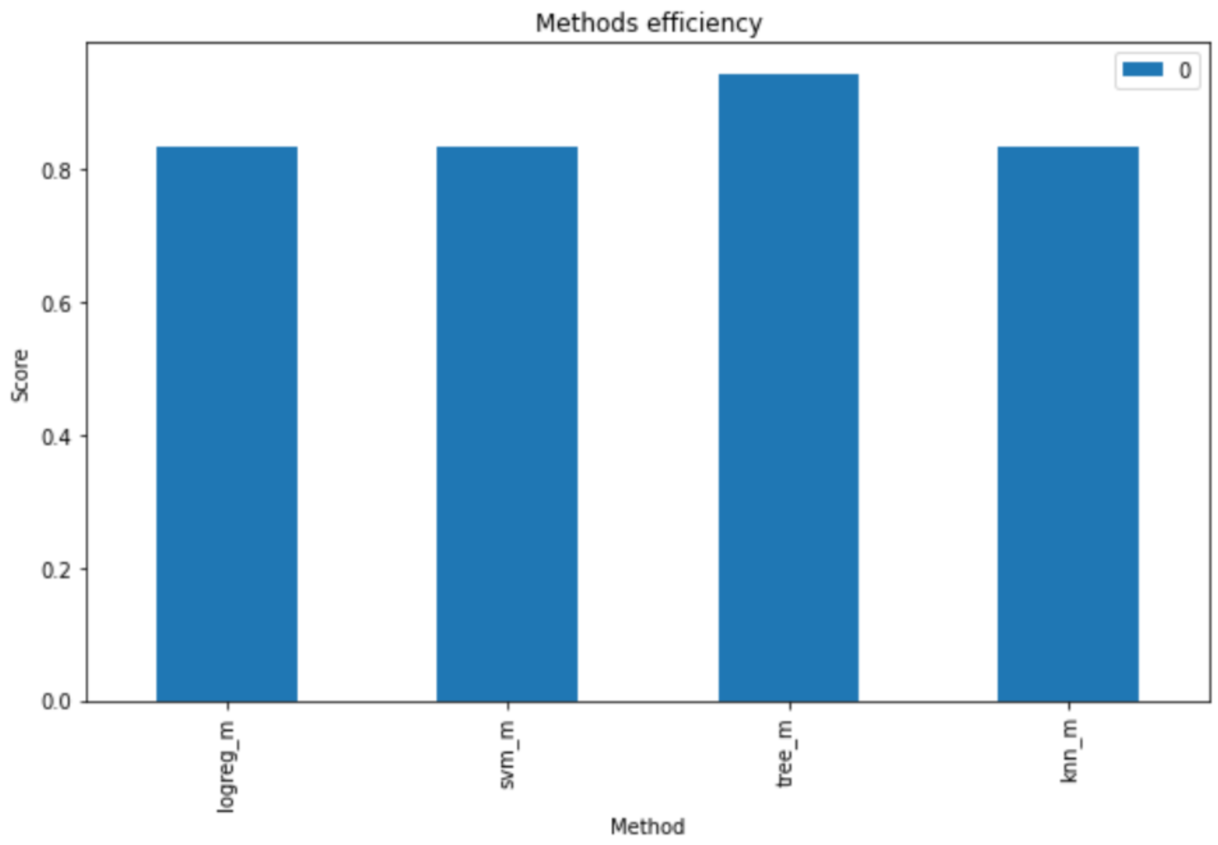


Section 6

Predictive Analysis (Classification)

Classification Accuracy

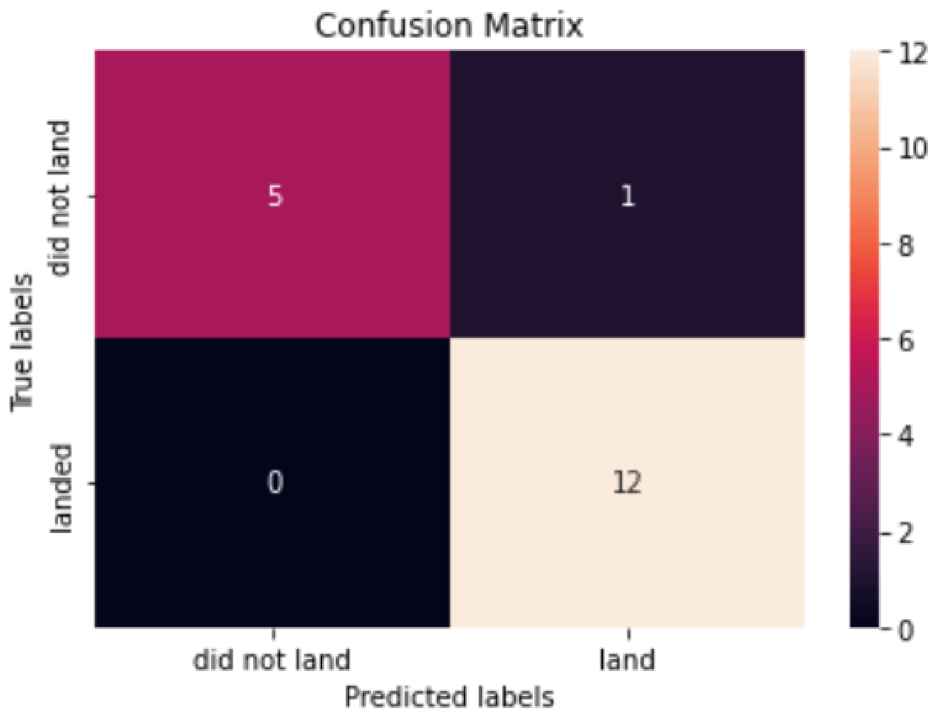
The method performs best is Decision Tree Classifier, with the score 0.94!



Confusion Matrix

The confusion matrix for the best performing model shows very good results. It have only one mistake where predicted success ended with failure.

So high accuracy was achieved by chance, normally all 4 model have the same score 0.83.



Conclusions

- The method that performs best in successful landing prediction is Decision Tree Classifier (score 0.94).
- All 4 Classifier models are good at predicting the result.
- Successful ratio significantly increased and stopped at 0.9.
- SSO orbit type have the best success ratio but used only for payload below 4500.
- ES-L1, SSO, HEO, MEO, VLEO, SO, GEO orbit types are used only when the rocket became more reliable.
- Some interesting info about launch sites location was acquired.

Appendix

Here I will provide some useful
code snippets for people who are curious.

lambda + explicit copy() to avoid the warring

```
data_falcon9 =
launch_df[launch_df['BoosterVersion'].map
(lambda x : x != "Falcon 1")].copy()
```

Its better to use a["title"] + rstrip()

```
# Customer
if row[6].a is not None:
	customer = row[6].a["title"]
else:
	customer = "other“

# Launch outcome - Delete /n
launch_outcome = row[7].contents[0].rstrip()
```

Use only features that are correlated with Class

```
features = df[['Class','FlightNumber', 'PayloadMass', 'Orbit', 'LaunchSite', 'Flights', 'GridFins', 'Reused', 'Legs', 'LandingPad',
'Block', 'ReusedCount', 'Serial']]

features_one_hot = pd.get_dummies(data=features, columns=["Orbit", "LaunchSite", "LandingPad", "Serial"], dtype=float)

features_one_hot = features_one_hot.astype(float)


features_one_hot.corr()["Class"]

correlation = features_one_hot.corr()["Class"]

correlation = pd.DataFrame(correlation)

correlation.drop(labels=["Class"], axis=0,inplace=True)

correlation.reset_index(inplace=True)


new_columns = []

for index, row in correlation.iterrows():

    if (row["Class"]>0.3) | (row["Class"]<-0.3):

        new_columns.append(row["index"])


features_one_hot = features_one_hot[new_columns]
```

Options for Dropdown and marks for RangeSlider

```
options_dropdown=[]
Launch_Sites = pd.unique(spacex_df["Launch Site"]).tolist()
Launch_Sites.append("ALL")

for i in range(len(Launch_Sites)):
    options_dropdown.append({'label': Launch_Sites[i], 'value': Launch_Sites[i]})



marks_rs = {}
marks_arr = list(map(str,np.arange(0,10001,500)))

for mark in marks_arr:
    marks_rs.update({mark : mark})
```

Thank you!