SpaceX Falcon-9 First Stage Landing Predictions

ZHEREN LI

1. Executive Summary
2. Introduction
3. Methods
4. Results
5. Conclusion

Executive Summary

- Data collection
- Data Wrangling
- Exploratory Data Analysis
- EDA with Visualization
- Interactive Map with Folium
- Predictive Analysis with Machine Learning

Introduction

- Primary goals of SpaceX Falcon9

  - Reliable Access to Space
  - Reusable Rocketry
  - Commercial Satellite Launches
  - International Space Station Resupply
  - Crewed Spaceflight

Source: https://i0.wp.com/spacenews.com/wp-content/uploads/2020/10/40126461411_a6e49a61f2_k.jpg?fit=2048%2C1365&ssl=1

Data Collection

	Flight No.	Date	Time	Version Booster	Launch Site	Payload	Payload mass	Orbit	Customer	Launch outcome	Booster landing
0	1	4 June 2010	18:45	F9 v1.0B0003.1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	Failure
1	2	8 December 2010	15:43	F9 v1.0B0004.1	CCAFS	Dragon	0	LEO	NASA	Success	Failure
2	3	22 May 2012	07:44	F9 v1.0B0005.1	CCAFS	Dragon	525 kg	LEO	NASA	Success	No attempt\n
3	4	8 October 2012	00:35	F9 v1.0B0006.1	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	No attempt
4	5	1 March 2013	15:10	F9 v1.0B0007.1	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	No attempt\n
...	...	...	...	...	...	...	...	...	...	...	...
116	117	9 May 2021	06:42	F9 B5B1051.10	CCSFS	Starlink	15,600 kg	LEO	SpaceX	Success\n	Success
117	118	15 May 2021	22:56	F9 B5B1058.8	KSC	Starlink	~14,000 kg	LEO	SpaceX	Success\n	Success
118	119	26 May 2021	18:59	F9 B5B1063.2	CCSFS	Starlink	15,600 kg	LEO	SpaceX	Success\n	Success
119	120	3 June 2021	17:29	F9 B5B1067.1	KSC	SpaceX CRS-22	3,328 kg	LEO	NASA	Success\n	Success
120	121	6 June 2021	04:26	F9 B5	CCSFS	SXM-8	7,000 kg	GTO	Sirius XM	Success\n	Success

- Data collected from webscrapping via Request and Beautifulsoup
- Data frame was cleaned and filtered to only include Falcon 9 launches

Data Wrangling

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

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

We can use the following line of code to determine the success rate:

```
df["Class"].mean()
```

```
0.6666666666666666
```

- Number of launches
- Number of occurrences of each orbit
- Mission outcomes
- Landing outcomes

EDA with SQL

Task 1

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

```
%sql select distinct(LAUNCH_SITE) from SPACEXTBL
```

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

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

Task 2

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

```
%sql select * from SPACEXTBL 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

- 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 succesful landing outcome in ground pad was acheived.

Task 3

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

```
%sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596

Task 4

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

```
%sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

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 SPACEXTBL where Landing_Outcome = 'Success (ground pad)'
```

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

min(DATE)
2015-12-22

EDA with SQL

Task 6

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 SPACEXTBL WHERE Landing_Outcome = 'Success (drone ship)' and PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000
```

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

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

Task 7

List the total number of successful and failure mission outcomes

```
%sql select count(Mission_Outcome) from SPACEXTBL WHERE Mission_Outcome = 'Success' or Mission_Outcome = 'Failure (in flight)'
```

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

count(Mission_Outcome)
99

Task 8

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

```
%sql select Booster_Version from SPACEXTBL where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

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

Booster_Version
F9 B5 B1048.4

Task 9

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,4,2) AS Month, Booster_Version, Launch_site FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Failure%drone%' AND SUBSTR(Date,7,4) = '2015'
```

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

Month	Booster_Version	Launch_Site

Task 10

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.

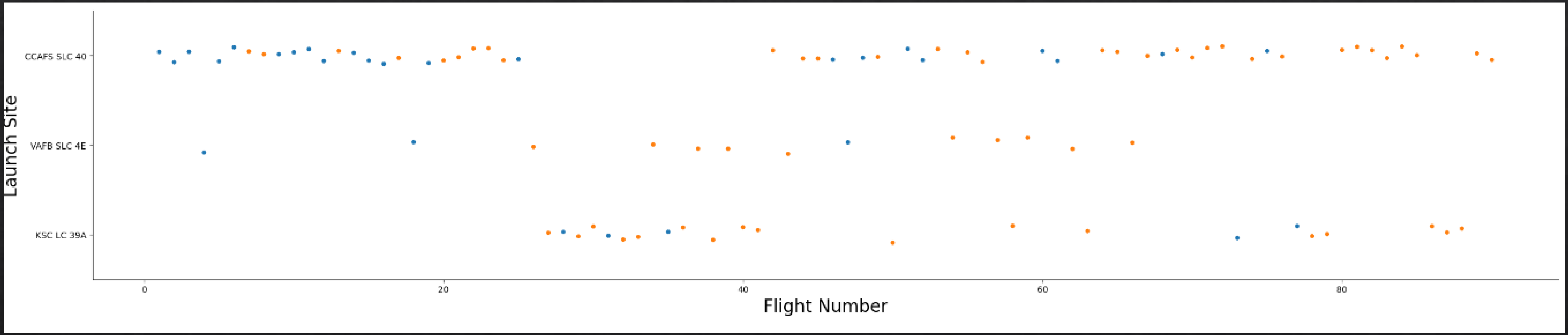
```
%sql SELECT Landing_Outcome, COUNT(*) AS Numbers FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success%' AND Date BETWEEN '04-06-2010' AND '20-03-2017' GROUP BY Landing_Outcome ORDER BY Numbers DESC
```

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

Landing_Outcome	Numbers

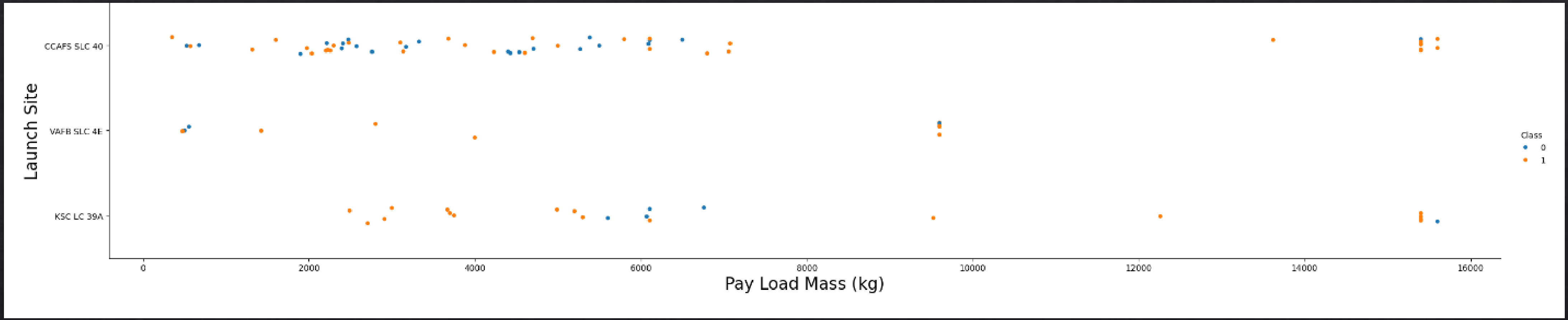
- List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- List the total number of successful and failure mission outcomes
- List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
- List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
- 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

EDA Visualizations



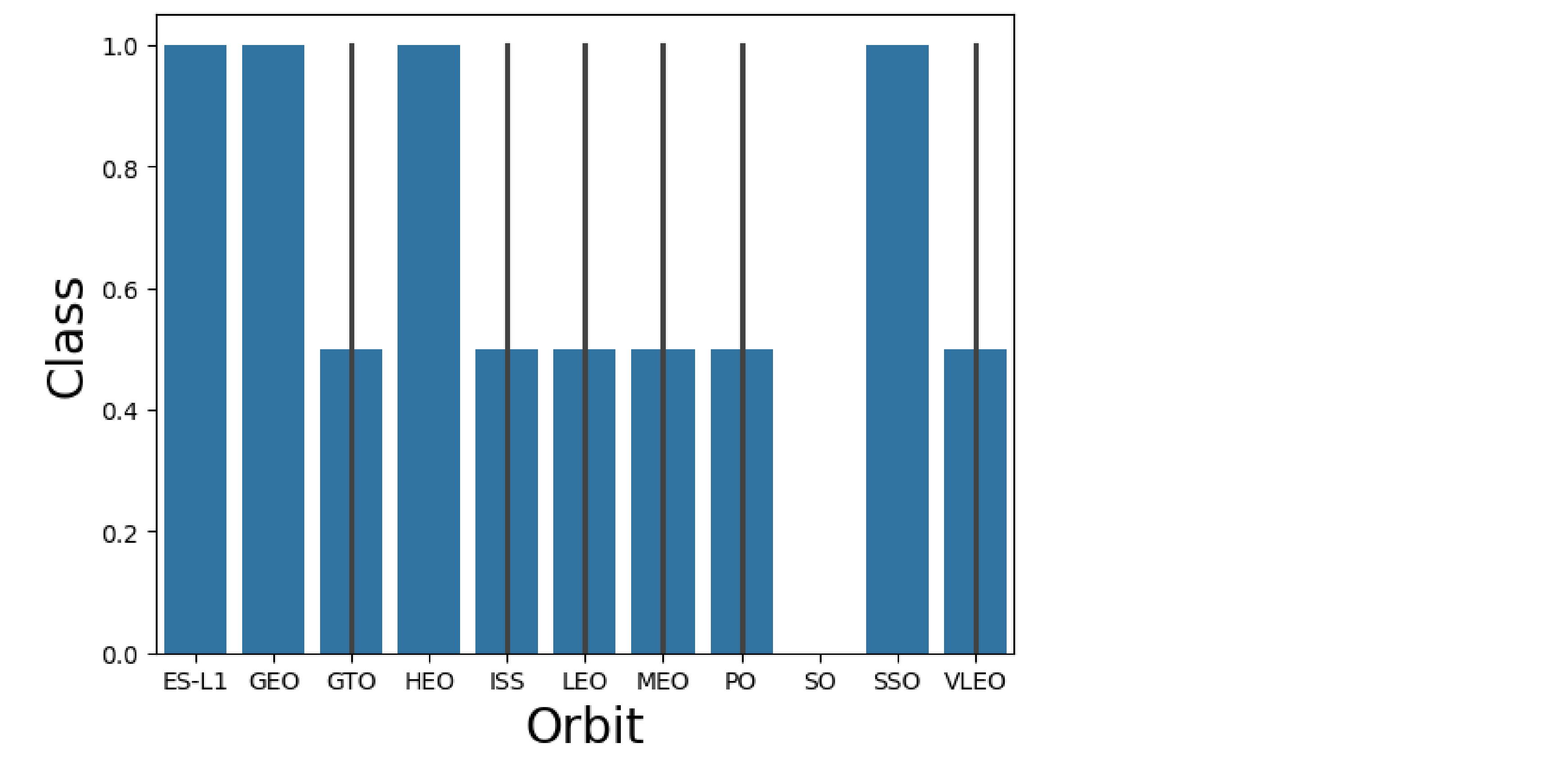
Visualize the relationship between Flight Number and Launch Site

EDA Visualizations



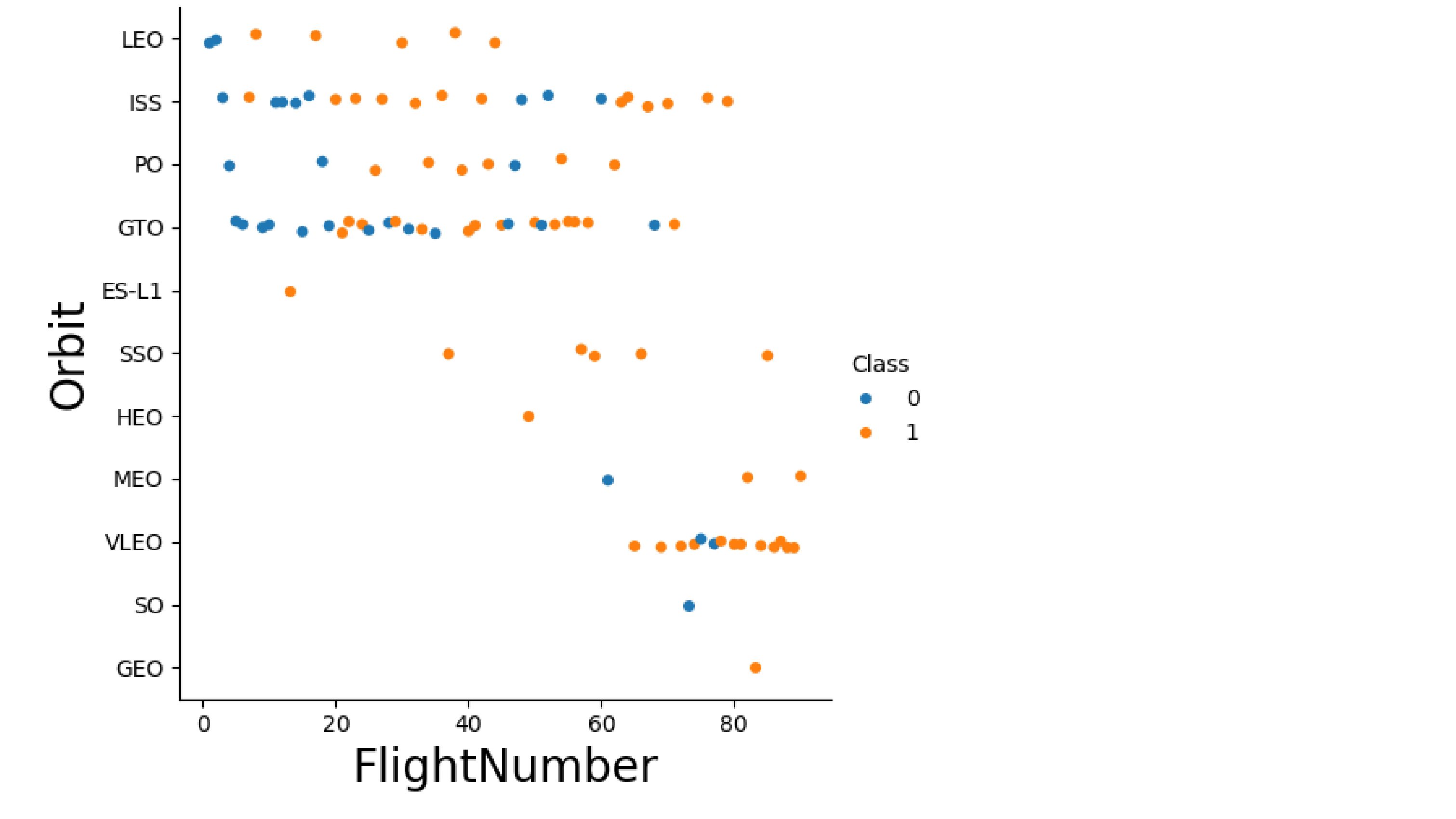
Visualize the relationship between Payload and Launch Site

EDA Visualizations



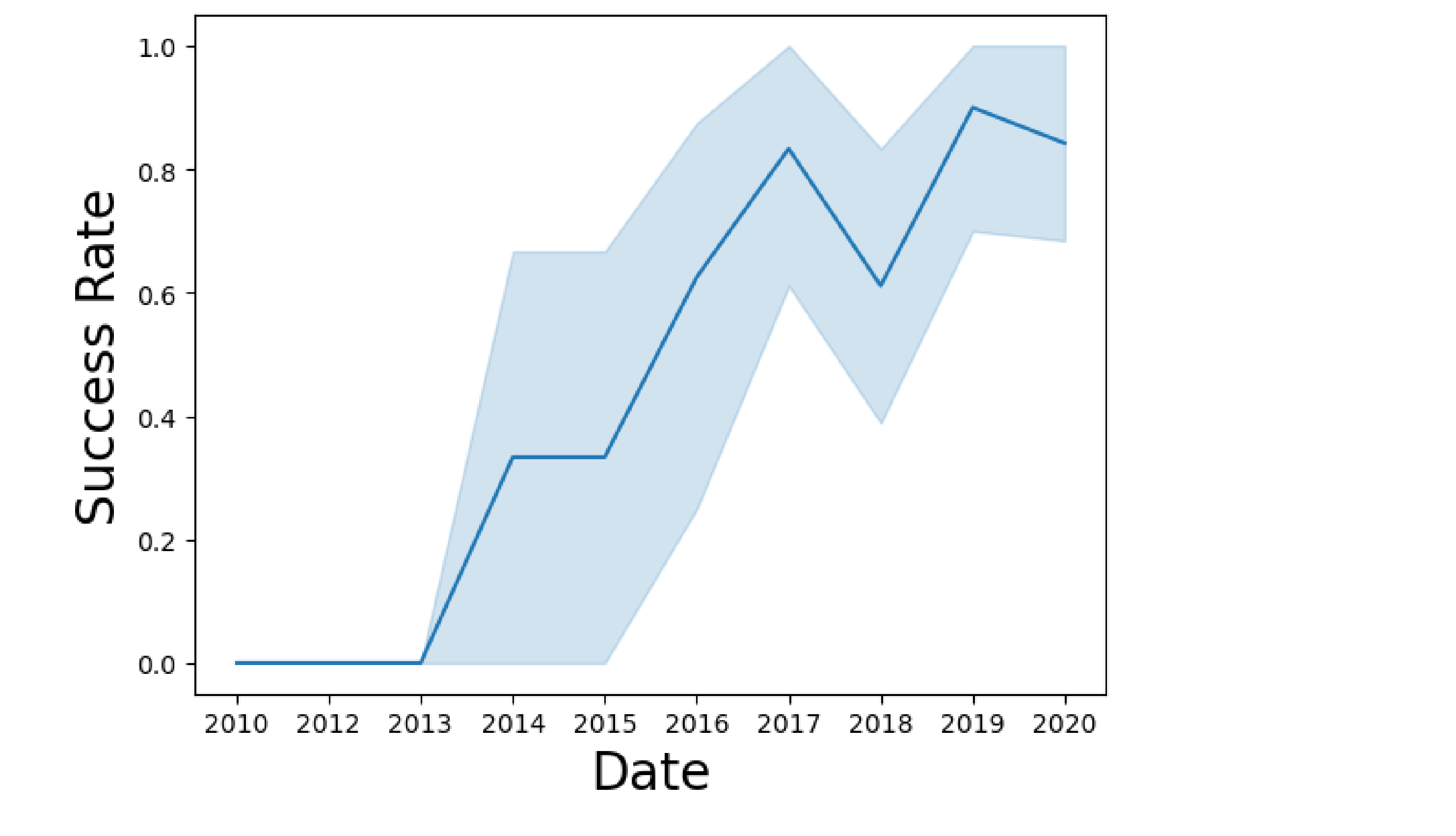
Visualize the relationship between success rate of each orbit type

EDA Visualizations



Visualize the relationship between Payload and Orbit type

EDA Visualizations



a line chart with x axis to be the extracted year and y axis to be the success rate

EDA Visualizations

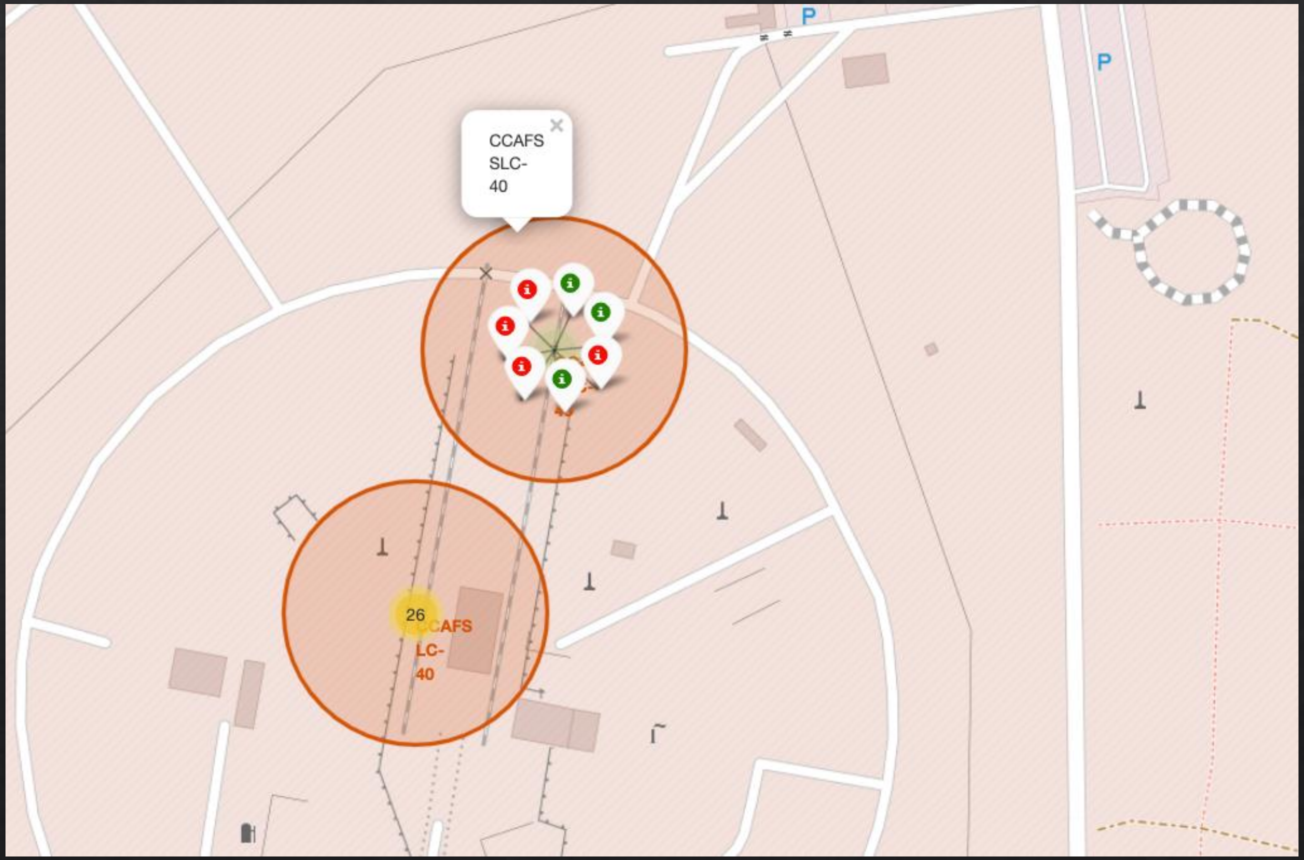
```
### TASK  8: Cast all numeric columns to `float64`
features_one_hot.astype(float)
```

	FlightNumber	PayloadMass	Flights	GridFins	Reused	Legs	Block	ReusedCount	Orbit_ES-L1	Orbit_GEO	...	Serial_B1048	Serial_B1049	Serial_B1050	Serial_B1051	Serial_B1054	Serial_B1056	Serial_B1058	Se
0	1.0	6104.959412	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	2.0	525.000000	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	3.0	677.000000	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	4.0	500.000000	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	5.0	3170.000000	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
85	86.0	15400.000000	2.0	1.0	1.0	1.0	5.0	2.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
86	87.0	15400.000000	3.0	1.0	1.0	1.0	5.0	2.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	
87	88.0	15400.000000	6.0	1.0	1.0	1.0	5.0	5.0	0.0	0.0	...	0.0	0.0	0.0	1.0	0.0	0.0	0.0	
88	89.0	15400.000000	3.0	1.0	1.0	1.0	5.0	2.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
89	90.0	3681.000000	1.0	1.0	0.0	1.0	5.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

90 rows × 80 columns

Create dummy variables to categorical columns
Cast all numeric columns to `float64`

Interactive Map with Folium



Predictive Analysis with Machine Learning

TASK 4

Create a logistic regression object then create a GridSearchCV object `logreg_cv` with cv = 10. Fit the object to

```
parameters ={'C':[0.01,0.1,1],
             'penalty':['l2'],
             'solver':['lbfgs']}
```

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()
logreg_cv = GridSearchCV(lr, param_grid=parameters,scoring='accuracy', cv=10)
logreg_cv.fit(X_train, Y_train)
logreg_cv.best_params_
```

```
{'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
```

We output the `GridSearchCV` object for logistic regression. We display the best parameters using the data attr

```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
accuracy : 0.8464285714285713
```

TASK 6

Create a support vector machine object then create a `GridSearchCV` object `svm_cv` with cv = 10. Fit the object to find the best

```
parameters = {'kernel':('linear', 'rbf','poly','rbf', 'sigmoid'),
              'C': np.logspace(-3, 3, 5),
              'gamma':np.logspace(-3, 3, 5)}
svm = SVC()
```

```
svm_cv = GridSearchCV(svm, param_grid=parameters,scoring='accuracy', cv=10)
svm_cv.fit(X_train, Y_train)
svm_cv.best_params_
```

```
{'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
```

```
print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
print("accuracy :",svm_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
accuracy : 0.8482142857142856
```

TASK 7

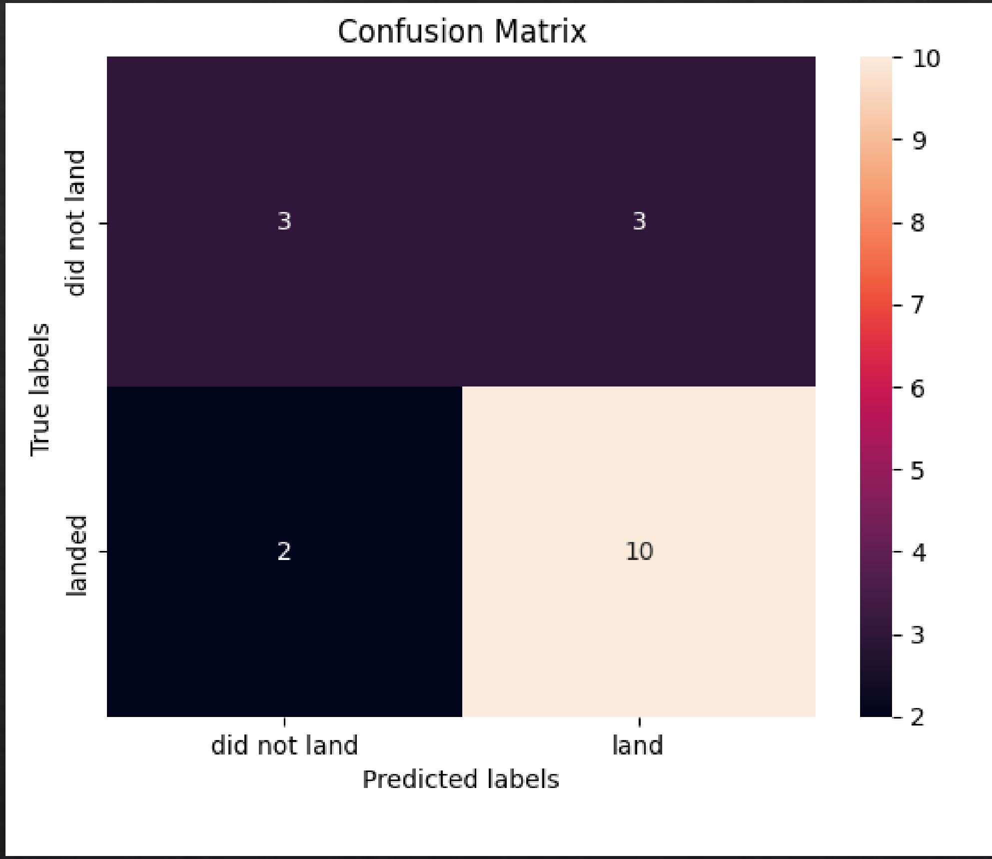
Calculate the accuracy on the test data using the method `score` :

```
svm_cv.score(X_test, Y_test)
```

```
0.8333333333333334
```

Accuracy: 0.8482142857142856

Predictive Analysis with Machine Learning



All methods yields about the same results

Conclusion

- There is a correlation between success rate and launch site
- There is a positive correlation between payload mass and success rate
- SO orbit type has the least success rate
- The accuracy is about 84%