IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Manuel Albeza Guijarro
2023-09-17

Outline

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

Executive Summary

In recent customer feedback sessions, 52% of **LAUNCH SUCCESS RATE** of **Space X Falcon 9** have expressed a need for a simpler, cheaper version of our product. In surveys of customers who have chosen watches from competitors, the price aspect is mentioned 87% of the time. To better serve our current customers and to expand into new markets, we need to develop a series of watches that we can sell at a price suitable for this market.

Our new series of watches will start out 20% cheaper than our current cheapest option, with the possibility of options 40% cheaper or more, depending on material and movement. To offer these prices, we will do the following:

Offer watches made of new materials, possibly including silicone or wood.

Use high quality quartz movement instead of automatic movement with internal mechanism.

Introduce customizable legging options, focused on choice and flexibility over traditional luxury.

Please note that all watches will undergo rigorous quality control in order to maintain the same exceptional quality speed and precision that our current products offer.

Introduction

Regarding Launch Sites Locations Analysis of **Space X Falcon 9**. That will help us achieve FY 2022 Goal 3: Expand the brand.

These new offerings have the potential to generate over $3 million in annual revenue, which will help us achieve FY 2022 Goal 1: $7 million in annual revenue.

Early customer feedback sessions indicate that cheaper options will not impact the value or prestige of the luxury brand, although this is a risk that must be taken into account during design.

To mitigate that risk, the product marketing team will begin working on their go-to-market strategy six months before launch.

Cheaper and more varied offerings not only allow us to enter a new market, but will also expand our brand in a positive way. With the attention gained from these new offerings, plus anticipated demand for cheaper watches, we expect to increase market share by 2% annually.

To learn more, read our go-to-market strategy and customer feedback documentation.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

Data collection that has been followed in this analysis can be carried out through 4 research methods:

1. **Analytical method**.

   Review each data in depth and in an orderly manner; goes from the general to the particular to obtain conclusions.

2. **Synthetic method**.

   Analyzes and summarizes information; Through logical reasoning he arrives at new knowledge.

3. **Deductive method**.

   Starting from general knowledge to reach singular knowledge.

4. **Inductive method**.

   From the analysis of particular data, he reaches general conclusions.

Data Collection – SpaceX API

	FlightNumber	PayloadMass	Flights	Block	ReusedCount	Orbit_ES-L1	Orbit_GEO	Orbit_GTO	Orbit_HEO	Orbit_ISS	...	Serial_B1058
0	1.0	6104.959412	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
1	2.0	525.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
2	3.0	677.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	...	0.0
3	4.0	500.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0
4	5.0	3170.000000	1.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...
85	86.0	15400.000000	2.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	0.0
86	87.0	15400.000000	3.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	1.0
87	88.0	15400.000000	6.0	5.0	5.0	0.0	0.0	0.0	0.0	0.0	...	0.0
88	89.0	15400.000000	3.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	0.0
89	90.0	3681.000000	1.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0

90 rows × 83 columns

Data Wrangling

There are four broad **steps** in the **Wrangling process** that has been followed in this analysis :

1. **Discovery**

   In the discovery stage, you will basically prepare yourself for the rest of the process. Here, you'll think about the questions you want to answer and the type of data you'll need to answer them. You will also locate the data you plan to use and examine its current form to discover how you will clean, structure, and organize your data in the next stages.

2. **Transformation**

   During the transformation stage, you'll act on the plan you developed during the discovery stage. This piece of the process can be broken down into four components: structuring, normalizing and denormalizing, cleaning, and enriching.

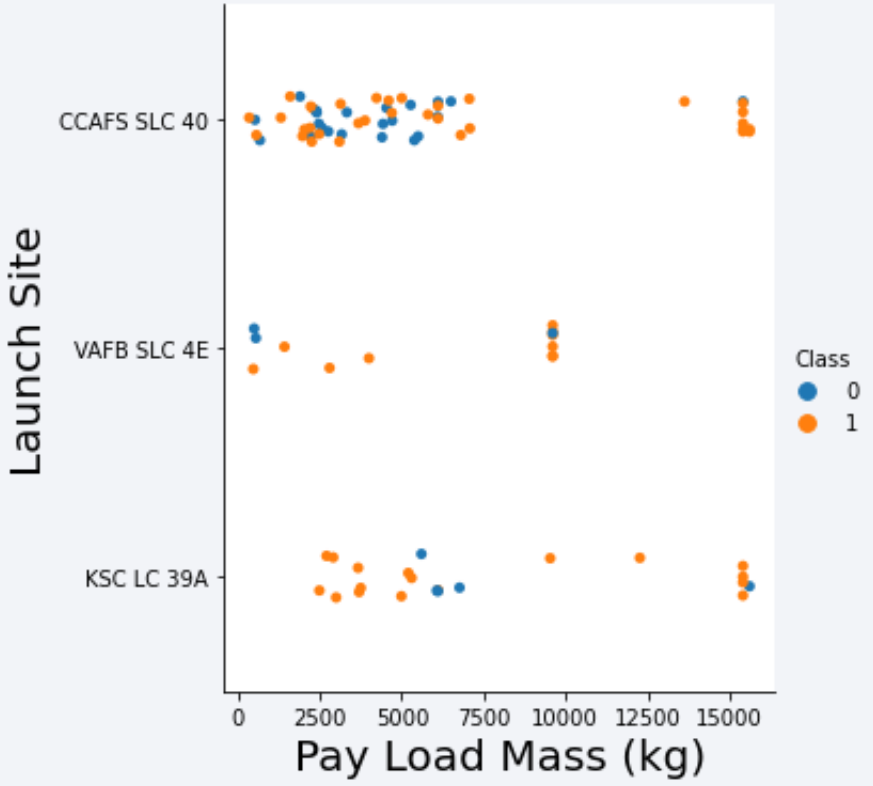
3. **Validation**

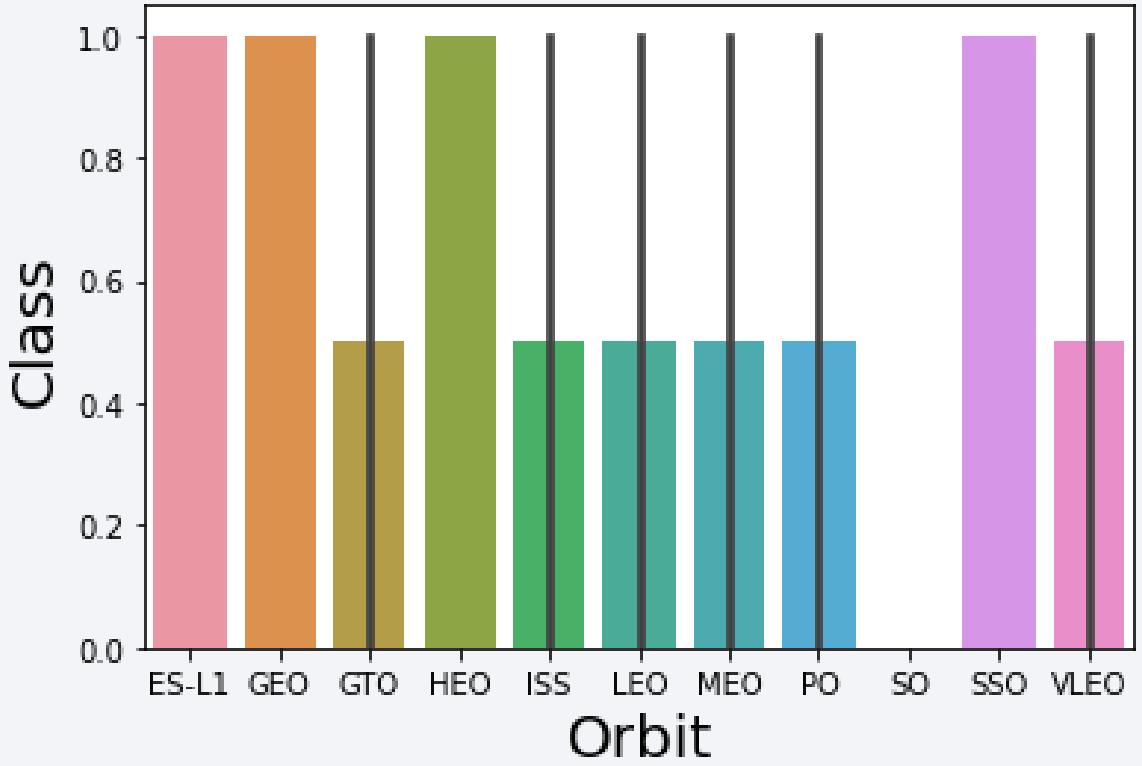
   During the validation step, you essentially check the work you did during the transformation stage, verifying that your data is consistent, of sufficient quality, and secure. This step may be completed using automated processes and can require some programming skills.

4. **Publishing**

   After you've finished validating your data, you're ready to publish it. When you publish data, you'll put it into whatever file format you prefer for sharing with other team members for downstream analysis purposes.

EDA with Data Visualization





EDA with SQL

```
!pip install sqlalchemy==1.3.9
!pip install ibm_db_sa
!pip install ipython-sql
```

```
import ibm_db
import ibm_db_sa
import sqlalchemy
```

Connect to the database

Let us first load the SQL extension and establish a connection with the database

```
%load_ext sql
```

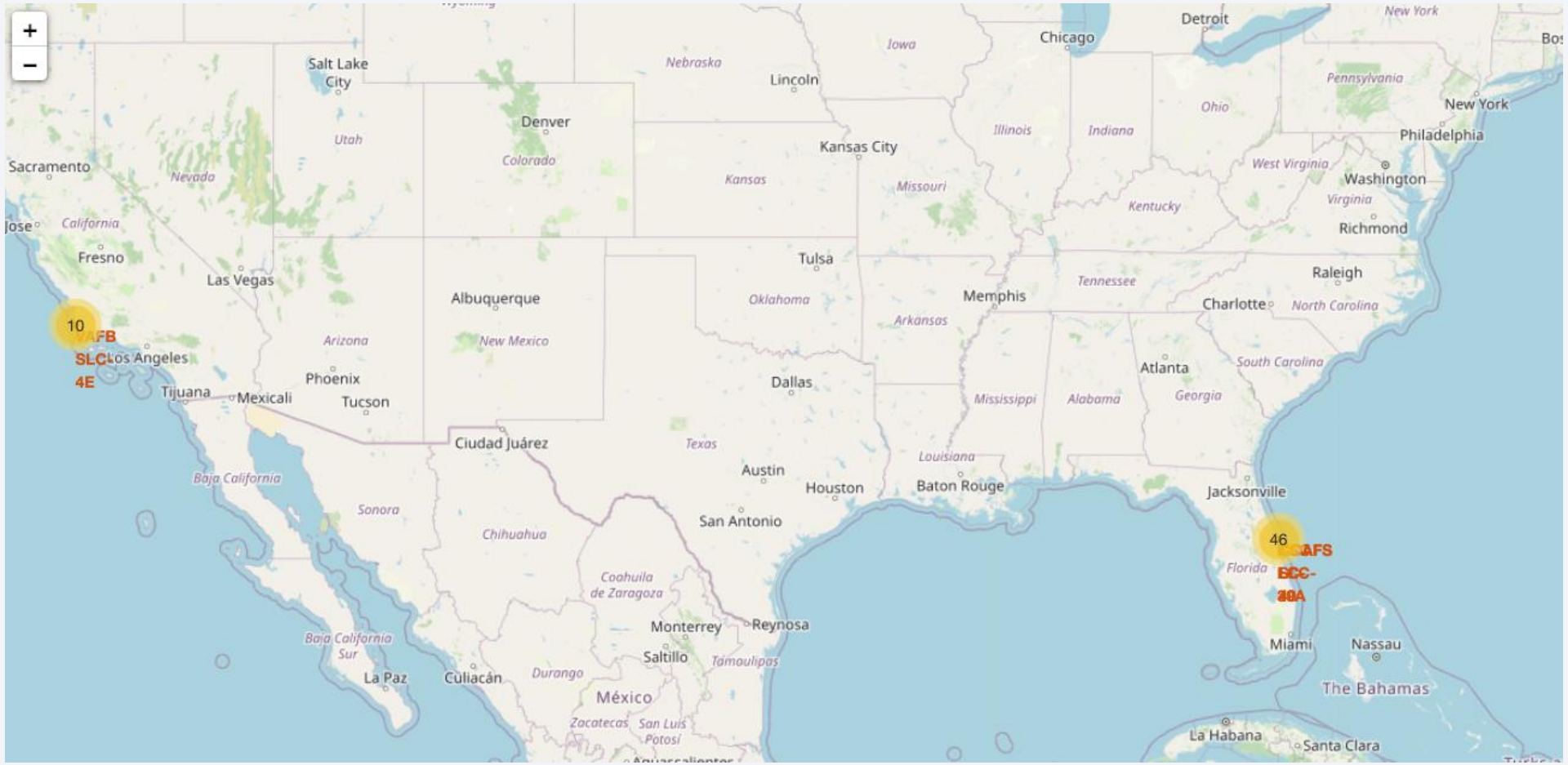
Display 5 records where launch sites begin with the string 'CCA'

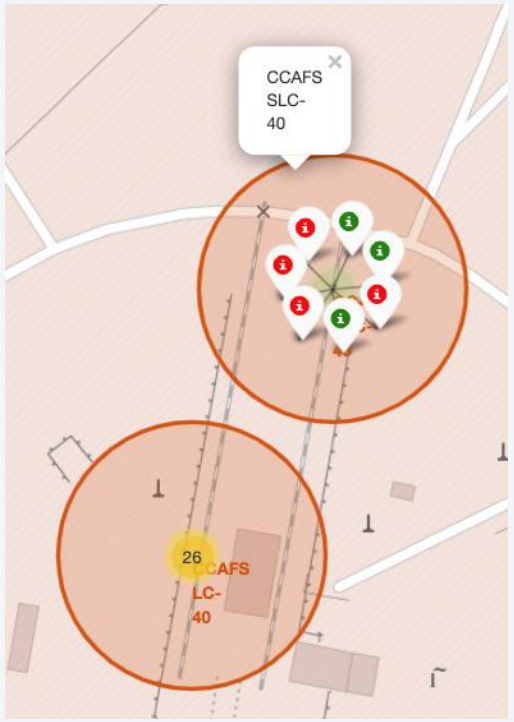
```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
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	None	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	None	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	None	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	None	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	None	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Build an Interactive Map with Folium





Build a Dashboard with Plotly Dash

```
#TASK 2.4: Creating Callbacks
# Define the callback function to update the input container based on the selected statistics
@app.callback(
    Output(component_id='select-year', component_property='disabled'),
    Input(component_id='dropdown-statistics',component_property='value'))

def update_input_container(selected_statistics):
    if selected_statistics =='Yearly Statistics':
        return False
    else:
        return True

#Callback for plotting
# Define the callback function to update the input container based on the selected statistics
@app.callback(
    Output(component_id='output-container', component_property='children'),
    [Input(component_id='select-year', component_property='value'), Input(component_id='dropdown-statistics', component_pro


def update_output_container(recession_option, year_option):
    if recession_option == 'Recession Period Statistics':
        # Filter the data for recession periods
        recession_data = data[data['Recession'] == 1]
    if year_option == 'Yearly Statistics':
        # Filter the data for recession periods
        recession_data = data[data['Year'] == 1]
```

Predictive Analysis (Classification)

Create a decision tree classifier object then create a `GridSearchCV` object `tree_cv` with cv = 10. Fit the object to find the best parameters from the dictionary `parameters`.

```
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier()
```

```
# Instantiate the GridSearchCV object: tree_cv
tree_cv = GridSearchCV(tree, parameters, cv=10)

# Fit it to the data
tree_cv.fit(X_train, Y_train)
```

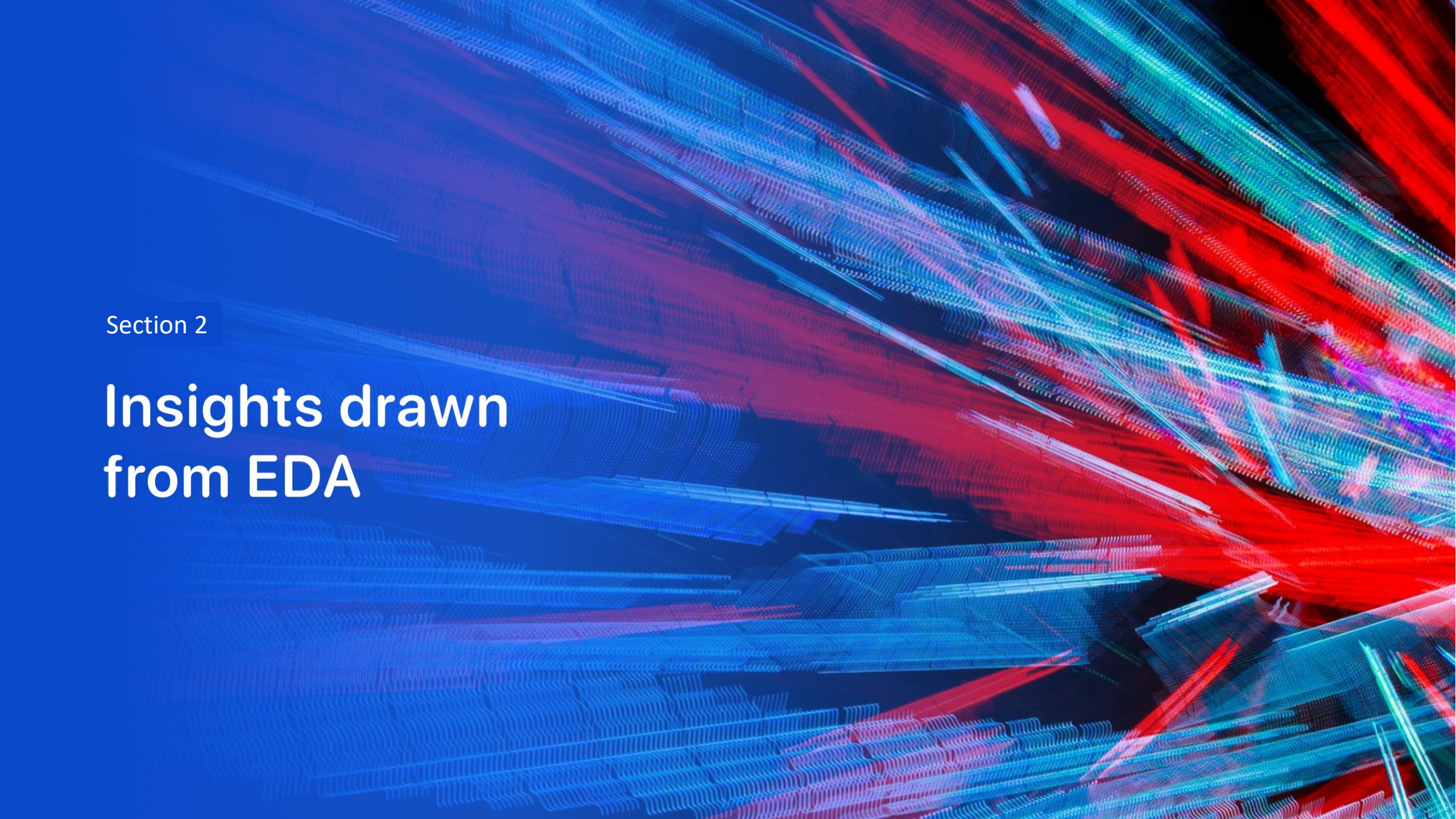
```
GridSearchCV(cv=10, estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [2, 4, 6, 8, 10, 12, 14, 16, 18],
                         'max_features': ['auto', 'sqrt'],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10],
                         'splitter': ['best', 'random']})
```

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

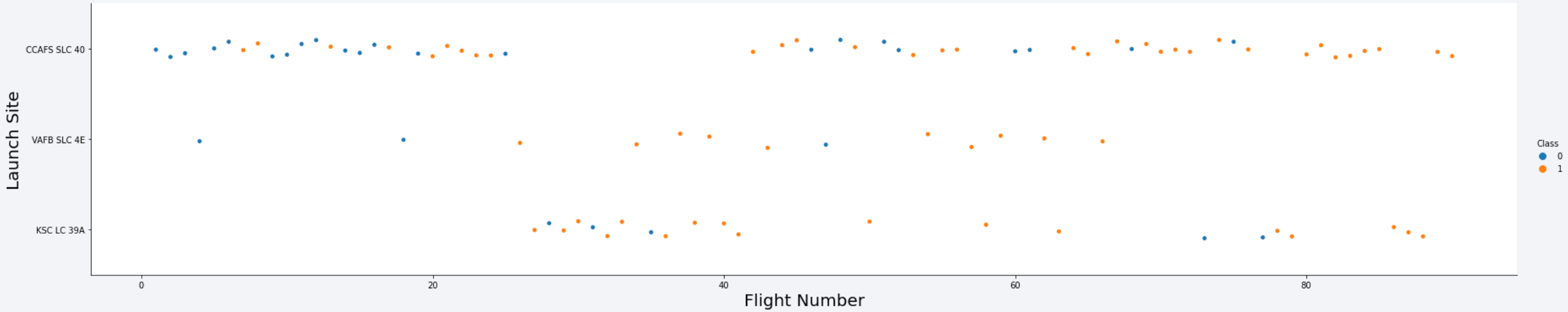
```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'auto', 'min_samples_leaf':
4, 'min_samples_split': 10, 'splitter': 'random'}
accuracy : 0.8857142857142856
```

Results

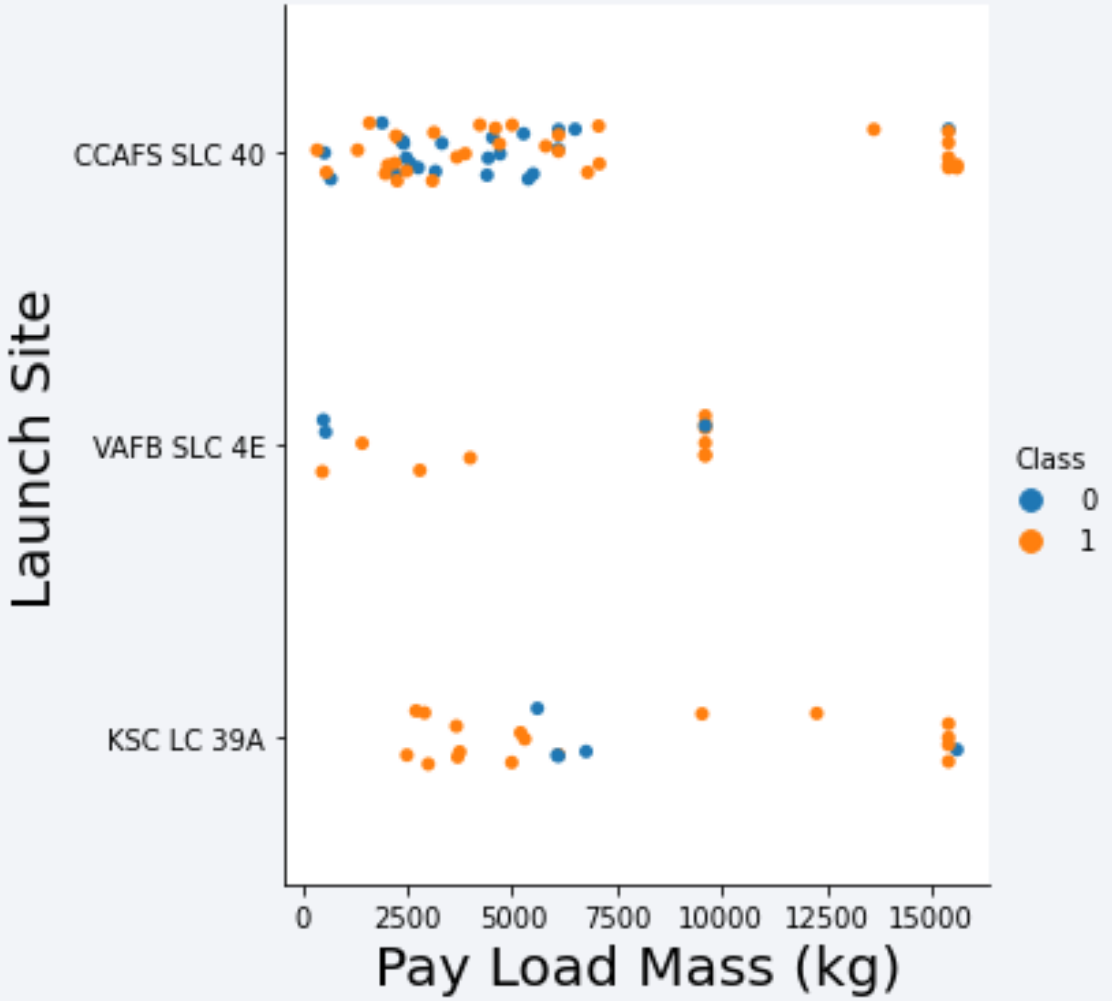

Confusion Matrix
3
3
0
12
True labels
did not land
landed
did not land
land
Predicted labels
12
10
8
6
4
2
0


Section 2
Insights drawn from EDA

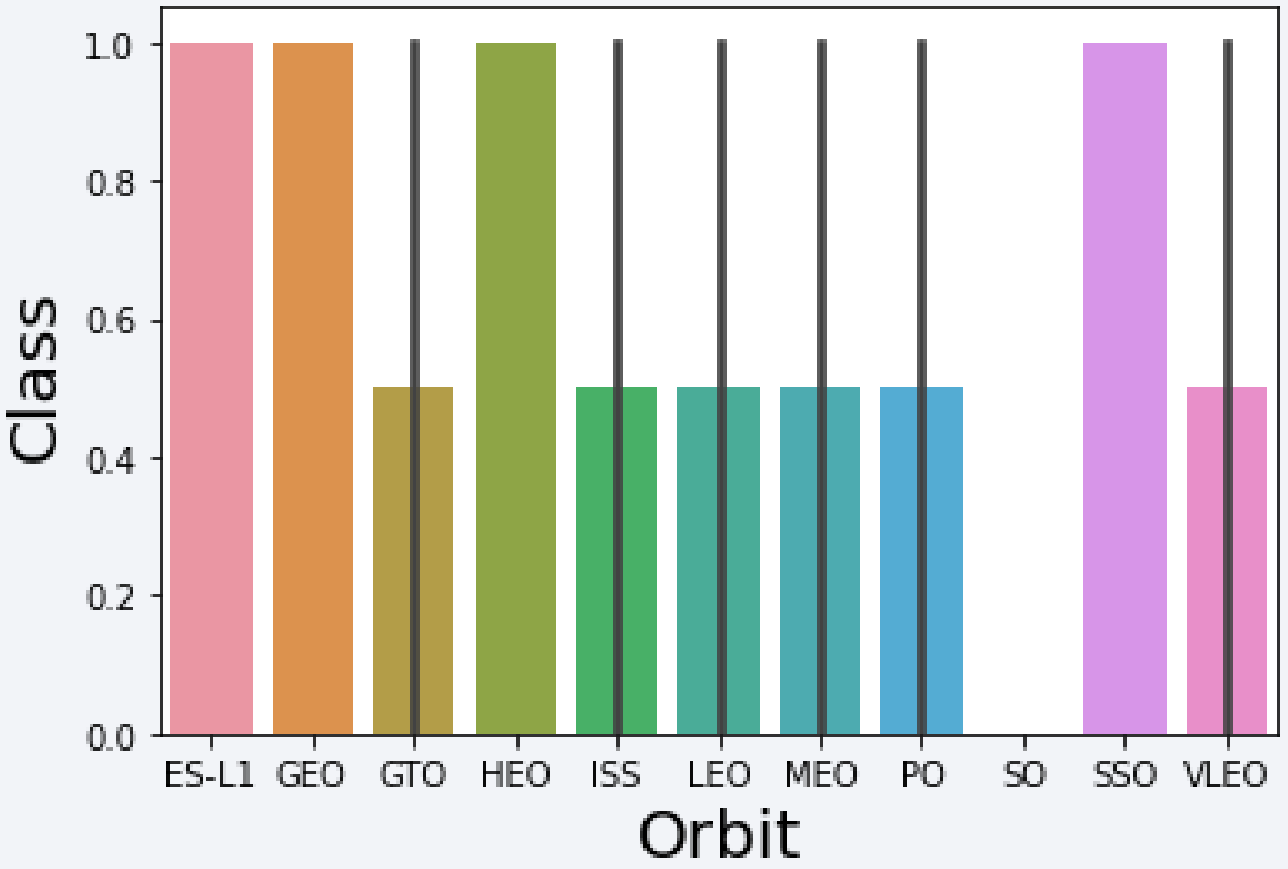
Flight Number vs. Launch Site



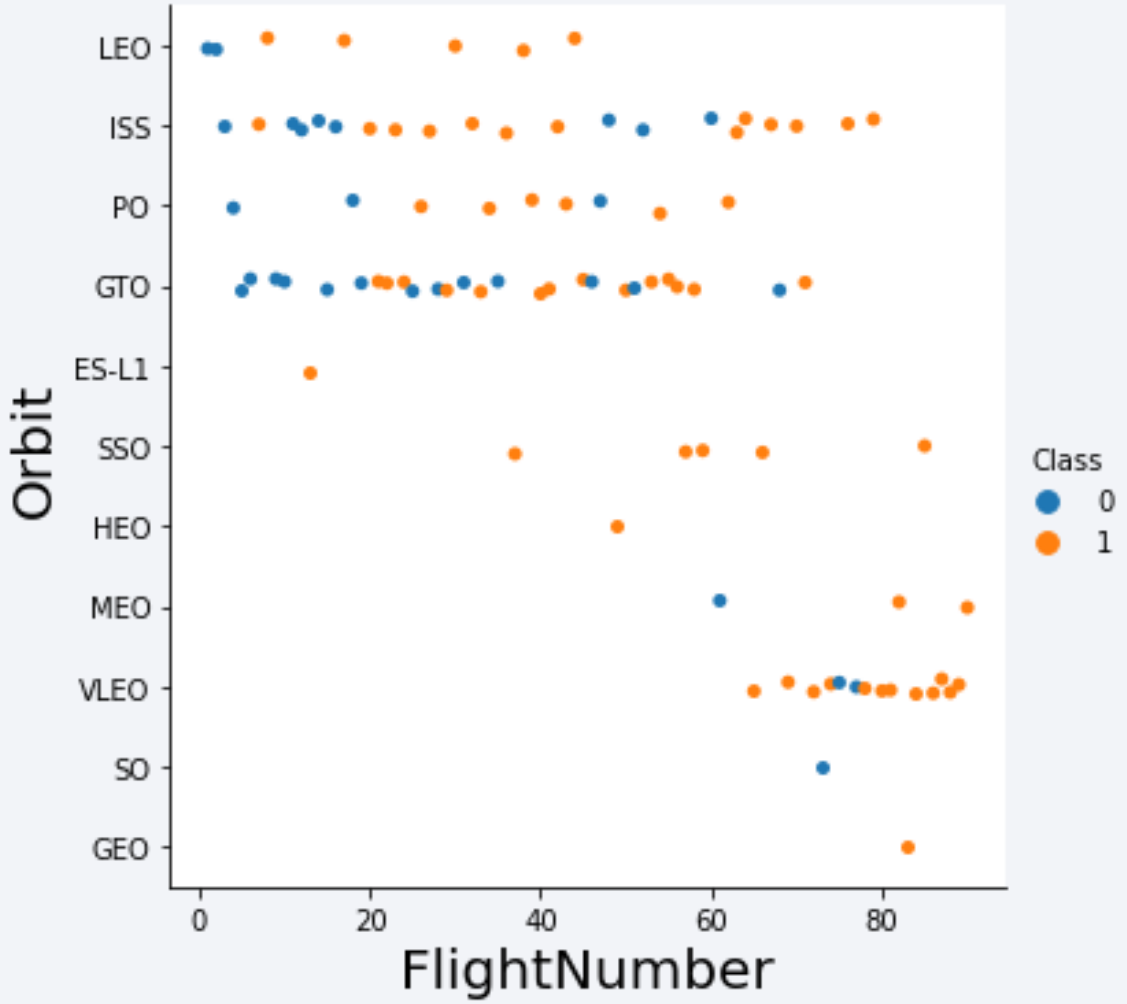
Payload vs. Launch Site



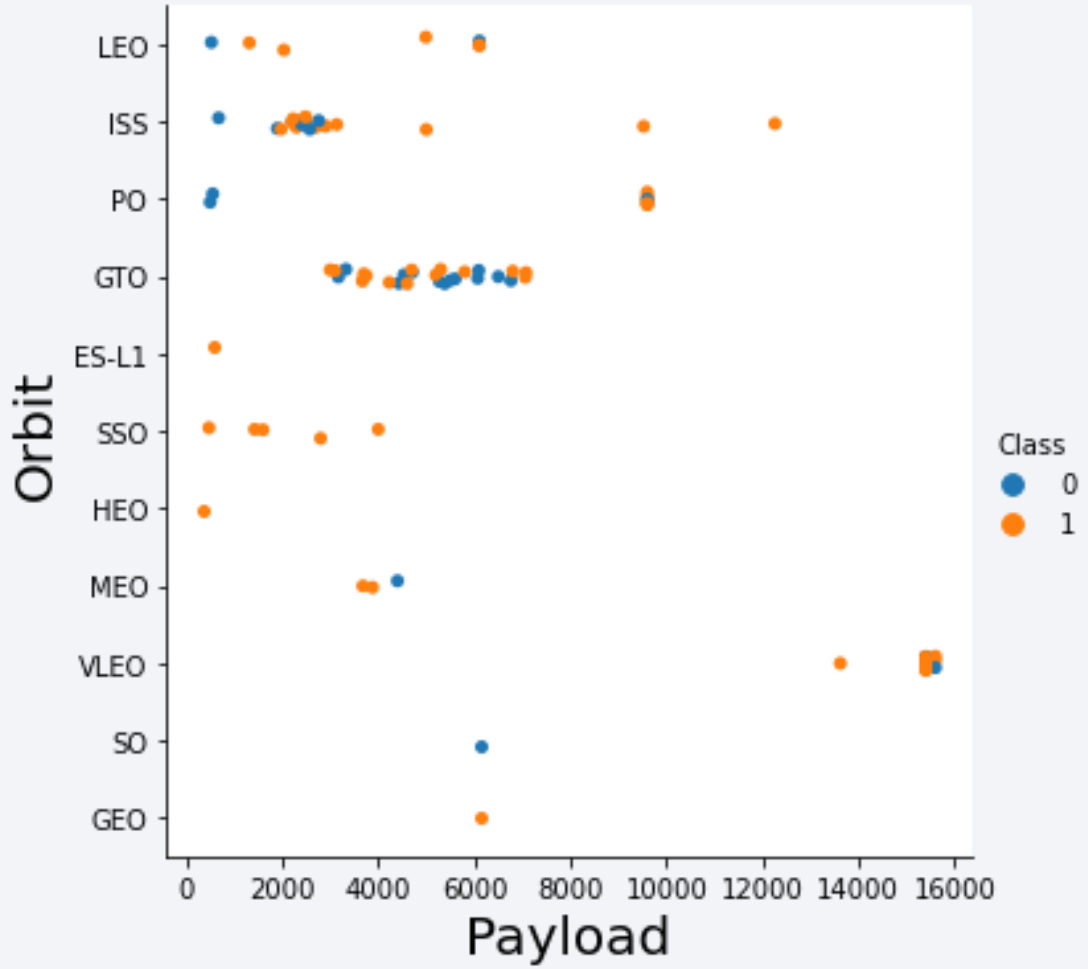
Success Rate vs. Orbit Type



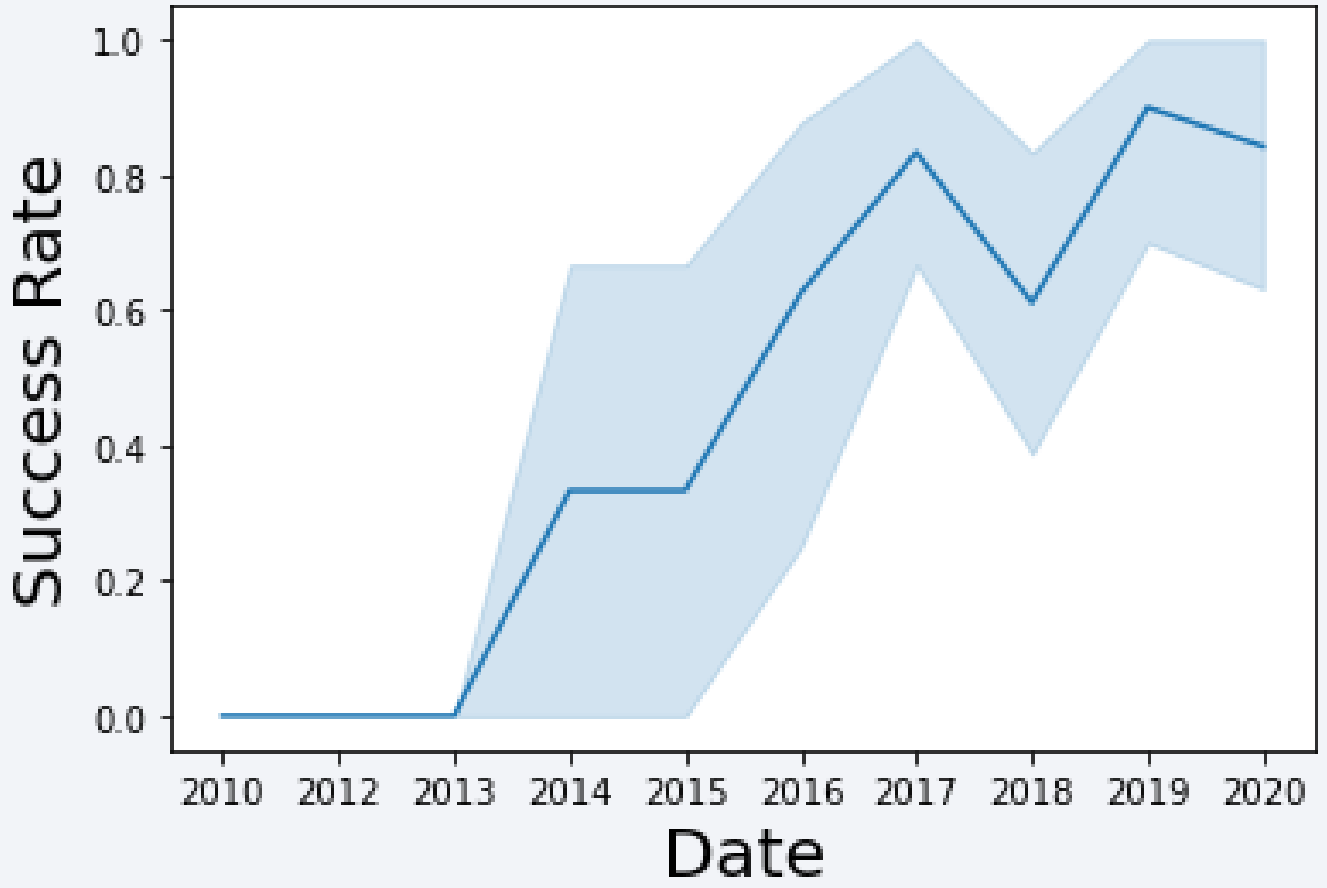
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

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

```
%sql SELECT Distinct LAUNCH_SITE FROM SPACEXTBL
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

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

Launch Site Names Begin with 'CCA'

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

```
%sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
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	None	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	None	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	None	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	None	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	None	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE CUSTOMER='NASA (CRS)'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

1
45596

Average Payload Mass by F9 v1.1

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'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

1
2928.400000

First Successful Ground Landing Date

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)'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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 PAYLOAD_MASS__KG_ between 4000 and 6000 AND LANDING__OUTCOME='Success (dror
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
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

List the total number of successful and failure mission outcomes

```
%sql SELECT COUNT(*) FROM SPACEXTBL WHERE MISSION_OUTCOME LIKE '%Success%' OR MISSION_OUTCOME LIKE '%Failure%'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

1
101

Boosters Carried Maximum Payload

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)
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
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

2015 Launch Records

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

```
%sql SELECT TO_CHAR(TO_DATE(MONTH("DATE"), 'MM'), 'MONTH') AS MONTH_NAME, \
    LANDING__OUTCOME AS LANDING__OUTCOME, \
    BOOSTER_VERSION AS BOOSTER_VERSION, \
    LAUNCH_SITE AS LAUNCH_SITE \
    FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND "DATE" LIKE '%2015%'
```

```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

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

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

Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

```
%sql SELECT "DATE", COUNT(LANDING__OUTCOME) as COUNT FROM SPACEXTBL \
    WHERE "DATE" BETWEEN '2010-06-04' and '2017-03-20' AND LANDING__OUTCOME LIKE '%Success%' \
    GROUP BY "DATE" \
    ORDER BY COUNT(LANDING__OUTCOME) DESC
```

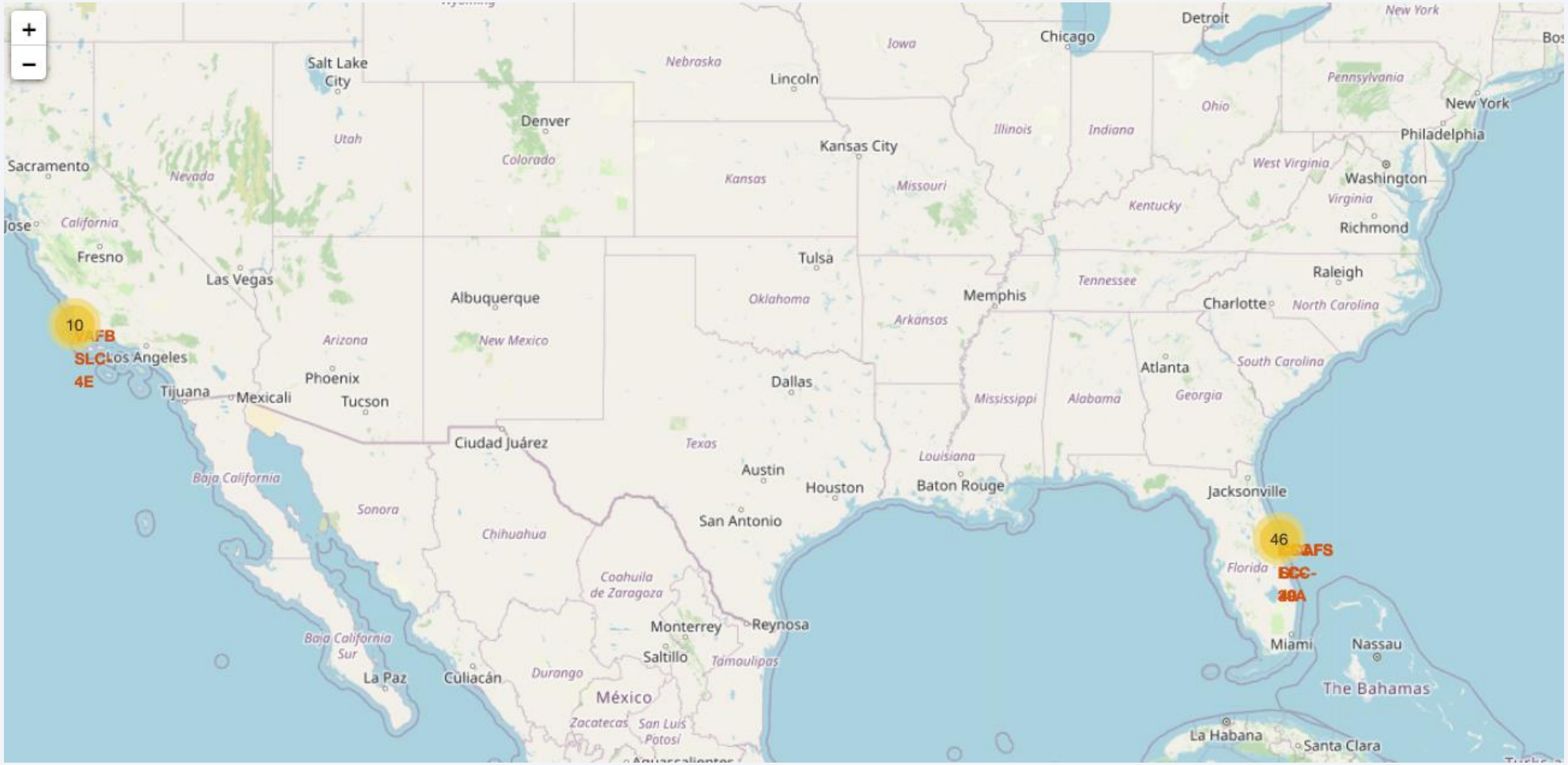
```
 * ibm_db_sa://kcq64325:***@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
Done.
```

DATE	COUNT
2015-12-22	1
2016-04-08	1
2016-05-06	1
2016-05-27	1
2016-07-18	1
2016-08-14	1
2017-01-14	1
2017-02-19	1

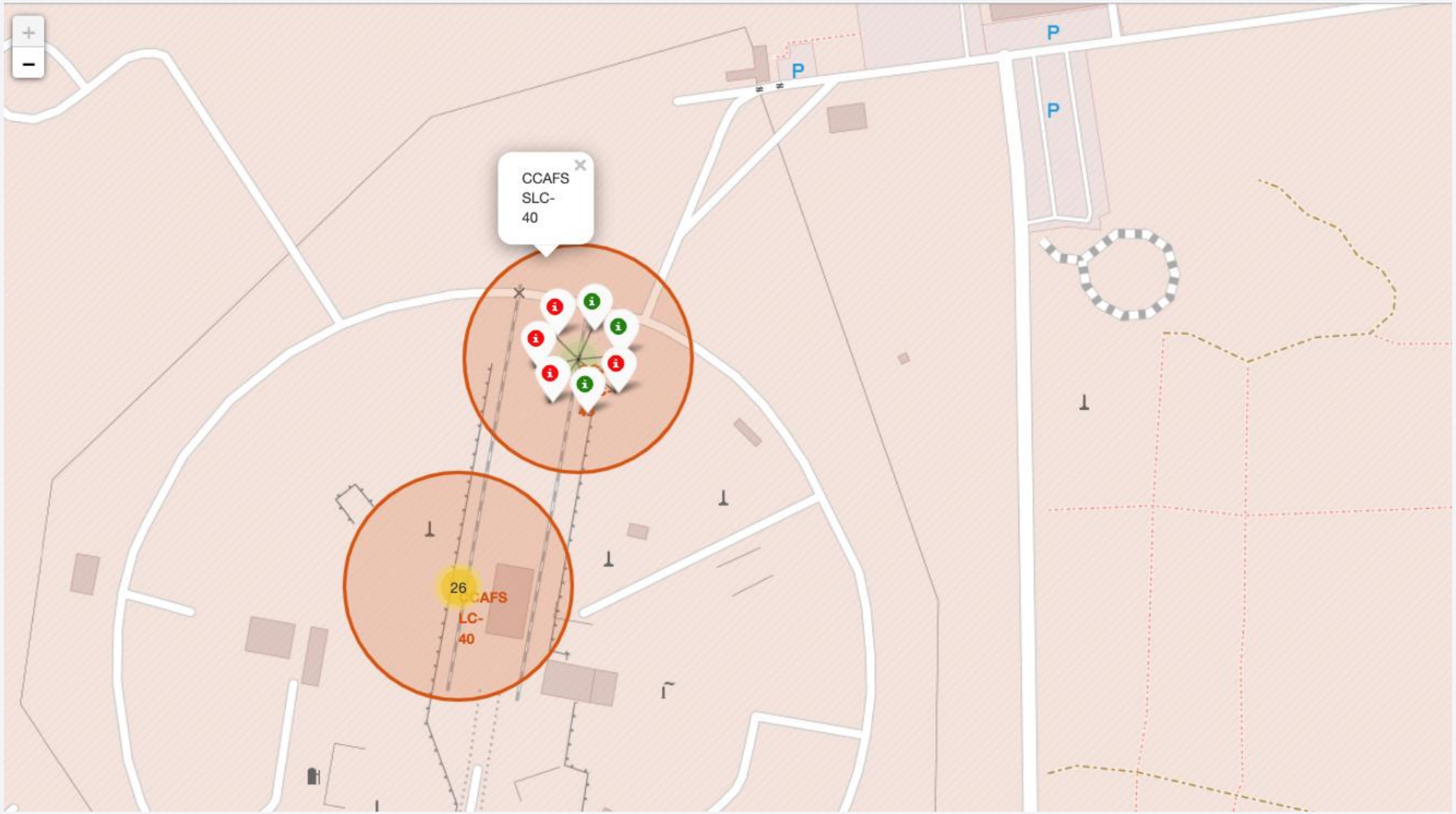
Section 3

Launch Sites Proximities Analysis

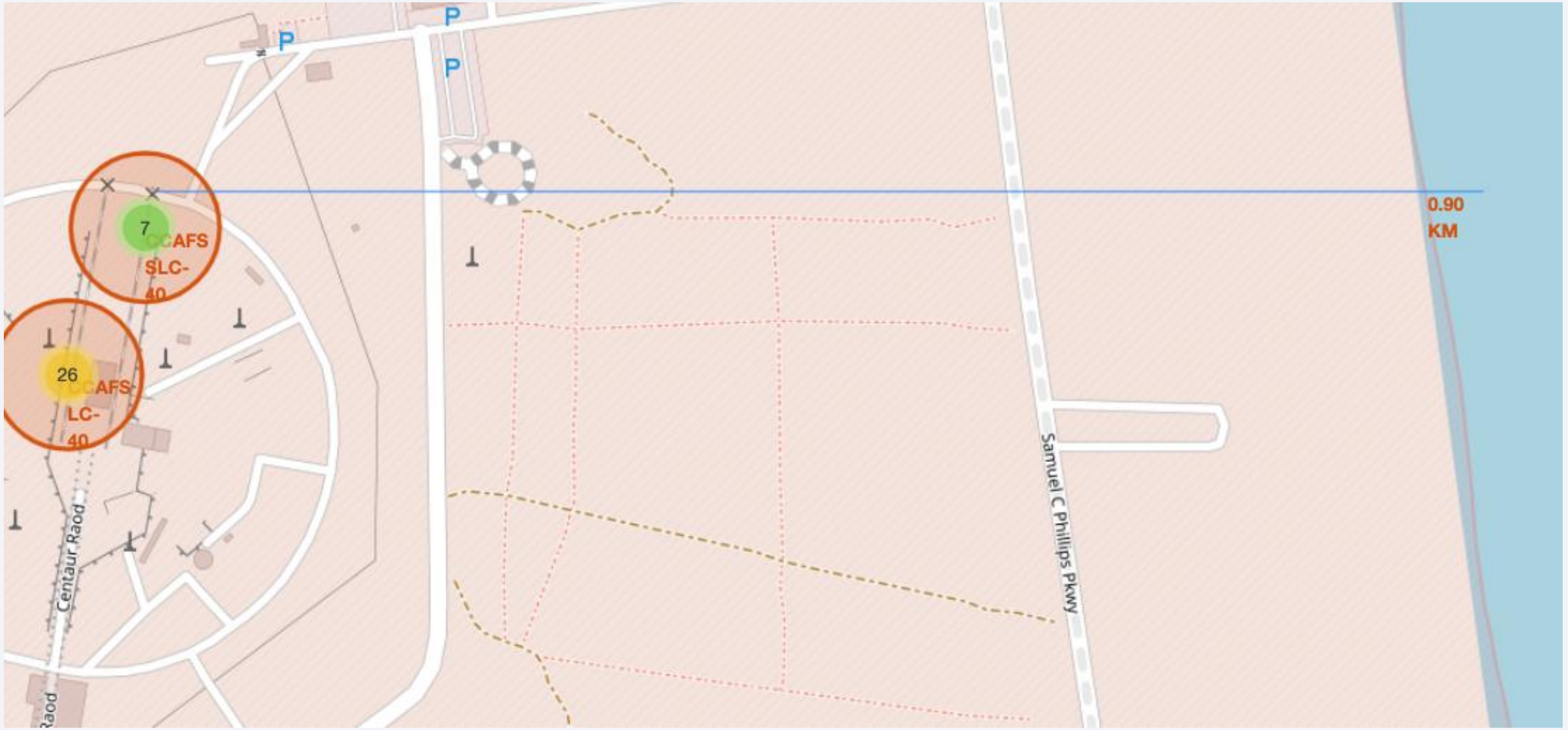
Success and Failed Launches Map

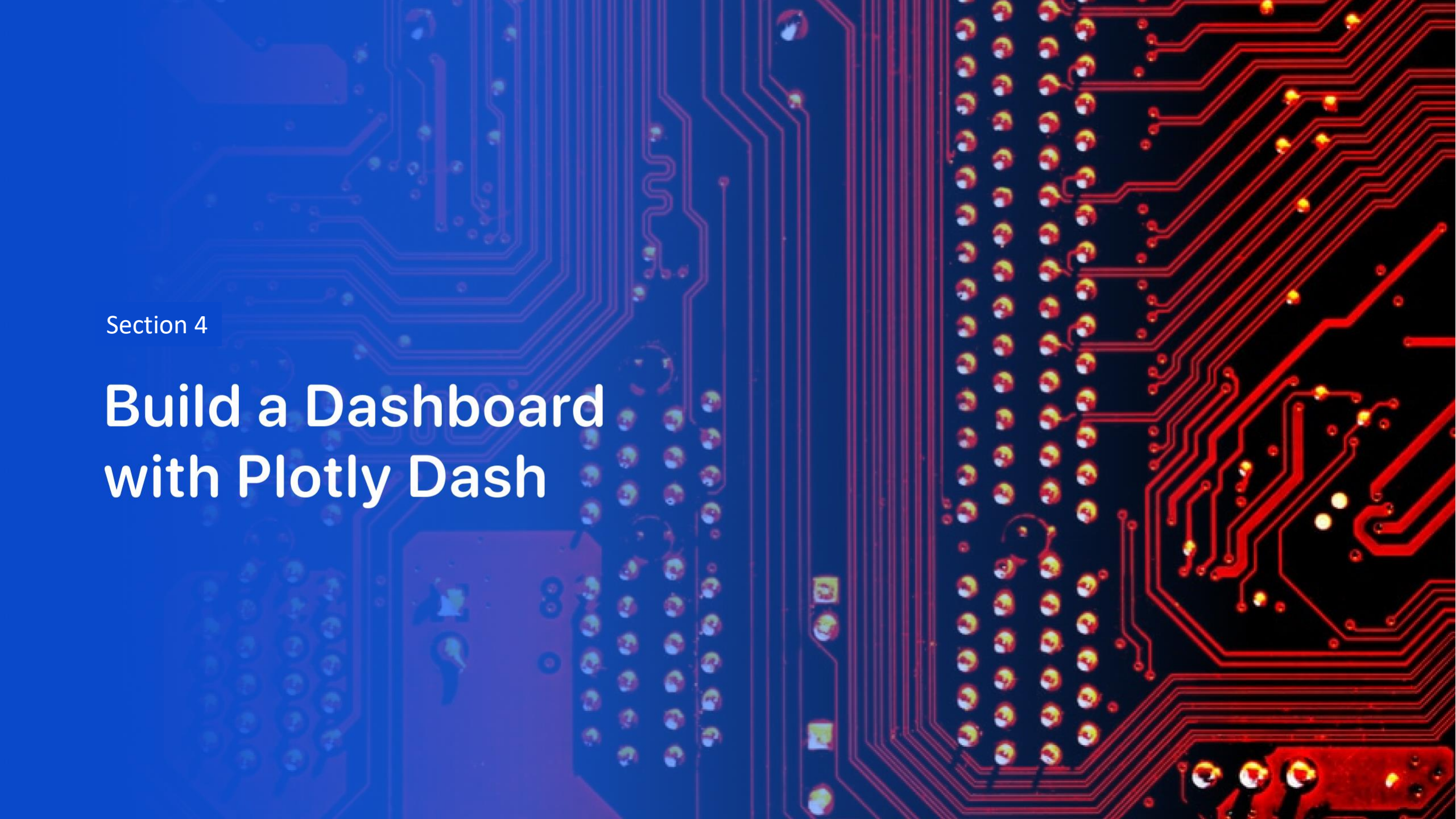


Map details 1

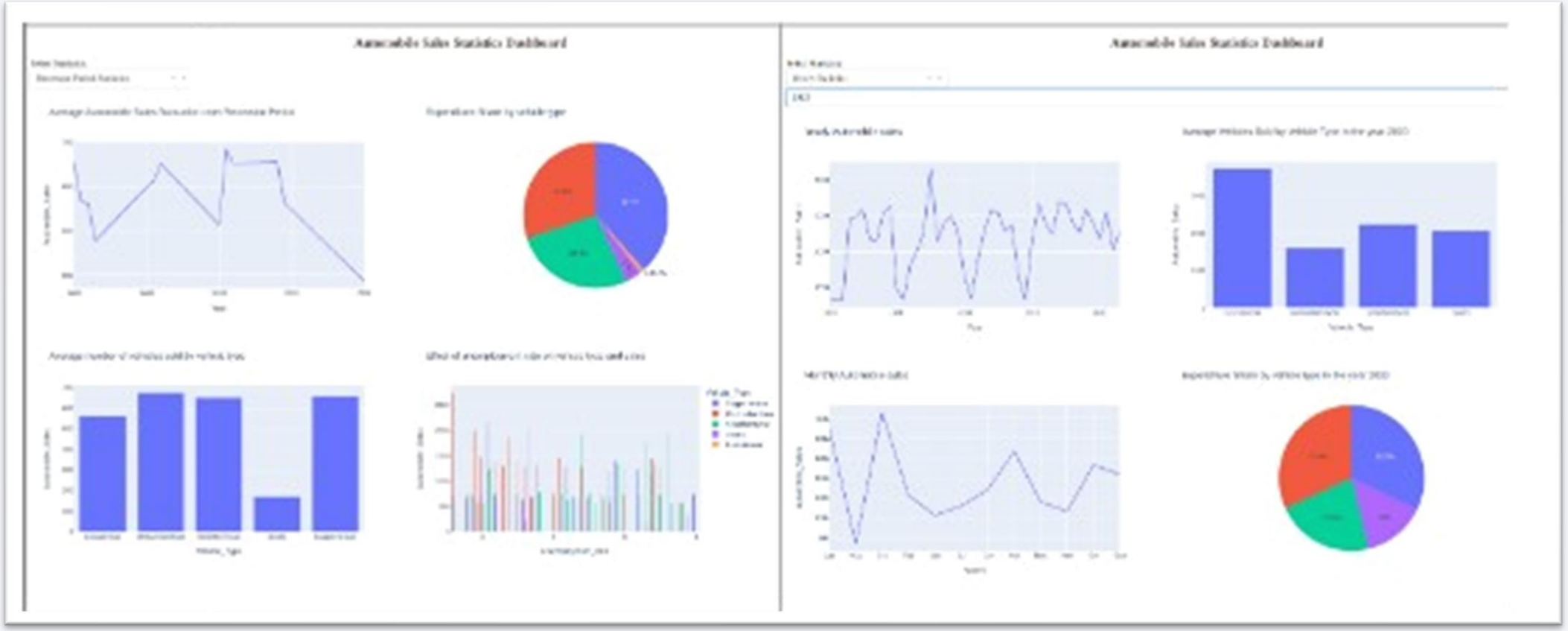


Map details 2




Section 4
Build a Dashboard with Plotly Dash

Dashboard Screenshot



Section 5

Predictive Analysis (Classification)

Classification Accuracy

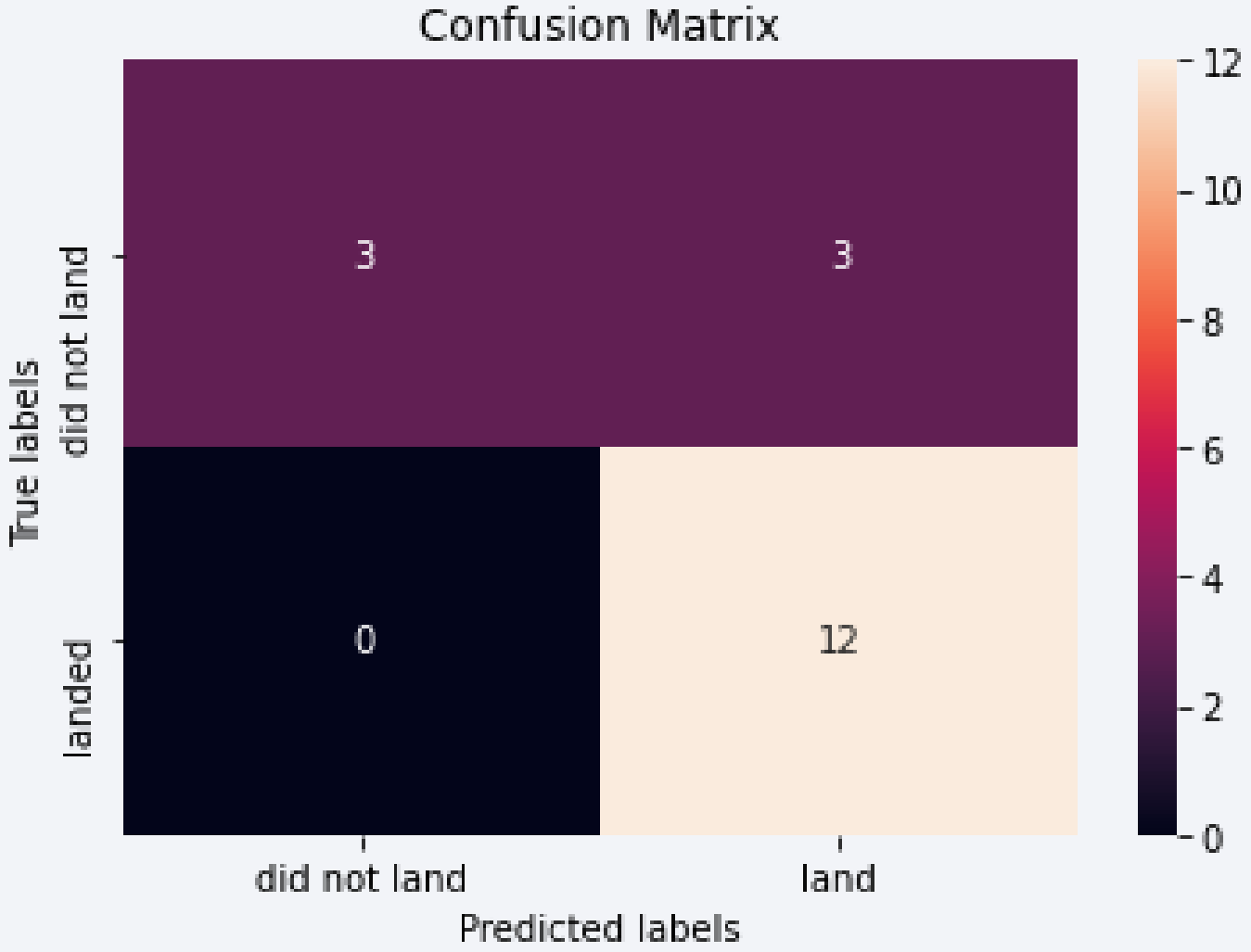
Find the method performs best:

```
predictors = [knn_cv, svm_cv, logreg_cv, tree_cv]
best_predictor = ""
best_result = 0
for predictor in predictors:

    predictor.score(X_test, Y_test)
```

```
0.8333333333333334
```

Confusion Matrix



Conclusions

This report shows all the activities carried out in this **Space X Falcon 9 project**. In these internships, we worked mainly on preventive and corrective maintenance of Space Rockets of various analysis, and after completing this internship and writing the final report, the following conclusions were reached:

The Space X Falcon 9 project is dedicated to the production of rockets material for open and private traveling, for the Internet, for independent national production companies, etc.

These teams had never received preventive maintenance regarding software maintenance, installed databases and hardware maintenance.

It was determined with this internship experience that all **Space Rocket** equipment requires preventive maintenance at least once every 3 months to ensure optimal operation throughout the year.

This lack of preventive maintenance resulted in the Space Rockets stopping working for an equivalent of 660 hours throughout the year, causing economic losses for the **Space X Falcon 9 project**.

Appendix

```
dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "BLUDB"            # e.g. "BLUDB"
dsn_port = "50000"                # e.g. "50000"
dsn_protocol = "TCPIP"            # i.e. "TCPIP"

%sql ibm_db_sa://kcq64325:0+bm77t47q9q5hn7@dashdb-txn-sbox-yp-dal09-04.services.dal.bluemix.net:50000/BLUDB
```

```
DB2/LINUXX8664
```

```
]: conda install -c conda-forge ipython-sql
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /opt/conda/envs/Python-3.8-main

  added / updated specs:
    - ipython-sql


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    certifi-2021.5.30          |   py38h578d9bd_0         141 KB  conda-forge
    ipython-sql-0.3.9          |py38h32f6830_1002          27 KB  conda-forge
    openssl-1.1.1k             |       h7f98852_0         2.1 MB  conda-forge
    prettytable-2.2.0          |     pyhd8ed1ab_0          23 KB  conda-forge
    python_abi-3.8             |           2_cp38           4 KB  conda-forge
    sqlparse-0.4.1             |     pyh9f0ad1d_0          34 KB  conda-forge
    ------------------------------------------------------------
                                       Total:         2.3 MB
```

Thank you!