Data Science In Winning Space Race

Kumar Durairaj

15-11-2024

Outline

- EXECUTIVE SUMMARY
- INTRODUCTION
- METHODOLOGY
- RESULTS
- CONCLUSION
- APPENDIX

Executive Summary

- **Problem statement**: SpaceX's Falcon 9 rocket launches sending spacecraft to the International Space Station, sending manned missions to Space with a cost of 62 million dollars; unlike other providers such as Blue Origin, Rocket Lab etc. cost upwards of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Sometimes the first stage does not land. Sometimes it will crash. Other times, Space X will sacrifice the first stage due to the mission parameters like payload, orbit, and customer. Instead of using rocket science to determine if the first stage will land successfully, we train a machine learning model and use public information by gathering information about Space X and creating dashboards, to predict if SpaceX reuse the first stage and the summary of methodology follows.

- **Summary of methodologies**
  - **Data collection methodology**:
    - **Data Collection** we work with SpaceX launch data that is gathered from an API, specifically the SpaceX REST API. This API give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
  - **Data Processing**
    - **Data wrangling** While observing the attributes Flight Number, booster etc. the column Outcome indicates if the first stage successfully landed. True ASDS / False ASDS mean the booster successfully / unsuccessfully landed to a drone These outcomes to be converted to Classes y (either 0 or 1). 0 is a bad outcome, that is, the booster did not land. 1 is a good outcome, that is, the booster did land. The variable Y will represent the classification variable that represents the outcome of each launch.
    - **Perform exploratory data analysis (EDA) using visualization and SQL** Performing EDA, observed that the success rate since 2013 has improved. different launch sites have different success rates. As combining attributes also gives us more information, incorporating those features help to determine what attributes are correlated with successful landings. The categorical variables will be converted using one hot encoding, preparing the data for a machine learning model that are used to predict if the first stage will successfully land.
  - **Interactive visual analytics using Folium and Plotly Dash:**
    - A dashboard application with the Python Plotly Dash package and Folium map support to find more insights from the SpaceX dataset easily than with static graphs. The interactive map and dashboard as interactive visual analytics enables users to find visual patterns more effectively and also to manipulate data in an interactive and real-time way.
  - **Predictive analysis using classification models**
  - At last**, b**uild a machine learning pipeline to predict if the first stage of the Falcon 9 lands successfully include: preprocessing, allowing us to standardize our data, and train_test_split, allowing us to split our data into training and testing data for training the model and perform Grid Search, allowing us to find the hyperparameters that allow a given algorithm to perform best. Using the best hyperparameter values, we determine the model with the best accuracy using the training data. Tested Logistic Regression, Support Vector machines, Decision Tree Classifier, and K-nearest neighbours models. Final output of the confusion matrix aids in deciding the best model.
- **Summary of Results**
- Increase in success rate since 2013
    - CCAFS LC-40 has a success rate of 60%, but if the mass is above 10,000 kg the success rate is 100%.
- KSC LC-39A and VAFB SLC 4E have a success rate of around 77%.

Introduction

- Project background and context

- During the present commercial space age, various companies viz. Virgin Galactic, Rocket Lab and Blue Origin manufactures are providing suborbital spaceflights, small satellite provider and sub-orbital and orbital reusable rockets respectively. Perhaps the most successful is SpaceX due to its accomplishments of sending spacecraft to the International Space Station, Starlink, a satellite internet constellation providing satellite Internet access, sending manned missions to Space etc. Furthermore, SpaceX can do the rocket launches with relatively inexpensive cost so that SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars while other providers cost upwards of 165 million dollars each. Spaces X's Falcon 9 launch like regular rockets. Much of the savings is because SpaceX can reuse or recover the first stage unlike other rocket providers. Stage two, or the second stage, helps bring the payload to orbit, but most of the work is done by the first stage and is much larger than the second stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. Even though SpaceX's Falcon 9 Can recover the first stage, Sometimes the first stage does not land or sometimes it will crash. Other times, Space X will sacrifice the first stage due to the mission parameters like payload, orbit, and customer.

- Thus, a real-world business problem is defined and formulated in which role of a data scientist working for a new rocket Company Space Y that would like to compete with SpaceX founded by Billionaire industrialist Allon Musk. Therefore, in this project, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch using various tools and methodologies.

- The problems to find answers are
  - Finding the price of each launch through gathering information about Space X and creating dashboards for working team.
  - Next, we train a machine learning model and use public information to predict if SpaceX will reuse the first stage successfully, instead of using rocket science.

Section 1

Methodology

Methodology

- **Data collection methodology**
  - **Data Collection** to work with SpaceX launch data that is gathered from an API, specifically the SpaceX REST API.
- **Data Processing**
  - **Data wrangling** to observe the attributes typically, the column Outcome i.e.True ASDS / False ASDS and other categorical variables and then to handle accordingly
  - **Exploratory data analysis (EDA)** using visualization and SQL to combine so that incorporating those features help to determine what attributes are correlated with successful landings for building a machine learning model.
- **Interactive visual analytics using Folium and Plotly Dash**, a dashboard application with the Python Plotly Dash package and Folium map support to find more insights from the SpaceX dataset
- **Predictive analysis using classification models**, at last**, b**uild a machine learning pipeline via standardize data, train_test_split, Grid Search, finding the hyperparameters and then plotting confusion matrix that aids in deciding the best model.

Data Collection

- Data sets collection
  - Request and parse the SpaceX launch data using the GET request
  - Filter the dataframe to only include Falcon 9 launches
  - Replace None values in the PayloadMass with the mean

- Data collection process (key phrases and flowcharts)
  - Keywords used in SpaceX's REST (Representational State of Resource) API calls
  - SpaceX provides an API (Application Programming Interface) that allows developers to access data about their launches, missions, vehicles, and more. REST calls are used to interact with this API. Here are some key phrases and parameters commonly used in SpaceX REST calls

  - https://github.com/kumard1963/Kumarhello-world/blob/main/1L1a_jupyter-labs-spacex-data-collection-api.ipynb

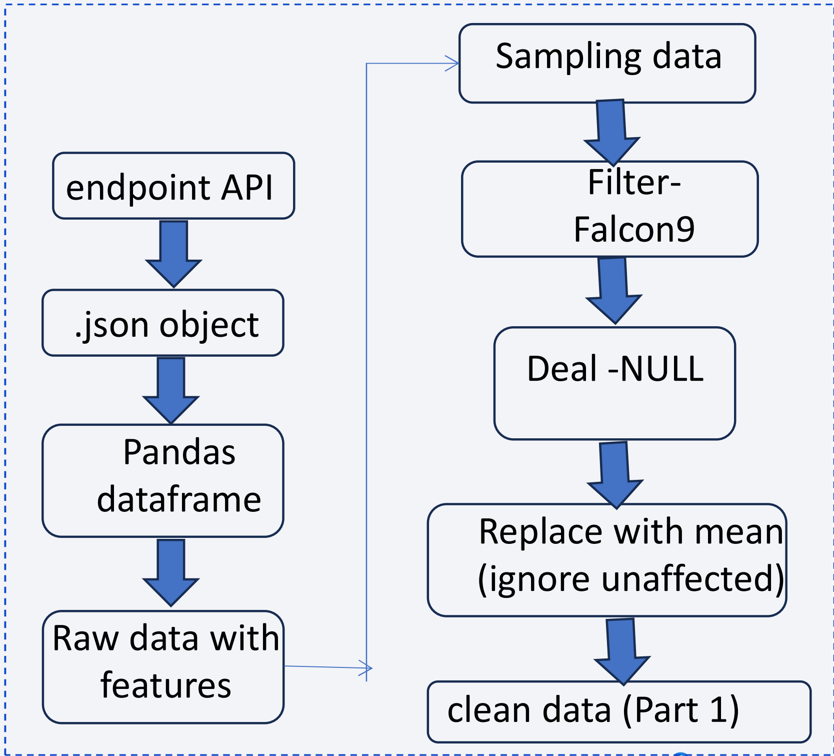
Data Collection – SpaceX API

:*Endpoints:*
1. *Launches*: `/launches` (https://api.spacexdata.com/v4/launches)
2. *Launch by ID*: `/launches/{id}` (Retrieves a specific launch by ID, v4/rockets
or v4/capsules or v4/cores or v4/launches/past)

Query Parameters:
1. *`limit`*: Specifies the number of results to return (e.g., `?limit=10`)
2. *`filter`*: Filters results by a specific field and value (e.g., filt = df['BoosterVersion']!= 'Falcon 1')
data_falcon9 = df.loc[filt]
Here filter creates a boolean Series where each entry is True if the corresponding list in the 'cores' column has a length of 1, and False otherwise data = data[data['cores'].map(len)==1]
Extract single value from the list data['cores'] = data['cores'].map(lambda x : x[0])
3 *NULL* Find missing values data_falcon9.isnull().sum()
4 *MEAN* Find mean plm_mean = data_falcon9['PayloadMass'].mean()
5. *REPLACE* Replace NULL with mean .replace(np.nan, plm_mean, inplace=True)
Common Response Fields:
1. *`BoosterVersion*: def getBoosterVersion(data)
2. *`launch_site`*: The site where the launch occurred def getLaunchSite(data)
3. *`status`*: The status of the launch (e.g., "success", "failure") from def getCoreData(data):

HTTP Methods:
1. *GET*: Retrieves data (static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json')

To use these key phrases, construct API calls like:
- `GET (response = requests.get(static_json_url) data = response.json() df = pd.json_normalize(data))
- `GET (Retrieve launch by ID, getBoosterVersion(data))



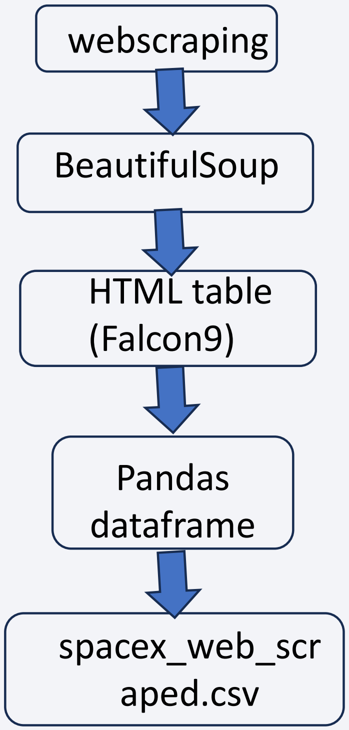
Data Collection - Scraping

Webscraping: Keyphrases
Get `response` object
1 HTTP GET method to request the Falcon9 Launch HTML page: response = requests.get(static_url)
2 Create a BeautifulSoup object from a response: soup = BeautifulSoup(response.content, 'html.parser')

helper functions to process web scraped HTML table
def date_time(table_cells):
def booster_version(table_cells):
def extract_column_from_header(row):

Function
1 Find_all function in the BeautifulSoup object, with element type `table: soup.find_all('table')
2`extract_column_from_header()` to extract column name one by one
3 Create a data frame by parsing the launch HTML tables
4 Create an empty dictionary with keys from the extracted column names
launch_dict= dict.fromkeys(column_names)
5 Fill up the `launch_dict` with launch records extracted from table rows
table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
6 Create a dataframe; df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
7 Export to csv: df.to_csv('spacex_web_scraped.csv', index=False)

- https://github.com/kumard1963/Kumarhello-world/blob/main/2L1b_jupyter-labs-webscraping.ipynb

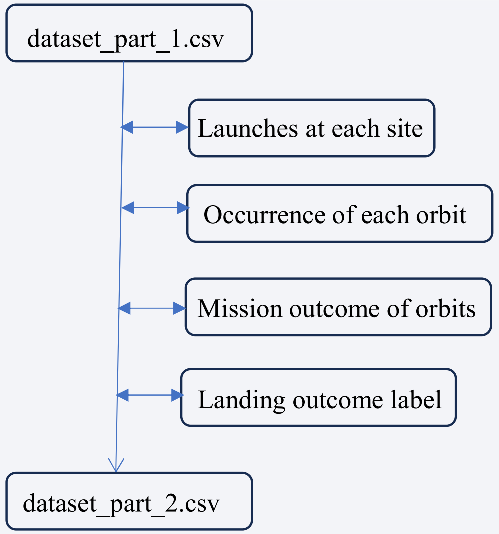


Data Wrangling

Description: In this lab, performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True and FALSE means successfully and unsuccessfully landed respectively.
Viz. TRUE Ocean & False Ocean, True RTLS (landed to a ground pad) & False RTLS and True ASDS (successfully landed on a drone ship) & False ASDS
In this lab we mainly converted those outcomes into Training Labels with 1 (TRUE) means the booster successfully landed 0 (FALSE) means it was unsuccessful

Key Phrases
method value_counts() on the column LaunchSite: df['LaunchSite'].value_counts()
Outcome: outcome in enumerate(landing_outcomes.keys())
.mean Find success rate: df["Class"].mean()

https://github.com/kumard1963/Kumarhello-world/blob/main/3L2_labs-jupyter-spacex-Data%20wrangling.ipynb



EDA with Data Visualization

Drill down into data, explore relationships, and identify patterns or anomalies interactively. Visualize the launch success yearly trend and create dummy variables to categorical columns

- Summary of plotted charts and why use of charts
  - Visualize the relationship between different parameters
    - Payload and flight number for how the Flight Number Payload variables would affect the launch outcome
    - Flight number and launch site to find the patterns in the scatter point plots
    - Payload mass and launch site to find relationship launching rockets with different masses
    - Orbits and class to find success rate
    - Flight number and orbit type to know outcomes at these orbits
  - Launch success rate and year to know success rate trend

  - https://github.com/kumard1963/Kumarhello-world/blob/main/4M2_L1_edadataviz.ipynb

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
- %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
- %sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
- %sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
- %sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
- %sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing_Outcome" = "Success (ground pad)";
- %sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing_Outcome" = "Success (drone ship)" AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
- %sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
- %sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);
- %sql SELECT substr(Date,7,4), substr(Date,4, 2), "Booster_Version", "Launch_Site", "Payload", "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing_Outcome" FROM SPACEXTBL WHERE substr(Date,7,4)='2015' AND "Landing_Outcome" = 'Failure (drone ship)';
- %sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDER BY Date DESC;
- https://github.com/kumard1963/Kumarhello-world/blob/main/M2_LSQL_jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Summary of map objects such as markers, circles, lines, etc. created and added to a folium map
  - Creation of Circles, Markers and lines added to a folium map
  - Use of objects
    - Launch locations as circles
    - Launch sites Markers on a map representing the success/failed launches for each site on the map
    - Line to measure distance between nearest coast to location and location to nearest railway station
- https://github.com/kumard1963/Kumarhello-world/blob/main/M3_InteVisAnalyt_Folium_lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

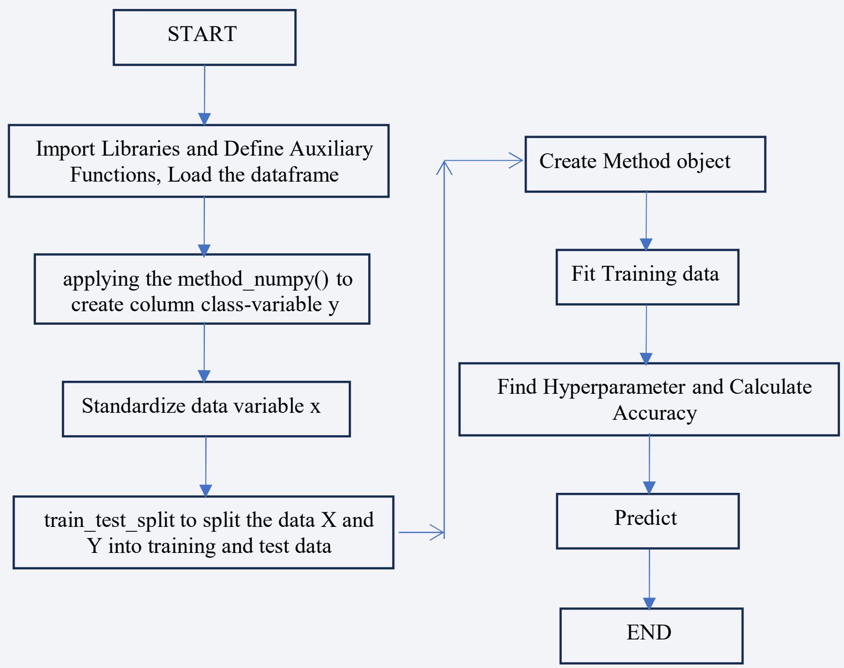
- Summary of plots/graphs and interactions added to a dashboard
  - SpaceX launch records dashboard
  - a pie chart visualizing launch success counts
  - a range slider to choose payload which in turn for identifying some visual patterns
  - payload-outcome scatter plot to observe how payload may be correlated with mission outcomes for selected site(s) including booster version
  - Plots and interactions
    - To perform interactive visual analytics on SpaceX launch data in real-time so that obtaining some insights
    - sites and launch outcomes, includes booster performance and lifting of varying payload masses
- https://github.com/kumard1963/Kumarhello-world/blob/main/spacex_dash_app_final_successratio.py

Predictive Analysis (Classification)

- **Summary**

  How built, evaluated, improved, and found the best performing classification model
  - standardize the data,
  - split the data into train and test set, For each model (SVM, Classification Trees, Logistic Regression and kNN),
  - find the best hyperparameters and calculate the accuracy of the test data,
  - find the method performs best using test data.
  - created a machine learning pipeline to predict if the first stage will land given the data from the preceding labs.
  - model development process as shown in flowchart

key phrases: Load data frame X= pd.read_csv(), numpy method Y = data['Class'].to_numpy(), standardize data X = transform.fit_transform(X), function train_test_split to split the data X and Y, train_test_split(X, Y, test_size=0.2, random_state=2), MMM_cv = GridSearchCV(M, parameters, cv=10), MMM_cv.fit(X_train, Y_train), print("tuned hpyerparameters :(best parameters) ",M_cv.best_params_)print("accuracy :",MMM_cv.best_score_), print("MMM test data accuracy :",MMM_cv.score(X_test, Y_test)), yhat=MMM_cv.predict(X_test)plot_confusion_matrix(Y_test,yhat)plt.show(), M – method (SVM, Classification Trees, Logistic Regression and kNN)



https://github.com/kumard1963/Kumarhello-world/blob/main/SpaceX_Machine%20Learning%20Prediction_Part_5_final.ipynb

https://github.com/kumard1963/Kumarhello-world/blob/main/SpaceX_ML%20Prediction_randmean_Part_5.ipynb

Results

Exploratory data analysis results

- Visualize the relationship between different parameters

Interactive analytics demo in screenshots

- Drill down into data, explore relationships, and identify patterns or anomalies interactively.
- visualize the launch success yearly trend
- create dummy variables to categorical column

Predictive analysis results

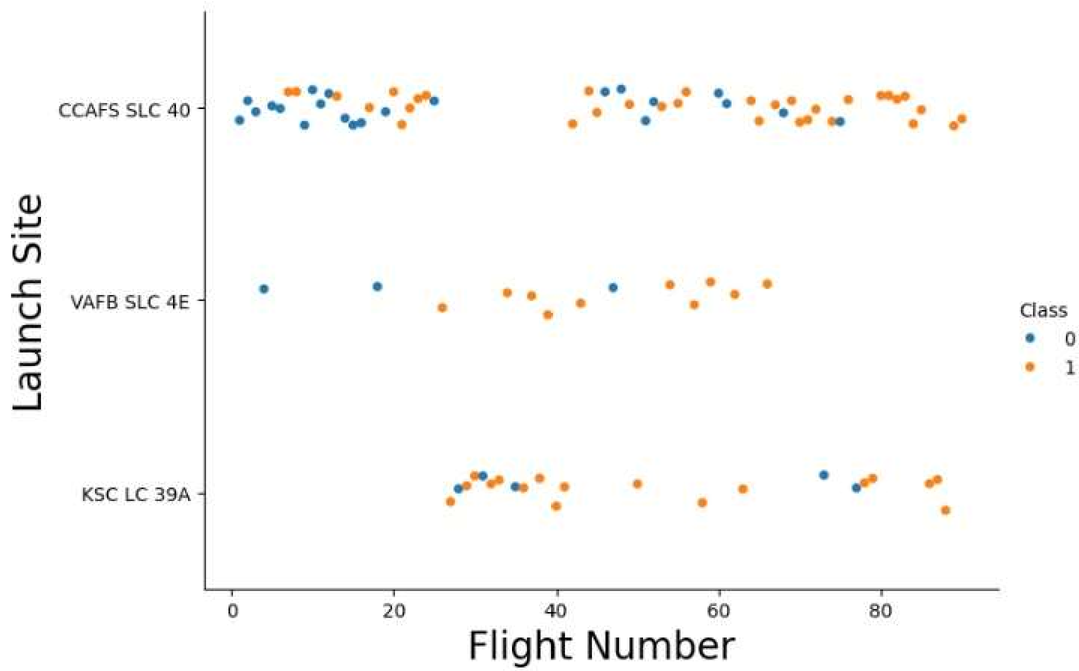
- Prediction using four models viz. SVM, Classification Trees, Logistic Regression and kNN and identifying the best performing model

Section 2

Insights drawn from EDA

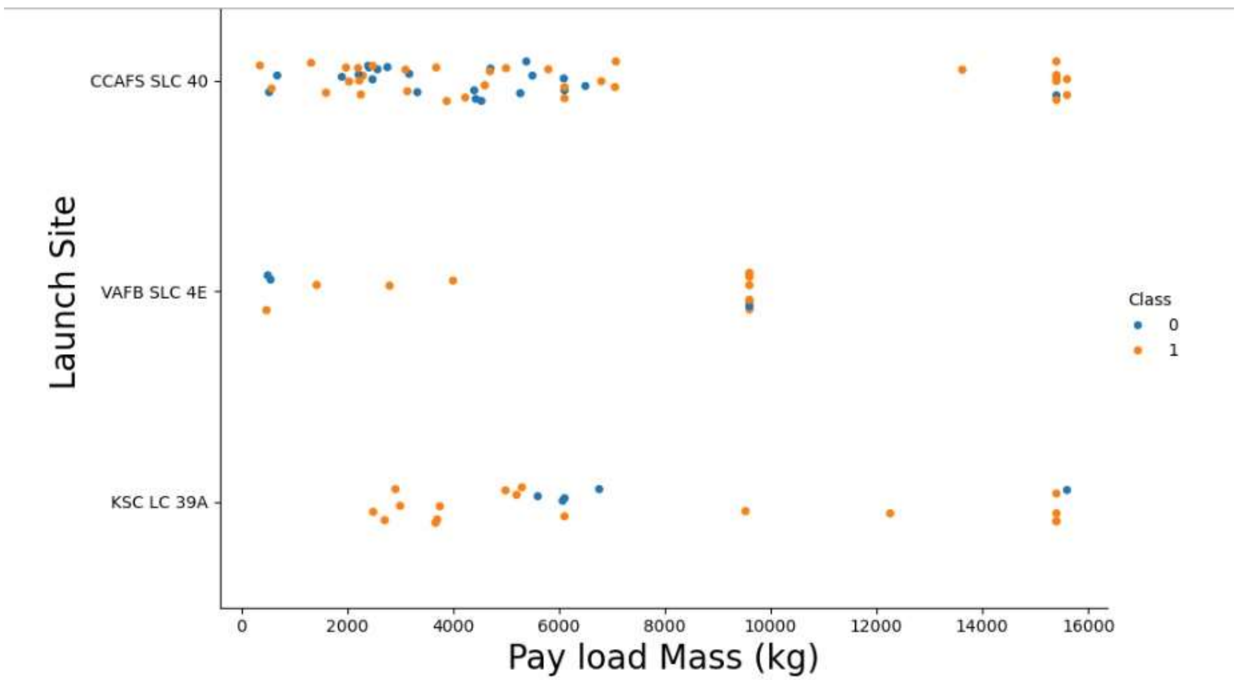
Flight Number vs. Launch Site

- The count of launches highest from CCAFS SLC 40. Less failure (22.7%) from KSC LC 39A



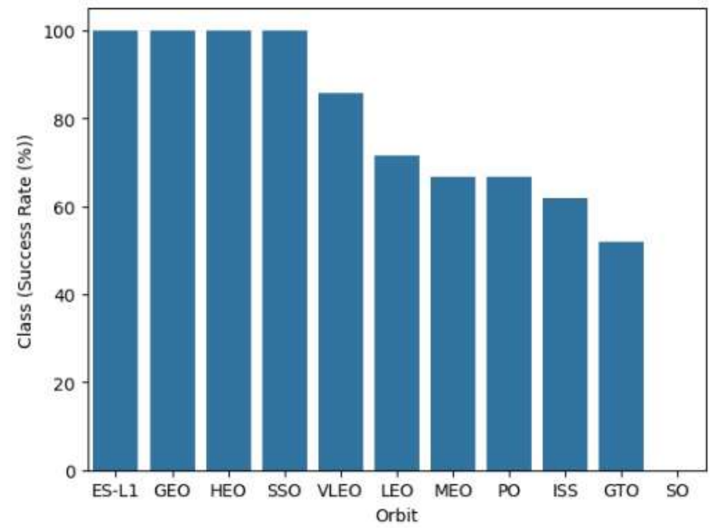
Payload vs. Launch Site

- High payload launch counts are maximum from CCAFS SLC 40 besides including highest launch counts



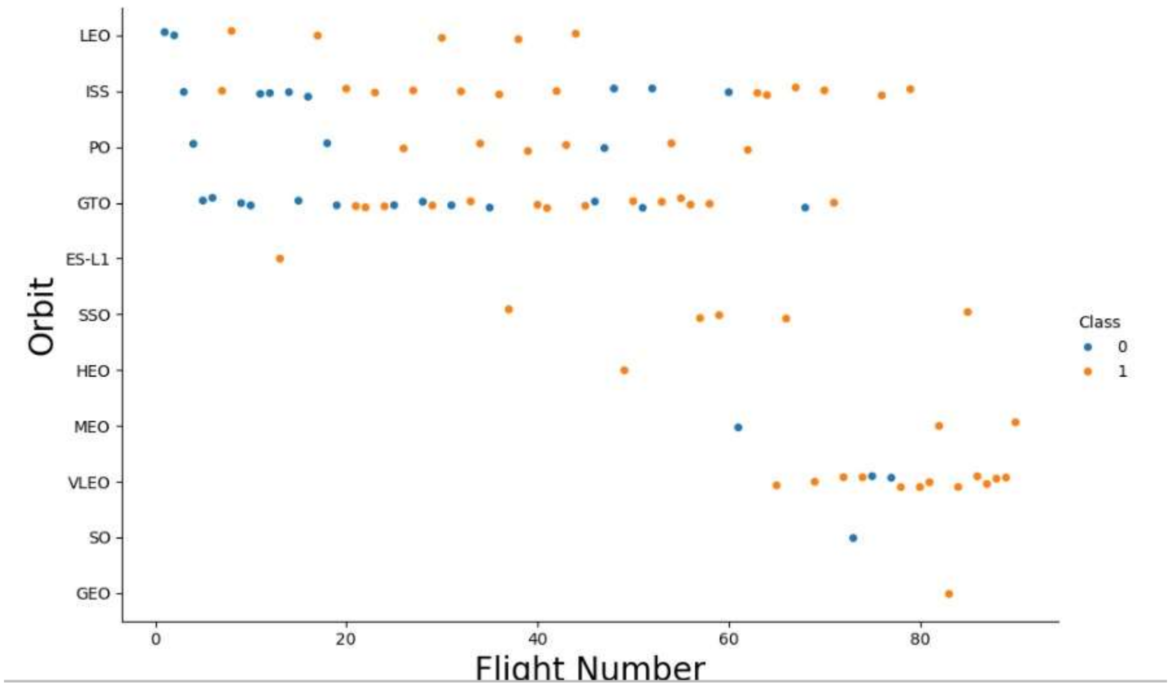
Success Rate vs. Orbit Type

- Success rate % maximum in orbit types ES-L1, GEO, HEO and SSO. Success rate in decreasing order from VLEO, MEO, PO, ISS and GTO. No information about SO type.



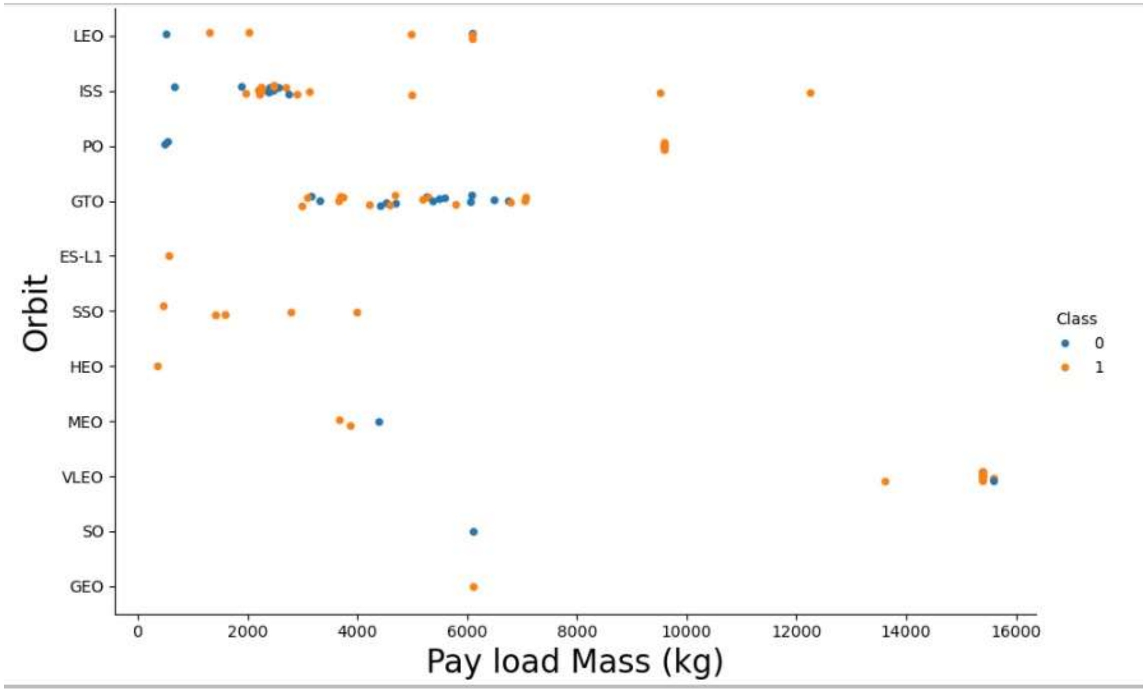
Flight Number vs. Orbit Type

- Total Number of flight numbers from ES-L1, GEO, HEO and SSO are 1, 1, 1 and 5 respectively with absence of failure.
- More flight numbers from GTO and ISS 27(13) and 21(8) respectively. Number of failures are within brackets.



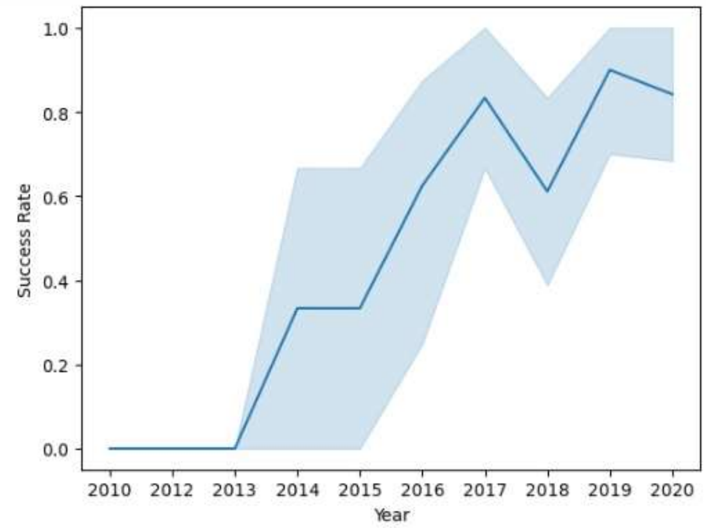
Payload vs. Orbit Type

- Counts of varying payloads maximum in orbit type GTO
- Counts maximum in Payload range 4000-6000
- Failure is very less for payload > 10000



Launch Success Yearly Trend

- Yearly trend of increase in success rate from 2013 onwards



All Launch Site Names

- Four unique launch sites and their names

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

Launch Site Names Begin with 'CCA'

- Payloads to Orbit type LEO(ISS) are small
- At the start Mission outcomes Successful. Initial two after no attempt in landing outcome.

Task 2

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

[32]:
```
%sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

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

[32]:

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

Total Payload Mass

- Total payload carried by boosters from NASA 45596kg

Task 3

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

[33]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

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

[33]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

Average Payload Mass by F9 v1.1

- Average payload mass (2534.7kg) carried by booster version F9 v1.1 for Customer MDA

Task 4

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

[34]:
```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version LIKE 'F9 v1.1%';
```

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

[34]:

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

First Successful Ground Landing Date

- Date of the first successful landing outcome on ground pad 22.12.2015

Task 5

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

Hint:Use min function

[39]:
```
%sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing_Outcome" = "Success (ground pad)";
```

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

[39]:

MIN(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Four numbers of F9 FT BXXXX series of boosters have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000 (**insight**).

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 * FROM 'SPACEXTBL'
%sql SELECT DISTINCT Booster_Version, Payload 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	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

Total Number of Successful and Failure Mission Outcomes

- Total number of successful mission outcomes 98+1+1(unclear payload status)
- Total number of failure mission outcomes 1(in flight)

Task 7

List the total number of successful and failure mission outcomes

```
[41]: %sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

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

[41]:

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

Boosters Carried Maximum Payload

- The booster which have carried the maximum payload mass F9 B5 Bxxxx.x series (insight)
- F9 B5 B1049 and B1051 carried thrice.

Task 8

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

```
[42]: %sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTBL);
```

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

[42]:

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

2015 Launch Records

- Two records on failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015
- Two are from CCAFS LC-40, but successful mission outcomes

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.

[31]:
```
"Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_Outcome", "Landing_Outcome" FROM SPACEXTBL WHERE substr(Date,0,5)='2015' AND "Landing_Outcome" = 'Failure (drone ship)'
```

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

[31]:

substr(Date,6,2)	substr(Date, 0, 5)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing_Outcome
01	2015	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
04	2015	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	Success	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.

```
[29]: %sql SELECT * FROM SPACEXTBL WHERE "Landing_Outcome" LIKE 'Success%' AND (Date BETWEEN '2010-06-04' AND '2017-03-20') ORDER BY Date DESC;
```

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

[29]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Success (drone ship)
2016-08-14	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-07-18	4:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Success (drone ship)
2016-05-06	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-04-08	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (drone ship)
2015-12-22	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

- 8 records 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
- 6 from CCAFS LC-40 between payload mass 2000-4000
- Highest payload mass (9600) from VAFB SLC 4E (Insight)

Section 3

Launch Sites Proximities Analysis

Launch site Locations Folium Map

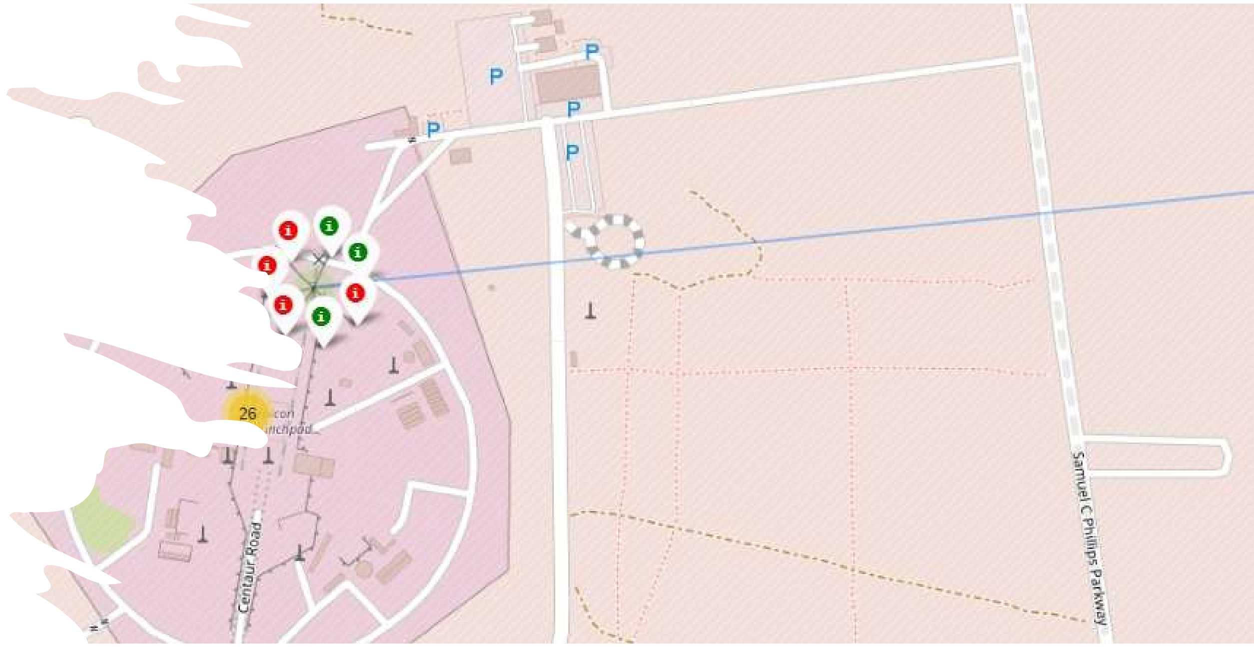
- Explored the generated folium map to include all launch sites' location markers on a global map
- Locations are with colored circles
- VAFB SLC 4E at west coast
- other combined at east coast
- While zooming,
- split into remaining three

Launch outcomes of locations

- launch outcomes on the map as green and red color-labeled markers representing success and failure of each launch respectively.
- Four locations as four spiral form of representation involving both east and west coast
- Totally 56 launches (west 10, east 46 (13+23+7))

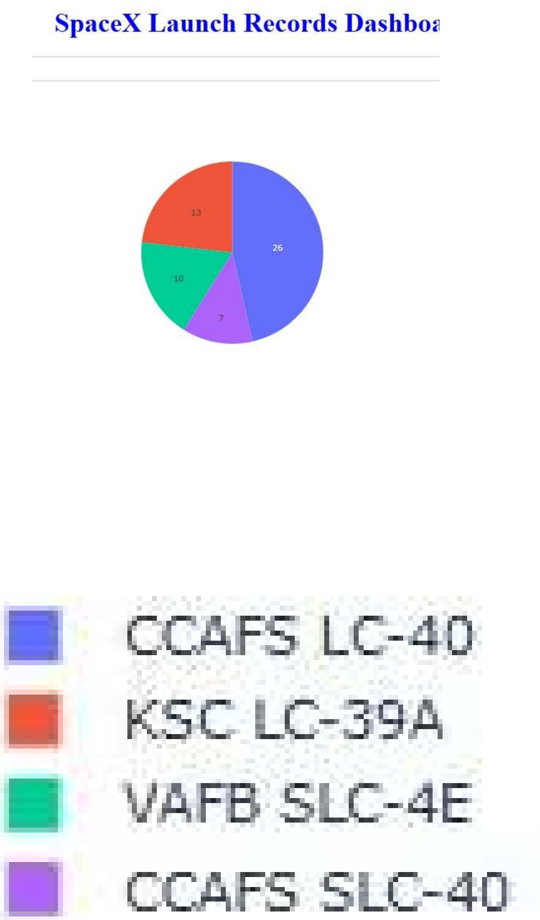
Proximities of launch sites

- At the top interactive display of latitude and longitude
- launch site of west VAFBSLC to its proximity railway – 1.3km
- launch site of east CCAFSSLC to its proximity coast – 0.86km
- Launch sites are nearer to coast and railways



Section 4

Build a Dashboard with Plotly Dash

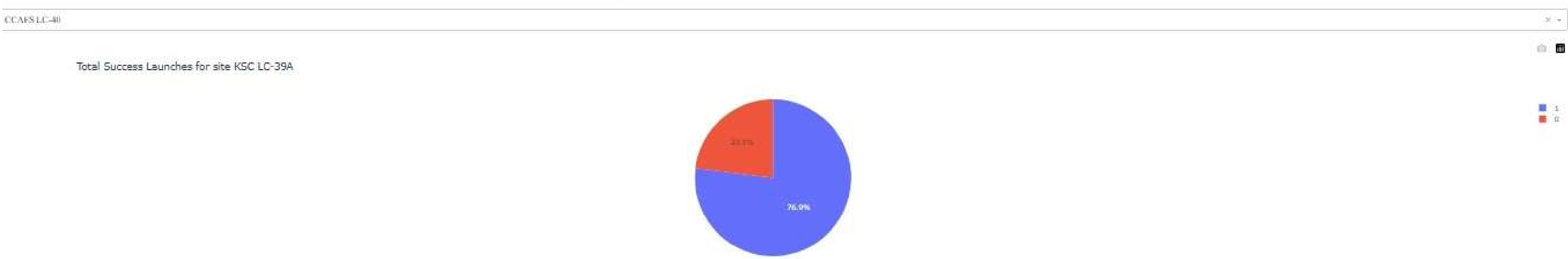


Launch success count for all sites, in a piechart

- Max. success counts from CCAFS LC-40
- Min. success counts from CCAFS SLC-40

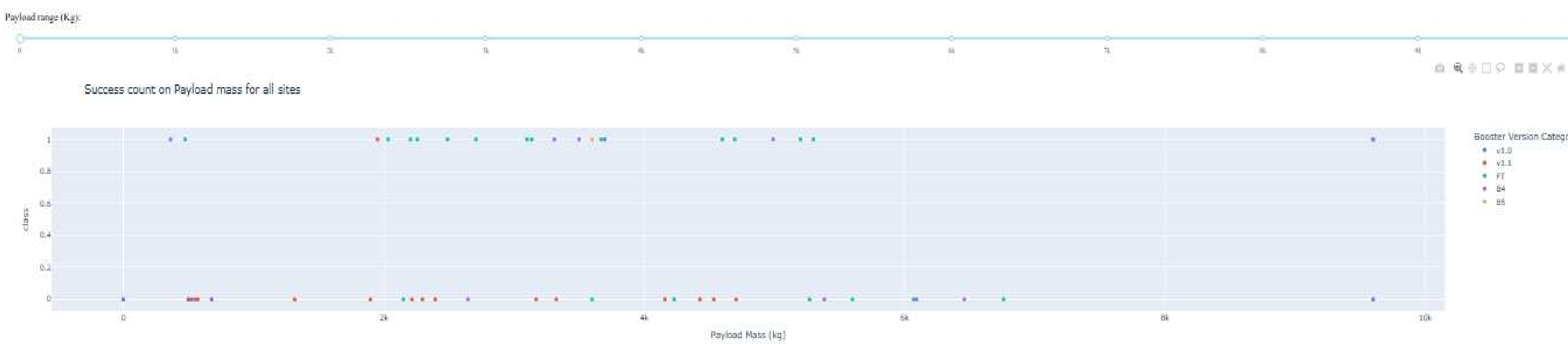
Dashboard findings piechart for the launch site - launch success ratio

Highest Launch Success ratio from KSC LC 39A is 76.9 , Other three are less and less



Dashboard **findings Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider**

- Dashboard findings on which payload range or booster version have the largest success rate
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.
- B4 lifted 9600kg successfully
- No failure of B5
- Success 58.8% in Payload 2000-4000kg
- V1.0 insignificant and max failure V1.1 besides one success

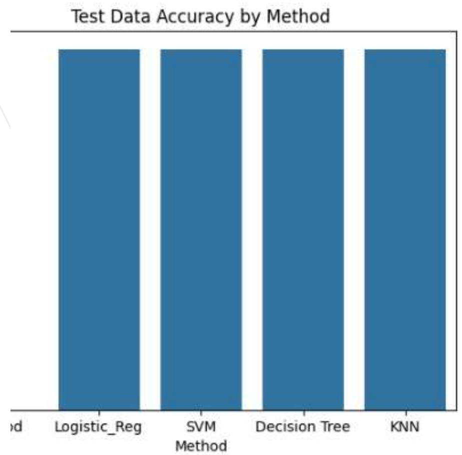


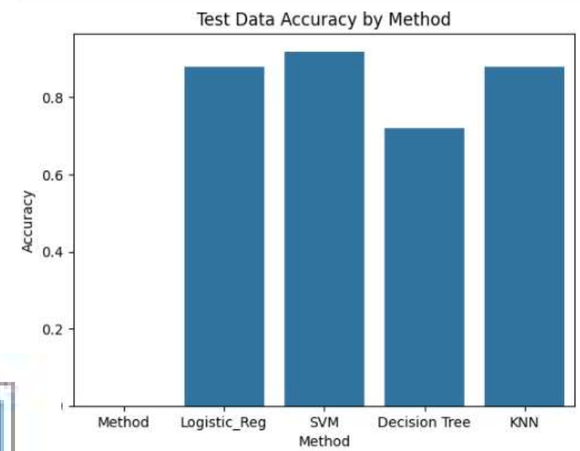
Section 5

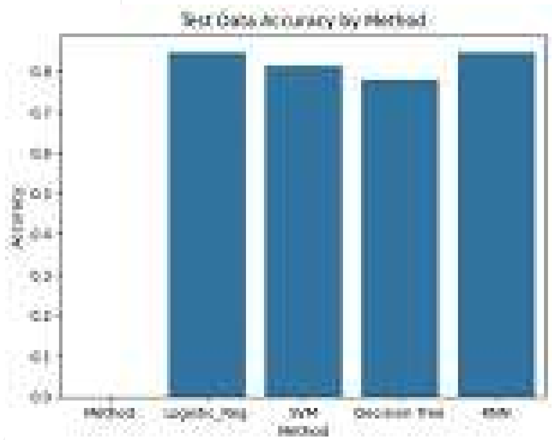
Predictive Analysis (Classification)

Classification Accuracy

- Visualize the built model accuracy for all built classification models, in a bar chart
    - All model has the same classification accuracy when test size (top)
    - **Innovation**
    - Highest in SVM when test size 0.27 and random state 8 (middle), signifies samples deficit.
    - implement all four models using for loop to iterate through different random states, calculated accuracy scores high in knn and log regression, and displayed the averaged confusion matrix (bottom)

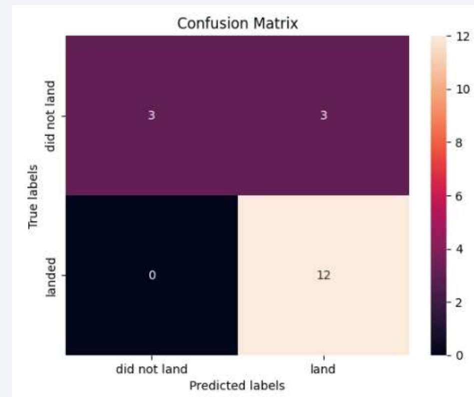


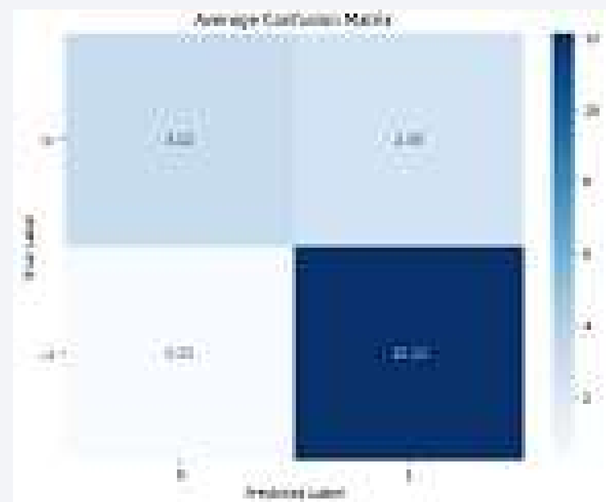




Confusion Matrix

- Examination of Confusion matrix of the models
    - Confusion matrix of all are same (top)
    - Examining the confusion matrix of all , we noticed that the problem is false positives.
    - True Postive - 12 (True label is landed, Predicted label is also landed)
    - False Postive - 3 (True label is not landed, Predicted label is landed)
  - Innovation
    - Confusion matrix items variation depends upon random state and sample size (bottom)





Conclusions

- Yearly trend of increase in success rate from 2013 onwards
- Two records in 2015, both mission outcomes successful whereas landing outcomes failure
- Date of the first successful landing outcome on ground pad 22.12.2015
- Four numbers of F9 FT BXXXX series of boosters have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
- Max. success counts from CCAFS LC-40
- The booster which have carried the maximum payload mass F9 B5 Bxxxx.x series.
- B4 lifted 9600kg successfully
- No failure of B5
- F9 B5 B1049 and B1051 carried thrice
- All four models showed same accuracy (0.833) as inferred from bar chart

Appendix

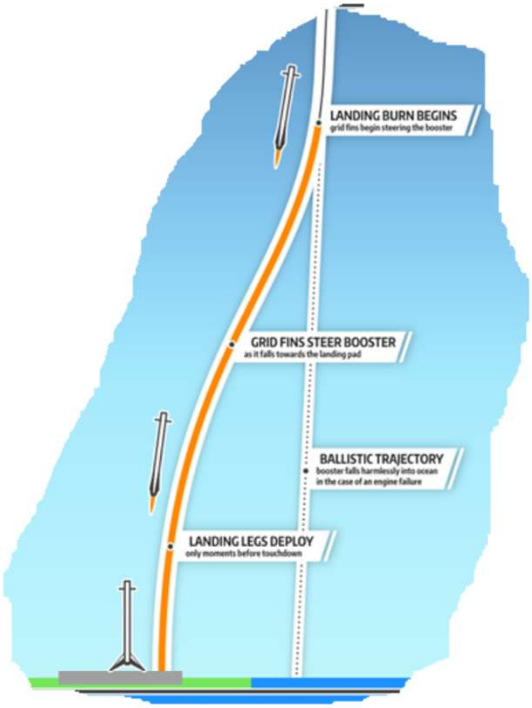
- **Code snippet for Test Data Accuracy by Methods as bar chart In Machine Learning (Support from Coursera Coach)**

import pandas as pdimport seaborn as snsimport matplotlib.pyplot as plt# Reset the index to convert the method names into a columntransposed_report_reset = transposed_report.reset_index()# Rename the columns for claritytransposed_report_reset.columns = ['Method', 'Test Data Accuracy']# Ensure the 'Method' column is of type stringtransposed_report_reset['Method'] = transposed_report_reset['Method'].astype(str)# Convert 'Test Data Accuracy' to numerictransposed_report_reset['Test Data Accuracy'] = pd.to_numeric(transposed_report_reset['Test Data Accuracy'], errors='coerce')# Now you can create the bar plotsns.barplot(data=transposed_report_reset, x='Method', y='Test Data Accuracy')plt.xlabel('Method')plt.ylabel('Accuracy')plt.title('Test Data Accuracy by Method')plt.show()

- **Plotly Dash interactive Board Count in pie chart instead percentage (Support from Ms Sathyapriya)**

```
@app.allback(Output(component_id='success-pie-chart', component_property='figure'),
           Input(component_id='site-dropdown', component_property='value'))
def get_pie_chart(entered_site):    filtered_df = spacex_df
   if entered_site == 'ALL':       # Group data to get counts for each launch site
       all_sites_df = filtered_df.groupby(['Launch Site', 'class']).size().reset_index(name='count')        # Create pie chart with counts instead of per
centages
       fig = px.pie(all_sites_df, values='count',       names='Launch Site',     title='Success Count for all launch sites')
       fig.update_traces(textinfo='value')  # Display counts directly, not percentages        return fig
  else:       # Filter for selected site and calculate counts
       site_df = filtered_df[filtered_df['Launch Site'] == entered_site]        site_counts = site_df.groupby(['class']).size().reset_index(name='count')
      fig = px.pie(site_counts, values='count',        names='class',                 title=f"Total Success Launches for site {entered_site}")
       fig.update_traces(textinfo='value')  # Show counts instead of percentages        return fig
```

Thank you



LANDING BURN BEGINS
grid fins begin steering the booster
GRID FINS STEER BOOSTER
as it falls towards the landing pad
BALLISTIC TRAJECTORY
booster falls harmlessly into ocean
in the case of an engine failure
LANDING LEGS DEPLOY
only moments before touchdown