Data Collection and Preprocessing Phase

Date	15 July 2024
Team ID	739760
Project Title	SpaceX Falcon 9 First Stage Landing Success Predictor
Maximum Marks	6 Marks

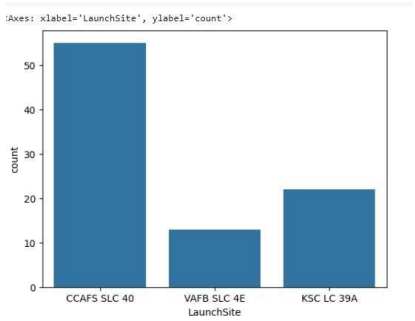
Data Exploration and Preprocessing Template

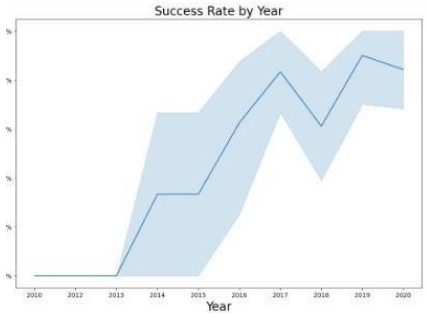
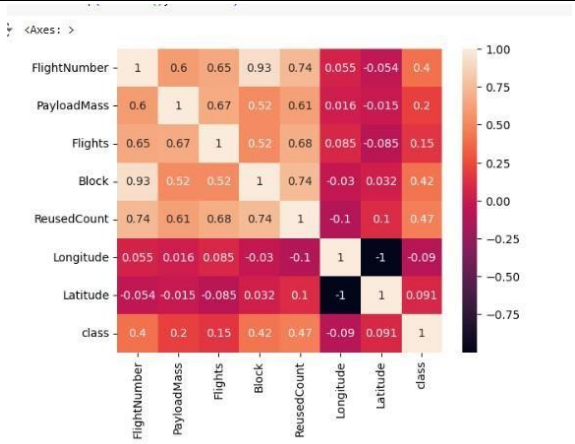
Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

Section	Description
Data Overview	statistics

```
df.describe()
```

	FlightNumber	PayloadMass	Flights	Block	ReusedCount	Longitude	Latitude	class
count	90.000000	90.000000	90.000000	90.000000	90.000000	90.000000	90.000000	90.000000
mean	45.500000	6104.959412	1.788889	3.500000	1.655556	-86.366477	29.449963	0.666667
std	26.124701	4694.671720	1.213172	1.595288	1.710254	14.149518	2.141306	0.474045
min	1.000000	350.000000	1.000000	1.000000	0.000000	-120.610829	28.561857	0.000000
25%	23.250000	2510.750000	1.000000	2.000000	0.000000	-80.603956	28.561857	0.000000
50%	45.500000	4701.500000	1.000000	4.000000	1.000000	-80.577366	28.561857	1.000000
75%	67.750000	8912.750000	2.000000	5.000000	3.000000	-80.577366	28.608058	1.000000
max	90.000000	15600.000000	6.000000	5.000000	5.000000	-80.577366	34.632093	1.000000

Univariate Analysis	

Bivariate Analysis	
Multivariate Analysis	
Outliers and Anomalies	Identification and treatment of outliers.

Data Preprocessing Code Screenshots

Loading Data	Code to load the dataset `df=pd.read_csv("/content/spacex_launch_data.csv")`

Handling Missing Data	`[8] df.isnull().sum()/df.count()*100`
	
	FlightNumber 0.000
	Date 0.000
	BoosterVersion 0.000
	PayloadMass 0.000
	Orbit 0.000
	LaunchSite 0.000
	Outcome 0.000
	Flights 0.000
	GridFins 0.000
	Reused 0.000
	Legs 0.000
	LandingPad 40.625
	Block 0.000
	ReusedCount 0.000
	Serial 0.000
	Longitude 0.000
	Latitude 0.000
	dtype: float64
	
	· Removing columns
	
	`[9] df.drop(["LandingPad"],axis=1,inplace=True)`

Data Transformation	`orbit_le=LabelEncoder().fit(df['Orbit'])`
	`df['Orbit']=orbit_le.transform(df['Orbit'])`
	`launchsite_le=LabelEncoder().fit(df['LaunchSite'])`
	`df['LaunchSite']=launchsite_le.transform(df['LaunchSite'])`
	`longitude_le=LabelEncoder().fit(df['Longitude'])`
	`df["Longitude"]=longitude_le.transform(df["Longitude"])`
	`latitude_le=LabelEncoder().fit(df['Latitude'])`
	`df["Latitude"]=latitude_le.transform(df["Latitude"])`
	`gridfins_le=LabelEncoder().fit(df['GridFins'])`
	`df["GridFins"]=gridfins_le.transform(df["GridFins"])`
	`legs_le=LabelEncoder().fit(df['Legs'])`
	`df['Legs']=legs_le.transform(df['Legs'])`
	
	`44] Scaler=preprocessing.StandardScaler()`
	`x_train=Scaler.fit_transform(x_train)`
	`x_test=Scaler.transform(x_test)`

Feature Engineering	`landing_outcomes=df["Outcome"].value_counts()`
	`landing_outcomes`
	
	Outcome
	True ASDS 41
	None None 19
	True RTLS 14
	False ASDS 6
	True Ocean 5
	False Ocean 2
	None ASDS 2
	False RTLS 1
	Name: count, dtype: int64
	
	`19] for i,outcome in enumerate(landing_outcomes.keys()):`
	`print(i,outcome)`
	
	0 True ASDS
	1 None None
	2 True RTLS
	3 False ASDS
	4 True Ocean
	5 False Ocean
	6 None ASDS
	7 False RTLS
	
	`20] bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])`
	`bad_outcomes`
	
	{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
	
	`21] landing_class=[0 if i in set(bad_outcomes) else 1 for i in df["Outcome"]]`
	
	`22] df["class"]=landing_class`
	`df[["class"]].head(8)`
	
	class

Save Processed Data	`filename="project.pkl"`
	`pickle.dump(lr,open(filename,'wb'))`
	
	`pickle.dump(orbit_le, open('orbit_le.pkl', 'wb'))`
	`pickle.dump(launchsite_le, open('launchsite_le.pkl', 'wb'))`
	
	`pickle.dump(Scaler, open('scaler.pkl', 'wb'))`