Experiment 1

Aim: Introduction to Data science and Data preparation using Pandas steps.

Theory:

Data science is the study of data that helps us derive useful insight for business decision making. Data Science is all about using tools, techniques, and creativity to uncover insights hidden within data. It combines math, computer science, and domain expertise to tackle real-world challenges in a variety of fields.

Data science involves these key steps:

- **Data Collection:** Gathering raw data from various sources, such as databases, sensors, or user interactions.
- **Data Cleaning:** Ensuring the data is accurate, complete, and ready for analysis.
- **Data Analysis:** Applying statistical and computational methods to identify patterns, trends, or relationships.
- **Data Visualization:** Creating charts, graphs, and dashboards to present findings clearly.
- **Decision-Making:** Using insights to inform strategies, create solutions, or predict outcomes.

Dataset Overview:

The dataset captures passenger satisfaction data from airline services and includes features related to demographic information, flight experience, and various service ratings. It consists of several columns, each providing crucial insight for customer experience analysis.

Below is a breakdown of the features:

1. **srno**
   Serial number for indexing rows; no analytical significance.

2. **id**
   Unique identifier for each passenger record.

3. **Age**
   Age of the passenger. Important for segmenting user experiences by age group.

4. **Flight Distance**
   Distance covered during the flight. Influences comfort and service satisfaction levels.

5. **Inflight wifi service**
   Rating for inflight Wi-Fi service, ranging from 0 (not rated) to 5 (excellent).

6. **Departure/Arrival time convenient**
   Passenger rating for the convenience of scheduled flight times.

7. **Ease of Online booking**
   How easily passengers were able to book their tickets online.

8. **Gate location**
   Rating of the gate location within the airport for convenience and accessibility.

9. **Food and drink**
   Passenger satisfaction with the quality of food and beverages served.

10. **Online boarding**
    Evaluation of the boarding process conducted online.

11. **Seat comfort**
    Passenger comfort level with seating during the flight.

12. **Inflight entertainment**
    Passenger rating of entertainment services available on the flight.

13. **On-board service**
    Evaluation of the service provided by staff on board.

14. **Leg room service**
    Satisfaction rating for legroom space.

15. **Baggage handling**
    Rating based on how well baggage was handled.

16. **Checkin service**
    Evaluation of the check-in experience at the airport.

17. **Inflight service**
    Rating of overall inflight services (excluding entertainment or Wi-Fi).

18. **Cleanliness**
 Cleanliness rating of the aircraft and facilities.

19. **Departure Delay in Minutes**
 Minutes of delay at the departure time. Important for reliability analysis.

20. **Arrival Delay in Minutes**
 Minutes of delay at the arrival. Strongly correlated with departure delay.

Problem Statement:

The airline dataset captures various aspects of a passenger's flight experience, including service ratings, flight logistics, and demographic attributes. The primary objective is to leverage Artificial Intelligence (AI) to enhance passenger satisfaction, optimize airline operations, and personalize services**.**

Key AI-driven goals include:

1. Passenger Satisfaction Prediction
 Build a classification model to predict whether a passenger is satisfied based on flight experience, service ratings, delays, and comfort levels.

2. Service Optimization
 Identify which service parameters (e.g., inflight entertainment, seat comfort, Wi-Fi quality) have the most impact on satisfaction using feature importance techniques.

3. Delay Impact Analysis
 Use regression or classification to evaluate how departure and arrival delays influence passenger satisfaction and identify thresholds where satisfaction drops.

4. Personalized Experience Recommendation
 Implement clustering (e.g., KMeans) to group passengers based on preferences and suggest personalized in-flight services for different segments.

5. Anomaly Detection
 Use unsupervised learning techniques to identify unusual patterns in flight services or feedback, which could highlight operational inefficiencies or critical

incidents.

Code:

1] This returns a tuple indicating the number of rows and columns in the DataFrame. This code prints "dataset info" and displays the DataFrame's structure including data types and non-null counts using `df.info()`.
2] It then prints "dataset description" and shows summary statistics like mean, standard deviation, and percentiles for numerical columns with `df.describe()`.

```
   srno      id  Gender      Customer Type  Age   Type of Travel     Class  \
0     0   70172    Male     Loyal Customer   13  Personal Travel  Eco Plus
1     1    5047    Male  disloyal Customer  180  Business travel  Business
2     2  110028  Female     Loyal Customer   26  Business travel  Business
3     3   24026  Female     Loyal Customer   25  Business travel  Business
4     4  119299    Male     Loyal Customer   61  Business travel  Business

   Flight Distance  Inflight wifi service  Departure/Arrival time convenient  \
0              460                      3                                  4
1              235                      3                                  2
2             1142                      2                                  2
3              562                      2                                  5
4              214                      3                                  3

   ...  Inflight entertainment  On-board service  Leg room service  \
0  ...                       5                 4                 3
1  ...                       1                 1                 5
2  ...                       5                 4                 3
3  ...                       2                 2                 5
4  ...                       3                 3                 4

   Baggage handling  Checkin service  Inflight service  Cleanliness  \
0                 4                4                 5            5
1                 3                1                 4            1
2                 4                4                 4            5
3                 3                1                 4            2
4                 4                3                 3            3
```

3] This code checks each column in the DataFrame for missing values and sums them up.

To check null values and remove maximum missing values

```
df.isnull().sum()
```

	0
ID	0
Plate Type	0
Primary Customer City	34
Primary Customer State	34
Registration Start Date	0
Registration Expiration Date	0
Registration Usage	0
Vehicle Type	0
Vehicle Weight	0
Vehicle Year	0
Vehicle Make	0
Vehicle Model	0
Vehicle Body	0
Primary Color	0
Vehicle Declared Gross Weight	0
Fuel Code	0
Vehicle Recorded GVWR	0
Vehicle Name	0
Type	0

4] This code replaces zeros in the DataFrame with NA values, then removes all rows containing any missing data to create a cleaned DataFrame.
It prints the cleaned DataFrame and confirms no missing values remain by summing NA entries for each column.

```python
df.replace(0, pd.NA, inplace=True)
df_cleaned = df.dropna()
print(df_cleaned.isna().sum())
```

```
Plate Type                    0
Primary Customer City         0
Registration Usage            0
Vehicle Type                  0
Vehicle Weight                0
Primary Color                 0
Vehicle Category              0
Z_Score                       0
Vehicle Weight Normalized     0
dtype: int64
```

Dropping unnecessary features

```python
df.drop(['ID', 'Primary Customer State', 'Registration Start Date', 'Registration Expiration Date',
         'Vehicle Year', 'Vehicle Make', 'Vehicle Model', 'Vehicle Body',
         'Vehicle Declared Gross Weight', 'Fuel Code', 'Vehicle Recorded GVWR',
         'Vehicle Name', 'Type'], axis=1, inplace=True)
```

This code removes duplicate rows from the cleaned DataFrame.

```python
[35] df = df_cleaned.drop_duplicates()
     df.head()
```

	Plate Type	Primary Customer City	Registration Usage	Vehicle Type	Vehicle Weight	Primary Color	Vehicle Category	Z_Score	Vehicle Weight Normalized
71	Passenger	SOUTHINGTON	Regular	SUV	5530	Gray	Light-Duty (Class 1-2)	-0.047744	0.376398
233	Passenger	GLASTONBURY	Regular	SUV	4774	Gray	Light-Duty (Class 1-2)	-0.533065	0.282484
276	Passenger	WALLINGFORD	Regular	Passenger	5200	Black	Light-Duty (Class 1-2)	-0.25959	0.335404
279	Passenger	WILLIMANTIC	Regular	Passenger	4200	Black	Light-Duty (Class 1-2)	-0.901549	0.21118
337	Passenger	WINDSOR	Regular	SUV	4800	Gray	Light-Duty (Class 1-2)	-0.516374	0.285714

```python
df.shape
```

```
(678, 20)
```

5] This code creates dummy data out of plate type as commercial and passenger. This helps to convert categorical data to numerical data and helps in analysis in the algorithm

6] The `head()` method to display the first ten rows of the DataFrame, to identify outliers manually we use the standardization approach (z score method). We find mean and standard deviation of the vehicle weight and calculate its z score; if its less than -3 or greater than 3 means its an outlier.

Detect outliers manually

```
df.head(10)
```

	Plate Type	Primary Customer City	Registration Usage	Vehicle Type	Vehicle Weight	Primary Color	Vehicle Category
71	Passenger	SOUTHINGTON	Regular	SUV	5530	Gray	Light-Duty (Class 1-2)
233	Passenger	GLASTONBURY	Regular	SUV	4774	Gray	Light-Duty (Class 1-2)
276	Passenger	WALLINGFORD	Regular	Passenger	5200	Black	Light-Duty (Class 1-2)
279	Passenger	WILLIMANTIC	Regular	Passenger	4200	Black	Light-Duty (Class 1-2)
337	Passenger	WINDSOR	Regular	SUV	4800	Gray	Light-Duty (Class 1-2)
416	Passenger	BRISTOL	Regular	SUV	5530	Silver	Light-Duty (Class 1-2)
420	Passenger	WEST SIMSBURY	Regular	Passenger	5600	Blue	Light-Duty (Class 1-2)
616	Passenger	NEW HAVEN	Regular	Passenger	3840	Black	Light-Duty (Class 1-2)
782	Passenger	GLASTONBURY	Regular	SUV	4800	Silver	Light-Duty (Class 1-2)
857	Passenger	GUILFORD	Regular	SUV	4709	Red	Light-Duty (Class 1-2)

```python
#By Z-score method
mean_vehicle_weight = df['Vehicle Weight'].mean()
std_vehicle_weight = df['Vehicle Weight'].std()

print(f"Mean of Vehicle Weight: {mean_vehicle_weight}")
print(f"Standard Deviation of Vehicle Weight: {std_vehicle_weight}")

# Calculate the Z-score for each vehicle weight
df['Z_Score'] = (df['Vehicle Weight'] - mean_vehicle_weight) / std_vehicle_weight

print(df[['Vehicle Weight', 'Z_Score']])

# Identify outliers based on the Z-score
outliers = df[df['Z_Score'].abs() > 3]
print(outliers)
```

```
Mean of Vehicle Weight: 5604.371681415929
Standard Deviation of Vehicle Weight: 1557.7315042063165
      Vehicle Weight   Z_Score
71              5530 -0.047744
233             4774 -0.533065
276             5200  -0.25959
279             4200 -0.901549
337             4800 -0.516374
...              ...       ...
52470           8450  1.826777
52609           8000  1.537896
52648           6450  0.542859
52670           8250  1.698385
52684           5300 -0.195394

[678 rows x 2 columns]
        Plate Type Primary Customer City Registration Usage Vehicle Type  \
15639  Combination             THOMASTON        Combination          SUV
23230  Combination             FAIRFIELD        Combination        Truck
32996  Combination           WILLIMANTIC        Combination          Van
50047  Combination               WINDSOR        Combination        Truck

      Vehicle Weight Primary Color         Vehicle Category   Z_Score
15639          10550         White   Light-Duty (Class 1-2)  3.174891
23230          10550         White   Light-Duty (Class 1-2)  3.174891
32996          10360        Orange  Medium-Duty (Class 3-6)  3.052919
50047          10500         White  Medium-Duty (Class 3-6)  3.142793
```

7] We normalize the data across the vehicle weights on a scale of 0 to 1.

8] This is our normalized data with respect to vehicle weight and can be used to analyse the vehicle weight distribution across its type and color.

```
[38] print("The first 10 rows of normalized data are :")
     df.head(10)
```

The first 10 rows of normalized data are :

	Plate Type	Primary Customer City	Registration Usage	Vehicle Type	Vehicle Weight	Primary Color	Vehicle Category	Z_Score	Vehicle Weight Normalized
71	Passenger	SOUTHINGTON	Regular	SUV	5530	Gray	Light-Duty (Class 1-2)	-0.047744	0.376398
233	Passenger	GLASTONBURY	Regular	SUV	4774	Gray	Light-Duty (Class 1-2)	-0.533065	0.282484
276	Passenger	WALLINGFORD	Regular	Passenger	5200	Black	Light-Duty (Class 1-2)	-0.25959	0.335404
279	Passenger	WILLIMANTIC	Regular	Passenger	4200	Black	Light-Duty (Class 1-2)	-0.901549	0.21118
337	Passenger	WINDSOR	Regular	SUV	4800	Gray	Light-Duty (Class 1-2)	-0.516374	0.285714
416	Passenger	BRISTOL	Regular	SUV	5530	Silver	Light-Duty (Class 1-2)	-0.047744	0.376398
420	Passenger	WEST SIMSBURY	Regular	Passenger	5600	Blue	Light-Duty (Class 1-2)	-0.002806	0.385093
616	Passenger	NEW HAVEN	Regular	Passenger	3840	Black	Light-Duty (Class 1-2)	-1.132655	0.16646
782	Passenger	GLASTONBURY	Regular	SUV	4800	Silver	Light-Duty (Class 1-2)	-0.516374	0.285714
857	Passenger	GUILFORD	Regular	SUV	4709	Red	Light-Duty (Class 1-2)	-0.574792	0.27441

Conclusion:

Thus we have pre-processed our dataset by various techniques mentioned and can be used for analysis and trained under algorithms for predictions.