Data Preparation and Exploration: A Detailed Guide with Examples and Python Code

---

1. Introduction to Data Preparation

Data preparation is the process of cleaning, transforming, and organizing raw data into a usable format for analysis, modeling, or reporting. It is a critical step in the data science workflow because high-quality data leads to more accurate and reliable insights. Poor data quality can result in misleading conclusions, poor model performance, and wasted effort.

---

2. Steps in Data Preparation

Step 1: Data Collection

Collecting raw data from multiple sources such as:

- Databases (SQL, NoSQL)
- APIs (RESTful services)
- CSV/Excel files
- Web scraping

Python Example:

```
import pandas as pd
# Load from CSV
df = pd.read_csv('sales_data.csv')
# Load from Excel
df_excel = pd.read_excel('customers.xlsx')
```

Output Preview:

```
   customer_id   age   income gender  online_purchase  store_purchase
0            1  28.0  45000.0   Male            250.0           100.0
1            2  34.0  52000.0   Female          300.0           150.0
2            3  45.0  61000.0   Male            400.0           200.0
```

Step 2: Data Cleaning

a) Handling Missing Values

```
# Fill missing prices with mean
df['price'].fillna(df['price'].mean(), inplace=True)
```

Output:

```
df['price'].isnull().sum()
0
```

b) Removing Duplicates

```
# Remove duplicated records
df.drop_duplicates(inplace=True)
```

Output:

```
df.duplicated().sum()
0
```

c) Correcting Data Types and Formats

```
# Convert date column to datetime
df['date'] = pd.to_datetime(df['date'])
```

Output:

```
df.dtypes
customer_id              int64
age                    float64
income                 float64
gender                  object
online_purchase        float64
store_purchase         float64
price                  float64
date           datetime64[ns]
dtype: object
```

d) Handling Outliers

```
# Remove outliers using IQR
Q1 = df['income'].quantile(0.25)
Q3 = df['income'].quantile(0.75)
IQR = Q3 - Q1
filtered_df = df[(df['income'] >= Q1 - 1.5 * IQR) & (df['income'] <= Q3 + 1.5 *
IQR)]
```

Output:

```
filtered_df.shape
(95, 7)
```

Step 3: Data Transformation

a) Standardization and Normalization

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
df[['age', 'income']] = scaler.fit_transform(df[['age', 'income']])
```

Output:

```
df[['age', 'income']].head()
        age    income
0 -1.214678 -0.892450
1 -0.542135 -0.372156
2  0.421251  0.653435
```

b) Encoding Categorical Variables

```
# Binary encoding
df['gender'] = df['gender'].map({'Male': 0, 'Female': 1})
```

Output:

```
df['gender'].value_counts()
0    52
```

```
1    48
Name: gender, dtype: int64
```

c) Feature Engineering

```
# Create a new feature
df['total_purchase'] = df['online_purchase'] + df['store_purchase']
```

Output Preview:

```
df[['online_purchase', 'store_purchase', 'total_purchase']].head()
   online_purchase  store_purchase  total_purchase
0            250.0           100.0           350.0
1            300.0           150.0           450.0
2            400.0           200.0           600.0
```

Step 4: Data Integration

```
merged_df = pd.merge(df_sales, df_customers, on='customer_id')
```

Output:

```
merged_df.shape
(100, 10)
```

Step 5: Data Reduction

a) Feature Selection

```
df = df[['customer_id', 'gender', 'age', 'income', 'total_purchase']]
```

Output:

```
df.columns
Index(['customer_id', 'gender', 'age', 'income', 'total_purchase'],
dtype='object')
```

b) Dimensionality Reduction (PCA)

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
reduced_data = pca.fit_transform(df[['age', 'income', 'total_purchase']])
```

Output:

```
reduced_data.shape
(100, 2)
```

---

3. Data Exploration (EDA)

3.1 Descriptive Statistics

```
print(df.describe())
```

Output:

```
        customer_id     gender         age      income  total_purchase
count    100.000000  100.00000  100.00000  100.00000       100.000000
mean      50.500000    0.48000    0.00000    0.00000       425.000000
std       29.011492    0.50253    1.00000    1.00000       100.000000
```

3.2 Visual Exploration

(رسوم بيانية لا تُعرض كنص، ولكن تُنتج مباشرة عند تنفيذ الكود.)

---

6. Output of Final Workflow Checks

```
print(df.head())
```

Output Preview:

```
   customer_id  gender       age    income  total_purchase
0            1       0 -1.214678 -0.892450           350.0
```

```
1            2       1 -0.542135 -0.372156           450.0
2            3       0  0.421251  0.653435           600.0
```

```
print(df[['PCA1', 'PCA2']].describe())
```

Output:

```
             PCA1        PCA2
count  100.000000  100.00000
mean     0.000000    0.00000
std      1.000000    1.00000
```

End of Document