DSPL Assignment 02

Name – Shrutik Gupta

Batch – T12

Roll No. – 32

Aim: Data preparation and visualization using NumPy and Pandas.

Theory:

Data preparation and visualization are essential components of any Data Science or Machine Learning project. They ensure that the data is clean, meaningful, and ready for analysis or training models.

---

1. Data Preparation

Data preparation refers to the process of cleaning, organizing, and transforming raw data into a usable format. It includes several key steps:

Why is Data Preparation Important?

- **Ensures Data Quality**: Raw data is often incomplete, inconsistent, or contains errors (e.g., missing values, outliers). Poor-quality data can lead to inaccurate models and unreliable insights.
- **Enhances Model Performance**: Well-prepared data ensures that the algorithms learn patterns effectively, resulting in better model accuracy.
- **Reduces Noise**: Irrelevant or redundant features are removed, reducing the complexity of the model.
- **Makes Data Usable**: Converts raw, messy data into a format that machine learning algorithms can process.

Key Steps in Data Preparation

1. **Data Collection**:
    - Gather data from multiple sources like databases, APIs, sensors, etc.
2. **Data Cleaning**:

- Handle missing values (e.g., imputation, deletion).
- Remove duplicates and inconsistencies.
- Identify and address outliers.

3. **Data Transformation**:
   - Standardize or normalize data to bring features to a similar scale.
   - Encode categorical variables using techniques like one-hot encoding.
4. **Feature Engineering**:
   - Create new features or modify existing ones to improve model performance.
   - Perform dimensionality reduction (e.g., PCA) to reduce feature space.
5. **Data Splitting**:
   - Divide the dataset into training, validation, and test sets.

---

2. Data Visualization

Data visualization involves representing data graphically to uncover patterns, trends, and insights. It is a critical step in exploratory data analysis (EDA).

Why is Data Visualization Important?

- **Understand the Data**:
  - Identify trends, correlations, and distributions.
  - Spot anomalies, patterns, or outliers that might not be evident in raw data.
- **Communicate Insights**:
  - Convey findings effectively to stakeholders using intuitive visual representations.
- **Model Understanding**:
  - Help understand the relationships between features, enabling better feature selection.
- **Improves Decision-Making**:

- Provides a clear, concise view of the data to support better decisions.

Common Data Visualization Techniques

1. **Univariate Analysis**:
   - Histograms, box plots, and bar charts to analyze individual variables.
2. **Bivariate Analysis**:
   - Scatter plots, line charts, and correlation heatmaps to explore relationships between two variables.
3. **Multivariate Analysis**:
   - Pair plots, parallel coordinates plots, and 3D plots to visualize interactions between multiple variables.
4. **Time-Series Data**:
   - Line charts and area charts to observe trends over time.
5. **Categorical Data**:
   - Pie charts, bar charts, and stacked bar charts for category-wise distribution.
6. **Geospatial Data**:
   - Maps for visualizing data across geographical regions.

Importance in Machine Learning Applications

1. **EDA (Exploratory Data Analysis)**:
   - Visualization tools help understand the dataset before modeling. For example:
     - Identifying correlated features that might cause multicollinearity.
     - Observing class imbalance in target variables.
2. **Feature Selection**:
   - Helps identify which features are relevant for building a predictive model.

3. **Model Debugging**:
    - Visualizations such as learning curves and feature importance plots assist in diagnosing and improving model performance.
4. **Model Interpretation**:
    - Tools like SHAP or LIME provide visual explanations of model predictions, building trust and transparency.

In [4]:
```python
import pandas as pd
```

In [6]:
```python
df1 = pd.read_csv('C:/Users/lab1002/Downloads/archive/deliveries.csv')
df2 = pd.read_csv('C:/Users/lab1002/Downloads/archive/matches.csv')
```

In [7]:
```python
print(df1.head())
print(df2.head())
```

```
   match_id  inning           batting_team                 bowling_team  over  \
0    335982       1  Kolkata Knight Riders  Royal Challengers Bangalore     0
1    335982       1  Kolkata Knight Riders  Royal Challengers Bangalore     0
2    335982       1  Kolkata Knight Riders  Royal Challengers Bangalore     0
3    335982       1  Kolkata Knight Riders  Royal Challengers Bangalore     0
4    335982       1  Kolkata Knight Riders  Royal Challengers Bangalore     0

   ball       batter   bowler  non_striker  batsman_runs  extra_runs  \
0     1   SC Ganguly  P Kumar  BB McCullum             0           1
1     2  BB McCullum  P Kumar   SC Ganguly             0           0
2     3  BB McCullum  P Kumar   SC Ganguly             0           1
3     4  BB McCullum  P Kumar   SC Ganguly             0           0
4     5  BB McCullum  P Kumar   SC Ganguly             0           0

   total_runs extras_type  is_wicket player_dismissed dismissal_kind fielder
0           1     legbyes          0              NaN            NaN     NaN
1           0         NaN          0              NaN            NaN     NaN
2           1       wides          0              NaN            NaN     NaN
3           0         NaN          0              NaN            NaN     NaN
4           0         NaN          0              NaN            NaN     NaN
       id   season        city        date match_type player_of_match  \
0  335982  2007/08   Bangalore  2008-04-18     League     BB McCullum
1  335983  2007/08  Chandigarh  2008-04-19     League      MEK Hussey
2  335984  2007/08       Delhi  2008-04-19     League     MF Maharoof
3  335985  2007/08      Mumbai  2008-04-20     League      MV Boucher
4  335986  2007/08     Kolkata  2008-04-20     League       DJ Hussey

                                        venue                        team1  \
0                       M Chinnaswamy Stadium  Royal Challengers Bangalore
1  Punjab Cricket Association Stadium, Mohali              Kings XI Punjab
2                            Feroz Shah Kotla             Delhi Daredevils
3                            Wankhede Stadium               Mumbai Indians
4                                Eden Gardens        Kolkata Knight Riders

                         team2                  toss_winner toss_decision  \
0        Kolkata Knight Riders  Royal Challengers Bangalore         field
1          Chennai Super Kings          Chennai Super Kings           bat
2             Rajasthan Royals             Rajasthan Royals           bat
3  Royal Challengers Bangalore               Mumbai Indians           bat
4              Deccan Chargers              Deccan Chargers           bat

                        winner   result  result_margin  target_runs  \
0        Kolkata Knight Riders     runs          140.0        223.0
1          Chennai Super Kings     runs           33.0        241.0
2             Delhi Daredevils  wickets            9.0        130.0
3  Royal Challengers Bangalore  wickets            5.0        166.0
4        Kolkata Knight Riders  wickets            5.0        111.0

   target_overs super_over method    umpire1         umpire2
0          20.0          N    NaN  Asad Rauf     RE Koertzen
1          20.0          N    NaN  MR Benson      SL Shastri
2          20.0          N    NaN  Aleem Dar  GA Pratapkumar
3          20.0          N    NaN   SJ Davis       DJ Harper
4          20.0          N    NaN  BF Bowden     K Hariharan
```

1. Find the null values in each column

In [8]:
```python
print("Null values in deliveries.csv:")
print(df1.isnull().sum())

print("\nNull values in matches.csv:")
print(df2.isnull().sum())
```

```
Null values in deliveries.csv:
match_id                 0
inning                   0
batting_team             0
bowling_team             0
over                     0
ball                     0
batter                   0
bowler                   0
non_striker              0
batsman_runs             0
extra_runs               0
total_runs               0
extras_type         246795
is_wicket                0
player_dismissed    247970
dismissal_kind      247970
fielder             251566
dtype: int64

Null values in matches.csv:
id                    0
season                0
city                 51
date                  0
match_type            0
player_of_match       5
venue                 0
team1                 0
team2                 0
toss_winner           0
toss_decision         0
winner                5
result                0
result_margin        19
target_runs           3
target_overs          3
super_over            0
method             1074
umpire1               0
umpire2               0
dtype: int64
```

2. Merge the two datasets

In [9]:
```python
merged_df = pd.merge(df1, df2, left_on='match_id', right_on='id', how='inner')
```

3. Find the null values in each column again

In [10]:
```python
print("Null values after merging:")
print(merged_df.isnull().sum())
```

```
Null values after merging:
match_id                 0
inning                   0
batting_team             0
bowling_team             0
over                     0
ball                     0
batter                   0
bowler                   0
non_striker              0
batsman_runs             0
extra_runs               0
total_runs               0
extras_type         246795
is_wicket                0
player_dismissed    247970
dismissal_kind      247970
fielder             251566
id                       0
season                   0
city                 12397
date                     0
match_type               0
player_of_match        490
venue                    0
team1                    0
team2                    0
toss_winner              0
toss_decision            0
winner                 490
result                   0
result_margin         4124
target_runs            309
target_overs           309
super_over               0
method              257274
umpire1                  0
umpire2                  0
dtype: int64
```

4. Check info and shape after merging

In [11]:
```python
print("Information of merged dataset:")
print(merged_df.info())

print("\nShape of merged dataset:")
print(merged_df.shape)
```

```
Information of merged dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 260920 entries, 0 to 260919
Data columns (total 37 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   match_id          260920 non-null  int64
 1   inning            260920 non-null  int64
 2   batting_team      260920 non-null  object
 3   bowling_team      260920 non-null  object
 4   over              260920 non-null  int64
 5   ball              260920 non-null  int64
 6   batter            260920 non-null  object
 7   bowler            260920 non-null  object
 8   non_striker       260920 non-null  object
 9   batsman_runs      260920 non-null  int64
 10  extra_runs        260920 non-null  int64
 11  total_runs        260920 non-null  int64
 12  extras_type       14125 non-null   object
 13  is_wicket         260920 non-null  int64
 14  player_dismissed  12950 non-null   object
 15  dismissal_kind    12950 non-null   object
 16  fielder           9354 non-null    object
 17  id                260920 non-null  int64
 18  season            260920 non-null  object
 19  city              248523 non-null  object
 20  date              260920 non-null  object
 21  match_type        260920 non-null  object
 22  player_of_match   260430 non-null  object
 23  venue             260920 non-null  object
 24  team1             260920 non-null  object
 25  team2             260920 non-null  object
 26  toss_winner       260920 non-null  object
 27  toss_decision     260920 non-null  object
 28  winner            260430 non-null  object
 29  result            260920 non-null  object
 30  result_margin     256796 non-null  float64
 31  target_runs       260611 non-null  float64
 32  target_overs      260611 non-null  float64
 33  super_over        260920 non-null  object
 34  method            3646 non-null    object
 35  umpire1           260920 non-null  object
 36  umpire2           260920 non-null  object
dtypes: float64(3), int64(9), object(25)
memory usage: 73.7+ MB
None

Shape of merged dataset:
(260920, 37)
```

5. Get the columns in the right datatype

In [12]:
```python
merged_df['date'] = pd.to_datetime(merged_df['date'])
categorical_cols = ['batting_team', 'bowling_team', 'season', 'match_type', 'venue', 'team1', 'team2', 'toss_winner', 'toss_decision', 'winner', 'result', 'super_over']
merged_df[categorical_cols] = merged_df[categorical_cols].astype('category')
```

In [13]:
```python
print("Updated Information of merged dataset:")
print(merged_df.info())
```

```
Updated Information of merged dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 260920 entries, 0 to 260919
Data columns (total 37 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   match_id          260920 non-null  int64
 1   inning            260920 non-null  int64
 2   batting_team      260920 non-null  category
 3   bowling_team      260920 non-null  category
 4   over              260920 non-null  int64
 5   ball              260920 non-null  int64
 6   batter            260920 non-null  object
 7   bowler            260920 non-null  object
 8   non_striker       260920 non-null  object
 9   batsman_runs      260920 non-null  int64
 10  extra_runs        260920 non-null  int64
 11  total_runs        260920 non-null  int64
 12  extras_type       14125 non-null   object
 13  is_wicket         260920 non-null  int64
 14  player_dismissed  12950 non-null   object
 15  dismissal_kind    12950 non-null   object
 16  fielder           9354 non-null    object
 17  id                260920 non-null  int64
 18  season            260920 non-null  category
 19  city              248523 non-null  object
 20  date              260920 non-null  datetime64[ns]
 21  match_type        260920 non-null  category
 22  player_of_match   260430 non-null  object
 23  venue             260920 non-null  category
 24  team1             260920 non-null  category
 25  team2             260920 non-null  category
 26  toss_winner       260920 non-null  category
 27  toss_decision     260920 non-null  category
 28  winner            260430 non-null  category
 29  result            260920 non-null  category
 30  result_margin     256796 non-null  float64
 31  target_runs       260611 non-null  float64
 32  target_overs      260611 non-null  float64
 33  super_over        260920 non-null  category
 34  method            3646 non-null    object
 35  umpire1           260920 non-null  object
 36  umpire2           260920 non-null  object
dtypes: category(12), datetime64[ns](1), float64(3), int64(9), object(12)
memory usage: 52.8+ MB
None
```

6. Derive an index field and add it to the data set.

In [14]:
```python
merged_df['index'] = merged_df['match_id'].astype(str) + '_' + merged_df['inning'].astype(str)
```

In [15]:
```python
print(merged_df)
```

```
        match_id  inning           batting_team                 bowling_team  \
0         335982       1  Kolkata Knight Riders  Royal Challengers Bangalore
1         335982       1  Kolkata Knight Riders  Royal Challengers Bangalore
2         335982       1  Kolkata Knight Riders  Royal Challengers Bangalore
3         335982       1  Kolkata Knight Riders  Royal Challengers Bangalore
4         335982       1  Kolkata Knight Riders  Royal Challengers Bangalore
...          ...     ...                    ...                          ...
260915   1426312       2  Kolkata Knight Riders          Sunrisers Hyderabad
260916   1426312       2  Kolkata Knight Riders          Sunrisers Hyderabad
260917   1426312       2  Kolkata Knight Riders          Sunrisers Hyderabad
260918   1426312       2  Kolkata Knight Riders          Sunrisers Hyderabad
260919   1426312       2  Kolkata Knight Riders          Sunrisers Hyderabad

        over  ball       batter         bowler  non_striker  batsman_runs  \
0          0     1   SC Ganguly        P Kumar  BB McCullum             0
1          0     2  BB McCullum        P Kumar   SC Ganguly             0
2          0     3  BB McCullum        P Kumar   SC Ganguly             0
3          0     4  BB McCullum        P Kumar   SC Ganguly             0
4          0     5  BB McCullum        P Kumar   SC Ganguly             0
...      ...   ...          ...            ...          ...           ...
260915     9     5      SS Iyer     AK Markram      VR Iyer             1
260916     9     6      VR Iyer     AK Markram      SS Iyer             1
260917    10     1      VR Iyer  Shahbaz Ahmed      SS Iyer             1
260918    10     2      SS Iyer  Shahbaz Ahmed      VR Iyer             1
260919    10     3      VR Iyer  Shahbaz Ahmed      SS Iyer             1

        ...                 winner   result result_margin  target_runs  \
0       ...  Kolkata Knight Riders     runs         140.0        223.0
1       ...  Kolkata Knight Riders     runs         140.0        223.0
2       ...  Kolkata Knight Riders     runs         140.0        223.0
3       ...  Kolkata Knight Riders     runs         140.0        223.0
4       ...  Kolkata Knight Riders     runs         140.0        223.0
...     ...                    ...      ...           ...          ...
260915  ...  Kolkata Knight Riders  wickets           8.0        114.0
260916  ...  Kolkata Knight Riders  wickets           8.0        114.0
260917  ...  Kolkata Knight Riders  wickets           8.0        114.0
260918  ...  Kolkata Knight Riders  wickets           8.0        114.0
260919  ...  Kolkata Knight Riders  wickets           8.0        114.0

       target_overs super_over method        umpire1      umpire2      index
0              20.0          N    NaN      Asad Rauf  RE Koertzen   335982_1
1              20.0          N    NaN      Asad Rauf  RE Koertzen   335982_1
2              20.0          N    NaN      Asad Rauf  RE Koertzen   335982_1
3              20.0          N    NaN      Asad Rauf  RE Koertzen   335982_1
4              20.0          N    NaN      Asad Rauf  RE Koertzen   335982_1
...             ...        ...    ...            ...          ...        ...
260915         20.0          N    NaN  J Madanagopal  Nitin Menon  1426312_2
260916         20.0          N    NaN  J Madanagopal  Nitin Menon  1426312_2
260917         20.0          N    NaN  J Madanagopal  Nitin Menon  1426312_2
260918         20.0          N    NaN  J Madanagopal  Nitin Menon  1426312_2
260919         20.0          N    NaN  J Madanagopal  Nitin Menon  1426312_2

[260920 rows x 38 columns]
```

7. Identify the outliers in each column

In [16]:
```python
numeric_cols = ['batsman_runs', 'total_runs', 'result_margin', 'target_runs', 'target_overs']
for col in numeric_cols:
    Q1 = merged_df[col].quantile(0.25)
    Q3 = merged_df[col].quantile(0.75)
    IQR = Q3 - Q1
    outliers = merged_df[(merged_df[col] < Q1 - 1.5 * IQR) | (merged_df[col] > Q3 + 1.5 * IQR)]
    print(f"Outliers in {col}: {len(outliers)}")
```

```
Outliers in batsman_runs: 43749
Outliers in total_runs: 44719
Outliers in result_margin: 28956
Outliers in target_runs: 5854
Outliers in target_overs: 5013
```

8. What was the count of matches played in each season?

In [17]:
```python
matches_per_season = merged_df.groupby('season')['match_id'].nunique()
print(matches_per_season)
```

```
season
2007/08    58
2009       57
2009/10    60
2011       73
2012       74
2013       76
2014       60
2015       59
2016       60
2017       59
2018       60
2019       60
2020/21    60
2021       60
2022       74
2023       74
2024       71
Name: match_id, dtype: int64
```

```
C:\Users\lab1002\AppData\Local\Temp\ipykernel_8092\1355830781.py:1: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  matches_per_season = merged_df.groupby('season')['match_id'].nunique()
```

9. How many runs were scored in each season?

In [18]:
```python
runs_per_season = df2.groupby('season')['total_runs'].sum()
print(runs_per_season)
```

```
season
2007/08    17937
2009       16353
2009/10    18883
2011       21154
2012       22453
2013       22602
2014       18931
2015       18353
2016       18862
2017       18786
2018       19901
2019       19434
2020/21    19416
2021       18637
2022       24395
2023       25688
2024       25971
Name: total_runs, dtype: int64
```

```
C:\Users\lab1002\AppData\Local\Temp\ipykernel_8092\1585962320.py:1: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  runs_per_season = merged_df.groupby('season')['total_runs'].sum()
```

10. Find the top 10 teams which has won the most number of matches

In [40]:
```python
top_teams = df2['winner'].value_counts().head(10)
print(top_teams)
```

```
winner
Mumbai Indians                 144
Chennai Super Kings            138
Kolkata Knight Riders          131
Royal Challengers Bangalore    116
Rajasthan Royals               112
Sunrisers Hyderabad             88
Kings XI Punjab                 88
Delhi Daredevils                67
Delhi Capitals                  48
Deccan Chargers                 29
Name: count, dtype: int64
```

11. Find the Top 10 Players With Most Player of Match

In [28]:
```python
top_players = df2['player_of_match'].value_counts().head(10)
print(top_players)
```

```
player_of_match
AB de Villiers    25
CH Gayle          22
RG Sharma         19
V Kohli           18
DA Warner         18
MS Dhoni          17
YK Pathan         16
RA Jadeja         16
SR Watson         16
AD Russell        15
Name: count, dtype: int64
```

12. Find the Percentage of choosing field after winning the toss
13. Find the Percentage of choosing to bat first after winning the toss

In [39]:
```python
toss_field_percentage = (merged_df[merged_df['toss_decision'] == 'field'].shape[0] / merged_df.shape[0]) * 100
toss_bat_percentage = (merged_df[merged_df['toss_decision'] == 'bat'].shape[0] / merged_df.shape[0]) * 100
print(f"Field after winning toss: {toss_field_percentage:.2f}%")
print(f"Bat after winning toss: {toss_bat_percentage:.2f}%")
```

```
Field after winning toss: 64.18%
Bat after winning toss: 35.82%
```

14. Find the ratio of Bat to Field count

In [34]:
```python
bat_count = df2['toss_decision'].value_counts().get('bat', 0)
field_count = df2['toss_decision'].value_counts().get('field', 0)
ratio = bat_count / field_count if field_count != 0 else "Undefined"
print(f"Bat-to-Field ratio: {ratio}")
```

```
Bat-to-Field ratio: 0.5553977272727273
```

15. Find the count of Loss vs Wins vs Ties

In [31]:
```python
results_count = df2['result'].value_counts()
print(results_count)
```

```
result
wickets      578
runs         498
tie           14
no result      5
Name: count, dtype: int64
```

16. Who won the most tosses?

In [30]:
```python
most_tosses_won = df2['toss_winner'].value_counts().idxmax()
print(f"Most tosses won by: {most_tosses_won}")
```

```
Most tosses won by: Mumbai Indians
```

17. Who has hit the most number of 4's?
18. Who has hit the most number of 6's?

In [29]:
```python
most_fours = df1[df1['batsman_runs'] == 4]['batter'].value_counts().idxmax()
most_sixes = df1[df1['batsman_runs'] == 6]['batter'].value_counts().idxmax()
```

```
print(f"Post #{i} bit by: {most_focus}")
print(f"Post #{i} bit by: {most_views}")
```

```
Post #74 bit by: S Shanes
Post #74 bit by: CW Sayle
```