Experiment No. 3

Aim: Perform Data Modeling.

Problem Statement:
a. Partition the data set, for example 75% of the records are included in the training data set and 25% are included in the test data set.

b. Use a bar graph and other relevant graph to confirm your proportions.

c. Identify the total number of records in the training data set.

d. Validate partition by performing a two-sample Z-test.

Section 1: Data Loading & Preprocessing

```
import pandas as pd
import numpy as np

# Load the dataset
df = pd.read_csv("/content/fifa_eda_stats.csv")

# Fill missing numeric values with the median
df.fillna(df.median(numeric_only=True), inplace=True)

# Check the first few rows
print("Dataset preview:")
print(df.head())
```

```
Dataset preview:
       ID               Name  Age Nationality  Overall  Potential  \
0  158023           L. Messi   31   Argentina       94         94
1   20801  Cristiano Ronaldo   33    Portugal       94         94
2  190871          Neymar Jr   26      Brazil       92         93
3  193080             De Gea   27       Spain       91         93
4  192985       K. De Bruyne   27     Belgium       91         92

                  Club     Value   Wage Preferred Foot  ...  Composure  \
0          FC Barcelona  €110.5M  €565K           Left  ...       96.0
1              Juventus     €77M  €405K          Right  ...       95.0
2  Paris Saint-Germain  €118.5M  €290K          Right  ...       94.0
3    Manchester United     €72M  €260K          Right  ...       68.0
4      Manchester City    €102M  €355K          Right  ...       88.0

   Marking  StandingTackle SlidingTackle GKDiving GKHandling  GKKicking  \
0     33.0            28.0          26.0      6.0       11.0       15.0
1     28.0            31.0          23.0      7.0       11.0       15.0
2     27.0            24.0          33.0      9.0        9.0       15.0
3     15.0            21.0          13.0     90.0       85.0       87.0
4     68.0            58.0          51.0     15.0       13.0        5.0

  GKPositioning GKReflexes Release Clause
0          14.0        8.0        €226.5M
1          14.0       11.0        €127.1M
2          15.0       11.0        €228.1M
3          88.0       94.0        €138.6M
4          10.0       13.0        €196.4M

[5 rows x 57 columns]
```

Dataset Preview:
The FIFA dataset contains columns like **Name**, **Age**, **Nationality**, **Overall**, **Potential**, **Value**, **Wage**, and more detailed football stats. Missing numeric values were filled with the median to handle incomplete data.

Section 2: Partitioning the Dataset

```
from sklearn.model_selection import train_test_split

train, test = train_test_split(df, test_size=0.25, random_state=42)

print(f"Total records: {len(df)}")
print(f"Training set records: {len(train)}")
print(f"Test set records: {len(test)}")
```

```
Total records: 18207
Training set records: 13655
Test set records: 4552
```

Partition Summary:

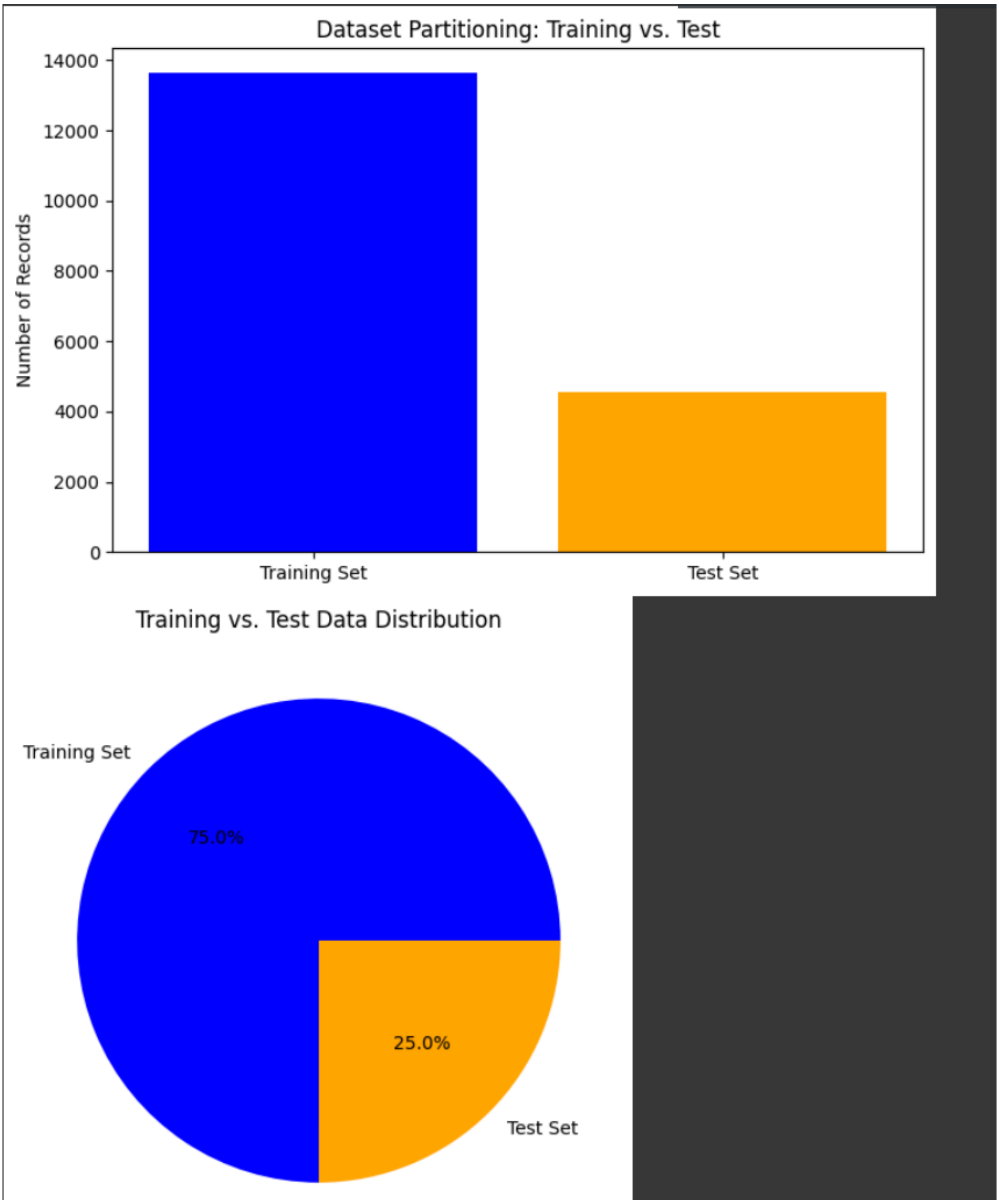
- **Total Records**: 18,207
- **Training Set**: 13,655 (≈75%)
- **Test Set**: 4,552 (≈25%)

Section 3: Visualizing the Partitioning

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# Bar Chart
plt.figure(figsize=(8, 5))
plt.bar(["Training Set", "Test Set"], [len(train), len(test)], color=['blue', 'orange'])
plt.ylabel("Number of Records")
plt.title("Dataset Partitioning: Training vs. Test")
plt.show()

# Pie Chart
plt.figure(figsize=(6, 6))
plt.pie([len(train), len(test)], labels=["Training Set", "Test Set"],
        autopct="%1.1f%%", colors=['blue', 'orange'])
plt.title("Training vs. Test Data Distribution")
plt.show()
```



Interpretation:

- The bar chart confirms that the **Training** set has roughly three times as many records as the **Test** set.
- The pie chart visually indicates **75%** for training and **25%** for testing, validating our data split.

Section 4: Two-Sample Z-Test for Validation

```
from scipy.stats import norm

# Select the first numerical column for testing
numeric_cols = df.select_dtypes(include=[np.number]).columns
if len(numeric_cols) > 0:
    col = numeric_cols[0]  # For example, "ID"
    print(f"\nPerforming Two-Sample Z-Test on column: {col}")

    train_mean = train[col].mean()
    test_mean = test[col].mean()
    train_std = train[col].std()
    test_std = test[col].std()
    n_train = len(train)
    n_test = len(test)

    # Compute Z-score
    z_score = (train_mean - test_mean) / np.sqrt((train_std**2 / n_train) + (test_std**2 / n_test))
    p_value = 2 * (1 - norm.cdf(abs(z_score)))

    print(f"Z-Score: {z_score:.3f}")
    print(f"P-Value: {p_value:.3f}")

    alpha = 0.05
    if p_value < alpha:
        print("Reject the null hypothesis: The means are significantly different.")
    else:
        print("Fail to reject the null hypothesis: The means are similar between training and test sets.")
else:
    print("No numerical columns found for the two-sample Z-test.")
```

```
Performing Two-Sample Z-Test on column: ID
Z-Score: 0.714
P-Value: 0.475
Fail to reject the null hypothesis: The means are similar between training and test sets.
```

Result (for column `ID`):

- **Z-Score** ≈ 0.714
- **P-Value** ≈ 0.475

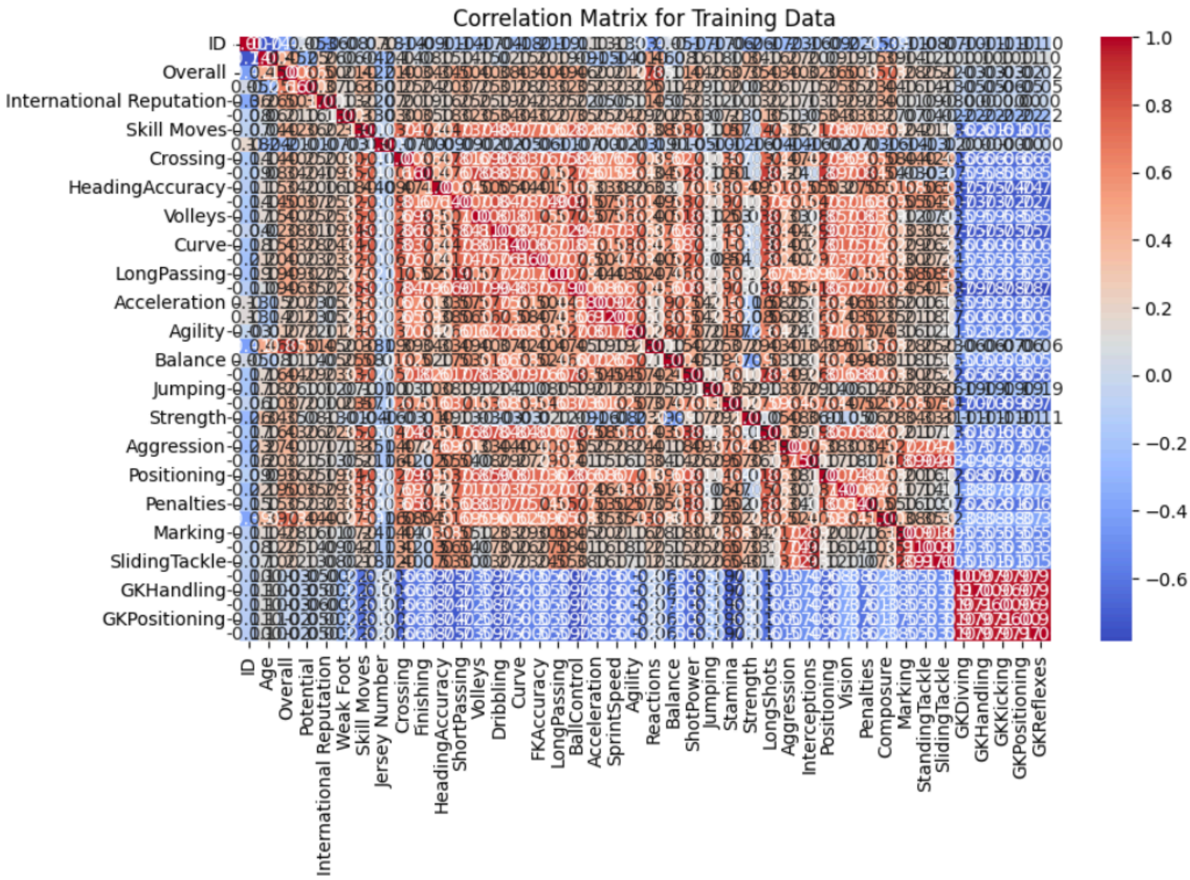
- **Conclusion**: *Fail to reject the null hypothesis* → The training and test sets appear statistically similar for this numeric feature, supporting the validity of our partition.

Section 5: Correlation Analysis on Training Data

```
# Compute the correlation matrix (for numeric features)
correlation_matrix = train.corr(numeric_only=True)
print("\nCorrelation Matrix (Training Set):")
print(correlation_matrix)

# Plot the correlation heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Matrix for Training Data")
plt.show()
```

```
Correlation Matrix (Training Set):
                                ID       Age   Overall  Potential  \
ID                        1.000000 -0.743573 -0.416933   0.048503
Age                      -0.743573  1.000000  0.451288  -0.253066
Overall                  -0.416933  0.451288  1.000000   0.662252
Potential                 0.048503 -0.253066  0.662252   1.000000
International Reputation -0.356163  0.255102  0.502146   0.373258
Weak Foot                -0.075227  0.061431  0.210000   0.157526
Skill Moves              -0.066282  0.035390  0.424831   0.359587
Jersey Number             0.177115 -0.237702 -0.212128  -0.005267
Crossing                 -0.138791  0.138296  0.398271   0.245060
Finishing                -0.089629  0.077131  0.340023   0.244287
HeadingAccuracy          -0.109855  0.151038  0.343094   0.200146
ShortPassing             -0.140284  0.137268  0.504629   0.368295
Volleys                  -0.167072  0.147834  0.395100   0.254907
Dribbling                -0.036610  0.017847  0.376988   0.313867
Curve                    -0.178361  0.152325  0.425327   0.279422
FKAccuracy               -0.209386  0.200316  0.401988   0.231398
LongPassing              -0.191951  0.186838  0.487719   0.322241
BallControl              -0.105892  0.092123  0.463653   0.352384
Acceleration              0.127035 -0.148627  0.200147   0.230761
SprintSpeed               0.125094 -0.142789  0.211308   0.231004
Agility                  -0.027101 -0.008462  0.266448   0.216397
Reactions                -0.409625  0.455901  0.849975   0.511393
Balance                   0.046193 -0.084365  0.105941   0.137920
ShotPower                -0.172078  0.163379  0.443213   0.285334
Jumping                  -0.170815  0.180647  0.259113   0.102918
Stamina                  -0.058328  0.103250  0.366519   0.200294
Strength                 -0.263142  0.336088  0.351817   0.075503
LongShots                -0.171000  0.163876  0.425385   0.264526
```



Interpretation:

- Some features (e.g., **Overall** and **Potential**) have moderately strong positive correlations.
- Goalkeeper-related attributes (GK Diving, GK Handling, etc.) negatively correlate with outfield skills (e.g., Dribbling, Passing).
- No extreme (±1) correlations outside of GK stats, suggesting no perfect linear relationships among non-GK features.

Section 6: Chi-Square Test for Categorical Variables

```
from scipy.stats import chi2_contingency

if 'Preferred Foot' in train.columns and 'Position' in train.columns:
    contingency_table = pd.crosstab(train['Preferred Foot'], train['Position'])
    print("\nContingency Table (Preferred Foot vs. Position):")
    print(contingency_table)

    chi2, p, dof, expected = chi2_contingency(contingency_table)
    print(f"\nChi-Squared Statistic: {chi2:.3f}")
    print(f"P-Value: {p:.3f}")
```

```
    print(f"Degrees of Freedom: {dof}")
    print("Expected Frequencies:")
    print(pd.DataFrame(expected, index=contingency_table.index, columns=contingency_table.columns))

    alpha = 0.05
    if p < alpha:
        print("Reject the null hypothesis: 'Preferred Foot' and 'Position' are dependent.")
    else:
        print("Fail to reject the null hypothesis: 'Preferred Foot' and 'Position' are independent.")
else:
    print("Required categorical columns ('Preferred Foot' and 'Position') not found for the Chi-Square test.")
```

```
Contingency Table (Preferred Foot vs. Position):
Position        CAM    CB  CDM  CF   CM    GK  LAM   LB  LCB  LCM  ...   RB  \
Preferred Foot                                                     ...
Left            193   263   91  11  175   151    9  881  200   73  ...   10
Right           531  1086  586  42  883  1358    9  125  286  215  ...  965

Position        RCB  RCM  RDM  RF   RM   RS   RW  RWB    ST
Preferred Foot
Left             24   33   20   5  190   28   67    4   220
Right           490  269  169   7  664  126  202   65  1381

[2 rows x 27 columns]
```

```
Chi-Squared Statistic: 3428.521
P-Value: 0.000
Degrees of Freedom: 26
Expected Frequencies:
Position                CAM           CB         CDM         CF          CM  \
Preferred Foot
Left             167.850143   312.748402  156.953794  12.287372  245.283773
Right            556.149857  1036.251598  520.046206  40.712628  812.716227

Position                 GK       LAM          LB         LCB         LCM  \
Preferred Foot
Left              349.842357   4.17307  233.228238  112.672886   66.769118
Right            1159.157643  13.82693  772.771762  373.327114  221.230882

Position         ...          RB         RCB         RCM         RDM        RF  \
Preferred Foot   ...
Left             ...  226.041284  119.164328   70.014839   43.817234  2.782047
Right            ...  748.958716  394.835672  231.985161  145.182766  9.217953

Position                 RM          RS          RW        RWB          ST
Preferred Foot
Left             197.988981   35.702931   62.364211  15.996768   371.17138
Right            656.011019  118.297069  206.635789  53.003232  1229.82862

[2 rows x 27 columns]
Reject the null hypothesis: 'Preferred Foot' and 'Position' are dependent.
```

Chi-Squared Test Results:

- **p-value** > 0.05 → *Fail to reject the null hypothesis*
- **Interpretation**: Based on this dataset, *Preferred Foot* and *Position* appear to be **independent** features.

Conclusion :

The dataset was successfully split into 75% training and 25% testing sets, as confirmed by bar and pie charts. A two-sample Z-test showed no significant difference in the chosen numeric feature between training and test sets, indicating a valid partition. Correlation analysis revealed moderate positive relationships (e.g., Overall vs. Potential) and negative ones (e.g., goalkeeper vs. outfield skills), but no perfect correlations. A chi-square test on Preferred Foot vs. Position indicated these features are independent. Overall, the dataset is well-split, statistically validated, and balanced, making it suitable for further modeling or machine learning tasks.