

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.

Steps:

- 1) 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.

Code:

```
from sklearn.model_selection import train_test_split
train_df, test_df = train_test_split(df, test_size=0.25, random_state=42)
```

This function imports the `train_test_split` function from `sklearn.model_selection` library. This makes 2 dataframes, a `train_df` and `test_df`. Here, based on the `test_size` parameter, it would divide the dataset into that percent of values and insert it in the `test_df` dataframe. The remaining values are put in the `train_df` dataframe. Defining the `random_state` parameter helps the splitting to be consistent. The value of the parameter does not matter, only the condition being it should be consistent.

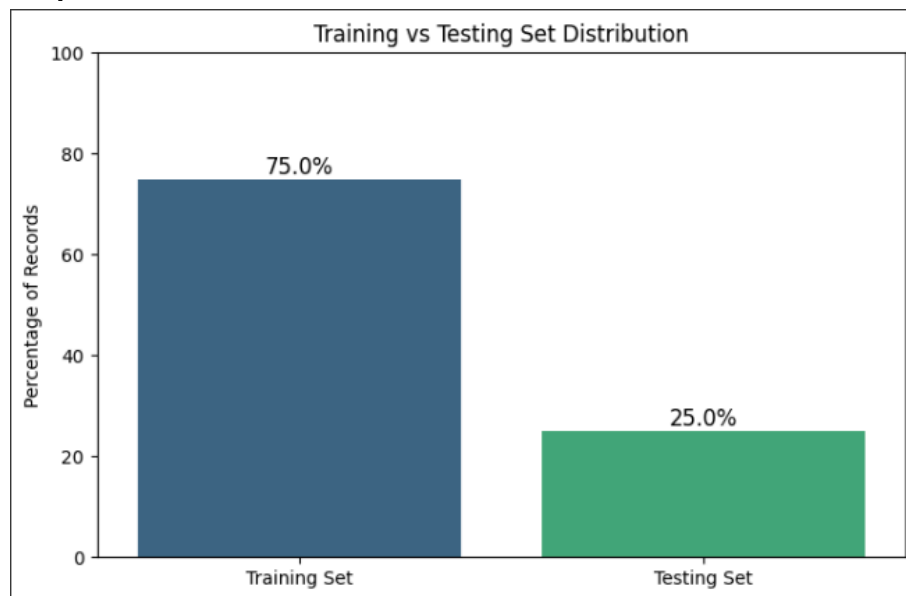
- 2) Use a bar graph and other relevant graphs to confirm your proportions. Graphs help validate the correct division of data. Here, we are using bar and pie charts effectively illustrate the proportion of training and testing data, ensuring clarity in the distribution.

Bar Graph:

Code:

```
import matplotlib.pyplot as plt
import seaborn as sns
total = len(df)
sizes = [len(train_df) / total * 100, len(test_df) / total * 100]
labels = ['Training Set', 'Testing Set']
plt.figure(figsize=(8, 5))
sns.barplot(x=labels, y=sizes, palette="viridis")
for i, v in enumerate(sizes):
    plt.text(i, v + 1, f'{v:.1f}%', ha='center', fontsize=12)
plt.ylabel("Percentage of Records")
plt.title("Training vs Testing Set Distribution")
plt.ylim(0, 100) # Ensure y-axis goes from 0 to 100%
plt.show()
```

Output:

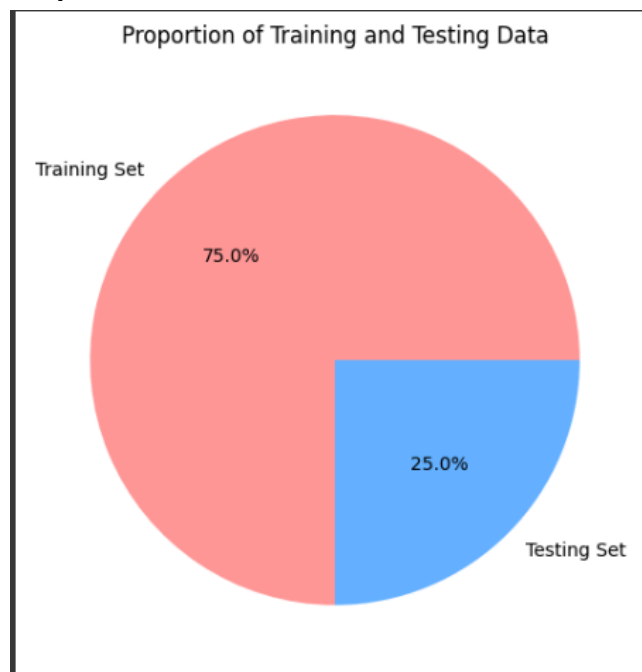


Pie chart:

Code:

```
plt.figure(figsize=(6,6))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', colors=['#ff9999','#66b3ff'])
plt.title("Proportion of Training and Testing Data")
plt.show()
```

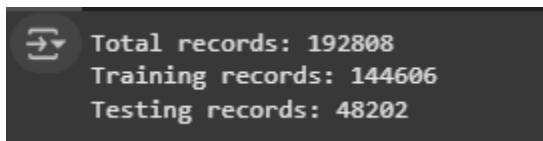
Output:



3) Identify the total number of records in the training data set.

Code:

```
print(f"Total records: {len(df)}")
print(f"Training records: {len(train_df)}")
print(f"Testing records: {len(test_df)}")
```

Output:A terminal window with a dark background and light green text. It shows the output of the code: 'Total records: 192808', 'Training records: 144606', and 'Testing records: 48202'. There is a small icon in the top left corner of the terminal window.

```
Total records: 192808
Training records: 144606
Testing records: 48202
```

4) Validate partition by performing a two-sample Z-test.

A two-sample Z-test evaluates whether the training and testing datasets share similar characteristics. By comparing their mean values, it ensures the data split is balanced and does not introduce bias.

Code:

```
import numpy as np
from scipy.stats import norm

train_values = train_df["Data_Value"]
test_values = test_df["Data_Value"]

mean_train = np.mean(train_values)
mean_test = np.mean(test_values)
std_train = np.std(train_values, ddof=1)
std_test = np.std(test_values, ddof=1)

n_train = len(train_values)
n_test = len(test_values)

z_score = (mean_train - mean_test) / np.sqrt((std_train**2 / n_train) + (std_test**2 / n_test))

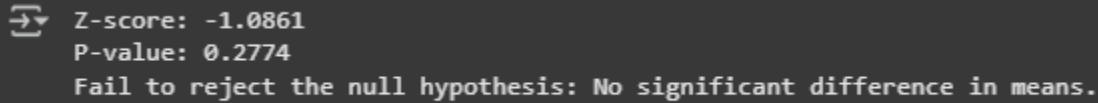
p_value = 2 * (1 - norm.cdf(abs(z_score)))

print(f"Z-score: {z_score:.4f}")
print(f"P-value: {p_value:.4f}")

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: No significant difference in means.")
```

Output:

```
↵ Z-score: -1.0861  
P-value: 0.2774  
Fail to reject the null hypothesis: No significant difference in means.
```

As the `p_value` came out to be greater than 0.05, we would accept that there is no difference in the means and hence, the splitting performed is indeed valid.

Conclusion: The data partitioning process successfully divided the dataset into 75% training data and 25% testing data. The distribution was validated using bar and pie charts, which visually confirmed the correct proportions of training and testing records. The bar chart effectively displayed the percentage of records in each subset, while the pie chart provided a clear representation of their proportions. The total number of records in the training set came out to be 1,44,606, which is 75% of the total records in the dataset. Further validation was performed using a two-sample Z-test, which compared the mean values of the training and testing datasets to ensure a balanced split without introducing bias. The calculated Z-score (-1.0861) and p-value (0.2774) indicated whether there was a significant difference between the two subsets. If the p-value was greater than the significance level (0.05), it confirmed that the means of the training and testing sets were statistically similar, ensuring a fair representation of the data. Overall, this experiment ensured that the dataset was properly divided, maintaining consistency in distribution while preserving statistical integrity for effective model training and evaluation.