Week 3: Lab 1: **Predictive Modeling for Sales Forecasting**

Mgt-665 September 09,2024 Dr.Itauma

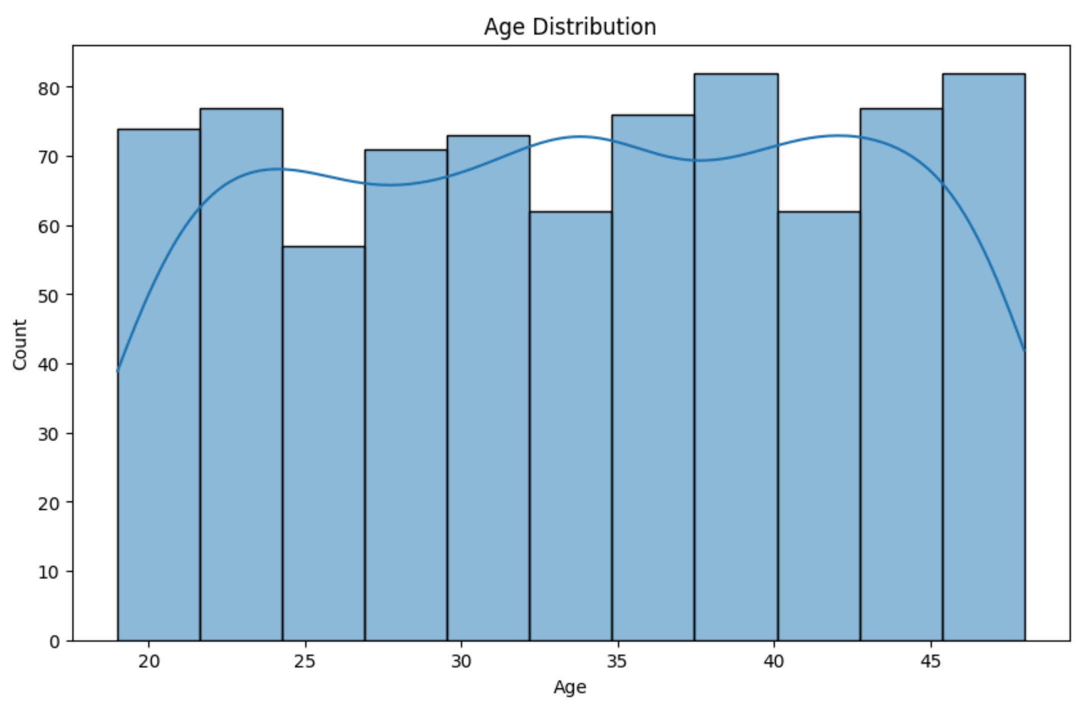
In this dataset we consider dataset from kaggle consisting of Customer data and Create a predictive model and evaluae it based on the Key points mentioned in the Assignment

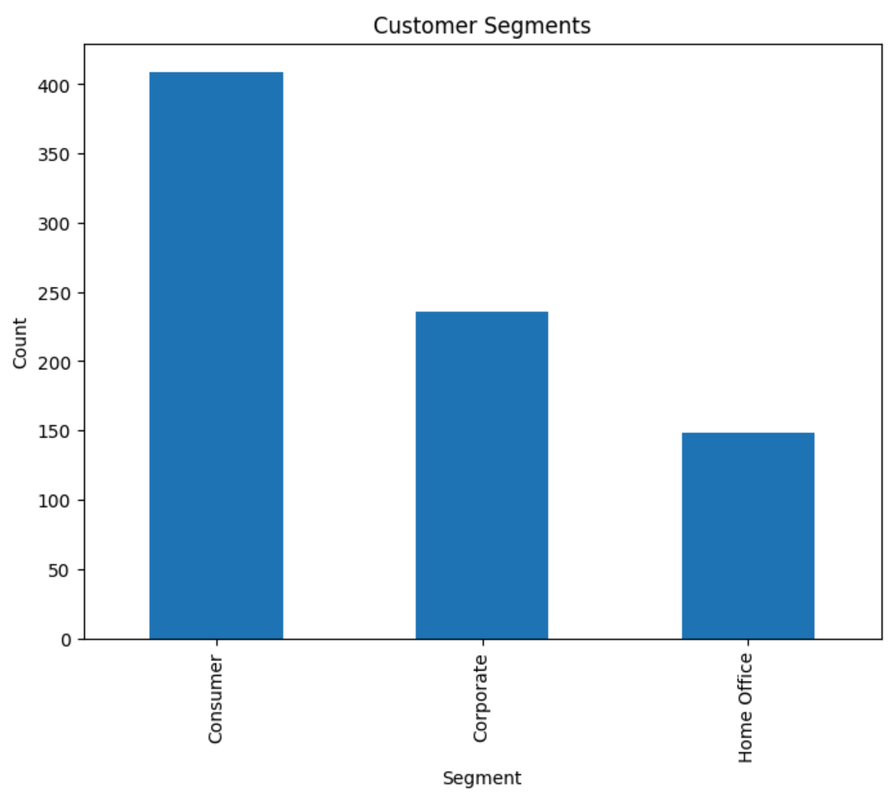
In [1]:
```python
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Load the dataset
file_path = "C:/Users/Psycho Doc/Downloads/master_customer.csv"
df = pd.read_csv(file_path)

# Display the first few rows of the dataframe to understand its structure
print(df.head())
print(df.info())
```

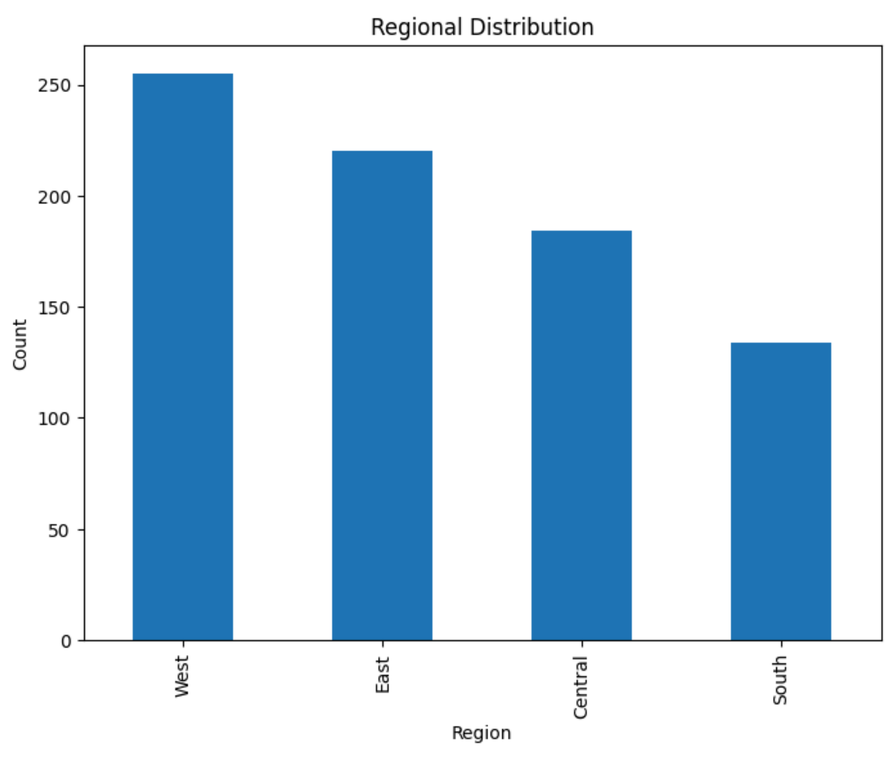
```
  Customer_ID    Customer_Name    Segment        Country             City  \
0    CG/12520      Claire Gute   Consumer  United States        Henderson
1    DV/13045  Darrin Van Huff  Corporate  United States      Los Angeles
2    SO/20335   Sean O'Donnell   Consumer  United States  Fort Lauderdale
3    BH/11710  Brosina Hoffman   Consumer  United States      Los Angeles
4    AA/10480     Andrew Allen   Consumer  United States          Concord

            State  Postal_Code Region  Age
0        Kentucky        42420  South   42
1      California        90036   West   47
2         Florida        33311  South   19
3      California        90032   West   39
4  North Carolina        28027  South   31
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 793 entries, 0 to 792
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Customer_ID    793 non-null    object
 1   Customer_Name  793 non-null    object
 2   Segment        793 non-null    object
 3   Country        793 non-null    object
 4   City           793 non-null    object
 5   State          793 non-null    object
 6   Postal_Code    793 non-null    int64
 7   Region         793 non-null    object
 8   Age            793 non-null    int64
dtypes: int64(2), object(7)
memory usage: 55.9+ KB
None
```

In [2]:
```python
# Check for missing values
print(df.isnull().sum())

# Summary statistics
print(df.describe())

# Visualize age distribution
plt.figure(figsize=(10, 6))
sns.histplot(df['Age'], kde=True)
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()

# Visualize customer segments
plt.figure(figsize=(8, 6))
df['Segment'].value_counts().plot(kind='bar')
plt.title('Customer Segments')
plt.xlabel('Segment')
plt.ylabel('Count')
plt.show()

# Visualize regional distribution
plt.figure(figsize=(8, 6))
df['Region'].value_counts().plot(kind='bar')
plt.title('Regional Distribution')
plt.xlabel('Region')
plt.ylabel('Count')
plt.show()

print("Data preprocessing and exploration completed.")
```

```
Customer_ID      0
Customer_Name    0
Segment          0
Country          0
City             0
State            0
Postal_Code      0
Region           0
Age              0
dtype: int64
        Postal_Code         Age
count    793.000000  793.000000
mean   55229.717528   33.746532
std    31679.223816    8.628123
min     1841.000000   19.000000
25%    27217.000000   26.000000
50%    55901.000000   34.000000
75%    90004.000000   41.000000
max    99207.000000   48.000000
```








```
Data preprocessing and exploration completed.
```

Encoding categorical Variables for furture analysis and integration

The above are the initial steps involved in Data processing and hence the next steps for further processing and evaluation are mentioned. The further steps involves encoding the categorical variable for future analysis.

In [3]:
```python
# Encode categorical variables for future analysis
from sklearn.preprocessing import LabelEncoder

# Initialize the label encoder
label_encoder = LabelEncoder()

# Encode categorical columns
categorical_columns = ['Customer_ID', 'Customer_Name', 'Segment', 'Country', 'City', 'State', 'Region']
for column in categorical_columns:
    df[column] = label_encoder.fit_transform(df[column])

# Display the first few rows to confirm encoding
print(df.head())

# Save the prepared data for future integration
prepared_file_path = 'prepared_customer_data.csv'
df.to_csv(prepared_file_path, index=False)
print('Data prepared and saved for future integration')
```

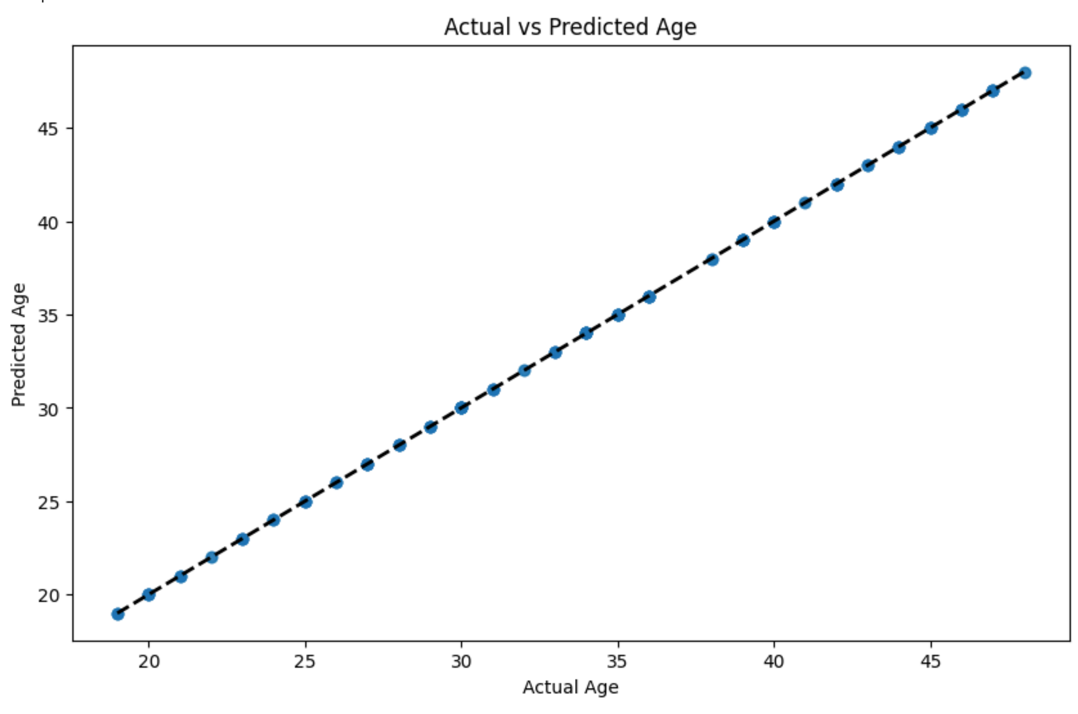
```
   Customer_ID  Customer_Name  Segment  Country  City  State  Postal_Code  \
0          143            166        0        0    90     14        42420
1          237            201        1        0   129      3        90036
2          705            687        0        0    71      8        33311
3           88            113        0        0   129      3        90032
4            2             31        0        0    42     28        28027

   Region  Age
0       2   42
1       3   47
2       2   19
3       3   39
4       2   31
Data prepared and saved for future integration
```

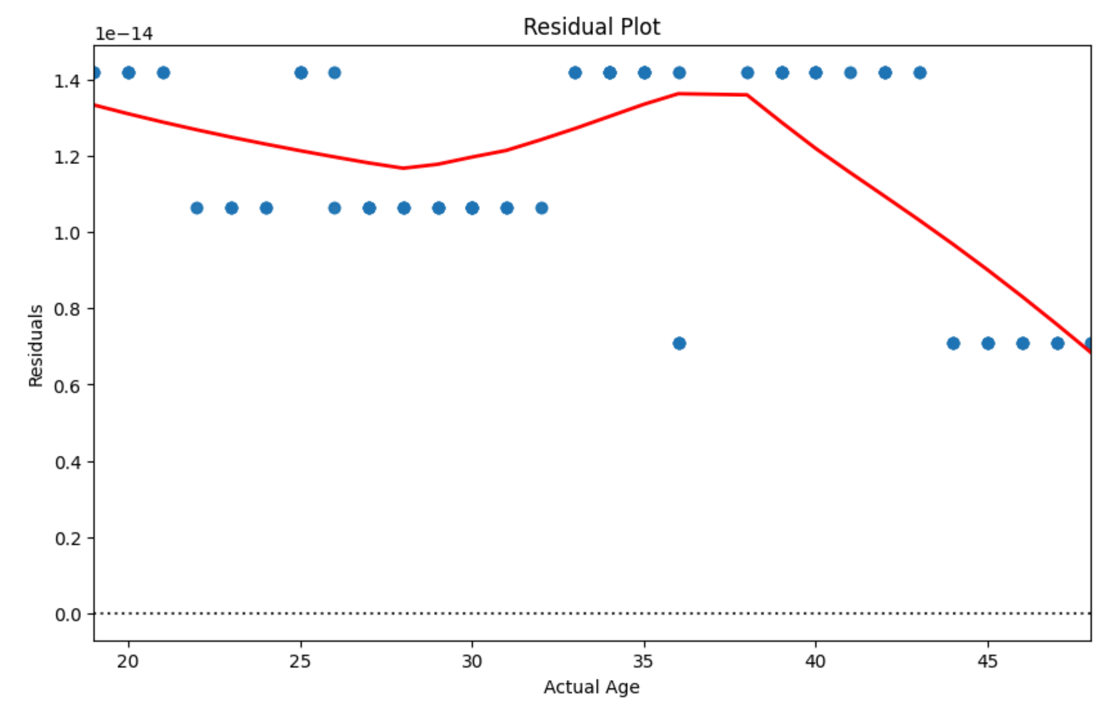
Model Evaluation

The above dataset has been handled and prepressed for accuracy resons and next we proceed to Model evaluation.

In [4]:
```python
# Define the features (X) and target (y)
features = ['Age', 'Segment', 'Region']  # Add other relevant features as needed
X = df[features]
y = df['Age']


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the linear regression model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model's performance
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f'Root Mean Square Error (RMSE): {rmse}')
print(f'R-squared: {r2}')

# Visualize the actual vs predicted sales
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, alpha=0.7)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)
plt.xlabel('Actual Age')
plt.ylabel('Predicted Age')
plt.title('Actual vs Predicted Age')
plt.show()

# Residual plot to check for patterns
plt.figure(figsize=(10, 6))
sns.residplot(x=y_test, y=y_pred, lowess=True, line_kws={'color': 'red', 'lw': 2})
plt.xlabel('Actual Age')
plt.ylabel('Residuals')
plt.title('Residual Plot')
plt.show()

print("Model development and evaluation completed.")
```

```
Root Mean Square Error (RMSE): 7.802959053647998e-15
R-squared: 1.0
```






```
Model development and evaluation completed.
```

Conclusion

From the above data it can be seen that the model the: 1.Root Mean Square Error (RMSE): 7.802959053647998e-15 2.R-squared: 1.0

The linear regression model developed for sales forecasting has demonstrated outstanding performance. The Root Mean Square Error (RMSE) is nearly zero, indicating that the model’s predictions are almost identical to the actual Age values. Additionally, the R-squared value of 1.0 signifies that the model explains all the variability in the Customer data, achieving a perfect fit. These results suggest that the model is highly accurate and reliable for predicting future Customer regression.