In [2]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
import joblib

# Load the combined DataFrame
combined_df = pd.read_csv('CombinedDataFrame.csv')

# Specify the correct column names based on your Phase 3 code
rental_price_column = 'Rental_Price'
area_column = 'Area'

# Check if the specified columns exist
if rental_price_column not in combined_df.columns or area_column not in combined_df.columns:
    print(f"Error: The specified columns '{rental_price_column}' or '{area_column}' do not exist in the combined DataFrame. Please verify the colum
    exit()

# Drop rows with missing values in the target variables
combined_df = combined_df.dropna(subset=[rental_price_column, area_column])

# Drop non-numeric features
X = combined_df.select_dtypes(include=['float64', 'int64'])

# Feature selection and target variable
y_rental_price = combined_df[rental_price_column]
y_area = combined_df[area_column]

# Train-test split
X_train_rental, X_test_rental, y_train_rental, y_test_rental = train_test_split(X, y_rental_price, test_size=0.2, random_state=42)
X_train_area, X_test_area, y_train_area, y_test_area = train_test_split(X, y_area, test_size=0.2, random_state=42)


# Check for missing values in features
if X_train_rental.isnull().values.any() or X_test_rental.isnull().values.any() or X_train_area.isnull().values.any() or X_test_area.isnull().values
    print("Error: There are missing values in the features. Please handle missing values before training the models.")
    exit()

# Initialize models
model_rental_price = RandomForestRegressor(random_state=42)
model_area = RandomForestRegressor(random_state=42)

# Fit models
model_rental_price.fit(X_train_rental, y_train_rental)
model_area.fit(X_train_area, y_train_area)

# Predictions
y_pred_rental_price = model_rental_price.predict(X_test_rental)
y_pred_area = model_area.predict(X_test_area)

# Evaluate models
mse_rental_price = mean_squared_error(y_test_rental, y_pred_rental_price)
mse_area = mean_squared_error(y_test_area, y_pred_area)

print(f'Mean Squared Error for Rental Price: {mse_rental_price}')
print(f'Mean Squared Error for Area: {mse_area}')

# Save the best models
joblib.dump(model_rental_price, 'best_model_rental_price.joblib')
joblib.dump(model_area, 'best_model_area.joblib')
```

```
Mean Squared Error for Rental Price: 164836.0
Mean Squared Error for Area: 161604.0
```

Out[2]:
```
['best_model_area.joblib']
```

Data Loading:

The script loads a combined DataFrame from 'CombinedDataFrame.csv' using pandas. Column Specification:

Specifies target variable columns: 'Rental_Price' and 'Area'. Column Existence Check:

Verifies if specified columns exist; exits if not. Handling Missing Values:

Drops rows with missing 'Rental_Price' or 'Area'. Feature Selection:

Selects numeric features. Train-Test Split:

Splits data into training and testing sets. Missing Values Check:

Exits if missing values in selected features. Model Initialization:

Initializes RandomForestRegressor models for rental prices and areas. Model Training:

Fits models using training data. Predictions:

Generates predictions on the test set. Model Evaluation:

Computes Mean Squared Error for rental prices and areas. Save Models:

Saves trained models using joblib.

In [ ]: