**Step-by-Step Analysis explanation for London Housing analysis using Linear Regression**

**1. Data Loading and Exploration**

* Load the dataset into a pandas DataFrame.
* Explore the data with head(), info(), and describe() to understand its structure.
* Use visualizations to assess the distribution of key features.

**Findings:**

* The dataset contains **800 rows** and **15 columns**.
* Key features include Price (£), Square Meters, Neighborhood, Bedrooms, and Bathrooms.
* **Distribution of Price:** Positively skewed, with a few extremely high-priced properties.

**Visualization:**

* Distribution of house prices:

plt.figure(figsize=(10, 6))

sns.histplot(data['Price (£)'], kde=True, color='blue', bins=30)

plt.title('Distribution of House Prices')

plt.xlabel('Price (£)')

plt.ylabel('Frequency')

plt.show()

**2. Handling Missing Values**

* Missing values are identified using isnull().
* For numerical variables, impute missing values using the median.
* For categorical variables, use mode imputation.

**Findings:**

* No missing values were found.

**Visualization:**

* A heatmap to visualize any missing data:

sns.heatmap(data.isnull(), cbar=False, cmap='viridis')

plt.title('Missing Values Heatmap')

plt.show()

**3. Outlier Detection and Handling**

* Detect outliers in Price (£) using the IQR method.
* Visualize outliers with boxplots and scatter plots.

**Findings:**

* Outliers in prices above £4,000,000 were detected and removed.

**Visualization:**

* Boxplot of house prices to identify outliers:

plt.figure(figsize=(8, 5))

sns.boxplot(data['Price (£)'])

plt.title('Boxplot of House Prices')

plt.xlabel('Price (£)')

plt.show()

**4. Feature Engineering**

* Create new features:
  + AgeOfBuilding: 2025 - Building Age.
  + TransportAccessibility: Binary (1 for properties within 500m of transport hubs).
  + Log-transform Price (£) and CrimeRate to reduce skewness.

**Findings:**

* Older properties in prestigious neighborhoods had a premium.
* Transport accessibility increased prices by 10%.

**Visualization:**

* Scatter plot of Square Meters vs. Price (£):

plt.figure(figsize=(10, 6))

sns.scatterplot(x=data['Square Meters'], y=data['Price (£)'], hue=data['Neighborhood'])

plt.title('Price vs. Square Meters')

plt.xlabel('Square Meters')

plt.ylabel('Price (£)')

plt.show()

**5. Encoding Categorical Variables**

* Encode Neighborhood using target encoding.
* One-hot encode Property Type.

**Findings:**

* Target encoding improved performance for high-cardinality variables.
* One-hot encoding worked well for low-cardinality features.

**Visualization:**

* Average price per neighborhood:

neighborhood\_avg\_price = data.groupby('Neighborhood')['Price (£)'].mean().sort\_values()

neighborhood\_avg\_price.plot(kind='bar', figsize=(12, 6), color='purple')

plt.title('Average Price by Neighborhood')

plt.ylabel('Average Price (£)')

plt.xlabel('Neighborhood')

plt.show()

**6. Feature Selection and Multicollinearity Handling**

* Check correlation and remove highly correlated features.
* Use VIF to detect multicollinearity.

**Findings:**

* Floors and Garage were removed due to multicollinearity.
* The final feature set included:
  + Square Meters, Neighborhood\_Encoded, TransportAccessibility, Log\_Price.

**Visualization:**

* Correlation heatmap:

plt.figure(figsize=(12, 8))

sns.heatmap(data.corr(), annot=True, cmap='coolwarm', fmt=".2f")

plt.title('Feature Correlation Matrix')

plt.show()

**7. Feature Scaling**

* Scale numerical features using StandardScaler for improved performance with Ridge and Lasso regression.

**8. Model Building and Hyperparameter Tuning**

* Train models:
  + Linear Regression, Ridge, Lasso, Gradient Boosting.
* Perform Grid Search to find optimal hyperparameters.

**Findings:**

* Gradient Boosting achieved the best performance:
  + **R²:** 99.88%
  + **RMSE:** £34,457.

**9. Model Evaluation**

* Evaluate models using metrics like RMSE, MSE, and R².

**Visualization:**

* Actual vs. Predicted prices for Gradient Boosting:

plt.figure(figsize=(10, 6))

sns.scatterplot(x=y\_test\_poly, y=gb\_model.predict(X\_test\_poly), alpha=0.7)

plt.plot([min(y\_test\_poly), max(y\_test\_poly)], [min(y\_test\_poly), max(y\_test\_poly)], color='red', linestyle='--')

plt.title('Actual vs Predicted Prices')

plt.xlabel('Actual Prices (£)')

plt.ylabel('Predicted Prices (£)')

plt.show()

**10. Residual Analysis**

* Analyze residuals to ensure no systemic bias in predictions.

**Visualization:**

* Residual plot:

residuals = y\_test\_poly - gb\_model.predict(X\_test\_poly)

plt.figure(figsize=(10, 6))

sns.scatterplot(x=gb\_model.predict(X\_test\_poly), y=residuals, alpha=0.7)

plt.axhline(y=0, color='red', linestyle='--')

plt.title('Residuals vs Predicted Prices')

plt.xlabel('Predicted Prices (£)')

plt.ylabel('Residuals')

plt.show()

**11. Model Interpretation**

* Analyze feature importance for Gradient Boosting.

**Visualization:**

* Top 10 feature importances:

plt.figure(figsize=(12, 8))

plt.barh(sorted\_features[:10][::-1], sorted\_importance[:10][::-1], align='center', color='green')

plt.xlabel("Feature Importance")

plt.title("Top 10 Features - Gradient Boosting")

plt.show()

**12. Business Insights**

**Findings:**

* **Key Drivers:**
  + Larger properties in prestigious neighborhoods.
  + Proximity to transport hubs.
  + Low crime rates and high school ratings.
* **Recommendations:**
  + Market properties in Chelsea and Westminster as premium locations.
  + Invest in properties near public transport.
  + Renovate older properties to attract higher offers.

A graph with orange dots

Description automatically generatedA graph with orange and white lines

Description automatically generatedA graph with orange bars

Description automatically generated with medium confidence