BAHRIA UNIVERSITY, Karachi Campus

Department of Software Engineering

Assignment # 02 – Fall 2025	Course: Machine Learning
Course Code: AIC-301	Class: BSE – 5(B)
Instructor: Engr. AAMANA	Shift: Morning
Name: MIR HAMZA	Enrollment: 02-131232-057
Date: 13-10-2025	Due Date: 22 NOV 2025

Question 1: Data Preprocessing

Perform data preprocessing to handle missing values and clean inconsistent entries.

```
import pandas as pd import numpy as np df.drop_duplicates(inplace=True) df =
df.dropna(subset=['total_sqft', 'bath', 'balcony', 'size', 'location', 'price'])
df['bath'] = df['bath'].fillna(df['bath'].median()) df['balcony'] =
df['balcony'].fillna(df['balcony'].median())
```

Question 2: Feature Engineering

Create normalized values for comparison, such as total square feet normalization.

```
df['price_per_sqft'] = df['price'] * 100000 / df['total_sqft'] df['total_sqft_norm'] =
(df['total_sqft'] - df['total_sqft'].mean()) / df['total_sqft'].std()
```

Question 3: Conflict Resolution

Resolve data conflicts by detecting and removing outliers.

```
df = df[df['total_sqft'] / df['bhk'] >= 300] def remove_pps_outliers(data): df_out =
pd.DataFrame() for key, subdf in data.groupby('location'): m =
subdf['price_per_sqft'].mean() st = subdf['price_per_sqft'].std() reduced_df =
subdf[(subdf['price_per_sqft'] > (m - st)) & (subdf['price_per_sqft'] <= (m + st))] df_out
= pd.concat([df_out, reduced_df], ignore_index=True) return df_out df =
remove_pps_outliers(df)
```

Question 4: Visualization

Scatter plot comparing 2BHK and 3BHK prices using orange and red colors.

```
import matplotlib.pyplot as plt def plot_bhk_scatter(location): temp = df[(df['location']
== location) & (df['bhk'].isin([2, 3]))] plt.figure(figsize=(10,6))
plt.scatter(temp[temp['bhk']==2]['total_sqft'], temp[temp['bhk']==2]['price'],
color='orange', label='2 BHK') plt.scatter(temp[temp['bhk']==3]['total_sqft'],
temp[temp['bhk']==3]['price'], color='red', label='3 BHK') plt.xlabel("Total Square
Feet") plt.ylabel("Price (in lakhs)") plt.title(f"2BHK vs 3BHK Prices in {location}")
plt.legend() plt.show() plot_bhk_scatter("Rajaji Nagar")
```

Figure 1: Scatter Plot (2BHK vs 3BHK)

Question 5: Correlation Analysis

Analyze correlations using a YlOrRd heatmap.

```
import seaborn as sns corr = df[['total_sqft', 'bath', 'balcony', 'bhk', 'price']].corr()
plt.figure(figsize=(8,5)) sns.heatmap(corr, annot=True, cmap='YlOrRd')
plt.title("Correlation Matrix of Numerical Features") plt.show()
```

Figure 2: Correlation Heatmap

Question 6: Machine Learning Predictions

Use regression models to predict prices and evaluate performance metrics.

```
from sklearn.model_selection import train_test_split from sklearn.linear_model import
LinearRegression from sklearn.ensemble import RandomForestRegressor from sklearn.metrics
import mean_absolute_error, mean_squared_error, r2_score non_numeric = ['area_type',
'availability', 'society'] df = df.drop(columns=[col for col in non_numeric if col in
df.columns]) dummies = pd.get_dummies(df['location']) df_model = pd.concat([df,
```

```
dummies.drop('other', axis=1)], axis=1) df_model.drop(columns=['size', 'location',
'price_per_sqft'], inplace=True) for col in df_model.columns: if df_model[col].dtype ==
'object': df_model[col] = pd.to_numeric(df_model[col], errors='coerce')
df_model.dropna(inplace=True) X = df_model.drop('price', axis=1) y = df_model['price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
lr = LinearRegression() rf = RandomForestRegressor(random_state=42) lr.fit(X_train,
y_train) rf.fit(X_train, y_train) y_pred_lr = lr.predict(X_test) y_pred_rf =
rf.predict(X_test) def evaluate(model_name, y_true, y_pred): mae =
mean_absolute_error(y_true, y_pred) rmse = np.sqrt(mean_squared_error(y_true, y_pred)) r2
= r2_score(y_true, y_pred) print(f"{model_name} Performance:") print(f" MAE : {mae:.2f}")
print(f" RMSE: {rmse:.2f}") print(f" R2 : {r2:.3f}\n") evaluate("Linear Regression",
y_test, y_pred_lr) evaluate("Random Forest", y_test, y_pred_rf)
```