**Assignment: Predicting Housing Prices**

**1. Load the Data**

* Load the Boston Housing dataset from sklearn.datasets.
* Print the dataset's shape, feature names, and a brief description.

**2. Explore and Preprocess the Data**

* Explore the dataset:
  + Print the first few rows.
  + Check for any missing values.
  + Describe the dataset to understand the statistical properties.
* Handle missing values if any.
* Split the data into features (X) and target variable (y).

**3. Split the Data**

* Split the data into training and testing sets using train\_test\_split from sklearn.model\_selection.

**4. Train a Random Forest Regressor**

* Import RandomForestRegressor from sklearn.ensemble.
* Instantiate the model and fit it on the training data.
* Predict the target variable on the test data.

**5. Evaluate the Random Forest Model**

* Evaluate the model's performance using metrics like Mean Squared Error (MSE) and R² score from sklearn.metrics.
* Print the evaluation results.

**6. Train an XGBoost Regressor**

* Import XGBRegressor from xgboost.
* Instantiate the model and fit it on the training data.
* Predict the target variable on the test data.

**7. Evaluate the XGBoost Model**

* Evaluate the model's performance using the same metrics as for the Random Forest model.
* Print the evaluation results.

**8. Hyperparameter Tuning**

* Perform hyperparameter tuning for both models:
  + Use GridSearchCV or RandomizedSearchCV from sklearn.model\_selection to find the best hyperparameters.
* Print the best hyperparameters and retrain the models with these settings.
* Evaluate the performance of the tuned models.

**9. Compare the Models**

* Compare the performance of the Random Forest and XGBoost models before and after tuning.
* Discuss which model performed better and why.

**Implementation Steps (Without Code)**

1. **Load the Data**
   * Import the Boston Housing dataset.
   * Load the data and assign features to X and the target to y.
2. **Explore and Preprocess the Data**
   * Print the dataset description.
   * Check the shape and feature names.
   * Convert the data to a DataFrame for better exploration.
   * Check for missing values and handle them if present.
   * Split the data into training and testing sets.
3. **Train a Random Forest Regressor**
   * Import the RandomForestRegressor.
   * Instantiate and fit the model on the training data.
   * Make predictions on the test data.
4. **Evaluate the Random Forest Model**
   * Use metrics such as Mean Squared Error (MSE) and R² score to evaluate the model.
   * Print the evaluation results.
5. **Train an XGBoost Regressor**
   * Import the XGBRegressor.
   * Instantiate and fit the model on the training data.
   * Make predictions on the test data.
6. **Evaluate the XGBoost Model**
   * Use the same evaluation metrics to assess the XGBoost model.
   * Print the evaluation results.
7. **Hyperparameter Tuning**
   * Use GridSearchCV or RandomizedSearchCV to find the best hyperparameters for both models.
   * Print the best hyperparameters.
   * Retrain the models with the best settings and evaluate their performance.
8. **Compare the Models**
   * Compare the performance of both models based on evaluation metrics.
   * Discuss the strengths and weaknesses of each model and which performed better for this dataset.

By following this outline and implementing the code yourself, you will gain practical experience with Random Forest and XGBoost for regression tasks. Good luck!