1. **XGBoost with Hyperparameter Tuning on the Pima Indians Diabetes Dataset**
   o **Dataset**: Use the *Pima Indians Diabetes Dataset* from Kaggle.
   o **Tasks**:
       1. Load the dataset, preprocess it, and split it into training and testing sets (80% train, 20% test).
       2. Initialize an XGBoost Classifier and tune `max_depth` (values: 3, 5, 7), `learning_rate` (values: 0.01, 0.1, 0.2), and `n_estimators` (values: 50, 100, 150).
       3. For each combination of parameters, evaluate the model using accuracy, precision, recall, and F1-score.
       4. Plot ROC curves for the top three configurations and compare their AUC values.
       5. Analyze which combination provides the best balance between precision and recall, discussing any observed trade-offs.

2. **XGBoost Regression on Boston Housing Dataset**
   o **Dataset**: Use the *Boston Housing Dataset* (available in `sklearn.datasets`).
   o **Tasks**:
       1. Load and split the dataset into training and testing sets (80% train, 20% test).
       2. Initialize an XGBoost Regressor with `n_estimators=200` and experiment with `subsample` (values: 0.6, 0.8, 1.0) and `colsample_bytree` (values: 0.6, 0.8, 1.0).
       3. Train models with different combinations of `subsample` and `colsample_bytree` values.
       4. Evaluate each model using R² and Mean Squared Error (MSE) and plot the results.
       5. Discuss how subsampling impacts the trade-off between computational efficiency and accuracy.

3. **Imbalanced Classification with XGBoost on Credit Card Fraud Detection**
   o **Dataset**: Use the *Credit Card Fraud Detection Dataset* from Kaggle.
   o **Tasks**:
       1. Load and preprocess the dataset, ensuring class imbalance is handled (fraudulent vs. non-fraudulent transactions).
       2. Split the data into training and testing sets (75% train, 25% test).
       3. Implement XGBoost with tuning for `scale_pos_weight` to handle the class imbalance (try values: 1, 10, 25, 50).
       4. Evaluate each model configuration using the confusion matrix, precision, recall, and F1-score.
       5. Create a bar plot showing precision and recall for each `scale_pos_weight` value and discuss how this parameter improves the handling of imbalanced data.