 **Import Libraries**: This step involves importing the necessary Python libraries:

* os: For interacting with the operating system, such as file paths.
* pandas as pd: For data manipulation and analysis, specifically reading and writing CSV files.
* PIL (Python Imaging Library): For opening and manipulating image files.
* torchvision.models and torchvision.transforms: For accessing pre-trained deep learning models and applying image transformations.
* torch: For performing tensor operations and model inference in PyTorch.
* sklearn.ensemble.GradientBoostingRegressor: For using the Gradient Boosting Regressor model for regression tasks.

 **Load Data**: Reads the training and test datasets from CSV files into Pandas DataFrames. The pd.read\_csv function is used to load the data from the specified file paths into DataFrames named train\_data and test\_data.

 **Assign Image Paths**: Constructs the full file paths for images in the training and test datasets. The file names are extracted from the image\_link column by splitting the URL and taking the last part (the actual file name). These file names are then appended to the directory paths (train\_images/ and test\_images/) to form the complete paths.

 **Define Entity-Unit Mapping**: Sets up a dictionary entity\_unit\_map that maps various entities (like width, depth, height) to their possible units of measurement (like centimetre, inch, meter). This is used to standardize units for different types of measurements in the dataset.

 **Map Default Units**: Defines a function get\_default\_unit that returns a default unit for each entity type. This function looks up the entity name in the entity\_unit\_map and returns the first unit in the set for that entity. This default unit is then added to the test\_data DataFrame based on the entity\_name column.

 **Image Feature Extraction**:

* Loads a pre-trained ResNet50 model using models.resnet50(pretrained=True). This model is pre-trained on a large dataset (ImageNet) and is used to extract features from images.
* Sets the model to evaluation mode with model.eval() to prepare it for inference.
* Defines a series of image transformations using transforms.Compose, including resizing the image to 224x224 pixels, converting it to a tensor, and normalizing it using mean and standard deviation values specific to ImageNet.
* Defines the function extract\_image\_features to process each image: open the image file, apply transformations, pass it through the model to get feature vectors, and return these feature vectors as numpy arrays.

 **Prepare Training Data**:

* Applies the extract\_image\_features function to the image\_path column in both the training and test datasets to get feature vectors.
* Extracts feature vectors (X\_train) and labels (y\_train) from the training data. The dropna() method is used to remove any rows with missing data.

 **Train Model**:

* Initializes a Gradient Boosting Regressor model with GradientBoostingRegressor().
* Trains the model using the fit method with the feature vectors and labels. The model learns the relationship between the features and the target labels during this step.

 **Make Predictions**:

* Uses the trained model to make predictions on the test data. The predict method is called with the feature vectors from the test dataset to generate predicted values.

 **Format Predictions**:

* Defines the format\_prediction function to format the predicted values by appending the appropriate unit. If the value is missing, it returns an empty string.
* Applies this function to the value\_pred column in the test data to generate the final formatted predictions.

 **Save Predictions**:

* Writes the formatted predictions to a CSV file named test\_out.csv using the to\_csv method of the Pandas DataFrame. Only the index and prediction columns are included in the output file.

 **Run Sanity Checker** (Optional):

* Executes an external script sanity.py to verify that the output file test\_out.csv meets the expected format requirements. This step ensures that the predictions are correctly formatted and ready for submission or further use.