# AI TechStack 2025
# Week 4 Task

September 7, 2025

## 1 Cat and Dog Classification

### Objective

Develop a classical machine learning pipeline to classify images of cats and dogs using Histogram of Oriented Gradients (HOG) for feature extraction and Decision Tree (DT) for classification.

### Task Description

You are provided with a dataset of cat and dog images in the `'dataset'` directory, with filenames starting with `cat` or `dog` (e.g., `cat.1.jpg`, `dog.1.jpg`). Implement a pipeline to:

- Extract HOG features from grayscale images resized to 128×128 pixels.
- Split the dataset into 80% training and 20% test sets, ensuring class balance.
- Train a Decision Tree classifier on the extracted features.
- Test your model and show result.

Use `scikit-image` for HOG feature extraction, `scikit-learn` for Decision Tree classification and data splitting, and `numpy-matplotlib` for data handling and visualization. Ensure the code supports adjustable parameters (e.g., HOG settings, train-test split ratio). Visualize sample results, including original images, HOG representations, and predicted labels.

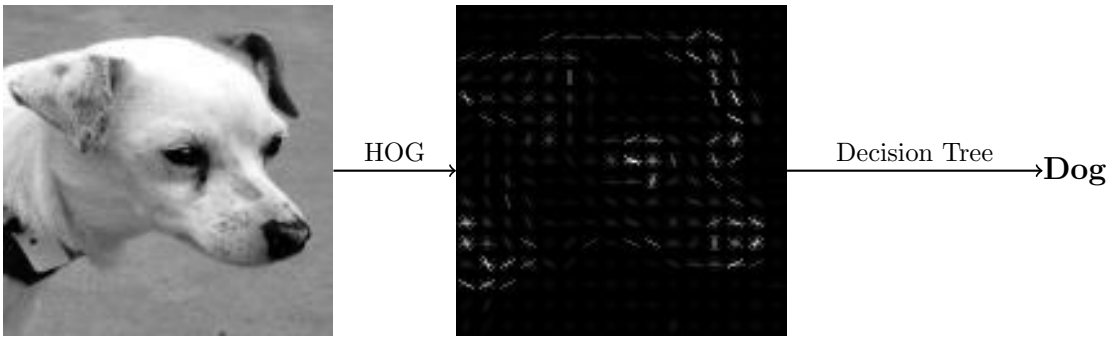


Figure 1: Pipeline visualization: An original sample image (left) is processed to extract HOG features (middle), which are then classified by a Decision Tree to predict the label (right, shown as "Dog").

### Steps to Follow

1. **Data Preparation:**
   - Load images from the `'dataset'`.
   - Assign labels (0 for cat, 1 for dog) based on filenames.
2. **HOG Feature Extraction:**

- Extract HOG features using `scikit-image` with adjustable parameters.
- Store HOG features and visualizations for each image.

3. **Train-Test Split:**
   - Split the dataset into 80% training and 20% test sets, using stratified sampling to maintain class balance.
4. **Decision Tree Classification:**
   - Train a Decision Tree classifier on the training set's HOG features.
   - Evaluate the model on the test set.
5. **Evaluation and Visualization:**
   - Compute accuracy on the test set.
   - Visualize sample original images, HOG representations, and predicted labels from the test set.

## Improvements

Consider enhancements like:

- Using alternative classifiers (e.g., Random Forest, SVM) for improved accuracy.
- Adjusting HOG parameters (e.g., `pixels_per_cell`) to capture more detailed features.
- Implementing data augmentation or preprocessing to enhance robustness.

## Evaluation and Submission

Submissions are evaluated based on test set accuracy and quality of the train-test split (e.g., class balance). Submit:

- Complete pipeline code (data loading, train-test split, HOG extraction, classification).
- Visualization of sample results (original images, HOG, and predicted labels).
- Brief report on approach, test accuracy, dataset split details, and any improvements.
- Summery of what you learn from HOG and DT.