Team Name: LSD

Members: Khan Farhan Yusuf, Leung Yau Yee, Tang Jiaxuan

1. Dataset analysis

The dataset was split into **50,000 training** samples, **2,000 public testing** samples, and **8,000 private testing** samples.

- Number of categories

```
data_path = './data/train.csv'
dataset_path = './data/train_ims'
df = pd.read_csv(data_path)

category_counts = df['label'].value_counts()
print("Category Counts:")
print(category_counts)
```

```
Category Counts:
label
2    5038
8    5020
1    5012
0    5010
3    5007
7    5000
4    4995
5    4993
9    4970
6    4955
Name: count, dtype: int64
```

- Visualisation

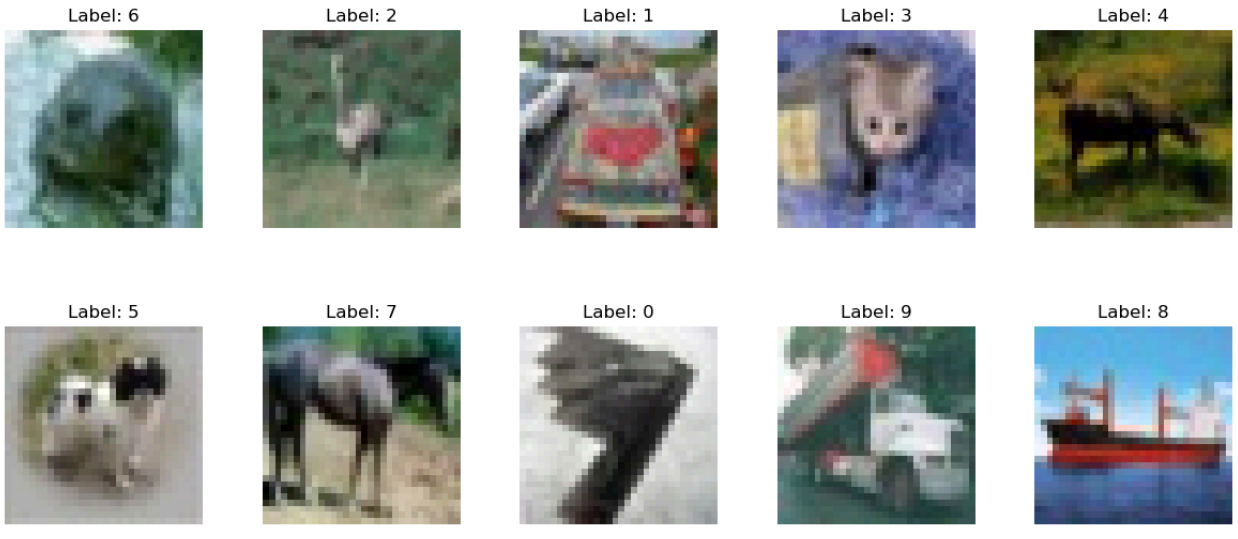
```
plt.figure(figsize=(10, 6))
plt.bar(category_counts.index, category_counts.values, color='skyblue')
plt.xlabel('Label')
plt.ylabel('Number of Images')
plt.title('Number of Images per Category')
plt.show()

fig, axs = plt.subplots(2, 5, figsize=(15, 6))
plt.subplots_adjust(hspace=0.5)
unique_labels = df['label'].unique()

for i, label in enumerate(unique_labels):
    ax = axs[i // 5, i % 5]
    img_name = df[df['label'] == label].iloc[0]['im_name']
    img_path = os.path.join(dataset_path, img_name)

    try:
        img = Image.open(img_path)
        ax.imshow(img)
        ax.set_title(f'Label: {label}')
        ax.axis('off')
    except Exception as e:
        print(f"Could not load image: {img_name}, Error: {e}")

plt.show()
```



2. Classifier exploration

For this experiment, we explored two classifiers: **Support Vector Machines (SVM)** and **Random Forest Classifier (RFC)**. SVM was chosen due to its effectiveness in handling high-dimensional datasets, while RFC was selected for its robustness and ability to capture non-linear relationships. Both classifiers were evaluated to determine their suitability for the given classification task.

The dataset was split into **50,000 training** samples, **2,000 public testing** samples, and **8,000 private testing** samples.

The SVM classifier utilised HOG feature extraction which works by analysing the **gradients** (changes in intensity) of an image to capture the structure, edge, and shape information. The RBF kernel was chosen for this classifier as it is optimal for non-linear problems such as image classification, as it maps the data into a higher-dimensional space where it becomes linearly separable. It also utilised GridSearchCV.

The Random Forest classifier utilised HOG and Gabor feature extraction as well as analysed the images in greyscale. It then scaled the features using StandardScaler, and utilised PCA dimensionality reduction. It then handled class imbalance with SMOTEENN and utilised the bagging classifier as well as GridSearchCV.

Classifier performance was evaluated using **accuracy**. The SVM classifier had a 64% accuracy while the random forest classifier had a 50% accuracy. The **higher accuracy of the SVM classifier** can be explained by:

1. SVM's ability to handle high-dimensional HOG features more effectively.
2. The use of the RBF kernel, which captures non-linear patterns in the data.
3. The absence of PCA, preserving more information in the feature space for SVM.
4. Random Forest's sensitivity to high-dimensional features and potential overfitting.

3. Final pipeline

Since we discovered higher accuracy in SVM we finally adopted it. The final pipeline has the following structure:

Data -> GrayScale -> HOG -> StandardScaler -> PCA -> SVM with rbf kernel

(where we implement grayscale in HOG class)

Preprocessing Steps:

- Images are converted to grayscale when necessary (if ndim != 3)(`rgb2gray`).
- Resizing is applied uniformly to match a predefined shape (`32x32` in this case) using `skimage.transform.resize`. (if some image is not 32x32)

A custom `HOGFeatureExtractor` class is implemented to extract features from the images:

- **Histogram of Oriented Gradients (HOG):**
    - Captures texture and edge information, essential for object classification.
    - Key parameters:
        - `resize_shape`: Resizes images to standard dimensions (e.g., `32x32`).
        - `orientations`: Number of gradient orientation bins (e.g., `9`).
        - `pixels_per_cell`: Size of each cell for computing histograms (e.g., `8x8`).
        - `cells_per_block`: Number of cells per block for normalization (e.g., `3x3`).
    - Returns a feature vector for each image, which is then passed to the classifier.

Pipeline construction:

- **HOG Extraction:** Extracts meaningful features from images.
- **Feature Standardisation:** A `StandardScaler` is used to standardize the feature set, ensuring a mean of 0 and variance of 1, which is critical for PCA.
- **Dimensionality Reduction:** PCA is applied to reduce the dimensionality of the feature space:
    - Retains 99% of the variance (`n_components=0.99`), reducing noise and computational load.
- **Classification: SVM** with an RBF kernel:
    - Hyperparameters: `C=10` for regularization and `gamma='auto'` for kernel coefficient.

Fine-tune Parameters:

- GridSearch was used to find the optimal parameters, where cross validation cv=3 was used to determine best accuracy. The optimal parameters were then set on the pipeline.

Evaluation:

- The final pipeline returned an accuracy of 0.65950. Greater accuracy could have been achieved if majority voting had been used.