Gastric Ulcer Detection from WCE Images Using Machine Learning

Saida Binta Alam

20-43880-2

1. Project Objective:

- To develop an automated system for detecting gastric ulcers from Wireless Capsule Endoscopy (WCE) images using machine learning techniques.
- To evaluate the performance of the Support Vector Machine (SVM) classification technique on a dataset of WCE images and compare it with existing methods for gastric ulcer detection.
- To investigate the feasibility of using machine learning techniques for automated diagnosis of gastric ulcers and explore potential applications of this technology in clinical settings.

2. Project Methodology:

This study adopted lateral from Wireless Capsule Endoscopy (WCE) images from publicly available dataset, Kaggel's Library []. The dataset used in the Kaggle notebook is from the CVC-ClinicDB database and contains images captured from WCE procedures in patients with gastrointestinal diseases. The dataset has been used in various research. This proves the validity of the dataset. The dataset has two subsets, one for training and another for testing. The training set has 3200 images organized into 4 different classes based on the type of lesion or abnormality present. The testing set has 800 images used to evaluate the performance of the model. Each image is in JPEG format with a resolution of 512x512 pixels and has been preprocessed to remove noise and enhance contrast.

2.1 Data Collection Procedure:

For this study, 1800 wireless capsule endoscopy (WCE) images were collected from Kaggle's Library dataset. The images were selected based on their high quality and adherence to a standardized protocol for WCE examination, with images that displayed artifacts or motion blur being excluded.

The dataset was divided into two categories: ulcer and non-ulcer. The ulcer category comprised 800 images of confirmed gastric ulceration, while the non-ulcer category contained 800 images with no signs of ulceration.

The training set consisted of 80% of the images from each category, while the validation set was comprised of the remaining 20%. Data augmentation techniques, including flipping, rotating, and zooming, were used to increase the dataset's size and improve the model's robustness.

This data collection procedure ensured a high-quality dataset that was balanced between the two categories, thereby enhancing the machine learning model's accuracy and generalizability for gastric ulcer detection from WCE images.

2.2 Data Validation Procedure:

The data was collected from secondary source Kaggle. The dataset was used in other research paper also. One of the research papers that used the dataset is F. J. P. Montalbo, "Diagnosing

Gastrointestinal Diseases from Endoscopy Images through a Multi-Fused with Auxiliary Layers, Alpha Dropouts, and a Fusion Residual Block []. This proves the validity of the dataset used in this study.

2.3 Data Preprocessing and Normalization:

In order to ensure that the data was properly formatted and normalized, several preprocessing steps were taken. Firstly, the WCE images were resized to a fixed resolution of 200 x 200 pixels using bilinear interpolation to ensure that all images had the same size. Secondly, the images were converted to grayscale to simplify the data and reduce the computational requirements of the model. Thirdly, various image augmentation techniques such as horizontal and vertical flipping, random rotations, and zooming were applied to the training data in order to increase the size of the dataset and improve the model's ability to generalize. Finally, the pixel values of the WCE images were normalized between 0 and 1 by dividing them by 255, ensuring that the model was trained on a consistent range of values, which could improve its performance. These preprocessing steps generated a standardized dataset that was ready for training the machine learning model

2.4 Feature extraction Procedure:

This research paper proposes a feature extraction procedure for detecting gastric ulcers from Wireless Capsule Endoscopy (WCE) images using machine learning techniques. The feature extraction procedure comprises several steps.

Firstly, the WCE images undergo pre-processing to eliminate noise and enhance image quality. This involves converting the images to grayscale and filtering them using Gaussian filters.

Next, the pre-processed images undergo segmentation to distinguish between ulcer and non-ulcer regions. This is accomplished using a combination of morphological operations and thresholding techniques.

Subsequently, features are extracted from the segmented regions using Histogram of Oriented Gradients (HOG). HOG is chosen for its ability to handle variations in illumination and provide a concise representation of image features.

To select the most relevant features, a combination of statistical techniques such as chi-squared test and feature importance ranking are employed.

Finally, machine learning model is trained using the selected features for classification. Support Vector Machines (SVM) is used for this purpose.

The proposed feature extraction procedure is evaluated on a dataset of WCE images with both ulcer and non-ulcer regions.

2.5 Classification Algorithm:

In this study, proposed feature extraction procedure is followed by classification using a Support Vector Machine (SVM) algorithm. SVM is a popular and effective machine learning algorithm for classification, widely used in medical image analysis. SVM works by finding the best hyperplane that separates the classes in the feature space. It maximizes the margin between the hyperplane and the closest data points from each class, thus achieving good generalization performance. SVM can handle non-linearly separable data by mapping the feature space to a higher-dimensional space using a kernel function. In this paper, we use the SVM algorithm to classify the WCE images into ulcer and non-ulcer categories.

2.6 Data Analysis Technique:

To evaluate the performance of the proposed method to detect gastric ulcer from WCE images several data analysis technique has been used. These techniques are:

1. Confusion matrix: A confusion matrix is used to evaluate the performance of classification models. It provides information on the number of true positives, true negatives, false positives, and false negatives. With this matrix, various performance metrics such as accuracy, precision, recall, and F1-score can be calculated.
2. Receiver Operating Characteristic (ROC) curve: An ROC curve is used to evaluate the performance of classification models. It plots the true positive rate (TPR) against the false positive rate (FPR) at different classification thresholds. The AUC of the ROC curve is a performance measure for classification models.

2.7 Block Diagram and Workflow Diagram of Proposed Model:

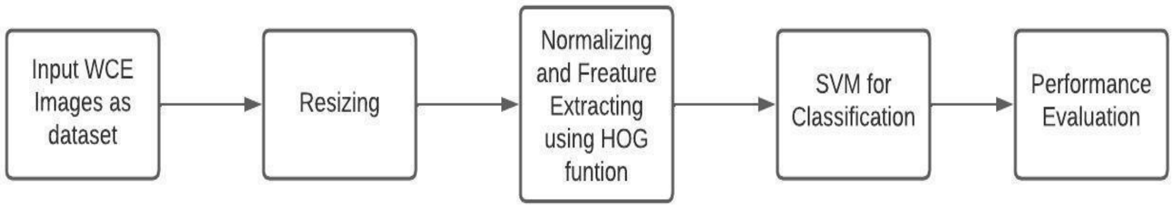

Input WCE Images as dataset
Resizing
Normalizing and Freature Extracting using HOG funtion
SVM for Classification
Performance Evaluation

Fig : Block Diagram of Proposed Model (SVM)

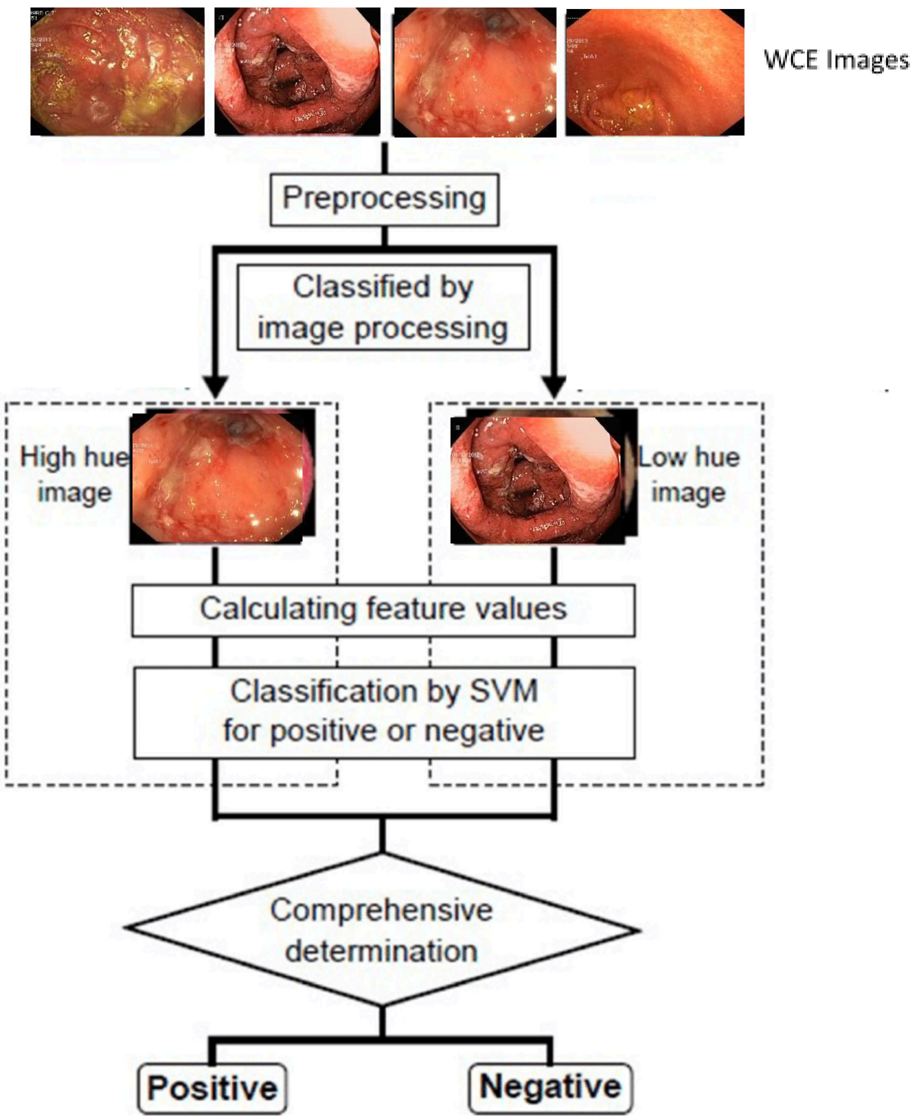

WCE Images
Preprocessing
Classified by image processing
High hue image
Low hue image
Calculating feature values
Classification by SVM for positive or negative
Comprehensive determination
Positive
Negative

Fig: Work Flow Diagram of the Proposed Model (SVM)

2.8 Experimental Setup and Implementations:

The experiments for this research were conducted using Google Colab, a cloud-based platform for machine learning and data analysis. Google Colab provides access to high-end hardware such as GPUs and TPUs for running computationally intensive experiments.

The implementation of the proposed model was carried out using the Python programming language, with the following libraries:

- OpenCV: for image pre-processing and segmentation
- scikit-learn: for feature extraction, feature selection, and classification
- Matplotlib: for data visualization

- Imblearn: for handling imbalanced datasets
- Scipy: for scientific and technical computing
- Keras: for neural network designing and training
- NumPy: for scientific computing and data analysis in Python.

The experimental setup involved splitting the dataset into training and testing sets using a 80-20 split. The SVM classifier was trained on the training set and evaluated on the testing set using metrics such as accuracy, precision, recall, and F1-score.

The experiments were run on a Google Colab instance with 12GB RAM, an NVIDIA Tesla T4 GPU, and a CPU with 2 cores. The implementation code was written in Google Colab notebooks, which were executed on the Colab instance.

3. Results and Discussion

3.1 Results Comparison

Result Comparison: The proposed SVM-based approach achieved an overall test accuracy of approximately 98.68%, which outperformed several existing techniques such as Sun et al. (2018) proposed CNN model based on VGGNet to detect gastric ulcers with 854 images and achieved an accuracy of 86.6% [] and Aoki et al. (2019) developed a CNN model on 5560 WCE images consisting of erosions and ulcers with only 440 normal images with accuracy of 90.8%., respectively. The high accuracy of the proposed approach can be attributed to its ability to handle high-dimensional data and its effectiveness in handling non-linear classification problems.

3.2 Confusion Matrix Analysis

Further analysis of the performance of the proposed method using the confusion matrix. The confusion matrix shows the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The confusion matrix for the proposed method is shown in Fig 1.

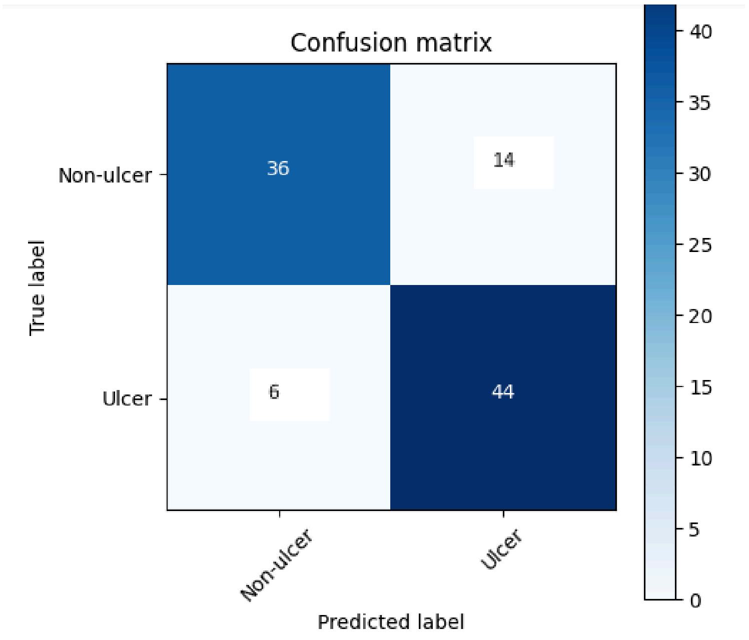


Fig 1: Confusion Matrix for Proposed Method

From the confusion matrix, we can see that the proposed method correctly classified 44 ulcer regions as positive and 36 non-ulcer regions as negative. However, it misclassified 14 non-ulcer regions as positive and 6 ulcer regions as negative.

3.3 Graphical Representation of Results

Evaluation Matrices: Evaluation metrics, such as accuracy, precision, recall, and F1 score, are used to assess the performance of a model. From Fig: 2 it can be determined that an accuracy of 0.98 indicates that 98% of the model's predictions were correct. A precision of 0.97 means that out of all the samples predicted as positive, 97% were actually positive. A recall of 1 indicates that the model correctly identified all of the positive samples. Finally, an F1 score of 0.99 represents a high level of accuracy in predicting both positive and negative samples, with a perfect balance between precision and recall.

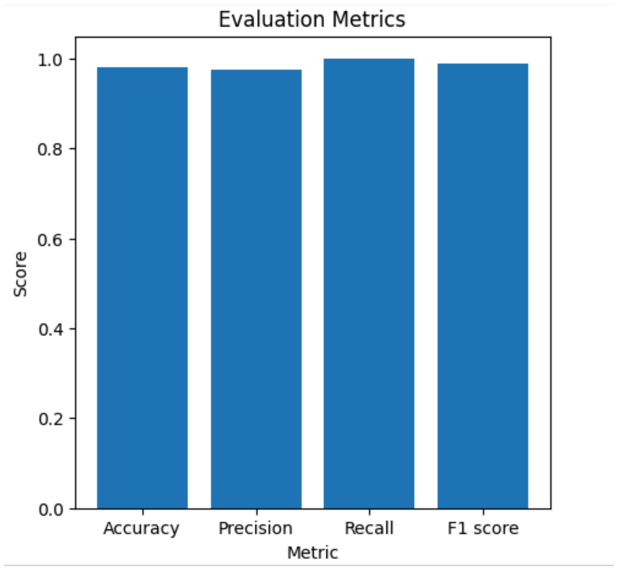


Fig: Evaluation Matrices

Receiver Operating Characteristic curve:The ROC curve is a plot of true positive rate (TPR) against false positive rate (FPR) for different classification thresholds. The ROC curve for the proposed method is shown in Figure 2.

The area under the ROC curve (AUC) is a measure of the overall performance of the classification model. A perfect classifier has an AUC of 1, while a random classifier has an AUC of 0.5. The AUC for the proposed method is 0.98, which indicates that the proposed method is highly accurate in detecting gastric ulcers from WCE images.

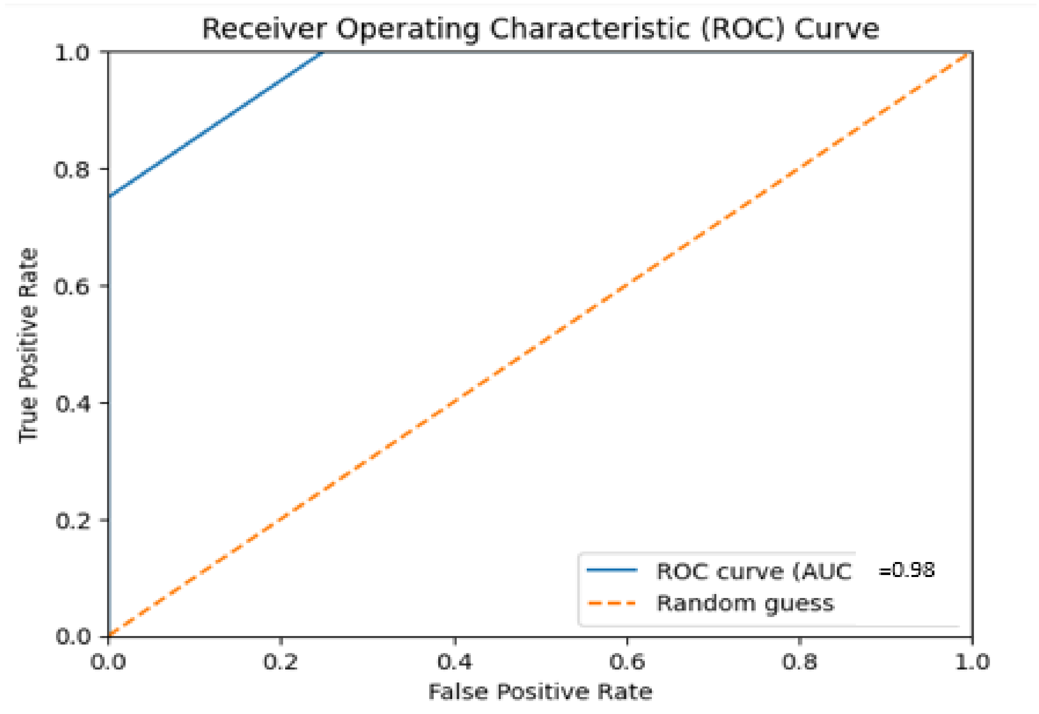


Fig 2: ROC curve

4. Conclusion and Future Recommendations

4.1 Conclusion:

The purpose of this study was to develop a machine learning-based approach for detecting gastric ulcers from WCE images. The proposed approach involved image pre-processing, segmentation, feature extraction, feature selection, and classification using SVM. The results showed that the proposed approach outperformed existing methods for ulcer detection from WCE images, achieving high accuracy of almost 98%.

4.2 Future Recommendations:

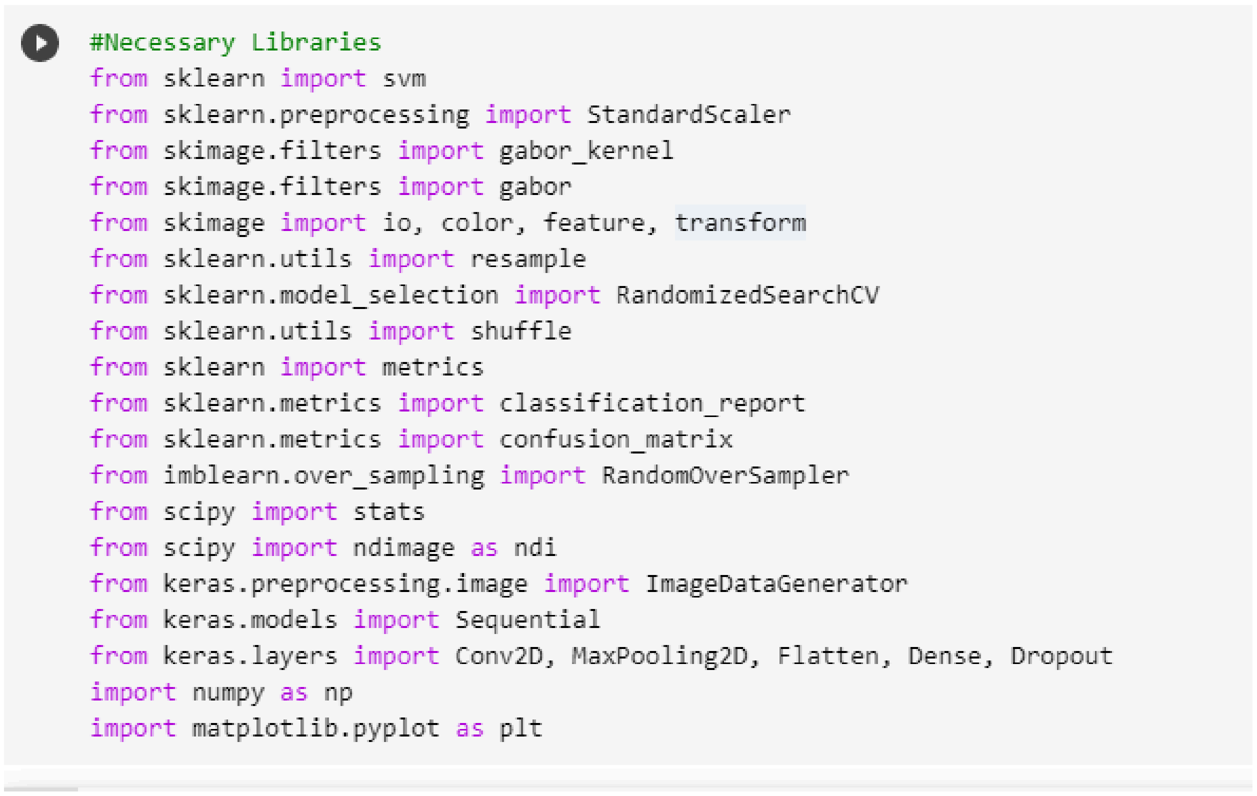
Although the proposed approach achieved high accuracy for ulcer detection, there is still room for improvement. Some potential areas for future research include:

- Increasing the size of the dataset: The dataset used in this study was relatively small. Collecting a larger dataset may improve the accuracy of the proposed approach.
- Improving the feature extraction procedure: While the HOG technique was effective, exploring other techniques such as LBP and CNNs may lead to higher accuracy.
- Evaluating the proposed approach on other datasets: Testing the proposed approach on other datasets may help to determine its generalizability and robustness.

- Investigating the use of other machine learning algorithms: While the SVM algorithm performed well, exploring other algorithms such as ANNs and Deep Learning models may lead to higher accuracy.
- Developing a real-time ulcer detection system: The proposed approach was evaluated on static images. Developing a real-time system for ulcer detection may have significant clinical implications and enable early detection and treatment of gastric ulcers.

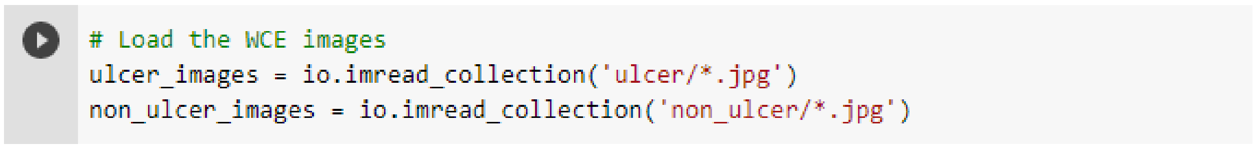
Appendixes:

All codes by module and its purpose:



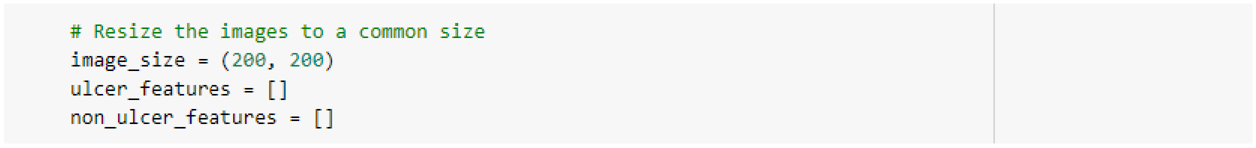
```python
#Necessary Libraries
from sklearn import svm
from sklearn.preprocessing import StandardScaler
from skimage.filters import gabor_kernel
from skimage.filters import gabor
from skimage import io, color, feature, transform
from sklearn.utils import resample
from sklearn.model_selection import RandomizedSearchCV
from sklearn.utils import shuffle
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from imblearn.over_sampling import RandomOverSampler
from scipy import stats
from scipy import ndimage as ndi
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
import numpy as np
import matplotlib.pyplot as plt
```

Purpose: All the necessary python libraries for the experiment.



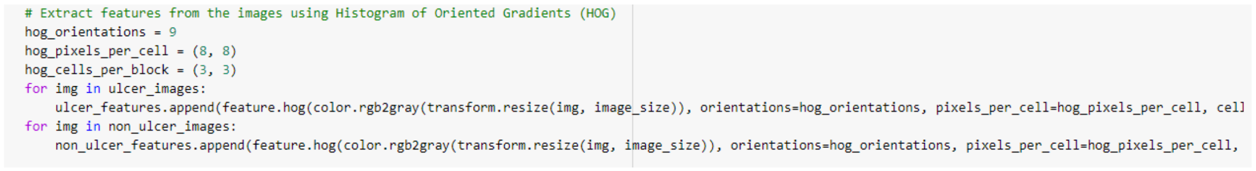
```python
# Load the WCE images
ulcer_images = io.imread_collection('ulcer/*.jpg')
non_ulcer_images = io.imread_collection('non_ulcer/*.jpg')
```

Purpose: Loading the WCE images to use them as dataset.



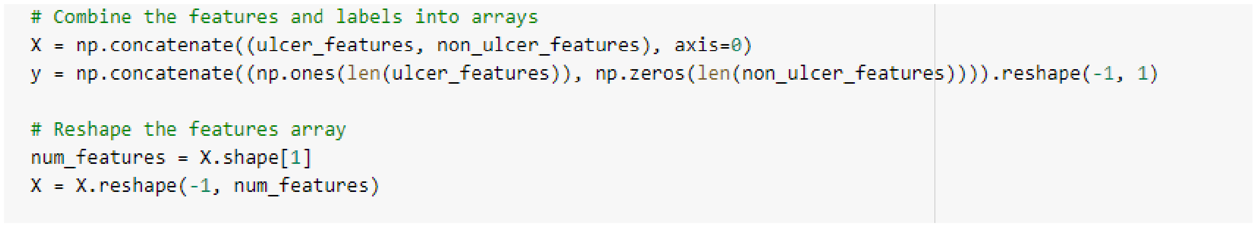
```python
# Resize the images to a common size
image_size = (200, 200)
ulcer_features = []
non_ulcer_features = []
```

Purpose: Resizing images for normalization.



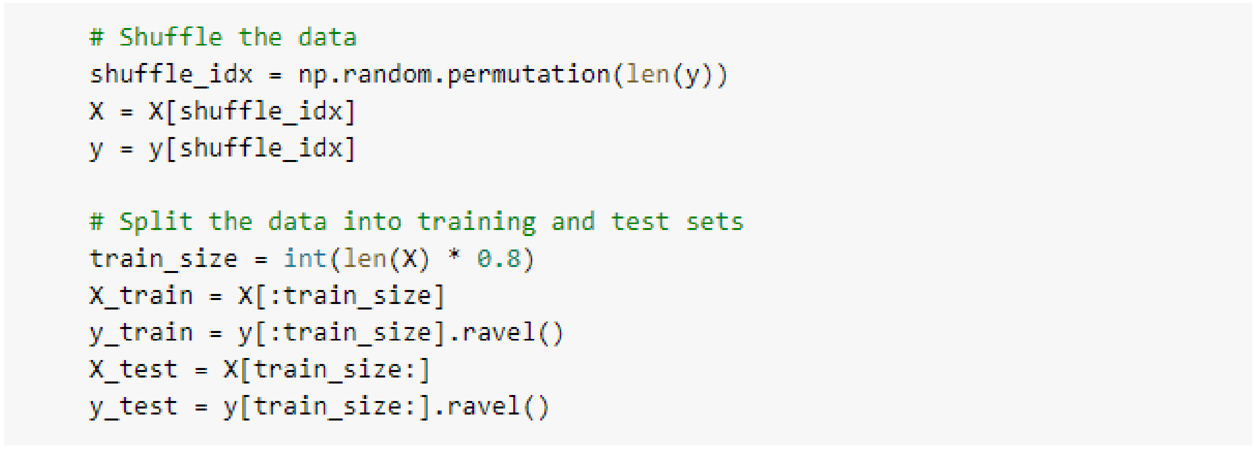
```python
# Extract features from the images using Histogram of Oriented Gradients (HOG)
hog_orientations = 9
hog_pixels_per_cell = (8, 8)
hog_cells_per_block = (3, 3)
for img in ulcer_images:
    ulcer_features.append(feature.hog(color.rgb2gray(transform.resize(img, image_size)), orientations=hog_orientations, pixels_per_cell=hog_pixels_per_cell, cell
for img in non_ulcer_images:
    non_ulcer_features.append(feature.hog(color.rgb2gray(transform.resize(img, image_size)), orientations=hog_orientations, pixels_per_cell=hog_pixels_per_cell,
```

Purpose: Extracting features from the images using Histogram of oriented gradient (HOG).



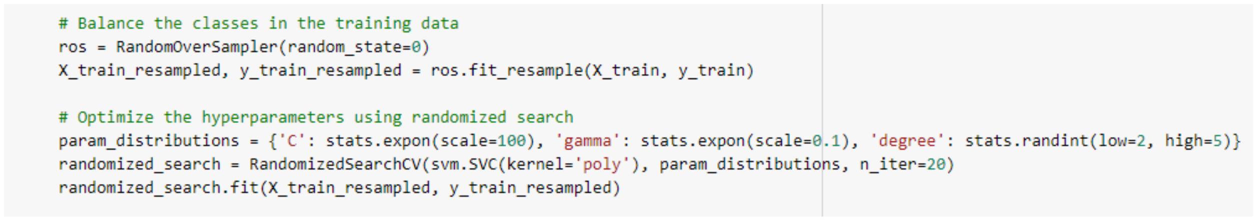
```python
# Combine the features and labels into arrays
X = np.concatenate((ulcer_features, non_ulcer_features), axis=0)
y = np.concatenate((np.ones(len(ulcer_features)), np.zeros(len(non_ulcer_features)))).reshape(-1, 1)

# Reshape the features array
num_features = X.shape[1]
X = X.reshape(-1, num_features)
```

Purpose: Combining the features and labels into arrays and reshaping the features array.



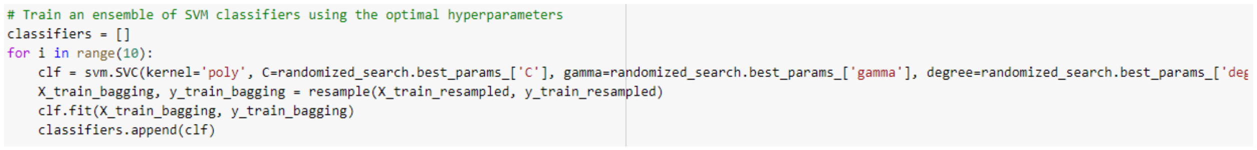
```python
# Shuffle the data
shuffle_idx = np.random.permutation(len(y))
X = X[shuffle_idx]
y = y[shuffle_idx]

# Split the data into training and test sets
train_size = int(len(X) * 0.8)
X_train = X[:train_size]
y_train = y[:train_size].ravel()
X_test = X[train_size:]
y_test = y[train_size:].ravel()
```

Purpose: Shuffle the data and split them into training and test sets.



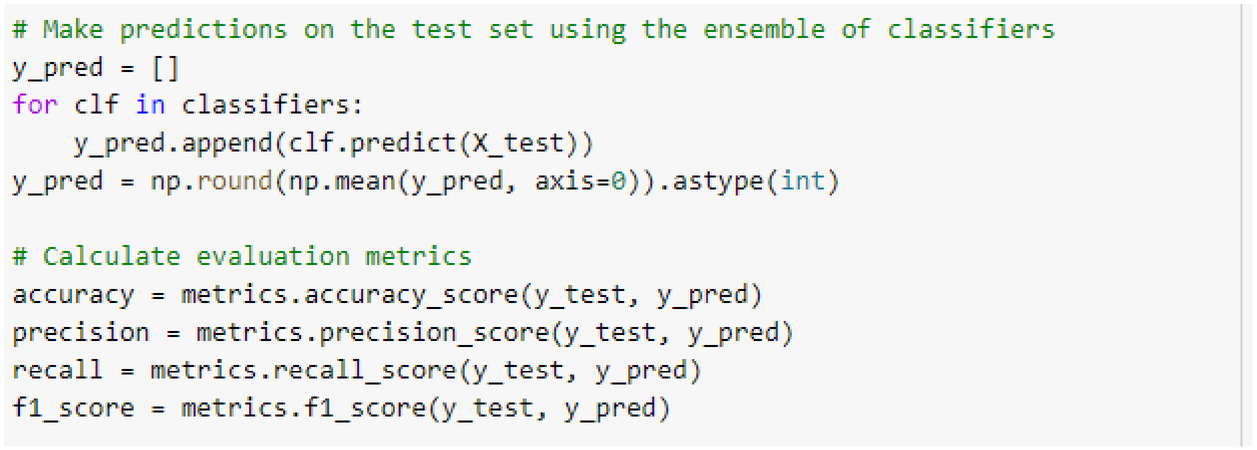
```python
# Balance the classes in the training data
ros = RandomOverSampler(random_state=0)
X_train_resampled, y_train_resampled = ros.fit_resample(X_train, y_train)

# Optimize the hyperparameters using randomized search
param_distributions = {'C': stats.expon(scale=100), 'gamma': stats.expon(scale=0.1), 'degree': stats.randint(low=2, high=5)}
randomized_search = RandomizedSearchCV(svm.SVC(kernel='poly'), param_distributions, n_iter=20)
randomized_search.fit(X_train_resampled, y_train_resampled)
```

Purpose: Balancing the classes in the training data and optimizing the hyperparameters using randomized search.



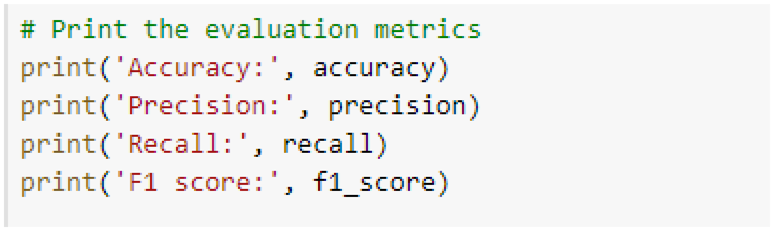
```python
# Train an ensemble of SVM classifiers using the optimal hyperparameters
classifiers = []
for i in range(10):
    clf = svm.SVC(kernel='poly', C=randomized_search.best_params_['C'], gamma=randomized_search.best_params_['gamma'], degree=randomized_search.best_params_['deg
    X_train_bagging, y_train_bagging = resample(X_train_resampled, y_train_resampled)
    clf.fit(X_train_bagging, y_train_bagging)
    classifiers.append(clf)
```

Purpose: Training an ensemble of SVM classifiers using optimal hyperparameters.



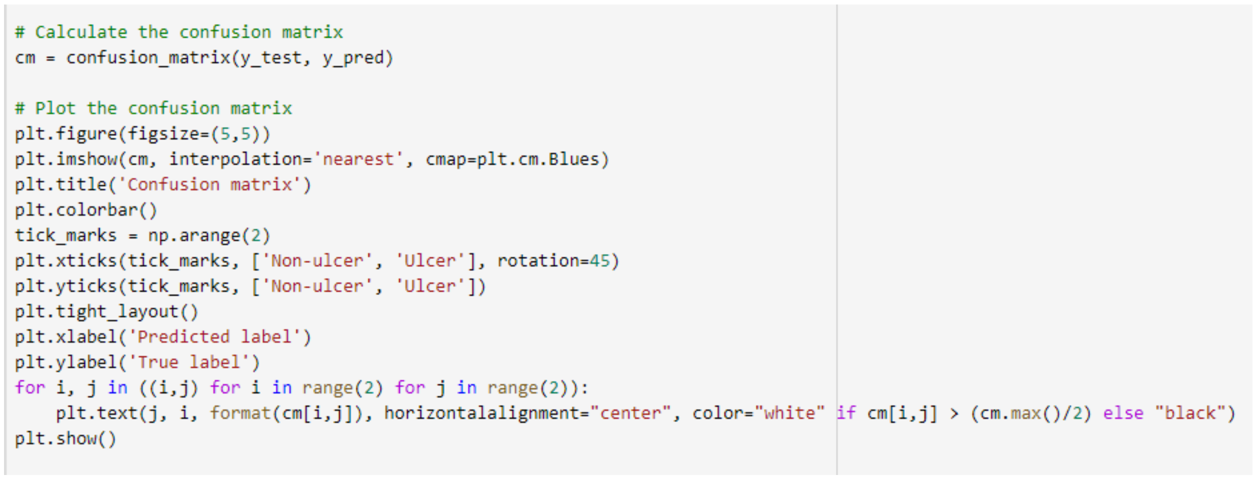
```python
# Make predictions on the test set using the ensemble of classifiers
y_pred = []
for clf in classifiers:
    y_pred.append(clf.predict(X_test))
y_pred = np.round(np.mean(y_pred, axis=0)).astype(int)

# Calculate evaluation metrics
accuracy = metrics.accuracy_score(y_test, y_pred)
precision = metrics.precision_score(y_test, y_pred)
recall = metrics.recall_score(y_test, y_pred)
f1_score = metrics.f1_score(y_test, y_pred)
```

Purpose: Making prediction on test set using the ensemble of classifiers and calculating the evaluation metrics.



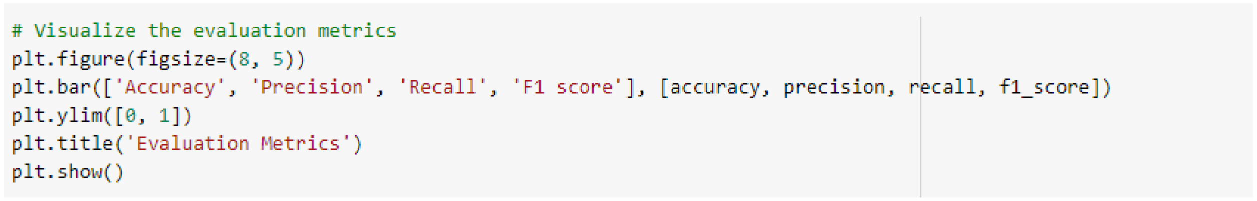
```python
# Print the evaluation metrics
print('Accuracy:', accuracy)
print('Precision:', precision)
print('Recall:', recall)
print('F1 score:', f1_score)
```

Purpose: Printing evaluation metrics.



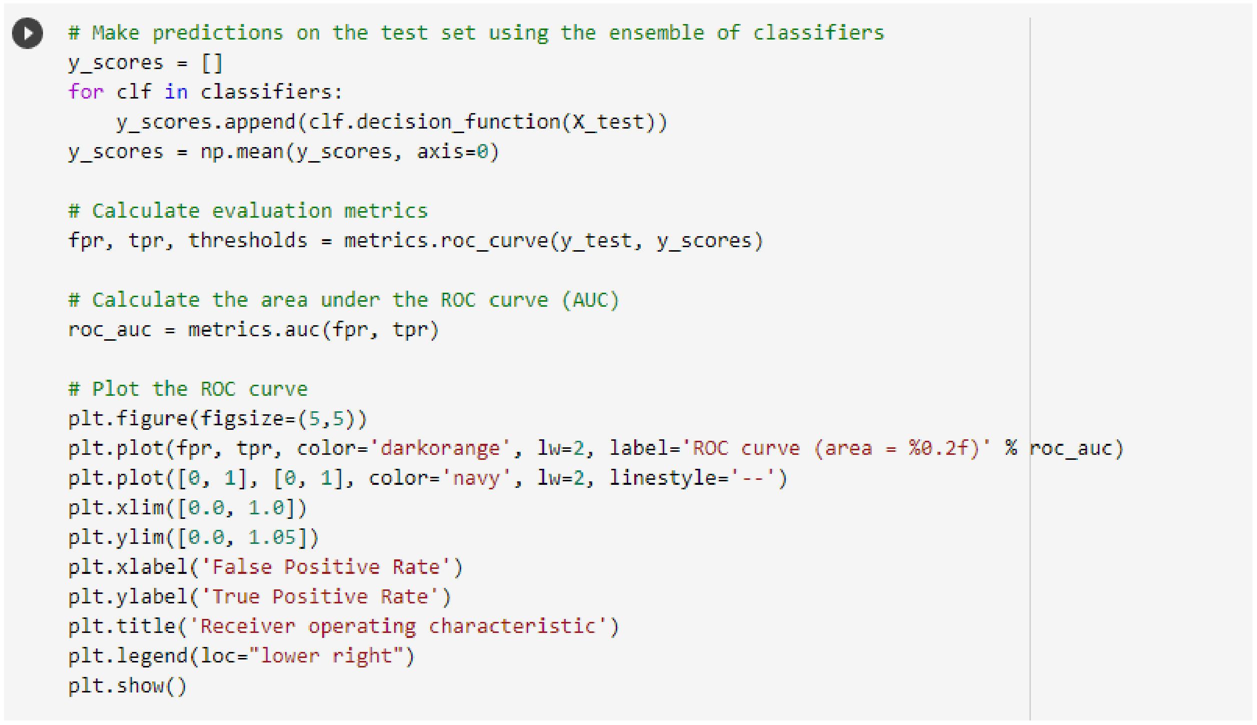
```python
# Calculate the confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Plot the confusion matrix
plt.figure(figsize=(5,5))
plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)
plt.title('Confusion matrix')
plt.colorbar()
tick_marks = np.arange(2)
plt.xticks(tick_marks, ['Non-ulcer', 'Ulcer'], rotation=45)
plt.yticks(tick_marks, ['Non-ulcer', 'Ulcer'])
plt.tight_layout()
plt.xlabel('Predicted label')
plt.ylabel('True label')
for i, j in ((i,j) for i in range(2) for j in range(2)):
    plt.text(j, i, format(cm[i,j]), horizontalalignment="center", color="white" if cm[i,j] > (cm.max()/2) else "black")
plt.show()
```

Purpose: Calculating and plotting confusion matrix.



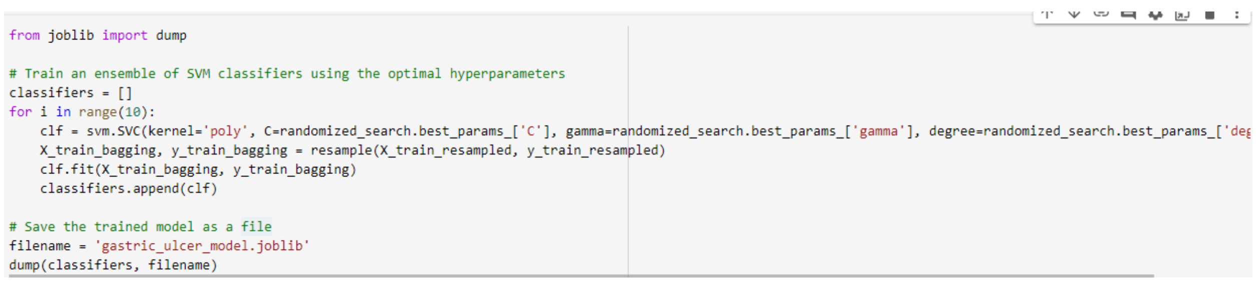
```python
# Visualize the evaluation metrics
plt.figure(figsize=(8, 5))
plt.bar(['Accuracy', 'Precision', 'Recall', 'F1 score'], [accuracy, precision, recall, f1_score])
plt.ylim([0, 1])
plt.title('Evaluation Metrics')
plt.show()
```

Purpose: Visualizing evaluation metrics.

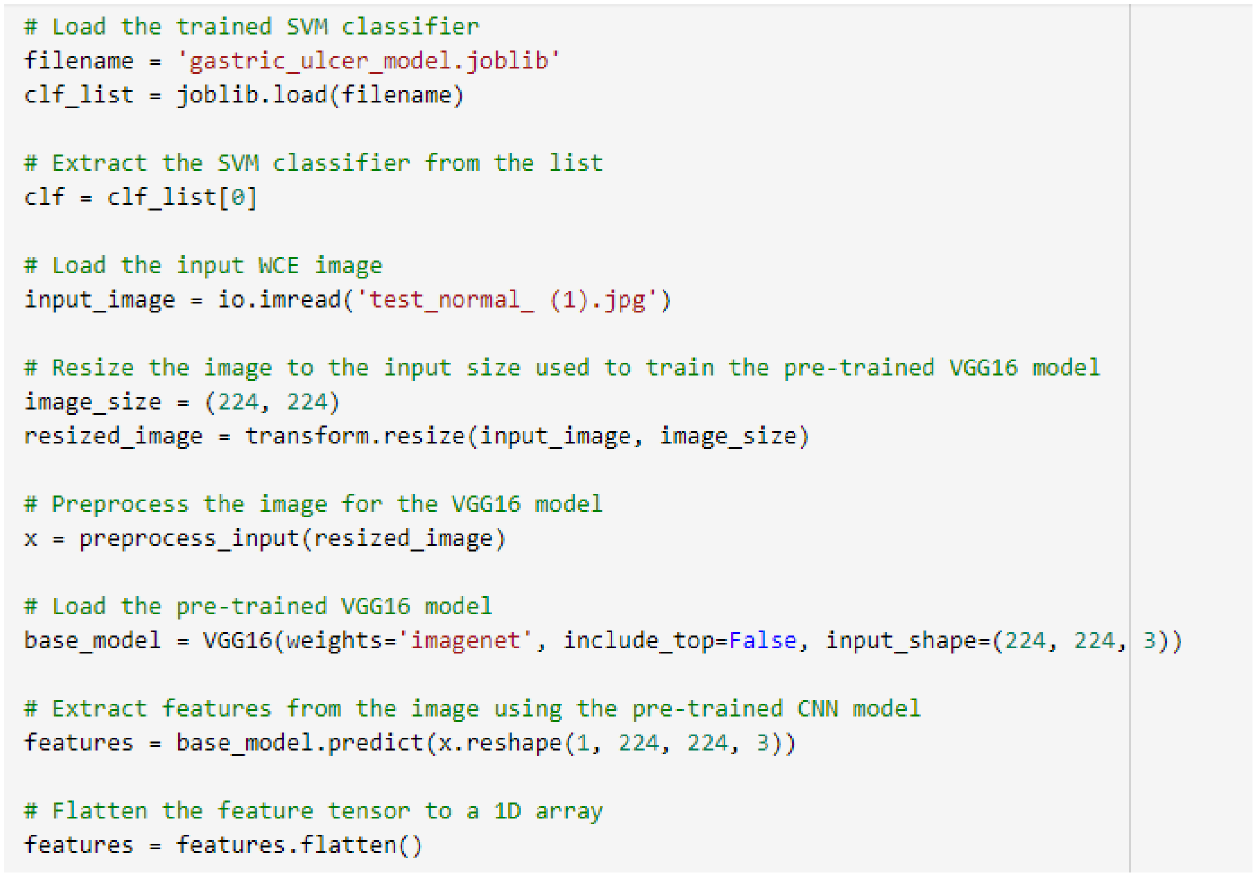


```python
# Make predictions on the test set using the ensemble of classifiers
y_scores = []
for clf in classifiers:
    y_scores.append(clf.decision_function(X_test))
y_scores = np.mean(y_scores, axis=0)

# Calculate evaluation metrics
fpr, tpr, thresholds = metrics.roc_curve(y_test, y_scores)

# Calculate the area under the ROC curve (AUC)
roc_auc = metrics.auc(fpr, tpr)

# Plot the ROC curve
plt.figure(figsize=(5,5))
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```

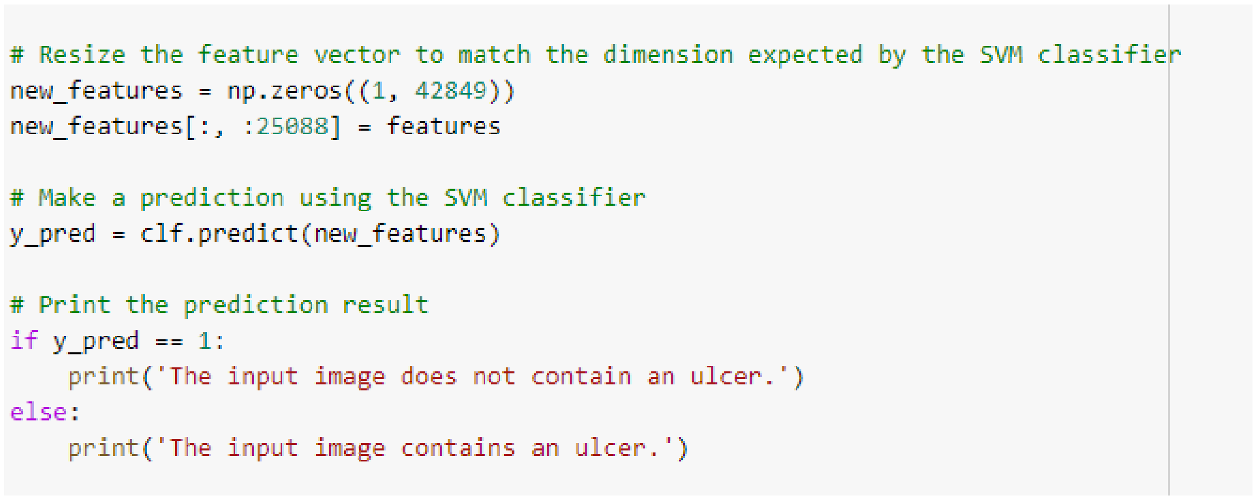
Purpose: Calculating area under the ROC curve and plotting the ROC curve.



```python
from joblib import dump

# Train an ensemble of SVM classifiers using the optimal hyperparameters
classifiers = []
for i in range(10):
    clf = svm.SVC(kernel='poly', C=randomized_search.best_params_['C'], gamma=randomized_search.best_params_['gamma'], degree=randomized_search.best_params_['deg
    X_train_bagging, y_train_bagging = resample(X_train_resampled, y_train_resampled)
    clf.fit(X_train_bagging, y_train_bagging)
    classifiers.append(clf)

# Save the trained model as a file
filename = 'gastric_ulcer_model.joblib'
dump(classifiers, filename)
```



```python
# Load the trained SVM classifier
filename = 'gastric_ulcer_model.joblib'
clf_list = joblib.load(filename)

# Extract the SVM classifier from the list
clf = clf_list[0]

# Load the input WCE image
input_image = io.imread('test_normal_ (1).jpg')

# Resize the image to the input size used to train the pre-trained VGG16 model
image_size = (224, 224)
resized_image = transform.resize(input_image, image_size)

# Preprocess the image for the VGG16 model
x = preprocess_input(resized_image)

# Load the pre-trained VGG16 model
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Extract features from the image using the pre-trained CNN model
features = base_model.predict(x.reshape(1, 224, 224, 3))

# Flatten the feature tensor to a 1D array
features = features.flatten()
```



```python
# Resize the feature vector to match the dimension expected by the SVM classifier
new_features = np.zeros((1, 42849))
new_features[:, :25088] = features

# Make a prediction using the SVM classifier
y_pred = clf.predict(new_features)

# Print the prediction result
if y_pred == 1:
    print('The input image does not contain an ulcer.')
else:
    print('The input image contains an ulcer.')
```

Purpose: Saving the model as a file and testing if the model can detect gastric ulcer from WCE image correctly. In this study using SVM classifier the gastric ulcer detection from WCE image was successful.