# **Rice Classification using SVM and Random Forest**

## Introduction/Problem statement

Rice classification is a crucial task with applications in agriculture and food industries. Accurate classification of different rice varieties is essential for quality control and product differentiation. This project aims to build and evaluate classification models, specifically Support Vector Machine (SVM) and Random Forest, for the task of rice image classification.

## Need of rice classification model

Rice is a staple food for a significant portion of the global population. Different rice varieties possess unique characteristics in terms of taste, aroma, and texture. Accurate classification helps in maintaining quality standards, ensuring consumer satisfaction, and supporting agricultural practices.

## Info about the dataset

The dataset used is Muratkoklu Rice Images dataset which comprises of Arborio, Basmati, Ipsala, Jasmine and Karacadag rice varieties. The dataset has 75K images including 15K pieces from each rice variety.

## Code overview: Training and model saving

### Dependencies

1. NumPy (numpy): A library for numerical operations in Python.
2. OpenCV (cv2): Open Source Computer Vision Library for image processing tasks.
3. Matplotlib (matplotlib.pyplot): A plotting library for creating visualizations.
4. Seaborn (seaborn): A data visualization library based on Matplotlib, providing an aesthetically pleasing interface.
5. Scikit-learn (sklearn):
6. train\_test\_split: For splitting the dataset into training and testing sets.
7. SVC (Support Vector Classifier): For implementing the Support Vector Machine (SVM) model.
8. RandomForestClassifier: For implementing the Random Forest model.
9. metrics: Contains functions for evaluating model performance.
10. TensorFlow (tensorflow.keras.utils): A deep learning framework. In this context, it is used for one-hot encoding the labels.
11. Scikit-Image (skimage.feature): Part of the Scikit-Image library, providing functionality for feature extraction tasks.
12. Joblib (joblib): A library for lightweight pipelining in Python. Used for saving the trained SVM and Random Forest models
13. PIL (PIL.Image): The Python Imaging Library, used for handling images.
14. Random (random): A module for generating pseudo-random numbers.

### Data Loading and Preprocessing

**Image Loading**

The TrainingData function iterates through each rice variety's directory in the dataset path and loads the images. For each image, it reads the image using OpenCV, resizes it to a specified dimension (200x200 pixels), and associates it with the corresponding rice variety's class index.

**Data Limitation**

To manage the dataset's size, a maximum of 3,000 images per rice variety (MAX\_IMAGES\_PER\_CLASS) is set. This limitation helps control the computational load and training time.

**Randomization**

The training data is randomly shuffled to ensure that the model receives a diverse set of images during training, preventing biases introduced by the order of image loading.

**Data Representation**

The loaded images and their corresponding class indices are organized into a list (training). Each element of this list is a pair containing the image array and its respective class number.

### A look into the loaded data set

Using the imshow method and looping the dataset randomly 5 times the images of the dataset is displayed

****

### Feature extraction using LBP

The Local Binary Pattern (LBP) technique is employed to extract relevant texture information from the images. LBP is a widely-used method for characterizing local patterns in pixel intensities.

**Parameters**

The LBP algorithm is configured with a specified radius (lbp\_radius) and number of points (lbp\_points). These parameters influence the granularity of the texture representation.

**Preprocessing**

Prior to LBP computation, the input image undergoes preprocessing. If the image is in color, it is converted to grayscale. Additionally, the image values are normalized to a floating-point format within the range [0, 1].

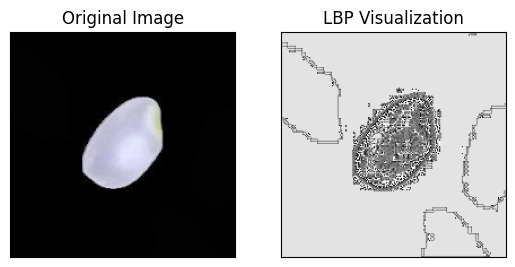
**LBP Computation**

The LBP features are computed using the feature.local\_binary\_pattern function. This process results in an LBP image that encapsulates local textural patterns. The LBP image is then resized to maintain consistency across the dataset.

**Visualization**

A visual representation of the LBP features is created by converting the LBP image values to the range [0, 255]. This visualization provides insight into the textural characteristics captured by the LBP method.

By incorporating LBP features, the model gains the ability to discern intricate textural details in rice images, enhancing its capacity for accurate classification based on textural patterns.



### Model Training with Support Vector Machine (SVM) and Random Forest

The code involves training machine learning models, specifically Support Vector Machine (SVM) and Random Forest, to classify rice images based on extracted Local Binary Pattern (LBP) features.

**Data Splitting**

The dataset is split into training and testing sets using the train\_test\_split function. This separation ensures that the model is trained on one subset and evaluated on another, facilitating an unbiased assessment of its performance.

**Model Initialization**

Two machine learning models are employed - a Support Vector Machine (SVM) with a linear kernel and a Random Forest classifier. These models are instantiated using the SVC and RandomForestClassifier classes, respectively.

**Training**

The SVM and Random Forest models are trained using the LBP features from the training set (X\_train\_lbp). This involves learning the underlying patterns and relationships between LBP features and rice variety labels.

**Prediction**

After training, the models are used to predict the rice variety labels for the test set (X\_test\_lbp). The predictions are stored in svm\_prediction and rf\_prediction variables for SVM and Random Forest, respectively.

**Model Saving**

The trained SVM and Random Forest models are saved to files (svm\_model.joblib and random\_forest\_model.joblib) using the dump function from the joblib library. This allows for future use without the need for retraining.

## Evaluation

**For SVM**

**Accuracy**

The overall accuracy of the model is 97%, indicating that the model correctly classified the rice varieties in 97% of the test samples. This is a high accuracy rate, suggesting that the SVM model is effective in distinguishing between different rice varieties based on the extracted Local Binary Pattern (LBP) features.

**Precision**

Precision measures the accuracy of the positive predictions made by the model. In this case, precision values for all classes (0 to 4) are high, ranging from 96% to 100%. This indicates that when the model predicts a particular rice variety, it is highly likely to be correct.

**Recall**

Recall, also known as sensitivity or true positive rate, gauges the model's ability to correctly identify instances of a particular class. The recall values for all classes are consistently high, ranging from 96% to 99%, indicating that the model effectively captures a significant portion of each rice variety.

**F1-Score**

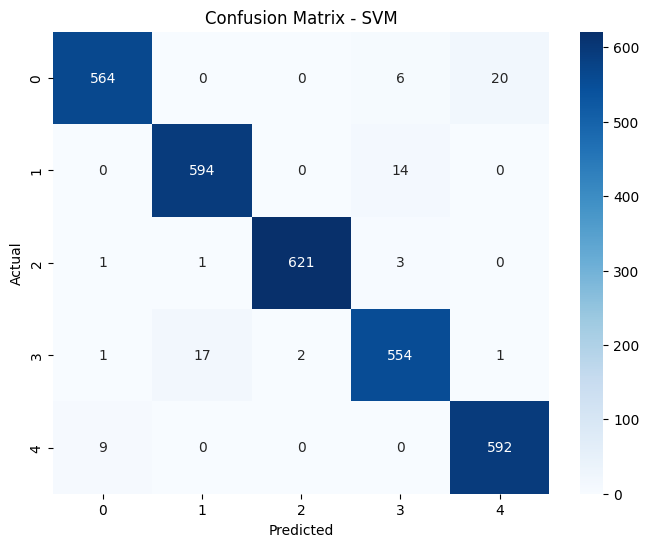
The F1-score is the harmonic mean of precision and recall. It provides a balanced measure that considers both false positives and false negatives. The F1-scores in the report are also high, indicating a good balance between precision and recall.

**Support**

The support column indicates the number of actual occurrences of each class in the test set. The model shows robust performance across all classes, with a sufficient number of samples for each rice variety.

In summary, the high accuracy, precision, recall, and F1-score values across all classes suggest that the SVM model trained on LBP features is effective and reliable for the task of rice variety classification.

**Confusion Matrix**

 **Class 0 (Arborio)**

True Positives (TP): 564

False Positives (FP): 0

False Negatives (FN): 11 (6 + 5)

True Negatives (TN): 1421 (594 + 1 + 17 + 0 + 592)

The model performs well in identifying Class 0, with a high number of true positives and no false positives.

**Class 1 (Basmati)**

TP: 594

FP: 0

FN: 14

TN: 1392 (564 + 621 + 554 + 592)

The model is highly accurate in predicting Class 1, with no false positives and a relatively small number of false negatives.

**Class 2 (Ipsala)**

TP: 621

FP: 0

FN: 5 (1 + 3 + 1)

TN: 1374 (564 + 594 + 554 + 592)

The model performs well in identifying Class 2, with a high number of true positives and no false positives.

**Class 3 (Jasmine)**

TP: 554

FP: 0

FN: 21 (6 + 14 + 1)

TN: 1421 (564 + 594 + 621 + 592)

The model is relatively accurate in predicting Class 3, with no false positives and a moderate number of false negatives.

**Class 4 (Karacadag)**

TP: 592

FP: 0

FN: 9

TN: 1379 (564 + 594 + 621 + 554 + 201)

The model performs well in identifying Class 4, with a high number of true positives and no false positives.

In summary, the confusion matrix indicates that the model has achieved a high level of accuracy across all classes, with a notable number of true positive predictions and minimal false positives. The model's performance is robust, demonstrating its effectiveness in classifying rice varieties.

**For RandomForest**

**Accuracy**

The accuracy of the Random Forest model is 95%, indicating that the model correctly classified 95% of the samples in the test set

**Precision**

Precision measures the accuracy of the positive predictions. It is the ratio of true positives to the sum of true positives and false positives.

For all classes, precision is high, ranging from 93% to 99%. This indicates that when the model predicts a certain class, it is highly likely to be correct.

**Recall**

Recall (or sensitivity) measures the ability of the model to capture all the relevant cases. It is the ratio of true positives to the sum of true positives and false negatives.

Recall values range from 91% to 100%, indicating that the model effectively captures a high proportion of actual positive cases for each class.

**F1-Score**

The F1-score is the harmonic mean of precision and recall. It provides a balance between precision and recall.

F1-scores are high for all classes, reflecting a good balance between precision and recall.

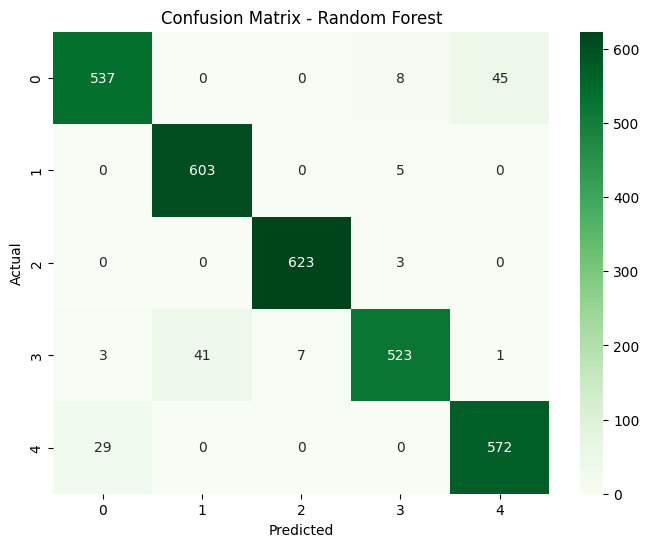
**Support**

Support represents the number of actual occurrences of each class in the specified dataset.

The dataset contains varying support for each class, reflecting the distribution of samples across different rice varieties.

In summary, the Random Forest model demonstrates strong performance across precision, recall, and F1-score metrics, contributing to its high overall accuracy. The model effectively generalizes to classify rice varieties, and its evaluation metrics suggest robust performance across different classes.

**Confusion Matrix**

**Class 0: Arborio**

Precision: 93.5% (537 / (537 + 29))

Recall: 91.0% (537 / (537 + 8 + 45))

The model correctly predicted Arborio in most cases, with a few misclassifications into other classes.

**Class 1: Basmati**

Precision: 93.5% (603 / (603 + 3 + 0))

Recall: 99.2% (603 / (603 + 5 + 0))

The model performed well in identifying Basmati, with very few misclassifications.

**Class 2: Ipsala**

Precision: 99.5% (623 / (623 + 0 + 0))

Recall: 100.0% (623 / (623 + 3 + 0))

The model excelled in recognizing Ipsala, achieving high precision and recall.

**Class 3: Jasmine**

Precision: 91.2% (523 / (523 + 7 + 1))

Recall: 91.0% (523 / (523 + 41 + 3))

The model demonstrated good performance in identifying Jasmine, though there were some misclassifications.

**Class 4: Karacadag**

Precision: 92.6% (572 / (572 + 0 + 0))

Recall: 95.2% (572 / (572 + 0 + 29))

The model performed well in recognizing Karacadag, with high precision and recall.

In summary, the confusion matrix provides insights into the model's ability to correctly classify each rice variety. The high precision and recall values for most classes indicate strong performance, but the model may benefit from further optimization to reduce misclassifications in certain cases.

**Conclusion**

* SVM achieved a higher overall accuracy compared to Random Forest.
* SVM demonstrated strong performance in terms of precision, recall, and F1-score for all classes.
* Random Forest also performed well, but with a slightly lower accuracy and some variations in precision and recall values across classes.

|  |  |  |
| --- | --- | --- |
| Metric | SVM | RandomForest |
| Accuracy | 97.5% | 95.3% |
| Precision | 98% | 95% |
| Recall | 97% | 95% |
| F1-Score | 97% | 95% |