Food Analytical Methods (2023) 16:721–748
https://doi.org/10.1007/s12161-023-02445-0

Artificial Intelligence Aided Adulteration Detection and Quantification for Red Chilli Powder

Tanmay Sarkar[1] · Tanupriya Choudhury[2] · Nikunj Bansal[2] · V. R. Arunachalaeshwaran[2] · Mars Khayrullin[3] · Mohammad Ali Shariati[3,4] · Jose Manuel Lorenzo[5,6]

Received: 19 December 2022 / Accepted: 10 January 2023 / Published online: 27 January 2023

Abstract

Food adulteration imposes a significant health concern on the community. Being one of the key ingredients used for spicing up food dishes. Red chilli powder is almost used in every household in the world. It is also common to find chilli powder adulterated with brick powder in global markets. We are amongst the first research attempts to train a machine learning-based algorithms to detect the adulteration in red chilli powder. We introduce our dataset, which contains high quality images of red chilli powder adulterated with red brick powder at different proportions. It contains 12 classes consists of 0%, 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, and 100% adulterant. We applied various image color space filters (RGB, HSV, Lab, and YCbCr). Also, extracted features using mean and histogram feature extraction techniques. We report the comparison of various classification and regression models to classify the adulteration class and to predict the percentage of adulteration in an image, respectively. We found that for classification, the Cat Boost classifier with HSV color space histogram features and for regression, the Extra Tree regressor with Lab color space histogram features have shown the best performance.

Keywords Food fraud · Machine learning · Computer vision · Image analysis · Food authentication

✉ Tanmay Sarkar
tanmays468@gmail.com

✉ Jose Manuel Lorenzo
jmlorenzo@ceteca.net

Tanupriya Choudhury
tanupriya1986@gmail.com

Nikunj Bansal
nikunj8126@gmail.com

V. R. Arunachalaeshwaran
avr5309@gmail.com

Mars Khayrullin
89049755219@ya.ru

Mohammad Ali Shariati
shariatymohammadali@gmail.com

[1] Department of Food Processing Technology, Malda Polytechnic, West Bengal State Council of Technical Education, Govt. of West Bengal, Malda 732102, India

[2] Informatics Cluster, School of Computer Science, University of Petroleum and Energy Studies (UPES), Dehradun 248007, Uttarakhand, India

[3] Department of Scientific Research, K.G. Razumovsky Moscow State University of Technologies and management (The First Cossack University), 73 Zemlyanoy Val, Moscow 109004, Russian Federation

[4] Department of Scientific Research, Russian State Agrarian University - Moscow Timiryazev Agricultural Academy, Moscow 127550, Russian Federation

[5] Centro Tecnológico de La Carne de Galicia, Avda. Galicia nº 4, Parque Tecnológico de Galicia, San Cibrao das Viñas, 32900 Ourense, Spain

[6] Área de Tecnología de los Alimentos, Facultad de Ciencias de Ourense, Universidad de Vigo, 32004 Ourense, Spain

Introduction

Food adulteration is an illegal practice of adding cheaper ingredients or fake substances to pure food products to increase their quantity, making them unhygienic to eat. It poses a significant health concern, and when consumed, it deprives necessary nutrition in the human body (Manasha and Janani 2016). Adulterated foods can cause various health issues ranging from mild to life-threatening, including allergic reactions, skin diseases, loss of vision, cancer, and heart diseases (Bansal et al. 2017). Sometimes, it shows immediate side effects in the human body, including vomiting, abdominal pain, and diarrhoea.

Individuals in the food supply chain perform adulteration intentionally to maximize profits. They reduce the food quality to increase its quantity. They put consumers' health in danger for personal economic gains. Adulterants are also used to expand the shelf life. Improper processing of these food items also leads to abdominal pain and food poisoning or other food infections, usually with fever (Nascimento et al. 2017).

Food adulteration also results in economic losses to the countries as it leads to a decrement in consumer demand. It impacts consumer confidence. Food quality assurance efforts are significant to make consumers' trust intact. Food adulteration at the production level usually happens due to a lack of monitoring and testing measures (Bansal et al. 2017).

Red chilli powder is one of the key ingredients used for spicing up Indian cooking. It is rich in vitamins and minerals. It also comes in several varieties like Kashmiri Chilli powder, Guntur Chilli, ghost pepper, bird-eye Chilli, Byadagi Daggi Chilli, etc. (Jamaluddin et al. 2022). The variety of the Chilli powder depends on the type of Chilli used and the process followed for drying and grinding. As it is typically consumed in small amounts, it does not contribute much towards one's diet. Red chilli powder typically constitutes 88% water, 0.3% protein, 1.3% carbohydrates, 0.8% sugar, 0.2% fiber, and 0.1% fat and has a Caloric value of 6 Cal. It contains vitamins like vitamin C, vitamin B_6, vitamin K_1, and vitamin A (Khan et al. 2019). It also contains minerals like potassium and copper. Consumption of red chilli powder has many benefits, such as providing pain relief, promoting weight loss, fighting inflammation, clearing congestion, boosting immunity, preventing stomach ulcers, boosting immunity, etc. (Ayob et al. 2021; Mi et al. 2022). It is not uncommon to find Chilli powder adulterated with brick powder in Indian markets. Typically, people identify this kind of adulteration using visual cues like adulterated Chilli powder looks more orangish instead of the rich color of unadulterated Chilli powder.

Artificial Intelligence enables computers to do tasks that earlier needs human interference. Computer vision is a field of artificial intelligence that allows computers to see, understand, and recognize digital content by processing images and videos (Ma et al. 2016). Adulteration in food is detectable by training algorithms on the labelled dataset. Therefore, the need to create a genuine dataset in presence of a domain expert for mapping food images with various percentages/ratios of adulteration or purity is a must. Image Augmentation techniques are used to make the image classification-based computer vision model more robust and image pre-processing techniques to improve image quality (Subashini 2010). Various filters including grayscale (Vincent 1993), RGB (Li et al. 2016), HSV (Sadhukhan et al. 2019), median smoothening (Vijaykumar et al. 2010), adaptive thresholding (Bao and Zhang 2003), Canny edge detection (Sarkar et al. 2022), Sobel edge detection (Gao et al. 2010; Hussein et al. 2011), and YCrCb (Lakhwani et al. 2015) are used to enhance features in the images. Feature extraction techniques are also used to extract features from the training dataset consisting of images.

Machine learning algorithms enable computers to make informative decisions based on raw data. Deep learning (subset of machine learning) algorithms are able to learn complex features from unstructured data as they mimic the way the human brain works. Researchers have used neural networks containing one or more hidden layers consisting of neurons to extract and learn complex information. Deep learning (DL)-based image recognition methods are spread across various fields. DL is being used to recognize food objects in an image and to recognize possible food allergens in an image (Salim et al. 2021). DL also provide food recommendations to the user based on their past choices (content-based filtering for food recommendation) (Bianchini et al. 2016). DL is also being used in detecting food adulteration by mixing cheap ingredients in order to maximize the quantity (Goyal et al. 2022).

Machine learning is essential to detect food adulteration in an image captured using mobile phone or any other device (Goyal et al. 2022). This will enable the detection of food adulteration even if it is not distinguishable by a human being. Deep learning-based food adulteration detection models are able to be deployed at a large scale as in this digital age major chunk of the population is having smartphones and access to an internet connection (Rateni et al. 2017). Deep learning models efficiently and precisely can detect food adulterants and their percentage using an image. Recent advances in adulteration detection using DL models for various foods ensures that this methodology is well tested and effective (Calle et al. 2022b).

The aim of this study is to detect the adulteration of red chilli powder; adulterated with red brick dust/powder. We have considered various machine learning algorithms for detecting the percentage of adulteration in red chilli powder.

Literature Review

Machine learning (ML) algorithms outperformed the earlier rule-based classical algorithms. ML enables systems to define rules on their own by recognizing hidden

patterns in data. In order to make ML model perform precisely and accurately, there is a need to provide the best suitable and as many features during training the model. Image processing is generally used to improve image quality and extract useful features from an image. Various filters including grayscale, RGB, HSV, median smoothening, adaptive thresholding, canny edge detection, Sobel edge detection, and YCrCb are used to enhance features in the images. Feature extraction using various methods including neural networks is also used to extract various features from the images.

Researchers have trained the Linear Discriminant Analysis (LDA) Classifier on 1080 greyscale images for detecting the quality of wheat seeds into 9 categories with an accuracy of 98.15% by extracting the features using various feature extraction matrices including local similarity numbers (LSN), local similarity patterns (LSP), gray level run length matrix (GLRM), gray level cooccurrence matrix (GLCM), and local binary patterns (LBP) (Pourreza et al. 2012). Various image processing techniques have been applied to differentiate the quality of rice by using various filters including canny edge detection, adaptive thresholding, median smoothing, and grayscale (Pratibha et al. 2017) (Table 1). Researchers have designed the automated mango grading system. They used the fuzzy rule-based algorithm by processing video images captured using a CCD camera including background elimination and contour detection using the Graph Contour tracking system. They have predicted the maturity using RFE-SVM and gradation with multi attributes decision method (MADM). They reported an average grading precision rate of 90% (Nandi 2014). Scholars have applied image processing techniques via performing various morphological operations including dilation, erosion, and intensity of border to classify oranges based on maturity level (Carolina and David 2014).

Researchers have trained PLS-DA and LDA models to detect the adulteration of mined beef with pork and vice-versa using the dataset consisting of 18 different wavelengths of 220 meat samples spread across nine adulteration classes (Ropodi et al. 2015). They reported accuracy of 98.48%. Researchers have trained three variants of Partial Least Squares Regression (PLSR) including R-PLSR, A-PLSR, and KM-PLSR with the data retrieved by visible near infrared hyperspectral imaging. They reported an R^2 score of 0.96 and RMSEP of 2.83% for R-PLSR; R^2 score of 0.97 and RMSEP of 2.61% for A-PLSR; and R^2 score of 0.96 and RMSEP of 3.05% For KM-PLSR (Kamruzzaman et al. 2016).

ANN classifier has been used to classify healthy (65% images) and unhealthy nuts (35% images) based on two different sets of features. First consists of 22 original features including 6 texture properties and 16 features extracted using gray level co-occurrence matrix (GLCM) at angles of 0, 45, 90, and 135. The second consists of a feature set obtained from the principal component analysis (PCA) algorithm. They split the dataset into 70:15:15 for training, validation and testing respectively. They reported an accuracy of 81.8% using the first set of features and 100% using the second set of features on the testing dataset (Khosa and Pasero 2014).

Decision tree, random forest, SVM, K-nearest neighbor (KNN), and ANN classifiers have been used to detect white rice adulteration using the dataset containing 330 samples spread across 7 different ratios. They reported that SVM and random forest outperformed other classifiers and are more robust in distinguishing white rice and adulterated mixtures (Lim et al. 2017). Scholars have proposed the model based on ANN, PLSR, and PCR algorithms for recognizing milk adulteration using the dataset containing milk samples mixed with bromothymol blue. They used RGB values and luminosity as features (Kobek 2017). MLP classifier has been considered to classify three varieties of rice, the dataset contains 222 images of each variety of rice, a total of 666 images. They used PCA for feature ranking and extracted 41 textural features and 17 morphological features from the images. They obtained the classification accuracy of 55.93 to 84.62% on the test dataset (Fayyazi et al. 2017).

Researchers have applied a window local segmentation algorithm to detect the surface defects in oranges on the dataset comprised of 1191 grey level images of oranges (Rong et al. 2017). Anami et al. 2019 trained BPNN, SVM, and k-NN classifiers to detect paddy adulteration into 7 different varieties and then classify it into 5 different adulteration levels (10%, 15%, 20%, 25%, and 30%). They extracted color features, GLCM features, and LBP features from RGB images. They also applied PCA and sequential forward floating selection (SFFS) algorithm for feature selection. They reported an accuracy of 41.31% using the BPNN classifier trained on extracted color features, the accuracy of 44.74% when trained on extracted GLCM features, and accuracy of 39.03% when trained on extracted LBP features. They also reported an accuracy of 35.80% using SVM classifier trained on extracted color features, accuracy of 37.00% when trained on extracted GLCM features, and accuracy of 35.00% when trained on extracted LBP features. They also reported an accuracy of 36.40% using k-NN classifier trained on extracted color features, accuracy of 34.40% when trained on extracted GLCM features, and accuracy of 34.71% when trained on extracted LBP features. They found significant improvement in performance after applying feature selection algorithms and using combined color-texture features. On Selected features using SFFS algorithm, they reported an accuracy of 91% using BPNN classifier, 86.91% using SVM classifier, and 82.14% using k-NN classifier. On Selected features using PCA algorithm, they reported an accuracy of 93.31% using BPNN classifier, 89.29% using SVM classifier, and 83.66% using k-NN classifier (Anami et al. 2019).

Table 1 Image analysis combined with machine learning for food quality determination

Models	Methodology/dataset preparation	Result	Reference
Linear discriminant analysis (LDA) classifier	Used 1080 greyscale images spread across 9 classes of quality of wheat seeds. Features extracted using local similarity numbers (LSN), local similarity patterns (LSP), gray level run length matrix (GLRM), gray level cooccurrence matrix (GLCM), and local binary patterns (LBP)	Accuracy of 98.15%	(Pourreza et al. 2012)
RFE-SVM and multi attributes decision method (MADM)	Used video images captured using CCD camera. Performed background elimination and contour detection using graph contour tracking system	Accuracy > 90%	(Nandi 2014)
R-PLSR	Used visible near infrared hyperspectral images	R^2 score of 0.96 and RMSEP of 2.83%	(Kamruzzaman et al. 2016)
A-PLSR		R^2 score of 0.97 and RMSEP of 2.61%	
KM-PLSR		R^2 score of 0.96 and RMSEP of 3.05%	
ANN	X-ray images of healthy (65% images) and unhealthy nuts (35% images). 22 original features including 6 texture properties, and 16 features extracted using gray level co-occurrence matrix (GLCM) at angles of 0, 45, 90, and 135	Accuracy of 81.8%	(Khosa and Pasero 2014)
	X-ray images healthy (65% images) and unhealthy nuts (35% images). Features set consist of 3 features obtained from principal component analysis (PCA)	Accuracy of 100%	
Decision tree, random forest, SVM, K-nearest neighbor (KNN), and ANN	330 samples spread across 7 different ratios of white rice adulteration	Random forest and SVM classifiers are more robust	(Lim et al. 2017)
MLP classifier (three-layer feed forward neural network)	222 images of 3 variety of rice, a total of 666 images. Then applied PCA for feature ranking and extracted 41 textural features and 17 morphological features from the images	Accuracy 55.93%, – 84.62%	(Fayyazi et al. 2017)
BPNN	7 different paddy varieties and then classify it into 5 different adulteration levels (10%, 15%, 20%, 25%, and 30%). Extracted color features, GLCM features, and LBP features from RGB images	91.00%	(Anami et al. 2019)
		86.91%	
	SFFS algorithm for feature selection	82.14%	
	PCA algorithm for feature selection	93.31%	
		89.29%	
		83.66%	
ANN	Sesame oil (SEO) adulterated with two types of Rapeseed oil (RO). Samples have prepared with the 30, 40, 50, 60, 65, 70, 75, 80, 85, 90, 95, and 100 (mL/100 mL) SEO	MAE is 6.4033, and RMSE is 7.8072	(Soltani Firouz et al. 2021)
Polynomial SVR		MAE is 8.7776, and RMSE is 10.9904	
RBF SVR		MAE is 4.988,8 and RMSE is 6.0129	
LDA	The dataset comprised of 100% squeezed apple, pineapple, and orange juices adulterated with grape juice. It consists of 7 adulteration classes, e.g., 5%, 10%, 15%, 20%, 30%, 40%, and 50%	97.67%	(Calle et al. 2022a)
SVM		88.37%	
RF		97.67%	
CNN	The used dataset comprised of 195 sample (mutton) images	They found that accuracy was increased by 7.33%, and RMSE decreased by 0.01 when detecting adulteration in entirely heated mutton	(Zhang et al. 2022)
ANN	Detection of fructose and glucose in honey samples. The dataset comprised images of 4 adulteration classes 10%, 20%, 30%, and 50%	95%	(Shafiee et al. 2016)
SVM		92%	
LDC		90%	
Fisher		89%	
Parzen		84%	

Table 1 (continued)

Models	Methodology/dataset preparation	Result	Reference
KNN	Detection of sugar syrup in honey samples. The dataset consists of 8675 rows representing spectral instances of honey samples spread across four adulteration classes 5%, 10%, 25%, and 50%	91.16%	(Al-Awadhi and Deshmukh 2022)
Linear SVM	Original features considered	90.99%	
RBF SVM		85.79%	
KNN	Detection of sugar syrup in honey samples. The dataset consists of 8675 rows representing spectral instances of honey samples spread across four adulteration classes 5%, 10%, 25%, and 50%	89.57%	
Linear SVM	PCA features considered	89.06%	
RBF SVM		91.03%	
KNN	Detection of sugar syrup in honey samples. The dataset consists of 8675 rows representing spectral instances of honey samples spread across four adulteration classes 5%, 10%, 25%, and 50%	96.39%	
Linear SVM	LDA features considered	96.36%	
RBF SVM		96.29%	
Logistic Regression –discriminant analysis (LRDA)	Detection of honey adulteration with syrup. The dataset consist of 1525 rows (253 spectra from pure honey samples and 1272 spectra from honey adulteration classes 2%, 4%, 8%, 16% and 32%). The dataset has been splitted into two portions: 66.66% and 33.33% for training and validation respectively	84.3%	(Boateng et al. 2022)
Artificial Neural Network –discriminant analysis (ANNDA)		95.5%	
Partial least squares–discriminant analysis (PLSDA)		80.2%	
Support vector machines – discriminant analysis (SVMDA)		98.1%	
Gradient boosting –discriminant analysis (GBDA)		98.8%	
CNN	Detection of tartrazine-colored rice flour adulteration in turmeric powder. The dataset consists of spectral images of turmeric powder mixed with tartrazine-colored rice flour adulterant in 4 weight percentages which are 0%, 5%, 10%, and 15%	94.35%	(Lanjewar et al. 2022)
SqueezeNet with Adam optimizer	Detection of common coffee green beans adulteration in Luwak Coffee. The dataset consists of 4 classes based on purity: very low (0–25%), low (25–50%), medium (50–75%), and high (75–100%)	80.47%	(Hendrawan et al. 2022)
GoogleNet with Adam optimizer		89.65%	
ResNet50 with Adam optimizer		86.52%	
AlexNet with Adam optimizer		69.53%	
LDA,KNN, RF, SVM	Detection of the exogenous protein in milk powder. The dataset has been prepared by mixing four types of exogenous proteins with milk powder	average accuracy of 93.9% has been obtained using SVM	(Huang et al. 2022b)
CNN		Average Accuracy of 97.8%	
SVM	The special green coffee has been distinguished from traditional green coffee using multispectral imaging	Accuracy of 96% and AUC of 0.97	(Pinheiro Claro Gomes et al. 2022)
XGBoost		Accuracy of 86% and AUC of 0.84	
Random forest		Accuracy of 88% and AUC of 0.80	
CatBoost		Accuracy of 86% and AUC of 0.84	
Decision tree	AI-based spectroscopic sensor system to detect milk adulterants in real time. The dataset consists of 4 adulterant classes, namely dextrose, hydrogen peroxide, ammonium sulfate, sodium salicylate, and pure milk class	91.7%	(Sowmya and Ponnusamy 2021)
Naïve Bayes		90%	
Linear discriminant analysis (LDA)		88.1%	
SVM		90%	
Neural network without hyperparameter		92.7%	
Neural network with hyperparameter		100%	

Table 1 (continued)

Models	Methodology/dataset preparation	Result	Reference
Decision tree (DT)	The seven pine nuts have been categorised (Pinus bungeana, Pinus yunnanensis, Pinus thunbergii, Pinus armandii, Pinus massoniana, Pinus elliottii andnt Pinus taiwanensis) through near-infrared (NIR) spectroscopy	87%	(Huang et al. 2022a)
Random forest (RF)		89%	
Multilayer perceptron (MLP)		99%	
Support vector machine (SVM)		94%	
Naive Bayes (NB)		80%	
VGG16		90.5%	
VGG19		82.6%	
Xception		95.7%	
InceptionV3		48.5%	
ResNet50		96.4%	
k-NN	Volatile organic compound sensor to detect the presence of formaldehyde in different fruits	95%	(Brighty et al. 2021)
Siamese network	Black peppercorns adulterated with papaya seed. The dataset consists 2000 images of each class	92%	(Fatima et al. 2021)
MLP	Chickpea powder adulterated ginger powder. The dataset consists of 3360 images spread across seven classes (pure ginger powder, pure chickpea powder, adulterant 10%, adulterant 20%, adulterant 30%, adulterant 40%, and adulterant 50%). The HOG image feature extraction method have been considered	87.65%	(Jahanbakhshi et al. 2021a)
Fuzzy		61.46%	
k-NN		96.13%	
SVM		78.42%	
GBT		87.35%	
EDT		90.33%	
KNN	Detection of adulteration in coconut milk. The dataset consists of FTIR spectral dataset for 42 coconut milk samples in which 14 were pure samples, 14 were adulterated with water at 10% concentration, and 14 were adulterated with water at 20% concentration Original features have been considered	44.44%	(Al-Awadhi and Deshmukh 2021)
Linear SVM		37.78%	
RBF SVM		32.22%	
KNN	PCA features have been considered	26.67%	
Linear SVM		37.78%	
RBF SVM		40%	
KNN	They used LDA features have been considered	93.33%	
Linear SVM		86.67%	
RBF SVM		91.11%	
PLS (number of factors = 2)	Detection of ripened papaya adulterant in chilli sauce. The dataset consists of 9 adulterated classes (10%w/w, 20%w/w, 30%w/w, 40%w/w, 50%w/w, 60%w/w, 70%w/w, 80%w/w, and 90%w/w)	R^2 score of 0.99 and RMSEP of 1.71% w/w	(Lapcharoensuk et al. 2020)
SVM		R^2 score of 0.99 and RMSEP of 2.18% w/w	
BPNN (full spectrum)		R^2 score of 0.80 and RMSEP of 12.01% w/w	
BPNN (PCs)		R^2 score of 0.99 and RMSEP of 3.27% w/w	

Table 1 (continued)

Models	Methodology/dataset preparation	Result	Reference
ResNet	Wheat flour detection on chickpea flour (gluten-free product). The dataset have been prepared by adding low amounts of wheat flour in chickenpea flour. The dataset consists of 14 classes in which 12 are adulterated samples (1, 2, 3, 4, 5, 7.5, 10, 15, 20, 30, 40, and 50 ppm), along with the pure wheat flour and pure chicken pea samples	Overall accuracy was greater than 93%	(Pradana-López et al. 2022)
Support vector regression (AO-SO)	Avocado Oils (AO) adulteration with cheap oils (soybean oil (SO), corn oil (CO), or rapeseed oil (RO))	R^2 score of 0.974 and RMSEP of 0.053	(Jin et al. 2022)
Support vector regression (AO-CO)	LF-NMR parameters for both adulterated and pure AO have been used as data points	R^2 score of 0.980 and RMSEP of 0.048	
Support vector regression (AO-RO)		R^2 score of 0.892 and RMSEP of 0.109	
Support vector regression (AO-SO/CO/RO)		R^2 score of 0.907 and RMSEP of 0.089	
Partial least squares regression (PLSR)	Palm oil adulterated Sunflower oil Attenuated total reflection (ATR) derived through mid-infrared spectroscopy have been used as input feature	R^2 score of 0.98	(Srinath et al. 2022)

Materials and Methods

Sample preparation

Full grown red chilli of variety Bullet Lanka-5 (*Capsicum annuum* L.) were collected from Kaliachak, Malda (24°86'N 88°01' E). The samples were rinsed with double distilled water, followed by Sun drying (37 ± 3 °C, relative humidity 75–81%, and for consecutive 3 days from 9.00 am to 5.00 pm). The final moisture content of the dried samples were 4–5% (dry weight basis) and ash content was 4.5%. The dried samples were grinded with mixer grinder (500 W, grinding jar capacity 0.5 L, hybrid motor) for 9 min (3 min per cycle, in total 3 cycles, 18,000 rpm). The final product was passed through a screen of 60 mesh, and the undersized powdered samples were considered as pure red chilli powder.

Burnt clay bricks (IS: 1077–1992) were procured from local market, and passed through jaw crusher, the crushed brick powder was passed through a screen of 60 mesh, and the undersized powdered materials were considered as adulterant.

The chilli powder sample was mixed thoroughly with the brick powder (5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, and 50%) to prepare a homogeneous mixture of adulterated samples.

Image Acquisition

In total, 12 sets of samples were prepared, 100% pure red chilli powder, 100% brick powder, and 10 sets of adulterated samples (5–50%). A total of 20 equal sized cells were drawn on the white A4 paper sheet an white A4 paper sheet (210 mm × 297 mm, 80 gsm). The samples were spread over the paper in thin layer. Thus, 20 equal sized cells were there loaded with similar samples (Fig. 1). In total, $12 \times 20 = 240$ images were captured.

Each of the cells were captured separately with Realme 8 Pro (Android 11, Realme UI 2.0, Adreno 618, Octa-core, 108 MP, f/1.9, 26 mm (wide), 1/1.52", 0.7 µm, PDAF, 8 MP, f/2.3, 119˚, 16 mm (ultrawide), 1/4.0", 1.12 µm, 2 MP, f/2.4, (macro), 2 MP, f/2.4, (depth)) without flash. The perpendicular distance between each of the cells and the camera lens was 15 cm. The schematic diagram for image acquisition has been shown in Fig. 2.

Color Space Representation

The color space representation chosen to represent images matters, as it can make ignoring or emphasizing certain information easier. For example, in HSV

representation emphasizing color (Hue) is easy and it is also easy to ignore brightness (Value) information. In our study we have explored four different color space representations which are discussed below.

RGB Single RGB pixel consists of intensities of red, green, and blue components of light illuminating that spot in the image. These intensities can take values from 0 to 255. RGB is one of simplest and straight forward ways of representing an image digitally.

HSV Single HSV pixel consists of hue, saturation, and value (brightness) components of light illuminating that spot in the image. This representation is better at modelling the way humans perceive color when compared to RGB. Hue can take values from 0 to 360. Saturation and value can take values from 0 to 100. The formula for conversion from RGB to HSV is given below. The inverse formula can be used for converting the other way around.

$$R^{'} = \frac{R}{255} \tag{1}$$

$$G^{'} = \frac{G}{255} \tag{2}$$

$$B^{'} = \frac{B}{255} \tag{3}$$

$$C_{max} = \max\left(R^{'}, G^{'}, B^{'}\right) \tag{4}$$

$$C_{min} = \min\left(R^{'}, G^{'}, B^{'}\right) \tag{5}$$

$$\Delta = C_{max} - C_{min} \tag{6}$$

Hue calculation:

$$H = \begin{matrix} 0^0 \Delta = 0 \\ 60^0 \times \left(\frac{G^{'} - B^{'}}{\Delta} \mathrm{mod} 6\right), C_{max} = R^{'} \\ 60^0 \times \left(\frac{B^{'} - R^{'}}{\Delta} + 2\right), C_{max} = G^{'} \\ 60^0 \times \left(\frac{R^{'} - G^{'}}{\Delta} + 4\right), C_{max} = B^{'} \end{matrix} \tag{7}$$

Saturation calculation:

$$S = \begin{matrix} 0, C_{max} = 0 \\ \frac{\Delta}{C_{max}}, C_{max} \neq 0 \end{matrix} \tag{8}$$

Value calculation:

$$\mathrm{V} = C_{max} \tag{9}$$

Lab This color space representation represents lightness, and green–red and yellow-blue values of pixels. It is also referred to as L*a*b* color space. It is commonly used for detecting minor variations in color. The L component takes value from 0 to 100, a and b components value from –128 to 128. Conversion from RGB color space (IEC 61,966–2-1:1999) to the CIE Lab colorspace is carried out using the D65 illuminant function and aperture angle of "2" for the observer (van der Walt 2014).

YCrCb This color space represents an image using luma, blue-difference, and red-difference chroma components. This representation is perceptually uniform. The luma component varies from 0 to 1. The blue-difference and red-difference components vary from –0.5 to 0.5. The digital version of the below RGB to YCrCb is used in this study.

$$Y^{'} = 16 + \left(65.481 \times R^{'} + 128.553 \times G^{'} + 24.996 \times B^{'}\right) \tag{10}$$

$$C_B = 128 + \left(-37.797 \times R^{'} - 74.203 \times G^{'} + 112 \times B^{'}\right) \tag{11}$$

$$C_R = 128 + \left(112 \times R^{'} - 93.786 \times G^{'} - 18.214 \times B^{'}\right) \tag{12}$$

Feature Extraction

Instead of using images directly we use feature extraction methods for speeding up training and inferencing, ease of model comprehension, improving stability of the model.

Color Histogram This feature extraction method is based on making histogram for each individual channel and concatenating the bin-count values of every channel. 256 was the number of bins picked for RGB, Lab, YCrCb color space images. 360 was the number of bins picked for HSV images. This representation only has color information and it ignores spatial information. This is suitable for our application as we want to be sensitive to the color of the Chilli powder while being insensitive to its geometric distribution.

Channel Mean Value This feature extraction method simply utilizes the mean channel intensities of the image. This is primarily used for providing comparison with color histogram features.

Classification Techniques

Classification is performed on the extracted features to determine the level of adulteration. The techniques used for classification are discussed below.

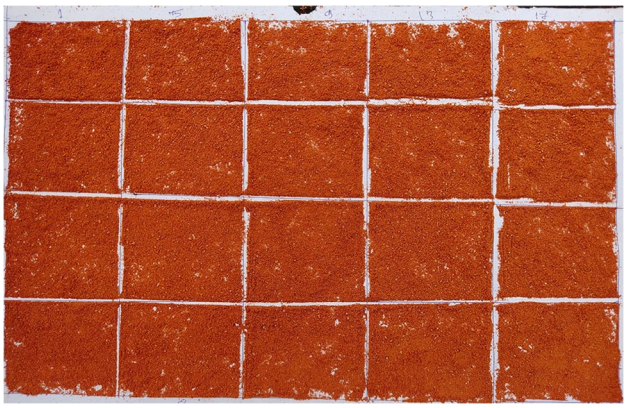

Fig. 1 Prepared samples spread in thin layer over 20 equal sized cells on an A4 white paper

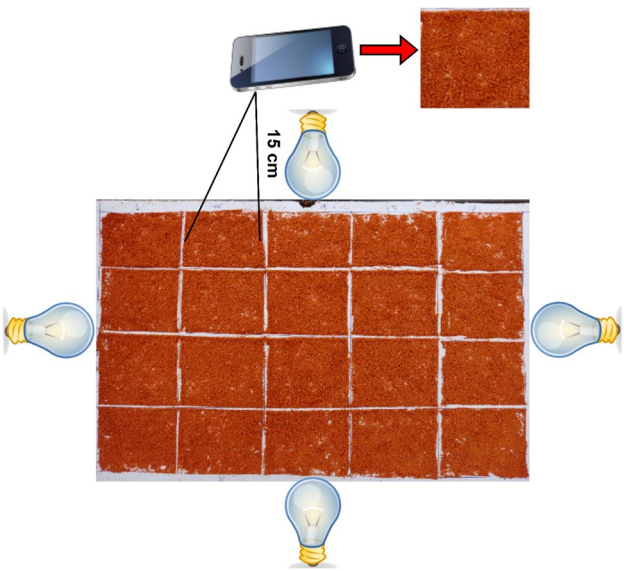

Fig. 2 Schematic diagram for image acquisition with smartphone

Cat Boost Classifier Cat Boost (Dorogush et al. 2018) classifier is a open-source gradient boosted tree-based classifier method by Yandex. It is highly scalable making it suitable for big data applications. It is also one of the best-known algorithms for performing classification on tabular data. This algorithm is implemented using the Cat Boost library. Cat Boost classifier name consist of two words which are "Category" and "Boosting". It works well with the various type of categorical data such as text, image, and audio. It uses ensemble-based learning technique named as Boosting to create an ensemble of well chosen strong and diverse models by combining all weak learners. It relies on the ordering principle and called target based with prior (TBS). It automatically converts categorical data into numerical data using various statistics on combination of categorical features and combination of categorical and numerical features without using any explicit pre-processing technique. It uses oblivious decision trees, where the same splitting rule is used across all intermediate nodes within the same level of tree. It is a robust classifier as it is less prone to overfitting. It also lowers the need for parameter tuning, providing great results using default parameters only. The Oblivious Decision Tree used by CatBoost Classifier has been shown in Fig. 3.

Random Forest Classifier Random forest (Breiman 2001) classifier is a model based on an ensemble of decision trees. It uses bagging technique for creating the ensemble. In bagging technique, we have various base learners. Similarly, random forest consist of various decision trees as base learners. We pick some sample of rows (known as row sampling) and features (known as feature sampling) from the dataset various times with replacement to train various decision trees (known as base learners). Then, we use the majority vote on decision trees inference to calculate the prediction and accuracy of Radom Forest Algorithm. That's why, Random Forest Classifier shows low variance while Decision Trees shows high variance. An implementation by the scikit-learn (Pedregosa et al. 2011) library is used by us. The working of Random Forest Classifier has been shown in Fig. 4.

Extra Trees Classifier Extra Trees classifier (Geurts et al. 2006) is a variant of the random forest classifier which also shows low variance. Unlike random forest it uses entire dataset instead of subsampling with replacement. It also sometimes uses random splits instead of always picking locally optimum splits like random forest classifier. An implementation of this algorithm is provided by the scikit-learn (Pedregosa et al. 2011) library.

Extreme Gradient Boosting Classifier Extreme gradient boosting (Chen and He 2017) is a technique similar to gradient boosting classifier but it utilizes a second-order Taylor approximation making it behave more like Newton–Raphson method rather than gradient descent in function space. It splits the tree level wise (depth wise) (Fig. 5). There are various hyper parameters that can be optimized. XGBoost library is used for implementing this classifier.

Light Gradient Boosting Machine Classifier Light gradient boosting machine (Ke et al. 2017) classifier is fast, distributed, high performance gradient boosting-based model that uses gradient-based one side sampling and exclusive feature bundling for better performance. It splits the tree leaf wise (best first) (Fig. 6). Like XGBoost, there are various hyper parameters that can be optimized. It can lead to overfitting which can be minimized by defining the depth for splitting. The LightGBM library is used for implementing this algorithm.

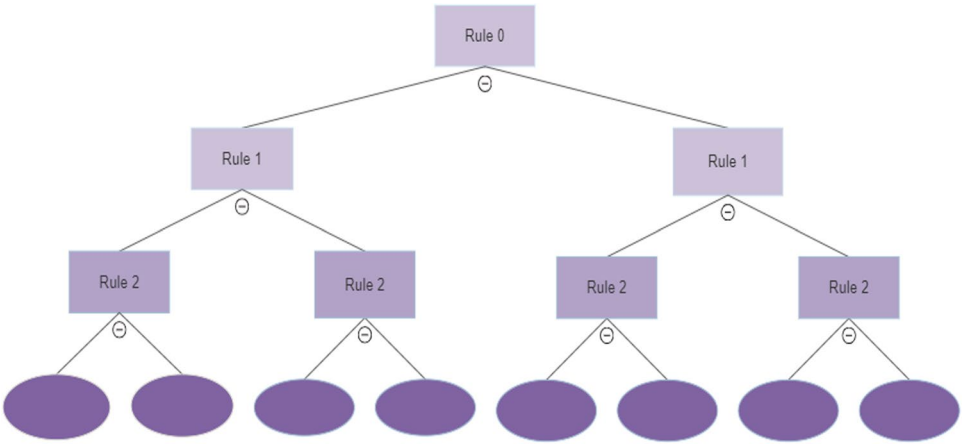

Fig. 3 The oblivious decision tree used by CatBoost classifier

Gradient Boosting Classifier Gradient boosting classifier (Friedman 2001; Hastie et al. 2009) algorithm is similar to Ada Boost Classifier with the main difference being this algorithm has a fixed base estimator. It also has a differentiable loss function and training is done by utilizing the technique of gradient boosting. In Gradient boosting algorithm, suppose we have some function Y(β0, β1). We want to minimize this function by starting with some β0, β1 and keep changing β0, β1 to reduce Y(β0, β1) until we hopefully end up at a minimum.

Repeat until convergence

$$\beta j := \beta j - \alpha \frac{\partial}{\partial \beta_j} Y(\beta 0, \beta 1) for j = 0 and j = 1 \tag{13}$$

The implementation by scikit-learn library is used (Pedregosa et al. 2011).

K Nearest Neighbors Classifier K nearest neighbors (KNN) (Fix and Hodges 1989) classifier is a lazy learner that picks the nearest k points and predicts the modal class of the k points. Different spatial metrics can be used for picking nearest points (Fig. 7). An implementation of KNN by scikit-learn library is used in this study.

This method may utilize any distance metric like Euclidean distance, Manhattan distance, Minkowski distance, etc. Distance between vectors x_i and y_i in a n-dimensional vector space is given for the above-mentioned metrics below,

$$D_{Euclidean}(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{14}$$

$$D_{Manhattan}(x, y) = \sum_{i=1}^{n} |x_i - y_i| \tag{15}$$

$$D_{Minkowski}(x, y) = \left(\sum_{i=1}^{n} \left| x_i^p - y_i^p \right| \right)^{\frac{1}{p}} for some real p \geq 1 \tag{16}$$

Ridge Classifier Ridge classifier (McDonald 2009) method converts output to range –1 to 1 and solves it as a ridge regression problem. One versus all approach is used for multi-class classification.

This model converts class 0 into value of –1 and keeps class 1 values unmodified. Then it applies Ridge regression to model the data (Fig. 8). Then it predicts class 1 for given input if the value of Ridge regression output is greater than 0, otherwise it predicts class 0. Typically, the parameters are optimized either analytically using a formula or estimated using methods based on gradient descent or Newton–Raphson method. Ridge classifier is realized by utilizing the scikit-learn library (Pedregosa et al. 2011).

Support Vector Machine Classifier Support vector machine (SVM) classifier uses support vector machine and kernel trick for performing classification (Platt 2000). One versus rest technique is utilized for multi-classification. SVM implementation by scikit-learn is utilized in this study.

This technique utilizes a support vector w to describe a hyperplane that nearly separates the two classes (Fig. 9), in such a way that the margin between the two classes is large. The support vector is learned by minimizing the following cost function l,

$$l = \lambda \parallel w \parallel + \frac{1}{n} \sum_{i=1}^{n} \max\left(0, 1 - y_i \left(w^T x_i - b\right)\right) \tag{17}$$

Here, λ is a hyperparameter that determines the trade-off between the size of the margin and the classification accuracy. b is some real constant, n denotes number of inputs, x_i denotes ith input and y_i denotes ith output. The loss function is typically minimized using L-BFGS technique.

Ada Boost Classifier Ada Boost (Freund and Schapire 1997; Zhu et al. 2006) uses a technique called adaptive boosting for ensembling several decision stumps (weak-learners) into one strong learner. A decision tree with one depth is known as a decision stump. In boosting techniques, there are various sequential base learners, where incorrectly classified records by first base learner will pass to the next base learner,

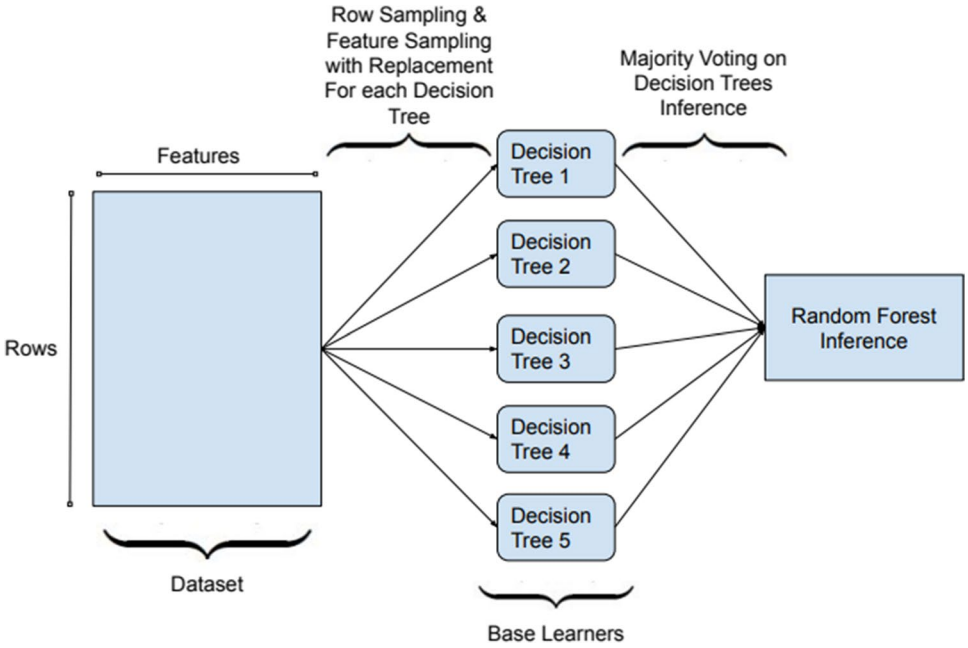

Fig. 4 The working of random forest classifier

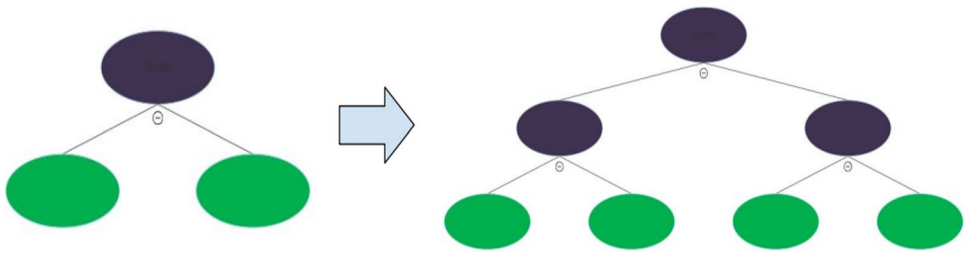

Fig. 5 Level-wise tree growth in XGBoost

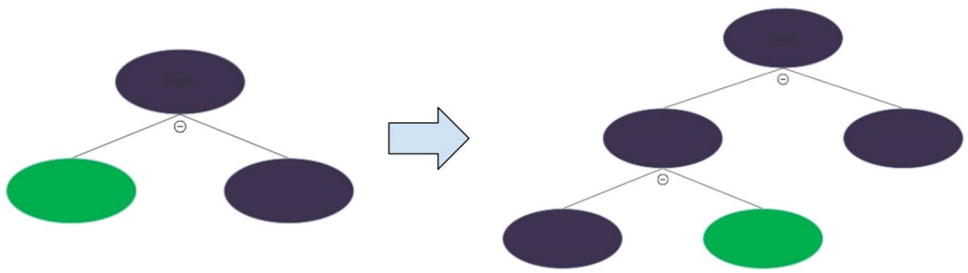

Fig. 6 Leaf-wise tree growth in LightGBM

incorrectly classified records by second base learner will pass to the next base learner and so on. In Ada Boost algorithm (Fig. 10), all the base learners are decision trees and weights will decide the sequence of base learners. We calculate either entropy or Gini index of all decision stumps (for each feature) to calculate the updated weights and normalized weights to decide the sequence of all base learners. An implementation by scikit-learn library is utilized (Pedregosa et al. 2011).

Logistic Regression Logistic regression (Kleinbaum and Klein 2002) performs binary classification using the logistic function applied on a linear regression model to get probabilities (Fig. 11). One versus rest approach is used for multi-class classification. This technique is utilized by us with the help of scikit-learn library (Pedregosa et al. 2011).

This technique models the log odds of binary classification as a linear combination of the inputs. The probability of true class $p(x)$ associated with input x is computed as follows where β_0 and β_1 are trainable parameters,

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \tag{18}$$

The log loss function l is used for optimization and parameter estimation. Here, p_k denotes the true class

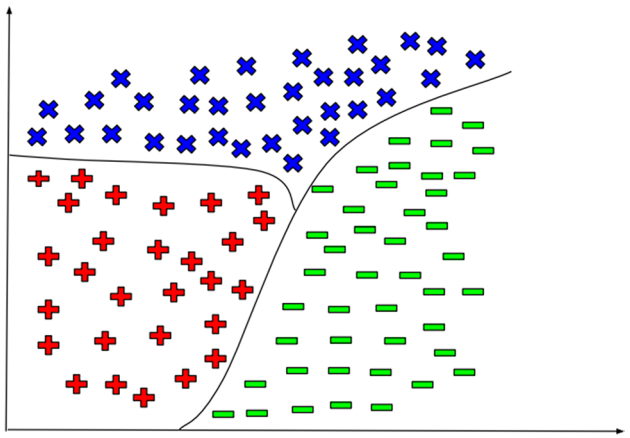

Fig. 7 K nearest neighbors classifier (KNN)

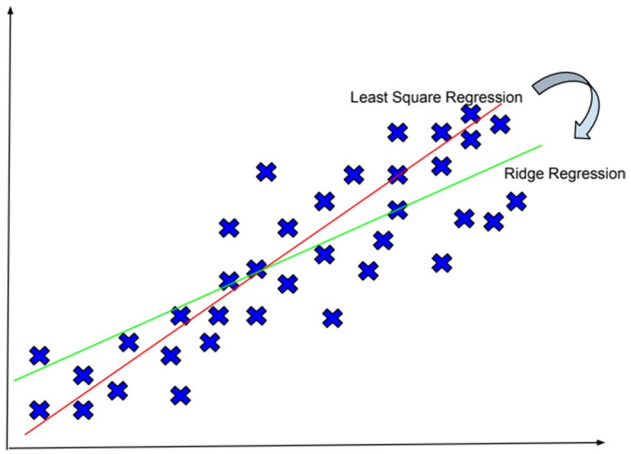

Fig. 8 Ridge classifier

probability associated with input at index k and y_k denotes actual class for input at index k.

$$l = \sum_{k=1} \left(y_k \ln\left(p_k\right) + \left(1 - y_k\right)\ln\left(1 - p_k\right)\right) \tag{19}$$

The parameters are usually solved by direct formula or gradient descent-based techniques.

Dummy Classifier This classification technique predicts the modal class and is used as a baseline. An implementation by scikit-learn with the 'prior' strategy is utilized by us.

Quadratic Discriminant Analysis (Quadratic Classifier) Quadratic discriminant analysis (Tharwat 2016) is a generalization of linear discriminant analysis. This technique is implemented using sckit-learn library in this study. Unlike linear discriminant analysis, this technique does not make the assumption that measurements of each class are identically distributed. This technique utilizes quadratic decision surfaces for performing classification (Fig. 12). This technique works for binary classification. It can be used for multi-class classification using one vs rest technique. This technique works by computing the likelihood ratio and comparing it against some threshold value t; based on the comparison the prediction is made. Let μ_0 and μ_1 be the means associated with both classes and Σ_0 and Σ_1 be the covariance matrices associated with both the classes. Then the likelihood ratio l_r is computed as follows,

$$l_r = \frac{\sqrt{2\pi\left|\Sigma_1\right|^{-1}\exp\left(-\frac{1}{2}\left(x-\mu_1\right)^T\Sigma_1^{-1}\left(x-\mu_1\right)\right)}}{\sqrt{2\pi\left|\Sigma_0\right|^{-1}\exp\left(-\frac{1}{2}\left(x-\mu_0\right)^T\Sigma_0^{-1}\left(x-\mu_0\right)\right)}} \tag{20}$$

The decision boundary we get when thresholding l_r is quadratic in nature. Hence, we call it quadratic discriminant analysis.

Classification Performance Evaluation

Different classifiers are trained on 70% of the total data using threefold cross-validation. Then the classifiers are evaluated on the remaining 30% test data and metrics like accuracy, precision, recall, F_1 score, Cohen's kappa (κ), Matthew's correlation coefficient (MCC), and ROC-AUC score (receiver operator characteristic curve – area under the curve score) are computed.

The above-mentioned metrics are computed using the formulas presented below, where TP denotes the number of true positive predictions, TN is the number of true negative predictions, FN is the number false negative predictions (i.e., type II error), FP is the number false positive predictions (i.e., type I error), P is the total number of positive tuples, and N is the total number of negative tuples. As we are dealing with multi-class classification one vs rest technique is utilized, i.e., the one particular class is treated as positive class and remaining classes are treated as negative class. This is done considering every class as a positive class and the final metrics are computed by averaging over metrics computed by treating one class as positive class or by a natural generalization.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} = \frac{TP + TN}{P + N} \tag{21}$$

$$Precision = \frac{TP}{TP + FP} \tag{22}$$

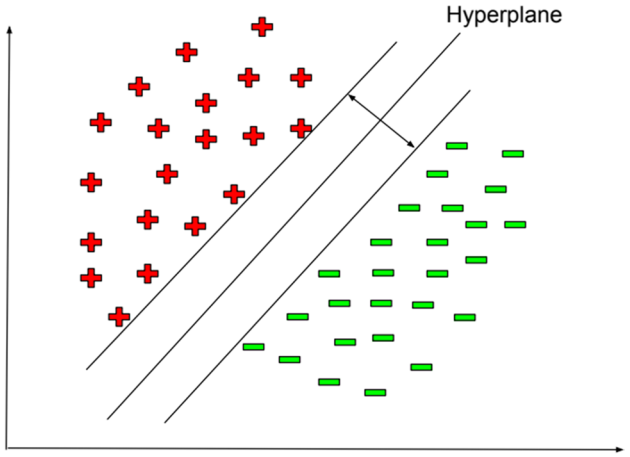

Fig. 9 Support vector machine (SVM) classifier

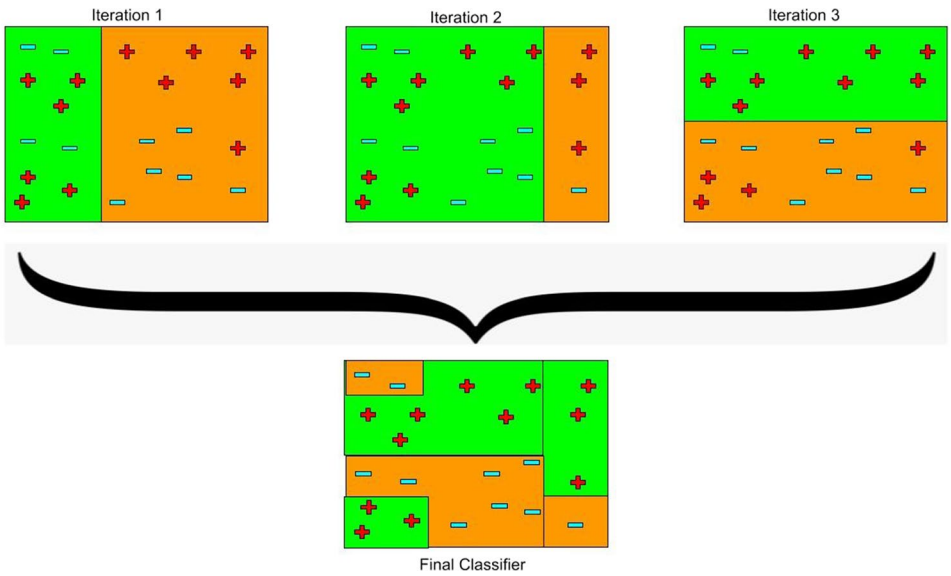

Fig. 10 Adaboost classifier

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{P} \tag{23}$$

$$F_1 Score = \frac{2TP}{2TP + FP + FN} \tag{24}$$

$$\kappa = \frac{2 \times (TP \times TN - FN \times FP)}{(TP + FP) \times (FP + TN) + (TP + FN) \times (FN + TN)} \tag{25}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{26}$$

The ROC curve plots the false positive rate (FPR) against the true positive rate (TPR) as the classification threshold is varied, where true positive rate is the same as recall and FPR is defined below. The area under this curve is the AUC score.

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} \tag{27}$$

Regression Techniques

Different regression models are trained to predict the percentage of chilli powder present in the given observation. Once again, instead of using the image directly the extracted features from the image are used for this purpose.

Ada Boost Regressor Ada boost regressor (Drucker 1997) uses a technique called adaptive boosting for ensembling several decision stumps (weak-learners) into one strong

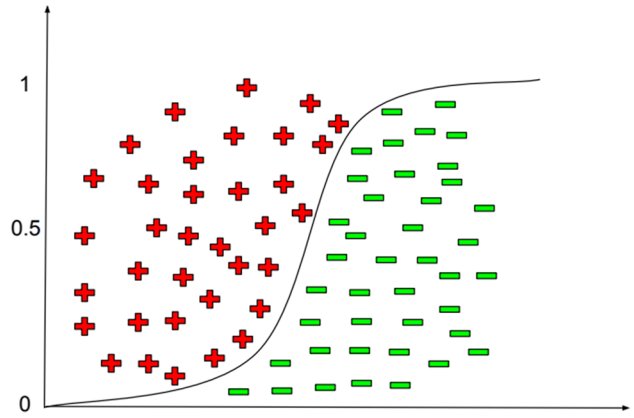

Fig. 11 Logistic regression classifier

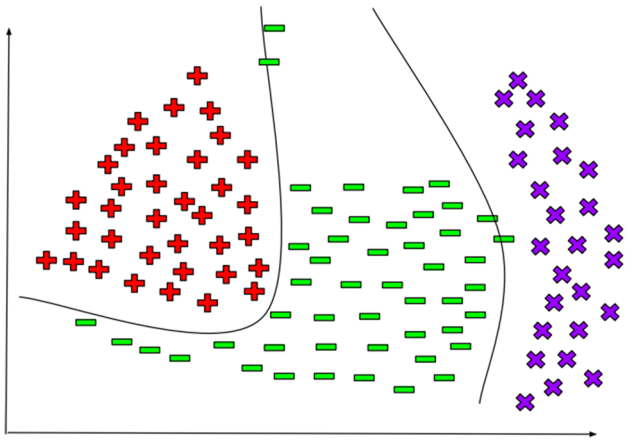

Fig. 12 Quadratic discriminant analysis (quadratic classifier)

learner. A decision tree with one depth is known as a decision stump. It uses ensemble-based learning technique named as Boosting to create an ensemble of well chosen strong and diverse models by combining all weak learners. An implementation by scikit-learn library is utilized (Pedregosa et al. 2011).

Gradient Boosting Regressor Gradient boosting regression algorithm (Friedman 2001) starts by making a single leaf, instead of a tree or stump. This leaf represents an initial guess for the target feature values of all of the sample. When we predict a continuous value, the first guess is the average value. Then, Gradient Boosting algorithm builds a tree. Like AdaBoost, this tree is based on the errors made by the previous tree. But unlike Adaboost, this tree is usually larger than a stump. Gradient Boost still restricts the size of the tree. Thus, like AdaBoost, gradient boost builds fixed sized trees based on the previous tree's errors. Also, it scales all trees by the same amount. An implementation by scikit-learn library is utilized (Pedregosa et al. 2011).

Light Gradient Boosting Machine Regressor This is an enhancement of the gradient boosting regressor algorithm. LightGBM Regressor is fast, distributed, high performance gradient boosting-based model that uses gradient-based one side sampling and exclusive feature bundling for better performance (Ke et al. 2017). It splits the tree leaf wise (best first) (Fig. 6). Like XGBoost, there are various hyper parameters that can be optimized. It can lead to overfitting which can be minimized by defining the depth for splitting. The LightGBM library is used for implementing this algorithm.

Extreme Gradient Boosting Regressor Extreme gradient boosting (Chen and He 2017) is a technique similar to gradient boosting regression algorithm but it utilizes a second-order Taylor approximation making it behave more like Newton–Raphson method rather than gradient descent in function space. It splits the tree level wise (depth wise) (Fig. 5). There are various hyper parameters that can be optimized. XGBoost library is used for implementing this classifier.

Random Forest Regressor Random forest regressor (Breiman 2001) is a model based on an ensemble of decision trees. It uses bagging technique for creating the ensemble. In bagging technique, we have various base learners. Similarly, random forest consists of various decision trees as base learners. We pick some sample of rows (known as row sampling) and features (known as feature sampling) from the dataset various times with replacement to train various decision trees (known as base learners). Then, we use the majority vote on decision trees inference to calculate the prediction and accuracy of radom forest algorithm. An implementation by the scikit-learn (Pedregosa et al. 2011) library is used by us. The working of random forest regressor has been shown in Fig. 4.

Extra Trees Regressor Extra Trees regressor (Geurts et al. 2006) is a variant of the random forest regressor which also shows low variance. Unlike random forest it uses entire dataset instead of subsampling with replacement. It also sometimes uses random splits instead of always picking locally optimum splits like random forest regressor. An implementation of this algorithm is provided by the scikit-learn (Pedregosa et al. 2011) library.

Cat Boost Regressor Cat boost regressor (Dorogush et al. 2017) is a a open-source gradient boosted regression method by Yandex. It is highly scalable making it suitable for big data applications. It is also one of the best-known algorithms for performing regression on tabular data. This algorithm is implemented using the Cat Boost library. Cat Boost regressor name consist of two words which are "Category" and "Boosting." It works well with the various type of categorical data such as text, image, and audio. It uses ensemble-based learning technique named as Boosting to create an ensemble of well chosen strong and diverse models by combining all weak learners. It automatically converts categorical data into numerical data using various statistics on combination of categorical and numerical features without using any explicit pre-processing technique.

K Nearest Neighbors Regressor This technique utilizes a lazy regressor that predicts the mean (or weighted mean in weighted nearest neighbors) of the k nearest neighbors (Song et al. 2017). It has higher prediction power as compare to linear regression as it takes care of the non-linearity. Implementation from scikit-learn of the KNN regressor algorithm is used in this study.

Linear Regressor This regression technique works by fitting a line of best fit to the data. It uses the mean squared error function as the loss function (Schneider et al. 2010). The mean squared error loss (L) is defined as follows,

$$L = \frac{1}{N}\sum_{i}^{N}\left(y_i - \hat{y}\right)^2 \tag{28}$$

where N is the total number of training points, y_i and $\hat{y_i}$ are the actual output and the estimated output.

Huber Regressor This regressor is similar to a linear regressor however it uses Huber loss instead of standard loss of mean squared error making it more robust when dealing with outliers (Huber and Ronchetti 2009). Scikit-learn-based implementation is used in this study. The Huber loss function on residual a with parameter δ is defined as follows:

$$L_{\delta}(a) = \left\{ \frac{a^2}{2} for |a| \le \delta, \delta \times \left(|a| - \frac{\delta}{2}\right) otherwise. \right\} \tag{29}$$

Bayesian Ridge Regressor This technique adds to the basic linear regressor model an additional L2-regularization term in the loss function (MacKay 1992). It also uses Bayesian techniques for determining priors and hyperparameter selection. Scikit-learn-based Bayesian ridge regression implementation is used in this study. L2-regularization term is mathematically defined as follows,

$$L_2 = \sum_{i}^{N} w_i^2 \tag{30}$$

where w_i denotes the weight of the i^{th} input parameter out of N total input parameters.

Lasso Regressor This technique adds an L1-regularization term to the loss function of basic linear regression (Ranstam and Cook 2018). This gives it the capability making the contributions of certain terms zero. Scikit-learn-based Lasso regression implementation is used in this study. The L1-regularization term is defined similarly to the L2-regularization term as follows,

$$L_1 = \sum_{i}^{N} |w_i| \tag{31}$$

Elastic Net Regressor This technique utilizes both L1 and L2 regularization with the basic linear regression model (Hans 2012). This gives it capabilities of Lasso and Ridge regression to some degree. Scikit-learn-based elastic net regression is used in this study.

Lasso Least Angle Regressor This technique utilizes the least angle regressor algorithm in lasso mode (Januaviani et al. 2019). This provides some guarantees on convergence and is usually preferred over least angle regressor. Scikit-learn-based Lasso least angle regressor is used in this study.

Least Angle Regressor Least angle regressor (LAR) algorithm is a forward-stepwise algorithm for regression related to Lasso regression (Efron et al. 2004). This algorithm works stagewise and does not provide any guarantees on convergence. Scikit-learn-based least angle regressor is used in this study.

Dummy Regressor This regressor predicts the arithmetic mean of the target class. This technique is used only as a baseline.

Regression Performance Evaluation

Different regressors are trained on 70% of the total data using threefold cross-validation. Then the regressors are evaluated on the remaining 30% test data and metrics like MAE (mean absolute error), MAPE (mean absolute percentage error), MSE (mean squared Error), R^2 score (Pearson correlation coefficient R squared), RMSE (root mean squared error), and RMSLE (root mean squared logarithmic error) are computed.

The above-mentioned metrics are defined below, where N is the number of predictions, $\bar{y}$ is the mean of actual values, y_i is the i^{th} actual value and $\hat{y}$ is the ith predicted value.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}| MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}| \tag{32}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|^2 MSE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|^2 \tag{33}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|^2} RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|^2} \tag{34}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(y_i - \hat{y}\right)^2}{\sum_{i=1}^{N}\left(y_i - \bar{y}\right)^2} \tag{35}$$

$$MAPE = \frac{100\%}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}}{y_i}\right| \tag{36}$$

$$RMSLE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\log\left(\hat{y}+1\right) - \log\left(y_i+1\right)\right)^2} \tag{37}$$

Model Development Process

To create the machine learning-based models, first we created the dataset by capturing the high-quality Images of red chilli powder adulteration with red brick powder at different proportions. We constructed 12 classes consist of 0%, 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%, and 100% adulterant. The dataset contains 240 images (20 images per class) spread across 12 classes. We split each 4 × zoom image into 4 segments and applied various image color space filters (RGB, HSV, Lab, and YCbCr). Then, extracted features using mean and histogram feature extraction techniques. We used various classification models to classify the adulteration percentage class. We trained Extra Trees Classifier, CatBoost Classifier, Random Forest Classifier, Light Gradient Boosting Machine, Gradient Boosting Classifier, K Neighbors Classifier, Naive Bayes, Decision Tree Classifier, Ridge Classifier, SVM—Linear Kernel, Linear Discriminant Analysis, Logistic Regression, Ada Boost Classifier, Quadratic Discriminant Analysis, Dummy Classifier, and Extreme Gradient Boosting Classifier. Then, we evaluated all the classification models by computing various classification model performance metrics.

We also used regression models to predict the percentage of adulteration in an image. We trained extra trees regressor, K neighbors regressor, CatBoost regressor, random forest regressor, gradient boosting regressor, extreme gradient boosting, light gradient boosting machine, AdaBoost regressor, least angle regression, Bayesian ridge, ridge regression, linear regression, lasso regression, elastic net, Huber regressor, decision tree regressor, orthogonal matching pursuit, Lasso least angle regression, dummy regressor, and passive aggressive regressor to determine the percentage of red brick powder adulteration in red chilli powder using an image. Then, we evaluated all the regression models by computing various regression model performance metrics. We used various python libraries for implementing these machine learning models including scikit-learn, pandas, NumPy, scikit-image, XGBoost, LGBMBoost, CatBoost, and PyCaret. We used GoogleColab to train all the machine learning models. Also, threefold cross-validation was used during training process for the model to generalize better and prevent overfitting. The results of this process are presented and discussed in the following section. The work flow adopted to detect adulterated chilli powder and quantification of adulteration has been shown in Fig. 13. We have modeled our dataset for performing adulteration quantification. The link to the dataset for adulterated red chilli powder with brick powder is https://doi.org/10.17632/cpm7y44746.1. The code which was used for performing adulteration quantification is available at https://github.com/arun5309/chilli_adulteration_quantification.

Result and Discussion

The overall experimental results indicate that the combination of color space, feature extraction and prediction techniques adopted in this study yield good results and they do not take too much time to train or make predictions. The experimental results are discussed in detail in the remaining part of this section.

From the outcomes of the experiments carried out summarized in Tables 2–17 one can see that histogram feature extraction technique consistently outperformed mean feature extraction techniques in majority of cases. With approximately a difference of 0.08–0.3 in term of R^2 score for regression and by around 30% difference in classification accuracy. Also, regression techniques yield better and more useful prediction than classification techniques. This being due to classification techniques treating all classes representing adulteration ratio as independent and unrelated, in contrast to regression treating it on a spectrum. This is why classification models have a prediction accuracy around 90% and regression models can explain about 98% of the variance as inferred from the R^2 score.

To predict the percentage of adulteration, we found that Extra Tree regressor was best performing with an R^2 score of 0.9812 and 0.9837 for YCbCr and Lab color spaces histogram features respectively. The YCbCr and Lab color spaces have similar performance characteristics in case of regression for histogram features, with around 0.98 R^2 score as seen from Tables 11 and Table 13. The HSV and RGB color spaces perform slightly worse in regression when using histogram feature extraction technique. Around, 0.97 and 0.96 R^2 respectively as seen from Tables 15 and Table 17. Performance of RGB and YCbCr color space is somewhat worse than Lab for channel mean value-based features. With R^2 scores of 0.87, 0.86, and 0.9 respectively as seen from Tables 16, 10, and 12. However, HSV perform significantly worse than other color spaces for regression using channel mean value features. With an R^2 of 0.61 as seen in Table 14 R^2 score of regression methods with different color spaces for grading chilli powder images is reported in Fig. 14.

To classify the adulteration class, we found that Cat Boost classifier was best performing with an Accuracy of 0.9049 and 0.908 for YCbCr and HSV color spaces histogram features respectively. Also, YCbCr and Lab have similar performance characteristics. With a classification accuracy score of around 60% for channel mean value features and 90% score for histogram-based features as seen from Tables 2, 3, 4, and 5. HSV has similar performance Charteristics to YCbCr and Lab when it comes to histogram features (accuracy score of around 90%) but it performs significantly worse when it comes to channel mean value features (accuracy score of 44%) as seen from

Table 6 and Table 7. RGB performs somewhat worse than other color spaces when histogram features (around 87% classification accuracy from Table 8) are used but surprisingly outperforms YCbCr and Lab color spaces by a small margin (approximately 63% classification accuracy from Table 9) when channel mean value features are used. Classification accuracy of classifiers with different color spaces for grading chilli powder images is reported in Fig. 15. Figure 16 shows the confusion matrix related to the image grading of red chilli powder adulteration with red brick powder. Most of the images are predicted correctly by the classifier, and a few images were misclassified, with a difference of 5% adulteration in actual and predicted classes.

One can infer from Tables 10, 11, 12, 13, 14, 15, 16, and 17 the tree-based algorithms and KNN are the regressors with the best performance across color space and feature extraction methods. There are only minor variations in their performance characteristics and achieving best scores on all the regression experiments. Other regressors perform somewhat or significantly worse than the regressors mentioned previously.

Once again, the tree-based algorithms perform well in classification algorithms as seen from Tables 2 to Table 9. With Ada boost being a notable exception, which performs slightly worse than SVM. These results aren't surprising as these algorithms have great success when dealing with tabular data. KNN, SVM, logistic regression and naive Bayes algorithms lag slightly behind the tree-based algorithms in terms of performance. Quadratic and linear discriminant analysis classifiers had a good success when it came to classification based on channel mean value-based features often outperforming the classifiers (including tree-based ones) mentioned earlier.

Most of these algorithms are fast to train and can be trained in a few seconds on datasets of this size and hardware configuration used in this study as seen from Tables 2–17. This indicates that these algorithms can be deployed on low-end devices in real-time scenarios and these would still yield decent results. However, CatBoost, LightGBM, and gradient boosted trees took much longer to train than other algorithms. These anomalies can be explained by stating that these three algorithms do not utilize hardware acceleration (which they were designed to take advantage of) in our benchmarks rather than having inefficiencies in the design of these algorithms.

The methods employed in this study perform nearly as well as other methods employed in the literature (Jahanbakhshi et al. 2021b), while consuming little computational resources for both training and inferencing. This show cases the practicality and utility of the methods employed in this study especially in situations with low computational resources and lack of advanced scientific equipment. For example, performing adulteration quantification of chilli powder using a low end smartphone.

Tree-based techniques outperformed most other techniques in both regression and classification. KNN closely followed the tree-based techniques in terms of performance in both regression and classification. SVM, logistic regression, naive Bayes classifier, and linear and quadratic discriminant analysis had good performance in case of classification, although discriminant analysis-based methods were only successful when using channel mean value-based features for classification.

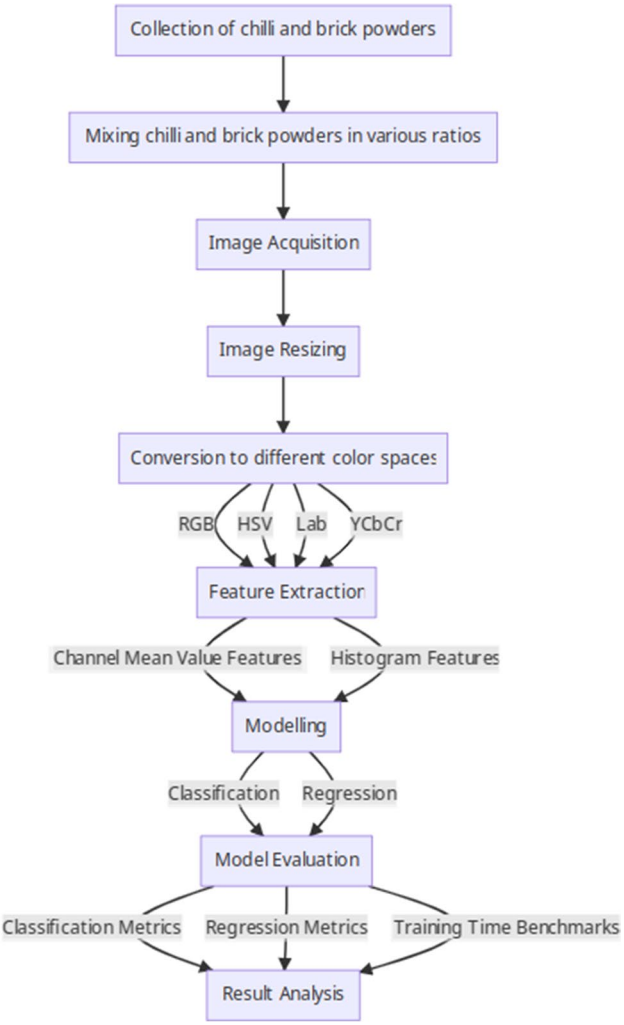

Fig. 13 The work flow adopted to detect adulterated chilli powder and quantification of adulteration

We also saw that regression-based techniques were more successful than classification-based techniques for solving this problem. The reason being strictly grouping percentage of classification into bins isn't a very good idea in practice and for this particular application small amounts of errors are acceptable in practice. This is clear when we compare Tables 10–17 with Tables 2–9.

The Lab and YCbCr color spaces had a good performance overall and produced stable results. The RGB color space had moderate performance and stability although it produced a few surprisingly good results. The HSV had performance comparable to that of Lab and YCbCr but it was the most unstable among the four color spaces and had a few performance

Table 2 Evaluation of classifiers in terms of accuracy, AUC score, recall, precision, F1 score, Kappa, MCC, and TT (Sec) with YCbCr mean features in the testing phase

Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (Sec)
Quadratic discriminant analysis	0.6043	0.9389	0.5994	0.5981	0.5927	0.568	0.5697	0.03
Linear discriminant analysis	0.5859	0.9252	0.5733	0.5841	0.5726	0.5472	0.5495	0.0233
Extra trees classifier	0.5583	0.8966	0.5531	0.5535	0.5484	0.5177	0.5191	0.6067
CatBoost classifier	0.5445	0.9197	0.539	0.5401	0.5345	0.5027	0.5043	9.3333
Logistic regression	0.5414	0.9101	0.5303	0.5666	0.5289	0.4986	0.5014	1.71
Random forest classifier	0.5276	0.8985	0.5223	0.5156	0.5131	0.4842	0.4858	0.6467
Light gradient boosting machine	0.5138	0.8887	0.5077	0.525	0.5118	0.4691	0.4703	0.6467
K neighbors classifier	0.4985	0.8613	0.4933	0.5007	0.4875	0.4524	0.4546	0.1667
Extreme gradient boosting	0.4985	0.8825	0.4921	0.4996	0.4816	0.4515	0.4529	0.7621
Decision TREE CLASSIFIER	0.4969	0.7257	0.4921	0.4963	0.4933	0.4509	0.4517	0.0267
Gradient boosting classifier	0.4801	0.8825	0.4708	0.4786	0.4756	0.4321	0.433	1.3933
Naive Bayes	0.3712	0.814	0.3688	0.3713	0.3276	0.314	0.3216	0.0233
Ridge classifier	0.3205	0.0	0.3068	0.1739	0.2029	0.2555	0.2748	0.0267
Ada Boost classifier	0.2117	0.6577	0.204	0.1827	0.1415	0.1366	0.1689	0.1333
SVM—linear kernel	0.1596	0.0	0.157	0.1313	0.0738	0.0822	0.1103	0.0833
Dummy classifier	0.0874	0.5	0.0833	0.0076	0.0141	0.0	0.0	0.02

dips. This clearly highlights that the choice of color space is extremely important when performing analysis on chilli powder. As the distinction between chilli powder and brick powder is made primarily based on color.

Overall, from the results of extensive experimentation carried out in this study one can infer that histogram-based feature technique outperforms channel mean value-based feature extraction, YCbCr and Lab color spaces outperform HSV and RGB, tree-based algorithms have best performance characteristics across the board and training time isn't a significant factor for datasets of this size with modern computational resources. It is also clear that regression techniques are more suitable than classification for quantifying amount of adulteration in red chilli powder. It is clear from these results that the approach taken in this study is a promising one for adulteration quantification. However, further research is needed to extend this result into adulteration quantification problems on other foods.

Conclusion

Many models and features extraction techniques have been compared and it is clear that these techniques can be used for efficiently in the detecting and quantifying the percentage of brick powder present in given chilli powder sample. Utilizing feature extraction techniques instead of utilizing the entire image makes our method suitable for large-scale and mobile application. The use of simple statistical machine learning algorithms instead of deep learning algorithms makes the model size smaller as well. In this study many image processing and ML techniques were employed and compared for performing adulteration quantification of chilli powder. To classify the adulteration class, we found that Cat Boost classifier was best performing with an accuracy of 0.9049 and 0.908 for YCbCr and HSV color spaces histogram features respectively. To predict the percentage of adulteration, we found that Extra Tree regressor was best performing with an R^2 score of 0.9812 and 0.9837 for YCbCr and Lab color spaces histogram features respectively. The results achieved were comparable to that of other results found in the literature employing DL (deep learning) methods. Evidencing that the methodology adopted in this study is viable, efficient, and especially suited for use in low resource and mobile environments. In the future a mobile application can be made for utilizing this technology in a more accessible manner.

Acknowledgements We acknowledge Mr. Aniket Ghosh, Mr. Gopal Roy, Mr. Prasun Kumar Saha final year student (2022 batch) of department of Food Processing Technology and Sri Snehashis Guha, PIC Malda polytechnic, Malda for their support to conduct this study. Thanks to GAIN (Axencia Galega de Innovación) for supporting this research (grant number IN607A2019/01).

Author Contribution T.R.: conceptualization, methodology, investigation, validation, formal analysis, writing-original draft preparation; T.C.: methodology, investigation, validation, formal analysis, contribution in writing; V.R.A.: methodology, investigation, validation, formal analysis, and contribution in writing in relevant section; M.K.: data analysis, writing-review and editing, final draft supervision, and monitoring; M.A.S.: review and editing, final draft supervision, and monitoring; J.M.L.: review and editing, final draft supervision and monitoring. All authors read and approved the final manuscript.

Funding Funding for open access charge: Universidade de Vigo/CISUG.

Table 3 Evaluation of classifiers in terms of accuracy, AUC score, recall, precision, F1 score, kappa, MCC, and TT (Sec) with YCbCr histogram features in the testing phase

Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (Sec)
Extra trees classifier	0.9049	0.9933	0.9031	0.9106	0.904	0.8962	0.8968	0.6567
CatBoost classifier	0.9049	0.9943	0.9034	0.9106	0.9044	0.8962	0.8967	613.72
Random forest classifier	0.8957	0.9912	0.8939	0.9006	0.895	0.8861	0.8866	0.8633
Light gradient boosting machine	0.8681	0.9904	0.8662	0.8764	0.8675	0.856	0.8569	8.1033
Extreme gradient boosting	0.8481	0.988	0.8347	0.8576	0.8465	0.8378	0.8321	10.2476
Gradient boosting classifier	0.8237	0.982	0.8219	0.8327	0.8235	0.8075	0.8083	36.3733
K neighbors classifier	0.822	0.9742	0.8193	0.8288	0.8194	0.8057	0.8068	0.2967
Naive Bayes	0.7638	0.9593	0.7643	0.7725	0.7627	0.7422	0.7433	0.07
Decision tree classifier	0.7424	0.8595	0.741	0.7538	0.7406	0.7188	0.72	0.16
Ridge classifier	0.5429	0.0	0.5249	0.6535	0.4903	0.4993	0.5151	0.0467
SVM—linear kernel	0.4998	0.0	0.4972	0.4815	0.4262	0.454	0.4868	0.23
Linear discriminant analysis	0.4478	0.8165	0.4438	0.4594	0.4463	0.3974	0.3989	0.21
Logistic regression	0.4295	0.9165	0.4012	0.3889	0.3296	0.3729	0.4044	1.31
Ada Boost classifier	0.293	0.708	0.2768	0.2331	0.2246	0.2216	0.319	0.85
Quadratic discriminant analysis	0.2807	0.6071	0.2751	0.2804	0.2648	0.2139	0.2173	0.13
Dummy classifier	0.092	0.5	0.0833	0.0085	0.0155	0.0	0.0	0.0733

Table 4 Evaluation of classifiers in terms of accuracy, AUC score, recall, precision, F1 score, kappa, MCC, and TT (Sec) with Lab mean features in the testing phase

Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (Sec)
Quadratic discriminant analysis	0.5951	0.9417	0.5887	0.6016	0.5854	0.5579	0.5599	0.03
extra trees classifier	0.5951	0.9103	0.591	0.6056	0.5933	0.558	0.5593	0.6033
Linear discriminant analysis	0.5844	0.925	0.5751	0.6033	0.5781	0.546	0.5485	0.02
CatBoost classifier	0.5691	0.9168	0.5655	0.5852	0.5712	0.5297	0.5308	8.06
Extreme gradient boosting	0.5643	0.9088	0.556	0.5714	0.5674	0.5261	0.523	0.7412
Random forest classifier	0.5537	0.9051	0.5497	0.5694	0.5534	0.5128	0.514	0.6333
K neighbors classifier	0.5506	0.8798	0.5454	0.5565	0.5417	0.5095	0.5114	0.16
Light gradient boosting machine	0.5261	0.885	0.5213	0.5391	0.5271	0.4828	0.4839	0.6267
Gradient boosting classifier	0.4893	0.8784	0.4859	0.5032	0.4902	0.4427	0.4437	1.23
Decision tree classifier	0.4601	0.7056	0.4583	0.469	0.4596	0.4109	0.4118	0.02
Naive Bayes	0.4003	0.81	0.3917	0.4328	0.3582	0.3449	0.3532	0.0233
Ada Boost classifier	0.2807	0.6927	0.2738	0.2711	0.2253	0.2132	0.2347	0.1233
SVM—linear kernel	0.2745	0.0	0.2711	0.2149	0.1943	0.2096	0.2553	0.0833
Ridge classifier	0.2669	0.0	0.2569	0.2012	0.175	0.1967	0.229	0.02
Logistic regression	0.227	0.7788	0.2137	0.1039	0.1158	0.152	0.1832	0.98
Dummy classifier	0.0874	0.5	0.0833	0.0076	0.0141	0.0	0.0	0.0167

Data Availability All the data used in the manuscript are available in the tables and figures.

Code Availability Not applicable.

Declarations

Competing interests The authors declare no competing interests.

Ethics Approval Not applicable.

Consent to Participate All authors has given their full consent to participate.

Consent for Publication All authors has given their full consent for publication.

Conflict of Interest Tanmay Sarkar declares that she has no conflict of interest. Tanupriya Choudhury declares that she has no conflict of interest. Nikunj Bansal declares that he has no conflict of interest. VR Arunachalaeshwaran declares that she has no conflict of interest. Mars Khayrullin declares that she has no conflict of interest. Mohammad

Table 5 Evaluation of classifiers in terms of accuracy, AUC score, recall, precision, F1 score, kappa, MCC, and TT (Sec) with Lab histogram features in the testing phase

Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (Sec)
Extra trees classifier	0.9003	0.9942	0.9	0.9054	0.8979	0.8912	0.8921	0.68
CatBoost classifier	0.8988	0.9954	0.8986	0.9044	0.8961	0.8895	0.8905	398.6767
Light gradient boosting machine	0.888	0.9922	0.8877	0.8985	0.8886	0.8778	0.8786	5.5433
Extreme gradient boosting	0.8823	0.9878	0.8812	0.8976	0.8815	0.8739	0.8734	6.9874
Random forest classifier	0.8757	0.9924	0.8747	0.879	0.8733	0.8644	0.8652	0.7867
Gradient boosting classifier	0.8435	0.9843	0.8426	0.8656	0.8459	0.8293	0.8308	25.63
K neighbors classifier	0.8206	0.9697	0.8193	0.8299	0.8185	0.8042	0.8053	0.2967
Linear discriminant analysis	0.8052	0.9762	0.8049	0.8171	0.8016	0.7874	0.7892	0.2167
Naive Bayes	0.7485	0.9598	0.7475	0.7678	0.7455	0.7255	0.7275	0.0967
Decision tree classifier	0.7347	0.8554	0.7337	0.7441	0.7338	0.7105	0.7116	0.1233
Ridge classifier	0.6227	0.0	0.6128	0.6524	0.5881	0.5877	0.5953	0.0867
Logistic regression	0.5645	0.9253	0.5516	0.6	0.5091	0.5235	0.539	1.22
SVM—linear kernel	0.4752	0.0	0.4706	0.5496	0.4396	0.428	0.4726	0.2333
Ada boost classifier	0.2745	0.7175	0.2645	0.2259	0.2011	0.206	0.2911	0.7067
Quadratic discriminant analysis	0.2699	0.6019	0.2682	0.2932	0.2667	0.2036	0.2063	0.1233
Dummy classifier	0.0874	0.5	0.0833	0.0076	0.0141	0.0	0.0	0.07

Table 6 Evaluation of classifiers in terms of accuracy, AUC score, recall, precision, F1 score, kappa, MCC, and TT (Sec) with HSV mean features in the testing phase

Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (Sec)
CatBoost classifier	0.4463	0.8654	0.4516	0.4546	0.4446	0.3957	0.3967	9.0133
Quadratic discriminant analysis	0.4325	0.8918	0.4373	0.444	0.4262	0.3807	0.3832	0.03
k neighbors classifier	0.4279	0.8138	0.4363	0.449	0.4268	0.3762	0.3787	0.16
Random forest classifier	0.4203	0.8435	0.4246	0.4277	0.4187	0.3673	0.3682	0.64
Extreme gradient boosting	0.4174	0.8454	0.4199	0.4142	0.4120	0.3509	0.3549	0.8269
Extra trees classifier	0.4049	0.8503	0.4089	0.4152	0.404	0.3504	0.3514	0.6033
Light gradient boosting machine	0.4018	0.8314	0.4067	0.4141	0.4029	0.3469	0.3477	0.6767
Gradient boosting classifier	0.3972	0.8396	0.4022	0.4091	0.3975	0.3418	0.3428	1.4267
Linear discriminant analysis	0.3743	0.8637	0.3777	0.3782	0.3637	0.3154	0.3184	0.0267
Decision tree classifier	0.3605	0.6508	0.3652	0.3636	0.3583	0.3019	0.3026	0.0233
Naive Bayes	0.3344	0.802	0.3446	0.3807	0.3075	0.2738	0.2824	0.0233
Ada boost classifier	0.2194	0.6514	0.2201	0.1746	0.1651	0.144	0.1757	0.1367
Ridge classifier	0.2162	0.0	0.2118	0.1399	0.1186	0.1387	0.1643	0.0267
Logistic regression	0.1641	0.7032	0.1441	0.1053	0.0727	0.074	0.0983	1.01
SVM—linear kernel	0.1502	0.0	0.1554	0.1194	0.0907	0.0725	0.1018	0.08
Dummy classifier	0.0997	0.5	0.0833	0.0099	0.0181	0.0	0.0	0.02

Ali Shariati declares that he has no conflict of interest. Jose Manuel Lorenzo declares that he has no conflict of interest.

References

Al-Awadhi MA, Deshmukh RR (2021) Detection of adulteration in coconut milk using infrared spectroscopy and machine Learning. Sana'a, Yemen, In: 2021 Int Conf Modern Trends Inf Commun Technol Ind (MTICTI) 1–4. https://doi.org/10.1109/MTICTI53925.2021.9664764

Al-Awadhi MA, Deshmukh RR (2022) Honey Adulteration detection using hyperspectral imaging and machine learning. Vijayawada, India, In: 2022 2nd Int Conf Artif Intell Signal Proc (AISP) 1–5. https://doi.org/10.1109/AISP53593.2022.9760585

Anami BS, Malvade NN, Palaiah S (2019) Automated recognition and classification of adulteration levels from bulk paddy grain samples. Inf Process Agric 6:47–60. https://doi.org/10.1016/j.inpa.2018.09.001

Ayob O, Hussain PR, Suradkar P et al (2021) Evaluation of chemical composition and antioxidant activity of Himalayan red chilli varieties. LWT 146:111413. https://doi.org/10.1016/j.lwt.2021.111413

Table 7 Evaluation of classifiers in terms of accuracy, AUC score, recall, precision, F1 score, kappa, MCC, and TT (Sec) with HSV histogram features in the testing phase

Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (Sec)
CatBoost classifier	0.908	0.995	0.9095	0.9115	0.9065	0.8995	0.9001	722.9233
Light gradient boosting machine	0.8973	0.993	0.8981	0.9012	0.8968	0.8878	0.8883	10.9167
Extreme gradient boosting	0.8955	0.9908	0.8915	0.9005	0.8845	0.8857	0.8782	11.2163
Extra trees classifier	0.8819	0.9903	0.8825	0.8857	0.8811	0.871	0.8715	0.6867
Random forest classifier	0.8742	0.9899	0.8761	0.8768	0.8732	0.8627	0.8631	0.9233
Gradient boosting classifier	0.8497	0.9851	0.8508	0.857	0.8495	0.8359	0.8366	54.7067
K neighbors classifier	0.8282	0.9825	0.8306	0.8383	0.8245	0.8125	0.8138	0.3833
Decision tree classifier	0.7622	0.8699	0.7643	0.777	0.7587	0.7403	0.7421	0.1967
Naive Bayes	0.7516	0.9492	0.7547	0.7608	0.7474	0.7286	0.7302	0.1067
Linear discriminant analysis	0.7516	0.9657	0.755	0.7632	0.7511	0.7287	0.73	0.2733
Ridge classifier	0.6058	0.0	0.5967	0.6943	0.5632	0.569	0.5806	0.12
SVM—linear kernel	0.5874	0.0	0.5786	0.6567	0.546	0.5493	0.5725	0.27
Logistic regression	0.5153	0.9212	0.4944	0.5078	0.4467	0.4688	0.4886	1.5033
Quadratic discriminant analysis	0.2684	0.6003	0.2662	0.3228	0.2682	0.2007	0.2037	0.1033
Ada Boost classifier	0.2377	0.6356	0.2407	0.1556	0.162	0.1578	0.2928	1.15
Dummy classifier	0.0982	0.5	0.0833	0.0096	0.0176	0.0	0.0	0.0733

Table 8 Evaluation of classifiers in terms of accuracy, AUC score, recall, precision, F1 score, kappa, MCC, and TT (Sec) with RGB mean features in the testing phase

Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (Sec)
Quadratic discriminant Analysis	0.6349	0.9483	0.6294	0.6303	0.6256	0.6014	0.6028	0.0267
Linear discriminant analysis	0.5767	0.9272	0.5682	0.598	0.5735	0.5374	0.5396	0.03
CatBoost classifier	0.5429	0.9093	0.5394	0.5465	0.5383	0.5009	0.5021	10.2533
Random forest classifier	0.5322	0.8927	0.5292	0.536	0.5256	0.4892	0.4905	0.6367
Extra trees classifier	0.5306	0.8908	0.5265	0.5326	0.5244	0.4874	0.4887	0.6067
K neighbors classifier	0.5168	0.8692	0.5156	0.5298	0.5076	0.4728	0.4755	0.16
Light gradient boosting machine	0.4969	0.8768	0.4916	0.4941	0.49	0.4506	0.4516	0.66
Extreme gradient boosting	0.4869	0.8755	0.4822	0.5049	0.4889	0.4465	0.4441	0.7216
Gradient boosting classifier	0.48	0.8748	0.4742	0.5008	0.4809	0.4323	0.4336	1.2667
Naive Bayes	0.4064	0.8313	0.3968	0.4262	0.3642	0.3508	0.3585	0.0267
Decision tree classifier	0.3911	0.6679	0.3857	0.3928	0.3858	0.3353	0.3364	0.0233
Ridge classifier	0.2423	0.0	0.2246	0.1275	0.1332	0.1664	0.1962	0.0233
Logistic regression	0.2178	0.7734	0.199	0.1006	0.1075	0.1365	0.1764	1.09
SVM—linear kernel	0.1963	0.0	0.1921	0.1625	0.1261	0.1243	0.1643	0.08
Ada Boost classifier	0.1778	0.6237	0.1669	0.1326	0.1117	0.0952	0.1335	0.1267
Dummy classifier	0.0982	0.5	0.0833	0.0096	0.0176	0.0	0.0	0.0233

Bansal S, Singh A, Mangal M et al (2017) Food adulteration: sources, health risks, and detection methods. Crit Rev Food Sci Nutr 57:1174–1189. https://doi.org/10.1080/10408398.2014.967834

Bao P, Zhang L (2003) Noise reduction for magnetic resonance images via adaptive multiscale products thresholding. IEEE Trans Med Imaging 22:1089–1099. https://doi.org/10.1109/TMI.2003.816958

Bianchini D, De Antonellis V, Franceschi N, Melchiori M (2016) PREFer: a prescription-based food recommender system. Comput Stand Interfaces 54:64–75. https://doi.org/10.1016/j.csi.2016.10.010

Boateng AA, Sumaila S, Lartey M et al (2022) Evaluation of chemometric classification and regression models for the detection of syrup adulteration in honey. LWT 163:113498. https://doi.org/10.1016/j.lwt.2022.113498

Breiman L (2001) Random Forests. Mach Learn 45:5–32. https://doi.org/10.1023/A:1010933404324

Brighty SPS, Harini GS, Vishal N (2021) Detection of adulteration in fruits using machine learning. Chennai, India, In: 2021 Sixth Int Conf Wirel Commun, Signal Proc Netw (WiSPNET) 37–40. https://doi.org/10.1109/WiSPNET51692.2021.9419402

Calle JLP, Barea-Sepúlveda M, Ruiz-Rodríguez A et al (2022a) Rapid Detection and quantification of adulterants in fruit juices using machine learning tools and spectroscopy data. Sensors (Basel) 22:. https://doi.org/10.3390/s22103852

Calle JLP, Ferreiro-González M, Ruiz-Rodríguez A et al (2022b) Detection of adulterations in fruit juices using machine learning methods over FT-IR Spectroscopic data. Agron 12. https://doi.org/10.3390/agronomy12030683

Table 9 Evaluation of classifiers in terms of accuracy, AUC score, recall, precision, F1 score, kappa, MCC, and TT (Sec) with RGB histogram features in the testing phase

Model	Accuracy	AUC	Recall	Prec	F1	Kappa	MCC	TT (Sec)
CatBoost classifier	0.868	0.9905	0.8659	0.8782	0.8675	0.8558	0.8569	966.5133
Extra trees classifier	0.865	0.9914	0.8622	0.873	0.8648	0.8525	0.8533	0.75
Random forest classifier	0.8512	0.9883	0.8489	0.8596	0.8511	0.8374	0.8383	1.39
Light gradient boosting machine	0.8358	0.9831	0.8343	0.8405	0.8349	0.8207	0.8213	14.1967
Gradient boosting classifier	0.8129	0.9746	0.81	0.8237	0.8132	0.7956	0.7967	69.7867
Extreme gradient boosting	0.8098	0.9737	0.8057	0.8123	0.7962	0.7767	0.7660	17.8931
K neighbors classifier	0.7745	0.972	0.7723	0.7899	0.7718	0.7537	0.7556	0.4867
Linear discriminant analysis	0.75	0.9703	0.7488	0.7598	0.7484	0.7269	0.728	0.2233
Naive Bayes	0.7223	0.9376	0.7182	0.7399	0.7232	0.6966	0.6986	0.1333
Decision tree classifier	0.7086	0.8412	0.7079	0.7183	0.708	0.6819	0.6828	0.3267
Ridge classifier	0.5061	0.0	0.479	0.5443	0.4515	0.4582	0.4706	0.1067
SVM—linear kernel	0.4188	0.0	0.4073	0.5153	0.373	0.3652	0.4231	0.3467
Ada Boost classifier	0.2562	0.6983	0.2352	0.1915	0.1848	0.1807	0.2642	1.35
Logistic regression	0.2561	0.9002	0.228	0.1405	0.1387	0.1772	0.2185	2.09
Quadratic discriminant analysis	0.2039	0.5642	0.1959	0.2131	0.1943	0.1283	0.1306	0.1167
Dummy classifier	0.0997	0.5	0.0833	0.0099	0.0181	0.0	0.0	0.08

Table 10 Evaluation of regression models in terms of MAE, MSE, RMSE, R^2 score, RMSLE, MAPE, and TT (Sec) with YCbCr mean features in the testing phase

Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE	TT (Sec)
Extra trees regressor	6.1393	81.7625	8.9411	0.8657	0.5224	0.0881	0.495
K neighbors regressor	6.913	103.1967	10.0831	0.8325	0.4829	0.0977	0.077
CatBoost regressor	6.8921	107.5758	10.1721	0.8284	0.629	0.095	1.735
Random forest regressor	7.057	123.4021	10.9495	0.7992	0.6056	0.0945	0.648
Gradient boosting regressor	7.4397	126.5915	11.0774	0.7963	0.6388	0.1009	0.083
Extreme gradient boosting	7.2394	135.7678	11.4281	0.7799	0.5651	0.0974	0.372
Light gradient boosting machine	7.7986	140.5417	11.647	0.7788	0.6574	0.1052	0.108
AdaBoost regressor	10.6193	182.0709	13.383	0.7069	0.703	0.1446	0.073
Least angle regression	11.6622	212.3135	14.5206	0.6665	0.9727	0.1391	0.027
Bayesian Ridge	11.6577	212.3143	14.5206	0.6665	0.9733	0.1389	0.022
Ridge regression	11.6617	212.3127	14.5206	0.6665	0.9727	0.1391	0.022
Linear regression	11.6622	212.3136	14.5206	0.6665	0.9727	0.1391	0.503
Lasso regression	11.6136	212.9138	14.5409	0.6655	0.9795	0.1376	0.025
Elastic Net	11.5862	215.8965	14.6411	0.6609	0.9883	0.1362	0.027
Huber regressor	11.4072	231.0311	15.124	0.6408	1.034	0.1288	0.053
Decision tree regressor	7.9212	239.3625	15.2087	0.6267	0.6161	0.1036	0.026
orthogonal matching pursuit	17.0357	442.2286	20.9664	0.3035	1.0698	0.2085	0.025
Lasso least angle regression	19.1239	660.5793	25.5116	−0.01	1.2182	0.1933	0.019
Dummy regressor	19.1239	660.5793	25.5116	−0.01	1.2182	0.1933	0.012
Passive aggressive regressor	27.5446	1303.1353	33.0316	−0.8683	1.2907	0.3727	0.032

Carolina CPD, David NTD (2014) Classification of oranges by maturity, using image processing techniques. In: 2014 III Int Congr Eng Mechatron Autom (CIIMA) 1–5. https://doi.org/10.1109/CIIMA.2014.6983466

Chen T, He T (2017) Xgboost: extreme gradient boosting. R package version 0.4-2 1, no. 4:1–4

Dorogush A, Gulin A, Gusev G et al (2017) Fighting biases with dynamic boosting

Dorogush AV, Ershov V, Gulin A (2018) CatBoost: gradient boosting with categorical features support. ArXiv abs/1810.1

Drucker H (1997) Improving Regressors Using Boosting Techniques. Proc 14th Int Conf Mach Learn

Efron B, Hastie T, Johnstone I, Tibshirani R (2004) Least angle regression. Ann Stat 32:407–499. https://doi.org/10.1214/009053604000000067

Fatima N, Areeb QM, Khan IM, Khan MM (2021) Siamese network-based computer vision approach to detect papaya seed adulteration in black peppercorns. J Food Process Preserv n/a:e16043. https://doi.org/10.1111/jfpp.16043

Fayyazi S, Abbaspour-Fard M, Rohani A, et al (2017) Identification and classification of three iranian rice varieties in mixed bulks using image processing and MLP neural network. Int J Food Eng 13:. https://doi.org/10.1515/ijfe-2016-0121

Table 11 Evaluation of regression models in terms of MAE, MSE, RMSE, R^2 score, RMSLE, MAPE, and TT (Sec) with YCbCr histogram features in the testing phase

Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE	TT (Sec)
Extra trees regressor	2.1769	11.6405	3.3825	0.9812	0.1204	0.0348	1.76
CatBoost regressor	2.5993	13.6621	3.6558	0.9778	0.3543	0.037	91.151
Random forest regressor	2.5417	15.2997	3.8891	0.975	0.1602	0.04	5.662
K neighbors regressor	2.5556	19.0567	4.3392	0.9702	0.0653	0.0419	0.105
Light gradient boosting machine	2.4302	22.7591	4.2808	0.965	0.1167	0.0363	1.577
Gradient boosting regressor	2.9527	20.9606	4.3647	0.9631	0.2412	0.0443	3.944
Extreme gradient boosting	2.8861	24.3909	4.5173	0.9544	0.14	0.0436	4.595
AdaBoost regressor	4.4586	29.8909	5.4586	0.9518	0.0821	0.0721	1.614
Decision tree regressor	2.2847	32.3619	5.4586	0.945	0.117	0.0339	0.126
Bayesian ridge	5.1162	45.6186	6.6611	0.9286	0.4707	0.0726	1.432
Huber regressor	5.4408	49.1674	6.9946	0.9211	0.4873	0.079	0.334
Passive aggressive regressor	6.7361	70.5732	8.3542	0.8865	0.5047	0.0994	0.095
Ridge regression	9.5593	127.3591	11.2576	0.7989	0.8578	0.122	0.033
Orthogonal matching pursuit	6.3151	162.3748	10.5511	0.7577	0.6502	0.0753	0.046
Elastic net	19.0872	660.7388	25.5695	−0.0206	1.2135	0.1919	0.035
Lasso regression	19.09	661.0854	25.5763	−0.0211	1.2136	0.1919	0.025
Lasso least angle regression	19.09	661.0854	25.5763	−0.0211	1.2136	0.1919	0.288
Dummy regressor	19.09	661.0854	25.5763	−0.0211	1.2136	0.1919	0.021
Linear regression	30.6232	80,367.0216	155.2462	−108.6006	0.9654	0.2551	0.366

Table 12 Evaluation of regression models in terms of MAE, MSE, RMSE, R^2 score, RMSLE, MAPE, and TT (Sec) with Lab mean features in the testing phase

Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE	TT (Sec)
Extra trees regressor	5.3196	57.7846	7.5571	0.9073	0.315	0.0812	0.4
K neighbors regressor	5.2613	57.4525	7.5409	0.9058	0.2281	0.0813	0.059
Extreme gradient boosting	5.435	67.2599	8.0783	0.8935	0.2662	0.0839	0.359
Gradient boosting regressor	5.9356	69.4454	8.2571	0.8888	0.3318	0.0911	0.076
Random forest regressor	5.5862	71.3658	8.2912	0.888	0.3401	0.0839	0.453
CatBoost regressor	5.8977	73.035	8.4316	0.8817	0.4703	0.086	1.727
Light gradient boosting machine	6.1363	83.3297	9.0554	0.8664	0.4836	0.0889	0.1
Decision tree regressor	5.6389	116.3671	10.4813	0.8175	0.2599	0.0867	0.013
AdaBoost regressor	9.0258	122.617	11.0237	0.7999	0.2221	0.1352	0.07
Bayesian ridge	9.8702	154.8129	12.3958	0.7526	0.8823	0.1208	0.011
Linear regression	9.8714	154.8083	12.3957	0.7525	0.8819	0.1209	0.297
Least angle regression	9.8714	154.8084	12.3957	0.7525	0.8819	0.1209	0.011
Huber regressor	9.791	164.4072	12.7224	0.7447	0.9358	0.1155	0.021
Passive aggressive regressor	11.2917	282.9009	16.3489	0.5818	1.021	0.1244	0.014
Orthogonal matching pursuit	14.3854	324.9412	17.9768	0.4795	0.9077	0.1852	0.012
Ridge regression	15.6169	388.9784	19.6157	0.3933	1.0806	0.1752	0.011
Elastic net	19.1944	656.5993	25.4955	−0.022	1.1931	0.1963	0.012
Lasso regression	19.2049	657.649	25.5159	−0.0237	1.1934	0.1964	0.012
Lasso least angle regression	19.2049	657.649	25.5159	−0.0237	1.1934	0.1964	0.011
Dummy regressor	19.2049	657.649	25.5159	−0.0237	1.1934	0.1964	0.008

Fix E, Hodges JL (1989) Discriminatory analysis. nonparametric discrimination: consistency properties. Int Stat Rev / Rev Int Stat 57:238–247

Freund Y, Schapire RE (1997) A Decision-Theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55:119–139. https://doi.org/10.1006/jcss.1997.1504

Friedman JH (2001) Greedy function approximation: A grADIENT BOOsting machine. Ann Stat 29:1189–1232. https://doi.org/10.1214/aos/1013203451

Gao W, Zhang X, Yang L, Liu H (2010) An improved Sobel edge detection. In: 2010 3rd Int Conf Comput Sci Inf Technol 67–71. https://doi.org/10.1109/ICCSIT.2010.5563693

Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Mach Learn 63:3–42. https://doi.org/10.1007/s10994-006-6226-1

Goyal K, Kumar P, Verma K (2022) Food Adulteration detection using artificial intelligence: a systematic review. Arch Comput Methods Eng 29:397–426. https://doi.org/10.1007/s11831-021-09600-y

Table 13 Evaluation of regression models in terms of MAE, MSE, RMSE, R^2 score, RMSLE, MAPE, and TT (Sec) with Lab histogram features in the testing phase

Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE	TT (Sec)
Extra trees regressor	2.1022	10.5962	3.2301	0.9837	0.0964	0.0336	1.204
CatBoost regressor	2.594	13.4725	3.6386	0.979	0.3461	0.0377	58.849
Light gradient boosting machine	2.4987	14.8814	3.8278	0.9766	0.0887	0.0392	1.069
K neighbors regressor	2.2123	16.4404	4.0253	0.9749	0.0592	0.0353	0.115
Random forest regressor	2.671	16.9829	4.0848	0.9732	0.1838	0.0418	3.643
Gradient boosting regressor	2.9921	18.8418	4.2946	0.9705	0.1919	0.0464	2.538
AdaBoost regressor	4.5683	31.0152	5.5546	0.9516	0.0835	0.0742	1.209
Extreme gradient boosting	3.1433	34.6469	5.4134	0.9416	0.1599	0.0478	3.869
Decision tree regressor	2.4638	36.6451	5.9664	0.9409	0.0851	0.0382	0.09
Bayesian ridge	4.9238	41.4136	6.3669	0.9356	0.5321	0.0681	1.853
Huber regressor	5.312	47.0429	6.8116	0.9274	0.4398	0.0788	0.516
Passive aggressive regressor	6.3274	63.1105	7.8761	0.9032	0.4952	0.0917	0.125
Ridge regression	8.4369	104.8051	10.2153	0.8374	0.7745	0.1138	0.047
Orthogonal matching pursuit	5.5549	119.6985	9.0873	0.8022	0.5999	0.0711	0.051
Elastic net	19.4039	678.5002	25.941	−0.0124	1.2458	0.1938	0.037
Lasso regression	19.4171	679.9596	25.969	−0.0146	1.2463	0.1938	0.037
Lasso least angle regression	19.4171	679.9596	25.969	−0.0146	1.2463	0.1938	0.332
Dummy regressor	19.4171	679.9596	25.969	−0.0146	1.2463	0.1938	0.023
Linear regression	8.6548	3044.2143	27.5076	−4.708	0.7234	0.0808	0.525

Table 14 Evaluation of regression models in terms of MAE, MSE, RMSE, R^2 score, RMSLE, MAPE, and TT (Sec) with HSV mean features in the testing phase

Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE	TT (Sec)
Extra trees regressor	11.7513	264.632	16.2236	0.61	0.6238	0.163	0.499
Gradient boosting regressor	12.411	271.0578	16.3851	0.6045	0.6745	0.1727	0.082
K neighbors regressor	11.9439	271.3831	16.3626	0.604	0.5253	0.1685	0.059
CatBoost regressor	12.2256	273.0489	16.489	0.5967	0.6779	0.1684	1.819
Random forest regressor	11.8235	275.8455	16.5822	0.5949	0.5758	0.166	0.489
Light gradient boosting machine	12.1891	278.7153	16.6531	0.5914	0.6832	0.1681	0.109
Extreme gradient boosting	12.2018	305.6401	17.413	0.551	0.5537	0.1731	0.369
AdaBoost regressor	14.3081	308.7254	17.5566	0.5508	0.8308	0.1915	0.049
Decision tree regressor	13.2411	442.2593	20.9727	0.3485	0.6077	0.1956	0.013
Bayesian ridge	18.6808	486.5106	22.0275	0.2962	1.094	0.2338	0.013
Linear regression	18.6785	486.4666	22.0271	0.2961	1.0918	0.2344	0.315
Least angle regression	18.6785	486.4666	22.0271	0.2961	1.0918	0.2344	0.012
Huber regressor	18.6083	492.2907	22.1627	0.2867	1.0949	0.2307	0.023
Ridge regression	19.1253	558.9239	23.5791	0.1968	1.2187	0.2106	0.012
Orthogonal matching pursuit	19.4624	566.878	23.7459	0.1823	1.1959	0.2239	0.012
Passive aggressive regressor	19.3373	662.6501	25.6501	0.0521	1.2856	0.2007	0.013
elastic Net	19.7661	704.4997	26.4653	−0.0095	1.2951	0.1933	0.013
Lasso regression	19.7686	704.9996	26.4748	−0.0102	1.2953	0.1932	0.015
Lasso least angle regression	19.7686	704.9996	26.4748	−0.0102	1.2953	0.1932	0.012
Dummy regressor	19.7686	704.9996	26.4748	−0.0102	1.2953	0.1932	0.01

Hans C (2012) Elastic Net regression modeling with the orthant normal prior. J Am Stat Assoc 106:1383–1393. https://doi.org/10.1198/jasa.2011.tm09241

Hastie T, Tibshirani R, Friedman J (2009) Overview of supervised learning. In: Hastie T, Tibshirani R, Friedman J (eds) The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York, New York, NY, pp 9–41

Hendrawan Y, Widyaningtyas S, Fauzy MR (2022) Deep Learning to detect and classify the purity level of Luwak Coffee green beans. Pertanika J Sci Technol 30:1–18

Huang B, Liu J, Jiao J et al (2022) Applications of machine learning in pine nuts classification. Sci Rep 12:8799. https://doi.org/10.1038/s41598-022-12754-9

Huang W, Guo L, Kou W et al (2022) Identification of adulterated milk powder based on convolutional neural network and laser-induced

Table 15 Evaluation of regression models in terms of MAE, MSE, RMSE, R^2 score, RMSLE, MAPE, and TT (Sec) with HSV histogram features in the testing phase

Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE	TT (Sec)
Extra trees regressor	2.2381	14.1152	3.6526	0.9782	0.2944	0.0319	2.495
K neighbors regressor	1.9757	15.423	3.8429	0.9756	0.1737	0.0295	0.138
CatBoost regressor	2.7265	19.5153	4.2678	0.9691	0.4541	0.0357	117.64
Light gradient boosting machine	2.8227	23.4071	4.7183	0.9653	0.4628	0.0379	2.32
Gradient boosting regressor	3.0338	26.4167	4.9095	0.9604	0.3651	0.0419	5.485
Random forest regressor	2.7994	27.5393	5.104	0.9574	0.4176	0.0375	7.854
AdaBoost regressor	4.8365	36.0875	5.9019	0.9409	0.1221	0.075	2.369
Extreme gradient boosting	3.1642	43.6663	6.1759	0.9376	0.3396	0.0431	5.943
Bayesian ridge	5.0717	49.0621	6.8461	0.9276	0.5146	0.0707	2.585
Huber regressor	5.8692	54.2342	7.3361	0.9162	0.5555	0.0821	0.699
Decision tree regressor	2.4923	59.528	6.7441	0.904	0.2214	0.037	0.165
Passive aggressive regressor	7.7442	101.7538	10.0096	0.8455	0.7437	0.1	0.185
Ridge regression	9.6755	157.2631	12.4519	0.7671	0.9823	0.1092	0.052
Elastic net	19.5026	685.0945	25.9764	−0.0074	1.2572	0.1933	0.043
Lasso regression	19.5426	688.1162	26.0344	−0.012	1.2579	0.1937	0.04
Lasso least angle regression	19.5426	688.1162	26.0344	−0.012	1.2579	0.1937	0.031
Dummy regressor	19.5426	688.1162	26.0344	−0.012	1.2579	0.1937	0.022
Orthogonal matching pursuit	8.993	2242.1225	29.4821	−1.6718	0.758	0.0811	0.059
Linear regression	24.7118	5902.9842	60.3507	−8.6954	1.1621	0.25	0.415
Least angle regression	315,576.3561	28,234,355,243,531.117	1,946,846.6327	−4,4190,243,518.9504	5.0148	922.3563	0.296

Table 16 Evaluation of regression models in terms of MAE, MSE, RMSE, R^2 score, RMSLE, MAPE, and TT (Sec) with RGB mean features in the testing phase

Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE	TT (Sec)
K neighbors regressor	6.3821	84.9571	9.1022	0.8738	0.3918	0.0942	0.071
CatBoost regressor	6.8413	94.2953	9.6066	0.8575	0.6046	0.0962	1.9
Extra trees regressor	6.7374	95.1653	9.6534	0.8574	0.5851	0.0939	0.403
Random forest regressor	7.3826	127.053	11.1205	0.8082	0.6268	0.1019	0.608
Light gradient boosting machine	7.9889	134.0316	11.3538	0.7996	0.7119	0.1095	0.115
Gradient boosting regressor	8.2815	131.6208	11.3889	0.7995	0.7447	0.1115	0.081
Extreme gradient boosting	7.4399	143.3294	11.5695	0.7905	0.5888	0.104	0.4
Bayesian ridge	11.6478	209.7266	14.4369	0.6855	0.9803	0.1398	0.021
Linear regression	11.6517	209.7381	14.4378	0.6854	0.9797	0.1399	0.559
Least angle regression	11.6517	209.7382	14.4378	0.6854	0.9797	0.1399	0.024
Decision tree regressor	8.0761	226.0332	14.6747	0.667	0.5547	0.1178	0.034
Huber regressor	11.4172	227.3077	14.9604	0.6661	1.0461	0.1292	0.037
AdaBoost regressor	12.4355	230.0562	15.1221	0.6461	0.6964	0.1706	0.089
Passive aggressive regressor	13.1795	372.2691	19.1274	0.4534	1.1584	0.1371	0.022
Orthogonal matching pursuit	15.4404	430.9389	20.6294	0.36	1.128	0.1742	0.031
Ridge regression	16.5804	451.9826	21.1588	0.3304	1.164	0.1807	0.025
Elastic net	19.547	685.2949	26.0597	−0.0128	1.2555	0.1944	0.038
Lasso regression	19.558	686.2383	26.0778	−0.0142	1.2557	0.1945	0.024
Lasso least angle regression	19.558	686.2383	26.0778	−0.0142	1.2557	0.1945	0.021
Dummy regressor	19.558	686.2383	26.0778	−0.0142	1.2557	0.1945	0.012

breakdown spectroscopy. Microchem J 176:107190. https://doi.org/10.1016/j.microc.2022.107190

Huber PJ, Ronchetti EM (2009) Robust Statistics concomitant scale estimate. Wiley

Hussein WB, Moaty AA, Hussein MA, Becker T (2011) A novel edge detection method with application to the fat content prediction in marbled meat. Pattern Recognit 44:2959–2970. https://doi.org/10.1016/j.patcog.2011.04.028

Jahanbakhshi A, Abbaspour-Gilandeh Y, Heidarbeigi K, Momeny M (2021) Detection of fraud in ginger powder using an automatic sorting system based on image processing technique and deep learning. Comput Biol Med 136:104764. https://doi.org/10.1016/j.compbiomed.2021.104764

Jahanbakhshi A, Abbaspour-Gilandeh Y, Heidarbeigi K, Momeny M (2021) A novel method based on machine vision system and deep

Table 17 Evaluation of regression models in terms of MAE, MSE, RMSE, R^2 score, RMSLE, MAPE, and TT (Sec) with RGB histogram features in the testing phase

Model	MAE	MSE	RMSE	R^2	RMSLE	MAPE	TT (Sec)
Extra trees regressor	2.8334	21.5822	4.5952	0.966	0.3613	0.0408	2.867
CatBoost regressor	3.3243	26.1075	5.0631	0.9594	0.4585	0.0452	141.615
K neighbors regressor	2.6109	27.6179	5.0519	0.9553	0.1168	0.0385	0.104
Gradient boosting regressor	3.6087	31.8616	5.5505	0.9503	0.282	0.0532	6.149
Light gradient boosting machine	3.5264	35.673	5.8468	0.9445	0.4206	0.0502	2.188
Random forest regressor	3.7982	45.9219	6.6726	0.9283	0.487	0.0516	9.468
AdaBoost regressor	5.4596	46.6209	6.7919	0.9248	0.1005	0.0871	2.371
Extreme gradient boosting	4.0055	57.6698	7.2531	0.9109	0.3641	0.0568	5.038
Huber regressor	6.9521	86.103	9.2108	0.8613	0.6324	0.0944	0.481
Bayesian ridge	6.5798	88.2574	9.1698	0.8589	0.5775	0.091	1.686
Decision tree regressor	4.1579	109.1766	9.8545	0.8342	0.3642	0.0612	0.185
Orthogonal matching pursuit	7.9801	269.6692	14.4564	0.5873	0.6927	0.1024	0.044
Passive aggressive regressor	11.3231	296.3618	17.0601	0.5383	1.101	0.1101	0.126
Ridge regression	14.2347	357.281	18.8297	0.4444	1.1069	0.1494	0.039
Lasso regression	19.1187	657.6826	25.5311	−0.0196	1.2036	0.1955	0.034
Elastic net	19.1187	657.6826	25.5311	−0.0196	1.2036	0.1955	0.034
Lasso least angle regression	19.1187	657.6826	25.5311	−0.0196	1.2036	0.1955	0.033
Dummy regressor	19.1187	657.6827	25.5311	−0.0196	1.2036	0.1955	0.035
Linear regression	17.2964	966.975	28.123	−0.4268	0.8908	0.2246	0.376
Least angle regression	6177.5458	174,071,465.0959	8798.6929	−21,1456.7145	4.2741	79.2987	0.23

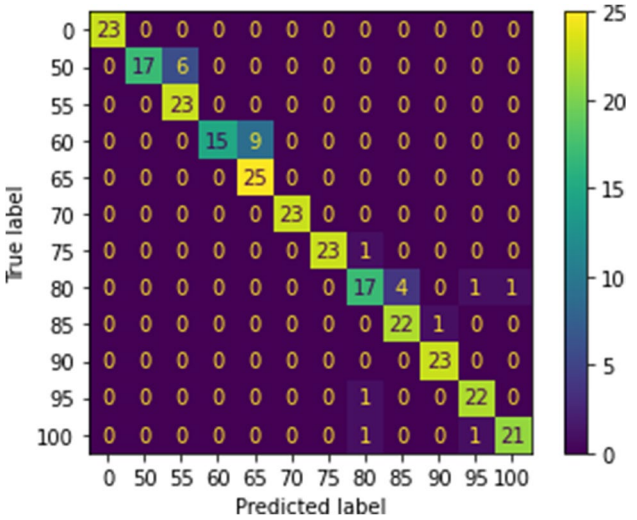

Fig. 14 The confusion matrix is related to the image grading of red chilli powder adulteration with red brick powder

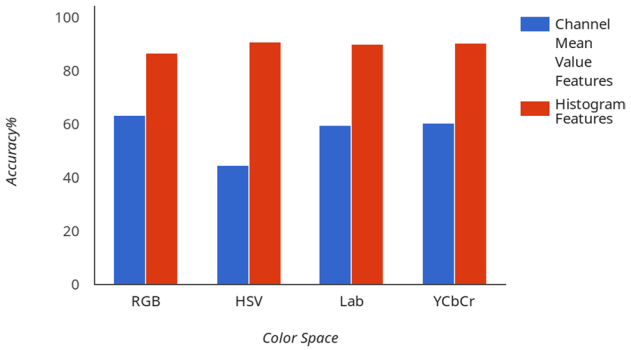

Fig. 15 Performance of classifiers with various color spaces for channel mean value and histogram features in grading red chilli powder

learning to detect fraud in turmeric powder. Comput Biol Med 136:104728. https://doi.org/10.1016/j.compbiomed.2021.104728

Jamaluddin F, Noranizan MA, Mohamad Azman E et al (2022) A review of clean-label approaches to chilli paste processing. Int J Food Sci Technol 57:763–773. https://doi.org/10.1111/ijfs.15293

Januaviani TMA, Gusriani N, Joebaedi K et al (2019) The best model of LASSO with the LARS (least angle regression and shrinkage) Algorithm Using Mallow's Cp

Jin H, Wang Y, Lv B et al (2022) Rapid detection of avocado oil adulteration using low-Field nuclear magnetic resonance. Foods 11. https://doi.org/10.3390/foods11081134

Kamruzzaman M, Makino Y, Oshita S (2016) Rapid and non-destructive detection of chicken adulteration in minced beef using visible near-infrared hyperspectral imaging and machine learning. J Food Eng 170:8–15. https://doi.org/10.1016/j.jfoodeng.2015.08.023

Ke G, Meng Q, Finley T et al (2017) LightGBM: A highly efficient gradient boosting decision tree. In: Guyon I, Luxburg U Von, Bengio S, et al. (eds) Advances in Neural Information Processing Systems. Curran Associates, Inc

Khan N, Ahmed MJ, Shah SZA (2019) Comparative analysis of mineral content and proximate composition from chilli pepper (Capsicum annuum L.) germplasm. Pure Appl Biol 8. https://doi.org/10.19045/bspab.2019.80075

Khosa I, Pasero E (2014) Defect detection in food ingredients using multilayer perceptron neural network. In: 2014 World Symp Comput Appl Res WSCAR 2014

Kleinbaum DG, Klein M (2002) Important special cases of the logistic model. In: Kleinbaum DG, Klein M (eds) Logistic Regression:

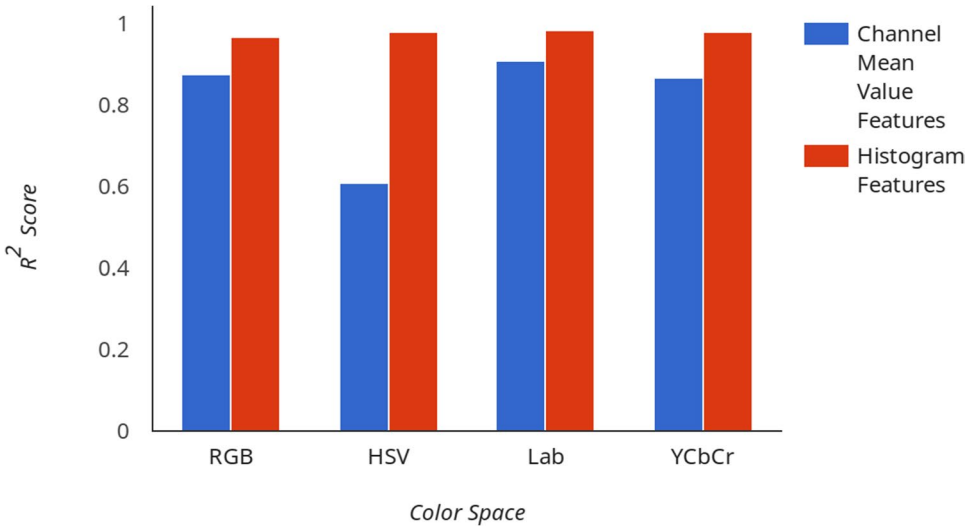

Fig. 16 Performance of regression methods with various color spaces for channel mean value and histogram features in grading red chilli powder

A Self-Learning Text. Springer, New York, New York, NY, pp 39–72

Kobek JA (2017) Vision based model for identification of adulterants in milk. http://su-plus.strathmore.edu/handle/11071/5652

Lakhwani K, Murarka P, Narendra M (2015) Color space transformation for visual enhancement of noisy color image. IET Image Process

Lanjewar MG, Morajkar PP, Parab J (2022) Detection of tartrazine colored rice flour adulteration in turmeric from multi-spectral images on smartphone using convolutional neural network deployed on PaaS cloud. Multimed Tools Appl 81:16537–16562. https://doi.org/10.1007/s11042-022-12392-3

Lapcharoensuk R, Danupattanin K, Kanjanapornprapa C, Inkawee T (2020) Combination of NIR spectroscopy and machine learning for monitoring chili sauce adulterated with ripened papaya. E3S Web Conf 187:4001. https://doi.org/10.1051/e3sconf/202018704001

Li Y, Huang J-B, Ahuja N, Yang M-H (2016) Deep joint image filtering BT - Computer Vision – ECCV 2016. In: Leibe B, Matas J, Sebe N, Welling M (eds). Springer International Publishing, Cham, 154–169

Lim DK, Long NP, Mo C et al (2017) Combination of mass spectrometry-based targeted lipidomics and supervised machine learning algorithms in detecting adulterated admixtures of white rice. Food Res Int 100:814–821. https://doi.org/10.1016/j.foodres.2017.08.006

Ma J, Sun D-W, Qu J-H et al (2016) Applications of computer vision for assessing quality of agri-food products: a review of recent research advances. Crit Rev Food Sci Nutr 56:113–127. https://doi.org/10.1080/10408398.2013.873885

MacKay DJC (1992) Bayesian interpolation. Neural Comput 4:415–447. https://doi.org/10.1162/neco.1992.4.3.415

Manasha S, Janani M (2016) Food Adulteration and its problems (Intentional, Accidental and Natural Food Adulteration). Int J Res Financ Int J Res Financ Mark 6:2231–5985

McDonald GC (2009) Ridge regression. WIREs. Comput Stat 1:93–100. https://doi.org/10.1002/wics.14

Mi S, Zhang X, Wang Y et al (2022) Effect of different genotypes on the fruit volatile profiles, flavonoid composition and antioxidant activities of chilli peppers. Food Chem 374:131751. https://doi.org/10.1016/j.foodchem.2021.131751

Nandi C (2014) Computer Vision Based Mango Fruit Grading System. https://doi.org/10.15242/IIE.E1214004

Nascimento CF, Santos PM, Pereira-Filho ER, Rocha FRP (2017) Recent advances on determination of milk adulterants. Food Chem 221:1232–1244. https://doi.org/10.1016/j.foodchem.2016.11.034

Pedregosa F, Varoquaux G, Gramfort A et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830

Pinheiro Claro Gomes W, Gonçalves L, Barboza da Silva C, Melchert WR (2022) Application of multispectral imaging combined with machine learning models to discriminate special and traditional green coffee. Comput Electron Agric 198:107097. https://doi.org/10.1016/j.compag.2022.107097

Platt J (2000) Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv Large Margin Classif 10

Pourreza A, Pourreza H, Abbaspour-Fard M-H, Sadrnia H (2012) Identification of nine Iranian wheat seed varieties by textural analysis with image processing. Comput Electron Agric 83:102–108. https://doi.org/10.1016/j.compag.2012.02.005

Pradana-López S, Pérez-Calabuig AM, Cancilla JC, Torrecilla JS (2022) Standard photographs convolutionally processed to indirectly detect gluten in chickpea flour. J Food Compos Anal 110:104547. https://doi.org/10.1016/j.jfca.2022.104547

Pratibha N, Hemlata M, Krunali M (2017) Analysis and identification of rice granules using image processing and neural network. International Journal of Electronics and Communication Engineering 10:25–33

Ranstam J, Cook JA (2018) LASSO regression. Br J Surg 105:1348

Rateni G, Dario P, Cavallo F (2017) Smartphone-based food diagnostic technologies: a review. Sensors (Basel) 17:. https://doi.org/10.3390/s17061453

Rong D, Rao X, Ying Y (2017) Computer vision detection of surface defect on oranges by means of a sliding comparison window local segmentation algorithm. Comput Electron Agric 137:59–68. https://doi.org/10.1016/j.compag.2017.02.027

Ropodi AI, Pavlidis DE, Mohareb F et al (2015) Multispectral image analysis approach to detect adulteration of beef and pork in raw meats. Food Res Int 67:12–18. https://doi.org/10.1016/j.foodres.2014.10.032

Sadhukhan T, Chatterjee S, Das RK et al (2019) Efficient Removal of noise from an image using HSV filtering. In: 2019 Global Conf Adv Technol (GCAT), Bangalore, pp 1–4

Salim NOM, Zeebaree SRM, Sadeeq MAM et al (2021) Study for food recognition system using deep learning. J Phys Conf Ser 1963:12014. https://doi.org/10.1088/1742-6596/1963/1/012014

Sarkar T, Mukherjee A, Chatterjee K et al (2022) Edge detection aided geometrical shape analysis of Indian gooseberry (Phyllanthus emblica) for freshness classification. Food Anal Methods. https://doi.org/10.1007/s12161-021-02206-x

Schneider A, Hommel G, Blettner M (2010) Linear regression analysis: part 14 of a series on evaluation of scientific publications. Dtsch Arztebl Int 107:776–782. https://doi.org/10.3238/arztebl.2010.0776

Shafiee S, Polder G, Minaei S et al (2016) Detection of honey adulteration using hyperspectral imaging. IFAC-PapersOnLine 49:311–314. https://doi.org/10.1016/j.ifacol.2016.10.057

Soltani Firouz M, Rashvand M, Omid M (2021) Rapid identification and quantification of sesame oils adulteration using low frequency dielectric spectroscopy combined with chemometrics. LWT 140:110736. https://doi.org/10.1016/j.lwt.2020.110736

Song Y, Liang J, Lu J, Zhao X (2017) An efficient instance selection algorithm for k nearest neighbor regression. Neurocomputing 251:26–34. https://doi.org/10.1016/j.neucom.2017.04.018

Sowmya N, Ponnusamy V (2021) Development of spectroscopic sensor system for an IoT application of adulteration identification on milk using machine learning. IEEE Access 9:53979–53995. https://doi.org/10.1109/ACCESS.2021.3070558

Srinath K, Kiranmayee AH, Bhanot S, Panchariya PC (2022) Detection of palm oil adulteration in sunflower oil using ATR-MIR spectroscopy coupled with chemometric algorithms. Mapan. https://doi.org/10.1007/s12647-022-00558-1

Subashini P (2010) Comparison of filters used for underwater image pre-processing. Int J Comput Sci Netw Secur 10:58–64

Tharwat A (2016) Linear vs. quadratic discriminant analysis classifier: a tutorial. Int J Appl Pattern Recognit 3:145. https://doi.org/10.1504/IJAPR.2016.079050

van der Walt S, Schönberger JL, Nunez-Iglesias J, Boulogne F, Warner JD, Yager N, Gouillart E, Yu T; scikit-image contributors (2014) scikit-image: Image processing in Python. PeerJ 2:e453. https://doi.org/10.7717/peerj.453

VijaykumarVanathiKanagasabapathy VRPTP (2010) Fast and efficient algorithm to remove gaussian noise in digital images. IAENG Int J Comput Sci 37:78–84

Vincent L (1993) Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. IEEE Trans Image Process 2:176–201. https://doi.org/10.1109/83.217222

Zhang Y, Zheng M, Zhu R, Ma R (2022) Detection of adulteration in mutton using digital images in time domain combined with deep learning algorithm. Meat Sci 192:108850. https://doi.org/10.1016/j.meatsci.2022.108850

Zhu J, Rosset S, Zou H, Hastie T (2006) Multi-class AdaBoost. Stat Interface 2:. https://doi.org/10.4310/SII.2009.v2.n3.a8