A Comparative Analysis of Optimizing Classification Techniques on Fruits Images Dataset

Youssef George Fouad   
Faculty of Engineering  
Ain Shams UniversityCairo, Egypt  
email: 19p9824@eng.asu.edu.eg

Kerollos Wageeh Youssef   
Faculty of Engineering  
Ain Shams UniversityCairo, Egypt  
email: 19p3468@eng.asu.edu.eg Jannah Ayman Amir  
Faculty of Engineering  
Ain Shams UniversityCairo, Egypt  
email: 19p1728@eng.asu.edu.eg

*Abstract*— Data mining, also known as data dredging or data archeology, provides several techniques to extract new and interpretable information from given datasets. In this paper, we compare the performance of several classification techniques, which are SVM, RBF, KNN, decision trees on pictures of 3 fruits (pineapple, cocos, and avocado) from fruits-360, an open-source dataset. The KNN classifier has given 97.9%, the highest accuracy score among the tested classifiers. In addition, we compare how principal component analysis (PCA), one of the most famous dimensionality reduction techniques, affects the performance of the classification tasks.

Keywords— Data Mining, Machine Learning, Knowledge Discovery in Database (KDD), Dimensionality Reduction, Principal component analysis (PCA), Support Vector Machine (SVM), Radial Basis Function (RBF), K-Nearest Neighbors (KNN), Decision Trees, Image Processing, Supervised Learning, Classification Techniques, Python.

# INTRODUCTION

Data mining is mainly used to transform raw data, that has no meaning by themselves, into useful, interpretable, and meaningful information. The new mantra in the data world is to gather whatever data one can gather wherever, however, and whenever, where these data points will definitely have value, sooner or later. That is why data is collected as these data can have value for an envisioned target or for an undecided or unknown purpose that might be decided or known later on. For instance, knowledge discovery in database (KDD) gives us methods and techniques to discover useful patterns and information from our raw data. The following figure gives an overview of the basic stages of the KDD process starting from the raw data, to selecting the target data we would work on, before preprocessing and transforming it to start the mining process that leads us to discovering some patterns, models, and useful information. This discovered information shall finally be evaluated and assessed before being used in business or further research use cases.

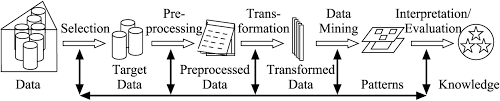


Figure : Data Mining in Action

After selecting target data from our enormous dataset, preprocessing, and transforming it, one of the famous data mining techniques is building learning models. Learning is one of the most important tasks of the different data mining tasks, and it can be either supervised or unsupervised.

In unsupervised learning, training data is unlabeled, and the algorithm only works to divide the given data into clusters of data trying to maximize similarities between points of the same cluster, also known as inter-cluster distance, and maximize dissimilarity between groups of data, also known as intra-cluster distance. Examples of unsupervised learning algorithms are K-means clustering model and Apriori technique.

On the other hand, supervised learning techniques teaches a model by training it on labeled training data, where each instance’s class is known prior to the training phase that helps the model learn how to classify new input data to determine which class that input data belongs to. Examples of supervised learning applications are regression applications and classification applications.

## Classification in machine learning

One of the most widely used applications of supervised learning is classification of input data, predicting its label and classifying each instance into a specific class. Classification techniques can be divided into 2 main types: Eager learners and Lazy learners.

Eager learning techniques first build models from the training dataset before the start of the testing phase where the model predicts the input instance class. Therefore, they tend to take longer time in the training phase and shorter time in the testing phase than those of the lazy models. Examples of the eager learning algorithms are support vector machine (SVM), and decision trees.

On the other hand, lazy learners, also known as instance-based learners, do not build any model prior to testing. The training phase is all about memorizing the training set to search for the nearest neighbor while predicting the label of the input instance. Thus, this type of classifier tends to have a much longer testing time and nearly no training time. The most famous example is K-nearest neighbors classifier (KNN).

# PROBLEM SPECIFICATION

The problem specified to be solved is mainly a classification problem. Image recognition won’t be used instead, every image is represented numerically by a grayscale for each pixel, to facilitate working with classification algorithms, as mentioned before.

## Classification problem

In general, the classification problem is a machine learning problem in which each instance of a given dataset is supposed to be classified into a class or a category.

## Binary classification

Binary classification is to classify a case to be Yes or No. For instance, how to know whether a person is suffering from a certain disease X or not. That is called a binary classification problem, as the person is classified to have such a disease or not without knowing the other disease that he suffers from. Another example would be taking a decision to instantly buy a product from an online portal or wait for couple of months in order to get maximum discount. Binary classification classifies the classes into Yes /No, Good/Bad, High/Low.

## Multinomial classification

Otherwise, the classification problem could be differentiating the data into one of three classes or more, as in this paper. For example, fruit classification, as will be mentioned later, product categorization or malware classification. The multinomial classification could follow one of two approaches, hard assignment, or soft assignment.

Hard assignment is to classify the data in one of the categories or classes. While soft assignment is to classify the data to a class corresponding to the highest probability, in which the data is probable to be classified to every class with certain probability. Thus, if there are 5 classes, for every data to be classified would have 5 probabilities one each class. For instance, a digit classification problem will have 10 classes, digits from 0 to 9, the output of the classification is the probability of the digit for each class, and the digit will be classified to the class with the highest probability.

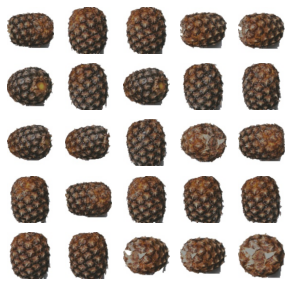


Figure : Samples of Pineapple and Cocos Images

## Fruit classification

Some fruit must be classified as pineapple, cocos, or avocado, only one of them. This classification would have great potential in manufacturing, for example. In a scenario, a factory producing many products from those three fruits, thus the fruits would be separated into three groups, one for every fruit, to allow a specific fruit to be used in such an appropriate product line. There exist more scenarios to benefit from the classification of those three fruits. Therefore, the fruit classification should be as accurate as possible, also not to be time consuming. If the classification is time consuming the generating power of the factory, mentioned in a previous example, would be reduced. Therefore, the chosen classifier with some specific hyperparameters should be giving the best accuracy in the shortest time.

# DATASET

Having a dataset of high-quality is essential for building a good classification model, also known as a classifier, where this dataset is usually split into a training set, a testing set, and a validation set for the purpose of building the different predictive models we are writing this research paper about. So, working on this paper, we used fruit-360 dataset, which is a good example of clean, intensive, and labeled, where each picture is labeled with its class name, dataset that provides a good amount of data to the models being built without letting them reach the undesired, overfitted state in which the model’s predictive accuracy percentage drops.

## Fruit-360 Dataset

The chosen dataset consists of tens of thousands of pictures, 80,653 images so far to be specific, as it gets updated every now and then by the author where new classes or new pictures for existing classes are added to the dataset that can be downloaded from the address pointed by reference [1]. The reader is encouraged to download and check the latest version from the above indicated version.

These pictures are of 118 different popular fruits and vegetables taken hundreds of colored shots for each of them from different angles on a white background. In addition, the pictures of every class, fruit, or vegetable, are split into training and test sets for training and testing purposes that will be discussed later in this paper. Each picture is an RGB picture where each of the red, green, blue color channels consists of 100x100 pixels, summing up to 100x100x3 values per individual picture.

Working on this comparative analytical paper, we decided to work on images of 3 fruits only, pineapple, cocos, and avocado. Our choice of these three fruits in specific gives us the opportunity to test the classification techniques in different ways as pineapples look like cocos in terms of color grades and cocos look like avocados in terms of the oval-like geometrical shape. 25 samples of pineapple images and 25 samples of cocos images in addition to 25 samples of avocado images are provided below in figures 2 and 3 showing how diverse the angles of the dataset pictures are.



Figure : Samples of Avocado Images

# APPROACH

The approach followed for our problem follows the common convention and framework when working on any data mining tasks, starting from collecting, and importing data to explore it before undergoing preprocessing to start the main mining task to extract useful information that we would later visualize in charts and graphs for presentation purposes.

## Data Importing and Integration

Starting with selecting some fruits to work on their images, pineapple, cocos, and avocado images were chosen out of the 118 fruits and vegetables available in the fruits-360 dataset as they have a lot of similarities so the classification task will not be so easy for the built model.

The training and the testing sets for each of the three fruits are imported as RBG images, before being combined in only one training set consisting of labeled training samples of all the three fruits and one testing set consisting of all imported testing samples of all three fruits’ image classes.

Every image is represented by 100 100 pixels, and each pixel is represented by 3 color values, one for the red channel, one for the green color channel and one for the blue color channel, as well to form an RGB image.

## Data Exploration

After importing the data, we shall explore our data to know more about it. Visualizations are made over the datasets to facilitate the understanding of how large the imported sets are. A pie chart and a bar chart are drawn to help us get a quick glance of the data looks like.

The pie chart shown in figure 4 illustrates the ratio percentage and count of the total of each of the training and testing datasets. As the graph suggests, there are 1407 training images and 475 testing samples.

Chart, pie chart

Description automatically generated

Figure : Total training to testing samples ratio and count

Digging deeper into the details of the previous pie chart, the bar chart in figure 5 breaks down the previously mentioned numbers into the detailed distribution of fruits among training and testing sets.

Chart, bar chart

Description automatically generated

Figure : Breakdown of fruits distribution among testing and training sets

## Data Preprocessing

Manipulation of data before mining tasks can be essential and can have a great impact on the processing speed and power needed to perform our mining task or even only for the algorithm to be able to deal with the given data; therefore, our input images had to undergo several types of data transformation as follows.

### Data discretization

The given input data is images that needs to be discretized into numbers before being processed or used in our classification models. Each input image is first discretized into a (100 x 100 x 3) array, where each value varies from 0 to 255.

### Data compression

After discretizing each image, we found ourselves dealing with a complex data form, so we need to reduce its complexity by compressing the RGB picture (100 x 100 x 3), where each pixel is represented by 3 values to grayscale ones (100 x 100), where each pixel is represented by a single value. This can be done using python library OpenCV that uses formula 1 shown below to calculate a single value from the three values of each pixel. In addition, one sample of each fruit is shown below in figure 6, once in RGB and the other after conversion to grayscale.

Equation : RGB to Grayscale Conversion

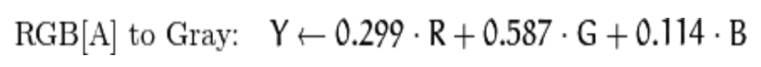






Figure : RGB to Grayscale Conversion Samples

### Data Linearization

All the mining tasks we are going to perform require that each instance of the data to be in linear form, also known as 1-dimension array; therefore, all instances are flattened from their 2-dimensions form (100 x 100) into 1-dimension array of (10,000) values.

### Dimensionality Reduction

After all the preprocessing applied to our dataset, we have reached a new list of 1-dimension arrays where each array is 10,000 features long which is still a huge number of features for processing; therefore, the last step of preprocessing would be applying principal component analysis (PCA) on our dataset to reduce the number of features to n-features.

Principal components can preserve enough information in the image for classification purposes. Moreover, PCA helps save time and processing power in the training phase of the eager learners. Figures 7, 8, 9 compares the raw picture with the same picture after gray scaling it and how the number of principal components affect the picture. Although reducing the instance features from 10,000 to only 2 looks like it completely damaged it but those 2 components have enough information that can result in acceptable accuracy results.



Figure : Sample Pineapple Image, Raw vs Grayscale vs PCA

A picture containing text, different

Description automatically generated

Figure : Sample Cocos Image, Raw vs Grayscale vs PCA

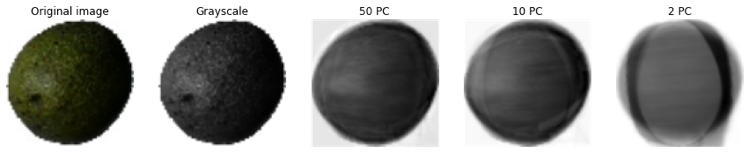


Figure : Sample Avocado Image, Raw vs Grayscale vs PCA

Moreover, reducing the dimensions of the given data can help in visualizing it; data of more than 3 dimensions cannot be visualized. For instance, figure 10 shows the distribution of our training dataset when reduced by PCA to 2 features only.

A picture containing scatter chart

Description automatically generated

Figure : Training Set Plotted in Two Dimensions

## Classification Algorithms

After data preprocessing is done, each image is represented as a one-dimensional numeric array, where each value ranges from 0 to 255. In this section, we will train and test several classification models, which are KNN, SVM, SVM with RBF kernel, and decision tree using the training dataset and then testing it using the testing dataset that we have input in the beginning of our process and underwent the same preprocessing steps as the training dataset.

Every classification algorithm is discussed in two manners, once with reduced form of data that underwent dimensionality reduction using principal components analysis (PCA), discussed above, and one more time without dimensionality reduced data.

Moreover, K-fold cross validation technique is used to minimize the error margin of the predictive classification model built throughout the training phase. As figure 10 suggests, the training set is split into five folds, where we chose to use K = 5, and on each of the five iterations, one-fold is used as validation set, while the other four are used as training set.

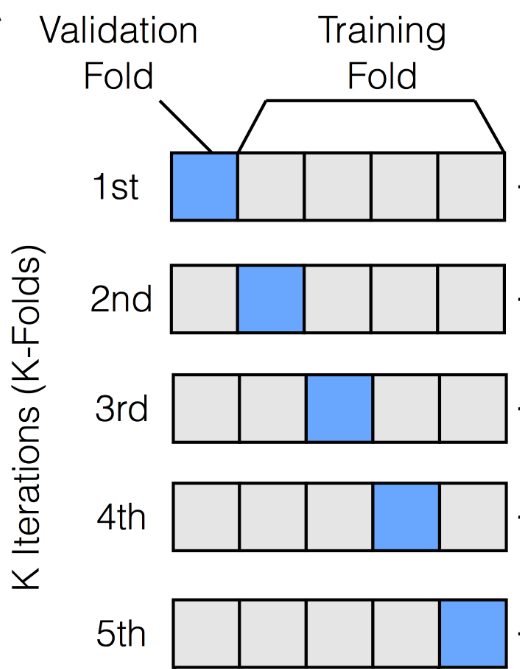
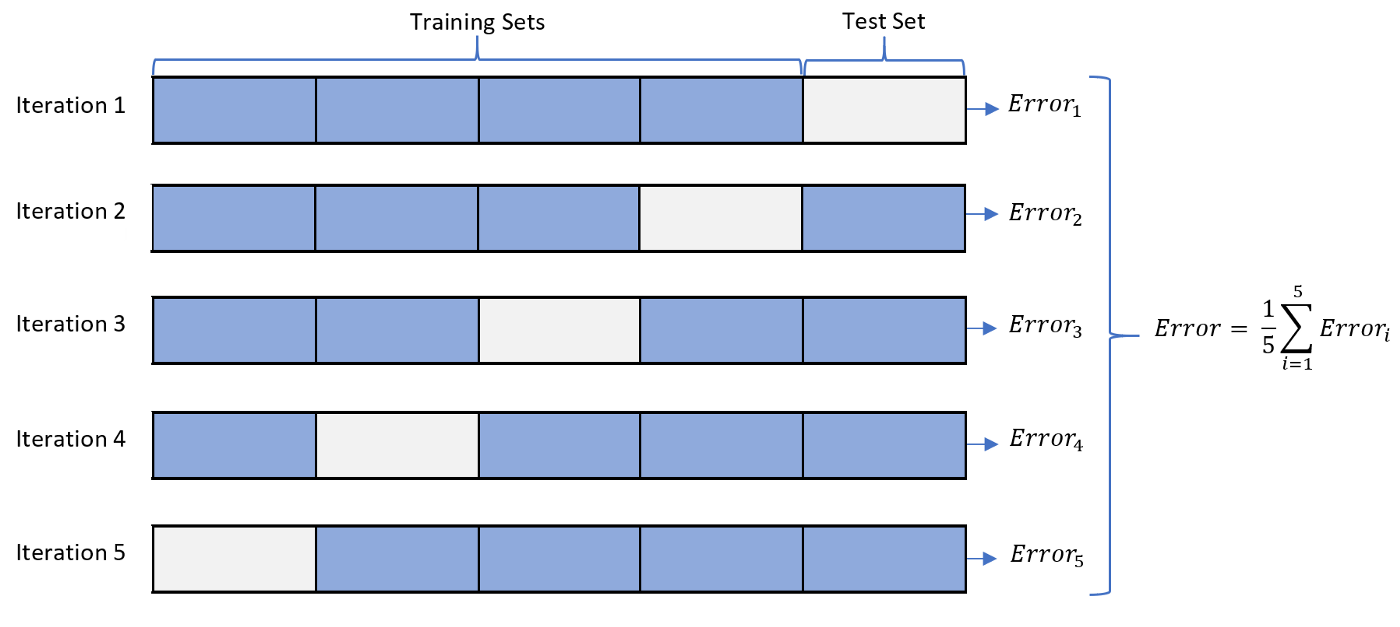


Figure : 5-Fold Cross Validation

Looking at equation 2, we find out how this technique helps in minimizing the error margin of the built model. Where is the error of the i-th iteration and the total error of the model is the average of the five error percentages calculated at the five iterations.

Equation : 5-fold Cross Validation Error



### Linear Support Vector Machine

SVM chooses the support vectors, which are the appropriate data points which shall maximize the margin (distance) between classes and reduce the percentage of the misclassified data, in order to find a linear hyperplane to be the decision surface.

The decision surface is a hyperdimensional surface which discriminates between the instances of classes, it could be of zero-dimensions (a point), 1-dimension (a line), 2-dimensions (a plane), etc. To generalize, for an input dataset of n features, the SVM decision plane is of n - 1 features.

We trained a classification linear SVM model using the original training data of 10,000 feature space once and tested it against the test set to check the training time and testing accuracy of the built model. This procedure was repeated several times with training and testing sets that have been dimensionally reduced using PCA with different range of principal components count, 2 to 100 principal components to check the number of components that resulted in the best possible classification accuracy. Note that in both manners, we used the previously mentioned 5-fold cross validation technique to get better results.

### SVM with Radial Basis Function Kernel

The support vector machine algorithm can be tweaked to separate linearly inseparable data by using a kernel function that takes low dimension feature space and converts it to high space where the data can be separable. One of the widely used kernel functions is the radial basis function which tries to perform exact function interpolation.

In our application, we trained an SVM model with RBF kernel in the same manner we did with the linear SVM, training with 5-fold cross validation and testing it and calculating the classification accuracy once on the 10,000 features data and once with the dimensionally reduced data for different number of principal components.

### K-Nearest Neighbor Classifier

Another classification algorithm that was trained and tested during the research is KNN that classifies a new point according to its position relative to the K nearest points. During the training phase, the model only memorizes the training dataset with no calculations. All calculations take place during testing where the distances between the test input instance and all other training points are calculated, and the decision is taken based on the K nearest ones.

During our research, we trained and tested the trained model once using the linearized grayscale data for different K values; K = 1 to K = 150. In addition, the same approach was followed for the dimensionally reduced data using PCA, using different number of principal components from 2 to 10 and a range of K values from 1 to 100. Lastly, we checked which number of components combined with which value of K resulted in the highest classification accuracy.

### Decision Trees

The last of the four classification techniques used was the decision trees where a tree is built during the training phase based on the entropy measure. The tree internal nodes represent features from the dataset, the branches represent the decision rules on which a decision is taken, and the tree leaf nodes represent the outcomes, classes or labels. The internal nodes basically ask a yes/ no question on which the answer determines how the tree is further split. Figure 12 illustrated decision trees in action.

Similar to the approach we followed for the other classifiers, we built a decision tree with the training set before testing it with the test set and checked the classification accuracy in addition to the number of levels of the generated tree. The approach was repeated for the reduced version of the training and testing sets for a range of number of principal components from 1 principal component to 100 principal components to check which number resulted in the highest accuracy at what number of tree levels, also known as tree depth.

Diagram

Description automatically generated

Figure : Decision Tree Classifier

# EVALUATION CRITERIA

Evaluating the experiments, we would consider two main factors, accuracy in the first place, in addition to time that should be considered, too. Robert L. Read stated that “Science is about truth, engineering is about compromise.”; therefore, one factor cannot be considered without the other.

On one hand, we consider the classification accuracy the main deciding factor of whether the algorithm is good enough to be accepted and used in classifying or predicting the label of more samples or not, but if the highest accuracy is obtained at a very long time hence a very high computational power, then accuracy can be compromised to obtain fairly good results at a much shorter time.

Furthermore, time factor can either be time taken during the training phase or that taken in the testing phase. Most applications are time critical as they require very short testing time but can withstand a long training time prior to testing, which eager classifiers can provide. Lazy classifiers, on the other hand, take no time in the training phase but take longer time in the testing phase as it computes all its calculations with each instance. Last but not least, some applications cannot compromise any testing accuracy for shorter time; therefore, the longer time solution is taken if it provides a higher classification accuracy.

In our research, some accuracy compromises was acceptable as the time difference was huge and sometimes, it was the other way round as we had to go for the higher time solution as the accuracy was much better.

# RESULTS AND ANALYSIS

After the approach was fully applied starting from importing and preprocessing until training and testing all the mentioned classification algorithms in chapter IV, results were collected. In this section, the results of these classification algorithms will be discussed as well as the analysis of these results in order to determine the best classifier in terms of time and testing accuracy for our fruit classification use case.

## Linear SVM

In linear SVM, the decision hyperplane will be, as if the trained data is of n dimensions, the decision hyperplane will be of n-1 dimensions.

### Linear SVM with PCA

PCA is done on the data with only 2 principal components, and it took 63 minutes and 45 seconds to finish training and testing, producing 68% accuracy. The resulted linear decision surface form it is plotted, to draw some results.

Scatter chart

Description automatically generated with medium confidence

Figure : Linear SVM Decision Plane in 2D

That accuracy resulted with that huge time is because the data is not linearly separable when projected to 2 dimensions, as shown in figure 13. Thus, the algorithm takes all that time trying to find 2 lines separating the 3 classes with maximum margin and lowest misclassification error. Therefore, drawing a graph representing the relationship between the applied principal components, from 1 to 100 principal components, and accuracy would be very time consuming and would not give high accuracy as well. Thus, a graph is drawn using principal components ranging from 10 to 100 along with the corresponding accuracy percentages, as shown in figure 14. Resulted in 56.6% maximum accuracy at 42 principal components in 7 minutes and 47 seconds, which is way shorter than that with 2 principal components.

Graphical user interface, chart, table, line chart

Description automatically generated

Figure : Linear SVM accuracies for different count of PCs

Consequently, the linear SVM would consume more time in lower dimensions than the higher dimensions, as in the higher dimensions the dataset would be linearly separable. However, that does not mean that the more separable the dataset, the more accuracy would be resulted. Moreover, when linear SVM is trained with PCA of 42 principal components, 56.6% acuuracy is achieved in 1.1 seconds. That reinforces the point of view stated, as the higher dimensions do not guarantee higher classification accuracy using linear SVM.

### Linear SVM without PCA

When applying the linear SVM on the entire training dataset without PCA, 10,000 dimensions, it resulted in accuracy of 96% consuming only 8.9 seconds to train and test the classification algorithm.

## SVM with RBF kernel

The SVM with RBF kernel is totally different from the linear SVM. In linear SVM, the decision hyperplane is linear as clarified previously, while changing the kernel of the SVM to be RBF, the decision hyperplane would be described with centers and standard deviations; one for every curve being drawn to differentiate between classes.

### SVM with RBF kernel with PCA

Firstly, applying PCA on the training dataset with 2 principal components results in 88% accuracy in 7.3 seconds. Although, SVM with RBF kernel finishes faster than the linear SVM, it is less accurate. The decision hyperplane concluded from SVM with RBF kernel using 2 principal components is shown below in figure 15.

A picture containing chart

Description automatically generated

Figure : SVM with RBF Kernel Decision Plane in 2D

As it could be seen, the decision hyperplane of the SVM with RBF kernel fits on the training dataset more than the linear SVM using 2 principal components, achieving higher accuracy.

Secondly, try to achieve the highest accuracy by trying principal components count ranging from 1 to 100 principal components, and the accuracy is calculated for every count of principal components, then plotted into a graph, as shown below in figure 16.

The maximum accuracy of 93.7% is reached at 3 principal components taking a total training and testing time of 0.6 seconds. Approximately after using 40 principal components, the accuracy tends to be constant at about 77%, that is due to the overfitting of the algorithm on the training dataset.

A picture containing text, shoji, crossword puzzle

Description automatically generated

Figure : RBF Accuracy for different count of PCs

### SVM with RBF kernel without PCA

Training the SVM classifying algorithm on the original 10,000 features training dataset resulted in 96.8% accuracy consuming a total of 17 seconds for training and testing.

Seemingly, if the time factor is critical it is preferred to use PCA with 3 principal components which would results in 93.7%, otherwise don’t use PCA at all and achieve 96.8% accuracy. As the difference between the two accuracies is not that much, it is not desired to consume 2833% the time (16.4 seconds), to increase the accuracy 3.1%

## KNN

KNN is a lazy classification algorithm which does not consume any time in training the algorithm and the training phase is only limited to memorizing the training dataset. Instead, all the time consumed is in testing phase, in which the distance from the data to be classified to each of the training point of the training dataset and choose the closest K points and decide the class of the data.

### KNN with PCA

In KNN with data dimensionally reduced using PCA, there are more than one hyperparameter to be used in order to train and test the algorithm, which are the nearest neighbors and the number of principal components. Nearest neighbors tested for a range of 1 to 100 nearest neighbors, while number of principal components tested ranged from 2 to 11 PCs. The maximum accuracy achieved was 93.05% with 9 principal components, and 13 nearest neighbors which only consumed 0.5 seconds to train and test the dataset.

It was decided to construct a 2D decision surface using KNN algorithm with 13 nearest neighbors and 2 principal components instead of 9 principal components, to visualize the decision boundary between the classes, as shown in figure 17. As mentioned before, there exist a region where the three classes are overlapping, hence the KNN with 2 principal components and 13 nearest neighbors is predicted not to give high accuracy. Indeed, it achieves 89% accuracy, which would be better if 13 principal components are used, as concluded before.

A picture containing chart

Description automatically generated

Figure :13-NN and 2-PC

### KNN without PCA

The KNN algorithm is trained and tested by the original dataset of 10,000 dimensions. Only the number of nearest neighbors is changed throughout the iterations, and the corresponding accuracy of each iteration are collected and represented on a graph, as shown in figure 18.

Seemingly, as the number of nearest neighbors increases the accuracy decreases. This is due to overfitting of the algorithm, as the algorithm does not learn from the training dataset, but it memorizes the training samples and fits too closely to it; instead of finding a general solution.

Chart, line chart

Description automatically generated

Figure : KNN Accuracy across different K

A range of the nearest neighbors is tested from 1 to 150 nearest neighbors to train and test the classifier. The achieved maximum accuracy was 97.9% with 3 nearest neighbors considered, consuming only 0.4 seconds.

Consequently, when applying KNN it is preferred not to use PCA in which using PCA achieves lower accuracy and consumes more time than without using PCA. KNN without PCA achieves 97.9% accuracy in 0.4 seconds, while KNN with PCA achieves 93% in 0.5 seconds.

To sum up, in KNN algorithm there is no need for using more exhaustive power whether to increase the number of nearest neighbors or increase the number of principal components. Better results are reached at smaller hyperparameters.

## Decision Tree

The Decision Tree is one of the algorithms which has low bias but high variance, as any small change in the input would change the output much. In general, its accuracy is not constant every time the algorithm is applied, and it leads to overfitting. It means there is a trade-off between predictive accuracy and generalization of pattern outside training data.

In applying the decision tree upon the training and testing datasets, the max depth hyperparameter is left untouched and the algorithm stops when nodes are expanded until all the leaves are pure nodes. The output of the algorithm is recorded to discuss its behavior on the max depth.

### DT with PCA

PCA is applied on the training dataset before it is passed to the algorithm to learn from it. The algorithm was applied to dataset with principal components from 1 to 100 principal components. For every count of principal components, the accuracy of the algorithm is collected to form a graph representing each count of principal components across the accuracy corresponding to it, as shown in figure 19. The figure illustrates that using a greater number of principal components does not increase accuracy, but it dramatically decreased. That is due to overfitting.

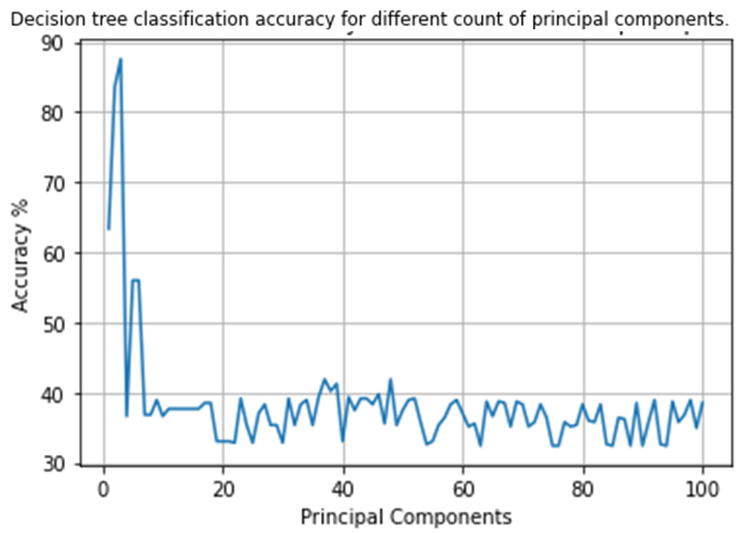


Figure : DT Accuracy Across Different Number of PCs

The maximum accuracy of 88.8% was reached at 3 principal components and a tree of 22 levels depth. Using 3 principal components the Decision tree algorithm consumes only 0.3 seconds to perform the training and testing phases.

### DT without PCA

When the Decision tree is applied on the training and testing datasets, variant of accuracies is achieved every time the algorithm is run, due to some reasons mentioned previously. The maximum accuracy reached is 97.5% with maximum depth of 7 levels consuming 1.7 seconds.

Consequently, the accuracy difference is huge on using PCA or not. Regardless of the increasing in time from using PCA to not using PCA, but the change in accuracy is reasonable. Using PCA, all the time consumed is on the depth of the decision tree, 22 levels, while without using PCA the time consumed is due to the large number of features, 10,000 feature, and the depth of the decision tree is only 7 levels. Therefore, not using PCA is preferred more than using PCA, as very high accuracy is predicted to be achieved.

## All Algorithms

In this part of results and analysis section, a comparison is made between the algorithms applied with PCA and without PCA, in terms of time and accuracy, as shown in table 1.

Table : Classifiers Accuracy and Time Consumed

Table

Description automatically generated with medium confidence

In Linear SVM, it is preferred not to use PCA, due to many reasons.

* Without using PCA, the time consumed is smaller to train and test the algorithm and achieves higher accuracy.
* At lower dimensions the feature space is not linearly separable, even at higher than 10 dimensions the feature space would be separable but with small misclassification error.

In SVM with RBF kernel, it is preferred to use PCA; as it is not about whether the feature space is separable or not, the RBF doesn’t create linear decision hyperplane, it is because

* Without using PCA the algorithm takes a lot of time in training with 10,000 dimensions, while using PCA the algorithm iterates over 3 dimensions only.
* The accuracy difference between using PCA and not using PCA, is not the huge difference, 3.1%, to consume that much time.

In KNN, all the time consumed is for testing no time is used during the training phase. It is preferred not to use PCA, as only more 0.1 second increases the accuracy 4.2%.

In Decision Tree, it is preferred not to use PCA; as the decision tree consumes more time in the training phase and the testing phase is so fast. Although the time consumed without using PCA is more than that with using PCA, the testing time of not using PCA is less than that of using PCA, because when not using PCA the resulted decision tree is of 7 levels only while that of using PCA is 22 levels.

# CONCLUSION

To conclude, pineapple, cocos, and avocado were chosen to be studied from fruits-360 dataset. A new dataset was constructed using the tree fruits. Each pixel of every image of the new dataset is converted into three values, RGB, then each pixel is converted to grayscale and linearized to be of 10,000 dimensions. Dimensionality reduction was applied using PCA technique. The dataset with PCA and without PCA was given for 4 classifiers, linear SVM, SVM with RBF kernel, KNN and Decision tree, to test their accuracy. Some results of the 4 classifiers were gathered and shown in figure 20.

Chart, bar chart

Description automatically generated

Figure : Best Achievable Classification Accuracy

If it is decided not to use PCA before training the algorithms, this would be the order of preference of the classification algorithms: KNN, DT, RBF, SVM, with KNN to be the most appropriate algorithm. Otherwise, the order would be RBF, KNN, DT, SVM, with RBF is the most prioritized algorithm over the others.

In case of critical testing time Decision tree without PCA is preferred with accuracy 97.5%, else the KNN without PCA takes precedence over the other classification algorithms achieving 97.9% accuracy.

##### ACKNOWLEDGMENT

We sincerely thank Dr. Nourhan Mohamed Zayed (Ain Shams University) and Eng. Mahmoud Soheil (Ain Shams University) for their guidance, support, and assistance in understanding essential data mining basic techniques and applications.

##### REFERENCES

1. Horea Muresan, [Mihai Oltean](https://mihaioltean.github.io/), [Fruit recognition from images using deep learning](https://www.researchgate.net/publication/321475443_Fruit_recognition_from_images_using_deep_learning), Acta Univ. Sapientiae, Informatica Vol. 10, Issue 1, pp. 26-42, 2018.
2. Y Mihai Oltean. (2018 February). Fruit-360, [2020.05.18.0]. Retrieved 2022.12.04 from kaggle.com/datasets/moltean/fruits