Machine classification on Fruit 360 and the MNIST Dataset

Rachel Conforti
ITCS-5156 Spring 2022 - Lee

February 28, 2022

Abstract

Machine classification is integrated into a lot of modern-day activities such as labeling emails as spam or not spam, facial recognition, or even assisting in self-driving cars. With this increasing need for image classification, a lot of testing needs to be done to see where different classifiers excel or don't excel. Within this paper, I explore how a dataset color scheme can affect the outcome on various models and compare results to the paper I modeled my project after.

1 Introduction

1.1 Problem Statement

My project [1] was inspired by Handwritten Digits Identification Using MNIST Database Via Machine Learning Models [2]. This paper took in the MNIST dataset and using specific preprocessing methods compared how the dataset would perform on five different models: Support Vector Machine, Decision Tree, Naïve Bayes, K-Nearest Neighbor, and Random Forest. The preprocessing involved separating their dataset in two ways. The first is the grayscale and the second one is the binary grayscale. My project mimicked this preprocessing on the Fruits 360 [5] dataset from Kaggle and compared the results.

1.2 Motivation and Challenges

The motivation for this project was to learn more about machine learning classification. I was interested in this topic because of the increasingly more popular uses of machine learning classification. Machine classification combined with other machine learning areas is creating incredibly complex and useful technology such as self-driving cars, facial expression classification, or medical uses such as identifying tumors. A common one we all use but may not be aware of is our emails automatically label emails as spam or not spam. This is a simple classification of passing an email through the model and determining the likelihood of it being spam or not. If it is spam, it will automatically go into the spam folder without you ever having to interact with the email. The idea of machine classification simplifying and improving people's lives is why I wish to learn more about it. There are common challenges within machine classification. A few that I faced during this project are dirty datasets, underfitting models, overfitting models, and long implementation time. Quite a few of the datasets I looked at for this project were poorly labeled, had a very small number of images to use, or were honestly beyond my knowledge of how to clean up the data. I will discuss below how I overcame these challenges.

1.3 Overview and Approach

The approach in this paper was I first experimented with the MNIST dataset. First by pulling in the data and ran the five models to give me an idea of how close I was to the paper's results and to give myself a baseline. MNIST is a benchmark dataset used in paper I modeled my project after. I chose the Fruit 360 dataset [5] found on Kaggle and tried to mimic the pre-processing method on this data set and run it against the five models and compare my results.

2 Backgrounds

2.1 Related Work 1 - Classification of Cape Gooseberry Fruit According to its Level of Ripeness Using Machine Learning Techniques and Different Color Spaces [4]

In 2009, they used machine learning on to classify seven stages of a cape gooseberry. Within the paper, they used three-color spaces RGB, HSV and L*a*b* against four different models: artificial neural networks, support vector machines (SVMs), decision trees, and K-nearest neighbor algorithms.

An interesting aspect of this paper is that they curated their own photos for this classification paper. They used Cape Gooseberries from Peru and using a conveyor belt and a VGA webcam they came up with seven different stages of fruit. From their seven stages of fruit, used a more advanced preprocessing method due to them taking their photos. This included image acquisition of storing pictures based on specific classes, image enhancement where they used Gaussian filters to smooth the photos, and then did segmentation to convert the images into grayscale images and using MATLAB to convert these into HSV and L*a*b* color spaces.

Using this they ran these datasets through their four models of artificial neural networks, support vector machines (SVMs), decision trees, and K-nearest neighbor algorithms to achieve the results below:

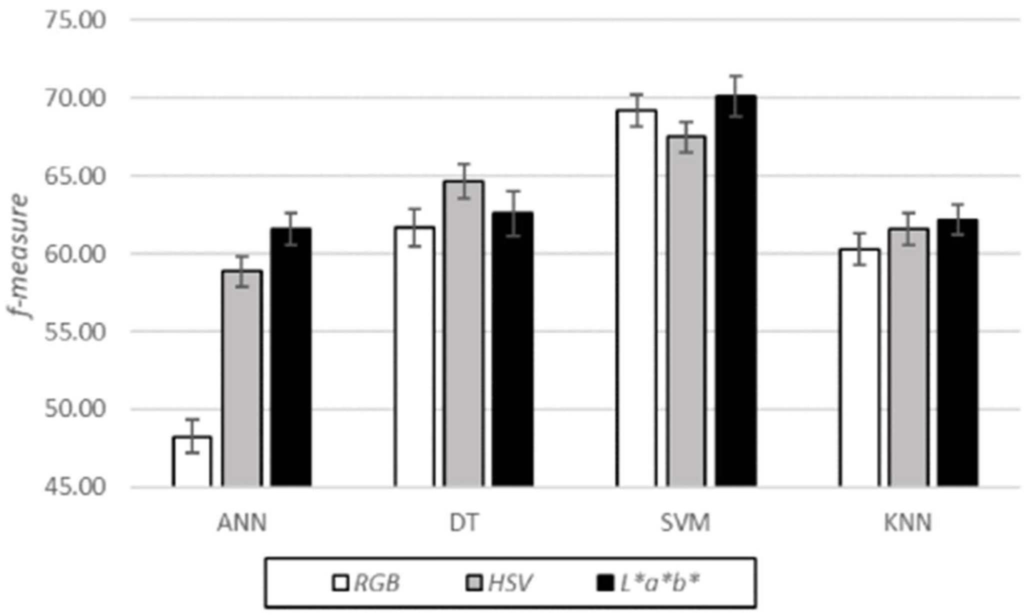


Figure 1: This is the results of the 3-color spaces and how they scored on each model [4]

2.2 Related Work 2 - Multi-Featured and Fuzzy-Filtered Machine Learning Model for Face Expression Classification [3]

In 2020 Kapil Juneja and Chhavi Rana used machine classification for facial expression classification [3]. They used photos and textural information to identify facial expressions of humans by dividing the face into smaller blocks and creating content, structure, and texture-sensitive features for each individual block.

They first apply a Gabor filter to their dataset to assist them with figuring out the expressions in the photo which can be seen below:

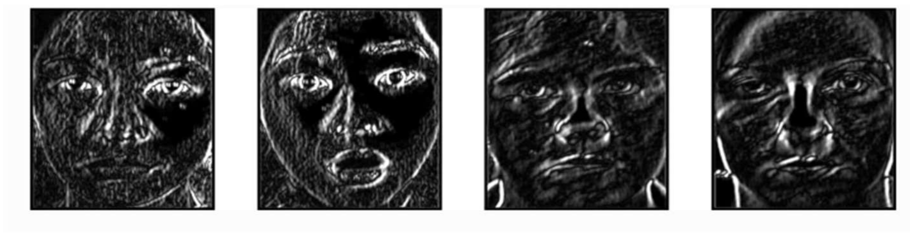


Figure 2: Level one feature extraction images [3]

This is considered the level 1 feature that is doing feature extraction. Their level two feature is feature quantification where the photos are broken down into smaller rectangular blocks that are used to extract ten textures and quantitative features.

The next step is to take the photos from this stage and create a fuzzy-based feature set filtration that aims to reduce the feature set and allow the SVM classifier to better classify the expression class. Below is results of their testing and out from the SVM of the facial expressions in two different sets, the JAFFE dataset and the CK + Dataset:

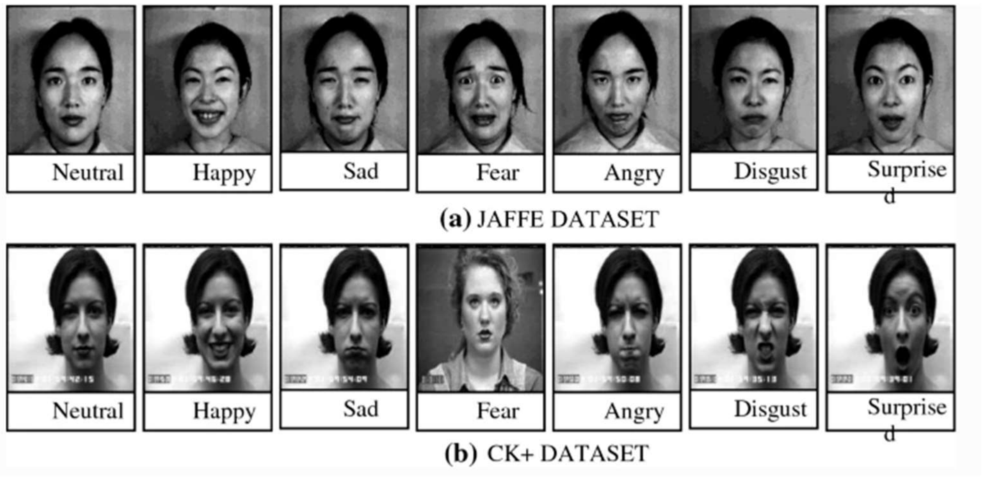


Figure 3: Results of using two machine classification to identify facial expressions [3]

2.3 Relation to my project

The paper about classification of cape gooseberry fruit was a very interesting project. I was impressed that they took their own photos of the fruit and the thoroughness of their research. This is in relation to my approach because of how they used machine classification to identify fruit stages in different color spaces theirs being RBG, HSV, L*a*b* and mine was grayscale and binary grayscale. They also compared it to multiple models and compared their results as seen above. This showed me how many opportunities there are in just fruit related machine classification.

The paper multi-featured and fuzzy-filtered machine learning model for face expression classification is similar to my project because a large amount of their work comes from the preprocessing of their data. They broke their dataset into a two-level feature set, created the context for each segment of their blocks, applied fuzzy composite filters to reduce noise and improve relevancy, and passed it into their model. Similar to what I did which was create two different datasets, pulled features from the dataset and applied image manipulation to improve the accuracy of my models.

3 Method

3.1 Framework

The method I decided to use for the project was to take it in the fruit 360 dataset [5] using the Type attribute are the following five possible values: fruit, vegetable, nut, seed, or other. Taking these five features and compare these using the two datasets created classifiers and then print out the evaluation of the models in multiple ways such as classification reports, bar graphs, and confusion matrices.

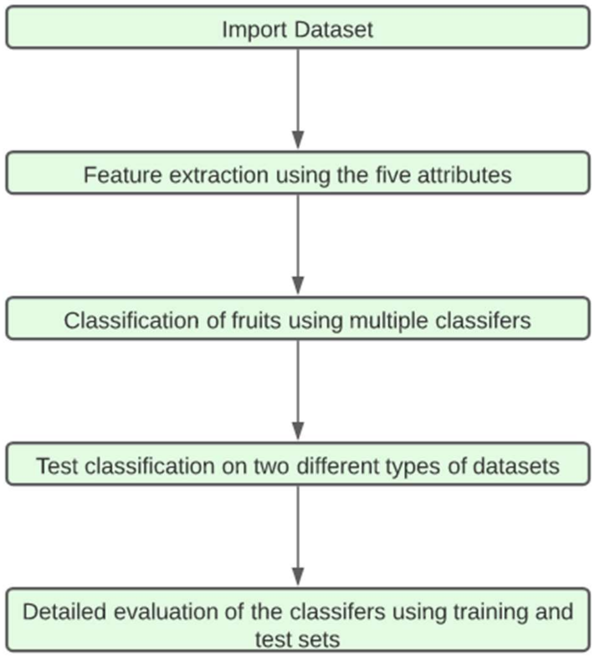

Import Dataset
Feature extraction using the five attributes
Classification of fruits using multiple classifers
Test classification on two different types of datasets
Detailed evaluation of the classifers using training and test sets

Figure 4: Experiment architecture

3.2 Machine Learning Models

1) Decision Tree Classifier – Decision tree classifiers are a common model used in machine learning classification because they are effective and easy to understand. The decision tree will split the dataset into smaller segments of data with two stages of predictions. According to Birjit Gope, "The primary objective is to create a training model that predicts the class or targeted values of the rules for learning decisions derived from training and past data" [2].

2) SVM - The support vector machine is a supervised learning model that distinguishes data points in a high dimensional space by optimizing the margin between classes [2]. SVM is known for its good accuracy and faster prediction compared to other models and is easy to scale with larger datasets.

3) Random Forest Classifier – Random Forest classification is a good model for image classification because it is good at gathering information about errors, the importance of variables, and gathering data outliers. It creates classification trees that create its predictions by voting by dominance. However, this model relies on estimation performance making it have low precision for minority classes [2].

4) Naïve Bayes – The Naïve Bayes depends on two simplifying assumptions, it's naïve and uses the Bayes theorem. It is naïve because it assumes that there are no secrets to impair the system for

predicting [2]. The Bayes theorem is a probabilistic classifier. The pros of Naïve Bayes are that is it less costly to run and works well for a small amount of training data [2].

5) K-Nearest Neighbor (KNN) – The K-Nearest Neighbor is the simplest model of the five in this project. KNN's approach is to use a lazy learner strategy that is focused on a feature similarity algorithm. This means that the algorithm is classifying data points based on the most common class in the k closest data points.

3.3 Data Preprocessing

Within my project I needed a way to efficiently pull in my dataset, change the color of my images, and prepare it for being passed into my models. The approach that I took for preprocessing was to use a global recursive search that uses the fruits name and file path to get the training and testing images. Once they were added to the list, I used the CV2 library which is also known as the OpenCV library. Using this library, I called on its imread_grayscale method which reads in my photo from the specific path and converts it to grayscale. This greatly increased the simplicity of transforming my dataset. From here, the Gaussian Blur method is called. I decided to use this in my project because looking at the dataset in the MNIST, it looked a bit blurred. Adding the Gaussian blur, it reduces the noise in the photos. Reducing the noise in the dataset is essential because it can cause inaccuracy in the models. For the binary grayscale dataset, the same steps are taken above and adding adaptiveThreshold method using the thresh binary call. This transforms my photos from grayscale to a data that is only white or black values. This is how I preprocessed my dataset for my project. The results below are for 3 of the 10 fruits in the dataset. The photo has been edited for to decrease the size so the image print out within the code will be in a different format.

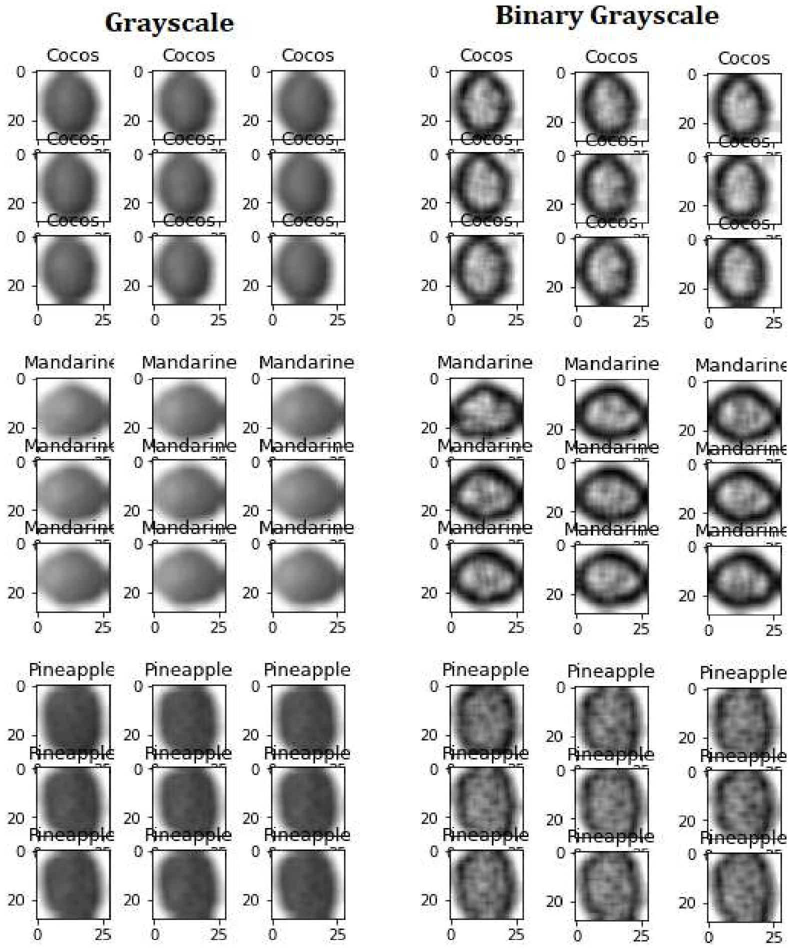


Figure 5: Visualization of binary grayscale and grayscale for Fruit 360

4 Experiments

The start of the project was playing with the MNIST dataset because that is the dataset used in the paper. Because I did not have the full capacity to replicate their models fully, I first passed my models on this dataset to give myself a baseline. Using the sklearn library digits I called the support vector machine, decision tree, naïve bayes, k-nearest neighbor, and random forest models. I also created graphs, tables, and confusion Matrix to give myself a baseline to compare my results to the paper and to the Fruits 360 dataset. MNIST data set is a benchmark that is highly used because it is easy to use, efficiently allows newcomers to machine learning to learn and evolve their skills on the dataset.

The goal of this project was to see how the preprocessing effect affected the models. The method I used for this project was to create 2 different data sets for the Fruit 360 dataset [5]. Within the paper, Handwritten Digits Identification Using MNIST Database Via Machine Learning Models they created two different types of reprocessing one for grayscale and the other for binary grayscale. I created my method to pull in the dataset and use the CV2 library also known as openCV to quickly and efficiently allow me to change the color scheme of the data set. The Fruit 360 dataset [5] is originally taken in normal RBG colors and converted into the appropriate pre-processing colors and 28 by 28 size. Now that the datasets are in the correct colors, I used a standard scaler. This standard scaler because without it the pre-process data sets are too high of a dimension and need to flatten them to pass them into each of my models. Now that my data is ready, I pass it into each classifier, train the classifier, and predict. From here I print the training accuracy and the testing accuracy to show how well we are doing. Along with the classification report to show individually how each one of the ten fruits is scoring to see where each model is doing well.

4.1 Overcoming Challenges

Within this project, I faced a lot of the common machine learning issues which are bad datasets, underfitting models, overfitting models, and long implementation times. These are issues that we have talked about in the class as well and was excited that I saw them in this project as well. It was interesting to see how quickly my dataset went from one end to the other.

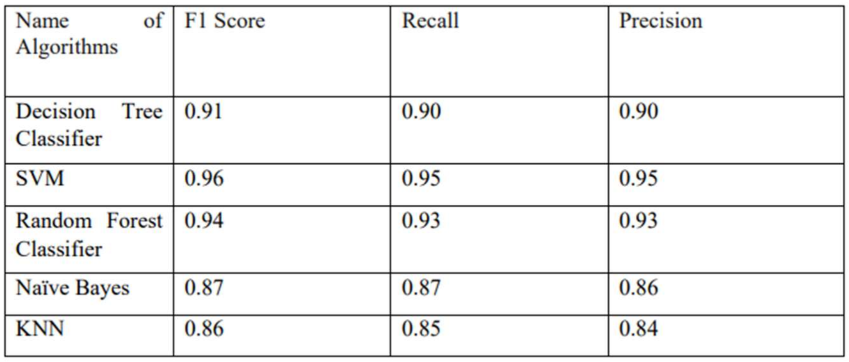
When exploring other datasets there were a lot of datasets that needed a large amount of work to clean up, process, and set up to get them into working order to be useable on my models. There were a few that I tried cleaning up but proved well beyond my personal skill set to get into working order. So, through this process, I was better able to look at datasets in Kaggle and the UC Irvine Machine Learning repository for datasets I could use. However, another issue that I found is a lot of the recommended datasets for image classification were benchmark datasets, so finding a dataset within my skillset was a challenge until I found Fruits 360.

An issue I faced was underfitting models within my project with the Fruit 360 dataset [5]. I was originally using five fruits but was receiving very poor results within my dataset. The highest accuracy I got between all five of my models was in the sixty percent range. The way I resolved this was by increasing the number of training images the models were using and doubling the number of fruits I was using to now include the full ten fruits: Banana, Cocos, Mandarine, Pineapple, Raspberry, Dates, Apple Red Delicious, Cactus fruit, Clementine, Granadilla. Then during this process as well, I went in the opposite direction of my model's becoming overfitting, a lot of my fruits were being misclassified once I increased the training image size. To overcome this challenge, I added the Gaussian Blur to slightly blur my fruit photos in the preprocessing to reduce the noise in the dataset. Once this was done, my models did improve to where they are now.

The last challenge that I faced is implementation times on my models. This is unfortunately an issue that I was not able to fully overcome within this project. My SVM models for my grayscale took 26 seconds to run and binary grayscale came in at 34 seconds running time. I felt like for the dataset size that I was working with for my project the time was a bit high, especially when looking at my other model's execution times. I am currently not sure how I would increase the time, however, if I were to continue working on this project, I would try to tweak my dataset and my models to improve the execution time.

4.2 Results

Below are the results of the MNIST Database Via Machine Learning Models [2] followed by my model results on the MNIST dataset and the Fruit 360 dataset. I was unable to perfectly replicate the results from the model paper and I will explain below.



Name of Algorithms	F1 Score	Recall	Precision
Decision Tree Classifier	0.91	0.90	0.90
SVM	0.96	0.95	0.95
Random Forest Classifier	0.94	0.93	0.93
Naïve Bayes	0.87	0.87	0.86
KNN	0.86	0.85	0.84

Figure 6: The results of the five models [2]

The results from the model paper which can be seen in the above table, achieved their highest accuracy with the SVM (support vector machine). This was what I would expect to be the highest since this model is notorious for performing well for image classification. On the other hand, their KNN was their lowest-performing model at 86% which is still a good accuracy. I was impressed with their results in the paper.

Name of Algorithms	**F1 Score**	**Recall**	**Precision**	**Time-Taken (in seconds)**
Decision Tree Classifier	0.84	0.84	0.84	0.11
SVM	0.99	0.99	0.99	0.13
Random Forest Classifier	0.95	0.95	0.95	0.35
Naïve Bayes	0.84	0.84	0.87	0.11
KNN	0.99	0.99	0.99	0.98

Figure 7: My results on the MNIST dataset [1]

The above results are my attempt at the MNIST dataset with the models. As you can see there is a large amount of overfitting happening in my results with the SVM and KNN both being at 0.99 percent accuracy. I did not spend nearly as much time tweaking the MNIST dataset since it is the benchmark dataset, however, it was interesting to see how well the models performed on this dataset. Using the sklearn built-in MNIST dataset, there were roughly 30-50 training images passed into the models. Given more time the first thing I would do is to expand the training and testing size to reduce the overfitting. An interesting note about these results is that I did get a similar trend in my Fruit 360 dataset with the SVM being the highest and Naïve Bayes being one of my lowest. Looking back, I should have realized there was tweaking that needed to be done to models here. I thought due to the larger dataset size for Fruits 360 it would perform closer to the model paper.

Algorithms for Grayscale	F1 Score	Recall	Precision	Time-Taken (in seconds)
Decision Tree	0.71	0.71	0.75	3.52
SVM	0.95	0.95	0.95	24.51
Random Forest	0.84	0.84	0.87	0.55
Naïve Bayes	0.62	0.61	0.65	0.28
KNN	0.94	0.94	0.95	1.58

Algorithms for Binary	F1 Score	Recall	Precision	Time-Taken (in seconds)
Decision Tree	0.68	0.68	0.71	4.51
SVM	0.92	0.92	0.92	33.90
Random Forest	0.83	0.83	0.85	0.67
Naïve Bayes	0.58	0.58	0.63	0.28
KNN	0.92	0.92	0.93	1.60

Figure 8: My results on the grayscale and binary grayscale Fruit 360 dataset [1]

As it can be seen above, the SVM is my highest scoring model at 95% accurate for grayscale and 92% accurate for binary grayscale. However, my lowest scoring model is my Naïve Bayes while the paper's lowest model is their K-Nearest Neighbors. As stated before, I did not reproduce the results of the paper, however, I was happy to see that I did get the SVM to perform the highest. There is a lot of work that needs to be done on my models. Plans for this project would be to research and tweak my models to better reproduce the results. Below are the confusion matrices of my SVM model and the Naïve Bayes.

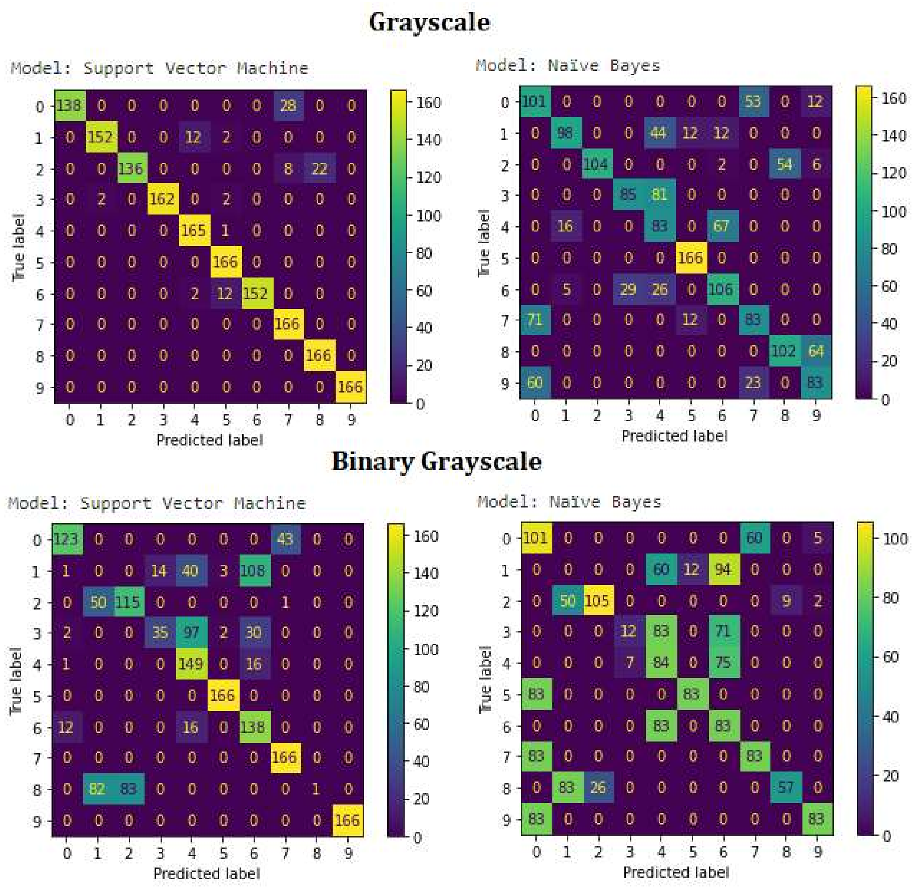


Figure 9: Confusion matrices for SVM and Naïve Bayes [1]

The top two are the confusion matrices for grayscale and the bottom two are for the binary grayscale. Zero through nine correspond to the ten fruits: Banana, Cocos, Mandarine, Pineapple, Raspberry, Dates, Apple Red Delicious, Cactus fruit, Clementine, Granadilla. I thought it was very interesting to see the difference between the SVM's and Naïve Bayes. The grayscale SVM was near perfect only mislabeling a few here or there but there is a clear and defined line. The Naïve Bayes on the other hand had a lot of mislabeled photos randomly throughout the dataset and only correctly predicted five which is date fruit. The Naïve Bayes scored lower than I would have liked because the paper scored an 84% accuracy while mine only scored 62%.

In the binary grayscale matrices, we can see how much of a decrease in efficiency the models had. There is no clear line, there is large mislabeling and this was my best model for binary grayscale. I was surprised to see that the binary grayscale did worse. It thought that creating harsh outlines on the fruit would help the models more accurately predict the fruit. The Naïve Bayes as shown are very confused on every fruit in the dataset. Both completely mislabeled the coco fruit which the grayscale also mislabeled but nearly to such an extent.

Unfortunately, I was unsuccessful in reproducing the results of the paper identically. I believe this happened because I need to do a better job at preprocessing my dataset and fine-tuning my models. I am disappointed that with the tweaking that I did do, the models were so far off. I believe I must look at the models themselves and not the preprocessing at this point to see how I could improve the accuracy. Another issue I would like to fix is the SVM model run time. They took a very long time to run, taking 25 and 34 seconds to run.

5 Conclusion

I have a lot of fun doing this project. The conclusion that I have learned during this class and in this project was machine learning is a lot more complicated than I anticipated and is a labor of love to do. I learned a lot and was excited about the challenge. There is a large amount of work that needs to go into researching the data, cleaning the dataset, prepping the data set, and then tweaking the data to give you better models. These are a lot of things I never thought of before this and was very interesting to deal with it firsthand without guidance from the professor. If I had more time on this project, I would spend more time tweaking my pre-processing, increasing the efficiency of my models, and finding a way to include the entire data set. I would like to have my results more accurately reflect the results of the paper. The implementation time of the SVM models is very high and I would like to bring that down while also increasing the accuracy of my Naïve Bayes.

5.1 Sharing agreement

No, I do not wish my paper or name to be shared.

5.2 My Contributions

The main adoption was doing two datasets – the MNIST and the Fruit 360. I did use the MNIST dataset, however to more to give myself a benchmark to use against the other dataset. I spent a large amount of my project on the Fruits 360, not wishing to spend much more time on the MNIST than creating the benchmark. I spent quite a bit of time interacting with datasets, roughly a week of trying various types of image classification datasets I found on the internet, however, a lot of them were beyond my knowledge in useability or had too few images. For example, I did get to pull in only had roughly 50 images for training and 30 for testing which I was not satisfied with, so I kept trying till I found one with better quality datasets. I spent quite a bit of my time tweaking the preprocessing process and reading about the cv2 library and how it is adjusting the color values in the dataset. Adjusting the testing size, color adjusting, and removing the noise in the dataset. I also created bar graphs and a confusion matrix to assist me with better visualization of where my models are going wrong.

References

[1] Conforti, Rachel. “RCONFORTI97/ITCS5156-Project” GitHub, 2022, github.com/rconforti97/ITCS5156-Project.

[2] Gope, Birjit, et al. “Handwritten Digits Identification Using Mnist Database Via Machine Learning Models.” IOP Conference Series. Materials Science and Engineering, vol. 1022, no. 1, IOP Publishing, 2021, p. 12108–, https://doi.org/10.1088/1757-899X/1022/1/012108.

[3] Juneja, K., Rana, C. Multi-Featured and Fuzzy-Filtered Machine Learning Model for Face Expression Classification. Wireless Pers Commun 115, 1227–1256 (2020). https://doi.org/10.1007/s11277-020-07620-8.

[4] W. Castro, J. Oblitas, M. De-La-Torre, C. Cotrina, K. Bazán and H. Avila-George, "Classification of Cape Gooseberry Fruit According to its Level of Ripeness Using Machine Learning Techniques and Different Color Spaces," in IEEE Access, vol. 7, pp. 27389-27400, 2019, doi: 10.1109/ACCESS.2019.2898223.

[5] Oltean, Mihai. “Fruits 360.” Kaggle, 12 Sept. 2021, www.kaggle.com/moltean/fruits.