

Summary Report: Model 2 (Support Vector Machine)

CS-4120

Omar Ebrahim and Kareem Eltouny

Instructor: Prof. Ahmed Elsheikh

5th December 2023

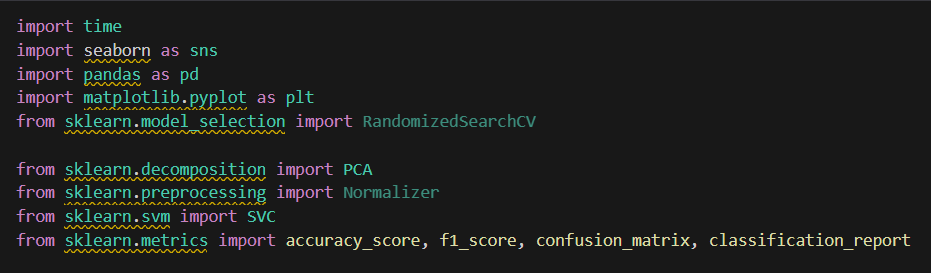
1. Background

The second model used for this project was Support Vector Machine. In the world of classification algorithms, SVM reigns supreme. Through the use of kernel functions, SVM is adept at managing non-linear relationships by translating the data into a higher dimension. This makes it a go-to choose for the purpose of binary and multi-class classification tasks, thanks to its capability to handle intricate decision borders and perform remarkably well on a variety of datasets. Based on its prominence in classification, we decided to select this as our second model in hopes of getting better accuracy results.

1. Importing the libraries

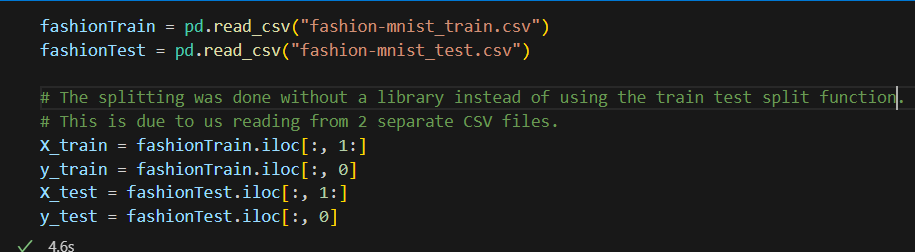
As was covered in the background section, scikit-learn was used to import the SVM as the second model used for this project, as well as the pre-processing techniques Normalizer and PCA. In addition, scikit-learn was used for accuracy, f1score, and classification\_report. Time was used for the training time. Finally, we implemented cross-validation in this model, and used the model\_selection library from sklearn to import RandomizedSearchCV.

For visualization purposes used for this model, we have also imported seaborn, pandas, matplotlib, and the confusion matrix.



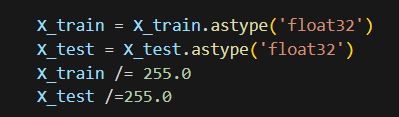
1. Importing and splitting the dataset

“fashionTrain” and “fashionTest” represent the two CSV files read using pandas. Due to this particular MNIST dataset appearing in two separate files with a set of ratios of training vs. testing, we decided to manually split them instead of using a built-in function from sci-kit.



1. Pre-Processing:
   1. Normalization

Given that the pixel values are stored as numbers with the range from 0 to 255, we can rescale this to save computation time. We normalized X and y as ‘float32’, which means that it is of single precision and cuts the memory in half. Moreover, this drastically decreases the training time of the SVM, although the model will be less precise as a result.



* 1. Principal Component Analysis

PCA was used to reduce the size of some of the dimensions present in the features. When observing the features of the fashion-mnist-test.csv file for example, we notice more presence of zeroes in comparison to the other pixel numbers.

Moreover, the purpose of the PCA here is transforming these large set of features into a slightly smaller version that still contains most of the information. In the implementation below, n\_components = 0.99 was the best value selected to maintain 99% of the variability in the data. As seen from the results below, we had 784 pixels decreased to 459 pixels which would help a lot in the computational time for our algorithm while also not losing a lot of information from the data as we maintained 99% of the variability

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a computer code

Description automatically generated

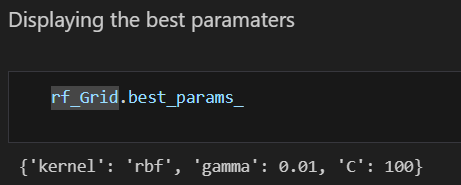
* 1. Cross Validation (Grid Search)

Although this took 19 minutes to perform, cross-validation was a new step we implemented to help us gain a new perspective as to what the ideal hyperparameters are for this model.

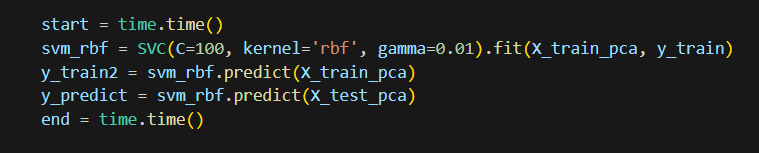
We implemented Randomized Grid Search with a list of potential hyperparameters for C and gamma, the results gave us 100 as the best C parameter, and 0.01 for gamma. Note that the kernel used was rbf (Gaussian Kernel),

A screenshot of a computer

Description automatically generated

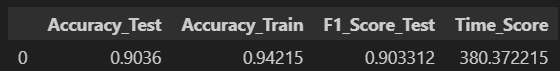


1. Training and Testing the Model:

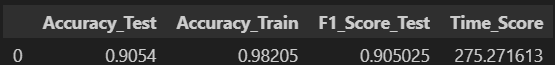


1. Getting the Accuracies:

When initially selecting the random hyperparameters, these were the results obtained:



After performing cross-validation through Randomized Grid Search, these were the new results:



Based on the model we have chosen, and the pre-processing techniques applied, we have attained and 90.54 % accuracy for testing and 98.2% for training. Moreover, there is no case of underfitting or overfitting with the model because of both accuracies being greater than 90.

As for the F1 score, we have attained 90.50 %, indicating a really strong precision in the chosen model which indicates a strong performance for the model. Finally, the total training/testing time recorded for the entire model was at 25.3 seconds.

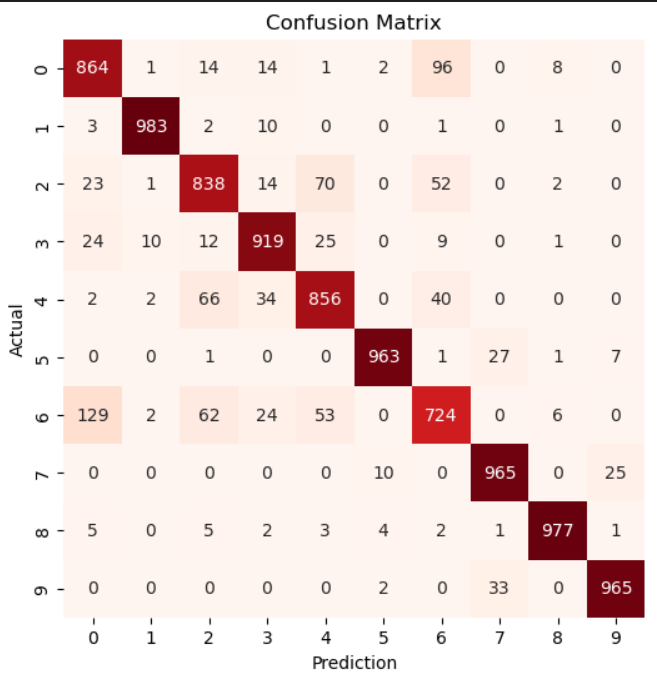
1. Visualization:

A screenshot of a computer screen

Description automatically generated



This classification report shows the precision, recall, and f1-score for each label in our dataset. We can deduce from these results that the least accurately predicted label is the label 6 which is the shirts which makes sense as it can easily be confused by the third least accurately predicted label which is label 0 (t-shits/tops). The best accurately predicted labels is label 1 and 8 which is are trouser and Bag respectively.

Based on this confusion matrix implementation, we have found the features 1,8,9,7,5,3 (highest to lowest) had the best predictions.

As for the factors that lowered the accuracy, we found that feature 0 (t-shirts/tops) and feature 6 (shirts) were confused with one another 129 times and lead to wrong predictions. In addition, feature 4 (coats) has feature 2 (pullovers) 66 times. Finally, feature 2 (pullover) and feature 6 (shirts) had confusion value of 62 times.