***TASK 1 (pre-processing and visualisation, 5 points)***

*Load the product image dataset and separate the labels [1 point] from the feature vectors [1 point]. How many samples are images of sneakers, how many samples are images of ankle boots [1 point]? Display at least one image for each class [2 point].*

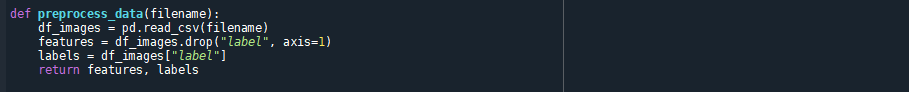
The *prepocess\_data(filename)* function takes the file parameter as an input and returns the processed data as an output. It takes the data from this file as a dataframe and processes it into ‘features’ and ‘labels’. The dataframe taken from this file is processed into ‘features’ and ‘labels’.

The first column entry in each row is the ‘label’, denoted with either 0 or 1; 0 denotes a sneaker, while 1 denotes an ankle boot. There are 7 thousand of each shoe type, totaling to 14000. The next 784 columns denote the color value for the pixel at that index. The grid to display these pixels was chosen as 28x28, calculated from the square root of the total amount of pixels.

All terminal output is formatted coherently.

Text

Description automatically generated



Text

Description automatically generatedA screenshot of a computer

Description automatically generated with low confidence

***TASK 2 (evaluation procedure, 9 points)***

*Create a k-fold cross-validation procedure to split the data into training [1 point] and evaluation subsets [1 point]. Parameterize the number of samples to use from the dataset to be able to control the runtime of the algorithm evaluation [1 point]. Start developing using a small number of samples and increase for the final evaluation.*

*Make the function flexible to accommodate different types of classifiers as required in tasks 3-6. Measure for each split of the cross-validation procedure the processing time required for training [1 point], the processing time required for prediction [1 point] and determine the confusion matrix [1 point] and accuracy score of the classification.*

*Calculate the minimum, the maximum, and the average of*

* *the training time per training sample [1 point]*
* *the prediction time per evaluation sample [1 point]*
* *and the prediction accuracy [1 point].*

The evaluate\_classifier() function act as a sort of framework for the other tasks. With no repetition of code, every other task may be run through it, where only the new respective classifier parameter is required for each task. A dictionary is returned as the summary of the given classifier, containing all of the important values for plotting.

Using the full size of the original dataframe (14000) may result in slow processing speeds due to hardware limitations. Thus, a parameterized sample size is preferred. Utilizing an easily changeable constant variable located at the top of the source code, one can quickly alter the sample size. With a sample size of 600, the total processing time is not drastically decreased, while still providing a sizable sample size for evaluation.

At each kfold, the training time, testing time, prediction accuracy and confusion matrix are displayed to the terminal in a formatted output.

The information summary dictionary is outputted to the terminal. This dictionary is also returned non-formatted.



Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

**TASK 3 (Perceptron, 3 points)**

Use the procedure developed in task 2 to train and evaluate the Perceptron classifier [1 point]. What is the mean prediction accuracy of this classifier [1 point]? Vary the number of samples and plot the relationship between input data size and runtimes for the classifier [1 point].

The evaluate\_perceptron() function is passed into the evaluate\_classifier() function, providing its respective classifier as a parameter. The summary dictionary is returned the entire way to the Main.

Text

Description automatically generated

Text

Description automatically generated A picture containing graphical user interface

Description automatically generated

**TASK 4 (Support Vector Machine, 5 points)**

*Use the procedure developed in task 2 to train and evaluate the Support Vector Machine classifier [1 point]. Use a radial basis function kernel and try different choices for the parameter 𝛾 [1 point]. Determine a good value for 𝛾 based on mean prediction accuracy [1 point]. What is the best achievable mean prediction accuracy of this classifier [1 point]? Vary the number of samples and plot the relationship between input data size and runtimes for the optimal classifier [1 point].*

Lorem ipsum dolor sit amet.

**TASK 5 (k-nearest Neighbours, 5 points)**

Use the procedure developed in task 2 to train and evaluate the k-nearest neighbour classifier [1 point]. Try different choices for the parameter k [1 point] and determine a good value based on mean prediction accuracy [1 point]. What is the best achievable mean prediction accuracy of this classifier [1 point]? Vary the number of samples and plot the relationship between input data size and runtimes for the optimal classifier [1 point].

Lorem ipsum dolor sit amet.

**TASK 6 (Decision trees, 3 points)**

*Use the procedure developed in task 2 to train and evaluate the Decision tree classifier [1 point]. What is the mean prediction accuracy of this classifier [1 point]? Vary the number of samples and plot the relationship between input data size and runtimes for the classifier [1 point].*

Lorem ipsum dolor sit amet.

**TASK 7 (comparison, 5 points)**

*Compare the training and prediction times of the four classifiers. What trend do you observe for each of the classifiers and why [4 points]? Also taking the accuracy into consideration, how would you rank the four classifiers and why [1 point]*

*Lorem ipsum dolor sit amet.*