Machine Learning

Exercise: 2

Author: Mylonaki Angeliki

Overview

Applying Project Template for Classification problems in MNIST dataset.

Loading Dataset

Using the link provided in the exercise description to download the dataset, we then load it into jupyter-notebook to analyse it. We only use a part of the dataset in our exercise, since calculations using the full dataset are not possible due to machine limitations.

Problem to be solved

Given the MNIST dataset we want to select and train a suitable model in order to be able to predict a digit when given a handwritten version of it.

Taking a peek of the data

We use functions provided by the “pandas” library to explore our data. In order to do this, we examine some basic statistics (mean, standard deviation etc), the data shape and datatypes.

In this step, we can also calculate the correlations between the features and visualize them. This is not possible in our case, since the machine used to implement this exercise had hardware limitations.

**Although we cannot really calculate correlations, we assume that some correlations exist.*

Preparing the Data

In order to help algorithms produce better results, we Normalize and Scale our data.

Feature Engineering

In this step, we can make enhancements to our features in order for the algorithms to perform better. Since we assumed that some correlations might exist, we use PCA to resolve that. There are also other ways to approach this. For example, we could drop some features that seem to give minimum information. Considering that an image is represented by a two-dimensional array we could drop the pixels in the four corners of the image, since it is unlikely that a value other than zero would be present there.

Algorithm Selection

We train multiple models using a variety of **classification** algorithms and cross validation to decide which one best fits our needs. We calculate the accuracy of each algorithm and plot in using box plots. Using this information, we can select the best algorithm.

Finalizing the Model

Once we have selected the best model for our case we have to tune the respective hyperparameters so as to get the best results. In our case, SVM (or SVC) produces the best results. In order to tune the hyperparameters, we provide the following parameter grid:

```
param_grid = {'C': uniform(),
        'kernel':[ "linear", "poly", "rbf", "sigmoid"],
        'gamma':uniform(),
        'coef0':uniform(),
        'shrinking':[True, False],
        'tol':uniform(),
        }
```

After that, we use the “best values” for each of the parameters, generated by RandomizedSearchCV, we create the respective model and train it using the training part of the data. We then use the generated model to predict the digits displayed in the test subset.

Applying Ensembles

Using **Ensembles** we can combine multiple algorithms to have more accurate results in prediction. In or case we use **Voting.** Since we have already trained multiple models to measure their accuracy, we already know the ones that behave better. Using the voting method we can assign a weight on each algorithm, giving the lowest weights on the ones that behave the worst. The final model uses all algorithms to generate predictions with respect to the weights assigned to them.