Classification Report on MNIST dataset – Voulgari Eleni

Introduction

In this assignment we are trying to evaluate some algorithms that are used for performing hand written digit recognition. This task is translated into a classification problem.

The dataset we are using is a part of the MNIST dataset. The original MNIST dataset is a large database of handwritten digits that is commonly used for training various image processing systems. It contains 60,000 training images and 10,000 testing images. We are going to use a part of the dataset which contais 42,000 samples.

To choose the best model for predicting the handwritten numbers, we experiment with different preprocessing methods and machine learning algorithms. The preprocessing methods contain the scaling of the data and the use of PCA algorithm to reduce the dimensionality of them. The algorithms we are using are: Random Forest, Linear Discriminant Analysis, K-Nearest Neighbors, Support Vector Machine, Naive Bayes classifier and Logistic Regression.

Step 1: Prepare Project

Our first step is to prepare the project which contains loading the libraries we are going to need and loading the dataset itself. The load of the dataset was done with the creation of a proper script, which downloaded the file in the right format (.csv) and then trasformed it into a pandas dataframe.

Step 2: Define Problem

Our next step is to define the task that we have. In this case, as we mentioned before, the task is to experiment with different models and settings and decide on the best model for recognizing hand written digits. This task is a classification problem.

Step 3: Exploratory Analysis

The next step is to understand our data. We take a "peek" of it and answer basic questions about it. The dataset contains gray-scale images of hand-drawn digits, from zero through nine.

As we can see, it contains 42,000 images. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. The dataset, has 785 columns. The first column, called "label", is the digit that was drawn by the user. The rest of the columns contain the pixel-values of the associated image.

There are no missing values and the instances are somehow evenly distributed between the classes.

Step 4: Prepare Data

The step of preparing our data contains data cleaning, data wrangling and collecting more data. In this case, we are not going to do any cleaning or collecting data, but we perform data wrangling by normalizing and scaling the data, so as to check if scaling the data will produce better results for the winning algorithm.

Step 5: Feature Engineering

The next step is feature engineering which contains feature selection, feature engineering (as in new features) and/or data transformations. In our case, we use PCA which is a form of feature engineering that reduce the dimensionality of the data. This is also being done so as to check if reducing the components of the data will produce better results for the winning algorithm. We are reducing the features to 100.

Step 6: Algorithm Selection

This step contains the selection of a set of algorithms to apply, the selection of evaluation metrics, and the evaluation and comparison of the algorithms.

As we already have mentioned we have selected the following algorithms to apply to the dataset: Random Forest, Linear Discriminant Analysis, K-Nearest Neighbors, Support Vector Machine, Naive Bayes classifier and Logistic Regression and the metric we are using is 'accuracy'.

So, we apply the algorithms and evaluate their accuracy by using the k-fold algorithm to find which of these is the best. The results are shown below:

```
RF: 0.888400 (+/- 0.012831)
LDA: 0.836400 (+/- 0.010613)
KNN: 0.933200 (+/- 0.008060)
SVM: 0.109400 (+/- 0.012200)
NB: 0.552000 (+/- 0.030146)
LR:  0.814400 (+/- 0.012484)
```

As we can conclude, the algorithm with the best accuracy score is K-Nearest Neighbours.

Step 7: Model Training

The following step contains the application of ensembles and the improvement of performance by hyperparameter optimisation.

As we can conclude from the previous step the model that produces the best accuracy, thus the one that predicts the hand written digits better than the others, is K-Nearest Neighbors. The accuracy of it if we use the original data is 0.927878787879.

In this step we check if the accuracy improves with the scaled data and the data we have transformed with PCA. The accuracy for the scaled data is 0.880606060606 and for the PCA data is 0.933333333333, so the accuracy seems to be a bit better if we reduce the features to 100.

By using the dataset with reduced features, we are using GridSearch to perform hyperparameter optimization and print the best hyperparameters. The best hyperparameters for KNN algorithm for this dataset are:
['n_neighbors': 5, 'metric': 'euclidean', 'weights': 'distance']
We then observe that the accuracy is somehow a bit improved to 0.936363636364.

We are using the voting classifier to combine the different machine learning classifiers and use a majority vote to predict the class labels. This does not seem to making the results better for the accuracy which is now 0.903582089552.

Step 8: Finalise Model

Last but not least, as long as we have decided the model that best predicts the hand written digits, which is K-Nearest Neighbors and have optimized its hyperparameters, that produce the highest accuracy, we now finalise our model. Finalising the model means that we create it from the whole training dataset.

```
model_param = KNN_param.fit(pca_data5000, target5000)
model_param_all = KNN_param.fit(data, target)
```