

Report on Homework 1

Gerardo Paladino

1918178

1 Introduction

1.1 Introduction

This paper represents a report on the Homework 1 of the Machine Learning course. The assignment is to realize and evaluate a supervised learning models on two datasets with different input space. In particular, the problem is a classification problem of 10-class datasets.

1.2 Some considerations

In a first inspection of the datasets, it is possible to observe that, in both cases, the datasets are balanced. Indeed, for each sample, many features are set to 0, and some features are different to 0. These features represent a discriminant for the class of that sample.

Another proof of the datasets's balance is the perfect equality of number of classes, as shown in figure 1.

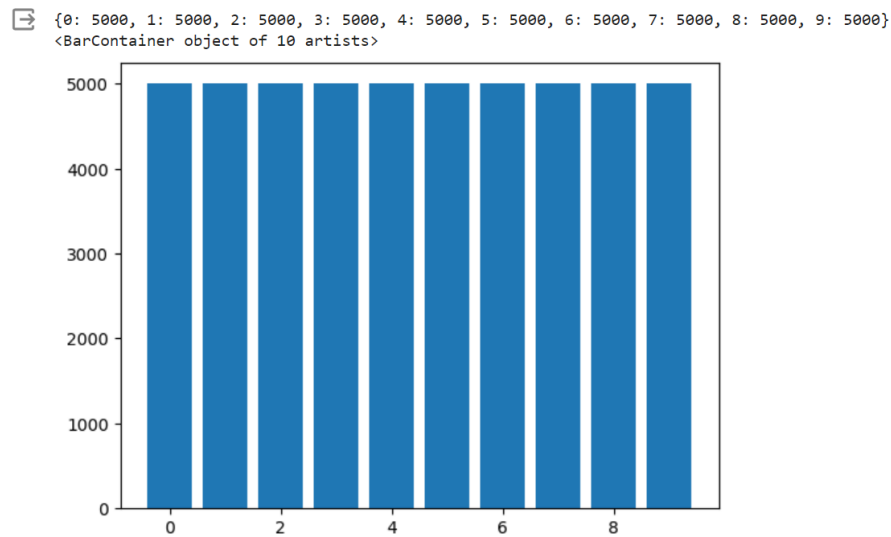


Figure 1: Diagram of 10 class. The x-axis represents a class, the y-axis represents a number of samples of both datasets that have corresponding class.

Another thing to take into account is the type of pre-process used for classification. In both classifications, the pre-process chosen is a standardization, which avoids measurements on different units of measurement or order of magnitude. Therefore the model learned on dataset 1 is more accurate than the one on dataset 2. Furthermore, the computational training time of dataset 2 is higher than dataset 1 as well. These results will be briefly discussed in the conclusion

section.

The models chosen for comparison are:

- Decision Tree
- SVM
- KNN

The hyperparameters list of the models are: for decision tree, the criteria of split in each iteration of tree, the strategy type to split at each node and the maximum depth of the tree; for SVM, a type of kernel, degree of the polynomial (if the kernel is poly) and the kernel coefficient (if the kernel is rbf, poly or sigmoid); for KNN, a number of nearest neighbors, the weight function and the algorithm used to compute the nearest neighbors.

In the end, the datasets are splitted in two subsets: 80% for train set and 20% for test set (the validation of the dataset is computed by K-Fold Cross-Validation).

2 Dataset 1

2.1 Shape

```
➡ Input shape: (50000, 100)
Output shape: (50000,)
Number of attributes/features: 100
Number of samples: 50000
Example:
x3365 = array([0.12018555, 1.118249 , 0.          , 2.0075612 , 4.1256638 ,
               0.          , 0.          , 2.2984993 , 0.          , 1.462071 ,
               0.          , 0.          , 0.          , 0.          , 0.          ,
               0.          , 4.5801663 , 0.          , 2.3051662 , 0.          ,
               2.14584  , 0.          , 0.          , 5.4590883 , 3.293236 ,
               0.          , 1.4857204 , 0.          , 2.9974902 , 0.          ,
               0.          , 0.          , 0.          , 7.189437 , 0.          ,
               0.5672025 , 1.4455842 , 2.118106 , 0.          , 0.5361991 ,
               4.9947267 , 5.524548 , 0.5045251 , 1.0253204 , 0.16632715,
               3.5827076 , 0.          , 0.25730067, 3.2998674 , 0.8849072 ,
               0.          , 0.5457191 , 0.          , 2.553366 , 0.45537165,
               0.          , 0.          , 0.45316783, 4.143042 , 4.6528616 ,
               1.0579351 , 0.          , 0.          , 0.          , 0.          ,
               0.1894842 , 0.          , 1.9096518 , 1.8443116 , 1.3028518 ,
               0.          , 0.45683283, 0.          , 0.          , 0.7842243 ,
               2.0525975 , 0.          , 0.          , 0.          , 0.          ,
               0.294677 , 0.          , 0.10570458, 0.          , 2.4540958 ,
               2.0452318 , 2.992731 , 1.1301323 , 3.1689377 , 0.2767097 ,
               3.0856464 , 0.04459978, 2.91205  , 0.25622556, 1.4924682 ,
               0.          , 1.107074 , 0.          , 0.          , 0.          ,
               ])
y3365 = 3
```

Figure 2: Dataset 1 shape.

The figure 2 shows the shape of dataset 1. In particular, the input shape represents rows and columns of the X matrix, where each row is one sample of

the dataset and each column is a particular feature, whereas the output shape represents a column vector of output. For completeness the figure shows an example of dataset (in this case the sample number 3365) with the relative class and the features's values, where many of this are set to 0.

2.2 Learning

2.2.1 K-Fold Cross-Validation

As mentioned before, the chosen models are the decision tree, SVM and KNN. It has been decided, for the choice of hyperparameters, to use a K-Fold Cross-validation, with the number of fold equal to 5 and the accuracy metric for the comparison. The output of the 3 validations is shown in figures 3, 4 and 5.

```
the hyperparameters choosen are criterion=gini, split=best, depth=12 and the max accuracy is 0.978
```

Figure 3: Output of K-Fold Cross-Validation for decision tree.

```
the hyperparameters choosen are kernel=poly, degree=1, gamma=scale and the max accuracy is 0.989
```

Figure 4: Output of K-Fold Cross-Validation for SVM.

```
the hyperparameters choosen are number of neighbors=9, algorithm=auto, weights=uniform and the max accuracy is 0.988
```

Figure 5: Output of K-Fold Cross-Validation for KNN.

Afterwards, a models have been created and fitted with these hyperparameters.

2.2.2 Evaluating

The first evaluation method is based on the classification report of each model. The classification report shows the precision and recall metrics, beyond an accuracy metric. In this case, as for dataset 2, the accuracy metric is enough, beacuse the models work well, however, for completeness, the other metrics have also been reported.

Precision and Recall table of decision tree is:					
	precision	recall	f1-score	support	
0	0.987	0.986	0.986	1036	
1	0.993	0.992	0.993	1008	
2	0.974	0.977	0.975	998	
3	0.941	0.941	0.941	958	
4	0.981	0.975	0.978	981	
5	0.954	0.957	0.955	1019	
6	0.977	0.989	0.983	1010	
7	0.986	0.985	0.985	969	
8	0.997	0.990	0.993	998	
9	0.990	0.990	0.990	1023	
accuracy			0.978	10000	
macro avg	0.978	0.978	0.978	10000	
weighted avg	0.978	0.978	0.978	10000	

Figure 6: Classification report of decision tree.

Precision and Recall table of SVM is:					
	precision	recall	f1-score	support	
0	0.993	0.991	0.992	1036	
1	0.997	0.994	0.996	1008	
2	0.991	0.988	0.989	998	
3	0.960	0.965	0.963	958	
4	0.991	0.993	0.992	981	
5	0.969	0.965	0.967	1019	
6	0.997	0.998	0.998	1010	
7	0.992	0.993	0.992	969	
8	0.996	0.997	0.996	998	
9	0.993	0.997	0.995	1023	
accuracy			0.988	10000	
macro avg	0.988	0.988	0.988	10000	
weighted avg	0.988	0.988	0.988	10000	

Figure 7: Classification report of SVM.

➡ Precision and Recall table of KNN is:

	precision	recall	f1-score	support
0	0.995	0.991	0.993	1036
1	0.998	0.996	0.997	1008
2	0.987	0.989	0.988	998
3	0.955	0.958	0.957	958
4	0.989	0.996	0.992	981
5	0.968	0.961	0.965	1019
6	0.995	0.997	0.996	1010
7	0.995	0.994	0.994	969
8	0.997	0.997	0.997	998
9	0.995	0.996	0.996	1023
accuracy			0.988	10000
macro avg	0.987	0.988	0.987	10000
weighted avg	0.988	0.988	0.988	10000

Figure 8: Classification report of KNN.

The second evaluation method is the confusion matrix, that shows the "confusion" of the classification of the different classes. The number in row i and column j represents the number of the cases that the classifier classified the "true" class i as class j .

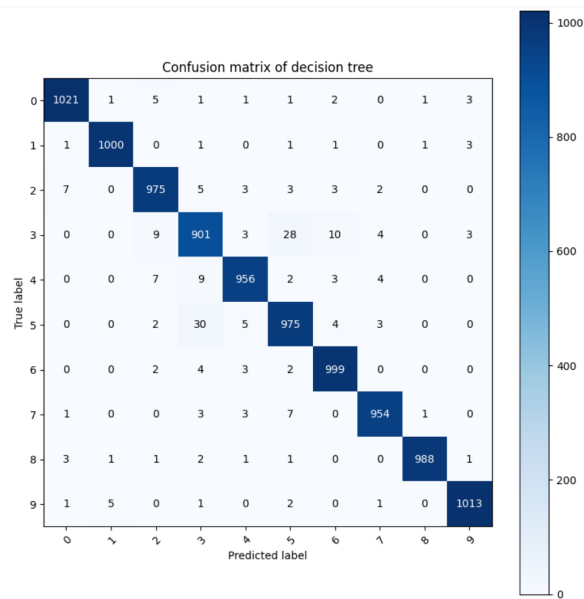


Figure 9: Confusion matrix of decision tree.

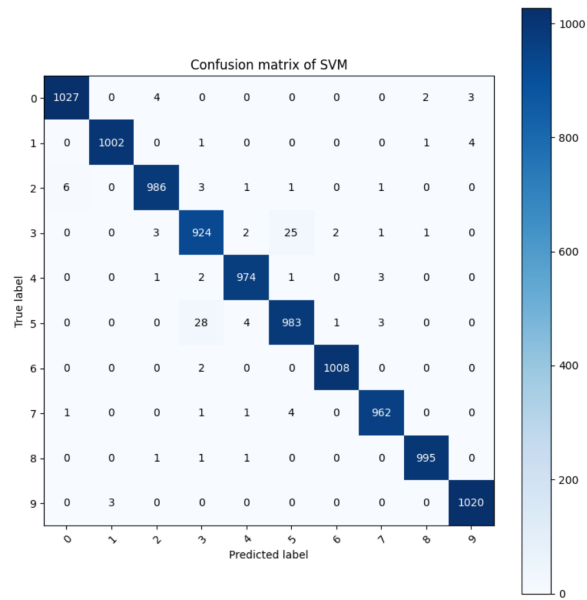


Figure 10: Confusion matrix of SVM.

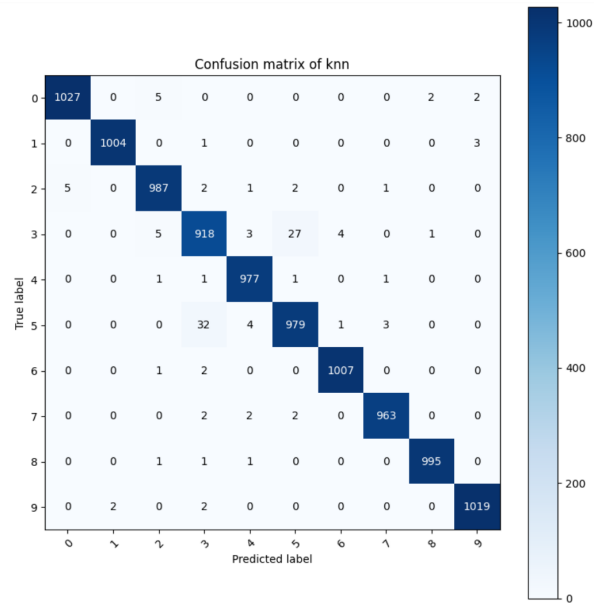
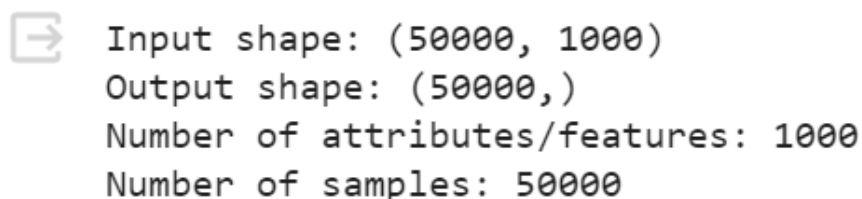


Figure 11: Confusion matrix of KNN.

The final classifier chosen is SVM.

3 Dataset 2

3.1 Shape

A screenshot of a Colab Notebook cell showing the shape of Dataset 2. The output is displayed in a monospaced font. To the left of the text is a small icon of a box with an arrow pointing right.

```
Input shape: (50000, 1000)
Output shape: (50000,)
Number of attributes/features: 1000
Number of samples: 50000
```

Figure 12: Dataset 2 shape.

The figure 12 shows the shape of dataset 2. The example of a sample is not reported, because too long, being 1000 the number of features, however, it is shown on Colab Notebook.

3.2 Learning

3.2.1 K-Fold Cross-Validation

Also in this case, it has been used a K-Fold Cross-Validation to validate the dataset and optimize the hyperparameters. The output of validations are shown in figure 13, 14 and 15.

```
the hyperparameters choosen are criterion=gini, split=best, depth=13 and the max accuracy is 0.958
```

Figure 13: Output of K-Fold Cross-Validation for decision tree.

```
the hyperparameters choosen are kernel=poly, degree=3, gamma=scale and the max accuracy is 0.974
```

Figure 14: Output of K-Fold Cross-Validation for SVM.

```
the hyperparameters choosen are number of neighbors=10, weights=distance,algorithm=auto, and the max accuracy is 0.972
```

Figure 15: Output of K-Fold Cross-Validation for KNN.

3.2.2 Evaluating

As above, the evaluation of the models was done with the classification reports and confusion matrices of each models, as represented in the next figures.

➞ Precision and Recall table of decision tree is:

	precision	recall	f1-score	support
0	0.974	0.972	0.973	1036
1	0.982	0.978	0.980	1008
2	0.965	0.947	0.956	998
3	0.877	0.892	0.885	958
4	0.938	0.959	0.949	981
5	0.900	0.914	0.907	1019
6	0.979	0.970	0.975	1010
7	0.974	0.962	0.968	969
8	0.989	0.987	0.988	998
9	0.982	0.977	0.979	1023
accuracy			0.956	10000
macro avg	0.956	0.956	0.956	10000
weighted avg	0.956	0.956	0.956	10000

Figure 16: Classification report of decision tree.

➞ Precision and Recall table of SVM is:

	precision	recall	f1-score	support
0	0.981	0.986	0.984	1036
1	0.989	0.988	0.989	1008
2	0.976	0.963	0.969	998
3	0.907	0.931	0.919	958
4	0.976	0.972	0.974	981
5	0.942	0.930	0.936	1019
6	0.986	0.986	0.986	1010
7	0.985	0.983	0.984	969
8	0.992	0.995	0.993	998
9	0.989	0.987	0.988	1023
accuracy			0.973	10000
macro avg	0.972	0.972	0.972	10000
weighted avg	0.973	0.973	0.973	10000

Figure 17: Classification report of SVM.

➡ Precision and Recall table of KNN is:

	precision	recall	f1-score	support
0	0.977	0.985	0.981	1036
1	0.988	0.989	0.989	1008
2	0.969	0.966	0.967	998
3	0.908	0.922	0.915	958
4	0.975	0.975	0.975	981
5	0.941	0.921	0.931	1019
6	0.983	0.986	0.985	1010
7	0.982	0.982	0.982	969
8	0.992	0.995	0.993	998
9	0.990	0.985	0.988	1023
accuracy			0.971	10000
macro avg	0.971	0.971	0.971	10000
weighted avg	0.971	0.971	0.971	10000

Figure 18: Classification report of KNN.

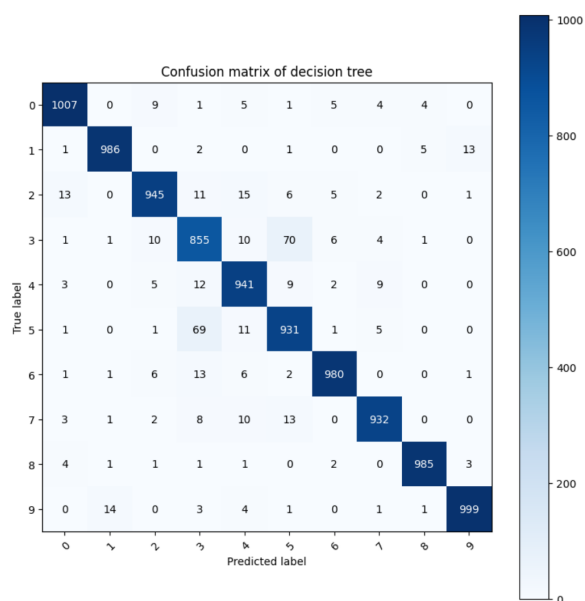


Figure 19: Confusion matrix of decision tree.

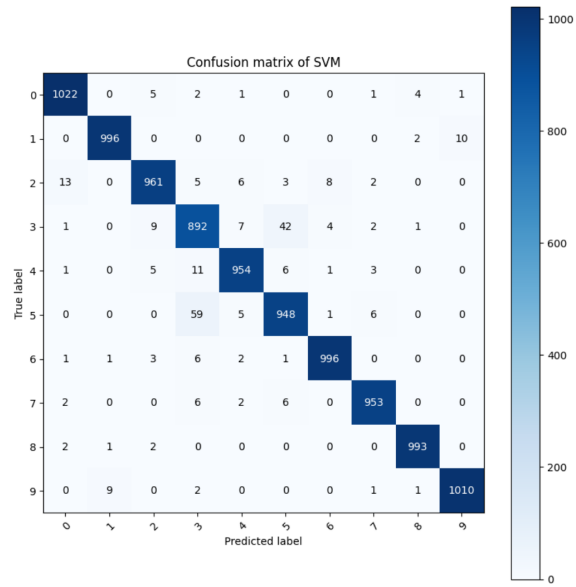


Figure 20: Confusion matrix of SVM.

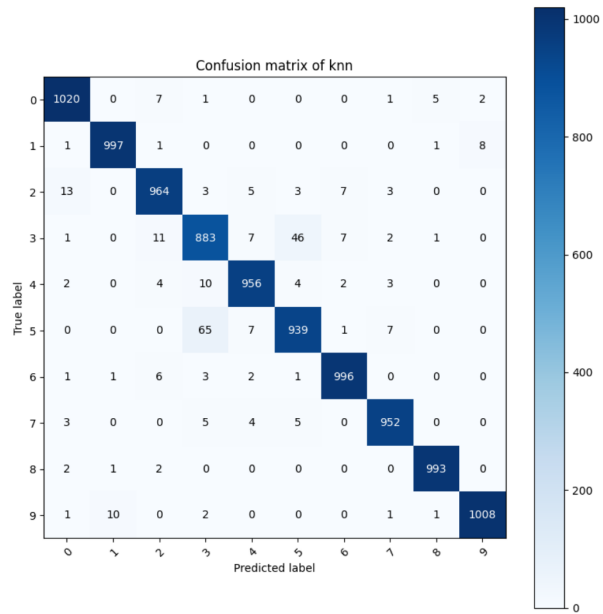


Figure 21: Confusion matrix of KNN.

Also in this case, the chosen classifier is SVM.

4 Conclusion

As expected, the final model for dataset 1 is more accurate than the model for dataset 2. In general, each models is a good candidate for the final model, since the accuracy is very high for all. A proof of this is the non-presence of overfitting, due to the K-Fold Cross-Validation, which does not present a greater performance on a particular fold. This is confirmed by the similarity of accuracy on train set, test set and fold test.

With regard to computational training time, it happened as expected, but with different performance. For dataset 1, the time of learning in validation process is about 30 minutes for decision tree, about 25 minutes for SVM and about 45 minutes for KNN. For dataset 2 the time is about 5 hours and 30 minutes for decision tree, about 2 hours and 45 minutes for SVM and about 7 hours and 50 minutes for KNN. In the last case, the time of SVM is not in scale with the others. This timing, being relative to validation process, also called tuning, depends on number of hyperparameters, and it is high for the product of single group of hyperparameters (for example, for decision tree, there are $3 \times 2 \times 30$ making, fit and evaluation of model). The single computational training time for fit the model is, for dataset 1, of 6 seconds for decision tree, 4 seconds for SVM and 0.006 seconds for knn (very fast!), for dataset 2, of 55 seconds for decision tree, 62 seconds for SVM and 0.009 seconds for KNN. In both cases, KNN is much faster than others. This speed is due to the simpleness of algorithm and calculus of this.

In the end, looking at the accuracy and timing, in the second case, the SVM model is better than decision tree about the accuracy, and better than KNN about the tuning timing. For this reason the choice is simple. Instead, in the first case, the accuracy of KNN is better than decision tree, but very similar to SVM. Therefore, both models are optimal choice.