University of Southern Denmark

SML-F21

Statistical Machine Learning

Applying Neural Networks and Support Vector Machines to hand-written digit recognition

Authors:
Ines Benomar
inben20@student.sdu.dk
Vincenzo Coppola
vicop20@student.sdu.dk
Erik Kockar
erkou20@student.sdu.dk
Karol Szurkowski
kaszu19@student.sdu.dk

Supervisors:
Norbert Krüger
Zhuoqi Cheng

May 31, 2021

Abstract

Human beings are the world's best pattern recognition machines... so far. Indeed, while humans are extremely skilled at recognizing letters, numbers and even faces, hope hasn't been lost yet in the field of machine learning. Methods like Artificial Neural Networks (ANN) and Support Vector Machines (SVM) are amongst some of the most popular in pattern and character recognition –a challenging topic in the field of image processing.

In this work, we chose to compare the efficiency of an SVM classifier and a feed-forward neural network in the application of a handwritten digit recognition, using the data-set provided in this course.

In order to analyze the performance of the two algorithms, they are both divided into training and testing phases and a comparative analysis of the accuracy of their results and run-time is performed. The input data used by both methods is pre-processed using Principal Component Analysis (PCA) and then fed to the chosen classification algorithm.

The results suggest that compared to the SVM, the MLP reached a higher accuracy using data from 20 people. Experiment conducted on all persons in dataset with split 50/50 shows that accuracy on testing data gave 85% success where on training data resulted in 96% succesfull detection. In the same test SVM classifier scored around 79% of succesfull classification for both testing and training data.

Contents

List of Figures

List of Tables

1 Introduction

In recent years, handwritten letter recognition has become one of the most challenging and popular research topic in the area of pattern recognition. That isn't so hard to grasp when we find ourselves having trouble reading a doctor's rushed handwriting, or we struggle to decipher an important note we left ourselves. The variety of handwritten styles and the variation between one person and another is bad enough for us, humans, let alone machines.

Over the course of this semester, we have studied numerous algorithms and pre-processing methods that have been proposed to decrease the processing time of this application as well as increase the accuracy of the recognition. Amongst them, Support Vector Machine (SVM) and Artificial Neural Networks (ANN), whose applications extend far beyond the scope of this one field, but that we chose for their robustness and popularity in literature.

This report present the results of the comparison of these two classifiers, first detailing each one of them and their results separately in sections 3 and 2 respectively, before comparing the best models obtained from each method in section 4. Finally, the work is commented and concluded upon in section 5.

All the data used in this project was collected the fellow students of this course. At the beginning of the semester, each student from the course was given a clean sheet with blank spaces to fill in the digits, allowing for a variety of handwritten styles. The total amount of data that was obtained came from 38 different people. Collected and filled sheets were then scanned using the 300 DPI (Dots Per Inch) resolution. An example filled sheet of 200 handwritten digits of cipher **8** with is presented in the figure 1.

The classification was conducted in two different approaches of the dataset – *All Persons In*, in which we choose the training and test data from a pool of all the people in the dataset, and *Disjunct* – in which the training data comes from a subgroup of people, we are predicting the digit of another.

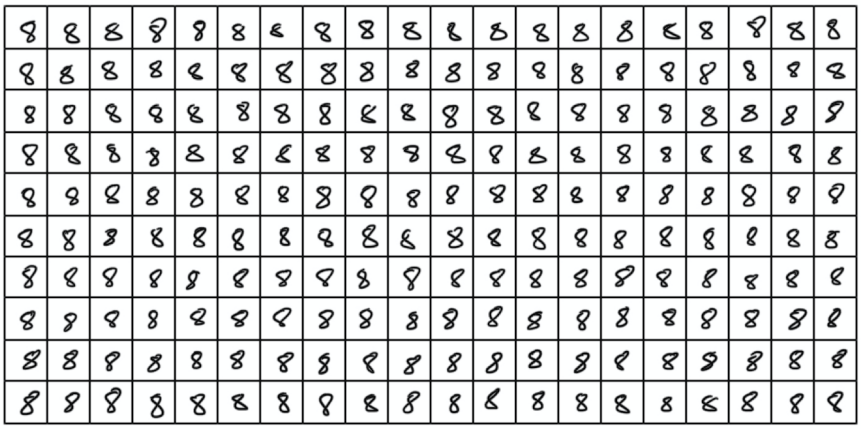

Figure 1: Image of one handwritten cipher generated from a tablet with an input stylus.

1.1 Preprocessing

For the pre-processing of our data, we have chosen the *Feature Scaling or Min-Max Normalization* in order to rescale the range of all values being between numbers 0 and 1. Since the images were saved in grayscale, the benefit achieved through the normalization process was making sure that the background is as close to the true white as possible, and the blackness of strokes of the pens is more evenly spread across pixels. Changes to the original digits made by min-max normalization can be observed in the figure 2.

Figure 2: Cipher 8 before (left) and after normalization(right).

Although we have experimented with the Gaussian smoothing and averaging of the saved ciphers using K-means clustering, due to the high execution time of those preprocessing techniques we disregarded them in order to focus instead on the comparison of the sole classification methods alone.

Having the execution of the classification algorithms in mind, we used the PCA decomposition method, with which we pre-processed the data for every inspected method to

obtain clear results.

1.1.1 Principal Component Anaysis

PCA is a statistical method that allows to compute the principal components and reduces the dimension of the data, such that we can project each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of its variation as possible [1]. The choice to use this method then makes sense as the dimension reduction aspect allows us to speed up a lot of the training and evaluation phases of the methods without sacrificing the accuracy of the classification.

As for the choice of the number of first principal components to use to describe the digits we have chosen to accumulate 85% of variance, which used 48 first principal components.

The example of pre-processed digit **8** can be observed in the figure 3.

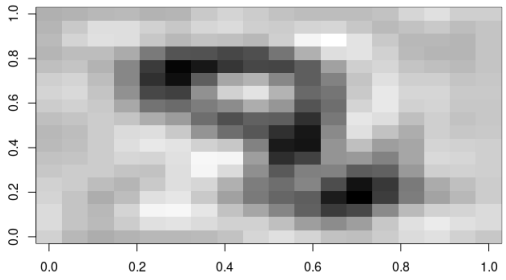

Figure 3: Cipher 8 reconstructed from 48 principle components saving 85% of variance.

2 Support Vector Machines - SVM

SVM (Support Vector Machine) is a supervised learning algorithm often used to solve machine learning problems. This algorithm can be used for classification and regression analysis. In case of hand-written numbers classification SVM needs to work with higher dimensional data, so standard SVM binary method is not enough. With n-dimensional datan SVM is constructing more hyper-planes or one n-dimensional hyper-planes. The goal of these latter is to separate the classe, thus with the largest margin to nearest training sample of a class, the optimal hyperplane is reached.

The separation of two data clusters with hyperplane can be seen in Figure (4)

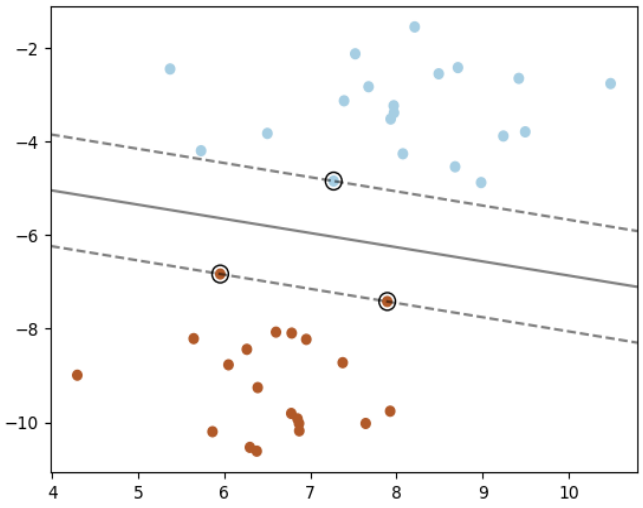

Figure 4: SVM hyperplane separation

The input used for this method n points is in the form of $(x_1, y_1), \ldots, (x_n, y_n)$, where y parameter is indicating the class to which the print x belongs and each x is a p - dimensional vector.

Any hyperplane can be written as the following equation:

$$w \cdot x + b = 0 \tag{1}$$

where:
w - normal vector to the hyperplane, determining hyperplane orientation

b - determines the offset of the hyperplane from the origin along the normal vector w

2.1 Kernels difference

While using the SVM approach, we can use different kernels as function parameters. A kernel is a set of mathematical functions which SVM uses, it takes the input data and transform it into the required form. There are different types of kernels that can be used with different SVM algorithms, namely:

1. Linear
2. Non-linear
3. Polynomial
4. Radial Basis Function (RBF)
5. Sigmoid

The most used and popular type of kernel is probably the RBF one, because along entire x-axis it has a localized and finite response. Difference between kernels classification with random data can be seen in the Figure (5).

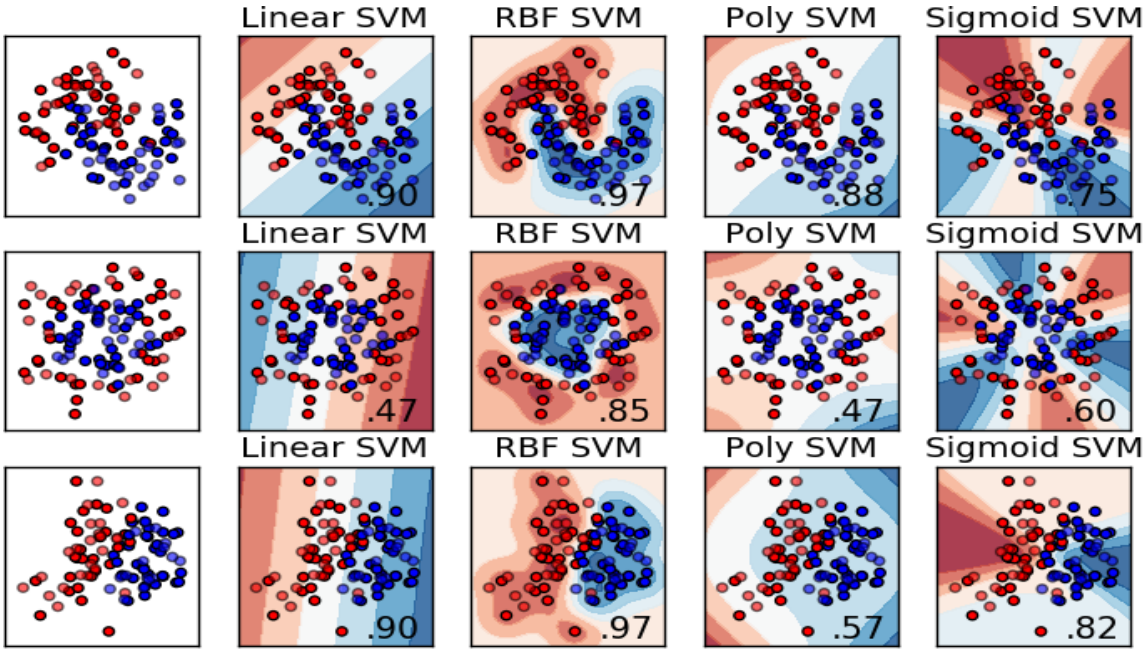

Figure 5: Difference in kernel types

2.2 SVM results

In the experiment, Linear, Polynomial and Radial kernels were applied to our hand-written digits classification problem and results of classification accuracy were obtained as can be seen in the following table:

Kernel	**Accuracy (%)**
Radial	13.4
Linear	86.5
Polynomial	90.4

From the previous table it can be clearly seen that the Polynomial kernel has achieved the best accuracy, with 90.4%. These results have also been plotted with respect to the cost in Figure (6).

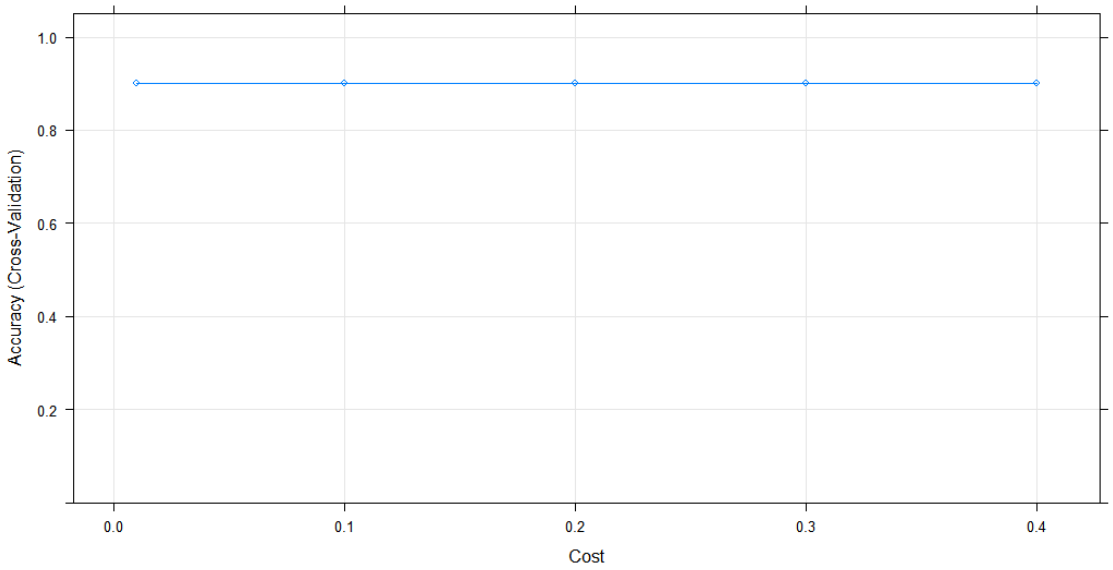

Figure 6: Polynomial kernel with degree = 2, scale = 0.1

2.3 All in

The *All in* data consists of 10 people extracted from the whole handwritten digits dataset.

At first, the Polynomial kernel was used on this *all in* dataset without any pre-processing in subsection 2.3.1. After which, the test data was pre-processed trough PCA and results can be seen in the figures in subsection 2.3.2 below.

2.3.1 without PCA

As first test without PCA was performed. Achieved Accuracy was around 91.1

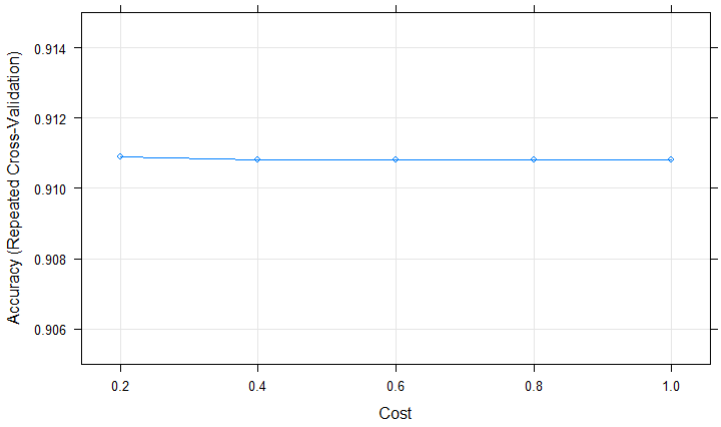

Figure 7: All in data without PCA pre-processing

2.3.2 with PCA

As a second test with the All in data, PCA was implemented to pre-process the data and reduce its dimensionality. As can be seen in Figure (8), the accuracy reached 91.7%, which is a slightly better result comparing to test without PCA –although the results are pretty close in both cases.

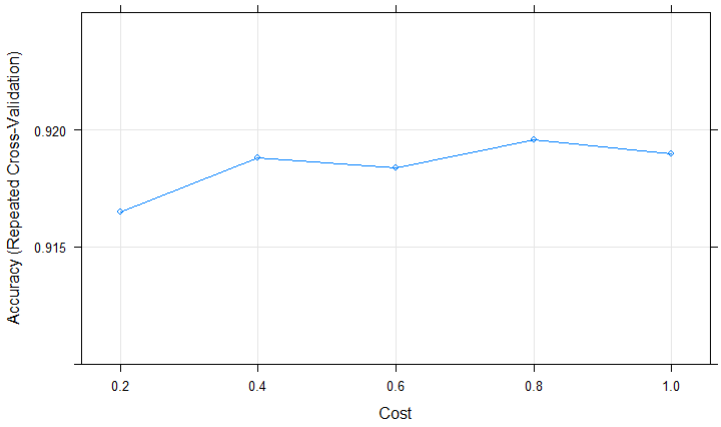

Figure 8: All in data with implemented PCA

2.4 Disjunct

Similarly to the tests performed on the *All in* data, we repeated the steps with the performed the *Disjunct* data.

2.4.1 without PCA

As can be seen in Figure (9) the best result with the SVM approach was achieved on the disjunct training dataset without PCA, with an accuracy of 96%.

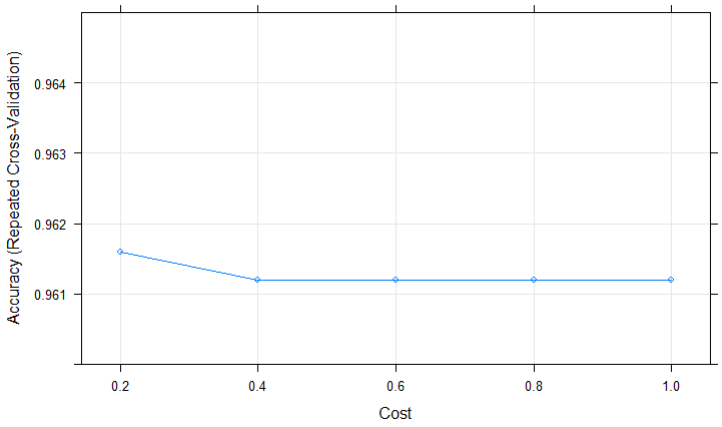

Figure 9: Disjunct data without PCA

2.4.2 with PCA

Finally, the results achieved on the dosjunct training dataset with the implemented PCA pre-processing method achieved an accuracy of 95% and can be seen in the figure below.

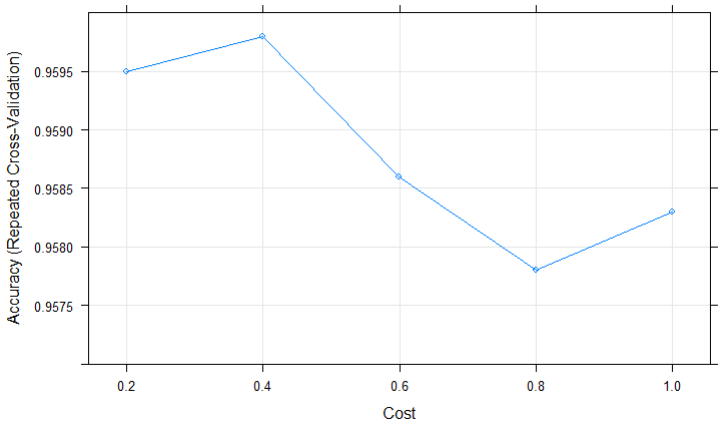

Figure 10: Disjunct data with implemented PCA

2.5 Conclusion on the best model from SVM

It can be concluded from the results of this section that for our particular used dataset the best accuracy is obtained with the disjunct data without PCA dimension reduction, with an accuracy of 96% as was shown in subsection 2.4.1. Parameters of the best model: $kernel = polydot$ $C = 0.2$, $degree = 2$, $scale = 0.5$

An interesting thing to note was that with the *all in* dataset, SVM performed better after PCA pre-processing even though the opposite is expected since PCA reducing dimensions which comes with data loss as well as expected lower accuracy.

3 Neural Networks

An Artificial Neural Network (or ANN) models the relationship between a set of input signals and an output signal using a model derived from our understanding of how a biological brain responds to stimuli from sensory inputs. Just as a brain uses a network of interconnected cells called neurons to create a massive parallel processor, ANN uses a network of artificial neurons or nodes to solve learning problems.

For the purpose of our project, we used ANNs for handwritten digit recognition application. This section will be structured in two different sets: first with the *all in* dataset with and without pre-processing, then with the *disjunct* dataset with and without processing. In each section, we compare the results of an MLP with two and three hidden layers in order to choose the best model.

The results are presented such that the accuracy of each layer of the MLP is plotted for the test and training data on the x axis.

For the two-layer neural network, we tried three different sets of pair of nodes presented in table 1.

name	(hidden layer 1, hidden layer 2)
Model 1	(40, 20)
Model 2	(30, 20)
Model 3	(30, 14)

Table 1: Architecture of checked MLP models with 2 hidden layers.

Same way of describing the model applies to the tree-layer neural network presented in table 2

name	(hidden layer 1, hidden layer 2, hidden layer 3)
Model 1	(80, 40, 20)
Model 2	(60, 40, 20)
Model 3	(50, 30, 14)

Table 2: Architecture of checked MLP models with 3 hidden layers.

For both of these neural nets, we used: 300 maximum iterations to learn, the $Tanh$ activation function with $Std\_Backpropagation$ used as a learning function with a gradient descent of 0.01 and a maximal difference between target value and output value tolerated of 0.

3.1 All in

3.1.1 without PCA

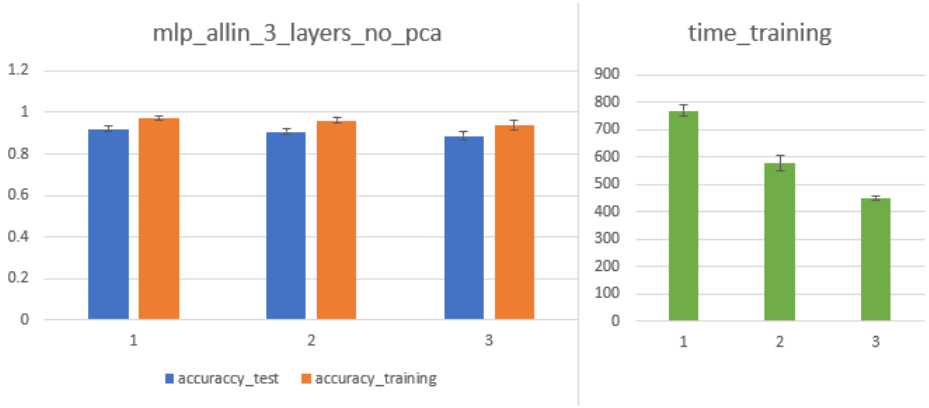

Figure 11: All In, 3 layers –No PCA

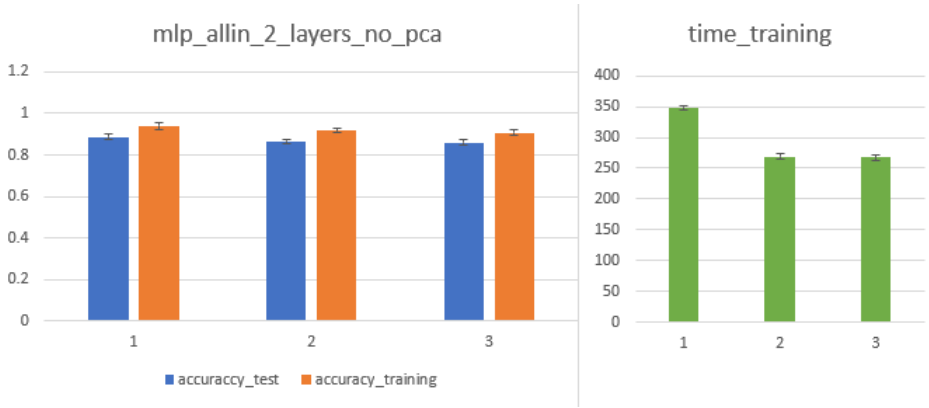

Figure 12: All In, 2 layers –No PCA

Comparing the results of the MLP with 2 and 3 hidden layers respectively, we can clearly see that the accuracy of the classification is high (reaching the 90s for both models) and the best accuracy was achieved by the first model with three layers with a test accuracy of 92% as can be observed in the figure 11.

It's worth noting that there is little difference between accuracy when more neurons are used, but the time of training grows significantly with more neurons in the network.

3.1.2 with PCA

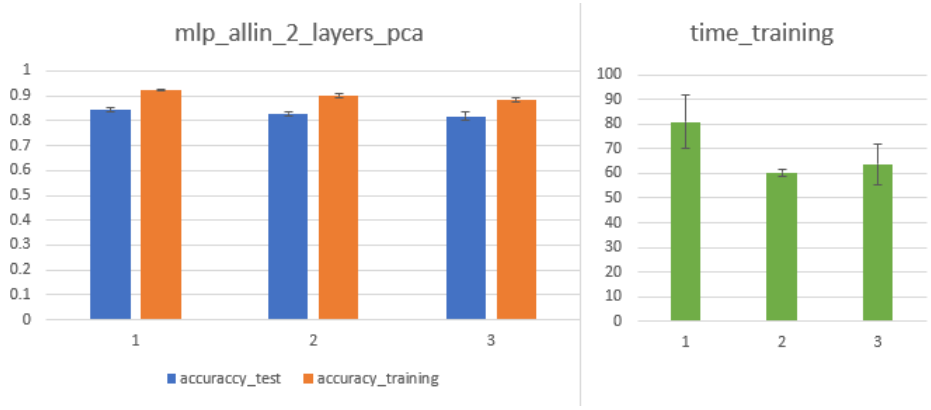

Figure 13: All In, 2 layers –with PCA

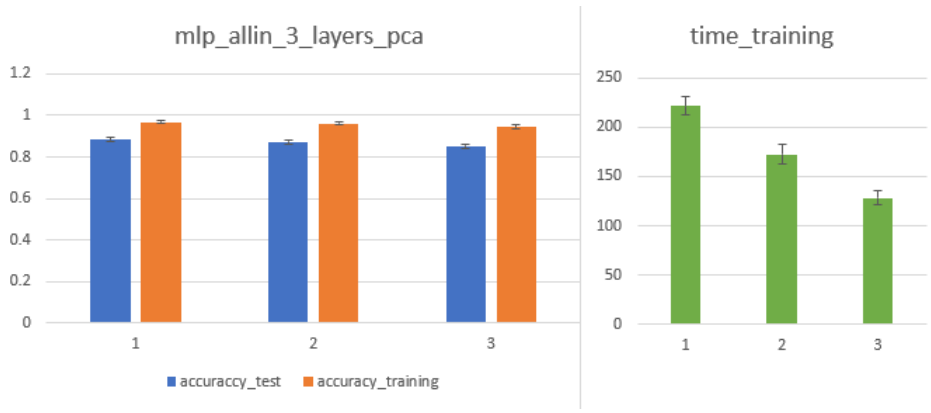

Figure 14: All In, 3 layers –with PCA

As expected, the accuracy drops with PCA since we lose dimensions which comes with data loss hence lower accuracy. Nevertheless from the figures 13 and 14, we can see that the best model accuracy is achieved with the three-layer MLP with the first set of nodes, at around 87%.

3.2 Disjunct

3.2.1 without PCA

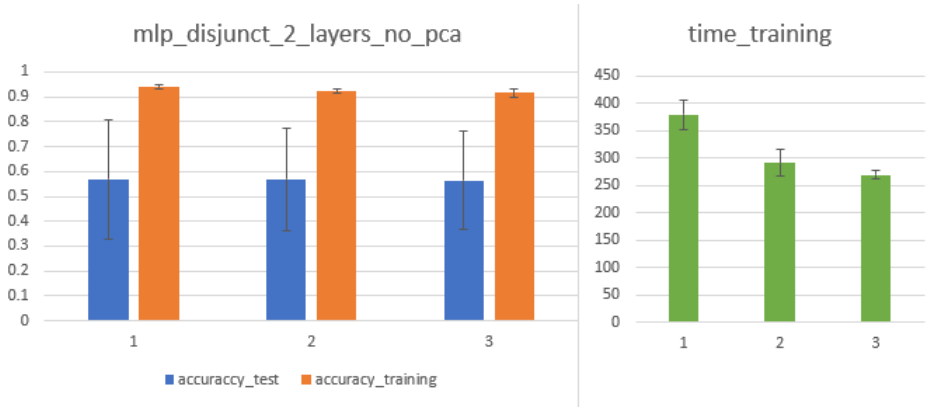

Figure 15: Disjunct, 2 layers –No PCA

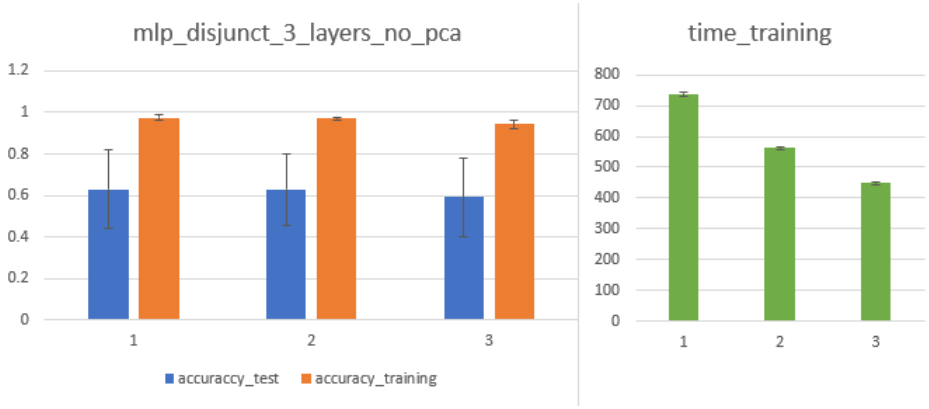

Figure 16: Disjunct, 3 layers –No PCA

Comparing the results of the 2 and 3 hidden layers used in MLP we can clearly see that the accuracy of classification do not differ as much in this case as they do with the "all persons in" data. This goes against what is expected, given that due to the nature of the disjunct dataset where, for cross validation only one person's handwriting is used as a training set, high variance in the accuracy test results is very much anticipated.

That being said, the best accuracy was achieved by models 1 and 2 at around 60% and can be observed in the figure 16.

3.2.2 with PCA

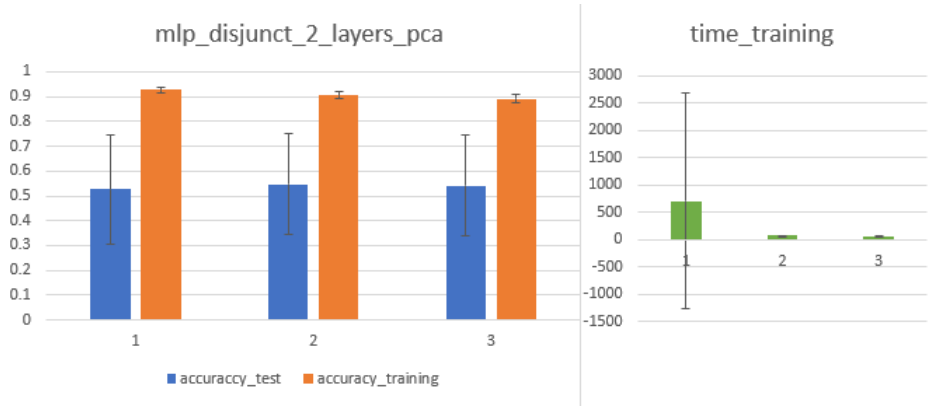

Figure 17: Disjunct, 2 layers –with PCA

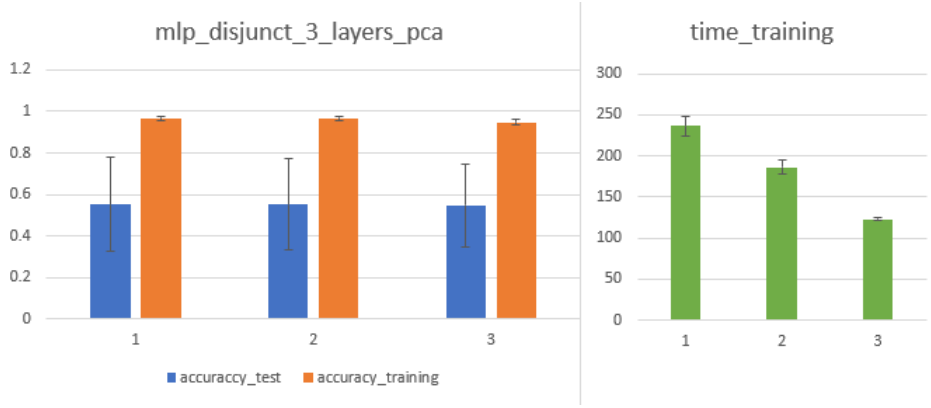

Figure 18: Disjunct, 3 layers –with PCA

Following the same reasoning as before, we note a drop in the accuracy obtained for both the 2 and 3 layers MLP from the PCA pre-processing with the highest performance barely reaching the 50% mark for all 6 models.

3.3 Conclusion on the best model from MLP

From the cross validation plots presented in the previous subsections, we can conclude that the best performing model out of all the combinations tried, was the 3-layers Model 1 with (80, 60 and 40) nodes in its respective hidden layers, without PCA and an all-in persons dataset. This model reached a test accuracy of 92% as observed in Fig. 11.

Another important thing to note is that seemingly all the plots generated with the disjunct dataset seem to be overfitting, whether we used pre-processing or not. Unfortunately, given the run-time needed to try each 2 and 3 hidden layers-MLP (multiple hours for each combination), we couldn't test out bigger data sizes to ensure this wasn't the main issue in our results.

4 Comparison of SVM and MLP

Using the best models from each of the two methods, we conducted a comparative analysis on the all-in and disjunct populations using data from scanned sheets from 20 people (505/50 split) as shown in the figures below.

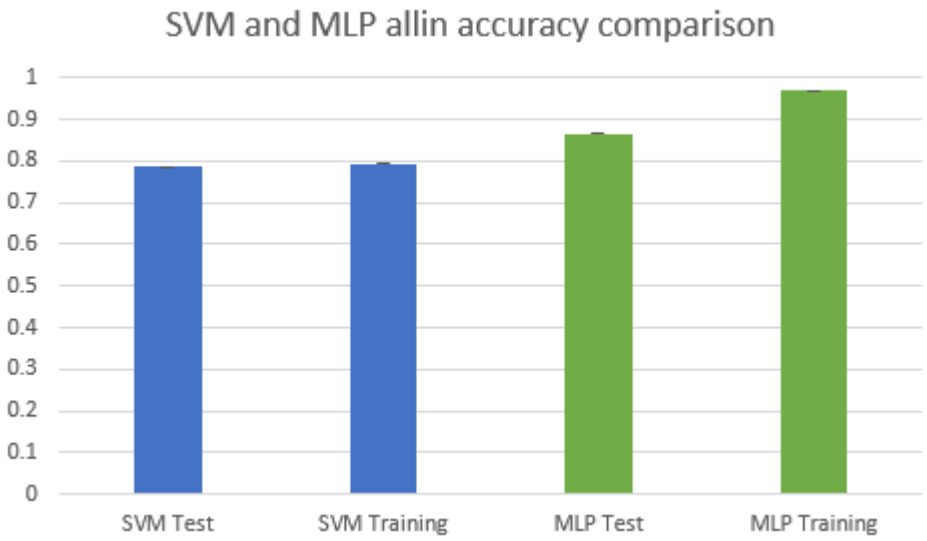

Figure 19: Comparison of accuracy of SVM and MLP method on all in dataset obtained from 20 people and split 50/50.

From the fig. 19, we can clearly see that the best model by far is the ANN reaching almost 86% of accuracy whereas the SVM is at about 80%.

However, it's interesting to note that there is little difference between the SVM's training and testing accuracy whereas the MLP shows a slight increase in performance on the training data. This is reflective of our findings in section 3 in which the neural network seemed prone to overfitting.

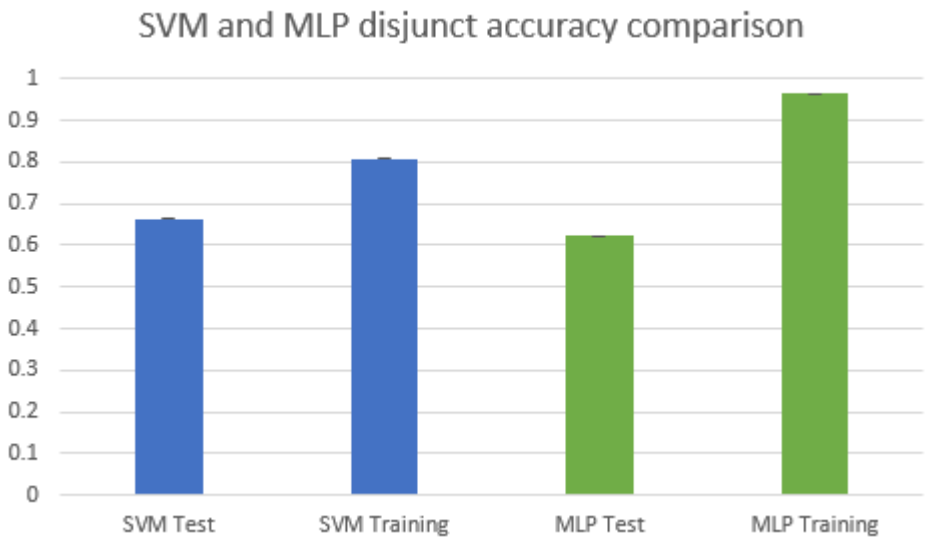

Figure 20: Comparison of accuracy of SVM and MLP method on disjunct dataset obtained from 20 people and split 50/50.

In Fig. 20, we observe the same overfitting that we noted in section 3 so it wasn't much of a surprise to see the SVM performing better than the neural network, at around 65%. Although, it's worth noting that both methods seem to under-perform with the disjunct data overall.

5 Conclusions

The purpose of this work was to compare two of the methods taught in our Statistical Machine Learning class this semester, and highlight their advantages and shortcomings in the application of a handwritten digit recognition system.

Overall, we found in section 3 that the best model using an ANN was a 3 hidden layers one without PCA, which make sense with respect to linear relationship between the depth of a neural net and its increase in performance.

Whereas with the the SVM approach, as explained in 2, we found that the best model was the one using a polynomial kernel with 2 degrees and a scale of 0.5, with PCA pre-processing.

In our comparative analysis of two methods, we found that the MLP performed better than the SVM with the *all-in* data-set at around 80%, and the opposite was true with the *disjunct* data-set where the SVM reached the best accuracy at around 65%.

All in all, our approach would have benefited from a larger data-set to inspect. Nevertheless results obtained show that those methods can be applied in real life to solve classification problems. The results might be even better if higher variance of PCA was used to reduce the dimensions, which was sacrificed to get the faster computational time.

References

[1] G. James, D. Witten, T. Hastie, R. Tibshirani, and M. T. Hastie, *An Introduction to Statistical Learning with Applications in R*. 2013.