**MACHINE LEARNING FROM DATA**

**Report: Lab Session 4 – K-Nearest Neighbors and Parzen windows**

**Names: Abdullah Qureshi, S M Rakib Hasan and Niina Hietamäki**

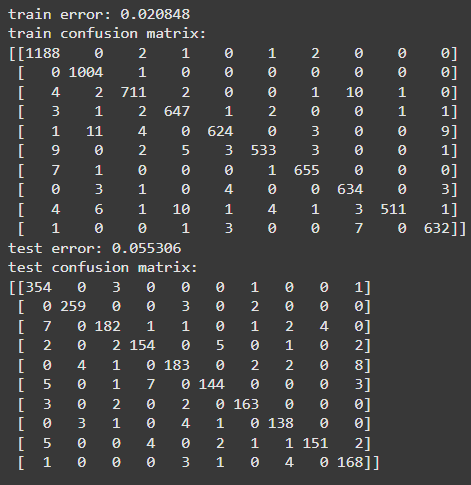
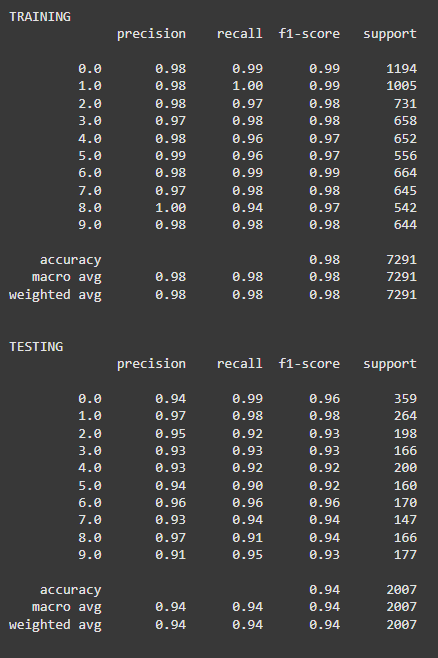
**Instructions**

* Download and uncompress the file **Mlearn\_Lab4.zip**
* Answer the questions in the document **Mlearn\_Lab4\_report\_surname.docx,** save and convert to pdf
* Write the new code in the same Colab Notebook **Mlearn\_lab4\_knn\_parzen.ipynb**

**Questions**

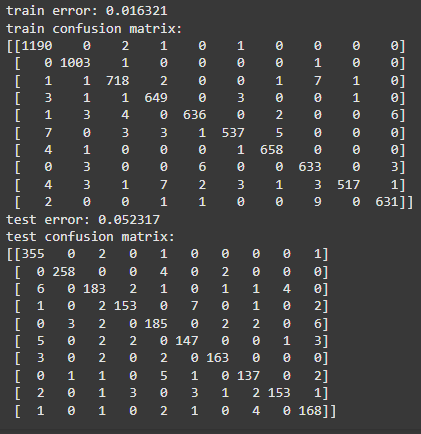
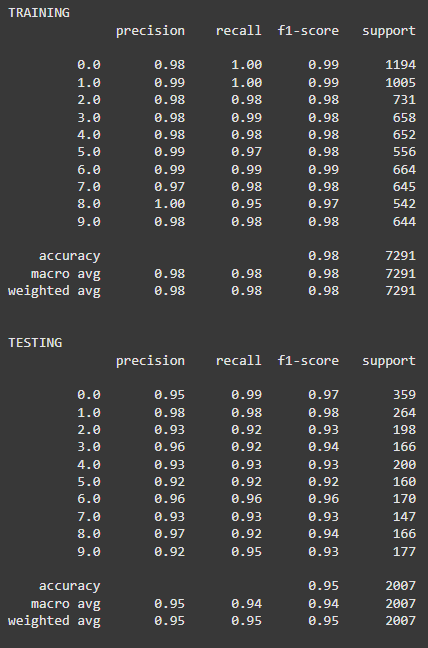
Q1. Copy the results obtained with kNN on the train and test sets, and discuss the results. What is the value of *k* (default)? Analyze the confusion matrices and identify the two most challenging classes.

KNeighborsClassifier uses 5 by default. Based on the results, the kNN classifier has a training error of 2,08% and a test error of 5.53%. This shows that the model generalized fairly well. From the test confusion matrix, it is observed that digits 3 and 5 are most frequently confused with each other, possibly due to their visual similarity. The overall accuracy on the test set is 94%, indicating that the model is quite accurate but still prone to errors with certain digits.



Q2. Run again the script, using PCA to reduce the dimensionality of the feature space, selecting d’=64 features. Observe the eigenvectors and the images reconstructed using only the first d’ eigenvectors (those with the highest eigenvalues). Discuss. Copy train and test results. Discuss the results and compare with the previous case (no PCA).

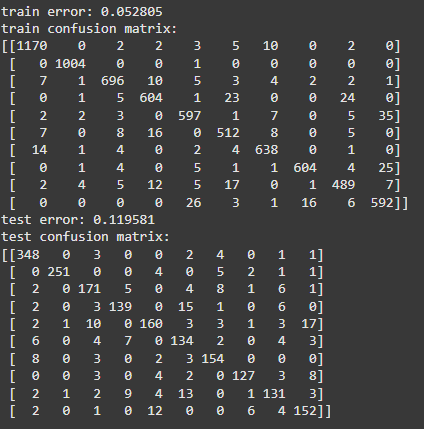
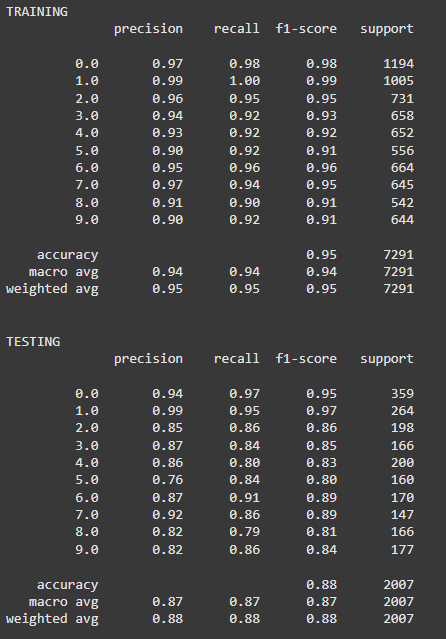
Using PCA with 64 features reduced the test error slightly from 5.53% (no PCA) to 5.23%. This means that PCA helped the model make fewer mistakes. The model also runs faster with PCA because it uses fewer features. The images reconstructed with PCA keep the main shapes, but some details are missing. Overall, PCA keeps the model accurate and makes it more efficient.



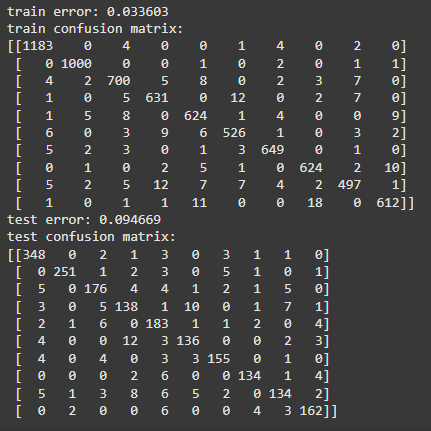
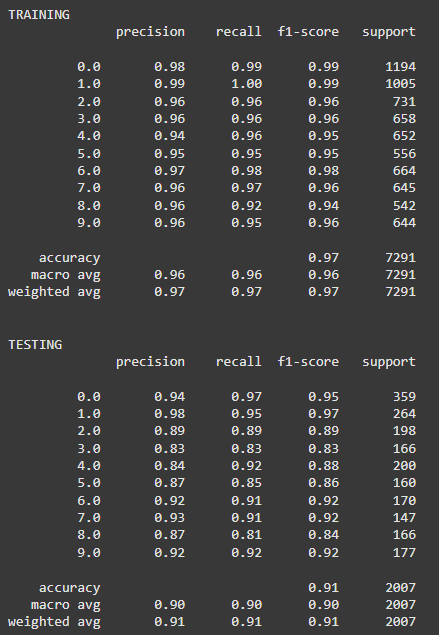
Q3. Repeat the previous analysis using PCA with d’=9 features, and MDA with d’=9 features. Discuss which method is the best for image reconstruction and which one is preferable for classification.

Using only 9 components, PCA keeps the main shapes of the images, but a lot of details are lost because it focuses on capturing the overall variance. MDA isn’t really made for image reconstruction, so it doesn’t keep the details as well as PCA. But when it comes to classifying the images, MDA does a better job because it’s focused on separating the classes, which makes it more accurate, as seen in the lower test error. So, PCA is better if you want to reconstruct images, while MDA is better for classification.

PCA with d’=9 features



MDA with d’=9 features



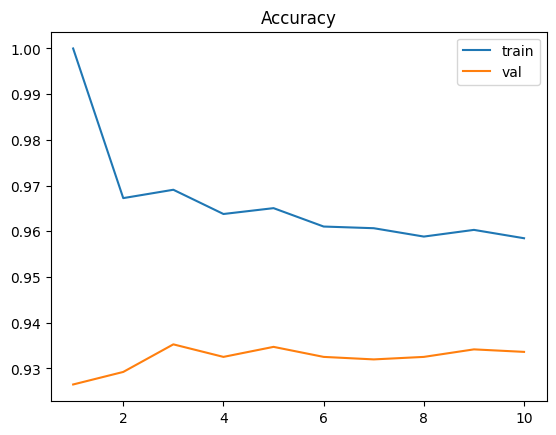
Q4. Find the optimal value of *k* on the training set. Use at least 10 values for *k*. Plot the train and validation errors as a function of *k*. Use the optimal value of *k* to compute the error on the test set. Discuss the results.

Using the values for K

Ks = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

The optimal value for K is 3 using GridSearch CV

{'clf\_\_n\_neighbors': 3}

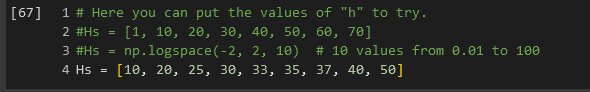


The test error on K is **0.101644**

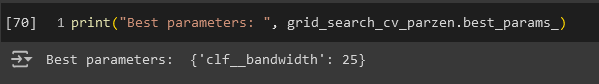
The results indicate that the optimal k for the KNN classifier is 3, as determined by GridSearchCV. At k=3, the model achieves a test error of 0.1016 which corresponds to an accuracy of 89.83%. This test performance suggests that the model generalizes reasonably well to unseen data. The gap between training accuracy (around 96%-97%) and validation accuracy (around 93%-94%) suggests some degree of overfitting particularly for lower values of k like k=1. By choosing k=3, the model manages to reduce overfitting while maintaining good performance on both training and validation sets.

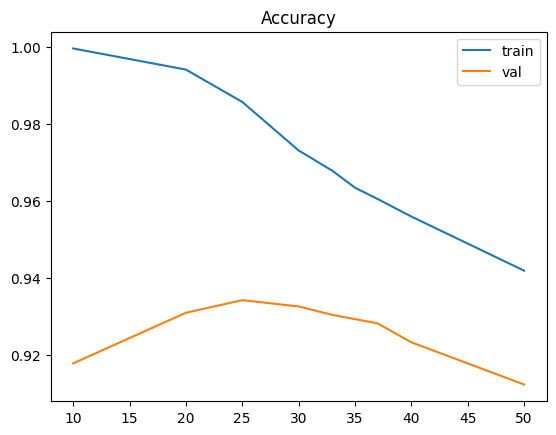
Q5. Use the same gridsearchcv strategy (with a single stratified split) to select the best parameter *h*. Try different values of *h* (for example 1, 10, 20, 30…. try more values). Plot the train and validation errors as a function of *h*. Use the optimal value of *h* to compute the error and other metrics on the test set. Discuss the results.

Values tried for Hs,



Through trial and error using multiple sets, the optimal value for H, 25, is found using GridSearch CV.

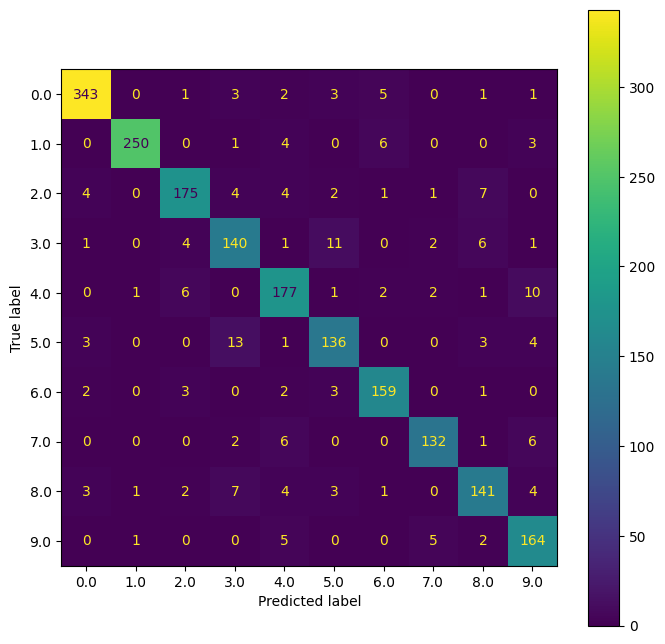


The training and validation curve of GridsearchCV-  


The test error obtained-

Test error: **0.094669**

Test confusion matrix-

****

From the GridSearchCV Plot, Training accuracy decreases as H increases, indicating reduced overfitting with larger bandwidths. The accuracy on both training and validation sets decreases for larger H, as shown in the plot, indicating that a smoother (wider) kernel reduces overfitting but may lose some critical details in the data. Validation accuracy peaks at H=25, indicating an optimal balance between bias and variance. The Test error = 0.094669 (90.53% accuracy), indicates good generalization to the test set.

High values along the diagonal of the Confusion matrix show that most classes are well-classified.