Exercise Sheet 6

1. Exercise: k-Nearest Neighbors Classification (MNIST)

(a) Load the digits dataset provided by the scikit-learn (sklearn) library and generate the following subsets. Note that each subset consists of data samples and their corresponding labels.

(i) `train_data`: The training subset of the datasest makes up 70% of the original dataset. This is the largest portion of the dataset and it's used directly for training the model.

(ii) `test_data`: The test subset of the datasest makes up 20% of the original dataset. The test dataset is a subset used to provide an unbiased evaluation of a final trained model. It is used used to verify the accuracy of the model built.

(iii) `val_data`: The validation subset of the datasest makes up 10% of the original dataset. The validation dataset is used to tune the parameters (hyperparameters) of a classifier.

(b) Write a Python function called `evaluate(y_pred, y_gt)` to determine the accuracy of a classification model's predictions. The function computes the accuracy as the proportion of correct predictions over the total number of predictions.

(c) Implement the t-SNE algorithm to reduce the data dimensionality down to four dimensions ($f_{\text{t-SNE}} : \mathbb{R}^{64} \rightarrow \mathbb{R}^{4}$). Apply this procedure for the training, test, and validation datasets.

(d) Implement a k-NN classifier as outlined in Exercise Sheet 4, then use the dimensionally reduced training dataset to train this k-NN algorithm. The k-NN classifier is applied to classify the handwritten digits (0-9) into their respective categories. Utilize the validation dataset to identify the most suitable hyperparameter k from $k \in \{1, 4, 8, 16\}$. Provide a table with the evaluation outcomes for each hyperparameter setting and clarify the choice of the hyperparameter you select as the most suitable one.

(e) Utilize the k-NN classifier, with the optimal hyperparameter k, on your test dataset and assess the performance of its predictions.

(f) Apply the t-SNE algorithm to reduce the data dimensionality of the original test dataset down to two dimensions ($f_{\text{t-SNE}} : \mathbb{R}^{64} \rightarrow \mathbb{R}^{2}$). Generate one figure with two distinct subplots that show the dimensionally reduced test dataset, using the predicted labels from the k-NN and the actual ground truth labels, respectively.

2. Exercise: Linear Classifier (Binary Classification)

Leverage the method of Least Squares to execute a task of binary classification. The classifier has the form

$$y(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{b} \tag{1}$$

with the decision rule

$$t(\boldsymbol{x}) = \begin{cases} 1 & \text{if } y(\boldsymbol{x}) \geq 0 \\ -1 & \text{otherwise} \end{cases}. \tag{2}$$

(a) Load the `classification.pkl` dataset provided in the Moodle course using the `pandas` library.

(b) Add a column of ones to the dataset features ($\tilde{\boldsymbol{X}} = [\boldsymbol{X}^{\mathrm{T}}\ \boldsymbol{1}]^{\mathrm{T}}$), thereby accounting for the bias term $\boldsymbol{b}$. This augmentation will be instrumental when you will carry out the matrix multiplication with the weight vector.

(c) Based on the class labeles, generate the ground truth vector $\boldsymbol{t}$.

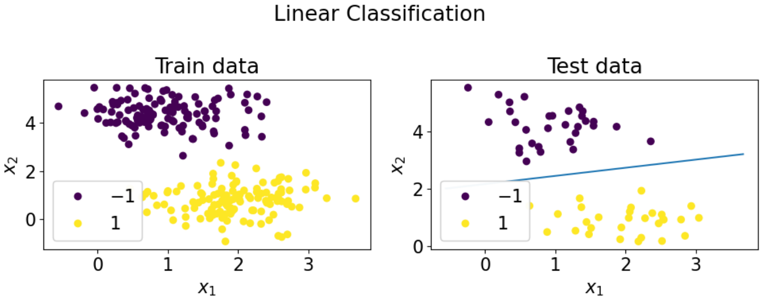
(d) Define a function named `fit` that accepts the input features matrix $\tilde{\boldsymbol{X}}$ and the target vector $\boldsymbol{t}$ as arguments and returns the weight vector $\boldsymbol{w}$. Remember that the formula for computing the weights $\boldsymbol{w}$ in a Least Squares setup is given by:

$$\boldsymbol{w} = (\tilde{\boldsymbol{X}}^{\mathrm{T}}\tilde{\boldsymbol{X}})^{-1}\tilde{\boldsymbol{X}}^{\mathrm{T}}\boldsymbol{t}. \tag{3}$$

(e) Define another function named `predict`, which accepts the input features matrix $\tilde{\boldsymbol{X}}$ and the weight vector $\boldsymbol{w}$ as arguments, and returns a prediction vector $\boldsymbol{t}_p$. The functions implements Eq. 1 and Eq. 2.

(f) Apply the function `fit` on the train data to determine the weight vector $\boldsymbol{w}$. Use the result to predict the labels of the test data using the function `predict`.

(g) Generate the following plot. The left side subplot shows the train samples with its labels and the right side subplot shows the predicted labels of the test samples as well as the decision boundary (blue line).


Linear Classification
Train data
Test data
x2
x1
● −1
● 1