**UE Machine Learning: Supervised Techniques**

Exercise 6 Report

Name: Omar Amr

Matrikel-Nr: k11776960

**Normalized Spectrum Kernel:**

The Normalized Spectrum Kernel depends on 2 main parameters. The first parameter is the length of sub-sequences which indicated the length of all possible substrings in a string while constructing the kernel matrix. The other parameter is the C parameter which indicates how much the SVM should avoid misclassifications in the training sample.

During the training process, a normalized kernel matrix is created for each combination which is used to train the model.

In the algorithm, grid search is used to try different variations of both parameters together. The best parameters which achieves the minimum cross-validation error is chosen to train the model again after trying all combinations in order to guarantee that the best model is selected by the end of the training process.

The function “train\_spectrum\_model” in the algorithm can work in 3 different ways depending on the input of the function. Either the mode parameter of the function is defined as “best”, which in this case trains a model with the best accuracy. The second mode is “search” which searches for the best combination of parameters and applies the combination that has the minimum cross-validation error. The last mode is “normal” which in this case another argument length has to be defined in order to train the model depending on the input of the user. [Examples are shown as comments in the code]

The best spectrum model has the following parameters: C = 10, Subsequence Length = 3 with cross-validation error = 0.209, putting this kernel as the worst model of all the other kernels.

Related Functions in code:

1. train\_spectrum\_model(mode = "search") # search mode tests all parametrs to get the best model.
2. train\_spectrum\_model(mode = "normal", input\_length = 4) # Set subsequence length depending on user input.
3. train\_spectrum\_model(mode = "best") # best mode trains the model with best parameters directly without searching.
4. predict\_spectrum\_test\_sample(testing\_set[,1]) # For prediction

**Linear Kernel (with one-hot encoding):**

Linear kernel depends on only the value of C which is explained previously. In this case a loop of different values of C is used in order to determine the best value for the training. Eventually the best C value is determined based on the cross-validation error and a model is trained with the best C in order to guarantee that the best model is the final one.

The best Linear kernel achieved a cross validation error of 0.1385 using a C=100. This cross-validation error puts the linear kernel in the second place of the three models.

Related Functions in code:

1. train\_linear\_model() # Search for parameters and apply best at the end.
2. linear\_trained\_model<<- ksvm(encoding\_matrix,training\_labels,type="C-svc", C=100, kernel='vanilladot', cross=10) # Train best model directly.
3. predict\_linear\_test\_sample(testing\_set[,1])

**RBF Kernel (with one-hot encoding):**

RBF kernel depends on two values. The C parameter which is explained previously and a sigma parameter which determines the width of the Gaussian distribution. Grid search is also used here to determine what is the best combination of both parameters that achieves best the result.

The best RBF kernel achieved a cross validation error of 0.113, which puts it in the first place in terms of cross-validation error. The parameters that achieve best results are C = 10 and Sigma = 0.001

This model with the pre-mentioned values for C and Sigma is used to predict the labels of the unlabelled data dataset included.

Related Functions in code:

1. train\_rbf\_model() # Search for parameters and apply best at the end.
2. rbf\_trained\_model <<- ksvm(encoding\_matrix,training\_labels,type="C-svc",C=10,kpar=list(sigma=0.001), kernel='rbfdot', cross=50) # Best parameters directly.
3. predict\_rbf\_test\_sample(testing\_set[,1])