Tuning and Cross Validation

`01-par_tuning-minimal.R` is an example script for the tuning and cross validation for a neural network. To apply the same method to other models, you need to make a few small changes as described below.

In total you need 5 scripts, but most of them are only called indirectly and you never have to look at them.

make changes and work with:

- `00-1-kfold_cv.R`
- `cvlistevaluate.R`

only get called:

- `00-2-rep_cv.R`
- `Decompose_Dataset.R`
- `helperfunctions.R` (actually I only need it for nnet, maybe you don't even need it yourself)
- to load the known and unknown dataset after Data Cleaning `./data/known-unknown-data.RData`

Prepare Datasets

write your own script that prepares the data how you need it:

1. **Decompose dataset** (splits the whole data into the 4 subsets for 4 trainings)
2. for neural network the next step would be to prepare the dataset for training, bring it in the right **format** (make it numerical, normalize it) - I don't know if you need such a step as well
3. give the **correct name**:

   ```
   known   <- training set
   unknown <- testing set
   ```

   `known` and `unknown` are the input of the next script that you call for training

Training

the next step is to do a **m times repeated, k-fold cross validation**

1. **choose settings** for cross validation and training, change "size" and "decay" to whatever parameters you want to tune on

   ```
   # settings for tuning and cross validation
   m <- 6 # repeated Cross Validation (same algorithm but different random split)
   k <- 3 # 3-fold cross validation
   parameters <- expand.grid("size" = seq(from = 3, to = 5, by = 2),
                             "decay" = c(0.01, 1))
   ```

2. **change the lines in `00-1-kfold_cv.R`** (from line 41), where the training actually happens

AND two times the command in line 29 and 32, add the package that you need for your model, you also add your package to the list at the beginning

```
[29] .packages = c("caret", "nnet", "pROC")
```

```
[41] # train nnet and make prediction
neunet <- nnet(return~. -order_item_id - tau, data = cv.train,
                                trace = FALSE, maxit = 1000,
                                size = parameters$size[n], decay =
parameters$decay[n])
yhat.val <- predict(neunet, newdata = cv.val, type = "raw")
```

3. **perform tuning/cv** (it's actually only one line!) `00-2-rep_cv.R` does the m times repeated cross validation. This script is very short and calls the `00-1-kfold_cv.R` script. But once you've changed the things from step 2, you don't have to do anything else anymore.

```
known   <- known.n                   # output of additional data preparation
unknown <- unknown.n                 #

# perform m-times repeatet k-fold cross validation
source(file="./nnet/00-2-rep_cv.R")
cv.list.i <- cv.list                 # store result of repeated cross validation
```

the output `cv.list` is a list of $m \cdot k \cdot \#settings$ measures (so far only AUC but can be extended to more measures) and the index of the settings that were used to make this prediction.

Evaluate the Results

`cvlistevaluate.R` is a crude code that evaluates the results from the training that were stored in `cv.list`. It combines the results of trainings that were made with the same settings ($m \cdot k$ times) and calculates the mean and variance.