Assignment 3.4

Read up on cross validation as a form of evaluation. What is done in cross validation? Consider especially leave-one-out cross validation and k-fold cross validation. Shortly explain the processes and their differences! For what purpose do we need either? How would a leave-one-out cross validation look in a concept learning setting?

Cross-validation (Rotation Estimation):

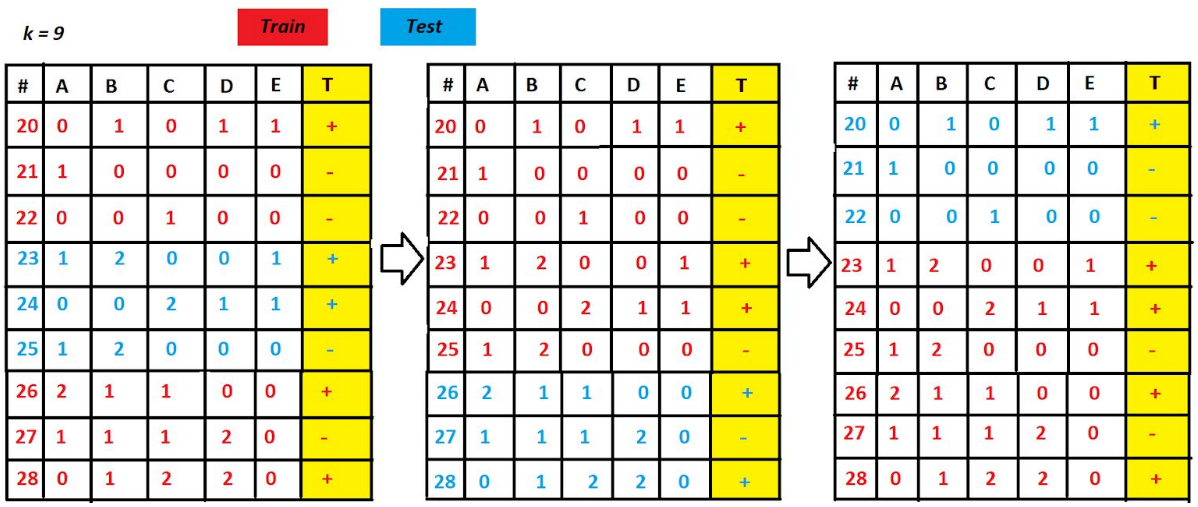
- It is a resampling procedure used to evaluate machine learning models on a limited data sample.
- The procedure has a single parameter called *k* that refers to the number of record groups that a given data sample is to be split into.

k-fold Cross-validation:

- In this flagship classifier evaluation method, the process of cross-validation is divided into *k*-folds.
- The value *k* is a hyperparameter and when a specific value for *k* is chosen it may be used in the reference to the model like *k=10 (optimal)* making it as 10-fold cross-validation.
- The larger the value of *k* higher the bias of the CV error and the lower variance.
- Process:
    - Shuffle the dataset randomly.
    - Split the dataset into *k* groups.
    - For each unique group perform:
        i. Take the group as a holdout or test data.
        ii. Take the rest of the group as the training data.
        iii. Fit a model on the training set and evaluate it on the test set.
        iv. Output the evaluation score.
    - Summarize the performance of the model using the mean of the scores.

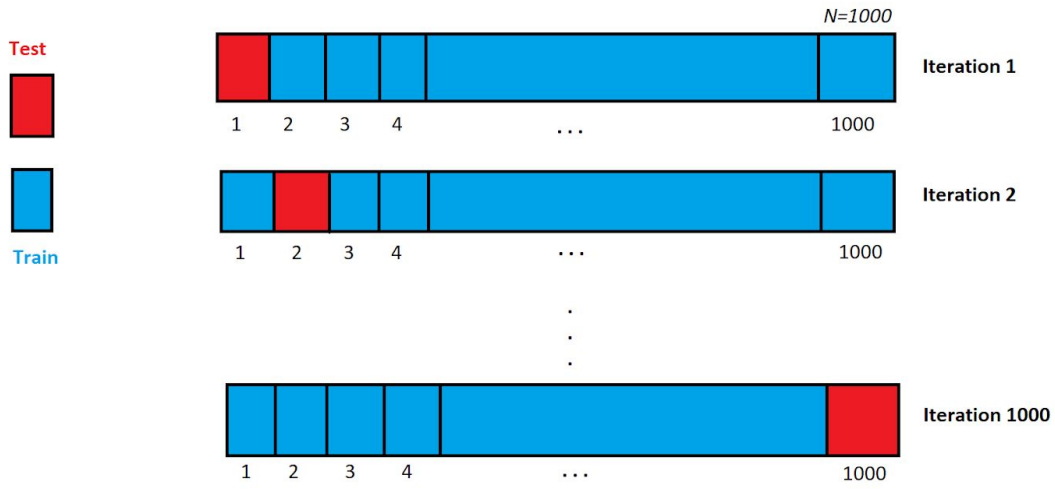
- Each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure.
- Meaning, each sample is given the chance to be used in the holdout set **once** and used to train the model ***k-1* times**.
- Purpose:

  Used to test the generalizability of the model.



Leave-One-Out Cross-Validation (LOOCV):

- It is a configuration of *k*-fold cross validation where *k* is set to the number of examples, *N* in the dataset (*k*=*N*).
- LOOCV is an extreme version of *k*-fold cross-validation that has the maximum computational cost.
- It requires one model to be created and evaluated for each example in the training dataset.
- Process:
  - Take 1 out of N as the test example (used for evaluation).
  - Rest as the training data (used to fit a model).
  - Choose a model (usually RandomForestClassifer with a *random_state*).
  - Save the score from each evaluation.
  - Present the final mean estimate as model performance.



- Purpose:

  The benefit of so many fit and evaluated models is a more robust estimate of model performance as each row of data is given an opportunity to represent the entirety of the test data.

- Recommended using when the size of the dataset is small or when estimated model performance is critical.

- Not recommended using when:

  I. The size of the dataset is humungous - Due to the fact of performing multiple evaluations per iterations.

  II. There are costlier models to fit.

  III. Prone to low bias - It performs as expected when training with the presented scenario and on the test data but, when made to evaluate on a new test example especially in a production setting then it underperforms.

LOOCV	*k*-fold CV
A special case of *k*-fold CV when $k = N$	Dataset grouped into chunks of *k* sizes ($k<N$)
N-1 for training; 1 out of *N* for test	*k-1* fold for training; 1-fold for test
Computationally expensive to perform	Computationally economic to perform
Convergence time is higher	Better convergence in comparison to LOOCV

LOOCV in a Concept Learning perspective

As concept learning can be viewed as a task of searching through a large set of hypotheses we can use it for determining whether a classifier evaluation strategy is apt or not.

- In LOOCV since we consider the entire dataset (*N*), it is directly proportional to the increase in the traces for *S* and *G,* the strategy would take computationally high time to converge.
- Depending upon the correct concept learning algorithm the learner is guaranteed to find the target concept.