Lecture 13: Validation

- Motivation
- The Holdout
- Re-sampling techniques
- Three-way data splits

Motivation

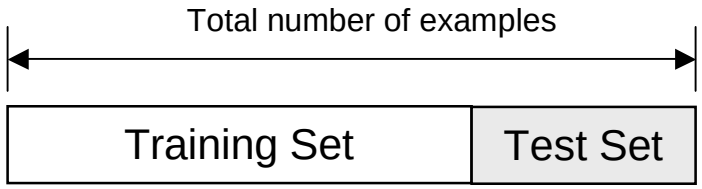
- **Validation techniques are motivated by two fundamental problems in pattern recognition: model selection and performance estimation**
- **Model selection**
  - Almost invariably, all pattern recognition techniques have one or more free parameters
    - The number of neighbors in a kNN classification rule
    - The network size, learning parameters and weights in MLPs
  - How do we select the “optimal” parameter(s) or model for a given classification problem?
- **Performance estimation**
  - Once we have chosen a model, how do we estimate its performance?
    - Performance is typically measured by the TRUE ERROR RATE, the classifier’s error rate on the ENTIRE POPULATION

Motivation

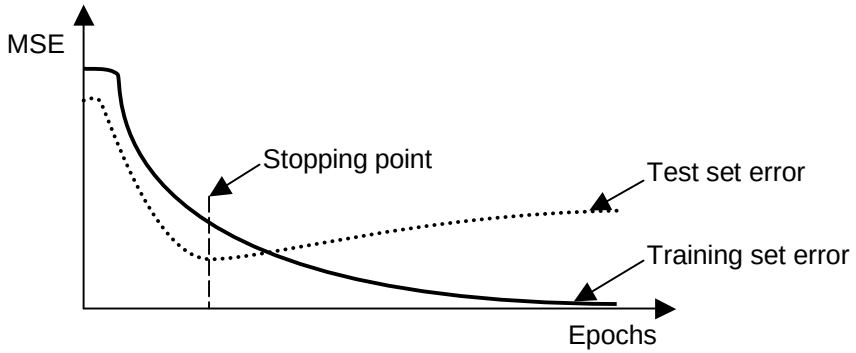
- **If we had access to an unlimited number of examples these questions have a straightforward answer**
  - Choose the model that provides the lowest error rate on the entire population and, of course, that error rate is the true error rate
- **In real applications we only have access to a finite set of examples, usually smaller than we wanted**
  - One approach is to use the entire training data to select our classifier and estimate the error rate
    - This naïve approach has two fundamental problems
      - The final model will normally overfit the training data
        - This problem is more pronounced with models that have a large number of parameters
      - The error rate estimate will be overly optimistic (lower than the true error rate)
        - In fact, it is not uncommon to have 100% correct classification on training data
  - A much better approach is to split the training data into disjoint subsets: the holdout method

The holdout method

- **Split dataset into two groups**
  - Training set: used to train the classifier
  - Test set: used to estimate the error rate of the trained classifier



- **A typical application the holdout method is determining a stopping point for the back propagation error**

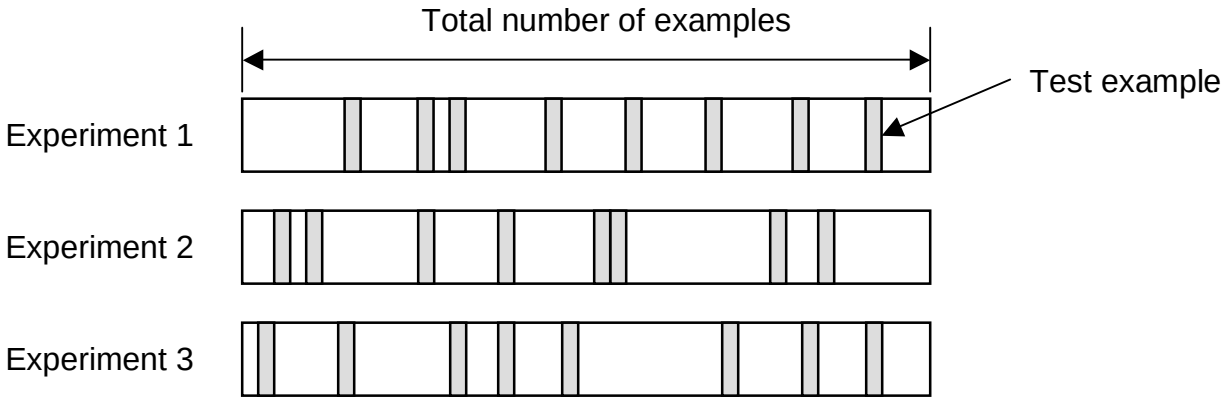


The holdout method

- **The holdout method has two basic drawbacks**
  - In problems where we have a sparse dataset we may not be able to afford the "luxury" of setting aside a portion of the dataset for testing
  - Since it is a single train-and-test experiment, the holdout estimate of error rate will be misleading if we happen to get an "unfortunate" split
- **The limitations of the holdout can be overcome with a family of resampling methods at the expense of more computations**
  - Cross Validation
    - Random Subsampling
    - K-Fold Cross-Validation
    - Leave-one-out Cross-Validation
  - Bootstrap

Random Subsampling

- **Random Subsampling performs K data splits of the dataset**
  - Each split randomly selects a (fixed) no. examples without replacement
  - For each data split we retrain the classifier from scratch with the training examples and estimate E_i with the test examples

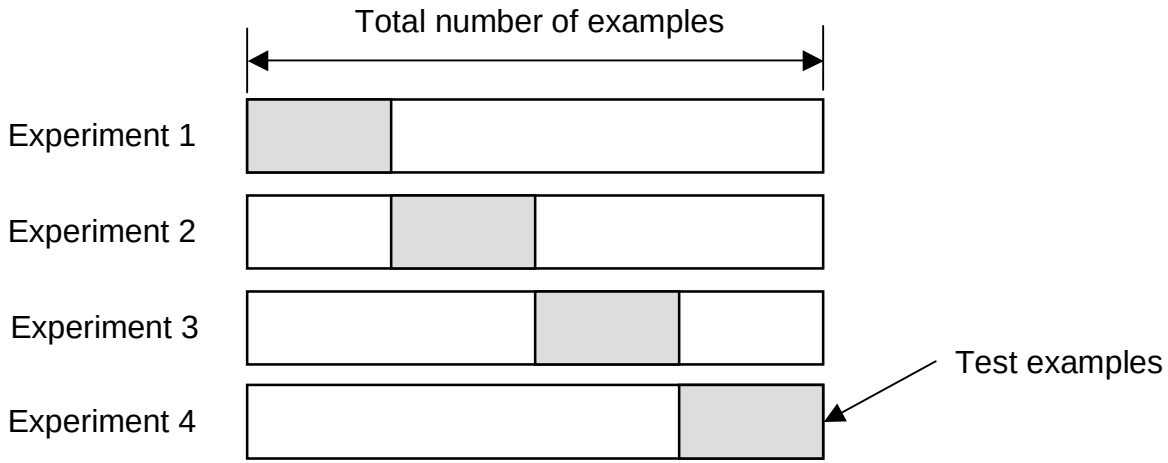


- **The true error estimate is obtained as the average of the separate estimates E_i**
  - This estimate is significantly better than the holdout estimate

$$E = \frac{1}{K}\sum_{i=1}^{K} E_i$$

K-Fold Cross-validation

- **Create a K-fold partition of the the dataset**
  - For each of K experiments, use K-1 folds for training and the remaining one for testing

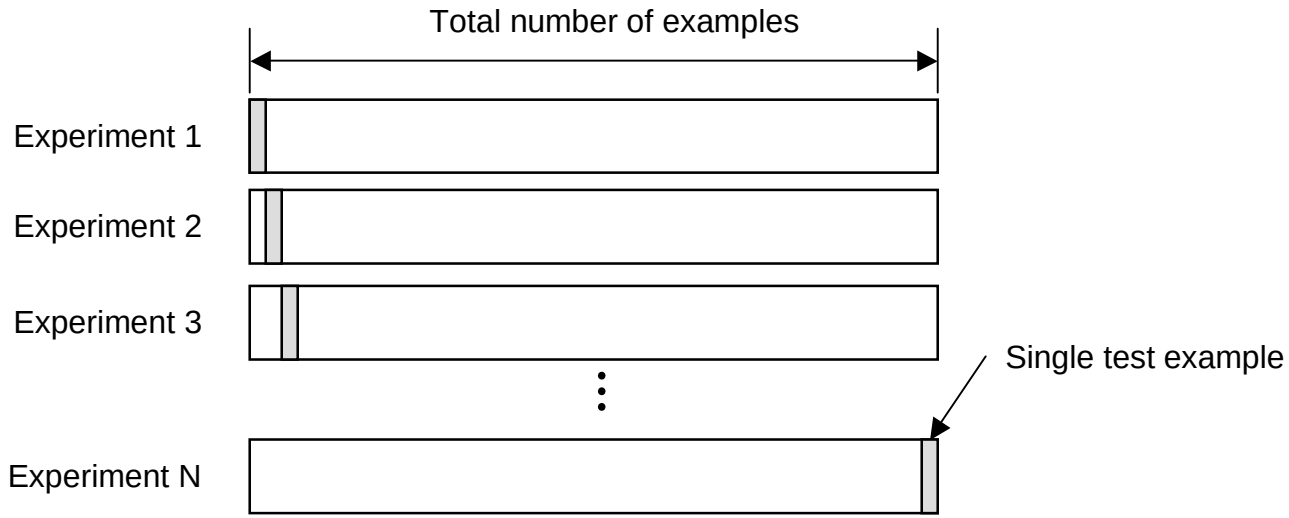


- **K-Fold Cross validation is similar to Random Subsampling**
  - The advantage of K-Fold Cross validation is that all the examples in the dataset are eventually used for both training and testing
- **As before, the true error is estimated as the average error rate**

$$E = \frac{1}{K}\sum_{i=1}^{K} E_i$$

Leave-one-out Cross Validation

- **Leave-one-out is the degenerate case of K-Fold Cross Validation, where K is chosen as the total number of examples**
  - For a dataset with N examples, perform N experiments
  - For each experiment use N-1 examples for training and the remaining example for testing



- **As usual, the true error is estimated as the average error rate on test examples**

$$E = \frac{1}{N}\sum_{i=1}^{N} E_i$$

How many folds are needed?

- **With a large number of folds**
  - \+ The bias of the true error rate estimator will be small (the estimator will be very accurate)
  - \- The variance of the true error rate estimator will be large
  - \- The computational time will be very large as well (many experiments)
- **With a small number of folds**
  - \+ The number of experiments and, therefore, computation time are reduced
  - \+ The variance of the estimator will be small
  - \- The bias of the estimator will be large (conservative or higher than the true error rate)
- **In practice, the choice of the number of folds depends on the size of the dataset**
  - For large datasets, even 3-Fold Cross Validation will be quite accurate
  - For very sparse datasets, we may have to use leave-one-out in order to train on as many examples as possible
- **A common choice for K-Fold Cross Validation is K=10**

Three-way data splits

- **If model selection and true error estimates are to be computed simultaneously, the data needs to be divided into three disjoint sets**
  - **Training set**: a set of examples used for learning: to fit the parameters of the classifier
    - In the MLP case, we would use the training set to find the "optimal" weights with the back-prop rule
  - **Validation set**: a set of examples used to tune the parameters of of a classifier
    - In the MLP case, we would use the validation set to find the "optimal" number of hidden units or determine a stopping point for the back propagation algorithm
  - **Test set**: a set of examples used only to assess the performance of a fully-trained classifier
    - In the MLP case, we would use the test to estimate the error rate after we have chosen the final model (MLP size and actual weights)
    - After assessing the final model with the test set, YOU MUST NOT further tune the model

Three-way data splits

- **Why separate test and validation sets?**
  - The error rate estimate of the final model on validation data will be biased (smaller than the true error rate) since the validation set is used to select the final model
  - After assessing the final model with the test set, YOU MUST NOT tune the model any further
- **Procedure outline**

1. Divide the available data into training, validation and test set
2. Select architecture and training parameters
3. Train the model using the training set
4. Evaluate the model using the validation set
5. Repeat steps 2 through 4 using different architectures and training parameters
6. Select the best model and train it using data from the training and validation set
7. Assess this final model using the test set

  - This outline assumes a holdout method
    - If CV or Bootstrap are used, steps 3 and 4 have to be repeated for each of the K folds

Three-way data splits