# Cross Validation

1. Generalization vs over-fitting

* Generalization: The ability of an algorithm to be effective across various inputs
* The performance of the ML model is constant across different datasets (with the same distribution as the train data)
* When a model performs well on the train set, but not on new / naïve data, the model over-fits to the training data

1. Training a ML model

* To prevent over-fitting, it is common practice to:
* Separate the data into a train and a test set
* Train the model on the train set
* Evaluate in the test set

1. Tuning hyperparams

* When evaluating different hyperparam spaces there is a risk of overfitting on the test set
* We select the best model based on performance over the test set
* Knowledge about the test set can ‘leak’ into the model -> lack or unknown generalization
* Common mistake in DS competitions

1. Another Hold-out sample

* Subsequently divide the train set into a train set and validation set
* Train model on most of train set
* Test performance on validation set
* Select best model
* Test best model’s performance on test set
* Problem: do we have a big enough dataset?
* We could be left out with very little data to train the model
* We have no metric of error
* Metric +/- error

1. Cross validation



* Train set divided into k folds
* Model trained in k-1 fold
* Model tested in the kth fold
* Repeat k times
* Final performance metric is the average
* Can determine an error (e.g., R squared = 0.8 +/- 0.1)

1. Hyperparams tuning with cross-val

* Hyperparams space 1,2,3 -> Cross-validation -> best hyperparam space

# Bias vs. Variance

(link: <https://www.youtube.com/watch?v=fDQkUN9yw44>)

# Cross-validation scheme

1. Schemes

* K-fold
* Leave one out (LOOCV)
* Leave p out (LPOCV)
* Repeated K-fold
* Stratified Cross-validation

1. Bias vs. Variance

* Underfitting – High bias
* Overfitting – High variance
* Complexity of the model: e.g., linear vs polynomial model
* Number of estimators in tree based algorithms, depth, etc.

1. Train set size vs. Bias (performance)

* Smaller datasets may lead to underfitted models

1. K-fold cross validation



* Divide train set into k folds (of equal size)
* Train model in k-1 folds
* Test model in kth fold
* Repeat k times -> train k models
* K performance values
* Final performance metric: mean +/- std
* Typical K is 5 or 10
* Higher K
  + Bigger train sets
  + Less model bias >< More variance
* No overlap of test sets in the different cross-validation rounds

1. Leave one out cross-validation

* K = n where n is the number of observations
* Divide train set into n folds
* Train model in n-1 fold
* Test model in nth observation
* Repeat n times -> train n models
* Final performance metric: mean +/- std
* Considerations
* Computationally expensive
* Models are almost identical as they are trained on practically the same training dataset -> high variance
* No overlap of test sets in the different folds
* Some metrics can’t be estimated when we only have 1 observation in the test sets (i.e., ROC-AUC, precision, recall)

1. Leave p out cross-validation

* Leaves out all possible subsets of p observations
* For n observations, this produces train-test pairs
* There is overlap between different test sets
* We have bigger validation sets -> better measure of performance (than LOOCV)
* Very computationally expensive

1. Repeated K-fold cross-validation

* Repeats K-fold cross-validation n times, each time making different data split
* The values of the training set are shuffled before making the split into the K fold
* Repeat n times: Shuffle data -> K-fold CV
* K \* n performance metrics
* There could be overlap between the tests sets in different repeats

1. Stratified K-fold Cross-validation

* Only for classification problem
* Procedure identical to K-fold cross-validation
* Ensures that each fold has a similar proportion of observations of each class
* Useful with (very) imbalanced datasets
* K performance metrics
* No overlap of test sets

1. Uses of cross-validation

* Estimate the generalization error of a given model
* Select best performing model from a group of models
* Different algorithms
* Different feature subsets
* Select hyperparams

1. Considerations

* Generally, use K-fold cross-validation with K equals 5 or 10
* Use stratified K-fold if target class is imbalanced
* If K is too small, the error estimate is pessimistically biased because of the difference in training set size between the original dataset and the cross-validation datasets
* Leave one out cross-validation works well for estimating continuous error functions (e.g., MSE), but it may perform poorly for discontinuous error functions (e.g., number of misclassified cases, precision and recall)

# Special Cross-validation schemes

1. Cross-validation schemes

* The cross-validation methods discussed so far assume that the data is **independent and identically distributed**
* If this is the case, a similar distribution of data is guaranteed in each fold of the cross-validation scheme
* Some data may not be independent and identically distributed
* **Grouped data** (data from the same subject)
* **Time series**
* These datasets require a tailored cross-validation scheme

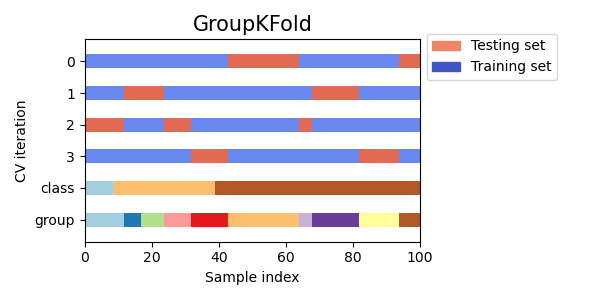
1. Grouped data

* Multiple observations come from the same subject
* Medical data collected from patients, with multiple samples taken from each patient
* Voice recognition of say, digits, where the digits are pronounced by various speakers
* We would like to know if a model trained on a particular set of groups **generalizes well to the unseen groups**
* To measure this, we need to ensure that all the samples in the validation fold come from groups that are **not represented at all** in the paired training fold

1. Grouped cross-validation

* Group K-fold CV
* Leave one group out CV
* Leave P groups out CV

1. Group K-fold CV



* Equivalent to K-fold cross validation
* Each subject is in a different testing fold
* Same subject is never in train and test fold at the same time
* The fold may be of different size

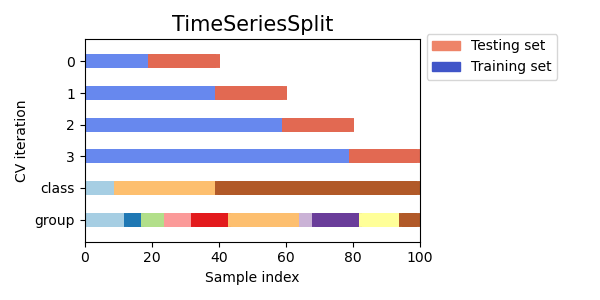
1. Leave one group out CV

* Equivalent to Leave one out CV
* Each subject is in a different testing fold
* We leave one subject at a time in the validation fold

1. Leave P Groups out CV

* Leaves out all possible combinations of p groups
* Similar to Leave p out CV

1. Cross validation for Time series



* We want to evaluate performance in ‘future’ observations
* Variation of K-fold cross-validation
* Returns first k folds as train set and the kth fold as test set
* Unlike standard CV methods, successive training sets are supersets of those that come before them

# Nested Cross validation

1. Uses of CV

* Estimate the generalization error of a given model
* Select best performing model from a group of models
* Different algorithms
* Different feature subsets
* Select hyperparams

1. Cross-validation & generalization error

* When we have 1 model
* Model selection, CV & generalization
* Validation tests are the same for all models
* Validation set ‘leaks’ info to model selection procedure
* Generalization error is optimistically biased
* We need a different test set to get an unbiased evaluation of the generalization error of the selected model (or selected hyperparams)
* Particularly important for DS competitions

1. Nested CV

* Outer loop – estimate generalization error
* Inner loop – select model, hyperparams

1. Considerations

* Computationally expensive
* Useful when we need a good estimation of the generalization error
* Different inner models may have different hyperparams, although it is expected to be among the top performing hyperparams