Model Assessment & Selection

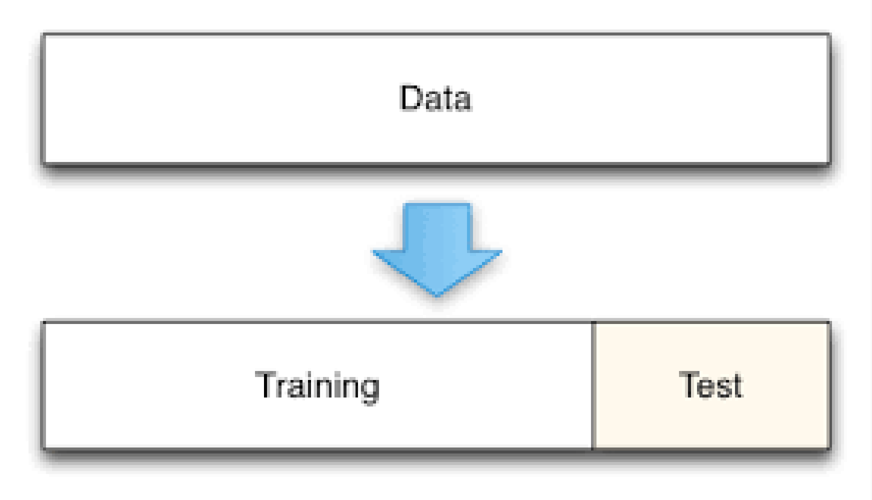
Week 6

Model Selection

Partitioning data for train and test
Mean square error
Finding the best hyper-parameters
Effects of Imbalanced class

Partitioning data for training and testing

- Reliable estimate of model performance (accuracy)
  - Training/test split of the data
    - Single
    - Multiple
  - Larger test set
  - Larger training set

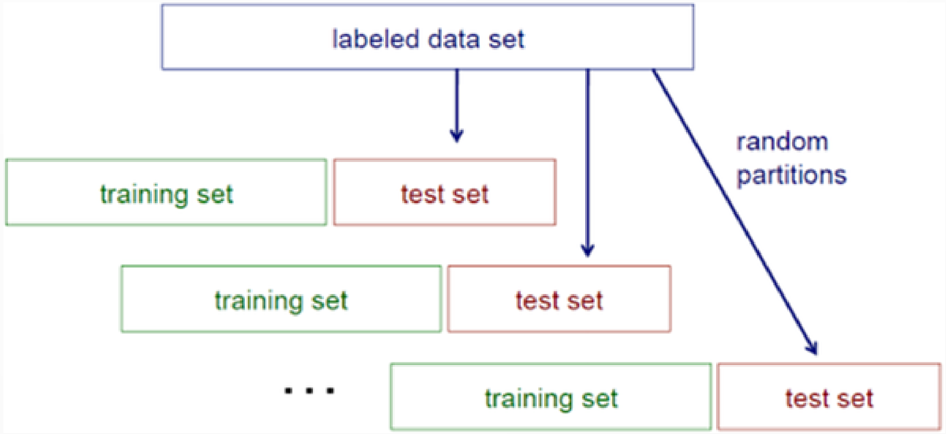


Partitioning data for training and testing

- Methods for splitting data:
  - Random sub-sampling
  - Stratified sampling
  - Cross validation

Partitioning data: Random sub-sampling

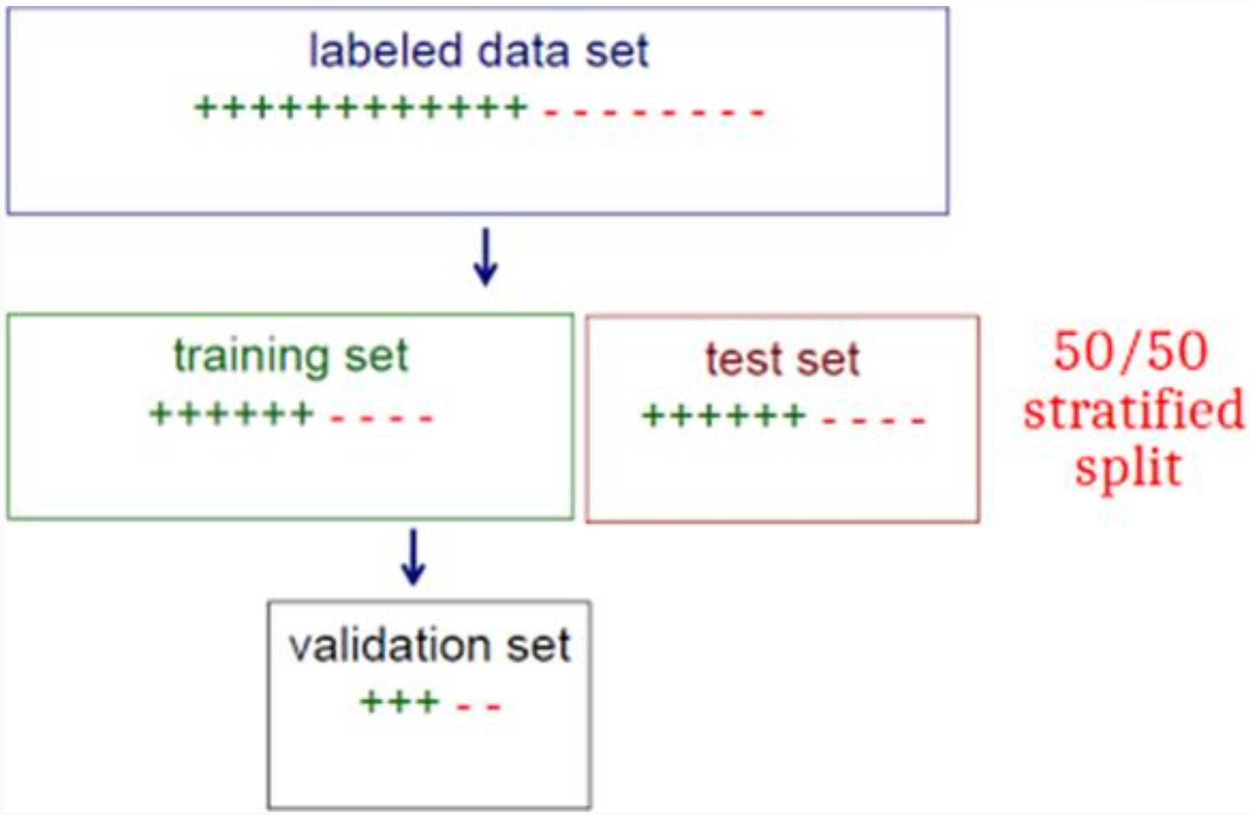
- A more reliable estimate of model performance can be obtained by random sub-sampling.
- Random sub-sampling
  - Partitions the data into random training and test sets in a specified ratio.



- For multiple splits / trials
  - Average the accuracies to get an averaged estimate.

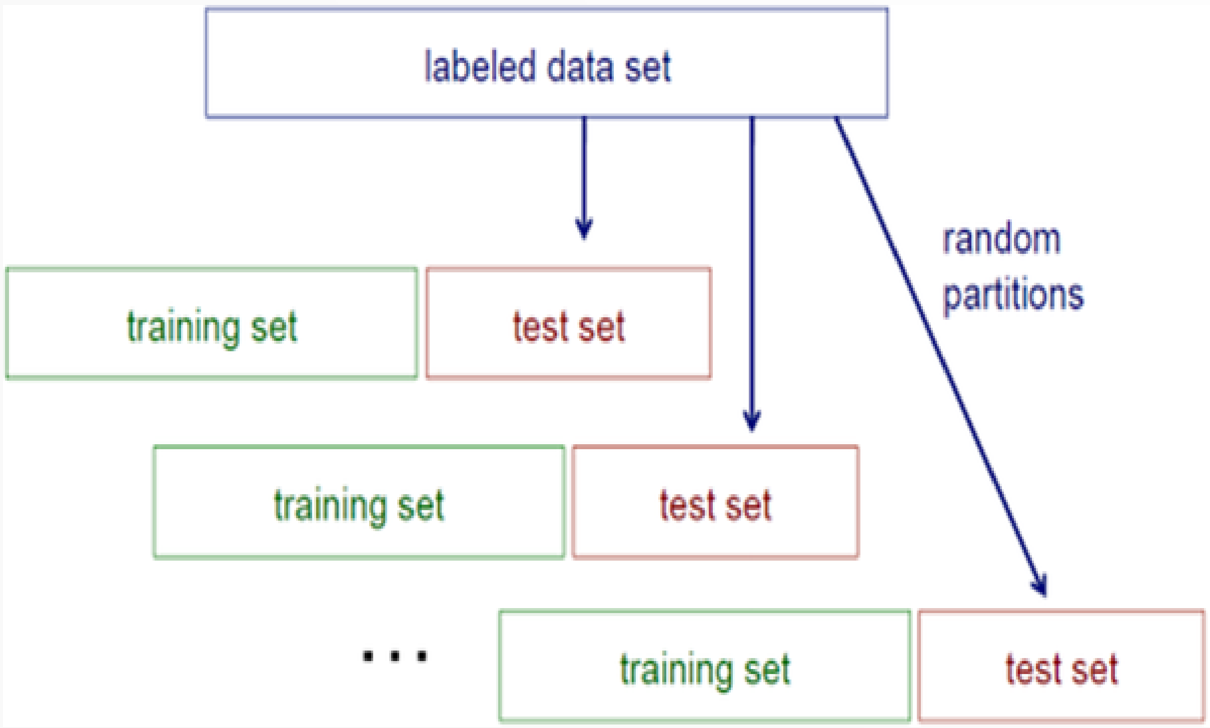
Partitioning data: Stratified Sampling

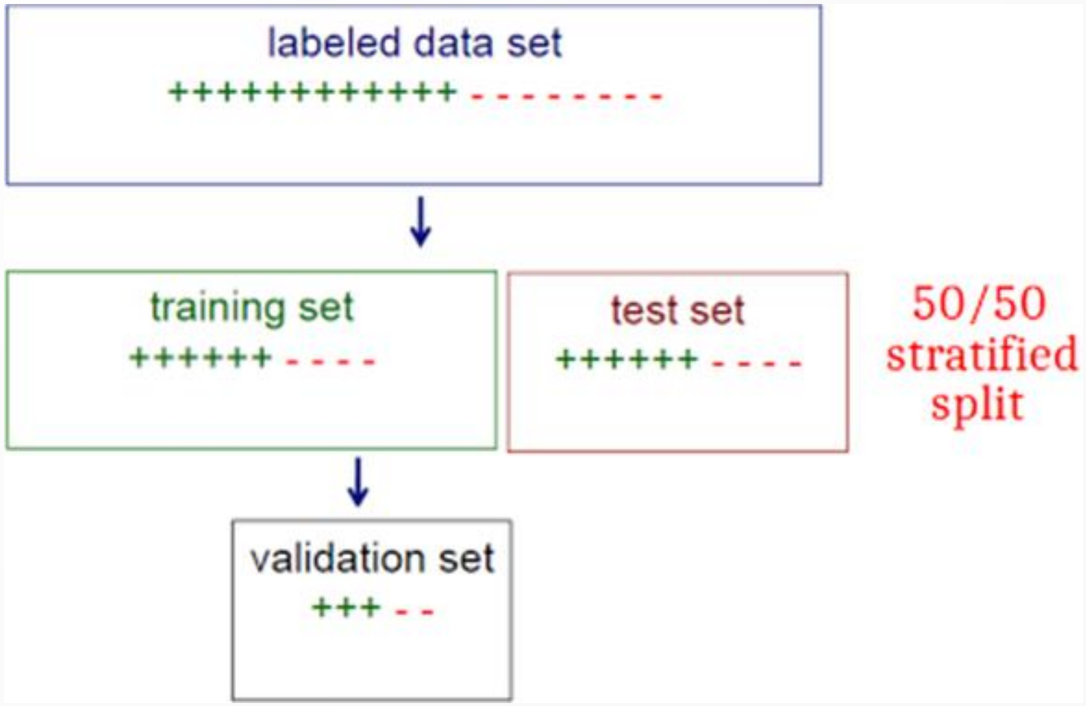
- Stratified sampling is a probability sampling technique
  - Divide the entire data into different subgroups or strata.
  - Randomly selects the samples proportionally from the different strata.



Partitioning data: Stratified Sampling...

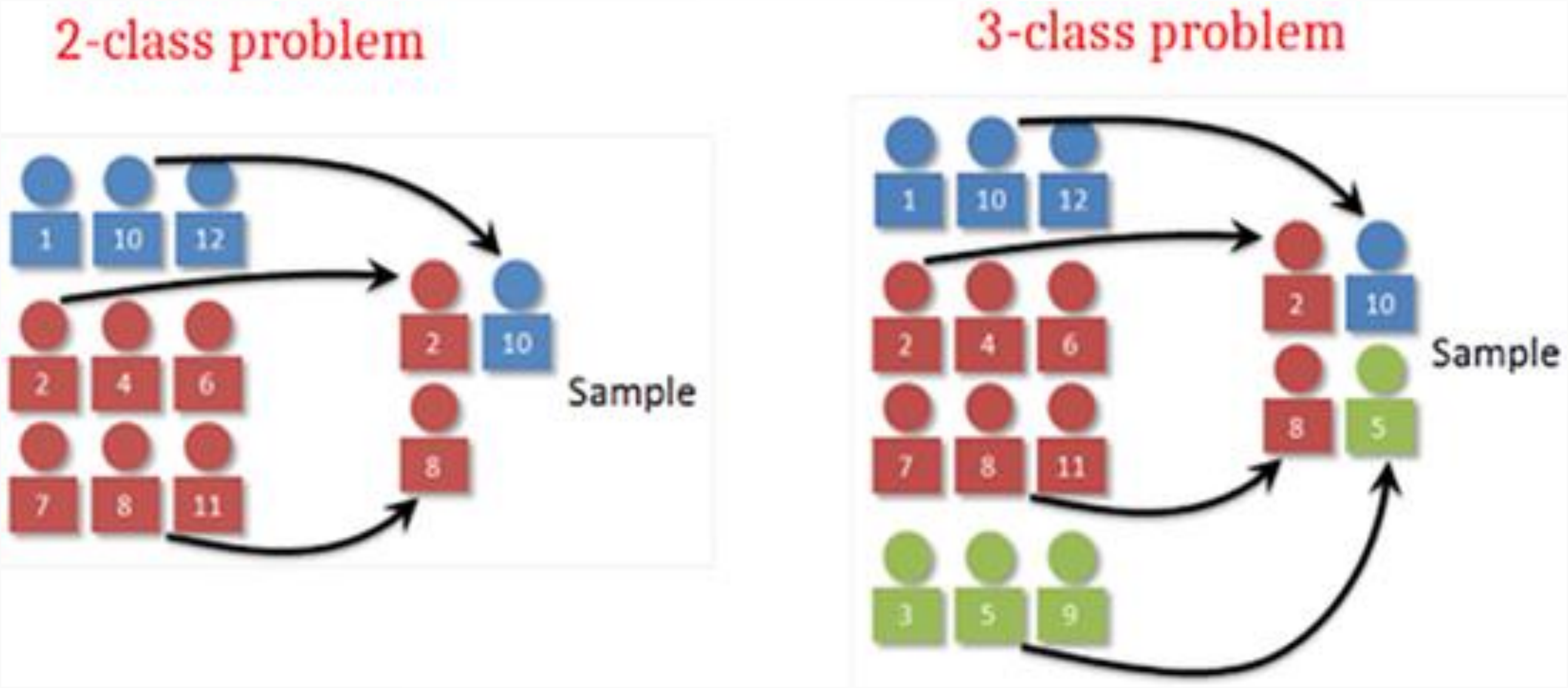
- Difference with random sampling:
  - When randomly selecting training (or validation) sets, class proportions may differ between training and test splits.





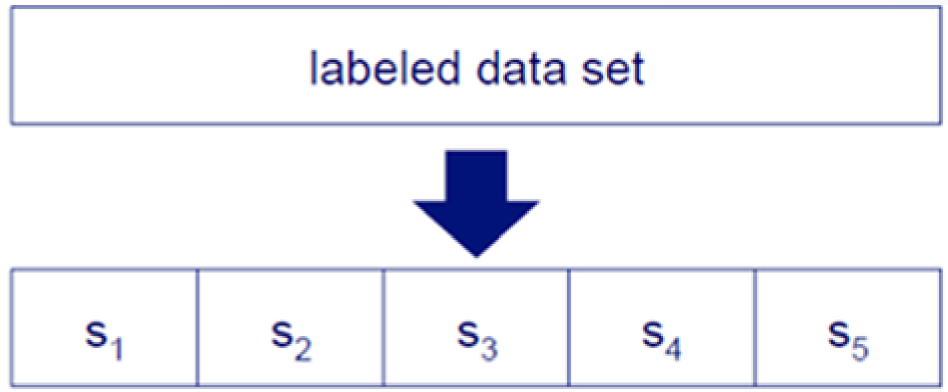
Partitioning data: Stratified Sampling…

- For multi-class problem:



Partitioning data: Cross-validation

- Cross-validation.
  - Evaluate models by partitioning the original sample into a training set to train the model, and a test set to evaluate it.
  - The main idea is to partition training data into k equal sized sub-samples (k-fold cross-validation).
  - Iteratively leave one sub-sample out for evaluating the model and train the model using the rest of the sub-samples.



iteration	train on	test on
1	s_2 s_3 s_4 s_5	s_1
2	s_1 s_3 s_4 s_5	s_2
3	s_1 s_2 s_4 s_5	s_3
4	s_1 s_2 s_3 s_5	s_4
5	s_1 s_2 s_3 s_4	s_5

Partitioning data: Cross-validation…

- In summary:
  - Accuracy can be averaged over multiple runs. (sounds like multiple splitting!)
  - In the special case, when k is equal to the number of instances n,
    - Leave-one-out cross-validation scheme.
  - Cross-validation makes efficient use of the available data for testing.

Iteration	Train on	Test on	Correct
1	S_2, S_3, S_4, S_5	S_1	110/200
2	S_1, S_3, S_4, S_5	S_2	170/200
3	S_1, S_2, S_4, S_5	S_3	160/200
4	S_1, S_2, S_3, S_5	S_4	130/200
5	S_1, S_2, S_3, S_4	S_5	160/200

Accuracy = 730/1000 = 73%

Mean Square Error in Linear Regression

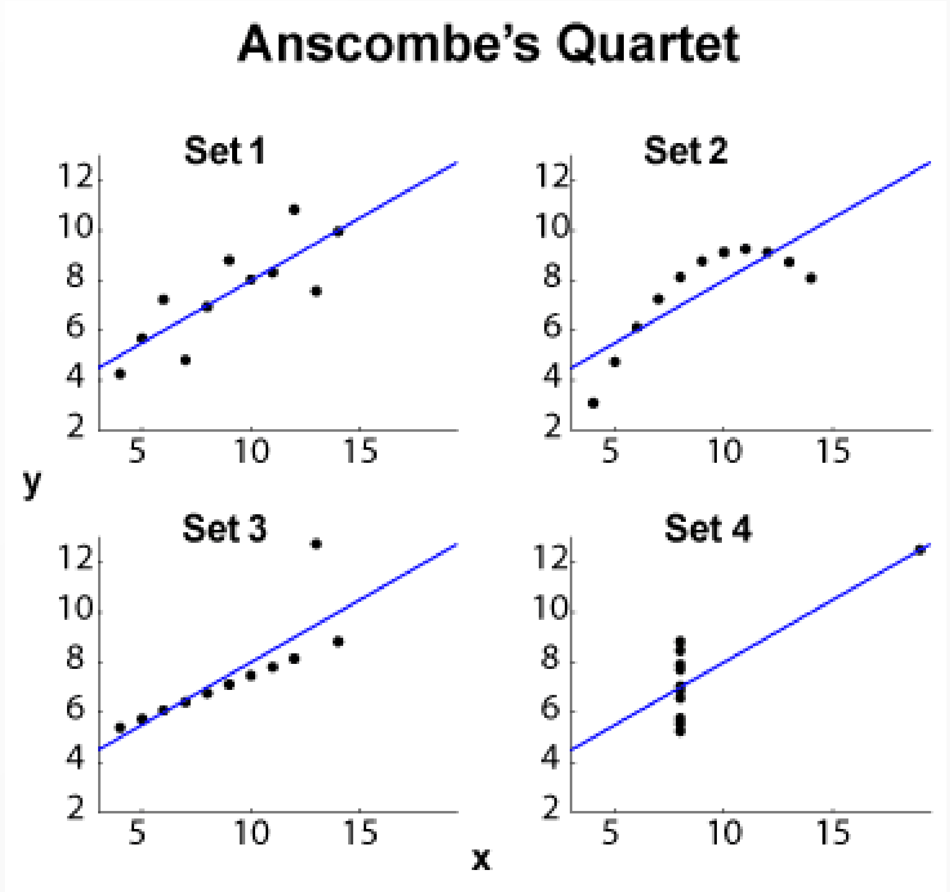
- Mean Square Error (MSE) is a metric used to evaluate linear models

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

- The goal of any model should be to reduce the MSE
  - Why?
  - A smaller MSE implies that there is relatively little difference between the estimated and observed outputs.
  - A well-fitted model should have a relatively low MSE value.
  - The ideal form has an MSE of zero
    - No difference between the estimated and observed parameters.

MSE in Linear Regression…
The effects of test size on MSE

- There are two cases in which you have to be extremely careful when using MSE to compare your results with the true outputs.
- Case-1: Anscombe's quartet comprises four datasets that have nearly identical simple descriptive statistics, yet appear very different when graphed.
- MSE is not a wise measurement in all these cases except in set 1 (as the MSE values will be the same in all 4 cases)



MSE in Linear Regression: The effects of test size on MSE…

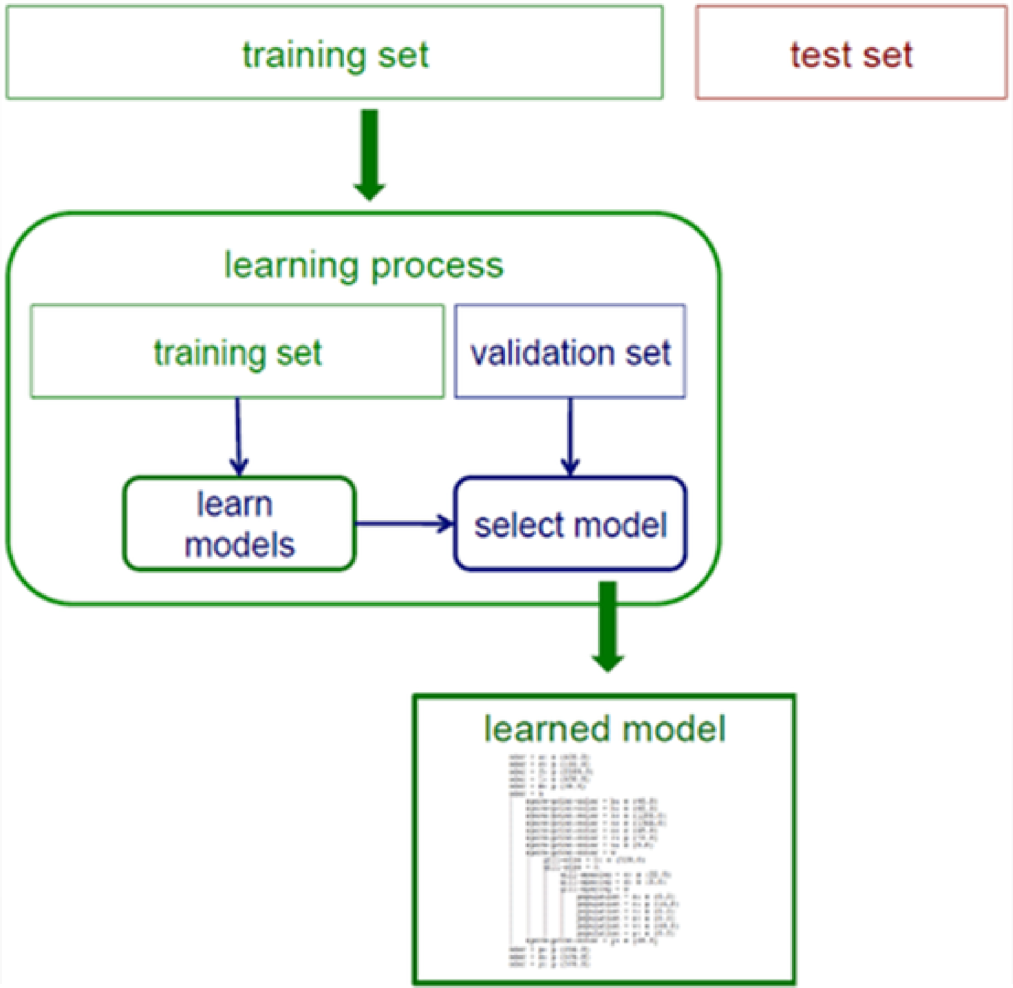
- Case-2: The number of test data points while calculating MSE of your model is important.
  - What if you have two models and you want to compare their performance based on MSE.
  - One of these models has 100 test points and the other one has been evaluated on 100000 test points.
  - Is it fair to compare such models?
    - Evaluate different models with the same number of test data points.

Finding the best hyper-parameters

- What is a hyperparameter?
  - a parameter whose value is set before the learning process begins.
  - the value of a hyperparameter in a model cannot be estimated from data.
  - often used in processes to help estimate model parameters.
  - often set by using heuristics
    - often tuned for a given predictive modeling problem

Finding the best hyper-parameters...

- To search for the best hyperparameters, we need to partition training data into separate training and validation sets
- We already know about training and test data.
- But what is validation set?



Finding the best hyper-parameters…

- A validation set:
  - a sample of data used to provide an unbiased evaluation of a model fit on the training dataset
  - fine-tune the model hyperparameters. How?
    - evaluate the performance of the model for different combinations of hyperparameter values (e.g. by means of a grid search process) and keep the best trained model or hyperparameters.
    - Why validation set?
      - Keeping test set unbiased to model training or hyperparameter selection
        - Since test set is used for comparing different models

Finding the best hyper-parameters...

- But, how can we exactly find the best hyper-parameter?
  - First, we need to decidea possible range for hyperparameter, i.e., a bounded interval such as [0,1]
  - We then define a search grid within the specified range. i.e., we would like to select these values {0, 10^−3, 10^−2, 10^−1, 1} for hyperparameter in order to evaluate the model with them.
  - Next, we train a model using each hyperparameter value from the search grid and assess its performance on a validation set (taken out from the training set).
  - Finally, We compute the performance on the validation set for each hyper-parameter value and select the one with the best performance.

Finding the best hyper-parameters...

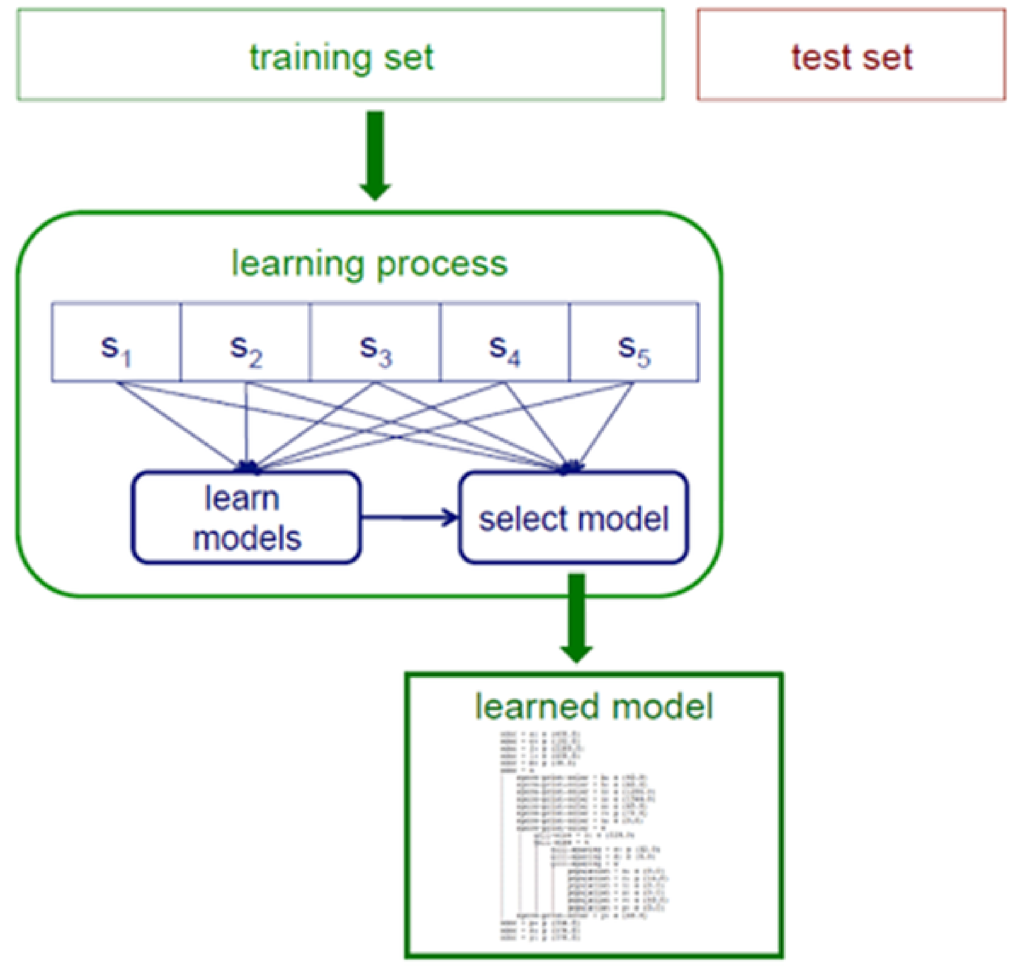
- Note: In the above example we just considered 4 cases to evaluate as a hyperparameter in the [0,1] interval.
- In this continues space, we may lose many other good options by restricting the search only to 4 values.
- It is obvious in this case we will be more accurate in finding the value of the hyperparamter.
- But this Grid-searching can be extremely computationally expensive

Finding the best hyper-parameters: Internal cross-validation

- All the techniques that we previously discussed for model assessment are applicable for training/validation set splitting:
  - Random subsampling
  - Stratified subsampling
  - Cross-validation
- Remember, this step is internal to the learning process and different from model assessment on the test data.

Finding the best hyper-parameters: Internal cross-validation…

- Let us examine how an internal cross-validation works.



Finding the best hyper-parameters: Internal cross-validation…

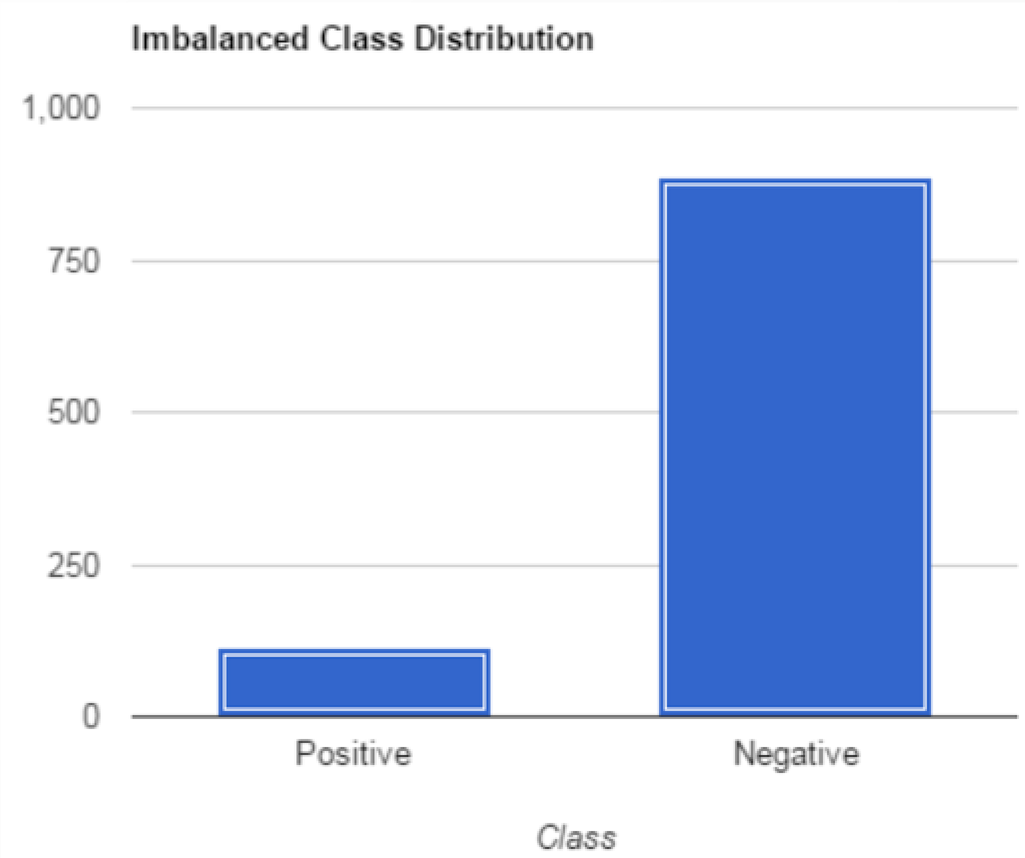
- Remember that we can select the best hyperparameter set by searching/or optimizing over all possible values.
- Let us show you 3 possible ways to navigate the hyperparameter space:
  - Grid-search (not so efficient). This is what we are using and explaining!
  - Random search (efficient in certain scenarios) [Bergstra et al. (JMLR 2012)
  - Bayesian optimization (efficient in general) [Snoek et al. (2012)]

Finding the best hyper-parameters: Effect of imbalanced class

- What is imbalanced data set?
  - Ratio of class distribution is skewed toward some class/es.
    - the total number of a class of data (i.e. positive) is far less than the total number of another class of data (i.e. negative).
- This problem is very common in practice
  - fraud detection
  - anomaly detection
  - medical diagnosis, etc.
- Most machine learning algorithms work best when the number of instances of each classes are roughly equal.

Finding the best hyper-parameters: Effect of imbalanced class

- Example: When developing a breast cancer diagnosis model, imbalanced class problem is encountered because the risk of a female being diagnosed with breast cancer (by their 85th birthday) is 1 in 8.
- This means that a representative training set will have 7 times more instances in negative class than the positive class (see figure below)



Finding the best hyper-parameters: Effect of imbalanced class

- So what are the possible solutions to overcome this problem?
  - Perform some actions on the data itself.
  - Improve the algorithm to be able to handle such phenomenon.
- At the data level: (Re-Sampling):
  - over-sampling the data from minority class
  - under-sampling the data from majority class.
- At the algorithmic level:
  - adjusting the costs
  - adjusting the decision threshold.

Finding the best hyper-parameters: Issues of imbalanced classes

- Now, let us have a close look on possible issues of imbalanced classes.
- Problem-1: Since the test data contains only few samples from the minority class, even a dumb classifier that always classifies an instance to the majority class will get very high accuracy!
  - This problem is dealt by using other evaluation metrics in place of accuracy.
- Problem-2: When doing random subsampling, it is possible that class proportion is not maintained in individual partition. In fact, we may not sample even one instance from the minority class.
  - This problem can be solved using Stratified Sampling.

Finding the best hyper-parameters: Issues of imbalanced classes

- But always remember:
  - Any pre-processing over entire data set (e.g. feature selection, or feature extraction) must not use the information that you are trying to predict (e.g. labels).
  - In the training process, you must not use any information that is not available during the training process.
    - Example: I was building a cancer prognosis model to predict whether a patient will survive 1 year from diagnosis or not?
      - I used the cause of death field as one of the features.
        - Clearly, this information is not available at the prediction time.
- If you modify your model again and again by looking at how it performs on a specified test set, then you may be overfitting on the test set.

Thank You.