Classifier Evaluation

Ying Shen, SSE, Tongji University

Classification error

- The errors committed by a classification model are generally divided into two types
  - Training errors
  - Generalization errors
- Training error is the number of misclassification errors committed on training records.
- Training error is also known as resubstitution error or apparent error.
- Generalization error is the expected error of the model on previously unseen records.

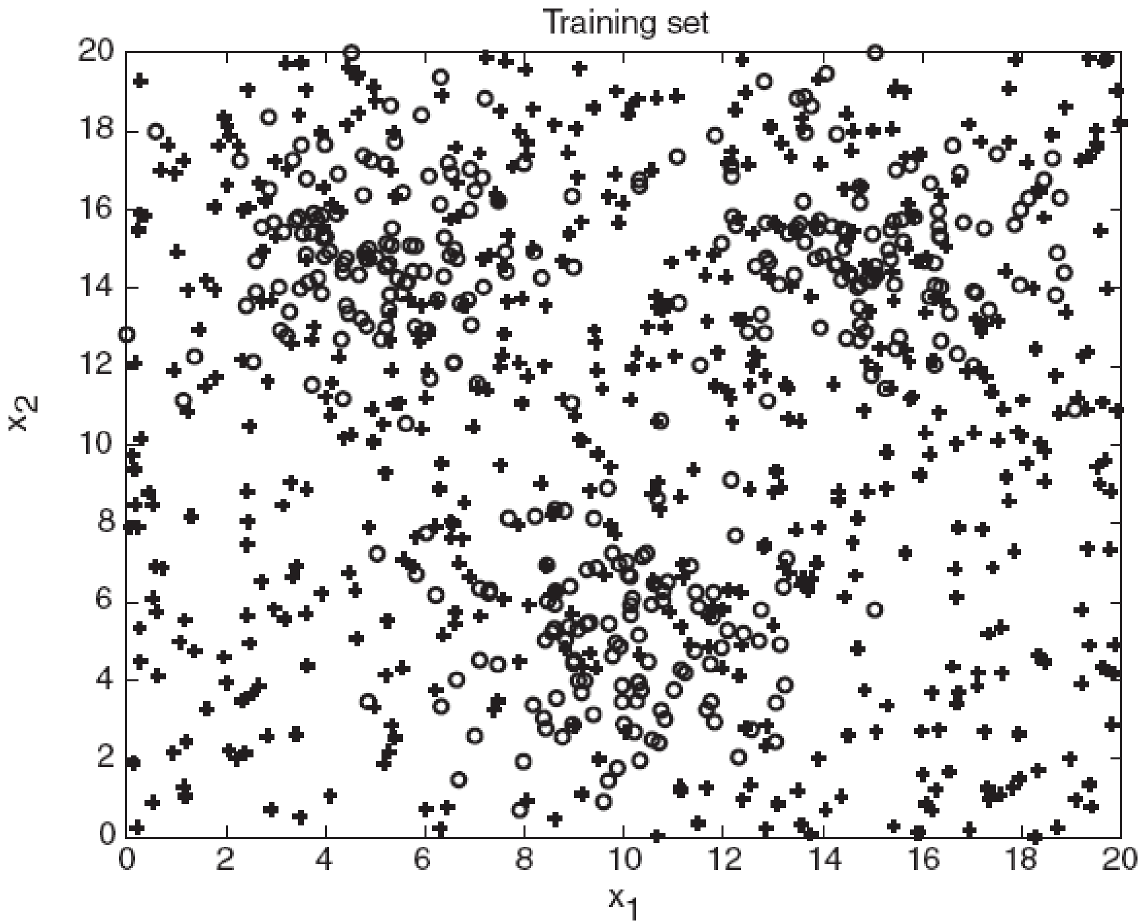
Classification error

- A good classification model should
  - Fit the training data well. (low training error)
  - Accurately classify records it has never seen before. (low generalization error)
- A model that fits the training data too well can have a poor generalization error.
- This is known as model overfitting.

Classification error

- We consider the 2-D data set in the following figure.
- The data set contains data points that belong to two different classes.
- 30% of the points are chosen for training, while the remaining 70% are used for testing.
- A decision tree classifier is applied to the training set.
- Different levels of pruning are applied to the tree to investigate the effect of overfitting

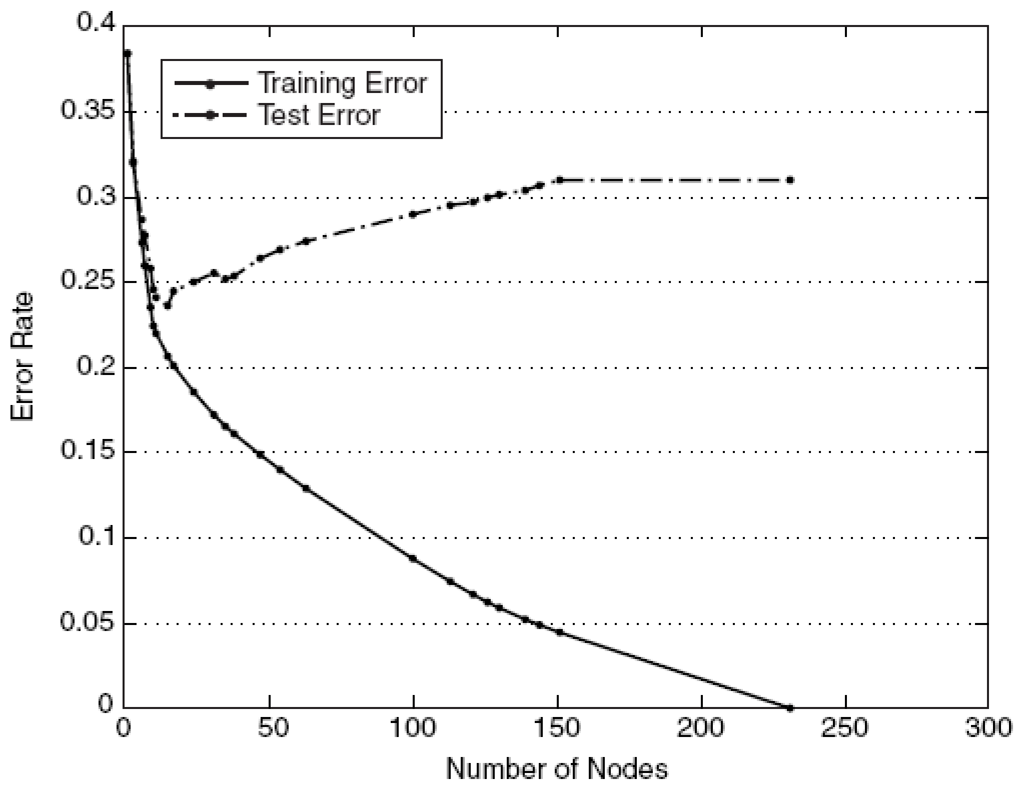
Classification error



Classification error

- The following figure shows the training and test error rates of the decision tree.
- Both error rates are large when the size of the tree is very small.
- This situation is known as model underfitting.
- Underfitting occurs because the model cannot learn the true structure of the data.
- It performs poorly on both the training and test sets.

Classification error



Classification error

- When the tree becomes too large
  - The training error rate continues to decrease.
  - However, the test error rate begins to increase.
- This phenomenon is known as model overfitting.

Overfitting

- The training error can be reduced by increasing the model complexity.
- However, the test error can be large because the model may accidentally fit some of the noise points in the training data.
- In other words, the performance of the model on the training set does not generalize well to the test examples.

Overfitting

- We consider a training and test set for a mammal classification problem.
- Two of the ten training records are mislabeled.
- Bats and whales are labeled as non- mammals instead of mammals.

Training set

Name	Body Temperature	Gives Birth	Four- Legged	Hibernates	Class Label
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
Bat	warm-blooded	yes	no	yes	no*
whale	warm-blooded	yes	no	no	no*
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no

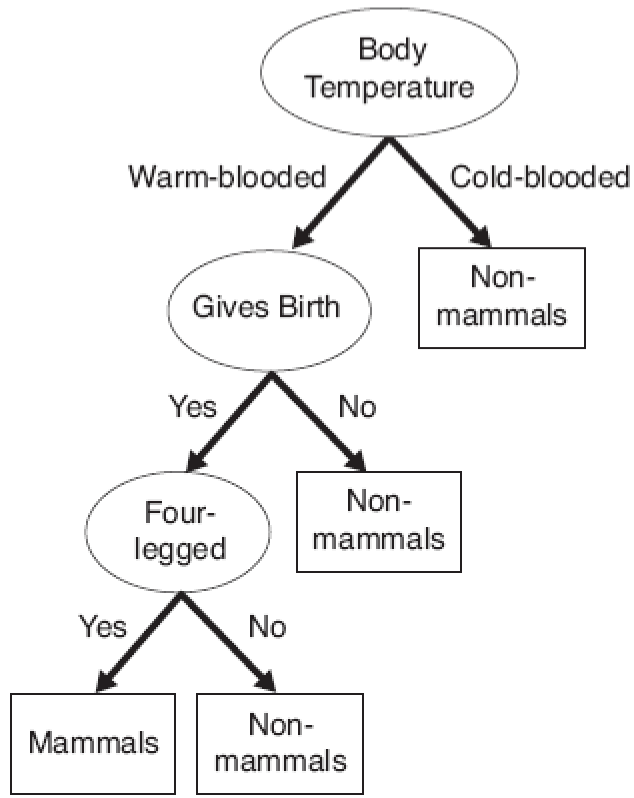
Test set

Name	Body Temperature	Gives Birth	Four- Legged	Hibernates	Class Label
human	warm-blooded	yes	no	no	yes
pigeon	warm-blooded	no	no	no	no
elephant	warm-blooded	yes	yes	no	yes
leopard shark	cold-blooded	yes	no	no	no
turtle	cold-blooded	no	yes	no	no
penguin	warm-blooded	no	no	no	no
eel	cold-blooded	no	no	no	no
dolphin	warm-blooded	yes	no	no	yes
spiny anteater	warm-blooded	no	yes	yes	yes
gila monster	cold-blooded	no	yes	yes	no

Overfitting

- A decision tree that perfectly fits the training data is shown in the following figure.
- The training error for the tree is zero.
- However, its error rate on the test set is 30%.

Overfitting



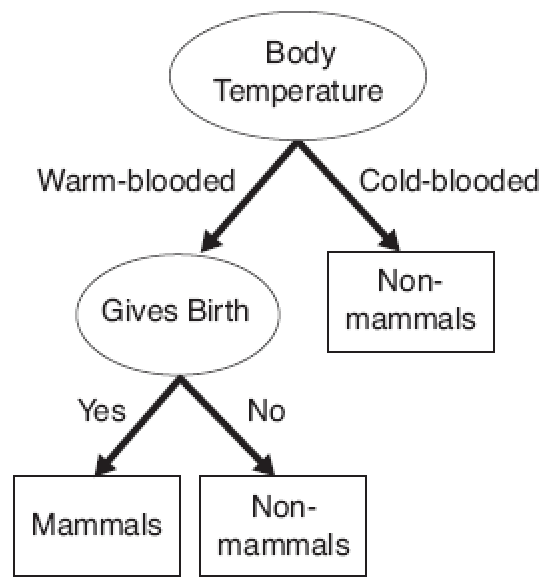
Overfitting

- Both humans and dolphins are misclassified as non-mammals.
- Their attribute values for Body Temperature, Gives Birth and Four-legged are identical to the mislabeled records in the training set.
- On the other hand, spiny anteater represents an exceptional case.
- The class label of the test record contradicts the class labels of other similar records in the training set.

Overfitting

- In contrast, the simpler decision tree in the following figure has
  - A somewhat higher training error rate (20%) but
  - A lower test error rate (10%).
- It can be seen that the Four-legged attribute test condition in the first model is spurious.
- It fits the mislabeled training records, which leads to the misclassification of records in the test set.

Overfitting



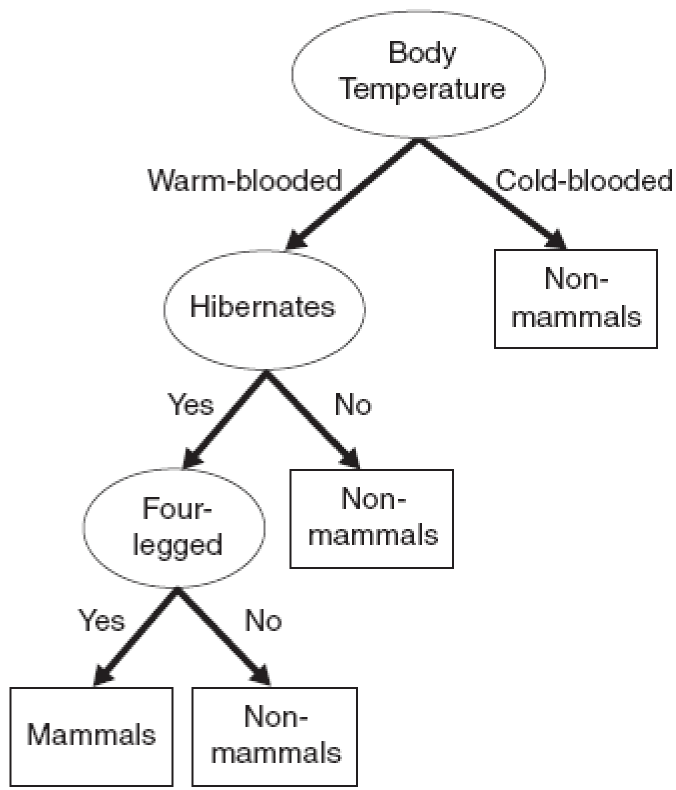
Overfitting

- Models that make their classification decisions based on a small number of training records are also susceptible to overfitting.
- We consider the five training records in the following table.
- The corresponding decision tree can label all the training records correctly.

Training set

Name	Body Temperature	Gives Birth	Four- Legged	Hibernates	Class Label
salamander	cold-blooded	no	yes	yes	no
guppy	cold-blooded	yes	no	no	no
eagle	warm-blooded	no	no	no	no
poorwill	warm-blooded	no	no	yes	no
platypus	warm-blooded	no	yes	yes	yes

Overfitting



Overfitting

- Although the training error is zero, the error rate on the previous test set is 30%.
- The model classifies all warm-blooded vertebrates that do not hibernate as non- mammals.
- As a result, humans, elephants and dolphins are misclassified.
- This is because there is only one training record (eagle) with such characteristics.

Generalization error estimation

- The ideal classification model is the one that produces the lowest generalization error.
- The problem is that the model has no knowledge of the test set.
- It has access only to the training set.
- We consider two approaches to estimate the generalization error
  - Resubstitution estimate
  - Estimates incorporating model complexity
  - Using a validation set

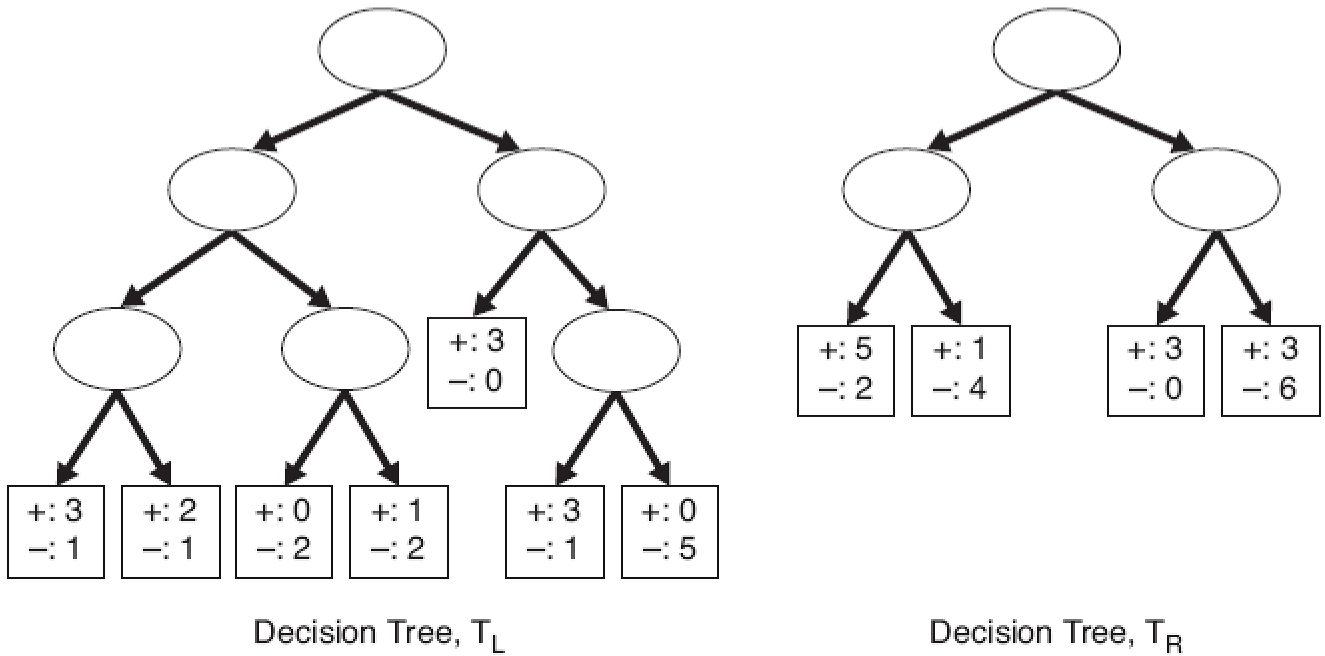
Resubstitution estimate

- The resubstitution estimate approach assumes that the training set is a good representation of the overall data.
- In other words, the training error can be used to provide an optimistic estimate for the generalization error.
- However, the training error is usually a poor estimate of generalization error.

Resubstitution estimate

- We consider the two decision trees shown in the following figure.
- The left tree T_L is more complex than the right tree T_R.
- The training error rate for T_L is $e(T_L)$=4/24=0.167.
- The training error rate for T_R is $e(T_R)$=6/24=0.25.
- Based on the resubstitution estimate, T_L is considered better than T_R.

Resubstitution estimate



Estimates incorporating model complexity

- The chance for model overfitting increases as the model becomes more complex.
- As a result, we should prefer simpler models.
- Based on this principle, we can estimate the generalization error as the sum of
  - Training error and
  - A penalty term for model complexity.

Estimates incorporating model complexity

- In the case of a decision tree, let
  - L be the number of leaf nodes.
  - n_l be the l-th leaf node.
  - $m(n_l)$ be the number of training records classified by node n_l.
  - $e(n_l)$ be the number of misclassified records by node n_l.
  - $\zeta(n_l)$ be a penalty term associated with the node n_l.
- The resulting error e_c of the decision tree can be estimated as follows:

$$e_c = \frac{\sum_{l=1}^{L}[e(n_l) + \zeta(n_l)]}{\sum_{l=1}^{L} m(n_l)}$$

Estimates incorporating model complexity

- We consider the previous two decision trees T_L and T_R.
- We assume that the penalty term is equal to 0.5 for each leaf node.
- The error estimate for T_L is

$$e_c(T_L) = \frac{4 + 7 \times 0.5}{24} = \frac{7.5}{24} = 0.3125$$

- The error estimate for T_R is

$$e_c(T_L) = \frac{6 + 4 \times 0.5}{24} = \frac{8}{24} = 0.3333$$

Estimates incorporating model complexity

- Based on this penalty term, T_L is better than T_R.
- For a binary tree, a penalty term of 0.5 means that a node should always be expanded into its two child nodes if it improves the classification of at least one training record.
- This is because expanding a node, which is the same as adding 0.5 to the overall error, is less costly than committing one training error.

Estimates incorporating model complexity

- Suppose the penalty term is equal to 1 for all the leaf nodes.
- The error estimate for T_L becomes 0.458.
- The error estimate for T_R becomes 0.417.
- Based on this penalty term, T_R is better than T_L.
- A penalty term of 1 means that a node should not be expanded unless it reduces the misclassification error by more than one training record.

Using a validation set

- In this approach, the original training data is divided into two smaller subsets.
- One of the subsets is used for training.
- The other, known as the validation set, is used for estimating the generalization error.

Using a validation set

- This approach can be used in the case where the complexity of the model is determined by a parameter.
- We can adjust the parameter until the resulting model attains the lowest error on the validation set.
- This approach provides a better way for estimating how well the model performs on previously unseen records.
- However, less data is available for training.

Handling overfitting in decision tree

- There are two approaches for avoiding model overfitting in decision tree
  - Pre-pruning
  - Post-pruning

Pre-pruning

- In this approach, the tree growing algorithm is halted before generating a fully grown tree that perfectly fits the training data.
- To do this, an alternative stopping condition could be used.
- For example, we can stop expanding a node when the observed gain in impurity measure falls below a certain threshold.

Pre-pruning

- The advantage of this approach is that it avoids generating overly complex sub-trees that overfit the training data.
- However, it is difficult to choose the right threshold for early termination.
- A threshold which is too high will result in underfitted models.
- A threshold which is too low may not be sufficient to overcome the model overfitting problem.

Post-pruning

- In this approach, the decision tree is initially grown to its maximum size.
- This is followed by a tree pruning step, which trims the fully grown tree.

Post-pruning

- Trimming can be done by replacing a sub- tree with a new leaf node whose class label is determined from the majority class of records associated with the sub-tree.
- The tree pruning step terminates when no further improvement is observed.

Post-pruning

- Post-pruning tends to give better results than pre-pruning because it makes pruning decisions based on a fully grown tree.
- On the other hand, pre-pruning can suffer from premature termination of the tree growing process.
- However, for post-pruning, the additional computations for growing the full tree may be wasted when some of the sub-trees are pruned.

Classifier evaluation

- There are a number of methods to evaluate the performance of a classifier
  - Hold-out method
  - Cross validation
  - Bootstrap

Hold-out method

- In this method, the original data set is partitioned into two disjoint sets.
- These are called the training set and test set respectively.
- The classification model is constructed from the training set.
- The performance of the model is evaluated using the test set.

Hold-out method

- The hold-out method has a number of well known limitations.
- First, fewer examples are available for training.
- Second, the model may be highly dependent on the composition of the training and test sets.

Hold-out method

- A training set which is too small may not be representative of the original data set.
- On the other hand, if the training set is too large, the estimated accuracy computed from the smaller test set is less reliable.

Cross validation

- In this approach, each record is used the same number of times for training, and exactly once for testing.
- To illustrate this method, suppose we partition the data into two equal-sized subsets.
- First, we choose one of the subsets for training and the other for testing.
- We then swap the roles of the subsets so that the previous training set becomes the test set, and vice versa.

Cross validation

- The estimated error is obtained by averaging the errors on the test sets for both runs.
- In this example, each record is used exactly once for training and once for testing.
- This approach is called a two-fold cross- validation.

Cross validation

- The k-fold cross validation method generalizes this approach by segmenting the data into k equal-sized partitions.
- During each run
  - One of the partitions is chosen for testing.
  - The rest of them are used for training.
- This procedure is repeated k times so that each partition is used for testing exactly once.
- The estimated error is obtained by averaging the errors on the test sets for all k runs.

Cross validation

- In the leave-one-out approach, each test set contains only one record.
- This approach has the advantage of utilizing as much data as possible for training.
- The drawback of this approach is that it is computationally expensive.
- Furthermore, since each test set contains only one record, the variance of the estimated error tends to be high.

Bootstrap

- In the bootstrap approach, the training records are sampled with replacement.
- If the original data has R records, a bootstrap sample of size R contains, on the average, about 63.2% of the records in the original data.
- This follows from the fact that the probability a record is chosen is $1-(1-1/R)^R$.
- When R is sufficiently large, the probability asymptotically approaches $1-e^{-1}=0.632$.

Bootstrap

- Records that are not included in the bootstrap sample become part of the test set.
- We construct a classification model from the bootstrap sample.
- The model is then applied to the test set to obtain an estimate of the accuracy a_b.
- The sampling procedure is then repeated B times to generate B bootstrap samples.

Bootstrap

- One of the more widely used bootstrap sampling approaches is the .632 bootstrap.
- This approach first combines
  - The accuracy of each bootstrap sample, a_b, with
  - The accuracy calculated from a training set that contains all the records, a_{orig}
- The combined value is then averaged across all the different bootstrap samples to obtain the overall accuracy a_{boot}

$$a_{boot} = \frac{1}{B}\sum_{b=1}^{B}(0.632a_b + 0.368a_{orig})$$