Data Mining

Model Overfitting

Introduction to Data Mining, 2nd Edition

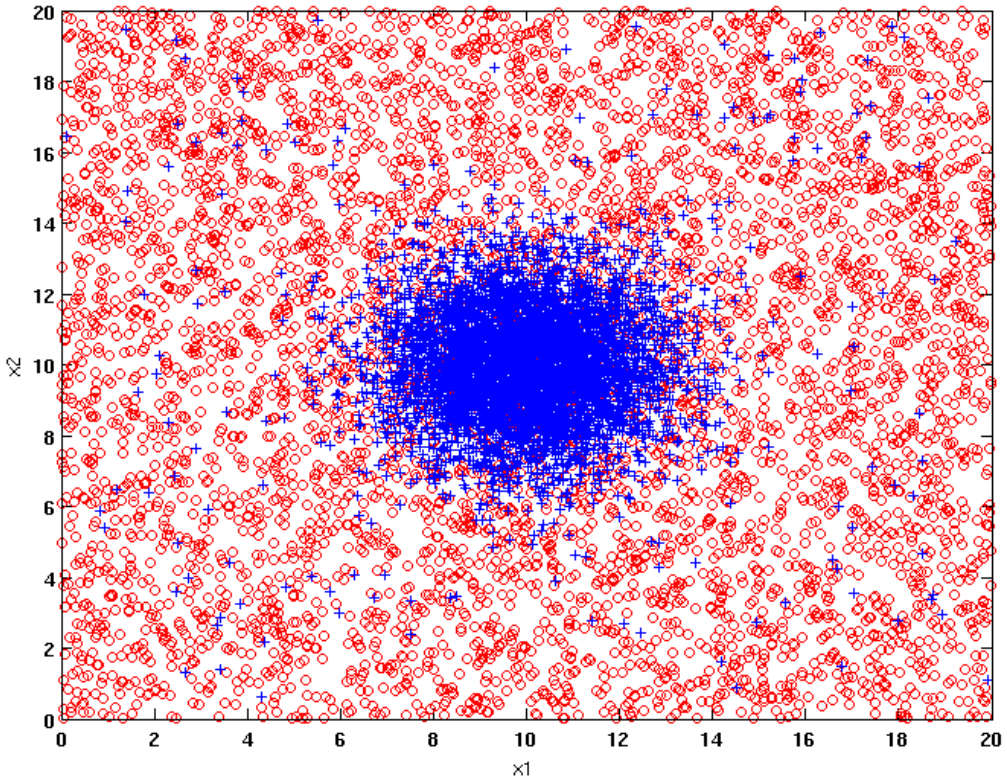
by

Tan, Steinbach, Karpatne, Kumar

Classification Errors

- Training errors (apparent errors)
  - Eğitim setinde yapılan hatalar
- Test errors
  - Test setinde yapılan hatalar
- Generalization errors
  - Aynı dağılımdan rastgele kayıtların seçimi üzeriden bir modelin beklenen hatası

Example Data Set



Two class problem:

+ : 5200 instances

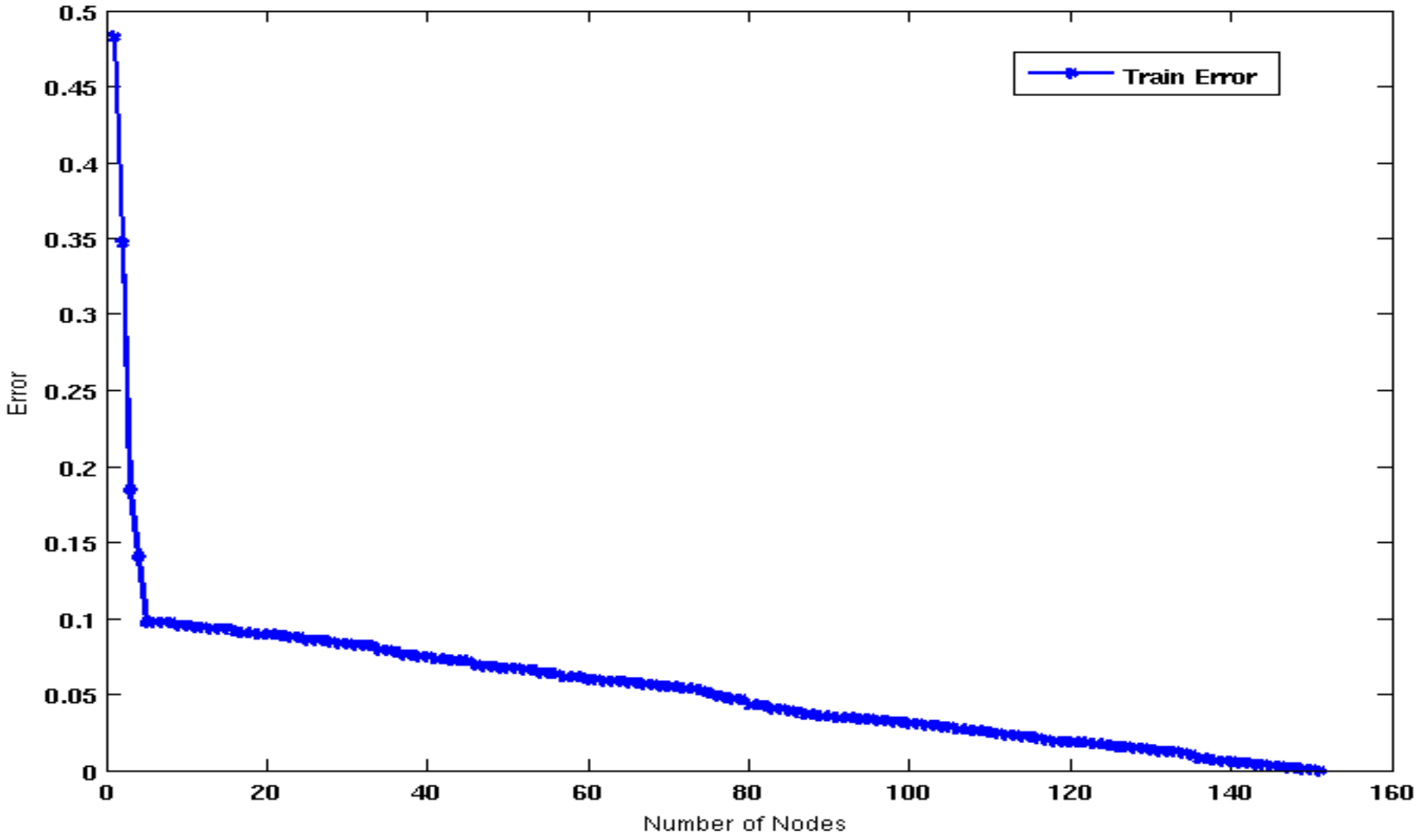
- **5000 instances generated from a Gaussian centered at (10,10)**
- **200 noisy instances added**

o : 5200 instances

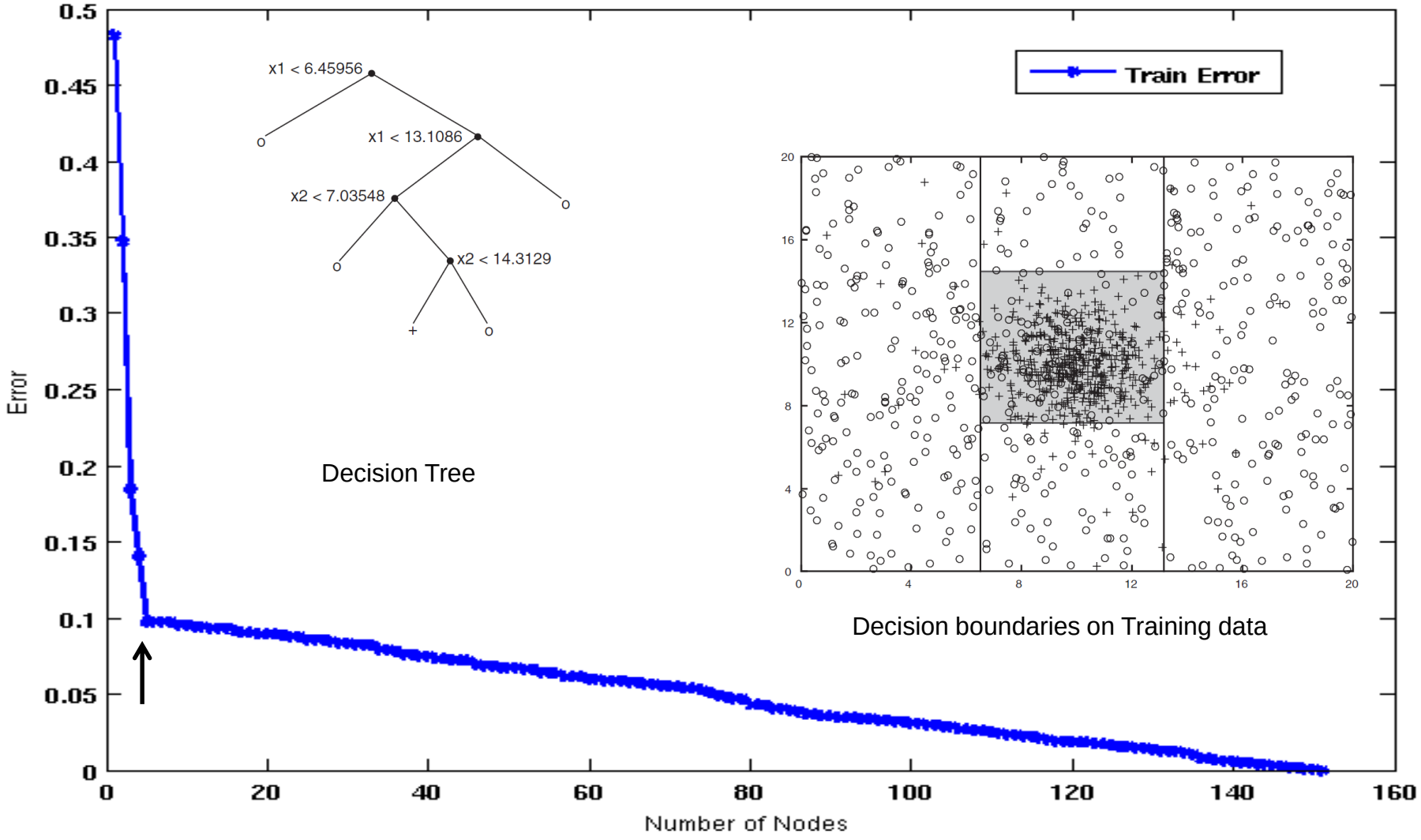
- **Generated from a uniform distribution**

10 % of the data used for training and 90% of the data used for testing

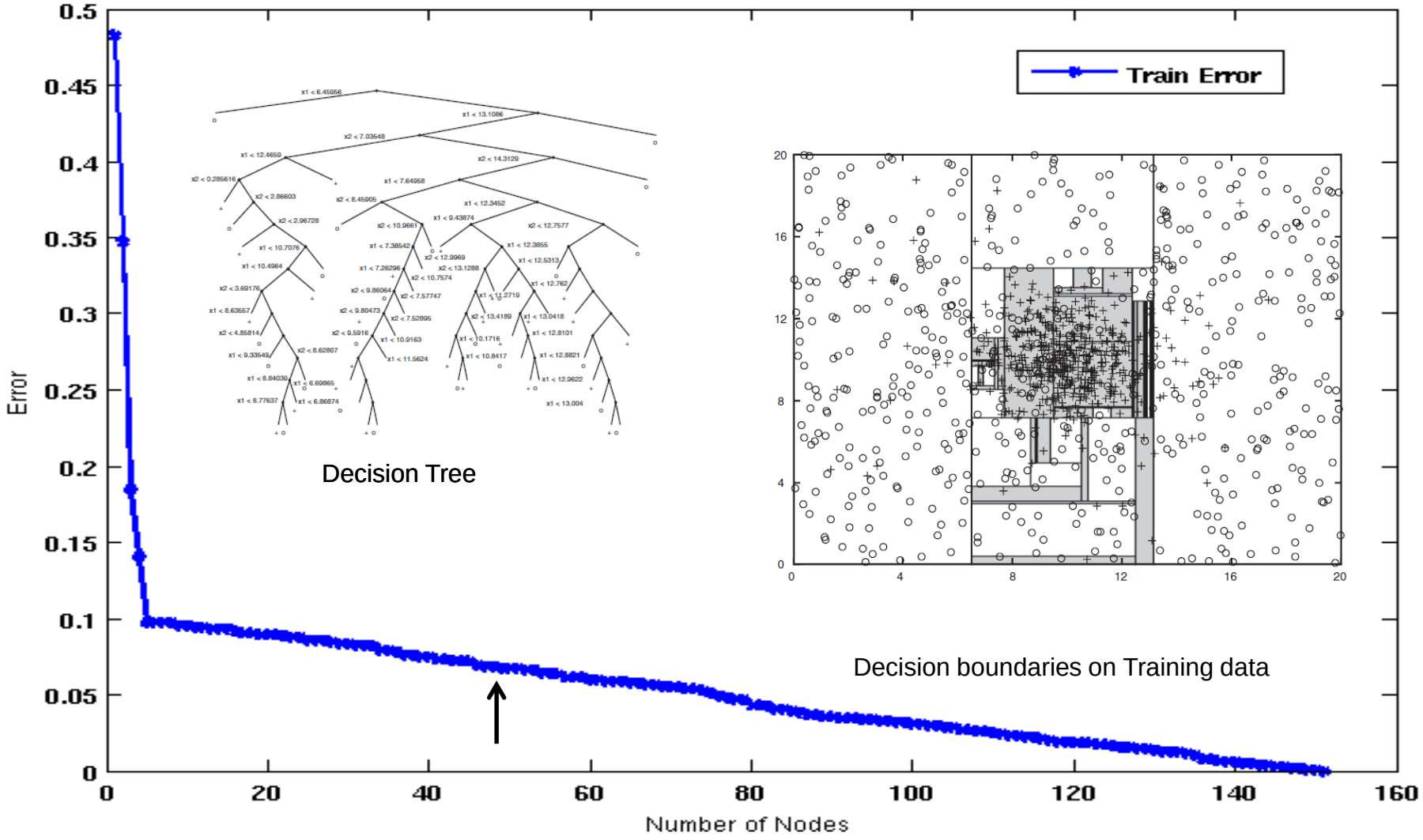
Increasing number of nodes in Decision Trees



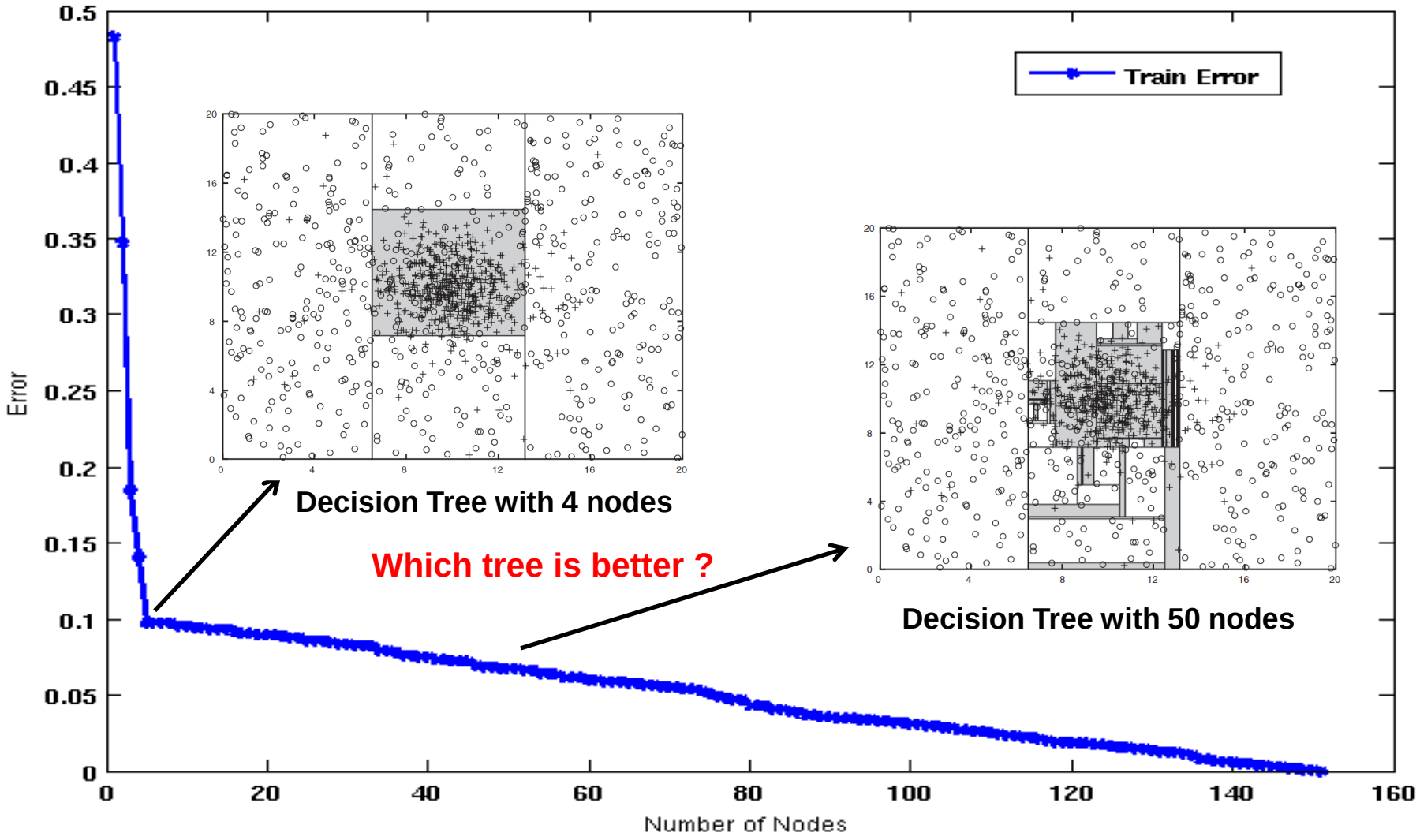
Decision Tree with 4 nodes



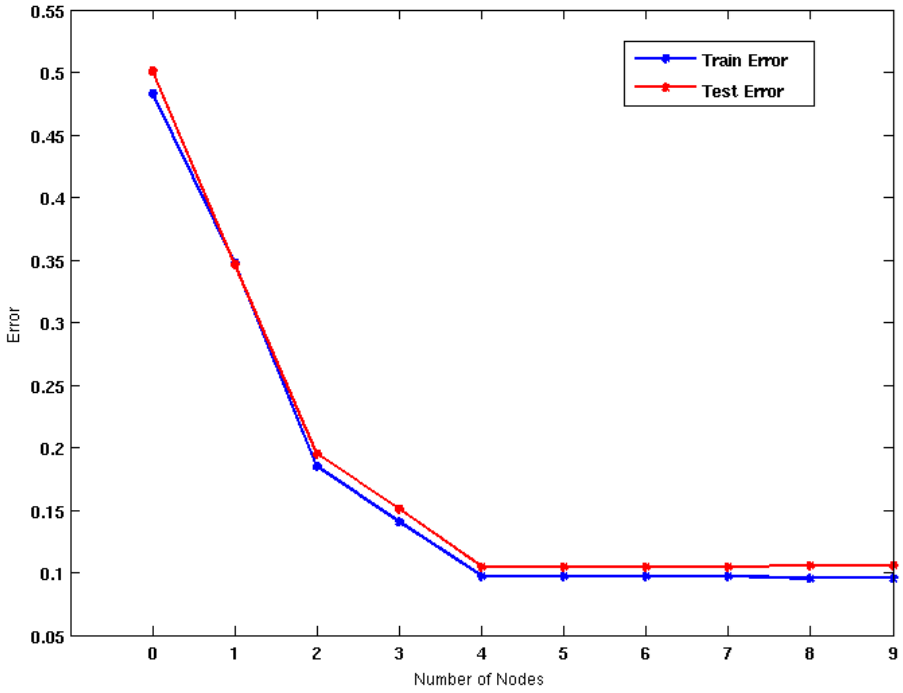
Decision Tree with 50 nodes

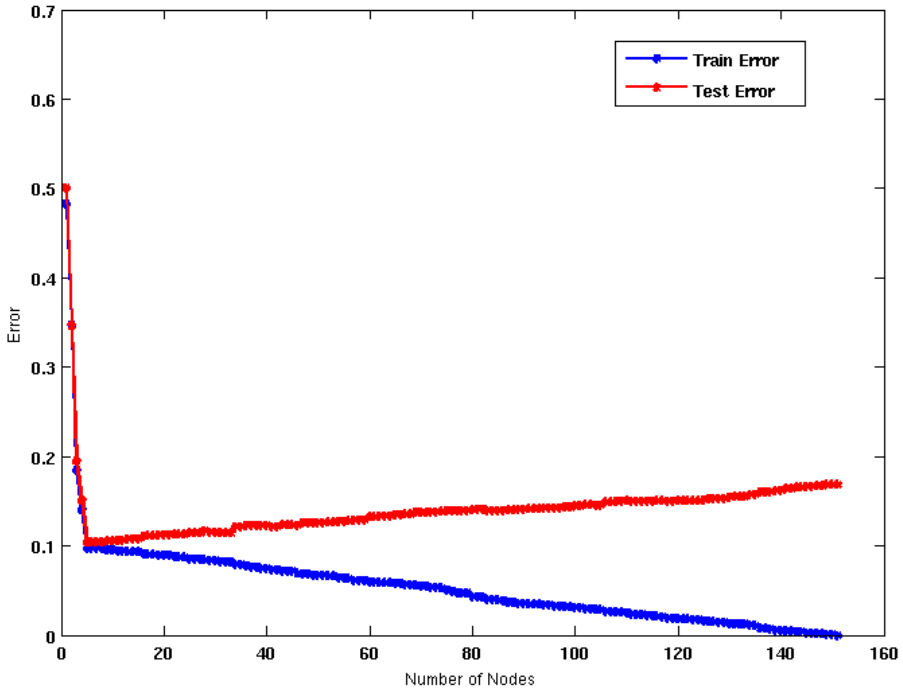


Which tree is better?



Model Overfitting



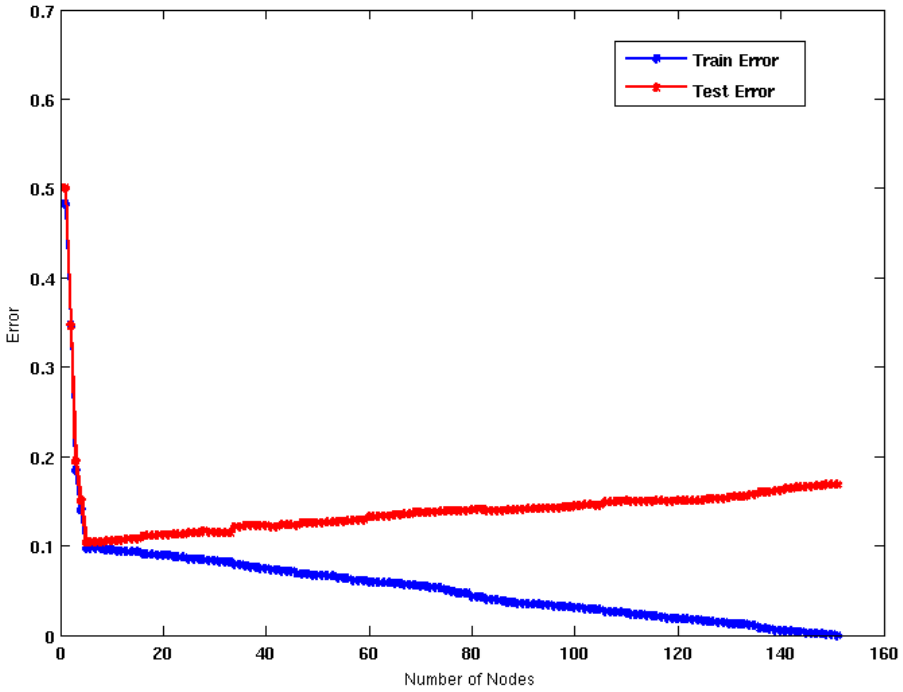


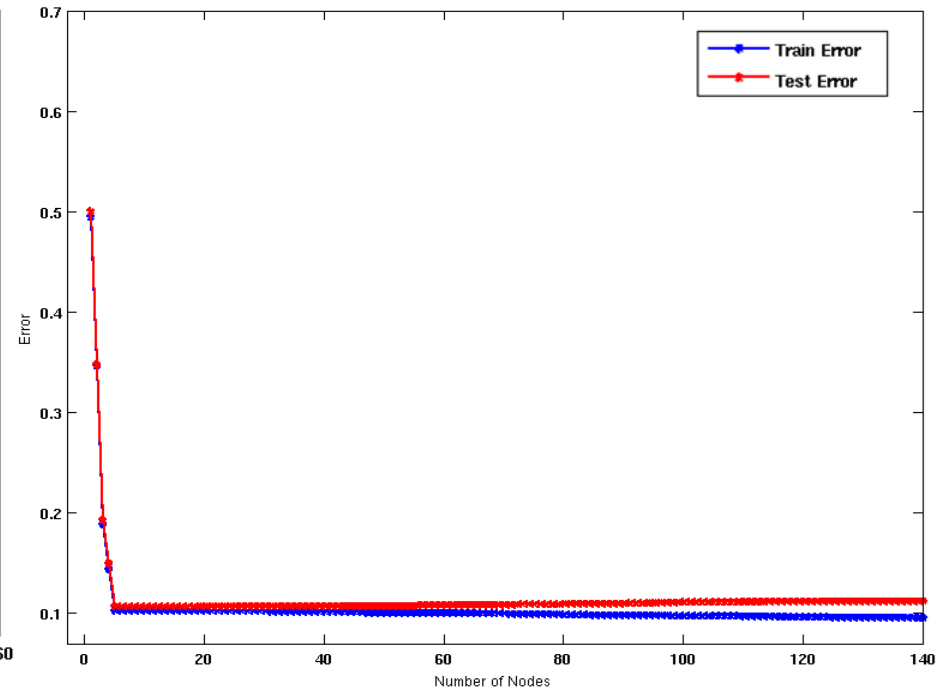
Underfitting: model çok basit olduğunda hem eğitim hem de test hataları büyüktür

Overfitting: model çok karmaşık olduğunda, eğitim hatası küçüktür ancak test hatası büyüktür

Overfitting ve ***underfitting*** model karmaşıklığıyla ilgili iki patolojidir.

Model Overfitting

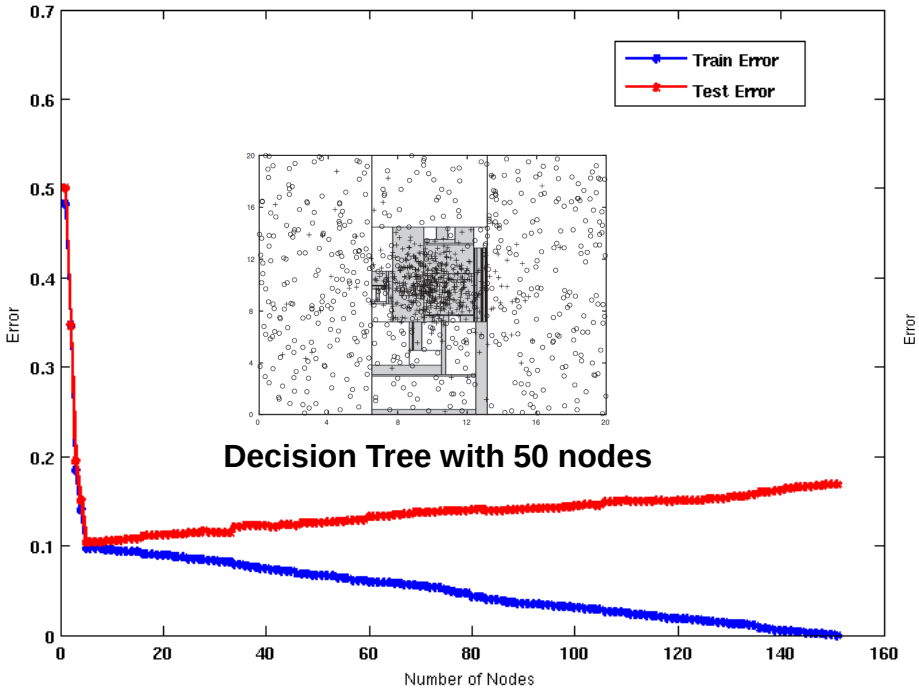


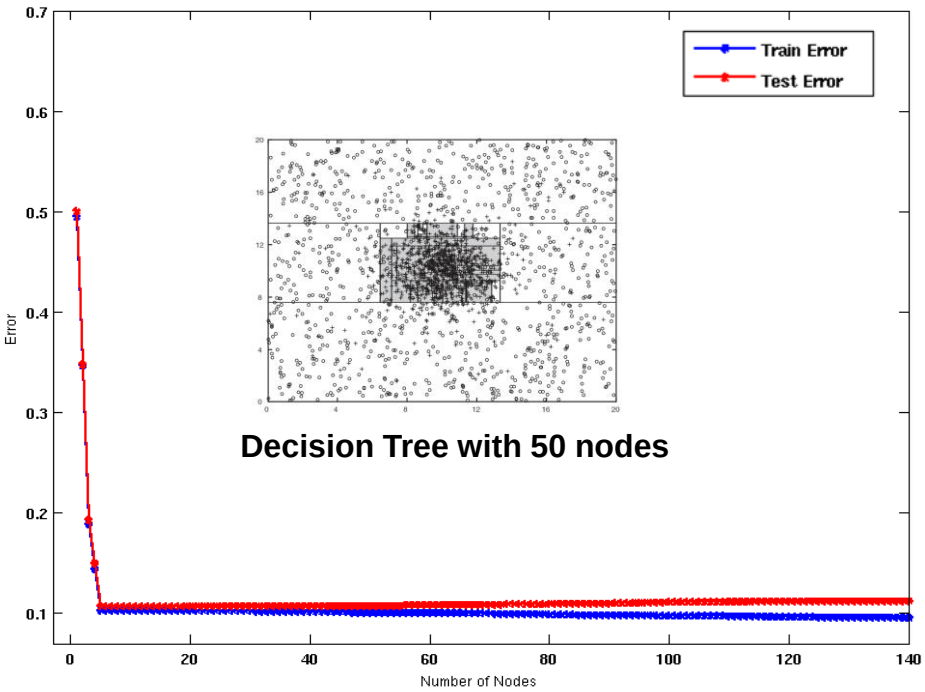


İki kat daha fazla veri örneği (data instances) kullanma

- Eğitim verileri temsili yetersizse (**under-representative**), artan düğüm sayısı ile test hataları artar ve eğitim hataları azalır
- Eğitim verilerinin boyutunu artırmak (**Increasing the size of training data**), belirli sayıda düğümde eğitim ve test hataları arasındaki farkı azaltır

Model Overfitting





İki kat daha fazla veri örneği (data instances) kullanma

- Eğitim verileri temsili yetersizse (**under-representative**), artan düğüm sayısı ile test hataları artar ve eğitim hataları azalır
- Eğitim verilerinin boyutunu artırmak (**Increasing the size of training data**), belirli sayıda düğümde eğitim ve test hataları arasındaki farkı azaltır

Reasons for Model Overfitting

- Limited Training Size
- High Model Complexity
  - Multiple Comparison Procedure

Effect of Multiple Comparison Procedure

- Önümüzdeki 10 işlem gününde borsanın yükselişİni/düşüşünü tahmin etme görevini düşünün

- Random guessing:

$$P(correct) = 0.5$$

- Arka arkaya 10 rastgele tahmin yapın :

$$P(\#correct \geq 8) = \frac{\binom{10}{8} + \binom{10}{9} + \binom{10}{10}}{2^{10}} = 0.0547$$

Day 1	Up
Day 2	Down
Day 3	Down
Day 4	Up
Day 5	Down
Day 6	Down
Day 7	Up
Day 8	Up
Day 9	Up
Day 10	Down

Effect of Multiple Comparison Procedure

- Approach:
  - 50 analist getirin
  - Her analist 10 rastgele tahmin (***random guess***) yapar
  - En fazla sayıda doğru tahminde bulunan analisti seçin
- En az bir analistin en az 8 doğru tahmin yapma olasılığı

$$P(\#correct \geq 8) = 1 - (1 - 0.0547)^{50} = 0.9399$$

Effect of Multiple Comparison Procedure

$$P(\#correct \geq 8) = 1 - (1 - 0.0547)^{50} = 0.9399$$

Her analistin en az sekiz defa doğru tahmin etme olasılığı düşük olsa da, bunları bir araya getirsek, bunu yapabilecek bir analist bulma olasılığımız yüksek.

Buna ek olarak, gelecekte böyle bir analistin rastgele tahmin yoluyla doğru tahminler yapmaya devam edeceğine dair hiçbir garanti yoktur.

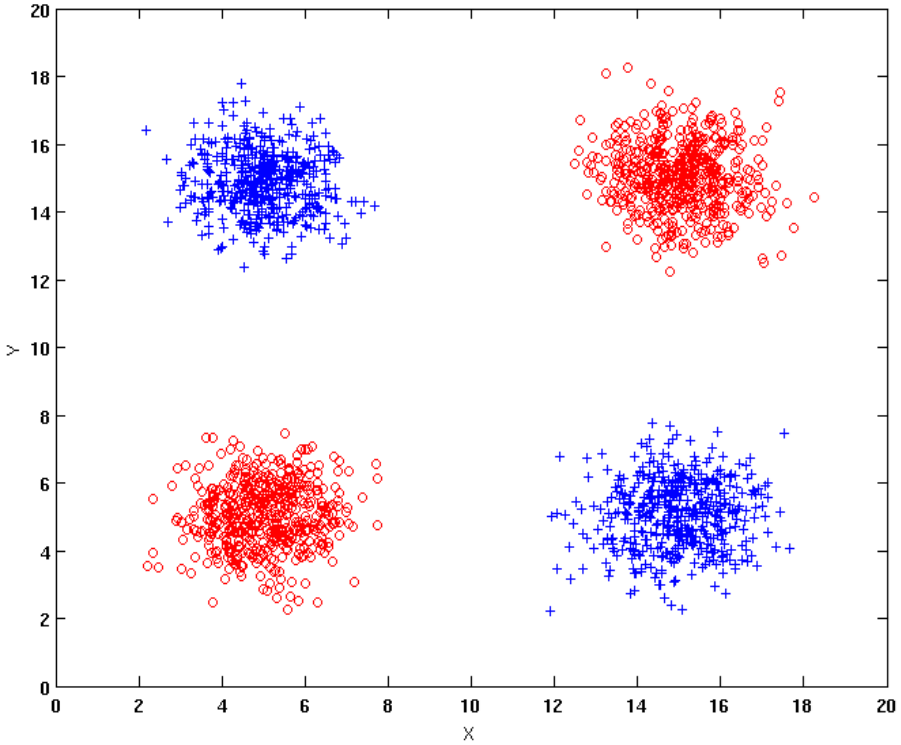
Effect of Multiple Comparison Procedure

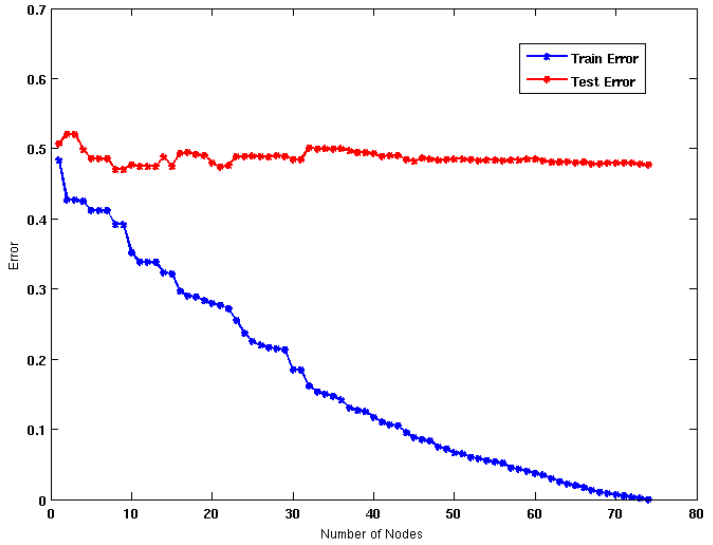
- Birçok algoritma aşağıdaki açgözlü stratejiyi (***greedy strategy***) kullanır:
  - İlk model: M
  - Alternatif model: M' = M $\cup$ γ, burada γ, modele eklenecek bir bileşendir (örneğin, bir karar ağacının test koşulu)
  - İyileştirme varsa M'yi tutun, Δ(M,M') > α

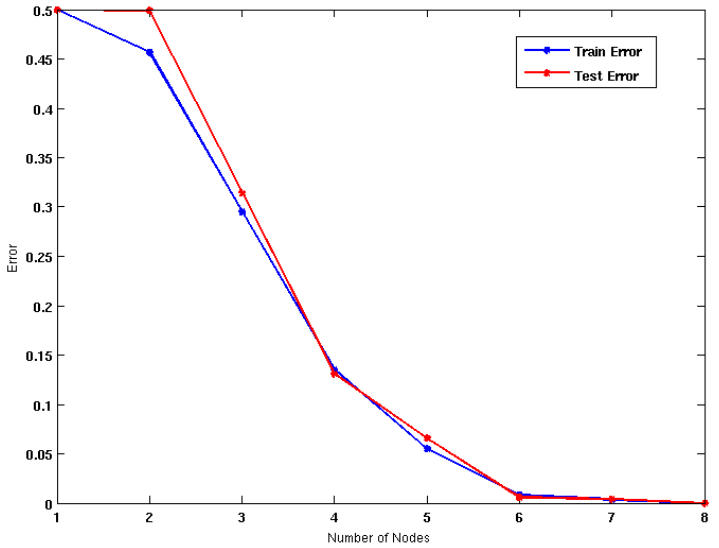
- Çoğu zaman, γ bir dizi alternatif bileşenden seçilir, $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_k\}$

- Birçok alternatif mevcutsa, modele istemeden alakasız bileşenler eklenebilir ve bu da modelin ezberlemesine (***overfitting***) neden olabilir.

Effect of Multiple Comparison - Example





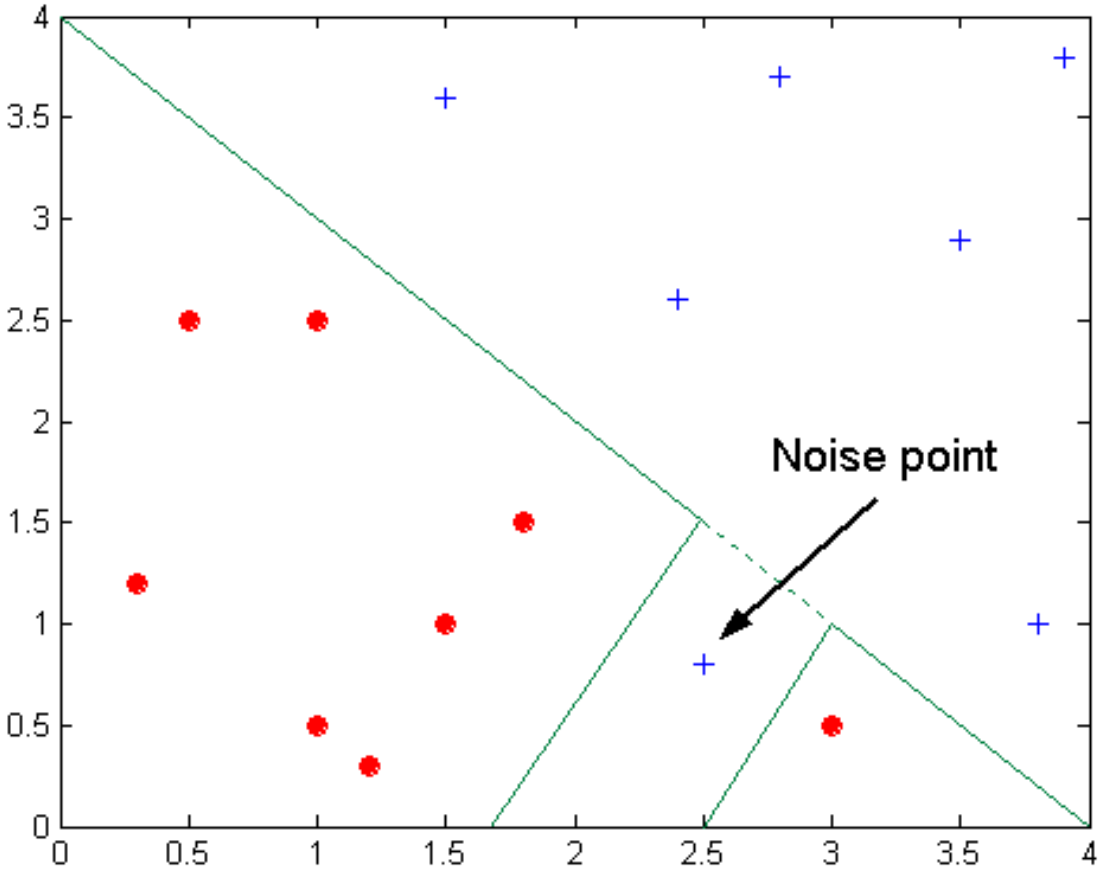


Use additional 100 noisy variables generated from a uniform distribution along with X and Y as attributes.

Use 30% of the data for training and 70% of the data for testing

Using only X and Y as attributes

Overfitting due to Noise



Decision boundary is distorted by noise point

Overfitting due to Noise

Table 4.3. An example training set for classifying mammals. Class labels with asterisk symbols represent mislabeled records.

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

Memelileri sınıflandırmak için kullanılan bir eğitim veri seti. Yıldız ile işaretli olalar yanlış etiketlenmiş kayıtlardır.

Overfitting due to Noise

Table 4.4. An example test set for classifying mammals.

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	cold-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

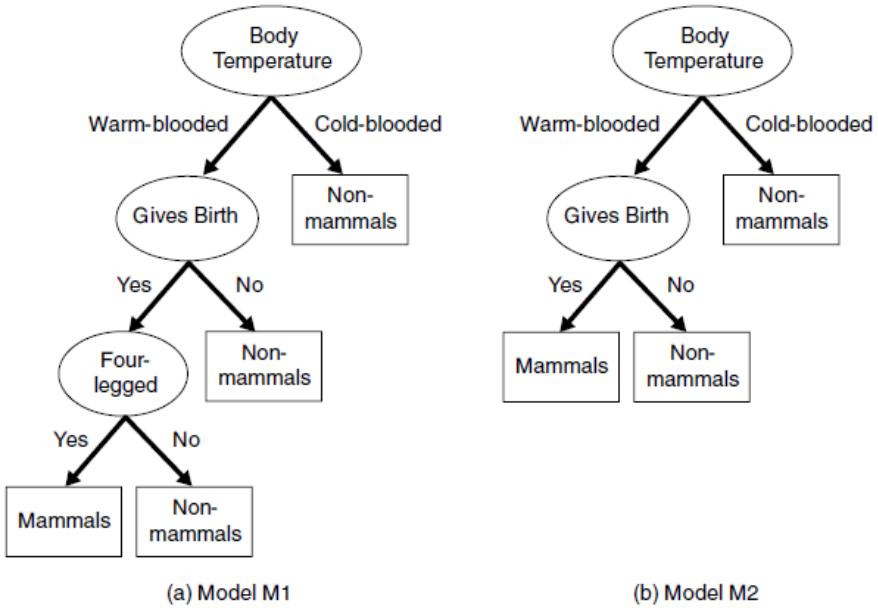


Figure 4.25. Decision tree induced from the data set shown in Table 4.3.

Test setinde daha düşük hata oranına sahip daha basit bir model var olduğu olduğu için, ilk karar ağacı ***M1*** 'in eğitim verilerini ezberlediği (**overfitted**) açıktır.

M1 modelindeki **Four-legged** öznitelik test koşulu yanlış etiketlenmiş eğitim kayıtlarına uyduğundan sahtedir (**spurious).** Bu da test setindeki kayıtların yanlış sınıflandırılmasına yol açar.

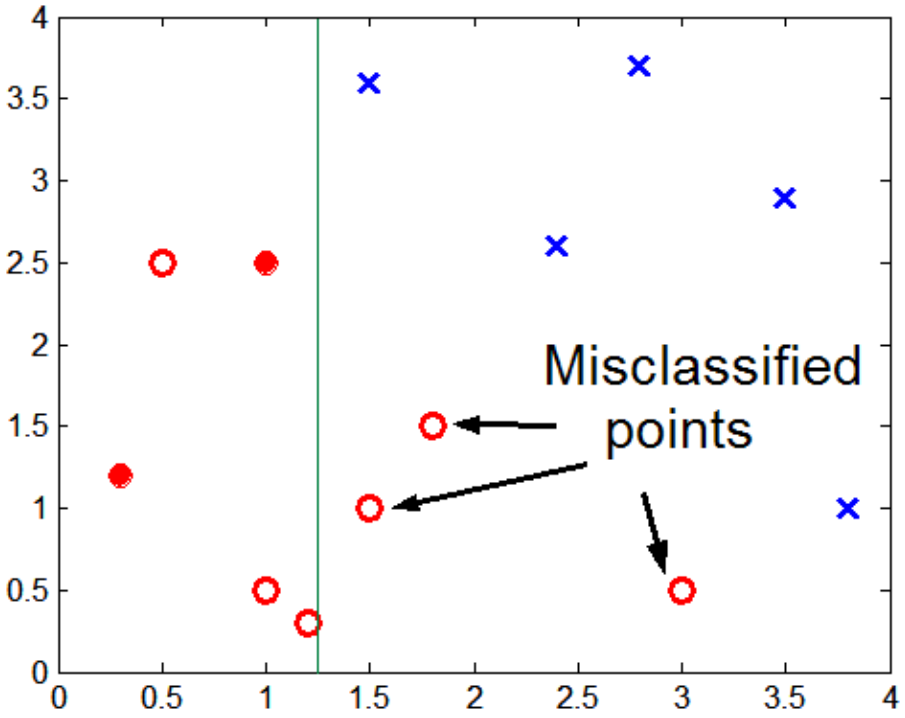
Model M1

Training error =0%
Test error =30%

Model M2

Training error =20%
Test error =10%

Overfitting due to Insufficient Examples



Diyagramın alt yarısındaki veri noktalarının yetersiz oluşu, o bölgenin sınıf etiketlerini doğru şekilde tahmin etmeyi zorlaştırır

- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

Notes on Overfitting

- Ezberleme/aşırı uyum (*Overfitting*), gerekenden daha karmaşık karar ağaçlarına neden olur

- Eğitim hatası (**Training error**), ağacın daha önce görülmemiş kayıtlar (**unseen records**) üzerinde ne kadar iyi performans göstereceğine dair iyi bir tahmin sağlamaz

- Genelleme hatalarını (*generalization error*) tahmin etmenin yollarına ihtiyacımız var

Model Selection

- Model oluşturma sırasında gerçekleştirilir
- Amaç, modelin aşırı karmaşık olmamasını sağlamaktır (ezberlemeyi önlemek için)
- Genelleme hatasını tahmin etmemiz/kestirmemiz gerekiyor
  - Using Validation Set (*Doğrulama veri seti kullanma*)
  - Incorporating Model Complexity (*Model karmaşıklığını dahil etme*)
  - Estimating Statistical Bounds (*İstatistiksel sınırların tahmin edilmesi*)

Using Validation Set

- Divide training data into two parts:
  - Training set:
    - use for model building
  - Validation set:
    - use for estimating generalization error
    - Note: validation set is not the same as test set

- Drawback:
  - Less data available for training

Incorporating Model Complexity

- Rationale: **Occam's Razor** (or **principle of parsimony**:)
  - Benzer genelleme hatalarının olduğu iki model verildiğinde, daha karmaşık model yerine basit modeli tercih etmelisiniz.
  - Karmaşık bir modelin, verilerdeki hatalarla yanlışlıkla uydurulma şansı daha yüksektir
  - Bu nedenle, bir modeli değerlendirirken model karmaşıklığı işin içerisine dahil edilmelidir

Gen. Error(Model) = Train. Error(Model, Train. Data) +
α x Complexity(Model)

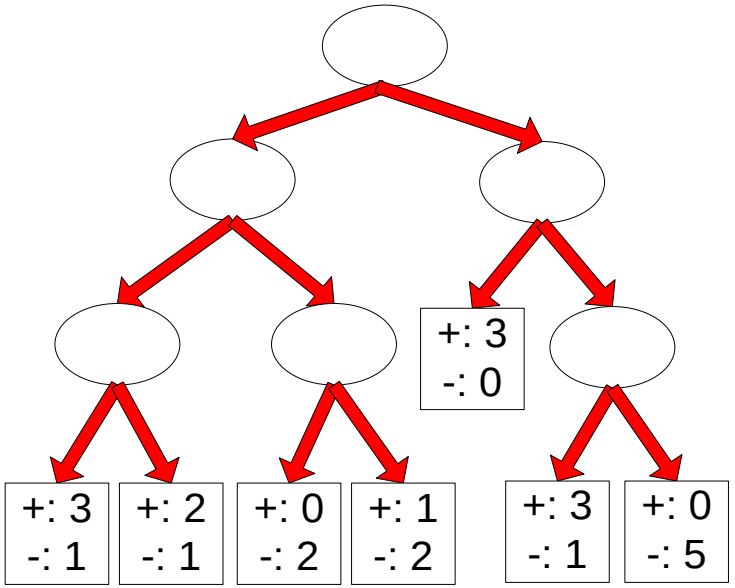
Estimating the Complexity of Decision Trees

- **Pessimistic Error Estimate** of decision tree T with k leaf nodes:

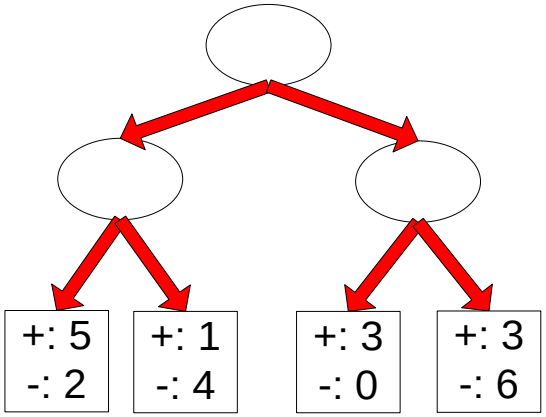
$$err_{gen}(T) = err(T) + \Omega \times \frac{k}{N_{train}}$$

  - err(T): error rate on all training records
  - Ω: trade-off hyper-parameter (similar to α)
    - Relative cost of adding a leaf node (penalty term for model complexity)
  - k: number of leaf nodes
  - N_{train}: total number of training records

Estimating the Complexity of Decision Trees: Example



Decision Tree, T_L



Decision Tree, T_R

$e(T_L) = 4/24$

$e(T_R) = 6/24$

$\Omega = 1$

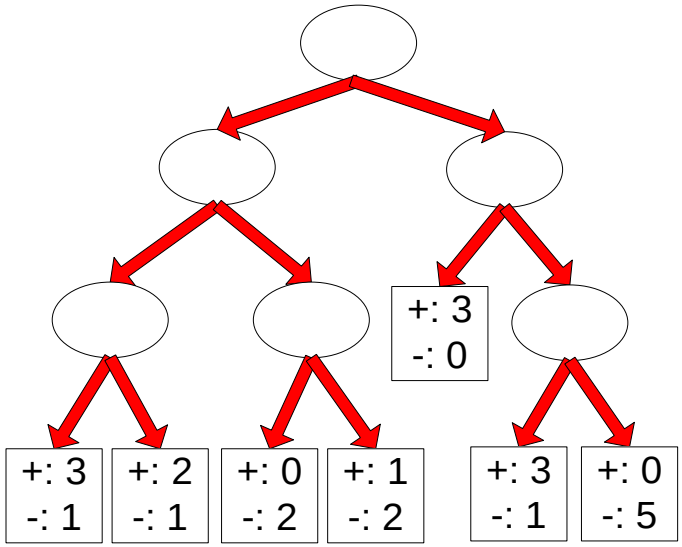
$e_{gen}(T_L) = 4/24 + 1*7/24 = 11/24 = 0.458$

$e_{gen}(T_R) = 6/24 + 1*4/24 = 10/24 = 0.417$

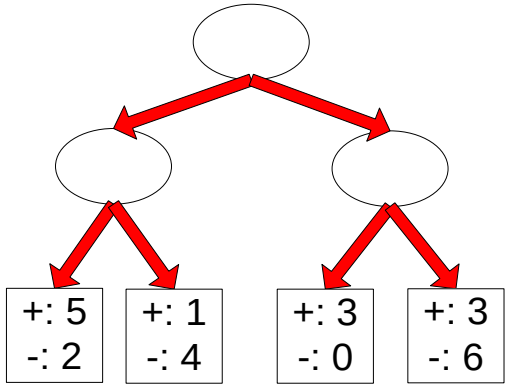
$$err_{gen}(T) = err(T) + \Omega \times \frac{k}{N_{train}}$$

Estimation of Generalization Errors

- Resubstitution Estimate:
  - Eğitim hatasını genelleme hatasının iyimser bir tahmini olarak kullanma
  - İyimser hata tahmini olarak anılır (optimistic error estimate)



Decision Tree, T_L



Decision Tree, T_R

$e(T_L) = 4/24$

$e(T_R) = 6/24$

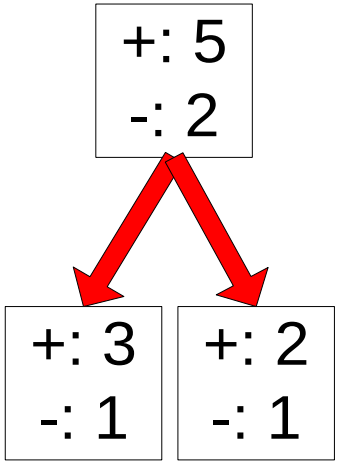
Estimating Statistical Bounds

By approximating a binomial distribution with a normal distribution, the following upper bound of the error rate *e* can be derived:

$$e'(N, e, \alpha) = \frac{e + \frac{z_{\alpha/2}^2}{2N} + z_{\alpha/2}\sqrt{\frac{e(1-e)}{N} + \frac{z_{\alpha/2}^2}{4N^2}}}{1 + \frac{z_{\alpha/2}^2}{N}}$$

α is the confidence level

N is the total number of training records used to compute *e*



Before splitting: e = 2/7, e'(7, 2/7, 0.25) = 0.503

e'(T) = 7 × 0.503 = 3.521

After splitting:

e(T_L) = 1/4, e'(4, 1/4, 0.25) = 0.537

e(T_R) = 1/3, e'(3, 1/3, 0.25) = 0.650

e'(T) = 4 × 0.537 + 3 × 0.650 = 4.098

$z_{\alpha/2}$ is the standardized value from a standard normal distribution

Therefore, do not split

Handling Overfitting in Decision Tree Induction

- Karar ağacı indükleme (*decision tree induction*) bağlamında modelin ezberlemesini önlemek için iki strateji
  - Pre-Pruning
  - Post-pruning

Model Selection for Decision Trees

- Pre-Pruning (Early Stopping Rule)
  - Algoritmayı tamamen büyümüş bir ağaç haline gelmeden durdurun
  - Bir düğüm için tipik durma koşulları(stopping conditions)
    - Tüm örnekler aynı sınıfa (the same class) aitse dur
    - Tüm öznitelik değerleri aynıysa dur
  - Daha kısıtlayıcı koşullar :
    - Örnek sayısı (number of instances) kullanıcı tarafından belirlenen bazı eşiklerin altındaysa dur
    - Örneklerin sınıf dağılımı mevcut özelliklerden bağımsızsa durun (Örn., χ^2 testi kullanarak)
    - **Mevcut düğümün** genişletilmesi impurity ölçütlerini (Örn., Gini veya information gain) iyileştirmiyorsa dur
    - Tahmini genelleme hatası belirli bir eşiğin altına düşerse dur

Model Selection for Decision Trees

- Post-pruning
  - Karar ağacını tamemen büyütün
  - Alt ağaç değişimi (*Subtree replacement*)
    - Karar ağacının düğümlerini aşağıdan yukarıya (bottom-up) doğru kırpın
    - Kırpmadan sonra genelleme hatası düzelirse, alt ağacı bir **yaprak düğüm (*leaf node*)** ile değiştirin
    - Yaprak düğümün sınıf etiketi, alt ağaçtaki örneklerin **çoğunluk sınıfından** (***majority class***) belirlenir
  - Subtree raising
    - Replace subtree with **most frequently used branch**

Example of Post-Pruning

Class = Yes	20
Class = No	10
Error = 10/30	

Training Error (Before splitting) = 10/30

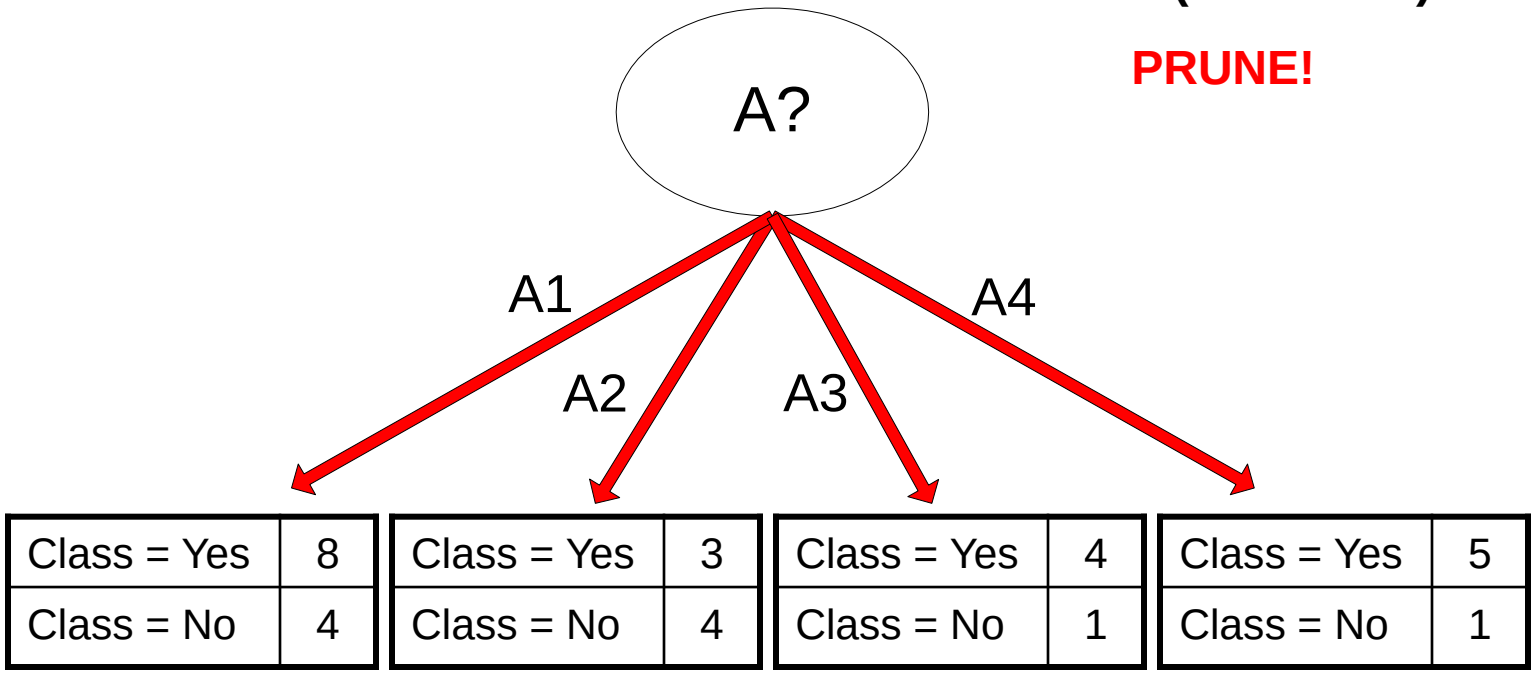
Pessimistic error = (10 + 0.5)/30 = 10.5/30

Training Error (After splitting) = 9/30

Pessimistic error (After splitting)

= (9 + 4 × 0.5)/30 = 11/30

PRUNE!



Class = Yes	8
Class = No	4

Class = Yes	3
Class = No	4

Class = Yes	4
Class = No	1

Class = Yes	5
Class = No	1

Examples of Post-pruning

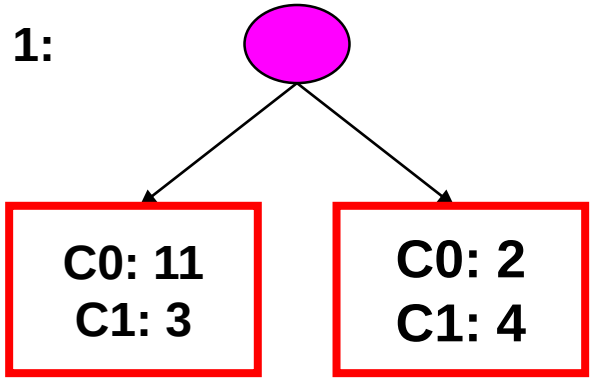
- Optimistic error?

  Don't prune for both cases

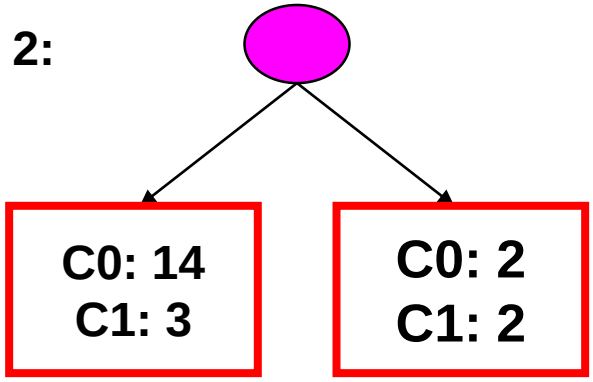
- Pessimistic error?

  Don't prune case 1, prune case 2

Case 1:



Case 2:

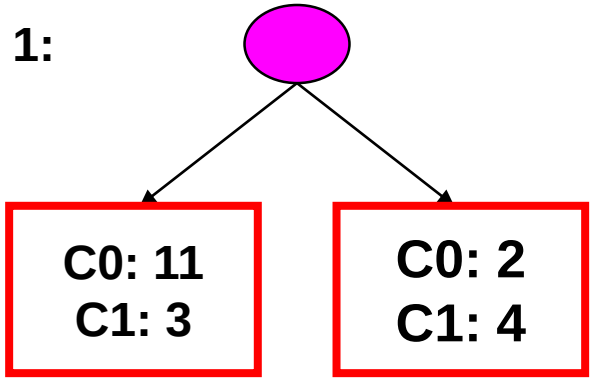


Examples of Post-pruning

– Optimistic error?

Class = C0	13
Class = C1	7
Error = 7/20	

Case 1:

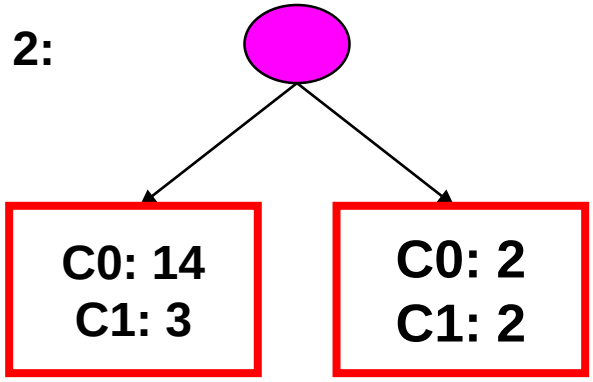


Optimistic error (After splitting) = 5/20

Don't prune for both cases

Class = C0	16
Class = C1	5
Error = 5/21	

Case 2:



Optimistic error (After splitting) = 5/21

Examples of Post-pruning

– Pessimistic error?

Class = C0	13
Class = C1	7
Error = 7/20	

Training Error (Before splitting) = 7/20

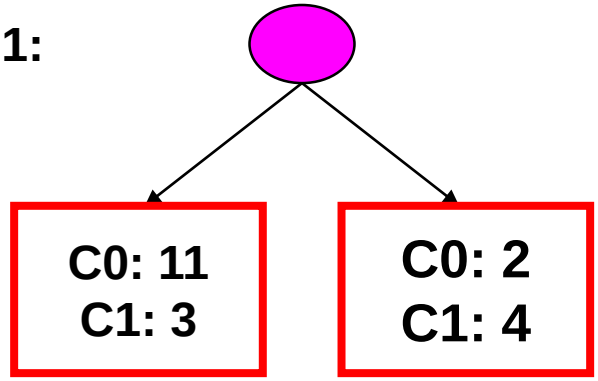
Pessimistic error = (7 + 0.5)/30 = 7.5/20

Training Error (After splitting) = 5/20

Pessimistic error (After splitting)

= (5 + 2 × 0.5)/20 = 6/20

Case 1:



DON'T PRUNE FOR CASE 1!

Class = C0	16
Class = C1	5
Error = 5/21	

Training Error (Before splitting) = 5/21

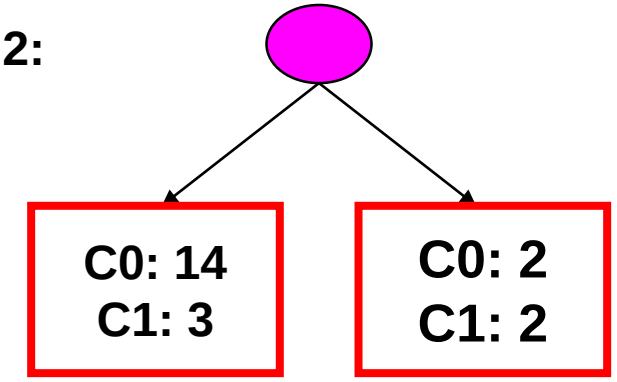
Pessimistic error = (5 + 0.5)/30 = 5.5/21

Training Error (After splitting) = 5/21

Pessimistic error (After splitting)

= (5 + 2 × 0.5)/21 = 6/21

Case 2:



PRUNE FOR CASE 2!

Examples of Post-pruning