Data Mining
Classification: Alternative Techniques

Lecture Notes for Chapter 4

Rule-Based

Introduction to Data Mining , 2nd Edition

by

Tan, Steinbach, Karpatne, Kumar

Rule-Based Classifier

- Kayıtları “if…then…” kurallarından oluşan bir koleksiyon kullanarak sınıflandırır

- Rule: (*Condition*) $\rightarrow$ *y*
  - burada
    - *Condition* (Koşul), özniteliklerin birleşimidir
    - *y*, sınıf etiketidir
  - *LHS*: rule antecedent (kural öncülü) or condition (koşul)
  - *RHS*: rule consequent (kural sonucu)
  - Sınıflandırma kurallarına örnekler:
    - (Blood Type=Warm) $\wedge$ (Lay Eggs=Yes) $\rightarrow$ Birds
    - (Taxable Income < 50K) $\wedge$ (Refund=Yes) $\rightarrow$ Evade=No

Rule-based Classifier (Example)

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
human	warm	yes	no	no	mammals
python	cold	no	no	no	reptiles
salmon	cold	no	no	yes	fishes
whale	warm	yes	no	yes	mammals
frog	cold	no	no	sometimes	amphibians
komodo	cold	no	no	no	reptiles
bat	warm	yes	yes	no	mammals
pigeon	warm	no	yes	no	birds
cat	warm	yes	no	no	mammals
leopard shark	cold	yes	no	yes	fishes
turtle	cold	no	no	sometimes	reptiles
penguin	warm	no	no	sometimes	birds
porcupine	warm	yes	no	no	mammals
eel	cold	no	no	yes	fishes
salamander	cold	no	no	sometimes	amphibians
gila monster	cold	no	no	no	reptiles
platypus	warm	no	no	no	mammals
owl	warm	no	yes	no	birds
dolphin	warm	yes	no	yes	mammals
eagle	warm	no	yes	no	birds

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Application of Rule-Based Classifier

- Bir *r* kuralı, ilgili örneğin öznitelikleri (attributes of the instance) kuralın koşulunu karşılıyorsa **x** örneğini kapsar (***covers***)

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
hawk	warm	no	yes	no	?
grizzly bear	warm	yes	no	no	?

The rule R1 covers a hawk => Bird

The rule R3 covers the grizzly bear => Mammal

Rule Coverage and Accuracy

- Bir sınıflandırma kuralının kalitesi, kapsam (**coverage**) ve doğruluk (**accuracy**) gibi ölçümler kullanılarak değerlendirilebilir.
- Bir veri kümesi *D* ve bir sınıflandırma kuralı verildiğinde $r : A \rightarrow y$
  - kuralın kapsamı (**coverage**), D veri kümesinde *r* kuralını tetikleyen kayıtların oranı olarak tanımlanır.
  - Öte yandan, kuralın doğruluğu (**accuracy**) veya güven faktörü (**confidence factor**), sınıf etiketleri *y*'ye eşit olan *r* tarafından tetiklenen kayıtların oranı olarak tanımlanır.
  - Bu ölçülerin tanımları

$$\text{Coverage}(r) = \frac{|A|}{|D|} = \frac{|LHS|}{n}$$

$$\text{Accuracy}(r) = \frac{|A \cap y|}{|A|} = \frac{|LHS \cap RHS|}{|LHS|}$$

Burada $|A|$ kural öncülünü (***rule antecedent***) karşılayan kayıtların sayısı, $|A \cap y|$ hem öncülü hem de sonucu (both ***antecedent*** *and* ***consequent***) karşılayan kayıtların sayısıdır ve $|D|$ toplam kayıt sayısıdır.

Rule Coverage and Accuracy

- **Coverage** of a rule:
  - Bir kuralın öncülünü (*antecedent of a rule*) karşılayan kayıtların oranı
- **Accuracy** of a rule:
  - Bir kuralın sonucunu da öncülünü de karşılayan kayıtların oranı

Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	**Single**	125K	**No**
2	No	Married	100K	**No**
3	No	**Single**	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	**Single**	85K	**Yes**
9	No	Married	75K	**No**
10	No	**Single**	90K	**Yes**

(Status=Single) → No

Coverage = 40%, Accuracy = 50%

How does Rule-based Classifier Work?

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?

A lemur triggers rule R3, so it is classified as a mammal

A turtle triggers both R4 and R5

A dogfish shark triggers none of the rules

Two important properties of the rule set

- Önceki örnek, kurala dayalı bir sınıflandırıcı tarafından oluşturulan kural kümesinin iki önemli özelliğini göstermektedir.
- **Mutually Exclusive Rules (*Karşılıklı/Birbirlerini Dışlayan Kurallar*)**
  - Eğer aynı kayıt tarafından *R*'deki herhangi iki kural (no two rules) tetiklenmezse, bu *R* kural kümesindeki kurallar birbirini dışlar (**mutually exclusive**).
  - Bu özellik, her kaydın *R*'de **en fazla** bir kural tarafından kapsanmasını sağlar. Karşılıklı dışlayıcı bir kural kümesi örneği Tablo 5.3'te gösterilmiştir.
- **Exhaustive Rules (*Eksiksiz/Kapsamlı Kurallar*)**
  - Her öznitelik değeri kombinasyonu için bir kural varsa, bir *R* kural grubu eksiksiz kapsamaya(**exhaustive coverage)** sahiptir.
  - Bu özellik, her kaydın *R*'deki **en az** bir kural tarafından kapsanmasını sağlar. «Body Temperature» ve «Gives Birth» özniteliklerinin ikili değişkenler olduğunu varsayarsak, Tablo 5.3'te gösterilen kural seti eksiksiz bir kapsamaya sahiptir.

Table 5.3. Example of a mutually exclusive and exhaustive rule set.

r_1: (Body Temperature = cold-blooded) $\longrightarrow$ Non-mammals
r_2: (Body Temperature = warm-blooded) $\wedge$ (Gives Birth = yes) $\longrightarrow$ Mammals
r_3: (Body Temperature = warm-blooded) $\wedge$ (Gives Birth = no) $\longrightarrow$ Non-mammals

Characteristics of Rule Sets: Strategy 1

- Mutually exclusive rules
  - Kurallar birbirinden bağımsızsa sınıflandırıcı, birbirini dışlayan kurallar içerir
  - Her kayıt en fazla bir kural tarafından kapsanır

- Exhaustive rules
  - Sınıflandırıcı, öznitelik değerlerinin her olası kombinasyonunu hesaba katıyorsa eksiksiz bir kapsamaya sahiptir
  - Her kayıt en az bir kural tarafından kapsanmaktadır.

Characteristics of Rule Sets: Strategy 2

- Rules are not mutually exclusive
  - Bir kayıt birden fazla kuralı tetikleyebilir
  - Çözüm?
    - *Ordered rule set*
    - *Unordered rule set – use voting schemes*

- Rules are not exhaustive
  - Bir kayıt herhangi bir kuralı tetiklemeyebilir
  - Çözüm?
    - *Use a default class*

Ordered Rule Set

Öncelik (**priority**) birçok şekilde tanımlanabilir (ör. doğruluk, kapsam, toplam açıklama uzunluğu veya kuralların oluşturulma sırasına göre).

- Kurallar, önceliğe (*priority*) göre sıralanır
  - Sıralı bir kural kümesi (ordered rule set), karar listesi olarak bilinir
- Sınıflandırıcıya bir test kaydı sunulduğunda
  - Tetiklediği en yüksek dereceli kuralın sınıf etiketine atanır
  - Kurallardan hiçbiri tetiklenmediyse, varsayılan sınıfa (*default class*) atanır

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
turtle	cold	no	no	sometimes	?

Unordered Rule Set

- Bu yaklaşım, bir test kaydının birden çok sınıflandırma kuralını tetiklemesine izin verir ve her kuralın sonucunu **belirli bir sınıf için bir oy (*vote*)** olarak değerlendirir.
- Oylar daha sonra test kaydının sınıf etiketini belirlemek için toplanır.
- Kayıt genellikle en çok oyu (**highest number of votes**) alan sınıfa atanır.
- Bazı durumlarda oylama, kuralın doğruluğuna göre ağırlıklandırılabilir.

Rule Ordering Schemes

- Rule-based ordering
  - Bireysel kurallar kalitelerine göre sıralanır
- Class-based ordering
  - Aynı sınıfa ait kurallar birlikte görünür

Rule-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Class-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Married}) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

Rule-based ordering scheme

- bireysel kuralları bazı kural kalite ölçülerine göre sıralar.
- her test kaydının, kendisini kapsayan "en iyi" kurala göre sınıflandırılmasını sağlar.
- potansiyel bir dezavantajı var
  - Daha düşük dereceli kuralları yorumlamak çok daha zordur çünkü kendilerinden önceki kuralların değillemesini varsayarlar.

Class-based ordering scheme

- Bu yaklaşımda, aynı sınıfa ait olan kurallar, *R* kural kümesinde **birlikte görünür**.
- Kurallar daha sonra **sınıf bilgilerine** göre toplu olarak sıralanır.
- Aynı sınıftaki kurallar arasında göreceli sıralama önemli değildir;
  - kurallardan biri tetiklendiği sürece, sınıf test kaydına atanacaktır.
- Bu, kural yorumlamasını biraz daha kolaylaştırır.
- İyi bilinen kural tabanlı (*rule-based)* sınıflandırıcıların çoğu (**C4.5rules** ve **RIPPER** gibi) sınıf tabanlı sıralama şemasını (**class-based ordering scheme**) kullandığından, bu bölümün geri kalanındaki tartışma esas olarak bu tür sıralama şemasına odaklanmaktadır.

Building Classification Rules

Sınıflandırma kuralları çıkarımı için iki geniş yöntem sınıfı vardır :

- Direct Method:
  - Kuralları doğrudan verilerden çıkarır
  - Örnekler: RIPPER, CN2, Holte's 1R

Doğrudan yöntemler (Direct methods), öznitelik uzayını daha küçük altuzaylara böler, böylece bir alt-uzaya ait olan tüm kayıtlar tek bir sınıflandırma kuralı kullanılarak sınıflandırılabilir.

- Indirect Method:
  - Kuralları diğer sınıflandırma modellerinden çıkarır (e.g. decision trees, neural networks, etc).
  - Örnekler : C4.5rules

Direct Method: Sequential Covering

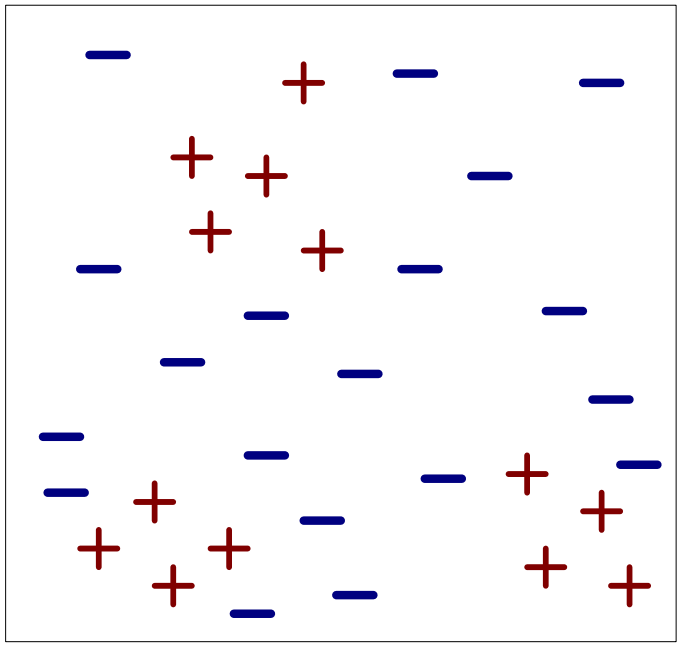
1. Start from an empty rule
2. Grow a rule using the Learn-One-Rule function
3. Remove training records covered by the rule
4. The rule is added to the bottom of the decision list *R*.
5. Repeat Step (2) and (3) until stopping criterion is met

A rule is desirable if it covers most of the positive examples and none (or very few) of the negative examples.

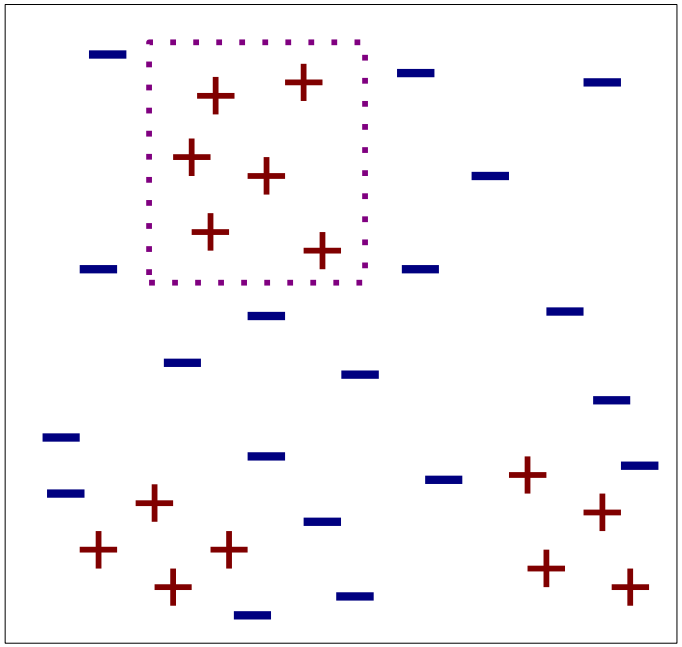
Learn-One-Rule Function
Learn-One-Rule fonksiyonunun amacı, eğitim setindeki pozitif örneklerin çoğunu ve negatif örneklerin hiçbirini (veya çok azını) kapsayan bir sınıflandırma kuralı çıkarmaktır. Bununla birlikte, arama uzayının üstel boyutu göz önüne alındığında, optimal bir kural bulmak hesaplama açısından pahalıdır.

Example of Sequential Covering

Pozitif örneklerin en büyük kısmını kapsadığı için önce R1 kuralı çıkarılır.



(i) Original Data

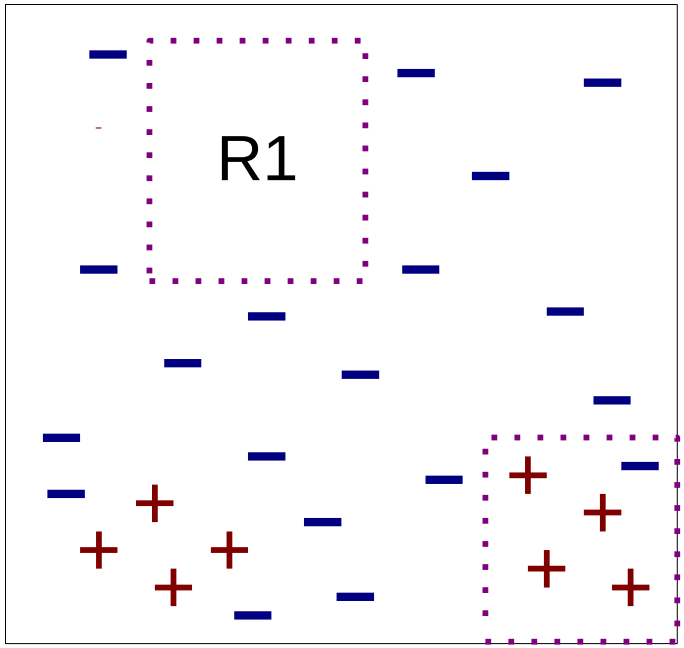


(ii) Step 1

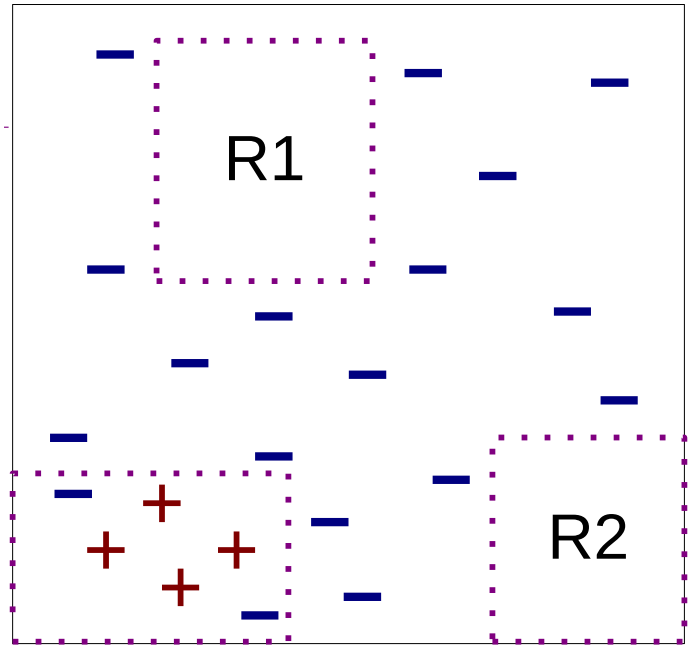
Böyle bir kural bulunduğunda, kuralın kapsadığı eğitim kayıtları elimine edilir.

Example of Sequential Covering…

R1 **tarafından kapsanan tüm eğitim kayıtları kaldırılır ve daha sonra algoritma bir sonraki en iyi kuralı, yani *R2*'yi aramaya devam eder.**



(iii) Step 2

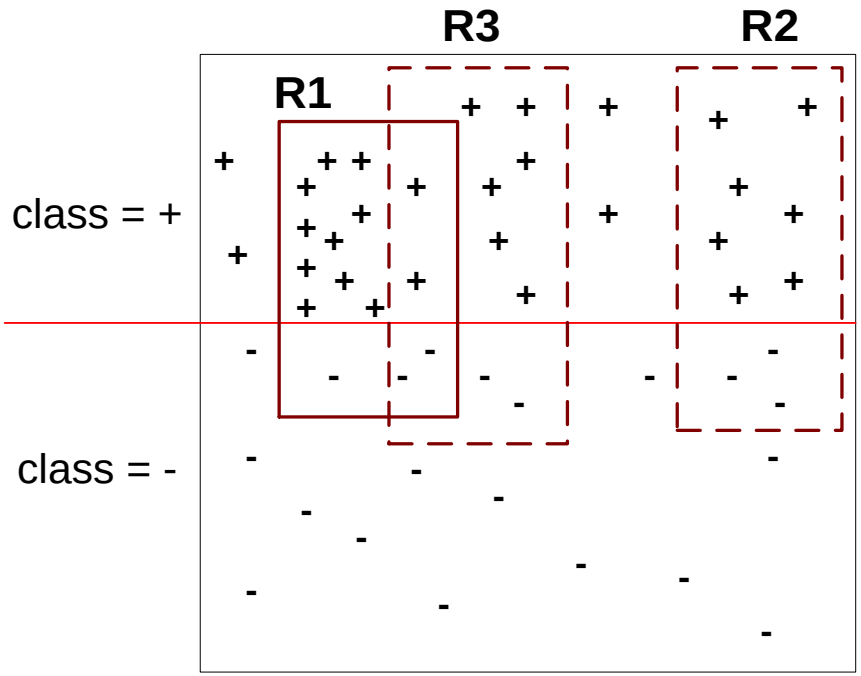


(iv) Step 3

Böyle bir kural bulunduğunda, kuralın kapsadığı eğitim kayıtları elimine edilir.

Instance Elimination

- Örnekleri (*instances*) neden ortadan kaldırmamız gerekiyor?
  - Aksi takdirde, sonraki kural önceki kuralla aynı olur
- Pozitif örnekleri neden kaldırıyoruz?
  - Sonraki kuralın farklı olmasını sağlama alıyor
- Negatif örnekleri neden kaldırıyoruz?
  - Kuralın doğruluğunun eksik değerlemesini önler
  - Diyagramda R2 ve R3 kurallarını karşılaştırın



Şekil 5.4. «sequential covering algorithm» ile eğitim kayıtlarının ortadan kaldırılması (elimination). R1, R2 ve R3, üç farklı kural tarafından kapsanan bölgeleri temsil eder

Şekil 5.4, 29 olumlu örnek ve 21 olumsuz örnek içeren bir veri kümesinden çıkarılan üç olası kuralı, *R1*, *R2* ve *R3*'ü gösterir. ***R1***, ***R2*** ve ***R3***'ün doğrulukları sırasıyla 12/15 (**%80**), 7/10 (**%70**) ve 8 / 12'dir (**%66.7**). En yüksek doğruluğa sahip olduğu için ilk önce *R1* oluşturulur.

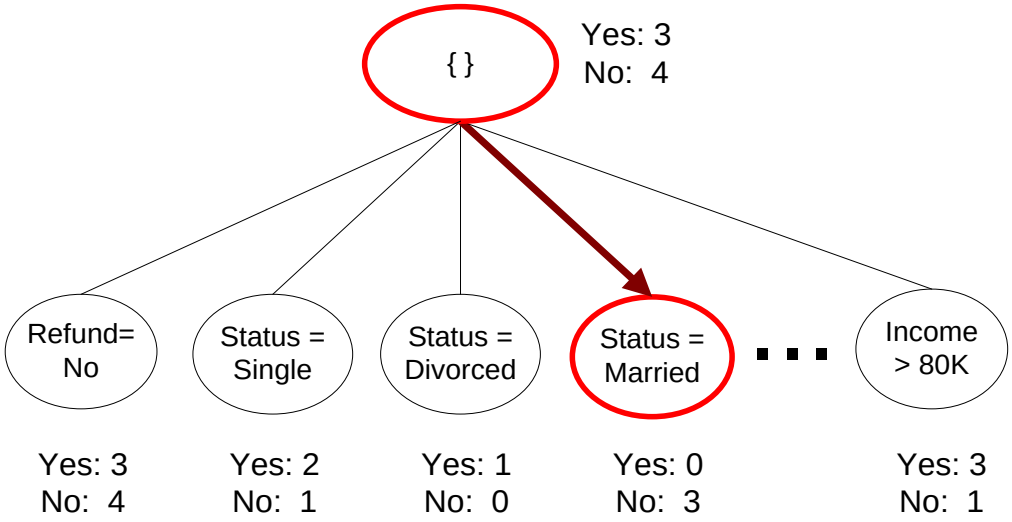
Instance Elimination

R1, *R2* ve *R3*'ün doğrulukları sırasıyla 12/15 (%80), 7/10 (%70) ve 8 / 12'dir (%66.7). En yüksek doğruluğa sahip olduğu için ilk önce *R1* oluşturulur.

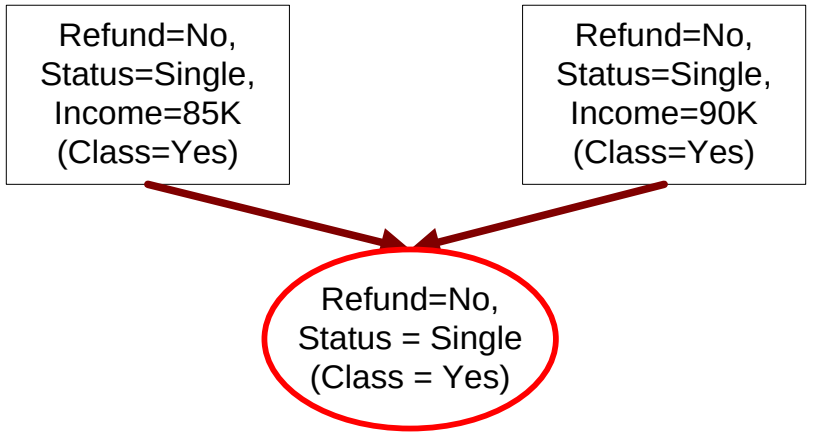
En yüksek doğruluğa sahip olduğu için önce *R1* oluşturulur. *R1* oluşturulduktan sonra, algoritma tarafından oluşturulan bir sonraki kuralın *R1*'den farklı olması için kuralın kapsadığı pozitif örneklerin kaldırılması gerektiği açıktır. Ardından, algoritmaya *R2* veya *R3* üretme seçeneği verildiğini varsayalım. *R2*, *R3'ten* daha yüksek doğruluğa sahip olsa da, *R1* ve *R3* birlikte 18 olumlu örneği ve 5 olumsuz örneği kapsar (genel olarak %78,3 doğrulukla sonuçlanır), oysa *R1* ve *R2* birlikte 19 olumlu örneği ve 6 olumsuz örneği kapsar (genel bir sonuçla sonuçlanır). %76 doğruluk). *R2* veya *R3'ün* doğruluk üzerindeki artan etkisi, doğrulukları hesaplanmadan önce *R1* kapsamındaki olumlu ve olumsuz örnekler kaldırıldığında daha belirgindir. Özellikle, *R1'in* kapsadığı olumlu örnekler çıkarılmazsa, *R3'ün* etkin doğruluğunu abartabiliriz ve olumsuz örnekler çıkarılmazsa, *R3'ün* doğruluğunu küçümseyebiliriz. İkinci durumda, *R3* tarafından işlenen yanlış pozitif hataların yarısı bir önceki kural olan *R1* tarafından zaten açıklanmış olsa bile, *R2'yi R3'e* tercih edebiliriz.

Rule Growing

- Two common strategies



(a) General-to-specific



(b) Specific-to-general

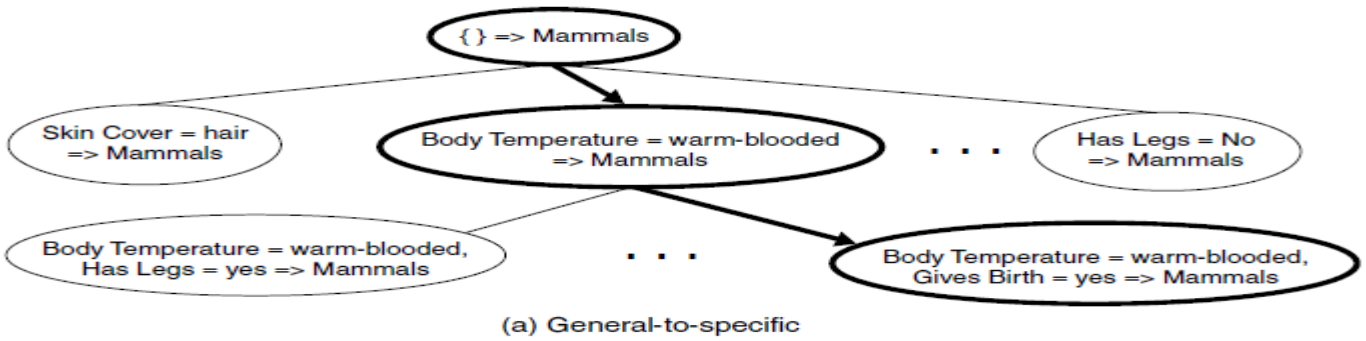
Recall: How to determine the Best Split

- Greedy approach:
  - Nodes with purer class distribution are preferred
- Need a measure of node impurity:

High degree of impurity **Low degree of impurity**

Rule Growing strategy: general-to-specific

- Bir başlangıç kuralı $r : \{\} \rightarrow y$ oluşturulur, burada sol taraf boş bir kümedir ve sağ taraf hedef sınıfı (*target class*) içerir.
- Bu haliyle kuralın kalitesi düşük (poor quality) çünkü eğitim setindeki tüm örnekleri kapsar.
- Kuralın kalitesini iyileştirmek için sonradan yeni bağlaçlar (**new conjuncts**) eklenir
- **Body Temperature=warm-blooded** bağlacı kural öncülünü oluşturmak için başlangıçta seçilir.
- Algoritma daha sonra olası tüm adayları araştırır ve greedy mantığıyla bir sonraki bağlacı, **Gives Birth=yes** kural öncülüne eklenecek olarak seçer.



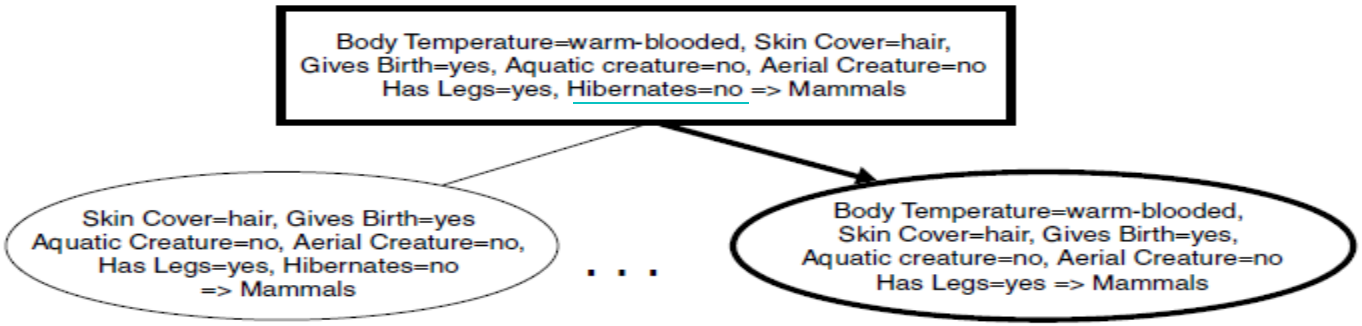
(a) General-to-specific

Bu süreç, durma kriteri karşılanana kadar devam eder (örneğin, eklenen bağlaç **kuralın kalitesini iyileştirmediğinde**).

The general-to-specific rule-growing strategy for the vertebrate classification problem.

Rule Growing strategy: specific-to-general

- Pozitif örneklerden biri, kural geliştirme süreci için ilk çekirdek (initial seed) olarak rastgele seçilir.
- İyileştirme adımı sırasında kural, bağlaçlarından biri kaldırılarak genelleştirilir
  - ➢ böylece **daha fazla sayıda pozitif örnekleri** kapsayabilir.
- İlk çekirdek olarak memeliler için pozitif bir örnek seçildiğini varsayalım.
  - Başlangıç kuralı, çekirdek öznitelik değerleriyle aynı bağlaçları içerir.
- Kapsamını (coverage) iyileştirmek için, kural **Hibernate=no** bağlacını kaldırılarak genelleştirilir.



(b) Specific-to-general

Rafine etme adımı, durma kriteri karşılanana kadar tekrar edilir, örneğin, **kural negatif örnekleri kapsamaya başladığında**

the specific-togeneral approach for the vertebrate classification problem.

Rule Evaluation

- Kural geliştirme sürecinde hangi bağlacın eklenmesi (veya kaldırılması) gerektiğini belirlemek için bir değerlendirme ölçütü gereklidir.
- **Accuracy** bariz bir seçimdir çünkü kural tarafından doğru bir şekilde sınıflandırılan eğitim örneklerinin oranını açık bir şekilde ölçer.
- Ancak, **accuracy**'nin olası handikapı şudur:
  - kuralın kapsamını (coverage) hesaba katmaz.
- Örneğin, aşağıdakileri içeren bir eğitim seti düşünün:
  - 60 pozitif örnek ve
  - 100 negatif örnek.
- Aşağıdaki iki aday kuralın bize verildiğini varsayalım:
  - Rule *r*1: 50 pozitif örnek ve 5 negatif örnek içerir,
  - Rule *r*2: 2 pozitif örneği içerir ve hiç negatif örnek içermez

Rule Evaluation

- Rule r_1 : covers 50 positive examples and 5 negative examples,
- Rule r_2 : covers 2 positive examples and no negative examples.

- r_1 ve r_2 için doğruluklar (accuracies) sırasıyla% 90,9 ve % 100'dür.

- Ancak, doğruluğu düşük olmasına rağmen r_1 daha iyi bir kuraldır.
- r_2 için yüksek doğruluk potansiyel olarak sahtedir (spurious) çünkü **kuralın** kapsamı (**coverage) çok düşüktür**.
- Bunu halletmek için aşağıdaki yaklaşım kullanılabilir

Rule Evaluation

- Foil's Information Gain

FOIL: First Order Inductive Learner – an early rule-based learning algorithm

  - R0: {} => class (initial rule)
  - R1: {A} => class (rule after adding conjunct)
  - Gain(R0, R1) = $t \left(log_2 \frac{p1}{p1+n1} - log_2 \frac{p0}{p0+n0} \right)$
  - where t: number of positive instances covered by both R0 and R1

    p0: number of positive instances covered by R0

    n0: number of negative instances covered by R0

    p1: number of positive instances covered by R1

    n1: number of negative instances covered by R1

- Since the measure is proportional to p_1 and $p_1/(p_1+n_1)$, it prefers rules that have high support count and accuracy.

Rule Evaluation example by FOIL's info. gain

$$\text{FOIL's information gain} = p_1 \times \left(\log_2 \frac{p_1}{p_1 + n_1} - \log_2 \frac{p_0}{p_0 + n_0} \right)$$

p0=60, n0=100

Rule *r*1 : covers 50 positive examples and 5 negative examples, i.e.

p1=50, n1=5

FOIL's info gain=

$$\mathbf{50}\left(log_2 \frac{\mathbf{50}}{\mathbf{55}} - log_2 \frac{\mathbf{60}}{\mathbf{160}}\right)\mathbf{=63,87}$$

p0=60, n0=100

Rule *r*2 : covers 2 positive examples and no negative examples, i.e.

p1=2, n1=0

FOIL's info gain=

$$\mathbf{2}\left(log_2 \frac{\mathbf{2}}{\mathbf{2}} - log_2 \frac{\mathbf{60}}{\mathbf{160}}\right)\mathbf{=2,83}$$

Therefore, r_1 is a better rule than r_2.

Stopping Criterion and Rule Pruning

- Stopping criterion
  - Kazancı (*gain)* hesaplayın
  - Kazanç ihmal edilebilecek kadar küçük ise, yeni kuralı atın
- Rule Pruning
  - Karar ağaçlarının sonradan budamasına (*post-pruning)* benzer
  - Reduced Error Pruning:
    - Kuraldaki bağlaçlardan birini kaldırın
    - Budamadan önce ve sonra doğrulama (on *validation set before and after pruning*) setindeki hata oranını karşılaştırın
    - Hata iyileşirse, bağlacı budayın

Summary of Direct Method

- Grow a single rule
- Remove Instances from rule
- Prune the rule (if necessary)
- Add rule to Current Rule Set
- Repeat

Direct Method: RIPPER

- RIPPER, yaygın olarak kullanılan bir kural indükleme (*induction*) algoritmasıdır.
- Bu algoritma, eğitim örneklerinin sayısıyla neredeyse doğrusal olarak ölçeklenir ve
  - özellikle dengesiz sınıf dağılımlarına (**imbalanced class distributions**) sahip veri kümelerinden modeller oluşturmak için uygundur.
- RIPPER, modelin ezberlemesini önlemek için bir doğrulama seti kullandığından gürültülü veri kümeleriyle (**noisy data sets**) de iyi çalışır.

Direct Method: RIPPER

- 2 sınıflı problem için, sınıflardan birini pozitif sınıf olarak seçin ve diğerini negatif sınıf olarak
  - Pozitif sınıf için kuralları öğrenin (the minority)
  - Negatif sınıf (the majority) varsayılan sınıf olacaktır
- Çok sınıflı (*multi-class*) problem için
  - Sınıfları artan sınıf yaygınlığına göre sıralayın (belirli bir sınıfa ait örneklerin oranı, yani **frekans**)
  - Önce en küçük sınıf için kural kümesini öğrenin, gerisini negatif sınıf olarak değerlendirin
  - Pozitif sınıf olarak sonraki en küçük sınıfla tekrarlayın

Direct Method: RIPPER

- Growing a rule:
  - Boş kuraldan başlayın (RIPPER, genelden özele (*general-to-specific strategy*) bir strateji kullanır)
  - FOIL'ın bilgi kazancını iyileştirdikleri sürece bağlaçlar ekleyin
  - Kural artık negatif örnekleri kapsamadığında eklemeyi durdurun
  - «incremental reduced error pruning» kullanarak kuralı hemen budayın (Yeni kural daha sonra doğrulama setindeki performansına göre budanır.)
  - Measure for pruning: v = (p-n)/(p+n)
    - p: validation setinde kuralın kapsadığı pozitif örneklerin sayısı
    - n: validation setinde kuralın kapsadığı negatif örneklerin sayısı
  - Pruning method: v'yi en maksimize eden son koşullar dizisini silin (Budama, kurala eklenen son bağlaçtan başlayarak yapılır.)

Direct Method: RIPPER

- Building a Rule Set:
  - Use sequential covering algorithm
    - Mevcut pozitif örnekler kümesini kapsayan en iyi kuralı bulur
    - Kuralın kapsadığı hem pozitif hem de negatif örnekleri ortadan kaldırır
  - Kural kümesine her kural eklendiğinde, yeni açıklama uzunluğunu (description length) hesaplanır
    - Yeni açıklama uzunluğu şimdiye kadar elde edilen en küçük açıklama uzunluğundan *d* bit daha uzun olduğunda yeni kural eklemeyi durdurur

Yeni kural, kural kümesinin toplam açıklama uzunluğunu en az *d* bit arttırırsa, RIPPER kural kümesine kural eklemeyi durdurur (varsayılan olarak, *d* 64 bit seçilir).

Indirect Methods

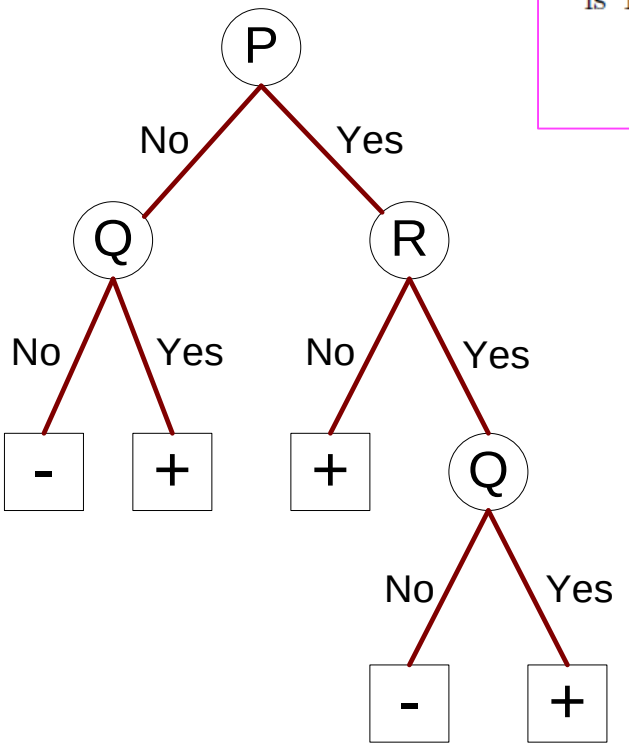
Example 5.2. Consider the following three rules from Figure 5.5:

$r2 : (\mathrm{P} = \mathrm{No}) \wedge (\mathrm{Q} = \mathrm{Yes}) \longrightarrow +$
$r3 : (\mathrm{P} = \mathrm{Yes}) \wedge (\mathrm{R} = \mathrm{No}) \longrightarrow +$
$r5 : (\mathrm{P} = \mathrm{Yes}) \wedge (\mathrm{R} = \mathrm{Yes}) \wedge (\mathrm{Q} = \mathrm{Yes}) \longrightarrow +$

Observe that the rule set always predicts a positive class when the value of Q is Yes. Therefore, we may simplify the rules as follows:

$r2'$: $(\mathrm{Q} = \mathrm{Yes}) \longrightarrow +$
$r3$: $(\mathrm{P} = \mathrm{Yes}) \wedge (\mathrm{R} = \mathrm{No}) \longrightarrow +$.

Decision Tree



Rule Set

r1: (P=No,Q=No) ==> -
r2: (P=No,Q=Yes) ==> +
r3: (P=Yes,R=No) ==> +
r4: (P=Yes,R=Yes,Q=No) ==> -
r5: (P=Yes,R=Yes,Q=Yes) ==> +

Converting a decision tree into classification rules.

Indirect Method: C4.5rules

- Budanmamış bir karar ağacından kuralları çıkarır
- Her kural için, r: A $\rightarrow$ y,
  - alternatif bir kural düşünün r′: A′ $\rightarrow$ y. Burada A′ başlangıçtaki kuraldaki bağlaçlardan birinin kaldırılmasıyla elde edilir.
  - r'nin karamsar hata oranını (*pessimistic error rate*) tüm r'lere karşılaştırın
  - Alternatif kurallardan biri daha düşük karamsar hata oranına sahipse budayın
  - Genelleme hatasını (*generalization error*) artık iyileştiremeyene kadar tekrarlayın

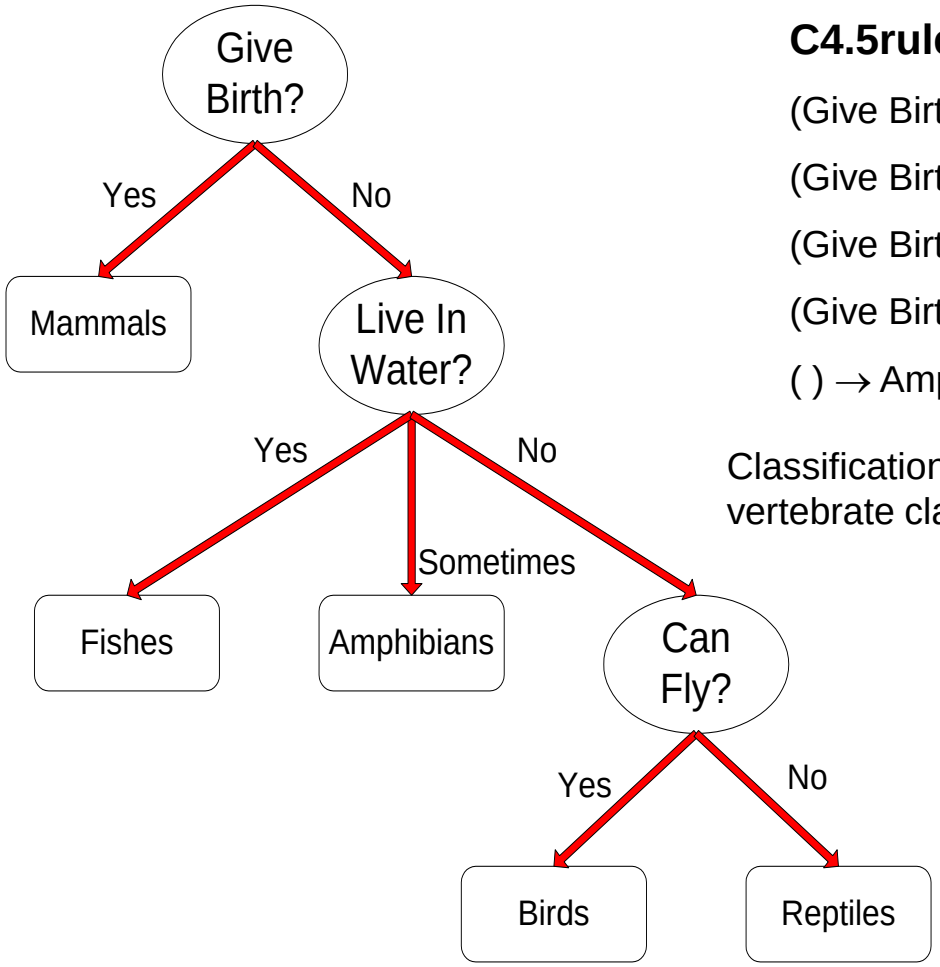
Indirect Method: C4.5rules

- Kuralları sıralamak yerine, kuralların alt kümelerini sıralayın (class ordering)
  - Her alt küme, aynı kural sonucunu (class) içeren bir kurallar koleksiyonudur.
  - Her alt kümenin açıklama uzunluğunu hesaplayın(description length)
    - Description length = L(error) + g L(model)
    - g, bir kural kümesindeki gereksiz özniteliklerin varlığını hesaba katan bir parametredir (varsayılan değer = 0,5)

Example

Name	Give Birth	Lay Eggs	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	no	yes	mammals
python	no	yes	no	no	no	reptiles
salmon	no	yes	no	yes	no	fishes
whale	yes	no	no	yes	no	mammals
frog	no	yes	no	sometimes	yes	amphibians
komodo	no	yes	no	no	yes	reptiles
bat	yes	no	yes	no	yes	mammals
pigeon	no	yes	yes	no	yes	birds
cat	yes	no	no	no	yes	mammals
leopard shark	yes	no	no	yes	no	fishes
turtle	no	yes	no	sometimes	yes	reptiles
penguin	no	yes	no	sometimes	yes	birds
porcupine	yes	no	no	no	yes	mammals
eel	no	yes	no	yes	no	fishes
salamander	no	yes	no	sometimes	yes	amphibians
gila monster	no	yes	no	no	yes	reptiles
platypus	no	yes	no	no	yes	mammals
owl	no	yes	yes	no	yes	birds
dolphin	yes	no	no	yes	no	mammals
eagle	no	yes	yes	no	yes	birds

C4.5rules versus RIPPER



C4.5rules:

(Give Birth=No, Can Fly=Yes) → Birds

(Give Birth=No, Live in Water=Yes) → Fishes

(Give Birth=Yes) → Mammals

(Give Birth=No, Can Fly=No, Live in Water=No) → Reptiles

( ) → Amphibians

Classification rules extracted from a decision tree for the vertebrate classification problem

RIPPER:

(Live in Water=Yes) → Fishes

(Have Legs=No) → Reptiles

(Give Birth=No, Can Fly=No, Live In Water=No) → Reptiles

(Can Fly=Yes, Give Birth=No) → Birds

() → Mammals

C4.5 versus C4.5rules versus RIPPER

C4.5 and C4.5rules:

			PREDICTED CLASS			
		Amphibians	Fishes	Reptiles	Birds	Mammals
ACTUAL CLASS	Amphibians	2	0	0	0	0
	Fishes	0	2	0	0	1
	Reptiles	1	0	3	0	0
	Birds	1	0	0	3	0
	Mammals	0	0	1	0	6

RIPPER:

			PREDICTED CLASS			
		Amphibians	Fishes	Reptiles	Birds	Mammals
ACTUAL CLASS	Amphibians	0	0	0	0	2
	Fishes	0	3	0	0	0
	Reptiles	0	0	3	0	1
	Birds	0	0	1	2	1
	Mammals	0	2	1	0	4

Advantages of Rule-Based Classifiers

- Karar ağaçlarına oldukça benzer özelliklere sahiptir
  - Karar ağaçları kadar ifade gücü yüksek
  - Yorumlaması kolay
  - Yeni örnekleri hızla sınıflandırabilir
  - (fakat) performansı Karar ağaçlarıyla kıyaslanabilir
  - Gereksiz özniteliklerle başa çıkabilir

- Dengesiz sınıfların üstesinden gelmek için daha uygun (imbalanced classes)

- Test setindeki eksik değerler ile başa çıkmak daha zor