

**IF4070 REPRESENTASI PENGETAHUAN & PENALARAN**  
**LAPORAN TUGAS**  
**IMPLEMENTASI RIPPLE DOWN RULE**



Disusun oleh :

13520042 – Jeremy S.O.N. Simbolon

13520092 – Vieri Mansyl

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**  
**INSTITUT TEKNOLOGI BANDUNG**  
**SEMESTER 1**  
**2023/2024**

# Penjelasan Algoritma

Algoritma *Ripple Down Rule* (RDR) merupakan salah satu metode untuk melakukan akuisisi pengetahuan dari seorang pakar. Alur berpikir dari algoritma RDR dapat dijabarkan sebagai berikut.

1. Lakukan pengecekan : apakah input dari data dapat dipenuhi dengan *node* saat ini ? Jika iya, maka label (*antecedent*) *node* saat ini ditandai sebagai **LAST TRUE**
2. Lakukan pencarian pada *child node* hingga mencapai *leaf node*. Lakukan tahap 1 pada setiap *node*
3. Pada akhir iterasi pada pohon yang terbentuk, lakukan validasi dengan pakar : apakah label dari **LAST TRUE** telah sesuai
  - a. Apabila pakar menyatakan tidak sesuai, maka akan dibentuk *node* dengan *precedent* dan *antecedent* yang sesuai dengan input data
  - b. Apabila pakar menyatakan sesuai, maka tidak terjadi perubahan pada pohon
4. Lakukan tahap 1 hingga 3 pada seluruh input data. Akhir iterasi akan menghasilkan pohon *rule*

## Dataset

Dataset yang digunakan adalah dataset klasifikasi binatang (zoo.csv). Berikut deskripsi untuk setiap fitur pada dataset.

Fitur	Tipe data	Penjelasan
animal_name	(unique) string	Nama hewan
hair	boolean	Ada-tidaknya bulu (rambut)
feathers	boolean	Ada-tidaknya bulu
eggs	boolean	Bisa-tidaknya bertelur
milk	boolean	Bisa-tidaknya menyusui
airborne	boolean	Bisa-tidaknya terbang
aquatic	boolean	Ada-tidaknya hidup di air
predator	boolean	Ada-tidaknya predator
toothed	boolean	Ada-tidaknya bergigi
backbone	boolean	Ada-tidaknya tulang belakang
breathes	boolean	Bisa-tidaknya bernafas
venomous	boolean	Ada-tidaknya bisa
fins	boolean	Ada-tidaknya sisik
legs	Numerik (0,2,4,5,6,8)	Jumlah kaki

tail	boolean	Ada-tidaknya ekor
domestic	boolean	Ada-tidaknya jinak
catsize	boolean	Ada-tidaknya berukuran besar
class_type	integer values in range [1,7]	Tipe kelas (tidak jelas)

## Implementasi

Pada pengembangan program, dilakukan pendefinisian 2 (dua) kelas, yaitu

- Node : merepresentasikan sebuah *rule*. Suatu instans node akan mencatat
  - *precedent* (premis)
  - *antecedent* (konsekuen)
  - *cornerstone* (fakta yang membentuk *rule*)
  - *except node* dan *else node* (*child node*)
  - *is\_root* (menyatakan *node* akar)

Kelas node memiliki method *getter* dan *setter* serta *match\_precedent* yang berfungsi untuk membandingkan kasus (input data) yang diterima terhadap *precedent* dari node terkait

- Tree : merepresentasikan pohon *rule*. Instans pohon hanya mencatat *root node*.

Kelas Tree memiliki method *traverse\_tree* dan *traverse\_tree\_by\_dataset* yang berfungsi untuk melakukan pembangkitan node-node sehingga membentuk pohon, serta method *start* untuk mengantisipasi input manual dari user, method *fit* untuk membangun model (pohon) berdasarkan input dataset, dan method *predict* untuk melakukan prediksi *antecedent*.

### Kasus 1 - input manual dari user

Pada kasus ini, kelas *Tree* menyediakan *prompt* UI untuk berinteraksi dengan user berupa *Command Line Interface* (CLI). User dapat memberikan entri data secara manual dan memberikan evaluasi berupa masukan (*expected antecedent*). Melalui *feedback* dari user, instans *Tree* dapat membangun node baru berdasarkan masukan dari user.

### Kasus 2 - input berupa dataset

Pada kasus ini, terlebih dahulu dilakukan *preprocessing data* pada dataset dilakukan dengan mengabaikan kolom yang densitas datanya merupakan *noise* serta mengabaikan baris pada tiap kolom yang mengandung *missing value*. Selanjutnya, dengan menggunakan fungsi *read\_dataset*, dataset awal akan ditransformasikan dan dipisah menjadi dataset bertipe **List of set of string** serta kumpulan label bertipe **list**

of **string**. Perlu ditekankan bahwa pada program implementasi RDR ini, terdapat beberapa batasan dan asumsi yang diterapkan, yaitu

- Fitur/atribut dari dataset yang dibaca memiliki tipe boolean dan tidak numerik
- Precedent yang dibentuk merupakan list of string yang merupakan kumpulan entri dengan nilai boolean TRUE. Sebagai contoh, pada dataset zoo.csv, salah satu precedent dari entri data berupa

{'backbone', 'milk', 'toothed', 'catsize', 'hair', 'breathes', 'predator'}

## Hasil

Berikut contoh input *test* pada pohon yang dibangun dengan menggunakan dataset zoo.csv.

Input : {'toothed', 'hair', 'breathes', 'milk', 'catsize', 'backbone', 'predator'}
Output: <b>gorilla</b> <pre>test = {'toothed', 'hair', 'breathes', 'milk', 'catsize', 'backbone', 'predator'} print(model.predict(test)) ✓ 0.0s gorilla</pre>
Input : {'feathers', 'eggs', 'airborne', 'backbone', 'breathes', 'tail'}
Output: <b>wren</b> <pre>test = {'feathers', 'eggs', 'airborne', 'backbone', 'breathes', 'tail'} print(model.predict(test)) ✓ 0.0s wren</pre>

## Tantangan yang dihadapi

Berikut tantangan yang dihadapi selama masa pengembangan program implementasi algoritma RDR.

- Perlunya waktu untuk mempelajari dan memahami algoritma RDR
- Pada awal pengembangan, memerlukan waktu yang lama untuk menentukan bagaimana merepresentasikan *precedent* dari sebuah *rule*, apakah berupa asumsi yang telah dijelaskan di [Kasus 2](#), atau sekaligus mencatat seluruh *rule*, baik dengan nilai boolean TRUE dan FALSE
- Pada saat pengembangan, ditemukan kesulitan dalam menggunakan dataset dengan fitur bertipe numerik, sehingga pada akhirnya, dataset dengan data

tersebut diabaikan dengan pemahaman bahwa penentuan *threshold* dari suatu fitur bertipe numerik ditentukan langsung oleh pakar

## Saran pengembangan

Berikut saran yang dapat kami berikan dalam pengembangan lanjutan terhadap program algoritma RDR kami.

- Algoritma yang dibangun dapat dikembangkan dengan memperhatikan bentuk representasi yang paling tepat untuk *precedent* dari tiap instans *rule*. Dengan begitu, proses pembangkitan *node* diharapkan lebih efisien
- Algoritma RDR yang dikembangkan memiliki kemampuan untuk *handling* data bertipe numerik dengan menentukan *threshold* berdasarkan nilai statistik, seperti *mean*, *max*, *min*, dan lainnya.

## Repositori *Github*

Berikut tautan repositori program yang mengimplementasi algoritma RDR.

<https://github.com/tastytypist/simple-rdr>

# Referensi

<https://www.kaggle.com/datasets/uciml/zoo-animal-classification?select=zoo.csv>

<https://docs.python.org/3/tutorial/classes.html>