Laporan Tugas Program 1 Mesin Pembelajaran

Ahmad Arsyel Abdul Hakim / 1301164193
IF-40-06

A. Analisis Masalah

Permasalahan dalam tugas program 1 ini adalah menentukan 40 klasifikasi data dari *testset* dengan dua label/target berbeda (>50K dan ≤50K). Diberikan 160 data *trainset* dengan tujuh atribut *input* (*age*, *workclass*, *education*, *marital-status*, *occupation*, *relationship*, dan *hours-per-week*) dan *output* (*income*) pada file *.csv* dan akan dipilih algoritma tertentu, sehingga label/target pada data *testset* dapat ditentukan.

B. Strategi Penyelesaian

Berdasarkan analisis di atas, algoritma *Naive Bayes* merupakan salah satu solusi yang dapat diterapkan pada model permasalahan klasifikasi dengan menghitung peluang sederhana di *trainset* yang telah memiliki target (*supervised learning*) untuk data *testset*. Pesebaran data dari *trainset* berupa kategorial.

Naive Bayesian Classification merupakan suatu klasifikasi berdasarkan aplikasi teorema Bayes dengan asumsi antar variabel saling bebas (independen). Dalam hal ini, diasumsikan bahwa kehadiran atau ketiadaan dari suatu kejadian tertentu dari suatu kelompok tidak berhubungan dengan kehadiran atau ketiadaan dari kejadian lainnya. Rumus probabilitas yang digunakan adalah sebagai berikut :

$$P(H|X) = \frac{P(X|H)\,P(H)}{P(X)}$$

Dengan $P(X|H)$ diubah untuk memaksimalkan nilai probabilitas.

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

Dalam mengimplementasikan algoritma ini, ada kondisi di mana hasil perhitungan dapat menghasilkan nilai nol, karena tidak ditemukannya data yang dipakai atau muncul di *trainset* sedangkan diperlukan di *testset*. Hal tersebut dapat diatasi dengan metode *Laplacian Correction*. Dengan menambahkan satu tuple fiktif pada setiap atribut yang mengakibatkan probabilitas bernilai nol.

Setelah dilakukan perhitungan algoritma *Naive Bayes*, nilai untuk P(*income* = *>50K*) = 0,75 dan P(*income* = *≤50K*) = 0,25. Implementasi program yang dibuat untuk menentukan target dari 40 *testset* yang disediakan adalah dengan bahasa pemrograman Lua dan IDE ZeroBrane Studio.

C. Penjelasan Fungsi

Pada Gambar 1, fungsi *p_high_income()* mengembalikan nilai perhitungan probabilitas untuk income yang memiliki target *>50K* di *trainset* dan diimplementasikan untuk menghitung *testset*. Tiap atribut, nilai yang dihasilkan akan dibagi dengan total kemunculan target *>50K*.

```
-- @param table Table that reference
-- @param age..hpw Value from key that wanted to calculate
local function p_high_income(table, age, wclass, edu, stat, occ, rel, hpw)
  local c1_income = counting_label(table, "income", 1)
  return (counting_based("age", age, 1)/c1_income) *
    (counting_based("workclass", wclass, 1)/c1_income) *
    (counting_based("education", edu, 1)/c1_income) *
    (counting_based("status", stat, 1)/c1_income) *
    (counting_based("occupation", occ, 1)/c1_income) *
    (counting_based("relationship", rel, 1)/c1_income) *
    (counting_based("hpw", hpw, 1)/c1_income)
end
```

Gambar 1. Fungsi menghitung nilai kemunculan label >50K.

Fungsi *counting_based()* mengembalikan jumlah kemunculan di sebuah atribut pada suatu kumpulan data (*traintest* atau *testset*) dengan target/label yang diinginkan (Gambar 2).

```
local function counting_based(tab_type, label, base_type, table)
  table = table or parse_CSV(TRAIN_PATH)
  base_type = base_type or 0
  local sum = 0
  for _, v in ipairs(table) do -- show parsing from CSV file
    if ((v[tab_type] == label) and (v.income == base_type)) then
      sum = sum + 1
    end
  end
  return sum
end
```

Gambar 2. Fungsi menghitung nilai berdasarkan atribut di target.

Hasil dari fungsi *p_high_income()* dan *p_low_income()* disimpan di array, kemudian akan dibandingkan satu sama lain di fungsi *determine_income()* sehingga mengembalikan kumpulan data di array berupa label/target dari *testset*. Array tersebut menjadi solusi yang dicari (Gambar 3).

```
local function determine_income(h_income, l_income)
  local income = 0
  if (h_income > l_income) then
    income = 1
  end
  return income
end
```

Gambar 3. Fungsi membandingkan nilai tertinggi terhadap kemunculan.