Praktikum 4
Adaline

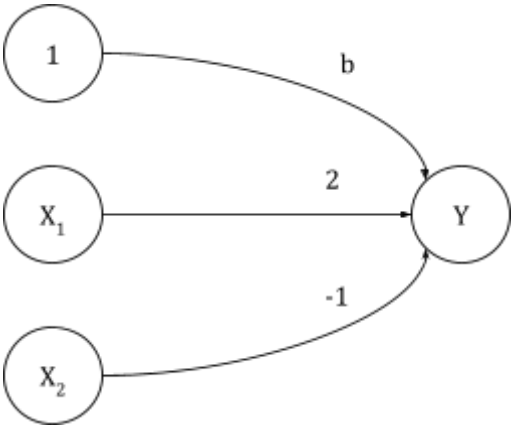
Tujuan

1. Praktikan mampu memahami algoritma Adaline
2. Praktikan mampu mengaplikasikan algoritma Adaline untuk melakukan klasifikasi

Dasar Teori

Adaline

Secara arsitektural, Adaline terdiri dari beberapa input neuron dan satu output neuron. Adaline juga menggunakan bias yang merupakan bobot dari sebuah neuron tambahan yang memiliki nilai aktivasi konstan sebesar 1. Gambar 4.1 memberikan satu contoh jaringan Adaline dengan dua input neuron.



Gambar 4.1 Adaline dengan dua *input neuron*.

Seperti pada Perceptron, nilai-nilai bobot pada Adaline dapat diubah menggunakan algoritma pelatihan Adaline. Jika beberapa jaringan Adaline dibuat dan dikombinasikan di mana *output* dari satu jaringan menjadi input untuk jaringan berikutnya, maka hasilnya adalah sebuah jaringan *multilayer*. Jaringan kombinasi Adaline seperti ini disebut dengan Madaline dan dibahas pada Bab 5.

Algoritma Pelatihan Adaline

Berikut adalah algoritma pelatihan Adaline.

1. Inisialisasi nilai-nilai bobot (termasuk *bias*) dan tentukan *learning rate* (α).
2. Untuk setiap data latih *s* dan target *t*, lakukan langkah 3–5.
3. Set nilai aktivasi setiap *input neuron*:
$$x_i = s_i$$
4. Hitung nilai *y_in* menggunakan perhitungan *weighted sum*:
$$y\_in = b + \sum_i x_i w_i$$
5. Lakukan perubahan nilai-nilai bobot:
$$b' = b + \alpha\,(t - y\_in)$$
$$w_i' = w_i + \alpha\,(t - y\_in)\,x_i$$
6. Jika perubahan nilai bobot tertinggi kurang dari suatu batas nilai tertentu, maka hentikan proses pelatihan. Jika tidak, maka lanjutkan proses pelatihan ke *epoch* berikutnya (kembali ke langkah 2)

Praktikum

1. Buka Google Colaboratory melalui tautan [tautan ini].
2. Tulis kode berikut ke setiap cell pada notebook tersebut.
   a. Fungsi *Step* Bipolar

```
def bipstep(y, th=0):
    return 1 if y >= th else -1
```

   b. Fungsi *Training* Adaline

```
import sys

def adaline_fit(x, t, alpha=.1, max_err=.1, max_epoch=-1,
verbose=False, draw=False):
    w = np.random.uniform(0, 1, len(x[0]) + 1)
    b = np.ones((len(x), 1))
    x = np.hstack((b, x))
    stop = False
    epoch = 0
```

```
    while not stop and (max_epoch == -1 or epoch < max_epoch):
        epoch += 1
        max_ch = -sys.maxsize

        if verbose:
            print('\nEpoch', epoch)

        for r, row in enumerate(x):
            y = np.dot(row, w)

            for i in range(len(row)):
                w_new = w[i] + alpha * (t[r] - y) * row[i]
                max_ch = max(abs(w[i] - w_new), max_ch)
                w[i] = w_new

            if verbose:
                print('Bobot:', w)

            if draw:
                plot(line(w), x, t)

        stop = max_ch < max_err

    return w
```

c. Fungsi *Testing* Adaline

```
def adaline_predict(X, w):
    Y = []

    for x in X:
        y_in = w[0] + np.dot(x, w[1:])
        y = bipstep(y_in)

        Y.append(y)

    return Y
```

d. Fungsi Hitung Akurasi

```
def calc_accuracy(a, b):
    s = [1 if a[i] == b[i] else 0 for i in range(len(a))]

    return sum(s) / len(a)
```

e. Logika AND

```
train = (1, 1), (1, -1), (-1, 1), (-1, -1)
target = 1, -1, -1, -1
w, epoch = adaline_fit(train, target, verbose=True, draw=True)
output = adaline_predict(train, w)
accuracy = calc_accuracy(output, target)

print('Output:', output)
print('Epoch:', epoch)
print('Target:', target)
print('Accuracy:', accuracy)
```

f. Logika OR

```
train = (1, 1), (1, -1), (-1, 1), (-1, -1)
target = 1, 1, 1, -1
w, epoch = adaline_fit(train, target, verbose=True, draw=True)
output = adaline_predict(train, w)
accuracy = calc_accuracy(output, target)

print('Output:', output)
print('Epoch:', epoch)
print('Target:', target)
print('Accuracy:', accuracy)
```

g. Logika AND NOT

```
train = (1, 1), (1, -1), (-1, 1), (-1, -1)
target = -1, 1, -1, -1
w, epoch = adaline_fit(train, target, verbose=True, draw=True)
output = adaline_predict(train, w)
accuracy = calc_accuracy(output, target)

print('Output:', output)
print('Epoch:', epoch)
print('Target:', target)
print('Accuracy:', accuracy)
```

h. Logika XOR

```
train = (1, 1), (1, -1), (-1, 1), (-1, -1)
target = -1, 1, 1, -1
w, epoch = adaline_fit(train, target, verbose=True, draw=False)
output = adaline_predict(train, w)
```

```
accuracy = calc_accuracy(output, target)

print('Output:', output)
print('Epoch:', epoch)
print('Target:', target)
print('Accuracy:', accuracy)
```

Analisis

1. Jelaskan tujuan dari parameter-parameter `alpha`, `max_err`, dan `max_epoch` yang ada pada fungsi `adaline_fit`.
2. Pada fungsi `adaline_fit`, apakah yang akan dilakukan oleh fungsi tersebut jika parameter `max_epoch` diberi nilai `-1`?
3. Amati jumlah *epoch* saat melakukan proses pelatihan menggunakan data logika AND, OR, dan AND NOT. Mengapa jumlah *epoch* pada ketiga proses pelatihan tersebut tidak sama?
4. Apakah yang terjadi saat melakukan proses pelatihan untuk data logika XOR? Mengapa bisa terjadi demikian?