BAB II
TEORI PENUNJANG

2.1 Mahasiswa

Mahasiswa adalah seseorang yang sedang dalam proses menimba ilmu ataupun belajar dan terdaftar sedang menjalani pendidikan pada salah satu bentuk perguruan tinggi, yang terdiri dari akademik, politeknik, sekolah tinggi, institut dan universitas [7].

Pengertian mahasiswa menurut KBBI adalah seseorang yang menuntut ilmu di perguruan tinggi. Di dalam dunia pendidikan, status mahasiswa adalah status tertinggi seorang murid di dunia pendidikan.

Mahasiswa itu sebagaimana dimaksud sebelumnya akan secara aktif mengembangkan potensinya dengan melakukan pembelajaran, mencarian kebenaran ilmiah, penguasaan, pengembangan, dan pengamalan dirinya di dalam suatu cabang Ilmu Pengetahuan dan Teknologi untuk menjadi seorang ilmuwan, intelektual, praktisi, dan profesional yang berbudaya [8], [9]. Mahasiswa akan memiliki kebebasan akademik dengan mengutamakan penalaran dan aklak mulia serta dapat bertanggung jawab sesuai dengan budaya akademik [8].

Mahasiswa tahun pertama adalah mahasiswa peralihan dari SMA menuju perkuliahan [10], [11]. Tuntutan akademis yang tinggi dirasakan oleh para mahasiswa tahun pertama [10], [11]. Pada tingkat pendidikan tinggi, mahasiswa dituntut untuk aktif dalam proses belajar mengajar melalui media yang ada, seperti perpustakaan, jurnal, maupun internet [10]. Keberhasilan mahasiswa dalam bidang akademik ditandai dengan prestasi akademik yang dicapai, ditunjukkan melalui indeks prestasi (IP) maupun indeks prestasi kumulatif (IPK) [10].

2.2 Indeks Prestasi Akademik

Pengertian prestasi akademik adalah hasil pelajaran yang diperoleh dari kegiatan belajar di sekolah atau perguruan tinggi yang bersifat kognitif dan

biasanya ditentukan melalui pengukuran dan penilaian [6]. Prestasi akademik pada penelitian ini dinilai berdasarkan Indeks Prestasi Kumulatif (IPK) [12]. Indeks Prestasi (IP) adalah penilaian keberhasilan studi semester yang dilakukan pada tiap akhir semester [12]. Penilaian ini meliputi semua mata kuliah yang direncanakan mahasiswa dalam Kartu Rencana Studi (KRS) [12].

2.3 Data Mining

Data mining merupakan analisis dari peninjauan kumpulan data untuk menemukan hubungan yang tidak diduga dan meringkas data dengan cara yang berbeda dengan sebelumnya, yang dapat dipahami dan bermanaat bagi pemilik data [13]. *Data mining* merupakan bidang dari beberapa bidang keilmuan yang menyatukan teknik dari pembelajaran mesin, mengenal pola, statistik, *database*, dan visualisasi untuk penanganan permasalahan pengambilan informasi dari database yang besar [13].

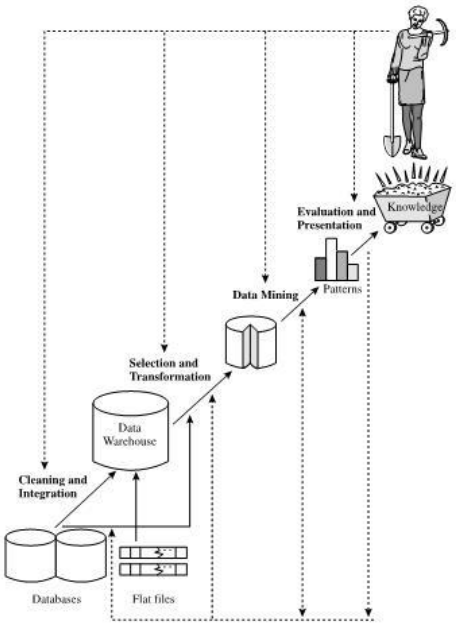
Istilah *data mining* memiliki beberapa padanan, seperti *knowledge discovery* ataupun *pattern recognition* [14], [15]. Kedua istilah tersebut sebenarnya memiliki ketepatannya masing-masing [14]. Istilah *knowledge discovery* atau penemuan pengetahuan tepat digunakan karena tujuan utama dari *data mining* memang untuk mendapatkan pengetahuan yang masih tersembunyi di dalam bongkahan data [14], [15]. Bila dalam tulisan ini digunakan istilah *data mining*, hal ini lebih didasarkan pada lebih populernya istilah tersebut dalam kegiatan penggalian pengetahuan data [14]. Namun demikian, istilah ini memiliki hakikat *(notion)* sebagai disiplin ilmu yang tujuan utamanya adalah untuk menemukan, menggali, atau menambang pengetahuan dari data atau informasi yang kita miliki [14].

Adapun tahapan-tahapan proses Knowledge Discovery in Databases dijelaskan sebagai berikut [13]–[17]:

1. *Cleaning*, yaitu membuang duplikasi data, memeriksa data yang tidak konsisten, dan memperbaiki kesalahan data seperti kesalahan dalam penulisan.
2. *Data integration*, yaitu tahapan yang perlu untuk mengurangi dan menghindari redudansi pada *dataset* sehingga dapat meningkatkan akurasi dan kecepatan saat proses *data mining*.
3. *Data selection*, yaitu proses untuk memilih data yang relevan untuk dianalisa yang diambil dari *database*.
4. *Data transformation*, yaitu proses di mana data diubah kedalam bentuk yang sesuai sehingga dapat digunakan untuk *data mining*. Untuk pola data input yang akan diskalakan ke dalam range 0 sampai 1 menggunakan rumus berikut:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \qquad (2.1)$$

   Keterangan: X' = data transformasi
   X_{min} = nilai minimum data
   X_{max} = nilai maksimum data
5. *Data mining*, yaitu proses di mana pencarian pola dalam data yang telah melewati tahap data transformation menggunakan metode dan algoritma tertentu.
6. *Pattern evaluation*, yaitu merupakan hasil dari *data mining* yang berupa pola-pola yang khusus.
7. *Knowledge*, yaitu memvisualisasikan kepada pengguna sehingga bisa digunakan sebagai acuan untuk mengambil keputusan.



Gambar 2.1. Tahapan Proses Knowledge Discovery in Databases.

2.4 Artificial Neural Network

Artificial Neural Network (ANN) adalah suatu jaringan yang memodelkan sistem syaraf otak manusia (yang disebut *neuron*) dalam melaksanakan tugas pengenalan pola, khususnya klasifikasi [17]. Pemodelan ini didasari oleh kemampuan otak manusia dalam mengorganisir *neuron* sehingga mampu mengenali pola secara efektif [17].

Salah satu kelebihan ANN adalah mampu menggambarkan model baik linier maupun non linier dengan rentang yang cukup luas [18]. ANN juga memiliki kemampuan untuk mendeteksi interaksi yang mungkin antara variabel prediksi [18]. ANN mampu untuk mendeteksi secara komplit tanpa keraguraguan meskipun dalam hubungan non linear antara variabel bebas dan terikat [18]. ANN memiliki kemampuan yang potensial untuk memprediksi secara akurat dan klasifikasi prestasi akademik mahasiswa di institusi yang lebih tinggi [19], [20].

Jaringan syaraf tiruan menggunakan elemen perhitungan *non linier* dasar yang disebut *neuron* yang diorganisasikan sebagai jaringan yang saling

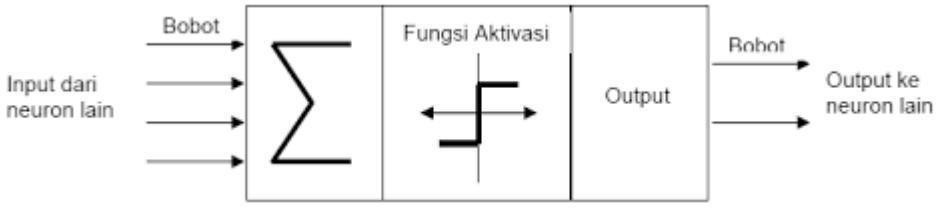
berhubungan, sehingga mirip dengan jaringan syaraf manusia. Hubungan antar neuron dikenal dengan nama bobot. *Neuron* merupakan unit pemroses informasi yang menjadi dasar dalam pengoperasian jaringan syaraf tiruan [20]. Neuron terdiri dari tiga elemen pembentuk, yaitu [20]:

1. Himpunan unit-unit yang dihubungkan dengan jalur koneksi.
2. Suatu unit penjumlah yang akan menjumlahkan sinyal-sinyal input yang sudah dikalikan dengan bobotnya.
3. Fungsi aktifasi yang menentukan apakah sinyal dari input neuron akan diteruskan ke neuron lain atau tidak.

Sebagaimana otak manusia yang terdiri banyak lapisan, suatu ANN adalah jaringan *neuron* yang diorganisasikan dalam bentuk lapisan-lapisan [20]. Secara umum, arsitektur jaringan dibagi ke dalam tiga macam, yaitu : *Single-Layer Feedforward Network, Multi Layer Perceptron,* dan *Recurrent Network* [17]. Pada penelitian ini arsitektur jaringan yang akan digunakan adalah *Multi Layer Perceptron*.

2.4.1 Komponen-komponen

Struktur neuron pada Artificial Neural Network (ANN) atau Jaringan Syaraf Tiruan terdiri dari beberapa neuron dan terdapat penghubung antara neuron-neuron yang dikenal dengan bobot seperti halnya otak manusia [21].



Gambar 2.2. Struktur Neuron Jaringan Syaraf Tiruan.

Berdasarkan gambar, informasi (disebut dengan : *input*) akan dikirim ke *neuron* dengan bobot kedatangan tertentu [21]. *Input* akan diproses oleh suatu fungsi perambatan yang menjumlahkan nilai-nilai semua bobot yang datang [21]. Hasil penjumlahan ini kemudian akan dibandingkan dengan suatu nilai ambang

(*treshold*) tertentu melalui fungsi aktivasi setiap *neuron* [21]. Apabila *input* tersebut melewati suatu nilai ambang tertentu, maka *neuron* tersebut akan diaktifkan, tapi jika tidak, maka *neuron* tersebut tidak akan diaktifkan [21]. Apabila *neuron* tersebut diaktifkan, maka *neuron* tersebut akan mengirimkan *output* melalui bobot-bobot output-nya ke semua *neuron* yang berhubungan dengannya demikian seterusnya [21].

2.4.2 Fungsi Aktivasi

Fungsi aktivasi memiliki peranan sangat penting dalam suatu jaringan syaraf tiruan di mana penggunaannya tergantung sesuai kebutuhan dan target yang diinginkan serta fungsi aktivasi yang akan menentukan besarnya bobot [21].

Ada beberapa fungsi aktivasi yang sering digunakan dalam jaringan syaraf tiruan, diantaranya [21]:

1. Fungsi *Sigmoid Biner* (*Logistic*)

   Fungsi ini sering diterapkan pada *neural network* ketika algoritma pembelajarannya menggunakan metode *backpropagation*. Fungsi ini memiliki kisaran antara 0 sampai 1 sehingga baik digunakan pada jaringan yang memiliki nilai keluaran kisaran antara 0 sampai 1. Berikut rumus fungsi *sigmoid biner* secara sistematis.

$$y = f(x) = \frac{1}{1+e^{-\alpha x}} \qquad (2.2)$$

$$f'(x) = \alpha f(x)[1 - f(x)] \qquad (2.3)$$

   Keterangan: *f(x)* = Fungsi aktivasi
   x = Jumlah sinyal-sinyal *input* yang terboboti
   α = Laju pembelajaran (*learning rate*)

2. Fungsi *Sigmoid Bipolar*

   Fungsi ini sama halnya dengan fungsi *sigmoid biner*, yang memiliki keluaran dengan kisaran antara -1 sampai 1. Berikut rumus fungsi *sigmoid bipolar*.

$$y = f(x) = \frac{1-e^{-x}}{1+e^{-\alpha x}} \quad (2.4)$$

$$f'(x) = \frac{\alpha}{2}[1 + f(x)][1 - f(x)] \quad (2.5)$$

3. Fungsi *Linear*

   Fungsi ini memiliki nilai keluaran yang sama dengan nilai masukkannya. Berikut rumus fungsi *linear*.

$$y = x \quad (2.6)$$

   Keterangan: y = *Output* atau keluaran

   x = Jumlah sinyal-sinyal *input* yang terboboti

2.4.3 Multi Layer Perceptron

Multi Layer Perceptron (MLP) adalah arsitektur jaringan yang memiliki banyak lapisan [17]. MLP memiliki satu atau lebih lapisan tersembunyi *(hidden layer)*, dengan *computation nodes* yang berhubungan disebut *hidden neurons* atau *hidden units* [17].

Multilayer perceptron secara literal memiliki beberapa *layers*, secara umum ada tiga *layers*: *input*, *hidden*, dan *output layer* [22]. *Input layer* menerima *input* (tanpa melakukan operasi apapun), kemudian nilai *input* (tanpa dilewatkan ke fungsi aktivasi) diberikan ke *hidden units* [22]. Pada *hidden units*, *input* diproses dan dilakukan perhitungan hasil fungsi aktivasi untuk tiap-tiap *neuron*, lalu hasilnya diberikan ke *layer* berikutnya [22]. *Output* dari *input layer* akan diterima sebagai *input* bagi *hidden layer* [22]. Begitupula seterusnya *hidden layer* akan mengirimkan hasilnya untuk *output layer* [22]. Arsitektur ini juga biasa disebut sebagai *multilayer feedforward neural network*.

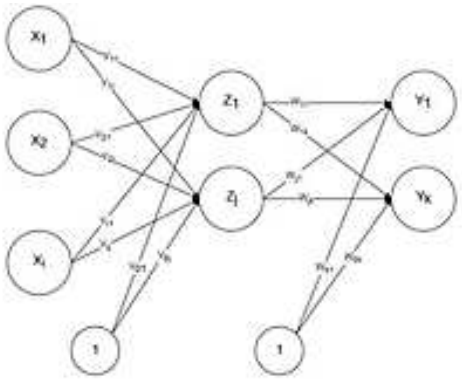
Pada dasarnya, Proses belajar MLP bertujuan menemukan bobot sinaptik yang paling tepat untuk mengklasifikasikan pola-pola himpunan data latih. Banyak algoritma pelatihan untuk melatih MLP, salah satu yang paling populer adalah *Back Propagation* (Propagasi Balik).

2.4.4 Backpropagation

Algoritma pelatihan *Backpropagation* adalah salah satu pelatihan jaringan saraf tiruan yang menggunakan model pembelajaran terawasi atau disebut juga *supervised learning*, di mana output yang diharap sudah diketahui sebelumnya [19], [22], [23]. Algoritma *backpropagation* memungkinkan untuk memperbarui skema bobot dengan nilai yang sangat kecil dalam jaringan yang kompleks [21]. Algoritma *backpropagation* menggunakan *error output* untuk mengubah nilai bobot-bobotnya dalam perambatan mundur (*backward*) [21]. Untuk mendapatkan *error* ini, tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu [21].

Pada jaringan syaraf tiruan *backpropagation* pembelajaran bersifat *iterative* dan didesain untuk meminimalkan *mean square error (MSE)* antara *output* yang dihasilkan dengan *output* yang diinginkan (target) [23]. Jaringan syaraf tiruan dengan pembelajaran *backpropagation* merupakan metode yang efisien untuk mempelajari hubungan antara *input* variabel dan *output* variable .[23]

Tiga tahapan dalam proses pelatihan *backpropagation* yaitu *input layer*, *hidden layer* dan *output layer* [24]. V_{ij} merupakan bobot dari *input* X_i ke unit *hidden layer* Z_j [24]. V_{0j} merupakan bobot yang menghubungkan bias dari *input* ke *hidden layer* Z_j [24]. Sementara bobot dari unit *hidden layer* Z_j ke unit *output* Y_k dilambangkan dengan W_{jk} (W_{0k} adalah bobot yang menghubungkan bias dari *hidden layer* Z_j ke *output layer* Y_k) [24].



Gambar 2.3. Arsitektur MLP dengan Algoritma Backpropagation.

2.4.5 Algoritma Pelatihan Backpropagation

Berikut adalah beberapa tahapan-tahapan dalam proses pelatihan *Backpropagation* diantaranya [21]:

1. Inisialisasi bobot dengan bilangan acak kecil.
2. Jika kondisi penghentian belum terpenuhi, kerjakan langkah 3-9.
3. Pada setiap pasangan data pelatihan, lakukan langkah 4-9.

Fase 1 ***Feedforward***

4. Tiap unit *input* (X_i , i=1,2,3, …, n) menerima sinyal X_i dan meneruskan sinyal tersebut ke semua unit pada lapisan yang ada di atasnya (lapisan tersembunyi *hidden layer*).
5. Masing-masing unit *hidden layer* dikalikan dengan faktor penim-bang dan dijumlahkan serta ditambah dengan biasnya. Seperti persamaan berikut:

$$Z\_in_j = V_{0j} + \sum_{i=1}^{n} X_i V_{ij} \qquad (2.7)$$

Keterangan:

Z_in_j = Keluaran untuk unit Z_j

V_{0j} = Nilai penimbang sambungan pada bias untuk unit Z_j

X_i = Nilai aktivasi dari unit X_i

V_{ij} = Nilai penimbang sambungan dari unit X_i ke unit Z_j

Gunakan fungsi aktivasi untuk menghitung sinyal *output*, seperti pada persamaan berikut:

$$Z_j = f(Z\_in_j) \qquad (2.8)$$

Keterangan:

Z_j = Nilai aktivasi dari unit Z_j

Bila yang digunakan adalah fungsi *sigmoid* maka bentuk fungsi tersebut sebagai berikut:

$$Z_j = \frac{1}{1+exp^{(-Z\_in_j)}} \qquad (2.9)$$

Kemudian mengirim sinyal tersebut ke semua unit pada *output layer*.

6. Tiap-tiap *unit* di *output layer* (Y_k, k = 1,2,3, … , m) menjumlahkan sinyal-sinyal masuk yang berbobot menggunakan persamaan (2.10).

$$Y\_in_k = W_{0k} + \sum_{j=1}^{p} Z_j W_{kj} \quad (2.10)$$

Keterangan:

Y_in_k = Total isinya masuk pada keluaran unit ke-k

W_{0k} = Nilai penimbang sambungan pada bias untuk unit Y_k

Z_j = Unit ke-j pada lapisan tersembunyi

W_{kj} = Nilai penimbang sambungan dari unit Y_k ke unit Z_j

Gunakan fungsi aktivasi untuk menghitung sinyal *output*, seperti pada persamaan berikut:

$$Y_k = \frac{1}{1+e^{-Y\_ink}} \quad (2.11)$$

Keterangan:

Y_k = Keluaran pada unit ke-k

Fase 2 *Backpropagation*

7. Tiap *unit* di *output layer* (Y_k, k = 1,2,3, … , m) menerima pola target berkaitan dengan pola pelatihan masuknya. Hitung galat informasi menggunakan persamaan berikut ini:

$$\delta_k = (t_k - Y_k) f'(Y\_in_k) = (t_k - Y_k) Y_k (1 - Y_k) \quad (2.12)$$

Keterangan:

δ_k = Faktor pengaturan nilai penimbang sambungan pada lapisan keluaran

t_k = Target yang harus dicapai

Kemudian hitung koreksi bobot (digunakan untuk memperbaiki W_{jk}) persamaan yang digunakan sebagai berikut:

$$\Delta W_{kj} = \alpha \delta_k Z_j \quad (2.13)$$

Keterangan:

ΔW_{kj} = Selisih antara $W_{kj}(t)$ dengan $W_{kj}(t + 1)$

δ_k = Faktor pengaturan nilai penimbang sambungan pada lapisan keluaran

α = Konstanta laju pelatihan (learning rate)

Hitung juga nilai koreksi bias (digunakan untuk memperbaiki W_{0k}) dengan menggunakan persamaan berikut ini:

$$\Delta W_{0k} = \alpha \delta_k \quad (2.14)$$

Keterangan:

ΔW_{0k} = Selisih antara $W_{0k}(t)$ dengan $W_{0k}(t+1)$

8. Tiap unit di *hidden layer* (Z_j = menjumlahkan delta masukannya (dari unit-unit pada lapisan di atasnya)) dengan menggunakan persamaan sebagai berikut:

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k W_{jk} \quad (2.15)$$

Keterangan:

δ_in_j = Jumlah kesalahan dari unit tersembunyi

W_{jk} = Nilai penimbang sambungan dari Z_j ke unit Y_k

Kalikan nilai ini dengan turunan fungsi aktivasinya untuk menghitung informasi *error* dengan persamaan berikut:

$$\delta_j = \delta\_in_j f'(Z\_in_j) = \delta\_in_j Z_j(1 - Z_j) \quad (2.16)$$

Keterangan:

δ_j = Faktor pengaturan nilai penimbang sambungan pada lapisan keluaran

δ_in_j = Jumlah kesalahan dari unit tersembunyi

Kemudian hitung koreksi bobot (digunakan untuk memperbaiki V_{ij}) menggunakan persamaan berikut ini:

$$\Delta V_{ij} = \alpha \delta_j X_i \quad (2.17)$$

Keterangan:

ΔV_{ij} = Selisih antara $V_{ij}(t)$ dengan $V_{ij}(t+1)$

δ_j = Faktor pengaturan nilai penimbang sambungan pada lapisan tersembunyi

X_i = Nilai aktivasi dari unit X_i

Hitung juga nilai koreksi bias (digunakan untuk memperbaiki V_{0j}) dengan menggunakan persamaan berikut ini:

$$\Delta V_{0j} = \alpha \delta_j \quad (2.18)$$

Keterangan:

ΔV_{0j} = Selisih antara $V_{0j}(t)$ dengan $V_{0j}(t + 1)$

9. Tiap unit di output layer (Y_k, k = 1,2,3, … , m) melakukan bias dan bobotnya (j = 0,1,2,3, … , p) dengan menggunakan persamaan sebagai berikut:

$$W_{jk}(baru) = W_{jk}(lama) + \Delta W_{jk} \quad (2.19)$$

Keterangan:

W_{jk} = Nilai penimbang sambungan dari Z_j ke unit Y_k

ΔW_{jk} = Selisih antara $W_{jk}(t)$ dengan $W_{jk}(t + 1)$

Setiap unit di *hidden layer* (Z_j, j = 1,2,3, … , p) melakukan perubahan bobot dan bias yang berasal dari tiap unit di *input layer* (X_i, i = 1,2,3, … , n) menggunakan persamaan sebagai berikut:

$$V_{ij}(baru) = V_{ij}(lama) + \Delta V_{ij} \quad (2.20)$$

Keterangan:

V_{ij} = Nilai penimbang sambungan dari Z_j ke unit Y_k

ΔV_{ij} = Selisih antara $V_{ij}(t)$ dengan $V_{ij}(t + 1)$

Guna mendapatkan karakteristik *backpropagation* yang terbaik sehingga *backpropagation* tersebut mampu mempelajari pola yang diberikan dengan benar maka pelatihan pola dilakukan secara berulang-ulang menggunakan data pelatihan dan parameter yang telah ditentukan [25].

2.4.6 Prediksi Metode ANN

Prediksi menggunakan metode ANN dengan algoritma *backpropagation* mendapatkan model jaringan terbaik setelah mempresentasikan hasil dari keakuratan jaringan tersebut [26]. Dikatakan kurang akurat ketika perbedaan antara hasil prediksi dengan data target (historis) sangat besar. Perbedaan kedua hal

tersebut yang sering dikenal dengan galat (*error*) [26]. Semakin besar *error* yang dihasilkan maka semakin tidak akurat hasil dari prediksi yang didapatkan [26].

Terdapat beberapa ukuran kinerja prediksi yang digunakan yaitu MSE (*Mean Square Error*) dan MAPE (*Mean Absolute Percentage Error*) [26]. Berikut ukuran kinerja prediksi yang dihasilkan.

1. *Mean Squared Error* (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(T_i - Y_i)^2 \qquad (2.21)$$

2. *Mean Absolute Percentage Error* (MAPE)

$$MAPE = \left(\frac{1}{n}\sum_{i=1}^{n}\frac{|T_i - Y_i|}{T_i}\right) * 100\% \qquad (2.22)$$

Keterangan:

T_i = Target data ke-i

n = Banyaknya data observasi

Y_i = Prediksi data ke-i

2.5 Python

Python adalah bahasa pemrograman yang dapat digunakan untuk banyak fungsi [27], [28]. Bahasa ini diciptakan oleh seorang *computer scientist* bernama Guido van Rossum pada tahun 1991 [29]. Tujuan dikembangkannya *Python* adalah agar bahasa ini mudah dibaca dan mampu mencakup berbagai paradigma pemrograman [27], [28].

Python adalah bahasa yang berakar pada ilmu komputer dan matematika [27], [28]. Bahasa ini didesain untuk *programer* yang memerlukan fleksibilitas tinggi dalam berbagai aktivitas pemrograman [27], [28].

Salah satu kekuatan terbesar *Python* adalah fleksibilitas penggunaannya yang termasuk *web framework, database connectivity, networking, web scraping, scientific computing, text* dan *image processing*. *Python* memiliki berbagai *library*

data science menarik, termasuk Scikit-Learn, Numpy, Scipy, dan Matplotlib yang merupakan *library data science* paling populer.

Berikut adalah contoh pseudocode algoritma *backpropagation* dalam bahasa pemograman *python* dengan *library numpy* :

```
def output_hidden(self, target_output, y, z, alpha, w):
    baris, kolom = y.shape
    tarOut = np.zeros((baris, kolom))
    for i in range(baris):
        for j in range(kolom):
            tarOut[i, j] = (target_output-y[i, j])*y[i, j]*(1-y[i, j])
    baris, kolom = tarOut.shape
    baris1, kolom1 = z.shape
    deltaW = np.zeros((kolom1+1, kolom))
    for i in range(kolom):
        for j in range(kolom1):
            deltaW[j+1, i] = round(alpha*tarOut[0, i]*z[i, j], 6)
        deltaW[0, i] = round(alpha*tarOut[0, i], 6)
    w_baru = w + deltaW
    return w_baru


def hidden_input(self, target_output, y, data, alpha, z, w, v):
    baris, kolom = y.shape
    tarOut = np.zeros((baris, kolom))
    for i in range(baris):
        for j in range(kolom):
            tarOut[i, j] = (target_output-y[i, j])*y[i, j]*(1-y[i, j])
    baris1, kolom1 = w.shape
```

```
    baris2, kolom2 = z.shape
    tarOutW = np.zeros((baris2, kolom2))
    for i in range(kolom2):
        tmp = 0
        for j in range(kolom):
            tmp = round(tmp+tarOut[0, j]*w[i+1, j], 6)
        tarOutW[0, i] = round(tmp*z[0, i]*(1-z[0, i]), 6)
    baris, kolom = tarOutW.shape
    n_data = data.shape[0]
    m, n = v.shape
    deltaV = np.zeros((m, n))
    for j in range(kolom):
        for i in range(n_data):
            deltaV[i+1, j] = round(alpha*tarOutW[0, j]*data[i], 6)
        deltaV[0, j] = round(alpha*tarOutW[0, j], 6)
    v_baru = np.round((v + deltaV), 6)
    return v_baru
```

2.6 MySQL

MySQL adalah sebuah *server database open source* yang termasuk populer keberadaannya [28]. *MySQL* umumnya digunakan bersamaan dengan *PHP* untuk membuat aplikasi *server* yang dinamis dan powerfull [28]. *MySQL AB* membuat *MySQL* tersedia sebagai perangkat lunak gratis di bawah lisensi *GNU General Public License (GPL),* tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus di mana penggunaannya tidak cocok dengan penggunaan *GPL* [28]. *MySQL* adalah *Relational Database Management System (RDBMS)* yang didistribusikan secara gratis di bawah lisensi *GPL (General Public License)* [28].

MySQL sebenarnya merupakan turunan salah satu konsep utama dalam database sejak lama, yaitu *SQL (Structured Query Language)* [30]. *SQL* adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis [30].

2.7 Microsoft Excel

Microsoft Excel adalah sebuah program aplikasi lembar kerja *spreadsheet* yang dibuat dan didistribusikan oleh *Microsoft Corporation* yang dapat dijalankan pada *Microsoft Windows* dan *Mac OS* [31]. Aplikasi ini memiliki fitur kalkulasi dan pembuatan grafik yang, dengan menggunakan strategi marketing *Microsoft* yang agresif, menjadikan *Microsoft Excel* sebagai salah satu program komputer yang populer digunakan di dalam komputer mikro hingga saat ini [31]. Aplikasi ini merupakan bagian dari *Microsoft Office System* [31].

2.8 Unified Modelling Language (UML)

UML (Unified Modelling Language) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis & desain, serta menggambarkan arsitektur dalam pemograman berorientasi objek [32]. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sistem dengan menggunakan diagram dan teks-teks pendukung [32].

UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan bahasa pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk baku [33].

2.8.1 Class Diagram

Class Diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem [32]. Kelas memiliki apa yang disebut atribut dan metode atau operasi [32]. *Class* adalah deskripsi

sekumpulan *object* yang memiliki kesamaan atribut, operasi, *relationship* dan *semantics* [33]. Dengan kata lain, sebuah *class* merupakan *blueprint*/ template/ cetakan dari satu atau lebih *object* [32], [33]. Diagram kelas dibuat agar pembuat program membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron [32].

2.8.2 Use Case Diagram

Use Case diagram merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat [32]. *Use Case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat [32], [33]. *Use Case* adalah layanan *(services)* atau fungsi-fungsi yang disediakan oleh sistem untuk pengguna-penggunanya [33]. *Use Case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu [25]. Setiap *Use Case* adalah suatu urut-urutan *(sequence)* transaksi yang saling berhubungan dan dilakukan oleh sebuah aktor dan sistem dalam bentuk sebuah dialog [33]. *Use Case Diagram* dibuat untuk memvisualisasikan/ menggambarkan hubungan antara aktor dan *Use Case* [33].

2.8.3 Activity Diagram

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau urutan aktifitas dalam sebuah proses [32], [33]. Diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor jadi aktivitas yang dapat dilakukan oleh sistem [32].

2.8.4 Sequence Diagram

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek [32], [33]. Menggambar diagram sekuen harus diketahui objek-objek yang terlibat dalam *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu [32]. *Sequence Diagram* digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai sebuah respon dari suatu kejadian untuk menghasilkan *output* tertentu [33].

2.9 Penelitian Sebelumnya

Adapun penelitian sebelumnya yang digunakan untuk dapat dijadikan bahan pertimbangan dan diharapkan dapat membantu dalam pembuatan sistem yang baru. Pada penelitian Jejen Jaenudin dengan judul *"Perbandingan Decision Tree Pada Algoritma C4.5 dan Id3 dalam Pengklasifikasian Indeks Prestasi Mahasiswa"* mengaplikasikan teknik *data mining* dengan algoritma C4.5 dan algoritma *iterative dichotomiser 3* (ID3) dalam membuat model aturan *decision tree* untuk mendukung peningkatan indeks prestasi mahasiswa pada Fasilkom UNSIKA. Model aturan tersebut diperoleh dari pengklasifikasian indeks prestasi mahasiswa berdasarkan predikat puas dan tidak puas. Hasil yang didapatkan pada penelitian ini algoritma C4.5 memiliki tingkat akurasi yang paling tinggi dibandingkan dengan algortima ID3, sehingga baik digunakan untuk pengklasifikasian indeks prestasi mahasiswa dengan persentase 96.34%.

Penelitian oleh Novhirtamely Kahar dengan judul *"Analisis Penerapan Artificial Neural Network dalam Penentuan Bidang Kompetensi Skripsi Mahasiswa"* mengaplikasikan teknik data mining dengan membandingkan algoritma *Neural Network Backpropagation* dengan *Perceptron*. Pada penelitian ini data yang dilatih sebanyak 100 data sedangkan proses pengujian menggunakan 80 data dan proses prediksi menggunakan 130 data. Hasil yang didapatkan pada penelitian ini kinerja model *Backpropagation* dalam penentuan bidang kompetensi skripsi mahasiswa lebih baik karena mampu memprediksi data dengan ketepatan 94,615%, sedangkan penerapan model *Perceptron* hanya mampu memprediksi data dengan ketepatan 83,077%.

Penelitian oleh Ade Pujianto dengan judul *"Perancangan Sistem Pendukung Keputusan untuk Prediksi Penerima Beasiswa Menggunakan Metode Neural Network Backpropagation"* mengaplikasikan teknik data mining dengan algoritma *Backpropagation* bertujuan dapat membantu pihak kemahasiswaan dalam mencari solusi dalam menentukan penerimaan beasiswa di Universitas AMIKOM Yogyakarta. Hasil yang didapatkan pada penelitian ini algoritma *Neural*

Network Backpropagation memiliki akurasi rata-rata tertinggi sebesar 99% serta hasil rata-rata nilai *error* terendah 0,000101.

Penelitian oleh Astrid Darmawan dengan judul *"Pembuatan Aplikasi Data Mining Untuk Memprediksi Masa Studi Mahasiswa Menggunakan Algoritma K-Nearest Neighborhood"* mengaplikasikan teknik data mining dengan algoritma *K-Nearest Neighborhood* bertujuan memanfaatkan data akademik mahasiswa yaitu Indeks Prestasi (IP) untuk memprediksi masa studi mahasiswa di Universitas Komputer Indonesia. Hasil yang didapatkan pada penelitian ini nilai *k* yang terbaik untuk memprediksi masa studi mahasiswa adalah nilai $k = 20$ dan $k = 30$ dengan tingkat keberhasilan 85%.

Penelitian oleh Gedeon dan Turner dengan judul "*Explaining Student Grades Predicted by A Neural Network*" mengaplikasikan *Neural Network* dengan algoritma *Feedforward* untuk menjelaskan prediksi nilai akhir mahasiswa. Hasil yang didapatkan pada penelitian ini algoritma *Neural Network Feedforward* memiliki akurasi ketepatan 94%.

Berdasarkan penelitian-penelitian sebelumnya yang telah disebutkan, penulis akan menggunakan algoritma *Neural Network Backpropagation* dalam memprediksi indeks prestasi mahasiswa karena memiliki tingkat akurasi yang baik dan mampu mengenali pola data baru yang lebih banyak. Selain itu algoritma *Neural Network Backpropagation* memiliki *fault tolerance*, gangguan dapat dianggap sebagai *noise* saja.