14

Penerapan Pembelajaran Mesin

> "Leading is not the same as being the leader. Being the leader means you hold the highest rank, either by earning it, good fortune or navigating internal politics. Leading, however, means that others willingly follow you–not because they have to, not because they are paid to, but because they want to."
>
> Simon Sinek

Bab ini memuat contoh penggunaan *machine learning* untuk dua permasalahan praktis yaitu: (1) sistem rekomendasi dan (2) sistem peringkasan dokumen. Dua domain ini dipilih karena tidak asing (*familiar*) bagi penulis. Seperti yang sudah dideskripsikan pada bab-bab sebelumnya, penerapan *machine learning* pada suatu domain membutuhkan pengetahuan/keahlian pada domain tersebut. Bab ini tidak akan membahas domain secara detail, tetapi secara abstrak (bertujuan memberikan gambaran/pengenalan). Untuk mengerti domain yang dibahas secara mendalam, silakan membaca sumber lainnya. Bab ini akan memuat secara sangat singkat, apa guna *machine learning* dan pada contoh kasus seperti apa teknik *machine learning* diterapkan pada permasalahan spesifik domain. Tujuan bab ini adalah untuk memberikan gambaran, bahwa mengetahui *machine learning* saja mungkin tidak cukup. Sekali lagi penulis ingin menekankan, pembaca harus mengerti domain aplikasi. Selain itu, bab ini mengilustrasikan bahwa model *machine learning* tidak berdiri sendiri. Artinya, model *machine learning* bisa jadi hanyalah sebuah modul (untuk mengeksekusi fungsi tertentu) pada sebuah perangkat lunak.

14.1 Sistem Rekomendasi

Bagian sistem rekomendasi ditulis oleh **Candy Olivia Mawalim** (*Japan Advanced Institute of Science and Technology*). Penulis ingin berterima kasih atas sumbangsih yang diberikan.

Salah satu penerapan pembelajaran mesin adalah sistem rekomendasi. Sistem rekomendasi dimotivasi oleh keinginan pemilik usaha untuk meningkatkan penjualan dengan cara mengerti pola pembelian pengguna. Aplikasi yang memanfaatkan sistem rekomendasi dapat kita temukan dalam kehidupan sehari-hari, misalnya Youtube yang memberikan rekomendasi video berdasarkan riwayat video yang telah kita lihat sebelumnya dan Amazon yang merekomendasikan produknya dengan cara menawarkan produk yang sering dibeli pengguna lain yang memiliki karakteristik yang "mirip" dengan kita. Ada dua komponen penting pada sistem rekomendasi yaitu: pengguna dan *item*. Pengguna adalah sang pengguna sistem, sementara *item* dapat berupa video, buku, dan lain sebagainya (produk/layanan yang ditawarkan sistem).

Secara umum, terdapat dua teknik untuk membangun sistem rekomendasi yaitu: (1) *content-based filtering* dan (2) *collaborative filtering*. Teknik pertama berfokus pada karakteristik pengguna secara spesifik. Teknik kedua berfokus pada selera terhadap suatu *item*. Dalam sistem rekomendasi, teknik *machine learning* dapat digunakan untuk memprediksi *item* yang mungkin disukai pengguna. Dua subbab berikutnya memuat aplikasi teknik *machine learning* untuk kedua teknik sistem rekomendasi(14.1.1 dan 14.1.2). Setelah proses *filtering* (memprediksi *item* yang mungkin disukai pengguna), biasanya ada tahapan *post-processing*. Misalnya menampilkan *item* yang bersesuaian dengan usia pengguna. Contoh lain adalah *post-processing* untuk memberikan rekomendasi yang cukup beranekaragam [100].

14.1.1 Content-based Filtering

Teknik membangun sistem rekomendasi berdasarkan *content-based filtering* memanfaatkan informasi mengenai profil seorang pengguna beserta uraian item yang sangat menarik bagi pengguna tersebut [101]. Profil pengguna diartikan sebagai karakteristik (atribut dan *behavior*) yang dimiliki pengguna. Atribut pengguna misalnya *gender*, kewarganegaraan, umur, dan lain-lain. Informasi mengenai *behavior* mencakup karakteristik *item* yang seorang pengguna sukai. Misalkan *item* adalah film, karakteristik film dapat ditentukan dari aktor yang terlibat dalam film, pembuat film, tahun pembuatan film, dan *genre* dari film (misalnya *action*, *horror*, *comedy*). Dengan karateristik ini, kita dapat menentukan kemiripan antar-film.

Kedua informasi ini dapat direpresentasikan sebagai vektor (ekuivalen dengan *feature vector*) agar mudah untuk dilakukan operasi aritmatika, dikenal dengan istilah ***user embedding*** dan ***item embedding***. Cara melakukan *embedding* untuk profil pengguna dan uraian item mirip dengan cara *word embedding* yang telah dijelaskan pada subbab 12.5.3, *one-hot encoding*.

Rekomendasi *item* yang diberikan pada pengguna adalah *item* yang paling mungkin disukai pengguna berdasarkan karakteristiknya. Agar mendapat gambaran lebih jelas, penulis mengajak pembaca untuk mengikuti tutorial pembuatan sistem rekomendasi film sangat sederhana menggunakan dataset MovieLens[1] berukuran kecil (100K). Dataset ini memuat berbagai informasi sebagai berikut:

1. Informasi *rating* pengguna untuk masing-masing film.
2. Informasi film, misalkan berupa *genre* dan tanggal *release*.
3. Demografik pengguna, misal usia, *gender*, pekerjaan, dan lain lain.

Untuk menyederhanakan tutorial, kami hanya akan menjelaskan contoh penggunaan *genre* film untuk membuat sistem rekomendasi. Pertama-tama, kita bangun representasi *item embedding*, yaitu *genre* untuk masing-masing film (Tabel 14.1). Setiap baris pada tabel tersebut merepresentasikan *item embedding* untuk suatu film. Perhatikan! Setiap sel pada tabel diisi nilai "1" apabila film memiliki *genre* yang tertera pada kolom bersesuaian, "0" apabila tidak.

MovieId	Adventure	Animation	Children	Comedy
6	1	1	1	1
22	1	0	1	0
50	0	0	0	1

Tabel 14.1: Representasi *item embedding* untuk film berdasarkan *genre*.

Kedua, kita bangun representasi *user embedding*, yaitu apakah pengguna menyukai suatu film atau tidak (*binary*). Suka atau tidaknya pengguna terhadap suatu film dapat dilihat berdasarkan *rating* yang ia berikan. Sebagai contoh sederhana, kita anggap apabila pengguna menyukai suatu film apabila memberi nilai *rating* lebih dari atau sama dengan 4. Sebagai contoh, perhatikan Tabel 14.2. Apabila user menyukai suatu film, nilai "1" diisi pada kolom "Like or Not", dan "0" apabila sebaliknya.

UserId	MovieId	Like or Not
1	6	0
1	22	0
1	50	1

Tabel 14.2: Representasi *user embedding* berdasarkan *rating* yang diberikan pengguna.

[1] https://grouplens.org/datasets/movielens/

Berdasarkan *item embedding* dan *user embedding* yang kita punya, kita ganti kolom MovieId pada Tabel 14.2 menggunakan baris pada *item embedding*. Sekarang, kita memiliki dataset *behavior* pengguna dengan UserId=1 (Tabel 14.3). Perhatikan! Tabel tersebut seperti dataset *machine learning* yang sudah kamu pelajari pada bab-bab sebelumnya. Diberikan *feature vector* dan kelas (Like or Not) yang berkorespondensi. Menggunakan data seperti

MovieId	Adventure	Animation	Children	Comedy	Like or Not
6	1	1	1	1	0
22	1	0	1	0	0
50	0	0	0	1	1

Tabel 14.3: Dataset *behavior* pengguna dengan UserId=1.

pada Tabel 14.3 yang dirata-ratakan, kita dapat menggunakan teknik *machine learning* untuk memprediksi apakah suatu pengguna akan menyukai film tertentu, berdasarkan *genre* yang dimuat oleh film tersebut. Sederhananya, bisa menggunakan K-*nearest-neighbor* dengan menghitung *cosine similarity* antara *item embedding* dan *user embedding*.

Teknik *content-based filtering* memiliki keunggulan dimana kita tidak memerlukan banyak informasi tentang pengguna lain. Kita hanya memerlukan informasi uraian *item* dan informasi karakteristik suatu pengguna. Hal ini mengakibatkan rekomendasi yang diberikan sangat bergantung pada kepribadian pengguna. Apabila pengguna tidak konsisten, sistem rekomendasi juga bingung.

14.1.2 Collaborative Filtering

Teknik ini diperkenalkan oleh Paul Resnick dan Hal Varian pada 1997 [102]. Prinsip *collaborative filtering* adalah asumsi bahwa selera penggunaan terhadap suatu *item* cenderung sama dari waktu ke waktu [103]. Pada contoh kasus sederhana untuk sistem rekomendasi film, teknik ini memanfaatkan informasi *rating* dari banyak pengguna. Kita dapat merepresentasikan tingkah laku (*behaviour*) semua pengguna menggunakan matriks utilitas dimana baris merepresentasikan profil pengguna dan kolom merepresentasikan *item*. Sebagai contoh, perhatikanlah Tabel 14.4.

Ada dua metode varian *collaborative filtering* yaitu: (1) *neighborhood-based collaborative filtering* (*memory-based method*) dan (2) *model-based collaborative filtering*. Metode *neighborhood-based collaborative filtering* bekerja dengan fakta bahwa pengguna yang "mirip" memiliki pola yang "mirip" dalam memberikan *rating* untuk *item* [104] (pada Tabel 14.4, pengguna yang mirip memiliki baris yang mirip). Selain itu, *item* yang memiliki kemiripan, akan memiliki pola *rating* yang mirip (pada Tabel 14.4, *item* yang mirip memiliki

	$item_1$	$item_2$	...	$item_N$
$user_1$	2	3	...	4
$user_2$	5	3	...	1
...				
$user_3$	4	1	...	2

Tabel 14.4: Matriks utilitas.

kolom yang mirip). Dengan itu, kita dapat menggunakan perhitungan kemiripan vektor untuk menghitung pengguna mana yang mirip dengan suatu pengguna p. Saat memberikan rekomendasi film bagi pengguna p, kita tunjukkan film-film yang pernah ditonton pengguna yang mirip dengannya, atau kita tunjukkan film-film yang mirip dengan film-film yang pernah ditonton oleh pengguna p.

Untuk *model-based collaborative filtering*, prediksi dibangun dengan menggunakan teknik *machine learning* [104]. Untuk suatu sistem dengan data yang *sparse*, matriks utilitas seperti Tabel 14.4 tidaklah efisien secara memori. Kita dapat memanfaatkan teknik-tenik seperti *matrix factorization*/*principal component analysis* dan *autoencoder* untuk mengurangi ukuran matriks. Silakan membaca lebih lanjut materi-materi tersebut (ingat kembali materi bab 12).

14.2 Peringkasan Dokumen

Meringkas dokumen berarti mengerti keseluruhan isi teks/dokumen, kemudian mampu menyampaikan kembali **sebanyak/seakurat mungkin** maksud dokumen asli, ke dalam bentuk yang lebih singkat [105, 106, 57]. Suatu ringkasan harus lebih pendek dibanding dokumen asli. Dengan demikian, sulit untuk dikatakan bahwa suatu ringkasan dapat memuat keseluruhan isi dokumen. Karena itu, ringkasan hanya memuat **sebanyak/seakurat mungkin** maksud dokumen asli, diberikan *constraint* jumlah kata maksimum pada ringkasan. Ada beberapa jenis peringkasan dokumen dilihat dari berbagai sudut pandang, misal:

1. Banyaknya dokumen yang diringkas, meringkas satu dokumen (*single-document summarization*) [107] atau banyak dokumen (*multi-document summarization*) [108] menjadi satu ringkasan.
2. Indikatif atau Informatif. Indikatif berarti menyediakan *pointer* ke bagian dokumen (misal membuat daftar isi). Informatif berarti meyampaikan sebanyak/seakurat mungkin maksud dokumen asli ke dalam bentuk lebih singkat. Pada masa sekarang, hampir seluruh riset peringkasan dokumen mengacu untuk menyediakan ringkasan yang informatif.
3. Domain. Hasil ringkasan bergantung pada domain dokumen, misalkan berita atau novel. Pada domain berita, hal yang penting untuk dimuat

dalam ringkasan adalah 5W1H (*what, who, when, whom, where, how*) [109], sementara pada novel mungkin kita ingin tahu kejadian-kejadian yang ada (beserta urutannya).

4. Generik, *query-based*, atau *tailored*. Generik berarti menghasilkan ringkasan untuk umum, dalam artian tidak ada *target user* secara spesifik, e.g., *review* buku [105]. *Query-based* artinya menghasilkan ringkasan dokumen berdasarkan *query* yang diberikan *user*, i.e., ringkasan adalah informasi spesifik yang dibutuhkan oleh *user* [110]. *Tailored* berarti menyajikan informasi dengan tipe spesifik. Misalkan pada berita terdapat informasi 5W1H, kita hanya ingin mencari tahu informasi *how*.
5. Ekstraktif [111] atau abstraktif [57]. Ekstraktif berarti ringkasan hanya memuat unit informasi yang ada di dokumen asli. Analoginya seperti memilih potongan dari dokumen asli sebagai ringkasan (*copy*). Abstraktif berarti ringkasan dapat memuat unit informasi yang mungkin tidak ada di dokumen asli. Analoginya seperti mengerti dokumen kemudian menulis ulang (parafrase).

Hal terpenting untuk diingat adalah peringkasan dokumen (apa yang diringkas, ringkasan harus memuat apa, dsb) sangat bergantung pada **tujuan ringkasan** (siapa pembaca ringkasan, untuk apa ringkasan dihasilkan). Diktat ini akan membahas *framework* (kerangka kerja/berpikir) peringkasan dokumen dan bagaimana *machine learning* membantu peringkasan dokumen.

Secara umum, ada beberapa tahap pada proses peringkasan dokumen secara otomatis, terlepas dari tipe peringkasan dokumen yang dijabarkan [105, 112, 113, 106, 114, 109, 108, 107, 57, 115]:

1. *Representation* – Melakukan pemisahan dan representasi unit informasi; pada umumnya adalah kalimat, kata, atau frase. Dalam artian, kita menganggap dokumen tersusun atas sekuens kalimat, kata, atau frase. Kita potong-potong unit informasi pada dokumen lalu, unit informasi dapat direpresentasikan sebagai vektor untuk tiap unitnya [116]. Kita pun dapat menggali hubungan antara unit informasi (misal direpresentasikan sebagai graf) untuk kemudian mencari unit mana yang sentral [109, 117].
2. *Filtering* – Menyaring informasi. Unit manakah yang penting/tidak penting. Unit mana yang dibutuhkan/tidak dibutuhkan. Unit mana yang relevan/tidak relevan. Unit mana yang representatif/tidak representatif. Teknik menyaring informasi sangat bergantung pada representasi unit pada tahap sebelumnya (e.g., vektor atau graf). Secara singkat, tahap ini memilih unit informasi berdasarkan tujuan. Model pembelajaran mesin biasanya paling sering digunakan pada tahap ini.
3. *Generation* – Menghasilkan (*generate*) ringkasan. Bagian ini membutuhkan pengetahuan mengenai bidang pemrosesan bahasa alami (*natural language processing*) (NLP). Pada diktat ini, tidak akan dibahas.

Teknik *machine learning* dapat berperan penting pada keseluruhan tahapan diatas, khususnya tahap kedua (*filtering*). Kami akan memberikan studi

kasus pada subbab 14.2.1 agar lebih jelas. Sistem peringkasan dokumen yang membagi-bagi proses menjadi tahapan-tahapan tersebut disebut *pipelined approach*. Artinya kita melakukan proses diatas sebagai tahap-tahap berbeda (independen satu sama lain). Selain *pipelined approach*, kita juga dapat memandang peringkasan dokumen dengan *single-view approach* (subbab 14.2.2), artinya keseluruhan proses tersebut terjadi bersamaan.

14.2.1 Pipelined Approach

Subbab ini akan memuat cerita singkat tentang peringkasan *paper* [112, 115]. Berdasarkan teori *argumentative zoning* [112], *paper* terdiri dari zona-zona dengan tujuan komunikatif yang berbeda. Dengan bahasa yang lebih mudah, tiap kalimat (unit informasi) pada *paper* memiliki tujuan komunikasi yang berbeda. Misalnya ada kalimat yang menyatakan tujuan penelitian, latar belakang penelitian, atau hasil penelitian. Tujuan komunikasi kalimat disebut *rhetorical categories* [118].

Ringkasan, dalam bentuk abstrak atau judul *paper* memiliki pola [119, 115]. Dalam artian, suatu ringkasan bisa jadi hanya memuat informasi dengan *rhetorical categories* tertentu (*tailored summary*). *Machine learning* dapat digunakan untuk mengklasifikasikan kalimat (unit informasi) ke kelas masing-masing [118, 115]. Pertama-tama setiap kalimat direpresentasikan menjadi *feature vector* (ingat kembali materi bab 3 dan bab 4). Sebagai contoh, *paper* [115] merepresentasikan kalimat pada *paper* sebagai *feature vector* berdasarkan fitur-fitur sebagai berikut:

1. Posisi. Lokasi kalimat dibagi dengan panjang teks (numerik).
2. Kata kunci (*lexicon*). Apakah kalimat memuat kata kunci yang eksklusif untuk *rhetorical category* tertentu (*binary* – ya atau tidak).
3. Bobot kalimat, i.e., seberapa "penting" kalimat tersebut. Hal ini dapat dihitung menggunakan TF-IDF (ingat bab 12) kata pada kalimat (numerik).
4. Kategori sebelumnya. *Rhetorical category* kalimat sebelumnya (nominal).

Setelah diubah menjadi *feature vector*, teknik *machine learning* digunakan untuk mengklasifikasikan kalimat menjadi kelas *rhetorical categories* yang sesuai. Dengan demikian, kita dapat menyaring kalimat berdasarkan tipe informasi mana yang kita inginkan, berdasarkan tujuan peringkasan. Selanjutnya, teknik NLP digunakan untuk memproses informasi yang disaring untuk menghasilkan ringkasan. Materi tersebut diluar bahasan diktat ini.

14.2.2 Single-view Approach

Pada *pipelined approach*, setiap tahapan peringkasan dokumen (*representation*, *filtering*, dan *generation*) dianggap sebagai tahap yang independen satu sama lain. Permasalahannya adalah kesalahan pada suatu tahap akan mempengaruhi tahap berikutnya. Opsi lain adalah dengan melakukan keseluruhan

proses sebagai satu tahapan yang utuh (*end-to-end*). Metode ini memiliki kaitan yang erat dengan teknik *machine translation*, yang pada masa sekarang populer didekati dengan teknik *deep learning* (*sequence to sequence encoder–decoder*) [57, 81, 80, 89, 120, 121]. Dokumen tersusun atas sejumlah T sekuens unit informasi (e.g., kata, frase) $\mathbf{u} = u_1, u_2, \ldots, u_T$. Sebuah ringkasan adalah M buah sekuens unit informasi $\mathbf{r} = r_1, r_2, \ldots, r_M$ dimana $M < T$. Tujuan peringkasan adalah untuk mencari $\mathbf{r}$ terbaik, sedemikian sehingga dapat memenuhi persamaan 14.1.

$$\arg\max_{\mathbf{r}} p(\mathbf{r} \mid \mathbf{u}) \tag{14.1}$$

Pada umumnya, terdapat suatu variabel tambahan θ yang mengendalikan (*govern*) probabilitas tersebut. Sehingga secara lebih tepat, persamaan 14.1 diubah menjadi persamaan 14.2.

$$\arg\max_{\mathbf{r}} p(\mathbf{r} \mid \mathbf{u}, \theta) \tag{14.2}$$

Perhatikan, persamaan 14.2 adalah bentuk yang dapat dimodelkan menggunakan *sequence to sequence encoder–decoder*. Selain peringkasan dokumen, banyak permasalahan pada bidang *natural language processing* memiliki bentuk yang "mirip", seperti: *part-of-speech tagging* [122], *named entity recognition* [123], mesin translasi [81] dan rekonstruksi paragraf [124].

14.3 Konklusi

Teknik *machine learning* sangatlah berguna untuk berbagai macam permasalahan. Tetapi perlu dipahami bahwa teknik yang ada belum mampu memodelkan proses berpikir manusia dengan benar. Proses berpikir manusia sangatlah kompleks, dan model yang ada sekarang ini adalah bentuk simplifikasi. Sebagai contoh, seorang bayi sekali melihat kucing mungkin akan mengetahui kucing lainnya walaupun tidak memiliki rupa yang sama persis. Sementara itu, model *machine learning* harus diberikan banyak sampel. Manusia memiliki otak yang luar biasa hebat karena mampu belajar dengan sedikit contoh.

Persoalan-persoalan yang diberikan pada buku ini secara umum mencakup *supervised* dan *unsupervised learning* saja. Terdapat satu pemodelan masalah penting lainnya yaitu *reinforcement learning* dimana kita ingin memaksimalkan hasil untuk sekuens aksi (sekuens keputusan). Setiap keputusan dapat diberikan bobot yang berbeda. Kemudian, agen memaksimalkan nilai untuk sekuens keputusan (*cummulative reward*).[2] Pada *supervised learning*, kita hanya perlu membuat satu keputusan saja (menggolongkan data ke kelas mana). Pada konferensi-konferensi, para *master* banyak menyebutkan

[2] https://en.wikipedia.org/wiki/Reinforcement_learning

masa depan *machine learning* berada pada *unsupervised* atau *reinforcement learning*. Aplikasi mutakhir yang ada sekarang ini didominasi oleh *supervised learning*. Pada kenyataannya, sebagian besar (hampir semua) data yang ada di dunia ini tidak berlabel. Karena itu, representasi data secara *unsupervised* menjadi topik riset hangat (*representation learning*).

Buku ini telah menjelaskan berbagai macam teknik, tetapi perlu diketahui bahwa materi yang diberikan adalah simplifikasi agar mampu dipahami secara intuitif. Selain itu, buku ini di desain sebagai materi pengantar saja. Untuk memahami teknik *machine learning* secara lebih jauh dan sempurna, penulis menyarankan untuk membaca buku referensi [8, 11].

Materi yang disajikan pada buku ini adalah persoalan-persoalan *machine learning* dari sudut pandang optimisasi. Yaitu mencari himpunan (*set of*) parameter agar model pembelajaran yang dibangun mampu memberi keputusan yang optimal. Kami menyarankan pembaca untuk mencari referensi lebih jauh tentang *machine learning* dari sudut pandang eksplorasi, yaitu mencari *concept space* dimana model dapat bekerja dengan "baik". Sebagai contoh, kita memiliki sebuah mesin (fisik) dengan berbagai macam konfigurasi (parameter). Apabila kita menjalankan mesin dengan konfigurasi parameter tertentu, mesin akan rusak. Tugas kita adalah mencari *parameters space* dimana mesin dapat berjalan dengan optimal dan tidak rusak. Sedangkan, pada persoalan optimisasi, kita mencari satu konfigurasi terbaik.

Machine learning adalah bidang yang sangat luas (lebih luas dari apa yang diceritakan pada buku ini) dan berkembang pesat. Penulis menyarankan pembaca untuk membaca makalah dari konferensi[3] *top-tier* untuk mengetahui perkembangan terkini. Sebagai contoh (diurutkan berdasarkan abjad):[4]

- AAAI. AAAI Conference on Artificial Intelligence
- ACL. Annual Meeting of Association for Computational Linguistics
- CVPR. IEEE Conference on Computer Vision and Pattern Recognition
- EMNLP. Empirical Methods in Natural Language Processing
- ICCV. IEEE International Conference on Computer Vision
- ICLR. International Conference on Learning Representation
- ICML. International Conference on Machine Learning
- IJCAI. International Conference on Artificial Intelligence
- INTERSPEECH. Conference of the International Speech Association
- NeurIPS (dahulu disebut NIPS). Neural Information Processing System
- SIGIR. ACM Special Interest Group in Information Retrieval
- SIGKDD. ACM Special Interest Group in Knowledge Discovery and Data Mining

[3] Kamu lebih mudah mengetahui informasi terkini dari konferensi dibanding jurnal karena proses *peer-review* yang lebih cepat. Pada bidang keilmuan ilmu komputer/teknik informatika, konferensi internasional lebih penting dibanding jurnal.

[4] Konferensi penting untuk bidang pemrosesan bahasa alami terdaftar pada https://aclanthology.coli.uni-saarland.de/

14.4 Saran Buku Lanjutan

Penulis ingin memberikan beberapa rekomendasi bacaan pembelajaran mesin selanjutnya (teoritis). Buku-buku berikut (diurutkan secara subjektif berdasarkan kualitas konten dan kemudahan dimengerti):

1. *Deep Learning* oleh Ian Goodfellow et al. [11]. Banyak yang menggganggap buku ini sebagai "kitab" *deep learning* modern. Buku ini juga mencakup materi matematika dasar (e.g., aljabar linier) yang dibutuhkan untuk mengerti *deep learning*.
2. *Neural Network Methods in Natural Language Processing* oleh Goldberg [1]. Buku ini ditujukan sebagai bahan transisi bagi peneliti bidang pemrosesan bahasa alami untuk menggunakan metode *neural network*. Apabila kamu tidak tertarik dengan pemrosesan bahasa alami, kamu bisa membaca bab 1-5 pada buku ini sebagai pengenalan *neural network*.
3. *Pattern Recognition and Machine Learning* oleh Bishop [8]. Menurut penulis, banyak yang tahu buku ini karena dianggap sebagai "kitab". Penjelasan buku ini sangat matematis dan relatif berat untuk dimengerti. Tetapi, kamu dapat menjadi *master* apabila memahami seluruh materi pada buku ini. Algoritma *machine learning* yang disajikan juga relatif lebih "klasik" dibanding rekomendasi pertama.
4. *Machine Learning* oleh Tom Mitchel [4]. Buku ini memuat materi *machine learning* yang cukup "klasik". Buku ini cocok sebagai materi pengenalan, tapi relatif kurang dalam.

Sedikit tambahan pesan sponsor karena penulis berlatar belakang dari bidang pemrosesan bahasa alami, penulis menyarankan membaca buku-buku (teoritis) berikut sebagai dasar pengetahuan pemrosesan bahasa alami (diurutkan dari konten paling dasar):

1. *Foundations of Statistical Natural Language Processing* oleh Christopher D. Mannning dan Hinrich Schutze [64]. Buku ini dapat dideskripsikan dengan satu kata, **TOP**. Buku ini memuat materi pemrosesan bahasa alami dengan sudut pandang matematis, dapat dianggap sebagai *framework* berpikir yang modern.
2. *Speech and Language Processing* oleh Daniel Jurafsky dan James H. Martin [12]. Buku ini jauh lebih tebal dari buku pertama dan memuat materi yang lebih luas. Penulis merekomendasikan buku ini untuk mengetahui permasalahan-permasalahan pemrosesan bahasa alami.
3. *An introduction to Information Retrieval* oleh Manning et al. [65]. Walaupun berjududul *information retrieval*, penulis pertama kali mengenal konsep *embedding* dari buku ini. Buku ini ditulis dengan apik.
4. *Neural Network Methods in Natural Language Processing* oleh Goldberg [1]. Apabila kamu telah membaca buku-buku yang disebutkan sebelumnya, kamu dapat membaca buku ini untuk transisi ke metode *neural network*.

Pada saat menulis buku ini, penulis berharap bisa menulis pada level diantara buku Tom Mitchel [4] dan Bishop [8] yaitu cukup matematis, lumayan mudah dipahami, dan lumayan dalam. Mudah-mudahan pembaca merasa tujuan ini tercapai dan menikmati membaca buku ini. Sebagai kalimat penutup, terimakasih sudah membaca sampai tahap ini.

Soal Latihan

14.1. Cold Start Problem

(a) Jelaskan apa itu *cold start problem* pada sistem rekomendasi, serta bagaimana cara menangani permasalahan tersebut!
(b) Pada teknik sistem rekomendasi manakah (*content-based filtering* atau *collaborative filtering*) *cold start problem* mungkin muncul? Mengapa?

14.2. Eksplorasi Sistem Rekomendasi

Bangunlah suatu sistem rekomendasi dengan teknik *collaborative filtering* pada dataset MovieLens dengan memanfaatkan library `recommenderlab`[5]!

14.3. Peringkasan Dokumen

(a) Presentasikanlah di kelasmu, *paper* sistem peringkasan dokumen otomatis oleh Kupiec et al. (1995) [116]![6] yang menggunakan *pipelined approach*!
(b) Presentasikanlah di kelasmu, *paper* sistem peringkasan dokumen otomatis oleh Cheng and Lapata [121]![7] yang menggunakan *single-view approach*!
(c) Jelaskan perbedaan, kelebihan, dan kelemahan pendekatan-pendekatan sistem peringkasan dokumen!

[5] https://cran.r-project.org/web/packages/recommenderlab/index.html
[6] https://dl.acm.org/citation.cfm?id=215333
[7] http://www.aclweb.org/anthology/P16-1046