BAB II. LANDASAN TEORI

2.1. Recommendation System

Menurut (Burke, 2002, p. 77) sistem rekomendasi menggambarkan preferensi pengguna dengan tujuan untuk memberikan saran akan *item* yang akan dibeli atau dikunjungi. Sistem ini telah menjadi aplikasi mendasar dalam bidang *e-commerce* dan informasi dengan memberikan saran optimal yang akan memangkas informasi yang tidak relevan yang umunya berjumlah cukup besar, dengan ini pengguna diarahkan terhadap *item* yang sesuai dengan kebutuhan dan preferensi mereka.

Sistem rekomendasi awalnya didefinisikan seperti "orang memberikan rekomendasi sebagai masukan, yang kemudian dikumpulkan oleh sistem dan mengarahkannya ke penerima yang tepat" (Resnick & Varian, Recommender Systems, 1997). Istilah ini kini memiliki konotasi yang lebih luas yang menggambarkan sebuah sistem yang menghasilkan rekomendasi individual sebagai output dan terkadang memiliki efek untuk membawa pengguna untuk melihat *item* yang menarik yang sudah dipersonalisasi. Sistem seperti itu memiliki daya tarik yang tinggi dalam lingkungan di mana jumlah informasi *online* jauh melebihi kemampuan individu untuk melakukan survei dan mengolahnya. Pada saat ini sistem rekomendasi merupakan bagian penting dari beberapa situs *e-commerce* seperti Amazon.com dan CDNow (Schafer, Konstan, & Riedl, 1999).

Ini adalah kriteria yang memisahkan sistem rekomendasi dari sistem pengambilan informasi (*search engine)*. Semantik dari *search engine* pada umumnya mengembalikan semua *item* yang sesuai dengan query dengan tingkat perbandingan *rating* (*page rank*). Teknik seperti *relevance feedback* memungkinkan *search engine* untuk memperbaiki hasil dari permintaan pengguna, dan memberikan bentuk sederhana dari rekomendasi. Mesin pencari ternama seperti Google mengabaikan perbedaan ini yang menyimpulkan bahwa jumlah dari halaman yang dihubungkan dari halaman tertentu (*linked)* menghasilkan hasil yang lebih relevan (Brin & Page, 1998).

Satu pembahasan yang umum dalam penelitian sistem rekomendasi adalah kebutuhan untuk menggabungkan teknik rekomendasi untuk mencapai kinerja yang maksimal. Semua teknik rekomendasi yang dikenal saat ini memiliki keuntungan dan juga kelemahan, dan banyak peneliti telah memilih untuk menggabungkan teknik dengan cara yang berbeda.

2.1.1. Teknik Sistem Rekomendasi

Teknik rekomendasi memiliki sejumlah klasifikasi yang memungkinkan (Terveen & Hill, 2001). Yang menarik dalam pembahasan ini bukanlah jenis *interface* atau properti dari interaksi pengguna dengan rekomendasi, melainkan dasar dari sumber data yang direkomendasikan yang digunakan untuk data yang telah dimasukkan. Secara khusus, sistem rekomendasi memiliki:

- Latar belakang dari sebuah data yang berisikan informasi yang sudah dimiliki sistem sebelum proses rekomendasi dimulai,

- *Input* data yang berisikan informasi hasil dari interaksi pengguna dengan sistem yang akan menghasilkan rekomendasi,
- Algoritma yang menggabungkan latar belakang dan input data untuk menyimpulkan hasil rekomendasi.

Tabel 1 Teknik Rekomendasi

Teknik	**Latar Belakang**	**Input**	**Proses**
Collaborative	*Ratings* dari U *items* dalam I	*Ratings* dari u *items* dalam I	Mengidentifikasi *user* dalam U yang mirip dengan *user* dalam u.
Content-based	*Features* dari *items* dalam I	*Ratings* dari u dalam I	Menghasilkan *classifier* yang memuat perilaku u dan menggunakannya terhadap i
Demographic	Informasi demografik tentang U dan *Ratings* mereka dalam I	Informasi demografik dari U	Mengidentifikasi pengguna yang memiliki demografik yang mirip.

Utility-based	*Features item* dalam I	Fungsi *utility* dalam *item* I yang mendeskripsikan preferensi u	Aplikasikan fungsi terhadap *items* dan tentukan *ranking* dari i
Knowledge-based	Features *item* dalam I. Informasi bagaimana *item* dapat mencukupi kebutuhan *user*	Deskripsi dari keperluan/ketertarikan dari u	Mencari pasangan dari ketertarikan u terhadap i

(Billsus & Pazzani, 1998)

Atas dasar ini, kita dapat membedakan lima teknik rekomendasi yang berbeda seperti yang ditunjukkan pada Tabel I. Asumsikan bahwa *I* adalah himpunan *item* di mana rekomendasi akan dibuat, *U* adalah himpunan pengguna yang diketahui preferensinya, *u* adalah pengguna untuk siapa rekomendasi harus dihasilkan, dan *i* adalah beberapa barang yang digunakan untuk memprediksi preferensi *u*.

Teknik *Collaborative Filtering* mungkin yang paling umum, paling banyak diterapkan dan paling stabil secara teknologi. Sistem rekomendasi kolaboratif mengagregat *ratings* atau hasil rekomendasi dari *items*, mengenali kesamaan antara pengguna atas dasar *rating* mereka, dan menghasilkan rekomendasi baru berdasarkan perbandingan antar-pengguna. Umumnya sebuah profil pengguna

dalam sistem kolaboratif terdiri dari vektor *item* dan penilaian mereka yang terus bertambah seiring pengguna berinteraksi dengan sistem dari waktu ke waktu. Dalam beberapa kasus, penilaian dapat berupa biner (seperti/tidak suka) atau bernilai (numerik) yang menunjukkan tingkat preferensi. Beberapa sistem signifikan yang menggunakan teknik ini adalah GroupLens/NetPerceptions (Resnick, Iacovou, Suchak, Berstrom, & Riedl, GroupLens: An Open Architecture for Collaborative Filtering of Netnews, 1994), Ringo / Firefly (Shardanand & Maes, 1995) dan *Recommender* (Hill, Stead, Rosenstein, & Furnas, 1995). Sistem ini dapat berupa *memory-based*, membandingkan pengguna terhadap satu sama lain secara langsung menggunakan korelasi atau langkah-langkah lain, atau *model-based*, di mana model berasal dari *history* data *rating* dan digunakan untuk membuat prediksi (Breese, Heckerman, & Kadie, 1998). Rekomendasi berbasis model telah menggunakan berbagai teknik pembelajaran termasuk jaringan saraf (Jennings & Higuchi, 1993), pengindeksan laten semantik (Foltz, 1990), dan jaringan Bayesian (Condliff, Lewis, Madigan, & Posse, 1999).

Keuntungan terbesar dari teknik kolaboratif adalah bahwa mereka benar-benar independen dari setiap objek yang direkomendasikan yang dimana umumnya dapat di baca oleh mesin dan dapat bekerja dengan baik untuk *item-item* yang kompleks seperti musik dan film, di mana berbagai variasi berperan untuk banyak variasi dalam preferensi. (Schafer, Konstan, & Riedl, 1999) menyebutnya korelasi orang-ke-orang.

2.2. Collaborative Filtering

Jumlah informasi yang ada di dunia meningkat jauh lebih cepat daripada kemampuan kita untuk memprosesnya. Banyak dari kita yang terkejut dengan jumlah buku-buku baru, artikel jurnal, dan prosiding konferensi yang terus bertambah setiap tahunnya. Teknologi telah secara dramatis membantu ilmuan dan edukator untuk penerbitan dan penyebaran informasi. Sekarang adalah waktu yang tepat untuk menciptakan teknologi yang dapat membantu kita menyaring semua informasi yang tersedia dan menemukan informasi yang tepat sesuai keperluan kita.

Salah satu teknologi seperti yang paling menjanjikan adalah *collaborative filtering* (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, GroupLens: An Open Architecture for Collaborative Filtering of Netnews, 1994). *Collaborative filtering* bekerja dengan membangun *database* preferensi antara produk dan pengguna. Sebagai contoh, pengguna baru, Budi dicocokkan dengan database untuk menemukan pengguna lain di sekelilingnya yang secara historis memiliki kemiripan dalam sifat dengan Budi. *Item* yang umumnya di sukai oleh pengguna lain yang memiliki kemiripan dalam sifat dengan Budi kemudian direkomendasikan kepada Budi, karena kemiripan sifat yang dimiliki oleh mereka. Collaborative filtering telah sangat sukses dalam penelitian, praktek, dan dalam kedua bidang seperti aplikasi penyaringan informasi dan aplikasi *E-commerce*. Namun, masih ada beberapa penelitian penting dalam mengatasi dua tantangan utama bagi sistem rekomendasi dengan menggunakan *collaborative filtering*.

Permasalahan pertama adalah tantangan untuk meningkatkan skalabilitas dari algoritma collaborative filtering. Algoritma ini dapat mencari puluhan ribu potensi *neighbors* secara *real time*, tapi tuntutan sistem modern saat ini adalah untuk mencari puluhan juta *neighbors* yang berpotensial. Selanjutnya, algoritma yang ada pada saat ini memiliki masalah performa dengan pengguna individu yang memiliki informasi yang besar pada suatu situs. Misalnya, jika situs menggunakan pola *browsing* sebagai indikasi preferensi konten, maka situs tersebut mungkin memiliki ribuan titik data untuk pengunjung yang paling sering mengunjunginya. Jumlah dari data inilah yang akan memperlambat pencarian jumlah tetangga yang bisa dicari per detik, yang selanjutnya akan mengurangi skalabilitas.

Tantangan kedua adalah untuk meningkatkan kualitas rekomendasi untuk pengguna. Pengguna perlu rekomendasi yang dapat mereka percayai untuk membantu mereka menemukan barang yang mereka akan suka atau perlukan. Pengguna akan cenderung untuk mencari secara acak untuk *item* yang mereka inginkan jika mereka menolak untuk menggunakan sistem *recommender* yang tingkat akurasinya tidak konsisten untuk mereka.

Dalam beberapa hal, arah dari dua tantangan ini bertolak belakang, karena semakin sedikit waktu algoritma yang dihabiskan untuk mencari *neighbors*, tentunya skalibilitasnya akan bertambah, dengan catatan hasil yang di hasilkan akan semakin buruk kualitasnya. Untuk alasan ini, sangatlah penting untuk mencoba mengatasi dua tantangan bersamaan, sehingga solusi yang d*item*ukan keduanya berguna dan praktis.

Tapestry adalah salah satu implementasi awal dari sistem *recommender* berbasis *collaborative filtering*. Sistem ini berdasarkan pendapat eksplisit orang-orang dari komunitas yang bersifat dekat seperti kelompok belajar atau grup dari sejumlah karyawan. Namun, sistem *recommender* bagi masyarakat yang besar tidak dapat bergantung pada setiap orang yang hanya mengetahui orang lain. Kemudian, beberapa sistem *recommender* otomatis yang berdasarkan *ratings* dikembangkan. Sistem penelitian GroupLens (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, GroupLens: An Open Architecture for Collaborative Filtering of Netnews, 1994) memberikan *pseudonim collaborative filtering* untuk berita dan film di Usenet. Ringo (Shardanand & Maes, 1995) dan *Video Recommender* (Hill, Stead, Rosenstein, & Furnas, 1995) adalah sistem berbasis email dan *web* yang menghasilkan rekomendasi terhadap musik dan film.

Teknologi lainnya juga telah diterapkan pada sistem *recomender*, termasuk jaringan Bayesian, *Clustering*, dan *Horting*. Jaringan Bayesian membuat model berdasarkan pada *training* set dengan *decion-tree* di setiap *node* dan *edge* yang mewakili informasi dari pengguna. Model dapat dibangun *off-line* dalam waktu hitungan jam atau hari. Model yang dihasilkan sangat kecil, sangat cepat, dan pada dasarnya memiliki tingkat akurasi yang mirip dengan metode *nearest neigbor* (Breese, Heckerman, & Kadie, 1998). Jaringan Bayesian mungkin terbukti praktis untuk lingkungan di mana pengetahuan tentang preferensi pengguna berubah secara bertahap (tidak terlalu cepat) sehubungan dengan waktu yang dibutuhkan untuk membangun model, tapi tidak cocok untuk lingkungan di mana model preferensi pengguna harus diperbarui dengan cepat atau sering.

Teknik *Clustering* bekerja dengan mengidentifikasi kelompok pengguna yang terlihat memiliki preferensi yang sama. Setelah *cluster* dibuat, prediksi bagi satu individu dapat dihasilkan dengan mencari rata-rata *ratings* dari pengguna lain dalam *cluster* itu. Beberapa teknik pengelompokan mewakili masing-masing pengguna dengan partisipasi parsial dalam berbagai kelompok. Prediksinya maka berupa rata-rata di *cluster* yang ditimbang dengan tingkat partisipasi. Teknik *clustering* biasanya menghasilkan rekomendasi yang tidak bersifat personal dibanding dengan metode lain, dan dalam beberapa kasus, clustering memiliki akurasi lebih buruk dari algoritma *nearest neighbor* (Breese, Heckerman, & Kadie, 1998). Setelah pengelompokan selesai, performa yang dihasilkan bisa sangat baik, karena ukuran kelompok yang harus dianalisis jauh lebih kecil. Teknik *clustering* juga dapat diterapkan sebagai langkah pertama untuk menyaring kandidat pengguna yang ditetapkan dalam algoritma *nearest neighbor* atau untuk penyebaran hasil komputasi dari *nearest neighbor* ke beberapa mesin *recommender* lainnya. Dengan membagi populasi ke dalam kelompok, cenderung akan mengurangi akurasi atau rekomendasi kepada pengguna di sekitar *cluster* itu.

Horting adalah teknik berbasis graf di mana node pengguna dan *edge* antara node menunjukkan tingkat kesamaan antara dua pengguna (Aggarwal, Wolf, Wu, & Yu, 1999). Prediksi diproduksi dengan menjalankan grafik ke node terdekat dan menggabungkan *rating* dari pengguna yang berada di dekatnya. Horting berbeda dari *nearest neighbor* karena grafik dapat berjalan melalui pengguna lain yang belum memberikan nilai kepada *item* yang di pilih, sehingga menjelajahi hubungan transitif yang tidak dilakukan oleh metode *nearest neighbor*. Dalam

satu penelitian yang menggunakan data sintetik, Horting menghasilkan prediksi yang lebih baik daripada algoritma *nearest neighbor* (Aggarwal, Wolf, Wu, & Yu, 1999).

(Schafer, Konstan, & Riedl, 1999) menyediakan taksonomi rinci dan contoh dari sistem *recommender* yang digunakan dalam *E-commerce* dan bagaimana mereka dapat memberikan personalisasi individu dan pada saat yang bersamaan juga dapat mendapatkan loyalitas pelanggan. Meskipun sistem ini telah sukses di masa lalu, d*item*ukan hasil dari penggunaan sistem ini yaitu beberapa keterbatasan mereka seperti masalah *sparsity* dalam *dataset*, masalah yang terkait dengan dimensi tinggi dan sebagainya. Masalah *sparsity* dalam sistem *recommender* telah dibahas dalam (Good, et al., 1999). Masalah yang terkait dengan dimensi tinggi dalam sistem *recommender* telah dibahas dalam (Billsus & Pazzani, 1998), dan penerapan teknik pengurangan dimensi untuk mengatasi masalah ini telah diteliti di (Sarwar B. , Karypis, Konstan, & Riedl, 2000).

2.2.1. User Based Collaborative Filtering

User based collaborative filtering pertama kali diperkenalkan oleh sistem penelitian GroupLens (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, GroupLens: An Open Architecture for Collaborative Filtering of Netnews, 1994) untuk memberikan prediksi yang dipersonalisasi untuk artikel dan berita di Usenet. Rincian dasar pelaksanaan *user based collaborative filtering* tetap sama seperti yang diusulkan dalam (Konstan, et al., 1997). Sistem *collaborative filtering* pada umumnya digunakan untuk memecahkan masalah prediksi atau masalah prediksi top-N. Untuk pengguna aktif U_a di set pengguna *U*, masalah dalam prediksinya adalah untuk memprediksi *rating* pengguna aktif yang akan

diberikan ke *item* itu, dari himpunan semua *item* U_a yang belum dinilai. Langkah-langkah yang dilakukan dalam *user based collaborative filtering* untuk membuat prediksi untuk pengguna U_a adalah sebagai berikut:

- Langkah 1: Kesamaan antara pengguna aktif U_a dan setiap pengguna lainnya dihitung.
- Langkah 2: Berdasarkan nilai kemiripan mereka dengan pengguna U_a, mengatur pengguna k, paling mirip dengan pengguna aktif U yang kemudian dipilih.
- Langkah 3: Akhirnya, prediksi untuk *item* ini dihasilkan dengan mengambil rata-rata tertimbang dari peringkat yang diberikan oleh k tetangga mirip dengan barang itu.

Pada langkah 1 untuk menghitung kesamaan antara pengguna korelasi koefisien Pearson digunakan. Dengan set *item* yang telah dinilai baik oleh pengguna u dan v dinotasikan dengan I, maka koefisien kemiripan ($Sim_{u,v}$) antara mereka dihitung sebagai:

$$(Sim_{u,v}) = \frac{-\sum i\in I(r_{u,i}-\bar{r}_u)(r_{v,i}-\bar{r}_v)}{\sqrt{\sum i\in I(\mathrm{r_{u,i}}-\bar{\mathrm{r}}_\mathrm{u})^2}\sqrt{\sum i\in I(\mathrm{r_{v,i}}-\bar{\mathrm{r}}_\mathrm{v})^2}} \quad (1)$$

Di sini $r_{u,i}$ menunjukkan peringkat pengguna *u* untuk *item i*, dan r_u adalah rata-rata *rating* yang diberikan oleh pengguna *i* dihitung dari semua yang dinilai oleh *u*. Demikian pula, r_v menunjukkan peringkat pengguna *v* untuk *item i*, dan r_v adalah rata-rata *rating* yang diberikan oleh pengguna *v* dihitung dari semua dinilai oleh *v*. Dalam beberapa kasus, untuk menghitung kesamaan vektor diperlukan *cosine similarity*. Dalam (Breese, Heckerman, & Kadie, 1998) melalui

evaluasi eksperimental, mereka telah menunjukkan persamaan vektor menggunakan pearson *correlation* menghasilkan akurasi yang lebih baik .

Setelah persamaan vektor dihitung, satu set pengguna yang paling mirip dengan pengguna aktif U_a dipilih pada langkah 2. Ada dua cara di mana satu set pengguna yang sama dapat dipilih. Salah satunya adalah untuk memilih semua pengguna yang korelasi dengan pengguna *U* di atas nilai korelasi tertentu yang sudah ditetapkan atau bisa juga dengan memilih satu set pengguna *top-k* berdasarkan kemiripannya.

Eksperimen telah menunjukkan bahwa pendekatan *top-k* menghasil hasil yang lebih baik daripada pendekatan *threshold* (Herlocker, Konstan, Borchers, & Riedl, 1999). Nilai *k* diperoleh dengan melakukan percobaan pada data yang bergantung pada *dataset* yang digunakan. Pada langkah 3 untuk menghitung prediksi untuk *item i* dari pengguna target *u*, sebuah rumus yang disesuaikan dengan bobot *item* digunakan untuk memperhitungkan fakta bahwa pengguna yang berbeda memiliki distribusi *ratings* yang berbeda.

$$(P_{u,i}) = \bar{r}_u + \frac{\sum v \in V \; Sim_{u,v} \; (r_{v,i} - \bar{r}_v)}{\sum v \in V \; |Sim_{u,v}|} \quad (2)$$

Dimana, *v* merupakan kumpulan *k* pengguna yang sama. Sementara untuk menghitung prediksi, yang digunakan hanya para pengguna di set *v*, yang telah dinilai *item I*.

2.2.2. Item Based Collaborative Filtering

Perbedaan utama antara *item based collaborative filtering* (Sarwar B. , Karypis, Konstan, & Riedl, 2001) dan *user based collaborative filtering* adalah bahwa *item based collaborative filtering* menghasilkan prediksi berdasarkan

model *item-item* yang memiliki persamaan dibandingkan dengan persamaan *user-user*. Dalam *item based collaborative filtering*, pertama, kesamaan di antara berbagai *item* dihitung. Kemudian dari set *item* yang dinilai oleh target pengguna, *k item* yang paling mirip dengan *item* yang dinilai oleh target akan dipilih. Untuk menghitung prediksi *item* yang sudah dipilih, rata-rata diambil dari *ratings* target pengguna di *k item* serupa yang sebelumnya dipilih. Bobot hitung yang digunakan adalah nilai koefisien kemiripan antara *item* target dan *k* barang yang serupa dinilai oleh target pengguna. Untuk menghitung *item-item* yang kesamaan, persamaan *cosine similarity* digunakan. Biarkan set pengguna yang dinilai kedua *item i* dan *j* akan dilambangkan dengan *U*, maka koefisien kemiripan $Sim_{i,j}$ antara mereka dihitung sebagai

$$(Sim_{u,j}) = \frac{\sum u \in U(r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum u \in U(\mathrm{r_{u,i}} - \mathrm{\bar{r}_u})^2}\sqrt{\sum u \in U(\mathrm{r_{u,j}} - \mathrm{\bar{r}_u})^2}} \quad (3)$$

Di sini $r_{u,i}$ menunjukkan peringkat pengguna *u* untuk *item i*, dan r_u adalah rata-rata *rating* yang diberikan oleh pengguna *u* dihitung lebih dari semuanya dinilai oleh *u*. Demikian pula, $r_{u,j}$ menunjukkan peringkat pengguna u untuk *item j*.

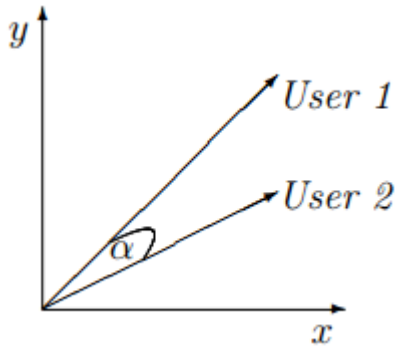
Untuk menghitung nilai prediksi untuk *item* sasaran *I* untuk pengguna target u, kita menggunakan rumus berikut.

$$(P_{u,i}) = \frac{\sum j \in I \; Sim_{i,j} * r_{u,j}}{\sum j \in I \; |Sim_{i,j}|} \quad (4)$$

Dalam persamaan 4, *I* mewakili set *k item* yang paling mirip dengan menargetkan *item i* yang telah dinilai oleh pengguna target *u*. Seperti disebutkan sebelumnya, $r_{u,j}$ menunjukkan peringkat pengguna *u* untuk *item j*

2.2.3. Cosine Similarity

Sudut cosinus antara dua vektor dapat dihitung dengan penggunaan Cosine Similarity (Deshpande & Karypis, 2004). Gambar 2.5 menggambarkan dua vektor di ruang dua dimensi. α adalah sudut cosinus antara dua vektor tersebut. Hasil *Cosine Similarity* adalah dalam rentang [0; 1], di mana 1 mewakili kesamaan penuh, dan 0 tidak ada kesamaan antara sudut dua vektor. Perhitungan yang mengambil pendekatan berbasis-*item* ke rekening didefinisikan oleh persamaan 2.3.3.1. Sedangkan persamaan 2.3.3.2 digunakan jika menggunakan pendekatan *user based*.



Gambar 1 Dua vektor yang dibuat oleh penggunaan peringkat pengguna di ruang dua dimensi (Kenji, 2016)

$$CS_{sim}(i,j) = \frac{\vec{\imath}.\vec{\jmath}}{\|\vec{\imath}\|.\|\vec{\jmath}\|} \quad (2.3.3.1)$$

$CS_{sim}(i,j)$ adalah kesamaan antara dua vektor *i* dan *j*. $\vec{\imath}$ merupakan vektor *i* dan $\vec{\jmath}$ adalah vektor *j*. $\vec{\imath}.\vec{\jmath}$ adalah dot produk dari vektor *i* dan vektor *j*. $\|\vec{\imath}\|$ adalah besarnya vektor $\vec{\imath}$ dan $\|\vec{\jmath}\|$ adalah besarnya vektor $\vec{\jmath}$.

$$CS_{sim}(u,v) = \frac{\vec{u}.\vec{v}}{\|\vec{u}\|.\|\vec{v}\|} \quad (2.3.3.2)$$

$CS_{sim}(u, v)$ adalah kesamaan antara dua vektor *u* dan *v*. $\vec{u}$ merupakan vektor *u* dan $\vec{v}$ adalah vektor *v*. $\vec{u} \,.\, \vec{v}$ adalah titik produk dari vektor *u* dan vektor *v*. $\|\vec{u}\|$ adalah besarnya vektor $\vec{u}$ dan $\|\vec{v}\|$ adalah besarnya vektor $\vec{v}$.

2.3. Yelp

Yelp adalah sebuah perusahaan multinasional Amerika yang berpusat di San Francisco, California. Mereka mengembangkan, host dan memasarkan Yelp.com dan aplikasi Yelp, yang menyediakan ulasan dari berbagai penggunanya tentang bisnis lokal, serta layanan reservasi online SeatMe dan pelayanan delivery makanan secara online Eat24. Perusahaan ini juga melatih usaha kecil tentang bagaimana menanggapi ulasan, mengadakan acara sosial untuk para pengulas, dan memberikan data tentang bisnis, termasuk skor pemeriksaan kesehatan.

Yelp didirikan pada tahun 2004 oleh mantan karyawan PayPal, Russel Simmons dan Jeremy Stoppelman. Yelp berkembang dengan cepat dan mengangkat beberapa pendanaan dari berbagai *investor*. Pada tahun 2010 mereka mandapatkan revenue sebesar $30 juta dan website mereka telah menghasilkan lebih dari 4.5 juta ulasan dari penggunanya. Dari 2009-2012, Yelp diperluas di seluruh Eropa dan Asia. Pada tahun 2009 itu memasuki beberapa negosiasi dengan Google untuk potensial akuisisi. Yelp menjadi perusahaan publik pada bulan Maret tahun 2012 dan menjadi menguntungkan untuk pertama kalinya dua tahun kemudian. Pada tahun 2016, Yelp.com memiliki 135 juta pengunjung bulanan dan 95 juta ulasan. Pendapatan perusahaan saat ini berasal dari bisnis periklanan.

2.4. Machine Learning

Definisi yang di berikan Tom Mitchell terhadap Mahcine Learning adalah "Sebuah program komputer yang belajar dari pengalaman E dihubungkan dengan beberapa *task* T dan beberapa pengukur kinerja P. Jika kinerja pada T yang diukur dengan P dapat ditingkatkan dengan pengalaman E." (Mitchell, 1997). Misalnya, jika kita menginginkan program mengenali dan mengklasifikasi kata tulisan tangan dalam gambar (tugas T), Kita dapat mengatur database kata-kata yang ditulis tangan dengan klasifikasi tertentu (pengalaman E) dan, jika telah berhasil belajar, program kita akan dapat mengenali kata tulisan tangan dengan benar.

Algoritma pembelajaran mesin telah terbukti dari nilai praktis yang besar dalam berbagai aplikasi domain. Mereka terutama berguna dalam masalah data mining di mana database besar mungkin berisi informasi implisit berharga yang dapat ditemukan secara otomatis (misalnya, untuk menganalisis hasil dari perawatan medis dari database pasien atau untuk belajar aturan umum untuk kelayakan kredit dari database keuangan). Pada saat ini masih kurang dipahami secara benar kelemahan manusia yang tidak memiliki pengetahuan yang dibutuhkan untuk mengembangkan algoritma yang efektif (misalnya, pengenalan wajah manusia dari gambar). Dan domain dimana program harus dinamis beradaptasi dengan perubahan kondisi (misalnya, mengendalikan proses manufaktur di bawah perubahan harga saham atau beradaptasi dengan perubahan membaca perminatan individu).

Pembelajaran mesin mengambil ide-ide dari satu set beragam ilmu Teknik Informatika, termasuk kecerdasan buatan, probabilitas dan statistik, kompleksitas komputasi, teori informasi, psikologi dan neurobiologi, teori kontrol, dan filsafat.

Masalah pembelajaran yang terdefinisi dengan baik membutuhkan tugas yang dengan baik telah ditentukan, kinerja metrik, dan sumber pengalaman pelatihan. Merancang pendekatan pembelajaran mesin melibatkan sejumlah pilihan desain, termasuk memilih jenis pengalaman pelatihan, fungsi target yang harus dipelajari, representasi untuk fungsi target ini, dan algoritma untuk belajar fungsi target dari contoh-contoh pelatihan (Mitchell, 1997).

1) Memilih Pengalaman Pelatihan: Pertama, kita harus memilih jenis pengalaman pelatihan yang akan dipelajari oleh sistem. Jenis pengalaman pelatihan yang tersedia dapat memiliki dampak yang signifikan terhadap keberhasilan atau kegagalan dari proses pembelajaran. Salah satu kuncinya adalah apakah pengalaman pelatihan memberikan umpan balik langsung atau tidak langsung mengenai pilihan yang dibuat oleh sistem kinerja. Sebuah atribut penting kedua dari pengalaman pelatihan adalah sejauh mana pembelajaran menguasai urutan contoh pelatihan. Sebuah atribut penting dari pengalaman pelatihan adalah seberapa baik itu mewakili distribusi contoh di mana kinerja sistem final P harus diukur.
2) Memilih Target Fungsi: Pilihan desain berikutnya adalah menentukan apa jenis pengetahuan yang akan dipelajari dan bagaimana ini akan digunakan oleh program.

3) Memilih Perwakilan untuk Fungsi Target: Sekarang kita telah menentukan fungsi yang ideal, kita harus memilih sebuah representasi bahwa program akan digunakan untuk menggambarkan fungsi yang akan melakukan proses pembelajaran. Hasil yang telah diuraikan rumusan asli dari masalah belajar adalah memilih jenis pengalaman pelatihan, fungsi target untuk dipelajari, dan representasi untuk fungsi target ini.
4) Memilih Fungsi Pendekatan Algoritma: Dalam rangka untuk mempelajari fungsi yang ingin dibuat maka kita membutuhkan sekumpulan contoh pelatihan.

Ada dua pengaturan utama yang digunakan oleh sistem pembelajaran mesin pada umumnya yaitu (Nilsson, 1998):

1) *Supervised Learning*, program ini dilatih pada set yang telah ditentukan dari contoh pelatihan, yang kemudian memfasilitasi kemampuannya untuk mencapai kesimpulan yang akurat ketika diberikan data baru.
2) *Unsupervised Learning*, bertugas mencari hubungan dalam data. Tidak ada contoh pelatihan yang digunakan dalam proses ini. Sebaliknya, sistem ini diberi kumpulan data dan bertugas mencari pola dan korelasi dalamnya. Sebuah contoh yang baik adalah mengidentifikasi kelompok erat dari teman-teman di jaringan data sosial.