BAB II
LANDASAN TEORI

2.1 Recommender System

Recommender system adalah perangkat lunak dan teknik yang menyediakan saran dari *items* yang diinginkan pengguna [5]. Saran tersebut berelasi terhadap macam – macam proses penentuan keputusan seperti apakah *items* yang akan dibeli?, apakah musik yang akan didengarkan?, atau apa berita online yang akan dibaca?. *Items* adalah objek dari rekomendasi. *Items* dapat menjadi sebuah karakter dengan segala kompleksitasnya dan nilai atau *utility* dari *items* itu sendiri.

Recommender system sendiri merupakan alat untuk memberikan saran atau membuat prediksi dari item/produk yang kita inginkan berdasarkan *user profile* atau *item profile* [2][4]. Produk tersebut dapat berupa buku, film, musik, handphone dll.

Recommender system juga dapat didefinisikan sebagai sebuah sistem cerdas yang dapat memberikan saran dengan sendirinya dan diprogram oleh seorang programmer. Sistem rekomendasi akan menawarkan kemungkinan dari penyaringan informasi personal sehingga hanya informasi yang sesuai kebutuhan dan preferensi pengguna yang akan ditampilkan di sistem dengan menggunakan teknik atau model rekomendasi. Informasi yang diberikan oleh pengguna dapat diperoleh secara eksplisit dan implisit. Informasi yang eksplisit adalah informasi yang langsung didapatkan dari pengguna. Sedangkan, informasi yang implisit adalah informasi yang diperoleh tanpa diketahui oleh pengguna.

Ada berbagai macam metode untuk membuat sistem rekomendasi. Metode yang digunakan haruslah sesuai dengan permasalahan dan dapat menghasilkan rekomendasi yang sesuai. Metode atau pendekatan yang dipilih pada sistem rekomendasi bergantung pada permasalahan yang akan diselesaikan, teknik rekomendasi yang berbeda-beda digunakan untuk aplikasi yang berbeda, dasar dari suatu tujuan dan objektif dari sebuah aplikasi. Beberapa metode untuk membuat sistem rekomendasi antara lain *Content based filtering*, *Collaborative filtering* , *Knowledge-based filtering* dan *hybrid based filtering*. Tetapi, untuk sistem rekomendasi E-Commerce ini metode yang digunakan adalah *Knowledge based filtering.*

2.1.1 Content based filtering

Content based fitering adalah hasil dari penelitian penyaringan informasi dalam sistem berbasis konten[4]. Metode ini membuat rekomendasi dengan menganalisis item yang telah memiliki bobot tinggi oleh pengguna tersebut. Menurut sharda, sistem rekomendasi berbasis

konten dimulai dengan memahami kebutuhan pengguna, preferensi dan kendala jika ada, kemudian informasi ini digabungkan dengan log interaksi user sebelumnya (jika ada) untuk membangun profil pengguna[4]. Sistem rekomendasi mencocokan profil pengguna dengan informasi tentang suatu produk yang telah tersimpan dalam basisdata.

Seperti yang telah dijelaskan sebelumnya, *Content based filtering* menyimpan kriteria pengguna ke dalam profil pengguna. profil pengguna deskripsi dari item yang disukai oleh pengguna personal dan interaksi antara pengguna dengan sistem rekomendasi. Interaksi bisa termasuk penyimpanan query penggunan yang terlihat dari aktifitas penggunan terhadap item.Riwayat dari interaksi pengguna dapat menyaring item yang telah dipertimbangkan user sebelumnya. Informasi pengguna bisa didapatkan dengan eksplisit langsung dari pengguna atau implisit dari *software agent*.

Content based filtering menggunakan asumsi bahwa item dengan *similarity* yang mendekati dengan *item* yang lainnya akan diberi *rating* yang sama juga[4]. generasi rekomendasi menitikberatkan sistem rekomendasi untuk melibatkan perbandingan *feature* yang diekstrak dari *item-item* yang tidak terlihat atau tidak diberi *rating* dengan gambaran konten yang terdapat pada profil pengguna. *Item-item* tersebut harus memiliki *similarity* yang cukup tinggi dengan profil pengguna untuk dapat direkomendasikan. Kelebihan *recommender system* dengan pendekatan *content based filtering* adalah memiliki kemampuan merekomendasikan *item* (contoh:film,lagu,artikel,smartphone dan lain-lain) yang sifatnya baru bagi user, karena prinsip kerjanya yaitu dengan melihat deskripsi *content* yang dikandung oleh *item* yang pernah diberi nilai *rating* tinggi sebelumnya oleh user.

Di dalam mendesain *content based filtering* terdapat dua permasalahan utama, yaitu menemukan sebuah representasi dari beberapa item dan membuat *profile* yang dapat membuat rekomendasi sebuah *item*.

2.1.2 Collaborrative filtering

Collaborrative filtering membuat rekomendasi mengenai preferensi user berdasarkan user yang memiliki informasi kesamaan *taste.* Asumsi yang dapat ditekankan pada method ini bahwa siapa yang setuju dengan masa lampau maka akan setuju juga di masa mendatang[10]. Dalam artian system dapat menyamai perilaku mengenai rekomendasi produk user kepada temannya karena dia akan tahu dan percaya bahwa *taste* yang *similiar* dengan temannya mempunyai kemiripan dengan produk yang disukai.

Kelebihan system rekomendasi yang menggunakan *Collaborrative filtering* adalah pendekatan yang dilakukan dapat bekerja di dalam domain yang kontennya sedikit berasosiasi

dengan *item* atau tempat dimana *content* tersebut sulit dianalisis menggunakan computer seperti ide, masukkan, atau opini. Metode *Collaborrative filtering* juga mempunyai kemampuan untuk menyediakan rekomendasi yang tidak terduga, misalnya dapat merekomendasikan *item* yang relevan kepada pengguna sekaligus tidak mengandung konten dari profil tersebut.[10]

Pada gambar 1 merupakan salah satu contoh dari sistem rekomendasi berdasarkan *collaborrative filtering*

Gambar 1 . freco.com contoh sistem rekomendasi *collaborative filtering*

Dengan alasan pendekatan tradisional *collaborative-filtering & content-based filtering* tidak menyediakan strategi rekomendasi yang efektif. karena kedua strategi tersebut melihatkan performa yang kurang memuaskan dengan penggunaan data yang terlalu banyak dan performa sistem rekomendasi tidak menunjukkan kemajuan tanpa partisipasi user banyak maka untuk menyelesaikan masalah diatas peneliti menggunakan review dari consumer yang berpengalaman dan sudah tersedia di internet untuk menentukan rekomendasi populer produk menurut preferensi dari consumer.[2]

2.1.3 Knowledge-based filtering

Knowledge-based filtering mengandalkan kedetilan *knowledge* mengenai karakteristik *item*. Manfaat dari system ini adalah mampu mengatasi perhitungan rekomendasi terhadap data yang dibutuhkan jika tidak ada data rating. Karakter dari *knowledge-based* adalah membuat basis pengetahuan tentang selera user dan merekomendasikan item berdasarkan pengetahuan tersebut. Rekomendasi dapat dihitung secara independent dari user ratings, selain itu *similarities* atau kesamaan antara profil user dan *item*.[10]

Dua tipe dasar dari *knowledge-based filtering* adalah *contrainst-based* dan *case-based.* Pendekatan kedua system tersebut mirip pada *terms* dari proses rekomendasi dimana user harus menetapkan persyaratan, dan system akan mencoba untuk mengidentifikasi sebuah solusi. Jika tidak ada solusi yang ditemukan user harus merubah persyaratan. Adapun perbedaan dari *constraint-based dan case-based* ialah :

Case-based fokus pada penemuan kemiripan item pada perbedaan type dari kemiripan yang diperhitungkan. *Case-based* juga menggunakan kemiripan *metrics* untuk menemukan item yang cocok dengan sebelumnya menggunakan *threshold* untuk menspesifikasikan user. *Contraint-based* fokus pada rekomendasi item yang ditetapkan oleh individu, pencarian untuk kumpulan item yang mengisi *rules* rekomendasi.

2.1.4 Hybrid-based filtering

Hybrid-based filtering adalah *recommender system* berdasarkan kombinasi dari berbagai macam teknik[4]. Sebuah sistem *hybrid-based* mengkombinasi teknik A dan B mencoba untuk menggunakan kelebihan dari A untuk mengatasi kekurangan dari teknik B. Sebagai contoh, teknik *Collaborative filtering* memiliki kekurangan dari masalah terkait masalah item baru, teknik tersebut tidak dapat merekomendasikan item yang tidak mempunyai *rating*. Masalah tersebut tidak membatasi teknik *Content-based filtering* ketika prediksi untuk item baru berdasarkan pada deskripsi atau *feature* item itu sendiri. Sebuah teknik baru *hybrid-based filtering* dapat dibuat dari dua atau lebih teknik *recommender system*.[4]

Berikut dijelaskan pada tabel 1 tentang kelebihan dan kekurangan dari *content-based, collaborative, knowledge-based* dan *hybrid-based recommendation.*

Tabel 1 . Kelebihan dan kekurangan metode *recommender system*

No	Metode	Kelebihan	Kekurangan

1.	*Content-based*	1.Saling bebas dengan user lain 2.Transparansi 3.Dapat merekomendasikan item baru yang belum pernah diberi rating	1.Keterbatasan fitur konten sehingga butuh domain knowledge. 2.*Overspecialization* 3.User baru akan mendapatkan rekomendasi yang kurang sesuai berdasarkan preferensi karena rating masih terbatas pada beberapa item saja.
2.	*Collaborative filtering*	1.Dapat mengidentifikasi preferensi yang tepat Tidak membutuhkan domain *knowledge* 2.Kualitas rekomendasi meningkat 3.Merekomendasikan secara personal	1.Bergantung pada historical data yang besar. 2. Menunjukan keanehan statistik di dalam data 3. Tidak senssitif terhadap perubahan preferensi.
3.	*Knowledge-based*	1.Tidak memerlukan rating *user* 2. feedback terhadap preferensi *user* jelas 3. sensitif terhadap perubahan preerensi	1. Teknik pengetahuan 2.Kemampuan rekomendasi statis

4.	*Hybrid-based*	1.Dapat mengidentifikasi preferensi dengan tepat 2.Tidak membutuhkan domain *knowledge*. 3.Kualitas rekomendasi meningkat. 4.*Feedback* terhadap preferensi *user* jelas 5.Dipengaruhi perubahan preferensi	1. Teknik pengetahuan

2.2 Explanation

Kemampuan Sebuah *recommender system* dapat terlihat terpercaya dan meyakinkan apabila menggunakan fasilitas penjelasan[4]. Sebuah penjelasan (*explanation*) dalam sebuah *recommender system* dapat memegang peranan penting dalam meningkatkan *user experience*. User dapat membuat keputusan dari hasil rekomendasi yang memiliki fasilitas penjelasan.

Berdasarkan [10] ada tujuh kemungkinan tujuan *explanation* dalam sebuah recmmender *system*, yaitu:

1. *Transparency*, menjelaskan bagaimana suatu *system* dapat bekerja.
2. *Scrutability*, mengijinkan pengguna untuk mengatakan kepada *system* bahwa sesuatu itu salah.
3. *Trustworthiness*, meningkatkan kepercayaan pengguna terhadap *system*.
4. *Effectiveness* , membantu pengguna untuk membuat suatu keputusan yang baik.
5. *Persuasiveness*, meyakinkan pengguna untuk mencoba atau membeli suatu produk.
6. *Efficiency*, membantu pengguna untuk membuat keputusan lebih cepat.
7. *Satisfaction*, meningkatkan kenyamanan pengguna pada saat berinteraksi dengan *system*.

Proses pembangkitan penjelasan dilakukan dengan cara penelusuran *backtrack* pada masing-masing lintasan *node* dan relasinya dalam *user* profil model, berawal dari suatu *node* produk yang akan direkomendasikan. Untuk menghasilkan penjelasan yang lebih natural,

pembangkitan penjelasan akan menggunakan template diapdukan dengan relasi serta info dari *node* hasil penelusuran *backtrack* pada *user* profil model . Template yang digunakan adalah sebagai berikut,

<produk> mendukung kebutuhan Anda, karena <produk> <relasi> <nodeinfo> <relasi> <nodeinfo>

2.3 Text Mining

Proses pembuatan sistem rekomendasi mencakup beberapa langkah. Adapun langkah pertama adalah proses *text mining* yang akan memetakan *review* customer ke dalam struktur *Ontology*[7].

2.3.1 Definisi Text Mining

Text Mining adalah proses menambang data berupa teks dengan sumber data biasanya dari dokumen dan tujuannya adalah mencari kata-kata yang mewakili dalam dokumen sehingga dapat dilakukan analisa keterhubungan dalam dokumen. Data teks akan diproses menjadi data numerik agar dapat dilakukan proses lebih lanjut. Sehingga dalam *text mining* ada istilah *preprocessing data,* yaitu proses pendahulu yang diterapkan terhadap data teks yang bertujuan untuk menghasilkan data numerik. Adapaun proses preprocessing yang dilakukan antara lain :

- Tokenizing

  Teks dalam bentuk mentah mereka, bagaimanapun, hanya rangkaian karakter tanpa informasi eksplisit tentang batas kata dan kalimat. Sebelum diproses lebih lanjut, teks perlu tersegmentasi ke dalam kata-kata dan kalimat. Proses ini disebut tokenization. *Tokenization* membagi urutan karakter menjadi kalimat dan kalimat ke dalam token. Tidak hanya kata-kata dianggap sebagi bukti, tetapi juga angka, tanda baca, tanda kurung dan tanda kutip.

- Filtering

  Tahap *filtering* adalah tahap mengambil kata-kata penting dari hasil token. Bisa menggunakan algoritma *stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). *Stoplist / stopword* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan bag-of-words. Contoh stopwords adalah “that”,”and”,”at”,”from” dan seterusnya.

- Stemming

  Tahap *stemming* adalah tahap mencari *root* kata dari tiap kata hasil *filtering*. Pada tahap ini dilakukan proses pengembalian berbagai bentukan kata ke dalam suatu

representasi yang sama. Tahap ini kebanyakan dipakai untuk teks berbahasa inggris dan lebih sulit diterapkan pada teks berbahasa Indonesia. Hal ini dikarenakan bahasa Indonesia tidak memiliki rumus baku yang permanen.

Proses *preprocessing* diatas sudah menyatu dalam tools yang penulis gunakan yaitu TMSK dan RIKTEXT.

2.4 Opinion Mining

2.4.1 Definisi Opinion Mining

Opinion mining merupakan cara untuk memperoleh informasi dari proses mining yang khusus terhadap opini-opini yang ada. Informasi tekstual di dunia terbagi menjadi 2 yaitu informasi fakta dan informasi opini. Fakta merupakan pernyataan yang bersifat objektif mengenai suatu entitas atau kejadian. Sedangkan opini merupakan suatu pernyataan yang bersifat subjektif yang mencerminkan persepsi atau pemikiran seseorang. *Opinion mining* juga dikenal dengan *sentimen analysis*. *Opinion Mining* sendiri memiliki definisi yaitu suatu dokumen yang mengandung opini-opini atau *sentimen analysis. Opinion Mining* sendiri memiliki definisi yaitu suatu dokumen yang mengandung opini-opini atau sentiment tentang suatu objek, *opinion mining* bertujuan untuk mengekstrak atribut dan komponen dari objek tersebut yang telah dikomentari dan bertujuan untuk menentukan suatu komentar tersebut positif, negatif, atau netral. Secara umum ada empat topik dalam *opinion mining* yaitu [5] :

- Model abstrak dari *opinion mining*.

  Merupakan topik dalam *opinion mining* yang berkonsentrasi mencari struktur kata-kata yang termasuk fitur ataupun opini dalam suatu kalimat.

- *Sentimen classification*.

  Merupakan topik dalam *opinion mining* yang berkonsentrasi untuk menentukan suatu kalimat opini positif, negatif , ataupun netral.

- *Feature-based opinion mining* and *summarization*.

  Merupakan topik dalam *opinion mining* yang berkonsentrasi untuk menentukan ringkasan berdasarkan orientasi opininya dan fitur yang diekstrak dalam opini tersebut.

- *Opinion mining* from *comparative sentence*.

  Merupakan topik dalam *opinion mining* yang berkonsentrasi untuk menentukan orientasi opini dalam kalimat yang bersifat membandingkan sesuatu dengan sesuatu yang lainnya.

Di berbagai referensi ada beberapa peneliti yang menganalisis opini dan *review* untuk mendapatkan informasi milik pengguna. Pada *opinion mining*, beberapa pendekatan menggunakan teknik *artificial intelligence* dan masalah pada *text mining* adalah mengidentifikasi *rating* milik konsumen atau opini (positif atau negatif) untuk sebuah produk *review*. *Sentiment analysis* fokus pada ekstraksi dari relevansi *feature* produk berdasarkan pada sentiment dari review consumer. NLP dan *supervised*, teknik *unsupervised learning* , *association rule* telah digunakan pada *sentiment analysis*. *Sentiment classification* mempercayakan klasifikasi dari *review* berdasarkan pada sifat berlawanan posistif dan negatif. *Text mining* dan *mutual information* digunakan pada *sentiment classification*. Semua pendekatan menganalisis review untuk mengekstrak feature produk dam mengklasifikasi opini tetapi mereka tidak menangkap konteks yang diekspresikan *review*. Untuk itu peneliti menggunakan *text mining tools* untuk mendapatkan rules klasifikasi yang mengidentifikasi kalimat secara kontekstual yang berisi kontekstual informasi dalam sebuah *review*[2].

2.4.2 Metode Feature Selection

Feature Selection lazim dan mudah untuk menyelesaikan masalah mengkategorikan *text*. Adapun metode yang digunakan peneliti untuk memetakan *review* ke ontology dengan metode **term strength**[14].

Di beberapa kasus, metode term strength *s(t)* dapat didefinisikan oleh sepasang angka sample yang acak dengan dua kategori yang berelasi dengan dokumen seperti:

$$S(t) = \frac{\text{Jumlah angka pada satu kategori}}{\text{Jumlah angka pada kedua ketegori}} \qquad \textbf{(1)}$$

Persamaan (1) diatas tidak jauh berbeda dengan rumus *precision* atau juga *proportion*. Hasil output TMSK dan RIKTEXT tools nantinya akan dicocokkan dengan kamus yang berisi kata-kata terkait domain *smartphone*. Kata-kata tersebut berisi tentang *specification* dari sebuah produk. Sehingga didapat nilai proporsi berdasarkan persamaan (1) didapat bahwa :

$$Proportion = \frac{\text{jumlah good pada kamus specification battery}}{\text{Jumlah bad dan good pada kamus specification battery}} \qquad \textbf{(2)}$$

2.5 Ontology

2.5.1 Definisi Ontology

Ontology adalah sebuah representasi eksplisit dari konsep terhadap domain yang menarik dengan karakteristik dan relasinya[3]. *Ontology* biasanya diekspresikan dalam bahasa *logic-based*, oleh karena itu mendetail dan perbedaan makna dapat dibuat berdasarkan

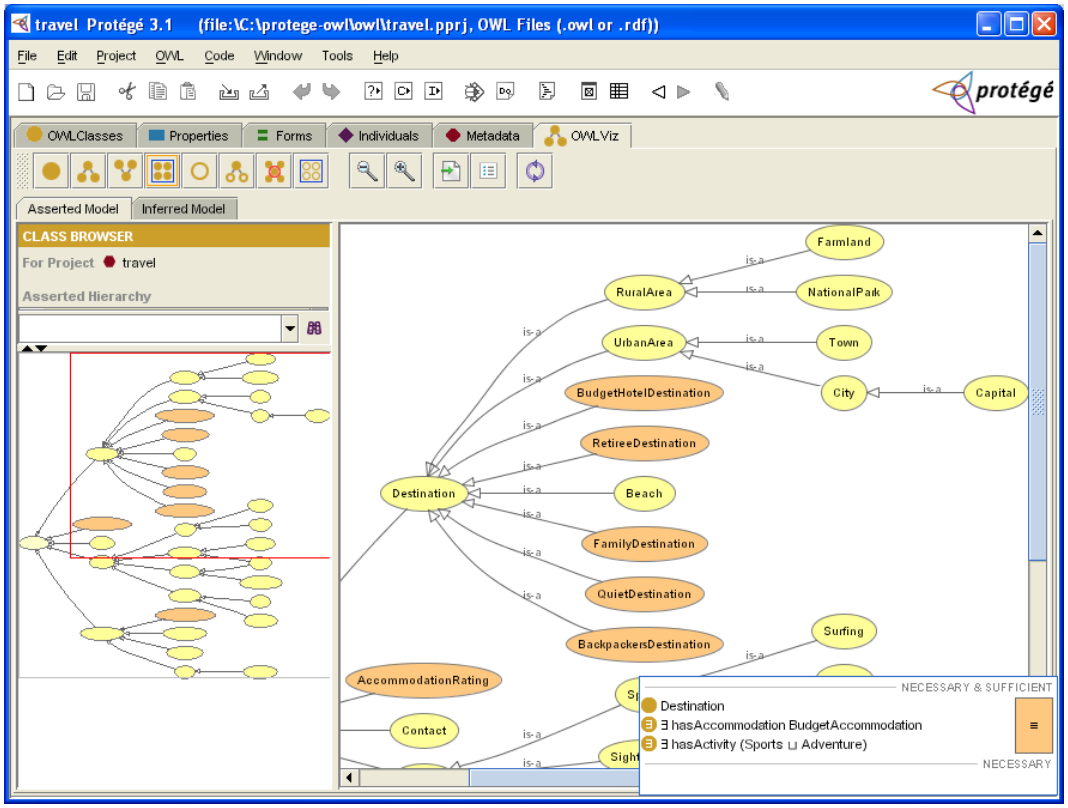
classess,properties, dan relations. *Ontology* dapat digunakan untuk meningkatkan komunikasi dua belah pihak antara manusia dan komputer. Tiga kegunaan dari *ontology* (Jasper & Uschold,1999) adalah :

- Untuk membantu komunikasi antara manusia.
- Untuk mencapai interoperability dan komunikasi diantara sistem software
- Untuk meningkatkan rancangan dan kualitas dari sistem software.

Pengertian tentang *ontology* memiliki banyak pengertian seperti yang dijelaskan pada berbagai sumber, termasuk yang dikemukakakn oleh beberapa ilmuwan. Neches dan rekannya memberikan definisi awal tentang *ontology* yaitu "sebuah *ontology* merupakan definisi dari pengertian dasar dan relasi *vocabulary* dari sebuah are sebagaimana dari kombinasi istilah dan relasi untuk mendefinisikan *vocabulary*". Gruber mendefinisikan yang sering digunakan oleh beberapa orang, definisi ersebut adalah "*Ontology* merupakan sebuah spesifikasi eksplisit dari konseptualisme".Sedangkan Barnaras pada proyek KACTUS memberikan definisi ontologi yang berdasarkan pada pengembangan ontologi. Definisi yang diberikan adalah "Sebuah ontologi memberikan pengertian untuk penjelasan secara eksplisit dari konsep terhadap representasi pengetahuan pada sebuah *knowledge base*"[7].

Ontology juga dapat didefinisikan sebagai graf bermakna yang berelasi satu sama lain antar domain vertex nya sehingga dapat menerjemahkan berdasarkan domain yang dituju. Dari definisi diatas penulis memakai ontologi sebagai representasi pengetahuan dari sebuah produk yang nantinya akan ditambahkan ekstraksi *feature review* dari pengguna.

Pada gambar 2 merupakan contoh dari *ontology* sebagai berikut.



Gambar 2 . Contoh Ontology dalam protégé

2.5.2 Komponen Ontology

Ontology menggunakan berbagai macam variasi stuktur tergantung dari penggunaan bahasa *ontology* termasuk sintaksis yang digunakan. *Ontology* bergantung kepada aplikasi yang digunakan, tidak hanya dari data yang terdapat dalam *ontology*. *Ontology* tidak melakukan fungsi perhitungan atau fungsi lainnya yang memproses *ontology*. *Ontology* hanya memberikan komponen kepada node dari graf suatu jaringan semantik. Seperti yang sudah disebutkan sebelumnya ontologi memiliki komponen sebagi berikut:

1. Entitas (*Entity*)

   Entity (juga dikenal sebagai *classes*, *objects* dan *categories*) menjelaskan konsep-konsep suatu domain. Sebuah entitas terdiri dari obyek-obyek yang merupakan penjelasan dari tugas, fungsi, aksi, strategi, dan sebagainya. Sebuah kelas juga bisa memiliki subkelas yang akan mempresentasikan konsep yang lebih spesifik daripada superkelasnya

2. Atribut (*Attribute*)

   Atribut adalah komponen dasar dari suatu obyek ontologi. Atribut menyatakan obyek-obyek dalam suatu domain yang diteliti yang digunakan untuk merepresentasikan elemen nyata seperti hewan, tanaman, dan manusia, maupun elemen abstrak seperti bilangan dan huruf.

3. Hubungan (*Relation*)

   Merupakan representasi sebuah tipe dari interaksi antara konsep dari sebuah domain. Secara formal dapat didefinisikan sebagai *subset* dari sebuah produk dari n *set*, R:C1 x C2 X . . . x Cn. Sebagai contoh dari relasi binari termasuk *subclass-of* dan *connected-to*. Relasi harus mampu mendefinisikan hubungan dari entitas yang ada.

4. Aksioma (*Axioms*)

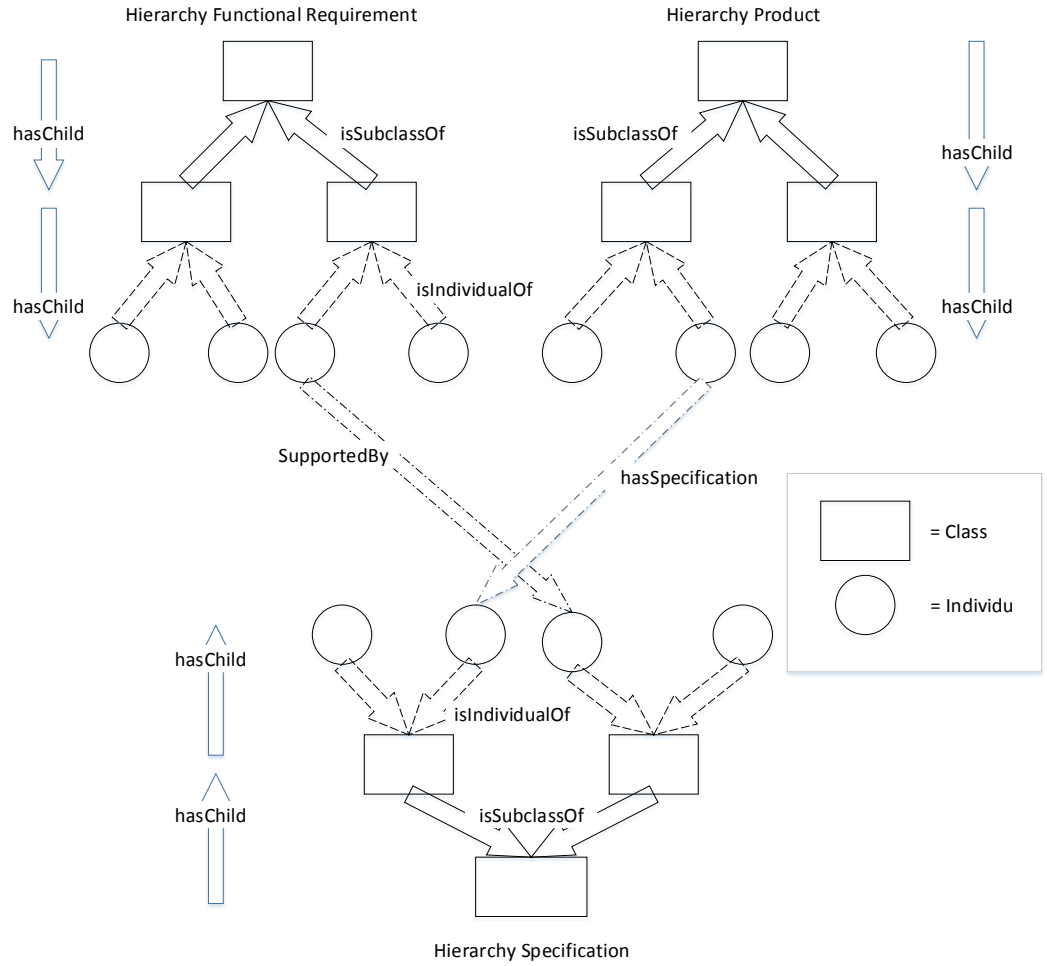
   Digunakan untuk memodelkan sebuah kalimat yang selalu benar.

2.5.3 Stuktur Model Ontology

Berdasarkan [12], suatu model ontology terdiri dari tiga kelas hirarki, yaitu kelas kebutuhan fungsional, produk, dan spesifikasi. Ketiga kelas tersebut masing-masing memiliki *subclasses* dan individu yang merepresentasikan sebuah hirarki.

Setiap kelas hirarki tersebut memiliki struktur *rooted tree* yang *node*-nya merupakan *subclasses* dari masing-masing kelas hirarki dan daunnya merupakan individu. Individu di sini berperan sebagai titik relasi antara satu kelas hirarki dengan kelas hirarki lainnya. Dalam struktur model ontology yang dijabarkan dalam [12], terdapat dua relasi, yaitu relasi antar kelas hirarki dan relasi dalam kelas hirarki. Relasi antar kelas hirarki menghubungkan dua individu antar kelas hiraki yang terdiri atas relasi *SuppBy* dan *HasSpec*. Sedangkan relasi dalam kelas hirarki menghubungkan dua *node* yang merupakan *subclasses* dalam masing-masing kelas hirarki yang terdiri atas relasi *HasChild*.

Pada gambar 3 dideskripsikan tentang struktur *ontology*



Gambar 3 . Struktur Ontology

2.6 Rekomendasi Produk

2.6.1 Hard Constraint dan Soft Constraint

Menurut [12], preferensi seorang pengguna dapat dibagi ke dalam dua kategori, yaitu *hard constraint* dan *soft constraint*. *Hard constraint* merupakan batasan preferensi yang dianggap pengguna sebagai kebutuhan mutlak yang harus dipenuhi. Sehingga produk-produk yang mendukung penuh terhadap suatu preferensi akan dikeluarkan dari daftar rekomendasi produk. Dalam penelitian ini beberapa preferensi yang dapat diset sebagai hard contraint adalah properti produk (harga, brand, sistem operasi) dan tipe produk.

Sementara *soft contraint* merupakan batasan suatu preferensi yang dianggap pengguna sebagai kebutuhan tambahan. Sehingga produk-produk yang tidak mendukung penuh terhadap suatu preferensi akan tetap dimasukkan ke daftar rekomendasi produk. Dalam penelitian ini preferensi yang dapat diset sebagai *soft constraint* hanya kebutuhan fungsional.

Pencarian produk berdasarkan *hard constraint* dan *soft constraint* menggunakan sebuah *query*. *Query* yang dilakukan memanfaatkan *semantic reasoning*, yaitu dengan cara menelusuri

nodes-nodes yang terkait berdasarkan relasi semantik pada *ontology*. Bahasa query yang digunakan adalah SPARQL karena support untuk sebuah *ontology* berformat .owl dan .rdf.

Menurut [12] dijelaskan bahwa untuk menetapkan suatu produk layak atau tidaknya untuk direkomendasikan terhadap suatu individu kebutuhan fungsional dapat mengacu pada persamaan (3) dibawah ini.

$$Parents(SpecF(f)) \oplus Parents(SpecF(f) \cap SpecP(p)) = \emptyset \qquad \textbf{(3)}$$

Dengan :

f = individu dari *functional requirement*

p = individu dari produk

SpecF(f) = himpunan dari komponen yang mendukung individu *functional requirment* f

SpecP (p) = himpunan dari komponen yang dimiliki individu produk p

Parents(Spec) = himpunan dari level atas dari hirarki suatu himpunan spesifikasi ada *ontology*.

2.6.2 Degree of Interest (DOI)

Berdasarkan [12], Derajat ketertarikan *(degree of interest* disingkat *DOI)* adalah sebuah nilai yang merepresentasikan tingkat ketertarikan pengguna terhadap sesuatu, dalam hal ini adalah *soft constraint* atau kebutuhan fungsional yang ada.

DOI bisa didapatkan dengan menanyakan langsung kepada pengguna saat berinteraksi dengan *system*. Cara ini cukup akurat karena *user* sendiri yang langsung memasukan *DOI*-nya.

DOI bisa didapatkan dengan menormalisasi dari nilai yang didapat oleh setiap kebutuhan fungsional. DOI didapt dari persamaan *(4)* dibawah ini:

$$DOI_i = \frac{i}{\sum_{i=1}^{n} i} \qquad \textbf{(4)}$$

Dengan, i = jumlah functional requirement sesuai preferensi pengguna

n = jumlah keseluruhan functional requirement

Nilai *DOI* ini nantinya digunakan dalam perhitungan bersama dengan *utility function* pada MAUT sebagai pertimbangan *system* dalam perankingan produk yang ada.

2.6.3 Multi Attribut Utility Theory (MAUT)

Berdasarkan [12][17], Multi Attribut Utility Theory (MAUT) adalah sebuah skema evaluasi yang sangat populer untuk mengevaluasi produk. Model preferensi berdasarkan metode MAUT menurut Chen dan Pu (2012) sebagai berikut :

$$U(\langle a_1, a_2, ..., a_n \rangle) = \sum_{i=1}^{n} w_i V_i(a_i) \text{ Dengan } \sum_{i=1}^{n} w_i = 1 \quad \textbf{(5)}$$

Dengan *U* adalah *utility function* dari tiap produk $(\langle a_1, a_2, ..., a_n \rangle)$, V_i merupakan *value function* untuk setiap atribut a_i dan w_i adalah *relative importance* dari a_i . *Utility function* digunakan untuk memberikan score kepada produk sesuai tingkat dukungannya terhadap suatu *functional requirement*. Adapun parameter yang digunakan untuk memberikan suatu *utility score* terhadap suatu produk adalah *supporting range, relative importance, opinion range , opinion relative importance,* dan DOI.

Supporting range adalah nilai *support level* terhadap spesifikasi yang dimiliki oleh suatu produk. Jika produk A memiliki komponen *Low End Camera,* sedangkan produk B memiliki *High End Network Speed,* bisa dipastikan *supporting range* milik produk B akan lebih tinggi dari produk B setelah dilakukan normalisasi.

Relative Importance adalah nilai keterkaitan suatu spesifikasi terhadap suatu *functional requirement*. Misalnya salah satu *functional requirement* yaitu *Selfie* membutuhkan spesifikasi Kamera Depan dan RAM, dapat dipastikan bahwa keterkaitan Kamera Depan akan lebih besar pengaruhnya daripada RAM untuk mendukung *functional requirement Selfie.* Maka nilai *relative importance* Kamera Depan akan lebih tinggi daripada RAM.

Opinion Range adalah nilai proporsi opini dari hasil ekstraksi *feature review* pengguna terhadap spesifikasi yang dimiliki oleh produk. Contohnya produk A memiliki spesifikasi *Opinion Low End Camera* sedangkan produk B memiliki *Opinion High End RAM,* dapat diartikan bahwa *opinion range* milik produk B lebih besar daripada produk A setelah dilakukan normalisasi.

Opinion Relative Importance adalah nilai keterkaitan suatu nilai proporsi opini spesifikasi terhadap *functional requirement.* Misalnya salah satu *functional requirement* yaitu *Selfie* membutuhkan spesifikasi Kamera Depan dan RAM, dapat dipastikan bahwa keterkaitan nilai proporsi Kamera Depan akan lebih besar pengaruhnya daripada RAM untuk mendukung *functional requirement Selfie.* Maka nilai *opinion relative importance* Kamera Depan akan lebih tinggi daripada RAM.

Hasil pengembangan persamaan (3) didapat *utility function* yang digunakan dalam penelitian ini adalah sebagai berikut:

$$U_i = \sum_{j=1}^{m} N(DOI_j) \times \left(\frac{\sum_{k=1}^{n} SR(C_k) \times RI(C_k) x \sum_{k=1}^{n} OR(C_k) \times OI(C_k)}{2} \right)$$

Dengan $0 < U_i \leq 1$ ***(6)***

Dengan :

U_i = nilai *Utility* dari produk ke-i

m = Jumlah *functional requirement* yang berelasi dengan produk tersebut

n = Jumlah *path* dari suatu produk ke suatu *functional requirement* ke j

C_K = Komponen pada *path* ke-k

SR = *Supporting Range*, dimana jarak dari Low,Middle Low, Middle, dst. Adalah 1 yang kemudian dilakukan normalisasi untuk digunakan sebagai *Supporting Range*

RI = *Relative Importance*, tingkat dukungan komponen terhadap suatu *functional requirement*

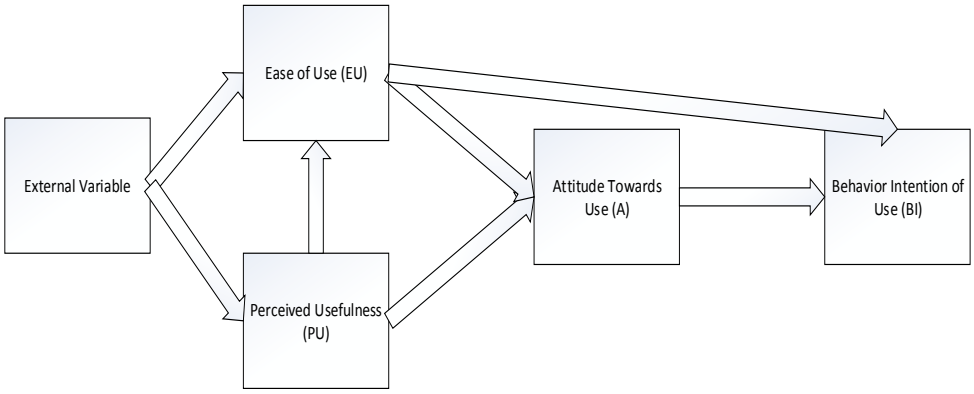
OR = *Opinion Range*, dimana jarak dari Low,Middle Low, Middle, dst. Adalah 1 yang kemudian dilakukan normalisasi untuk digunakan sebagai *Opinion Range*

OI = *Opinion Importance*, tingkat dukungan komponen terhadap suatu *functional requirement* ditambah hasil text mining

$N(DOI_j)$ = *Degree of Interest user* terhadap suatu *functional requirement* ke-*j* yang telah dinormalisasi

2.7 Technology Acceptance Model

Technology Acceptance Model (TAM), diperkenalkan pertama kali oleh Davis pada tahun 1989. TAM dibuat khusus untuk pemodelan adopsi pengguna sistem informasi. Menurut Davis (1989), tujuan utama TAM adalah untuk mendirikan dasar penelusuran pengaruh faktor eksternal terhadap kepercayaan, sikap (personalisasi), dan tujuan pengguna komputer[15]. TAM menganggap bahwa dua keyakinan variabel perilaku utama dalam mengadopsi sistem informasi, yaitu persepsi pengguna terhadap manfaat (*perceived usefulness*) dan persepsi pengguna terhadap penggunaan (*perceived ease of use*). *Perceived usefulness* diartikan sebagai tingkat di mana seseorang percaya bahwa menggunakan system tertentu dapat meningkatkan kinerjanya, dan *perceived ease of use* diartikan sebagai tingkat dimana seseorang percaya bahwa menggunakan system tidak diperlukan usaha apapun (*free of effort*). *perceived ease of use* juga berpengaruh pada *perceived usefulness* yang dapat diartikan bahwa jika seseorang merasa system tersebut mudah digunakan maka system tersebut berguna bagi mereka.



Gambar 4 . Model TAM

Berdasarkan model pada gambar 4 diatas, dapat dijelaskan bahwa *Behavioural Intention* (BI) mendefinisikan niat dan perilaku *user* dalam penggunaan *recommender system* dan menentukan bahwa teknologi itu dapat diterima. *Attitude towards Use* (A) menentukan sikap penggunaan *user* yang dapat mempengaruhi perilaku *user* atau *Behavioural Intention* (BI) , *Perceived usefulness* (PU) atau perasaan tentang kegunaan sebuah teknologi juga berefek pada perilaku *user* atau *Behavioural Intention* (BI). BI secara tidak langsung juga akibat dari perasaan kemudahan *user* atau *Perceived Ease of Use*(EU). *Attitude* (A) juga secara langsung berimplikasi dengan kedua PU dan EU, ketika PU secara langsung mempengaruhi EU. Selanjutnya *External Variables* seperti umur, jenis kelamin dll juga mempengaruhi EU dan PU yang jadi mediasi ke A. TAM menggunakan *Structural Equation Model* (SEM) untuk menghitung dan menganalisis hasil model yang dibentuk oleh TAM.

2.7.1 Structural Equation Model

SEM adalah gabungan dari analisis faktor dan regresi. Pada tahun 1950-an, SEM sudah mulai dikemukakan oleh para ahli statistik yang mencari metode untuk membuat model yang dapat menjelaskan hubungan diantara variabel-variabel. Persoalan timbul karena banyak variabel yang termasuk variabel laten yang menimbulkan kesulitan tersendiri dalam pengukurannya[15]. Dalam ilmu sosial banyak variabel, seperti motivasi seseorang, komitmen, kesetiaan pelanggan dan lainnya, yang dikategorikan sebagai *latent variable*. Sebagai contoh, peneliti tidak dapat begitu saja mengukur komitmen seseorang, karena komitmen adalah sesuatu yang kompleks, berbeda dengan frekuensi pembelian barang per minggu atau keinginan membeli barang. Untuk mengetahui komitmen seseorang, peneliti harus mengukur dengan sejumlah rincian lanjutan yang disebut dengan indikator ,seperti kepastian bertindak, keinginan mengulang tindakan, kepastian menolak alternative lain, dan sebagainya. Secara

ringkas *latent variable* adalah variabel yang mengharuskan adanya sejumlah indikator agar variabel laten tersebut dapat diukur. Tanpa sejumlah indikator maka *variabel latent* tidak dapat diukur begitu saja.

Informasi dibawah menjelaskan tahapan-tahapan yang dilakukan dalam pengujian model dengan analisis SEM.

- Perhatikan jumlah *variabel latent* dan hubungan antar *variabel latent*. Membuat dan menamai *variabel latent* adalah langkah awal yang biasa dilakukan untuk membuat sebuah model.

- Setelah *variabel latent* dibuat, selanjutnya dilakukan pemberian nama pada setiap variabel laten.

- Setelah itu harus diingat bahwa sebuah *variabel latent* harus mempunyai dua atau lebih indikator

- Setelah indikator dibuat, selanjutnya dilakukan pemberian nama pada setiap variabel manifest. Setelah semua *variabel latent* dan indikator tersusun gunakan tanda panah satu arah atau dua arah untuk menandai hubungan antar *variabel latent* atau indikator.