BAB II

TINJAUAN PUSTAKA

2.1 Data Mining

Menurut Larose (2005) *data mining* adalah proses menemukan korelasi-korelasi penuh arti, pola-pola dan *trend* dengan penyaringan melalui sejumlah data yang besar pada tempat penyimpanan, dan menggunakan teknologi pengenalan pola seperti yang terdapat pada teknik-teknik statistika dan matematika. *Data mining* dapat dibagi menjadi beberapa metode (Larose, 2005), di antaranya:

- *Classification*

  Classification adalah tindakan untuk memberikan kelompok pada setiap keadaan. Setiap keadaan berisi sekelompok atribut, salah satunya adalah *class attribute*. Metode ini digunakan untuk menemukan sebuah model yang dapat menjelaskan *class attribute* itu sebagai fungsi dari *input attribute*. *Classification* sering disebut *supervised learning* karena kelas-kelasnya sudah ditentukan sebelum menguji data.

- *Clustering*

  Clustering juga disebut sebagai *segmentation*. Metode ini digunakan untuk mengidentifikasi kelompok alami dari sebuah kasus yang didasarkan pada sebuah kelompok atribut, mengelompokkan data yang memiliki kemiripan atribut, dan mencocokkan profil *user* dengan informasi tentang suatu *item* yang ada di *database*.

- *Association*

  Association juga disebut sebagai *Market Basket Analysis*. Sebuah *problem* bisnis yang khas adalah menganalisa tabel transaksi penjualan dan mengidentifikasi produk-produk yang seringkali dibeli bersamaan oleh *customer*. Metode *association* bertujuan untuk mencari produk apa yang biasanya terjual bersamaan dan mencari tahu apa aturan yang menyebabkan kesamaan tersebut.

- *Regression*

  Metode *Regression* mirip dengan metode *Classification*. Perbedaannya adalah metode *regression* tidak bisa mencari pola yang dijabarkan sebagai *class*. Metode ini bertujuan untuk mencari pola dan menentukan sebuah nilai numerik.

- *Prediction*

  Prediksi dapat dilihat sebagai sebuah jenis klasifikasi. Perbedaannya dengan klasifikasi adalah bahwa prediksi memprediksi nilai keadaan di masa depan bukan keadaan saat ini. Contoh aplikasi prediksi adalah *machine learning*.

- *Sequence Discovery*

  Metode ini digunakan untuk menemukan pola urutan yang ada dalam data. Pola ini didasarkan pada pola urutan waktu beberapa kejadian. Pola ini mirip dengan *association* antar data yang saling berhubungan. Perbedaannya dengan *association* adalah hubungannya didasarkan pada waktu.

- *Time Series Analysis*

  Dengan analisa waktu berlanjut, nilai dari sebuah atribut diuji sepanjang waktu. Nilai-nilai tersebut biasanya akan diambil dalam jangka waktu tertentu. Analisa

ini biasanya digunakan untuk menilai kemiripan antara beberapa atribut, menilai sifat atribut, dan juga memprediksi nilai di masa yang akan datang.

2.2 Sistem Rekomendasi

Sistem rekomendasi merupakan sebuah alat personalisasi yang menyediakan pengguna sebuah informasi daftar *item-item* yang sesuai dengan keinginan atau ketertarikan masing-masing pengguna (Sebastia, dkk, 2009). Sistem rekomendasi memberikan kesimpulan dengan menganalisis informasi tentang pengguna dan menyaring informasi personal, sehingga hanya informasi yang sesuai dengan kebutuhan akan ditampilkan di sistem dengan menggunakan model rekomendasi.

Menurut McGinty dan Smyth (2006), sistem rekomendasi merupakan model aplikasi dari hasil observasi terhadap keadaan dan keinginan pelanggan. Oleh karena itu sistem rekomendasi memerlukan model rekomendasi yang tepat agar yang direkomendasikan sesuai dengan keinginan pelanggan, serta mempermudah pelanggan mengambil keputusan yang tepat.

Terdapat dua algoritma utama untuk merekomendasikan *item* kepada *user* (Sarwar, dkk, 2001), yaitu

- *Content-based Filtering*

  Algoritma ini merupakan hasil penyaringan informasi dalam sistem berbasis konten (Bogers dan Van den Bosch, 2007). Sistem rekomendasi berbasis konten dimulai dengan memahami profil *user*, preferensi dan kendala jika ada. Informasi ini digabungkan dengan *log* dari interaksi *user* sebelumnya (jika ada) untuk membangun profil *user* (Sharda N., 2007). Kemudian sistem rekomendasi mencocokkan profil *user* dengan informasi tentang suatu *item* yang ada di

database. Ada dua masalah pada algoritma *Content-based Filtering*. Masalah pertama adalah tidak semua domain dapat mengimplementasikan algoritma *Content-based Filtering* karena membutuhkan teknik *text-filtering*. Domain-domain yang menggunakan *item* berupa musik atau video tidak dapat mengimplementasikan algoritma ini. Masalah kedua adalah apabila seorang *user* jarang meng-*update* profilnya, maka rekomendasi yang diberikan hanya sebatas kriteria profilnya tersebut. Dengan kata lain, sistem tidak bisa memunculkan rekomendasi yang baru. Hal ini disebut dengan *over-specialization*.

- *Collaborative Filtering*

  Collaborative Filtering menggunakan penilaian dari *user* sebelumnya untuk meningkatkan kualitas materi yang direpresentasikan kepada *user*. Kemiripan atau *similarity* antar *item* menjadi bagian penting dari algoritma ini (Sebastia, dkk, 2009). Hal yang paling penting dari algoritma ini adalah satu-satunya cara untuk merekomendasikan *item* yang baru, *user* harus terlebih dahulu melakukan *rating* terhadap suatu *item* (Linden, dkk, 2003).

2.3 Item-based Collaborative Filtering

Item-based Collaborative Filtering merupakan algoritma rekomendasi yang didasari atas adanya *similarity* antara pemberian *rating* terhadap suatu *item* dengan *item* yang dipilih (Sarwar, dkk, 2001). Dari *rating* yang diberikan terhadap suatu *item* dan *item* lain, didapatlah nilai *similarity* terhadap kedua *item* tersebut. Nilai *similarity* antar *item* kemudian digabungkan dengan rumus prediksi *rating*, sehingga menghasilkan prediksi *rating* yang akan dijadikan rekomendasi. Algoritma ini muncul sebagai solusi untuk masalah skalabilitas, waktu dan memori karena komputasinya dilakukan secara

offline. Pada algoritma ini, nilai *similarity* antar *item* cenderung lebih jarang berubah dibandingkan dengan nilai *similarity* antar *user*.

Langkah pertama algoritma ini adalah mencari nilai *similarity* antar *item*. Terdapat dua rumus untuk menghitung nilai *similarity* antar *item*, yaitu

- *Cosine Similarity*

  Rumus untuk menghitung *similarity*-nya sebagai berikut:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

Gambar 2.1 Rumus *Cosine Similarity* (Sarwar, dkk, 2001)

  Keterangan:

  1. A_i adalah *rating user* terhadap *item* A.
  2. B_i adalah *rating user* terhadap *item* B.

- *Adjusted Cosine Similarity*

  Mirip dengan *Cosine Similarity* hanya saja setiap *rating user* terhadap suatu *item* harus dikurangi dengan rata-rata *rating user* tersebut. Rumus untuk menghitung *similarity*-nya sebagai berikut:

$$sim(i,j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_u)^2} \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_u)^2}}.$$

Gambar 2.2 Rumus *Adjusted Cosine Similarity* (Sarwar, dkk, 2001)

  Keterangan:

  1. *Sim(i,j)* adalah nilai *similarity* antara *item i* dan *item j*.
  2. $u \in U$ adalah himpunan *user u* yang me-*rating item i* dan *item j*.

3. $R_{u,i}$ adalah *rating user u* pada *item i*.
4. $R_{u,j}$ adalah *rating user u* pada *item j*.
5. $\bar{R}_u$ adalah rata-rata *rating user u*.

Setelah mendapatkan *similarity* antar *item*, langkah selanjutnya adalah menghitung prediksi *rating* yang akan diberikan kepada *user*.

2.4 Rumus prediksi Weighted Sum

Weighted Sum adalah rumus yang digunakan untuk memberikan prediksi *rating* terhadap seorang *user* pada suatu *item* (Sarwar, dkk, 2001). Setiap *rating* yang diberikan pada *item* yang berkorelasi akan dikalikan dengan nilai *similarity*-nya. Kemudian dibagi dengan jumlah nilai absolut dari *similarity* seluruh *item* yang berkorelasi. Rumus dapat dilihat pada gambar 2.3.

$$P_{u,i} = \frac{\sum_{\text{all similar items, N}}(s_{i,N} * R_{u,N})}{\sum_{\text{all similar items, N}}(|s_{i,N}|)}$$

Gambar 2.3 Rumus Prediksi *Weighted Sum* (Sarwar, dkk, 2001)

Keterangan:

- $P_{u,i}$ adalah prediksi untuk *user u* pada *item i*.
- $S_{i,N}$ adalah nilai *similarity* antara *item i* dan *item N*.
- $R_{u,N}$ adalah *rating user u* pada *item N*.

Rating user yang digunakan pada rumus *Weigthed Sum* harus dinormalisasi menjadi skala -1 sampai 1 seperti pada Gambar 2.4 dan hasilnya dikembalikan ke skala awal dengan rumus denormalisasi seperti pada Gambar 2.5.

$$NR_{u,N} = \frac{2(R_{u,N} - Min_R) - (Max_R - Min_R)}{(Max_R - Min_R)}$$

Gambar 2.4 Rumus Normalisasi *Rating* (Sarwar, dkk, 2001)

$$R_{u,N} = \frac{1}{2}((NR_{u,N} + 1) \times (Max_R - Min_R)) + Min_R$$

Gambar 2.5 Rumus Denormalisasi *Rating* (Sarwar, dkk, 2001)

Keterangan:

- $NR_{u,n}$ adalah normalisasi *rating user u* terhadap *item N*.
- $R_{u,N}$ adalah *rating user u* pada *item N*.
- Max_R adalah skala *rating* terbesar.
- Min_R adalah skala *rating* terkecil.

Weighted Sum terbagi dua, yaitu *Item-Item Weighted Sum* dan *Item-Regression Weighted Sum*. Pada *Item-Item Weighted Sum*, *rating user* yang digunakan adalah *rating user* secara mentah atau *raw*. Pada *Item-Regression*, *rating user* yang digunakan adalah *rating user* yang telah diolah dengan menggunakan *Simple Linear Regression*. *Simple Linear Regression* dijelaskan pada Gambar 2.6.

$$Y_i = a_0 + b_0 X_i$$

Gambar 2.6 *Simple Linear Regression* (Draper, 1981)

Keterangan:

- Y_i adalah variabel *response* atau variabel akibat.
- X_i adalah variabel *predictor* atau variabel faktor penyebab.

- a_0 adalah konstanta.
- b_0 adalah koefisien regresi.

2.5 Web Ontology Language (OWL)

Web Ontology Language (OWL) adalah suatu bahasa untuk memproses informasi yang terdapat dalam *website* (M. Smith, 2008). Ontologi sendiri dapat didefinisikan sebagai suatu cara untuk mendeskripsikan arti dan relasi dari istilah-istilah. Deskripsi tersebut berisi *classes*, *properties*, dan *instances*. Deskripsi ini dapat membantu sistem komputer dalam menggunakan istilah-istilah tersebut dengan cara yang lebih mudah (Bekke, 1992).

Penggunaan OWL dapat menambah *vocabulary* tambahan disamping semantik formal yang telah dibuat sebelumnya menggunakan XML dan RDF. Hal ini sangat membantu penginterpretasian mesin yang lebih baik terhadap isi Web (Wicaksana, 2006).

Menurut Sachs (2006) terdapat tiga alasan untuk menggunakan ontologi, yaitu

- Menjelaskan suatu domain secara eksplisit.

  Memberikan struktur hierarki dari istilah untuk menjelaskan sebuah domain dan bagaimana mereka berhubungan.
- Berbagi pemahaman dari informasi yang terstruktur.

  Perangkat lunak dapat menggunakan kumpulan informasi untuk menjawab permintaan *user* atau sebagai data *input* untuk aplikasi yang lain.
- Penggunaan ulang domain pengetahuan.

  Dengan digunakannya ontologi, suatu sistem dapat mengintegrasikan dengan beberapa ontologi yang sudah ada.

2.6 Ontologi Movie

Ontologi *movie* adalah ontologi yang digunakan untuk mendapatkan informasi tambahan pada suatu film seperti informasi personal aktor, nama karakter yang dimainkan oleh aktor tersebut, dan lain-lain. Ontologi ini dibangun dengan menggabungkan ontologi yang ada pada *dbpedia.org* dan *linkedmdb*.org. Beberapa *class* yang didefinisikan pada ontologi *movie* sebagai berikut :

- dbpedia-owl:birthDate untuk mendefinisikan tanggal lahir dari seorang aktor.
- dbpedia-owl:birthName untuk mendefinisikan nama asli berdasarkan akte kelahiran dari seorang aktor.
- movie:performance untuk mendefinisikan film apa saja yang diperankan oleh aktor tersebut dengan *identifier* alamat URI.
- movie:performance_character untuk mendefinisikan nama karakter yang dimainkan oleh aktor dalam suatu film.

2.7 Resource Development Framework (RDF)

Berdasarkan World Wide Web Consortium, Resource Description Framework (RDF) adalah bahasa untuk merepresentasikan informasi mengenai *resource* yang ada dalam World Wide Web (Bekke, 1992). RDF merupakan suatu metadata yang digunakan untuk mendeskripsikan alamat sumber daya pada web. Metadata ini dapat berupa judul, pengarang, hak cipta, dan lisensi dalam dokumen web. Elemen pernyataan dalam RDF terdiri dari subyek, predikat, dan obyek. Subyek dan predikat berisikan data yang berisikan sumber daya, sedangkan obyek dapat bertipe sumber daya maupun literal (Wicaksana, 2006).

2.8 Simple Protocol and RDF Query Language (SPARQL)

Simple Protocol and RDF Query Language (SPARQL) adalah sebuah bahasa untuk mengekspresikan *query* untuk mengakses data pada RDF dan OWL. Hasil *query* dari SPARQL dapat mengembalikan data dalam format XML, JSON, RDF, dan HTML. Berikut adalah contoh implementasi dari SPARQL:

Data:

```
<rdf:Description rdf:about="http://dig.csail.mit.edu/2008/webdav/timbl/foaf.rdf#ss">
    <foaf:name>Susie Stephens</foaf:name>
    <foaf:organization rdf:resource="http://dbpedia.org/resource/Eli_Lilly_and_Company"/>
</rdf:Description>
<rdf:Description rdf:about="http://bblfish.net/people/henry/card#me">
   <foaf:name>Henry Story</foaf:name>
```

Gambar 2.7 Potongan Data Awal
(Sumber: http://dig.csail.mit.edu/2008/webdav/timbl/foaf.rdf)

Query:

```
PREFIX foaf:  <http://xmlns.com/foaf/0.1/>
SELECT *
WHERE {
    ?person foaf:name ?name .
    ?person foaf:mbox ?email .
}
```

Gambar 2.8 Contoh SPARQL *Query*
(Sumber: http://www.cambridgesemantics.com/sparql-by-example/slides.html#q2)

Hasil:

person	name	email
<http://www.w3.org/People/karl/karl-foaf.xrdf#me>	"Karl Dubost"	<mailto:karl@w3.org>
<http://www.w3.org/People/Berners-Lee/card#amy>	"Amy van der Hiel"	<mailto:amy@w3.org>
<http://www.w3.org/People/Berners-Lee/card#edd>	"Edd Dumbill"	<mailto:edd@xmlhack.com>
<http://www.w3.org/People/Berners-Lee/card#dj>	"Dean Jackson"	<mailto:dean@w3.org>
<http://www.w3.org/People/Berners-Lee/card#edd>	"Edd Dumbill"	<mailto:edd@usefulinc.com>
<http://www.aaronsw.com/about.xrdf#aaronsw>	"Aaron Swartz"	<mailto:me@aaronsw.com>
<http://www.w3.org/People/Berners-Lee/card#i>	"Timothy Berners-Lee"	<mailto:timbl@w3.org>
<http://www.w3.org/People/EM/contact#me>	"Eric Miller"	<mailto:em@w3.org>
<http://www.w3.org/People/Berners-Lee/card#edd>	"Edd Dumbill"	<mailto:edd@xml.com>
<http://www.w3.org/People/Berners-Lee/card#dj>	"Dean Jackson"	<mailto:dino@grorg.org>
<http://www.w3.org/People/Berners-Lee/card#libby>	"Libby Miller"	<mailto:libby.miller@bristol.ac.uk>
<http://www.w3.org/People/Connolly/#me>	"Dan Connolly"	<mailto:connolly@w3.org>

Gambar 2.9 Hasil SPARQL *Query*
(Sumber: http://www.cambridgesemantics.com/sparql-by-example/slides.html#q2r)

2.9 Internet Movie Database (IMDB)

Internet Movie Database (IMDB) adalah *database online* yang berisi informasi tentang film, program televisi, aktor, biografi dan lain-lain. IMDB dapat diakses melalui situs http://www.imdb.com. Aktor dan *crew* dapat mem*posting* informasi tentang dirinya agar dapat diakses dan dilihat oleh *user* yang mengunjungi situs tersebut. Untuk dapat mengakses informasi mengenai suatu film, IMDB menyediakan *copy dataset* yang dapat diakses pada situs http://www.imdb.com/interfaces. *Dataset* diberikan dalam bentuk *plain text* dengan format total *user* yang melakukan *rating*, *rating* film, judul film dan tahun film. Salah satu contohnya sebagai berikut:

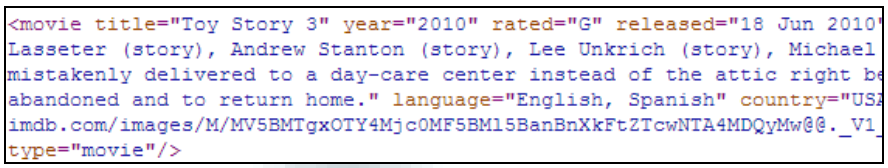
```
79294 9.1 "Avatar: The Last Airbender" (2005)
147399 9.6 "Band of Brothers" (2001)
93051 8.8 "Battlestar Galactica" (2004)
94249 8.7 "Boardwalk Empire" (2010)
78573 8.1 "Bones" (2005)
```

Gambar 2.10 Contoh *Dataset* Movie IMDB

Hasil *copy dataset* yang didapat dari http://www.imdb.com/interfaces tidak mengikutsertakan *identifier* (imdbid), *genre*, aktor, abstraksi dan *poster* gambar dari film IMDB tersebut. Oleh karena itu, digunakan *Application Program Interface* (API) tambahan, yaitu Online Movie Database API (OMDBAPI). OMDBAPI dapat menambahkan informasi film dari situs IMDB dan Rotten Tomatoes. API ini dapat diakses pada situs http://www.omdbapi.com. Contoh untuk mendapatkan informasi tentang film “Toy Story 3” menggunakan OMDBAPI sebagai berikut:

URL: http://www.omdbapi.com/?i=&t=Toy+Story+3.

Hasil dalam bentuk XML:



Gambar 2.11 Hasil OMDBAPI dalam bentuk XML

2.10 DBPedia

DBPedia merupakan *open source* data berbasis web dalam bentuk RDF *triples* berbasis data dari Wikipedia. Tujuan utama dari *knowledge* tersebut adalah memungkinkan pengguna untuk menggunakan data lengkap Wikipedia dan menggunakan data antar *website*. DBPedia mendeskripsikan lebih dari 3,5 juta benda yang dibagi ke dalam beberapa kategori, misalnya *people*, *places*, *music albums*, *films*, *video games*, *organization*, *species*, dan *disease* (Harbers, 2011). Untuk mengakses data dalam DBpedia, digunakan *query* SPARQL yang akan mengembalikan data dalam bentuk XML atau bentuk lainnya. *End point* atau titik akhir untuk mengakses data dari DBPedia adalah http://dbpedia.org/sparql. Informasi yang diambil dari DBPedia adalah tanggal lahir aktor, nama asli aktor dan foto aktor tersebut.

2.11 Linked Movie Database (LinkedMDB)

Linked Movie Database (LinkedMDB) adalah *open source* data berbasis web tentang film dalam bentuk RDF *triples*. Basis data yang digunakan pada LinkedMDB berasal dari berbagai sumber situs film seperti IMDB, RottenTomatoes.com dan FreeBase (Oktie Hassanzadeh dan Mariano Consens, 2008). Untuk mengakses data dalam LinkedMDB, digunakan *query* SPARQL yang akan mengembalikan data dalam bentuk XML atau bentuk lainnya. *End point* dari LinkedMDB adalah

http://linkedmdb.org/sparql. Informasi yang diambil dari LinkedMDB adalah nama karakter yang diperankan oleh seorang aktor.