BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Tahun 2001, Breiman merupakan orang yang pertama kali memperkenalkan *random forest* dalam teknik *bagging*. Pada penelitiannya, Breiman menyebutkan beberapa kelebihan dari random forest diantaranya dapat memberikan hasil akhir klasifikasi dengan baik disertai dengan tingkat error yang lebih rendah, mampu mengatasi jumlah data yang banyak dengan lebih baik serta merupakan sebuah algoritma yang baik untuk mengatasi *missing data* (Breiman, 2001). Kelas klasifikasi pada random forest ditentukan dengan cara melakukan *voting* pada beberapa pohon keputusan yang telah terbentuk. Pohon keputusan dengan *vote* yang paling banyak merupakan pemenang dari penentukan kelas klasifikasi (Ho, 1995). Beberapa penelitian yang menggunakan algoritma *random forest* dan membahas permasalahan prediksi *drop out* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Penelitian terkait

No.	Peneliti	Tugas	Domain Data
1.	Gutierrez dkk (2018)	Prediksi *drop out* berdasarkan rata-rata nilai matakuliah pada tiga semester pertama menggunakan *Logistic Regression, Decision Trees, Random Forest* dan *Naive Bayes*.	Data akademik (Systems Engineering Program at a private University in Bogot´a, Colombia)
2.	Bharadwaj dan Pal (2011)	Identifikasi para siswa yang membutuhkan perhatian khusus untuk mengurangi jumlah siswa yang gagal (*drop out*).	Data dari berbagai universitas dan institusi
3.	Devasia dkk (2016)	Prediksi akademik status (*drop out*) berdasarkan variabel seperti jenis kelamin, kebiasaan siswa, tempat tinggal, jumlah keluarga,	Data akademik (Amrita School of Arts and Sciences, Mysuru)

		status keluarga, kualifikasi ayah, kualifikasi ibu, pendapatan keluarga, pekerjaan ayah, pekerjaan ibu.	
4.	Dharmawan dkk (2018)	Prediksi *drop out* berdasarkan faktor non akademik seperti demografi, interaksi sosial siswa, keuangan, motivasi belajar dan personaliti juga dapat menjadi penyebab siswa yang *drop out*.	Data siswa pada beberapa universitas yang diambil menggunakan teknik sample acak menggunakan *Edward Personal Preference Schedules* (EPSS) tes.
5.	Sukhbaatar dkk (2018)	Prediksi *drop out* menggunakan *decision tree*, berdasarkan data 717 mahasiswa dari *log blended learning* mahasiswa seperti jumlah mengikuti course, dan jumlah mengikuti ujian. Hasil akhirnya dapat memprediksi mahasiswa yang *drop out* dengan akurasi 79%.	Data akademik (Data penilaian akhir dan *log file* dari program sarjana wajib di Fakultas Teknik Universitas Nasional)
6.	Shiratori (2018)	Prediksi kandidat *drop out* dengan menggunakan *logistic regression*, pada 719 data mahasiswa berupa data identitas dan nilai tiap semester, hasil akhir berupa model probabilitas transisi normal, kandidat *drop out* dan *drop out*.	Data akademik

Algoritma klasifikasi seperti *decision tree, k-nearest neighbor, artificial neural network, naïve bayes, support vector machine* dan *ensemble method* seperti

bagging, *adaboosting*, *random forest*, *ensemble filtering* dan *voting algorithm* telah digunakan untuk membuktikan bahwa penggunaan teknik *ensemble* dapat memberikan peningkatan nilai akurasi pada pengukuran algoritma klasifikasi dan prediksi (Amrieh dkk., 2016; Rahman dkk., 2017; Kumari dkk., 2018; Pan dkk., 2016).

2.2 Dasar Teori

2.2.1 *Drop-Out*

Berdasarkan Kamus Oxford, *drop-out* adalah seseorang yang meninggalkan sekolah atau kuliah sebelum menyelesaikan masa studi. Sedangkan definisi masa studi menurut Peraturan Kementerian Riset, Teknologi dan Pendidikan Tinggi adalah lama waktu terjadwal untuk mahasiswa menyelesaikan studinya. Batas waktu studi adalah batas waktu maksimal seorang mahasiswa dalam menyelesaikan studi. Program Sarjana memiliki 144 (seratus empat puluh empat) SKS minimal atau 160 (seratus enam puluh) SKS yang dapat diselesaikan dalam rentang waktu 8 (delapan) semester atau kurang dari 8 (delapan) semester atau batas maksimal lamanya 14 (empat belas) semester. Program Diploma memiliki 108 (seratus delapan) SKS minimal atau 120 (seratus dua puluh) SKS yang dapat diselesaikan dalam rentang waktu 10 (sepuluh) semester atau kurang dari 10 (sepuluh) semester atau batas maksimal lamanya 10 (sepuluh) semester (Kemeristekdikti, 2015).

Berikut ketentuan putus studi (*drop-out*) untuk pendidikan program diploma (D3) dan sarjana (S1) :

1) Ketentuan putus studi (*drop-out*) untuk mahasiswa dengan pendidikan program Diploma (S0) yaitu :
    a) Mahasiswa tidak dapat mengumpulkan total sks sebanyak 27 sks pada akhir tahun pertama atau;
    b) Mahasiswa tidak dapat mengumpulkan total sks sejumlah ≥ 27 sks namun mencapai IPK < 2,00 di akhir tahun pertama;
    c) Mahasiswa tidak dapat mengumpulkan jumlah total sks yang dipersyaratkan (108 sks) pada batas waktu maksimal masa studi;
    d) Jumlah total SKS yang dipersyaratkan (108 sks) dapat dikumpulkan oleh mahasiswa di batas waktu maksimal masa studi, namun IPK yang dimiliki

adalah < 2,00 atau mendapatkan nilai E atau mendapatkan nilai D sejumlah > 10% dari syarat minimal total SKS;

e) Mahasiswa yang tidak dapat membayarkan biaya UKT sesuai jadwal pada masing-masing semester dan yang bersangkutan tidak pula mengajukan *stop-out* (SO).

2) Ketentuan putus studi (*drop-out*) untuk mahasiswa dengan pendidikan program Sarjana (S1) yaitu :
   a) Mahasiswa tidak dapat mengumpulkan total sks sebanyak 52 sks pada akhir tahun kedua atau;
   b) Mahasiswa tidak dapat mengumpulkan total sks sejumlah ≥ 52 sks namun mencapai IPK < 2,00 di akhir tahun kedua;
   c) Mahasiswa tidak dapat mengumpulkan jumlah total sks yang dipersyaratkan (144 sks) pada masa akhir studi maksimal (10 semester);
   d) Jumlah total SKS yang dipersyaratkan (144 sks) dapat dikumpulkan oleh mahasiswa di batas waktu maksimal masa studi, namun mendapatkan nilai E atau mendapatkan nilai D sejumlah > 10% dari syarat minimal total SKS;
   e) Mahasiswa yang tidak dapat membayarkan biaya UKT sesuai jadwal pada masing-masing semester dan yang bersangkutan tidak pula mengajukan *stop-out* (SO).

2.2.2 *Business Intelligence* (BI)

Business intelligence adalah salah satu istilah yang menggabungkan antara arsitektur, *tools*, *database*, *analytical tools*, aplikasi dan metodelogi. Perusahaan yang telah menerapakan teknologi *business intelligence* dapat mengubah data yang sebelumnya tidak memiliki nilai tambah bagi perusahaan dapat diubah menjadi informasi yang memiliki nilai tambah yang bernilai tinggi untuk menentukan strategi bisnis ke depan (Akbar dkk., 2017). Tujuan dari BI adalah menggabungkan sekumpulan sumber data yang berbeda menjadi informasi yang menjelaskan bagaimana proses yang terjadi pada perusahaan dan memberikan kembali informasi tersebut kepada level manajemen dengan cara yang tepat dan cepat (Harokova dan Skalska, 2013).

Terdapat empat komponen dasar dari *business intelligence* yang saling berhubungan agar sebuh *business intelligence* dapat berfungsi yaitu (Akbar dkk., 2017) :

1) *Data Warehouse*

   Data warehouse berfungsi sebagai sumber data pada *business intelligence*. Sebuah *data warehouse* merupakan koleksi data yang terorientasi pada subjek, tidak mengalami perubahan, serta memiliki rentang waktu yang cukup lebar yang berfungsi dalam mendukung pengambilan keputusan level manajemen.

2) *Business Analytic*

   Business Analytic adalah sekumpulan peralatan yang digunakan untuk memanipulasi, menambang dan menganalisa data yang terdapat di dalam *data warehouse*.

3) *Report and Queries*

   Termasuk didalamnya segala bentuk pelaporan baik secara statis (tidak berubah) ataupun dinamis sesuai dengan perubahan data dan setiap macam *query* yang ada.

4) *Data, text and web mining* serta peralatan matematika level atas dan statistic

   Data mining adalah proses untuk menemukan hubungan atau informasi yang tidak diketahui didalam *database* besar ataupun *data warehouse* dengan menggunakan peralatan *intelligent*.

2.2.3 *Data Mining*

Data mining merupakan sebuah tahapan dalam menemukan aturan, pola atau informasi baru pada kumpulan data yang dipilih dengan menerapkan metode atau algoritma tertentu. Metode, teknik dan algoritma pada *data mining* memiliki banyak variasi. Pemilihan teknik, algoritma atau metode yang sangat tepat bergantung kepada tujuan dan proses *mining data* secara keseluruhan (Puti dkk., 2018). Kegiatan yang dilakukan dengan menggunakan teknik statistik, matematika, kecerdasan buatan maupun *machine learning* untuk mengestraksi dan memperoleh informasi baru dari sekumpulan besar data disebut dengan *data mining* (Turban dkk., 2004).

Ada banyak istilah lain yang mempunyai arti yang serupa terhadap *data mining* adalah *knowledge extraction* (ekstrasi pengetahuan), *knowledge discovery*

in database (KDD), *business intelligence* (kecerdasan bisnis), *data pattern analysis* (analisis data/pola), *data archaeology* dan *data dredging* (Larose, 2005). *Knowledge discovery in database* adalah serangkaian proses yang terdiri dari kegiatan mengumpulkan, menggunakan data lama untuk menemukan kembali pola, keteraturan dan relasi dalam data yang berskala besar.

1) Metode Pelatihan

   Secara umum, metode pelatihan yang terdapat dalam teknik *data mining* terbagi menjadi dua jenis yaitu :

   a) *Unsupervised learning*, yaitu metode yang tidak memiliki data latih dan label sebagai target sehingga dari data yang ada dihasilkan *rule*/formula/rumus/aturan yang dijadikan acuan untuk selanjutnya mengelompokan data tersebut menjadi 2 (dua) bagian atau 3 (tiga) bagian dan seterusnya. Metode analisis *unsupervised learning* adalah *Self-Organizing Map, DBSCAN, Hierarchical Clustering, Fuzzy C-Means, K-Means*.

   b) *Supervised learning*, yaitu metode yang memiliki data latih dan label sebagai target sehingga tujuannya pada metode ini adalah untuk mengelompokkan data yang baru kedalam data yang sudah ada. Dengan model ini prediksi pada masa depan dapat dilakukan dengan menggunakan data historis yang dapat dipelajari lebih lanjut sehingga mampu memberikan hasil yang akurat dan tepat. Metode analisis untuk *supervised learning* adalah *Decision Tree, Random Forest, Artificial Neural Network, Support Vector Machine, Naive Bayes Classifier, Nearest-Neighbor Classifier, Fuzzy K-Nearest Neighbor*.

2) Pengelompokan *Data Mining*

   Data mining dikelompokkan menjadi beberapa bagian berdasarkan pembagian tugasnya, yaitu :

   a) Deskripsi : untuk mengetahui pola dan tren yang tersembunyi pada data dengan cara mendeskripsikannya.

   b) Estimasi : untuk melakukan proses penilaian terhadap sesuatu.

   c) Prediksi : untuk mengetahui kejadian/hasil yang belum dapat diketahui/belum terjadi dimasa depan.

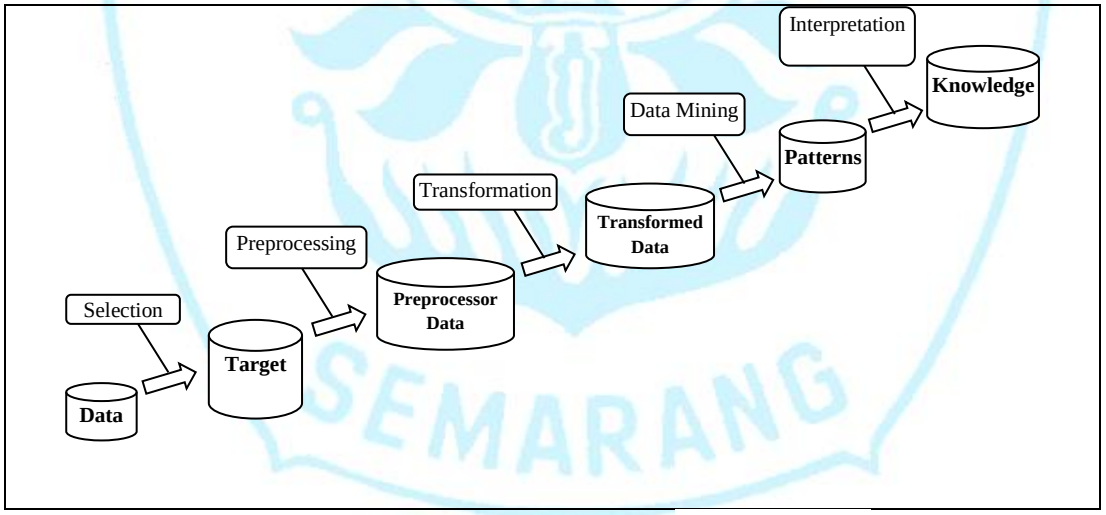
d) Klasifikasi : untuk melakukan kegiatan pengelompokkan terhadap sekumpulan data ke dalam label/kelas tertentu yang bersifat kategorik.

e) *Clustering* : untuk melakukan kegiatan pengelompokkan terhadap data/*record*/analisa/studi kasus ke dalam sebuah kelas yang memiliki kesamaan satu sama lain.

f) Asosiasi : untuk mengetahui korelasi yang terdapat pada beberapa kejadian yang terjadi dalam satu waktu.

2.2.4 Tahapan *Data Mining*

Terdapat lima proses dalam tahapan *data mining*. Kegiatan menyeleksi data dari sumber data ke data target menjadi proses pertama yang dilakukan. Proses kedua yaitu tahap preprocessing data yang berguna untuk memperbaiki kualitas dari data yang akan digunakan, proses ketiga yaitu transformasi yang berguna untuk menormalkan data, proses keempat yaitu tahapan *mining data* dan yang terakhir adalah proses interpretasi/evaluasi yang berguna untuk menghasilkan keluaran yaitu informasi baru yang dapat bermanfaat (Larose, 2005). Tahapan *data mining* secara detail pada Gambar 2.1.



Gambar 2.1 Tahapan *Data Mining* (Fayyad, 1996)

1) Data *selection* : menyeleksi data dari kumpulan data yang kemudian dijadikan sebagai dataset untuk proses *data mining*.

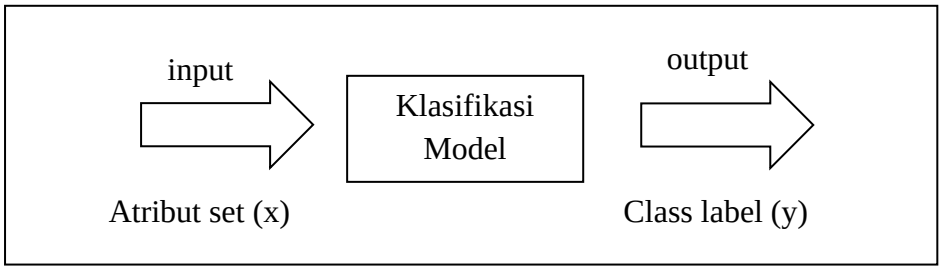
2) Pre-*processing* / *cleaning* : proses pembersihan terhadap data seperti melakukan pemeriksaan inkonsisten data, menghilangkan duplikasi data dan mengatasi permasalahan *missing data*.
3) *Transformation* : salah satu proses yang terjadi pada *transformation* adalah *coding*. Pola informasi dan jenis informasi menjadi syarat uutama dalam proses *coding*.
4) *Data mining* : proses untuk menemukan informasi, pola serta aturan baru pada kumpulan data yang dipilih berdasarkan metode, algoritma dan teknik.
5) *Interpretation* / *evaluation* : menterjemahkan hasil informasi yang didapat dari proses *data mining* sehingga dapat dimengeri oleh pihak yang membutuhkan.

2.2.5 Prediksi

Berdasarkan Kamus Besar Bahasa Indonesia (KBBI), perkiraan/prediksi/ramalan adalah proses yang dilakukan dengan menggunakan data masa lalu (historis) untuk menghasilkan pengetahuan, informasi baru dan nilai untuk kebutuhan masa depan. Hasil prediksi/ramalan/perkiraan dapat menjadi alat bantu untuk kegiatan perencanaan serta pengambilan keputusan. Adapun beberapa metode prediksi *data mining* yaitu *classification tree*, *regression*, *deviation detection* dan *naïve bayes classification*.

2.2.6 Klasifikasi

Klasifikasi merupakan kegiatan yang dilakukan guna menghasilkan model yang dapat menjelaskan perbedaan antara kelas data atau konsep data (Putri dkk., 2018). Ada beberapa teknik klasifikasi, diantaranya adalah *neural network, artificial neural network, rough sets, k-nearest neighbor, bayesian classifiers,* dan lain-lain. Ada 2 (dua) fase pada tahapan klasifikasi, yaitu *learning fase/training fase* (fase pembelajaran) dan fase klasifikasi. *Learning fase/training fase* adalah fase yang dilakukan untuk melatih data sehingga menghasilkan sebuah model/*rule*/pola/aturan. Fase klasifikasi dapat berfungsi untuk kegiatan prediksi terhadap label kelas data baru dengan menggunakan model yang telah dibangun yang dapat memberikan hasil akhir berupa akurasi dari model yang dihasilkan. Model klasifikasi dapat diilustrasikan pada Gambar 2.2.



Gambar 2.2. Model Klasifikasi (Sumarlin, 2015)

Menurut Gorunescu (2011), proses klasifikasi didasarkan pada empat komponen dasar sebagai berikut :

1) Kelas adalah *dependent variabel* dalam bentuk *categorical variabel model* yang mewakili label pada objek setelah proses klasifikasi. Contohnya adalah tipe gempa, loyalitas pelanggan dan sebagainya;
2) *Predictor* adalah variabel independen yang ditunjukkan berdasarkan karakteristik (atribut) data yang akan diklasifikasikan dan didasarkan pada klasifikasi yang telah dibuat. Contohnya seperti status perkawinan pelanggan, arah dan kecepatan angin, musim dan lokasi gempa, frekuensi pembelian;
3) *Training dataset* adalah seperangkat data yang berisi nilai-nilai dari kelas dan *predictor* yang berfungsi untuk mengklasifikasikan kelas yang sesuai menurut prediksi yang ada. Contohnya adalah *database* badai, kelompok pelanggan di supermarket (dihasilkan dari jajak pendapat internal) dan *database* penelitian gempa;
4) *Testing dataset* adalah sekumpulan *unseen data* yang digunakan untuk proses pengujian pada klasifikasi terhadap model yang dihasilkan sebelumnya serta memungkinkan untuk melakukan proses evaluasi akurasi klasifikasi terhadap hasil dari *testing dataset*.

2.2.7 Synthetic Minority Over-Sampling Technique (SMOTE)

Data yang tidak seimbang (*imbalance data*) merupakan keadaan dimana terdapat satu atau beberapa kelas yang memiliki lebih banyak sampel data dibandingkan dengan kelas lainnya. Jika data yang tidak seimbang ini digunakan sebagai sampel pembelajaran klasifikasi, maka model klasifikasi yang dihasilkan tidak akan mampu memprediksi masing-masing kelas secara maksimal (Soonthornphisaj dkk., 2018). Permasalahan *imbalance data* dapat diatasi dengan menggunakan salah satu teknik *resampling*, teknik ini digunakan untuk

menyeimbangkan kembali sampel pada dataset yang tidak seimbang sehingga distribusi kelas menjadi simetris pada tahap pembelajaran (Haixiang dkk., 2017).

SMOTE merupakan salah satu teknik *resampling* pada *over-sampling* yang digunakan untuk pembangkitan data minoritas sebanyak data mayoritas menggunakan teknik interpolasi. Algoritma SMOTE dimulai dari mencari tetangga untuk nilai *k*-terdekat untuk setiap *instance* minoritas, kemudian untuk setiap tetangga secara acak memilih titik dari garis yang menghubungkan tetangga dan *instance* tersebut. Algoritma ini mampu menghasilkan *instance* sintetis daripada duplikat instance kelas minoritas, oleh karena itu masalah *overfitting* dapat dihidari (Soonthornphisaj dkk., 2018).

Pembangkitan data sintetis yang bertipe data numerik berbeda dengan kategorik. Prosedur pembangkitan data sintetis dengan tipe data kategorik dilakukan dengan cara menentukan nilai yang paling sering muncul (modus) antara data asli dengan data tetangga terdekat, apabila terdapat nilai yang sama maka pilih isi data secara acak. Jadikan nilai yang diperoleh tersebut menjadi data sintetis yang baru (Chawla dkk., 2002).

Algoritma SMOTE ditulis dalam bentuk pseudcode sebagai berikut (Soonthornphisaj dkk., 2018)

Algoritma SMOTE (T, N, K)
Input : T (jumlah sampel minoritas); N (persentase SMOTE); K (jumlah tetangga terdekat)
Output : (N/100) * T sampel sintetis kelas minoritas

```
1.  (* Apabila N kurang dari 100%, kelas minoritas akan di SMOTE secara
    random)
2.  if N < 100
3.      then Acak kelas minoritas T
4.           T = (N/100) * T
5.           N = 100
6.  endif
7.  N = (int) (N/100) (* jumlah SMOTE diasumsikan dalam kelipatan integer
    dari perhitungan 100*)
8.  k = jumlah dari tetangga terdekat
```

```
9.    numattrs = jumlah dari atribut
10.   Sample [][] : array dari sample kelas minoritas awal
11.   newindex = menyimpan hitungan jumlah sampel sintetis yang dihasilkan,
      diinisialisasi ke 0
12.   Synthetic [][] = array dari sampel sintetis (* menghitung k tetangga yang
      berdekatan dari tiap sampel kelas minoritas *)
13.   for i ← 1 to T
14.       Hitung k tetangga yang berdekatan untuk setiap i, dan simpan
          indexnya ke nnarray
15.       Populate (N, i, nnarray)
16.   endfor
17.   Populate (N, i, nnarray) (* fungsi untuk membangkitkan sampel sintetis*)
18.   while N ≠ 0
19.       Pilih sebuah nomor random antara 1 dan k, sebut sebagai nn,
          langkah ini memilih satu dari k tetangga yang berdekatan untuk i.
20.       f attr ← 1 to numattrs
21.       Hitung : diff = Sampel [nnarray [nn][attr] − Sampel [i][attr]]
22.       Hitung : gap = nomor acak antara 0 sampai dengan 1
23.       Sintetis [newindex][attr] = Sampel[i][attr] + gap * diff
24.       endfor
25.       newindex++
26.       N = N -1
27.   endwhile
28.   return (* Akhir dari populasi *)
```

Atau dengan persamaan berikut (Khasana dkk., 2019) :

$$X_{syn} = X_i + (X_{knn} - X_i)\ x\ \delta \tag{2.1}$$

X_{syn} adalah data sintesis yang akan diciptakan, X_i adalah data yang akan direplikasi sedangkan X_{knn} adalah data yang memiliki jarak terdekat dari data yang akan direplikasi dan δ adalah nilai random antara 0 dan 1.

2.2.8 *Decision Tree* (DT)

Salah satu teknik *predictive data mining* adalah *decision tree* / pohon keputusan yang diperkenalkan oleh Hunt pada tahun 1989. *Predictive data mining* adalah tahapan menambang data dengan menggunakan *business intelligence* atau data lain untuk memprediksi sebuah tren yang mampu memberikan kontribusi yang bisa membantu pihak pengambil keputusan dalam proses mengambil keputusan. Proses kerja DT dilakukan dengan cara memprediksi suatu objek atau data kedalam beberapa kategori (kelas) dengan memperhatikan nilai kesesuaian pada atributnya (variabel target / *predictor variabel*). DT telah banyak dilakukan dalam berbagai macam bidang ilmu diantaranya, kedokteran (diagnosa medis), ilmu komputer, psikologi (teori keputusan prilaku), percobaan ilmiah, ramalan cuaca, ramalan bisnis dan sebagainya.

Secara konseptual algoritma DT yang diperkenalkan oleh Hunt dapat dilihat pada penjelesan berikut (Gorunescu, 2011) :

1. D_t adalah himpunan objek (data) pelatihan yang mencapai *node t*;
2. Jika D_t adalah sebuah set data kosong, maka *t* adalah sebuah simpul terminal (simpul daun), yang dilabeli sebagai kelas ϕ_t;
3. Jika D_t berisi objek yang termasuk pada kelas C_t, maka *t* juga merupakan simpul daun, dan diberikan label sebagai C_t;
4. Jika D_t berisi objek yang dimiliki lebih dari satu kelas, maka akan digunakan metode uji atribut untuk membagi objek menjadi himpunan yang lebih kecil.

Pohon keputusan dimulai dengan cara menghitung nilai *entropy* sebagai penentu tingkat ketidakmurnian atribut, nilai gain dan nilai information gain. Untuk menghitung nilai information gain digunakan rumus seperti pada persamaan 2.2 atau persamaan 2.3, menghitung nilai entropy menggunakan persamaan 2.4., sedangkan menghitung nilai gain menggunakan persamaan 2.5 atau persamaan 2.6.

$$Information\ Gain = I\ (p, n) \tag{2.2}$$

$$I\,(p, n) = \frac{-p}{p+n} log_2 \left(\frac{p}{p+n}\right) - \frac{n}{p+n} log_2 \left(\frac{n}{p+n}\right) \quad (2.3)$$

$$E\,(A) = \sum_{i=1}^{v} \frac{p_{i+}n_i}{p+n} \left(I(p,n)\right) \quad (2.4)$$

$$Gain = Information\ Gain - Entropy \quad (2.5)$$

$$Gain = I\,(p, n) - E\,(A) \quad (2.6)$$

I adalah *information gain*, *E (A)* adalah *entropy* variabel *A*, *v* adalah variabel, sedangkan *p* adalah jumlah kasus yang bernilai *true* (1) dan *n* adalah jumlah kasus yang bernilai *false* (0).

2.2.9 *Random Forest* (RF)

Pada tahun 1995, Tin Kam Ho dari Bell Labs pertama kali memperkenalkan *random forest*. Menurutnya, *random forest* adalah algoritma pembelajaran *ensemble* untuk tahapan *data mining* dengan tugas klasifikasi dan regresi (Ho, 1995). *Random forest* menggunakan teknik *bagging* untuk membangun *ensemble decision tree*. Pada penelitiannya, Breiman menyebutkan beberapa kelebihan dari *random forest* diantaranya dapat menghasilkan hasil akhir klasifikasi yang baik dan error yang lebih rendah, mampu mengatasi jumlah data yang cukup banyak secara lebih baik dan adalah salah satu algoritma yang efektif dalam permasalahan *missing data* (Breiman, 2001). Nilai akurasi pada sebuah *classifier* tunggal dapat ditingkatkan dengan menggunakan metode *ensemble*, caranya dengan menggunakan lebih dari satu *classifier* dari algoritma yang sama lalu mengkombinasikannya dengan cara *voting* untuk mendapatkan hasil dugaan klasifikasi akhir sebagai hasil akhir dari sebuah proses klasifikasi (Wezel dan Potharst, 2007).

Semakin banyak pohon yang digunakan, maka nilai akurasi akan semakin baik. Untuk melakukan prediksi pada sampel baru pada *random forest* dilakukan dengan cara memasukkan sampel ke dalam pohon keputusan yang sudah terbentuk untuk ditentukan kelas dari sampel tersebut. Langkah ini dilakukan secara berulang kali terhadap keseluruhan pohon keputusan yang terdapat pada

random forest. Hasil *voting* dari beberapa pohon keputusan yang terbentuk merupakan batasan yang dijadikan sebagai penentuan kelas pada proses klasifikasi. Pohon keputusan dengan *vote* yang paling banyak merupakan pemenang dari penentukan kelas klasifikasi (Ho, 1995). Untuk lebih jelas algoritma *Random Forest* terdapat pada Tabel 2.2 (Breiman, 2001).

Tabel 2.2 Algoritma *Random Forest* (Breiman, 2001)

Algoritma *Random Forest for Regression or Classification*
1. Untuk b = 1 *to B;* a) Buatlah sebuah contoh *bootstrap Z* dari *N* jumlah *training* data; b) Buatlah sebuah pohon acak (*random forest tree*) T_b dari contoh *bootstrap* Z yang sudah dilakukan pada langkah (a), lakukan secara berulangkali sampai mencapai simpul yang minimum n_{min}; **tahapan ini dinamakan *random feature selection*.** (i) Pilih variabel *m* secara acak dari variabel *p*; (ii) Pilih variabel/titik split terbaik diantara *m*; (iii) Bagi simpul menjadi beberapa simpul anak 2. *Output* dari *ensemble tree* $\{T_b\}_1^B$
Untuk membuat prediksi pada sebuah titik baru *x* : **Klasifikasi** : Misalkan $\hat{C}_b(x)$ adalah kelas prediksi dari *b* sebuah pohon acak. Kemudian $\hat{C}_{rf}^B(x)$ = *majority vote* $\{\hat{C}_b(x)\}_1^B$

Perlu diingat bahwa untuk setiap kali proses pembentukan pohon, kandidat variabel penjelas yang digunakan untuk melakukan pemisahan bukanlah seluruh variabel yang terlibat tetapi hanya sebagian variabel saja yang dipilih secara acak. Dengan kesimpulan akhir yang diperoleh akan terdapat banyak kumpulan pohon tunggal dengan ukuran dan bentuk yang berbeda-beda satu dengan lainnya (Zhu, 2008).

Menurut Breiman (2001) pada eksperimennya bahwa teknik *bagging* digunakan secara bersama-sama pada proses pembentukan pohon pada random

forest dengan pemilihan fitur secara acak. Setiap subset data pelatihan dibentuk dari data asli yang ada dengan melakukan pemilihan secara acak. Pohon-pohon yang dihasilkan pada *random forest* tidak mengalami *prune* atau pemangkasan.

Ada dua alasan untuk menggunakan teknik bagging pada *random forest*. Yang pertama adalah teknik *bagging* dapat meningkatkan akurasi ketika fitur acak yang dipersyaratkan pada pembentukan pohon acak digunakan. Yang kedua adalah teknik *bagging* dapat digunakan untuk memberikan estimasi berkelanjutan dari kesalahan generalisasi (PE*) dari *ensemble* pohon, dengan kata lain estimasi strength dan korelasi. Estimasi strength dan korelasi tersebut dikenal sebagai *Out-Of-Bag* (OOB) dalam algoritma *random forest* (Breiman, 2001). Untuk lebih jelas cara menghitung *out-of-bag* dalam metode *random forest* terdapat pada Tabel 2.3 (Breiman, 2001).

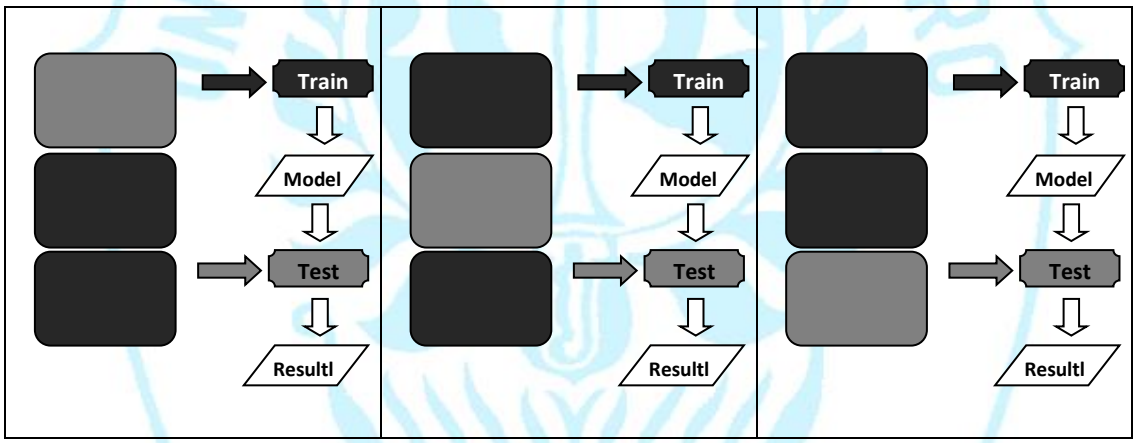
Tabel 2.3 Algoritma *Out-Of-Bag* pada *Random Forest* (Breiman, 2001)

Algoritma *Out-Of-Bag* pada *Random Forest* :
1. Asumsikan sebuah metode untuk membangun sebuah *classifier* dari set data pelatihan; 2. Diberikan set data *training* khusus T, kemudian bentuk set pelatihan dari *bootstrap* T_k, bentuk *classifier* $h(x, T_k)$ dan lakukan pemilihan menggunakan *vote* untuk membentuk *bagged predictor*; 3. Dari setiap *y,x* pada set data *training*, jumlah *vote* untuk *classifier* T_k adalah yang tidak mengandung *y,x*. Ini disebut dengan *out-of-bag classifier*; 4. Lakukan estimasi *out-of-bag* untuk kesalahan generalisasi adalah dengan cara mengecek tingkat kesalahan set data *training* dari *out-of-bag classifier* pada bentuk *classifier* dalam hal ini *tree* yang sudah dihasilkan sebelumnya.

2.2.10 *K-Fold Cross Validation*

Cross-validation adalah metode statistik digunakan untuk melakukan proses penilaian pada teknik analisis/algoritma data yang dipilih serta proses membandingkan kinerja terhadap masing-masing algoritma/teknik yang dipilih. Pada *cross-validation*, data dibagi ke dalam dua bagian yaitu data *training* dan data *testing*/validasi (Refaeilzadeh dkk., 2009; Yadav dan Sukhla, 2016). Baik set

data pelatihan dan data *testing*/validasi harus bersifat *cross-over* dalam putaran secara berturut-turut sedemikian rupa sehingga setiap blok data memiliki peluang untuk divalidasi. Bentuk dasar dari *cross-validation* adalah *k-fold cross-validation* (Refaeilzadeh dkk., 2009). Proses yang dilakukan pada *k-fold cross validation* dimulai dari membagi data terlebih dahulu menjadi *k* sama dnegan ukuran segmen atau bagian. Pelatihan model dilakukan pada bagian k-1 dan sisa bagian untuk dilakukan pengujian. Prosesnya akan dilakukan secara berulang kali sambil mengubah bagian tes satu persatu sampai pada pengujian telah dilakukan pada semua bagian (Yadav dan Sukhla, 2016). Gambar 2.3 memperlihatkan ilustrasi prosedur 3-*folds cross validation* dengan data pelatihan berwarna semakin gelap sedangkan data *testing*/validasi memiliki warna yang lebih ringan. Nilai *k* yang paling umum digunakan pada ranah *machine learning* maupun *data mining* untuk *cross validation* adalah nilai *k* = 10 sehingga menjadi 10-*folds cross validation*.



Gambar 2.3 Prosedur 3-Fold Cross-Validation (Refaeilzadeh dkk., 2009)

2.2.11 *Confusion Matrix* (Pengukuran Kinerja)

Confusion matrix adalah salah satu metode pengukuran kinerja yang terdapat pada algoritma klasifikasi. Pengukuran kinerja sistem klasifikasi memperlihatkan seberapa baik sistem dalam melakukan proses klasifikasi data. Pada kenyataannya, *confusion matrix* merupakan sebuah tabel yang memberikan informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi nyata. Peristiwa tersebut menggambarkan seberapa banyak data yang diprediksi secara benar atau salah oleh sistem (Gorunescu,

2011). Perhitungan kinerja model dalam *confusion matrix* berdasarkan pada nilai *true positive* (TP), *true negative* (TN), *false positive* (FP) dan *false negative* (FN). Bentuk dari *confusion matrix* ditampilkan pada Tabel 2.4.

Tabel 2.4 Confusion Matrix

Aktual	**Prediksi**	
	True	***False***
True	TP (*True Positive*)	FP (*False Positive*)
False	FN (*False Negative*)	TN (*True Negative*)

Berdasarkan nilai *true negative* (TN), *false positive* (FP), *false negative* (FN), dan *true negative* (TN) dapat diperoleh nilai akurasi. Nilai akurasi menggambarkan seberapa akurat sistem dapat mengklasifikasikan data secara benar. Perhitungan nilai akurasi dilakukan dengan persamaan 2.7 berikut ini :

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \; x\; 100\% \qquad (2.7)$$

Nilai *precision* adalah nilai perbandingan dari data yang diprediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif. Perhitugan nilai *precision* dapat dilakukan dengan persamaan 2.8 berikut ini :

$$Precision = \frac{(TP)}{(TP+FP)} \; x\; 100\% \qquad (2.8)$$

Nilai *recall* (sensitifitas) adalah nilai perbandingan dari data yang diprediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. Perhitungan nilai *recall* dapat dilakukan dengan persamaan 2.9 berikut ini :

$$Recall = \frac{(TP)}{(TP+FN)} \; x\; 100\% \qquad (2.9)$$

Nilai $F_{measure}$ adalah nilai timbal balik yang diperolah dari perhitungan *recall* dan presisi. Perhitugan nilai $F_{measure}$ dapat dilakukan dengan persamaan 2.10 berikut ini :

$$F_{measure} = 2\ x\ \frac{presisi\ x\ recall}{presisi + recall}\ x\ 100\% \quad (2.10)$$