BAB II
KAJIAN TEORI

2.1 *Sport Science*

Menurut (Jamen dan Ross 2004) *Sport science* merupakan penerapan prinsip-prinsip *science* untuk membantu meningkatkan prestasi olahraga. Secara umum terdapat 3 bidang dalam *sport science* yaitu: fisiologi, psikologi, dan biomekanika. *Sport science* adalah disiplin ilmu yang mempelajari penerapan dari prinsip-prinsip *science* dan teknik-teknik yang bertujuan untuk meningkatkan prestasi olahraga (Su, 2016). Selain James, Ross dan Su, Papastergiou berpendapat bahwa *sport science* merupakan disiplin ilmu yang luas yang terutama berkaitan dengan proses-proses yang menjelaskan perilaku dalam olahraga dan bagaimana kinerja atletik dapat ditingkatkan (Papastergiou, 2009).

Sport science merupakan disiplin interdisipliner yang memiliki tujuan dalam menggabungkan aspek teoritis serta praktis dan metode bidang informatika dan ilmu olahraga. Penekanan utama dari interdisipliner ditempatkan pada aplikasi dan penggunaan komputer berbasis tetapi juga teknik matematika dalam ilmu olahraga, yang bertujuan dengan cara ini pada dukungan dan kemajuan teori dan praktek dalam olahraga (Link dan Lames, 2009). Alasan mengapa ilmu komputer telah menjadi mitra penting bagi sains olahraga terutama terkait dengan "fakta bahwa penggunaan data dan media, desain model dan analisis sistem. Semakin membutuhkan dukungan alat dan konsep yang sesuai yang dikembangkan dan tersedia dalam ilmu komputer (Ghofrani dan Golsanamlou, 2012). Clyde berpendapat (Clyde, 1976) bahwa *sport science* merupakan disiplin yang mempelajari bagaimana tubuh manusia yang sehat bekerja selama latihan, dan bagaimana olahraga dan aktivitas fisik meningkatkan kesehatan dan kinerja dari perspektif seluler ke seluruh tubuh. Studi ilmu olahraga secara tradisional menggabungkan bidang fisiologi (fisiologi olahraga), psikologi (psikologi olahraga), anatomi, biomekanik, biokimia, dan biokinetik.

Sport science merupakan aplikasi ilmiah dari prinsip pengetahuan untuk membantu atlet dalam meningkatkan performanya. Para ilmuwan olahraga dan

konsultan kinerja terus meningkat dalam permintaan dan jumlah pekerjaan, denganfokus yang semakin meningkat dalam dunia olahraga dalam mencapai hasil terbaik yang mungkin. Melalui studi ilmu pengetahuan dan olahraga, para peneliti telah mengembangkan pemahaman yang lebih besar tentang bagaimana tubuh manusia bereaksi terhadap latihan, pelatihan, lingkungan yang berbeda dan banyak rangsangan lainnya (Ghofrani dan Golsanamlou, 2012).

Tabel 2.1 Referensi Sport Science

Referensi	Tujuan	Bidang Olahraga	Metode	Hasil
(Papastergiou, 2009)	Meninjau literatur ilmiah pada penggunaan komputer dan video *game* di Pendidikan kesehatan dan jasmani	Atletik	KONI	Permainan elektronik memiliki kekuatan untuk membantu seseorang mengadopsi gaya hidup sehat dan aktif secara fisik untuk hidup
(Thompson et al., 2008)	Menentukan apakah video *game* berguna dalam meningkatkan hasil kesehatan		EEG	Video *game* memilikipotensi untuk meningkatkan kesehatan diberbagai bidang.
(Hsiao, 2013)	Menganalisis keseriusan masalah di Taiwan dan merancang latihan penggabungan berbasis AR	Sepak bola	PLAY and GAMES	Peningatan yang signifikan dalam pembelajaran akademis dalam merancang latihan penggabungan dengan AR
(Liang & Liu, 2018)	Meningkatkan tingkat pelatihan fisik dan efesiensi pelatihan		KONI	System pengukuran *ring* otomatis berdasarkan teknologi pemrosesan gambar dapat memenuhi pelatihan pemotretan

Referensi	Tujuan	Bidang Olahraga	Metode	Hasil
(Haghighat, Rastegari, dan Nourafza, 2013)	Sistem penambangan data untuk memprediksi hasil olahraga dannmengevaluasi keuntungan dan kerugian dari setisp sistem		Pelatihan	Penambangan data yang diterima secara luas dapat memprediksi dan menjelaskan peristiwa yang tepa
(Decroix, De Pauw, Foster, & Meeusen, 2016)	Membangung system klasifikasi subjek dalam sport science		Fartlek	System klasifikasi terpadu biometrik, fisiologis
(Ekelund, Yngve, Sjostrom, & Westerterp, 2000)	Mengamati aktivitas CSA untuk penilaian jumlah total aktifitas fisik di atlet remaja		Interval training	CSA memonitor secara akurat dalam pengeluaran energi pada atlet
(Baca, Dabnichki, Heller, & Kornfeind, 2009)	Survei perkembangan trebaru dalam olahraga dan rekreasi dengan penekanan pada teknologi dan ok putasi teknik			Pengembangan system cerdas yang tidak hanya bisa menganalisis data tetapi menyarankan strategi dan intervensi

2.2 Bulutangkis

Bulutangkis atau istilah internasionalnya disebut dengan *badminton* diambil dari nama *badminton house*, satu tempat milik bangsawan Beaufort yang menempati rumah di Gloucestershire, Inggris (Okada et al., 2004). Berawal dari suatu hari yang hujan di akhir tahun 1860, dalam suatu pesta untuk orang dewasa. Dalam pesta ini kemudian disertakan pula satu permainan dari anak-anak hingga dewasa, dan permainan tersebut memakai peralatan yang dilengkapi dengan senar yang berfungsi untuk memukul bola dan juga dibentangkan net yang membagi area permainan menjadi dua yang dilewati oleh bola yang disebut *shuttlecock* (Walinono dkk, 2017). *Shuttlecock* harus dipukul dan melintas melewati atas net untuk smenyatakan bola masih dalam keadaan hidup. Tujuan awal adalah menjaga

shuttlecock tetap di udara dalam waktu selama mungkin. Dan di awal kemunculan dari olahraga ini, pukulan sederhana yang dilakukan untuk melewati atas net ditujukan hanya untuk kesenangan semata.

Gambar 2.1 Pukulan Dasar Bulutangkis

Dalam perkembangan selanjutnya tujuan sederhana dari aksi memukul bola berubah. Tujuan tidak lagi untuk kesenangan, melainkan membuat lawan yang berada di sisi lain lapangan menjadi sulit untuk bisa mengembalikannya. Aksi saling memukul ini dilakukan oleh dua orang atau regu dengan saling melontar dan menerima bola. Permainan tidak lagi sederhana dan berubah menjadi cepat dan membutuhkan ketangkasan. Dari sinilah muncul cikal bakal permainan modern bulutangkis.

Permainan bulutangkis merupakan salah satu cabang olahraga yang tumbuh dan berkembang pesat, mampu mengharumkan bangsa dan negara Indonesia (Juang, 2015). Menurut Rachmaningdiah dan Jannah (2016) menyatakan bulutangkis merupakan cabang olahraga yang termasuk ke dalam kelompok olahraga permainan, dapat dimainkan di dalam maupun di luar ruangan di atas lapangan yang di batasi dengan garis-garis dalam ukuran yang panjang dan lebar yang sudah ditentukan. Lebih lanjutnya lapangan dibagi dua sama besar dan dipisahkan oleh net yang terenggang di tiang net yang di tanam di pinggir lapangan (Rachmaningdiah dan Jannah, 2016).

Sedangkan menurut Kartiko dan Ishak Permainan bulutangkis merupakan salah satu cabang olahraga yang terkenal di dunia (Mahakharisma, 2014). Olahraga

ini menarik minat berbagai kelompok umur, berbagai tingkat ketrampilan, baik pria maupun wanita memainkan olahraga ini di dalam atau di luar ruangan untuk rekreasi juga sebagai persaingan. Bulutangkis adalah olahraga yang dimainkan dengan menggunakan, net, raket dan *shuttlecock* dengan teknik pukulan yang bervariasi mulai dari yang relatif lambat hinggga sangat cepat disertai dengan gerakan tipuan.

Menurut Himawanto (2000) permainan bulutangkis merupakan permainan yang bersifat individu yang dapat dilakukan dengan cara satu orang melawan satu orang atau dua orang melawan dua orang. Dalam hal ini permainan bulutangkis mempunyai tujuan bahwa seseorang pemain berusaha menjatuhkan *shuttlecock* di daerah permainan lawanan dan berusaha agar lawan tidak dapat memukul *shuttlecocok* dan menjatuhkan di daerah sendiri.

Menurut A. Komari (2005) Bulutangkis adalah olahraga yang dimainkan oleh dua orang dalam permainan tunggal dan empat orang dalam permainan ganda, pada sebuah lapangan yang di bagi dua dengan membentangkan net di tengahnya. Permainan bulutangkis menggunakan raket sebagai pemukul bola, dan bola dibuat dari rangkaian bulu beratnya antara 73 sampai 85 grain. Cara bermain bulutangkis adalah melewatkan *Shutlecock* diatas net agar dapat jatuh menyentuh lantai lapangan lawan dan untuk mencegah usaha yang sama dari lawan. Perlengkapan permainan bulutangkis adalah:

1. Lapangan yag rata dengan ukuran panjang 13,40 meter atau 44 *feet* dan lebar 6,10 meter atau 20 *feet* (Chint et al., 1995). Net atau jaring direntangkan di tengah-tengah lapangan sebagai batas pembagi dua lapangan. Tinggi net yang ada di tengah 1,524 meter atau 5 *feet* Tinggi net dekat tiang net atau di pinggir 1,55 meter atau 5 feet, 1 inchi (Chint et al., 1995).
2. Raket, raket digunakan sebagai pemukul bola, Panjang raket sekitar 26 *inchi* beratnya antara $3\frac{3}{4}$ sampai $5\frac{1}{2}$ ons.
3. Shuttlecock: shuttlecock adalah bola yang dipergunakan dalam permainan dibuat dari rangkaian bulu beratnya antara 73 sampai 85 grain. Pada umumnya berat shuttlecock yang digunakan adalah 76 grain (1 grain = 0,0648 gram).

Dalam permainan bulu tangkis di sebuah pertandingan ada yang dinamakan pembagian kelompok dalam bermain. Pembagian kelompok dalam pertandingan olahraga bulu tangkis itu sendiri dinamakan dengan partai-partai, sehingga ada berbagai macam jenis partai yang digunakan atau dipertandingkan dalam permainan olahraga bulu tangkis itu sendiri.

1. Tunggal Putra

   Tunggal putra merupakan salah satu partai dalam sebuah pertandingan bulu tangkis, dimana pertandingan tersebut dimainkan hanya oleh seorang pemain putra yang mana berhadapan secara langsung dengan satu orang juga yang menjadi lawan atau musuhnya. Dalam partai tunggal putra ini sistem pertahanan yang dimiliki tiap–tiap pihak relatif lebih sedikit atau lebih kecil apabila dibandingkan dengan sistem pertahanan yang ada pada partai yang mana permainannya menggunakan sistem pemain atau atlet ganda.

2. Tunggal Putri

   Dalam permainan partai yang satu ini, hampir sama dengan partai tunggal putra. Pada permainan yang ada di tunggal putri ini dilakukan sebuah pertandingan dimana dihadapkan sebanyak dua orang pemain dan dari keduanya sama-sama berjenis kelamin wanita atau putri. Dinamakan tunggal putri, karena dalam permainan ini tidak memiliki sebuah partner dalam melawan musuh melainkan melawannya dengan sendiri seperti tunggal putra di atas.

3. Ganda Putra

   Pada partai permainan bulu tangkis untuk yang satu ini agak sedikit berbeda dengan partai-partai yang sebelumnya, karena dalam partai ganda putra ini permainannya dimainkan oleh sebuah tim yang saling berlawanan yang mana tim tersebut berisikan oleh dua pasang pemain putra yang bertanding satu sama lain dengan mengandalkan kerjasama tim yang cukup baik agar setiap gerakan yang dipakai lebih efektif dan maksimal tentunya dalam bertanding nanti di lapangan.

4. Ganda Putri

   Di dalam permainan bulu tangkis ada juga partai ganda putri, yang mana peraturannya tidak jauh berbeda dengan apa yang ada pada permainan dalam partai ganda putra. Dimana bertemunya dua orang wanita dalam satu tim yang sama melawan tim lainnya yang anggotanya sama seperti tim lawannya. Dalam penggunaan lapangannya juga tidak jauh berbeda apabila dibandingkan dengan partai ganda putra, dan bahkan bisa dikatakan tidak ada perubahan untuk partai ganda putri ini sendiri.

5. Ganda Campuran

   Di dalam permainan bulu tangkis ada juga partai ganda campuran, Dimana dalam satu tim akan ada sepasang pemain wanita dan pria yang melawan pasangan pria dan wanita lainnya di tim lawan tentunya. Peraturan yang ada dalam partai ganda campuran ini tidak jauh berbeda dengan partai ganda lainnya yang sebelumnya sudah ada seperti ganda putra dan ganda putri. Dan juga untuk penggunaan lapangan dalam bermain tidak jauh berbeda dengan peraturan partai ganda lainnya

2.2.1 Teknik Dasar Permainan Bulutangkis

Teknik merupakan suatu proses gerakan dan pembuktian dalam praktek dengan sebaik mungkin untuk menyelesaikan tugas yang pasti dalam cabang olahraga. Dalam permainan bulutangkis teknik dasar harus dipelajari lebih dahulu guna mengembangkan mutu permainan, bulutangkis dimainkan oleh Ganda ataupun ada juga perorangan. Mengingat permainan bulutangkis ada yang ganda, maka kerjasama antar pemain mutlak diperlukan sifat toleransi antar kawan serta saling percaya dan saling mengisi kekurangan dalam regu (Firmansyah, 2013).

Permainan bulutangkis merupakan permainan yang bersifat individual yang dapat dilakukan dengan cara melakukan satu orang melawan satu orang atau dua orang melawan dua orang. Permainan ini menggunakan raket sebagai alat pemukul dan *shuttlecock* sebagai objek pukul, lapangan permainan berbentuk segi empat dan dibatasi oleh net untuk memisahkan antara daerah permainan sendiri dan daerah permainan lawan. Tujuan permainan bulutangkis adalah berusaha untuk menjatuhkan *shuttlecock* di daerah permainan lawan dan berusaha agar lawan tidak dapat memukul *shuttlecock* dan menjatuhkan di daerah permainan sendiri.

Pada saat permainan berlangsung masing-masing pemain harus berusaha agar *shuttlecock* tidak menyentuh lantai di daerah permainan sendiri. Apabila *shuttlecock* jatuh di lantai atau menyangkut di net maka permainan berhenti (Jiang dkk, 2013).

2.2.2 Strategi Bulutangkis

Bulutangkis adalah olahraga yang cepat dan dinamis. Memenangkan permainan, penggunaan taktik yang tepat sangat penting (Tong dan Hong, 2000) Taktik dan strategi adalah komponen yang sangat penting dalam permainan bulu tangkis. Strategi adalah rancangan atau konsep yang bersifat metodis sebelum permainan atau pertandingan berlangsung. Taktik adalah penerapan atau pelaksanaan dari strategi. Dengan taktik dan strategi yang tepat, seorang pemain dapat memenangkan suatu perrmainan dengan efisien. Taktik dan strategi menunjang pemain untuk bermain secara pandai. Seorang pemain mampu memaksa untuk membuka kelemahan lawannya dan menutupi kelemahannya sendiri dengan tepat. Pemain tidak perlu menghabiskan banyak waktu yang hanya membuang-buang tenaga, ketika taktik yang digunakan mampu menekan lawan.

Menurut (Komari, 2005) taktik adalah suatu proses gerakan dan pembuktian dalam praktek dengan sebaik mungkin untuk menyelesaikan tugas yang pasti dalam cabang olahraga. Dalam permainan bulutangkis teknik dasar harus dipelajari lebih dahulu guna mengembangkan mutu permainan, bulutangkis dimainkan oleh Ganda ataupun ada juga perorangan. Mengingat permainan bulutangkis ada yang ganda, maka kerjasama antar pemain mutlak diperlukan sifat toleransi antar kawan serta saling percaya dan saling mengisi kekurangan dalam regu.

Menurut (Subarjah, 2009) taktik merupakan rangkuman metode yang dipergunakan dalam melakukan gerakan suatu cabang olahraga. Pengusaaan teknik dasar dalam permainan bulutangkis merupakan salah satu unsur yang turut menentukan menang atau kalahnya suatu regu di dalam suatu pertandingan disamping unsur-unsur kondisi fisik, taktik, strategi dan mental.

Dalam permainan bulutangkisteknik dasar harus dipelajari lebih dahulu guna membangun mutu permainan, bulutangkis dimainkan oleh dua regu ataupun perorangan. Mengingat permainan bulutangkis ada yang beregu, maka kerjasama antar pemain sangat diperlukan sifat tolerensi antar kawan serta saling percaya dan

saling mengisi kekurangan dalam regu. Atlet, untuk dapat berprestasi semaksimal mungkin, maka suatu tim harus menguasai permainan bulutangkis supaya strategi yang diterapkan oleh pelatih akan berjalan disekitar pertandingan. Salah satu Teknik yang harus dikuasai yaitu Teknik pukulan, didalam permainan bulutangkis yang haru dikuasai oleh pemain diantaranya:

- Teknik Memukul Bola (*Shuttlecock*)

Memukul bola (*Shuttlecock*) merupakan ciri dalam permainan bulutangkis. Perinsip Teknik dalam permainan bulutangkis adalah untuk menyebrangkan bola ke daerah permainan lawan. Irfandy menyatakan (Irfandy dkk, 2017), Teknik pukulan adalah cara-cara melakukan pukulan pada permainan bul

utangkis dengan tujuan menerbangkan bola (*shuttlecock*) ke area permainan lawan. Jenis-jenis pukulan yang harus dikuasai oleh pemain bulutangkis diantaranya: pukulan servis, pukulan lob, pukulan dropshot, pukulan smash, pukulan drive, dan pukulan pengembalian servis (Irfandy dkk, 2017). Yane berpendapat (Yane, 2017) bahwa, macam-macam pukulan dalam permainan bulutangkis terutama adalah servis, lob, smash, dropshot, drive dan netting.

Dari kedua pendapat tersebut dapat disimpulkan bahwa Teknik pukulan yang harus dikuasai dalam permainan bulutangkis meliputi servis, lob, drive, dropshot, smash, netting dan pengembalian servis. Janis-jenis pukulan dapat dilakukan dengan forehand maupun backhand, kecuali pukulan servistinggi yang sulit dilakukan dengan pukulan backhand.

a. Pukulan Servis

Pukulan servis adalah Pukulan dengan raket yang menerbangkan *shuttlecock* ke bidang lapangan lawan secara diagonal dan bertujuan sebagai pembuka permainan (Irfandy dkk, 2017). Servis merupakan pukulan yang sangat menentukan dalam perolehan nilai, karena hanya pemain yang melakukan servis yang dapat mengendalikan jalanya permainan, misalnya sebagai strategi awal serangan (Yane, 2017). Aturan-aturan yang berkaitan dengan pukulan servis pada saat perkenaan (Randy dan Darmawan, n.d., 2017) adalah:

1. Bola maksimal berada sebatas pinggang pemain.
2. Mulai dari pergelangan, kepala raket harus condong ke bawah.
3. Kaki tidak menyentuh garis lapangan.

4. Kedua kaki berhubungan dengan lantai lapangan.
5. Tidak ada gerakan menipu. Gerakan raket dapat diperlambat atau dipercepat tetapi gerakan harus berkelanjutan tanpa adalnya istirahat.

Ada tiga macam jenis servis yang biasa dilakukan oleh pemain bulutangkis, yaitu servis Panjang, servis pendek dan servis tanggung. Servis Panjang adalah servis yang mengarahkan bola tinggi dan jauh. Bola diusahakan jatuh sedekat mungkin dengan garis belakang permainan lawan dengan demikian bola lebih sulit untuk diperkirakan dan dipukul, sehingga semua pengembalian lawan kurang efektif (Subarjah, 2009).

Pukulan pendek adalah servis yang dilakukan rendah, paling sering digunakan dalam partai ganda, karena lapangan untuk ganda lebih pendek, tetapi lebih lebar dari partai tunggal. servis ini dapat dilakukan dengan baik dengan forehand ataupun dengan backhand (Subarjah, 2009).

Servis tanggung sebenarnya hanya variasi saja dari servis pendek. Dilakukan dengan drive dan flick. Servis ini merupakan alternative yang baik dan membuat lawan hanya memilikisedikit waktu untuk bertindak (Subarjah, 2009).

a. Pukulan Lob (Clear)

Pukuan clear biasanya dilakukan dengan tinggi dan panjang. Gunanya untuk mendapatkan waktu untuk kembali ke posisi bagian tengah lapangan. Pukulan ini merupakan strategi yang digunakan khususnya untuk pemain tunggal. Pukulan clear yang bersifat bertahan merupakan pengembalian yang tinggi yang hampir sama dengan pukulan lob dalam tenis. Clear dapat dilakukan dengan pukulan overhand atau underhand, baik dari sisi forehand ataupu backhand untuk memaksa lawan bergerak mundur ke arah sisi belakang lapangannya.

Kegunaan utama dari pukulan clear adalah untuk membuat bola menjauh dari lawan dan membuatnya bergerak dengan cepat. Dengan mengarahkan bola ke belakang lawan atau dengan membuat dia bergerak lebih cepat dari yang dia inginkan, akan membuat dia kekurangan waktu dan membuatnya cepat lelah. Jika melakukan clear dengan benar maka lawan harus bergegas melakukan pukulan balasan dengan akurat dan efektif. Pukulan clear yang bersifat menyerang merupakan clear yang cepat

dan mendatar, yang berguna untuk menempatkan bola ke belakang lawan dan menyebabkan lawan melakukan pengembalian yang lemah. Pukulan clear yang bersifat bertahan memiliki lintasan yang tinggi dan panjang (Subarjah, 2009).

b. Pukulan Drive

Drive adalah pukulan datar yang mengarahkan bola dengan lintasan horisontal melintasi net. Baik drive forehand ataupun backhand mengarahkan bola dengan ketinggian yang cukup untuk melakukan clear pada bola dengan jalur yang datar atau sedikit menurun. Gerakan memukul hampir bersama dengan gerakan memukul dari samping dan biasanya dilakukan dari bagian samping lapangan. Pukulan drive memberi kesempatan untuk melatih footwork karena pukulan ini biasanya dilakukan pada ketinggian antara bahu dan lutut ke arah kiri atau kanan lapangan. Dengan demikian pukulan ini menekankan pada pencapaian bola dengan menyeret atau menggelincirkan kaki pada posisi memukul (Yuliawan dan Sugiyanto, 2014). Drive adalah pukulan pengembalian yang aman akan memaksa lawan mengembalikan bola tinggi. Jika pukulan kurang keras, pengembalian bola lebih mirip dengan pukulan push (mendorong bola) atau drive dari bagian tengah lapangan (Subarjah, 2009).

c. Pukulan Drop (Dropshot)

Pukulan *drop shot* adalah pukulan rendah dan pelan, tepat di atas net sehingga bola langsung jatuh ke lantai. Bola dipukul di depan tubuh dengan jarak lebih jauh dari pukulan *clear overhead,* dan permukaan raket dimiringkan untuk mengarahkan lebih ke bawah. Larinya bola lebih seperti diblok atau ditahan dari pada dipukul. Ciri yang paling penting dari pukulan *drop overhead* yang baik adalah gerakan tipuan. Jika gerakan dapat menipu lawan pukulan mungkin tidak dikembalikan sama sekali.

d. Pukulan Smash

Pukulan smash adalah pukulan yang cepat, diarahkan ke bawah dengan kuat dan tajam untuk mengembalikan bola pendek yang dipukul ke atas. Pukulan smash hanya dapat dilakukan dari posisi overhead. Bola dipukul dengan kuat tetapi harus diatur tempo dan keseimbanganya sebelum

mencoba mempercepat kecepatan smash. Ciri yang paling penting dari pukulan smash overhead yang baik selain kecepatan adalah sudut raket yang mengarah ke bawah. Bola dipukul di depan tubuh lebih jauh dari pukulan clear ata drop. Permukaan raket diarahkan untuk mengarahkan bola lebih ke bawah.

e. Netting

Pukulan netting atau jarring adalah salah satu jenis pukulan yang cukup sulit dalam permainan bulutangkis, karena permainan netting ini banyak memerlukan kecermatan yang penuh perasaan atau *feeling*. Factor tenaga dalam permainan netting hamper tidak diperlukan sma sekali. Pukulan dilakukan dengan tenang dan pasti. Dalam permainan net, bola harus diambil sewaktu bola masih berada diatas. Apabila bola diambil setelah dibawah, tempo permainan akan menjadi lambat dan hal ini memberi kesempatan lawan lebih siap untuk maju. Bola harus serendah mungkin dengan bibir jarring, hal ini mempertinggi terget kesulitan lawan memukul kembali bola, terutama ntuk menerobosnya. Irfandy menyatakan (Irfandy dan others, 2017), tujuan penempatan bola yang jatuh dekat net adalah agar lawan kesulitan untuk mengembalikan bola, karena jatuhnya dengan netmaka pengembalian bola lawan kemungkinan tanggung.

2.3 String Matching

2.3.1 Definisi String Matching

Menurut (Buulolo, 2013) string adalah susunan dari karakter-karakter (angka, *alphabet*, atau karakter yang lain) dan biasanya dipresentasikan sebagai struktur data array. String dapat berupa kata, frase atau kalimat (Buulolo, 2013). String matching diartikan sebagai sebuah permasalahan untuk menemukan pola susunan karakter string didalam string lain atau bagian isi dari teks.(Verykios, dkk, 2000). Pencocokan string adalah proses menemukan pola string tertentu dari volume teks yang besar (Aho dan Corasick, 1975). Pencocokan string mendeteksis pola string tertentu dari data yang disimpan. Saat ini, sebagian besar aplikasi menggunakan konsep pencocokan string untuk pengambilan data atau pencocokan pola dari sejumlah besar data (Syaroni dan Munir, 2005).

Pencarian string yang juga bisa disebut pencocokkan string (string matching) merupakan algoritma untuk melakukan pencarian semua kemunculan string pendek [0….n-1] yang disebut pattern di string yang lebih panjang teks [0…m-1] yang disebut teks (Hadiati, 2007). Dalam algoritma pencocokkan string, teks diasumsikan berada didalam memori, sehingga bila kita mencari string didalam sebuah arsip, maka semua isi arsip perlu dibaca terlebih dahulu kemudia disimpan didalam memori. Jika pattern muncul lebih dari sekali didalam teks, maka pencarian hanya akan memberikan keluaran berupa lokasi pattern ditemukan pertama kali (Syaroni dan Munir, 2005).

Algoritma string matching dapat diklasifikasikan menjaid 3 bagian menurut arah pencarianya (Valianta, 2016)

1. *From left to right*

   Dari arah yang paling alami, dari kiri ke kanan, yang merupakan arah untuk membaca. Algoritma yang termasuk kategori ini adalah algoritma brute force dan algoritma Knuth-Morris-Pratt.

2. *From right to left*

   Dari arah kanan ke kiri, arah yang biasanya menghasilkan hasi terbaik secara partikal. Algoritma yang termasuk kategori ini adalah algoritma boyer-moore.

3. *In a specific order*

   Dari arah yang ditentukan secara spesifik oleh algoritma tersebut, arah ini menghasilkan hasil terbaik secara teoritis. Algoritma yang termasuk kategori ini adalah algoritma *colossi* dan algoritma *crochemore-perrin*.

Beberapa konsep string matching (Hall dan Dowling, 1980) antara lain:

1. *Approximate string matching*, yaitu sebuah pencarian terhadap pola-pola string (mengandung beberapa proses yaitu menghitung jumlah karakter yang berbeda, penyisipan dan penghapusan karakter) sehingga mendekati pola atau pattern dari string yang dicari.
2. Algoritma pencarian string adalah sebuah proses pencarian dari suatu atau beberapa string yang ditemukan dlam sebuah kumpulan string atau teks. Jalan paling sederhana adalah dengan cara membaca karakter satu

persatu dan melakukan perhitungan kesalahan posisi yang ada dari string yang dicari.

2.3.2 Macam-macam Algoritma *String Matching*

Secara garis besar algoritma string matching dibedakan menjadi 2 (Adiwidya, 2009), yaitu:

1. *Exact string matching*
2. *Inexact string matching* atau *fuzzy string matching*

Berikut penjelasan lengkap dua algoritma string matching:

2.3.2.1 Exact String Matching

Exact string matching, merupakan pencocokan string secara tepat dengan susunan karakter dalam string yang dicocokkan memiliki jumlah maupun urutan karakter dalam string yang sama. Bagian algoritma ini bermanfaat jika pengguna ingin mencari string dalam dokumen yang sama persis dengan string masukan.

Beberapa algoritma *exact string matching* yang mengemuka antara lain:

1. *Brute Force*

Analisa dengan metode *brute force* adalah membandingkan karakter per karakter sampai ditemukanya pola yang dicari dari awal string sampai dengan akhir (Ghias, dkk, 1995). Dengan asumsi bahwa teks berada didalam array *T*[1..n] dan pattern berada didalam array *P*[1..m], maka algoritma brute force pencocokkan string adalah sebagai berikut:

a. Mula-mula *pattern P* dicocokkan pada awal teks *T*.
b. Dengan bergerak dari kiri ke kanan, bandingkan setiap karakter didalam *pattern P* dengan karakter yang bersesuaian didalam teks *T* sampai semua karakter yang dibandingkan cocok atau sama (pencarian berhasil), atau dijumpai sebuah ketidakcocokkan karakter (pencarian belum berhasil).
c. Bila pattern *P* belum ditemukan kecocokkan dengan teks *T* belum habis, geser *pattern P* satu karakter ke kanan dan ulangi langkah 2.

2. Knuth Morris Pratt

Algoritma dikembangkan oleh D. E. Knuth, bersama-sama dengan J. H. Morris dan V. R. Pratt (Knuth et al., 1977). Dengan algoritma KMP waktu

pencarian dalam pencocokkan pattern dan teks dapat berkurang dikarenakan algoritma ini melakukan sejumlah pergeseran lebih jauh sesuai dengan informasi ketidakcocokkan string antara teks dan pattern.

Dalam algoritma Knuth-Morris-Pratt (KMP), untuk setiap karakter yang dibandingkan kita bias memutuskan apakah berhasil atau gagal. Algoritma KMP membangun sebuah mesin automata yang status-statusnya adalah status dari string yang sedang kita cari dan setiap status memiliki fungsi berhasil dan gagal artinya status bias jadi semakin jauh atau tetap terhadap status akhir. Kita akan mendapatkan sebuah karakter dari teks saat kita berhasil dalam membandingkan dan akan me-*reuse* karakter bila kita gagal (Daptardar dan Shapira, 2004).

3. Algoritma *Boyer Moore*

Algoritma *boyer-moore* merupakan algoritma yang mempertibangkan *string matching* dengan efesiensi tinggi dari aplikasi. Algoritma ini melakukan pencocokkan karakter yang dimulai dari kanan ke kiri (Besta dan Stomp, 2002).

Algoritma *Boyer-Moore* dipublikasikan oleh S. Boyer, dan J. Strother Moore pada tahun 1997 (Charras dan Lecroq, 2004). Tidak seperti dua algoritma sebelumnya, algoritma *Boyer-Moore* memulai mencocokkan karakter dari sebelah kanan pattern. Ide dibalik algoritma ini adalah bahwa dengan memulai pencocokkan karakter dari kanan, dan bukan dari kiri, maka akan lebih banyak informasi yang didapat (Boyer). Jadi kita bisa melompati atau tidak melakukan perbandingan-perbandingan karakter yang diprediksikan akan gagal. Karakter paling kanan pada pola merupakan karakter pertama yang akan dicocokkan dengan teks. Prinsip-prinsip dasar dari algoritma *Boyer-Moore* yakni:

1. Pembacaan karakter dari kanan ke kiri.
2. *Preprocessing* dimana terdapat kondisi *bad karakter preprocessing* dan *good suffix preprocessing.*

2.3.2.1. *Inexact String Matching (fuzzy String Matching)*

Fuzzy string matching merupakan pencocokkan string secara samar, maksudnya pencocokkan string dimana string yang dicocokkan memiliki kemiripan dimana keduanya memiliki susunan karakter yang berbeda (mungkin jumlah atau urutanya) tetapi string-string tersebut memiliki kemiripan baik kemiripan tekstual/penulisan (*approximate string matching*) atau kemiripan ucapan (*phonetic*

string matching). Metode *fuzzy string matching* diarahkan untuk mencari nilai dari beberapa string yang mendekati dan tidak hanya menghasilkan cocok atau tidak cocok (Cayrol, dkk, 1982).

Beberapa konsep *Fuzzy string matching* (Karim, 2017) antara lain:

1. *Fuzzy String Matching* merupaka salah satu metode pencarian string yang menggunakan proses pendekatan terhadap pola string yang dicari.
2. Melakukan pencarian terhadap string yang sama dan juga string yang mendekati dengan string yang lain yang terkumpul dalam sebuah penampung atau kamus.
3. Kunci dari konsep pencarian ini adalah bagaimana memutuskan bahwa sebuah string yang dicari memiliki kesamaan dengan string tertampung dikamus, meskipun tidak sama persis dalam susunan karakternya. Untuk memutuskan ‘kesamaan’ ini dipergunakan sebuah fungsi yang diistilahkan sebagai similarity function. Fungsi akan bertugas memutuskan string hasil pencarian jika ditemukan string hasil pendekatan (sproksimasi).

Inexact string matching masih dibagi menjadi dua (Foggia, dkk, 2014) yaitu:

1. *Approximate String Matching,* merupakan pencocokkan string berdasarkan kemiripan penulisan (jumlah karakter, susunan karakter dalam dokumen). Tingkat kemiripan ditentukan dengan jauh tidaknya beda penulisan dua buah string yang dibandingkan tersebut dan nilai tingkat kemiripan ini ditentukan oleh pemrogram. Contoh: *c mpuler* dengan *compiler*, memiliki jumlah karakter yang sama tetapi ada dua karakter yang berbeda. Jika perbedaan dua karakter ini dapat ditoleransi sebagai sebuah kesalahan penulisan makadua string tersebut dikatakan cocok (Anderson, 2014).
2. *Phonetic String Matching,* adalah pencocokkan string dengan dasar kemiripan dari segi pengucapanya meskipun ada perbedaan penulisan dua string yang dibandingkan tersebut. Contoh *step* dengan *steb* dari tulisan berbeda tetapi dalam pengucapanya mirip, sehingga dua string tersebut diangap cocok. Contoh yang lain adalah *step,* dengan *steppe, step, stepp, stepe,*. Dalam pembagianya beberapa algoritma phonetic string matching anatara lain: soundex, metaphone, caverphone, phonex, NYSIIS, Jaro-Winkler dan lain-lain (Becker dkk, 2016).

Dalam penerapan kedua algoritma tersebut, sebenarnya phonetic string matching dapat dimanfaatkan untuk approximate string matching dengan batasan dua string yang dicocokkan masih memilii kemiripan ucapan. Phonetic string matching sering juga dimanfaatkan untuk approximate string matching karena phonetic string matching lebih mudah diimplementasikan. Phonetic string matching banyak digunakan dalam bahasa Inggris terdapat perbedaan antara penulisan dan pengucapan.

Riset Janani dan Vijayarani membandingkan algoritma string matching dengan menggunakan tiga algoritma, yaitu: *Enhanced Boore Moore algorithm*, *Enhanced Rabin Karp algorithm* dan *Enhanced Knuth-Morris-Pratt*. Hasil menentukan algoritma pencarian string terbaik dengan menggunakan algoritma Knuth-Morris-Pratt yang meiliki relevansi tertinggi yaitu, yaitu 100% untuk *single word,* 100% untuk *multiple words,* dan 100% untuk *file.* (Janani dan Vijayarani, 2016). Pada subbab selanjutkan akan dibahas mengenai algoritma Knuth-Morris-Pratt yang menjadi algoritma yang akan diimplementasikan pada penelitian ini.

2.4 Algoritma Knuth-Morris-Pratt

Algoritma *Knuth-Morris-Pratt* dikembangkan oleh D. E. Knuth, bersamasama dengan J.H Morris dan V. R. Pratt (Knuth et al., 1977). Pada algoritma *Knuth-Morris-Pratt* (KMP) informasi ketidak cocokan *pattern* dengan teks yang digunakan disimpan untuk menentukan jumlah pergeseran.

Algoritma KMP melakukan pergeseran lebih jauh sesuai dengan informasi yang disimpan, Tidak seperti algoritma *Brute Force* yang mencocokkan *string* dengan melakukan pengecekan dan melakukan pergeseran setiap satu karakter, pada algoritma *Knuth Morris Pratt* informasi ketidak cocokan *pattern* dengan teks disimpan untuk menentukan jumlah pergeseran. Sehingga algoritma *Knuth Morris Pratt* melakukan pergeseran lebih jauh sesuai dengan informasi yang disimpan, yang menyebabkan waktu pencarian dapat dikurangi secara signifikan.

Algoritma KMP melakukan proses awal terhadap *pattern* P dengan menghitung *prefix* yang mengindikasikan pergeseran s terbesar yang mungkin dengan menggunakan perbandingan yang dibentuk sebelum pencarian *string* (Amaludin, dkk, 2018). *Prefix* itu sendiri merupakan derajat pengulangan karakter

yang akan dapat menentukan karakter yang tidak perlu dicocokan kembali dan juga untuk melakukan pergeseran indeks untuk mengabaikan pencocokkan yang sis-sia.

I	1	2	3	4	5
P	J	A	J	A	N
Prefix	0	0	1	2	0

Gambar 2.2 Pemberian Prefix

Berikut adalah algoritma untuk penentuan *prefix* pada algoritma Knuth-Morris-Pratt seperti pada Tabel 2.2

Tabel 2.2 Pseude Code Nilai Prefix

Pseudocode KMP Pencarian Nilai Prefix	
1	Input : Pattern Output : Nilai Prefix
2	BEGIN
3	Length <- Panjang pattern
4	prefix <- c(0) #inisiasi variable nilai prefix
5	i <- 0 #inisiasi variable i
6	for(j in 2:n_pattern){
7	while(i > 0 && pattern[i+1] != pattern[b]){
8	i <- prefix[i]
9	end ehile
10	if(pattern[i+1] == pattern[j]){
11	i <- i+1
12	end if
13	prefix[j] <- i
14	end for
15	return prefix
	END

Setelah pencarian nilai *prefix* selesai, maka selanjutnya masuk pada algoritma utama Knuht-morris-Pratt seperti yang terlihat pada tabel 2.3.

Tabel 2.3 Pseude Code KMP

Algoritma 3 KMP	
1	Input: string, pattern, prefix Output: indeks hasil
2	**Begin**
3	teks ← masukkan panjang string
4	pattern ← masukkan panjang pattern

```
5     index ← 0 //inisiasi variabel index
6     I ← 0 //inisiasi variabel i
7     for j=1 to n_string do
8       while i > 0 and pattern[i+1] != teks[j] do
9         i <- prefix[i]
10      endwhile
11      if pattern[i+1] == teks[j] then
12        i <- i+1
13      endif
14      if i == n_pattern then
15        index <- c(index, j-n_pattern+1)
16        total <- total+1
17        i <- prefix[i]
18      endif
19    endfor
20  end
```

Berdasarkan *pseudocode* yang ditunjukkan pada 2.3 dapat diketahui langkah-langkah yang dilakukan algoritma *Knuth Morris Pratt* pada saat mencocokkan string pada teks (*string*) adalah sebagai berikut:

1. Algoritma *Knuth Morris Pratt* mulai mencocokkan *pattern* pada awal teks.
2. Dari kiri ke kanan, algoritma ini akan mencocokkan karakter per karakter *pattern* dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
    a. Karakter di *pattern* dan di teks yang dibandingkan tidak cocok (*mismatch*).
    b. Semua karakter di *pattern* cocok. Kemudian algoritma akan memberitahukan penemuan di posisi ini.
3. Algoritma kemudian menggeser *pattern* berdasarkan tabel next, lalu mengulangi langkah 2 sampai *pattern* berada di ujung teks.

Berikut contoh penggunaan algortima Knuth-Morris-Pratt pada pencocokan string, diberikan string “JAJAJAJANJNJAJAN” dengan Panjang enam belas karakter dan pattern “JAJAN” dengan Panjang pattern sebanyak lima karakter.

Cara kerja:

String S

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N

Pattern P

J	A	J	A	N

Langkah 1: bandingkan *pattern* 1 dengan string 1

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
J	A	J	A	N												

Pattern 1 cocok dengan string 1, maka selanjutnya mencocokan pattern 2 dengan string 2

Langkah 2: bandingkan pattern 2 dengan string 2

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
J	A	J	A	N												

Pattern 2 cocok dengan string 2, maka selanjutnya mencocokkan pattern 3 dengan string 3

Langkah 3: bendingkan pattern 3 dengan string 3

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
J	A	J	A	N												

Pattern 3 cocok dengan string 3, maka selanjutnya mencocokan pattern 4 dengan string 4

Langkah 4: bandingkan pattern 4 dengan string 4

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
J	A	J	A	N												

Pattern 4 cocok dengan string 4, maka selanjutnya mencocokan patter 5 dengan string 5.

Langkah 5: bandingkan pattern 5 dengan string 5

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
J	A	J	A	N												

Pattern 5 tidak cocok dengan string 5, maka selanju pattern bergeser sebanyak lima langkah dan mencocokan pattern 1 dengan string 6

Langkah 6: bandingkan pattern 1 dengan string 6:

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
					J	A	J	A	N							

Jika pattern 1 cocok dengan string 6, maka selanjutnya membandingkan patter 2 dengan string 7.

Langkah 7: bandingkan pattern 2 dengan string 7

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
					J	A	J	A	N							

Jika pattern 2 tidak cocok dengan string 7, maka selanjutnya membandingkan patter 3 dengan string 8.

Langkah 8: bandingkan pattern 3 dengan string 8

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
					J	A	J	A	N							

Jika pattern 3 tidak cocok dengan string 8, maka selanjutnya membandingkan patter 4 dengan string 9.

Langkah 9: bandingkan pattern 4 dengan string 9

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
					J	A	J	A	N							

Pattern 4 cocok dengan string 9, karena ada kecocokan untuk pattern tidak dilakukan pergeseran dan mencocokan pattern 5 dengan string 10.

Langkah 10: bandingkan pattern 5 dengan string 10.

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
					J	A	J	A	N							

Pattern 5 tidak cocok dengan string 10, selanjutnya melakukan pergeseran indeks sebanyak lima langkah karena dengan algoritma Knuth-Morris-Pratt memastikan di kedua langkah tersebut *pattern* tidak akan menemukan kecocokkan sempurna dari tiap karakter di *pattern* dengan teks yang se-indeks pada tiap langkahnya. Pencocokan berikutnya antara string pattern 1 dengan string 12

Langkah 11: bandingkan pattern 1 dengan string 11.

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
										J	A	J	A	N		

Pattern 1 tidak cocok dengan string 11, maka pattern bergeser satu langkah dan mencocokan pattern 1 dengan string 12.

Langkah 12: bandingkan pattern 1 dengan string 12

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
											J	A	J	A	N	

Pattern 1 tidak cocok dengan string 12, maka pattern bergeser satu langkah dan selanjutnya mencocokan pattern 1 dengan string 13.

Langkah 13: bandingkan pattern 1 dengan string 13

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
												J	A	J	A	N

Pattern 1 cocok dengan string 13, maka selanjutnya mencocokan pattern 2 dengan string 14.

Langkah 14: bandingkan pattern 2 dengan string 14

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
												J	A	J	A	N

Pattern 2 cocok dengan string 14, maka selanjutnya mencocokan pattern 3 dengan string 15.

Langkah 15: bandingkan pattern 3 dengan string 15

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
												J	A	J	A	N

Pattern 3 cocok dengan string 15, maka selanjutnya mencocokan pattern 4 dengan string 16.

Langkah 16: bandingkan pattern 4 dengan string 16

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
												J	A	J	A	N

Pattern 4 cocok dengan string 16, maka selanjutnya mencocokan pattern 5 dengan string 17.

Langkah 17: bandingkan pattern 5 dengan string 17

J	A	J	A	J	J	A	J	A	N	A	N	J	A	J	A	N
												J	A	J	A	N

Pada langkah 17, pattern 5 cocok dengan string 17 dan pada langkah 17 semua pattern sudah dicocokan sama semua string dan ditemukan sebuah pola kosa kata dalam string. Dalam menemukan sebuah pola *pattern* di dalam *string* akan dilakukan pergeseran beberapa kali untuk mencocokan setiap huruf pada *pettern* yang di mulai dari sebelah kiri untuk mencocokan setiap huruh pada *string*.

Contoh dari algoritma *Knuth Morris Pratt* bekerja ini menggunakan *pattern* (jajan) yang panjangnya lima huruf melakukan pencocokan dengan string yang panjangnya tujuh belas huruf acak untuk menemukan sebuah pola kata yang sama dengan pattern tersebut. Dengan melakukan pencocokan satu persatu di dalam *pattern* dengan seluruh huruh yang terdapat pada *string*.

2.5 Clustering

Clustering (pengelompokan data) mempertimbangkan sebuah pendekatan penting untuk mencari kesamaan dalam data dan menempatkan data yang sama ke dalam kelompok-kelompok. *Clustering* membagi kumpulan data ke dalam

beberapa kelompok dimana kesamaan dalam sebuah kelompok adalah lebih besar daripada diantara kelompok-kelompok (Xu, Damelin, Nadler, dan Wunsch II, 2010). Gagasan mengenai pengelompokan data atau *clustering*, memiliki sifat yang sederhana dan dekat dengan cara berpikir manusia, kapanpun kita dipresentasikan jumlah data besar ini ke dalam sejumlah kecil kelompok-kelompok atau kategori-kategori untuk memfasilitasi analisanya lebih lanjut. Selain dari itu sebagian besar data yang dikumpulkan dalam banyak masalah terlihat memiliki beberapa sifat yang melekat yang mengalami pengelompokan-pengelompokan natural (Jafari dan Naji, 2013).

Algoritma-algortima *clustering* digunakan secara ekstensif tidak hanya untuk mengkategorikan data, akan tetapi juga sangat bermanfaat untuk kompresi data dan konstruksi model. Melalui pencarian kesamaan dalam data, sesorang dapat mempresentasikan data yang sama dengan lebih sedikit symbol misalnya. Juga, jika kita dapat menemukan kelompok-kelompok data, kita dapat membangun sebuah model masalah berdasarkan pengelompokkan-pengelompokkan ini (Dubes dan Jain, 1988). *Clustering* sering dilaksanakan sebagai langkah pendahuluan dalam proses pengumpulan data. Dengan *cluster-cluster* yang dihasilkan dan digunakan sebagai input lebih lanjut ke dalam sebuah teknik yang berbeda (Lance dan Williams, 1967).

2.5.1 Analisis *Clustering*

Analisis berbasis *cluster* merupakan suatu teknik untuk membagi data kedalam beberapa kelompok (*cluster*) yang memiliki arti dan berguna. Jika kelompok yang memiliki arti adalah tujuannya, maka cluster-cluster harus dapat mengetahui struktur alami dari data. Semakin besar kesamaan (homogenitas) antar objek dalam suatu *cluster* dan semakin besar perbedaan antara *cluster*, maka *clustering* akan semakin baik (Tripathy, dkk, 2011).

Pada proses *clustering* tidak diperlukan label kelas untuk setiap data yang diproses karena nantinya label baru bisa diberikan ketika *cluster* sudah terbentuk. Karena tidak adanya label kelas maka *clustering* sering disebut juga *unsupervised learning* (Prasetyo dan Purwarianti, 2014). Prasetyo (2014) menyatakan bahwa proses *clustering* dapat dibedakan menjadi tiga jenis, yaitu dapat dibedakan menurut struktur *cluster*, keanggotaan data dalam *cluster*, dan kekompakan data dalam

cluster. Adapun penjabaran dari ketiga jenis proses *clustering* tersebut ditunjukkan secara rinci pada tabel 2.4

Tabel 2.4 Jenis-jenis Clustering

Proses Clustering		Deskripsi
Menurut struktur *cluster*	Hirarki	a. Satu data tunggal bisa dianggap sebagai sebuah *cluster.* b. Dua atau lebih *cluster* kecil dapat bergabung menjadi sebuah *cluster* besar. c. Begitu seterusnya hingga semua data dapat bergabung menjadi sebuah *cluster*.
	Partisi	a. Membagi set data ke dalam sejumlah *cluster* yang tidak bertumpang-tindih antara satu *cluster* dengan *cluster* lain. b. Setiap data hanya menjadi anggota satu *cluster* saja.
Menurut keanggotaan data dalam *cluster*	Ekslusif	Sebuah data bisa dipastikan hanya menjadi anggota satu *cluster* dan tidak menjadi angota di *cluster* lain.
	Tumpang tindih	Membolehkan sebuah data menjadi anggota di lebih dari satu *cluste*
Menurut kesamaan data dalam *cluster*	Lengkap	Jika semua data bisa bergabung, maka data kompak menjadi satu cluster, jika tidak data dikatakan menyimpang.
	Parsial	

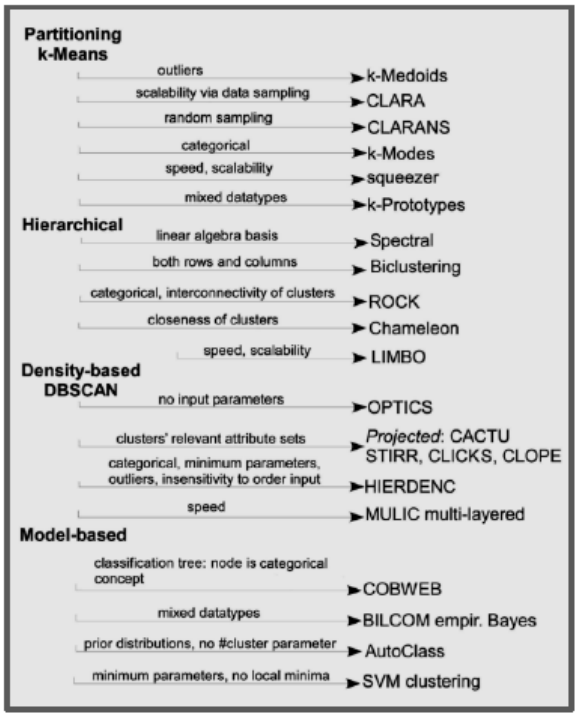
Karena tidak ada label kelas yang digunakan dalam prosesnya, oleh Prasetyo *clustering* dikatakan sangat cocok untuk melakukan *clustering* data yang label kelasnya memang sulit didapatkan pada saat pembangkitan fitur. Pada *clustering*, segera setelah *cluster* terbentuk, maka label kelas untuk setiap data dapat diberikan dengan cara mengamati keluaran yang dihasilkan oleh proses *clustering*. Karena tidak membutuhkan label kelas, kemiripan (*similarity*) harus didefinisikan berdasarkan atribut objek, di mana definisi tersebut bergantung pada algoritma *clustering* yang diterapkan. Algoritma *clustering* yang baik digunakan tergantung pada penerapan set data yang diproses.

Pada algoritma *clustering* terdahulu kebanyakan didesain dengan asumsi bahwa atribut dari data yang diolah merupakan data yang bersifat numerik. Namun,

hal tersebut tidak sepenuhnya benar pada kasus-kasus dalam dunia nyata, data bisa didapatkan dari berbagai macam tipe data seperti diskret *categorical* atau *structural* (Link dan Lames, 2009). Adapun tipe data yang dapat diteliti dalam analisis berbasis *cluster* menurut (Link dan Lames, 2009) adalah: *Clustering* pada data kategorikal, *Clustering* pada data teks, *Clustering* pada data multimedia, *Clustering* pada data *time-series*, *Clustering* pada rangkaian diskret, *Clustering* pada data berbasis jaringan dan *Clustering* pada data yang tidak pasti.

2.5.2 Clustering Pada Set Data Kategorikal

Proses *clustering* pada set data kategorikal membagi N buah objek ke dalam k buah *cluster*. Objek o memiliki m buah atribut {o1, o2,…., om}. Atribut oi, i=1..m, memiliki sebuah *domain* Di dari data kategorikal atau *boolean*. Adapun klasifikasi dari algoritma yang dapat digunakan untuk melakukan proses *clustering* pada data kategorikal diperlihatkan pada Gambar 2.3.



Gambar 2.3 Klasifikasi Clustering data Kategorikal

Dalam proses *clustering* data kategorikal, perhitungan *similarity* atau jarak antar data kategorikal tidak seperti pada data numerik yang kontinyu. Salah satu karekteristik dari data kategorikal adalah atribut kategorikal menggunakan nilai

diskret yang tidak memiliki susunan yang inheren yang ada pada atribut data numerik. Beberapa teknik dapat digunakan untuk mengukur similaritas pada *clustering* data kategorikal diantaranya, *hamming distance*, pendekatan probabilitas, *information-theoritic measures*, dan *context-based similarity measures* (Link dan Lames, 2009).

2.5.3 Algoritma *k-Means*

Secara historis algoritma *k-Means* menjadi salah satu algoritma yang paling penting dalam bidang data mining (Bhattacharya, dkk, 2009). Algoritma *k-Means* dapat diterapkan pada data yang direpresentasikan dalam r dimensi ruang tempat. Algoritma ini mengelompokkan set data *r*-dimensi, X = {x_i | *i=1,….*, *N* }, di mana $x_i \in R\ d$ yang menyatakan data ke-*i* sebagai titik data. Algoritma *k-Means* mempartisi X ke dalam *k cluster*, algoritma *k-Means* mengelompokkan semua titik data dalam X sehingga setiap titik *xi* hanya jatuh dalam satu *k* partisi. Yang diperhatikan adalah titik berada dalam *cluster* yang mana, dilakukan dengan cara memberikan setiap titik sebuah ID *cluster*. Titik dengan ID *cluster* yang sama berarti berada dalam satu cluster yang sama, sedangkan titik dengan ID *cluster* yang berbeda berada dalam *cluster* yang berbeda. Untuk menyatakan hal ini, biasanya dilakukan dengan vektor keanggotaan *cluster m* dengan panjang N, di mana *mi* bernilai ID cluster titik x_i.

Parameter yang harus dimasukkan ketika menggunakan algoritma *k-Means* adalah nilai *k*. Nilai *k* yang digunakan biasanya didasarkan pada informasi yang diketahui sebelumnya tentang berapa banyak *cluster* data yang muncul dalam X, berapa banyak *cluster* yang dibutuhkan untuk penerapannya, atau jenis *cluster* dicari dengan mengeksplorasi/melakukan percobaan dengan beberapa nilai k. Beberapa nilai k yang dipilih tidak perlu memahami bagaimana *k-means* mempartisi set data x (Prasetyo dan Purwarianti, 2014).

Dalam *k-means*, setiap *cluster* dari *k cluster* diwakili oleh titik tunggal dalam $\Re\ d$. Set representative *cluster* dinyatakan C={cj/J=1, ..., *k}*. Sejumlah *k representative cluster* tersebut tersebut disebut juga sebagai *cluster means* atau *cluster centroid* (atau *centroid* saja). Untuk set data dalam *x* dikelompokkan berdasarkan konsep kedekatan atau kemiripan. Meskipun konsep yang dimaksud untuk data-data yang berkumpul dalam satu cluster adalah data-data yang mirip,

tetapi kuantitas yang digunakan untuk mengukurnya adalah ketidakmiripan. (*dissimilarity*). Artinya, data-data dengan ketidakmiripan (jarak) yang kecil/dekat maka lebih besar kemungkinannya untuk bergabung dalam satu *cluster*. Matrik yang umum digunakan untuk ketidakmiripannya adalah *Euclidean* (Prasetyo dan Purwarianti, 2014).

Prasetyo (2014) menyatakan, pada saat data sudah dihitung ketidakmiripan terhadap setiap *centroid*, maka selanjutnya dipilih ketidakmiripan yang paling kecil sebagai *cluster* yang akan diikuti sebagai relokasi data pada *cluster* di sebuah iterasi. Relokasi sebuah data dalam *cluster* yang diikuti dapat dinyatakan dengan nilai keanggotaan a yang bernilai 0 atau 1. Nilai 0 jika tidak menjadi anggota sebuah data *cluster*, hanya satu yang bernilai 1, sedangkan lainnya 0 seperti dinyatakan oleh persamaan berikut:

$$a_{ij} = \begin{cases} 1, \arg\min\{d(x_{1,}c_{1,}) \\ 0, lainya \end{cases}$$

d(xi,cj) menyatakan ketidakmiripan (jarak) dri data ke-i ke *cluster* cj. Sementara relokasi centroid untuk mendapatkan titik centroid C didapatkan dengan menghitung rata-rata setiap fitur dari semua data yang tergabung dalam setiap *cluster*. Rata-rata sebuah fitur dari semua data dalam sebuah *cluster* dinyatakan oleh persamaan berikut:

$$C_1 = \frac{1}{Nk} \sum_{1=1}^{Nk} X_{jl}$$

Nk adalah jumlah data yang tergabung dalam *cluster*.

Jika diperhatikan dari angkanya yang selalu memilih *cluster* terdekat, maka sebenarnya K=-Means berusaha untuk meminimalkan fungsi objektif/fungsi biaya non-negatif, seperti dinyatakan oleh persamaan berikut:

$$J = \sum_{1=1}^{K} \sum_{1=1}^{K} a_{ic} d(x_i, c_1)^2$$

Algoritma clustering dengan *k-means* (Huang, 1998).

1. Insialisasi: tentukan nilai k sebagai jumlah *cluster* yang diinginkan dan metric ketidakmiripan (jarak) yang diinginkan. Jika perlu, tetapkan ambang batas perubahan fungsi objektif fungsi dan ambang batas perubahan posisi *centroid*.

2. Pilih K data dari set data X sebagai *centroid*.
3. Alokasikan semua data ke *centroid* terdekat dengan matriks yang sudah ditetapkan (memperbarui *cluster* ID setiap data).
4. Hitung kembali *centroid* C berdasarkan data yang mengikuti *clusteri* masingmasing.
5. Ulangi langkah 3 dan 4 hingga kondisi konvergen tercapai, yaitu (a) perubahan fungsi objektif sudah di bawah ambang batas yang diinginkan; atau (b) tidak ada data yang berindah *cluster*; atau (c) perubahan posisi centroid sudah di bawah ambang batas yang ditetapkan.

Algoritma k-*means* disajikan pada Algoritma k-*means* mengelompokkan set data x dalam langkah iterative. Berikut dua langkah utamanya, yaitu (1) penentua kembali ID *cluster* dari semua titik data dalam X, da (2) memperbarui representasi *cluster* (*centroid*) berdasarkan titik data dalam setiap cluster.Algoritma bekerja sebagai berikut: Pertama, representasi cluster diinisialisasi dengan memilih k data daa $\Re d$ secara acak. Selanjutnya, secara iterative melakukan dua langkah berikut sampai tercapai kondisi konvergen.

Langkah 1: Data *assignment*. Setiap data ditetapkan ke *centroid* terdekat dengan pemecahan hubungan apa adanya. Hasilnya berupa data yang terpartisipasi.

Langkah 2: *Relocation of "means"*. Setiap representasi *cluster* direlokasi ke pusat (center) dengan rata-rata aritmetika dari semua data yang ditetapkan masuk ke dalamnya. Rasionalnya langkah ini didasarkan pada observasi bahwa dalam memberikan set titik, representasi tunggal yang terbaik untuk set tersebut (dalam hal meminimalkan jumlah kuadrat jarak Euclidean diantara setiap titik data dan representative) adalah dari rata-rata dari titik data. Hal ini jugalah yang menyebabkan metode ini sering disebut dengan *cluster mean* atau atau *cluster centroid*, seperti nama yang dimiliki.

Algoritma *k-Means* mencapai kondisi konvergen ketika pengalokasian kembai ke titik data (dan juga lokasi *centroid* c_1) tidak lagi berubah. Proses dari iterasi higga dicapai kondisi konvergen juga dapat diamati dari nilai fungsi obyektif yang didapatkan. Pada kondisi ini yang semakin konvergen dapat diamati bahwa nilai fungsi objektif akan semakin menurun.

2.6 Bahasa Pemrograman R

R adalah Bahasa pemroraman dan perangkat lunak untuk analisis statistika dan grafik. R dibuat oleh Ross Ihaka dan Robert Gentleman di Universitas Auckland, selandia baru. Saat ini dikembangkan oleh R Development Core Team. Bahasa R menjadi standar de facto diantara statistikawan untuk pengembangan perangkat lunak statistika dan digunakan secara luas untuk pengembangan perangkat lunak statistika dan analisis data. Bahasa R merupakan bersi *open-source* dari bahasa pemrograman S. R menyediakan berbagai teknik statistika (permodelan linier dan nonlinier, uji statistik klasik, analisis deret waktu, klasifikasi, klasterisasi, dan sebagainya) serta grafik. R, sebagaimana S, dirancang sebagai bahasa komputer sebenarnya, dan mengizinkan penggunanya untuk menambah fungsi tambahan dengan mendefinisikan fungsi baru.

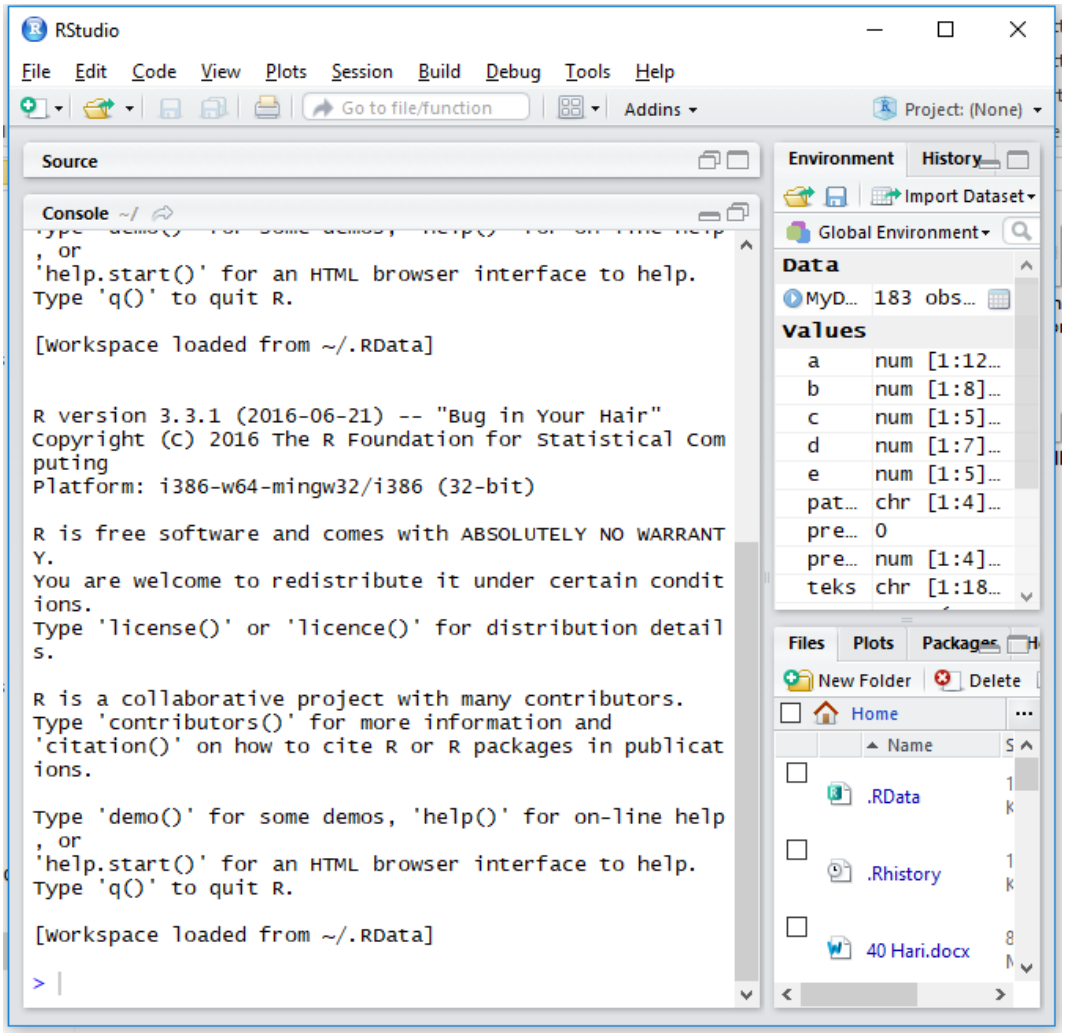
R project pertama kali dikembangkan oleh Robert Gentleman dan Ross Ihaka (nama R untuk sofware ini berasal dari huruf pertama nama kedua orang tersebut) yang bekerja di departemen statistik Universitas Auckland tahun 1995. Sejak saat itu software ini mendapat sambutan yang luar biasa dari kalangan statistikawan, industrial engineering, peneliti, programmer dan sebagainya. Pada saat ini, source code kernel R dikembangkan terutama oleh R Core Team yang beranggotakan 17 orang statistisi dari berbagai penjuru dunia.

Program R dapat diunduh secara gratis di http://cran.r-project.org/ dengan situs resminya di http://www.r-project.org/ . R pertama kali diluncurkan pada tahun 1997. R memiliki banyak kelebihan, diantaranya:

- R tersedia untuk berbagai system operasi
- Memiliki kemampuan membeuat graph yang canggih
- Mempunyai sintaks yang mudah dipelajari dengan menciptakan fungsi-fungsi buatan pengguna sendiri
- Memiliki banyak *package* yang dapat diunduh secara gratis

Namun, kelebihan utama bekerja dengan R adalah pemrograman ini menyediakan ekosistem ilmiah dari *package* yang disumbangkan oleh para penelitidan praktisi. Dan ada lebih dari 5000 *package* tersedia yang terklasifikasi ke lebih dari 20 tampilan tugas. Misalnya, tampilan tugas *Machine Learning*

berisikan lebih dari 30 tampilan *Package* untuk jaringan syaraf tiruan, random forest, ataupun *support vector machines.*



Gambar 2.4 RStudio

R Studio merupakan salah satu GUI untuk pemrograman bahasa R. Gambar 2.4, merupakan tampilan dari R Studio. R Studio dapat digunakan apabila program R telah di install. Tab console pada R Studio digunakan untuk menjalankan fungsi-fungsi atau sintaks-sintaks bahasa R.

Meskipun R mengutamakan penggunaan bahasa pemrograman, untuk pengguna awam dengan metode statistik dan bahasa pemrograman, dapat memanfaatkan paket R-commander yang telah disediakan di *library*. Dengan R-commander, pengguna dapat melakukan pengolahan data secara statistik dengan mudah. Hal ini sangat dimungkinkan karena melalui R-commander, pengguna bias

langsung melakukan pengolahan data dengan memilih menu-menu yang disediakan pada jendela R-commander.

Rexer's Annual Data Miner Survey 2010 menyebutkan bahwa R telah menjadi alat data mining yang digunakan oleh mayoritas pengguna. Salah satu penyebabnya adalah adanya Rattle, yaitu suatu *library* yang digunakan secara khusus untuk Data Mining melalui GUI (*Graphic User Interface*). Rattle dapat menyajikan ringkasan data secara statistik dan secara visual dari berbagai sumber data seperti Excel, SQL, ataupun XML, yang selanjutnya dapat mentransformasi data ke dalam bentuk yang siap untuk dimodelkan. Untuk pemodelannya dapat digunakan berbagai metode baik *supervised* maupun *unsupervised* dan sekaligus mampu membuat laporan secara grafis untuk menampilkan kerja model yang dibangun.

Pada R prompt kita dapat memasukkan berbagai macam ekspresi. Setiap symbol atau ekspresi memiliki banyak banyak fungsi, contohnya symbol "<-", symbol ini berfungsi sebagai operator penugasan atau *assignment operator* seperti berikut:

```
R> n <- 23
R> i <- 6
R> j <- 8
```

Gambar 2.5 Operator Penugasan 1

Dari ekspresi yang dimasukkan pada gambar 2.5, maka nilai n adalah 23, nilai I adalah 26 dan j adalah. Assignment operator tidak hanya untuk nilai angka saja, nilai string pun bisa, contohnya :

```
R> a <- "aku"
R> b <- "kamu"
R> c <- "kita"
```

Gambar 2 6 Operator Penugasan

Dari ekspresi y maka a berisikan kata "aku", q berisi kata "kamu", dan r berisi kata "kita". Dalam pemrograman R dapat

melakukan operasi perkalian, pembagian, pertambahan, dan pengurang. Berikut ini beberapa contoh dari operasi tambah, kurang, kali, dan bagi :

- Perkalian i dan j yang memiliki nilai i = 26 dan j = 14

```
R> i <- 26
R> j <- 14
R> i * j
R > [1] 364
```

Gambar 2.7 Operator Perkalian

- Pembagian i dan j yang memiliki nilai i = 26 dan j = 2

```
R> i <- 26
R> j <- 2
R> i / j
R> [1] 13
```

Gambar 2.8 Operator Pembagian

- Penjumlahan i dan j yang memiliki nilai i = 26 dan j = 16

```
R> i <- 26
R> j <- 14
R> i + j
R> [1] 40
```

Gambar 2.9 Operasi Penjumlahan

- Pengurangan i dan j yang memiliki nilai i = 26 dan j = 14

```
R> i <- 26
R> j <- 14
R> i - j
R> [1] 12
```

Gambar 2.7 Operasi Pengurangan

Di dalam bahasa R ada pula fungsi yang digunakan untuk membaca file yaitu *read.table("nama_file.format_file").* Dengan fungsi ini kita dapat melakukan proses data menggunakan fungsi atau program R yang telah dibuat. Contoh :

```
R> data <- read.table(table.txt)
```

Gambar 2.8 Membaca File di R

Berdasarkan perinta yang ditampilkan pada Gambar 2.11, maka variable "data" akan berisi nilai-nilai yang ada dalam file "table.txt".

Fungsi-fungsi yang kita buat dapat disimpan, forman penyimpanan file untuk pemrograman R adalah "nama_file".R. setelah program yang kita bangun tersimpan, maka cara untuk menjalankan setiap fungsi yang ada dalam file tersebut adalah dengan perintah source("direktori_file/nama_file.R"), contoh :

```
R> F:\AAAAA\Data\Skripsi\koding\cobadata.R
```

Gambar 2.9 Menjalankan File R

Dengan menggunakan perintah pada Gambar 2.12, maka fungsi yang ada dalam file "cobadata.R" dapat digunakan.

2.7 Package Kamila

Subbab ini akan membahas mengenai *package* Kamila dimulai dari clustering di R, definisi, instalasi dan contoh penggunaan.

2.7.1 R *Package*

R *Package* atau R *Library* merupakan kumpulan fungsi, data dan kode di dalam Bahasa R. di dalam R terdapat lebih dari 6 ribu *package* yang berada pada repository CRAN. Ketika pengguna melakukan instalasi R ke dalam computer maka beberapa *package* akan ikut terinstal secara otomatis, sedangkan untuk *package* yang lainya harus di instals sendiri secara manual supaya bisa digunakan.

Seperti yang dikatakan oleh (Foss, dkk, 2016) ada beberapa cara dalam menginstal *package*, adalah sebagai berikut:

1. Instalasi R *package* dari local

   Langkah-langkah yang bisa dilakukan adalah:

   a. Dari halaman CRAN pilih *package* yang diinginkan.
   b. Download *package source*
   c. Pada R GUI, pilih menu *package*, lalu pilih *instal package from local zip files.*
   d. Jika berhasil, R console akan memberi pesan *package successfully unpacked and MD5 sums checked.*

2. Install R *package* langsung dari CRAN repository

   Instal *package* dapat langsung dilakukan dari CRAN *repository* apabila sedang terhubung dengan internet. Berikut cara yang harus dilakukan:

   a. Instal.package (pkgs)

      Pkgs merupakan nama package yang diinstal, sehingga apabila ingin menginstal WriteXLS, maka pada R console ditulis script Instal.package(WriteXLS).

   b. Setelah menjalankan script tersebut, pilih CRAN mirror, misalnya "Indonesia (Jakarta)". Dan akan muncul pesan package 'WriteXLS' *successfully and MD5 sums checked* pada R *console*.

   c. Jika ingin menginstal dua package tinggal melakukan modifikasi, Instal.package(c("WriteXLS","twitteR")).

2.7.2 R *Package* Kamila

Package Kamila merupakan metode untuk mengelompokan data dengan jenis campuran dalam clustering. Hal ini dapat menyelesaikan berbagai metode dalam pengelompokan data dengan kategori kontinu dan campuran (Foss, dkk, 2016). Algoritma Kamila menggabungkan dua algoritma popular, algoritma k-means dan model campuran Gaussian-multinomial, keduanya berhasil diadaptasi dengan baik. Data yang sangat besar seperti k-means, kamila tidak membuat asumsi parameter yang kuat mengenai variable kontinu, namun Kamila berhasil menyeimbangkan variable kontinu dan kategori tanpa menentukan bobot, tetapi kamila didasarkan pada kedekatan jarak antar setiap data.