# BAB II
# TINJAUAN PUSTAKA DAN DASAR TEORI

## 2.1. Tinjauan Pustaka

Proses penambangan data *Twitter* terdiri dari *information extraction, information summarizatian, dan document classification* (Han dkk., 2000). Sebelumnya telah dilakukan penelitian analisis sentimen terhadap *Twitter* dengan berbagai metode seperti *Naïve Bayes Classification, Maximum Entropy*, ataupun *Support Vector Machine*. Didapatkan kesimpulan bahwa penggunaan metode *Support Vector Machine* (SVM) memberikan hasil paling baik dibandingkan metode lainnya, yaitu dengan tingkat keakuratan hingga 82,2% (Go dkk., 2009). Penelitian terkait dengan kemacetan lalu lintas secara *realtime* di Chicago dan New York (USA) dapat dilakukan dengan mengekstrak informasi dari lokasi geotag pengguna, waktu, tanggal dan isi teks *tweet* (Paule dkk., 2018).

Penggunaan media *Twitter* sebagai bahan untuk kasus okupansi lahan berdasarkan kondisi cuaca dengan algoritma *Naïve Bayes Classifier* dan SVM didapatkan hasil dimana metode SVM memiliki tingkat performansi yang lebih besar daripada *Naïve Bayes* dengan perbedaan tingkat akurasi sebesar 0.0015 dikarenakan ketidakmampuan *Naive Bayes* dalam mengatasi masalah data *outlier* (Marchel dkk., 2017). Terdapat penelitian sebelumnya yang juga menggunakan 2 algoritma yaitu *Naïve Bayes Classifier* dan SVM (Hassan dkk., 2011). Penelitian ini dilakukan dengan mengkategorisasi teks berbasis konten melalui Wikitology dengan menggunakan data dari 20 kelompok berita dan 1000 kategori, didapatkan hasil bahwa semakin banyak data *training*, keakuratan SVM sebakin besar, semakin banyak fitur dan pengayaan *external* untuk labelling, keakuratan *Naïve Bayes* semakin besar.

Algoritma pengelompokan *Ant Colony* dimana skema pengelompokan node terdekat diusulkan untuk mengatasi masalah routing lokasi dinamis (DLRP) (Gao dkk., 2016). DLRP dibagi menjadi dua bagian yang dibentuk oleh masalah alokasi lokasi (LAP) dan masalah perutean kendaraan (VRP) dalam lingkungan yang dinamis. Untuk menangani LAP, algoritma pengelompokan digunakan untuk

menangani lokasi node dan kota-kota sekitarnya di setiap kelas. skema imigran digunakan untuk mengatasi masalah routing lokasi dinamis (DLRP) dengan *Ant Colony* with clustering (KACO), dari hasil penelitian didapatkan hasil bahwa dibandingkan dengan algoritma ACO biasa seperti AS, EAS, MMAS, ACS, P-ACO, SA dan GA, KACO menghasilkan performa yang lebih baik.

Penelitian ini menggunakan pendekatan yang dilakukan oleh Gao dkk (2016) untuk mengatasi masalah pemilihan rute perjalanan dinamis dan pembelajaran mendalam dalam mendeteksi kecelakaan lalu lintas dari data media sosial seperti yang dilakukan oleh Hasan dkk (2012) menggunakan data *Twitter* seputar kejadian di jalan dengan metode *Support Vector Machine*. Data berbasis waktu tersebut dilakukan pembobotan dengan metode SAW sehingga didapatkan nilai visibilitas sebagai dasar perhitungan simpul probabilitas $P_{ij}^{k}$ (t) untuk menentukan rute tercepat secara real time dengan algoritma *Ant Colony* dan diaplikasikan pada masalah transportasi mobil pemadam kebakaran.

## 2.2. Dasar Teori

### 2.2.1. *Twitter Application Programming Interface* (API)

Untuk mempermudah interaksi antara pengembang dan layanan, *Twitter* telah menyediakan API yang bisa diakses secara publik melalui internet. API tesebut digunakan sebagai lisensi untuk mengunduh dan menggunakan data *Twitter*.

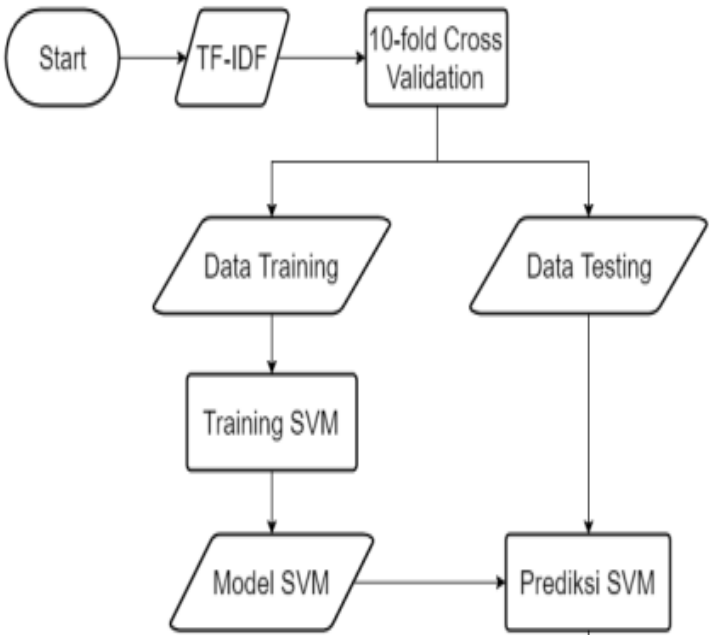
Salah satu aspek penting dari *Twitter* adalah karakteristiknya yang bersifat historis. Sebagai contoh, saat terjadi kemacetan atau kecelakaan, banyak pengguna mengirim jawaban ke *Twitter* (*tweet*) yang berhubungan dengan dua kejadian tersebut. Hal ini memungkinkan dilakukan deteksi terhadap kejadian dengan menginspeksi berbagai *tweet* yang masuk ke *Twitter*.

### *2.2.2. Support Vector Machine*

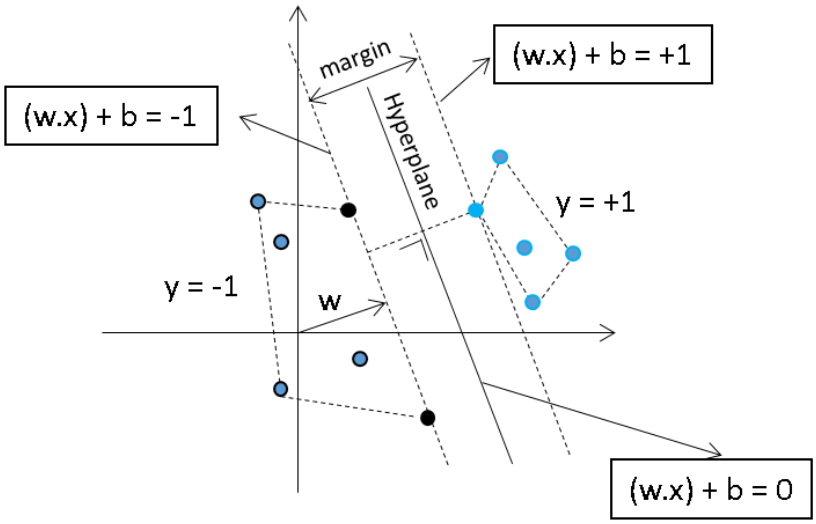
*Support Vector Machine* (SVM) merupakan salah satu metode klasifikasi dengan menggunakan *machine learning (supervised learning*) yang memprediksi

kelas berdasarkan model atau pola dari hasil proses *training*. Klasifikasi dilakukan dengan mencari *hyperplane* atau garis pembatas (*decision boundary*) yang memisahkan antara suatu kelas dengan kelas lain, yang dalam kasus ini garis tersebut berperan memisahkan *tweet* bersentimen positif (berlabel +) dengan *tweet* bersentimen negatif (berlabel -) (Han dkk., 2006). Dalam SVM ini yang pertama adalah pelatihan (*training*) yaitu mempelajari pola-pola teks pada *tweet* menjadi model, sesuai acuan yang diberikan, kemudian langkah kedua yaitu prediksi (*testing*) yaitu menggunakan model untuk mengklasifikasikan teks dan hasilnya merupakan prediksi dari klasifikasi. Pada rancangannya, *tweet* dan label dibagi menjadi data *training* dan *testing* seperti pada Gambar 2.1. Kemudian setelah dilakukan pelatihan, dilanjutkan dengan proses prediksi yang mana akan menghasilkan akurasi sistem. Gambar 2.2 menunjukkan *Hyperplane* terbentuk diantara class-1 dan +1, dimana

$$\text{Margin} = \frac{2}{\|\mathrm{w}\|} = \frac{2}{\sqrt{w_1^{\,2} + w_2^{\,2}}} \tag{2.1}$$




Gambar 2.1 Klasifikasi dengan *Support Vector Machine* (Han dkk., 2006)




Gambar 2.2 *Hyperplane* terbentuk diantara kelas -1 dan +1 (Sanger dkk., 2007)

### 2.2.3. Metode SAW

Metode *Simple Additive Weighting* (SAW) sering juga dikenal istilah metode penjumlahan terbobot. Konsep dasar metode SAW adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut. Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada. Formula untuk melakukan normalisasi tersebut ditampilkan pada persamaan 2.2 sebagai berikut:

$$r_{ij} = \begin{cases} \dfrac{x_{ij}}{\underset{i}{Max}\, x_{ij}} & \textit{jika } \text{j adalah atribut keuntungan (benefit)} \\ \dfrac{\underset{i}{Min}\, x_{ij}}{x_{ij}} & \text{jika j adalah atribut biaya (cost)} \end{cases} \tag{2.2}$$

dimana $r_{ij}$ adalah *rating* kinerja ternormalisasi dari alternatif Ai pada atribut C*j*; i=1,2,...,m dan j=1,2,...,n. Nilai preferensi untuk setiap alternatif (Vi) diberikan oleh persamaan 2.3 sebagai berikut:

$$V_i = \sum_{j=1}^{n} w_j r_{ij} \tag{2.3}$$

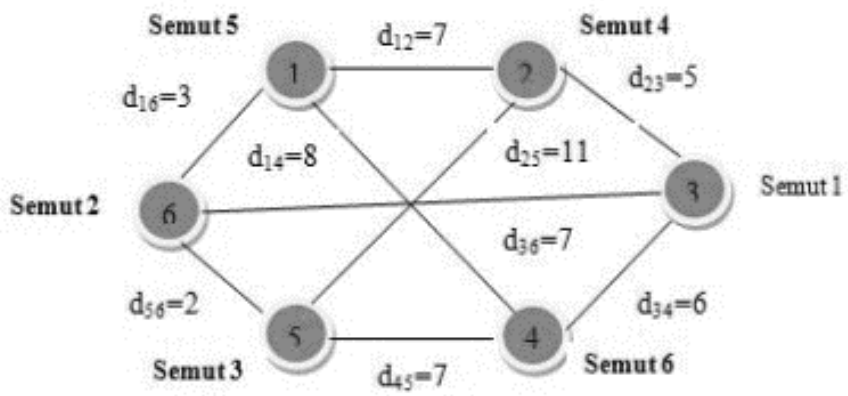
nilai Vi yang lebih besar mengindikasikan bahwa alternatif (Vi) tersebut lebih dipilih.

### 2.2.4. Algoritma *Ant Colony*

Algoritma *Ant Colony* adalah algoritma yang terinspirasi oleh perilaku mencari makan koloni semut. Semut menggunakan zat kimia khusus yang disebut feromon untuk berkomunikasi dan bertukar pengetahuan selama individu, dan antara individu dan lingkungan. Algoritma *Ant Colony* adalah algoritma adaptif kontinyu karena mereka dapat mentransfer informasi yang berguna dari lingkungan masa lalu dan beradaptasi dengan perubahan dinamis. Algoritma awal dikembangkan oleh Dorigo dan Colorni, yang disebut sistem semut, dan pertama kali diterapkan pada masalah perjalanan salesman (TSP) pada tahun 1991 (Dorigo dkk., 1997). Secara umum algoritma ACO untuk TSP mengikuti skema algoritma (Dorigo dkk., 2007) sebagai berikut:

```
procedure ACO Algorithm for TSPs
    Penetapan parameters, initialize pheromone trails
        while (termination condition not met) do
            ConstructAntSolutions
            ApplyLocalSearch (optional)
            Updatepheromones
        Endwhile
    End procedure ACO Algorithm for TSPs
```

Pada inisialisasi, misalkan sejumlah m = 6 ekor semut diletakkan secara acak pada n = 6 simpul pada graph tersebut. Masukkan setiap simpul awal dari semua semut ke dalam tabu list masing-masing semut. Dalam hal ini tabu1 = {3}, tabu2 = {6}, tabu3 = {5}, tabu4 = {2}, tabu5 = {1}, dan tabu6 = {4}. Intensitas jejak *pheromone* untuk semua busur diisi dengan konstanta, $r_{ij}(t) = c$, artinya semua busur memiliki intensitas sama karena memang belum ada seekor semutpun yang melewati busur.




Gambar 2.3**.** Inisialisai 6 ekor semut diletakkan secara acak

Semut (k) akan akan bergerak dari node (i) ke (j) dengan probabilitas $P_{ij}^{k}(t)$ yaitu:

$$P_{ij}^{k}(t) = \left\{\frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum[\tau_{ik}(t)]^{\alpha}[\eta_{ik}]^{\beta}}\right\} \quad (2.4)$$

dimana $\tau_{ij}$ adalah jumlah *pheromone* yang terdapat pada *edge* antara titik i dan titik j, $\eta_{ij}$ adalah *visibility*, α adalah sebuah parameter yang mengontrol bobot (weight) relatif dari *pheromone* dan β adalah parameter pengendali jarak ( α > 0 dan β > 0 ). Dalam memilih node selanjutnya (s) digunakan persamaan 2.5

$$s = argmax_{ucj_i^k}\{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}\} \quad (2.5)$$

Selanjutnya intensitas *pheromone* global di semua busur di-update menggunakan persamaan 2.6 sebagai berikut

$$\tau_{ij}(t+n) = (1-\xi)\tau_{ij} + \xi\tau_{ij}^{k} \quad (2.6)$$

dengan $\xi$ adalah tingkat evaporasi *pheromone* lokal, dan n adalah siklus semut. Selanjutnya intensitas *pheromone* lokal di semua busur di-update menggunakan persamaan 2.7

$$\tau_{ij}(t+n) = \rho\tau_{ij}(t) + \Delta\tau_{ij}^{k} \quad (2.7)$$

dengan $\rho$ adalah tingkat evaporasi *pheromone* global, dan $\Delta\tau_{ij}^{k} = Q/L_k$ adalah *pheromone* yang dikeluarkan oleh semut k dengan $L_k$ adalah panjang jalur semut k dan Q adalah konstanta.

### 2.2.5. *Elitis Ant Colony*

Elitis adalah pengembangan dari *Ant Colony*. Penambahan intensitas *pheromone* dari *tour terbaik* $T_{bs}$ adalah dengan memberi penambahan *quantity* $\frac{e}{C^{bs}}$ untuk setiap *edge*, dimana e parameter yang diberikan untuk mendefinisikan nilai *tour* terbaik ($T_{bs}$) dan $C_{bs}$ adalah panjang *tour* terbaik. Perubahan *pheromone* didefinisikan sebagai berikut :

$$\tau_{ij}(t+n) = \rho\tau_{ij}(t) + \Delta\tau_{ij}^{k} + \Delta\tau_{ij}^{bs} \qquad (2.8)$$

dimana $\Delta\tau_{ij}^{bs}$ didefinisikan sebagai berikut :

$$\Delta\tau_{ij}^{bs} = \begin{cases} \frac{1}{C^{bs}}, & jika \quad edge(r,s) \quad terdapat \quad pada \quad T^{bs} \\ 0, & yang \quad lainnya \end{cases} \qquad (2.9)$$

Kelebihan – kelebihan Elitis adalah sebagai berikut :

1. Jumlah iterasi yang dilakukan lebih sedikit, hal ini karena *edge – edge* pada *tour* terbaik mendapatkan penambahan *pheromone* lebih banyak dan menyebabkan ruang pencarian lebih sempit.
2. Ruang pencarian solusi lebih sempit, sehingga lebih cepat dalam menemukan solusi. Walaupun ruang pencarian lebih sempit tetapi Elitis tidak memperlihatkan terjadinya stagnasi, maksudnyasemut terus mencari kemungkinan adanya *tour* baru yang lebih baik.

### 2.2.6. MAX – MIN *Ant System* (MMAS)

MAX-MIN *Ant System* (MMAS) merupakan pengembangan dari algoritma *Ant Colony* selanjutnya, dengan beberapa perubahan utama sebagai berikut :

1. Penambahan *pheromone* bisa dilakukan pada *edge – edge* yang merupakan bagian dari *tour* terbaik yang ditemukan sejak awal algoritma (*best so-far-tour*) atau pada *tour* terbaik yang ditemukan pada iterasi tersebut (*iteration*

*best-tour*). Bisa juga penambahan *pheromone* pada keduanya, *best so-far-tour* dan *iteration best-tour* sekaligus.

2. Untuk mengatasi masalah pada perubahan pertama, maka MMAS memberikan batasan dalam pemberian nilai *pheromone* dengan interval [$\tau$ min ,$\tau$ max] .
3. Menginisialisasi *pheromone* dengan batas atas nilai *pheromone*, yang mana bersama dengan tingkat evaporasi *pheromone* yang kecil, meningkatkan eksplorasi *tour* sejak dimulainya pencarian.
4. *pheromone* diinisialisasi kembali pada saat terjadi stagnasi atau ketika tidak ditemukan *tour* yang sesuai dengan iterasi yang diinginkan.

Kelebihan – kelebihan MMAS adalah sebagai berikut :

1. Adanya batas nilai *pheromone* membuat pencarian tidak mengalami stagnasi dan bisa lebih fokus.
2. Penginisialisasian kembali membuat performa *Algoritma* menjadi lebih baik. Pada jumlah titik masalah yang besar akan membuat pencarian solusi lebih mudah dan tidak menimbulkan stagnasi. Karena, pada waktu terjadi stagnasi maka akan dilakukan inisialisasi kembali.