BAB 2

LANDASAN TEORI

2.1 Clickbait

Clickbait merupakan judul yang menggoda pembaca yang biasanya menggunakan bahasa provokatif untuk menarik perhatian (Anand, dkk, 2017). Sementara itu, dalam kamus besar Merriam-Webster, *Clickbait* diartikan sebagai judul artikel berita yang dirancang untuk membuat pembaca ingin mengeklik tautan yang ada di media *online* (Merriam-Webster, 2021). Biasanya judul artikel yang bersifat *clickbait* berfokus pada subyek selebritis, rumor, akun fiktif daripada isu-isu yang serius atau lebih akademik (Pengnate, 2016). Sebuah judul dapat dikatakan menggunakan unsur *clickbait* jika memenuhi empat unsur yang telah digabungkan oleh M Rizky Kertanegara pada tahun 2018, sebagai berikut:

1. Menggunakan kalimat atau frasa tanya, seperti ‘Tahukah Anda?...’.
2. Menggunakan kalimat atau frasa seruan, seperti ‘Wow!’, ‘Keren!’, dan ‘Luar Biasa!’.
3. Menggunakan *listicle*, yang merupakan istilah untuk memulai judul berita dengan nomor yang diikuti dengan kata benda dan kata sifat sebagai pesan penggoda yang sensasional, seperti ‘3 tempat wisata alam paling romantis di Indonesia’.
4. Menggunakan frasa catafora, yaitu frasa yang ditandai dengan penggunaan kata ‘ini’ yang menunjukkan waktu, tempat, atau situasi seperti ‘berita ini akan menghebohkan pikiran anda’, ‘nama-nama ini sering menjadi lelucon’.

2.2 Text Classification

Text classification adalah sebuah pekerjaan penting yang digunakan pada banyak aplikasi. *Text classification* dapat dilakukan dengan cara memberikan label pada dokumen-dokumen yang sudah diberi label sebelumnya. Langkah pertama dalam melakukan *text classification* adalah merepresentasikan dokumen sebagai vektor dalam sebuah *high-dimension vector*, dimana masing-masing dimensi menyesuaikan kepada nilai dari sebuah fitur. Kemudian vektor tersebut digunakan oleh sistem untuk melakukan klasifikasi dokumen ke dalam kategori yang benar (Nedjah, dkk, 2009).

2.3 Preprocessing

Pada tahap *preprocessing*, terdapat beberapa tahapan untuk mengolah data agar data tersebut dapat menjadi lebih terstruktur. Hal ini perlu dilakukan agar data dapat dibaca oleh sistem dengan baik. Tahapan-tahapan yang diperlukan untuk *preprocessing* dalam penelitian ini adalah sebagai berikut:

1. Case Folding

   Tahap ini dilakukan karena penggunaan huruf kapital atau sejenisnya terkadang tidak mempunyai kesamaan. *Case folding* bertujuan untuk mengubah semua dokumen teks menjadi huruf kecil (Pernanda, 2019).

2. Tokenizing

   Pada tahap ini, semua teks yang telah melewati tahap *case folding* akan dipecah menjadi beberapa bagian, satu bagian berisi satu kata dari teks yang digunakan (Yunus, 2020).

3. Filtering

Filtering adalah tahap mengambil kata-kata yang penting dari hasil tahap *tokenizing* dengan menggunakan algoritma *stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata yang penting) (Nugroho, 2019).

4. Stemming

Tahap *Stemming* berfungsi untuk memperkecil jumlah indeks yang berbeda dari suatu dokumen teks dan melakukan pengelompokan kata-kata lain yang memiliki kata dasar dan arti yang serupa namun memiliki bentuk yang berbeda karena imbuhan yang berbeda (INFORMATIKALOGI, 2020).

2.4 Term Frequency-Inverse Document Frequency

Term Frequency-Inverse Document Frequency (TF-IDF) adalah metode yang digunakan untuk menghitung bobot dari setiap kata yang telah diekstrak. Model pembobotan TF-IDF merupakan metode yang mengintegrasikan model Term Frequency (TF) dan Inverse Document Frequency (IDF). Term Frequency (TF) merupakan proses untuk menghitung jumlah kemunculan term dalam satu dokumen (judul berita) dan Inverse Document Frequency (IDF) digunakan untuk menghitung term yang muncul di keseluruhan dokumen (judul berita) yang dianggap sebagai term umum, yang dinilai tidak penting (Luqyana, dkk, 2018).

2.5 Extreme Gradient Boosting

Algoritma Extreme Gradient Boosting merupakan salah satu algoritma yang populer dan paling banyak digunakan karena algoritma ini termasuk sebagai algoritma yang *powerful*. Algoritma ini adalah kombinasi antara algoritma

Gradient Descent dan Boosting yang disebut dengan Gradient Boosting Machine. Extreme Gradient Boosting pertama kali dikenalkan pada *Higgs Boson Competition*, dimana pada kompetisi ini algoritma Extreme Gradient Boosting menjadi algoritma yang paling banyak digunakan. Algoritma Extreme Gradient Boosting melakukan optimasi 10 kali lebih cepat dibandingkan dengan implementasi dari Gradient Boosting Machine lainnya (Chen & Guestrin, 2016).

Algoritma Extreme Gradient Boosting merupakan implementasi lanjutan dari algoritma Gradient Boosting. Extreme Gradient Boosting menggunakan prinsip *ensemble* yaitu menambahkan model baru untuk memperbaiki model sebelumnya yang masih memiliiki kesalahan sampai model tidak memiliki kesalahan lagi (Islam, dkk, 2021). Prinsip *ensemble* juga menggabungkan beberapa *tree* yang lemah menjadi sebuah model yang kuat sehingga menghasilkan prediksi dengan tingkat akurasi yang tinggi (Syahrani, dkk, 2019).

Prediksi dilakukan dengan cara menjumlahkan semua nilai yang terdapat pada setiap *tree* lalu memasukkan nilai tersebut ke dalam fungsi logistik. Algoritma Extreme Gradient Boosting akan meminimumkan fungsi objektif yang dapat dilihat pada persamaan 2.1 (Syukron, dkk, 2020).

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \qquad ...(2.1)$$

Dimana n merupakan jumlah *tree* yang ingin digunakan (dapat ditetapkan menggunakan variabel n_estimators), l merupakan fungsi untuk mengukur perbedaan antara prediksi target y_i dan $\hat{y}_i$, $f_t(x_i)$ merupakan model baru yang dibangun, Sedangkan Ω adalah fungsi untuk membuat model yang dapat

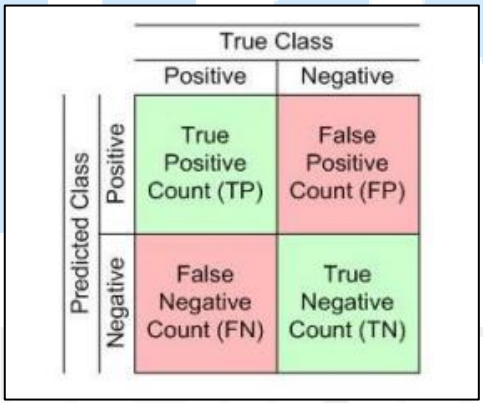
menghindari *overfitting*. Nilai *gain* bisa didapatkan untuk penentuan *splitting node* atau pembagian cabang dalam model *tree*. Persamaan 2.2 merupakan rumus untuk mencari nilai *gain* (Syukron, dkk, 2020).

$$L_{split} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma \qquad ...(2.2)$$

Tiga suku dari persamaan masing-masing mewakili *leaf* kiri, leaf kanan, dan *leaf* asli. IL dan IR merupakan node kiri dan node kanan setelah dibagi, λ dan γ merupakan koefisien, dengan nilai default yang telah ditetapkan untuk λ = 1 dan γ = 0, nilai g_i dan h_i merupakan turunan pertama dan kedua *loss function* pada algoritma Extreme Gradient Boosting (Syukron, dkk, 2020).

2.6 Metode Evaluasi

Metode evaluasi yang digunakan untuk melakukan evaluasi performa dari algoritma Extreme Gradient Boosting adalah Confusion Matrix. Contoh dari Confusion Matrix dapat dilihat pada gambar 2.



Gambar 2. 1 Confusion Matrix
(Syukron, dkk, 2020)

Pada gambar 2 terdapat 4 nilai yaitu True Positive (data positif yang diprediksi dengan benar), False Positive (data positif yang diprediksi dengan

salah), False Negative (data negatif yang diprediksi dengan salah), dan True Negative (data negatif yang diprediksi dengan benar). Dengan menggunakan 4 nilai tersebut, nilai akurasi *recall* bisa didapatkan. Nilai akurasi merupakan keakuratan prediksi secara keseluruhan, nilai *recall* merupakan perbandingan prediksi benar negatif dibandingkan dengan keseluruhan hasil yang diprediksi negatif, nilai *precision* merupakan perbandingan prediksi benar negatif dibandingkan dengan keseluruhan data yang benar negatif, dan nilai *F1 Score* merupakan perbandingan antara nilai rata-rata *recall* dan *precision.* Nilai akurasi dan *recall* dapat dihitung menggunakan rumus yang ada di persamaan 2.3, 2.4, 2.5, dan 2.6 (Syukron, dkk, 2020).

$$Akurasi = \frac{TP + TN}{TP + FP + TN + FN} \quad ...(2.3)$$

$$Recall = \frac{TN}{TN + FP} \quad ...(2.4)$$

$$Precision = \frac{TN}{TN + FN} \quad ...(2.5)$$

$$F1\ Score = \frac{2 * (Recall * Precision)}{Recall + Precision} \quad ...(2.6)$$