BAB 2

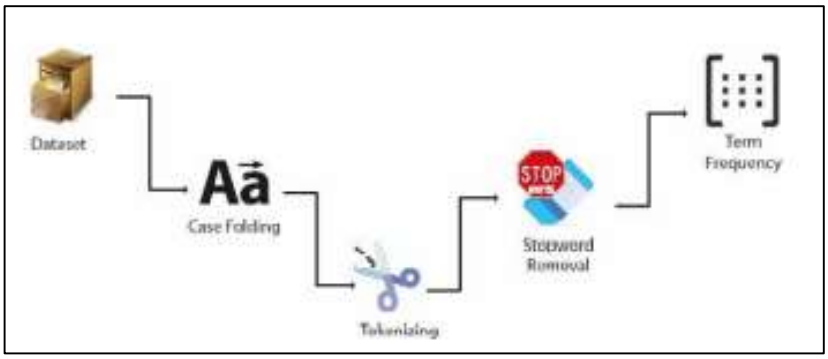
LANDASAN TEORI

2.1 Hoax

Hoaks merupakan berbagai rangkaian informasi yang memang sengaja disesatkan, tetapi "dijual" sebagai kebenaran (Silverman, 2015). Sedangkan Werme mendefiniskan hoaks atau *fake news* sebagai berita palsu yang mengandung informasi yang sengaja menyesatkan orang dan memiliki agenda politik tertentu (Aditiawarman dkk., 2019). Hoaks bukan hanya sekadar *misleading* alias menyesatkan, informasi dalam berita palsu juga tidak memiliki landasan faktual, tetapi disajikan seolah-olah sebagai serangkaian fakta (Athaillah dkk., 2020).

2.2 Praproses Teks

Praproses teks diperlukan dalam penelitian *natural language processing* (NLP) untuk mengubah data yang awalnya tidak terstruktur menjadi terstruktur. Praproses teks merupakan proses pembersihan dan mengodekan data menjadi nilai numerik sebelum diberikan untuk model pembelajaran mesin (Nugroho, 2019). Data uji dan data latih akan melalui tahap praproses berupa *case folding, tokenizing, stopword removal,* dan *term frequency*. Praproses teks dilakukan untuk memperoleh matriks *term* dan frekuensi *term*. Berikut merupakan diagram tahapan praproses teks.



Gambar 2.1 Tahap praproses teks
(Rahutomo dkk., 2019)

2.2.1 Case Folding

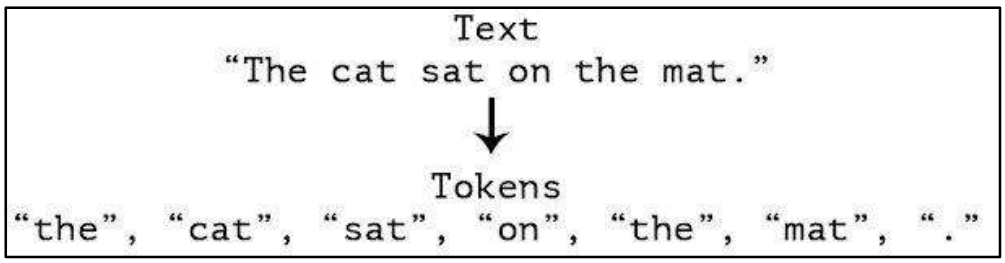
Case folding merupakan salah satu praproses teks yang paling sederhana dan efektif (Nugroho, 2019). *Case folding* bertujuan untuk mengubah seluruh huruf dalam dokumen menjadi huruf kecil. Karakter lain selain huruf 'a' hingga 'z' dianggap sebagai *delimiter* dan dihilangkan. Berikut ini merupakan uraian proses *case folding* (Nugroho, 2019).

1. Seluruh teks dalam dokumen diubah menjadi huruf kecil (*lowercase*)
2. Angka dihapus bila tidak relevan dengan penelitian.
3. Tanda baca atau simbol dihapus.
4. Karakter kosong (*whitespace*) dihapus.

2.2.2 Tokenizing

Tokenizing merupakan proses memotong kalimat menjadi bagian-bagian yang disebut token (Nugroho, 2019). Pada saat *tokenizing*, tanda baca dianggap sebagai pemisah kata atau *delimiter* dan tidak berpengaruh akan proses teks. *Token*

diartikan sebagai turunan dari urutan karakter dalam dokumen yang dikelompokkan sebagai unit semantik untuk selanjutnya diproses (Rahutomo dkk., 2019). Berikut merupakan ilustrasi dari pemisahan kalimat menjadi *token*.



Gambar 2.2 Ilustrasi tahap tokenisasi
(Mayo, 2020)

2.2.3 Stopword Removal

Filtering atau *stopword removal* merupakan proses pengambilan kata penting dari hasil tokenisasi dengan algoritma *stoplist* atau *wordlist* (Nugroho, 2019). Algoritma *stoplist* bertujuan untuk membuang kata kurang penting, sedangkan algoritma *wordlist* menyimpan kata penting. *Stopword* adalah kata-kata umum yang sering muncul dan dianggap tidak memiliki makna. Contoh stopword Bahasa Indonesia yaitu "yang", "di', "dan", "dari", dll. Dengan menghapus *stopword* yang dianggap memiliki informasi rendah, diperoleh kata-kata yang penting. Penggunaan *stopword list* dapat secara signifikan mengurangi jumlah kata yang harus disimpan dalam basis data (Rahutomo dkk., 2019).

2.2.4 TF-IDF

TF-IDF adalah metode *term frequency* yang menggunakan konsep *inverse document frequency*. TF-IDF menyimpan nilai evaluasi kata yang ada pada suatu

kalimat dan membandingkannya dengan seluruh kata pada dokumen (Madan, 2019). Maka, TF-IDF digunakan untuk memperoleh skor relevansi tiap kata dalam dokumen. Berikut merupakan rumus perhitungan TF-IDF (Madan, 2019).

$$TF = \frac{Number\ of\ repetitions\ of\ word\ in\ a\ document}{Number\ of\ words\ in\ a\ document} \quad (2.1)$$

$$IDF = Log(\frac{Number\ of\ documents}{Number\ of\ documents\ containing\ the\ word}) \quad (2.2)$$

Berikut merupakan hasil yang akan diperoleh setelah data melalui proses TF-IDF (Madan, 2019).

1. Seberapa penting sebuah kata dalam kalimat.
2. Seberapa penting sebuah kata dalam dokumen berdasarkan frekuensi kemunculannya.
3. Mengabaikan kata yang salah eja dengan teknik n-gram.

2.3 Algoritma Naïve Bayes

Algoritma Naïve Bayes merupakan salah satu metode untuk klasifikasi yang menggunakan probabilitas dan statistik yang dikemukakan oleh seorang ilmuwan Inggris bernama Thomas Bayes. Algoritma ini digunakan untuk memprediksi peluang yang akan terjadi di masa depan berdasarkan pengalaman atau hal yang telah terjadi sebelumnya. Ciri utama yang membedakan Naïve Bayes adalah asumsi yang begitu kuat (naif) akan independensi dari masing masing kondisi atau kejadian.

Performa Naïve Bayes bekerja dengan sangat baik jika dibandingkan dengan model classifier yang lain. Hal ini telah dibuktikan pada beberapa penelitian, seperti yang tertulis pada jurnal yang ditulis oleh Xhemali dkk. (2009) dengan judul "Naive Bayes vs. Decision Trees vs. Neural Networks in the classification of training web pages" yang menyimpulkan bahwa Naïve Bayes Classifier memiliki tingkat akurasi yang lebih baik dibandingkan model lainnya.

Algoritma Naive Bayes hanya membutuhkan data latih yang sedikit untuk dapat menentukan estimasi yang diperlukan dalam proses klasifikasi. Hanya varian dari suatu variabel yang dibutuhkan dari sebuah kelas untuk menentukan klasifikasi. Berikut merupakan tahapan dari algoritma Naïve Bayes (Syarli dan Muin, 2016).

1. Hitung jumlah kelas atau label.
2. Hitung jumlah kasus per kelas.
3. Kali semua variabel kelas.
4. Bandingan hasil per kelas.

Adanya keterkaitan antar atribut dapat mengurangi akurasi dalam asumsi *independence*. Namun, algoritma Naïve Bayes dinilai dapat menghasilkan hasil yang bagus dalam klasifikasi dokumen dan *spam filtering*. Selain itu, Naïve Bayes jauh lebih cepat dibandingkan dengan metode lainnya. Pada pembuatan *ensemble model* untuk penelitian ini, digunakan 3 jenis metode Naïve Bayes yaitu Gaussian Naïve Bayes, Multinomial Naïve Bayes, dan Complement Naïve Bayes.

2.3.1 Algoritma Gaussian Naïve Bayes

Gaussian Naïve Bayes biasa diimplementasikan dalam klasifikasi teks dengan fitur bernilai kontinu (Monika, 2020). Saat mengaplikasikan Gaussian Naïve Bayes, nilai *likelihood* pada fitur yang ada diasumsikan sebagai *gaussian* (Zhang, 2004). Distribusi *gaussian* merupakan cara paling umum untuk mengasumsikan nilai dari data yang bersifat kontinu (Gao, 2020). *Gaussian* atau distribusi normal dianggap sebagai cara yang paling mudah untuk digunakan untuk memperkirakan distribusi data karena hanya perlu memperkirakan nilai *mean* dan *standard deviation* dari *training data*.

2.3.2 Algoritma Multinomial Naïve Bayes

Multinomial Naïve Bayes diimplementasikan pada data yang terdistribusi secara multinomial dan biasa digunakan dalam klasifikasi teks (Scikit-learn, 2019). Multinomial Naïve Bayes digunakan untuk menentukan apakah suatu dokumen merupakan bagian dari sebuah kategori. Fitur yang digunakan pada *classifier* ini yaitu frekuensi kemunculan kata dalam dokumen (Monika, 2020). Berikut merupakan rumus menghitung probabilitas dokumen d bagian dari kelas c.

$$P(c|d) \propto P(c) \prod_{1 \le k \le n_d} P(t_k|c) \tag{2.3}$$

Diketahui:

$P(t_k|c)$: kemungkinan munculnya t_k dalam dokumen kelas c.

$P(c)$: probabilitas awal sebuah dokumen ada pada kelas c.

n_d: jumlah token pada dokumen d.

Kompleksitas waktu uji dan waktu latih yang dibutuhkan untuk melakukan *scan* data adalah linear. Karena diperlukan *scan* data minimal satu kali, Naïve Bayes dinilai memiliki *time complexity* yang optimal. Hal ini yang menyebabkan Naïve Bayes sering digunakan untuk permasalahan klasifikasi teks (Cambridge University Press, 2008).

2.3.3 Algoritma Complement Naïve Bayes

Complement Naïve Bayes (CNB) merupakan adaptasi dari Multinomial Naïve Bayes (MNB) yang sesuai untuk digunakan pada *imbalanced datasets*. Secara spesifik, CNB menggunakan statistik dari nilai komplemen setiap kelas untuk menghitung *weight* dari model. Berikut merupakan cara kerja Complement Naïve Bayes (Rennie, et al., 2003).

1. Pada setiap kelas, hitung seberapa besar probabilitas suatu instance bukan bagian dari kelas.
2. Pilih nilai terkecil dari hasil perhitungan (1).

Nilai probabilitas terkecil dipilih karena memiliki kemungkinan tertinggi bahwa sebenarnya *instance* yang dihitung adalah bagian dari kelas yang dipilih (Alokesh, 2020).

2.4 Algoritma Logistic Regression

Logistic Regression merupakan model statistik yang dalam bentuk dasarnya menggunakan fungsi logistik untuk memodelkan variabel biner (Tolles dan Meurer, 2016). Model logistik biner memiliki variabel dependen dengan dua kemungkinan nilai, pada penelitian ini berupa valid / hoaks yang diwakili oleh label ‘1’ dan ‘0’.

Logaritma peluang nilai dengan label '1' merupakan kombinasi linier dari variabel independent yang disebut prediktor. Fungsi logistic mengubah logaritma peluang menjadi probabilitas. Model Logistic Regression hanya memodelkan probabilitas output dan tidak melakukan klasifikasi secara statistik, meski dapat digunakan untuk membuat *classifier* dengan memilih nilai *cutoff* dan mengklasifikasikan input dengan probabilitas (Walker dan Duncan, 1967). Logistic Regression dapat diaplikasikan dalam berbagai bidang, termasuk pembelajaran mesin. Dalam pembelajaran mesin, Logistic Regression digunakan untuk menemukan hubungan antara fitur dan probabilitas prediksi (Agrawal, 2017). Model logistik seringkali digunakan sebagai *meta learner* dalam ensemble learning masalah klasifikasi karena cukup sederhana dan mampu menghasilkan intrepretasi dari prediksi yang dibuat oleh *base learners* (Brownlee, 2020).

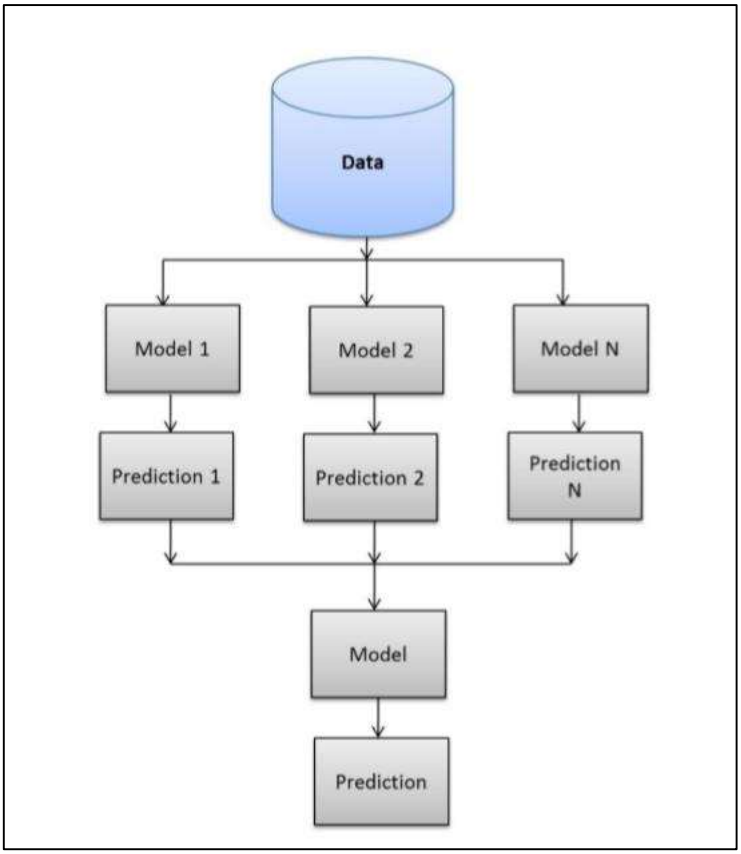
2.5 Ensemble Learning Menggunakan Teknik Stacking

Stacking adalah metode pembelajaran *ensemble* yang menggabungkan beberapa algoritma pembelajaran mesin melalui *meta learning*, dimana algoritma tingkat dasar dilatih menggunakan *training dataset* lengkap dan kemudian hasil akhir dari semua model tingkat dasar dijadikan sebagai fitur (Patel, 2020). Dengan menggabungkan beberapa jenis model yang berbeda, metode *stacking* diharapkan dapat menghasilkan akurasi yang lebih besar dalam klasifikasi. Gambaran cara kerja stacking dapat dilihat pada Gambar 2.3.

Berikut ini merupakan tahapan cara kerja *stacking* (Charan, 2017).

1. Pisahkan *training data* menjadi 2 *disjoint set*.
2. Latih beberapa *base learner* pada bagian pertama.

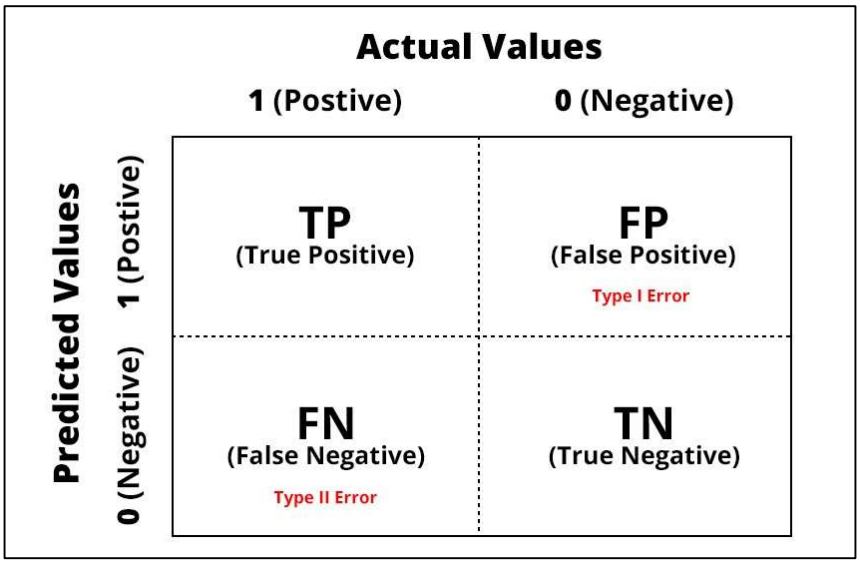
3. Uji *base learner* pada bagian kedua dan buat prediksi.
4. Pakai hasil prediksi dari (3) sebagai input untuk melatih *meta level learner*.



Gambar 2.3 Gambaran cara kerja metode stacking
(Patel, 2020)

2.6 Metrik Evaluasi

Evaluasi dilakukan untuk mengukur hasil kinerja sistem. Untuk mengukur kinerja, dapat digunakan *confusion matrix*. *Confusion matrix* memberi informasi perbandingan dari hasil klasifikasi model dengan data klasifikasi sebenarnya yang diketahui. Berikut ini merupakan diagram *confusion matrix*.



Gambar 2.4 Confusion matrix
(Nugroho, 2019)

Diketahui:

TP: *True Positive* yaitu data positif yang diprediksi benar.

TN: *True Negative* yaitu data negatif yang diprediksi benar.

FP: *False Positive* yaitu data negatif namun diprediksi sebagai data positif.

FN: *False Negative* yaitu data positif namun diprediksi sebagai data negatif.

Confusion matrix digunakan untuk mengukur performa dalam permasalahan klasifikasi karena mampu memberikan informasi tentang TP, FP, FN dan TN. Informasi tersebut dapat digunakan untuk menghitung *performance metrics* untuk mengukur kinerja model. Beberapa *performance metrics* yang umum digunakan yaitu *precision*, *recall, dan f1-score*. *Precision* digunakan untuk mengukur seberapa akurat antara data dan hasil prediksi model. Sedangkan *recall* menggambarkan tingkat keberhasilan model dalam menemukan kembali sebuah informasi. Berikut merupakan rumus *precision*, *recall* dan *f1-score* (Athaillah dkk., 2020).

1. *Precision* yaitu proporsi kasus dengan hasil positif.

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.4}$$

2. *Recall* yaitu banyaknya kasus dengan hasil positif yang diidentifikasikan dengan benar.

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.5}$$

3. *F1-score* yaitu perbandingan nilai rata-rata *precision* dan *recall*.

$$f1 - score = 2 \text{ x } \frac{\text{Recall x Precision}}{Recall + Precision} \tag{2.6}$$