LAPORAN TUGAS KLASIFIKASI TEKS

Pembuatan Model *Sentiment Analysis* Menggunakan *Traditional Machine Learning Algorithm* dengan *Bag of Words Feature*

IF5153 - Pemrosesan Bahasa Alami

oleh :

Mohammad Rifqi Farhansyah / 13521166

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2024

Daftar Isi

Daftar Gambar

Bab I Penjelasan Kode

Pada tugas ini telah diimplementasikan *script* dari model *sentiment analysis* menggunakan *traditional machine learning algorithm* dengan *feature* berupa *bag of words.* Algoritma yang digunakan harus terdiri atas 3 algoritma yang berbeda. Sementara itu, dataset yang digunakan berasal dari salah satu repositori milik IndoNLU. Dataset dapat diakses melalui pranala berikut ini.

Pada implementasi telah dilakukan beberapa proses, di antaranya *preprocessing* data, *exploratory data analysis, training*, dan lain sebagainya. Secara umum, implementasi tersebut memiliki struktur seperti di bawah ini:

```
.gitignore
MultinomialNaiveBayes_13521166.ipynb
RandomForest_13521166.ipynb
README.md
SVM_13521166.ipynb

├─.ipynb_checkpoints
│   MultinomialNaiveBayes_13521166-checkpoint.ipynb
│   RandomForest_13521166-checkpoint.ipynb
│   SVM_13521166-checkpoint.ipynb
│
├─doc
│   Laporan_13521166.pdf
│
└─smsa_doc-sentiment-prosa
    test_preprocess.tsv
    test_preprocess_masked_label.tsv
    train_preprocess.tsv
    valid_preprocess.tsv
    vocab.txt
    vocab_uncased.txt
```

Setiap implementasi algoritma akan ditandai dengan format file bernama <NamaAlgoritma>*13521166.ipynb,* sementara *doc* merupakan *folder* yang berisi *l*aporan pengerjaan, serta *smsa_doc-sentiment-prosa* merupakan folder yang berisi dataset tugas ini. Kode implementasi program dapat diakses melalui pranala berikut ini.

I.1 Kakas yang Digunakan

Implementasi tugas ini menggunakan beberapa kakas bantuan yang digunakan untuk *task-task* tertentu, diantaranya:

1. sklearn.feature_extraction.text.CountVectorizer digunakan untuk mengubah data teks menjadi matriks (bag-of-words), hasilnya akan merepresentasikan teks sebagai vektor numerik yang kemudian digunakan sebagai input ke model pembelajaran mesin.
2. sklearn.model_selection.GridSearchCV digunakan untuk mencari parameter terbaik (misalnya, nilai alpha dalam MultinomialNB) dengan memvalidasi silang berbagai kombinasi parameter dan memilih yang memberikan performa terbaik berdasarkan skor evaluasi (misalnya F1-score).
3. nltk.tokenize.word_tokenize digunakan untuk memisahkan teks menjadi kata-kata (tokenisasi), sehingga teks dapat diproses lebih lanjut untuk analisis atau ekstraksi fitur.
4. nltk.corpus.stopwords digunakan untuk menghilangkan kata-kata umum yang sering muncul tetapi biasanya tidak memiliki banyak arti (seperti "dan", "atau", "tetapi") untuk meningkatkan akurasi model.
5. sklearn.metrics.accuracy_score, precision_score, recall_score, f1_score, classification_report digunakan untuk mengukur kualitas prediksi model pembelajaran mesin, memberikan gambaran tentang bagaimana baik atau buruk model dalam mengklasifikasikan data.
6. collections.Counter digunakan untuk menghitung jumlah kemunculan setiap elemen dalam data, misalnya untuk menghitung frekuensi kata atau emosi.
7. pandas digunakan untuk memanipulasi dan menganalisis data, misalnya untuk memuat, membersihkan, dan memproses dataset.
8. numpy digunakan sebagai dasar untuk operasi numerik seperti manipulasi matriks yang diperlukan untuk model pembelajaran mesin.
9. matplotlib.pyplot digunakan untuk membuat visualisasi seperti grafik batang untuk menganalisis distribusi kesalahan klasifikasi.

10. re berguna untuk menangani ekspresi reguler, misalnya untuk membersihkan teks atau menemukan pola dalam data teks.
11. string menyediakan fungsi untuk memanipulasi string, seperti menghapus tanda baca atau mengubah format teks.
12. warnings digunakan untuk mengelola pesan peringatan dalam kode, misalnya, mengabaikan peringatan yang tidak relevan saat menjalankan eksperimen.
13. NLTK (Natural Language Toolkit) digunakan untuk praproses teks, seperti tokenisasi dan penghapusan stopwords.

I.2 Konfigurasi Dataset dan Pembacaan Data

Dari file dataset yang diberikan, akan terdapat 5 file secara keseluruhan yang digunakan dalam pembangunan implementasi tugas ini. File-file tersebut adalah training data, validation data, testing data (dengan *masking label*), serta testing data (tanpa *masking label*). Selain itu, akan digunakan pula *vocabulary* yang telah menyimpan seluruh token data dalam implementasi ini.

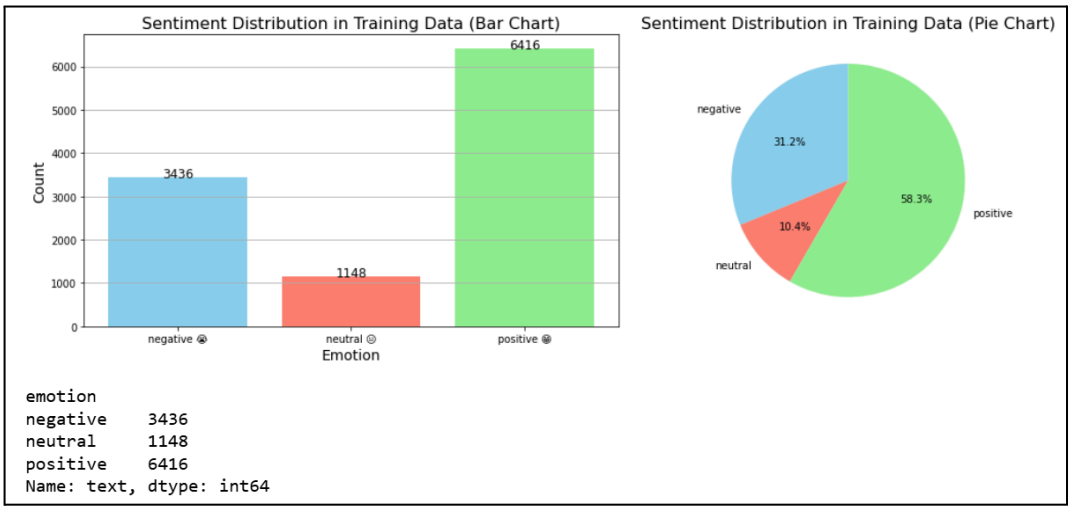
Pada implementasi ini, *load dataset* dilakukan dalam direktori lokal penulis, sehingga untuk melakukan pengujian pada perangkat lain, perlu dilakukan proses pengunduhan dataset ulang dari repositori.

I.3 *PreProcessing*

Penulis telah melakukan pengujian dengan beberapa kombinasi yang hendak dilakukan untuk meningkatkan kualitas model akhir. Beberapa *preprocessing* yang diujikan adalah perubahan karakter ke huruf kecil, tokenisasi, penghilangan huruf berulang, penghilangan angka, penghilangan tanda baca, serta penghapusan *stop words*. Akan tetapi, kombinasi yang menghasilkan nilai metriks evaluasi terbaik adalah elaborasi antara perubahan tiap karakter ke *lowercase* serta tokenisasi.

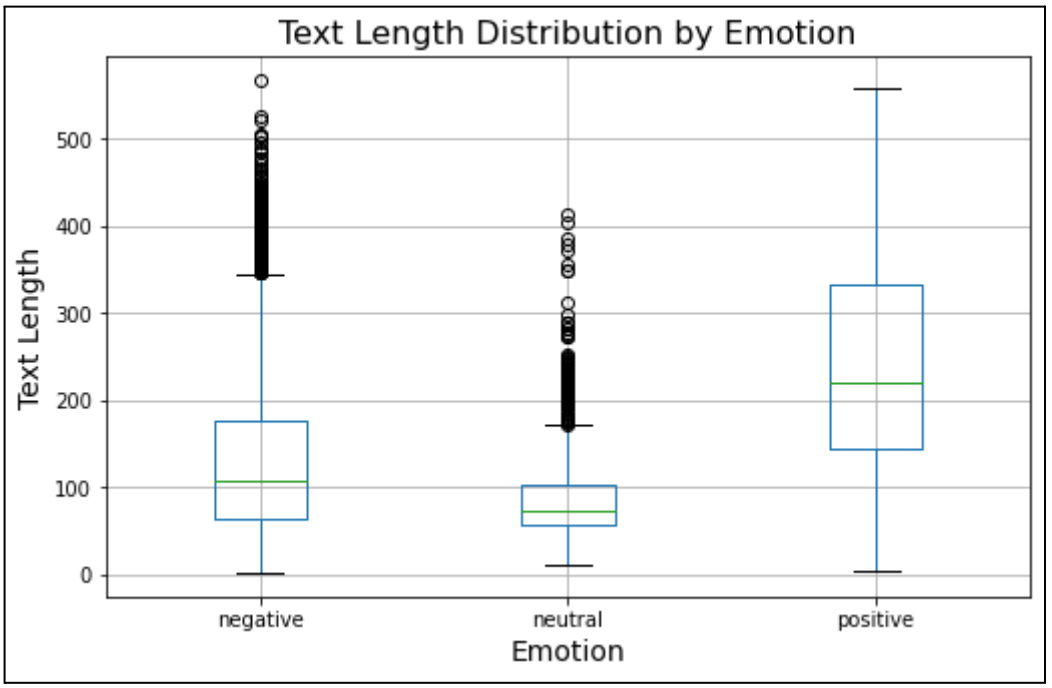
I.4 *Exploratory Data Analysis*

Penulis melakukan beberapa visualisasi dengan maksud tertentu sebelum melatih model yang hendak digunakan. Beberapa visualisasi yang dibuat, antara lain:



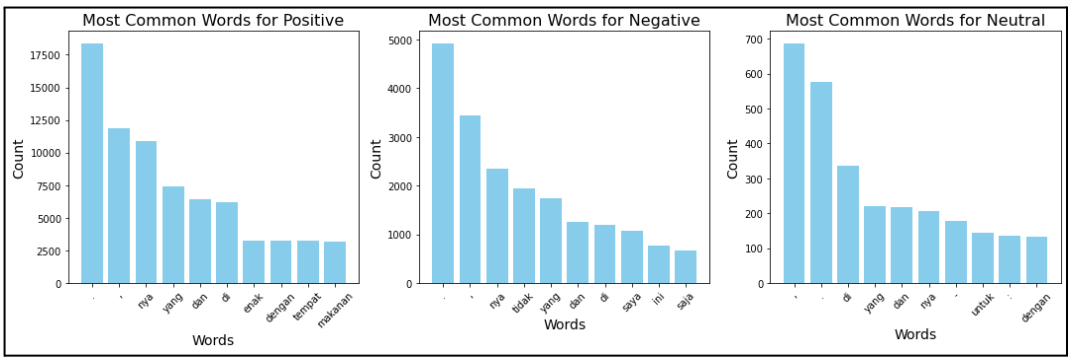
Gambar 1. Ilustrasi Distribusi Label dalam Dataset

Terdapat pula visualisasi yang menggambarkan beberapa *outlier* dalam konteks panjang kalimat masukan.



Gambar 2. Distribusi Panjang Kalimat tiap Emosi

Selain itu, penulis juga memvisualisasikan kata-kata yang paling sering muncul dalam sebuah emosi.



Gambar 3. Kata-kata dengan Frekuensi Kemunculan ter-Tinggi

I.5 *Training Preparation*

Tahap pelatihan diawali dengan menyiapkan data yang hendak digunakan beserta menentukan model mana yang harus dipersiapkan. Kemudian, dilakukan proses *encode label* dari dataset untuk mempermudah representasi dari kalimat konvensional menjadi sebuah bilangan. Dalam waktu bersamaan, akan dilakukan proses pembuatan *Bag of Words* (BoW) melalui penggunaan *CountVectorizer*, dimana setiap token akan digolongkan berdasarkan frekuensi penggunaannya. Vecorizer akan dihilangkan apabila terdapat kasus duplikasi serta dilakukan transformasi terhadap data-data yang tersedia.

Bab II Skenario Eksperimen

Eksperimen dilakukan dalam 3 *source file* berbeda, sesuai dengan penjelasan pada pembukaan Bab II. Eksperimen diwali dengan menggunakan sebuah algoritma bernama SVM untuk kemudian digunakan dalam menemukan *hyperparameter* serta *preprocessing* terbaik. Langkah-langkah yang perlu dilakukan setelah langkah pertama di atas, antara lain:

II.1 Proses *Training*

Setelah dataset berhasil dikumpulkan, akan dilakukan proses pelatihan / *training*. Proses pelatihan untuk tiap *source file* yang berbeda akan juga memiliki penggunaan algoritma yang berbeda. Pada implementasi kode ini, terdapat 3 algoritma yang digunakan, yaitu: Support Vector Machine, Mulinomial Naive Bayes, serta Random Forest. Hasil dari tahap ini selanjutkan akan diperbandingkan satu dengan yang lain menggunakan sebuah metriks evaluasi ttertentu.

II.2 Evaluasi Model

Penilaian tiap model akan didasarkan pada beberapa metriks evaluasi, antara lain: *precision, recall, accuracy,* dan F1-score. Untuk tiap implementasi model, akan pula dicari sebuah kombinasi paramater atau konfigurasi yang menyebabkan sebuah model mampu menghasilkan nilai metriks evaluasi terbaik. Selain perbandingan intramodel terdapat pula perbandingan nilai antara beberapa model, hingga dihasilkan sebuah model terbaik di antara yang lainnya.

II.3 Algoritma yang Digunakan

Berikut merupakan rincian deskripsi dari ketiga algoritma yang digunakan dalam impementasi tugas ini:

1. Multtinomial Naive Bayes

   Multinomial Naive Bayes adalah model yang digunakan khusus untuk data dengan distribusi multinomial, yang umum digunakan dalam kasus

klasifikasi teks. Model ini sangat efektif ketika fitur yang digunakan adalah frekuensi atau probabilitas kemunculan kata, seperti yang diterapkan dalam Bag of Words atau TF-IDF. Pada beberapa eksperimen, dilakukan *tuning hyperparameter* untuk model **Multinomial Naive Bayes** menggunakan **GridSearchCV**. Parameter yang dioptimalkan adalah **alpha**, yang berfungsi sebagai smoothing parameter untuk menghindari masalah pembagian dengan nol dalam pembelajaran. Variasi alpha yang dicoba dalam eksperimen adalah [0.1, 0.5, 0.75, 1.0, 2.0]. Alpha terbaik yang ditemukan dari eksperimen adalah 0.5, yang memberikan skor **f1_macro** terbaik.

2. Support Vector Machine (SVM)

   Support Vector Machine adalah algoritma pembelajaran mesin yang mencari hyperplane terbaik untuk memisahkan kelas dalam ruang fitur. Pada kasus klasifikasi teks ini, SVM digunakan untuk mencari hyperplane yang memisahkan kategori sentimen (seperti positif, netral, negatif). Beberapa algoritma yang digunakan adalah linear, radial basis kernel (RBF), sigmoid, dan poly. Sementara itu, hyperparameter tuning dilakukan menggunakan GridSearchCV dengan parameter utama yang dioptimalkan adalah C (regularization parameter). Nilai C yang besar memungkinkan model untuk lebih "keras" dalam pemisahan data, sedangkan nilai C yang kecil akan membuat model lebih "lunak". Pada eksperimen ini, C dicoba dengan nilai [0.1, 1, 10, 20]. Kombinasi parameter terbaik yang ditemukan adalah C: 10 dan kernel: rbf.

3. Random Forest

   Random Forest merupakan salah satu algoritma *ensemble learning* yang terdiri dari banyak *decision tree.* .Algoritma ini bekerja dengan cara membangun beberapa pohon keputusan (decision tree) dari subset acak data latih, kemudian menggabungkan prediksi dari semua pohon tersebut untuk menghasilkan prediksi akhir berdasarkan **voting** atau **average**. Pada kasus sentiment analysis, Random Forest akan membangun beberapa decision tree berdasarkan fitur-fitur yang diperoleh dari representasi teks seperti **Bag of**

Words. Setiap pohon akan memprediksi kelas sentimen, dan keputusan akhir dibuat berdasarkan **mayoritas suara** dari semua pohon. Random Forest biasanya diuji dengan berbagai jumlah pohon (n_estimators) dan kedalaman pohon (max_depth) untuk menemukan konfigurasi terbaik.

Bab III Hasil Eksperimen

III.1 Hasil Validasi

Berikut adalah tabel skor akurasi, precision, recall, dan F1-Score untuk tiap proses perhitungan metriks evaluasi data validasi eksperimen.

	Parameter Eksperimen	Akurasi	Precision	Recall	F1-Score
MultinomialNB	Alpha 1.0	0.85952	0.85529	0.80796	0.82691
	Alpha 0	0.58253	0.69327	0.49901	0.46449
	Alpha 0.5	0.85793	0.84116	0.82129	0.82973
	Hyperparam Tuning	0.85793	0.84116	0.82129	0.82973
Support Vector Machine	RBF Kernel	0.86349	0.83120	0.81049	0.81873
	Linear Kernel	0.85555	0.81949	0.81092	0.81347
	Poly Kernel	0.68809	0.81350	0.53776	0.48553
	Sigmoid Kernel	0.74285	0.71602	0.67997	0.69514
	Hyperparam Tuning	**0.87301**	0.83983	**0.84534**	**0.84126**
Random Forest	Gini Criterion	0.86349	0.85787	0.77376	0.80406
	Entropy Criterion	0.86269	0.86516	0.78167	0.81243
	Log Loss Criterion	0.85634	0.85442	0.77608	0.80553
	Hyperparam Tuning	0.86507	**0.86935**	0.78590	0.81651

Model hyperparameter SVM (linear kernel dan nilai C=0.1) menghasilkan nilai metriks evaluasi terbaik dengan angka akurasi mencapai 0.87301.

III.2 Hasil Test

Berikut adalah tabel skor akurasi, precision, recall, dan F1-Score untuk tiap proses perhitungan metriks evaluasi data tetst eksperimen.

	Parameter Eksperimen	Akurasi	Precision	Recall	F1-Score
MultinomialNB	Alpha 1.0	0.63000	0.70014	0.56781	0.56771
	Alpha 0	0.48200	0.53261	0.40099	0.34016
	Alpha 0.5	0.62400	0.66943	0.58052	0.57291
	Hyperparam Tuning	0.62400	0.66943	0.58052	0.57291
Support Vector Machine	RBF Kernel	0.72400	0.77405	0.66286	0.67629
	Linear Kernel	0.73400	0.72186	0.68317	0.69374
	Poly Kernel	0.44600	0.40591	0.36365	0.26222
	Sigmoid Kernel	0.51400	0.54745	0.45343	0.45735
	Hyperparam Tuning	**0.76600**	**0.78626**	**0.70735**	**0.72183**
Random Forest	Gini Criterion	0.67800	0.73716	0.60374	0.61381
	Entropy Criterion	0.67800	0.74905	0.59291	0.59703
	Log Loss Criterion	0.65600	0.71532	0.57525	0.57762
	Hyperparam Tuning	0.67000	0.72212	0.58425	0.58635

Model hyperparameter SVM (linear kernel dan nilai C=0.1) menghasilkan nilai metriks evaluasi terbaik dengan angka akurasi mencapai 0.76600.

Bab IV Error Analysis

Berikut adalah tabel jumlah kesalahan serta masing-masing komposisi *false sentiment* di dalamnya.

	Parameter Eksperimen	Jumlah Kesalahan	Negative	Neutral	Positive
MultinomialNB	Alpha 1.0	185 (37.00%)	7	62	116
	Alpha 0	259 (51.80%)	12	84	163
	Alpha 0.5	188 (37.60%)	6	54	128
	Hyperparam Tuning	188 (37.60%)	6	54	128
Support Vector Machine	RBF Kernel	138 (27.60%)	5	53	80
	Linear Kernel	133 (26.60%)	31	**47**	**55**
	Poly Kernel	277 (55.40%)	**4**	88	185
	Sigmoid Kernel	243 (48.60%)	63	71	109
	Hyperparam Tuning	**117 (23.40%)**	8	48	61
Random Forest	Gini Criterion	161 (32.20%)	18	63	80
	Entropy Criterion	161 (32.30%)	15	68	78
	Log Loss Criterion	172 (34.40%)	16	68	78
	Hyperparam Tuning	165 (33.00%)	17	69	79

Model hyperparameter SVM menghasilkan jumlah kesalahan terkecil, yaitu: 117 kesalahan (23.40%) yang tersusun atas 8 label negative, 48 label neutral, dan 61 label positive.