Laporan Eksperimen I
Pemrosesan Bahasa Alami

Kelas Pemrosesan Bahasa Alami

Oleh:

Hilal Ramadhan Utomo (1301194236)

Hajarot Najiha (1301194259)

Program Studi Sarjana Informatika

Fakultas Informatika

Telkom University

Bandung

2022

1. Eksperimen

1.1. Penjelasan Eksperimen

Pada eksperimen pertama ini, kami diminta untuk melakukan eksplorasi dan eksperimen pemodelan bahasa dengan N-gram dan neural-based. Korpus yang kami gunakan untuk kedua metode tersebut sama, yakni all_kindle_review.csv, sebuah korpus teks berbahasa Inggris yang mengandung ulasan buku termasuk *rate* atau nilai dari tiap bukunya oleh sang pengulas. Terdapat sebuah data yang berisikan tentang review dan perasaan pembaca terhadap sebuah buku kindle. Mereka juga memberikan rating terhadap buku tersebut. Kemudian akan dilakukan klasifikasi terhadap data tersebut berdasarkan rating yang mereka berikan, dan dilakukannya prediksi dengan review baru
Metadata:
asin = ID dari produk
helpful = menunjukkan seberapa membantu rating yang diberikan contoh: 8/10.
rating = Rating dari produk.
reviewText = review dari user.
reviewTime = waktu yang digunakan saat mereview.
reviewerID = ID dari reviewer
reviewerName = nama dari reviewer.
summary = catatan singkat dari reviewer.
unixReviewTime = timestamp.
Bahasa pemrograman yang kita gunakan merupakan Python.

1.2. Pre-Processing

Terdapat beberapa metode yang akan digunakan pada pre-processing. Di antaranya:

1. Lower casting
2. Penghapusan char tertentu
3. Tokenization
4. Stemming dan lemmatization
5. Penghapusan stopword

Pada lower casting, teks yang mengandung huruf kapital akan diubah menjadi huruf kecil. Setelahnya, beberapa teks yang tidak dibutuhkan dihapus. Seperti format HTML, format URL, tanda baca, angka, dan spasi ganda.

Berikutnya tokenisasi dilakukan, yakni membagi kata per kata dari kalimat yang ada menjadi unit-unit tersendiri bernama token. Sebagai contoh, kalimat, “Budi bermain bola” setelah melalui tokenisasi akan menjadi, “Budi”, “bermain”, “bola”.

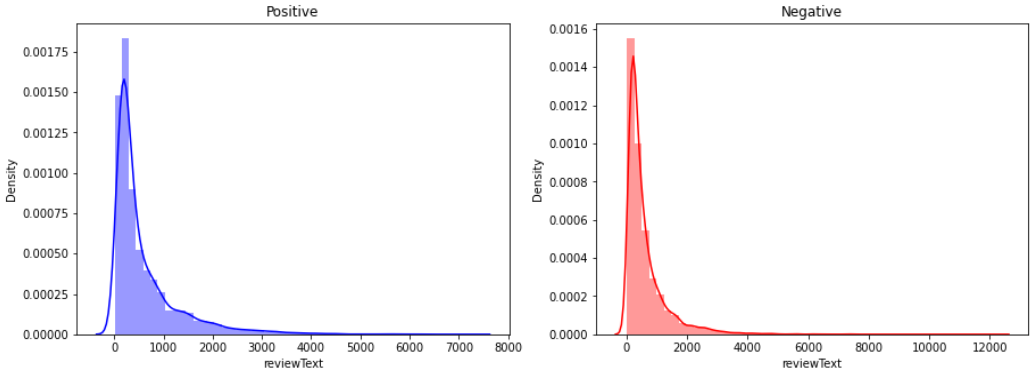
Kemudian teks melalui proses stemming dan lemmanization. Stemming merupakan proses menghapus imbuhan pada suatu kata. Contoh, kata “mengubah” setelah melalui stemming akan menjadi “ubah”. Lemmatization merupakan proses yang mirip dengan stemming, hanya saja proses ini juga melibatkan konteks kalimat.

Terakhir, beberapa stopword dihapus. Karena dataset yang digunakan menggunakan Bahasa Inggris, stopword yang dihapus juga berbahasa Inggris. Daftar kata yang dijadikan stopword dapat dilihat di tabel 1.

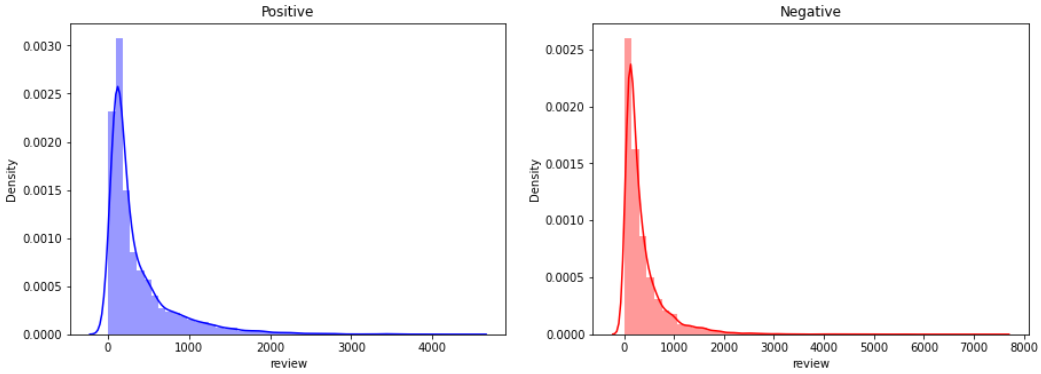
1.3. Eksplorasi Data

Eksplorasi data berguna untuk melihat secara visual perbedaan yang dihasilkan dari preprocessing yang sudah dilakukan

Sebelum pre-processing:



Setelah pre-processing:



Berdasarkan grafik tersebut, terlihat bahwa sebagian dari data terhapus agar lebih mudah diproses.

1.4. N-Gram dan Neural Based

1.4.1. Eksperimen N-gram

1.4.1.1. TF-IDF

Term frequency (TF) adalah frekuensi dari kata t pada dokumen/data d. misalkan kata “bagus” (t) muncul sebanyak 10 kali dalam data (d) maka kita akan menggunakan rumus dari TF yaitu sebagai berikut.

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d}, & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0, & \text{if } \mathrm{tf}_{t,d} = 0 \end{cases}$$

Sementara itu, Inverse Document Frequency (IDF) adalah invers dari ratio data yang memuat suatu kata dan total dari data. Rumus dari IDF yaitu sebagai berikut.

$$idf = log\frac{n}{df}$$

Setelah mengetahui apa itu TF-IDF, kita akan mencari tf-idf dengan menggunakan N-gram. N-gram adalah model yang digunakan untuk memprediksi kata berikutnya yang mungkin dari kata N-1 sebelumnya. N-gram mempunyai beberapa jenis penguraian, antara lain unigram, bigram, trigram dan seterusnya. Dalam kasus ini, kita akan mencari tf-idf menggunakan unigram dan bigram. Untuk mencarinya. Kita menggunakan bantuan Tfidf Vectorizer dengan n-gram_range = 1,2. Setelah itu akan dilakukan fit transform kepada tf-idf yang sudah didapatkan.

Setelah itu, akan dilakukan data split untuk membagi x dan y train maupun test. Pada tahap ini, kita menggunakan bantuan library sklearn. Model_selection dengan import train_test_split. Input yang digunakan sebagai X yaitu tf-idf yang sudah didapatkan, sedangkan Y adalah label(rating_prep).

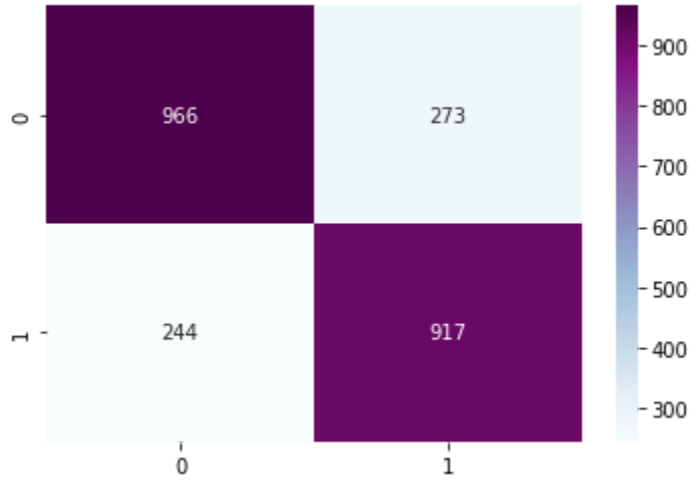
1.4.1.2. Modelling

Pada modelling, kami menggunakan Random Forest. Random forest merupakan metode dari decision tree, yakni diagram alir berbentuk pohon yang memiliki root node untuk mengumpulkan data dan leaf node yang digunakan untuk memecahkan masalah serta membuat keputusan. Dengan itu, kita akan mendapatkan nilai accuracy, precision, recall dan f1-score.

Selain itu, kami juga menambahkan berapa waktu eksekusi yang dibutuhkan dengan feature engineering berupa n-gram. Adapun hasil dari modelling dengan feature engineering n-gram adalah sebagai berikut.

```
******** Random Forest Classifier N gram **********
MultinomialRFC Accuracy: 0.7845833333333333
MultinomialRFC Precision: [0.79834711 0.77058824]
MultinomialRFC Recall: [0.77966102 0.78983635]
MultinomialRFC f1_score: [0.78889343 0.78009358]
Waktu yang diperlukan: 0:03:38.608813
```

Hasil confusion matrix



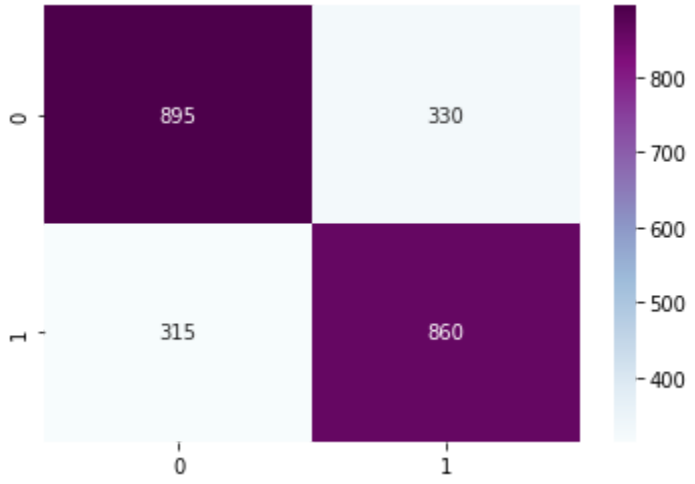
1.4.2. Eksperimen Neural Based

Neural-based menggunakan bantuan vector, dan konsep dari neural network ini untuk mendapatkan vector tersebut. Pada umumnya, akan ada tiga layer yang terdiri dari input, hidden dan output. Dalam proses ini kita menggunakan bantuan library spacy untuk mendapatkan vector. Setelah vector didapatkan, kita mengubah tipenya dari pandas series ke bentuk numpy array. Setelah convert dilakukan, dan di reshape, vector akan digabungkan kembali untuk menjadi output yang diharapkan dari proses neural network dengan bantuan concatenate dari numpy.

Proses terakhir sebelum masuk ke modelling adalah split data, dengan bantuan sklearn.model_selection dengan import train_test_split. Dan didapatkan X dan Y untuk train dan test. Tahap terakhir adalah modelling, dengan metode yang sama seperti N-Gram, yaitu random forest, Adapun hasilnya sebagai berikut

```
******** Random Forest Classifier Neural Based **********
MultinomialRFC Accuracy: 0.73125
MultinomialRFC Precision: [0.73966942 0.72268908]
MultinomialRFC Recall: [0.73061224 0.73191489]
MultinomialRFC f1_score: [0.73511294 0.72727273]
Waktu yang diperlukan: 0:00:14.709866
```

Hasil confusion matrix:



2. Kesimpulan

Berdasarkan hasil eksperimen yang sudah dilakukan, proses fitur ekstraksi dari N-gram lebih cepat dibandingkan dengan neural based. Hal ini dikarenakan dalam neural based tidak terbatas dalam banyakanya kata input data. Sedangkan dalam N-gram dibatasi oleh model yang kita pilih. Namun dalam pemrosesan di modelling, neural based lebih cepat dibandingkan dengan N-gram.

LAMPIRAN

Code:

https://colab.research.google.com/drive/1XB8S-4-JxsbaDA-tqwDpL3Y4vDNtzwt2?usp=sharing

Video presentasi:

https://drive.google.com/drive/folders/1MlD0ka2go4FMT6_1Ha4WFJNx9xAHVMHT?usp=sharing

REFERENSI

https://socs.binus.ac.id/2019/12/31/n-gram/

https://info.cambridgespark.com/latest/word-embeddings-in-python

https://www.nltk.org/

https://towardsdatascience.com/understanding-random-forest-58381e0602d2

https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learningalgorithms/

https://medium.com/mlearning-ai/nlp-tokenization-stemming-lemmatization-and-part-ofspeech-tagging-9088ac06876