Penggunaan Text Mining untuk Penentuan Model Machine Learning dalam Analisis Sentimen Pelanggan Airy Rooms

Alvina Maharani Hasibuan, Matthew Martianus Henry, M. Iqbal
(Anaraunga Team)

IPB University
Bogor Indonesia

Inspiring Innovation with Integrity
in Agriculture, Ocean and Biosciences for a Sustainable World

PENDAHULUAN

Latar Belakang

Dewasa ini, perkembangan teknologi informasi semakin pesat. Perkembangan tersebut menjadikan masyarakat semakin mudah menghasilkan data dan menyimpan data melalui perangkat keras yang dimiliki oleh sebagian besar masyarakat, seperti laptop, PC, *smartphone*, dsb. Masyarakat modern sering kali menghasilkan data tanpa mereka sadari. Diperkirakan, pada 2012 terdapat data berukuran 2.8 zettabytes (2.8 x 10^21), berkembang dari hanya 160 exabytes (160 x 10^18) pada 2006 (Marsland S. 2015). Data yang dihasilkan sangatlah besar dan seringkali memiliki *noise* dan sebaran yang *imbalance*. Beberapa data juga memiliki error dan *missing value* yang akan membuat pengambilan keputusan menjadi tidak sesuai. Data yang dihasilkan perlu diolah terlebih dahulu sebelum dijadikan landasan untuk pengambilan keputusan.

Machine learning yang bersifat *supervised learning* adalah teknik yang paling memungkinkan untuk komputasi dan pengolahan bahasa (Bengfort *et al* 2018). Dalam hal pengolahan data berbentuk teks, diperlukan beberapa teknik analisis seperti *data cleaning*, *exploratory data analysis*, *feature engineering*, *validation scheme*, dan *modeling*. Penggunaan teknik tersebut pada data teks memungkinkan analis data mengambil keputusan yang tepat karena datanya sudah siap untuk diproses.

Salah satu aplikasi yang turut menghasilkan data dalam jumlah besar adalah Airy Rooms. Airy Rooms adalah salah satu platform teknologi jaringan yang bermitra dengan hotel di seluruh Indonesia. Pada platform ini, pengguna dapat melakukan transaksi *booking* hotel yang sesuai keinginan. Pengguna juga dapat memberikan komentar pada fitur yang tersedia untuk mengutarakan apa yang dirasakan oleh pengguna. Dengan banyaknya review yang dihasilkan, maka ukuran data yang dihasilkan sangatlah besar. Ukuran data yang terlalu besar menyulitkan analis dalam menilai *review* dari setiap orang. Menilai suatu *review* sebagai positif atau negatif secara manual dari ukuran data yang sangat besar tentu tidak efisien.

Permasalahan tersebut dapat diselesaikan dengan proses *text mining* dan *text classification*. Setelah proses text mining selesai dilakukan, dilakukan eksperimen untuk menentukan model machine learning yang paling cocok untuk menganalisis review dari Airy Rooms.

Rumusan Masalah

Apa model machine learning yang paling cocok untuk mengklasifikasi *review* positif dan negatif dari Airy Rooms?

Tujuan Penelitian

Berdasarkan latar belakang yang telah dikemukakan, tujuan dari penelitian ini adalah:

1. Menentukan model machine learning yang paling cocok diterapkan untuk klasifikasi *review* positif dan negatif dari Airy Rooms.
2. Melakukan penelitian dengan menggunakan model *Logistic Regression* untuk mengklasifikasikan dokumen teks.

Manfaat Penelitian

Penelitian diharapkan dapat memberikan rujukan kepada analis data dalam membuat suatu model *machine learning* untuk keperluan *text mining*.

METODE PENELITIAN

Dataset

Dataset yang digunakan dalam penelitian ini diunduh melalui sumber https://www.kaggle.com/c/penyisihan-datavidia-7-0/data. Pada link tersebut, terdapat data train.csv yang digunakan untuk melatih model *machine learning*, data test.csv yang digunakan untuk menguji model *machine learning*, dan sample_submission.csv yang menjadi contoh data untuk format pengumpulan. Pada data train.csv, terdapat 3 fitur yaitu *review_id*, *review_text*, dan *category*. Pada data test.csv, hanya terdapat 2 fitur yaitu *review_id* dan *review_text*. Pada sample_submission.csv, terdapat 2 fitur yaitu *review_id* yang sama persis nilainya dengan *review_id* pada test.csv dan *category* yang berisi nilai 0 dan 1 secara acak. Model *machine learning* yang dibuat akan digunakan untuk menentukan nilai fitur *category* untuk data test.csv lalu nilainya ditaruh pada feature category di sample_submission.csv.

Lingkungan Pengembangan

Perangkat lunak yang digunakan dalam penelitian ini adalah sebagai berikut:

a. Microsoft Excel 2013
b. Bahasa Pemrograman Python 3
c. Platform Google Collab

TAHAP PENELITIAN

Tahap penelitian ini secara garis besar terdiri dari *preprocessing*, *exploratory data analysis*, *feature engineering*, dan *modelling*. Keseluruhan tahapan diatas tidak selalu bersifat sekuensial. Pada kasus Airy Rooms, prosesnya juga akan melibatkan *text mining*.

Menurut Hearst (1999), *text mining* adalah proses otomatisasi untuk menemukan informasi yang tidak diketahui sebelumnya dari data yang tidak terstruktur. Dengan *text mining,* mesin berupaya mencari kata- kata yang dapat mewakili dokumen untuk analisis keterhubungan antar dokumen. *Text mining* diawali dengan proses *tokenization* atau pemisahan teks menjadi

kata tunggal. Setelah itu, dilakukan pembuangan stopwords dan dilakukan stemming untuk mengembalikan suatu kata ke bentuk dasarnya.

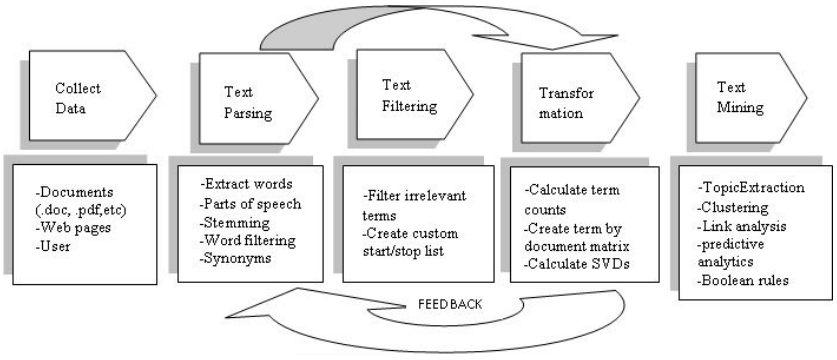
Preprocessing

Preprocessing adalah proses persiapan data sebelum dianalisis lebih lanjut. Tahap *preprocessing* secara umum meliputi penanganan *missing value*, *noise* dan pencilan pada data. *Preprocessing* juga dilakukan untuk mengatasi inkonsistensi pada data. Tahap *preprocessing* juga dikenal sebagai *data cleaning* (Han *et al* 2012).

Missing value merupakan salah satu baris yang satu atau lebih fiturnya tidak memiliki nilai. Pandas, salah satu *library* pada bahasa pemrograman Python merepresentasikan *missing value* sebagai NaN. Missing value dapat diatasi dengan beberapa cara, seperti menghilangkan baris yang terdapat missing value atau mengisi missing value dengan nilai seperti median, rata rata, atau nilai lain yang paling memungkinkan yang ditentukan dengan metode lain seperti regresi, *decision tree*, dsb. Membuang missing value akan menyebabkan ukuran data berkurang. Hal ini dapat mengakibatkan model *machine learning* tidak mampu membuat prediksi dengan lebih akurat. Akan tetapi, mengisi missing value juga dapat menimbulkan bias baru yang dapat berakibat pada prediksi yang salah. Penentuan langkah yang paling tepat untuk mengatasi missing value dapat ditentukan setelah proses *exploratory data analysis*. Penting untuk diketahui bahwa *missing value* bukan berarti nilai tersebut error. Pada beberapa kasus, timbulnya *missing value* dapat disebabkan karena tidak adanya nilai yang memungkinkan untuk mengisi data tersebut.

Noise merupakan nilai pada fitur yang memiliki ragam yang besar. Ragam yang besar tersebut dapat disebabkan oleh adanya pencilan pada data. Noise pada data dapat diatasi dengan membuang data yang menyebabkan fitur tersebut memiliki ragam yang besar.

Pada *text mining*, tahap *preprocessing* juga meliputi pembersihan teks untuk menghilangkan *stopwords* dan proses *stemming* untuk mengubah kata ke bentuk kata dasarnya. *Stopwords* merupakan kata yang frekuensinya besar tetapi tidak memiliki makna tertentu.



1. **Import Datasets & Libraries**

Preprocessing diwali dengan mengimport *library* Python yang dibutuhkan dalam pengolahan data seperti Numpy, Pandas, Matplotlib, dan Seaborn. *Library* Warnings juga ditambahkan untuk mengatasi peringatan yang mungkin muncul setelah menjalankan baris kode.

Setelah menambahkan *library* yang dibutuhkan, dataset train.csv, test.csv, dan sample_submission.csv dibaca oleh *library* pandas dan disimpan sebagai variabel train, test, dan sample.

```
# Importing necessary library
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
# Import Datasets
train = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')
sample = pd.read_csv('sample_submission.csv')
```

Selanjutnya, train.csv akan disebut sebagai data train, test.csv akan disebut sebagai data test, dan sample_submission.csv sebagai sample.

2. **Text Cleaning**

2.1 Removing Non-Alphabetic Characters

Pada tahap ini, *string* yang mengandung tanda baca, angka dan unsur tulisan yang bukan alfabet akan dihapus, karena unsur-unsur tulisan bukan alfabet tidak dapat merepresentasikan sentimen dari suatu kalimat.

2.2 Removing Stopwords

Setelah *string* dibersihkan, pengerjaan dilanjutkan dengan membuang *stopwords* yang terkandung di dalam string. *Stopwords* merupakan kata yang memiliki frekuensi tinggi tetapi tidak memiliki makna yang berarti. Contoh stopwords pada Bahasa Indonesia adalah *antar*, *baru*, *biasa*, dan lain-lain.

2.3 Stemming

Stemming merupakan proses mengubah kata pada teks menjadi kata dasarnya. Stemming untuk Bahasa Indonesia dilakukan menggunakan *library* Sastrawi.

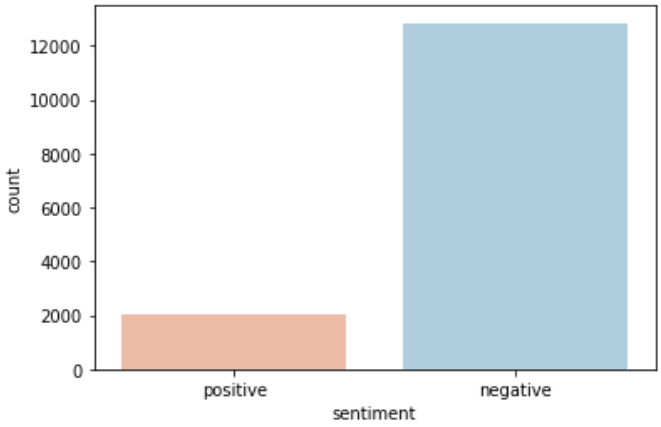
```
# Stemming string with sastrawi
corpus = []
stemmer = factory.create_stemmer()
for string in strings :
    string = string.split()
    string = [stemmer.stem(word) for word in string]
    string = ' '.join(string)
    corpus.append(string)
```

Exploratory Data Analysis (EDA)

Exploratory data analysis (EDA) merupakan proses untuk mendapatkan insight dari data dengan melakukan beberapa pendekatan pada data, seperti visualisasi data, visualisasi distribusi target, dsb. EDA merupakan ranah baru dari statistik yang dikenalkan oleh John Tukey pada 1962 dan menjadi dasar dari statistika modern (Bruce P. dan Bruce A. 2017).

1. **Target Distribution**

Visualisasi distribusi target dilakukan untuk melihat apakah data yang akan kita proses memiliki sebaran target yang seimbang. Bar chart dibawah ini menunjukkan bahwa ***review*** **negatif (86.4%)** lebih mendominasi dibandingkan ***review*** **positif** (**13.6%**).



2. **Visualisasi Data**

Pada tahap ini, dilakukan visualisasi dari data train terhadap *review* dari para pengguna untuk melihat kata atau frasa dalam data train yang frekuensinya tinggi. Visualisasi dilakukan dengan menambahkan *library* Wordcloud.

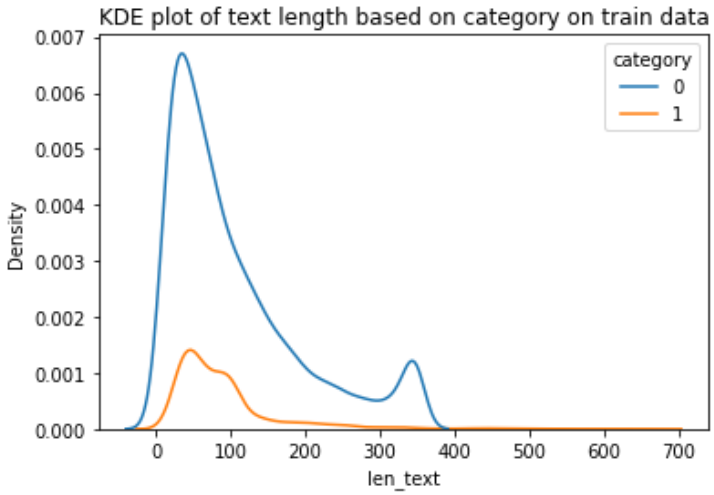
All sentiment Words in Reviews

Berdasarkan wordcloud diatas, kata atau frasa seperti *kamar mandi*, *dan*, *tidak ada*, *hotel*, *untuk*, *saya*, *yg*, *kurang bersih* memiliki frekuensi tinggi pada data train. Untuk mendapatkan visualisasi data yang lebih jelas pada setiap sentimen, dibuat wordcloud yang berisi kata kata dengan frekuensi tinggi untuk setiap sentimen.

Positive Sentiment Words in Reviews

Negative Sentiment Words in Reviews

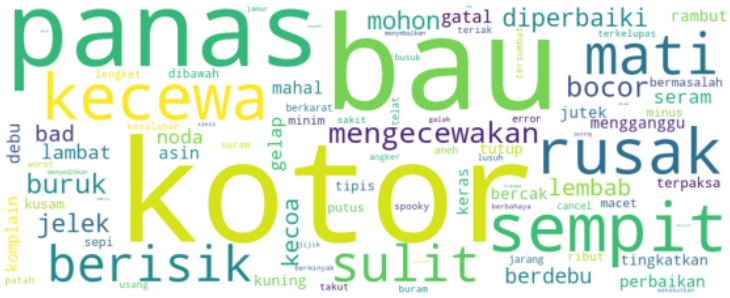
Selain melihat distribusi kata ada *review*, dibuat grafik untuk melihat panjang teks dari setiap *review*. Berdasarkan grafik dibawah ini, terlihat bahwa sebagian besar kalimat *review* negatif lebih panjang dibandingkan dengan kalimat *review* positif.



Seluruh proses diatas dilakukan sebelum dilakukan proses text *cleaning*. Dari visualisasi tersebut, peneliti kemudian menelaah setiap kata yang cenderung menggambarkan sentimen positif dan negatif kemudian mengelompokkannya ke dalam *list* berisi kata sentimen positif dan negatif. Wordcloud-nya menjadi seperti berikut.

Positive Sentiment Words

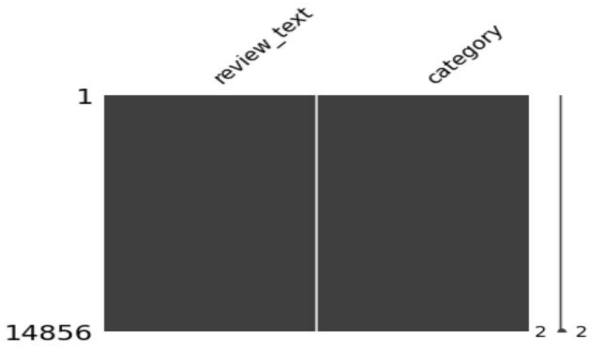
Negative Sentiment Words



Dari wordcloud diatas, terlihat bahwa sebagian besar pelanggan yang memberi review negatif mengeluhkan kamar yang kotor dan bau. Sementara itu, sebagian besar pelanggan yang memberi review positif menyebutkan bahwa kamarnya bagus dan nyaman serta pelayan yang ramah.

3. Missingno Matrix

```
import missingno
missingno.matrix(train_clean, figsize=(6,4))
```



Matriks diatas digunakan untuk melihat letak nilai yang hilang pada setiap kolom dan korelasi antara nilai yang hilang. Dari gambar di atas, terlihat bahwa tidak ada missing value pada fitur *review_text* dan *category*. Hal ini terlihat dari hasil visualisasi gambar yang seluruh grafiknya berwarna hitam yang menunjukkan bahwa seluruh nilai tidak ada yang kosong.

Feature Engineering

1. Bag of Words

Proses *feature engineering* diawali dengan memisahkan setiap kata dengan membuat *bag of words* dengan fungsi CountVectorizer dari library sklearn. Fungsi CountVectorizer akan menjadikan setiap kata sebagai fitur sebuah *data frame* dan setiap kalimat sebagai indeks yang direpresentasikan dengan angka seperti di bawah ini.

```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=10500)
X = cv.fit_transform(corpus).toarray()
```

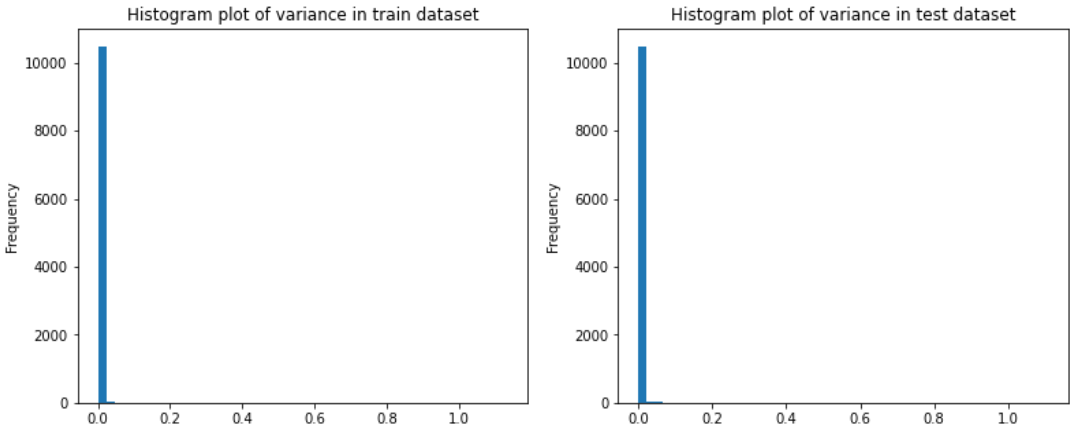
```
test_df.head(2)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2 rows × 10569 columns

2. Analysis of Variance

Analisis lebih dalam dilakukan melalui visualisasi ragam pada data train dan data test. Dari grafik dibawah, terlihat bahwa banyak kata dengan ragam 0 atau mendekati 0 yang memiliki frekuensi tinggi pada data.



Kata dengan ragam 0 menggambarkan bahwa sebarannya sangatlah seragam. Hal ini menunjukkan bahwa kata tersebut hanya disebutkan pada satu kalimat dan tidak disebut pada kalimat lainnya. Kata dengan ragam 0 tersebut menyebabkan model *machine learning* tersebut tidak bisa membaca pola data dari variabel tersebut. Kata-kata tersebut akan dihapus dari *data frame* agar model *machine learning* bisa semakin efektif memprediksi sentimennya. Variabel atau *feature* dengan ragam 0 dapat dideteksi dan dihapus dengan baris kode berikut.

```
# Extract the index where variance is 0.0 in train data
cnt_train = []
for i in range(0,cols_num) :
   if train_df[i].nunique() == 1 :
       cnt_train.append(i)

# Remove columns with index from cnt_train
train_df.drop(cnt_train, axis=1, inplace=True)
test_df.drop(cnt_train, axis=1, inplace=True)
```

Modelling

Modelling merupakan tahap terakhir dalam proses pembuatan model *machine learning*. Pada tahap ini, peneliti akan bereksperimen dengan beberapa model untuk menentukan model terbaik yang akan digunakan.

1. **Logistic Regression**

Salah satu contoh algoritma *machine learning* yang digunakan untuk teks klasifikasi ini adalah *logistic regression. Logistic regression* ini bekerja dengan mengukur hubungan linear antara variabel-variabel bebas dengan variabel tidak bebas. Algoritma ini merupakan salah satu algoritma yang cocok untuk *text classification* karena variabel tak bebas dipengaruhi oleh variabel bebas. Dengan kata lain terdapat hubungan linear antara variabel bebas dan tak bebas. Akurasi dari model *logistic regression* juga meningkat seiring dengan banyaknya variabel tak bebas yang mempengaruhinya.

Logistic regression terdapat di *library sklearn* dengan *class linear_model.* Pada model tersebut ditambahkan parameter ***class_weight*** yaitu ***"weight"***. Parameter ini digunakan karena sebaran data dari variabel target tersebut tidak seimbang sehingga dengan mengatur parameter tersebut akan meningkatkan akurasi dari prediksi model *machine learning*.

```
from sklearn.linear_model import LogisticRegression
Logreg = LogisticRegression(random_state=0,class_weight='weight')
```

2. **Validation**

Pada tahap ini, dilakukan pengukuran akurasi atau skor dari prediksi algoritma machine learning menggunakan metric dari *library* sklearn. Untuk kompetisi ini, *metric* yang digunakan adalah F1 *score*. Untuk itu, peneliti menambahkan fungsi f1_score dari sklearn.metrics. F1 *score* efektif untuk mengukur performa dari model klasifikasi terutama untuk sebaran nilai yang tidak seimbang (*imbalanced*) pada variabel target.

Proses selanjutnya setelah mengetahui *metric* yang digunakan adalah menentukan strategi validasi. Strategi validasi yang digunakan yaitu *train/test split*. Strategi ini akan membagi data train menjadi *training data* yg digunakan untuk fitting model dan *validating data* yang digunakan untuk mengukur performa dari model tersebut. Teknik ini efektif terutama untuk ukuran data yang besar.

```
from sklearn.model_selection import train_test_split
X_train, X_val, y_train, y_val = train_test_split(X, target,
test_size = 0.3, random_state = 0)
```

Setelah data train terbagi, dilakukan pengujian akurasi dengan fungsi f1_score. Keseluruhan proses mulai dari *preprocessing* hingga *modelling* menghasilkan akurasi untuk model *logistic regression* sebesar 75.32%.

```
from sklearn.metrics import f1_score
logreg.fit(X_train, y_train)
y_pred_logreg = logreg.predict(X_val)
score_logreg = f1_score(y_val, y_pred_logreg)
print(score_logreg)

yhat = logreg.predict(X_test)
sample['category'] = yhat
```

```
0    3288
1     426
Name: category, dtype: int64
```

Saat model *logistic regression* digunakan untuk memprediksi data test, diperoleh hasil bahwa output 0 (***review* negatif**) yang dihasilkan jauh lebih banyak dibandingkan output 1 (***review* positif**).

3. Eksperimen Lanjutan

Setelah keseluruhan proses mulai dari *preprocessing* hingga *modelling*, peneliti kemudian melakukan banyak eksperimen untuk meningkatkan akurasi dari model *machine learning* yang dibuat. Beberapa model lain selain *logistic regression* yang diuji oleh peneliti adalah *gaussian naive bayes*, *multinomial naive bayes*, *support vector classifier*, *multi layer perceptron*, dan *random forest*. Peneliti juga melakukan *tuning* setiap model untuk mendapatkan hasil terbaik.

Proses lainnya yang akhirnya memberikan skor tertinggi yaitu *modelling* dataset secara langsung tanpa melakukan proses *text cleaning*. Dataset gabungan dari data train dan data test yang dimasukkan ke fungsi CountVectorizer kemudian dilakukan *modelling* tanpa proses *text cleaning* rupanya menghasilkan akurasi tertinggi dibandingkan eksperimen lainnya.

PENUTUP

Kesimpulan

Analisis sentimen dengan model *logistic regression* pada dataset *review* hotel Airy Rooms menghasilkan nilai akurasi dengan F1 *Score* sebesar 75,32%. Hasil ini kemudian meningkat menjadi 78.32% setelah peneliti mencoba melakukan *modelling* secara langsung tanpa melalui proses *stemming* dan *text cleaning*. Oleh karena itu, modelling kasus *sentiment analysis* pada Airy Rooms tidak memerlukan adanya proses *stemming* dan *text cleaning* untuk menghasilkan nilai yang tinggi.

Saran

Pemodelan lebih lanjut dapat dilakukan dengan *deep learning*. *Deep Learning* merupakan model yang diperkirakan oleh peneliti jauh lebih efektif dibandingkan *logistic regression*. Sama seperti model *logistic regression*, performa *deep learning* akan meningkat dengan semakin bertambah besarnya data. Keuntungan lainnya dari deep learning adalah proses prediksi yang dilakukan *deep learning* bersifat *end to en*d, sedangkan model *machine learning* lainnya berupaya membuat prediksi dengan membagi permasalahan menjadi beberapa bagian lalu menggabungkan beberapa hasil prediksinya diakhir menjadi satu hasil.

Akan tetapi, *modelling* dengan *deep learning* membutuhkan waktu yang sangat lama. Keterbatasan spesifikasi perangkat keras yang dimiliki peneliti menyebabkan peneliti tidak mampu melakukan modelling secara optimal dengan *deep learning* sehingga pada akhirnya peneliti memutuskan menggunakan model *logistic regression*.

DAFTAR PUSTAKA

Bengfort B, Bilbro R, Ojeda T. 2018. Applied Text Analysis with Python. Sebastopol (): O'Reilly Media.

Fayyad UM. 1996. Advances in Knowledge Discovery and Data Mining. MIT Press.USA.

Haddi E, Liu X, Shi Y. 2013. The role of text pre-processing in sentiment analysis. *Procedia Computer Science*. 17:26-32. https://doi.org/10.1016/j.procs.2013.05.005

Han J, Kamber M, Pei J. 2012. Data Mining: Concepts and Techniques.Waltham (USA). Morgan Kaufmann Publishers.

Hearst MA. 1999. Untangling Text Data Mining. ACM Digital Library [internet].[diunduh 30 Januari 2021].

Marsland S. 2015. Machine Learning: An Algorithmic Perspective Second Edition. Boca Raton (USA). CRC Press.

Witten IH, Frank E, Pal CJ, Hall MA. 2011. Data Mining: Practical Machine Learning Tools and Techniques. Burlington (USA): Morgan Kaufman Publishers.