FINAL PROJECT – KELOMPOK 6

TEKS-KNN

Mata Kuliah Pengantar Pemrosesan Data Multimedia

Disusun Oleh :

1. I Made Widi Arsa Ari Saputra (2108561081)
2. Ni Luh Eka Suryaningsih (2108561096)
3. I Ketut Adian Jayaditya (2108561101)

PROGRAM STUDI INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS UDAYANA

2023

BAB I

PENDAHULUAN

1.1 Latar belakang

Penelitian mengenai analisis sentimen atau emosi dari ulasan pengguna telah menjadi topik yang signifikan dalam bidang pemrosesan bahasa alami. Dalam era digital saat ini, ulasan pengguna dapat ditemukan di berbagai platform seperti media sosial, situs web e-commerce, dan forum diskusi. Analisis sentimen atau emosi dari ulasan ini memberikan wawasan berharga bagi perusahaan untuk memahami pendapat dan persepsi pengguna terhadap produk atau layanan mereka. Salah satu pendekatan yang populer untuk mengatasi masalah ini adalah menggunakan algoritma K-Nearest Neighbors (KNN).

Algoritma K-Nearest-Neighbor adalah sebuah metode yang bisa digunakan untuk klasifikasi objek berdasarkan atribut dan sampel latih yang telah tersedia [1]. Metode ini sederhana dan kunci dari metode ini adalah parameter k yang ditentukan oleh pengguna [2]. Algoritma K-Nearest Neighbor (KNN) adalah salah satu algoritma yang banyak digunakan di dunia machine learning untuk kasus klasifikasi. Algoritma ini termasuk dalam kategori supervised learning, yaitu algoritma yang mempelajari pola dari data yang sudah terklasifikasi sebelumnya[3]. Cara kerja algoritma KNN adalah dengan membandingkan data baru dengan data yang sudah ada dalam dataset. Algoritma ini mencari sejumlah K data dengan jarak terdekat dari data baru, kemudian menentukan kelas dari data baru tersebut berdasarkan mayoritas kelas dari K data terdekat tersebut Sebelum melakukan klasifikasi, nilai K harus ditentukan terlebih dahulu. Penentuan nilai K ini tidak ada rumus pastinya, namun satu tips yang dapat dipertimbangkan adalah jika kelas berjumlah genap maka sebaiknya nilai K-nya ganjil, sebaliknya jika kelas berjumlah ganjil maka sebaiknya nilai K-nya genap. Algoritma KNN memiliki kelebihan dan kekurangan. Kelebihannya adalah mudah dipahami dan diimplementasikan, serta dapat digunakan untuk klasifikasi data yang kompleks. Sedangkan kekurangannya adalah sensitif terhadap data pencilan (outlier) dan membutuhkan waktu yang cukup lama untuk melakukan klasifikasi pada dataset yang besar

Dalam laporan ini, kami akan merancang dan mengimplementasikan sebuah sistem aplikasi untuk mengidentifikasi sentimen atau emosi dari ulasan menggunakan algoritma KNN. Kami akan mengumpulkan dataset ulasan yang sudah diberi label sentimen/emosi sebagai data pelatihan. Selanjutnya, kami akan membagi dataset menjadi dua subset, yaitu subset pelatihan untuk melatih model KNN dan subset pengujian untuk menguji performa model. Kami akan menerapkan prinsip "tf-idf" untuk mengubah ulasan menjadi representasi numerik yang dapat digunakan oleh algoritma KNN. Kami juga akan melakukan evaluasi performa menggunakan metrik seperti akurasi, presisi, recall, dan F1-score.

1.2 Rumusan Masalah

1.2.1 Bagaimana performa dari model klasifikasi KNN menggunakan seleksi fitur chi squere dalam melakukan klasifikasi teks ulasan?

1.2.2 Bagaimana kombinasi dari hyperparameter k dan Chi-square terbaik untuk menghasilkan model dengan akurasi maksimal

1.3 Tujuan

1.3.1 Mengetahui performa dari model klasifikasi KNN menggunakan seleksi fitur chi squere dalam melakukan klasifikasi teks ulasan

1.3.2 Mengetahui kombinasi hyperparameter k dan Chi-square terbaik untuk menghasilkan model dengan akurasi maksimal

1.4 Manfaat

1.4.1 Masyarakat

1. Sebagai tools untuk mengidentifikasi sentimen dari suatu produk
2. Menambah wawasan terkait klasifikasi teks menggunakan algoritma KNN dengan seleksi fitur Chi-square

1.4.2 Ilmu Pengetahuan

1. Sebagai bahan untuk penelitian terkait selanjutnya pada bidang klasifikasi teks dan atau algoritma KNN
2. Pengembangan kurikulum pada penelitian program studi Informatika

BAB II

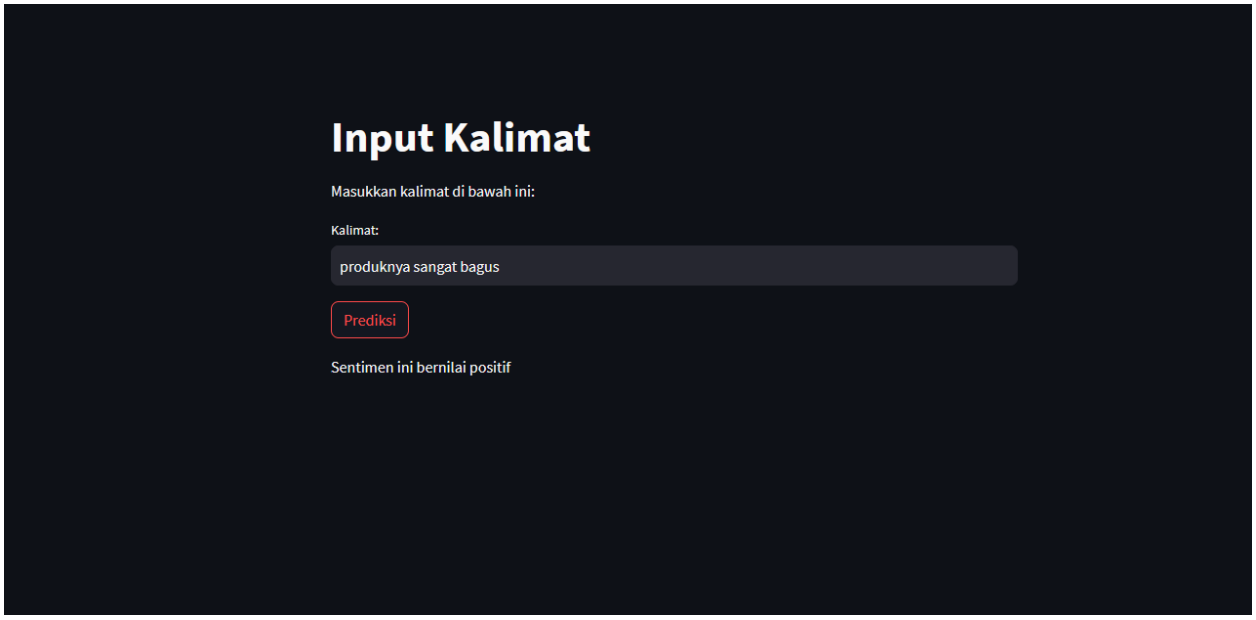
ISI

2.1 Manual Aplikasi

2.1.1 Fitur sistem dan antar muka

Sistem ini berbasis web dengan bantuan 1 framework untuk melakukan pembuatan aplikasinya. Framework yang digunakan adalah framework dari bahasa pemrograman Python yang bernama streamlit yang digunakan sebagai frontend atau pembuatan antar muka.

Pada sistem yang kami buat memiliki fitur untuk dapat melakukan analisis sentimen ulasan secara positif atau negatif dengan menangkap variabel string input user dan dianalisis menggunakan trained model sebelumnya dengan melakukan aksi pada tombol 'Prediksi'. Selain itu aplikasi yang kami buat itu dapat berjalan pada platform berbasis web.



Gambar.1 Tampilan awal aplikasi

2.2 Source Code Modul

2.2.1 Import *library* dan *dataset*

```
import pandas as pd
import re
from nltk.corpus import stopwords
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from nltk.tokenize import word_tokenize
from sklearn.feature_selection import chi2
from sklearn.feature_extraction.text import
TfidfVectorizer
```

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score,
precision_score, recall_score, f1_score,
ConfusionMatrixDisplay, confusion_matrix
import matplotlib.pyplot as plt

df = pd.read_excel("dataset_tugas.xlsx")
df = df.drop("No", axis=1)
X = df.drop("Label", axis=1)
Y = df['Label']
```

Pada modul ini dilakukan import library dan dataset yang digunakan untuk training dan testing. Dataset berekstensi .xlsx yang ditangkap menggunakan fungsi *read_excel* pada modul pandas. Kemudian didefinisikan dataframe 'X' yang berisikan kumpulan dokumen pada dataset, dan dataframe 'Y' yang berisikan kumpulan label dari dokumen pada dataset.

2.2.2 *Case Folding*

```
def case_folding(data):
    """
    Lower case kalimat
    menghilangkan new line, karakter selain huruf dan
spasi,
    serta angka

    """
    data = data.lower()
    data = data.replace('\n',' ')
    data = re.sub('[^\w\s]+', ' ',data)
    data = re.sub('\d+', '', data)
    return data

X['Reviews'] = X['Reviews'].apply(case_folding)
```

Pada modul ini, dilakukan pembersihan kalimat. pemebersihan kalimat terdiri dari: menurunkan semua character menjadi huruf kecil, menghilangkan seluruh *pindah baris* (\n), menghilangkan semua character *non string*, dan menghilangkan karakter angka. Hasil dari proses ini disimpan pada kolom 'Reviews'.

2.2.3 Tokenisasi

```
def tokenisasi(text):
    return word_tokenize(text)

X['Reviews_token'] = X["Reviews"].apply(tokenisasi)
```

Pada modul ini, dilakukan pemisahan kata-kata pada dokumen kedalam bentuk token kata. Hasil dari proses ini disimpan pada kolom baru dengan nama 'Reviews_token'.

2.2.4 Normalisasi kata

```
kamus_alay =
pd.read_csv("colloquial-indonesian-lexicon.csv")
kamus_alay = kamus_alay.filter(['slang', 'formal'],
axis=1)

dict_normalisasi1 = dict(kamus_alay.values)
dict_normalisasi2 = {
    "murceeee" : "murah",
    "thank" : "terima kasih",
    "enggak" : "tidak",
    "makasihh" : "terima kasih",
    "badaiii" : "badai",
    "pertahankann" : "pertahankan",
    "chek" : "cek",
    "tengkyu" : "terima kasih",
    "dtang" : "datang",
    "terimakasih" : "terima kasih",
    "papa" : "kenapa",
    "banget" : "sangat",
    "kayak" : "seperti",
    "cuman" : "hanya",
    "kalo" : "kalau",
    "pcs" : "pieces",
    "tau" : "tahu",
    "gede" : "besar",
    "nih" : "ini",
    "oke" : "bagus",
    "kak" : "kakak"
}
def normalisasi_pertama(document):
    """
    Melakukan normalisasi kata berdasarkan data pada
kamus_alay
    Me-return list baru yang berisi token yang sudah
```

```
berkata baku sesuai dengan data
    """
    return [
        dict_normalisasi1[token] if token in
dict_normalisasi1 else token
        for token in document
    ]

def normalisasi_kedua(document):
    """
    Melakukan normalisasi kata berdasarkan data yang sudah
didefinisikan sebelumnya (static)
    Me-return list baru yang berisi token yang sudah
berkata baku sesuai dengan data
    """
    return [
        dict_normalisasi2[token] if token in
dict_normalisasi2 else token
        for token in document
    ]

X["Reviews_normalisasi"] =
X['Reviews_token'].apply(normalisasi_pertama).apply(normal
isasi_kedua)
X['Reviews_token'].apply(normalisasi_pertama).apply(normal
isasi_kedua)
```

Pada modul ini, dilakukan proses normalisasi untuk mengubah token kata yang tidak baku menjadi baku dengan bantuan kamus alay yang diperoleh dari artikel *Colloquial Indonesian Lexicon*. Token kata yang tidak baku akan diubah menjadi bentuk bakunya sesuai dengan kamus alay dan *dictionary* yang telah dibuat dengan cara statik.

2.2.5 *Stopwords*

```
list_stopwords = stopwords.words('indonesian')
stopwords_buatan = ["nya","lu","sih","hhe"]
list_stopwords_baru = list_stopwords + stopwords_buatan


def hapus_stopwords(words):
    """
    Me-return list baru yang berisi kata yang tidak berada
di list stopwords
    """
    return [
        word for word in words if word not in
list_stopwords_baru
```

```
    ]

X["Reviews_stopword"] =
X["Reviews_normalisasi"].apply(hapus_stopwords)
```

Pada modul ini, dilakukan proses stopword removal menggunakan library nltk.words('indonesian'). Pada bagian ini juga dibentuk stopwords tambahan yang disimpan pada variabel array 'stopword_buatan'. Logika pada fungsi ini adalah, untuk setiap kata pada kalimat, kembalikan kata apabila tidak dalam list stopwords. Hasil dari proses ini disimpan pada kolom baru dengan nama 'Reviews_stopword'.

2.2.6 *Stemming*

```
factory = StemmerFactory()
stemmer = factory.create_stemmer()


def stemming(term):
    return stemmer.stem(term)

X["Reviews_clean"] = [' '.join(map(str, l)) for l in
X["Reviews_stopword"]]
X["Reviews_stem"] = X["Reviews_clean"].apply(stemming)
```

Pada modul ini, dilakukan proses stemming kata atau pengembalian kata ke bentuk dasarnya. modul ini menggunakan library Sastrawi dengan object StemmerFactory(). Hasil dari proses ini disimpan pada kolom baru dengan nama 'Reviews_stem'.

2.2.7 TF-DF dan seleksi fitur *Chi - Square*

```
documents = X['Reviews_stem']
vectorizer = TfidfVectorizer()
X_tfidf = vectorizer.fit_transform(documents)

chi2_scores, _ = chi2(X_tfidf, Y)
scores_dict = dict(zip(vectorizer.get_feature_names_out(),
chi2_scores))

"""
    Eksperimen nilai fitur terbaik dari chi square
    10% = 181
    20% = 363
    30% = 544
    40% = 726
"""
k=544
```

```
top_features = sorted(scores_dict, key=scores_dict.get,
reverse=True)[:k]

new_vectorizer = TfidfVectorizer(vocabulary=top_features)
X_new = new_vectorizer.fit_transform(documents)
feature_names = new_vectorizer.get_feature_names_out()
df_tfidf = pd.DataFrame(X_new.toarray(),
columns=feature_names)
```

Pada modul ini, dilakukan perhitungan TF-IDF pada dokumen yang telah di-*stemming* sebelumnya. Kemudian, hasil dari perhitungan tersebut akan dilakukan seleksi fitur menggunakan *Chi-Square* berdasarkan beberapa percobaan nilai fitur yang dipertahankan, yaitu 181, 163, 544, dan 726.

2.2.8 Data split

```
X_train,X_test,Y_train,Y_test=train_test_split(df_tfidf,Y,
test_size=0.2,random_state=21)
```

Pada modul ini dilakukan *data split* untuk memecah data menjadi data *training* dan data *testing* dengan rasio 80% untuk data *training* dan 20% untuk data testing.

2.2.9 K-NN

```
"""
Eksperimen nilai k=3, 5, 7, 9
"""
model = KNeighborsClassifier(n_neighbors=9)
model.fit(X_train, Y_train)
y_pred = model.predict(X_test)
accuracy_score(Y_test, y_pred)

accuracy_model = accuracy_score(Y_test, y_pred)
precision_model = precision_score(Y_test, y_pred)
recall_model = recall_score(Y_test, y_pred)
skor_f1_model = f1_score(Y_test, y_pred)

print("Akurasi ", accuracy_model)
print("Precision ", precision_model)
print("Recall ", recall_model)
print("F1-skor ", skor_f1_model)


cm = confusion_matrix(Y_test, y_pred,
labels=model.classes_)
```

```
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=model.classes_)
disp.plot()
plt.title("Confusion Matrix dengan nilai chi square 544
dan k = 9")
plt.show()
```

Pada modul ini, dilakukan membangun model klasifikasi *K-Nearest Neighbors* dengan beberapa percobaan nilai k, yaitu 3, 5, 7, dan 9. Model ini akan dievaluasi menggunakan *Confusion Matrix* dengan metrik pengukuran akurasi, *precision*, *recall*, dan *f1-score*.

2.2.10 Save model

```
with open('new_vectorizer.pkl', 'wb') as f:
    pickle.dump(new_vectorizer, f)

with open('knn_model.pkl', 'wb') as f:
    pickle.dump(model, f)
```

Pada modul ini dilakukan penyimpanan model yang telah ditraining. Ada 2 model yang disimpan, yaitu new_vectorizer.pkl dan knn_model.pkl. model ini akan digunakan pada fase deploy website.

BAB III

PENUTUP

Pada penelitian ini percobaan dilakukan terhadap 831 dokumen berupa ulasan atau *review* dalam Bahasa Indonesia yang terbagi ke dalam 2 kategori, yaitu positif (1) dan negatif (0). Percobaan identifikasi sentimen dilakukan menggunakan model klasifikasi *K-Nearest Neighbors* dan seleksi fitur menggunakan *Chi - Square*. *Hyperparameter tuning* juga akan dilakukan untuk mencari model klasifikasi dengan performa terbaik. Beberapa nilai *hyperparameter* yang akan di-*tuning* ialah nilai k = 3, k = 5, k = 7, k =9 dan nilai fitur yang dipertahankan ketika seleksi fitur chi square ialah 10% (181), 20% (363), dan 30% (544). Berikut merupakan hasil dari beberapa percobaan yang dilakukan.

1. **Perbandingan Performa Model Klasifikasi**

K	Fitur teratas	*Accuracy*	*Precision*	*Recall*	*F1-Score*
3	181	87.42%	83.56%	87.14%	85.31%
3	363	80.23%	84.90%	64.28%	73.17%
3	544	89.22%	83.33%	92.85%	87.83%
3	726	91.61%	86.84%	94.28%	90.41%
5	181	84.43%	85.48%	75.71%	80.30%
5	363	73.05%	82.05%	45.71%	58.71%
5	544	89.82%	85.33%	91.42%	88.27%
5	726	91.61%	85.00%	97.14%	90.66%
7	181	79.04%	83.01%	62.85%	71.54%
7	363	67.06%	77.77%	30.00%	43.29%
7	544	92.21%	85.18%	98.57%	91.39%
7	726	91.01%	84.81%	95.71%	89.93%
9	181	75.44%	83.72%	51.42%	63.71%
9	363	65.26%	80.00%	22.85%	35.55%
9	544	91.01%	84.81%	95.71%	89.93%

9	**726**	**89.22%**	**82.50%**	**94.28%**	**88.00%**

Dari beberapa percobaan yang dilakukan, didapatkan kombinasi *hyperparameter* terbaik yaitu dengan nilai k = 7 dan nilai fitur yang dipertahankan berjumlah 544. *Hyperparameter* ini mendapatkan performa akurasi terbesar dengan nilai 92.21%.

Kesimpulannya metode seleksi fitur Chi-square pada algoritma klasifikasi KNN dapat digunakan dengan baik pada proses klasifikasi teks ulasan. Nilai tertinggi pada precision 85.48%, akurasi 92.21%, recall 98.57%, dan F1-score 91.39%. Kombinasi hyperparameter yang menghasilkan akurasi terbaik adalah k=7 dengan sejumlah 544 fitur teratas.

Dalam melakukan deploy model KNN pada modul streamlit. Ada baiknya melakukan save vectorizer sebagai model agar model vectorizer dapat digunakan untuk vektorisasi kalimat string baru dari user.

REFERENSI

[1] Y. Yahya and W. Puspita Hidayanti, “Penerapan Algoritma K-Nearest Neighbor Untuk Klasifikasi Efektivitas Penjualan Vape (Rokok Elektrik) pada ‘Lombok Vape On,’” Infotek J. Inform. dan Teknol., vol. 3, no. 2, pp. 104–114, 2020, doi: 10.29408/jit.v3i2.2279.

[2] A. Alim Murtopo, B. Priyatna, and R. Mayasari, “Signature Verification Using The K-Nearest Neighbor (KNN) Algorithm and Using the Harris Corner Detector Feature Extraction Method,” Buana Inf. Technol. Comput. Sci. (BIT CS), vol. 3, no. 2, pp. 35–40, 2022, doi: 10.36805/bit-cs.v3i2.2763.