LAMPIRAN

1. Pengambilan Data

```
from google_play_scraper import Sort, reviews

result, continuation_token = reviews(
    'id.go.kpu.sirekap2024',
    lang='id',
    country='id',
    sort=Sort.MOST_RELEVANT,
    count=10000000,
    filter_score_with=None
)
```

Lampiran 1 Proses Pengambilan Data

```
import pandas as pd
import numpy as np

df = pd.Dataframe(np.array(result),columns=['review'])
df = df.join(pd.Dataframe(df.pop('review').tolist()))

df.head()
```

Lampiran 2 Memasukan Data Ke *Dataframe* Pandas

```
df = df[['userName', 'score', 'at', 'content', 'thumbsUpCount']]
df.sort_values(by='at', ascending=False)

df.head()
```

Lampiran 3 *Filtering* dan *Sorting* Kolom

```
df.to_csv("scrapped_data.csv", index = False, sep='|')
```

Lampiran 4 Ekspor Data Ke File CSV

```
import pandas as pd

file_csv = 'scrapped_data.csv'

df = pd.read_csv(file_csv, sep='|')

print(df)
```

Lampiran 5 Input File CSV

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
ratings, counts =
zip(*sorted(dict(df['Rating'].value_counts()).items()))
plt.bar(ratings, counts, alpha=0.7, edgecolor='black', align='center')
plt.title('Distribusi Rating')
plt.xlabel('Rating')
plt.ylabel('Frekuensi')
plt.xticks(ratings)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()

rating_counts = df['Rating'].value_counts().sort_index()

for rating, count in rating_counts.items():
    print(f"Rating {rating}: {count} kali")
```

Lampiran 6 Proses Distribusi *Rating*

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
ratings, counts =
zip(*sorted(dict(df['Rating'].value_counts()).items()))
plt.bar(ratings, counts, alpha=0.7, edgecolor='black', align='center')
plt.title('Distribusi Rating')
plt.xlabel('Rating')
plt.ylabel('Frekuensi')
plt.xticks(ratings)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()

rating_counts = df['Rating'].value_counts().sort_index()

for rating, count in rating_counts.items():
    print(f"Rating {rating}: {count} kali")
```

Lampiran 7 Proses *Wordcloud*

2. *Pra-proses* Data

```
digit_count = len(str(len(df)))

df.insert(0, 'ID', df.index.map(lambda x: 'd' +
str(x+1).zfill(digit_count)))

print(df)
```

Lampiran 8 Proses Pemberian ID Disetiap Data

```
import time

def lowercase(text):
    return text.lower()

start_time = time.time()
df['komentar_lowercase'] = df['Komentar'].apply(lowercase)
end_time = time.time()
time_taken = end_time - start_time

print(df[['Komentar', 'komentar_lowercase']])
print("Waktu proses:", time_taken, "detik")
```

Lampiran 9 Proses *Lower Case*

```
import re

def remove_unnecessary_char(text):
    text = re.sub('((www\.[^\s]+)|(https?://[^\s]+)|(http?://[^\s]+))','
',text)
    text = re.sub('\n',' ',text)
    text = re.sub('\r',' ',text)
    text = re.sub(r'\\x..',' ',text)
    text = re.sub('  +', ' ', text)
    text = re.sub('[^0-9a-zA-Z]+', ' ', text)
    return text

start_time = time.time()
df['komentar_remove_char'] =
df['komentar_lowercase'].apply(remove_unnecessary_char)
end_time = time.time()
time_taken = end_time - start_time


print(df[['komentar_lowercase', 'komentar_remove_char']])
print("Waktu proses:", time_taken, "detik")
```

Lampiran 10 Proses Remove Unnecessary Character

```
import re

# Path untuk file input dan output
file_path = "kbi.txt"
output_file_path = "4 kata lebih list kata.txt"

# Membaca konten file
with open(file_path, "r", encoding="utf-8") as file:
    text = file.read()

# Ekstraksi kata-kata dan konversi ke huruf kecil
words = re.findall(r'\b\w+\b', text.lower())

# Menghapus kata-kata yang mengandung karakter numerik
words = [word for word in words if not any(char.isdigit() for char in word)]

# Menghapus kata-kata yang hanya terdiri dari satu hingga tiga huruf
words = [word for word in words if len(word) >= 4]

# Menghapus duplikat kata dengan menggunakan set
unique_words = set(words)

# Mengurutkan kata-kata sesuai dengan abjad
sorted_words = sorted(unique_words)

# Membuka file untuk menulis hasil output
with open(output_file_path, "w", encoding="utf-8") as output_file:
    for word in sorted_words:
        output_file.write(word + "\n")

print(f"Kata-kata unik telah disimpan di file {output_file_path}")
```

Lampiran 11 Source Code Kamus "4 lebih list kata KBBI.txt"

```
with open('4 lebih list kata KBBI.txt', 'r', encoding='utf-8') as file:
    kamus_baku = set(file.read().splitlines())

def detect_and_insert_unknown_words(text, baku_dict):
    words = text.split()
    unknown_words = [word for word in words if word.lower() not in baku_dict]
    return ', '.join(unknown_words)

df['Kata_Tidak_Baku'] = df['komentar_stemming'].apply(lambda x: detect_and_insert_unknown_words(x, kamus_baku))

print(df['Kata_Tidak_Baku'])
```

Lampiran 12 Cek Kata Tidak Baku

```
def detect_and_create_dataframe(text, baku_dict):
    words = text.split()
    kata_tidak_baku = set(word.lower() for word in words if word.lower()
not in baku_dict)
    return pd.Dataframe({'Kata_Tidak_Baku': list(kata_tidak_baku)})

df_kata_tidak_baku = pd.concat([detect_and_create_dataframe(row,
kamus_baku) for row in df['komentar_stemming']], ignore_index=True)

df_kata_tidak_baku = df_kata_tidak_baku.drop_duplicates()
df_kata_tidak_baku_sorted =
df_kata_tidak_baku.sort_values(by='Kata_Tidak_Baku')

print(df_kata_tidak_baku_sorted)
```

Lampiran 13 Cek List Kata Tidak Baku

```
alay_dict = pd.read_csv('kamus_tidak_baku.csv', sep=",",
encoding='latin-1', header=None)
alay_dict = alay_dict.rename(columns={0: 'original',
                                      1: 'replacement'})

alay_dict_map = dict(zip(alay_dict['original'],
alay_dict['replacement']))
```

Lampiran 14 Proses Input "kamus_tidak_baku"

```
def normalize_alay(text):
    return ' '.join([alay_dict_map[word] if word in alay_dict_map
                     else word for word in text.split(' ')])

start_time = time.time()
df['komentar_spellchecker'] = df['komentar_remove_char'].apply
(normalize_alay)
end_time = time.time()
time_taken = end_time - start_time

print(df[['komentar_remove_char', 'komentar_spellchecker']])
print("Waktu proses:", time_taken, "detik")
```

Lampiran 15 Proses *Spell Checker*

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
factory = StemmerFactory()
stemmer = factory.create_stemmer()

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

start_time = time.time()

df['komentar_stemming'] = df['komentar_spellchecker'].apply(stemming)

end_time = time.time()
time_taken = end_time - start_time
```

Lampiran 16 Proses *Stemming*

```
import numpy as np

df['komentar_stemming'] = df['komentar_stemming'].replace('', np.nan)

baris_nan = df[df['komentar_stemming'].isnull()]
jumlah_nan = df['komentar_stemming'].isnull().sum()
print(f"Jumlah NaN pada kolom 'komentar_stemming': {jumlah_nan}")
print(baris_nan)
```

Lampiran 17 Cek Data Kosong

```
df_cleaned = df.dropna(subset=['komentar_stemming'])
df_cleaned
```

Lampiran 18 Hapus Data Kosong

3. Proses Klasifikasi dan Evaluasi *F1-Score Wordnet*

```
!pip install Deep-translator

from deep_translator import GoogleTranslator
langs_dict = GoogleTranslator().get_supported_languages(as_dict=True)
import time

def translate_komentar(text):
    text_translated = GoogleTranslator(source='id',
target='en').translate(text)
    return text_translated

from tqdm import tqdm

tqdm.pandas()

start_time = time.time()
df['Translated'] =
df['komentar_stemming'].progress_apply(translate_komentar)
end_time = time.time()
time_taken = end_time - start_time

print(df[['komentar_stemming', 'Translated']])
print("Waktu proses:", time_taken, "detik")
```

Lampiran 19 Proses Penerjemahan Data Ulasan ke dalam Bahasa Inggris

```
from TextBlob import TextBlob

def scoring_sentiment(text):
    analysis = TextBlob(text)
    sentiment_score = analysis.sentiment.polarity
    # print("Sentiment polarity:", sentiment_score)
    score_Wordnet = 0

    if sentiment_score > 0.5:
        score_Wordnet = '5 star'
    elif 0.2 <= sentiment_score <= 0.5:
        score_Wordnet = '4 star'
    elif -0.2 < sentiment_score < 0.2:
        score_Wordnet = '3 star'
    elif -0.5 <= sentiment_score <= -0.2:
        score_Wordnet = '2 star'
    else:
        score_Wordnet = '1 star'

    return score_Wordnet, sentiment_score
```

Lampiran 20 Fungsi Hitung Score

```
start_time = time.time()
score_Wordnet, sentiment_score =
zip(*df['Translated'].apply(scoring_sentiment))
df['Score_Wordnet'] = score_Wordnet
df['Sentiment_Score'] = sentiment_score
end_time = time.time()
time_taken = end_time - start_time

print(df[['komentar_stemming', 'Score_Wordnet','Sentiment_Score']])
print("Waktu proses:", time_taken, "detik")
```

Lampiran 21 Proses *Wordnet*

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, f1_score
from matplotlib.ticker import FixedLocator

cm = confusion_matrix(df['Rating'], df['Score_Wordnet'])

Macro_f1 = f1_score(df['Rating'], df['Score_Wordnet'], Average='Macro')
print(f"Macro F1-Score: {Macro_f1}")

fig, ax = plt.subplots(figsize=(10, 8), dpi=100)
cax = ax.matshow(cm, cmap=plt.cm.Greens)
fig.colorbar(cax)

tick_marks = np.arange(len(np.unique(df['Rating'])))
ax.set_xticks(tick_marks)
ax.set_yticks(tick_marks)
ax.set_xticklabels(list(range(1, 6)))
ax.set_yticklabels(list(range(1, 6)))

plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Wordnet')

thresh = cm.max() / 2
for i in range(len(cm)):
    for j in range(len(cm[i])):
        value = cm[i][j]
        text_color = 'white' if value > thresh else 'black'
        ax.text(j, i, str(value), va='center', ha='center',
color=text_color)
plt.show()
```

Lampiran 22 Proses Evaluasi *F1-Score Wordnet*

4. Proses Klasifikasi dan Evaluasi *F1-Score Decision Tree* CART

```
X = df['komentar_stemming']
y = df['Rating']
```

Lampiran 23 Proses Penentuan Input dan Kelas

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer()
features = vectorizer.fit_transform(X)
```

Lampiran 24 Proses Ekstraksi Fitur TF-IDF

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, Precision_score, Recall_score, f1_score
from sklearn.model_selection import KFold, cross_val_predict

def cross_validation(model, _X, _y, _cv):
    kf = KFold(n_splits=_cv, shuffle=True, random_state=42)
    fold_count = 1
    results = {}
    fold_predictions = []
    overall_cm = np.zeros((len(np.unique(_y)), len(np.unique(_y))), dtype=int)

    for train_index, test_index in kf.split(_X):
        X_train, X_test = _X[train_index], _X[test_index]
        y_train, y_test = _y[train_index], _y[test_index]

        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)
        fold_predictions.append(y_pred)
        cm = confusion_matrix(y_test, y_pred)

        results[f"Fold {fold_count}"] = {
            "Confusion Matrix": cm,
            "Precision": Precision_score(y_test, y_pred, Average='Macro'),
            "Recall": Recall_score(y_test, y_pred, Average='Macro'),
            "F1 Score": f1_score(y_test, y_pred, Average='Macro')
        }
        overall_cm += cm
        fold_count += 1

    for key, value in results.items():
        print(key + ":")
        print("Confusion Matrix:")
        print(value["Confusion Matrix"])
        print("Precision:", value["Precision"])
        print("Recall:", value["Recall"])
        print("F1 Score:", value["F1 Score"])
        print()

        fig, ax = plt.subplots(figsize=(10, 8), dpi=100)
```

```
41          im = ax.imshow(value["Confusion Matrix"],
    interpolation='nearest', cmap=plt.cm.Greens)
42          ax.figure.colorbar(im, ax=ax)
43
44          class_names = np.unique(_y)
45          ax.set(xticks=np.arange(len(class_names)),
46                 yticks=np.arange(len(class_names)),
47                 xticklabels=class_names, yticklabels=class_names,
48                 title=f'Confusion Matrix - {key}',
49                 ylabel='True label',
50                 xlabel='Predicted label')
51
52          thresh = value["Confusion Matrix"].max() / 2.
53          for i in range(value["Confusion Matrix"].shape[0]):
54              for j in range(value["Confusion Matrix"].shape[1]):
55                  ax.text(j, i, format(value["Confusion Matrix"][i, j],
    'd'),
56                          ha="center", va="center",
57                          color="white" if value["Confusion Matrix"][i, j]
    > thresh else "black")
58
59          plt.show()
60
61      print("Overall Confusion Matrix:")
62      print(overall_cm)
63      print()
64
65      fig, ax = plt.subplots(figsize=(10, 8), dpi=100)
66      cax = ax.matshow(overall_cm, cmap=plt.cm.Greens)
67      fig.colorbar(cax)
68
69      class_names = np.unique(_y)
70      ax.set_xticks(np.arange(len(class_names)))
71      ax.set_yticks(np.arange(len(class_names)))
72      ax.set_xticklabels(class_names)
73      ax.set_yticklabels(class_names)
74      plt.xlabel('Predicted')
75      plt.ylabel('Actual')
76      plt.title('Overall Confusion Matrix')
77
78      thresh = overall_cm.max() / 2.
79      for i in range(overall_cm.shape[0]):
80          for j in range(overall_cm.shape[1]):
81              ax.text(j, i, format(overall_cm[i, j], 'd'), ha='center',
    va='center',
82                      color='white' if overall_cm[i, j] > thresh else
    'black')
83
84      plt.show()
85
86      y_pred = cross_val_predict(model, _X, _y, cv=_cv)
87      cm_overall = confusion_matrix(_y, y_pred)
88
89      Precision_Macro = Precision_score(_y, y_pred, Average='Macro')
90      print(f"\nPrecision Macro: {Precision_Macro}")
91
92      Recall_Macro = Recall_score(_y, y_pred, Average='Macro')
93      print(f"Recall Macro: {Recall_Macro}")
```

```
94
95        f1_score_Macro = f1_score(_y, y_pred, Average='Macro')
96        print(f"F1-Score Macro: {f1_score_Macro}")
97
98        return results, fold_predictions
```

Lampiran 25 Proses Cross-Validation 10 *Fold*

```
1   kfold = KFold(n_splits=10, shuffle=False)
2   for fold, (train_index, test_index) in enumerate(kfold.split(X, y), 1):
3       X_train, X_test = X[train_index], X[test_index]
4       y_train, y_test = y[train_index], y[test_index]
5
6       print(f'Fold {fold}:')
7       print(f'  - Train data: {len(X_train)} samples')
8       print(f'  - Test data: {len(X_test)} samples')
9       print(f'  - Train Index: {train_index
10  }')
```

Lampiran 26 Menampilkan Jumlah Data Setiap Pembagian *Fold*

```
1   from sklearn.tree import DecisionTreeClassifier
2
3   cart_model = DecisionTreeClassifier(
4       criterion='gini',
5       max_depth=None,
6       min_samples_split=2,
7       min_samples_leaf=1,
8       max_features=None,
9       class_weight=None,
10      random_state=0
11  )
```

Lampiran 27 Proses Input Model Klasifikasi *Decision Tree* CART

```
1   import time
2
3   start_time = time.time()
4   cv_results, fold_predictions = cross_validation(cart_model, features, y,
    _cv=10)
5
6   end_time = time.time()
7   time_taken = end_time - start_time
8   print("Waktu proses:", time_taken, "detik")
```

Lampiran 28 Proses Visualisasi Model dengan Menjalankan *Cross Validation*

5. Dokumen

UMKT
Program Studi
Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Muhammadiyah Kalimantan Timur
Berkarakter | Berwawasan | Berkemajuan

Telp. 0541-748511 Fax. 0541-766832
Website http://informatika.umkt.ac.id
email: informatika@umkt.ac.id

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Nomor : 055-001/KET/FST.1/A/2024
Lampiran : -
Perihal : **Keterangan Pengambilan Data Sekunder**

Assalamu'alaikum Warrahmatullahi Wabarrakatuh

Puji Syukur kepada Allah Subhanahu wa ta'ala yang senantiasa melimpahkan Rahmat-Nya kepada kita sekalian. Amin.

Dengan surat ini, kami menerangkan bahwa mahasiswa berikut:

No	Nama	NIM
1	Emyzar Haflida Tanjung	2011102441240
2	Lilis Sagita	2011102441198
3	Sri Ramadani	2011102441177
4	Muhammad Fariz Ijlal Rafi	2011102441124
5	Nurlita	2011102441070

Melakukan penelitian dengan pengambilan data sekunder di Google Playstore data yang diambil yaitu data ulasan aplikasi "Sirekap" dari rating 1-5.

Demikian hal ini disampaikan, atas kerjasamanya kami ucapkan terima kasih.

Wassalamu'alaikum Warrahmatullahi Wabarrakatuh

Samarinda, 18 Dzulhijjah 1445 H
25 Juni 2024 M

Ketua Program Studi S1 Teknik Informatika

Arbansyah, S.Kom., M.TI
NIDN. 1118019203

Kampus 1 : Jl. Ir. H. Juanda, No.15, Samarinda
Kampus 2 : Jl. Pelita, Pesona Mahakam, Samarinda

Lampiran 29 Surat Keterangan Pengambilan Data Sekunder

KARTU KENDALI BIMBINGAN LAPORAN KARYA ILMIAH

Nama : Muhammad Fariz Ijlal Rafi
NIM : 2011102441124
Nama Dosen Pembimbing : Naufal Azmi Verdikha, S.Kom., M.Eng.
Judul : ANALISIS KOMPARASI MODEL KLASIFIKASI UNTUK ULASAN APLIKASI SIREKAP 2024 MENGGUNAKAN CART DAN METODE *WORDNET*

No	Tanggal	Uraian Pembimbingan	Paraf Dosen
1	26/01/2024	Diskusi mengenai data yang akan digunakan dalam penelitian	
2	27/01/2024	Pemilihan metode yang akan digunakan dalam penelitian	
3	28/01/2024	Evaluasi awal data ulasan yang akan digunakan dan literatur terkait metode CART dan *Wordnet*.	
4	29/01/2024	Pembahasan metode pengumpulan data dan teknik preprocessing yang relevan.	
5	07/02/2024	Review hasil sementara preprocessing data dan perbaikan berdasarkan saran.	
6	15/02/2024	Diskusi mengenai penerapan model CART pada dataset dan evaluasi awal hasilnya.	
7	20/03/2024	Evaluasi penerapan metode *Wordnet* untuk analisis sentimen pada data ulasan.	
8	18/04/2024	Perbandingan hasil klasifikasi antara model CART dan metode *Wordnet*, serta interpretasi hasilnya.	
9	30/04/2024	Review dan saran perbaikan laporan hasil penelitian	
10	10/05/2024	Pembahasan mengenai integrasi temuan dalam laporan dan persiapan dokumen akhir.	
11	26/06/2024	Finalisasi laporan penelitian dan persiapan untuk presentasi akhir.	

Mengetahui

Dosen Pembimbing

(.....)

Ketua Program Studi

(.....)

Lampiran 30 Kartu Kendali Bimbingan Laporan Karya Ilmiah

SKRIPSI MUHAMMAD FARIZ IJLAL RAFI

by Teknik Informatika Universitas Muhammadiyah Kalimantan Timur

Submission date: 25-Jul-2024 10:56AM (UTC+0800)
Submission ID: 2422080893
File name: SKRIPSI_MUHAMMAD_FARIZ_IJLAL_RAFI.docx (1.96M)
Word count: 7128
Character count: 44307

SKRIPSI MUHAMMAD FARIZ IJLAL RAFI

ORIGINALITY REPORT

14%	13%	7%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	repository.its.ac.id Internet Source	2%
2	ijcs.stmikindonesia.ac.id Internet Source	1%
3	repository.uin-suska.ac.id Internet Source	1%
4	jurnal.umk.ac.id Internet Source	<1%
5	dspace.uii.ac.id Internet Source	<1%
6	dspace.umkt.ac.id Internet Source	<1%
7	gudangjurnal.com Internet Source	<1%
8	text-id.123dok.com Internet Source	<1%
9	pdfcookie.com Internet Source	<1%

RIWAYAT HIDUP

Penulis lahir di Suliliran Baru pada 9 Agustus 2002 sebagai anak pertama dari tiga bersaudara, dari pasangan Kusnadi dan Surtinah. Saat ini, penulis tinggal di Jl. K.H. Wahid Hasyim, Gg. Sungai II, RT 011, Sempaja Selatan, Samarinda Utara. Penulis menyelesaikan pendidikan di SD Negeri 013 Pasir Belengkong pada tahun 2014, kemudian melanjutkan ke SMP Negeri 03 Pasir Belengkong dan lulus pada tahun 2017, serta menamatkan pendidikan di SMA Negeri 01 Pasir Belengkong pada tahun 2020. Saat ini, penulis sedang menempuh pendidikan di Universitas Muhammadiyah Kalimantan Timur di Kota Samarinda dengan jurusan Teknik Informatika Internasional dan belajar di gedung Fakultas Sains dan Teknologi. Penulis pernah menjalani program magang pada semester 7 di PT. Barqun Digital Teknologi, sebuah perusahaan software house di Samarinda. Saat ini, penulis sedang menyelesaikan tugas akhir atau skripsi