Tugas NLP 4

Rahmat Kurnianto 1301188593

Hasil Kondisi awal

Bandingkan performansi dengan base model, yaitu model dengan parameter default.

In [18]:
```
base_model = LogisticRegression(random_state = 8)
base_model.fit(features_train, labels_train)
accuracy_score(labels_test, base_model.predict(features_test))
```

Out[18]: 0.9431137724550899

In [19]:
```
best_classifier.fit(features_train, labels_train)
accuracy_score(labels_test, best_classifier.predict(features_test))
```

Out[19]: 0.9401197604790419

- --

1. Membuat fitur

   Tanpa proses normalisation

   - Variable yang di ubah

In [12]:
```
nrows = len(df)
lemmatized_text_list = []

for row in range(0, nrows):

    # Create an empty list containing lemmatized words
    lemmatized_list = []

    # Save the text and its words into an object
    #text = df.loc[row]['Content_Parsed_4']#no1
    text = df.loc[row]['Content']
    text_words = text.split(" ")

    # Iterate through every word to lemmatize
    for word in text_words:
        lemmatized_list.append(wordnet_lemmatizer.lemmatize(word, pos="v"))

    # Join the list
    lemmatized_text = " ".join(lemmatized_list)

    # Append to the list containing the texts
    lemmatized_text_list.append(lemmatized_text)
```

   - Hasil akurasi

In [18]:
```
base_model = LogisticRegression(random_state = 8)
base_model.fit(features_train, labels_train)
accuracy_score(labels_test, base_model.predict(features_test))
```

Out[18]: 0.9251497005988024

In [19]:
```
best_classifier.fit(features_train, labels_train)
accuracy_score(labels_test, best_classifier.predict(features_test))
```

Out[19]: 0.9341317365269461

   Tanpa proses lemmatisation

   - Variabel yang diubah

In [17]:
```
df['Content_Parsed_6'] = df['Content_Parsed_5']
#df['Content_Parsed_6'] = df['Content_Parsed_4']

for stop_word in stop_words:

    regex_stopword = r"\b" + stop_word + r"\b"
    df['Content_Parsed_6'] = df['Content_Parsed_6'].str.replace(regex_stopword, '')
```

   - Hasil Akurasi

Bandingkan performansi dengan base model, yaitu model dengan parameter default.

```
In [18]:
base_model = LogisticRegression(random_state = 8)
base_model.fit(features_train, labels_train)
accuracy_score(labels_test, base_model.predict(features_test))
```

```
Out[18]: 0.9251497005988024
```

```
In [19]:
best_classifier.fit(features_train, labels_train)
accuracy_score(labels_test, best_classifier.predict(features_test))
```

```
Out[19]: 0.9341317365269461
```

Tanpa menghilangkan stopwords

- Variable yang di ubah

```
In [26]:
list_columns = ["File_Name", "Category", "Content", "Content_Parsed_6"]
df = df[list_columns]

df = df.rename(columns={'Content_Parsed_5': 'Content_Parsed'})
#df = df.rename(columns={'Content_Parsed_6': 'Content_Parsed'})
```

- Hasil Akurasi

Bandingkan performansi dengan base model, yaitu model dengan parameter default.

```
In [18]:
base_model = LogisticRegression(random_state = 8)
base_model.fit(features_train, labels_train)
accuracy_score(labels_test, base_model.predict(features_test))
```

```
Out[18]: 0.9461077844311377
```

```
In [19]:
best_classifier.fit(features_train, labels_train)
accuracy_score(labels_test, best_classifier.predict(features_test))
```

```
Out[19]: 0.9461077844311377
```

2. tfidf dengan nilai "max_features"

- Set min 150

```
In [33]:
# Parameter election
ngram_range = (1,2)
min_df = 10
max_df = 1.
max_features = 150 #no 2
```

Hasil akurasi

```
In [18]:
base_model = LogisticRegression(random_state = 8)
base_model.fit(features_train, labels_train)
accuracy_score(labels_test, base_model.predict(features_test))
```

```
Out[18]: 0.9041916167664671
```

```
In [19]:
best_classifier.fit(features_train, labels_train)
accuracy_score(labels_test, best_classifier.predict(features_test))
```

```
Out[19]: 0.9041916167664671
```

- Set maks 450

```
# Parameter election
ngram_range = (1,2)
min_df = 10
max_df = 1.
max_features = 450 #no 2
```

Hasil akurasi

In [18]:
```
base_model = LogisticRegression(random_state = 8)
base_model.fit(features_train, labels_train)
accuracy_score(labels_test, base_model.predict(features_test))
```

Out[18]: 0.9401197604790419

In [19]:
```
best_classifier.fit(features_train, labels_train)
accuracy_score(labels_test, best_classifier.predict(features_test))
```

Out[19]: 0.9461077844311377

3. Jika mengunakan Bahasa indonesia
    - Pada proses Normalisation di Bahasa inggris diganti jadi jadi proses normalisasi Bahasa Indonesia. contoh dalam bahasa inggris untuk meyatakan jamak pada kata noun mengukan akhiran ‘s menandakan kepunyaan, sedangkan pada bahasa tidak ada
    - Pada proses Lemmatisation Bahasa inggris di ubah jadi Bahasa Indonesia. contoh pada kata tempel jadi tempelan
    - stopwords karena aturan penggalan kata yang tidak relavan pada bahasa indonesia dengan inggris berbeda, contoh euy, anjir, dll