LAMPIRAN

Lampiran 1 *Import Document* ke *library pandas*

```
import pandas as pd

#Import Data Excel
df = pd.read_excel("data.xlsx",sheet_name="Master")
data = pd.DataFrame(df)

data.head()
```

	No	Kritik	Bidang
0	1.0	Pelayanan agar lebih ditingkatkan	Pelayanan
1	2.0	Lahan parkir diperluas	Sarana dan Prasarana
2	3.0	Loket diperbanyak, ada tempat bermain untuk an...	Sarana dan Prasarana
3	4.0	Dokter spesialis ditingkatkan	Sumber Daya Manusia
4	5.0	Pelayanan kualitasnya ditingkatkan	Pelayanan

Lampiran 2 *Encoding label*

```
#delete column No
del data["No"]

#change text to numeric on Bidang
data["Bidang"] = data["Bidang"].replace("Pelayanan",1)
data["Bidang"] = data["Bidang"].replace("Sarana dan Prasarana",2)
data["Bidang"] = data["Bidang"].replace("Sumber Daya Manusia",3)
data["Bidang"] = data["Bidang"].replace("Administrasi dan Manajemen",4)
data["Bidang"] = data["Bidang"].replace("Peralatan",5)
data["Bidang"] = data["Bidang"].replace("Netral",0)
data.head()
```

	Kritik	Bidang
0	Pelayanan agar lebih ditingkatkan	1
1	Lahan parkir diperluas	2
2	Loket diperbanyak, ada tempat bermain untuk an...	2
3	Dokter spesialis ditingkatkan	3
4	Pelayanan kualitasnya ditingkatkan	1

Lampiran 3 Proses c*asefolding*

```
#lower case (Case folding)
data["Kritik"] = data["Kritik"].str.lower()
data.head()
```

	Kritik	Bidang
0	pelayanan agar lebih ditingkatkan	1
1	lahan parkir diperluas	2
2	loket diperbanyak, ada tempat bermain untuk an...	2
3	dokter spesialis ditingkatkan	3
4	pelayanan kualitasnya ditingkatkan	1

Lampiran 4 Proses *cleaning*

```
import string
import re #regex lib

#Cleaning
def hapus_format(text):
    #hapus tab, new line, ans back slice
    return text.replace('\\t'," ").replace('\\n'," ").replace('\\u'," ").replace('\\'," ")

data["Kritik"] = data["Kritik"].apply(hapus_format)

def hapus_ascii(text):
    #hapus non ASCII (emoticon, chinese word, dll)
    return text.encode('ascii', 'replace').decode('ascii')

data["Kritik"] = data["Kritik"].apply(hapus_ascii)

def hapus_number(text):
    #hapus number
    return re.sub(r"\d+", " ", text)

data["Kritik"] = data["Kritik"].apply(hapus_number)

def hapus_punc(text):
    #hapus punctuation
    return text.translate(str.maketrans("","",string.punctuation))

data["Kritik"] = data["Kritik"].apply(hapus_punc)

def hapus_spasi(text):
    #hapus spasi berlebih
    return re.sub('\s+', ' ', text)

data["Kritik"] = data["Kritik"].apply(hapus_spasi)

def hapus_single(text):
    #hapus single char
    return re.sub(r"\b[a-zA-Z]\b", " ", text)

data["Kritik"] = data["Kritik"].apply(hapus_single)

#hapus spasi awal dan akhir
def hapus_awal_akhir(text):
    return re.sub('\s+',' ',text)

data["Kritik"] = data["Kritik"].apply(hapus_awal_akhir)

display(data)
```

Lampiran 5 Hasil proses *cleaning*

	Kritik	Bidang
0	pelayanan agar lebih ditingkatkan	1
1	lahan parkir diperluas	2
2	loket diperbanyak ada tempat bermain untuk ana...	2
3	dokter spesialis ditingkatkan	3
4	pelayanan kualitasnya ditingkatkan	1
...	...	...
1026	pegawainya ramah dan cepat penanganan nya	3
1027	bisa rapid antigen untuk naik pesawat	1
1028	pelayanan lama dan perawat yang mengobrol	1
1029	pelayanan dan informasi sangat mudah	4
1030	bagian depan kasir kurang koordinasi	4

1031 rows × 2 columns

Lampiran 6 Proses *tokenizing*

```
#tokenize

from nltk.tokenize import word_tokenize

data["token"]=data["Kritik"].apply(word_tokenize)
data["token"].head()
```

[11]

```
0                [pelayanan, agar, lebih, ditingkatkan]
1                             [lahan, parkir, diperluas]
2    [loket, diperbanyak, ada, tempat, bermain, unt...
3                    [dokter, spesialis, ditingkatkan]
4                [pelayanan, kualitasnya, ditingkatkan]
Name: token, dtype: object
```

Lampiran 7 Proses *normalization*

```
#normalization

kata_normal = pd.read_excel("normalization list.xlsx")

normal_dict={}

for index, row in kata_normal.iterrows():
    if row[0] not in normal_dict:
        normal_dict[row[0]]=row[1]

def normal_kata(document):
    return [normal_dict[term] if term in normal_dict else term for term in document]

data["normalize"]=data["token"].apply(normal_kata)

data["normalize"].head(10)
```

✓ 0.4s

```
0              [pelayanan, agar, lebih, ditingkatkan]
1                          [lahan, parkir, diperluas]
2    [loket, diperbanyak, ada, tempat, bermain, unt...
3                  [dokter, spesialis, ditingkatkan]
4              [pelayanan, kualitasnya, ditingkatkan]
5            [parkiran, agar, lebih, diperluas, lagi]
6               [waktu, pelayanan, agar, dipercepat]
7                      [parkir, diatur, lebih, baik]
8    [lahan, parkir, diperluas, khususnya, untuk, p...
9    [membuat, website, untuk, daftar, online, agar...
Name: normalize, dtype: object
```

Lampiran 8 Proses *stopwords*

```
#stopwords

from nltk.corpus import stopwords

#stopwords from nltk
list_stopword = stopwords.words('indonesian')

#tambahan kata stopwords manual
list_stopword.extend(["yg", "dg", "rt", "dgn", "ny", "d", 'klo',
                      'kalo', 'amp', 'biar', 'bikin', 'bilang',
                      'gak', 'ga', 'krn', 'nya', 'nih', 'sih',
                      'si', 'tau', 'tdk', 'tuh', 'utk', 'ya',
                      'jd', 'jgn', 'sdh', 'aja', 'n', 't',
                      'nyg', 'hehe', 'pen', 'u', 'nan', 'loh', 'rt',
                      '&amp', 'yah', 'gk', 'dtg', 'msh', 'tdi'])

#corpus stopwords Tala
tala_stopword = pd.read_csv("tala-stopwords.txt", names=["stopwords"], header=None)

#add tala corpus on list
list_stopword.extend(tala_stopword["stopwords"][0].split(' '))

#membuat corpus stopword
list_stopwords = set(list_stopword)

#menghapus stopwords pada list token
def hapus_stopwords(words):
    return [words for words in words if words not in list_stopwords]

data["normalize_stopwords"] = data["normalize"].apply(hapus_stopwords)

data["normalize_stopwords"].head(10)

list_stopwords.head()
```

Lampiran 9 Hasil proses *stopwords*

```
0                    [pelayanan, ditingkatkan]
1                  [lahan, parkir, diperluas]
2    [loket, diperbanyak, bermain, anak, bosan]
3             [dokter, spesialis, ditingkatkan]
4       [pelayanan, kualitasnya, ditingkatkan]
5                         [parkiran, diperluas]
6                      [pelayanan, dipercepat]
7                                [parkir, diatur]
8     [lahan, parkir, diperluas, parkir, mobil]
9           [website, daftar, online, menunggu]
Name: normalize_stopwords, dtype: object
```

Lampiran 10 Proses *stemming*

```
#import sastrawi package
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import swifter

#create stemmer
factory = StemmerFactory()
stemmer= factory.create_stemmer()

#stemmed
def stemmed_wrap(term):
    return stemmer.stem(term)

term_dict={}

for document in data["normalize_stopwords"]:
    for term in document:
        if term not in term_dict:
            term_dict[term]=' '

print(len(term_dict))
print("------------------")

for term in term_dict:
    term_dict[term] = stemmed_wrap(term)
    print (term, ":" , term_dict[term])

print(term_dict)
print("------------------")

#add column stemmed term
def get_term(document):
    return [term_dict[term] for term in document]

data["normalize_stop_stem"] = data["normalize_stopwords"].swifter.apply(get_term)

data.head(10)
```

Lampiran 11 Hasil proses *stemming*

```
pelayanan : layan
ditingkatkan : tingkat
lahan : lahan
parkir : parkir
diperluas : luas
loket : loket
diperbanyak : banyak
bermain : main
anak : anak
bosan : bosan
dokter : dokter
spesialis : spesialis
kualitasnya : kualitas
parkiran : parkir
dipercepat : cepat
diatur : atur
mobil : mobil
website : website
daftar : daftar
online : online
menunggu : tunggu
sarana : sarana
prasarana : prasarana
...
rapid : rapid
pesawat : pesawat
```

Lampiran 12 Hasil proses *casefolding* hingga *stemming*

	Kritik	Bidang	token	normalize	normalize_stopwords	normalize_stop_stem
0	pelayanan agar lebih ditingkatkan	1	[pelayanan, agar, lebih, ditingkatkan]	[pelayanan, agar, lebih, ditingkatkan]	[pelayanan, ditingkatkan]	[layan, tingkat]
1	lahan parkir diperluas	2	[lahan, parkir, diperluas]	[lahan, parkir, diperluas]	[lahan, parkir, diperluas]	[lahan, parkir, luas]
2	loket diperbanyak ada tempat bermain untuk ana...	2	[loket, diperbanyak, ada, tempat, bermain, unt...	[loket, diperbanyak, ada, tempat, bermain, unt...	[loket, diperbanyak, bermain, anak, bosan]	[loket, banyak, main, anak, bosan]
3	dokter spesialis ditingkatkan	3	[dokter, spesialis, ditingkatkan]	[dokter, spesialis, ditingkatkan]	[dokter, spesialis, ditingkatkan]	[dokter, spesialis, tingkat]
4	pelayanan kualitasnya ditingkatkan	1	[pelayanan, kualitasnya, ditingkatkan]	[pelayanan, kualitasnya, ditingkatkan]	[pelayanan, kualitasnya, ditingkatkan]	[layan, kualitas, tingkat]
5	parkiran agar lebih diperluas lagi	2	[parkiran, agar, lebih, diperluas, lagi]	[parkiran, agar, lebih, diperluas, lagi]	[parkiran, diperluas]	[parkir, luas]
6	waktu pelayanan agar dipercepat	4	[waktu, pelayanan, agar, dipercepat]	[waktu, pelayanan, agar, dipercepat]	[pelayanan, dipercepat]	[layan, cepat]
7	parkir diatur lebih baik	3	[parkir, diatur, lebih, baik]	[parkir, diatur, lebih, baik]	[parkir, diatur]	[parkir, atur]
8	lahan parkir diperluas khususnya untuk parkir ...	2	[lahan, parkir, diperluas, khususnya, untuk, p...	[lahan, parkir, diperluas, khususnya, untuk, p...	[lahan, parkir, diperluas, parkir, mobil]	[lahan, parkir, luas, parkir, mobil]
9	membuat website untuk daftar online agar tidak...	4	[membuat, website, untuk, daftar, online, agar...	[membuat, website, untuk, daftar, online, agar...	[website, daftar, online, menunggu]	[website, daftar, online, tunggu]

Lampiran 13 Fungsi pembobotan dengan TF-IDF

```
def tf_idf(data, max_features=2000, ngram=(1,1)):
    # Hitung T vektor
    tf_vect = CountVectorizer(max_features=max_features, ngram_range=ngram)
    TF_vect = tf_vect.fit_transform(data["Join_Kata"])
    # Normalisasi tf vektor
    normal_tf_vect = normalize(TF_vect, norm='l1', axis=1)
    # Hitung IDF
    idf = TfidfVectorizer(max_features=max_features, smooth_idf=False, ngram_range=ngram)
    tfs = idf.fit_transform(data["Join_Kata"])
    # Hitung tf x idf
    tfidf = normal_tf_vect.multiply(tfs).toarray()
    tfidf = tfidf.astype(np.float64)
    # print(idf.get_feature_names())

    return tfidf
```

Lampiran 14 Fungsi *KFold Crossvalidation*

```
# KFold Cross Validation
from sklearn.metrics import r2_score, precision_score, classification_report
def fungsiFold(model, _X, _y, _cv, return_train_score=True):
    _score = {
        'acc' : 'accuracy',
        'prec' : make_scorer(precision_score, average='macro', zero_division=0),
        'rec' : make_scorer(recall_score, average='macro'),
        'f1' : make_scorer(f1_score, average='macro')
    }
    _X = np.array(_X)
    _y = _y.values.reshape(1031,)
    return cross_validate(estimator=model, X=_X, y=_y, cv=_cv ,scoring=_score, return_train_score=True, error_score=True)
```

Lampiran 15 Fungsi model SVM OvO

```
clf = svm.SVC(C=10, kernel='linear', decision_function_shape='ovo', max_iter=10000)
```

Lampiran 16 Fungsi model SVM OvR

```
1  clfOvR = svm.LinearSVC(C=10, multi_class='ovr', max_iter=10000)
```