Contents Description Assignments Grading Materials Absences

BINAR PLATINUM CHALLENGE

Analisis sentimen melalui flask API

Start Here

Contents Description Assignments Grading Materials Absences

Table of Contents.

#1

Pendahuluan

Kehadiran dunia yang saling berhubungan secara digital menghadirkan celah yang lebih luas dalam membagikan opini berdiskusi secara bebas melewati sosial media dan dunia virtual.

Kecendrungan prilaku manusia untuk mengekspresikan pendapat maupun komentar didukung penuh oleh pemilik sosial media maupun e-commerce untuk menaikan traffic mereka. Mulai dari ruang diskusi, kontroversi, maupun interaksi antar pengguna kerap kali memberikan efek positive maupun negative didalam ruang digital. Nuansa-nuansa didalam sebuah komentar didalam diskusi tersebut dapat diekstraksi dan dapat diimaanfatkan berbagai pihak. Sebagai potensi mendeteksi arus yang terjadi didalam sebuah ruang diskusi virtual untuk menghasilkan user engagement yang lebih baik.

#2

Metode Penelitian

Didalam studi ini kami mencoba untuk membedah lebih dalam dari data yang disediakan dalam bentuk teks yang sudah dilengkapi dengan label sentimen terhadap kalimat-kalimat yang ada.

pada penelitian kali ini, peneliti menggunakan metode LSTM (*Long Short Term Memory*) dan *Neural Network*. Penggunaan LSTM diharapkan dapat memberikan hasil terbaik untuk sequential data dan neural network dipilih dengan pertimbangan model yang dapat ditraining dengan jangka waktu lebih efisien.

Workflow

1

Dataset Preparation

menyiapkan *raw* data yang akan digunakan untuk proses training

2

Exploratory Data Analysis

investigasi terhadap data untuk mengetahui bentuk dan karakteristik data itu sendiri

3

Data Cleaning Process

membersihkan data dengan metode text normalization

4

Feature Extraction

Ekstraksi fitur text yang mengubah data mentah menjadi bentuk yang dapat diproses oleh machine learning model

Workflow

5

Model Training Process (LSTM & NN)

proses belajar model untuk menghasilkan output sentimen yang diharapkan

6

Evaluation

evaluasi pad kualitas parameter output model

7

Visualization

visualisasi terhadap *learning curve* pada kedua model untuk dianalisa

8

API Creation (*GUI*)

memproduksi API yang dapat digunakan oleh user untuk mendapatkan output sentimen yang akurat

#1 activity

Dataset Preparation

```
[2] # import library yang diperlukan
    # membaca TSV TXT file menggunakan pandas
    import pandas as pd
    import re
    import sklearn
    import nltk

    nltk.download('stopwords')

    !pip install Sastrawi
    from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Collecting Sastrawi
  Downloading Sastrawi-1.0.1-py2.py3-none-any.whl.metadata (909 bytes)
Downloading Sastrawi-1.0.1-py2.py3-none-any.whl (209 kB)
   ---------------------------------------- 209.7/209.7 kB 3.5 MB/s eta 0:00:00
Installing collected packages: Sastrawi
Successfully installed Sastrawi-1.0.1
```

```
[3] df = pd.read_csv('train_preprocess.tsv.txt', delimiter = "\t", names = ['text','label'])
```

#2 activity

EDA (*Exploratory Data Analysis*)

```
[ ] df
```

	text	label
0	warung ini dimiliki oleh pengusaha pabrik tahu...	positive
1	mohon ulama lurus dan k212 mmbri hujjah partai...	neutral
2	lokasi strategis di jalan sumatera bandung . t...	positive
3	betapa bahagia nya diri ini saat unboxing pake...	positive
4	duh . jadi mahasiswa jangan sombong dong . kas...	negative
...	...	...
10995	tidak kecewa	positive
10996	enak rasa masakan nya apalagi kepiting yang me...	positive
10997	hormati partai-partai yang telah berkoalisi	neutral
10998	pagi pagi di tol pasteur sudah macet parah , b...	negative
10999	meskipun sering belanja ke yogya di riau junct...	positive

11000 rows × 2 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
[ ] df.shape
```

```
(11000, 2)
```

```
[ ] label=df.label.value_counts()
    label
```

label	count
positive	6416
negative	3436
neutral	1148

dtype: int64

#3 activity

Data Cleaning Process

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

```
[ ] # fungsi stemming
    def stemming(text):
      text = stemmer.stem(text)
      return text
```

Regex Cleaning

```
def cleaning(text):
  # membuat tulisan lower case
  text = text.lower()
  # menghilangkan whitespaces didepan & belakang
  text = text.strip()

  # menghilangkan USER tag
  text = re.sub('user', ' ', text)
  # menghilangkan URL tag
  text = re.sub('url', ' ', text)
  # menghilangkan "RT" tag
  text = re.sub('rt', ' ', text)
  # menghilangkan random url
  text = re.sub(r'https?:[^\s]+', '', text)

  # menghilangkan tab
  text = re.sub('\t', ' ', text)
  #  menghilangkan random /xf character
  text = re.sub('x[a-z0-9]{2}', ' ', text)
  #  menghilangkan code "newline"
  text =  text.replace('\\n', ' ')
  # menghilangkan symbol tersisa
  text = re.sub(r'[^a-zA-Z0-9]', ' ', text)
  text = re.sub(r"[-()\"#/@;:{}`+=~|.!?,'0-9]", "  ", text)
  # menghilangkan sisa whitespaces
  text = re.sub(r' \s+', ' ',text)
  # menghilangkan whitespaces kembali
  text = text.strip()


  return text
```

#4 activity

Feature Extraction

Bag of Words

bag of word saya pakai karna memberikan prediksi yang lebih baik pada prediksi sentimen

```
[6] df = pd.read_csv('stemmed.csv')
```

```
[7] data_preprocessed = df.text_clean.tolist()
```

```
[8] from sklearn.feature_extraction.text import CountVectorizer
```

```
[9] count_vect = CountVectorizer()
    count_vect.fit(data_preprocessed)

    X = count_vect.transform(data_preprocessed)
    print('data exctraction is done')
```

```
data exctraction is done
```

```
[10] import pickle

     pickle.dump(count_vect, open('feature.p', 'wb'))
```

LSTM Tokenization Feature Extraction

```
[ ] import pickle
    import setuptools.dist
    from tensorflow.keras.preprocessing.text import Tokenizer
    from tensorflow.keras.preprocessing.sequence import pad_sequences
    from collections import defaultdict

    max_features = 100000
    tokenizer = Tokenizer(num_words=max_features, split=' ', lower=True)
    tokenizer.fit_on_texts(total_data)
    with open('tokenizer.pickle', 'wb') as handle:
      pickle.dump(tokenizer, handle, protocol=pickle.HIGHEST_PROTOCOL)
      print('tokenizier.pickle has created!')

    X = tokenizer.texts_to_sequences(total_data)

    vocab_size = len(tokenizer.word_index)
    maxlen = max(len(x) for x in X)

    X = pad_sequences(X)
    with open('x_pad_sequences.pickle', 'wb') as handle:
        pickle.dump(X, handle, protocol=pickle.HIGHEST_PROTOCOL)
        print('x_pad_sequences.pickle has been created!')
```

#5 activity

Model Training Process (NN & *LSTM*)

```
[14] from sklearn.neural_network import MLPClassifier

     model = MLPClassifier(activation='logistic')
     model.fit(X_train,y_train)

     print('training is done')
```

```
training is done
```

```
[15] import pickle
     pickle.dump(model, open('model-sentiment.p', 'wb'))
```

NN Training

```
embed_dim = 100
units = 428

modelL = Sequential()
modelL.add(Embedding(max_features, embed_dim, input_length=X.shape[1]))
modelL.add(LSTM(units, dropout=0.5))
modelL.add(Dense(3, activation='softmax'))

modelL.compile(loss = 'binary_crossentropy', optimizer='adam', metrics = ['accuracy'])
print(modelL.summary())

adam = optimizers.Adam(learning_rate=0.001)
modelL.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics = ['accuracy'])

es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=4, restore_best_weights=True)
history = modelL.fit(X_train, y_train, epochs=12, batch_size=15, validation_data=(X_test, y_test), verbose=1, callbacks=[es])
```

Model: "sequential_12"

Layer (type)	Output Shape	Param #
embedding_12 (Embedding)	?	0 (unbuilt)
lstm_12 (LSTM)	?	0 (unbuilt)
dense_12 (Dense)	?	0 (unbuilt)

```
 Total params: 0 (0.00 B)
 Trainable params: 0 (0.00 B)
 Non-trainable params: 0 (0.00 B)
None
Epoch 1/12
587/587 ━━━━━━━━━━━━━━━━━━━━ 123s 204ms/step - accuracy: 0.7251 - loss: 0.6713 - val_accuracy: 0.8559 - val_loss: 0.4098
Epoch 2/12
587/587 ━━━━━━━━━━━━━━━━━━━━ 126s 215ms/step - accuracy: 0.8976 - loss: 0.2854 - val_accuracy: 0.8700 - val_loss: 0.3499
Epoch 3/12
587/587 ━━━━━━━━━━━━━━━━━━━━ 119s 202ms/step - accuracy: 0.9289 - loss: 0.1853 - val_accuracy: 0.8823 - val_loss: 0.3590
Epoch 4/12
587/587 ━━━━━━━━━━━━━━━━━━━━ 119s 202ms/step - accuracy: 0.9615 - loss: 0.1108 - val_accuracy: 0.8923 - val_loss: 0.3387
Epoch 5/12
587/587 ━━━━━━━━━━━━━━━━━━━━ 118s 200ms/step - accuracy: 0.9752 - loss: 0.0828 - val_accuracy: 0.8905 - val_loss: 0.3978
Epoch 6/12
```

LSTM Training

#6 activity

Evaluation & Prediction (*NN*)

Classification report

```
[16] from sklearn.metrics import classification_report

     test = model.predict(X_test)

     print('testing is done')
     print(classification_report(y_test, test))
```

```
testing is done
              precision    recall  f1-score   support

    negative       0.74      0.71      0.73       714
     neutral       0.70      0.60      0.65       223
    positive       0.84      0.88      0.86      1263

    accuracy                           0.80      2200
   macro avg       0.76      0.73      0.74      2200
weighted avg       0.79      0.80      0.80      2200
```

Cross Validation

```
--------------------
Training ke -  5
              precision    recall  f1-score   support

    negative       0.70      0.76      0.73       670
     neutral       0.80      0.60      0.69       245
    positive       0.86      0.86      0.86      1285

    accuracy                           0.80      2200
   macro avg       0.79      0.74      0.76      2200
weighted avg       0.81      0.80      0.80      2200

====================


Rata-rata Accuracy: 0.7986363636363637
```

#6 activity

NN Prediction

```
[25] # original_text = 'mantap sekali rasa syukur dan rasa cukup.'
original_text = 'suka makan orang'

# Feature Extraction
text = count_vect.transform([cleansing(original_text)])

# Prediksi Sentimenya
result = model.predict(text)[0]
print('Text sentiment analysis:')
print()
print(result)
```

```
Text sentiment analysis:

negative
```

```
[26] # original_text = 'mantap sekali rasa syukur dan rasa cukup.'
original_text = 'Rasa, Syukur kita ucapkan'

# Feature Extraction
text = count_vect.transform([cleansing(original_text)])

# Prediksi Sentimenya
result = model.predict(text)[0]
print('Text sentiment analysis:')
print()
print(result)
```

```
Text sentiment analysis:

positive
```

Evaluation & Prediction (*NN*)

#6 activity

Evaluation & Prediction (*LSTM*)

Classification_report

```
[ ] from sklearn import metrics

    predictions = modelL.predict(X_test)
    y_pred = predictions
    matrix_test = metrics.classification_report(y_test.argmax(axis=1), y_pred.argmax(axis=1))

    print('testing is done')
    print(matrix_test)
```

```
69/69 ━━━━━━━━━━━━━━━━━━━━ 5s 67ms/step
testing is done
              precision    recall  f1-score   support

           0       0.84      0.89      0.86       685
           1       0.84      0.76      0.80       233
           2       0.93      0.92      0.93      1282

    accuracy                           0.89      2200
   macro avg       0.87      0.86      0.86      2200
weighted avg       0.89      0.89      0.89      2200
```

Cross-Validation

```
Training ke -  5
              precision    recall  f1-score   support

           0       0.85      0.85      0.85       685
           1       0.80      0.78      0.79       233
           2       0.92      0.93      0.92      1282

    accuracy                           0.89      2200
   macro avg       0.86      0.85      0.85      2200
weighted avg       0.89      0.89      0.89      2200

=====================



Rata-rata Accuracy: 0.8795454545454545
```

#6 activity

Evaluation & Prediction (*LSTM*)

```
[47] from keras.models import load_model

     input_text = ' makan bang, jangan diem diem bae'

     sentiment = ['negative', 'neutral', 'positive']

     text = [cleansing(input_text)]
     predicted = tokenizer.texts_to_sequences(text)
     guess = pad_sequences(predicted, maxlen=X.shape[1])

     model = load_model('modelLSTM.h5')
     prediction = model.predict(guess)
     polarity = np.argmax(prediction[0])

     print('Text: ', text[0])
     print('Sentiment: ', sentiment[polarity])
```

```
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile_metrics` will be empty until you train or evaluate the model.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 443ms/step
Text:  makan bang jangan diem diem bae
Sentiment:  positive
```

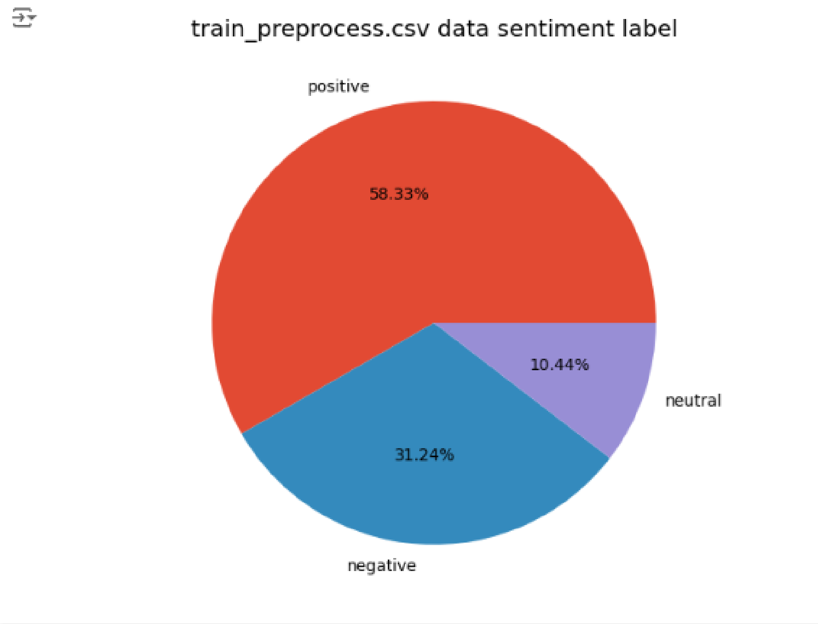
```
[48] from keras.models import load_model

     input_text = ' rasa syukur, cukup'

     sentiment = ['negative', 'neutral', 'positive']

     text = [cleansing(input_text)]
     predicted = tokenizer.texts_to_sequences(text)
     guess = pad_sequences(predicted, maxlen=X.shape[1])

     model = load_model('modelLSTM.h5')
     prediction = model.predict(guess)
     polarity = np.argmax(prediction[0])

     print('Text: ', text[0])
     print('Sentiment: ', sentiment[polarity])
```

```
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile_metrics` will be empty until you train or evaluate the model.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 318ms/step
Text:  rasa syukur cukup
Sentiment:  positive
```

#7 activity

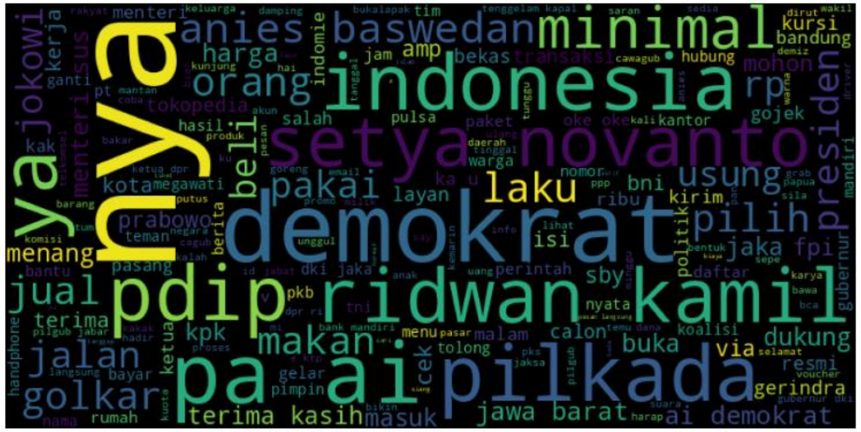
Visualization

```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.axis('equal')

ax.pie(count_value, labels = label_value, autopct='%1.2f%%')
plt.title('train_preprocess.csv data sentiment label')
plt.show()
```



EDA

```
label_neutral = df.loc[df['label'] == 'neutral']
text = ' '.join(map(str,(label_neutral['stopwords'])))

wordcloud = WordCloud(width=800, height=400).generate(text)
plt.figure(figsize=(12, 6))
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```




```
label_positive = df.loc[df['label'] == 'positive']
text = ' '.join(map(str,(label_positive['stopwords'])))

wordcloud = WordCloud(width=800, height=400).generate(text)
plt.figure(figsize=(12, 6))
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```

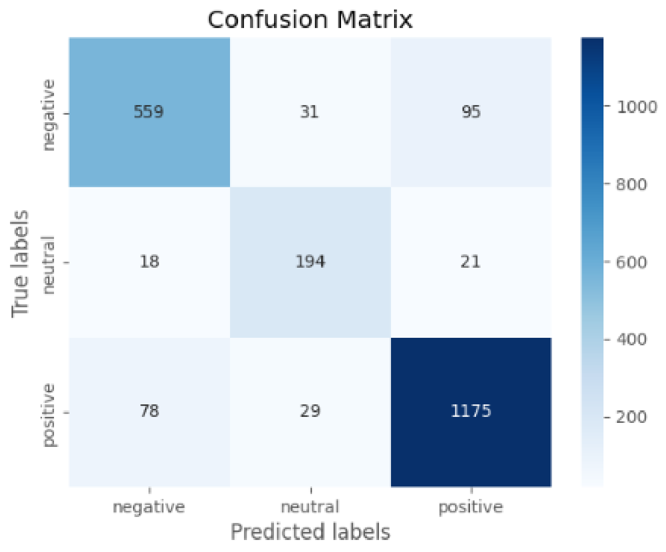
wordcloud EDA

#7 activity

Visualization

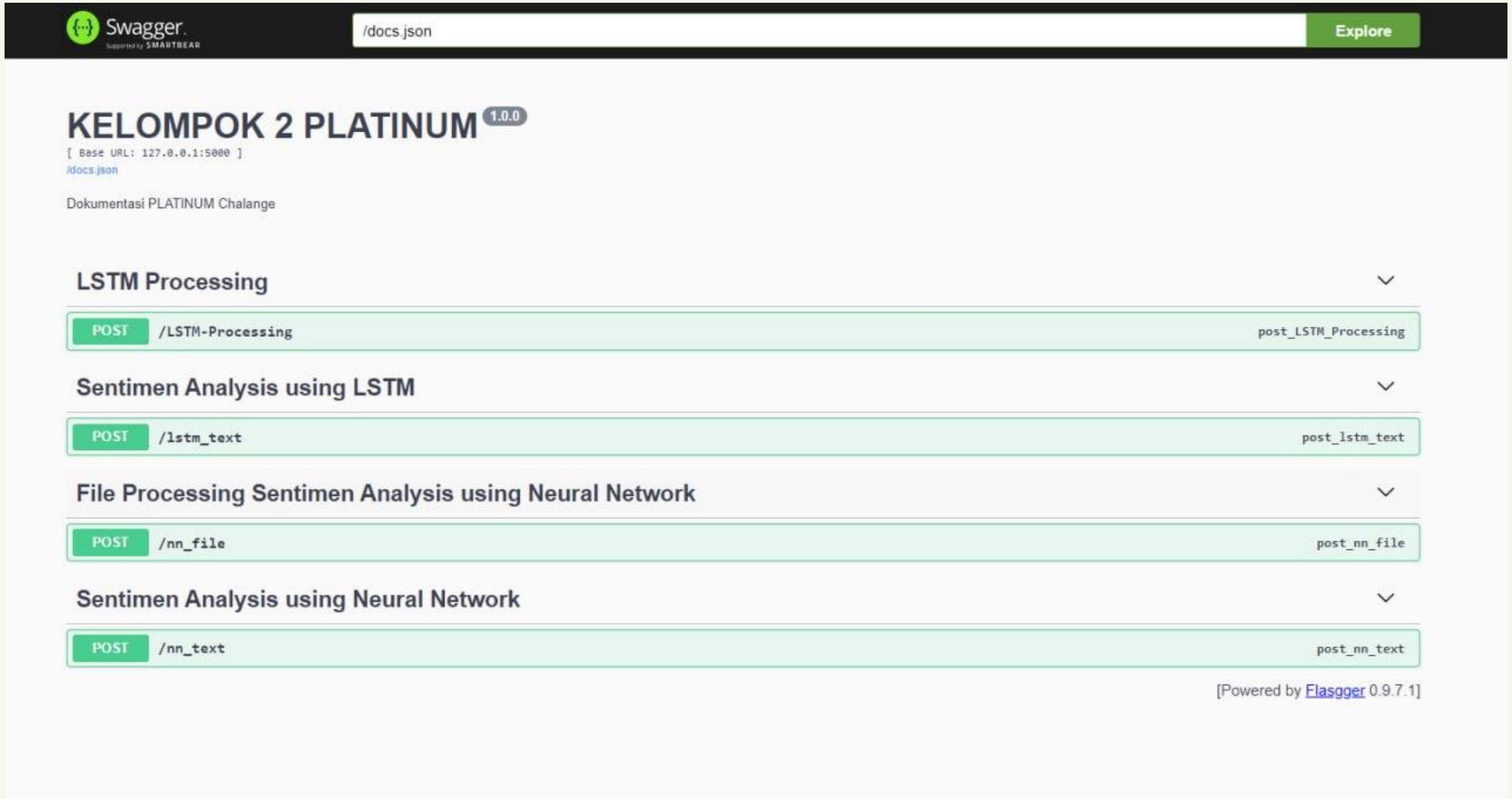
```
y_pred=np.argmax(y_pred, axis=1)
y_test=np.argmax(y_test, axis=1)
```

```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['negative', 'neutral', 'positive'], yticklabels=['negative', 'neutral', 'positive'])
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.title('Confusion Matrix')
plt.show()
```



Model Training Confusion Matrix

#8 activity



API Creation

#8 activity

API Creation

Responses

Response content type: application/json

Curl

```
curl -X POST "http://127.0.0.1:5000/nn_file" -H "accept: application/json" -H "Content-Type: multipart/form-data" -F "upload_file=@data.csv;type=text/csv"
```

Request URL

```
http://127.0.0.1:5000/nn_file
```

Server response

Code	Details
200	Response body


```
{
  "data": [
    {
      "sentiment": "negative",
      "text": "Tweet"
    },
    {
      "sentiment": "neutral",
      "text": "- disaat semua cowok berusaha melacak perhatian gue. loe lantas remehkan perhatian yg gue kasih khusus ke elo. basic elo cowok bego ! ! !'"
    },
    {
      "sentiment": "negative",
      "text": "RT USER: USER siapa yang telat ngasih tau elu?edan sarap gue bergaul dengan cigax jifla calis sama siapa noh licew juga'"
    },
    {
      "sentiment": "positive",
      "text": "41. Kadang aku berfikir, kenapa aku tetap percaya pada Tuhan padahal aku selalu jatuh berkali-kali. Kadang aku merasa Tuhan itu ninggalkan aku sendirian. Ketika orangtuaku berencana berpisah, ketika kakakku lebih memilih jadi Kristen. Ketika aku anak ter"
    },
    {
      "sentiment": "negative",
      "text": "USER USER AKU ITU AKU\\n\\nKU TAU MATAMU SIPIT TAPI DILIAT DARI MANA ITU AKU'"
    }
  ],
  "description": "Result of Sentiment Analysis using NN",
  "status_code": 200
}
```


Download

Response headers

```
connection: close
content-length: 997
content-type: application/json
date: Sun04 Aug 2024 07:09:25 GMT
server: Werkzeug/3.0.3 Python/3.9.6
```

Responses

Code	Description

Sentiment Generation

#4

Hasil dari Pemrosesan Data

Jika kalimat yang dimasukan positive maka akan muncul

#1

Curl

```
curl -X POST "http://127.0.0.1:5000/nn_text" -H "accept: application/json" -H "Content-Type: application/x-www-form-urlencoded" -d "text=sayang"
```

Request URL

```
http://127.0.0.1:5000/nn_text
```

Server response

Code	Details
200	Response body


```
{
  "data": {
    "sentiment": [
      "positive"
    ],
    "text": "sayang"
  },
  "description": "Result of Sentiment Analysis using NN",
  "status_code": 200
}
```


Download

Jika kalimat yang dimasukan negative maka akan muncul

#2

Curl

```
curl -X POST "http://127.0.0.1:5000/nn_text" -H "accept: application/json" -H "Content-Type: application/x-www-form-urlencoded" -d "text-babi%20kau"
```

Request URL

```
http://127.0.0.1:5000/nn_text
```

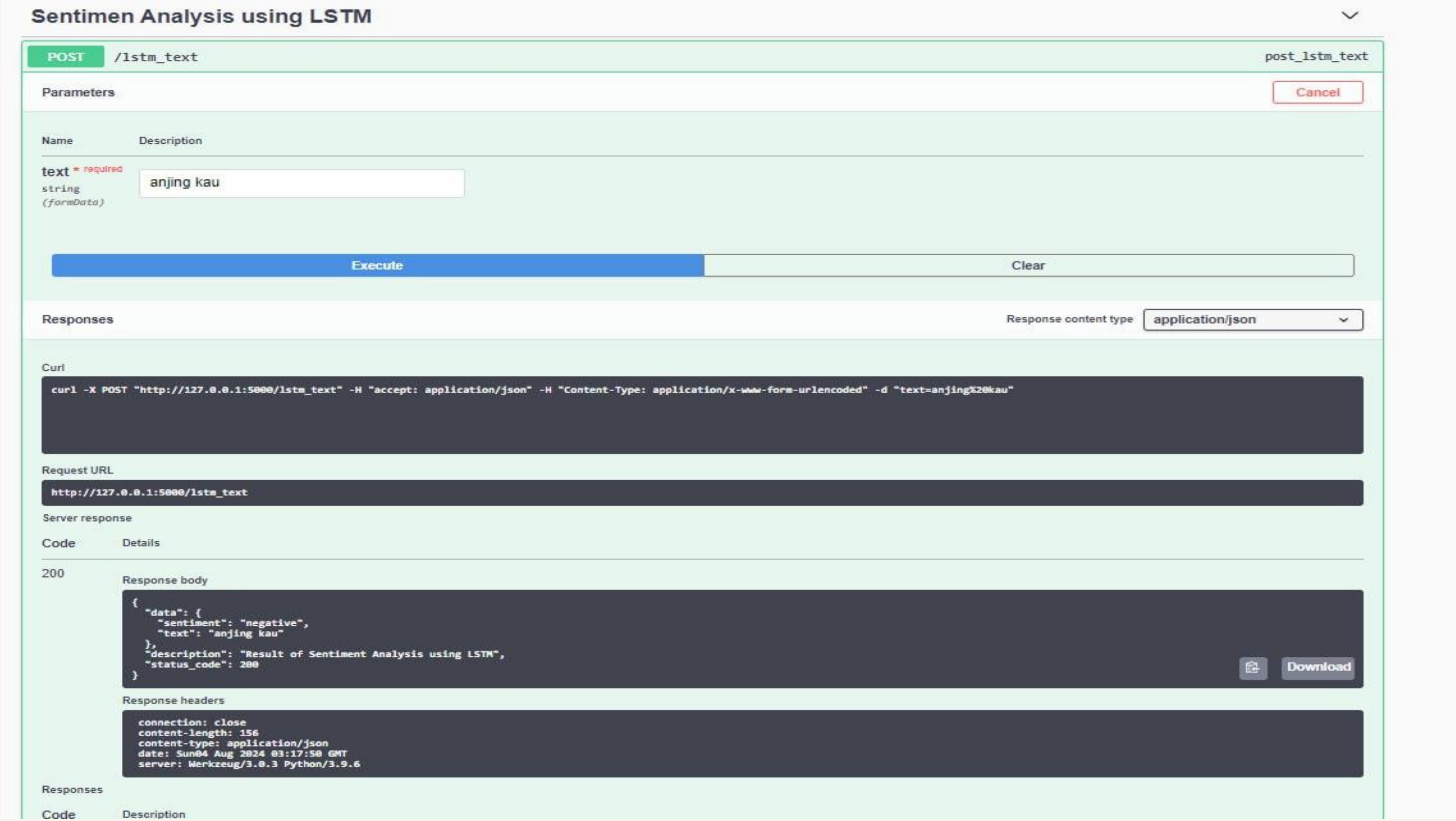
Server response

Code	Details
200	Response body

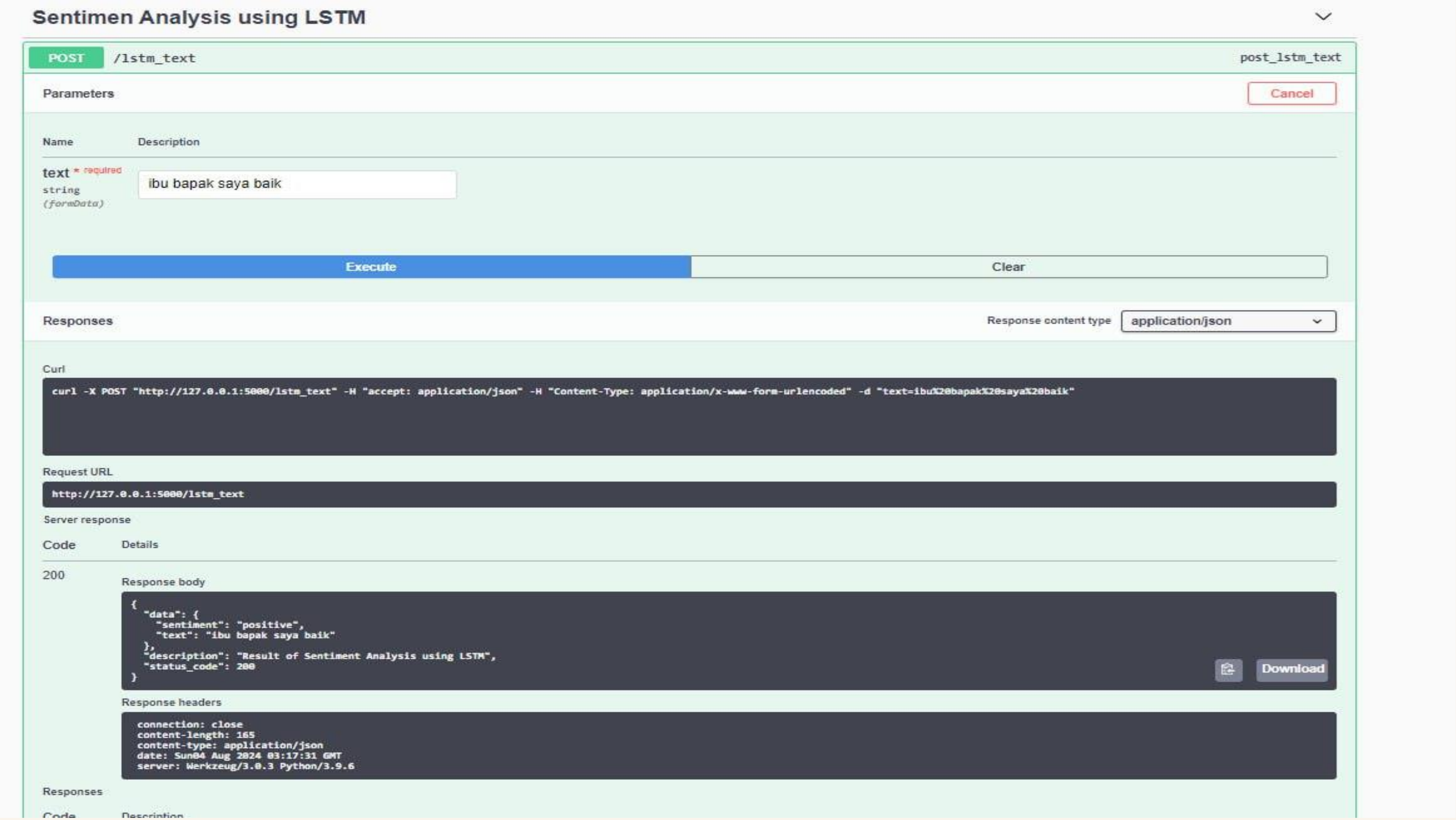

```
{
  "data": {
    "sentiment": [
      "negative"
    ],
    "text": "babi kau"
  },
  "description": "Result of Sentiment Analysis using NN",
  "status_code": 200
}
```


Download

#4 Hasil Deploy Data LSTM negative



#5 Hasil Deploy Data LSTM positive



Contents | Description | Assignments | Grading | Materials | Absences

Kesimpulan

Dari hasil prediksi yang dilakukan didalam API terhadap kedua model, menunjukkan hasil yang tidak begitu berbeda. yaitu, memberikan analisis sentimen yang cukup akurat didalam API. nilai akurasi yang dapat dihasilkan model NN sebesar 0.8 dan model LSTM sebesar 0,88-0,89. Dapat disimpulkan dari 100 kali tes, model LSTM seharusnya dapat memberikan prediksi lebih baik dengan ketidakakuratan serendah 12 hasil tes.

Contents | Description | Assignments | Grading | Materials | Absences

Thank you