**Laporan Study Case Course**

**Pemrosesan Bahasa Alami**

****

**Disusun oleh:**

**Rofiq Samanhudi**

**(202210370311260)**

**PROGRAM STUDI INFORMATIKA**

**FT UMM**

**2025**

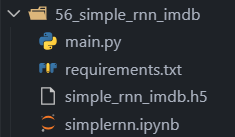
1. **Course Udemy**
   1. **Complete Machine Learning & NLP Bootcamp with MLOps Deployment**

Dalam era transformasi digital, kebutuhan akan keahlian di bidang Data Science, Kursus *Complete Machine Learning & NLP Bootcamp with MLOps Deployment* dari Udemy telah menjadi perjalanan pembelajaran yang sangat berharga dalam menguasai bidang *Data Science*, *Machine Learning* (ML), *Deep Learning* (DL), *Natural Language Processing* (NLP), dan *MLOps*. Kursus ini dirancang untuk membawa peserta dari tingkat pemula hingga mahir melalui pendekatan yang komprehensif, menggabungkan konsep teoretis, aplikasi praktis, dan penggunaan alat-alat standar industri. Kurikulumnya mencakup dasar-dasar algoritma ML dan NLP, topik lanjutan seperti *deep learning* dan *transformer models*, serta proyek dunia nyata di berbagai domain seperti kesehatan, keuangan, dan e-commerce.

Selama kursus, saya mempelajari prinsip-prinsip matematika yang mendasari algoritma ML dan NLP, seperti aljabar linier, kalkulus, dan teori probabilitas. Saya juga mendapatkan pengalaman praktis dengan alat-alat seperti *TensorFlow*, *PyTorch*, dan *scikit-learn*, serta memahami teknik optimasi seperti *hyperparameter tuning* dan evaluasi model. Laporan ini bertujuan untuk mendokumentasikan proyek-proyek yang telah saya selesaikan, keterampilan yang diperoleh, dan tantangan yang berhasil diatasi, dengan fokus pada salah satu proyek utama dari kursus ini.

* + 1. **Section 56: End-to-End Deep Learning Project with Simple RNN**
* **Deskripsi proyek**

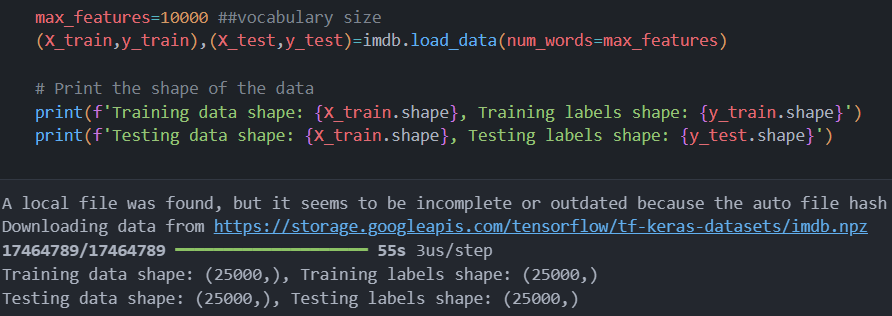
Pada bagian ke-56 dari kursus, saya mengerjakan proyek *end-to-end* yang berfokus pada pembangunan model *deep learning* menggunakan *Simple Recurrent Neural Network* (RNN) untuk analisis sentimen pada ulasan film dari dataset IMDB. Proyek ini mencakup seluruh siklus pengembangan model, mulai dari pra-pemrosesan data, pelatihan model, hingga penerapan model dalam aplikasi berbasis web menggunakan *Streamlit*. Tujuan proyek ini adalah untuk mengklasifikasikan ulasan film sebagai positif atau negatif berdasarkan teks yang diberikan.



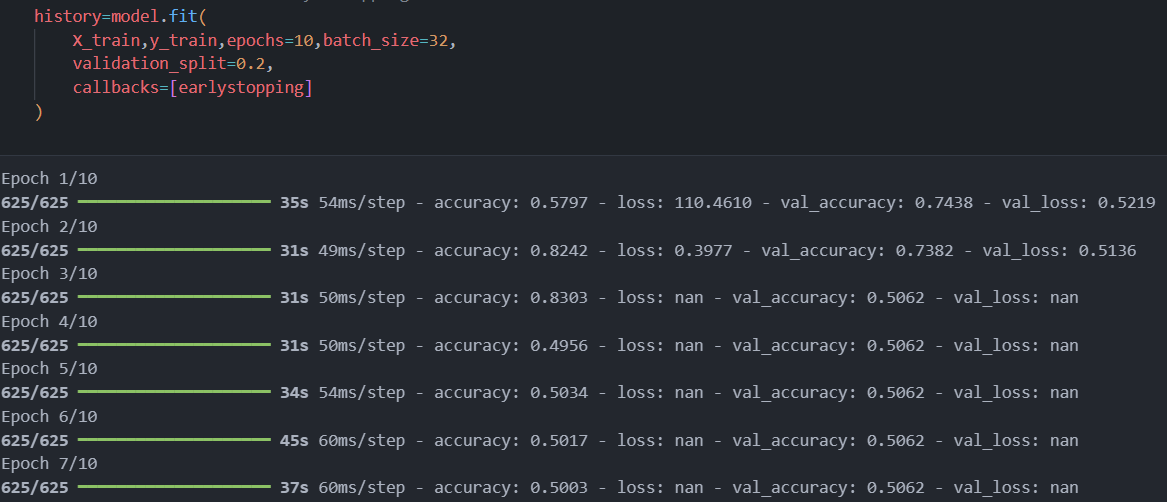
* **simplernn.ipynb**

File ini berisi kode untuk membangun dan melatih model *Simple RNN*. Langkah-langkah utama meliputi:

* Pemuatan dan Pra-pemrosesan Data: Dataset IMDB dimuat dengan ukuran kosakata sebanyak 10.000 kata. Ulasan diproses dengan *padding* untuk memastikan panjang *sequence* seragam (maksimum 500 kata).



* **Pembangunan Model:** Model dibangun menggunakan *TensorFlow* dengan arsitektur berlapis:
  + *Embedding Layer* untuk mengubah kata menjadi vektor berdimensi 128.
  + *Simple RNN Layer* dengan 128 unit dan aktivasi ReLU untuk memproses data berurutan.
  + *Dense Layer* dengan aktivasi *sigmoid* untuk klasifikasi biner (positif/negatif).
* **Pelatihan Model**: Model dilatih selama 10 epoch dengan batch size 32, menggunakan optimizer Adam dan loss function binary crossentropy. Early Stopping diterapkan untuk mencegah overfitting dengan memantau validation loss.



* **Penyimpanan Model:** Model yang telah dilatih disimpan dalam format .h5 untuk digunakan pada tahap penerapan.

Salah satu tantangan utama adalah ketidakstabilan pelatihan model, di mana nilai loss menjadi nan pada beberapa epoch. Hal ini kemungkinan disebabkan oleh aktivasi ReLU pada Simple RNN, yang dapat menyebabkan exploding gradients. Solusi potensial di masa depan adalah menggunakan aktivasi tanh atau menerapkan gradient clipping.



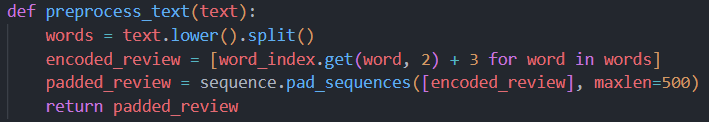
* **Main.py**

File ini berisi kode untuk membangun aplikasi web interaktif menggunakan *Streamlit* yang memungkinkan pengguna memasukkan ulasan film dan mendapatkan prediksi sentimen. Langkah-langkah utama meliputi:

* **Pemuatan Model**: Model yang telah dilatih dimuat dari file simple\_rnn\_imdb.h5.



* **Pra-pemrosesan Input Pengguna**: Teks ulasan pengguna diubah menjadi *sequence* numerik menggunakan *word index* dari dataset IMDB, kemudian diproses dengan *padding* agar sesuai dengan panjang input model (500 kata).



* **Prediksi Sentimen**: Model memprediksi sentimen ulasan, dengan skor di atas 0.5 diklasifikasikan sebagai positif dan di bawah 0.5 sebagai negatif.



* **Antarmuka Pengguna**: Aplikasi *Streamlit* menyediakan area teks untuk input pengguna, tombol untuk memicu prediksi, dan menampilkan hasil berupa label sentimen serta skor probabilitas.



Tantangan pada tahap ini adalah memastikan konsistensi pra-pemrosesan teks pengguna dengan data pelatihan, terutama dalam menangani kata-kata yang tidak ada dalam *word index*.

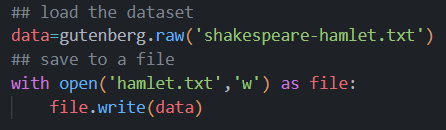
* + 1. **Section 58: LSTM and GRU End Deep Learning Project-Prediction Next Word**
* **Deskripsi Proyek**

Pada bagian ke-58 dari kursus *Complete Machine Learning & NLP Bootcamp with MLOps Deployment*, saya mengerjakan proyek *end-to-end* yang berfokus pada pembangunan model *deep learning* untuk memprediksi kata berikutnya dalam sebuah urutan teks menggunakan *Long Short-Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU). Proyek ini menggunakan teks *Hamlet* karya Shakespeare sebagai dataset, yang memberikan tantangan karena kekayaan dan kompleksitas bahasanya. Proyek ini mencakup siklus lengkap pengembangan model, mulai dari pengumpulan data, pra-pemrosesan, pembangunan model, pelatihan, evaluasi, hingga penerapan dalam aplikasi web interaktif menggunakan *Streamlit*. Tujuan utama proyek ini adalah membangun model yang mampu memprediksi kata berikutnya berdasarkan urutan kata yang diberikan oleh pengguna.

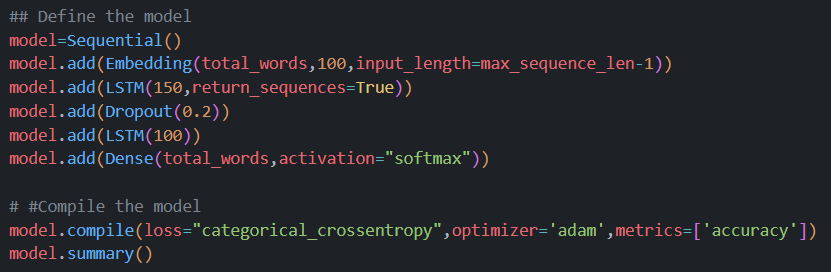
* **experiments.ipynb**

File ini berisi kode untuk pengumpulan data, pra-pemrosesan, pembangunan, dan pelatihan model LSTM serta GRU. Langkah-langkah utama meliputi:

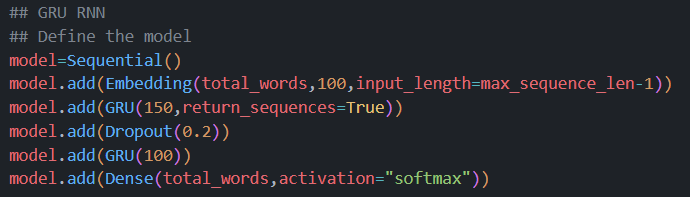
* **Pengumpulan Data**: Dataset diambil dari teks *Hamlet* yang tersedia melalui pustaka NLTK (*Natural Language Toolkit*). Teks disimpan dalam file hamlet.txt untuk diproses lebih lanjut.



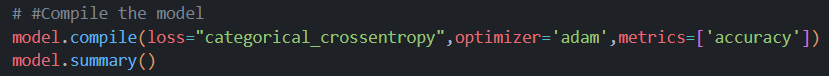
* **Pra-pemrosesan Data**:
  + Teks diubah menjadi huruf kecil untuk konsistensi.
  + *Tokenizer* dari *TensorFlow* digunakan untuk mengindeks kata-kata, menghasilkan total 4.818 kata unik (*total\_words*).
  + Urutan input dibuat dengan menggeser jendela kata untuk membentuk *sequence* (misalnya, "to be or" menjadi input untuk memprediksi "not").
  + *Sequence* dipadatkan (*padded*) ke panjang maksimum 14 kata menggunakan *pad\_sequences* untuk memastikan input seragam.
  + Data dibagi menjadi fitur (*x*: urutan kata) dan label (*y*: kata berikutnya), dengan label diubah ke format *one-hot encoding* menggunakan *to\_categorical*.
  + Dataset dibagi menjadi 80% data pelatihan dan 20% data pengujian menggunakan *train\_test\_split*.
* **Pembangunan Model**: Dua arsitektur model dibangun:
  + **LSTM Model**:
    - *Embedding Layer* dengan dimensi 100 untuk mengubah indeks kata menjadi vektor.
    - Dua lapisan LSTM (150 unit dengan *return\_sequences=True* dan 100 unit) untuk menangkap pola urutan.
    - *Dropout Layer* (0.2) untuk mencegah *overfitting*.
    - *Dense Layer* dengan aktivasi *softmax* untuk memprediksi probabilitas kata berikutnya.



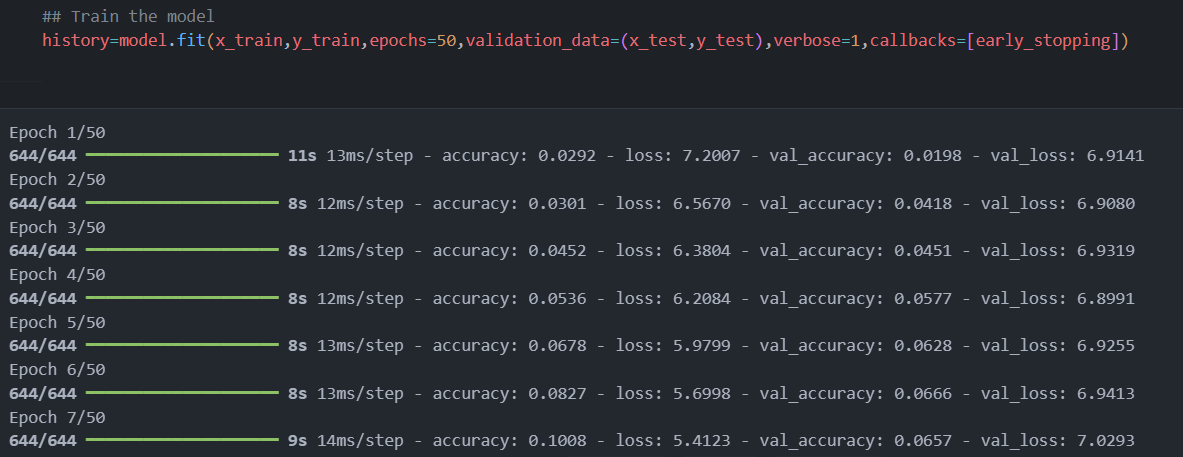
* **GRU Model**:
  + Struktur serupa dengan LSTM, tetapi menggunakan lapisan GRU (150 unit dan 100 unit) yang lebih efisien dalam beberapa kasus.



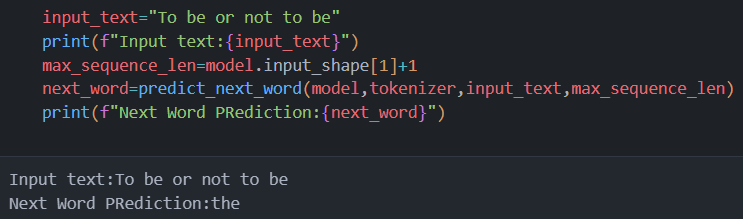
* Model dikompilasi dengan *loss function* *categorical\_crossentropy*, *optimizer* Adam, dan metrik akurasi.



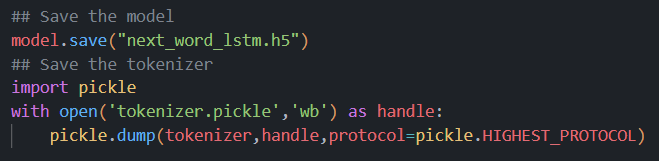
* **Pelatihan Model**:
  + Model dilatih selama maksimum 50 *epoch* dengan *Early Stopping* (memantau *validation loss* dengan *patience=3*) untuk mencegah *overfitting*.
  + Pelatihan dihentikan setelah 7 *epoch* karena *validation loss* tidak membaik, dengan akurasi pelatihan sekitar 10,08% dan *validation loss* sekitar 7,0293.
  + Tantangan utama adalah akurasi model yang relatif rendah, kemungkinan karena kompleksitas teks *Hamlet* dan keterbatasan ukuran dataset. Solusi potensial termasuk menambah data, menggunakan model pra-latih seperti BERT, atau menyesuaikan hiperparameter.



* **Evaluasi Model**:
  + Model diuji dengan memprediksi kata berikutnya untuk input seperti "To be or not to be" (diprediksi: "the") dan "Barn. Last night of all, When yond same" (diprediksi: "lord").
  + Hasil prediksi menunjukkan bahwa model mampu menangkap beberapa pola bahasa, tetapi kinerjanya terbatas pada konteks yang kompleks.



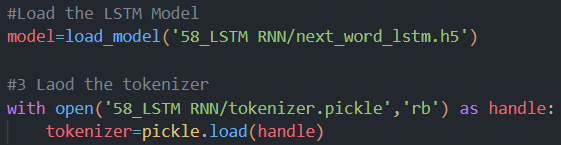
* **Penyimpanan Model**: Model disimpan sebagai next\_word\_lstm.h5, dan *tokenizer* disimpan sebagai tokenizer.pickle untuk digunakan pada tahap penerapan.



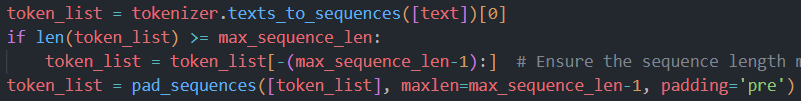
* **App.py**

File ini berisi kode untuk aplikasi *Streamlit* yang memungkinkan pengguna memasukkan urutan kata dan mendapatkan prediksi kata berikutnya. Langkah-langkah utama meliputi:

* + **Pemuatan Model dan Tokenizer**: Model LSTM/GRU dimuat dari next\_word\_lstm.h5, dan *tokenizer* dimuat dari tokenizer.pickle.



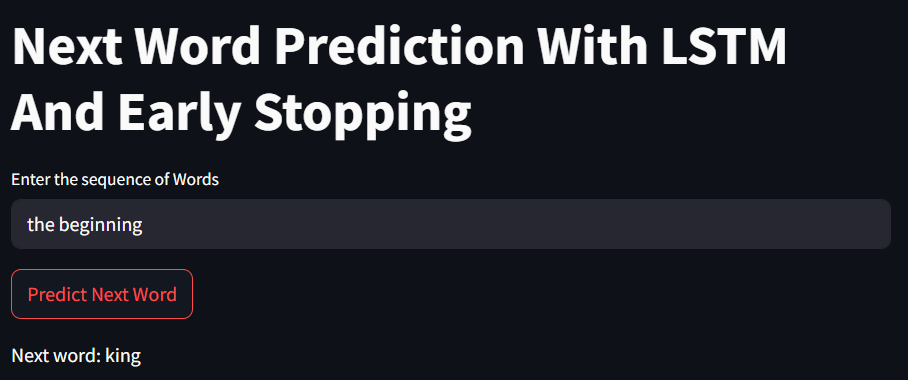
* + **Pra-pemrosesan Input Pengguna**:
    - Teks pengguna diubah menjadi *sequence* numerik menggunakan *tokenizer*.
    - *Sequence* dipotong atau dipadatkan ke panjang maksimum 13 kata (sesuai dengan input model) menggunakan *pad\_sequences*.



* + **Prediksi Kata Berikutnya**: Model memprediksi kata berikutnya dengan memilih indeks kata dengan probabilitas tertinggi (*argmax*).



* + **Antarmuka Pengguna**:
    - Aplikasi *Streamlit* menampilkan judul "Next Word Prediction With LSTM And Early Stopping".
    - Pengguna memasukkan teks melalui kolom input (default: "To be or not to").
    - Tombol "Predict Next Word" memicu prediksi, dan hasil ditampilkan dalam format "Next word: [kata]".



* + Tantangan pada tahap ini adalah memastikan pra-pemrosesan input pengguna konsisten dengan data pelatihan, terutama dalam menangani kata-kata di luar *word index* (diabaikan atau diganti dengan token default).

1. **Course MySkill**
   1. **MySkill: NLP Case Study - Sentiment Analysis**

* Informasi Proyek:

Nama: Ihza Mahendra  
Posisi: Data Science at Startup Company  
Judul Course: NLP Case Study: Sentiment Analysis  
Platform Implementasi: Google Colab  
Tipe Projek: Project Portofolio

* Deskripsi Proyek

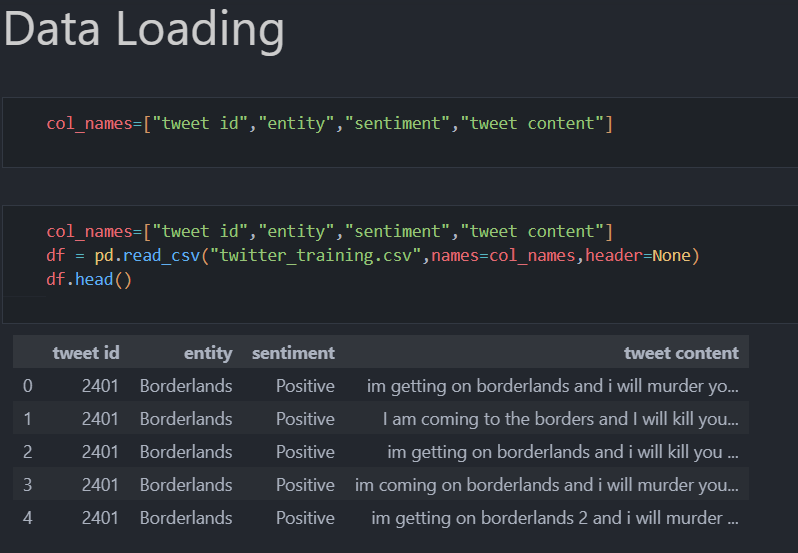
Projek ini merupakan bagian dari course MySkill dengan fokus pada studi kasus Natural Language Processing (NLP) terkait Sentiment Analysis. Projek ini mencakup beberapa tahapan utama, yaitu:

* **Data Cleaning**: Proses pembersihan data teks untuk memastikan kualitas data yang digunakan dalam analisis.
* **Exploratory Data Analysis (EDA)**: Eksplorasi data untuk memahami karakteristik dan pola dalam dataset.
* **Modelling**: Pembuatan model machine learning untuk melakukan analisis sentimen.

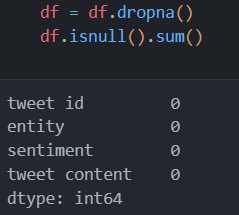
Materi ini diimplementasikan menggunakan bahasa pemrograman **Python** pada platform **Google Colab**, yang memungkinkan pengolahan data secara efisien dan interaktif. Projek ini bertujuan untuk mengaplikasikan teknik-teknik NLP dalam memproses dan menganalisis teks guna menentukan sentimen yang terkandung di dalamnya, seperti sentimen positif, negatif, atau netral.

* 1. **Tahapan Proyek**
     1. **Data Loading**

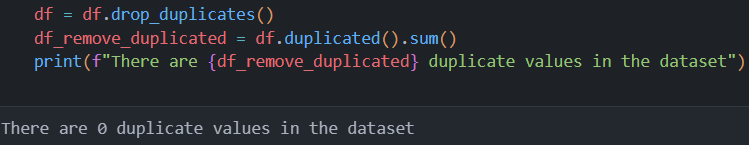
Pada tahap **Data Loading**, dataset yang digunakan adalah file twitter\_training.csv, yang berisi data tweet untuk analisis sentimen. Proses ini dilakukan untuk memuat data ke dalam lingkungan kerja agar dapat diolah lebih lanjut dalam tahapan analisis sentimen. Dataset ini mencakup informasi tweet dari berbagai entitas seperti Borderlands, Amazon, dan Microsoft, dengan label sentimen seperti Positive, Negative, Neutral, dan Irrelevant.

****

* + 1. **Data Cleaning**
* Remove null values
  + Dataset diperiksa untuk nilai null, terutama pada kolom tweet content dan sentiment, karena keduanya penting untuk analisis sentimen. Artikel MySkill menyebutkan bahwa terdapat 686 baris dengan nilai null pada kolom tweet content, yang harus dihapus untuk memastikan kualitas data.
  + Metode:
    - Menggunakan df.isnull().sum() untuk mengidentifikasi nilai null:



* Remove duplicate values
* Deskripsi: Dataset memiliki banyak duplikat pada kolom tweet content, sebagaimana ditunjukkan pada tahap Data Loading (contoh: tweet berulang dengan tweet id 2401). Artikel MySkill tidak menyebutkan langkah ini secara eksplisit, tetapi ini merupakan praktik standar untuk memastikan data unik.
* Metode:
  + Menggunakan df.duplicated(subset=['tweet content']).sum() untuk menghitung duplikat.
  + Menghapus duplikat menggunakan df.drop\_duplicates(subset=['tweet content']).



* Remove outliner
* Deskripsi: Outlier diidentifikasi berdasarkan panjang teks pada kolom tweet content. Artikel MySkill tidak membahas outlier secara langsung, tetapi menyebutkan bahwa tweet dengan panjang karakter yang tidak wajar (misalnya, terlalu pendek atau panjang) perlu diperhatikan. Saya mendefinisikan outlier sebagai teks dengan <3 kata atau >280 karakter (batas tweet pada masa pengumpulan data).
  + 1. **Exploratory Data Analysis**

Sebelum melakukan pemodelan, kita harus melihat terlebih dahulu distribusi atau kecenderungan dari dataset. Bila distribusi data cenderung memiliki sentimen positif, maka model yang kita bangun juga harus memiliki keluaran yang sama.

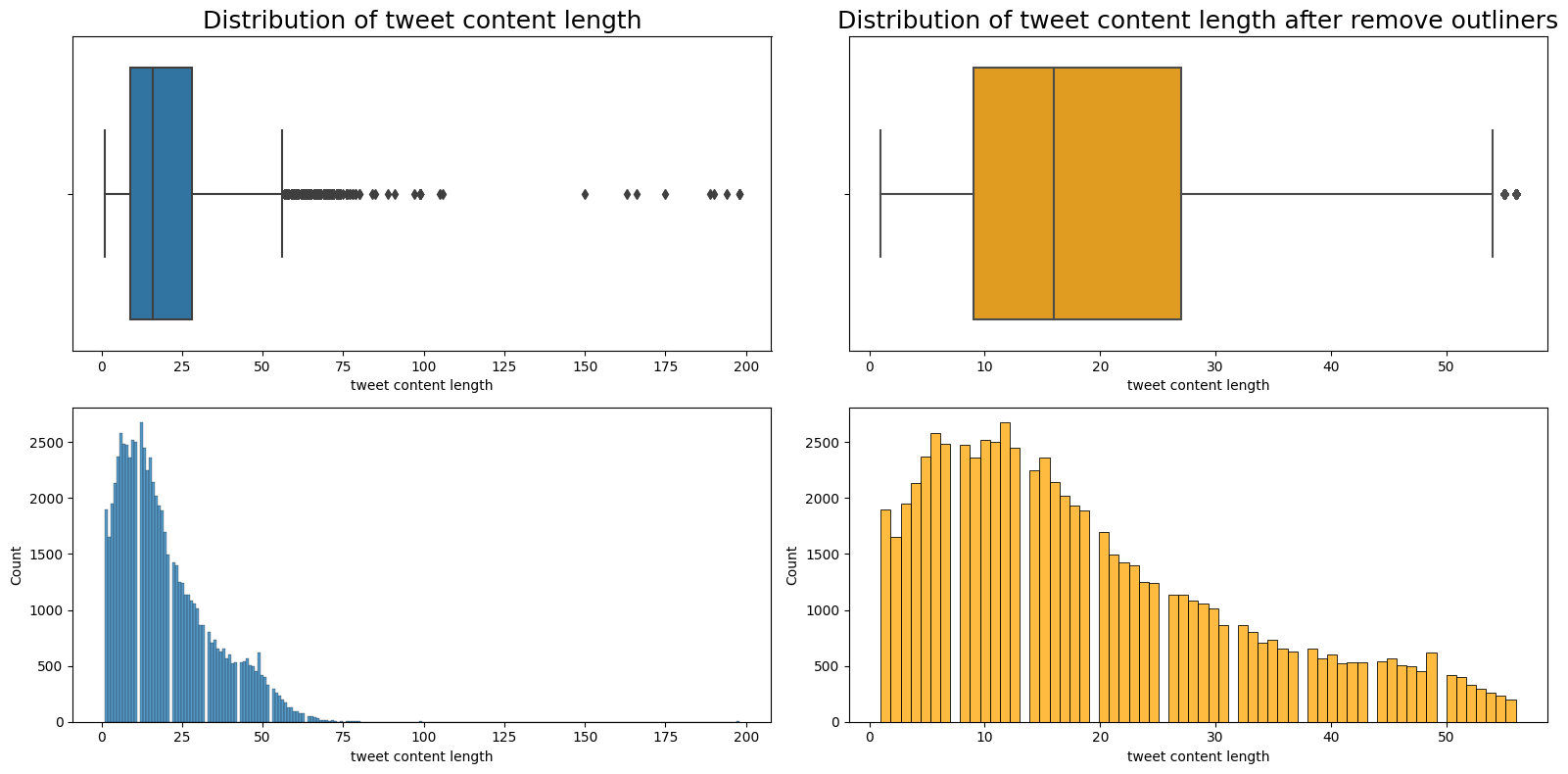
* Analysis Insight

****

Pada plot *wordcloud*, semakin besar ukuran kata pada sentimen tertentu menunjukkan frekuensinya semakin banyak. Pada gambar di atas, kata *game* adalah frekuensi terbanyak pada tiap *wordcloud* negatif, positif, netral, dan irelevan. Hal ini bisa saja terjadi karena kata *game* disandingkan juga dengan kata-kata lain yang memang berlabel sentimen positif, negatif, netral, atau irelevan.

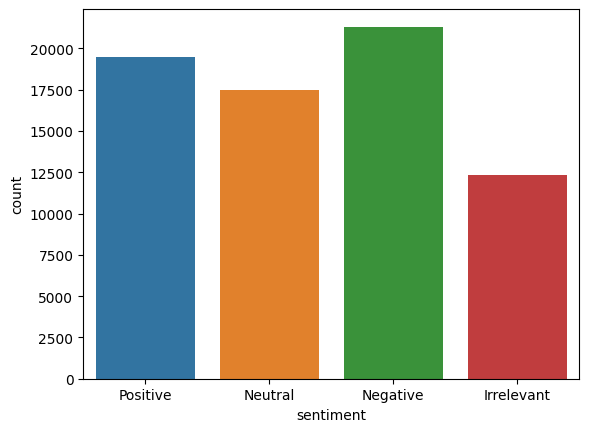
* + 1. **Text Preprocessing**

Pada kasus analisis sentimen, kita akan membuang data konten yang terlalu panjang, misalnya >50 kata.



Perbedaan distribusi sebelum dan setelah cleaning data. Sumber: dokumentasi pribadi.

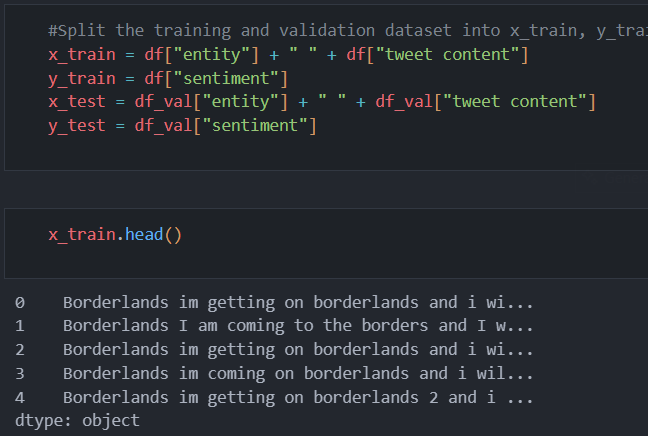
Setelah kita melakukan *cleaning data*, distribusi kelas terlihat hampir seragam. Dengan kata lain, selisih antar kelas sentimen tidak terlalu jauh.



Distribusi data sentimen setelah cleaning data. Sumber: dokumentasi pribadi.

* + 1. **Data Splitting**

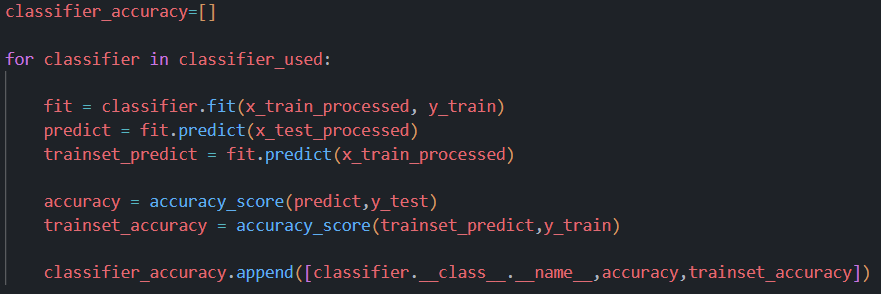
Tahap Data Splitting bertujuan untuk membagi dataset menjadi dua bagian: data latih (train) dan data uji (test). Data latih digunakan untuk melatih model machine learning, sedangkan data uji digunakan untuk mengevaluasi performa model pada data yang belum pernah dilihat sebelumnya. Dataset yang digunakan memiliki 72.745 baris setelah pembersihan, dengan kolom tweet content sebagai fitur utama dan sentiment sebagai label.

****

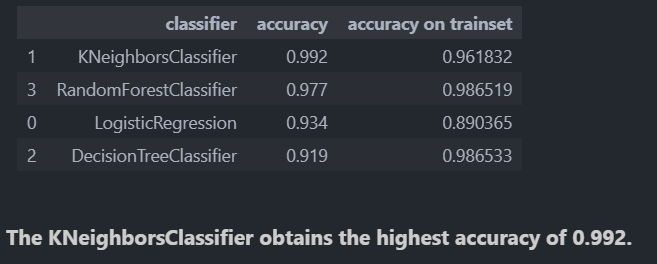
* + 1. **Models**

Tahap Modelling bertujuan untuk melatih model machine learning guna mengklasifikasikan sentimen tweet (Positive, Negative, Neutral, Irrelevant) berdasarkan teks pada kolom tweet content. Dataset telah melalui tahap Data Cleaning, EDA, Data Splitting, dan preprocessing menggunakan pipeline (CountVectorizer dan TfidfTransformer). Pada bagian ini, empat model classifier digunakan: Logistic Regression, KNeighborsClassifier, DecisionTreeClassifier, dan RandomForestClassifier. Model dievaluasi berdasarkan akurasi pada data latih dan data uji.

* Langkah-langkah Modelling
  + 1. Pemilihan Model
* Deskripsi: Empat model classifier dipilih untuk eksperimen:
  + LogisticRegression(): Model linier yang cocok untuk klasifikasi teks.
  + KNeighborsClassifier(): Model berbasis jarak yang efektif untuk data dengan fitur numerik seperti TF-IDF.
  + DecisionTreeClassifier(): Model berbasis pohon keputusan yang sederhana namun rentan terhadap overfitting.
  + RandomForestClassifier(): Model ensemble yang menggabungkan banyak pohon keputusan untuk meningkatkan akurasi dan mengurangi overfitting.



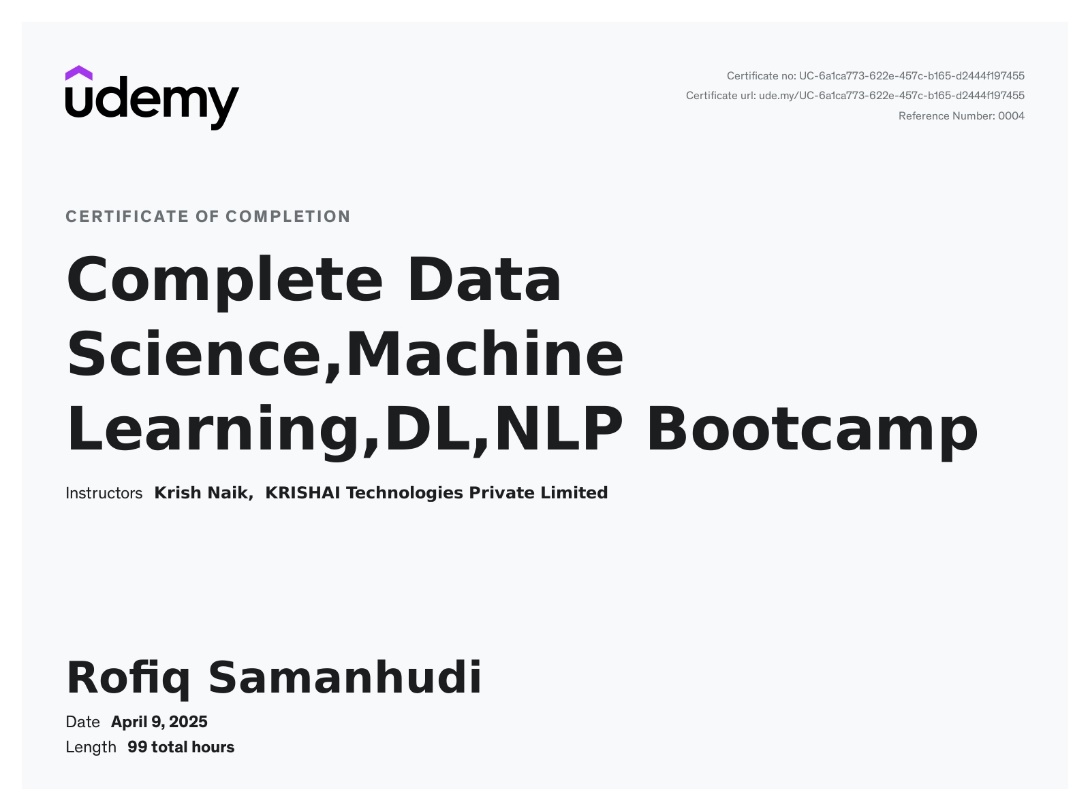
* + 1. Hasil dan Perbandingan Accuracy



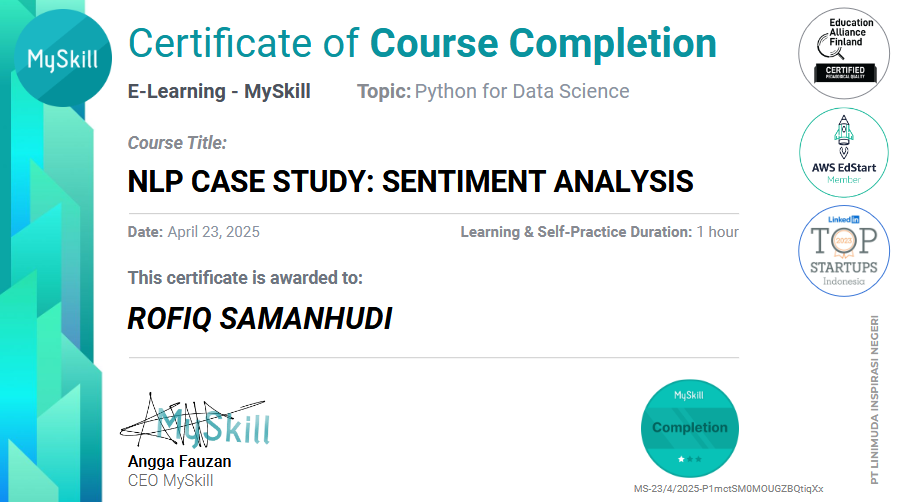
* Hasil: **KNeighborsClassifier** mencapai akurasi tertinggi pada data uji, yaitu **99.2%**, dengan akurasi pada data latih sebesar 96.18%.

1. Lampiran

* Sertifikat Udemy:



* Sertifikan MySkill:



* Link Git (SourceCode):

<https://github.com/Descanden/UTS-NLP.git>