**Sistem Rekomendasi Film(collaborative Filtering)**

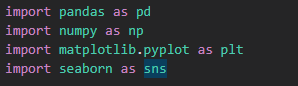
Percobaan kali membuat sistem rekomendasi film dengan metode *collaborative filtering* yang memanfaatkan *ratings* film dari tiap user untuk memprediksi *ratings* dari *user* lain yang nantinya menampilkan rekomendasi film lain. Tutorial pembuatan dan dataset sistem rekomendasi dapat diakses di link berikut:

<https://www.youtube.com/watch?v=6N2vo3JZg2c&list=PL3VpLbLyLE56IVsWwn_rlv9IGGNiyab9e&t=4127s>

<https://github.com/GilangAgungS/Film-Recommendation-Tensor>

Secara garis besar pembuatan sistem rekomendasi ini dibagi menjadi 4 tahap yaitu pengambilan dataset, memisahkan dataset menjadi data *train* dan data *test*, pembuatan model dengan Tensorflow, dan *training* model yang telah dibuat. Berikut adalah langkah-langkah dari pembuatan sistem rekomendasi film:

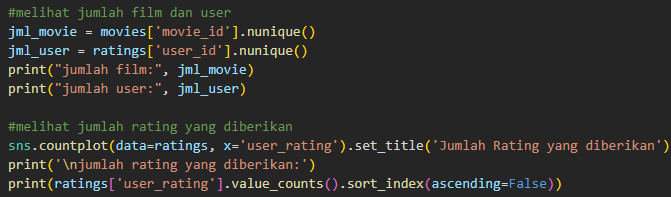
**1. *Import Library* untuk Mengolah Dataset**



**2. *Load* Dataset “ratings.csv” dan “movies.csv”**



**3. Melihat beberapa informasi dari data yang digunakan**

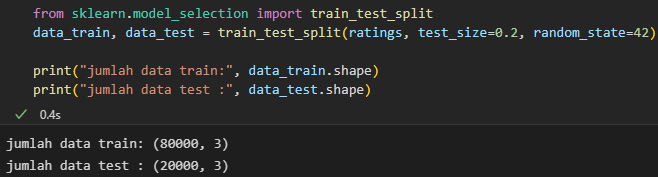


Berikut adalah hasil dari kode di atas:

|  |  |
| --- | --- |
|  |  |

Kebanyakan user memberikan *rating* 4 pada film yang ditonton

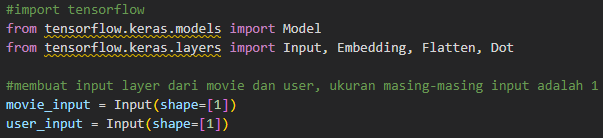
**4. Memisahkan data menjadi data *train* dan data *test***



Data dibagi dengan proporsi 20% untuk *testing* dan 80% untuk *training*, menggunakan random\_state agar hasil *training* tidak berubah ketika dijalankan lagi.

**5. Membuat model**

Membuat input layer dengan ukuran 1



Membuat embedding layer

Embedding size/dimension dapat diatur sesuai keinginan, namun embedding size yang terlalu kecil menyebabkan model hanya menghafal data sehingga rekomendasi yang dimunculkan hanya itu-itu saja bahkan untuk *user* yang berbeda, sedangkan embedding size terlalu besar rawan terjadi overfitting.



Membuat flatten layer

Flatten layer berfungsi untuk mengubah data array multidimensi menjadi vektor 1 dimensi agar bisa digunakan untuk input dalam fully connected layer



Membuat output layer

Output layer berfungsi untuk menghasilkan rekomendasi



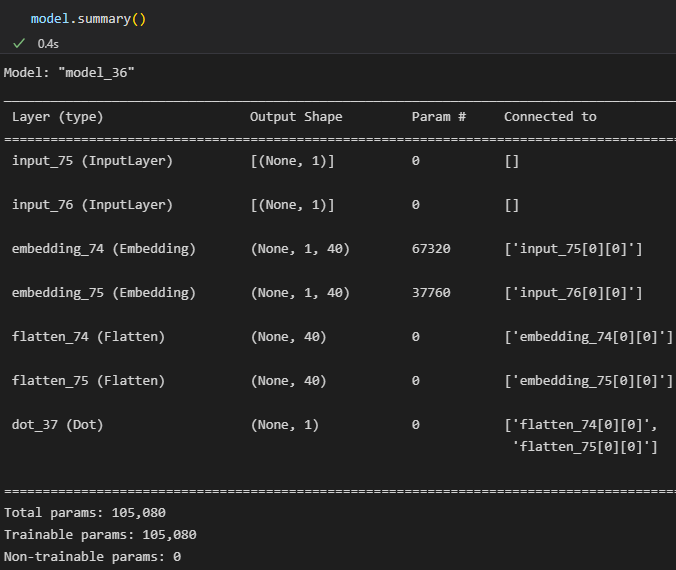
Menyusun layer-layer menjadi model

Masukkan input layer yang akan dimasukkan dan output layernya



Melihat rangkaian layer yang telah dibuat sebelumnya

Seluruh rangkaian layer sesuai dengan kode yang kita buat sebelumnya, mulai dari 2 input layer, 2 embedding layer dengan embedding size / dimension sebesar 40, flatten layer untuk mengubah data menjadi vektor 1 dimensi, dan diakhiri dengan output layer(Dot).



**6. Training model**

Konfigurasi hyperparameter dengan jenis optimizer ‘adam’ dan perhitungan loss dengan MSE.

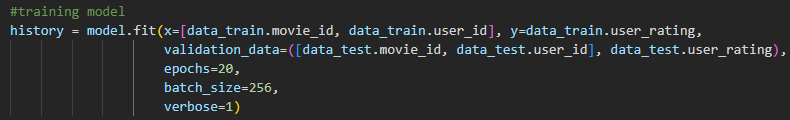


Artikel tentang optimizer ‘adam’: <https://medium.com/@saritilawah9/adam-optimizer-80cc267522af>

Artikel tentang MSE(Mean Square Error):

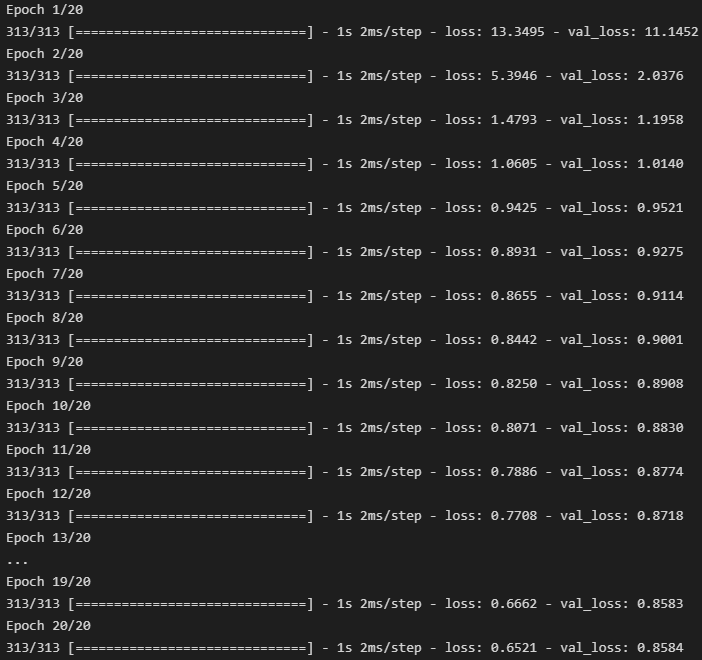
<https://dqlab.id/kriteria-jenis-teknik-analisis-data-dalam-forecasting>

Melakukan training dengan x sebagai features yang berisi data\_train, y sebagai target prediksi, validation\_data berisi data test yang sebelumnya sudah dipisah, pelatihan model diulang sebanyak 20 kali(epochs) dengan 256 data dalam sekali training, verbose=1 artinya memunculkan proses training dan menunjukkan loss dalam tiap epochs.

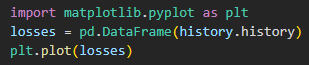


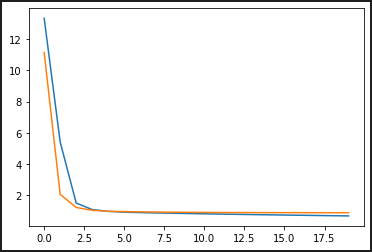
Hasil:

Dari hasil val\_loss di epochs terakhir, kebanyakan loss berada di angka 0,8.

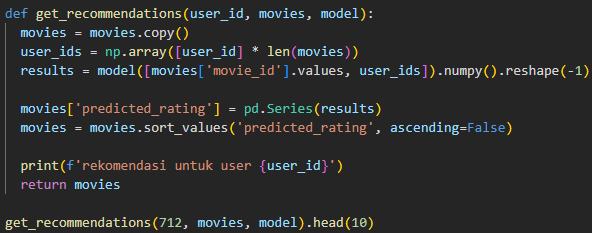


Menampilkan grafik training(loss dan validation loss)



Grafik di bawah menunjukkan bahwa loss pada data training semakin mengecil, sedangkan loss pada data validasi pada epochs ke-19, jika epochs pada training ditambah maka berpotensi untuk overfitting(nilai loss pada data validasi semakin naik).  


**7. Mencoba prediksi ratings dengan model yang telah dibuat**



Hasil:  

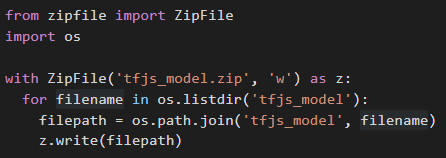

**8. Menyimpan Model**

Simpan dengan format .h5

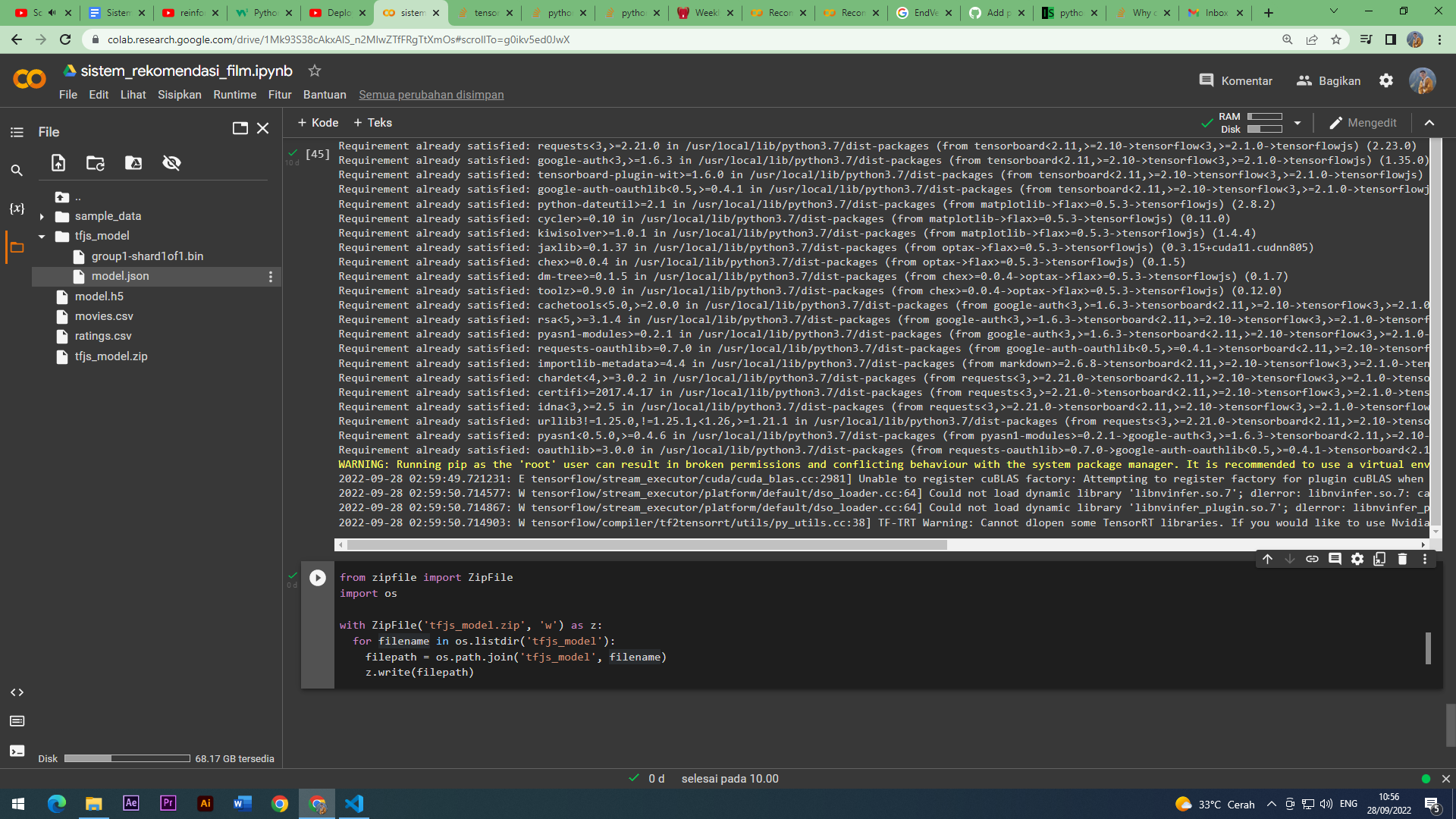


Convert model menjadi format .json, untuk mempermudah proses convert kita bisa upload file .ipynb dan dataset ke google colab, jalankan semua cell kemudian jalankan kode berikut





Berikut adalah hasilnya:



Kita bisa download file hasil convert ke memori local