Book Recommendation System

Dataset from: https://www.kaggle.com/datasets/zygmunt/goodbooks-10k

In [1]:

```
import pandas as pd
import numpy as np

from tensorflow.keras.layers import Input, Embedding, Flatten, Dot, Dense, Concatenate
from tensorflow.keras.models import Model
```

In [2]:

```
dataset = pd.read_csv('https://github.com/Alireza-Akhavan/datasets_and_models/raw/main/ratings.csv')
```

In [3]:

```
dataset.head()
```

Out[3]:

	book_id	user_id	rating
0	1	314	5
1	1	439	3
2	1	588	5
3	1	1169	4
4	1	1185	4

In [4]:

```
dataset.shape
```

Out[4]:

```
(981756, 3)
```

In [5]:

```
from sklearn.model_selection import train_test_split
train, test = train_test_split(dataset, test_size=0.2, random_state=42)
```

In [6]:

```
train.head()
```

Out[6]:

	book_id	user_id	rating
341848	3423	4608	2
964349	9811	36373	5
645459	6485	2957	4
74960	750	42400	3
358670	3591	36886	5

In [7]:

```
test.head()
```

Out[7]:

	book_id	user_id	rating
646451	6495	19643	5
614851	6175	8563	4
974393	9920	52110	3
21471	215	33864	5
272540	2728	16587	3

In [8]:

```
# number of unique users

n_users = len(dataset.user_id.unique())
n_users
```

Out[8]:

```
53424
```

In [9]:

```
# number of unique books

n_books = len(dataset.book_id.unique())
n_books
```

Out[9]:

```
10000
```

The model we will have will consist of the following main components:

- Input: Inputs for both books and users

- Embedding Layers: Embeddings for books and users
- Dot: Combines the embeddings using a dot product

In [10]:

```
book_input = Input(shape=[1], name="Book-Input")
book_embedding = Embedding(n_books+1, 5, name="Book-Embedding")(book_input)
book_vec = Flatten(name="Flatten-Books")(book_embedding)

user_input = Input(shape=[1], name="User-Input")
user_embedding = Embedding(n_users+1, 5, name="User-Embedding")(user_input)
user_vec = Flatten(name="Flatten-Users")(user_embedding)

prod = Dot(name="Dot-Product", axes=1)([book_vec, user_vec])
model = Model([user_input, book_input], prod)
model.compile('adam', 'mean_squared_error')
```

In [12]:

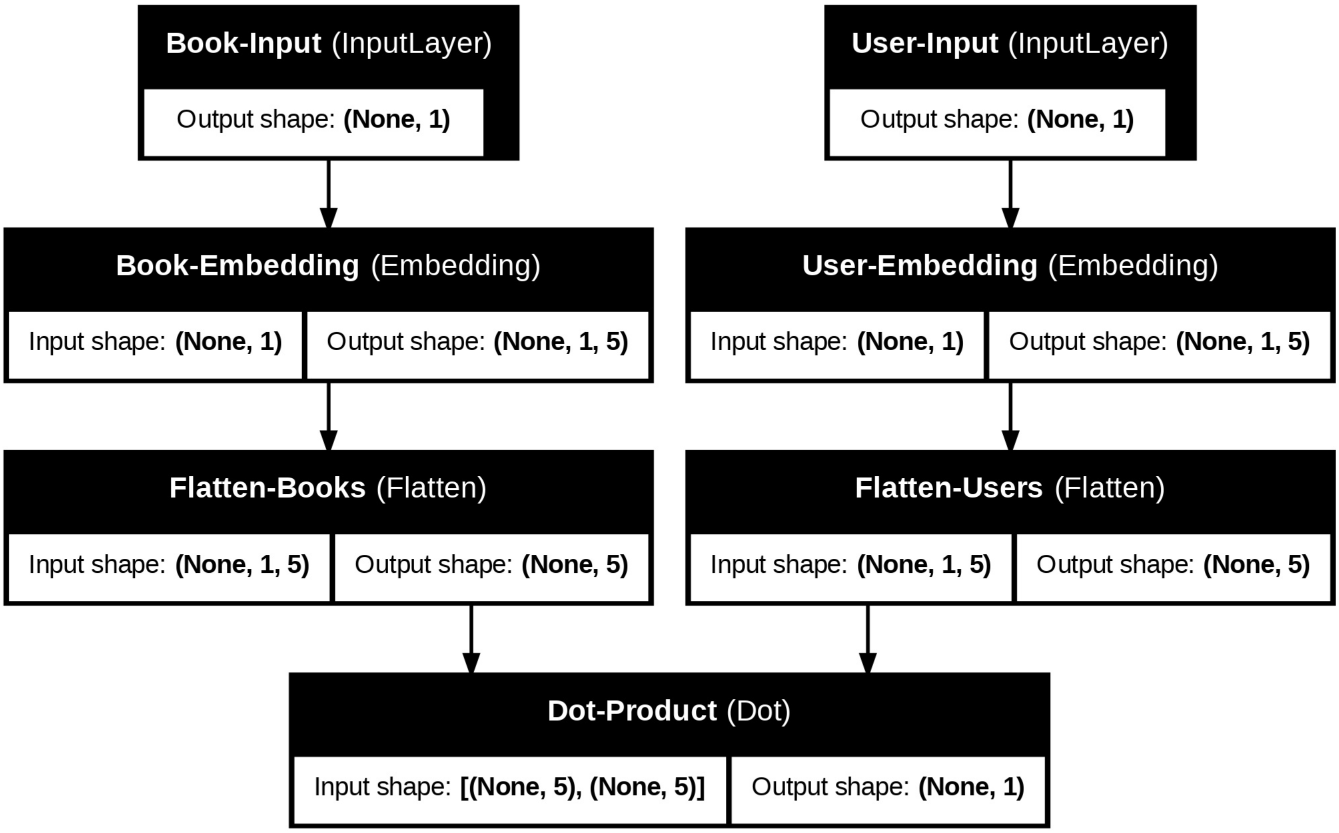
```
model.summary()
```

```
Model: "functional"

┏━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━┓
┃ Layer (type)        ┃ Output Shape      ┃    Param # ┃ Connected to      ┃
┡━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━┩
│ Book-Input          │ (None, 1)         │          0 │ -                 │
│ (InputLayer)        │                   │            │                   │
├─────────────────────┼───────────────────┼────────────┼───────────────────┤
│ User-Input          │ (None, 1)         │          0 │ -                 │
│ (InputLayer)        │                   │            │                   │
├─────────────────────┼───────────────────┼────────────┼───────────────────┤
│ Book-Embedding      │ (None, 1, 5)      │     50,005 │ Book-Input[0][0]  │
│ (Embedding)         │                   │            │                   │
├─────────────────────┼───────────────────┼────────────┼───────────────────┤
│ User-Embedding      │ (None, 1, 5)      │    267,125 │ User-Input[0][0]  │
│ (Embedding)         │                   │            │                   │
├─────────────────────┼───────────────────┼────────────┼───────────────────┤
│ Flatten-Books       │ (None, 5)         │          0 │ Book-Embedding[0… │
│ (Flatten)           │                   │            │                   │
├─────────────────────┼───────────────────┼────────────┼───────────────────┤
│ Flatten-Users       │ (None, 5)         │          0 │ User-Embedding[0… │
│ (Flatten)           │                   │            │                   │
├─────────────────────┼───────────────────┼────────────┼───────────────────┤
│ Dot-Product (Dot)   │ (None, 1)         │          0 │ Flatten-Books[0]… │
│                     │                   │            │ Flatten-Users[0]… │
└─────────────────────┴───────────────────┴────────────┴───────────────────┘

 Total params: 317,130 (1.21 MB)
 Trainable params: 317,130 (1.21 MB)
 Non-trainable params: 0 (0.00 B)
```

In [11]:

```
from tensorflow import keras
keras.utils.plot_model(model, show_shapes=True, show_layer_names=True)
```

Out[11]:



In [13]:

```
model.evaluate([test.user_id, test.book_id], test.rating)
```

```
6136/6136 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - loss: 15.8460
```

Out[13]:

```
15.828319549560547
```

In [14]:

```
predictions = model.predict([test.user_id.head(10), test.book_id.head(10)])

for i in range(0,10):
   print(predictions[i], test.rating.iloc[i])
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 187ms/step
[-0.00025442] 5
[-0.0012523] 4
[0.0028373] 3
[-0.00327236] 5
[0.00350545] 3
[-0.00407772] 3
[-0.00248902] 3
[-0.00065142] 4
[0.0013981] 3
[-0.00204168] 5
```

Training

In [19]:

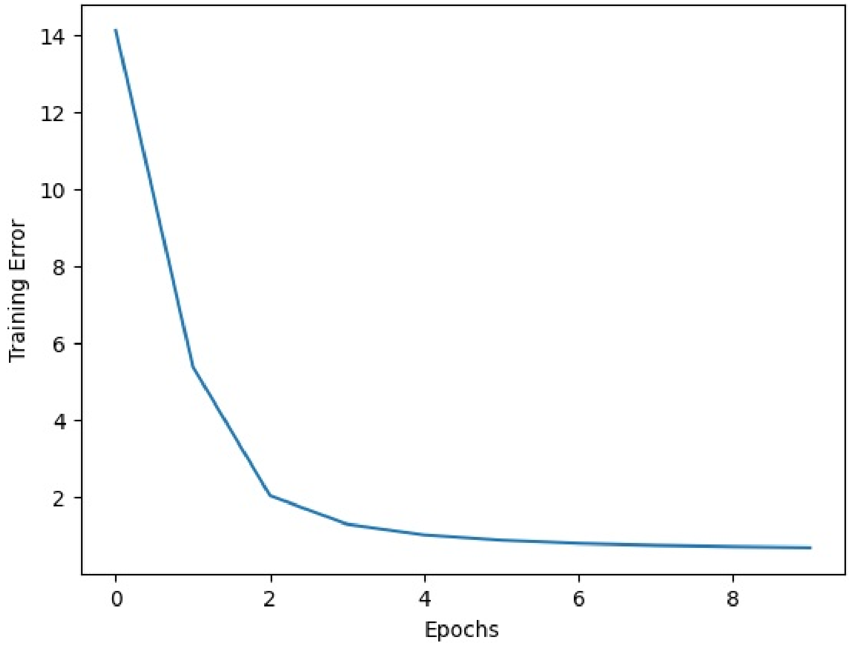
```
from keras.models import load_model
import os

if os.path.exists('regression_model.keras'):
    model = load_model('regression_model.keras')
else:
    history = model.fit([train.user_id, train.book_id], train.rating, epochs=10,batch_size=64, verbose=1)
    model.save('regression_model.keras')
```

In [18]:

```
import matplotlib.pyplot as plt

plt.plot(history.history['loss'])
plt.xlabel("Epochs")
plt.ylabel("Training Error")
```

Out[18]:

```
Text(0, 0.5, 'Training Error')
```



```
In [16]: model.evaluate([test.user_id, test.book_id], test.rating)

         6136/6136 ━━━━━━━━━━━━━━━━━━━━ 9s 2ms/step - loss: 0.9399

Out[16]: 0.9380558133125305

In [17]: predictions = model.predict([test.user_id.head(10), test.book_id.head(10)])

         [print(predictions[i], test.rating.iloc[i]) for i in range(0,10)]

         1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
         [5.1756186] 5
         [4.099106] 4
         [4.0260844] 3
         [4.6336107] 5
         [3.5826578] 3
         [3.971473] 3
         [3.7924268] 3
         [4.6222005] 4
         [4.1563315] 3
         [4.090555] 5

Out[17]: [None, None, None, None, None, None, None, None, None, None]
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js