In [18]:
```
import pandas as pd
import numpy as np
```

In [19]:
```
df = pd.read_csv('./car-details-v3.csv')
df.head()
```

Out[19]:

	name	year	selling_price	km_driven	fuel	seller_type	transmission	ow
0	Maruti Swift Dzire VDI	2014	450000	145500	Diesel	Individual	Manual	F Ow
1	Skoda Rapid 1.5 TDI Ambition	2014	370000	120000	Diesel	Individual	Manual	Seco Ow
2	Honda City 2017-2020 EXi	2006	158000	140000	Petrol	Individual	Manual	Th Ow
3	Hyundai i20 Sportz Diesel	2010	225000	127000	Diesel	Individual	Manual	F Ow
4	Maruti Swift VXI BSIII	2007	130000	120000	Petrol	Individual	Manual	F Ow

In [36]:
```
# clear data (biranje kljucnih kolona i ciscenje podataka)

df = df[['name', 'year', 'selling_price', 'km_driven', 'fuel', 'seller_ty

# The df.dropna(inplace=True) command in Pandas removes rows or columns c
# missing values (NaN) directly from the DataFrame df without creating a
df.dropna(inplace=True)

# TODO: ovo je minimalno ciscenje, u realnom sistemu mora postojati vise
```

In [38]:
```
# Pretvaranje kategorija u brojeve

# Svaka kategorija postaje integer (npr. Diesel=0, Petrol=1 …).
# Ovaj pristup je jednostavan, ali ne modeluje semantičke odnose (embeddi

df['fuel'] = df['fuel'].astype('category').cat.codes
df['seller_type'] = df['seller_type'].astype('category').cat.codes
df['transmission'] = df['transmission'].astype('category').cat.codes
```

In [39]:
```
# Normalizacija numerickih atributa

# Skaliranje u opseg [0,1] omogućava da neuronska mreža lakše konvergira.
# Kategorije ostaju neskalirane – što je u redu na ovom nivou.

from sklearn.preprocessing import MinMaxScaler
```

```
scaler = MinMaxScaler()
df[['year','selling_price','km_driven']] = scaler.fit_transform(df[['year
```

In [40]:
```
# kreiranje matrice item feature-a
item_features = df[['year','selling_price','km_driven','fuel','seller_typ
item_features.shape

# To je ulaz za item tower.

# Dimenzija:
# (broj_automobila, 6)
```

Out[40]: (8128, 6)

In [24]:
```
# simulacija korisnickog profila
user_profile = {
    "year": 0.8,                # preferira novija kola
    "selling_price": 0.4,       # srednji budzet
    "km_driven": 0.2,           # mala kilometraza
    "fuel": 0,                  # 0 = Benzin (u kodiranju)
    "seller_type": 0,           # svejedno
    "transmission": 1           # 1 = Automatika
}
```

In [41]:
```
# Pretvaranje u tensor:
user_vec = np.array(list(user_profile.values())).reshape(1, -1)
```

In [26]:
```
# Two-Tower Model (minimalni TensorFlow MVP)

# user network
import tensorflow as tf
from tensorflow.keras import layers, Model

embedding_dim = 16
```

In [42]:
```
# USER tower

# input: 6 numeričkih feature-a
# Jedan skriveni sloj (32 neurona)
# Embedding sloj od 16 dimenzija

user_input = layers.Input(shape=(6,))
u = layers.Dense(32, activation='relu')(user_input)
u = layers.Dense(embedding_dim)(u)
user_tower = Model(user_input, u)
```

In [43]:
```
# item network (tower)

# identicna struktura i za item input
item_input = layers.Input(shape=(6,))
i = layers.Dense(32, activation='relu')(item_input)
i = layers.Dense(embedding_dim)(i)
item_tower = Model(item_input, i)
```

In [29]:
```
# Loss funkcija (dot product)
user_emb = user_tower(user_input)
item_emb = item_tower(item_input)
```

```
# Dot-product model za merenje sličnosti
dot = layers.Dot(axes=1)([user_emb, item_emb])

# Model pokušava da nauči da:
# slični parovi (user, item) → dot product približava 1
# neslični parovi → dot product približava 0

model = Model(inputs=[user_input, item_input], outputs=dot)
model.compile(optimizer='adam', loss='mse')
```

In [46]:

```
# Formiranje pozitivnih i negativnih parova

# pozitivni parovi = automobili cije karakteristike lice na korisnicke pr
# negativni parovi = potpuno suprotne karakteristike

# Pozitivni uzorci = automobili sa automatikom i benzinom
positive_items = df[(df['fuel'] == user_profile['fuel']) &
                    (df['transmission'] == user_profile['transmission'])]

# Negativni uzorci
negative_items = df[(df['fuel'] != user_profile['fuel'])]

# Sampling (uzorkovanje) - ukupno 400 primera
pos_samples = positive_items.sample(200, replace=True)[['year','selling_p
neg_samples = negative_items.sample(200, replace=True)[['year','selling_p
```

In [31]:

```
# Formiranje ulaza:
X_user = np.vstack([
    np.repeat(user_vec, len(pos_samples), axis=0),
    np.repeat(user_vec, len(neg_samples), axis=0)
])

X_item = np.vstack([pos_samples, neg_samples])

y = np.hstack([np.ones(len(pos_samples)), np.zeros(len(neg_samples))])
```

In [33]:

```
## Trening modela
# Mala epoha, minimalno treniranje — dovoljno za MVP.
model.fit([X_user, X_item], y, epochs=5, batch_size=32)

## Generisanje embeddinga za sve automobile
item_emb_matrix = item_tower.predict(item_features)
user_embedding = user_tower.predict(user_vec)

# Sada imaš embedding vektore:
# User embedding: (1, 16)
# Item embedding: (N_items, 16)

## Top N poruka
from sklearn.metrics.pairwise import cosine_similarity

# Preporuke — kosinusna slicnost
scores = cosine_similarity(user_embedding, item_emb_matrix)[0]

# Uzima 10 automobila sa najvecim kosinusnim skorom
top_n_idx = np.argsort(scores)[::-1][:10]
```

```
Epoch 1/5
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step - loss: 0.0943
Epoch 2/5
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step - loss: 0.0909
Epoch 3/5
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step - loss: 0.0863
Epoch 4/5
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step - loss: 0.0836
Epoch 5/5
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step - loss: 0.0839
254/254 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
```

In [48]:

```
recommended_cars = df.iloc[top_n_idx]
recommended_cars[['name','year','selling_price','km_driven']]
```

Out[48]:

	name	year	selling_price	km_driven
3306	Maruti Eeco CNG 5 Seater AC BSIV	1.000000	0.037011	0.002118
5815	Maruti Alto 800 LXI CNG	1.000000	0.034102	0.006778
7543	Maruti Alto 800 CNG LXI Optional	0.972973	0.030090	0.004236
35	Maruti Alto 800 CNG LXI Optional	0.972973	0.030090	0.004236
402	Maruti Eeco CNG 5 Seater AC	1.000000	0.038114	0.014827
1225	Maruti Eeco CNG 5 Seater AC BSIV	0.972973	0.042126	0.004236
5789	Maruti Alto K10 LXI CNG	0.972973	0.040120	0.008473
6488	Maruti Wagon R LXI CNG Optional	0.972973	0.043129	0.014827
2513	Maruti Wagon R LXI CNG	0.945946	0.036108	0.016945
5557	Maruti Eeco CNG 5 Seater AC BSIV	0.918919	0.034102	0.010591

In [ ]: