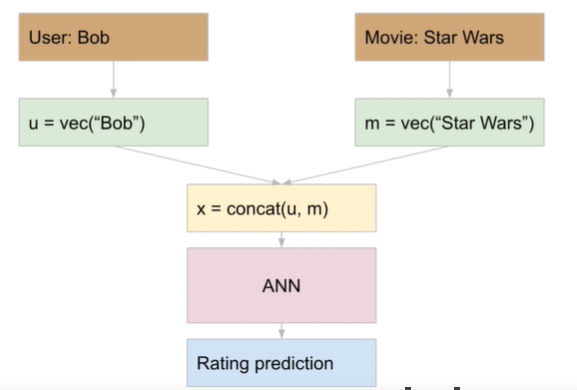
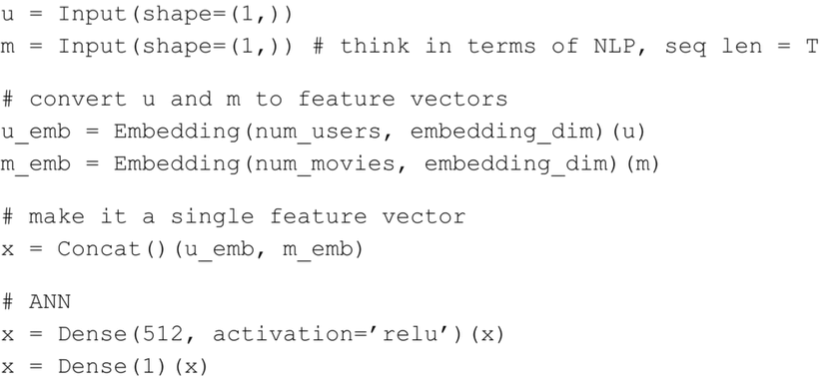
1. Recommender systems

* E.g.: Movie ratings
* Ratings dataset must be incomplete (if every user has watched and rated all movies -> nothing left to recommend)
* How to recommend?
* Bob rated every star wars movie a 5
* Bob rated a few Star Trek movies a 5
* Bob rated Avatar a 5
* Probably a fan of sci-fi
* Probably likes movies with space travel, aliens
* Given a dataset of triples (user, item, rating)
* Fit a model to the data f(u, m) -> r
* What should it do?
  + If the user u and movie m appeared in the dataset, then the predicted rating should be close to the true rating
  + The function should predict what user u would rate movie m, even if it didn’t appear in the training set
* Neural networks (function approximators) do this!
* Given a user, get predicted for every unseen movie
* Sort by predicted rating (descending)
* Recommend movies with the highest predicted rating
* How to build the model
* Problem: both users and movies are categorical
* Neural networks do matrix multiplications
* Can’t multiply a category by a number
* Embedding! Map each category to a vector
* A neural network for recommenders



* Pseudocode:



* Functional API required
* 2 inputs appear in parallel
* Sequential layer won’t support this
* Matrix factorization – well known algorithm in recommenders
* When word embeddings became popular, the main algorithms were word2vec and GloVe
* They both do things like ‘king-man = queen-woman’
* GloVe is nothing more than matrix factorization