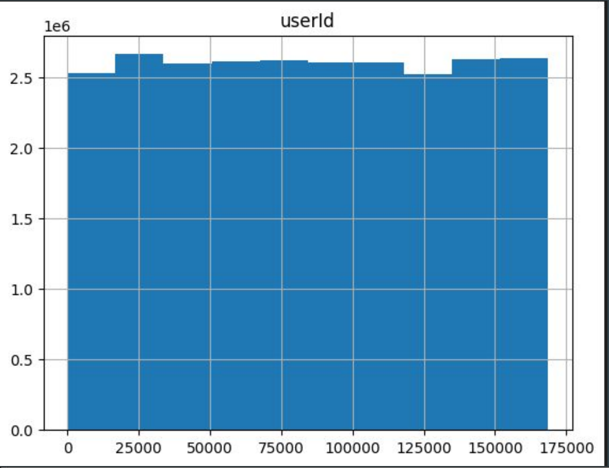
Recommender System

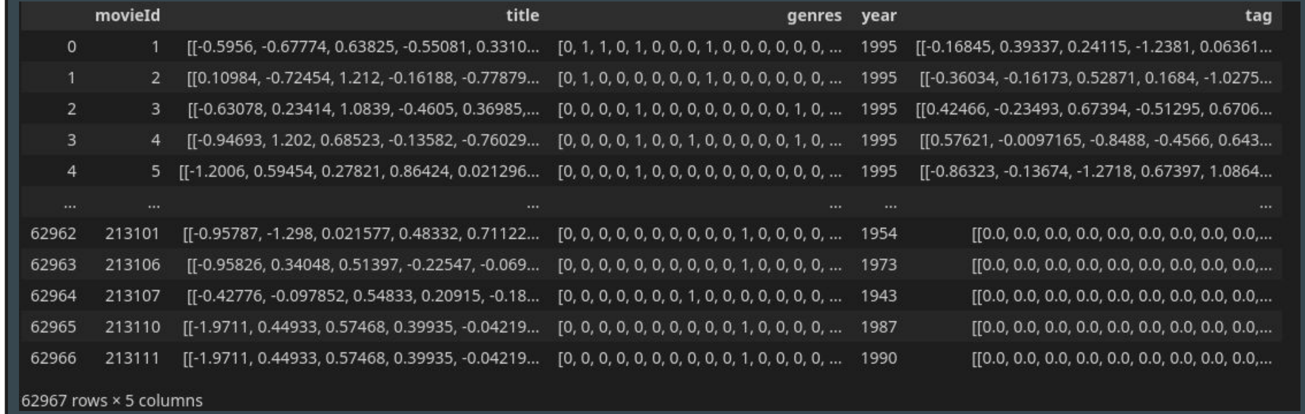
By Michael Nafe & Nicholas Bacon

Data Augmentation: Go Big (= RAM overflow, long runtime, regret)

- Goal: Combine, clean, organize data into entirely numerical matrices
    - Combined two sets of movies data from MovieLens
        - “MovieLens 1M movie ratings. Stable benchmark dataset. 1 million ratings from 6000 users on 4000 movies”
        - “MovieLens 25M movie ratings. Stable benchmark dataset. 25 million ratings and one million tag applications applied to 62,000 movies by 162,000 users.”
    - The rating datasets combined to make a 168,581 x 62,967 (num_users x num_movies) matrix
        - Had to parse the data using a dev defined object that accumulated then saved to disk 3 lists:
            - the row coordinate of, the col coordinate of, the data that is not a 0
    - Combined the movie datasets into a 62,967 x 319 matrix by
        - peeling the year off the end of the movie titles and adding that as a numeric feature column
        - encoding the movie genres using a one-hot encoding
        - using Glove to embed the title of each movie and, for those that had them, a tag into a 6 x 25 = 150 length vector each and added those as feature columns.
    - Combined for a 168,581 x 4 users matrix with the feature columns being: gender, age, occupation, zip code
        - encoded Male/Female -> -1/1, truncated the zips at 5 digits, removing any “-” (wanted everything numeric).

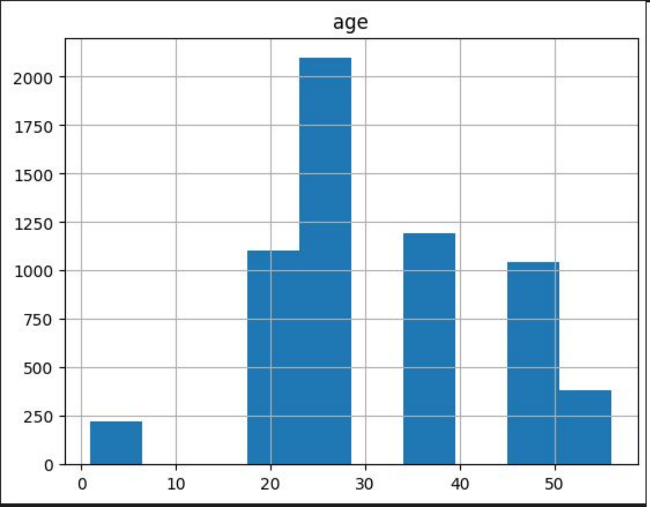
Inspecting augmented data

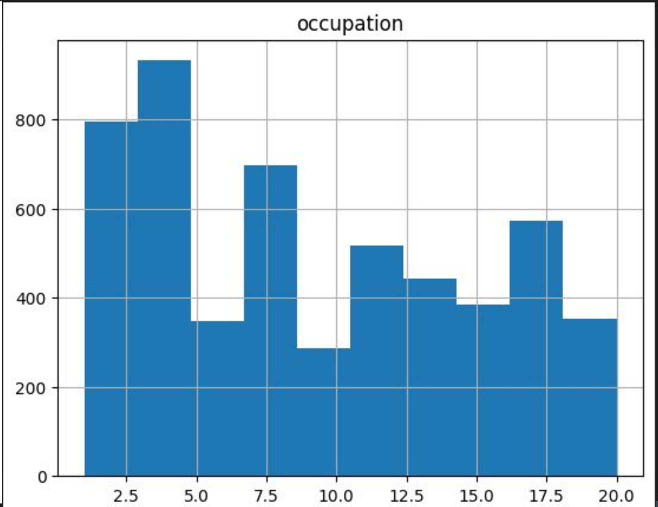


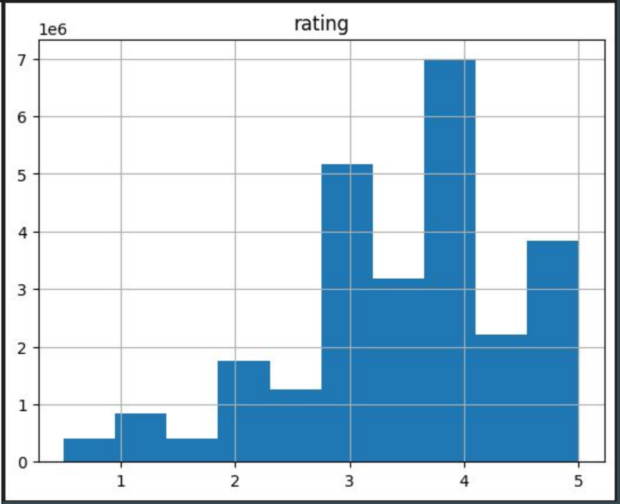


	movieId	title	genres	year	tag
0	1	[[-0.5956, -0.67774, 0.63825, -0.55081, 0.3310...	[0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, ...	1995	[[-0.16845, 0.39337, 0.24115, -1.2381, 0.06361...
1	2	[[0.10984, -0.72454, 1.212, -0.16188, -0.77879...	[0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, ...	1995	[[-0.36034, -0.16173, 0.52871, 0.1684, -1.0275...
2	3	[[-0.63078, 0.23414, 1.0839, -0.4605, 0.36985,...	[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, ...	1995	[[0.42466, -0.23493, 0.67394, -0.51295, 0.6706...
3	4	[[-0.94693, 1.202, 0.68523, -0.13582, -0.76029...	[0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, ...	1995	[[0.57621, -0.0097165, -0.8488, -0.4566, 0.643...
4	5	[[-1.2006, 0.59454, 0.27821, 0.86424, 0.021296...	[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...	1995	[[-0.86323, -0.13674, -1.2718, 0.67397, 1.0864...
...	...	...	...	...	...
62962	213101	[[-0.95787, -1.298, 0.021577, 0.48332, 0.71122...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, ...	1954	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...
62963	213106	[[-0.95826, 0.34048, 0.51397, -0.22547, -0.069...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, ...	1973	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...
62964	213107	[[-0.42776, -0.097852, 0.54833, 0.20915, -0.18...	[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, ...	1943	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...
62965	213110	[[-1.9711, 0.44933, 0.57468, 0.39935, -0.04219...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, ...	1987	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...
62966	213111	[[-1.9711, 0.44933, 0.57468, 0.39935, -0.04219...	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, ...	1990	[[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,...

62967 rows × 5 columns



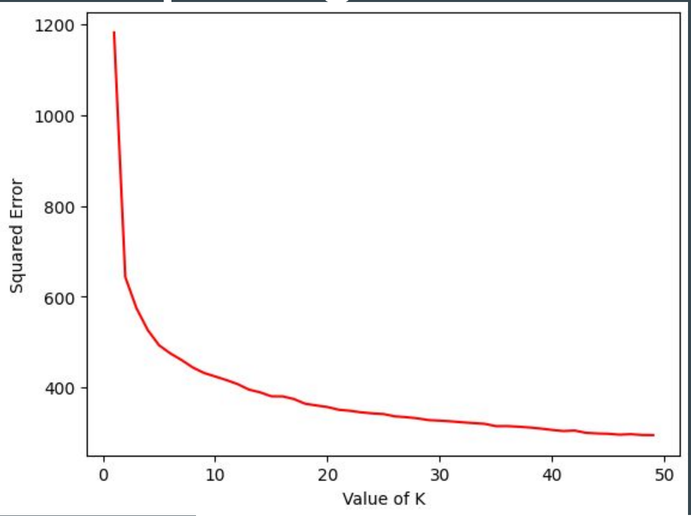




Recommender Construction: M=Movie_matrix, U=User_matrix, R=Rating_matrix

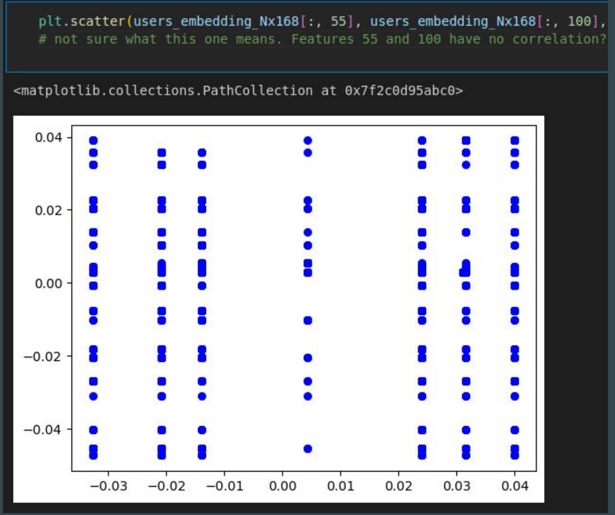
- Goal: Would having an "***a***" such that M*U = a*R be useful? Plus, Similarity Matrix and Clustering
  - U= 168,581 x 4, Movie_matrix = 62,967 x 319 -> embed(U)*transpose(M) = (168,581 x K) * (K x 62,967)
    - So as to lighten the load, we removed the title embedding feature columns from the movies matrix -> (62,967 x 168), this gave us K=168
    - Embedded each user's [gender, age, occupation, zip code] -> a vector of length 168 using a keras embedding layer.
      - Layer does not accept negatives, so reencode Male/Female from -1/1 -> 21/22
      - Used a vocab size of "number of unique values in the users matrix" ~ 3,400 (mostly zip codes).
        - This binned all the unique values in the users matrix into numbers 0 - about 3,400
    - Finding "***a***" = overflow RAM, headaches, moved on to other goals
  - Make an upper triangular matrix of the cosine similarity for each movie, w.r.t. each movie.
    - 0 is counted as negative in cosine sim, so centered the entire movies matrix around 0: ~ (-7.0, 7.0)
    - 62,967 + 62,966 + ... + 2 + 1 = 1,982,453,028 floats. Causing RAM overflow too.
      - Solution: partition the work, compute cosine sim of ~3,500 movies to all movies, write to disk, dump the saved state, go to next partition, repeat. Took about 2 - 3 hours to complete.
  - Cluster the users.
    - Users now embedded as vectors of length 168.
    - Find a good k for k means, cluster using that k, save the kmeans model

Inspecting Recommender Construction



Best k = 5, 6

Plotting the user embedding features against each other

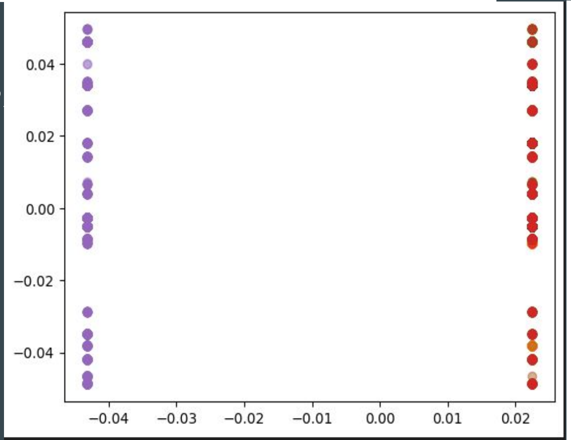
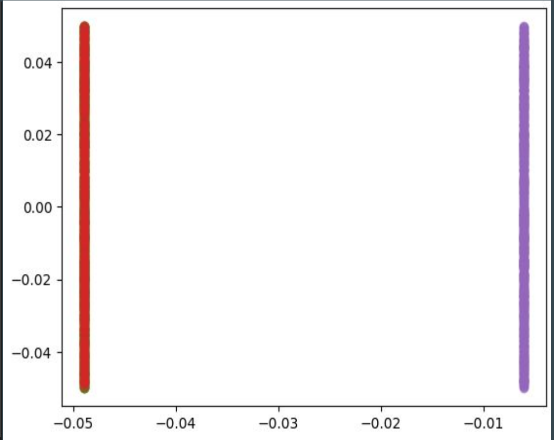
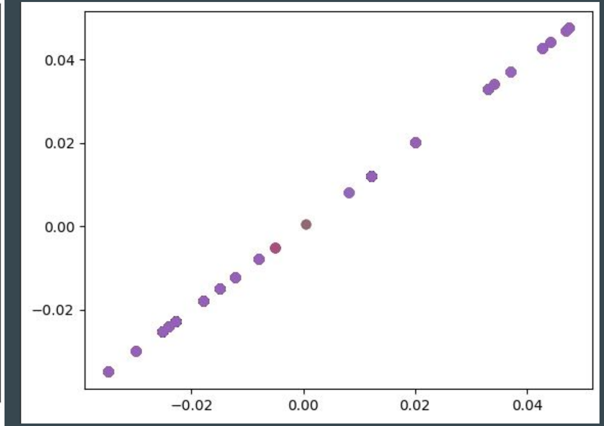


```
plt.scatter(users_embedding_Nx168[:, 55], users_embedding_Nx168[:, 100],
# not sure what this one means. Features 55 and 100 have no correlation?
```

```
<matplotlib.collections.PathCollection at 0x7f2c0d95abc0>
```

Below, the same feature, column 112 (part of the occupation embedding), on x and y

With K = 5, choosing various features as the x, y dims ->

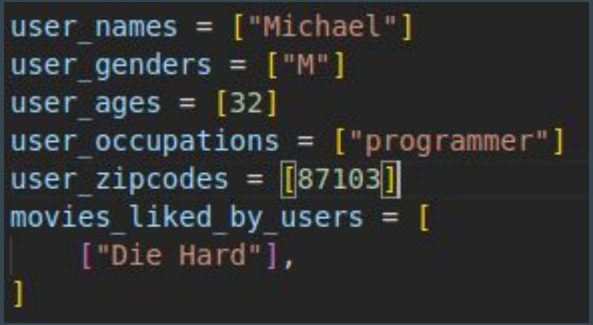




Recommender Recommendations

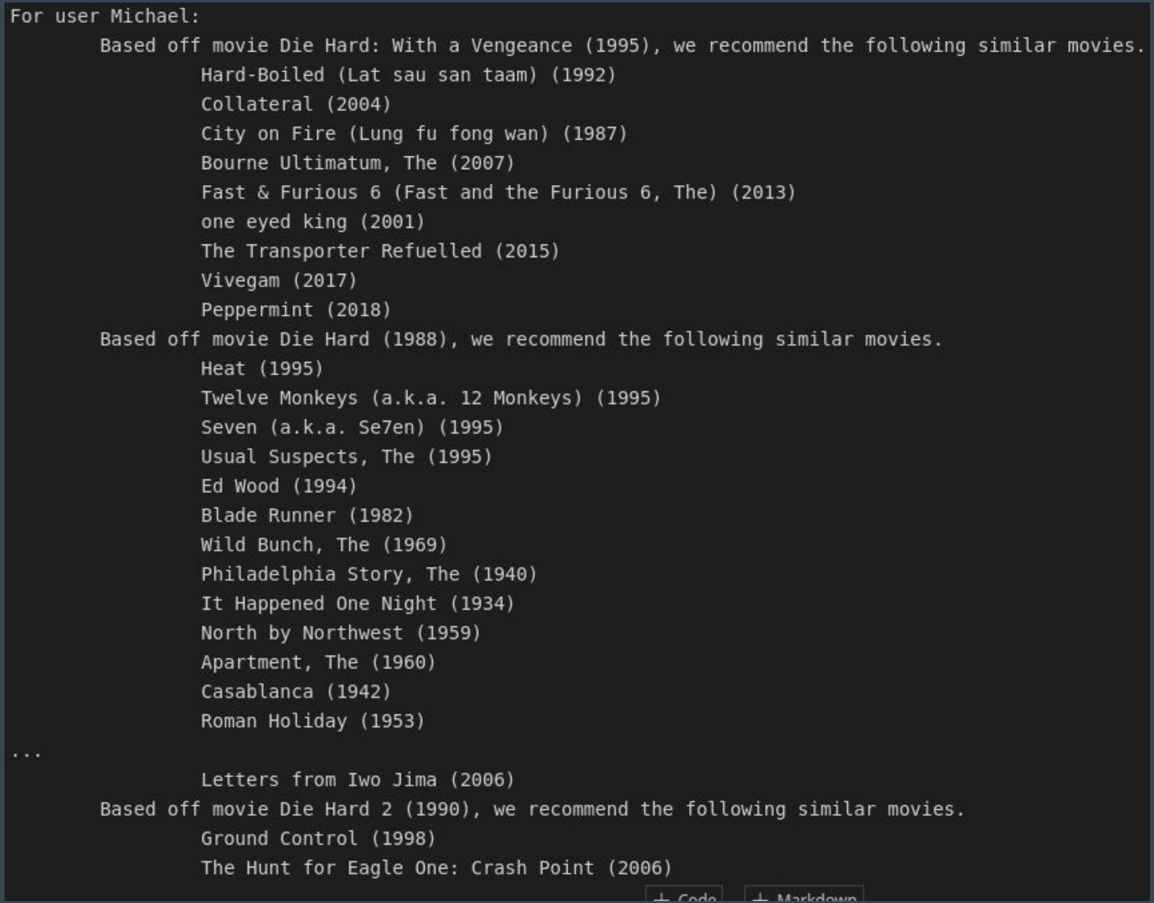
- Goal: Given a user's characteristics and a/some movie(s) they like, recommend movies based on them being
  - cosine similar to the movie the user says they like
    - the cosine sim matrix saved as 18 ~3500x62,967 matrices.
    - find the right matrix in memory, load it, find the right row, return the movies who are >= .95 in cos similarity, dump the partition state.
  - movies most liked by other users who like the movie(s) the user likes
    - Find user rows in the sparse ratings matrix that also rated the same movie high, get those movies that these users rated above a certain threshold, then across those ratings from theses users, sum up the ratings (handles two users rating the same movie high = a better recommendation?), then return the 15ish movies with the highest summed ratings
  - movies most liked by other users similar to the user in embedded characteristics: gender, age, occ, zip
    - Embed the user's characteristics the same way the users were embedded during recommender construction.
      - Had to use the same set of unique vocab here, so if the given user's zip code did not exist in the unique vocab, then try finding the closest match by stepping the zip code value in a direction and checking if this zip code exists in our unique vocab.
        - Example: 87109 does not exist, but 87103 does
    - Use the kmeans model to predict what cluster the given user fits in, then get the ratings of the other users who fall into that cluster from the sparse ratings matrix and find the movies most liked by these users in the ratings matrix.

Inspecting Recommendations

Cosine similar to the movie the user says they like



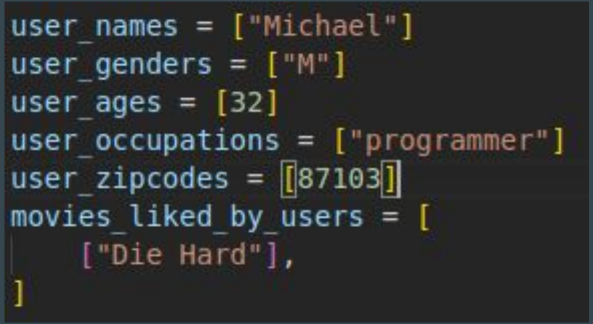
```
user_names = ["Michael"]
user_genders = ["M"]
user_ages = [32]
user_occupations = ["programmer"]
user_zipcodes = [87103]
movies_liked_by_users = [
    ["Die Hard"],
]
```



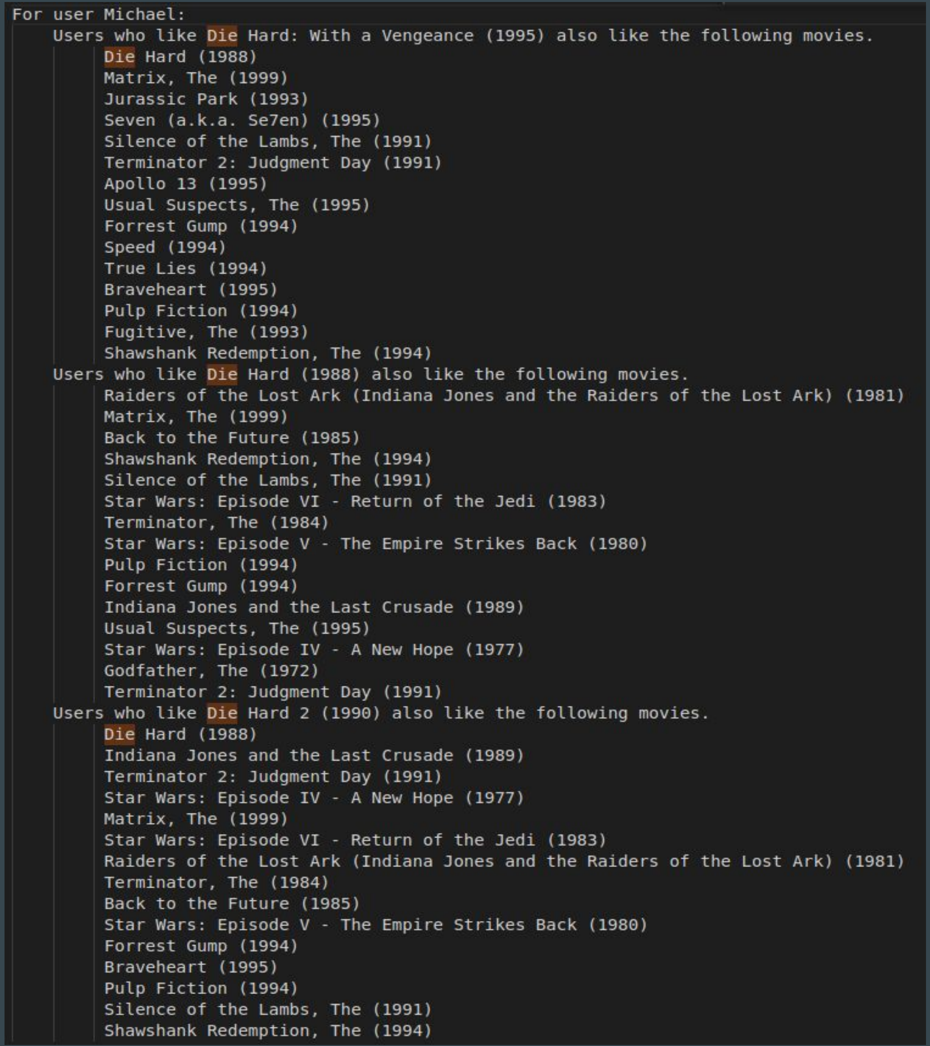
```
For user Michael:
        Based off movie Die Hard: With a Vengeance (1995), we recommend the following similar movies.
                Hard-Boiled (Lat sau san taam) (1992)
                Collateral (2004)
                City on Fire (Lung fu fong wan) (1987)
                Bourne Ultimatum, The (2007)
                Fast & Furious 6 (Fast and the Furious 6, The) (2013)
                one eyed king (2001)
                The Transporter Refuelled (2015)
                Vivegam (2017)
                Peppermint (2018)
        Based off movie Die Hard (1988), we recommend the following similar movies.
                Heat (1995)
                Twelve Monkeys (a.k.a. 12 Monkeys) (1995)
                Seven (a.k.a. Se7en) (1995)
                Usual Suspects, The (1995)
                Ed Wood (1994)
                Blade Runner (1982)
                Wild Bunch, The (1969)
                Philadelphia Story, The (1940)
                It Happened One Night (1934)
                North by Northwest (1959)
                Apartment, The (1960)
                Casablanca (1942)
                Roman Holiday (1953)
...

                Letters from Iwo Jima (2006)
        Based off movie Die Hard 2 (1990), we recommend the following similar movies.
                Ground Control (1998)
                The Hunt for Eagle One: Crash Point (2006)
```

Inspecting Recommendations

movies most liked by other users who like the movie(s) the user likes



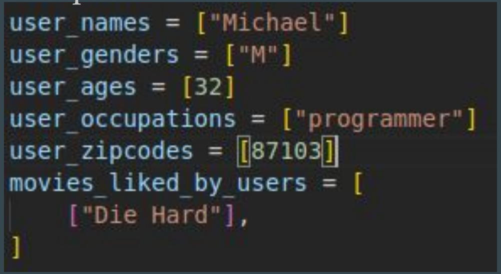
```python
user_names = ["Michael"]
user_genders = ["M"]
user_ages = [32]
user_occupations = ["programmer"]
user_zipcodes = [87103]
movies_liked_by_users = [
    ["Die Hard"],
]
```



```
For user Michael:
    Users who like Die Hard: With a Vengeance (1995) also like the following movies.
        Die Hard (1988)
        Matrix, The (1999)
        Jurassic Park (1993)
        Seven (a.k.a. Se7en) (1995)
        Silence of the Lambs, The (1991)
        Terminator 2: Judgment Day (1991)
        Apollo 13 (1995)
        Usual Suspects, The (1995)
        Forrest Gump (1994)
        Speed (1994)
        True Lies (1994)
        Braveheart (1995)
        Pulp Fiction (1994)
        Fugitive, The (1993)
        Shawshank Redemption, The (1994)
    Users who like Die Hard (1988) also like the following movies.
        Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)
        Matrix, The (1999)
        Back to the Future (1985)
        Shawshank Redemption, The (1994)
        Silence of the Lambs, The (1991)
        Star Wars: Episode VI - Return of the Jedi (1983)
        Terminator, The (1984)
        Star Wars: Episode V - The Empire Strikes Back (1980)
        Pulp Fiction (1994)
        Forrest Gump (1994)
        Indiana Jones and the Last Crusade (1989)
        Usual Suspects, The (1995)
        Star Wars: Episode IV - A New Hope (1977)
        Godfather, The (1972)
        Terminator 2: Judgment Day (1991)
    Users who like Die Hard 2 (1990) also like the following movies.
        Die Hard (1988)
        Indiana Jones and the Last Crusade (1989)
        Terminator 2: Judgment Day (1991)
        Star Wars: Episode IV - A New Hope (1977)
        Matrix, The (1999)
        Star Wars: Episode VI - Return of the Jedi (1983)
        Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)
        Terminator, The (1984)
        Back to the Future (1985)
        Star Wars: Episode V - The Empire Strikes Back (1980)
        Forrest Gump (1994)
        Braveheart (1995)
        Pulp Fiction (1994)
        Silence of the Lambs, The (1991)
        Shawshank Redemption, The (1994)
```

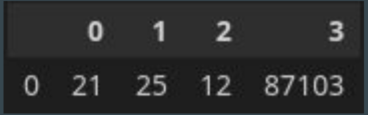
Inspecting Recommendations

movies most liked by other users similar to the user in embedded characteristics: gender, age, occ, zip



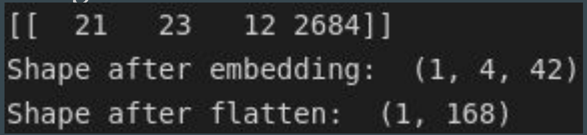
```
user_names = ["Michael"]
user_genders = ["M"]
user_ages = [32]
user_occupations = ["programmer"]
user_zipcodes = [87103]
movies_liked_by_users = [
    ["Die Hard"],
]
```

Encoding the user



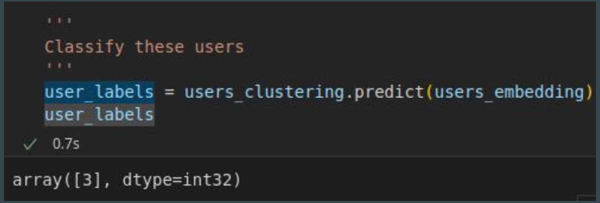
	0	1	2	3
0	21	25	12	87103

Embedding the user

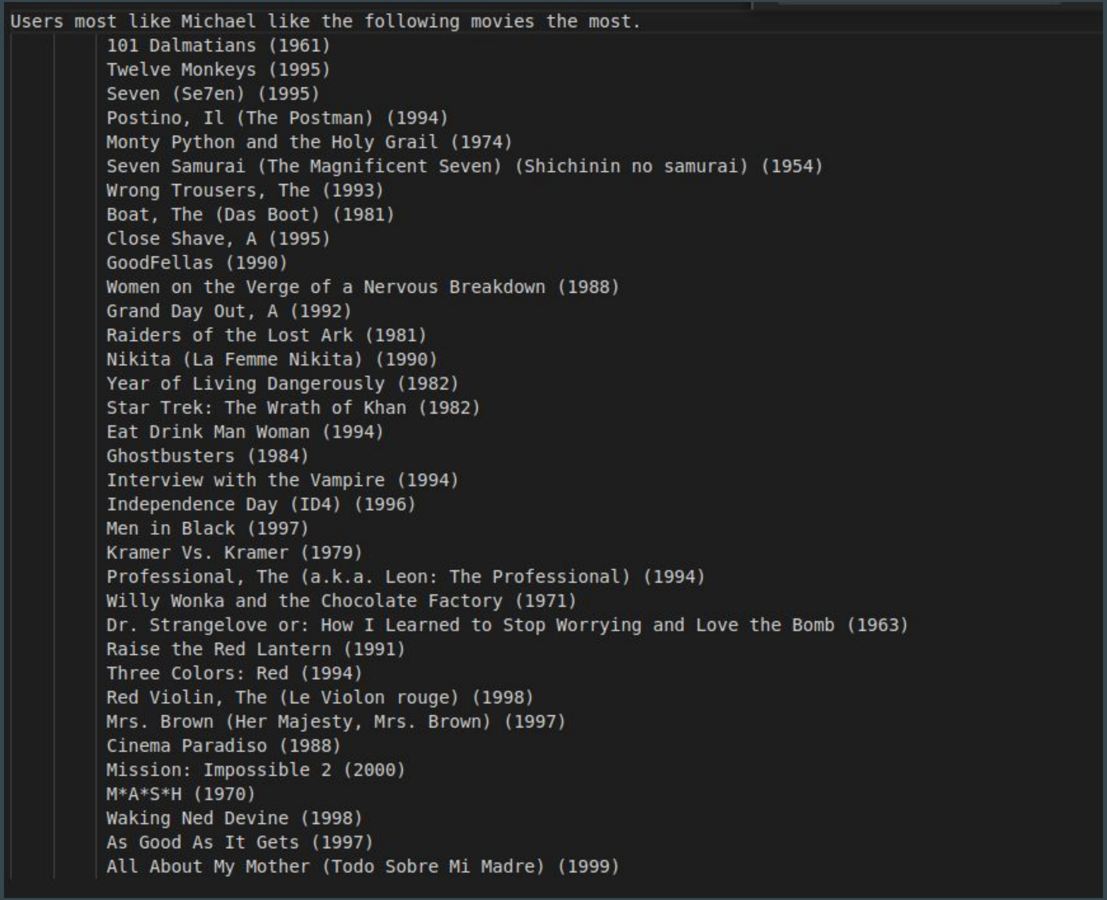


```
[[  21   23   12 2684]]
Shape after embedding:  (1, 4, 42)
Shape after flatten:  (1, 168)
```

Predict the user's cluster:



```
'''
Classify these users
'''
user_labels = users_clustering.predict(users_embedding)
user_labels
✓ 0.7s

array([3], dtype=int32)
```



```
Users most like Michael like the following movies the most.
            101 Dalmatians (1961)
            Twelve Monkeys (1995)
            Seven (Se7en) (1995)
            Postino, Il (The Postman) (1994)
            Monty Python and the Holy Grail (1974)
            Seven Samurai (The Magnificent Seven) (Shichinin no samurai) (1954)
            Wrong Trousers, The (1993)
            Boat, The (Das Boot) (1981)
            Close Shave, A (1995)
            GoodFellas (1990)
            Women on the Verge of a Nervous Breakdown (1988)
            Grand Day Out, A (1992)
            Raiders of the Lost Ark (1981)
            Nikita (La Femme Nikita) (1990)
            Year of Living Dangerously (1982)
            Star Trek: The Wrath of Khan (1982)
            Eat Drink Man Woman (1994)
            Ghostbusters (1984)
            Interview with the Vampire (1994)
            Independence Day (ID4) (1996)
            Men in Black (1997)
            Kramer Vs. Kramer (1979)
            Professional, The (a.k.a. Leon: The Professional) (1994)
            Willy Wonka and the Chocolate Factory (1971)
            Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1963)
            Raise the Red Lantern (1991)
            Three Colors: Red (1994)
            Red Violin, The (Le Violon rouge) (1998)
            Mrs. Brown (Her Majesty, Mrs. Brown) (1997)
            Cinema Paradiso (1988)
            Mission: Impossible 2 (2000)
            M*A*S*H (1970)
            Waking Ned Devine (1998)
            As Good As It Gets (1997)
            All About My Mother (Todo Sobre Mi Madre) (1999)
```

Recommender Conclusion

- Recommendations based off other movies
  - cosine similar to the movie the user says they like
    - the recommended movies are mostly similar in genre, tag description tone.
    - notice this way did not result in any variation of Die Hard recommending another variation of Die Hard, but did recommend other Bruce Willis movies, Twelve Monkeys, for example.
    - recommendations here appear to do a better job recommending along the tail.
- Recommendations based off other users:
  - movies most liked by other users similar to the user in embedded characteristics: gender, age, occ, zip
    - this one only really used 4 features, but appears to have done a decent job.
      - Most of the movies listed there that I recognize, I do like, but there are some surprises too
    - If we had more and more descriptive for the task features about the user, this could probably do a better job.
      - Having only k = 5 clusters of people across 168,581 users seems hardly discriminative.
      - Could probably extrapolate data learned from the other bullet point methods on this slide and turn them into user-dependent features, such as “likes action movies”, “does not like old movies”, etc.
- Recommendations based off of both
  - movies most liked by other users who like the movie(s) the user likes
    - from one Die Hard to the next, a lot of the same movies are recommended, all of them admittedly similar to one another in genre, at the very least, plus in tag description tone.
      - this probably indicates this method results in recommendations that keep users in a bubble.
    - notice this way did result in Die Hard recommending another variation of Die Hard, as well as another Bruce Willis movies, Pulp Fiction, this time.