Northeastern University

Data Mining Project Report (Draft)

Recommendation Systems for Yelp Dataset

Aditya Priyadarshi, Abhay Kasturia, Xingxing Liu, Varun Nandu and Gautam Vashisht

Supervised by
Dr. Nate Derbinsky

November 22, 2017

1 Introduction and Related Work

A recommender system or a recommendation system is a subclass of information filtering systems that seeks to predict the rating or preference that a user would give to an item [2]. Recommendations systems have become very relevant today given the presence of e-commerce website like Amazon and Netflix as well as other platforms like Facebook and Youtube. These are utilized in a variety of areas such as movies, music, videos, news, books, research articles, search queries and products in case of Amazon. Two most common methods to build a recommendation system are collaborative filtering and content-based filtering. Collaborative filtering methods use user's past behaviors and behaviors of similar users to find items which a user might like. Content-based methods use the features of the items liked by the user to suggest similar items. There are also hybrid recommendation system which combine both of these techniques.

2 Dataset and Analysis

2.1 Dataset

The original dataset described in the Yelp Dataset Challenge 10 [1] has 4.7M reviews and 1M tips by 1.1M users for 156K businesses spread across 12 cities. The data is given in json format which include business.json, review.json, user.json, checkin.json and tip.json.Each business has name, address, star rating and textual reviews. Each individual review data consists of anonymized IDs for the business, user and review, star rating, review type, review text and votes on how useful, funny or cool the review is.

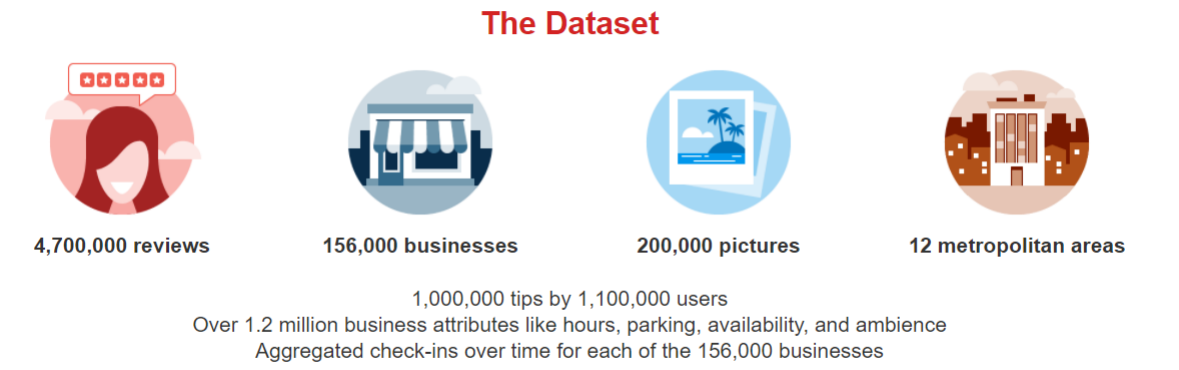

Figure 1: Dataset Details

2.2 Analysis

We did an initial analysis of the dataset. Below sections present our analysis.

2.2.1 User data

There are 1183362 total users whose reviews are present in the dataset. We plotted a histogram to understand the distribution of user reviews. Looking at the histogram, we can observe that most of the user have very few reviews and some top users have significant number of reviews. Majority of user have 25 or less reviews which is also shown by a mean of 23..72 and standard deviation of 80.5. The maximum number of reviews given by any user is 11656.

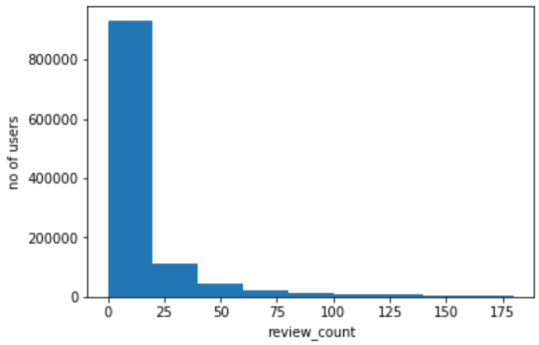

Figure 2: Review count per user

In addition to number of reviews, we also looked at distribution of star ratings given by a user. Looking at the histogram, we can observe that more users give higher rating which is shown by a median of 3.89 star rating. Mean and standard deviation for the same are 3.71 and 1.10 respectively. In order to group the reviews as positive, average and negative reviews, we have used the following method. We assume that if the rating lies in the range of (mean standard deviation, mean) which is 2.6 to 3.7, we will categorize it as average. Reviews lower than 2.6 will be considered as a negative review and anything greater than 3.7 will be considered as positive reviews with two extremes being 0 and 1.

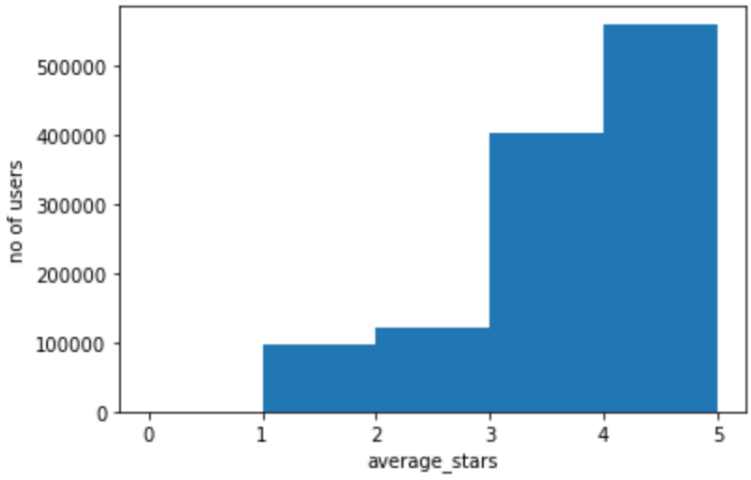

Figure 3: Rating per user

We also did some analysis to see the user growth on yelp. User growth has started declining

after an increase in users joining from 2005 to 2014 .

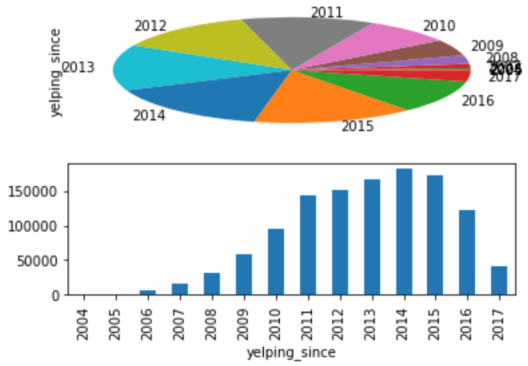

Figure 4: User Growth

2.2.2 Business Data

There are total 156639 business in the dataset. We grouped business according to city and business category to determine popular cities and categories. Below pie-charts give idea about popular cities and categories.

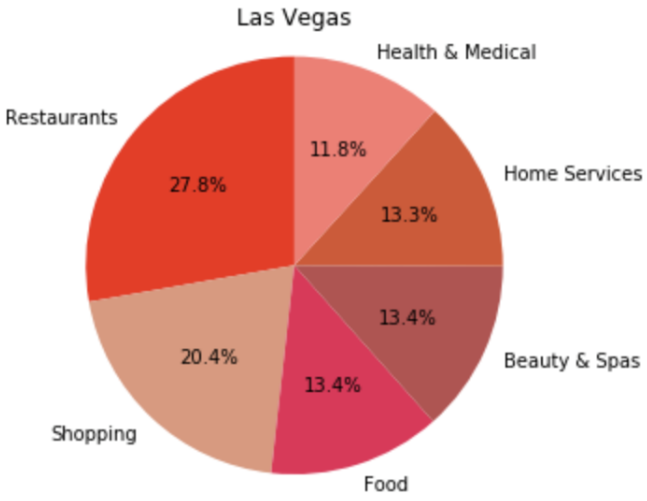

Figure 5: Top cities

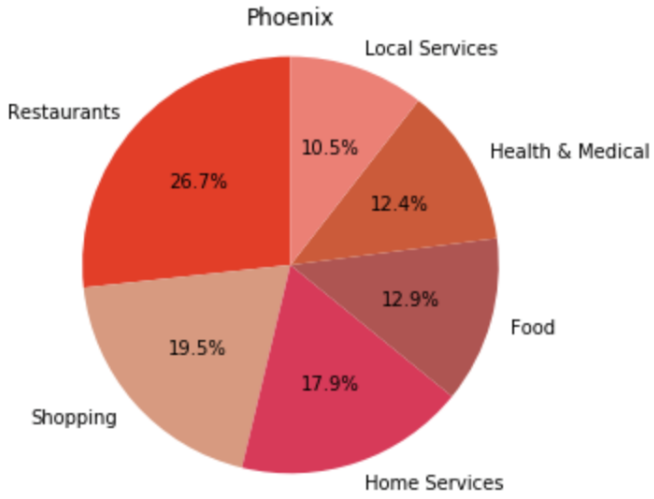

Figure 6: Top Business Categories

We did more analysis into sub-categories of our most popular category i.e. restaurants to map its distribution.

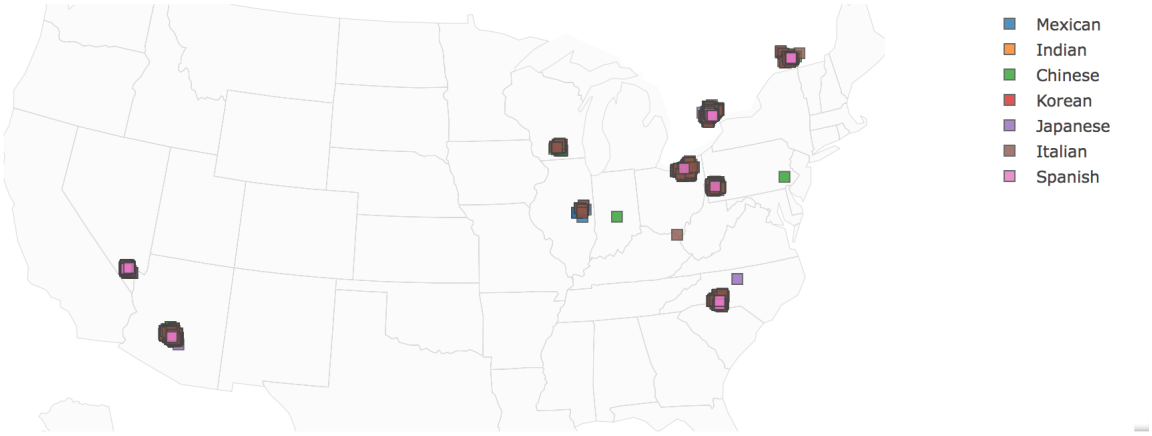

Figure 7: Restaurant Sub-categories

2.2.3 Checkin Data

Finally, we did analysis on use checkin data to find out popular timing in the top cities shown in our earlier analysis.

	city	time
0	Chandler	[(2:00, 34479), (19:00, 36988), (1:00, 39656)]
1	Charlotte	[(22:00, 64241), (17:00, 64472), (23:00, 72857)]
2	Cleveland	[(0:00, 21725), (22:00, 23961), (23:00, 25273)]
3	Edinburgh	[(12:00, 7720), (17:00, 7822), (18:00, 8324)]
4	Glendale	[(0:00, 20438), (2:00, 20472), (1:00, 23372)]
5	Henderson	[(20:00, 51439), (2:00, 53762), (1:00, 54136)]
6	Las Vegas	[(20:00, 441484), (1:00, 447715), (2:00, 481180)]
7	Madison	[(1:00, 15968), (23:00, 18773), (0:00, 19747)]
8	Mesa	[(2:00, 30860), (19:00, 31840), (1:00, 34917)]
9	Montréal	[(22:00, 17128), (0:00, 17196), (23:00, 18541)]
10	Phoenix	[(2:00, 170967), (19:00, 173996), (1:00, 188606)]
11	Pittsburgh	[(16:00, 36995), (22:00, 41485), (23:00, 43827)]
12	Scottsdale	[(0:00, 116834), (1:00, 128978), (19:00, 142432)]
13	Tempe	[(2:00, 49823), (19:00, 53595), (1:00, 54233)]
14	Toronto	[(0:00, 67390), (22:00, 67789), (23:00, 75615)]

Figure 8: Checkin Times

3 Methods

3.1 Clustering Based Approach

Clustering is based on the reviews of users. From the reviews we can build vector for each user and business with preference on different categories of businesses. By computing the distance of these two vectors, we can recommend businesses to user with the most similar preference vector value. There are basically four steps:-

1. Compose the preference vector of both users and businesses.
2. Clustering on user to generate a more general user preference vector
3. Calculating similarity of user preference and business preference
4. Generating recommendations

3.1.1 Vector Calculating

Preference Vector is built on a common top 20 categories of a certain type of business such as 'Restaurant'. That's saying 20 kinds of restaurant like 'Spanish', 'Chinese', 'Mexican', 'Indian' and so on.

```
['Food', 'Nightlife', 'Bars', 'Sandwiches', 'Fast Food', 'American (Traditional)', 'Pizza', 'Italian', 'Burgers',
 'Breakfast & Brunch', 'Mexican', 'American (New)', 'Chinese', 'Cafes', 'Coffee & Tea', 'Japanese', 'Chicken Wings',
 'Seafood', 'Event Planning & Services', 'Salad']
```

Figure 9: Top 20 Categories

Based on users' each review, we can get the preference rating from 1 to 5 for a particular business, from which we can generate a rating value for each of the category in that particular

business. For those categories user does not have review on, we assign a value of 1. Thus we get a vector of rating value on all top 20 categories for each user. The same procedure goes to business. Vector comes with format '[1,2,3, ...]', with respect to a list of categories '[Indian, Chinese, Mexican, Spanish, ...]'.

```
{ 'American (New)': 4, 'American (Traditional)': 4, 'Bars': 3, 'Breakfast & Brunch': 4,  'Burgers': 0, 'Cafes': 0,
'Chicken Wings': 0, 'Chinese': 0, 'Coffee & Tea': 0, 'Event Planning & Services': 3.75, 'Fast Food': 0,
'Food': 3.5714285714285716, 'Italian': 3.5, 'Japanese': 0, 'Mexican': 4, 'Nightlife': 3.3333333333333335,
'Pizza': 4, 'Salad': 0, 'Sandwiches': 0, 'Seafood': 2 }
```

Figure 10: Top 20 Categories

3.1.2 Clustering on Users

Given the vector of each user, we can calculate the Euclidean distance of each pair of users. Then perform average agglomerative clustering on the users' distance matrix. Dendogram after performing clustering on 8000 users is as shown in figure. Based on the dendogram we can decide on a cut-off point to get number of clusters. From the dendogram we selected 10 clusters.

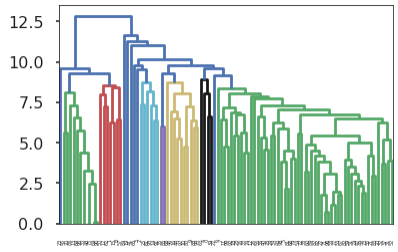

Figure 11: Dendogram after Clustering

3.2 Collaborative Filtering

Collaborative Filtering is the method to implement recommendation system. It is the way to recommend item to user 'u1' by collaborating the choice of item of other users similar to user 'u1'. We create a n X m matrix, where n is the number of users and m is the number of business, with each cell representing the rating given by a user to that particular business. Each row in matrix represent the vector of star rating given by user for all the businesses.

business_id user_id	-MhfebM0QIsKt87iDN-FNw	-cYOKJ5kbVZqzSYQlzZcqA	-lC6gIVhl7vY6W_dnw08YA	-pV9kWNoA9vyHfM_auYecA	03SYJLErY8XpNfY-qIDZcw	0nyN
---1lKK3aKOuomHnwAkAow	NaN	NaN	NaN	NaN	NaN	
--Nnm_506G_p8MxAOQna5w	NaN	NaN	NaN	NaN	NaN	
--P-Qvza7AED8gnDrZkMgA	NaN	NaN	NaN	NaN	NaN	
--ZNfWKj1VyVElRx6-g1fg	NaN	NaN	NaN	NaN	NaN	
-00MbjbaOlSrcuV7jOVRlg	NaN	NaN	NaN	NaN	NaN	

5 rows × 227 columns

Figure 12: UserVsBusiness - Stars Rating Value

While doing this, we face two issues 1. As expected, there are a lot of missing values in matrix(for items which user has not given rating). We treat missing values as the average computed after following the second step - which will always be zero. 2. There is an issue of handling the rating given by soft users and hard users i.e some users may rate the business they like with a 3 star ratings and there are some users who may rate the business they don't like with a 3 star ratings. To normalize these ratings for each user we use the centered cosine similarity. We normalize the ratings by subtracting the row mean for each user.[8]

To recommend new businesses to a target user 'target', we find the cosine similarity between all other users and 'target'. Top 'k' businesses rated with positive average star rating, by users of having cosine similarity greater than 1, will be recommended to user 'target'.[9]

3.3 Collaborative Deep Learing

We are using Collaborative Deep Learning [7] (CDL) approach suggested by Hao Wang and team. We looked at various deep learning approaches towards building recommendation systems and we choose this work as it was generally applicable compared to other techniques which either target music or videos recommendations. CDL is a hierarchical Bayesian model. Stacked Denoising autoencoders [10] are used for feature learning and cleaning the noise from the input. In below sections, I will be brief about the input parameters and neural network architecture.

3.3.1 Notation and Problem Formulation

As explained in [7], collection of J items (business) is represented by J-by-S matrix X_c, where row j is the bag-of-words vector $X_{c,j*}$ for item j based on a vocabulary size S. Assuming I users, an I-by-J rating matrix R. Given part of the ratings in R and the content information Xc, the problem is to predict the other ratings in R. Here, X_c is the clean matrix after using SDAE and X_o is the noise-corrupted matrix.

3.3.2 Stacked Denoising Autoencoders

SDAE [10] is a feedforward neural network for learning representations (encoding) of the input data by learning to predict the clean input itself in the output as shown in figure

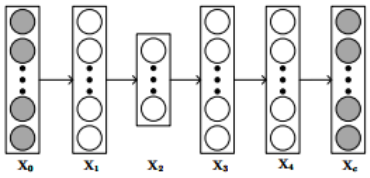

Figure 13: Stacked Denoising Autoencoders

3.3.3 Collaborative Deep Learing Model

Below figures show the model in detail, where W is weight matrix. The part inside the dashed rectangle represents an SDAE. An example SDAE with L=2 is shown. Here, λ_w, λ_n, λ_u, λ_s and λ_v are hyper-parameters. The middle layer $X_{L/2}$ serves as a bridge between the ratings and content information.

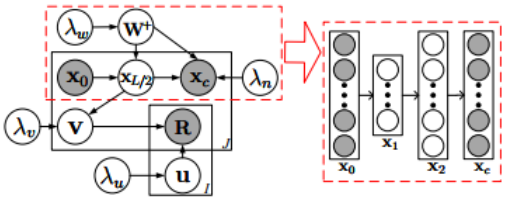

Figure 14: Graphical model of CDL

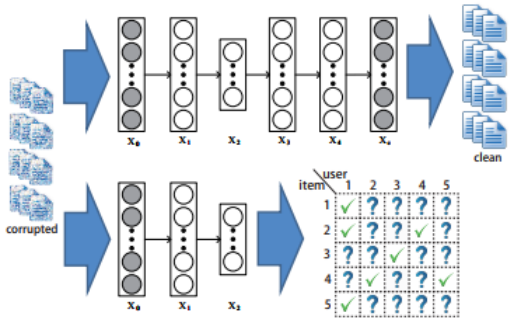

Figure 15: NN representation for degenerated CDL

4 Experiments and Results

4.1 Clustering Based Approach

4.1.1 Calculating Similarity and Generating Recommendations

Based on users clustering dendogram shown in figure, we generate an average vector representing user preference vector. Then we do an inner product of user vector and business vector. Smaller the value, more similar the user and the business. From the results, we choose top 10 or 20 business with the minimum value as the recommendations. Below is a sample for user with user is: 'McLqvYLBQoCFQikU9dC4cQ' searching for Restaurants.

```
Quaker Steak & Lube,1000 Airport Blvd,Coraopolis,PA
Michelle's Cafe,548 N Center St, Ste A,Lagrange,OH
Wendy's,27400 Chagrin Blvd,Beachwood,OH
Serendipity Bistro,393 Jane Street,Toronto,ON
Eggspectation,190 Sainte-Catherine Ouest,Montréal,QC
Pine House Cafe,21-5661 Steeles Avenue E,Toronto,ON
Five Star Cafe,1001 N Central Ave, Ste 10,Phoenix,AZ
Chickadee's Cafe,1617 Elizabeth Ave,Charlotte,NC
Bojangles' Famous Chicken 'n Biscuits,7047 South Blvd,Charlotte,NC
King Pin Grill & Pizza,8925 N 12th St,Phoenix,AZ
Bojangles' Famous Chicken 'n Biscuits,7701 Gateway Ln,Concord,NC
KFC,5940 Spring Mountain Rd,Las Vegas,NV
Guys Pizza Co,859 E 222nd St,Euclid,OH
Wild Wing,17 Worthington Avenue,Brampton,ON
Chunky Chicken,6001 14th Avenue, Unit B2,Markham,ON
Guys Pizza Co,5900 Ridge Rd,Parma,OH
Guys Pizza Co,1838 Coventry Rd,Cleveland Heights,OH
Bojangles' Famous Chicken 'n Biscuits,1101 W Sugar Creek Rd,Charlotte,NC
McDonald's,796 Burnhamthorpe Road W,Mississauga,ON
Church's Chicken,7925 S Rainbow Blvd,Las Vegas,NV
```

Figure 16: Results for User with id 'McLqvYLBQoCFQikU9dC4cQ'

4.1.2 Evaluation of the clustering

We used cophenetic correlation to evaluate the quality of the clustering. Since, for 8000 users our cophenetic correlstion value is about 0.7, we can assume that the quality of clusters formed is good.

```
size = 8000; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.68441209386784496
size = 5000; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.69580000510359319
size = 3000; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.70658970892720374
size = 2000; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.73391695357663456
size = 1000; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.73100111995588535
size = 500; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.73846506222107178
size = 100; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.83762182127852514
size = 50; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.87879134020736049
size = 10; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.91836580287816971
size = 3; c, coph_dists = cophenet(Z, pdist(X)); c is ideally close to one, c=0.55192082653918684
```

Figure 17: Cophenetic Correlation Distance for Different User Groupings

4.2 User-User Collaborative Filtering

For the initial setup, we have worked on 100,000 rows of reviews.json file. We have created a matrix of 78276 users and 4224 businesses.We randomly choose one user to find the set of similar users of count 200. Based on positive average star ratings given by 200 users, 531 set of businesses are recommended to users.

5 Future work

1. Modify original collaborative deep learning implementation[11] of paper [7] to analyze our dataset.

2. All the algorithms are run on a small subset of data. Look into distributive processing framework(Spark), to perform the same on all the data or leverage the Google Cloud Platform.

3. Finalize evaluation criteria. The general approach that we plan to take would be:

   (a) Divide training and test data based on the timeline of reviews - to get past activity and future activity.

   (b) Define users in future activity as target users. Use the past activity of the target users to make recommendations.

   (c) Evaluate the recommendation as correct, if the recommendation shows up in the users future activity.

   (d) Get the total accuracy based on the same and evaluate our model!

4. Handle the problem of cold start - where a new business or a user is added. The general approach here would be to recommend the most popular business to the new user. Have to search for a way to tackle new businesses.

References

[1] Yelp Dataset Challenge `https://www.yelp.com/dataset_challenge`

[2] Recommendation System Wiki `https://en.wikipedia.org/wiki/Recommender_system`

[3] Collaborative filtering `https://en.wikipedia.org/wiki/Collaborative_filtering`

[4] Collaborative filtering Techniques `https://www.youtube.com/watch?v=h9gpufJFF-0`

[5] Movie Recommenation System `https://beckernick.github.io/matrix-factorization-recommender/`

[6] Paul Covington, Jay Adams, Emre Sargin *Deep Neural Networks for YouTube Recommendations*, ACM 2016.

[7] Hao Wang, Naiyan Wang, Dit-Yan Yeung*Collaborative Deep Learning for Recommender Systems*, ACM 2015

[8] Lecture on CF - Stanford `https://www.youtube.com/watch?v=h9gpufJFF-0&t=854s`

[9] Recommendation System Netflix `https://www.youtube.com/watch?v=bLhq63ygoU8&t=5598s`

[10] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. *Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.* JMLR, 11:33713408, 2010

[11] CDL Implementation `https://github.com/hexiangnan/neural_collaborative_filtering`