40.016: The Analytics Edge
Week 11 Lecture 1

RECOMMENDATION SYSTEMS (PART 1)

Term 5, 2022

SINGAPORE UNIVERSITY OF
TECHNOLOGY AND DESIGN

Outline

1. Recommendation systems
2. Netflix prize
3. Clustering
4. Hierarchical clustering
5. K-means clustering

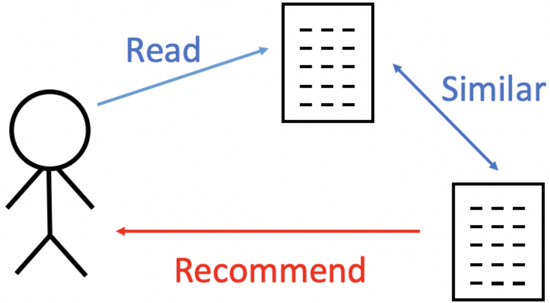
Outline

1. Recommendation systems
2. Netflix prize
3. Clustering
4. Hierarchical clustering
5. K-means clustering

Recommendation systems

- Personalize the user experience for online applications
- Leverage data on items and customers (e.g., likes, purchase history)
- A key challenge: they must
    - be fast
    - be accurate
    - work work with large/small datasets
    - some variables may have large variance
- Common underlying analytics:
    - clustering (Today)
    - collaborative and content filtering (Wednesday)

Recommendation systems (cont'd)



Outline

1. Recommendation systems
2. Netflix prize
3. Clustering
4. Hierarchical clustering
5. K-means clustering

Netflix cinematch

- Netflix introduced its recommendation system – called Cinematch – in 2000.
- According to Netflix, Cinematch was accurate within half a star 75 % of the time (Netflix users are asked to give a rating, from one to five stars, for any movie they watch).
- A more accurate recommendation system was an attractive option.
- Why Netflixs Algorithm Is So Binge-Worthy
  `https://www.youtube.com/watch?v=nq2QtatuF7U`
- Why your Netflix thumbnails don't look like mine
  `https://www.youtube.com/watch?v=axCBA3VD5dQ&feature=youtu.be`

Netflix prize

- In October 2006, Neflix announced that it would award a 1-million dollar prize to the first developer of a recommendation system that could beat Cinematch at predicting customer ratings by more than 10% on a test dataset.
- Netflix provided a training dataset of 100,480,507 ratings that 480,189 users gave to thousands of movies (18,000 movies).
- In addition, Netflix provided the ratings of 2,817,131 data points from the same subscribers over the same set of movies as a testing dataset.

Netflix prize (cont'd)

- First large-scale data competition – led to online platform for data competition (Kaggle)
- Any participating team had to predict the ratings on the entire testing dataset, while
    - Public leaderboard (roughly half of the data): was informed on the accuracy
    - Private leaderboard (the other half): was used to determine the ultimate winner
- Only the judges knew the partition in the testing dataset.

Netflix prize (cont'd)

- The predictions – which could be any number, not necessarily in the set $\{1, 2, 3, 4, 5\}$ – were scored against the true ratings in terms of Root Mean Squared Error (RMSE, to be minimized).
- The trivial algorithm that used the average rating for each movie from the training set produced a RMSE of 1.0540 on the testing set.
- Cinematch scored an RMSE of 0.9514 on the testing data using the training set to build the model (roughly, a 10% improvement).
- To win the prize, a team had to beat Cinematch by another 10% – that is, 0.8572 of the testing set.

Timeline of Netflix prize

- October 2006: the competition was launched
- November 2007: BellKor (a team of 3 AT&T lab researchers) won the 50,000 USD progress prize with a 8.43% improvement over Cinematch
- June 2009: Team BellKor Pragmatic Chaos achieves 10.05% improvement
- September 2009: Team BellKor Pragmatic Chaos officially wins the competition
    - A variety of method were used in the final winning predictive algorithm, including collaborative filtering, regression models, and LASSO...
    - A few words from the winner: `https://www.youtube.com/watch?v=ImpV70uLxyw`

MovieLens

- MovieLens is a recommendation system setup by GroupLens, a research lab based at the University of Minnesota (`https://grouplens.org`).
- GroupLens collects data on movie ratings and makes them available for research.

Summary

Data: The data contains information on the ratings of movies provided by multiple customers.

Model: clustering can be used to categorize movies and customers, while collaborative and content filtering can be used to make recommendations on which movie to watch.

Value and Decision: an effective recommendation system provides an edge over competitors by increasing sales and improving the customers satisfaction.

Outline

1. Recommendation systems
2. Netflix prize
3. Clustering
4. Hierarchical clustering
5. K-means clustering

Supervised learning

In supervised learning, one has typically access to a set of p features (or predictor variables) $\{x_1, \cdots, x_p\}$ and an output, or response variable, y.

- The goal is to predict y from $\{x_1, \cdots, x_p\}$.
- We have seen different supervised learning algorithms, such as linear regression, logistic regression, discrete choice models, CARTs, and Random Forests.

Unsupervised learning

In unsupervised learning, one has only access to a set of p features $\{x_1, \cdots, x_p\}$.

- The goal is to identify patterns within the data.
- One main challenge: it is far more subjective, since there is no clear way to perform cross-validation or validate the results on a testing set.
- However, these techniques are very important in various domains, such as recommendation systems.
- Clustering, for example, can be used to identify groups of shoppers based on their browsing and purchasing histories on, say, Amazon; then, an individual can be preferentially shown items based on the purchase histories of other shoppers in the same cluster.

Clustering (Descriptive analytics)

Goal: to partition observations into distinct groups so that

- the observations within each group are quite similar to each other (min. the within cluster variance)
- while observations in different groups are quite different from each other (max. the inter-cluster variance)

Note:

- It is an unsupervised problem, because we are trying to discover clusters within a dataset (and not to predict the outcome of a given variable).
- To make this concrete, we must define what it means for two or more observations to be similar or different. Indeed, this is often a domain-specific consideration that must be made based on knowledge of the data being studied.

Example 1

- Suppose that we have a set of n observations, each with p features.
  - n observations: tissue samples for patients with breast cancer
  - p features: measurements collected for each tissue sample; these could be clinical measurements, such as tumor stage or grade, or they could be gene expression measurements
- We may have a reason to believe that there is some heterogeneity among the n tissue samples; for instance, perhaps there are a few different unknown subtypes of breast cancer.
- Clustering could be used to find these subgroups. This is an unsupervised problem because we are trying to discover structure – in this case, distinct clusters – on the basis of a data set.

Example 2

- Another application of clustering arises in marketing.
- We may have access to a large number of measurements (e.g. median household income, occupation, distance from nearest urban area, and so forth) for a large number of people.
- Our goal is to perform market segmentation by identifying subgroups of people who might be more receptive to a particular form of advertising, or more likely to purchase a particular product.
- The task of performing market segmentation amounts to clustering the people in the data set.

Clustering: Difference w.r.t. PCA

They both seek to simplify the data via a small number of summaries, but their mechanisms are different:

- PCA looks to find a low-dimensional representation of the observations that explain a good fraction of the variance
- Clustering looks to find homogeneous subgroups (cluster) among the observations

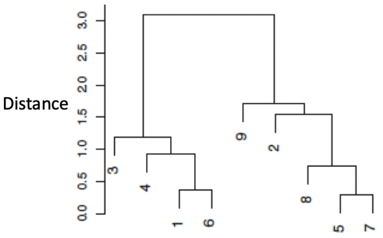
Common methods

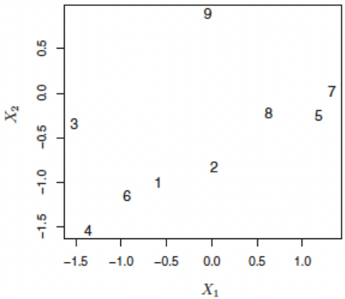
- **Hierarchical clustering** (bottom-up)
  - we do not know in advance how many clusters we want
  - end up with a tree-like visual representation of the observations
- **K-means clustering:** (top-down)
  - we partition the observations into a pre-specified number of clusters

Outline

1. Recommendation systems
2. Netflix prize
3. Clustering
4. Hierarchical clustering
5. K-means clustering

A motivating example from James et al. (2014)


Dendrogram (≠ CART)


$n = 9, p = 2$

- Each leaf of the dendrogram represents one of the 9 observations
- As we move up the tree, some leaves begin to fuse into branches
- As we move higher up the tree, branches themselves fuse, either with leaves or other branches.

A motivating example from James et al. (2014) (cont'd)

- The earlier (lower in the tree) fusions occur, the more similar the groups of observations are to each other.
- On the other hand, observations that fuse later (near the top of the tree) can be quite different.
- The height of this fusion, as measured on the vertical axis, indicates how different the two observations are.

A motivating example from James et al. (2014) (cont'd)

- Observations 5 and 7 are quite similar to each other, since they fuse at the lowest point on the dendrogram.
- However, it is tempting but incorrect to conclude from the figure that observations 9 and 2 are quite similar to each other on the basis that they are located near each other on the dendrogram.
- In fact, based on the dendrogram, observation 9 is no more similar to observation 2 than it is to observations 8, 5, and 7.
- Therefore, we cannot draw conclusions about the similarity of two observations based on their proximity along the horizontal axis. Rather, we draw conclusions about the similarity of two observations based on the location on the vertical axis where branches containing those two observations first are fused.

A motivating example from James et al. (2014) (cont'd)

- We move on to the issue of identifying clusters on the basis of a dendrogram.
- To identify clusters across a dendrogram, make a horizontal cut across the dendrogram: the number of vertical lines that it crosses determines the number of clusters.
- The distinct sets of observations beneath the cut can be interpreted as clusters.
- The furthest this horizontal line can move up and down without touching one of the horizontal lines the better it is.

Types of linkage

How do we define the dissimilarity between two clusters when we have multiple obs. in a cluster? Common types of **linkage**:

- Complete (maximal intercluster dissimilarity)
- Single (minimal intercluster dissimilarity)
- Average (mean intercluster dissimilarity)
- Centroid (dissimilarity between centroids)
- Ward's dissimilarity (balance inter and intra cluster variance): how much the sum of squares will increase when we merge them

Example?

Algorithm

Step 1. (Initialization) Begin with n observations and a measure (such as Euclidean distance) of all the $n(n-1)/2$ pairwise dissimilarities. Treat each observation as its own cluster.

Step 2. For $i = n, n-1, ..., 2$:

- Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters.
- Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters

Hierarchical clustering

- A very attractive aspect of hierarchical clustering: one single dendrogram can be used to obtain any number of clusters.
- In practice, people often look at the dendrogram and select by eye a sensible number of clusters, based on the heights of the fusion and the number of clusters desired.
- The term hierarchical refers to the observation that the clusters obtained by cutting the dendrogram at a particular height are nested within clusters obtained by cutting it at a greater height.
- However, on an arbitrary data set, this assumption of hierarchical structure might be unrealistic.

Example

- Suppose that our observations correspond to a group of people with a 50-50 split of males and females, evenly split among Americans, Japanese, and French.
- We can imagine a scenario in which the best division into two groups might split these people by gender, and the best division into three groups might split them by nationality.
- In this case, the true clusters are not nested, in the sense that the best division into three groups does not result from taking the best division into two groups and splitting up one of those groups.
- Consequently, this situation could not be well-represented by hierarchical clustering.

Outline

1. Recommendation systems
2. Netflix prize
3. Clustering
4. Hierarchical clustering
5. K-means clustering

Notation

- Let n be the number of observations.
- Let K denote the number of clusters.
- Let $C_1, C_2, \cdots, C_K$ denote sets containing the indices of the observations in each cluster. These sets satisfy two properties:
  - $C_1 \cup C_2 \cup \cdots \cup C_K = \{1, \cdots, n\}$ (each observation belongs to at least one of the K clusters)
  - $C_k \cap C_{k'} = \{\}$ for all $k' \neq k$ (clusters are non-overlapping)

Idea of K-means clustering

A good clustering is one for which the within-cluster variation is as small as possible. That means we want to solve the following problem:

$$\min_{C_1, \cdots, C_K} \sum_{k=1}^{K} W(C_k).$$

We quantify the within-cluster variation with the squared Euclidean distance:

$$W(C_k) = \frac{1}{|C_k|} \sum_{i, i' \in C_k} \underbrace{\sum_{j=1}^{p} (x_{ij} - x_{i'j})^2}_{\text{sq. Euclidean dist.}} \quad .$$

- p: number of variables
- C_k: points in cluster k

Idea of K-means clustering (cont'd)

Putting the previous equations together, we get

$$\min_{C_1,\cdots,C_K} \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

This is the optimization problem we want to solve.

- This is in fact a very difficult problem to solve precisely, since there are almost K^n ways to partition n observations into K clusters.
- This is a huge number unless K and n are tiny!
- The K-means algorithm uses an heuristic search process to provide a local optimum.

Algorithm

Step 1. Randomly assign a number, from 1 to K, to each of the observations (initial cluster assignment)

Step 2. Iterate until the cluster assignment does not change:

(a) For each of the K clusters, compute the cluster centroid (for the k-th cluster, the centroid is the vector of the p feature means for the observations in the k-th cluster)

(b) Assign each observation to the cluster whose centroid is closest

Discussion

- The algorithm is guaranteed to decrease the value of the objective function at each step.
- The following identity holds

$$\frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \bar{x}_{kj})^2,$$

  where $\bar{x}_{kj} = \frac{1}{|C_k|} \sum_{i \in C_k} x_{ij}$ is the mean for feature j in cluster C_k.
- In Step 2(a) the cluster means for each feature are the constants that minimize the sum-of-squared deviations
- In Step 2(b), reallocating the observations can improve the objective function

Illustration of the search process for a toy case

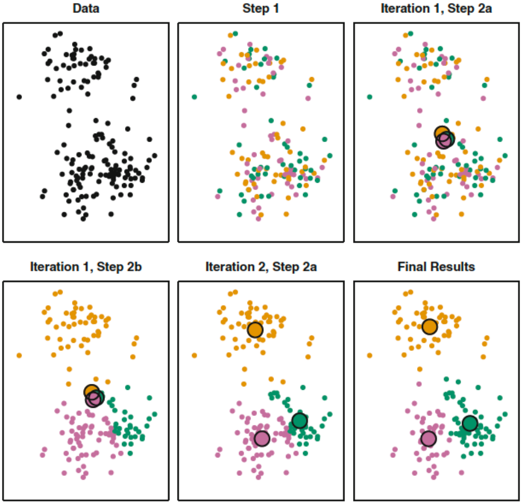


Figure: $n = 50$, $p = 2$, $K = 3$. Source: James et al. (2014)

A couple of considerations

- The results will depend on the initialization
- So, it is good practice to run the algorithm multiple times (with different initialization)
- We will the pick the configuration that minimizes the value of the objective function

Final remarks

Some important decisions we generally have to make:

- Should the observations or features first be standardized in some way?
- For hierarchical clustering:
    - What type of linkage should be used?
    - Where should we cut the dendrogram in order to obtain clusters?
    - (What dissimilarity measure should be used?)
- For K-means clustering:
    - How many clusters should we use?

Back to R!

- Hierarchical clustering

  `distances <- dist(Data, method="euclidean")` Calculate distance between observations

  `cluster <- hclust(distances, method="ward.D2")` Hierarchical clustering

  `plot(cluster)` Plot the dendrogram

  `Groups <- cutree(cluster, k)` Cut the dendrogram into k clusters

- K-means clustering

  `cluster <- kmeans(Data,centers,nstart)` K-means clustering using nstart random initializations

  `cluster$tot.withinss` Total within cluster sum of squares (min.)

References

Clustering and Recommendation systems:

- Teaching notes.
- James et al. (2014) *An Introduction to Statistical Learning with Applications in R*, Springer, 2014. Chapter 10.3.

Netflix:

- Koren Y. *The Bellkor solution to the Netflix grand prize.* Netflix prize documentation 81.2009 (2009): 1-10.
- From the Labs: Winning the Netflix Prize
  `https://www.youtube.com/watch?v=ImpV70uLxyw`
- Why Netflixs Algorithm Is So Binge-Worthy
  `https://www.youtube.com/watch?v=nq2QtatuF7U`
- Why your Netflix thumbnails don’t look like mine
  `https://www.youtube.com/watch?v=axCBA3VD5dQ&feature=youtu.be`