Data Mining and Machine Learning
K-means and Apriori

Gergely Horváth

October 6, 2021

Outline

1. Mining Association Rules
   - Apriori algorithm
2. Instance-based learning
3. Clustering
4. K-means

Mining Association Rules

Basic concepts:

- Let us have the following rule: $X \rightarrow Y$
- Support: $Supp(X \text{ and } Y) = \text{No. of transactions containing both } X \text{ and } Y$
- Confidence: $Conf(X \text{ and } Y) = \frac{Supp(X \text{ and } Y)}{Supp(X)}$
- Lift: $Lift(X \text{ and } Y) = \frac{Conf(X \text{ and } Y)}{Supp(Y)} = \frac{Supp(X \text{ and } Y)}{Supp(X) \cdot Supp(Y)}$
- The value for $Lift(X \text{ and } Y)$ will tell us about rule $X \rightarrow Y$, if:
  - $Lift(X \text{ and } Y) = 1 \rightarrow$ no association
  - $Lift(X \text{ and } Y) > 1 \rightarrow$ higher probability for Y when X is given
  - $Lift(X \text{ and } Y) < 1 \rightarrow$ lower probability for Y when X is given

Apriori algorithm

Steps:

1. Consider the support of individual cases first (e.g. $Supp('Sunny')$)
2. Consider the support of two individual cases first (e.g. $Supp('Sunny' \text{ and } 'Hot')$)
3. Same goes for 3 and 4 individual cases (5 would not make sense, *hint: no. of columns*)
4. Now that you have the support value for everything, you can compute the confidence and/or lift for any rule you please

Exercise: have some experiment with the provided code (*apriori.py*)

Outlook	Temp.	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Instance-based learning – KNN algorithm

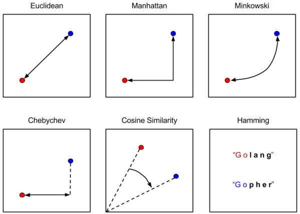
Basic principle

Given a reference instance, that we want to classify, we assign a label based of other 'similar' instances predicted previously (or part of the training set). $\rightarrow$ K-Nearest-Neighbour Classifier

What does 'similar' mean?

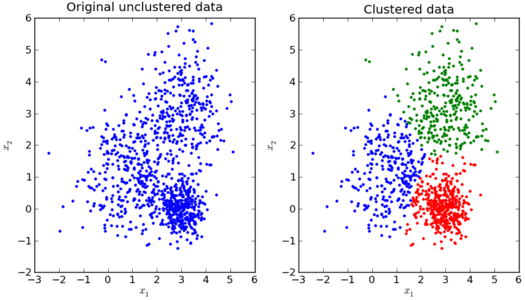
- Start to think in terms of n-dimensional feature spaces!
- 'Similar' means that they are close to each other in that space
- Therefore we have to compute a distance to assess similarity!
- Examples for distance metrics: Euclidean distance, Minkowski distance, or some more abstract way

$$d_M(i, j) = \left(\sum_k^n |x_{ik} - x_{jk}|^p \right)^{1/p}$$



Clustering

- Unsupervised learning
- Looking for more "similar" instances
- Similarity measure can be defined infinitely many ways:
  - Euclidean distance
  - Minkowski distance
  - Many more...
- Applications:
  - Medicine: PET-scan tissue types
  - Bioinformatics: sequence analysis (homologous sequences into gene families)
  - Recommendation systems: recommendations based on similar users



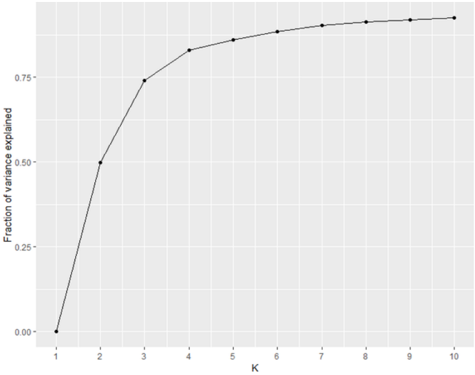
K-means – Steps

1. Select a value for K (select the number of clusters you would like to have), and initialize reference variance to infinity
2. Select K data point randomly (to initialize K clusters)
3. Compute the distance of each instances from each clusters, chose the minimum (instances assigned to clusters)
4. Compute the mean for each cluster
5. Do step 3 and 4 again, but with the new means as reference points for clusters, until the clusters are not modified
6. Add up the variances for each cluster, and store this set-up of initial data points and set is as reference variance if the overall variance is lower than the reference variance (instead of variance, Frobenius-norm difference between two iterations steps)
7. Do steps 2 – 6 N-times

K-means – Cont.

How to choose K?

- Prior knowledge
- Plot variance reduction as a function of K as a decision point (or inertia $\rightarrow$ in Sklearn a summed squared distance)



K-means – Verdict

Can we do better? Yes! Where?

- Result sensitive to the chosen value for K
- Cannot handle anisotropically distributed data
- Different variances
- The issue of random initialization of clusters
- Handling of large sample size ($\rightarrow$ MiniBatchKMeans)