

Introduction to Machine Learning

NPFL 054

<http://ufal.mff.cuni.cz/course/npfl054>

Barbora Hladká
hladka@ufal.mff.cuni.cz

Martin Holub
holub@ufal.mff.cuni.cz

Charles University,
Faculty of Mathematics and Physics,
Institute of Formal and Applied Linguistics

Lecture #2

Outline

- **Basic data analysis**
 - data for the Movie recommendation task
 - data for the Verb Pattern Recognition task
- **Clustering**
 - USArrest data set

Movie recommendation task (MOV)

Predict the user's rating for a given movie

	Toy Story (1995)	Star Wars (1977)	Some Like It Hot (1959)
Peter	?	5	4
Paul	2	5	?
Mary	2	4	?

E.g., predict Mary's rating for the *Some Like it Hot* movie

MOV – Getting examples

- Create a database of movies to be rated by users
- Set up a rating scale allowing users to rate movies
- Record users' ratings
- Typically, the dataset of ratings is sparse.
So do some pruning, like require a minimum of twenty ratings per user.

MOV – Available Data

Basic statistics

number of votes	100,000
number of movies	1,682
number of users	943

- Data comes from the MovieLens datasets
 - for more details, go to the course web page

MOV – Available data

- **About users**

	age	gender	occupation	zip code
Peter	19	M	student	58644
Mary	50	F	healthcare	60657

- **About movies**

title	action	...	IMDb rating	director
Toy Story	0	...	8.3	John Lasseter
Some Like It Hot	0	...	8.3	Billy Wilder
Star Wars	1	...	8.7	George Lucas

MOV – Available data

Data representation

	1	2	3	4	5-8	9-33
vote id	MOVIE	USER	RATING	TIMESTAMP	user features	movie features
1	1	1	5	1997-09-23 00:02:38	24 M technician 85711	Toy Story (1995) ...
...
100,000	1682	916	3

See the feature description [mov.pdf](#) at our course webpage.

Data analysis

Machine learning process

- ① Formulating the task (e.g., predict user's rating for a given movie)
- ② Getting data (e.g., MOV data)
 - Data analysis
- ③ Building predictor
- ④ Evaluation

Data analysis

Deeper understanding the task by statistical view on the data

We exploit the data in order to make prediction of the target value.

- Build intuition and understanding for both the task and the data
- Ask questions and search for answers in the data
 - **What values do we see?**
 - **What associations do we see?**
- Do plotting and summarizing

Data analysis

We focus on

- Recap of methods for basic data exploration
- Analyzing distributions of values
- Analyzing association between features
- Analyzing association between features and target value

We recommend you to read [this overview](#) of mathematical foundations needed for our course.

Methods for basic data exploration

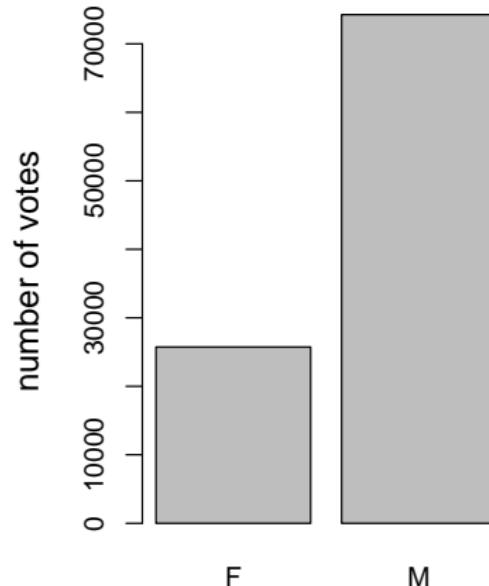
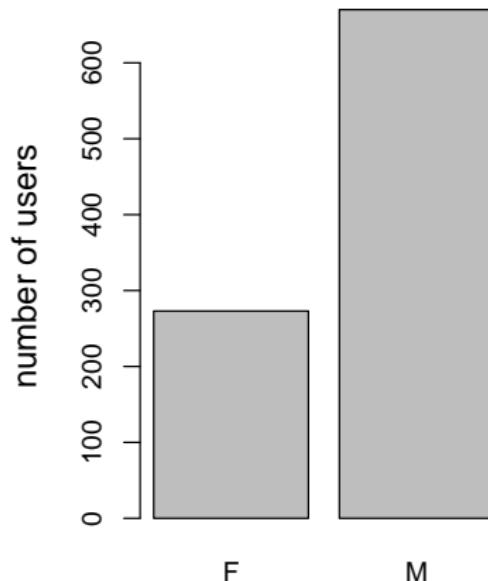
Frequency tables display the frequency of categorical feature values.

```
# frequency of voting men and women  
> source("load-mov-data.R") # see the course web page  
> table(examples$gender)  
F      M  
25740 74260
```

Methods for basic data exploration

Bar plots visualize frequency tables

Barplot (barplot-gender.R)



Methods for basic data exploration

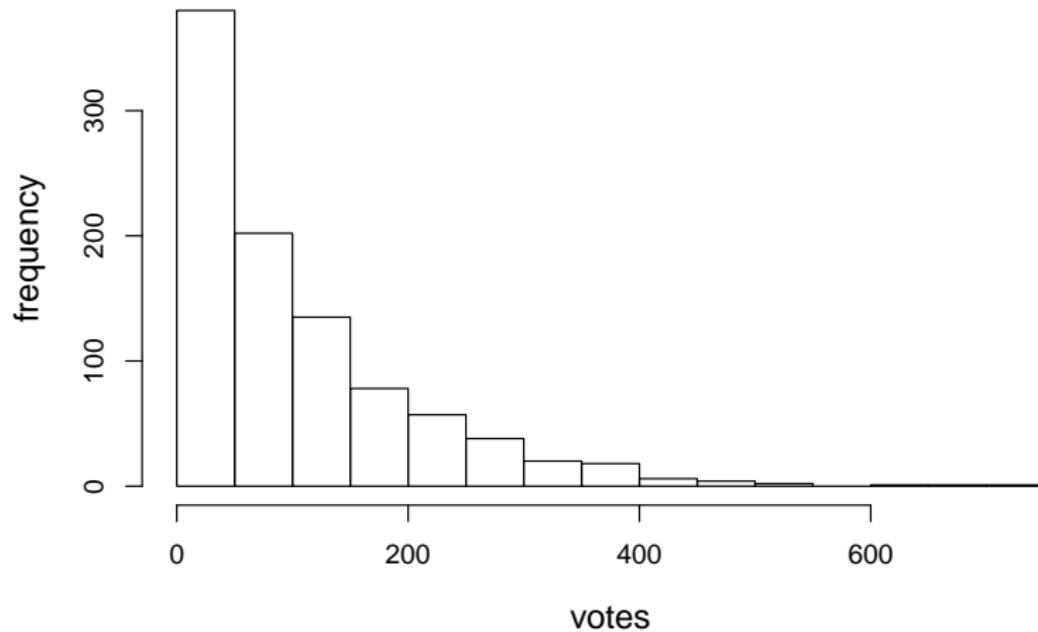
Histograms visualize distribution of feature values.

Add a new feature VOTES for the number of votes of the users

```
# get the number of votes for each user
> v <- as.data.frame(table(examples$user))
> users$votes <- v$Freq
> min(users$votes)
[1] 20
> max(users$votes)
[1] 737
```

Methods for basic data exploration

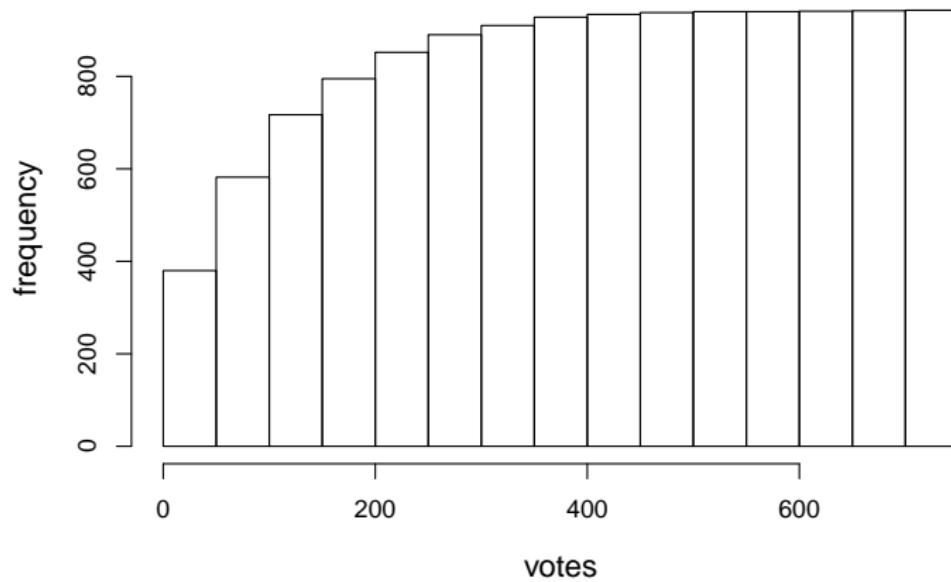
Histogram (histogram-votes.R)



Methods for basic data exploration

Cumulative histograms visualize cumulative frequencies.

Cumulative histogram (cumulative-histogram-votes.R)



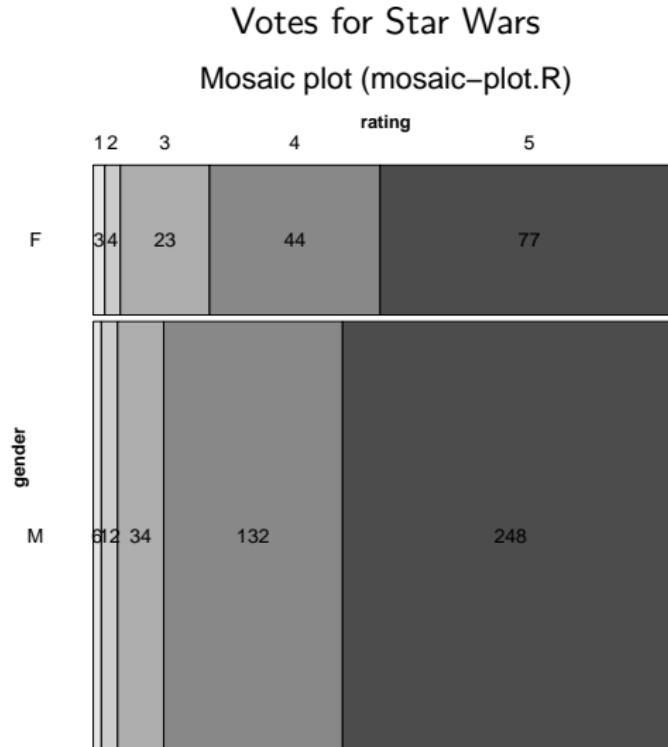
Methods for basic data exploration

Contingency tables display the frequency of values for combination of two categorical features.

```
> # Star Wars ratings
> movie <- subset(examples, movie == 50); nrow(movie) # 583
> # construct contingency table for gender and rating
> ct <- table(movie$gender, movie$rating)
> margin.table(ct)           # total sum
[1] 583
> addmargins(ct)            # adds marginal sums by default
      1   2   3   4   5 Sum
F     3   4  23  44  77 151
M     6  12  34 132 248 432
Sum   9  16  57 176 325 583
> round(prop.table(ct),3)    # prop.table generates proportions
      1     2     3     4     5
F 0.005 0.007 0.039 0.075 0.132
M 0.010 0.021 0.058 0.226 0.425
```

Methods for basic data exploration

Mosaic plots visualize contingency tables.



Methods for basic data exploration

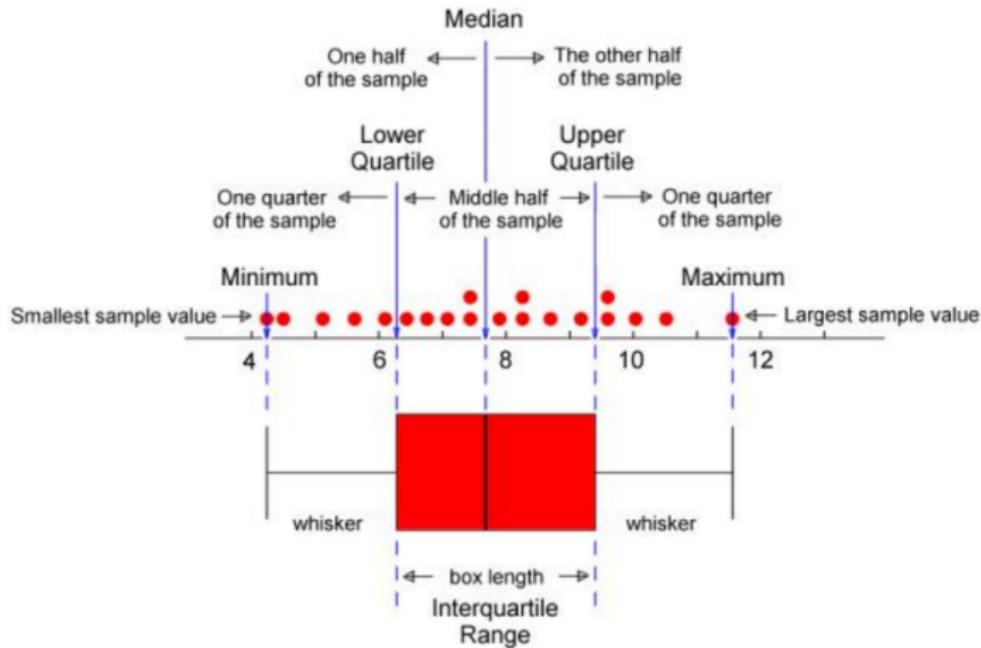
Measures of center and variation

```
> summary(users$votes) # five-number summary
   Min. 1st Qu. Median      Mean 3rd Qu.      Max.
   20       33     65      106     148      737

> mean(users$votes)
[1] 106.4
> median(users$votes)
[1] 65
> sd(users$votes) # standard deviation
[1] 100.93
```

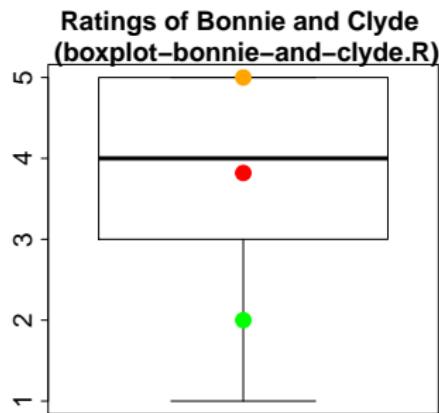
Methods for basic data exploration

Box-and-whiskers plots visualize five-number summaries.



Methods for basic data exploration

Box-and-whiskers plots

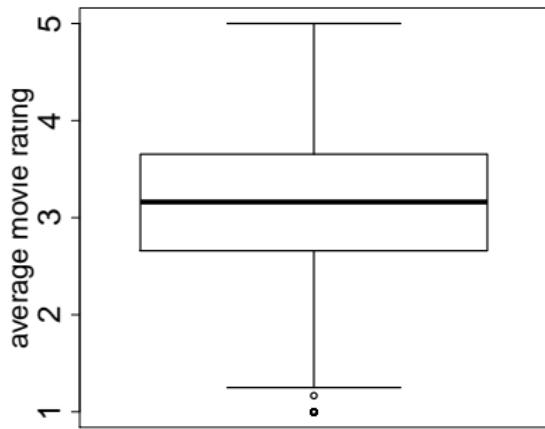


- the sample of 122 ratings
 - 1 2 3 4 5
 - 2 7 35 45 33
- $\text{median}(\mathbf{x}) = 4$
- $\bar{x} = 3.82$
- $sd(\mathbf{x}) = 0.95$
- the bottom whisker is much longer than the top whisker
- Peter's rating is in green and Mary's rating in orange

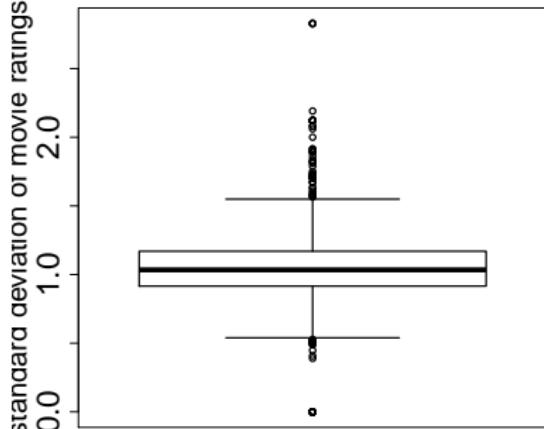
Methods for basic data exploration

- Boxplots are of a great importance to detect outliers and extreme values
- Outlier (Extreme value) is an observation that is distant from other observations, typically if it falls more than 1.5 (3)*($Q_3 - Q_1$) above Q_3 or below Q_1 .

(boxplot-movie-rating-mean-sd.R)



(boxplot-movie-rating-mean-sd.R)



Methods for basic data exploration

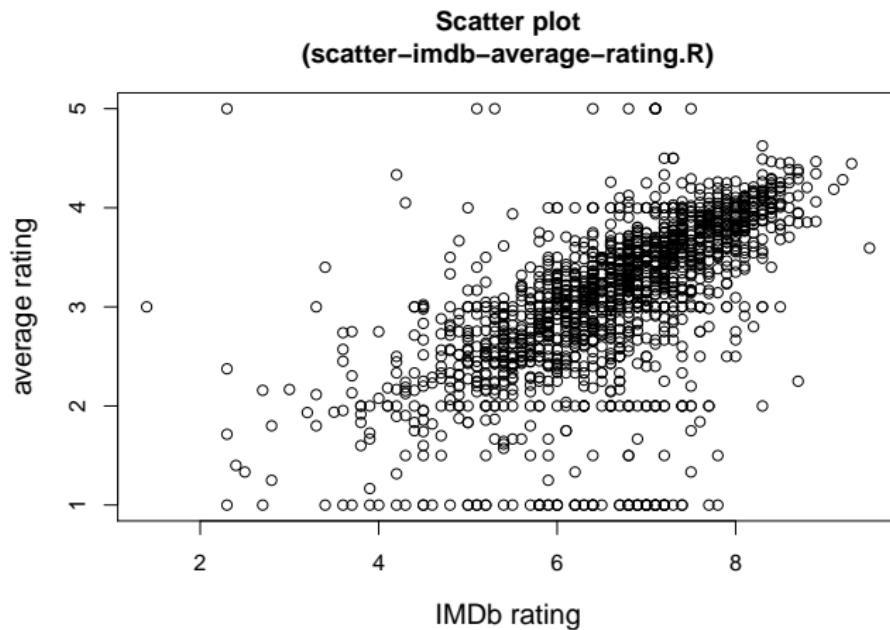
Boxplots are of a great importance to detect outliers and extreme values

```
> boxplot <- boxplot(tapply(votes$rating, votes$movie, sd))
# analyze outliers
> boxplot$out[1:2]
    247      314
1.788854 0.000000
>
> subset(votes, movie == 247) # Turbo: A Power Rangers Movie
   user movie rating           timestamp
38147    38    247      5 1998-04-13 03:04:20
38148     1    247      1 1997-09-26 04:40:19
38149   374    247      1 1997-12-01 01:35:22
38150   222    247      1 1997-11-05 08:29:58
38151   782    247      1 1998-04-02 08:48:20

> movies[movies$movie == 247,]
247 Turbo: A Power Rangers Movie (1997) 28-Mar-1997 ...
```

Methods for basic data exploration

Scatter plots display values of two numerical features.



Association between feature and target value

Numerical features

Covariance $\text{cov}(X, Y)$ is a measure of the joint variability of two random variables X and Y

$$\text{cov}(X, Y) = E[(X - EX)(Y - EY)]$$

The magnitude of the covariance is not easy to interpret because it is not normalized and hence depends on the magnitudes of the variables.

Therefore normalize the covariance → correlation coefficient

Association between feature and target value

Numerical features

Pearson correlation coefficient is a measure of the linear relationship between two variables

- **For a population**

$$-1 \leq \rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sqrt{\text{var}X \text{var}Y}} \leq +1$$

- perfect negative correlation if $\rho = -1$
- perfect positive correlation if $\rho = +1$
- not linear relationship if $\rho = 0$

- **For a sample**

$$-1 \leq r_{X,Y} = \frac{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{s_X s_Y} \leq +1$$

Association between feature and target value

Numerical features

$r(\text{Peter's rating}, \text{imdb_rating})$	0,51
$r(\text{Paul's rating}, \text{imdb_rating})$	0,44
$r(\text{Mary's rating}, \text{imdb_rating})$	0,37
$r(\text{Peter's rating}, \text{Mary's rating})$	0,29
$r(\text{Peter's rating}, \text{Paul's rating})$	0,29
$r(\text{Paul's rating}, \text{Mary's rating})$	0,24

Association between feature and target value

Categorical features

Pearson's χ^2 test

This test compares observed frequencies O_{ij} with theoretical frequencies E_{ij} that we would expect in case of statistical independence of X and Y . Test statistic

$\chi^2 = \sum_{i=1}^r \sum_{j=1}^s \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$ follows a χ^2 distribution with $(r - 1)(s - 1)$ degrees of freedom when the null hypothesis is true (r/s is the number of rows/columns in the contingency table).

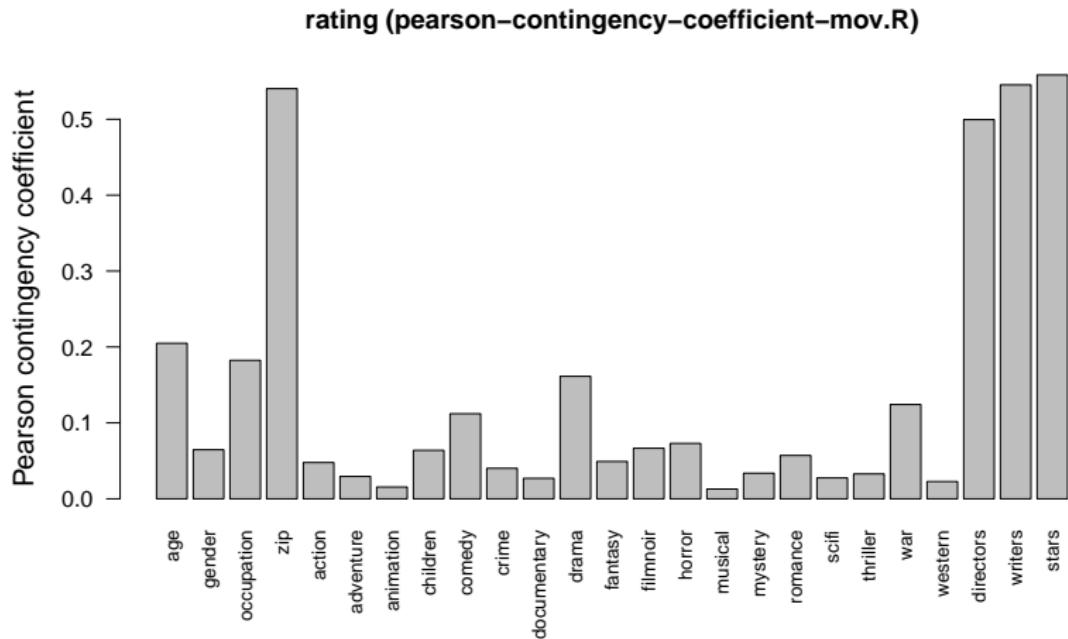
Pearson contingency coefficient

$$0 < \sqrt{\frac{\chi^2}{n + \chi^2}} < 1$$

- perfect correlation if $\rightarrow 1$
- no correlation if $\rightarrow 0$

Association between feature and target value

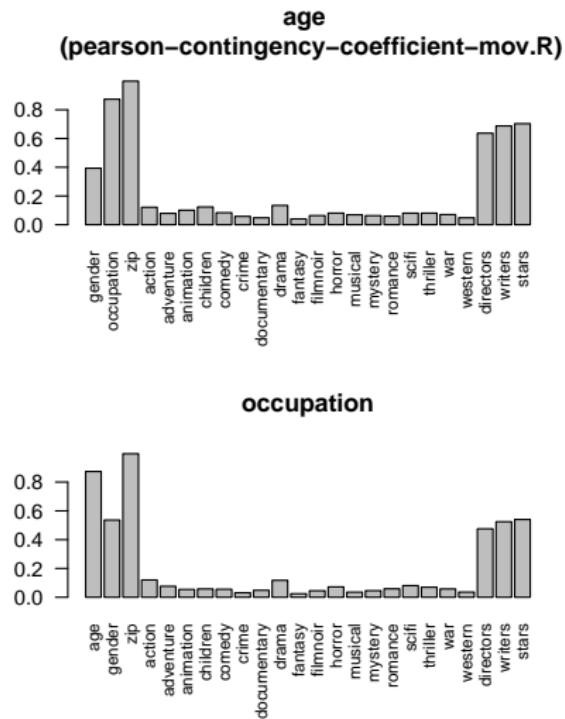
Categorical features



Association between features

Categorical features

Pearson contingency coefficient



Analyzing values

Feature frequency

- Feature frequency of binary features

$$\text{fr}(A_j) = \#\{\mathbf{x}_i \mid x_i^j > 0\}$$

where A_j is the j -th feature (binary), \mathbf{x}_i is the feature vector of the i -th instance, and x_i^j is the value of A_j in \mathbf{x}_i .

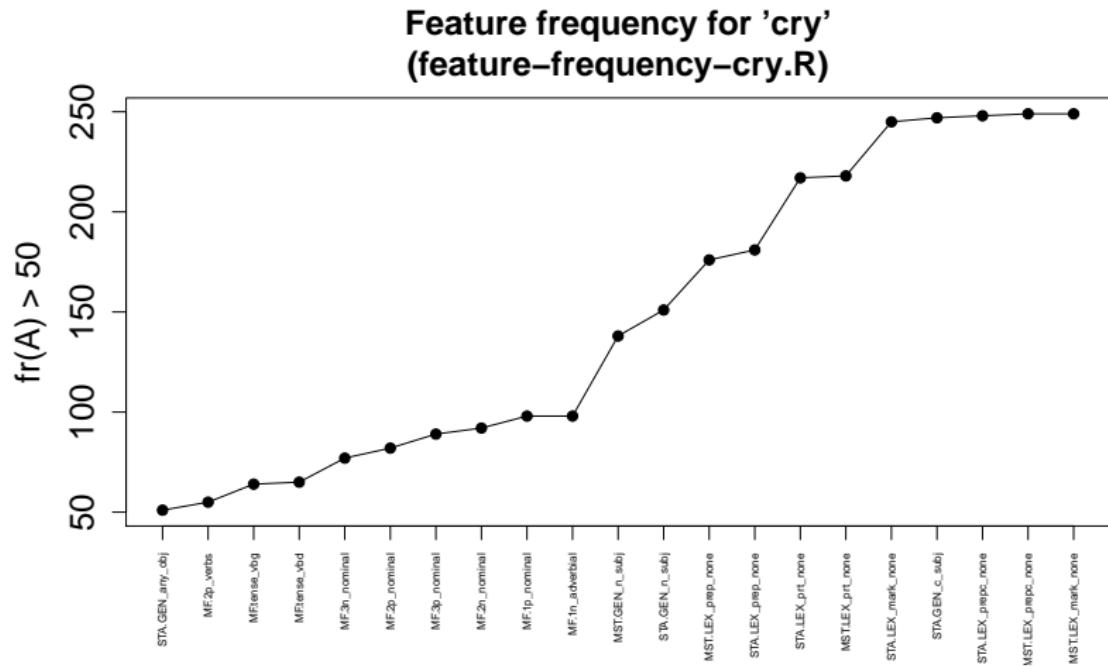
Analyzing values

Feature frequency – VPR data (cry)

```
> examples <- read.csv("cry.development.csv", sep="\t")
> c <- examples[,-c(1,ncol(examples))]
> nrow(examples)
[1] 250
> length(names(c)) # get the number of features
[1] 363
# compute feature frequency using the fr function (see feature-frequency-cry.R)
> ff <- apply(c, 2, fr) # apply fr to columns ('2') of c
> table(sort(ff))
  0   1   2   3   4   5   6   7   8   9   10  12  14  15  16  20 
181  47  26  12   9   3   5   6   4   4   7   1   3   1   2   1 
21  24  25  26  28  29  30  31  32  34  35  39  41  42  46  48  49 
  3   1   1   2   1   1   3   5   2   2   1   1   1   1   1   3   1 
51  55  64  65  77  82  89  92  98  138 151 176 181 217 218 245 
  1   1   1   1   1   1   1   1   2   1   1   1   1   1   1   1 
247 248 249
  1   1   2
```

Analyzing values

Feature frequency – VPR data (cry)



Analyzing values

Feature frequency – VPR data (cry)

Filter out uneffective features from the CRY data

```
> examples <- read.csv("cry.development.csv", sep="\t")
> n <- nrow(examples)
> ## remove id and target class tp
> c <- examples[,-c(1,ncol(examples))]

> ## remove features with 0s only
> c.1 <- c[, !lapply(c,fr) == 0 ]
> ## remove features with 1s only
> c.2 <- c.1[, !lapply(c.1,fr) == n ]
> ## remove column duplicates
> c.effective <- data.frame(t(unique(t(as.matrix(c.2)))))

> ncol(c)           # get the number of input features
[1] 363
> ncol(c.effective) # get the number of effective features
[1] 168
```

Analyzing values

Entropy – VPR data (cry)

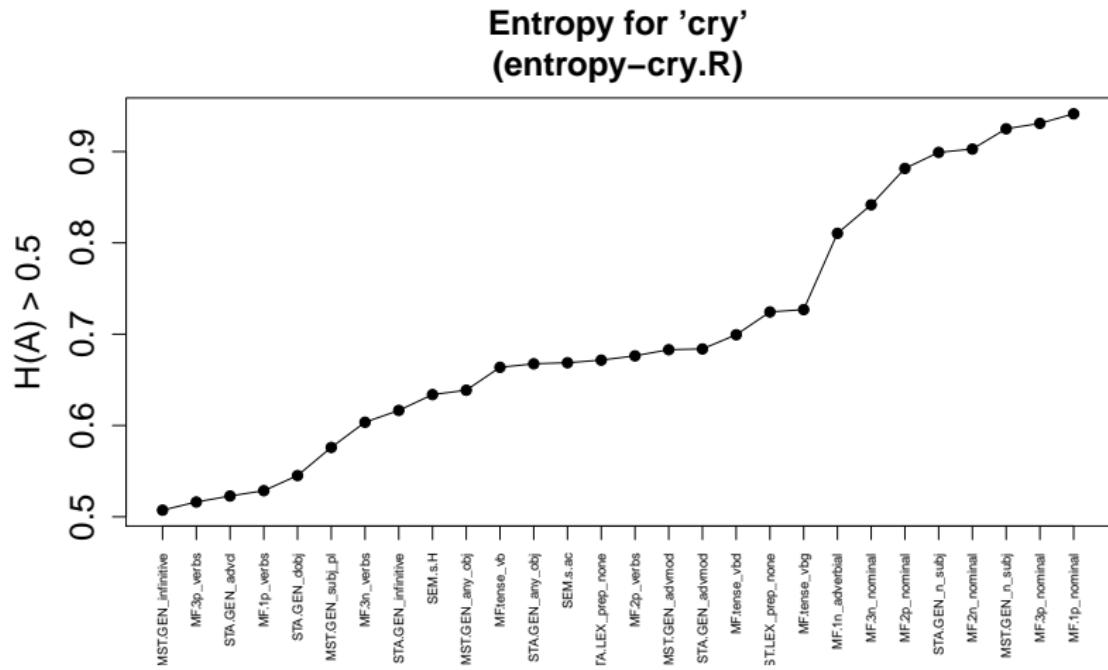
Entropy is a measure of the uncertainty in a random variable

$$H(X) = - \sum_{x \in X} \Pr(x) \log_2 \Pr(x)$$

```
# compute entropy using the entropy function (see entropy-cry.R)
> e <- apply(c, 2, entropy)
> table(sort(round(e,2)))
   0 0.04 0.07 0.09 0.12 0.14 0.16 0.18 0.2 0.22 0.24 0.28 0.31 0.33
181    49    27    13     9     4     5     6     4     4     7     1     3     1
   0.34 0.4 0.42 0.46 0.47 0.48 0.51 0.52 0.53 0.54 0.55 0.56 0.57 0.58
      2     1     3     1     1     2     1     1     3     5     3     1     2     1
   0.62 0.64 0.65 0.69 0.71 0.73 0.76 0.82 0.83 0.85 0.88 0.89 0.91 0.94
      1     1     1     1     4     1     1     1     1     1     1     1     1     1
   0.95 0.97 0.99
      1     3     1
```

Analyzing values

Entropy – VPR data (cry)



Association between feature and target value

Conditional entropy – VPR data (cry)

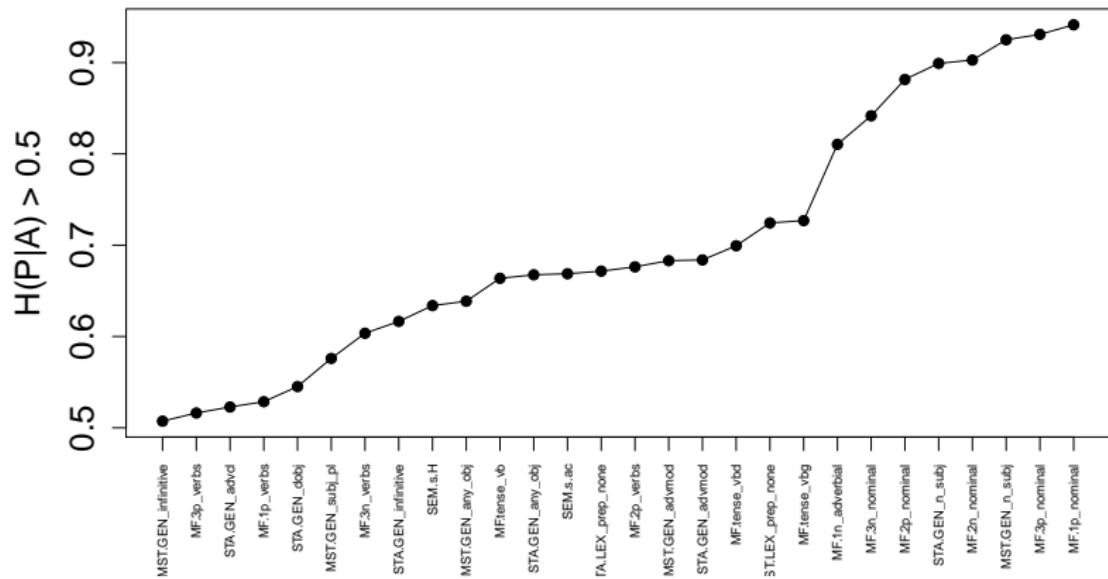
$$H(Y|X) = - \sum_{x \in X, y \in Y} \Pr(x, y) \log_2 \Pr(y|x)$$

```
# compute conditional entropy using entropy.cond (see entropy-cry.R)
ce <- apply(c, 2, entropy.cond, y=examples$tp)
table(sort(round(ce,2)))
  0 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1 0.11 0.12 0.13 0.14 0.15 0.16
181   3   47    1    7   19    1    7    6    1    7    2    2    1    2    4
0.17 0.18 0.2 0.21 0.22 0.23 0.25 0.26 0.27 0.28 0.29 0.3 0.31 0.32 0.37 0.39
  5    2    3    3    5    1    1    1    2    1    4    2    1    1    1    3
0.44 0.45 0.46 0.48 0.49 0.5 0.51 0.52 0.53 0.55 0.58 0.6 0.62 0.63 0.64 0.66
  1    2    1    2    1    1    1    2    1    1    1    1    1    1    1
0.67 0.68 0.7 0.72 0.73 0.81 0.84 0.88 0.9 0.92 0.93 0.94
  3    3    1    1    1    1    1    2    1    1    1
```

Association between feature and target value

Conditional entropy – VPR data (cry)

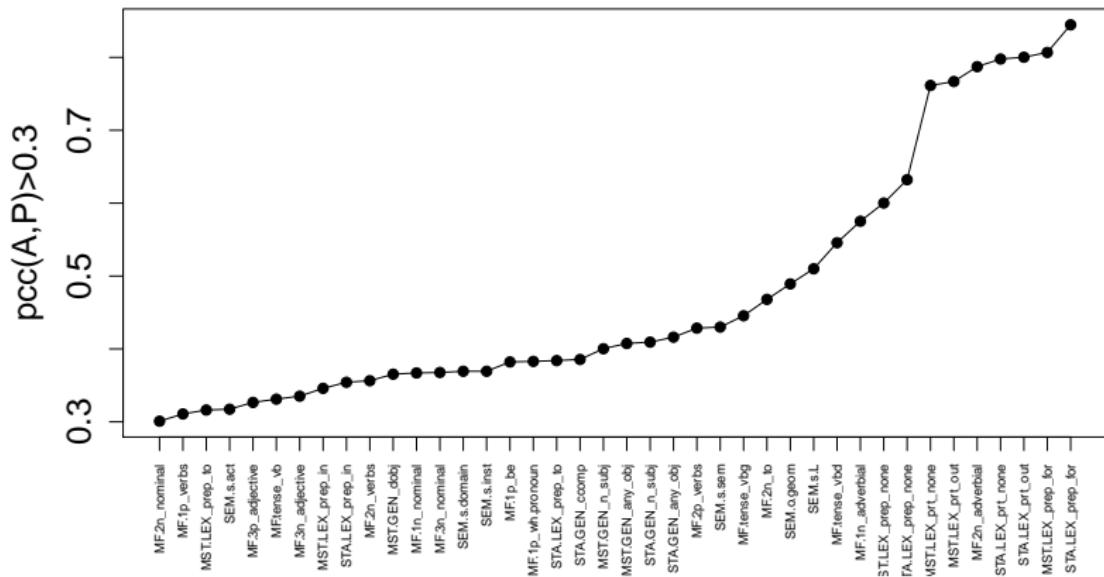
Conditional entropy for 'cry'
(entropy-cry.R)



Association between feature and target value

Pearson contingency coefficient – VPR data (cry)

Pearson contingency coefficient for 'cry'
(pearson-contingency-coefficient-vpr.R)

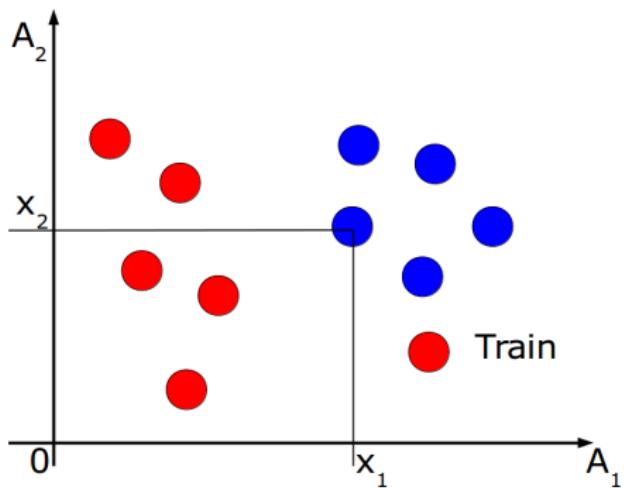


Clustering

Supervised vs. Unsupervised learning

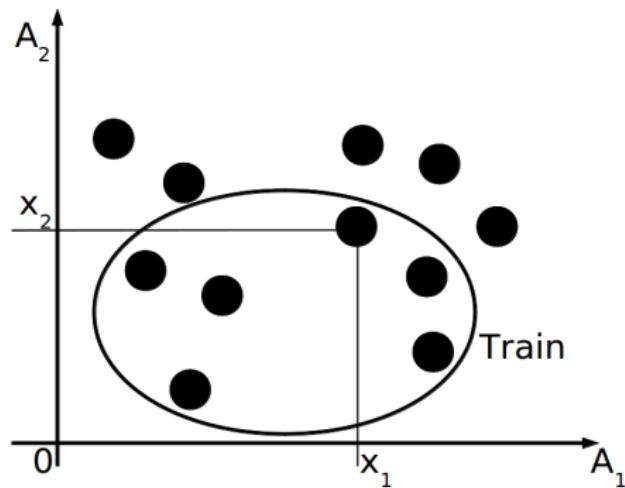
Supervised learning

$$Data = \{\langle \mathbf{x}, y \rangle : \mathbf{x} \in X, y \in Y\}$$



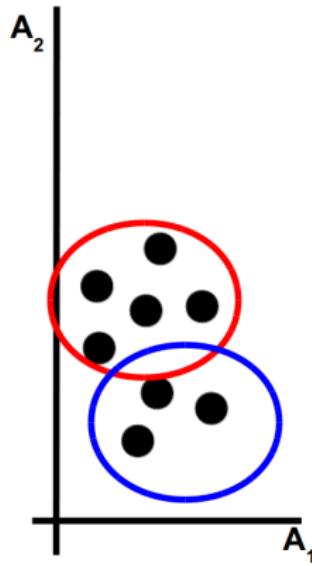
Unsupervised learning

$$Data = \{\mathbf{x} : \mathbf{x} \in X\}$$



Clustering

Clustering finds homogenous subgroups among the instances in the unlabeled data.



Clustering

Discovering structure

The most common criteria

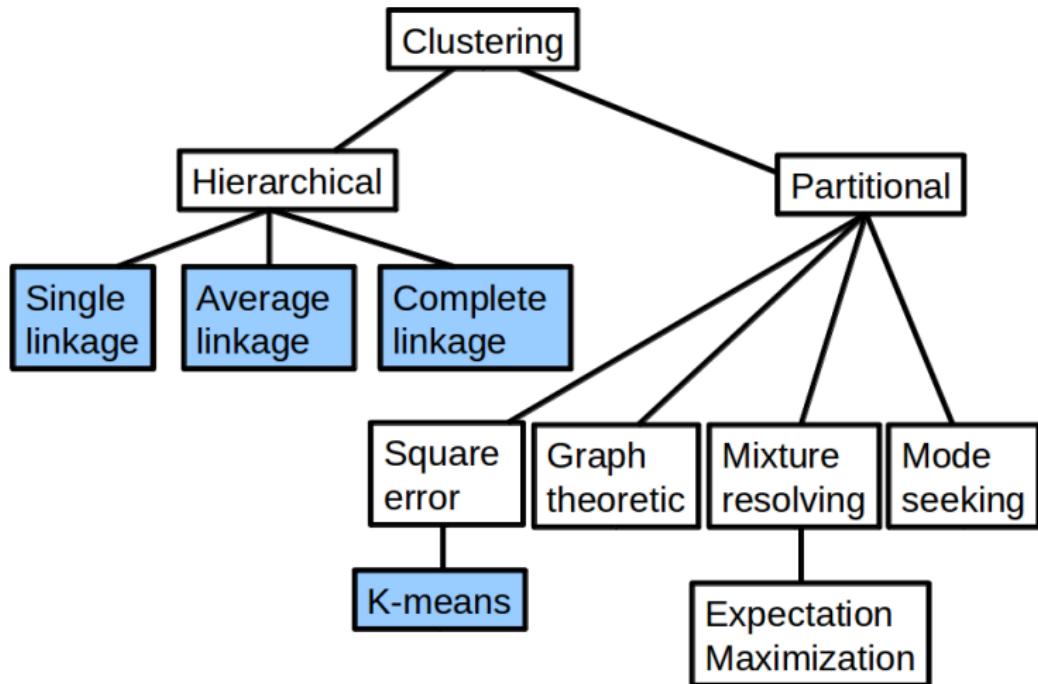
- **Homogeneity**

Objects within a same cluster should be similar each other

- **Separation**

Objects in different clusters should be dissimilar from each other

Clustering algorithms



Credits: (Kononenko, Kukar, 2007)

Clustering

Similarity metrics

The most common one

- **Cosine similarity**

$$\text{sim}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\sqrt{||\mathbf{u}||^2 \cdot ||\mathbf{v}||^2}}$$

Clustering

Dissimilarity metrics

Dissimilarity can be thought of as distance. The most common ones:

- **Euclidean distance** – continuous features

$$d(\mathbf{u}, \mathbf{v}) = \sqrt{\sum_{i=1}^m (u_i - v_i)^2}$$

- **Manhattan distance** – continuous features

$$d(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^m |u_i - v_i|$$

- **Hamming distance** – categorical features

$$d(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^m (u_i \neq v_i)$$

Clustering algorithms

Notation

- $Data = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$
- A set of k clusters $C = \{C_1, C_2, \dots, C_k\}$ containing the indeces of the instances
- $C_1 \cup \dots \cup C_k = \{1, 2, \dots, n\}$
- $C_j \cap C_i = \emptyset, \forall i \neq j$
- i -th cluster centroid $\mu(C_i) = \frac{1}{|C_i|} \sum_{j \in C_i} \mathbf{x}_j$

Clustering algorithms

Loss functions

- **Within-cluster variation** $L(C_i)$

$$L(C_i) = 2 \sum_{l \in C_i} d(\mathbf{x}_l, \mu(C_i))^2$$

(originally $L(C_i) = \frac{1}{|C_i|} \sum_{l,j \in C_i} d(\mathbf{x}_l, \mathbf{x}_j)^2$)

- **Total within-cluster variation** $L(C_1, \dots, C_k)$

$$L(C_1, \dots, C_k) = \sum_{i=1}^k L(C_i)$$

The most common choice of d involves Euclidean distance.

K-means algorithm

Optimization problem

$$\operatorname{argmin}_{C_1, \dots, C_k} L(C_1, \dots, C_k) = \operatorname{argmin}_{C_1, \dots, C_k} \sum_{i=1}^k L(C_i) \quad (1)$$

K-means algorithm

① Create clusters C_1^0, \dots, C_k^0

- randomly assign a number, from 1 to k , to each of the instance so that each cluster contains at least one instance

② while a *stopping criteria* is not met do

a) **centroid update:** for all clusters $C_i^t, i = 1, \dots, k$ do

$$\mu(C_i^t) = \frac{1}{|C_i^t|} \sum_{j \in C_i^t} \mathbf{x}_j$$

b) **data assignment:** for all clusters $C_i^t, i = 1, \dots, k$ do

$$C_i^{t+1} = \{j; d(\mathbf{x}_j, \mu(C_i^t))^2 \leq d(\mathbf{x}, \mu(C_i^t))^2, \forall I \neq i\}$$

Stopping criteria: no data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached

K-means

Convergence animation

Dataset USArrests from the base R distribution

- statistics on arrests per 100,000 residents in each of the 50 US states in 1973

```
> attributes(USArrests)
$names
[1] "Murder"    "Assault"   "UrbanPop"  "Rape"
# UrbanPop is the percent of the population living in urban areas
$class
[1] "data.frame"
$row.names
[1] "Alabama"      "Alaska"     "Arizona"    "Arkansas"
[5] "California"   "Colorado"   "Connecticut" "Delaware"
[9] "Florida"       "Georgia"    "Hawaii"    "Idaho"
[13] "Illinois"      "Indiana"    "Iowa"      "Kansas"
[17] "Kentucky"      "Louisiana" "Maine"     "Maryland"
[21] "Massachusetts" "Michigan"   "Minnesota" "Mississippi"
[25] "Missouri"      "Montana"    "Nebraska"  "Nevada"
[29] "New Hampshire" "New Jersey" "New Mexico" "New York"
[33] "North Carolina" "North Dakota" "Ohio"      "Oklahoma"
[37] "Oregon"        "Pennsylvania" "Rhode Island" "South Carolina"
[41] "South Dakota"   "Tennessee"  "Texas"     "Utah"
[45] "Vermont"        "Virginia"   "Washington" "West Virginia"
[49] "Wisconsin"      "Wyoming"
```

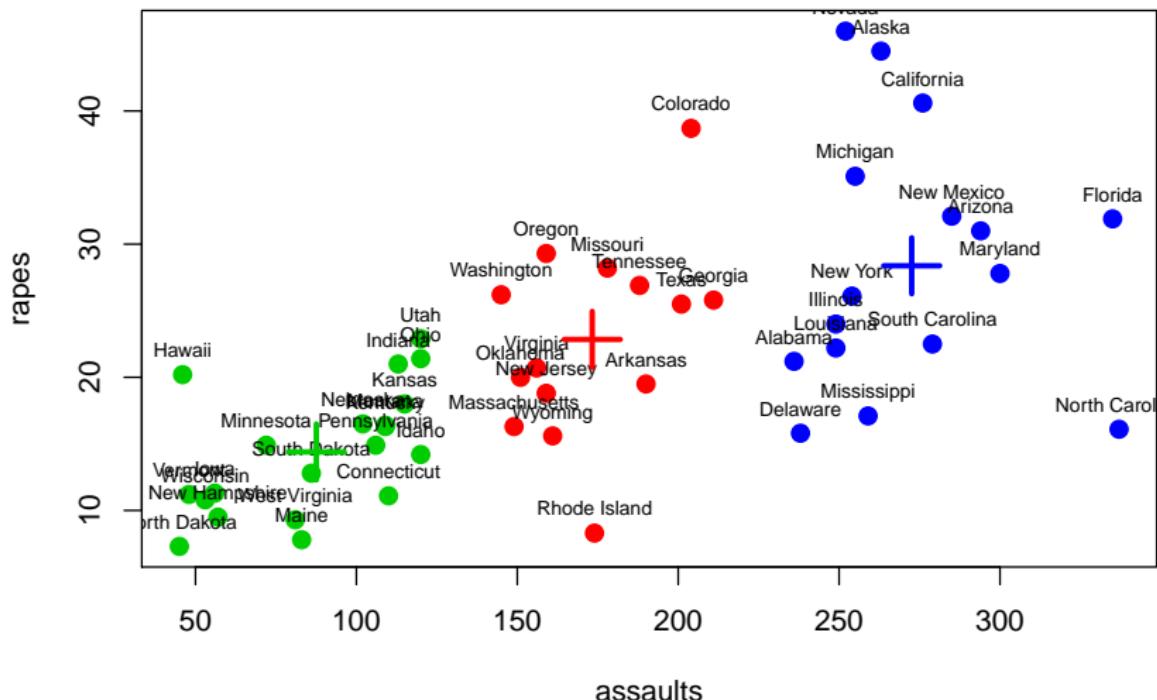
K-means algorithm with the USArrests data

```
> str(USArrests)
'data.frame':      50 obs. of  4 variables:
 $ Murder   : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
 $ Assault  : int  236 263 294 190 276 204 110 238 335 211 ...
 $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
 $ Rape     : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8  ...

> d <- USArrests
> examples <- d[,c(2,4)]
> km.3 <- kmeans(examples, 3, nstart=20)
> km.3$tot.withinss
[1] 38435.53
> km.3$withinss
[1] 15847.167  7109.191 15479.168
```

K-means algorithm with the USArrests data

K-means, K = 3



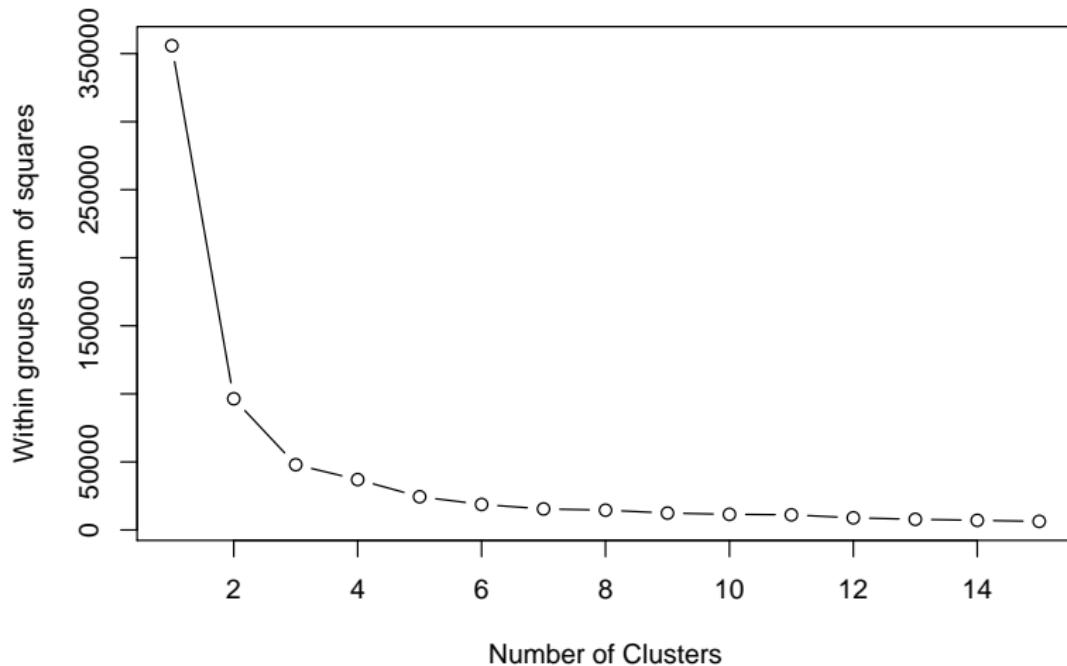
K-means algorithm with the USArrests data

Multiple initial cluster assignments (nstart)

```
> km.1 <- kmeans(examples, 6, nstart=1)
> km.1$tot.withinss
[1] 13711.87
>
> km.20 <- kmeans(examples, 6, nstart=20)
> km.20$tot.withinss
[1] 10282.92
```

K-means algorithm with the USArrests data

Which K to choose? Use e.g. **Elbow method**



K-means algorithm

Remarks

- The results depend on the initial clusters. The standard solution is to try a number of different starting points (see `nstart` in R).
This is an annoyance that must be handled in an implementation.
- The results depend on the metric used to measure similarity.
- The results depend on the value of K .

Hierarchical clustering methods

- do not require specification of the number of clusters.
- do produce tree-based representation of the instances, called **dendrogram**

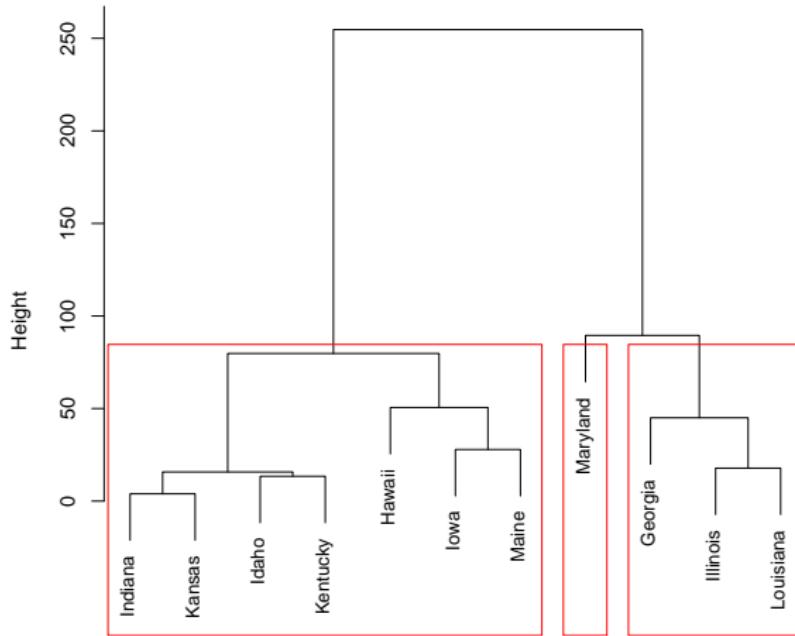
Dendrogram

is a rooted binary tree where

- the root node represents an input data set $Data$, $|Data| = n$
- the internal nodes represent the groups of instances
- each non-terminal node has two daughter nodes
- each terminal node represents one of the input instances (n terminal nodes)

Dendrogram

height = distance



How to read a dendrogram

- location of instances on the horizontal axis says nothing about the similarity
- location on the vertical axis: dissimilarity between the clusters when they were merged
- cutting the dendrogram \sim getting clusters

Hierarchical clustering methods

Agglomerative (bottom-up) clustering

- ① Start with each instance in its own singleton cluster
- ② At each step, greedily merge 2 most similar clusters
- ③ Stop when there is a single cluster of all examples, else go to 2

Divisive (top-down) clustering

- ① Start with all instances in the same cluster
- ② At each step, remove the “outsiders” from the least cohesive cluster
- ③ Stop when each example is in its own singleton cluster, else go to 2

Agglomerative (bottom-up) hierarchical methods

- ① for $i := 1$ to n do $C_i := \{\mathbf{x}_i\}$ end
- ② $C := \{C_1, C_2, \dots, C_n\}$
- ③ $j := n + 1$
- ④ while $|C| > 1$
 - ① $(C_{n_1}, C_{n_2}) := \operatorname{argmax}_{C_u, C_v \in C \times C} \operatorname{sim}(C_u, C_v)$
 - ② $C_j = C_{n_1} \cup C_{n_2}$
 - ③ $C := C \setminus \{C_{n_1}, C_{n_2}\} \cup C_j$
 - ④ $j := j + 1$

Agglomerative hierarchical methods

Work with distance (dissimilarity) measures

- dissimilarity between instances $d(\mathbf{x}_i, \mathbf{x}_j)$
- dissimilarity between clusters $d(C_u, C_v)$
 - then 4.1 in the algorithm is

$$(C_{n_1}, C_{n_2}) := \operatorname{argmin}_{C_u, C_v \in C \times C} d(C_u, C_v)$$

where $d(C_u, C_v)$ is a **linkage function**

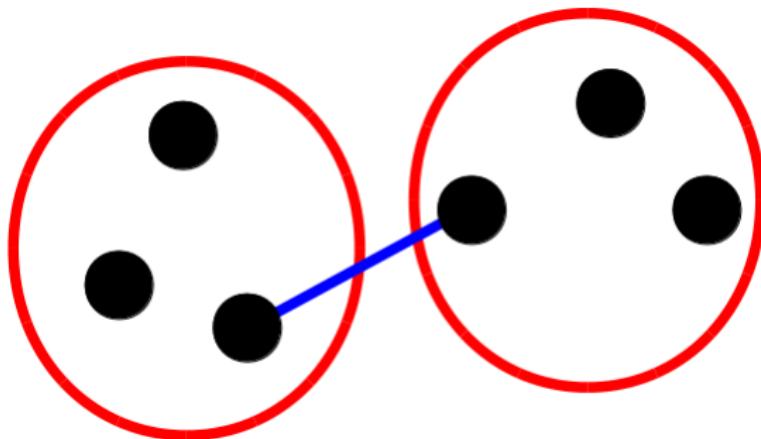
The choice of linkage function determines how we measure dissimilarity between clusters.

Dissimilarity between clusters

Single linkage clustering

The minimum dissimilarity between instances of each cluster

$$d(C_u, C_v) = \min_{\mathbf{x}_i \in C_u, \mathbf{x}_j \in C_v} d(\mathbf{x}_i, \mathbf{x}_j)$$

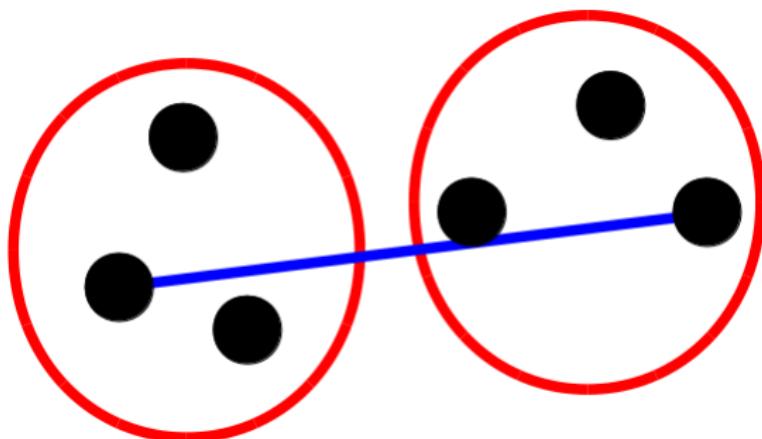


Dissimilarity between clusters

Complete linkage clustering

The maximum dissimilarity between instances of each cluster

$$d(C_u, C_v) = \max_{\mathbf{x}_i \in C_u, \mathbf{x}_j \in C_v} d(\mathbf{x}_i, \mathbf{x}_j)$$

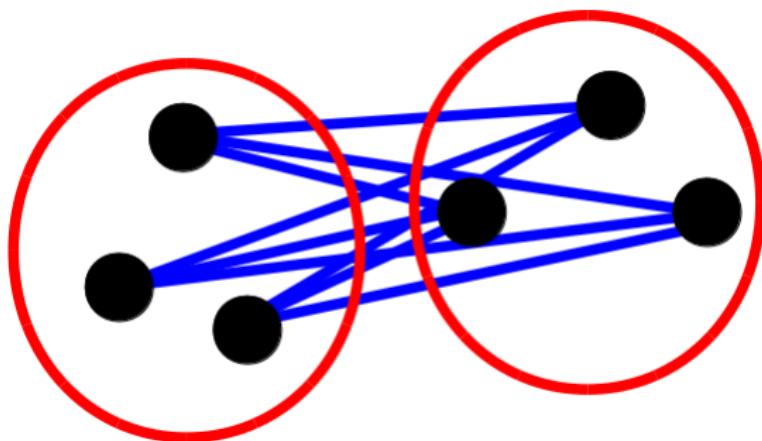


Dissimilarity between clusters

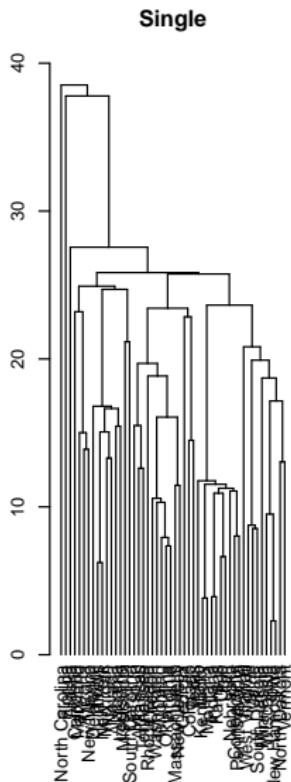
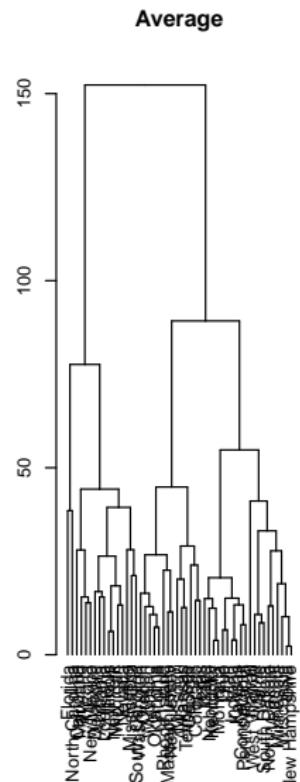
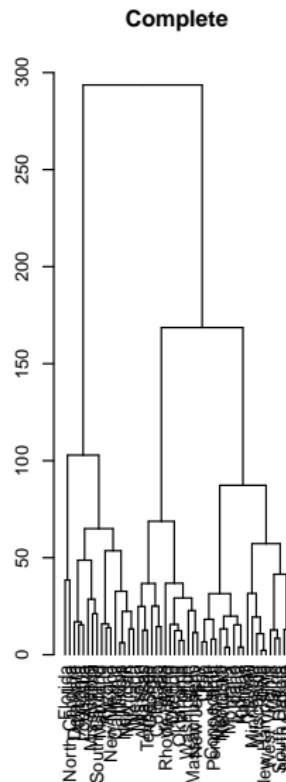
Average linkage clustering

The mean dissimilarity between instances of each cluster

$$d(C_u, C_v) = \frac{1}{|C_u||C_v|} \sum_{x_i \in C_u} \sum_{x_j \in C_v} d(x_i, x_j)$$



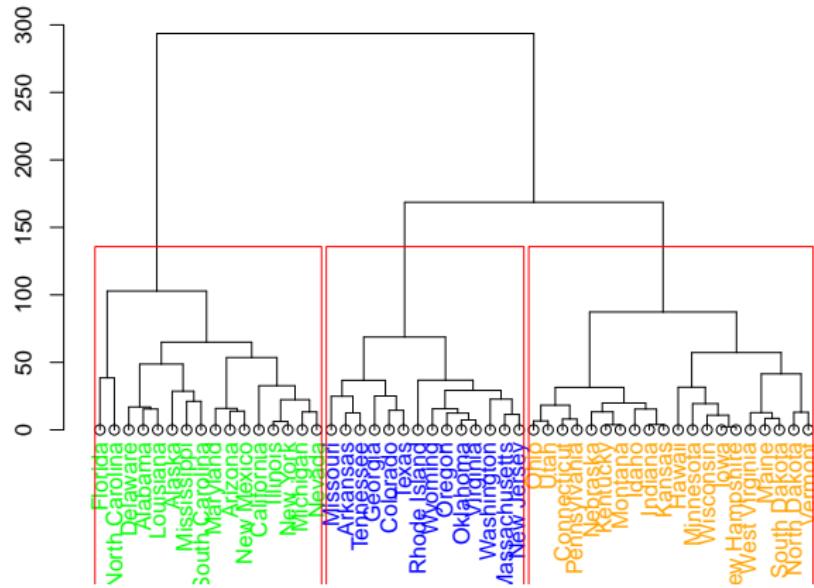
USAReest and the linkage methods



Cutting the dendrogram

```
> hc.complete <- hclust(dist(d), method = "complete")
# draw dendrogram with red borders around the 3 clusters
> rect.hclust(hc.complete, k=3, border="red")
```

USArrests data, hclust with complete linkage, k=3



Cutting the dendrogram

Cut the dendrogram at height h . The interpretation of h is

- complete linkage: for each instance \mathbf{x}_i , EVERY other instance \mathbf{x}_j in its cluster satisfies $d(\mathbf{x}_i, \mathbf{x}_j) \leq h$
- single linkage: for each instance \mathbf{x}_i , there is ANOTHER instance \mathbf{x}_j in its cluster satisfies $d(\mathbf{x}_i, \mathbf{x}_j) \leq h$
- average linkage: no interpretation

Agglomerative hierarchical methods

Stopping criteria

- **Distance criterion**

When the clusters are too far apart to be merged

- **Number criterion**

When there is sufficiently small number of clusters

Key takeaways

- Examine the data before diving into the building predictor.
- Spot issues with data range, units, data type, and missing or invalid values.
- Visualization gives a sense of data distribution and relationships among variables.
- Visualization helps answer questions about the data.
- The goal of clustering is to discover or draw out similarities among subsets of your data.
- Different units cause different distances and potentially different clusterings.
- Different clustering algorithms will give different results. Consider different approaches, with different numbers of clusters.
- Consider the results from different heuristics for estimating the best number of clusters and explore various numbers of clusters.

Summary of Lecture #2

Examination Requirements

- Methods for basic data exploration – plotting and summarizing
- Association between features
- Clustering algorithms: K-means, hierarchical agglomerative