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

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter</td>
<td>?</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Paul</td>
<td>2</td>
<td>5</td>
<td>?</td>
</tr>
<tr>
<td>Mary</td>
<td>2</td>
<td>4</td>
<td>?</td>
</tr>
</tbody>
</table>

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.
Basic statistics

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>number of votes</td>
<td>100,000</td>
</tr>
<tr>
<td>number of movies</td>
<td>1,682</td>
</tr>
<tr>
<td>number of users</td>
<td>943</td>
</tr>
</tbody>
</table>

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

• About users

<table>
<thead>
<tr>
<th></th>
<th>age</th>
<th>gender</th>
<th>occupation</th>
<th>zip code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter</td>
<td>19</td>
<td>M</td>
<td>student</td>
<td>58644</td>
</tr>
<tr>
<td>Mary</td>
<td>50</td>
<td>F</td>
<td>healthcare</td>
<td>60657</td>
</tr>
</tbody>
</table>

• About movies

<table>
<thead>
<tr>
<th>title</th>
<th>action</th>
<th>...</th>
<th>IMDb rating</th>
<th>director</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toy Story</td>
<td>0</td>
<td>...</td>
<td>8.3</td>
<td>John Lasseter</td>
</tr>
<tr>
<td>Some Like It Hot</td>
<td>0</td>
<td>...</td>
<td>8.3</td>
<td>Billy Wilder</td>
</tr>
<tr>
<td>Star Wars</td>
<td>1</td>
<td>...</td>
<td>8.7</td>
<td>George Lucas</td>
</tr>
</tbody>
</table>
## MOV – Available data

### Data representation

<table>
<thead>
<tr>
<th>vote</th>
<th>MOVIE</th>
<th>USER</th>
<th>RATING</th>
<th>TIMESTAMP</th>
<th>5-8</th>
<th>9-33</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>1997-09-23 00:02:38</td>
<td>24 M technician</td>
<td>Toy Story (1995)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100,000</td>
<td>1682</td>
<td>916</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

See the feature description `mov.pdf` at our course webpage.
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
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
We focus on

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

We recommend you to read this overview of mathematical foundations needed for our course.
Frequency tables display the frequency of categorical feature values.

```r
# 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)**

![Barplot showing number of users and votes by gender](image-url)
Methods for basic data exploration

**Histograms** visualize distribution of feature values.

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

```r
# 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)

![Histogram](histogram−votes.R)

```
votes
frequency
0 200 400 600
0 100 200 300
```
Methods for basic data exploration

Cumulative histograms visualize cumulative frequencies.
**Contingency tables** display the frequency of values for combination of two categorical features.

```r
> # 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

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>3</td>
<td>4</td>
<td>23</td>
<td>44</td>
<td>77</td>
<td>151</td>
</tr>
<tr>
<td>M</td>
<td>6</td>
<td>12</td>
<td>34</td>
<td>132</td>
<td>248</td>
<td>432</td>
</tr>
<tr>
<td>Sum</td>
<td>9</td>
<td>16</td>
<td>57</td>
<td>176</td>
<td>325</td>
<td>583</td>
</tr>
</tbody>
</table>

> round(prop.table(ct),3) # prop.table generates proportions

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>0.005</td>
<td>0.007</td>
<td>0.039</td>
<td>0.075</td>
<td>0.132</td>
</tr>
<tr>
<td>M</td>
<td>0.010</td>
<td>0.021</td>
<td>0.058</td>
<td>0.226</td>
<td>0.425</td>
</tr>
</tbody>
</table>
```
Mosaic plots visualize contingency tables.

Votes for Star Wars

Mosaic plot (mosaic-plot.R)
Methods for basic data exploration

Measures of center and variation

```r
> 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.

Box plot for a sample of 20 instances:
Box-and-whiskers plots

- the sample of 122 ratings
  1  2  3  4  5
  2  7 35 45 33
- median($x$) = 4
- $\bar{x}$ = 3.82
- $sd(x) = 0.95$
- 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 $1.5 \times (Q_3 - Q_1)$ above $Q_3$ or below $Q_1$. 

![Boxplot of average movie ratings and standard deviation](boxplot-movie-rating-mean-sd.R)
**Methods for basic data exploration**

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

```r
> 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 ...
```
Scatter plots display values of two numerical features.
**Covariance** \( \text{cov}(X, Y) \) is a statistical 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 \( \rightarrow \) correlation coefficient
Pearson correlation coefficient is a measure of the linear relationship between two variables

- **For a population**

\[-1 \leq \rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sqrt{\sigma_X \sigma_Y}} \leq +1\]

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

- **For a sample**

\[-1 \leq r_{X,Y} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{s_xs_y} \leq +1\]
### Association between feature and target attribute

#### Numerical variables

<table>
<thead>
<tr>
<th>Feature Relationship</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter’s rating, imdb_rating</td>
<td>0.51</td>
</tr>
<tr>
<td>Paul’s rating, imdb_rating</td>
<td>0.44</td>
</tr>
<tr>
<td>Mary’s rating, imdb_rating</td>
<td>0.37</td>
</tr>
<tr>
<td>Peter’s rating, Mary’s rating</td>
<td>0.29</td>
</tr>
<tr>
<td>Peter’s rating, Paul’s rating</td>
<td>0.29</td>
</tr>
<tr>
<td>Paul’s rating, Mary’s rating</td>
<td>0.24</td>
</tr>
</tbody>
</table>
Association between feature and target attribute
Categorical variables

**Pearson’s $\chi^2$ 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 $\chi^2$ 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 $\to 1$
- no correlation if $\to 0$
Association between feature and target attribute

Categorical variables

RATING is a target attribute

<table>
<thead>
<tr>
<th>Feature</th>
<th>Pearson Contingency Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>0.0</td>
</tr>
<tr>
<td>gender</td>
<td>0.1</td>
</tr>
<tr>
<td>occupation</td>
<td>0.2</td>
</tr>
<tr>
<td>zip</td>
<td>0.3</td>
</tr>
<tr>
<td>action</td>
<td>0.4</td>
</tr>
<tr>
<td>adventure</td>
<td>0.5</td>
</tr>
<tr>
<td>animation</td>
<td>0.6</td>
</tr>
<tr>
<td>children</td>
<td>0.7</td>
</tr>
<tr>
<td>comedy</td>
<td>0.8</td>
</tr>
<tr>
<td>crime</td>
<td>0.9</td>
</tr>
<tr>
<td>documentary</td>
<td>1.0</td>
</tr>
<tr>
<td>drama</td>
<td></td>
</tr>
<tr>
<td>fantasy</td>
<td></td>
</tr>
<tr>
<td>filmnoir</td>
<td></td>
</tr>
<tr>
<td>horror</td>
<td></td>
</tr>
<tr>
<td>musical</td>
<td></td>
</tr>
<tr>
<td>mystery</td>
<td></td>
</tr>
<tr>
<td>romance</td>
<td></td>
</tr>
<tr>
<td>scifi</td>
<td></td>
</tr>
<tr>
<td>thriller</td>
<td></td>
</tr>
<tr>
<td>war</td>
<td></td>
</tr>
<tr>
<td>western</td>
<td></td>
</tr>
<tr>
<td>directors</td>
<td></td>
</tr>
<tr>
<td>writers</td>
<td></td>
</tr>
<tr>
<td>stars</td>
<td></td>
</tr>
</tbody>
</table>
Association between features
Categorical variables

Pearson contingency coefficient

age

(pearson-contingency-coefficient-mov.R)

gender

occupation

stars
Feature frequency of binary features

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

where $A_j$ is the $j$-th feature (binary), $x_i$ is the feature vector of the $i$-th instance, and $x_i^j$ is the value of $A_j$ in $x_i$. 

Analyzing values
Feature frequency
Analyzing values
Feature frequency – VPR data (cry)

```r
> 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  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  2  1  1  1  1  1  1  1  1
          247 248 249
     1  1  2
```
Analyzing values
Feature frequency – VPR data (cry)

Feature frequency for 'cry'
(feature-frequency-cry.R)
Filter out uneffective features from the CRY data

```r
> examples <- read.csv("cry.development.csv", sep="	")
> 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, function(x) sum(x) == 0)]
> ## remove features with 1s only
> c.2 <- c.1[, !lapply(c.1, function(x) sum(x) == 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
```
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)

```r
> e <- apply(c, 2, entropy)
> table(sort(round(e,2)))
```

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0.04</th>
<th>0.07</th>
<th>0.09</th>
<th>0.12</th>
<th>0.14</th>
<th>0.16</th>
<th>0.18</th>
<th>0.2</th>
<th>0.22</th>
<th>0.24</th>
<th>0.28</th>
<th>0.31</th>
<th>0.33</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>181</td>
<td>49</td>
<td>27</td>
<td>13</td>
<td>9</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0.34</th>
<th>0.4</th>
<th>0.42</th>
<th>0.46</th>
<th>0.47</th>
<th>0.48</th>
<th>0.51</th>
<th>0.52</th>
<th>0.53</th>
<th>0.54</th>
<th>0.55</th>
<th>0.56</th>
<th>0.57</th>
<th>0.58</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0.62</th>
<th>0.64</th>
<th>0.65</th>
<th>0.69</th>
<th>0.71</th>
<th>0.73</th>
<th>0.76</th>
<th>0.82</th>
<th>0.83</th>
<th>0.85</th>
<th>0.88</th>
<th>0.89</th>
<th>0.91</th>
<th>0.94</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0.95</th>
<th>0.97</th>
<th>0.99</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
Analyzing values
Entropy – VPR data (cry)

Entropy for 'cry'
(entropy–cry.R)
Association between feature and target attribute

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)))
Association between feature and target attribute
Conditional entropy – VPR data (cry)

Conditional entropy for 'cry'
(entropy–cry.R)
Association between feature and target attribute
Pearson contingency coefficient – VPR data (cry)

Pearson contingency coefficient for 'cry'
(pearson-contingency-coefficient-vpr.R)
Clustering
Supervised vs. Unsupervised learning

Supervised learning

\[ Data = \{ \langle x, y \rangle : x \in X, y \in Y \} \]

Unsupervised learning

\[ Data = \{ x : x \in X \} \]
Clustering finds homogenous subgroups among the instances in the unlabeled data.
Clustering

Discovering structure

The most common criteria

- **Homogenity**
  Objects within a same cluster should be similar each other

- **Separation**
  Objects in different clusters should be dissimilar from each other
Clustering algorithms

Hierarchical
- Single linkage
- Average linkage
- Complete linkage

Partitional
- Square error
- Graph theoretic
- Mixture resolving
- Mode seeking
- K-means
- Expectation Maximization

Credits: (Kononenko, Kukar, 2007)
The most common one

- **Cosine similarity**

\[
\text{sim}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{u}|| \cdot ||\mathbf{v}||}
\]

(Euclidean dot product \( \mathbf{u} \cdot \mathbf{v} = ||\mathbf{u}|| \cdot ||\mathbf{v}|| \cos \theta \))
Dissimilarity can be thought of as distance. The most common ones:

- **Euclidean distance** – continuous features
  \[
  d(u, v) = \sqrt{\sum_{i=1}^{m} (u_i - v_i)^2}
  \]

- **Manhattan distance** – continuous features
  \[
  d(u, v) = \sum_{i=1}^{m} |u_i - v_i|
  \]

- **Hamming distance** – categorical features
  \[
  d(u, v) = \sum_{i=1}^{m} (u_i \neq v_i)
  \]
Clustering algorithms

Notation

- Data = \{x_1, x_2, \ldots, x_n\}
- A set of k clusters C = \{C_1, C_2, \ldots, C_k\} containing the indices of the instances
- C_1 \cup \cdots \cup C_k = \{1, 2, \ldots, n\}
- C_j \cap C_i = \emptyset, \forall i \neq j
- i-th cluster centroid \( \mu(C_i) = \frac{1}{|C_i|} \sum_{j \in C_i} x_j \)
Clustering algorithms
Loss functions

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

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

(originally $L(C_i) = \frac{1}{|C_i|} \sum_{l, j \in C_i} d(x_l, x_j)^2$)

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

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

The most common choice of $d$ involves Euclidean distance.
argmin_{C_1, \ldots, C_k} L(C_1, \ldots, C_k) = \argmin_{C_1, \ldots, C_k} \sum_{i=1}^{k} L(C_i) \quad (1)
K-means algorithm

1. Create clusters $C_1^0, \ldots, C_k^0$
   - 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 $C_i^t$, $i = 1, \ldots, k$ do
      $$\mu(C_i^t) = \frac{1}{|C_i^t|} \sum_{j \in C_i^t} x_j$$
   b) data assignment: for all clusters $C_i^t$, $i = 1, \ldots, k$ do
      $$C_{i}^{t+1} = \{ j; d(x_j, \mu(C_i^t))^2 \leq d(x_j, \mu(C_{l}^t))^2, \forall l \neq i \}$$

Stopping criteria: no data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached
Dataset USArrests from the base R distribution

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

```r
> 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

```r
> 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)] # Assault, Rape
> 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-maps algorithm with the USArrests data

K–means, K = 3, nstart = 20

Rape

Assault

States plotted on the graph include Alabama, Alaska, Arizona, Arkansas, and so on.
K-means algorithm with the USArrests data

Multiple initial cluster assignments (nstart)

```r
> 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

![Graph showing the within groups sum of squares versus the number of clusters. The curve shows a steep decrease initially, followed by a much slower decrease, indicating the elbow point where adding more clusters does not significantly decrease the within groups sum of squares.](image-url)
Remarks

- The results depend on the initial clusters. The standard solution is to try a number of different starting points (see \texttt{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**
is a rooted binary tree where

• the root node represents an input data set \( \text{Data}, |\text{Data}| = n \)

• 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 $C_i := \{x_i\}$ end

2. $C := \{C_1, C_2, \ldots, C_n\}$

3. $j := n + 1$

4. while $|C| > 1$
   1. $(C_{n_1}, C_{n_2}) := \arg\max_{C_u, C_v \in C \times C} \text{sim}(C_u, C_v)$
   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** $d(x_i, x_j)$

• **dissimilarity between clusters** $d(C_u, C_v)$
  • then 4.1 in the algorithm is

$$(C_{n1}, C_{n2}) := \arg\min_{C_u, C_v \in C \times C} d(C_u, C_v)$$

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

\[ d(C_u, C_v) = \min_{x_i \in C_u, x_j \in C_v} d(x_i, x_j) \]
Dissimilarity between clusters
Complete linkage clustering

The maximum dissimilarity between instances of each cluster

\[ d(C_u, C_v) = \max_{x_i \in C_u, x_j \in C_v} d(x_i, x_j) \]
The mean dissimilarity between instances of each cluster

\[ d(C_u, C_v) = \frac{1}{|C_u| |C_v|} \sum_{x_i \in C_u} \sum_{x_j \in C_v} d(x_i, x_j) \]
Linkage methods - Illustration

```r
> round(dist(m),2)
     1  2  3  4  5
1 2.69
2 2.52 1.58
3 4.28 2.42 1.77
4 2.51 1.12 2.50 3.53
5 2.81 2.15 0.58 1.72 3.08
```
USA Arrest and the linkage methods

Complete

Average

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

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

- complete linkage: for each instance $x_i$, EVERY other instance $x_j$ in its cluster satisfies $d(x_i, x_j) \leq h$

- single linkage: for each instance $x_i$, there is ANOTHER instance $x_j$ in its cluster satisfies $d(x_i, x_j) \leq h$

- 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