# Methods to Learn: Last Lecture

<table>
<thead>
<tr>
<th></th>
<th>Vector Data</th>
<th>Set Data</th>
<th>Sequence Data</th>
<th>Text Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classification</strong></td>
<td>Logistic Regression; Decision Tree; KNN SVM; NN</td>
<td></td>
<td></td>
<td>Naïve Bayes for Text</td>
</tr>
<tr>
<td><strong>Clustering</strong></td>
<td>K-means; hierarchical clustering; DBSCAN; DBSCAN; Mixture Models</td>
<td></td>
<td></td>
<td>PLSA</td>
</tr>
<tr>
<td><strong>Prediction</strong></td>
<td>Linear Regression GLM*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Frequent Pattern Mining</strong></td>
<td>Apriori; FP growth</td>
<td></td>
<td>GSP; PrefixSpan</td>
<td></td>
</tr>
<tr>
<td><strong>Similarity Search</strong></td>
<td></td>
<td></td>
<td></td>
<td>DTW</td>
</tr>
</tbody>
</table>
# Methods to Learn

<table>
<thead>
<tr>
<th></th>
<th>Vector Data</th>
<th>Set Data</th>
<th>Sequence Data</th>
<th>Text Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classification</strong></td>
<td>Logistic Regression; Decision Tree;</td>
<td></td>
<td></td>
<td>Naïve Bayes for Text</td>
</tr>
<tr>
<td></td>
<td>KNN; SVM; NN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Clustering</strong></td>
<td>K-means; hierarchical clustering;</td>
<td></td>
<td></td>
<td>PLSA</td>
</tr>
<tr>
<td></td>
<td>DBSCAN; Mixture Models</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Prediction</strong></td>
<td>Linear Regression; GLM*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Frequent Pattern</strong></td>
<td>Apriori; FP growth</td>
<td></td>
<td>GSP; PrefixSpan</td>
<td></td>
</tr>
<tr>
<td><strong>Mining</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Similarity Search</strong></td>
<td></td>
<td></td>
<td>DTW</td>
<td></td>
</tr>
</tbody>
</table>
K Nearest Neighbor

• Introduction
• kNN
• Similarity and Dissimilarity
• Summary
Lazy vs. Eager Learning

- **Lazy vs. eager learning**
  - **Lazy learning** (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
  - **Eager learning** (the above discussed methods): Given a set of training tuples, constructs a classification model before receiving new (e.g., test) data to classify

- Lazy: less time in training but more time in predicting
- **Accuracy**
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form an implicit global approximation to the target function
  - Eager: must commit to a single hypothesis that covers the entire instance space
Lazy Learner: Instance-Based Methods

- Instance-based learning:
  - Store training examples and delay the processing (“lazy evaluation”) until a new instance must be classified

- Typical approaches
  - *k*-nearest neighbor approach
    - Instances represented as points in, e.g., a Euclidean space.
  - *Locally weighted regression*
    - Constructs local approximation
K Nearest Neighbor

• Introduction
• kNN
• Similarity and Dissimilarity
• Summary
The $k$-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space
- The nearest neighbor are defined in terms of a distance measure, $\text{dist}(X_1, X_2)$
- Target function could be discrete- or real-valued
- For discrete-valued, $k$-NN returns the most common value among the $k$ training examples nearest to $x_q$
- Voronoi diagram: the decision surface induced by 1-NN for a typical set of training examples
**kNN Example**

\[ X = (\text{length, lightness}) \]

Classes = \{salmon, sea bass, eel\}

Task: Identify fish given its (length, lightness)

\[ K = 5 : 3 \text{ sea bass, 1 eel, 1 salmon } \Rightarrow \text{ sea bass} \]
**kNN Algorithm Summary**

- Choose K
- For a given new instance $X_{new}$, find K closest training points w.r.t. a distance measure
- Classify $X_{new} = \text{majority vote among the K points}$
Discussion on the $k$-NN Algorithm

- $k$-NN for real-valued prediction for a given unknown tuple
  - Returns the mean values of the $k$ nearest neighbors
- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the $k$ neighbors according to their distance to the query $x_q$
    - **Give greater weight to closer neighbors**
    - $y_q = \frac{\sum w_i y_i}{\sum w_i}$, where $x_i$’s are $x_q$’s nearest neighbors
    - $w_i = \frac{1}{d(x_q, x_i)^2}$
    - $w_i = \exp(-d(x_q, x_i)^2 / 2\sigma^2)$
- Robust to noisy data by averaging $k$-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
  - To overcome it, axes stretch or elimination of the least relevant attributes
Selection of k for kNN

- The number of neighbors k
  - Small k: overfitting (high var., low bias)
  - Big k: bringing too many irrelevant points (high bias, low var.)

- More discussions:
  [Link to Bias-Variance discussion](http://scott.fortmann-roe.com/docs/BiasVariance.html)
K Nearest Neighbor

• Introduction
• kNN
• Similarity and Dissimilarity
• Summary
Similarity and Dissimilarity

- **Similarity**
  - Numerical measure of how alike two data objects are
  - Value is higher when objects are more alike
  - Often falls in the range $[0, 1]$  

- **Dissimilarity (e.g., distance)**
  - Numerical measure of how different two data objects are
  - Lower when objects are more alike
  - Minimum dissimilarity is often 0
  - Upper limit varies

- **Proximity** refers to a similarity or dissimilarity
Data Matrix and Dissimilarity Matrix

- **Data matrix**
  - n data points with p dimensions
  - Two modes

- **Dissimilarity matrix**
  - n data points, but registers only the distance
  - A triangular matrix
  - Single mode

\[
\begin{pmatrix}
  x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\
  \vdots & \ddots & \vdots & \ddots & \vdots \\
x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\
  \vdots & \ddots & \vdots & \ddots & \vdots \\
x_{n1} & \cdots & x_{nf} & \cdots & x_{np}
\end{pmatrix}
\]

\[
\begin{pmatrix}
  0 & & & & \\
  d(2,1) & 0 & & & \\
  d(3,1) & d(3,2) & 0 & & \\
  \vdots & \vdots & \vdots & \ddots & \\
  d(n,1) & d(n,2) & \cdots & \cdots & 0
\end{pmatrix}
\]
Proximity Measure for Nominal Attributes

- Can take 2 or more states, e.g., red, yellow, blue, green (generalization of a binary attribute)
- **Method 1**: Simple matching
  - \( m \): # of matches, \( p \): total # of variables
  - \( d(i, j) = \frac{p - m}{p} \)
- **Method 2**: Use a large number of binary attributes
  - creating a new binary attribute for each of the \( M \) nominal states
Proximity Measure for Binary Attributes

• A contingency table for binary data

<table>
<thead>
<tr>
<th></th>
<th>Object i</th>
<th>Object j</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>q</td>
<td>r</td>
<td>q + r</td>
</tr>
<tr>
<td>0</td>
<td>s</td>
<td>t</td>
<td>s + t</td>
</tr>
<tr>
<td>sum</td>
<td>q + s</td>
<td>r + t</td>
<td>p</td>
</tr>
</tbody>
</table>

• Distance measure for symmetric binary variables:

\[ d(i, j) = \frac{r + s}{q + r + s + t} \]

• Distance measure for asymmetric binary variables:

\[ d(i, j) = \frac{r + s}{q + r + s} \]

• Jaccard coefficient (similarity measure for asymmetric binary variables):

\[ \text{sim}_\text{Jaccard}(i, j) = \frac{q}{q + r + s} \]
Dissimilarity between Binary Variables

Example

<table>
<thead>
<tr>
<th>Name</th>
<th>Gender</th>
<th>Fever</th>
<th>Cough</th>
<th>Test-1</th>
<th>Test-2</th>
<th>Test-3</th>
<th>Test-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jack</td>
<td>M</td>
<td>Y</td>
<td>N</td>
<td>P</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Mary</td>
<td>F</td>
<td>Y</td>
<td>N</td>
<td>P</td>
<td>N</td>
<td>P</td>
<td>N</td>
</tr>
<tr>
<td>Jim</td>
<td>M</td>
<td>Y</td>
<td>P</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

- Gender is a symmetric attribute
- The remaining attributes are asymmetric binary
- Let the values Y and P be 1, and the value N 0

\[
\begin{align*}
\text{d(} & \text{jack, mary) } = \frac{0 + 1}{2 + 0 + 1} = 0.33 \\
\text{d(} & \text{jack, jim) } = \frac{1 + 1}{1 + 1 + 1} = 0.67 \\
\text{d(} & \text{jim, mary) } = \frac{1 + 2}{1 + 1 + 2} = 0.75
\end{align*}
\]
Standardizing Numeric Data

- **Z-score:**
  \[ z = \frac{x - \mu}{\sigma} \]
  - \(X\): raw score to be standardized, \(\mu\): mean of the population, \(\sigma\): standard deviation
  - the distance between the raw score and the population mean in units of the standard deviation
  - negative when the raw score is below the mean, “+” when above

- An alternative way: Calculate the mean absolute deviation

  \[ s_f = \frac{1}{n} \left( |x_{1f} - m_f| + |x_{2f} - m_f| + \ldots + |x_{nf} - m_f| \right) \]

  where

  \[ m_f = \frac{1}{n} (x_{1f} + x_{2f} + \ldots + x_{nf}) \]

  - **standardized measure (z-score):**
    \[ z_{if} = \frac{x_{if} - m_f}{s_f} \]

- Using mean absolute deviation is more robust than using standard deviation
Example:
Data Matrix and Dissimilarity Matrix

Data Matrix

<table>
<thead>
<tr>
<th>point</th>
<th>attribute1</th>
<th>attribute2</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>x2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>x3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>x4</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Dissimilarity Matrix
(with Euclidean Distance)

<table>
<thead>
<tr>
<th></th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>3.61</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td>2.24</td>
<td>5.1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td>4.24</td>
<td>1</td>
<td>5.39</td>
<td>0</td>
</tr>
</tbody>
</table>
**Distance on Numeric Data: Minkowski Distance**

- **Minkowski distance**: A popular distance measure

\[ d(i, j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h} \]

where \( i = (x_{i1}, x_{i2}, \ldots, x_{ip}) \) and \( j = (x_{j1}, x_{j2}, \ldots, x_{jp}) \) are two \( p \)-dimensional data objects, and \( h \) is the order (the distance so defined is also called L-\( h \) norm)

- **Properties**
  - \( d(i, j) > 0 \) if \( i \neq j \), and \( d(i, i) = 0 \) (Positive definiteness)
  - \( d(i, j) = d(j, i) \) (Symmetry)
  - \( d(i, j) \leq d(i, k) + d(k, j) \) (Triangle Inequality)

- A distance that satisfies these properties is a **metric**
Special Cases of Minkowski Distance

- $h = 1$: **Manhattan** (city block, $L_1$ norm) distance
  - E.g., the Hamming distance: the number of bits that are different between two binary vectors
    \[
    d(i, j) = |x_{i_1} - x_{j_1}| + |x_{i_2} - x_{j_2}| + \ldots + |x_{i_p} - x_{j_p}|
    \]

- $h = 2$: (L$_2$ norm) **Euclidean** distance
  \[
  d(i, j) = \sqrt{|x_{i_1} - x_{j_1}|^2 + |x_{i_2} - x_{j_2}|^2 + \ldots + |x_{i_p} - x_{j_p}|^2}
  \]

- $h \to \infty$. “supremum” (**$L_{\text{max}}$** norm, $L_{\infty}$ norm) distance.
  - This is the maximum difference between any component (attribute) of the vectors
    \[
    d(i, j) = \lim_{h \to \infty} \left( \sum_{f=1}^{p} |x_{i_f} - x_{j_f}|^h \right)^{\frac{1}{h}} = \max_{f} |x_{i_f} - x_{j_f}|
    \]
Example: Minkowski Distance

Dissimilarity Matrices

**Manhattan (L₁)**

<table>
<thead>
<tr>
<th>L</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>5</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td>3</td>
<td>6</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td>6</td>
<td>1</td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

**Euclidean (L₂)**

<table>
<thead>
<tr>
<th>L₂</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>3.61</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td>2.24</td>
<td>5.1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td>4.24</td>
<td>1</td>
<td>5.39</td>
<td>0</td>
</tr>
</tbody>
</table>

**Supremum**

<table>
<thead>
<tr>
<th>L₂</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>3</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td>2</td>
<td>5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>
Ordinal Variables

- Order is important, e.g., rank
- Can be treated like interval-scaled
  - replace $x_{if}$ by their rank $r_{if} \in \{1, \ldots, M_f\}$
  - map the range of each variable onto $[0, 1]$ by replacing $i$-th object in the $f$-th variable by
    \[z_{if} = \frac{r_{if} - 1}{M_f - 1}\]
  - compute the dissimilarity using methods for interval-scaled variables
Attributes of Mixed Type

• A database may contain all attribute types
  • Nominal, symmetric binary, asymmetric binary, numeric, ordinal
• One may use a weighted formula to combine their effects

\[ d(i, j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}} \]

• \( f \) is binary or nominal:
  \( d_{ij}^{(f)} = 0 \) if \( x_{if} = x_{jf} \), or \( d_{ij}^{(f)} = 1 \) otherwise
• \( f \) is numeric: use the normalized distance
• \( f \) is ordinal
  • Compute ranks \( r_{if} \) and \( z_{if} = \frac{r_{if} - 1}{M_f - 1} \)
  • Treat \( z_{if} \) as interval-scaled
A **document** can be represented by thousands of attributes, each recording the *frequency* of a particular word (such as keywords) or phrase in the document.

<table>
<thead>
<tr>
<th>Document</th>
<th>team</th>
<th>coach</th>
<th>hockey</th>
<th>baseball</th>
<th>soccer</th>
<th>penalty</th>
<th>score</th>
<th>win</th>
<th>loss</th>
<th>season</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document1</td>
<td>5</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Document2</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Document3</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Document4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

- Other vector objects: gene features in micro-arrays, ...
- Applications: information retrieval, biologic taxonomy, gene feature mapping, ...
- Cosine measure: If $d_1$ and $d_2$ are two vectors (e.g., term-frequency vectors), then
  \[
  \cos(d_1, d_2) = \frac{(d_1 \bullet d_2)}{||d_1|| \cdot ||d_2||},
  \]
  where $\bullet$ indicates vector dot product, $||d||$: the length of vector $d$
Example: Cosine Similarity

- \( \cos(d_1, d_2) = \frac{(d_1 \cdot d_2)}{||d_1|| \times ||d_2||} \),
  where \( \cdot \) indicates vector dot product, \( | |d|| \): the length of vector \( d \)

- Ex: Find the similarity between documents 1 and 2.

\[
\begin{align*}
  d_1 &= (5, 0, 3, 0, 2, 0, 0, 2, 0, 0) \\
  d_2 &= (3, 0, 2, 0, 1, 1, 0, 1, 0, 1)
\end{align*}
\]

\[
\begin{align*}
  d_1 \cdot d_2 &= 5 \times 3 + 0 \times 0 + 3 \times 2 + 0 \times 0 + 2 \times 1 + 0 \times 1 + 0 \times 1 + 2 \times 1 + 0 \times 0 + 0 \times 1 = 25 \\
  | |d_1|| &= (5^2 + 0^2 + 3^2 + 0^2 + 2^2 + 0^2 + 1^2 + 0^2 + 1^2 + 0^2 + 0^2 + 1^2)^{0.5} = (42)^{0.5} = 6.481 \\
  | |d_2|| &= (3^2 + 0^2 + 2^2 + 0^2 + 1^2 + 1^2 + 1^2 + 0^2 + 1^2 + 0^2 + 0^2 + 1^2)^{0.5} = (17)^{0.5} = 4.12 \\
  \cos(d_1, d_2) &= 0.94
\end{align*}
\]
K Nearest Neighbor

- Introduction
- kNN
- Similarity and Dissimilarity
- Summary
Summary

• Instance-Based Learning
  • Lazy learning vs. eager learning; K-nearest neighbor algorithm; Similarity / dissimilarity measures