Similarity and Clustering

The Process of Data Science

Dr. Amilcar Soares

Summer 2019
If two objects can be represented as feature vectors, then we can compute the distance between them.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Person A</th>
<th>Person B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>23</td>
<td>40</td>
</tr>
<tr>
<td>Years at current address</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Residential status</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>(1=Owner, 2=Renter, 3=Other)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Euclidean distance

\[
\text{Distance}(A, B) = \sqrt{(X_B - X_A)^2 + (Y_B - Y_A)^2}
\]
Euclidean distance

$$d(A, B) = \sqrt{(23 - 40)^2 + (2 - 10)^2 + (2 - 1)^2} = 18.8$$
Example: Whiskey analytics

1. **Color**: yellow, very pale, pale, pale gold, gold, old gold, full gold, amber, etc. (14 values)
2. **Nose**: aromatic, peaty, sweet, light, fresh, dry, grassy, etc. (12 values)
3. **Body**: soft, medium, full, round, smooth, light, firm, oily. (8 values)
4. **Palate**: full, dry, sherry, big, fruity, grassy, smoky, salty, etc. (15 values)
5. **Finish**: full, dry, warm, light, smooth, clean, fruity, grassy, smoky, etc. (19 values)

<table>
<thead>
<tr>
<th>Whiskey</th>
<th>Distance</th>
<th>Descriptors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bunnahabhain</td>
<td></td>
<td>gold; firm, med, light; sweet, fruit, clean; fresh, sea; full</td>
</tr>
<tr>
<td>Glenglassaugh</td>
<td>0.643</td>
<td>gold; firm, light, smooth; sweet, grass; fresh, grass</td>
</tr>
<tr>
<td>Tullibardine</td>
<td>0.647</td>
<td>gold; firm, med, smooth; sweet, fruit, full, grass, clean; sweet; big, arome, sweet</td>
</tr>
<tr>
<td>Ardbeg</td>
<td>0.667</td>
<td>sherry; firm, med, full, light; sweet; dry, peat, sea; salt</td>
</tr>
<tr>
<td>Bruichladdich</td>
<td>0.667</td>
<td>pale; firm, light, smooth; dry, sweet, smoke, clean; light; full</td>
</tr>
<tr>
<td>Glenmorangie</td>
<td>0.667</td>
<td>p.gold; med, oily, light; sweet, grass, spice; sweet, spicy, grass, sea, fresh; full, long</td>
</tr>
</tbody>
</table>
Example: Whiskey analytics

- Category values are not mutually exclusive
  - 68 binary features
    - $68 = 14 + 12 + 8 + 15 + 19$
- A vector representing a whiskey consists of 68 binary-valued elements.
A simple classifier that relies on a similarity measure defined over pairs of instances of the data
Nearest Neighbours
### Nearest Neighbours for Predictive Modeling

<table>
<thead>
<tr>
<th>Customer</th>
<th>Age</th>
<th>Income (1000s)</th>
<th>Cards</th>
<th>Response (target)</th>
<th>Distance from David</th>
</tr>
</thead>
<tbody>
<tr>
<td>David</td>
<td>37</td>
<td>50</td>
<td>2</td>
<td>?</td>
<td>0</td>
</tr>
<tr>
<td>John</td>
<td>35</td>
<td>35</td>
<td>3</td>
<td>Yes</td>
<td>(\sqrt{(35 - 37)^2 + (35 - 50)^2 + (3 - 2)^2} = 15.16)</td>
</tr>
<tr>
<td>Rachael</td>
<td>22</td>
<td>50</td>
<td>2</td>
<td>No</td>
<td>(\sqrt{(22 - 37)^2 + (50 - 50)^2 + (2 - 2)^2} = 15)</td>
</tr>
<tr>
<td>Ruth</td>
<td>63</td>
<td>200</td>
<td>1</td>
<td>No</td>
<td>(\sqrt{(63 - 37)^2 + (200 - 50)^2 + (1 - 2)^2} = 152.23)</td>
</tr>
<tr>
<td>Jefferson</td>
<td>59</td>
<td>170</td>
<td>1</td>
<td>No</td>
<td>(\sqrt{(59 - 37)^2 + (170 - 50)^2 + (1 - 2)^2} = 122)</td>
</tr>
<tr>
<td>Norah</td>
<td>25</td>
<td>40</td>
<td>4</td>
<td>Yes</td>
<td>(\sqrt{(25 - 37)^2 + (40 - 50)^2 + (4 - 2)^2} = 15.74)</td>
</tr>
</tbody>
</table>
How many neighbours and how much influence

$k$ Nearest Neighbors

- $k = ?$
- $k = 1 ?$
- $k = n ?$
Majority versus weighted scoring

Neighbours have different similarity to the instance we are trying to predict. Shouldn't this influence prediction?

<table>
<thead>
<tr>
<th>Name</th>
<th>Distance</th>
<th>Similarity weight</th>
<th>Contribution</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rachael</td>
<td>15.0</td>
<td>0.004444</td>
<td>0.344</td>
<td>No</td>
</tr>
<tr>
<td>John</td>
<td>15.2</td>
<td>0.004348</td>
<td>0.336</td>
<td>Yes</td>
</tr>
<tr>
<td>Norah</td>
<td>15.7</td>
<td>0.004032</td>
<td>0.312</td>
<td>Yes</td>
</tr>
<tr>
<td>Jefferson</td>
<td>122.0</td>
<td>0.000067</td>
<td>0.005</td>
<td>No</td>
</tr>
<tr>
<td>Ruth</td>
<td>152.2</td>
<td>0.000043</td>
<td>0.003</td>
<td>No</td>
</tr>
</tbody>
</table>
Geometry, Over-fitting and Complexity
1-NN
30-NN

Nearest-neighbor classification (k = 30)
kNN prediction observations

● Intelligibility
  ○ Easy to describe how a single instance is decided
  ○ Knowledge embedded in model not understandable

● Dimensionality & domain knowledge
  ○ Numeric attributes may have vastly different ranges
  ○ Attributes may be irrelevant to similarity
    ■ Feature selection (manual)
  ○ Tune similarity/distance function manually
kNN prediction observations

- Sensitivity to outliers
- Irregular concept boundaries
- Selecting $k$: a complexity parameter
  - Cross-validation or holdout testing to determine optimal $k$
- Computational cost
  - Training
  - Testing
  - Use Kd-trees, hashing
Similarity and Heterogeneous attributes

- Handling categorical attributes?
  - Binary: map to 0 and 1
  - Non-binary?
- Scaling of attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Person A</th>
<th>Person B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex</td>
<td>Male</td>
<td>Female</td>
</tr>
<tr>
<td>Age</td>
<td>23</td>
<td>40</td>
</tr>
<tr>
<td>Years at current address</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Residential status (1=Owner, 2=Renter, 3=Other)</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Income</td>
<td>50,000</td>
<td>90,000</td>
</tr>
</tbody>
</table>
Calculating scores from neighbours

- Majority vote classification
  \[ c(x) = \arg \max_{c \in \text{classes}} \text{score}(c, \text{neighbors}_k(x)) \]

  - Majority scoring function
    \[ \text{score}(c, N) = \sum_{y \in N} [\text{class}(y) = c] \]

  - \([\text{class}(y) = c]\) is one if statement is true, and zero otherwise
Calculating scores from neighbours

- Similarity-moderated classification
  \[
  \text{score}(c, N) = \sum_{y \in N} w(x, y) \times [\text{class}(y) = c]
  \]
  
  - Weight
    \[
    w(x, y) = \frac{1}{\text{dist}(x, y)^2}
    \]
Calculating scores from neighbours

- Similarity-moderated scoring (class probability estimation)
  
  \[
  p(c \mid x) = \frac{\sum_{y \in \text{neighbors}(x) : \text{class}(y) = c} w(x, y)}{\sum_{y \in \text{neighbors}(x)} w(x, y)}
  \]

- Similarity-moderated regression
  Where \( t(y) \) is the target value for \( y \)
  
  \[
  f(x) = \frac{\sum_{y \in \text{neighbors}(x)} w(x, y) \times t(y)}{\sum_{y \in \text{neighbors}(x)} w(x, y)}
  \]
Given a data set, form groups of instances that are similar within the same group and different across different group. Unlike prediction/classification, clustering is exploratory in nature.
Clustering

● Application of similarity
● Find groups of instances where
  ○ Instances within groups are similar
  ○ Instances in different groups are not so similar
Clustering around centroids
Clustering around centroids
K-means clustering
K-means clustering
K-means clustering

- Efficient (what is the computational cost per iteration)
- Choice of k is a challenge:
  - Engage the user
  - Clustering quality metrics
Estimating the number of clusters (Wikipedia) - Not in the midterm

- Rule of thumb $k \approx \sqrt{n/2}$
- The elbow method
  - Percent of variance explained (F-test)
    $F = \frac{\text{between-group variability}}{\text{within-group variability}}$
  - Between-group-variability $= \sum_{i=1}^{K} n_i (\bar{Y}_i - \bar{Y})^2 / (K - 1)$
    $n_i$: number of instances in cluster $i$
    $\bar{Y}_i$: sample mean of cluster $i$
    $\bar{Y}$: overall mean of the data, $K$: number of clusters
  - Within-group-variability $= \sum_{i=1}^{K} \sum_{j=1}^{n_i} (Y_{ij} - \bar{Y}_i)^2 / (N - K)$,
    $Y_{ij}$: jth instance in ith cluster
    $N$: overall number of instances
  - Not always clear where the elbow is
Clustering when only a similarity measure is available

- Similarity is the result of a complex process, not of vector distance
  - Instances have categorical and numerical attributes
  - Text document similarity that takes into account semantics
- Centroid cannot be defined
- K-medoids algorithm (Wikipedia):
  - Partitioning Around Medoids (PAM)
  - Python implementation

...
Case study: Clustering Business News Stories

- Thomson-Reuters news corpus
- Vocabulary: all words in the corpus except very rare and very common ones
  - Very rare: in fewer than $m$ documents (e.g. $m=5$)
  - Very common: in more than $n$ documents (e.g. $n=0.5$ of total number of docs)
- Each doc represented by a TFIDF vector
  - TFIDF = Term Frequency Inverse Document Frequency
- Cluster news stories into $k=9$ clusters using k-means.
- Inspect the clusters
  - Some are interesting and thematically coherent, others are not
  - Clustering is useful tool to uncover structure in the data we did not foresee
Cluster 2

Cluster 2. This cluster contains stories about Apple’s stock price movements, during and after each day of trading:

- Apple shares pare losses, still down 5 pct
- Apple rises 5 pct following strong results
- Apple shares rise on optimism over iPhone demand
- Apple shares decline ahead of Tuesday event
- Apple shares surge, investors like valuation
Cluster 4. This cluster contains various Apple announcements and releases. Superficially, these stories were similar, though the specific topics varied:

- Apple introduces iPhone "push" e-mail software
- Apple CFO sees 2nd-qtr margin of about 32 pct
- Apple says confident in 2008 iPhone sales goal
- Apple CFO expects flat gross margin in 3rd-quarter
- Apple to talk iPhone software plans on March 6
Cluster descriptions: what distinguishes a cluster?

● What differentiates a cluster from the other clusters?
  ○ Need an intelligible and concise description of the cluster
  ○ Highly dependent on the nature of the data and the domain

● Use a predictive (supervised) learning method with strong intelligibility
  ○ Decision trees as sets of rules
  ○ How do we apply the predictive method to the clustering result?
    ■ K-class prediction
    ■ One-vs-the rest prediction

● Visualize typical members of a cluster
  ○ Cluster centroid (not a real instance)
  ○ Instances closest to the centroid
Hierarchical clustering

- Agglomerative clustering is more common than divisive
- At each step:
  - Find the two closest data instances or clusters
  - Cluster them together
  - Repeat until all points are in a cluster
- Need to define
  - Distance metric: Euclidean, cosine, etc
  - Linkages: distances between clusters
  - Cutoff point to decide the final clusters
- They do not need to define the number of clusters in advance
- Returns how data can be grouped at different levels (if no cutoff is provided)
- Typically, more computationally expensive than flat algorithms
A simple clustering algorithm

- Given a distance between points
- **Initialize**: every point is a cluster
- **Iterate**:
  - Compute distances between all clusters (store for efficiency)
  - Merge two closest clusters
- Save both clustering and *sequence* of cluster operations
- “Dendrogram”
Iteration 1
Iteration 2
Iteration 3

- Builds up a sequence of clusters ("hierarchical")

- Algorithm complexity \(O(N^2)\)
Dendrogram

https://www.youtube.com/watch?v=XJ3194AmH40
DBSCAN

DBSCAN: Density Based Spatial Clustering of Applications with Noise

- Proposed by Ester, Kriegel, Sander, and Xu (KDD96)
- Relies on a density-based notion of cluster: A cluster is defined as a maximal set of density-connected points.
- Discovers clusters of arbitrary shape in spatial databases with noise
**DBSCAN**

*Basic Idea:*

Clusters are dense regions in the data space, separated by regions of lower object density

**Why Density-based clustering?**

Results of a $k$-medoid algorithm for $k=4$
DBSCAN

- **Intuition for the formalization of the basic idea**
  - For any point in a cluster, the local point density around that point has to exceed some threshold
  - The set of points from one cluster is spatially connected

- **Local point density at a point $p$ defined by two parameters**
  - $e$ – radius for the neighborhood of point $p$:
    - $N_e(p) := \{ q \in \text{data set } D \mid \text{dist}(p, q) \leq e \}$
  - $MinPts$ – minimum number of points in the given neighbourhood $N(p)$
Given $\varepsilon$ and $\text{MinPts}$, categorize the objects into three exclusive groups.

A point is a core point if it has more than a specified number of points (MinPts) within $\varepsilon$ These are points that are at the interior of a cluster.

A border point has fewer than MinPts within $\varepsilon$, but is in the neighborhood of a core point.

A noise point is any point that is not a core point nor a border point.

$\varepsilon = 1\text{unit}, \text{MinPts} = 5$
Input: The data set D
Parameter: $\varepsilon$, MinPts

For each object $p$ in D
  if $p$ is a core object and not processed then
    $C = \text{retrieve all objects density-reachable from } p$
    mark all objects in $C$ as processed
    report $C$ as a cluster
  else mark $p$ as outlier
end if

End For
MinPts = 5

1. Check the $\varepsilon$-neighborhood of $p$;
2. If $p$ has less than MinPts neighbors then mark $p$ as outlier and continue with the next object;
3. Otherwise mark $p$ as processed and put all the neighbors in cluster $C$.

1. Check the unprocessed objects in $C$;
2. If no core object, return $C$;
3. Otherwise, randomly pick up one core object $p_1$, mark $p_1$ as processed, and put all unprocessed neighbors of $p_1$ in cluster $C$. 

Class 7 - Similarity and Clustering
DBSCAN

● Advantages
  ○ Resistant to noise
  ○ Can handle clusters with different shapes and sizes

● Disadvantages
  ○ If the density varies, DBSCAN may not capture all clusters
  ○ The original version does not work well with high dimensional data

https://www.youtube.com/watch?v=sJQHz97sCZ0
DBSCAN

Original Points

(MinPts = 4, Eps = 9.75)

(MinPts = 4, Eps = 9.92)