

Similarity, Neighbors, and Clusters

1

Dr. Yi Long (Neal)

Most contents (text or images) of course slides are from the following textbook :
Provost, Foster, and Tom Fawcett. Data Science for Business: What you need to know about data mining and data-analytic thinking. " O'Reilly Media, Inc.", 2013

Outline

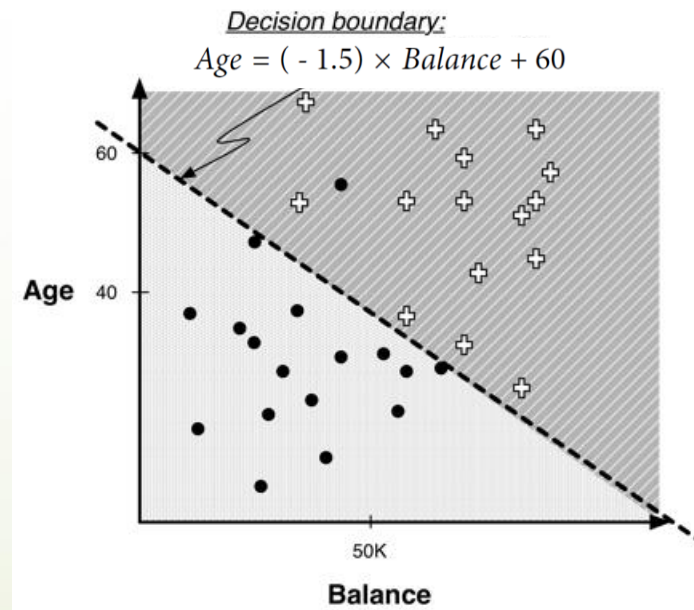
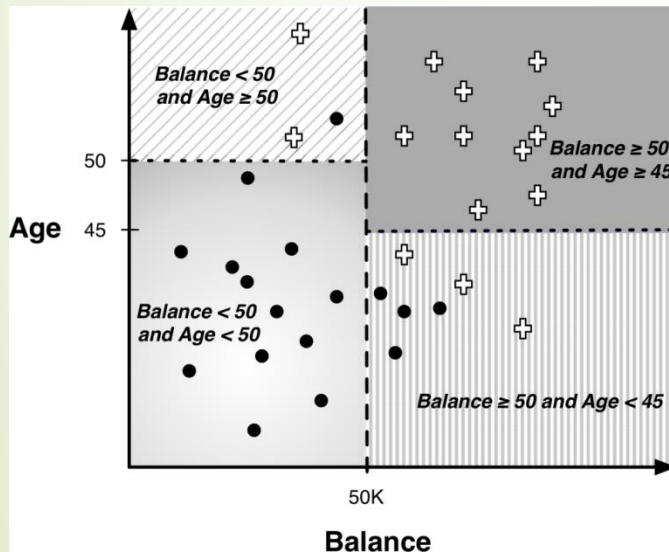
- Similarity and Distance
- Nearest-Neighbor Reasoning
- Clustering and Others
- Quiz

Applications of Similarity

- Similarity underlies many data science methods and solutions to business problems
- If two things (people, companies, products) are similar in some ways, they often share other characteristics as well
- Different sorts of business tasks involve reasoning from similar examples
 - ✓ Retrieve similar things: search engine
 - ✓ Classification or regression: K-Nearest-Neighbor
 - ✓ Clustering: grouping similar things tougher
 - ✓ Recommender system: Amazon --“People who like X also like Y”
 - ✓ Reasoning from similar cases beyond business applications: lawyers and doctors

Similarity under Segmentations

- Both classification trees and linear classifiers establish boundaries between regions of differing classifications.
- They have in common the view that instances sharing a common region in space should be similar (share the same label).
- What differs between classification trees and linear classifiers is how the regions/boundaries are represented and discovered.



Similarity and Distance

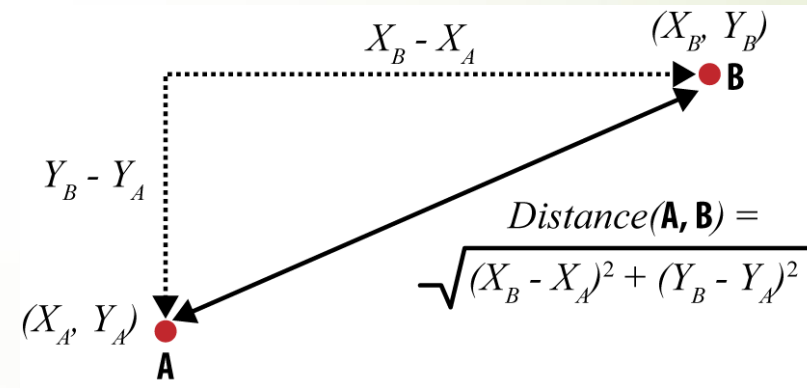
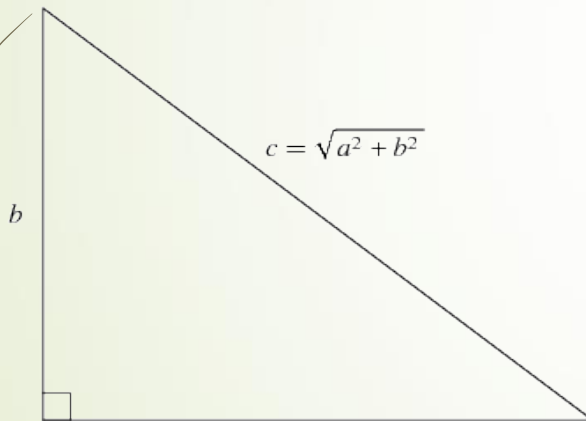
- Why not reason about the similarity or distance between objects directly?
- If we represent each object as a feature vector, then the closer two objects are in the space defined by the features, the more similar they are.
- For example :

Attribute	Person A	Person B
Age	23	40
Years at current address	2	10
Residential status (1=Owner, 2=Renter, 3=Other)	2	1

How to measure the similarity or distance between Person A and Person B ?

Euclidean Distance

- Euclidean distance is the "ordinary" straight-line distance between two points in Euclidean space.
- When each sample have two features, then each object is a point in a two-dimensional space. (Pythagorean theorem)



- When an object is described by n -dimensions features (d_1, d_2, \dots, d_n) , the general equation for Euclidean distance in n dimensions is shown as

$$\sqrt{(d_{1,A} - d_{1,B})^2 + (d_{2,A} - d_{2,B})^2 + \dots + (d_{n,A} - d_{n,B})^2}$$

Euclidean Distance (Example)

- Euclidean distance for Person A and Person B

$$\begin{aligned}d(A, B) &= \sqrt{(23 - 40)^2 + (2 - 10)^2 + (2 - 1)^2} \\ &\approx 18.8\end{aligned}$$

Attribute	Person A	Person B
Age	23	40
Years at current address	2	10
Residential status (1=Owner, 2=Renter, 3=Other)	2	1

- ✓ This distance is just a number—it has no units, and no meaningful interpretation.
- ✓ Distance is only really useful for comparing the similarity of one pair of instances to that of another pair, such as $d(A,B) \leq d(A,C)$

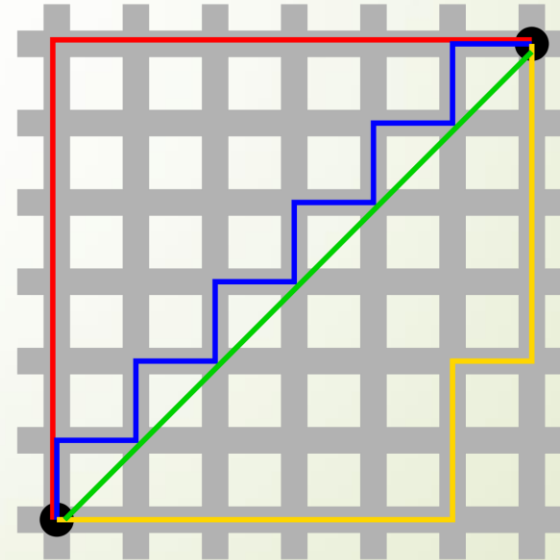
Other Distance Measure (1)

- Manhattan distance: L1-norm distance

$$d_{\text{Manhattan}}(\mathbf{X}, \mathbf{Y}) = \|\mathbf{X} - \mathbf{Y}\|_1 = |x_1 - y_1| + |x_2 - y_2| + \dots$$

✓ Application:

- ✓ Feature vector of integer values (number of students in different age group)
- ✓ Compressed sensing, ...

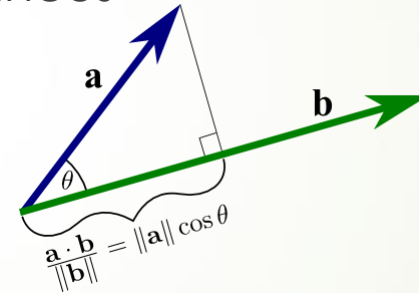


Other Distance Measure (2)

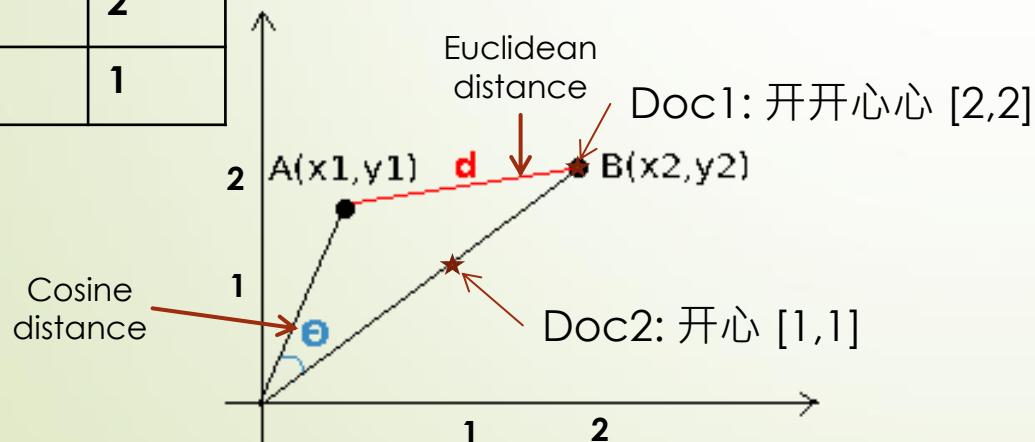
- Cosine distance: often used in text classification to measure the similarity of two documents
- ✓ Often used in text classification to measure the similarity of two documents
- ✓ Ignore differences in scale across instances

$$d_{\cosine}(X, Y) = 1 - \frac{X \cdot Y}{\|X\|_2 \cdot \|Y\|_2}$$

1



	开	心
Doc1	2	2
Doc2	1	1



$$A = \langle 7, 3, 2 \rangle \text{ and } B = \langle 2, 3, 0 \rangle$$



$$\begin{aligned}
 d_{\cosine}(A, B) &= 1 - \frac{\langle 7, 3, 2 \rangle \cdot \langle 2, 3, 0 \rangle}{\| \langle 7, 3, 2 \rangle \|_2 \cdot \| \langle 2, 3, 0 \rangle \|_2} \\
 &= 1 - \frac{7 \cdot 2 + 3 \cdot 3 + 2 \cdot 0}{\sqrt{49 + 9 + 4} \cdot \sqrt{4 + 9}} \\
 &= 1 - \frac{23}{28.4} \approx 0.19
 \end{aligned}$$

Other Distance Measure (3)

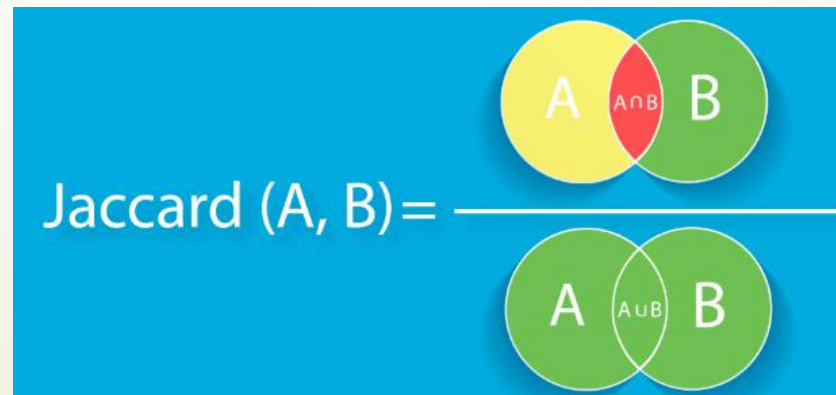
- Jaccard distance measure treats the two objects as *sets* of characteristics
 - ✓ The possession of a common characteristic between two item sets is important, but the common absence of a characteristic is **NOT**

$$d_{\text{jaccard}}(X, Y) = 1 - \frac{|X \cap Y|}{|X \cup Y|}$$

$X = (a, b, c)$

$Y = (b, c, d)$

$\rightarrow \text{Dist}(X, Y) = 1 - (2/4) = 0.5$



Other Distance/Similarity Measure (4)

- Hamming distance (equal length)
- Edit distance
- Longest common substring
- Longest common subsequence
- The Minkowski distance

Distance in Sklearn

<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.DistanceMetric.html>

Metrics intended for real-valued vector spaces:

identifier	class name	args	distance function
"euclidean"	EuclideanDistance	•	$\sqrt{\sum (x - y)^2}$
"manhattan"	ManhattanDistance	•	$\sum x - y $
"chebyshev"	ChebyshevDistance	•	$\max(x - y)$
"minkowski"	MinkowskiDistance	p	$\sum x - y ^p^{(1/p)}$
"wminkowski"	WMinkowskiDistance	p, w	$\sum w * (x - y) ^p^{(1/p)}$
"seuclidean"	SEuclideanDistance	V	$\sqrt{\sum (x - y)^2 / V}$
"mahalanobis"	MahalanobisDistance	V or VI	$\sqrt{(x - y)' V^{-1} (x - y)}$

Metrics intended for two-dimensional vector spaces: Note that the haversine distance metric requires data in the form of [latitude, longitude] and both inputs and outputs are in units of radians.

identifier	class name	distance function
"haversine"	HaversineDistance	$2 \arcsin(\sqrt{\sin^2(0.5 * dx) + \cos(x1) \cos(x2) \sin^2(0.5 * dy)})$

Metrics intended for integer-valued vector spaces: Though intended for integer-valued vectors, these are also valid metrics in the case of real-valued vectors.

identifier	class name	distance function
"hamming"	HammingDistance	$N_{\text{unequal}}(x, y) / N_{\text{tot}}$
"canberra"	CanberraDistance	$\sum x - y / (x + y)$
"braycurtis"	BrayCurtisDistance	$\sum x - y / (\sum x + \sum y)$

Outline


- Similarity and Distance
- Nearest-Neighbor Reasoning
- Clustering and Others
- Quiz

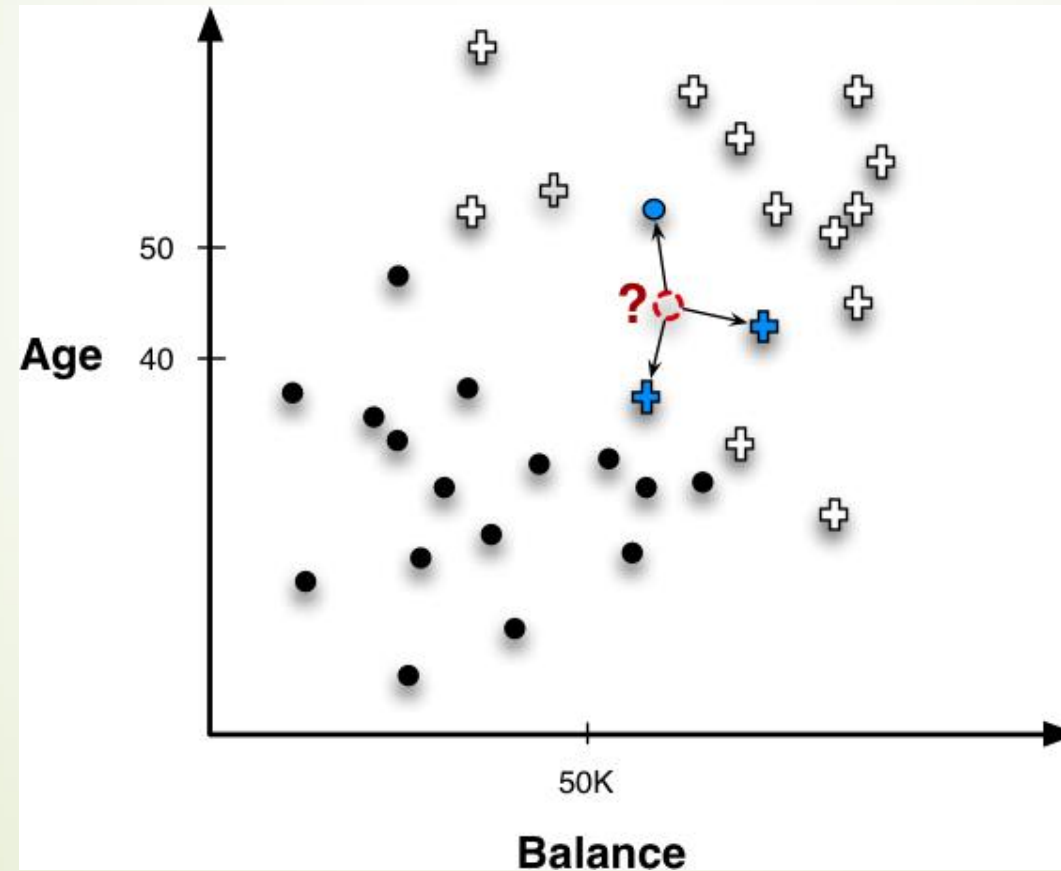
Nearest Neighbors for Predictive Modeling

- Simple procedure: given a new example whose target variable we want to predict, we scan through all the training examples and choose k examples that are the most similar to the new example.
- Then we predict the new example's target value, based on the k nearest neighbors' (known) target values.
- When we have need to make the prediction based on multiple nearest examples ($k > 1$), we can use **combining function** (like voting or averaging) to operating on these k neighbors' known target values
- How to choose k and **combining function**?

k-nearest neighbors algorithm (KNN/K-NN)

Example: KNN

When $k=3$ and combining function is done by voting , predictions is plus mark 



KNN for Classification

- When treating examples equally among the k nearest neighbors (as $\text{neighbors}_k(\mathbf{x})$ or \mathbf{N}) of feature vector \mathbf{x} , the combination function is majority votes

- ✓ The predicted class label $c(\mathbf{x})$ of a feature vector \mathbf{x} can be mathematically defined as follows

$$c(\mathbf{x}) = \arg \max_{c \in \text{classes}} \text{score}(c, \text{neighbors}_k(\mathbf{x})) \quad \text{with} \quad \text{score}(c, N) = \sum_{\mathbf{y} \in N} [\text{class}(\mathbf{y}) = c]$$

, and $[\text{class}(\mathbf{y}) = c]$ is 1 if $\text{class}(\mathbf{y}) = c$, otherwise 0

- ✓ We can further compute the probability of our classification as

$$\frac{\text{score}(c(\mathbf{x}), \text{neighbors}_k(\mathbf{x}))}{k}$$

- ✓ Usually, k is an odd number to break tie when selecting majority

Example: KNN for Classification

- Will David respond or not ($k=3$)?

Customer	Age	Income (1000s)	Cards	Response (target)	Distance from David
David	37	50	2	?	0
John	35	35	3	Yes	$\sqrt{(35 - 37)^2 + (35 - 50)^2 + (3 - 2)^2} = 15.16$
Rachael	22	50	2	No	$\sqrt{(22 - 37)^2 + (50 - 50)^2 + (2 - 2)^2} = 15$
Ruth	63	200	1	No	$\sqrt{(63 - 37)^2 + (200 - 50)^2 + (1 - 2)^2} = 152.23$
Jefferson	59	170	1	No	$\sqrt{(59 - 37)^2 + (170 - 50)^2 + (1 - 2)^2} = 122$
Norah	25	40	4	Yes	$\sqrt{(25 - 37)^2 + (40 - 50)^2 + (4 - 2)^2} = 15.74$

$classes = \{yes, no\}$

$score(yes, neighbors_3(\mathbf{x})) = 2$

$score(no, neighbors_3(\mathbf{x})) = 1$



$c(\mathbf{x}_{David}) = yes$

$p(yes|\mathbf{x}_{David}) = \frac{2}{3}$

- Predictions will change when $k=5$, but is it reasonable that Ruth and Rachel have equal weight in determining David's response?

KNN for Regression

- When treating each examples equally among the k nearest neighbors (denoted as $\text{neighbors}_k(\mathbf{x})$) of feature vector \mathbf{x} , the combination functions is average
- ✓ The predicted target value $v(\mathbf{x})$ of a feature vector \mathbf{x} can be mathematically defined as follows

$$v(\mathbf{x}) = \frac{\sum_{i \in \text{neighbors}_k(\mathbf{x})} v(i)}{k}$$

- What is the balance for David? is it reasonable that Ruth and Rachel have equal weight in determining David's response?

Customer	Age	Income (1000s)	Cards	Balance (target)	Distance from David
David	37	50	2	?	0
John	35	35	3	10k	$\sqrt{(35 - 37)^2 + (35 - 50)^2 + (3 - 2)^2} = 15.16$
Rachael	22	50	2	20k	$\sqrt{(22 - 37)^2 + (50 - 50)^2 + (2 - 2)^2} = 15$
Ruth	63	200	1	60k	$\sqrt{(63 - 37)^2 + (200 - 50)^2 + (1 - 2)^2} = 152.23$
Jefferson	59	170	1	80k	$\sqrt{(59 - 37)^2 + (170 - 50)^2 + (1 - 2)^2} = 122$
Norah	25	40	4	30k	$\sqrt{(25 - 37)^2 + (40 - 50)^2 + (4 - 2)^2} = 15.74$

$$v(\mathbf{x}) = \frac{10k + 20k + 30k}{3} = 20k, \text{ when } k=3$$

$$v(\mathbf{x}) = \frac{10k + 20k + 30k + 60k + 80k}{5} = 40k, \text{ when } k=5$$

Weighted Voting for Classification

- It is reasonable that each neighbor's contribution to the prediction of a sample is scaled by the similarity between the sample and each neighbor

$$c(\mathbf{x}) = \arg \max_{c \in \text{classes}} \text{score}(c, \text{neighbors}_k(\mathbf{x}))$$

with ~~$\text{score}(c, N) = \sum_{y \in N} [\text{class}(\mathbf{y}) = c]$~~ $\Rightarrow \text{score}(c, N) = \sum_{y \in N} w(\mathbf{x}, \mathbf{y}) \times [\text{class}(\mathbf{y}) = c]$

, and $[\text{class}(\mathbf{y}) = c]$ is 1 if $\text{class}(\mathbf{y}) = c$, otherwise 0

- ✓ $w(\mathbf{x}, \mathbf{y})$ is the weighting function based on the similarity between examples \mathbf{x} and \mathbf{y}

$$w(\mathbf{x}, \mathbf{y}) = \frac{1}{\text{dist}(\mathbf{x}, \mathbf{y})^2}$$

- ✓ The probability is also adjusted by the weight as

$$p(c \mid \mathbf{x}) = \frac{\sum_{y \in \text{neighbors}(\mathbf{x})} w(\mathbf{x}, \mathbf{y}) \times [\text{class}(\mathbf{y}) = c]}{\sum_{y \in \text{neighbors}(\mathbf{x})} w(\mathbf{x}, \mathbf{y})}$$

Example: KNN for Classification(weighted)

- Will David respond or not when contribution weighted by similarity?

Name	Distance	Similarity weight	Contribution	Class
Rachael	15.0	$0.004444 = \frac{1}{15^2}$	0.344	No
John	15.2	0.004348	0.336	Yes
Norah	15.7	0.004032	0.312	Yes
Jefferson	122.0	0.000067	0.005	No
Ruth	152.2	0.000043	0.003	No

$classes = \{yes, no\}$

$score(yes, neighbors_3(\mathbf{x})) = 0.008380$

$score(no, neighbors_3(\mathbf{x})) = 0.004444$



$c(\mathbf{x}_{David}) = yes$

$p(yes|\mathbf{x}_{David}) = \frac{0.008380}{0.008380 + 0.004444} = 0.6534$

- Predictions will **NOT** change when $k=5$, only the probability changes to 0.647

Weighted Voting for Regression

- It is reasonable that each neighbor's contribution to the prediction of a sample is scaled by the similarity between the sample and each neighbor
- ✓ The predicted target value $v(\mathbf{x})$ of a feature vector \mathbf{x} can be mathematically defined as follows (weighted average)

$$v(\mathbf{x}) = \frac{\sum_{\mathbf{y} \in \text{neighbors}_k(\mathbf{x})} w(\mathbf{x}, \mathbf{y}) \cdot v(\mathbf{y})}{\sum_{\mathbf{y} \in \text{neighbors}_k(\mathbf{x})} w(\mathbf{x}, \mathbf{y})}$$

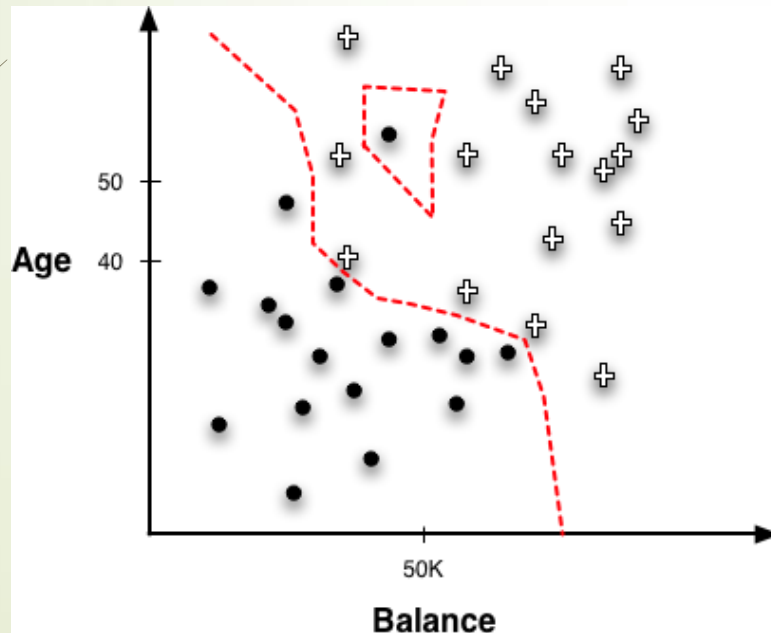
- ✓ $w(\mathbf{x}, \mathbf{y})$ is the weighting function based on the similarity between examples \mathbf{x} and \mathbf{y}

$$w(\mathbf{x}, \mathbf{y}) = \frac{1}{\text{dist}(\mathbf{x}, \mathbf{y})^2}$$

- ✓ Please compute predicted balance of David by yourself with weighted average

How to choose k in KNN

- k can be chosen from 1 to number of all training samples
- ✓ When $k=1$, KNN can always achieve perfect performance in samples, but sensitive to outliers, and hence have bad generalization performance (**overfitted**)



- ✓ The boundaries are erratic and follow the frontiers between training instances of different classes.
- ✓ The isolated instance creates a “negative island” around itself.

- ✓ When k is number of all training samples, the prediction is a constant, base rate

k as Complexity Parameter of KNN

- Number k controls the complexity of KNN : lower k ➔ more complex model
- Choose optimal k by (nested) cross-validation

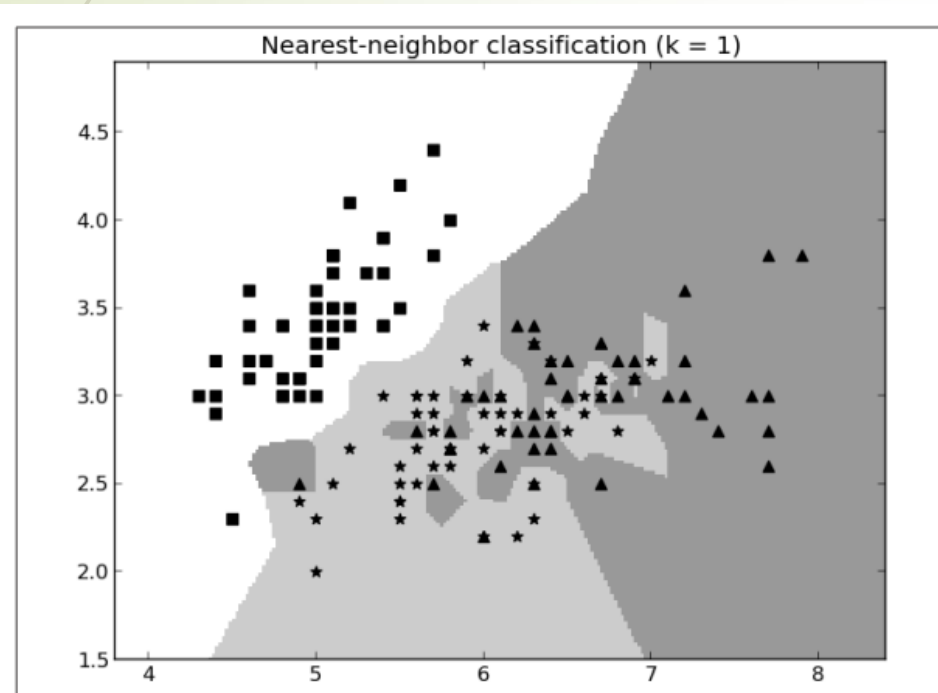


Figure 6-4. Classification boundaries created on a three-class problem created by 1-NN (single nearest neighbor).

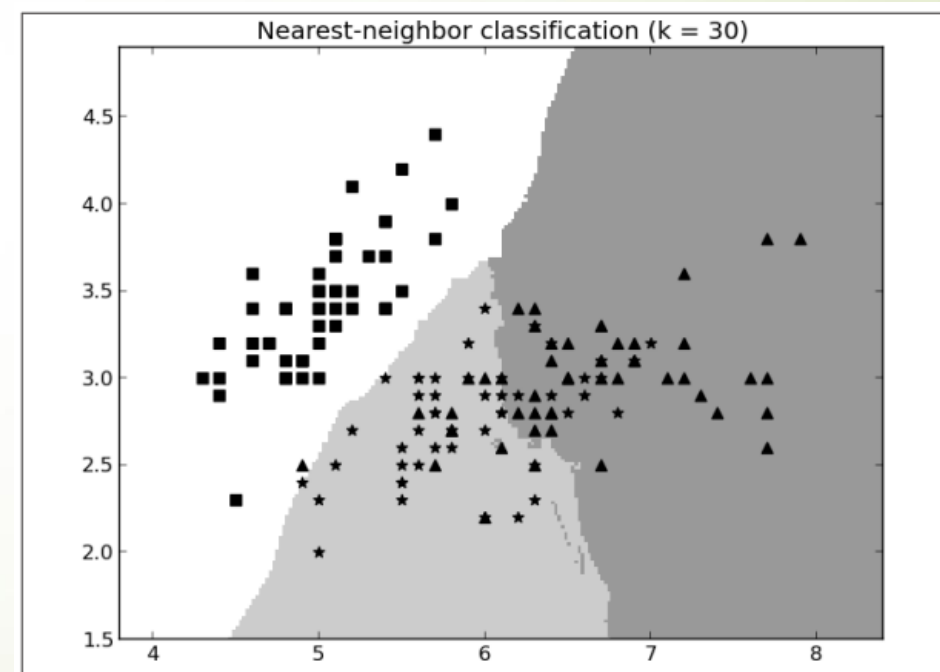


Figure 6-5. Classification boundaries created on a three-class problem created by 30-NN (averaging 30 nearest neighbors).

Issues with KNN

► Intelligibility

- ✓ It is reasonable when applying to recommender system, or help lawyers and doctors
- ✓ It is hard to explain to a customer that the reason of rejecting his credit card application is just that he is similar to some users who default
- ✓ KNN just stores all the training samples, but do not mine understandable “knowledge” from the samples
- ✓ If we use decision tree or linear model , we may explain “All else being equal, if your income had been \$20,000 higher you would have been granted credit card.”
- ✓ If model intelligibility and justification are critical, nearest-neighbor methods should be avoided.

Issues with KNN

► Dimensionality

- ✓ Take into account all features equally when calculating the distance between two instances.
- ✓ Linear model (adjusted weight), decision tree (sort by information gain)
- ✓ Curse of dimensionality: too many meaningless features will mislead the prediction
- ✓ Feature selection is usually required in advance

► Computational efficiency

- ✓ Training KNN is very fast because it usually involves only storing the instances.
- ✓ Huge computational cost in the prediction/classification step: for each new coming sample, we should compute the distance with all training samples to retrieve nearest neighbors
- ✓ Not applicable in decisions should be made very fast and repeated at large scale

KNN in Sklearn

<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>

weights : *str or callable, optional (default = 'uniform')*

weight function used in prediction. Possible values:

- 'uniform' : uniform weights. All points in each neighborhood are weighted equally.
- 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

algorithm : {'auto', 'ball_tree', 'kd_tree', 'brute'}, optional

Algorithm used to compute the nearest neighbors:

- 'ball_tree' will use `BallTree`
- 'kd_tree' will use `KDTree`
- 'brute' will use a brute-force search.
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed to `fit` method.

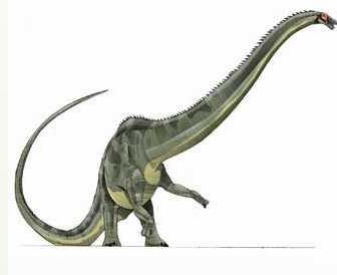
Note: fitting on sparse input will override the setting of this parameter, using brute force.

Outline

- Similarity and Distance
- Nearest-Neighbor Reasoning
- Clustering and Others
- Quiz

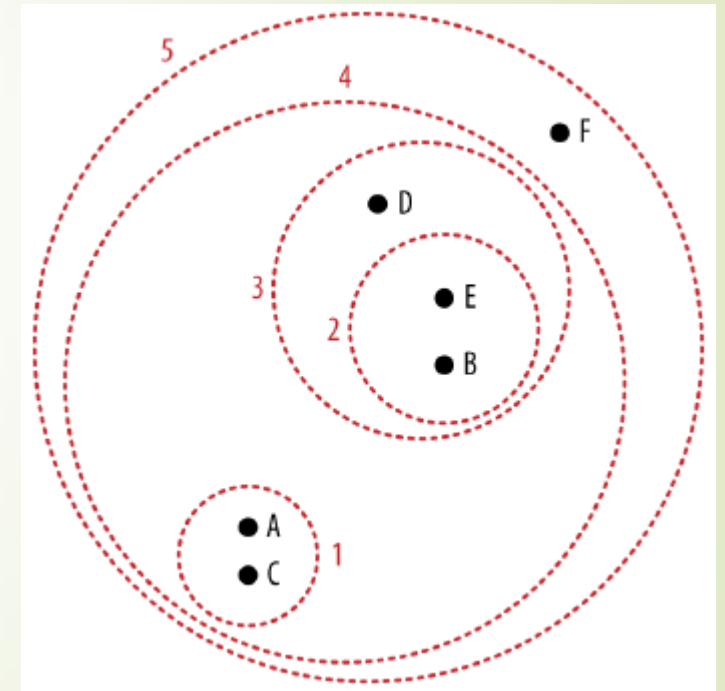
Clustering

- Clustering is another application of our fundamental notion of similarity.
- The basic idea is that we want to find groups of objects (consumers, businesses, whiskeys, etc.), where
 - ✓ The objects within groups are similar, but the objects in different groups are not so
 - ✓ Clustering is unsupervised segmentation: it relies on the similarity computed with by input feature vector rather than being driven by some given target characteristic



Hierarchical Clustering

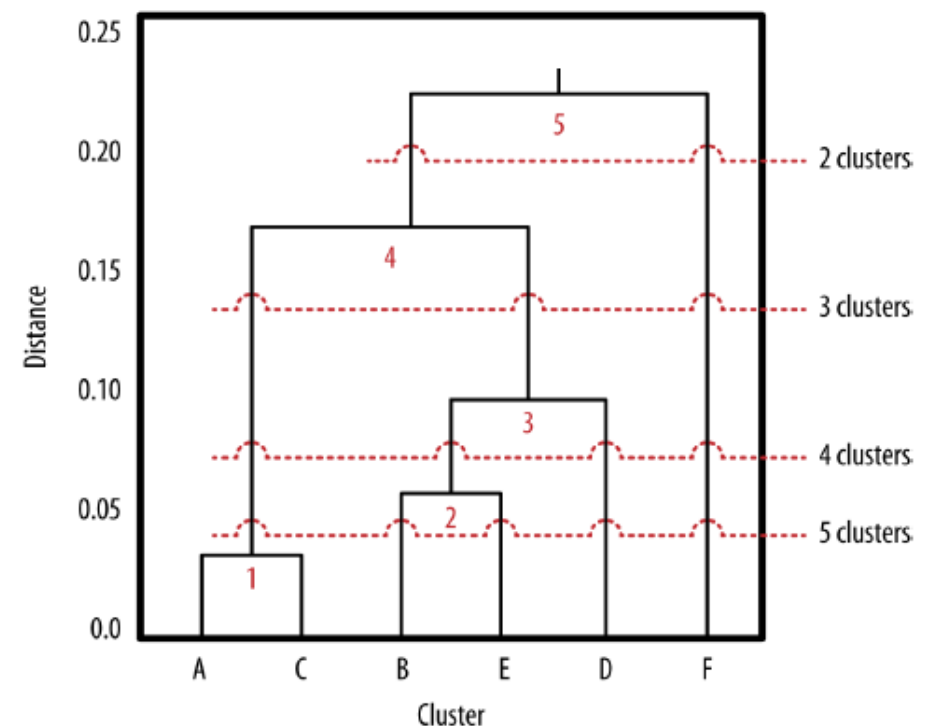
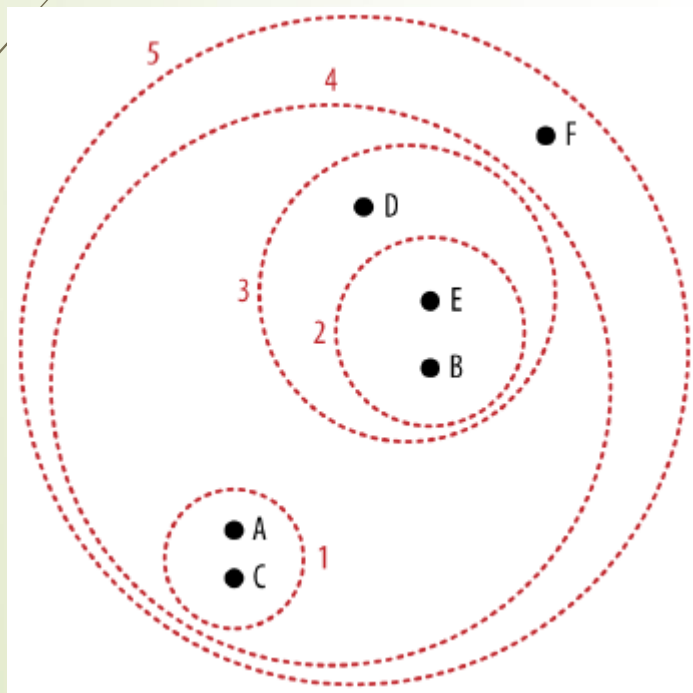
- Hierarchical clustering algorithm is intuitive:
 - ✓ Initialization: each instance form a separate cluster
 - ✓ Iteration: two most similar clusters are merged iteratively until only a single cluster remains
- **Linkage function:** measure the similarity between two clusters (more than 1 instances each)
 - ✓ Nearest point: the similarity between the closest instances in each of the clusters
 - ✓ Farthest point: the similarity between the farthest instances in each of the clusters
 - ✓ Average: the average of similarity score among all pairs of instances



Dendrogram

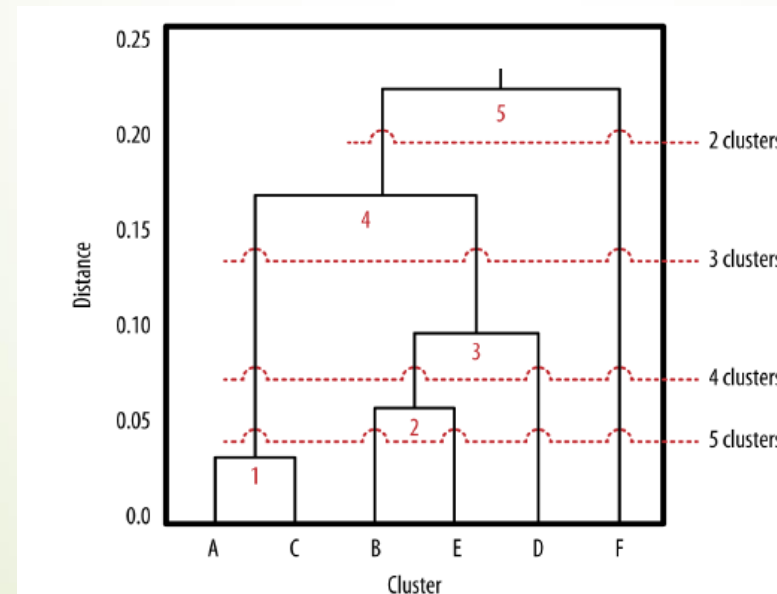
- Dendrogram shows explicitly the hierarchy of the clusters
- ✓ X-axis arranges the individual data points (in no particular order except to avoid line crossings)
- ✓ Y-axis represents the distance between the clusters

(**scikit-learn 0.22.2**) https://scikit-learn.org/stable/auto_examples/cluster/plot_agglomerative_dendrogram.html



Highlights of Hierarchical Clustering

- No need to determine the number of clusters to extract in advance
- Any distance/similarity measure between two instances can be adopted
- The grouping process is easy to be presented with dendrogram
- Cutting the diagram properly can give any desired number of clusters

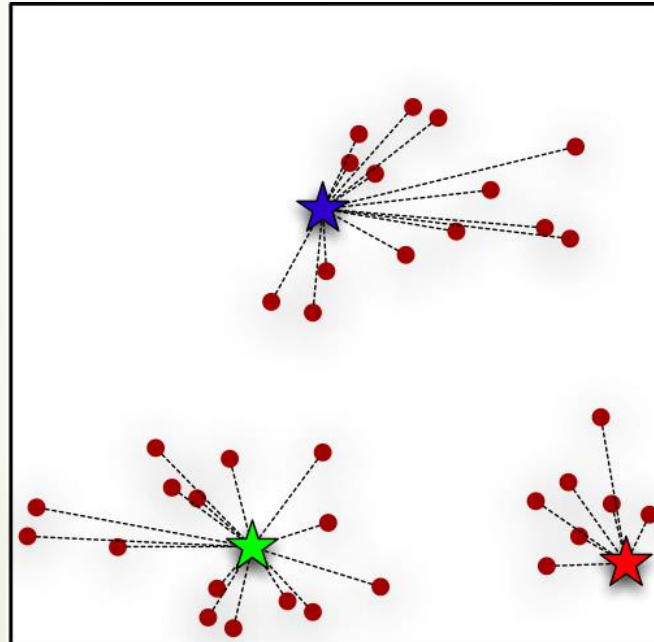


K-means Clustering Algorithm

- K-means algorithm tries to extract a **k** clusters
 - ✓ Initialization: creating k initial cluster centers, usually randomly
 - ✓ Assignment : assign each instance to nearest center, and form clusters
 - ✓ Update : calculate the means/center of each newly formed clusters
 - ✓ Repeat assign step and **update** steps until there is no change or some other stop criteria is satisfied
- The result of k-means clustering include
 - ✓ The **k** cluster centroids when cluster method terminates
 - ✓ Information on which of the data points belongs to each cluster.

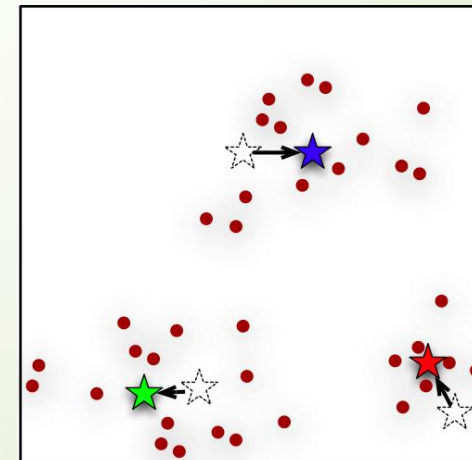
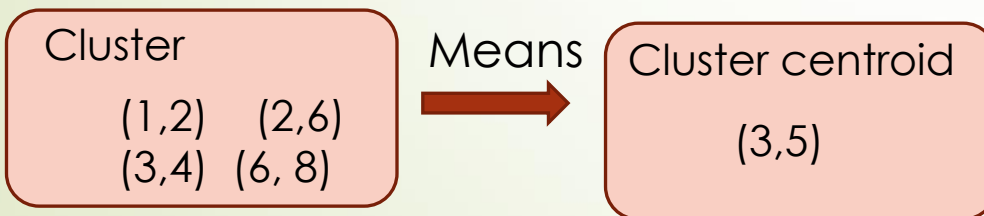
Assignment step

- **Assignment** : assign each instance to nearest center, and form clusters
 - The k-means clustering is usually used with **Euclidean distance** for fast convergence

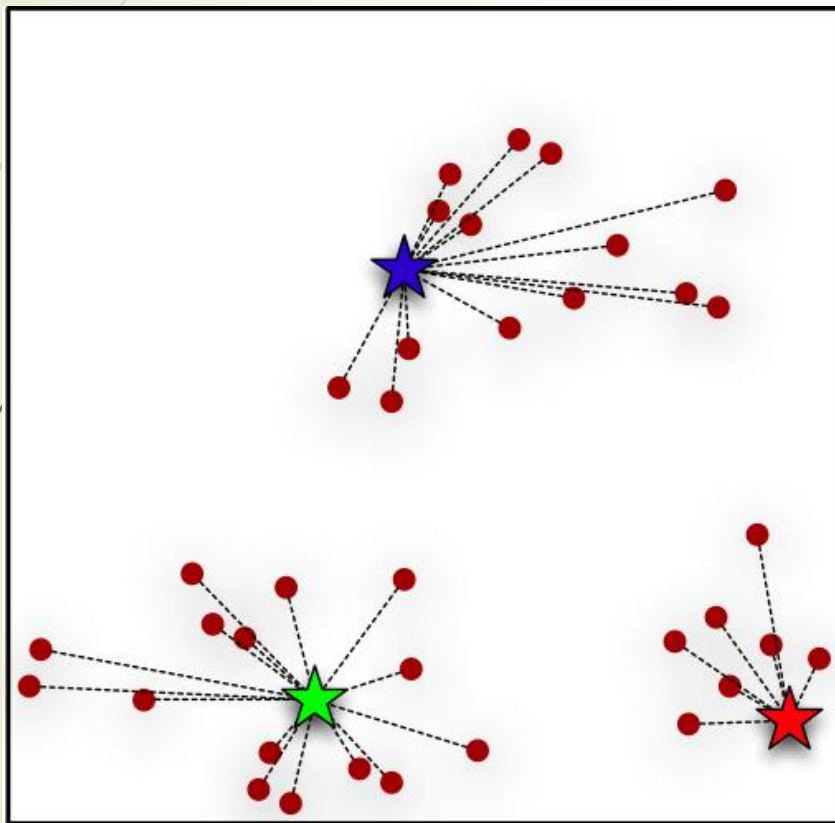


Update Step

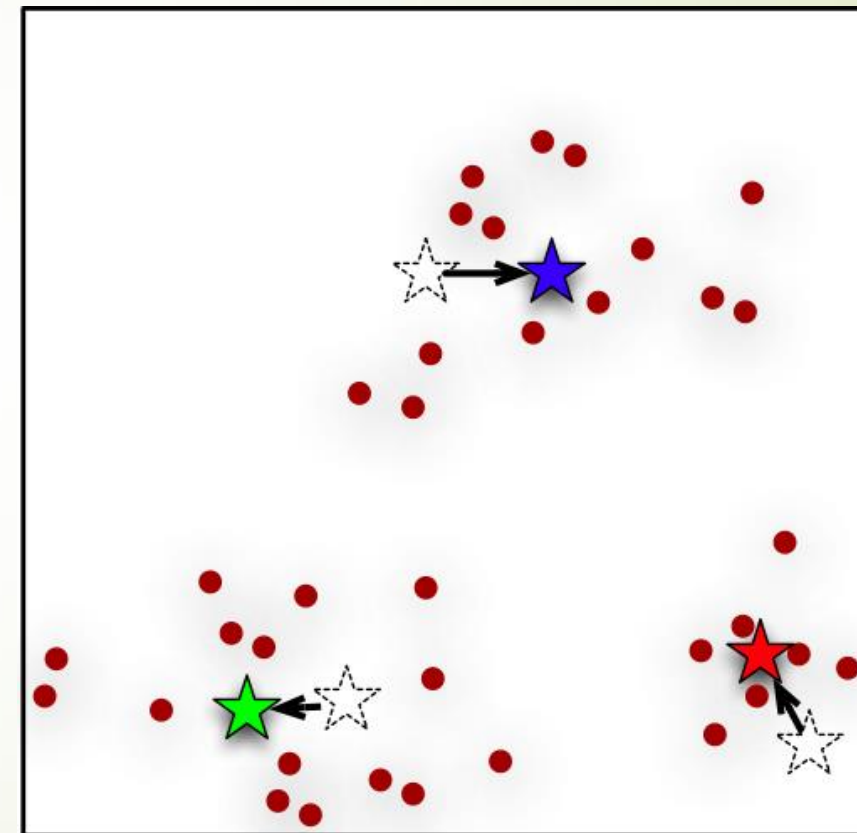
- Clustering data by focusing on the clusters themselves—the groups of instances. The most common
 - ✓ Represent each cluster by its “cluster center,” or centroid
- “Means” as centroid : compute the centroid of a cluster by the means of the values along each dimension for the instances in the cluster.
 - Minimize the sum of squared distance among cluster center and examples in the same cluster – individual clusters’ distortion



Iterations in K-Means



Assign step



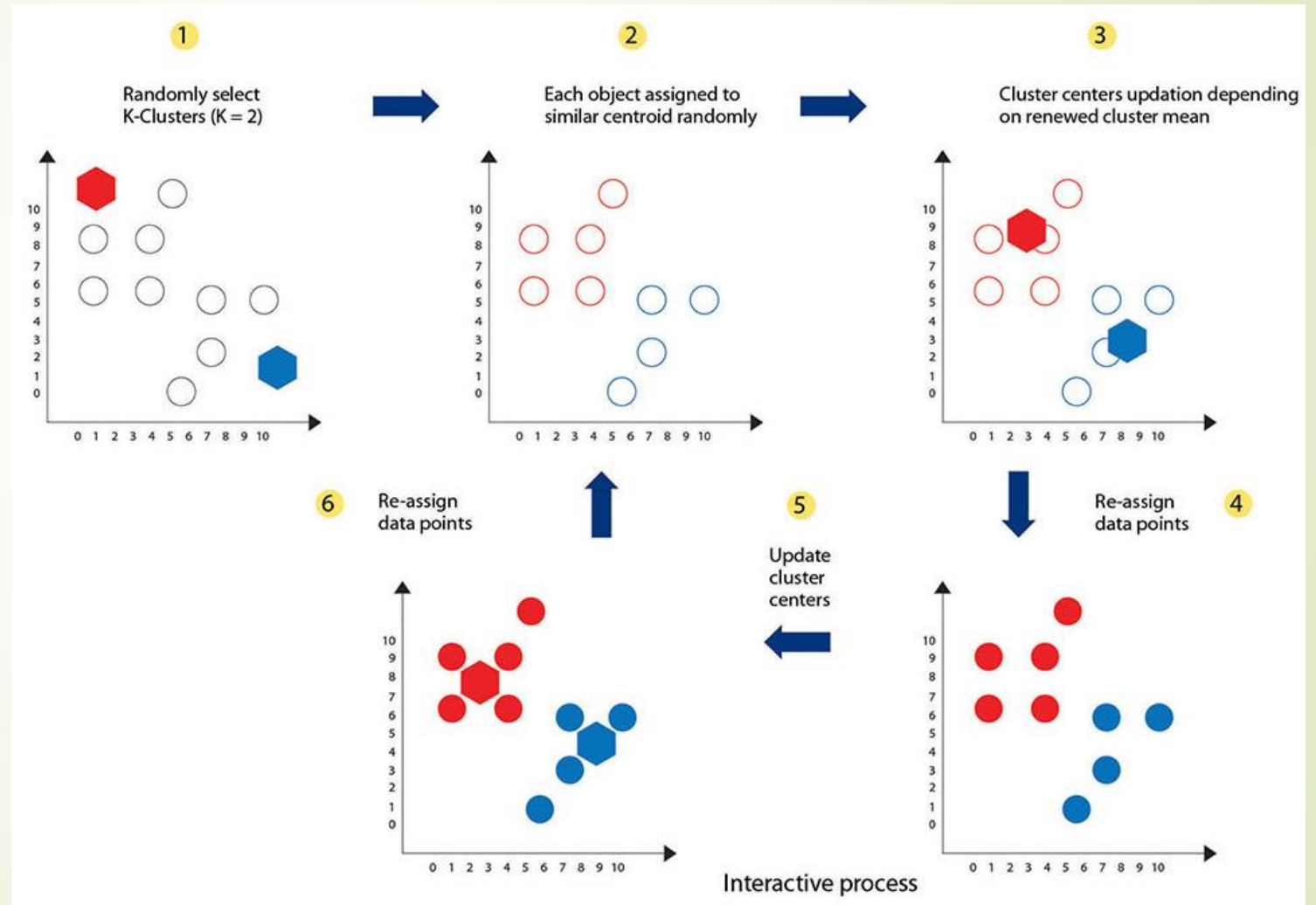
Update step

Shifting Cluster Centroid Until

- No changes made, or
- Few changes made, or
- Reach maximum iterations

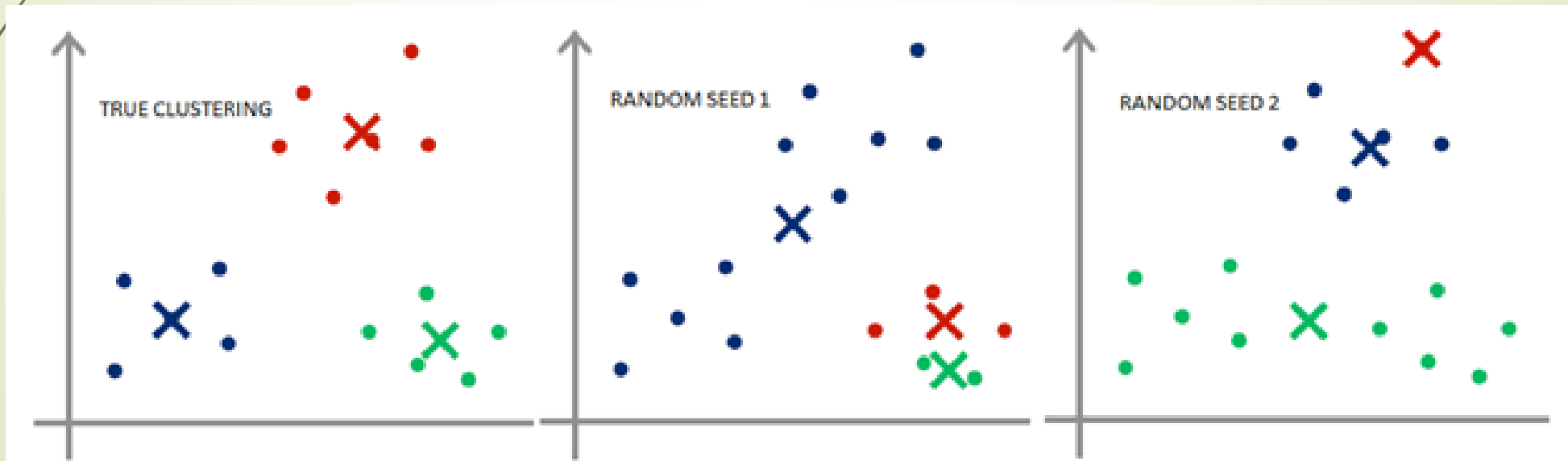
Animation of K-Means:

<http://tech.nitoyon.com/en/blog/2013/11/07/k-means/>



Initialization of K-means

- There is no guarantee that a single run of the k-means algorithm will result in a good clustering
 - ✓ Should be run many times, starting with different random centroids each time
 - ✓ Choose the best by all clusters' distortion (inertia in sklearn) : the sum of the squared differences between each data point and its corresponding centroid



How to Choose Best K

- Distortion is not a good metric to choose the best K

- ✓ It will decrease in general when the K increases

- Lots of criteria for choosing best : low average distance within clusters (ss_W), high average distance between clusters (ss_B),

- ✓ Calinski-Harabasz criterion in Tableau

$$\frac{ss_B}{ss_W} \times \frac{(N-k)}{(k-1)}$$

- ✓ Silhouette score in sklearn : https://scikit-earn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html

- ✓ Hartigan indexes

- ✓ ...

Kmeans in Sklearn

<https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

init : {'k-means++', 'random'} or ndarray of shape (n_clusters, n_features), default='k-means++'

Method for initialization, defaults to 'k-means++':

'k-means++': selects initial cluster centers for k-mean clustering in a smart way to speed up convergence. See section Notes in `k_init` for more details.

'random': choose k observations (rows) at random from data for the initial centroids.

If an ndarray is passed, it should be of shape (n_clusters, n_features) and gives the initial centers.

n_init : int, default=10

Number of time the k-means algorithm will be run with different centroid seeds. The final results will be the best output of n_init consecutive runs in terms of inertia.

max_iter : int, default=300

Maximum number of iterations of the k-means algorithm for a single run.

tol : float, default=1e-4

Relative tolerance with regards to inertia to declare convergence.

Example: Similar Whiskey

- How can we find the most similar whiskey of a given type of whiskey as a data scientist

✓ Construct 5 attributes that can describe the general whiskey as follows

Color: *yellow, very pale, pale, pale gold, gold, old gold, full gold, amber, etc.* (14 values)

Nose: *aromatic, peaty, sweet, light, fresh, dry, grassy, etc.* (12 values)

Body: *soft, medium, full, round, smooth, light, firm, oily.* (8 values)

Palate: *full, dry, sherry, big, fruity, grassy, smoky, salty, etc.* (15 values)

Finish: *full, dry, warm, light, smooth, clean, fruity, grassy, smoky, etc.* (19 values)

- ✓ The values of attributes are **NOT** mutually exclusive (e.g., Aberlour's palate is described as medium, full, soft, round and smooth).
- ✓ Use a feature vector of 68 ($=14+12+8+15+19$) binary (0/1) attributes to reprint a type of whiskey as $[0,1,1,0,\dots,1]$ (with 68 entries of 0/1)

Example: Similar Whiskey by Euclidean Distance

- Given a special type of whiskey(Bunnahabhain), we could compute the Euclidean distance between it and other whiskey, respectively
- We could rank the other whiskey by the distance in descending order as follows

Whiskey	Distance	Descriptors
Bunnahabhain	—	gold; firm,med,light; sweet,fruit,clean; fresh,sea; full
Glenglassaugh	0.643	gold; firm,light,smooth; sweet,grass; fresh,grass
Tullibardine	0.647	gold; firm,med,smooth; sweet,fruit,full,grass,clean; sweet; big,arome,sweet
Ardbeg	0.667	sherry; firm,med,full,light; sweet; dry,peat,sea;salt
Bruichladdich	0.667	pale; firm,light,smooth; dry,sweet,smoke,clean; light; full
Glenmorangie	0.667	p.gold; med,oily,light; sweet,grass,spice; sweet,spicy,grass,sea,fresh; full,long

Supervised versus Unsupervised Learning

- The trade-off between supervised and unsupervised Learning is about how effort is expended in the data mining
 - ✓ Supervised learning: we spent much more time defining precisely the problem we were going to solve, and then have a clear-cut evaluation question
 - ✓ Unsupervised problems often are much more exploratory, we have to spend more time later in the Evaluation stage



Outline

- Similarity and Distance
- Nearest-Neighbor Reasoning
- Clustering and Others
- Quiz

Lab Quiz

- **Deadline:** 17:59 p.m., April. 3, 2020
- Two questions accounting for **5%** of overall score
- **Upload** the **answer worksheet** and the accomplished **Python files** to the **Blackboard**
- You may submit **unlimited times** but only the **LAST** submission will be considered
- **Only the answers in answer sheet** will be referred for grading
- Note: **MUST attach ALL** the required files in every submission/resubmission, otherwise other files will be missing.

Lab Quiz

- You may consider use NearestNeighbors class to rewrite Q3 for exploring more
 - <https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.NearestNeighbors.html>