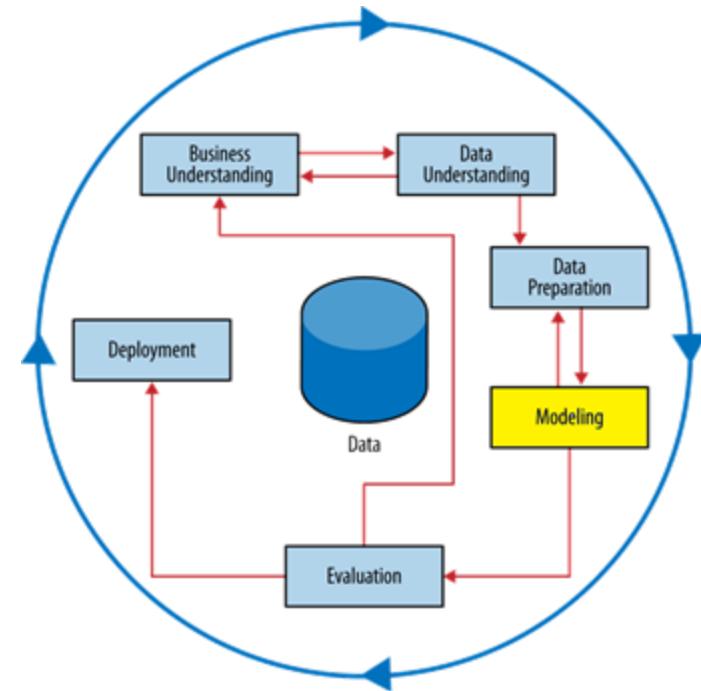


Similarity, Neighbors, and Clusters

PF6

Learning Goals

- ▶ Using **similarity** for **supervised** or **unsupervised** learning.
 - ▶ Supervised learning: classification (and regression)
 - ▶ Unsupervised learning: clustering
- ▶ Techniques:
 - ▶ Calculate **similarity** between instances with **distance** metrics
 - ▶ **K nearest neighbors** for classification
 - ▶ **K means** for clustering



Similarity

- ▶ **Similarity** is the core of many data mining methods
 - ▶ If two instances are similar in some ways, they often share common characteristics.
- ▶ Supervised learning
 - ▶ Instances that are similar (in features) are likely to have the similar target value.
 - ▶ **k -NN**: find k nearest neighbors for an instance, predict its target according to its neighbors' target.
 - ▶ The instance's neighbors are similar to it.
- ▶ Unsupervised learning
 - ▶ **k -Means**: group similar instances together into k clusters to understand the pattern of each cluster.

Distance Functions

- ▶ **Similarity** is measured by the **distance** between **two instances** across multiple feature dimensions.
 - ▶ Step 1: instance i and j are defined as a n -feature vector each
 - ▶ $x_i : (x_{i1}, x_{i2}, \dots, x_{in})$
 - ▶ $x_j : (x_{j1}, x_{j2}, \dots, x_{jn})$
 - ▶ Step 2: calculate their distance in each feature dimension (n), then take aggregation (e.g., sum) for all dimensions.
 - ▶ Shorter distance → Higher similarity

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

Euclidean Distance

- ▶ **Euclidean distance** between instance i and j is the **square root of sum** of the pairwise **squared distance** across n dimensions.
- ▶ Also called L2 norm, denoted by $\|X_1 - X_2\|_2$

$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{in} - x_{jn})^2}$$

- ▶ Euclidean distance between $x_A = (23, 2, 2)$ and $x_B = (40, 10, 1)$ is

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

- ▶ Demerits:
 - ▶ Affected by the scale of measurement, need to scale data first.
 - ▶ Not meaningful for categorical features with multiple values.

Manhattan Distance

- ▶ **Manhattan distance** between instance i and j is the **sum** of the pairwise **absolute distance** across n dimensions.
 - ▶ Also called L1 norm, denoted by $\|X_1 - X_2\|_1$

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{in} - x_{jn}|$$

- ▶ Manhattan distance for $x_A = (23, 2, 2)$ & $x_B = (40, 10, 1)$ is

$$d(A, B) = |23 - 40| + |2 - 10| + |2 - 1| = 26$$

- ▶ Demerits:
 - ▶ Affected by the scale of measurement, need to scale data first.
 - ▶ Not meaningful for categorical features with multiple values.

Cosine Distance

- ▶ **Cosine distance** involves the *products* of corresponding elements:
 - ▶ $\langle x_i, x_j \rangle$: multiply element-by-element and sum the products
 - ▶ $\|x_i\|_2$: L2-norm of a feature vector itself

$$d(x_i, x_j) = 1 - \frac{\langle x_i, x_j \rangle}{\|x_i\|_2 \times \|x_j\|_2}$$

**Cosine
Similarity**

- ▶ Cosine distance for $x_A = (23, 2, 2)$ & $x_B = (40, 10, 1)$ is

$$d(A, B) = 1 - \frac{23*40+2*10+2*1}{\sqrt{23^2+2^2+2^2} * \sqrt{40^2+10^2+1^2}} \approx 0.01$$

- ▶ Merits:
 - ▶ Cosine distance is always in the range of [0,1].
 - ▶ The difference of scales can be ignored.

Example: Text Similarity

- ▶ Frequency of three unique words (i.e., “*performance*”, “*transition*”, “*monetary*”) in three documents, each word is a feature.
 - ▶ Document A: $<7,3,2>$
 - ▶ Document B: $<2,3,0>$
 - ▶ Document C: $<70,30,20>$: replicate Document A 10 times
- ▶ **Question:** is document A more similar to document B or C?
- ▶ **Cosine distance** = 1 - Cosine similarity

$$d(A, B) = 1 - \frac{7 * 2 + 3 * 3 + 2 * 0}{\sqrt{7^2 + 3^2 + 2^2} * \sqrt{2^2 + 3^2 + 0^2}} = 0.19$$

Similarity=0.81

$$d(A, C) = 1 - \frac{7 * 70 + 3 * 30 + 2 * 20}{\sqrt{7^2 + 3^2 + 2^2} * \sqrt{70^2 + 30^2 + 20^2}} = 0$$

Similarity=1

Example: Text Similarity

- ▶ Calculate Euclidean & Manhattan distance: $d(A, B)$ and $d(A, C)$
 - ▶ Document A: $<7, 3, 2>$
 - ▶ Document B: $<2, 3, 0>$
 - ▶ Document C: $<70, 30, 20>$

▶ Euclidean distance

$$d(A, B) = \sqrt{(7 - 2)^2 + (3 - 3)^2 + (2 - 0)^2} \approx 5.39$$

$$d(A, C) = \sqrt{(7 - 70)^2 + (3 - 30)^2 + (2 - 20)^2} \approx 70.87$$

▶ Manhattan distance

$$d(A, B) = |7 - 2| + |3 - 3| + |2 - 0| = 7$$

$$d(A, C) = |7 - 70| + |3 - 30| + |2 - 20| = 108$$

Important to scale data first for Euclidean and Manhattan distance.

Nearest Neighbors for Target Prediction

- ▶ Given a new instance:
 - ▶ Retrieve all training data, select some **neighbors** that are **most similar** to it;
 - ▶ How many neighbours shall be selected?
 - ▶ Predict its target value based on the target of its nearest neighbors.
- ▶ **Classification**: apply a **combining function** (e.g., majority vote) to its nearest neighbors.
- ▶ Probability estimation: proportion of nearest neighbors with “Yes”

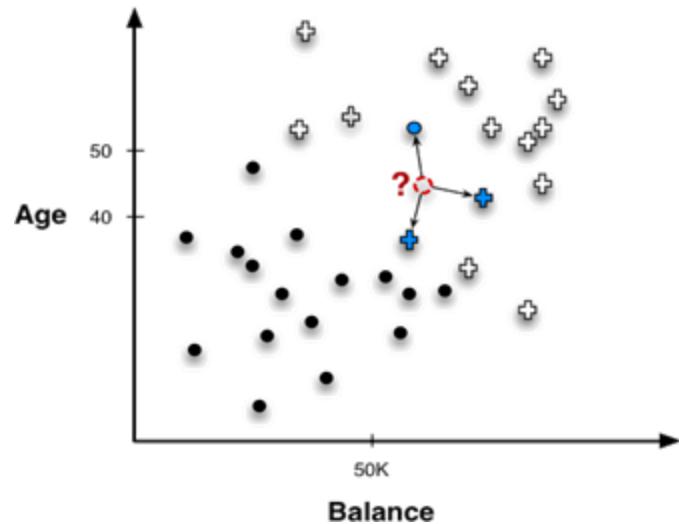


Figure 6-2. Nearest neighbor classification.

Example: Credit Card Promotion

- ▶ A binary classification task
 - ▶ 5 training instances with 3 features: *Age*, *Income*, *Number of Cards*.
 - ▶ Will David (a new instance) **respond** to the marketing offer (**Target value**: Yes or No) of a credit card company ?
- ▶ Compute David's **euclidean distance** to 5 training instances:

Customer	Age	Income (1000s)	Cards	Response (target)	Distance from David
David	37	50	2	?	
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	$\sqrt{(25 - 37)^2 + (40 - 50)^2 + (4 - 2)^2} = 15.74$

Example: Credit Card Promotion

- ▶ **Classification:** will David respond to the marketing offer?
 - ▶ David's 3 nearest neighbors:
 - ▶ Rachel (No)
 - ▶ John (Yes)
 - ▶ Norah (Yes)
 - ▶ Predicted *Response* value
 - ▶ Yes (majority vote)
 - ▶ Class probability
 - ▶ 2/3 of the chance is 'Yes'

Name	Response	Distance
David	?	
Rachel	No	15
John	Yes	15.16
Norah	Yes	15.74
Jefferson	No	122
Ruth	No	152.23

Euclidean Distance according to "Age", "Income" and "Cards".

- ▶ **Questions:**
 - ▶ How many neighbours should be selected in prediction?
 - ▶ Assign heavier weight to Rachel as she is more similar as David?

Similarity-moderated Classification

- ▶ **Weighted voting:** each neighbor is different in their importance in target prediction, according to their distance to the new instance.
- ▶ **Similarity weight:** inverse of the squared distance

$$w(x_i, x_j) = \frac{1}{d(x_i, x_j)^2}$$

The distance can be Euclidean, Manhattan, or Cosine, etc.

- ▶ **Contribution:** proportion of similarity weight (sum of all k neighbors' is 1)

$$\frac{w(x_i, x_j)}{\sum_k w(x_i, x_j)}$$

- ▶ When $k = 5$ and Euclidean Distance was used, will David respond?

Name	Response	Euclidean Distance	Similarity Weight	Contribution
Rachel	No	15	0.004444	0.344
John	Yes	15.16	0.004348	0.336
Norah	Yes	15.74	0.004032	0.312
Jefferson	No	122	0.000067	0.005
Ruth	No	152.23	0.000043	0.003

0.65 Yes
when $k = 5$

K Nearest Neighbors

- ▶ **k -NN** is the nearest neighbor method with k neighbors.
 - ▶ *Odd* numbers are convenient to break ties for majority vote classification.
 - ▶ A larger k value means the target estimation is more smoothed out among the neighbors.
- ▶ Example: credit card promotion – number of training instances ($N = 5$)
 - ▶ **Classification** by majority vote (target: *Response*)
 - ▶ $k = 1 \Rightarrow \text{No}$
 - ▶ $k = 3 \Rightarrow \text{Yes}$
 - ▶ $k = N \Rightarrow \text{No}$

What is the best k value then?

How Many Neighbors?

- ▶ 1-NN model overfits very strongly.
 - ▶ When predict on training data, each instance considers **itself** as its nearest neighbor:
 - ▶ The predicted target for each training instance is exactly its actual target (**100% training accuracy**)
 - ▶ It often leads to worse performance on unseen data.
 - ▶ Worse generalization performance.

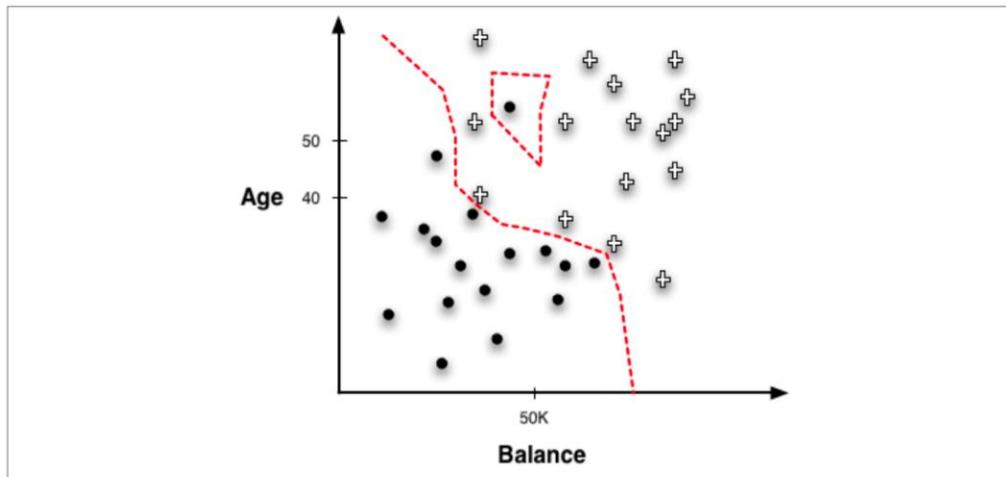
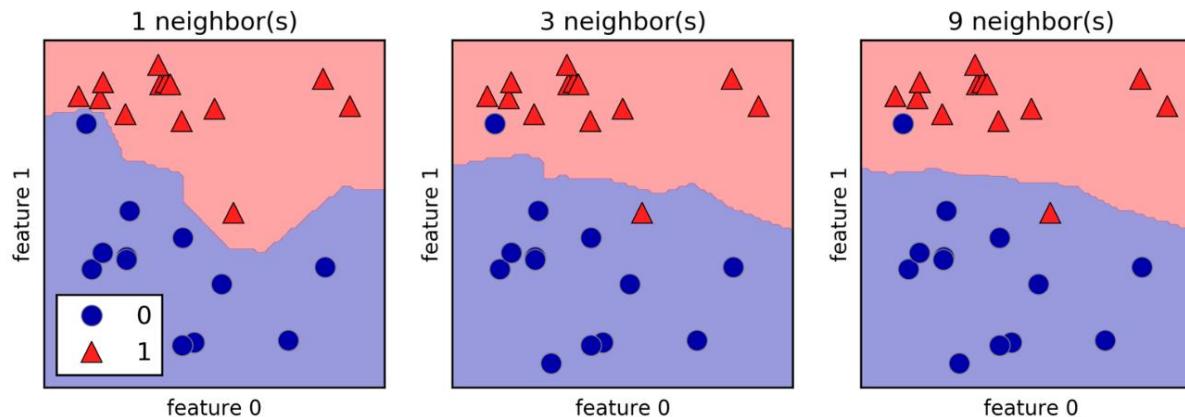


Figure 6-3. Boundaries created by a 1-NN classifier.

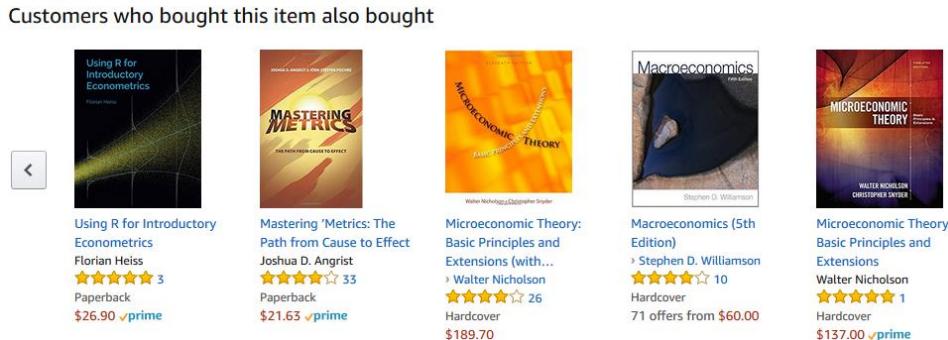
How Many Neighbors?

- ▶ **k is a complexity (hyper)parameter**
 - ▶ A larger k value means a simpler model (i.e., smoother decision boundary).
 - ▶ What the predicted target value when $k = \text{training data size}$?
 - ▶ **GridSearchCV** can be used to find the best k value.
 - ▶ Best k value is found by comparing their average generalization performance.
- ▶ Binary classification with different k values.



Issues with k-NN

- ▶ Intelligibility
 - ▶ Suitable for recommendation systems
 - ▶ Amazon: Book A was recommended to you, as you have bought book B and C (i.e., nearest neighbors for Book A).



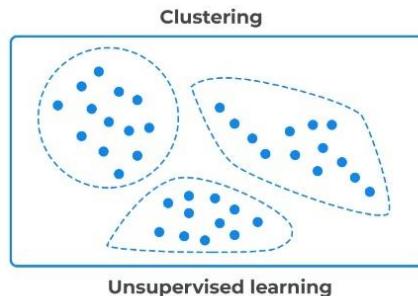
- ▶ May NOT be suitable when justification is required
 - ▶ Rejecting a loan application because of its similarity to other defaulted cases may not be acceptable.

Issues with k-NN

- ▶ Dimensionality
 - ▶ Having too many (irrelevant) features may confuse distance calculation → Curse of Dimensionality
 - ▶ Solutions: feature selection through domain knowledge or automated methods (e.g., PCA).
- ▶ Computational efficiency
 - ▶ Model training is very fast: simply store all training instances.
 - ▶ Querying the database to find nearest neighbors for a new instance (for target prediction) can be very expensive.
 - ▶ Impractical for online advertising which requires instant response.

Use Similarity for Clustering

- ▶ **Clustering** is a process of grouping data into clusters so that
 - ▶ Instances in the same cluster are **similar**;
 - ▶ But are very **dissimilar** to instances in other clusters.



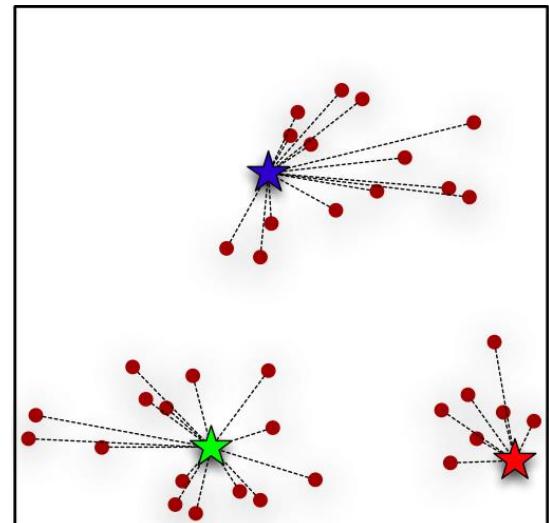
- ▶ Clustering is an **unsupervised** method
 - ▶ to explore the **natural structure** of the (training) data itself.
 - ▶ e.g., characterize customer groups based on their purchasing patterns.
 - ▶ to **predict** the cluster label for unseen data.
 - ▶ In this case, generalization performance of the model matters.
 - ▶ often used as a **preprocessing** step for supervised methods.
 - ▶ Estimated cluster labels used as target label.

K-Means Clustering (1/2)

1. **Initialization**: randomly pick k points (either actual instance or not) in the feature space as initial cluster centroids (m_1, \dots, m_k).
2. **Cluster allocation**: assign each instance p to its closest cluster centroid to form k clusters (C_1, \dots, C_k).
 - ▶ Compare Euclidean Distance between instance p and all k centroids.
3. **Calculate clustering quality (inertia)**: take the sum of squared distance (error) in each cluster, then sum up for all k clusters.

$$SSE = \sum_{i=1}^k \sum_{p \in C_i} d(p, m_i)^2$$

- p : instances in cluster C_i
- $d(p, m_i)$: Euclidean Distance between p and its centroid m_i across n feature dimensions

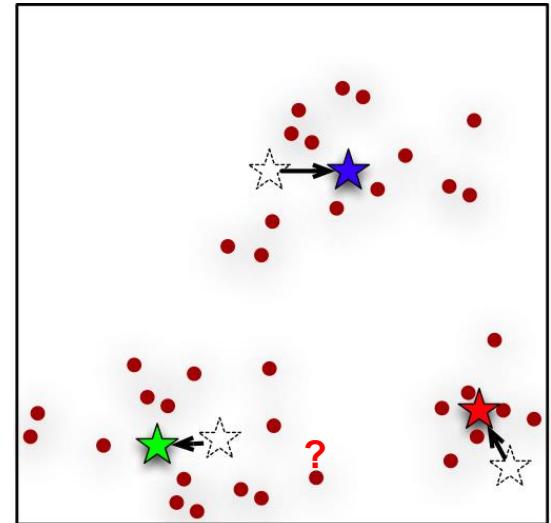


K-Means Clustering (2/2)

4. **Find actual centroids**: calculate the mean for all instances in each cluster (m_1^*, \dots, m_k^*), use them to replace old cluster centroids (m_1, \dots, m_k).

- ▶ With cluster centroids updated, their distance to each instance change as well.
- ▶ Need to allocate clusters again.

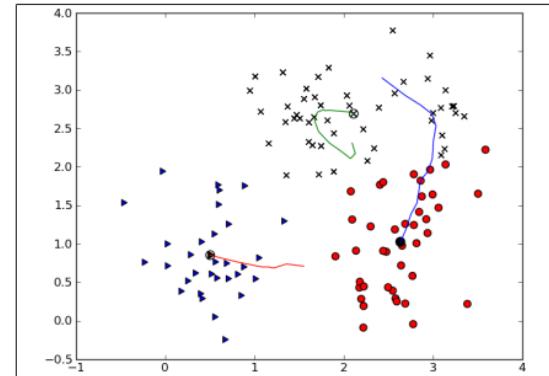
5. **Repeat Step 2 to 4** multiple times until improvement in **SSE** is negligible or none.



- ▶ Apply a clustering model (i.e., final centroids) for **cluster prediction**:
 - ▶ For a new instance, assign it to its closest cluster centroid.
 - ▶ Generalization performance is also measured by SSE.

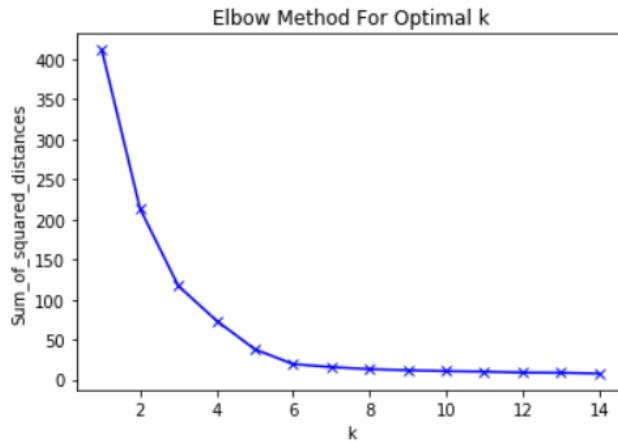
K-Means Clustering

- ▶ **3-Means clustering:** movement of cluster centroids through **16 iterations**.
 - ▶ The marker shape of each point represents their final cluster identity.
- ▶ With random initial centroids, no guarantee that a single run will yields good result (**local optimum** might be found).
 - ▶ Compare the clustering quality for different initial centroids, whichever returns the lowest SSE will be used.
 - ▶ Each initial setting is run for some iterations for comparison.
 - ▶ **k-means++:** push initial centroids away from each other and hence speeds up convergence.
 - ▶ The first centroid is selected at random, subsequent centroids are selected according to their distance to nearest existing centroid.



Find the Optimal Value for k

- ▶ Visualize k values vs. clustering quality on training data.
 - ▶ Often used when we only care the underlying structure of the data.



When k value increases, SSE will decrease.

Minimum k where the stabilization begins is a good choice (e.g., $k = 6$)

- ▶ If generalization is key, can we find best k value via GridSearchCV?
 - ▶ Compare k values according to their *mean cv score* on validation set.
 - ▶ The score is the opposite of SSE (negative SSE: the bigger, the better).
 - ▶ **Not recommended** as the biggest k value usually yields best cv score.
 - ▶ We should aim for a small k value in clustering tasks.

Clustering vs. Classification

- ▶ Interpreting the clustering result can be challenging.
 - ▶ What differentiates a cluster from the others?
 - ▶ A cluster centroid describes the average of cluster members, which can be very detailed.

- ▶ We can combine unsupervised methods (e.g., clustering) with supervised methods (e.g., classification).
 - ▶ Use predicted cluster labels to label instances (target).
 - ▶ Train a classifier (e.g., logistic regression) based on the features and the target.
 - ▶ The classifier can tell the relationship between features and the target in a more intelligent and concise way.