DSC478: Programming Machine Learning Applications

Roselyne Tchoua

rtchoua@depaul.edu

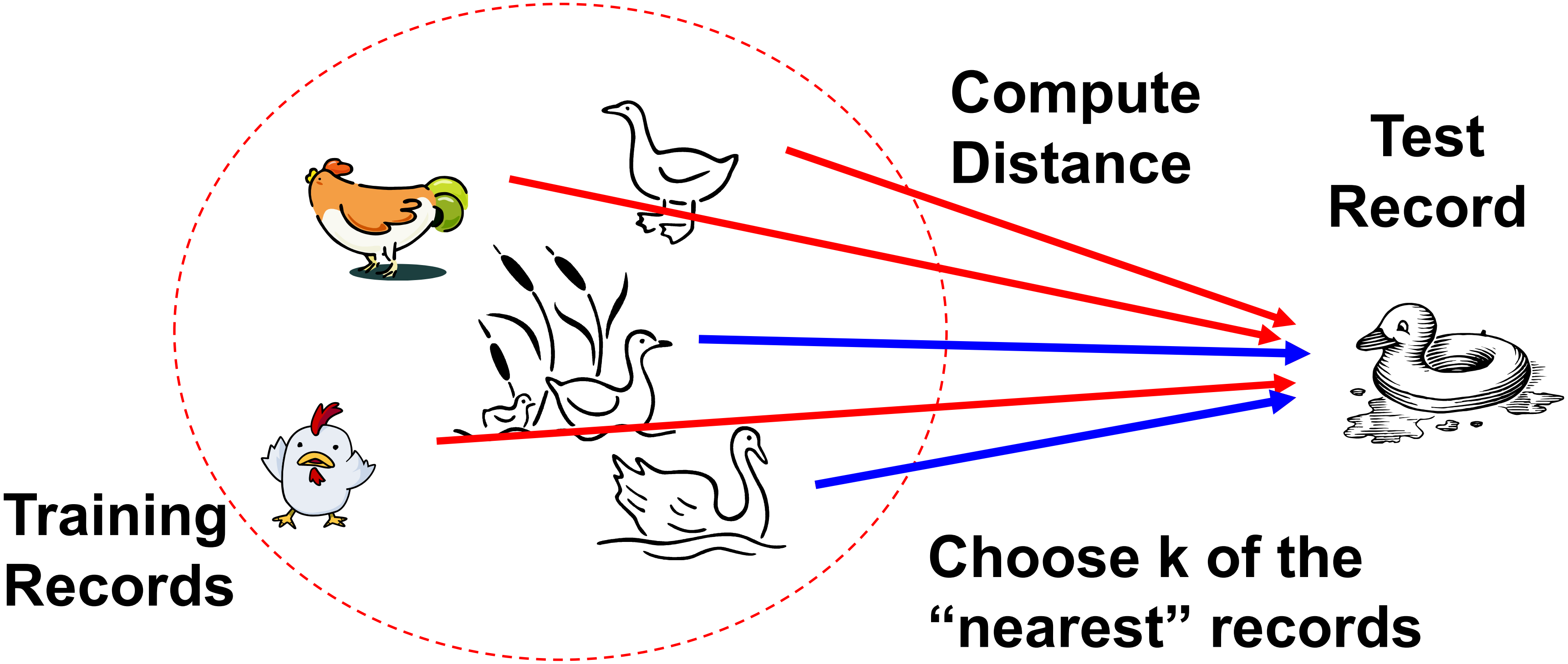
School of Computing, CDM, DePaul University

Distance-Based Classification

- Basic Idea: classify new instances based on their similarity to or distance from instances we have seen before
  - also called "**instance-based learning**"
  - Also called "**memory-based learning**" (MBR)
- Simplest form of MBR: Rote Learning
  - learning by memorization
  - save all previously encountered instance; given a new instance, find one from the memorized set that most closely "resembles" the new one; assign new instance to the same class as the "nearest neighbor"
  - more general methods try to find k nearest neighbors rather than just one
  - **but how do we define "resembles?"**
- MBR is "**lazy**"
  - defers all of the real work until new instance is obtained; no attempt is made to learn a generalized model from the training set
  - less data preprocessing and model evaluation, but more work has to be done at classification time

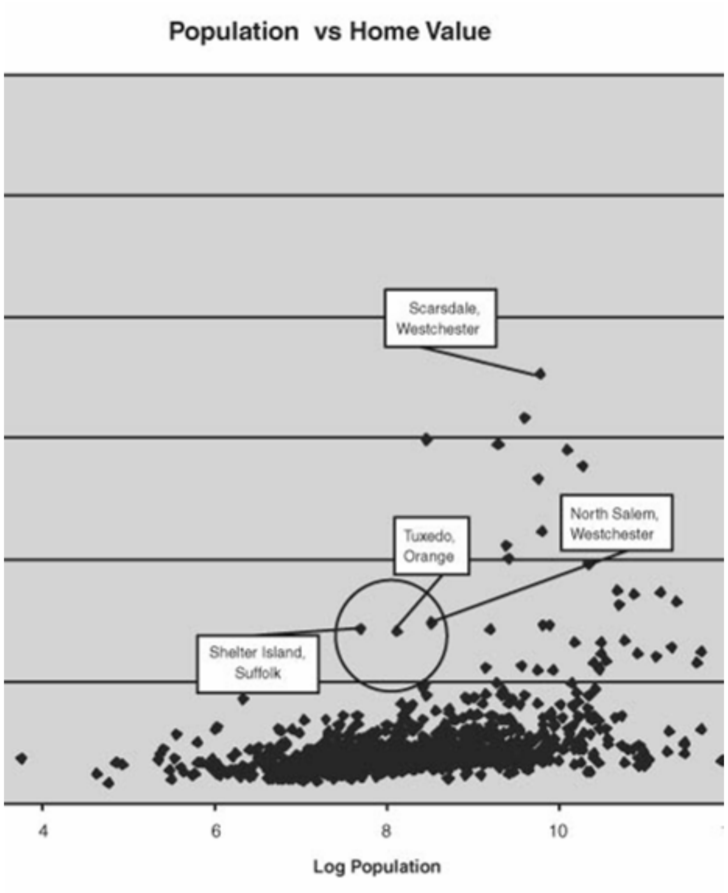
Nearest Neighbor Classifiers

Basic idea: If it walks like a duck, quacks like a duck, then it's probably a duck



Nearest Neighbor Example

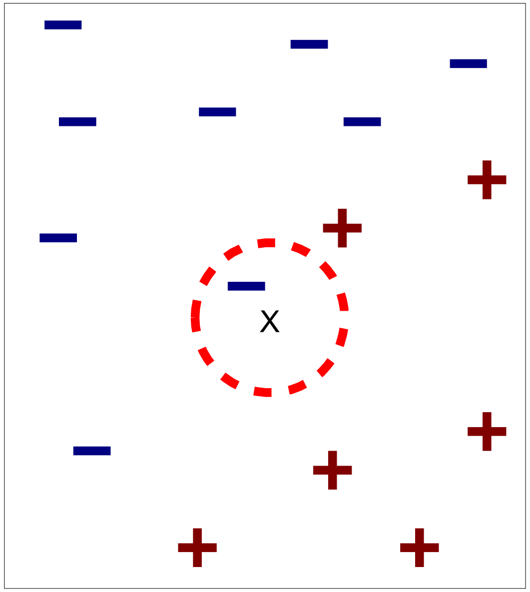
- Classify nearest neighbors based on descriptive variables – population & median home prices (**not geography in this example**)
- Range midpoint in 2 neighbors is \$1,000 & \$1,250 so Tuxedo rent should be \$1,125; 2nd method yields rent of \$977
- Actual midpoint rent in Tuxedo turns out to be \$1,250 (one method) and \$907 in another.



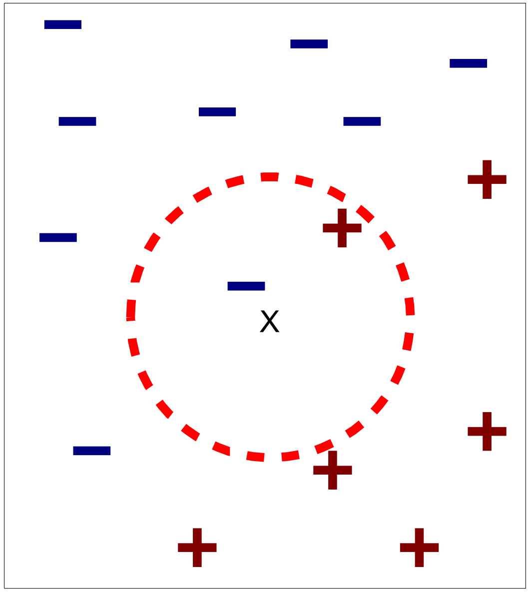
K-Nearest Neighbor Strategy

- Given object *x*, find the ***k* most similar** objects to *x*
  - The *k* nearest neighbors
  - **Variety of distance or similarity measures** can be used to identify and rank neighbors
  - **Note** that this requires comparison between *x* and all objects in the database
- **Classification**:
  - Find the class (category) label for each of the *k* neighbor
  - Use a voting or weighted voting approach to determine the majority class among the neighbors (a combination function)
    - **Weighted voting means the closest neighbors count more**
  - Assign the majority class label to *x*
- **Prediction**:
  - Identify the value of the target attribute for the *k* neighbors
  - Return the weighted average as the predicted value of the target attribute for *x*

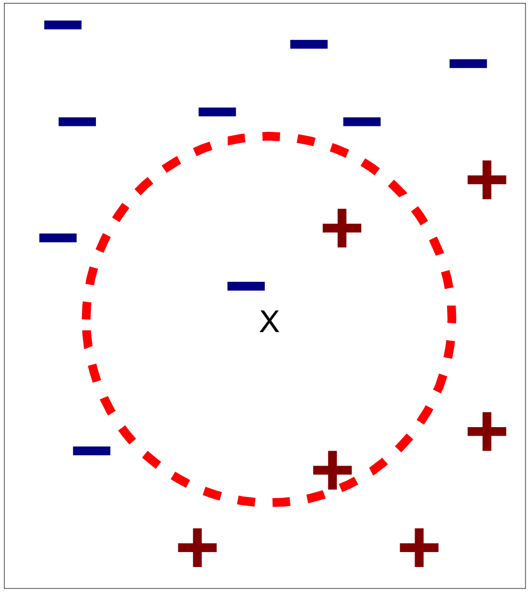
K-Nearest Neighbor Strategy



(a) 1-nearest neighbor



(b) 2-nearest neighbor



(c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k **smallest distance** to x

- If k is too small, it is sensitive to noise (noisy points)
- If it’s too large it can include points from the other classes

Impact of K on Predictions

- In general, different values of *k* affect the outcome of classification
- If k is too small, it is sensitive to noise (noisy points)
- If it's too large it can include points from the other classes
- We can associate a **confidence** level with predictions (this can be the % of neighbors that are in agreement)
- Problem is that no single category may get a majority vote
  - **Pick k = # of classes + 1** to avoid ties or use weighted voting scheme
- **If there is strong variations in results for different choices of *k*, this an indication that the training set is not large enough**

Voting Schemes

- “democracy”: Poll the neighbors for the answer and use the majority vote
- “shareholder democracy”: Give neighbors different weights:
  - Distance-based: **closer neighbors have more weight**
    - In Scikit-learn methods: ***uniform*** or ***distance***
  - “value” of the vote is the inverse of the distance (may need to add a small constant)
  - the weighted sum for each class gives the combined score for that class
  - to compute confidence, need to take weighted average
  - Design custom weighing scheme based on domain knowledge
  - **Prevents ties and helps distinguish between competing neighbors**

Voting Approach - Example

Will a new customer respond to solicitation?

ID	Gender	Age	Salary	Respond?
1	F	27	19,000	no
2	M	51	64,000	yes
3	M	52	105,000	yes
4	F	33	55,000	yes
5	M	45	45,000	no
new	F	45	100,000	?

Using the voting method without confidence

	Neighbors	Answers	k =1	k = 2	k = 3	k = 4	k = 5
D_man	4,3,5,2,1	Y,Y,N,Y,N	yes	yes	yes	yes	yes
D_euclid	4,1,5,2,3	Y,N,N,Y,Y	yes	?	no	?	yes

Using the voting method with a confidence

	k =1	k = 2	k = 3	k = 4	k = 5
D_man	yes, 100%	yes, 100%	yes, 67%	yes, 75%	yes, 60%
D_euclid	yes, 100%	yes, 50%	no, 67%	yes, 50%	yes, 60%

KNN for Document Categorization

	T1	T2	T3	T4	T5	T6	T7	T8	Cat
DOC1	2	0	4	3	0	1	0	2	**Cat1**
DOC2	0	2	4	0	2	3	0	0	**Cat1**
DOC3	4	0	1	3	0	1	0	1	**Cat2**
DOC4	0	1	0	2	0	0	1	0	**Cat1**
DOC5	0	0	2	0	0	4	0	0	**Cat1**
DOC6	1	1	0	2	0	1	1	3	**Cat2**
DOC7	2	1	3	4	0	2	0	2	**Cat2**
DOC8	3	1	0	4	1	0	2	1	?

KNN for Document Categorization

Using Cosine Similarity to find K=3 neighbors:

	T1	T2	T3	T4	T5	T6	T7	T8	Norm	Sim(D8,Di)
DOC1	2	0	4	3	0	1	0	2	5.83	0.61
DOC2	0	2	4	0	2	3	0	0	5.74	0.12
DOC3	4	0	1	3	0	1	0	1	5.29	0.84
DOC4	0	1	0	2	0	0	1	0	2.45	0.79
DOC5	0	0	2	0	0	4	0	0	4.47	0.00
DOC6	1	1	0	2	0	1	1	3	4.12	0.73
DOC7	2	1	3	4	0	2	0	2	6.16	0.72
DOC8	3	1	0	4	1	0	2	1	5.66	

e.g.: Sim(D8,D7) = (D8 • D7) / (Norm(D8).Norm(D7)

= (3x2+1x1+0x3+4x4+1x0+0x2+2x0+1x2) / (5.66 x 6.16)

= 25 / 34.87 = 0.72

KNN for Document Categorization

	T1	T2	T3	T4	T5	T6	T7	T8	Cat	Sim(D8,Di)
DOC1	2	0	4	3	0	1	0	2	**Cat1**	0.61
DOC2	0	2	4	0	2	3	0	0	**Cat1**	0.12
DOC3	4	0	1	3	0	1	0	1	**Cat2**	0.84
DOC4	0	1	0	2	0	0	1	0	**Cat1**	0.79
DOC5	0	0	2	0	0	4	0	0	**Cat1**	0.00
DOC6	1	1	0	2	0	1	1	3	**Cat2**	0.73
DOC7	2	1	3	4	0	2	0	2	**Cat2**	0.72
DOC8	3	1	0	4	1	0	2	1	5.66	

- Simple voting:
  - Cat for DOC 8 = Cat2 with confidence 2/3 = 0.67
- Weighted voting:
  - Cat for DOC 8 = Cat2
  - Confidence: (0.84 + 0.73) / (0.84 + 0.79 + 0.73) = 0.66

KNN for Collaborative Filtering

- Starts with a history of people's personal preferences
- Uses a **distance function** – people who like the same things are "close"
- Uses **"votes"** which are weighted by distances, so close neighbor votes count more
- Basically, judgments of a peer group are important
- Knowing that lots of people liked something is not sufficient…

KNN Example: Collaborative Filtering

- Collaborative Filtering Example
  - A movie rating system
  - Ratings scale: 1 = “hate it”; 7 = “love it”
  - Historical DB of users includes ratings of movies by Sally, Bob, Chris, and Lynn
  - Karen is a new user who has rated 3 movies, but has not yet seen “Independence Day”; should we recommend it to her?

	Star Wars	Jurassic Park	Terminator 2	Indep. Day
Sally	7	6	3	7
Bob	7	4	4	6
Chris	3	7	7	2
Lynn	4	4	6	2
Karen	7	4	3	?

Will Karen like “Independence Day?”

KNN Example: Collaborative Filtering

	Star Wars	Jurassic Park	Terminator 2	Indep. Day	*Average*	Cosine	Distance	Euclid	Pearson
Sally	7	6	3	7	5.33	0.983	2	2.00	0.85
Bob	7	4	4	6	5.00	0.995	1	1.00	0.97
Chris	3	7	7	2	5.67	0.787	11	6.40	-0.97
Lynn	4	4	6	2	4.67	0.874	6	4.24	-0.69
Karen	7	4	3	?	4.67	1.000	0	0.00	1.00

K	Prediction
1	6
2	6.5
3	5

K is the number of nearest neighbors used in to find the average predicted ratings of Karen on Indep. Day.

Example computation:

Pearson(Sally, Karen) = ((7-5.33)*(7-4.67) + (6-5.33)*(4-4.67) + (3-5.33)*(3-4.67))
/ SQRT($((7-5.33)^2 + (6-5.33)^2 + (3-5.33)^2) * ((7-4.67)^2 + (4-4.67)^2 + (3-4.67)^2)$) = 0.85

$$a'_k = (a_k - mean(A)) / std(A)$$
$$b'_k = (b_k - mean(B)) / std(B)$$
$$correlation(A, B) = A' \bullet B'$$

KNN Example: Collaborative Filtering (with KNN)

- In practice a more sophisticated approach is used to generate the predictions based on the nearest neighbors
- To generate predictions for a target user *a* on an item *i*:

$$\boxed{p_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^{k} (r_{u,i} - \bar{r}_u) \times sim(a,u)}{\sum_{u=1}^{k} sim(a,u)}}$$

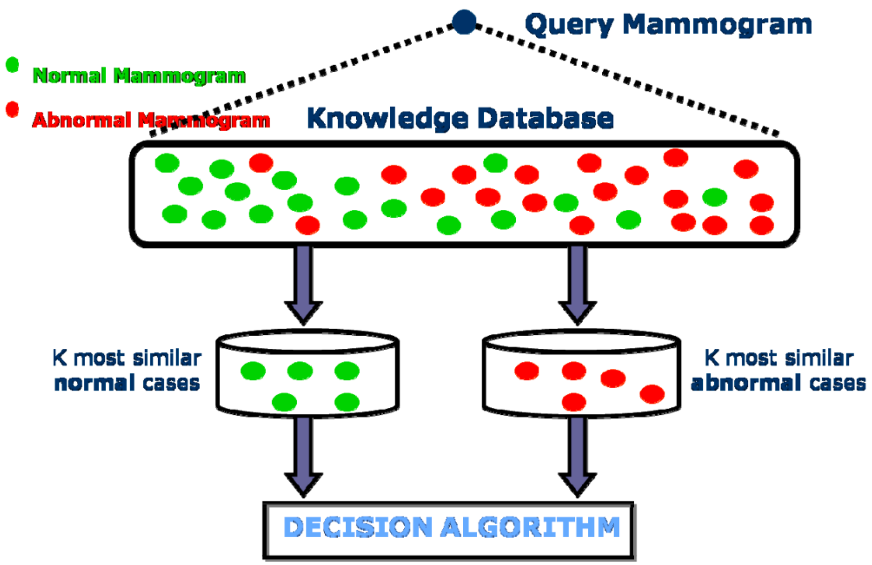
  - r_a = mean rating for user *a*
  - $u_1, \ldots, u_k$ are the *k*-nearest-neighbors to *a*
  - $r_{u,i}$ = rating of user u on item *I*
  - *sim*(*a*,*u*) = Pearson correlation between *a* and *u*

- This is a weighted average of deviations from the neighbors' mean ratings (and closer neighbors count more)

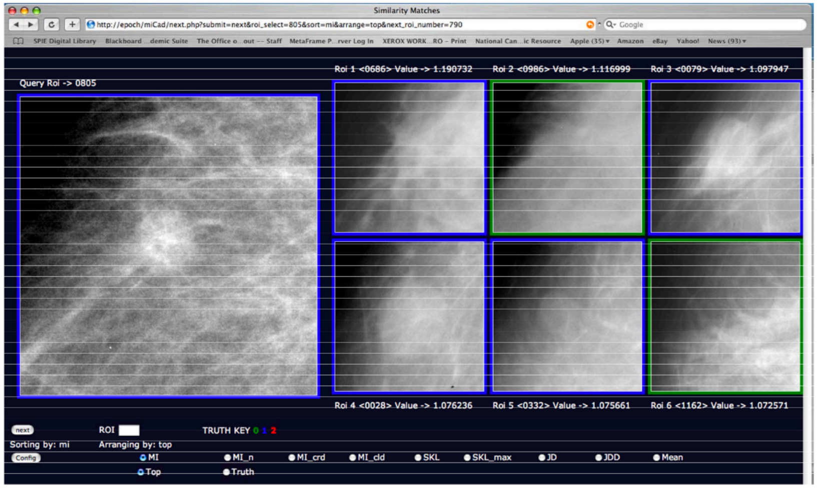
KNN Example: Computer-Aided Diagnosis

KNN for Medical Diagnosis - Dr. Georgia Tourassi of Duke University Medical Center developed a diagnostic aid for breast cancer based on MBR. The system not only produces a diagnosis; it also shows the physician the neighboring cases on which the diagnosis was based.

The Basic Idea



Example for Malignant Tumor

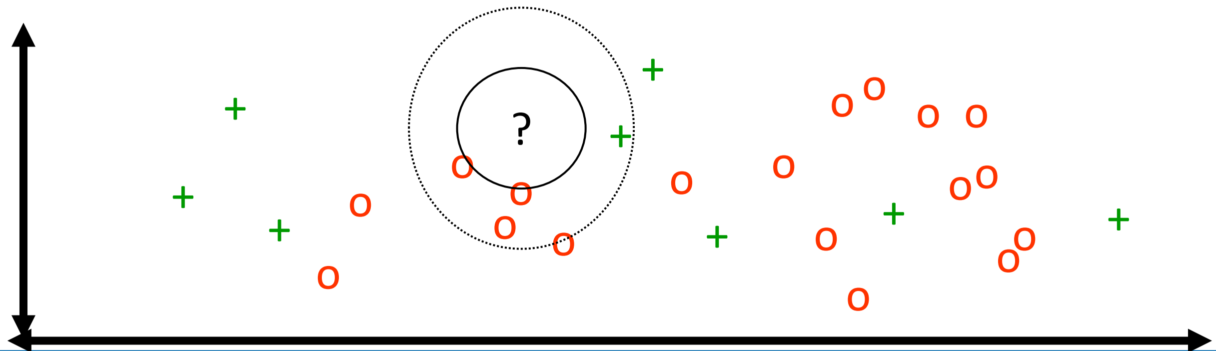


Six top retrieved cases (4/6 true masses)
Top retrieval is a malignant mass. (like the query)

Berry, M., and G. S. Linoff. "Data Mining Techniques: Theory and Practice Course Notes." *Cary, NC: The SAS Institute* (2009).

KNN Steps/Challenges

- Choosing appropriate historical data for use in training
- Choosing the most efficient way to represent the training data, then best "**closeness**" measure
- KNN is subject to the **curse of dimensionality** (i.e., presence of many irrelevant attributes) – again meaningful closeness to avoid counter intuitive results!
  - Tricks: Ignore features where both values are 0
- Choosing the (fast) distance function, voting function, and the number of neighbors – whose vote matters? How much?



Recall: Scale Effects

- Different features may have different measurement scales
  - e.g., patient weight in kg (range [50,200]) vs. blood protein values in ng/dL (range [-3,3])
- Consequences
  - Patient weight will have a much greater influence on the distance between samples
  - May bias the performance of the classifier
- Solution
  - Range and scale should be similar
  - Normalization or standardization (z-score)

KNN Weaknesses

- Remember to scale the data and reduce features
- Slow computation even though there are some tricks
  - Approximate distance computation
  - Ignore matches across all vectors etc.
- Requires storing all the data
- Does not give you an idea of the underlying structure of the data; you have no idea what an “average” or “exemplar” instance from each class looks like.

KNN Advantages

- Interpretability – Simple to understand and implement
- Complex boundaries
  - Makes no assumptions about the data (low bias)
- Fairly resistant to noise (with the right k, easy to tune)
- Ability to use data "as is" or almost just normalize the data but can work with pixels, ratings, …)
- Work with numeric and categorical data, all you need is a distance function!
- Works with classification and prediction
- **Often performs quite well, try it first on a new learning problem!**