Fundamentals of Computational Intelligence

Nearest Neighbors

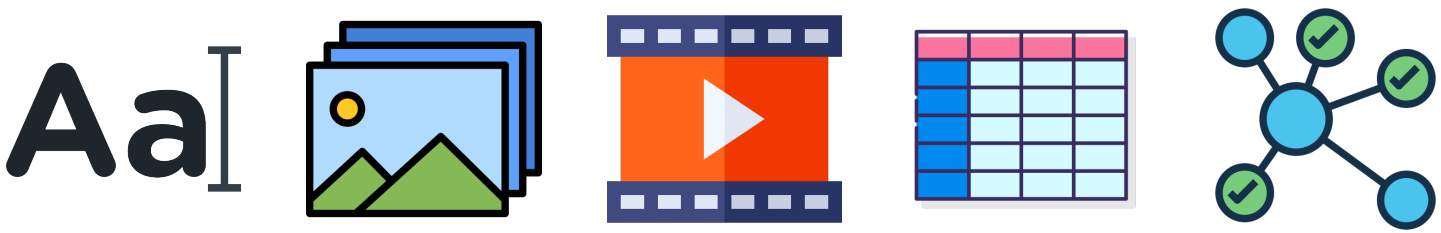
Mark Crowley

University of Waterloo
Department of Electrical and Computer Engineering

(based on slides provided by Amir-Hossein Karimi)

Preliminaries: Data and Dataset

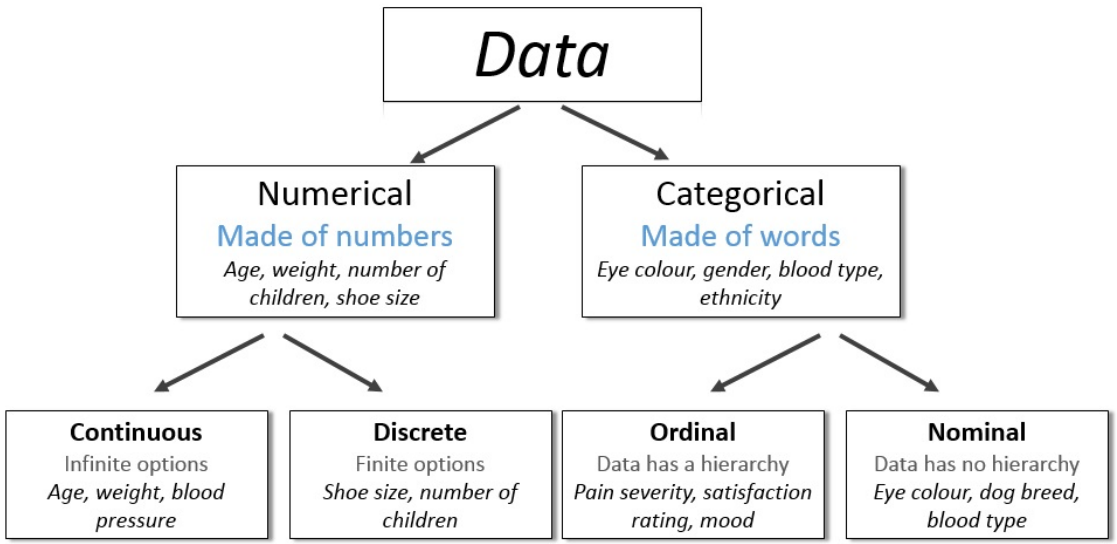
- *Data:* facts and statistics collected together into a form that is efficient for movement or processing

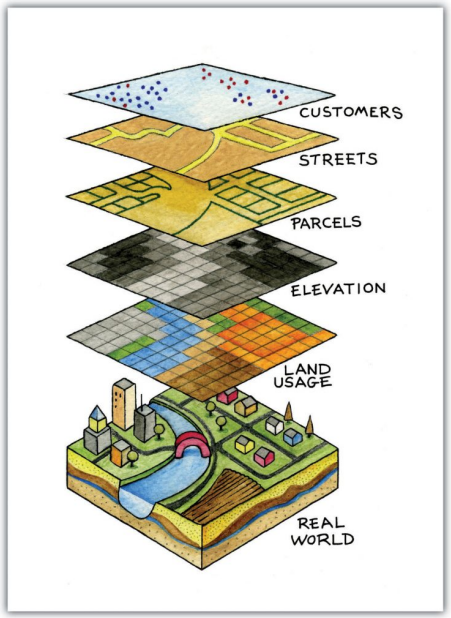


Different forms of data in the wild (l-r) text, images, videos, tabular, and graphs.

- *Dataset:* a structured *collection of data* generally associated with a unique body of work

Mixed Data Example: Geographical Information System (GIS)





Real-world data can be captured and summarized in different ways; all data are *approximations/summarizations* of underlying real-world phenomena. How does one characterize the planets in a solar system? 🤔 Answer in class.

Dataset cardinality and dimensionality

Data can capture

- too little information (e.g., approximations, as in digitization)
- too much information (e.g., redundant features or samples, as in temperature in C and F).

Datasets are characterized by

- *Cardinality:* the number of sample instances in the dataset
- *Dimensionality:* the number of features for each instance in the dataset

Big Data

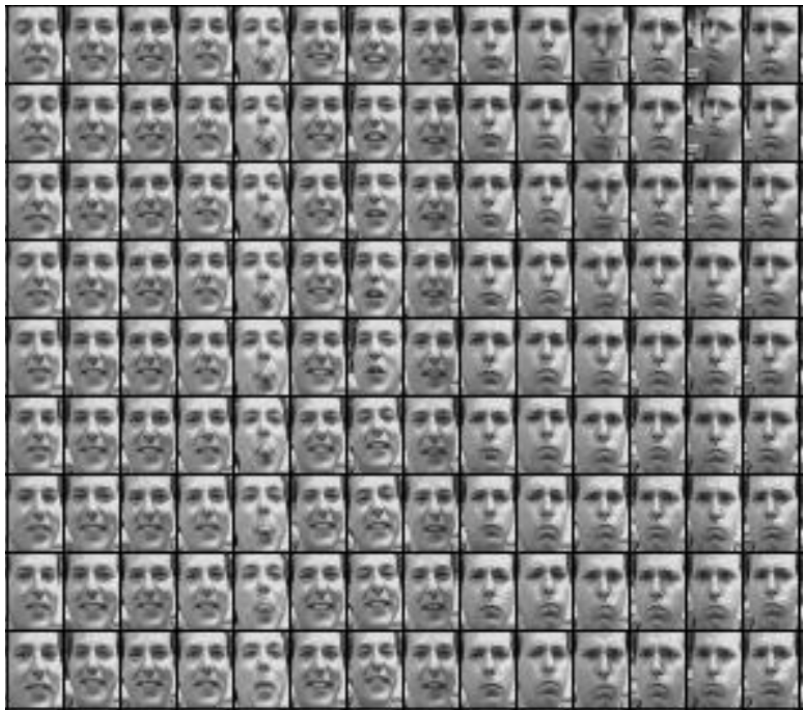
What is *Big Data*? 🤔 Answer in class.

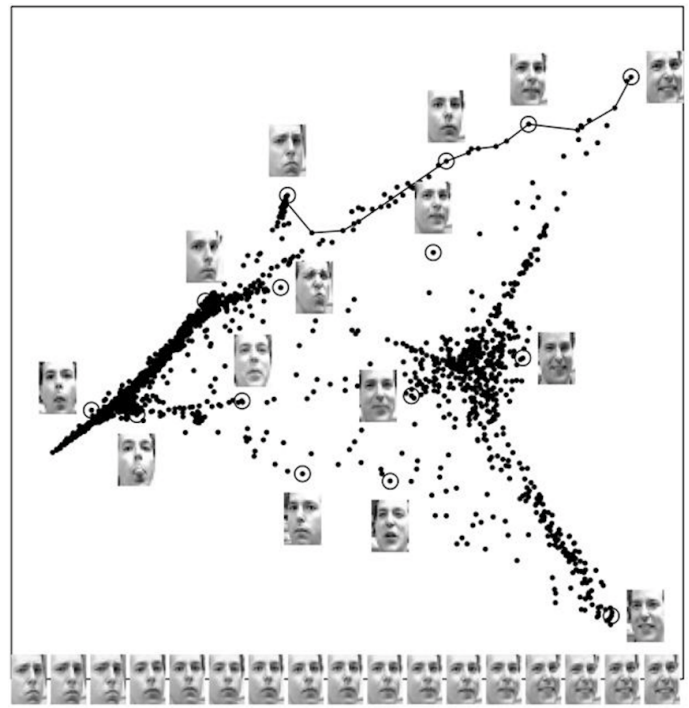
Example I - Redundant Samples or Dimensions

	Lion King	Top Gun I	Top Gun II	$\cdots$	Titanic
Susan	🤩	😑	😑	$\cdots$	❓
Akbar	🙃	🤩	❓	$\cdots$	🤢
Tom	🙄	❓	🙃	$\cdots$	🤢
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Leily	❓	🙄	🤢	$\cdots$	🤩

Netflix data contains $480{,}189$ users $\times$ $17{,}770$ movies. High-dimensional datasets likely contain redundant information which can be exploited for learning or recommendation (e.g., the similarity of movies or individuals' preferences). Which movies/individuals are similar? 🤔 Answer in class.

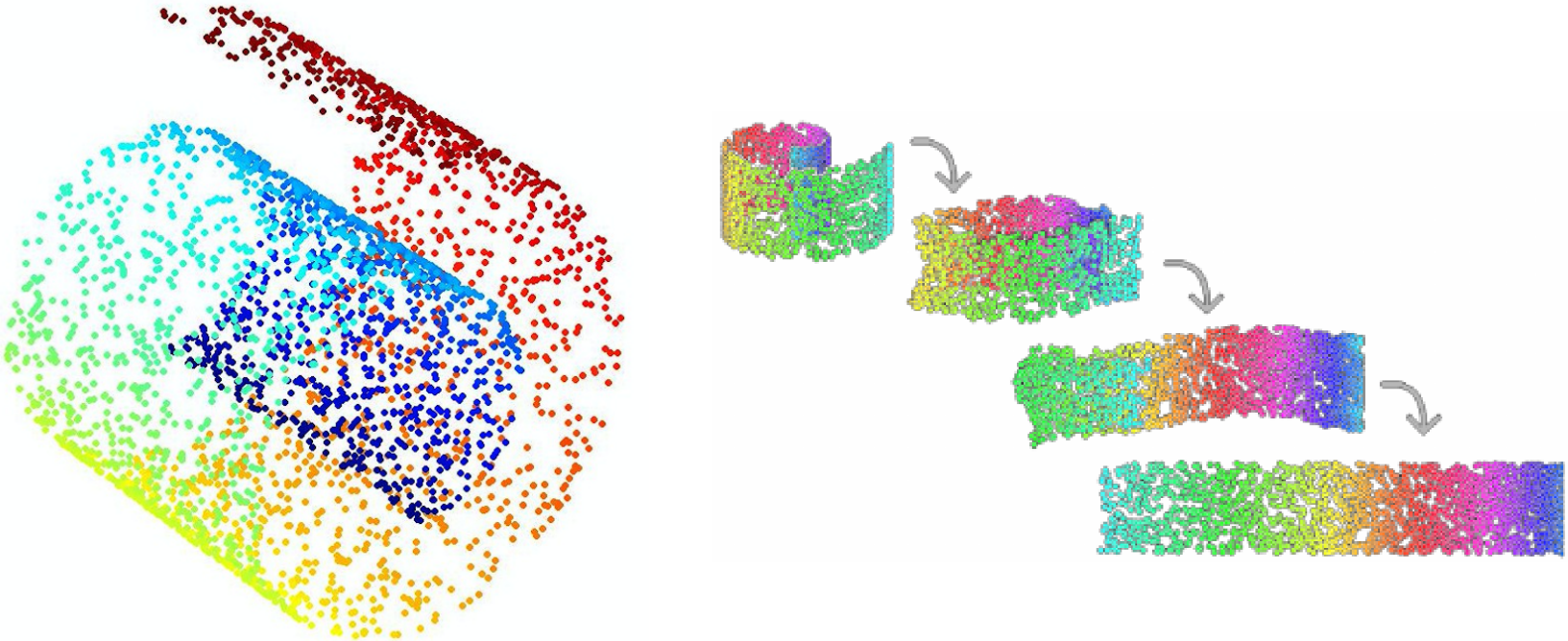
Example II - Degrees of Variations





(left) Although images are high dimensional vectors, their degrees of variation are small (e.g., angle/orientation/expression, but not relative positioning of nose and eyes). *(right)* Dimensionality reduction applied to high-dimensional data shows lower-dimensional patterns. Any patterns? 🤔 Answer in class.

Example III - Variety in Representation



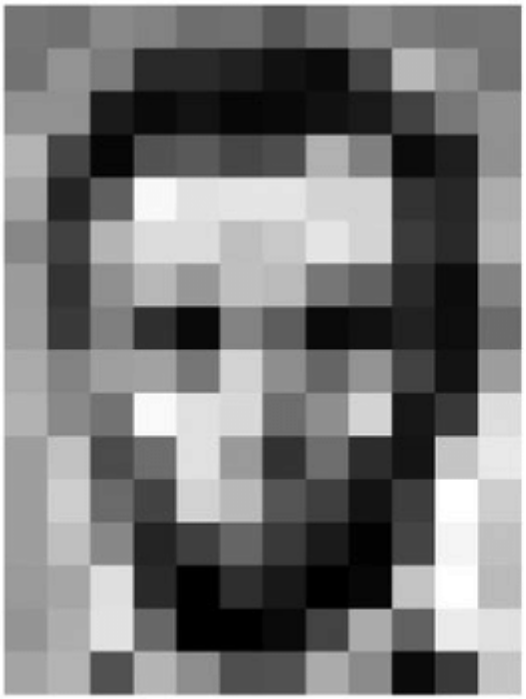
Datasets are often generated by unknown/complicated processes. Dimensionality reduction can help to *unfold* complex manifolds, to explain the local geometry of the space. Which is better; 2D or 3D? 🤔 Answer in class.

How is Data Represented?

- Machine learning algorithms need to *handle lots of types of data*: images, text, audio waveforms, credit card transactions, etc.
- *Representation*: means of measuring, comparing, and transforming data on a computer
- Common strategy: represent data in *Euclidean space* as vectors: $\mathbb{R}^d$
- *Vectors* are a great representation since we can do linear algebra!

How does one convert a natural image to a representation suitable for machine learning? 🤔 Answer in class.

Media Digitization

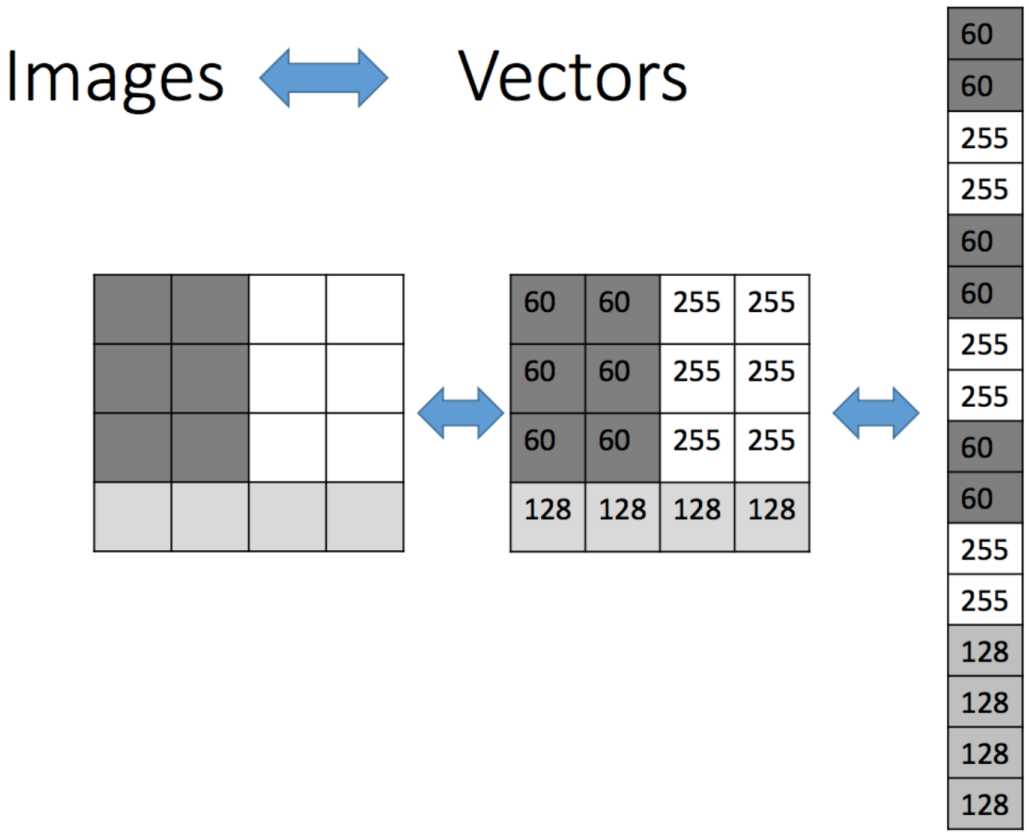


157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

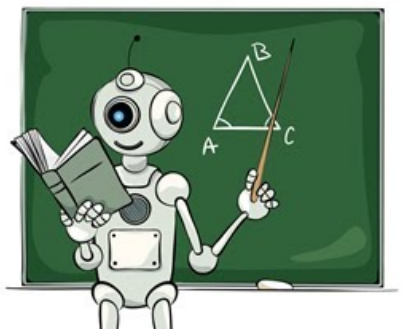
Media (e.g., images) can be converted to numbers for a computer to process. Once again, digitization is an approximation of the underlying data (e.g., what resolution to capture? how many bit channels to use?)

Image Vectorization



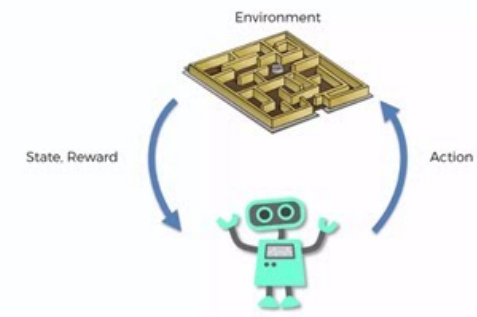
Training Dataset

- Today (and for much of this course) we focus on *supervised learning*.



VS

VS



	Supervised	*Unsupervised*	*Reinforced*
Label	✔	✘	✘
Goal	Match labels	Uncover patterns	Maximize reward
Example	Spam Detection	Customer Segmentation	Chess A

- This means we are given a *training set* consisting of data tuples consisting of (*input*, *label*), e.g.,

Training Dataset

- Today (and for much of this course) we focus on *supervised learning*.
- This means we are given a *training set* consisting of data tuples consisting of (*input*, *label*), e.g.,

Task	**Inputs**	**Labels**
object recognition	image	object category
image captioning	image	caption
document classification	text	document category
speech-to-text	audio waveform	🤔Answer in class.
⋮	⋮	⋮

Input Vectors

- Mathematically, our training set consists of a collection of pairs of an *input vector* $\mathbf{x} \in \mathbb{R}$ and its corresponding *target*, or *label*, t
  - *Regression*: t is an element of continuous reals (e.g. stock price)
  - *Classification*: t is an element of a discrete set $\{1, \ldots, C\}$
  - These days, t is often a highly structured object (e.g., image)
- Denote the *training set* $\mathcal{D}_{tr} = \{(\mathbf{x}^{(1)}, t^{(1)}), \ldots, (\mathbf{x}^{(N)}, t^{(N)})\}$
  - Note: these superscripts are indices, and have nothing to do with exponentiation!
- Later:
  - *testing set* $\mathcal{D}_{ts} = \{(\mathbf{x}^{(1)}, t^{(1)}), \ldots, (\mathbf{x}^{(M)}, t^{(M)})\}$
  - *validation set* $\mathcal{D}_{vl} = \{(\mathbf{x}^{(1)}, t^{(1)}), \ldots, (\mathbf{x}^{(P)}, t^{(P)})\}$

Nearest Neighbors

- Suppose we're given
  - a training set $\mathcal{D}_{tr} = \{(\mathbf{x}^{(1)}, t^{(1)}), \ldots, (\mathbf{x}^{(N)}, t^{(N)})\}$
  - a (test) input vector $\hat{\mathbf{x}}$ we'd like to classify. How? 🤔
- 💡 Idea: find the nearest input vector to $\hat{\mathbf{x}}$ in the training set and copy its label. What does "nearest" mean? 🤔
- Can formalize "nearest" in terms of Euclidean distance:

$$\text{distance}(\mathbf{x}^{(a)}, \mathbf{x}^{(b)}) = \|\mathbf{x}^{(a)} - \mathbf{x}^{(b)}\|_2 = \sqrt{\sum_{j=1}^{d} \left(x_j^{(a)} - x_j^{(b)}\right)^2}$$

What is distance$([0,1]^T, [2,1]^T)$? 🤔 Answer in class. What other distance measures? 🤔 Answer in class.

ℓ_p-norms

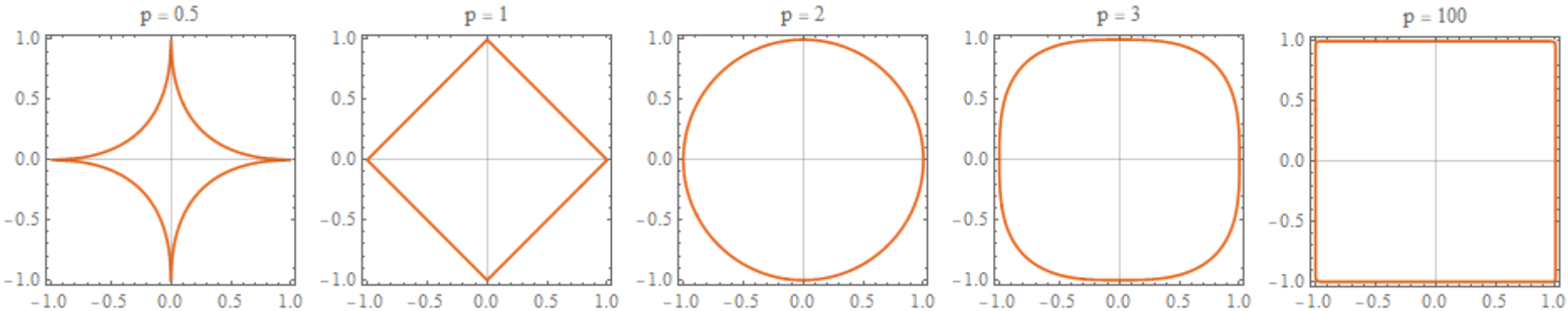
By definition, if $\mathbf{x}$ is a vector such that $\mathbf{x} = (x_1, \cdots, x_n)$, then:

$$\ell_p - \text{norm} := ||\mathbf{x}||_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}$$

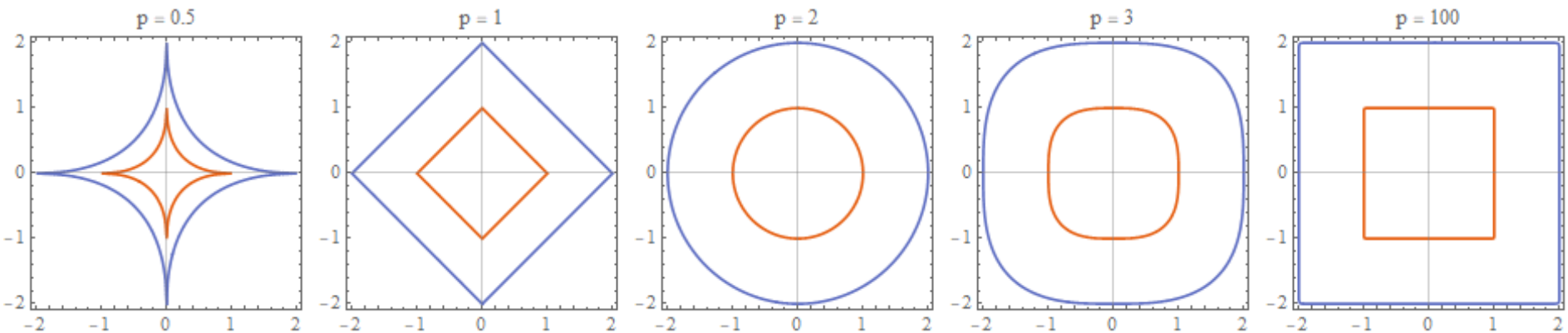
- $p = 1$: 🤔 Answer in class.
- $p = 2$: 🤔 Answer in class.
- $p \to \infty$: 🤔 Answer in class.

ℓ_p-norms

In 2 dimensions, identify and plot coordinates (x_1, x_2) that satisfy $\ell_1(\mathbf{x}) = 1$.
Repeat for $\ell_2(\mathbf{x}) = 1$

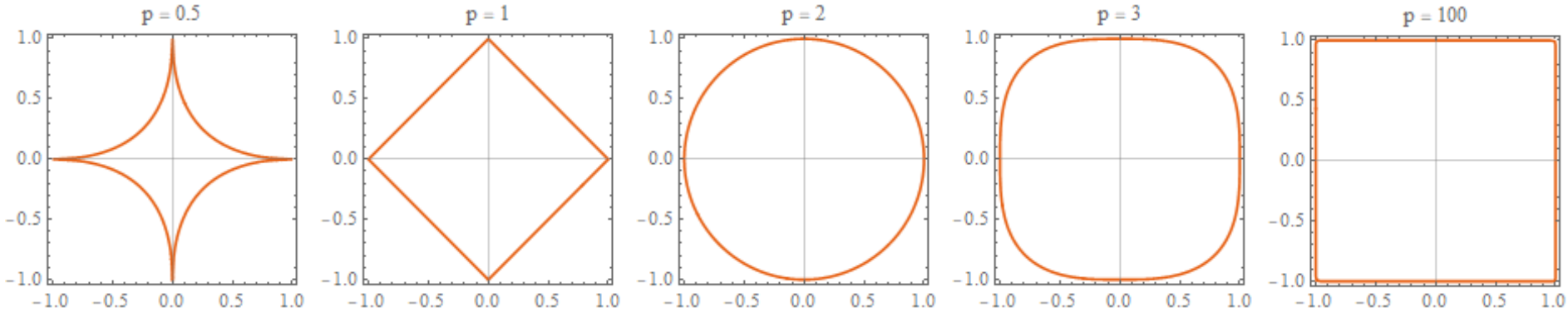


Depiction of ℓ_p-norms $= 1$ in $\mathbb{R}^2$ for various values of p.

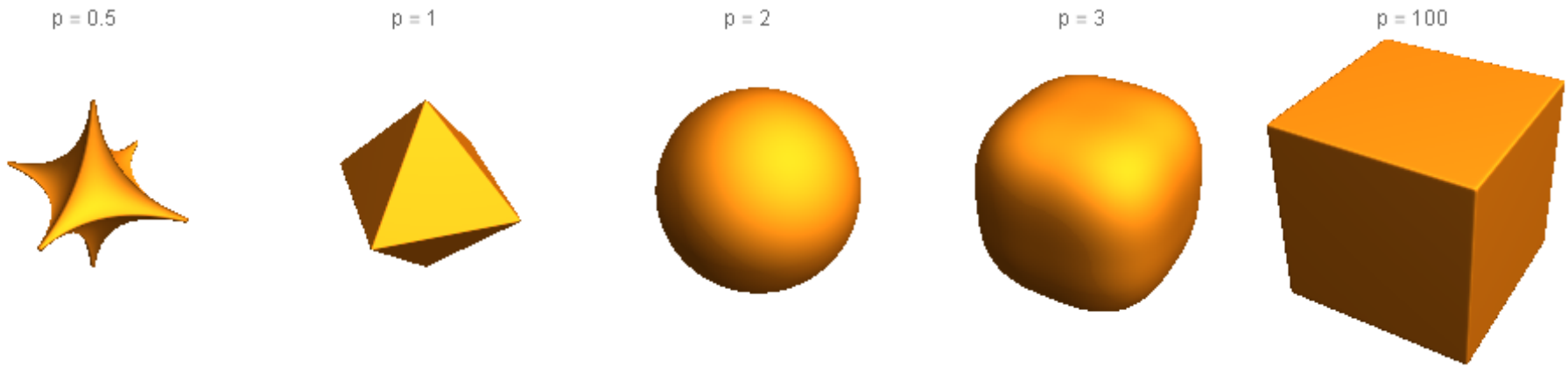


Depiction of ℓ_p-norms in $\mathbb{R}^2$ for various values of p; $\ell_p = ?$ 🤔 Answer in class.

ℓ_p-norms



Depiction of ℓ_p-norms $= 1$ in $\mathbb{R}^2$ for various values of p.



Depiction of ℓ_p-norms in $\mathbb{R}^3$ for various values of p; $\ell_p = 1$.

Homework (ungraded): can p take on values smaller than 1? Would it still satisfy the triangle equality?

Cosine Similarity

$$\text{cosine similarity}(\mathbf{a}, \mathbf{b}) := \cos(\theta) = \frac{\langle \mathbf{a}, \mathbf{b}, \rangle}{||\mathbf{a}||.||\mathbf{b}||} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \sqrt{\sum_{i=1}^{n} b_i^2}}$$

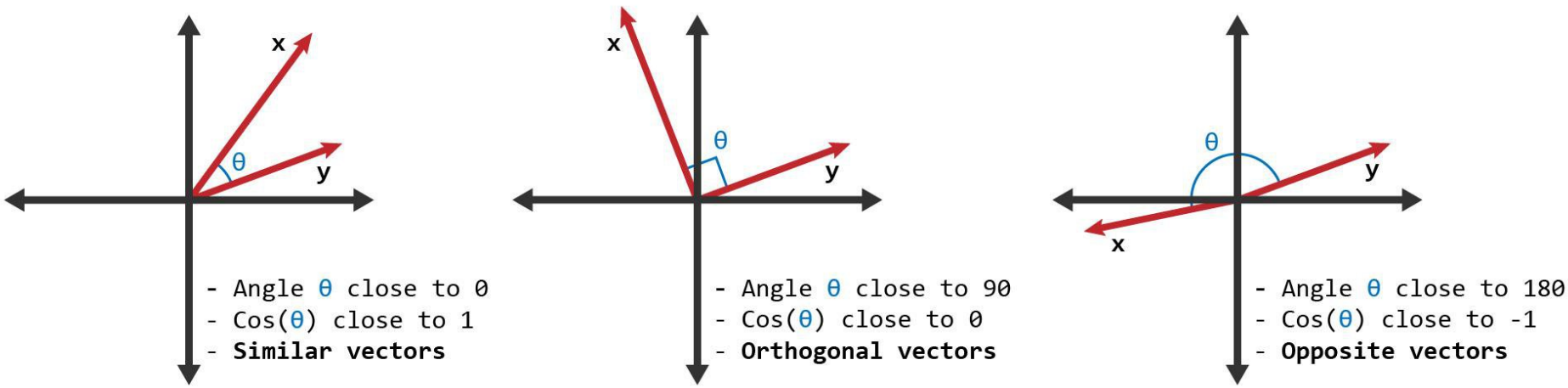


Image source

- *Cosine similarity* looks at the *angle* between *two vectors*
- *Euclidean similarity* at the *distance* between *two points*.
- When would you use cosine similarity over Euclidean similarity? 🤔

Cosine Similarity

$$\text{cosine similarity}(\mathbf{a}, \mathbf{b}) := \cos(\theta) = \frac{\langle \mathbf{a}, \mathbf{b}, \rangle}{||\mathbf{a}||.||\mathbf{b}||} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \sqrt{\sum_{i=1}^{n} b_i^2}}$$

Let's say you are in an e-commerce setting and you want to compare users for product recommendations:

- User 1 bought 1x eggs, 1x flour and 1x sugar.
- User 2 bought 100x eggs, 100x flour and 100x sugar
- User 3 bought 1x eggs, 1x Vodka and 1x Red Bull

Who is similar based on which metric? 🤔

- Answer in class.
- Answer in class.

Nearest Neighbors

Algorithm (Nearest Neighbors):

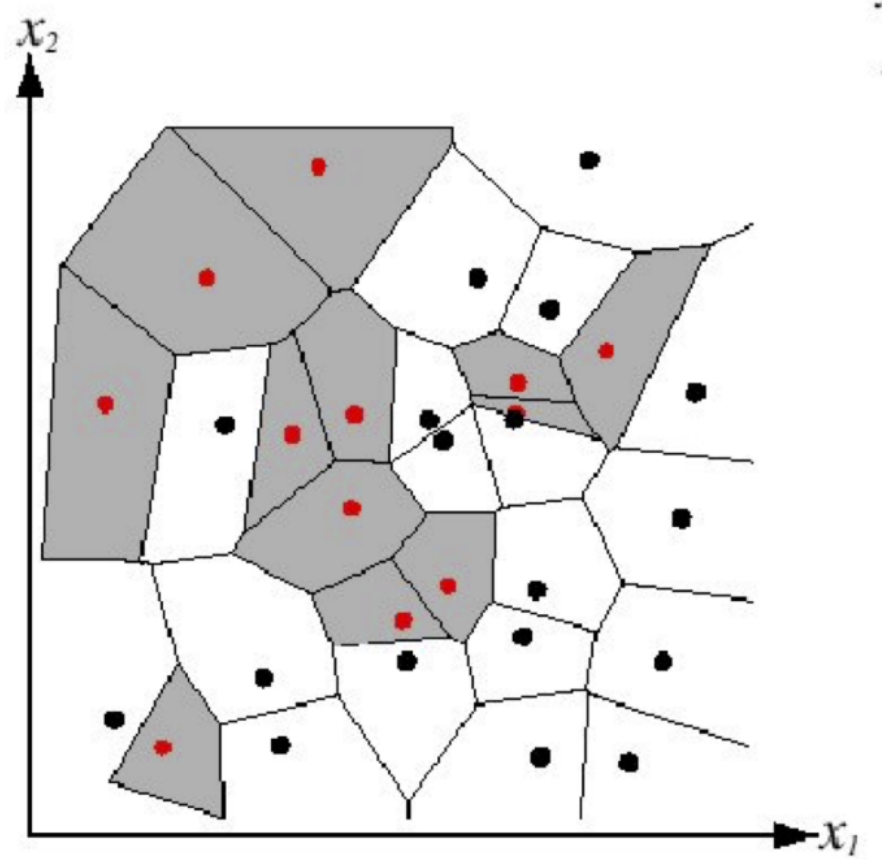
1. Find example $(\mathbf{x}^n, t^n)$ (from training set $\mathcal{D}_{tr}$) closest to $\hat{\mathbf{x}}$, i.e.,

$$\mathbf{x}^n = \underset{i \in \{1,\ldots,n\}}{\text{argmin}} \quad \text{distance}(\mathbf{x}^n, \hat{\mathbf{x}})$$

2. Output $\hat{y} = t^n$

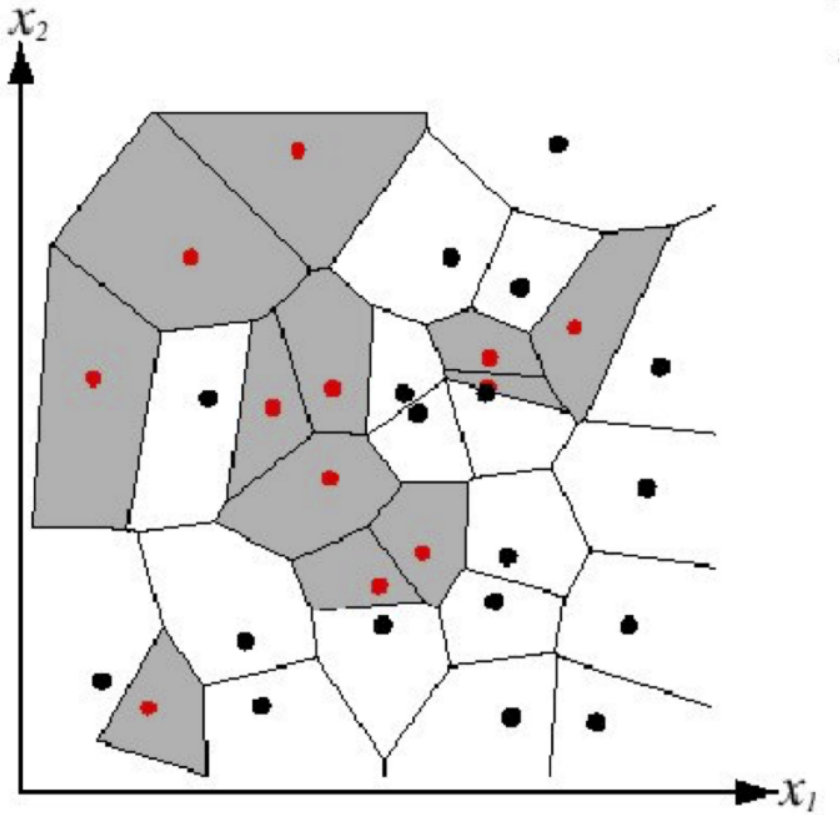
Nearest Neighbors: Decision Boundaries

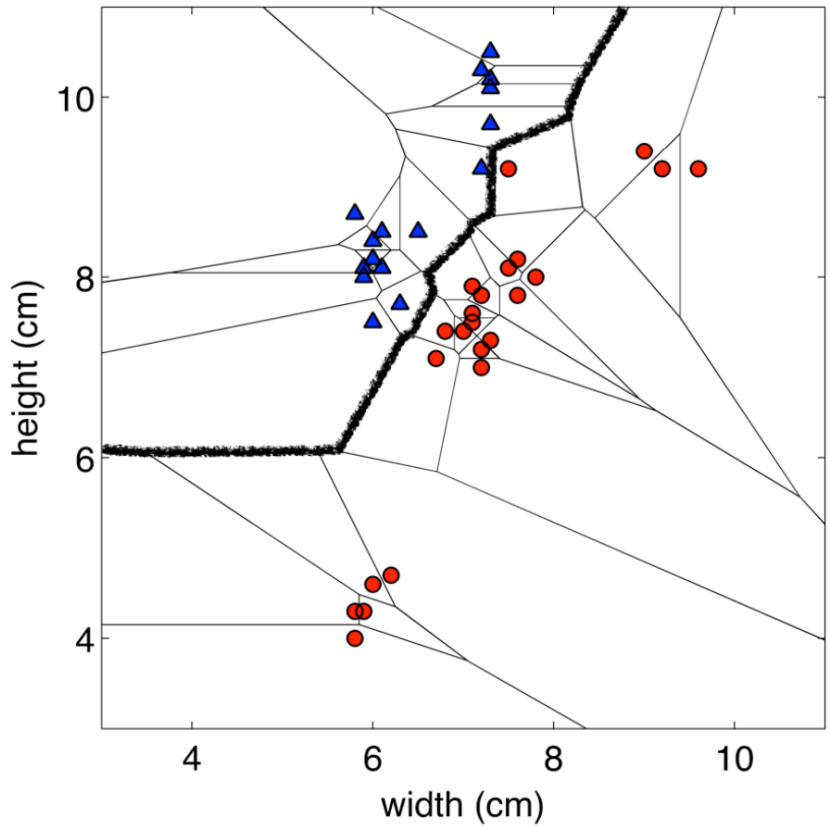
We can visualize the behavior in the classification setting using a *Voronoi diagram*. Each region is shaded according to the target class of the centroid of that region. How many target classes do we have here? 🤔 Answer in class.



Nearest Neighbors: Decision Boundaries

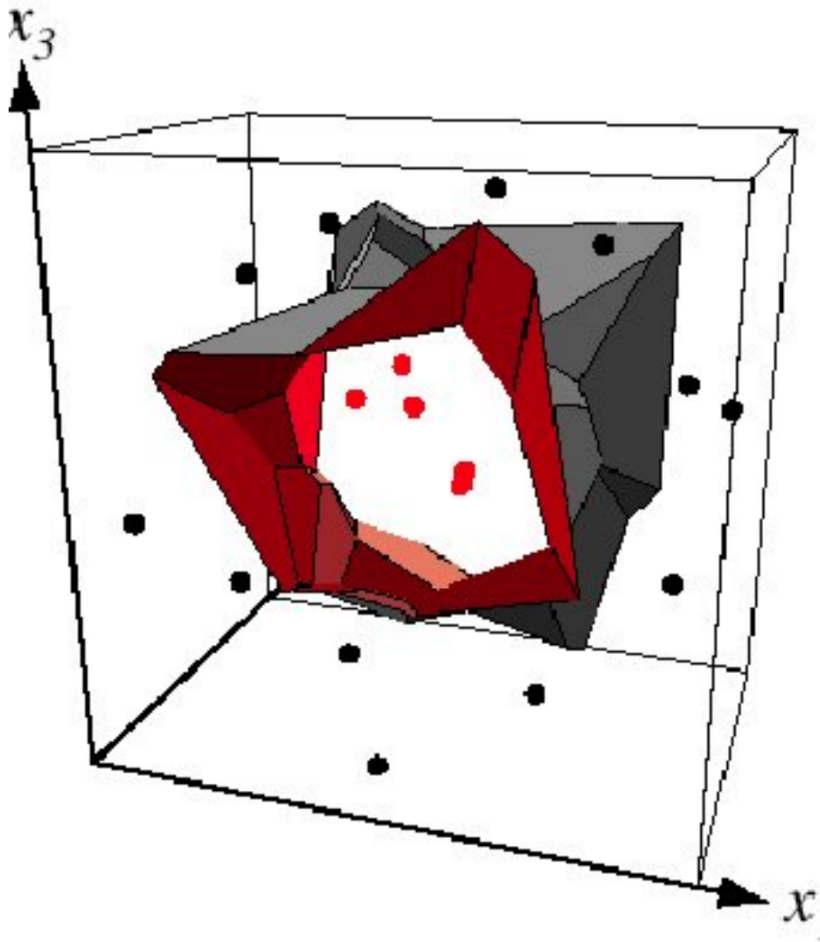
Decision boundary: the boundary between regions of input space assigned to different categories.





What is a *Decision Region?* 🤔 Answer in class.

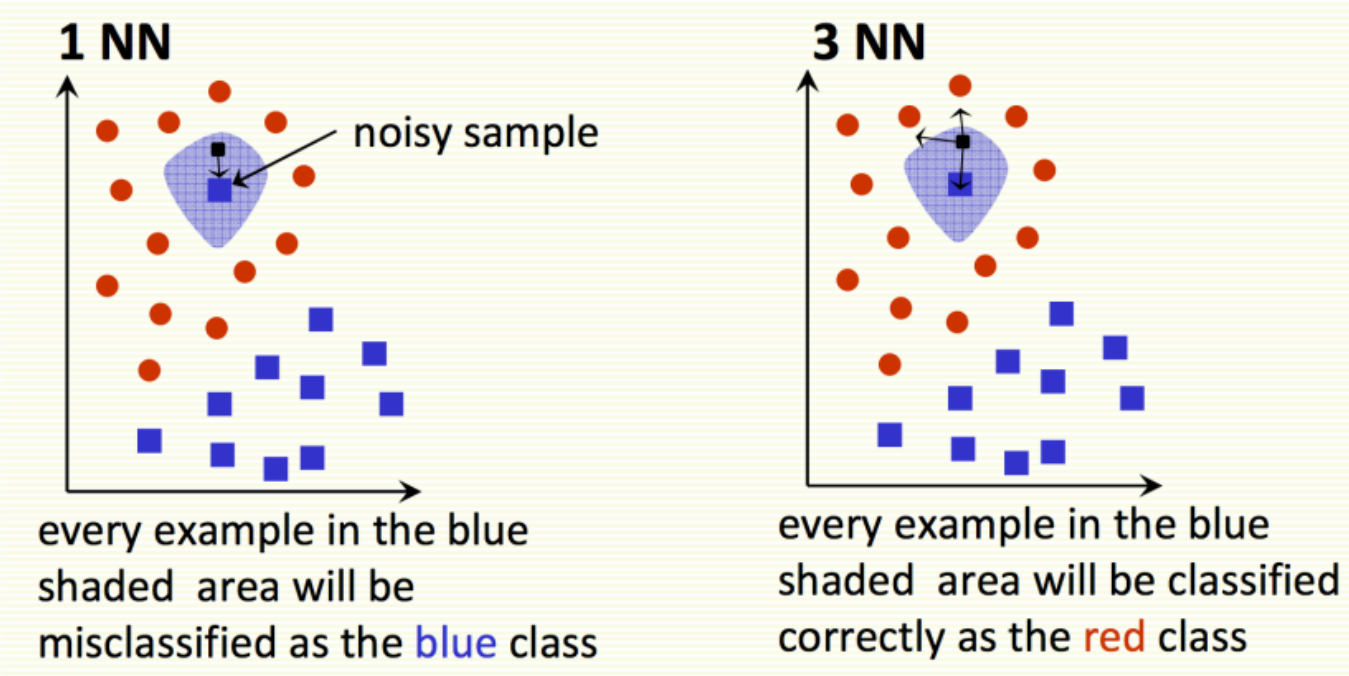
Nearest Neighbors: Decision Boundaries



Example: 2D decision boundary in 2D space.

(1-)Nearest Neighbors: Pitfalls

(1-)Nearest Neighbors decision regions may be sensitive to noise or mis-labeled data ("class noise"). Solution? 🤔 Answer in class.



[Pic by Olga Veksler]

K-Nearest Neighbors

- How to smoothen the decision regions of (1-)Nearest Neighbors? 🤔
  Answer in class.

Algorithm (K-Nearest Neighbors):

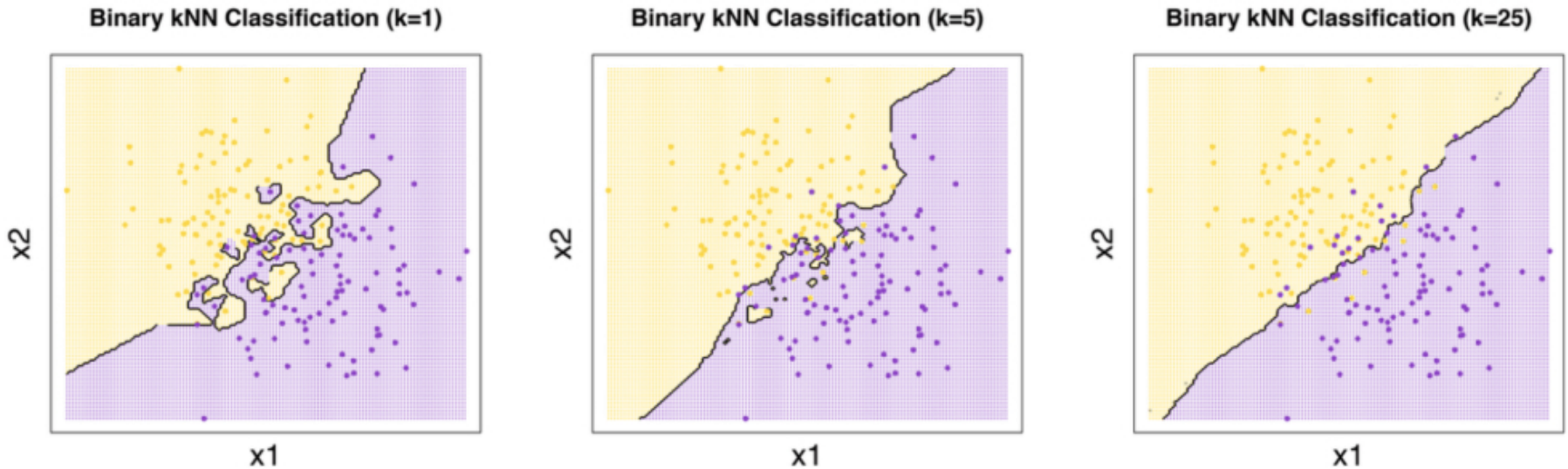
1. Find k examples from training set $\mathcal{D}_{tr}$ closest to test input $\hat{\mathbf{x}}$
2. Classification output is majority class:

$$\hat{y} = \underbrace{\underset{t^{(x)} \in \mathcal{L}}{\arg\max}}_{\text{Go through the class labels}} \underbrace{\sum_{i=1}^{k} \mathbb{I}(t^{(z)} = t^{(i)})}_{\text{Count how often a label appears in the k-nearest neighbor set}}$$

where $\mathcal{L}$ is the set of all class labels.

$\mathbb{I}\{statement\}$ is the identity function and is equal to one whenever the statement is true. We could also write this as $\delta(t^{(z)}, t^{(i)})$, with $\delta(a, b) = 1$ if $a = b$, 0 otherwise.

K-Nearest Neighbors



What is a good value for k? Any trade-offs? 🤔

Image source

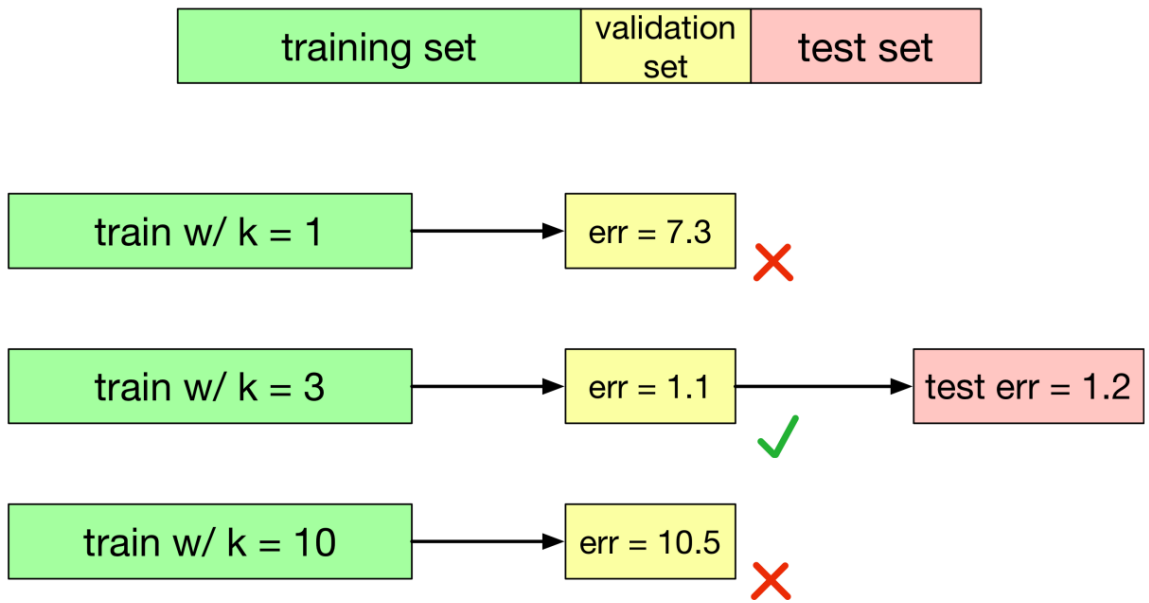
K-Nearest Neighbors

Tradeoffs in choosing k?

- Small k
  - Good at capturing fine-grained patterns
  - May *overfit*, i.e. be sensitive to random idiosyncrasies in the training data (e.g., outliers)
- Large k
  - Makes stable predictions by averaging over lots of examples
  - May *underfit*, i.e. fail to capture important regularities
- Balancing k
  - Optimal choice of k depends on number of data points n (not number of dimensions d nor number of classes c).
  - Nice theoretical properties if $k \to \infty$ and $\frac{k}{n} \to 0$.
  - Rule of thumb: choose $k < \sqrt{n}$.
  - We can choose k using validation set....

Validation and Test Sets

- k is an example of a *hyperparameter*
- Whereas a *learning algorithm*'s' *hyperparameter* is *selected* prior to training which controls the flow of training, a *model*'s *parameter* is *learned / tuned* during training and becomes intrinsic to the model
- We can select hyperparameters using a *validation set*:



What is a good value for k?

- The *test set* is *used only at the very end*, to measure the generalization performance of the final configuration; not to guide selection of k.

A Note on Generalization

- *Generalization*: a model's ability to adapt to new, previously unseen data, from the same distribution as the training set
- *The aim of learning*: so our algorithm *generalize* to unseen data.
- We can measure the *generalization error* using a *test set*.
- Selecting k based on *training error* is a bad idea. Why? 🤔 Answer in class.
- Bayes Error is a lower-bound on error (optimal error a model can have).

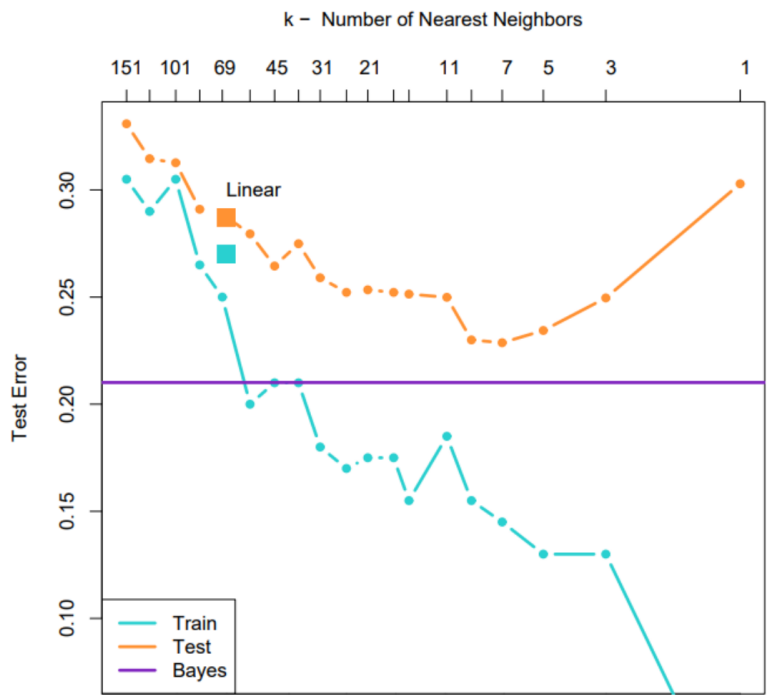


Image credit: "The Elements of Statistical Learning"

Training error is the error you'll have when you input your training set to your KNN as test set.

Validation and Test Sets

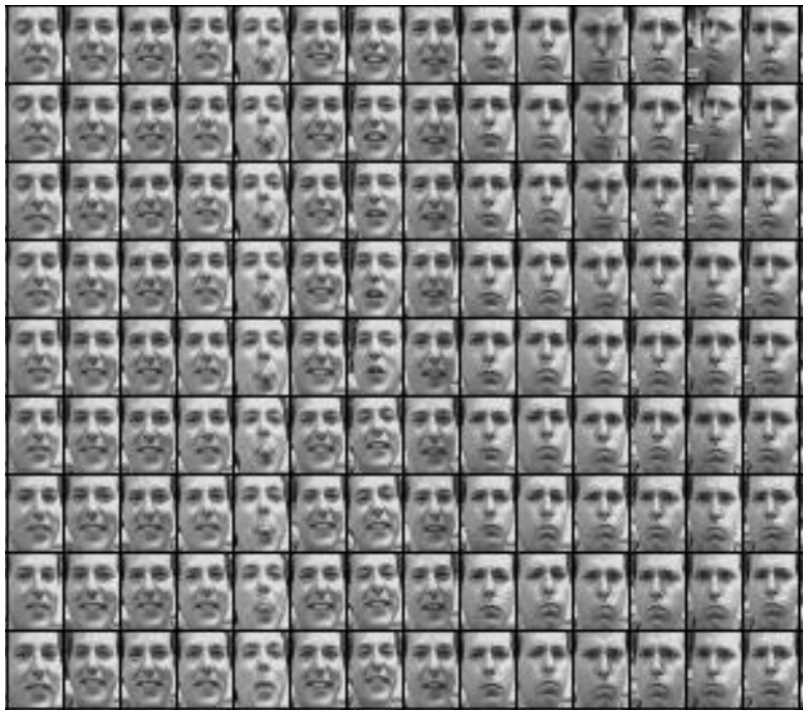
Selection of k in K-NN is not to be confused with *k-fold Cross Validation*; to be studied later...

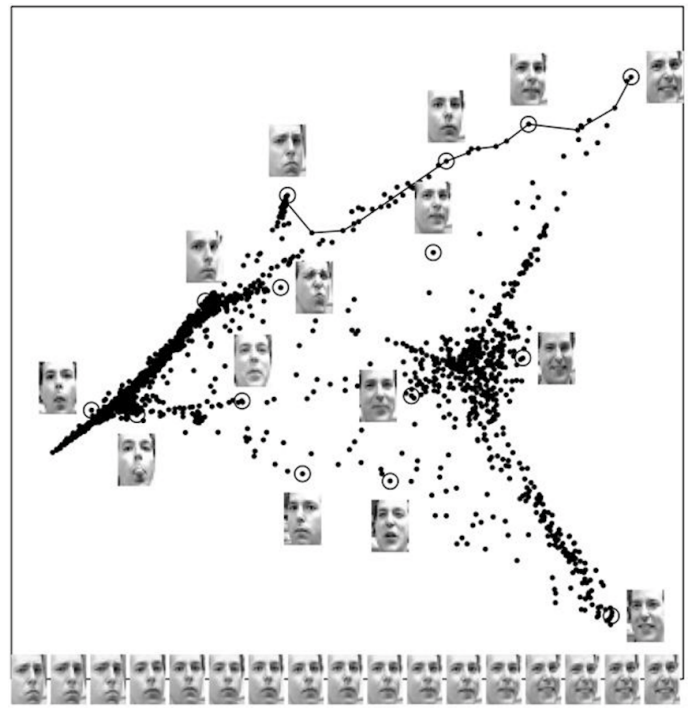
So now, we've selected k. Are there pitfalls with K-Nearest Neighbors? 🤔

Pitfalls: The Curse of Dimensionality

- Nearest neighbors intuition says that *if two points are close in input space, their outputs must be close* (the same).
- In high dimensions, "most" points are approximately the same distance (see assignment 2)
- As dimension increases
  - all points appear equi-distant; and
  - the nearest neighbor will compute distances based on more (potentially irrelevant) dimensions.
- So how do we select the label for a test point? 🤔 Answer in class.

Example II - *Few* Degrees of Variations

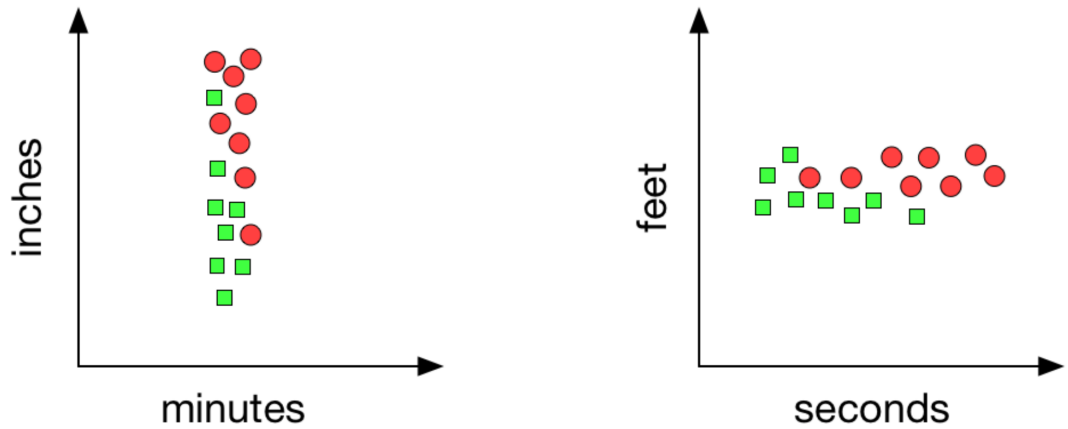




(left) Although images are high dimensional vectors, their degrees of variation are small (e.g., angle/orientation/expression, but not relative positioning of nose and eyes). *(right)* Dimensionality reduction applied to high-dimensional data shows lower-dimensional patterns.

Pitfalls: Normalization & Standardization

- Nearest neighbors can be sensitive to the ranges of different features.
- Often, the units are arbitrary:



- Simple fix: *standardize* each dimension to be *zero mean* and *unit variance*, i.e., compute the mean μ_j and std. deviation σ_j , and take

$$\tilde{x}_j = \frac{x_j - \mu_j}{\sigma_j} \tag{1}$$

- Caution: depending on the problem, the scale might be important!
- Caution: when scale is important, distance measure is important!

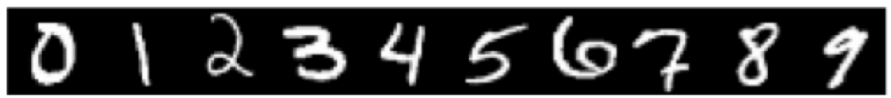
Pitfalls: Computational & Storage Cost

- Computational req. at *training time*? 🤔 Answer in class.
- Computational req. at *test time*, per query (naïve algorithm)? 🤔
  - Calculate D-dim Euclidean distances between $\hat{\mathbf{x}}$ and $\mathbf{x}^{(i)}$? 🤔 Answer in class.
  - Calculate D-dim Euclidean distances between $\hat{\mathbf{x}}$ and $\mathbf{x}^{(j)}\ \forall\ j$, i.e., all of training set? 🤔 Answer in class.
  - Sort the distances to find closest k instances? 🤔 Answer in class.
  - This *must be done for each query*, which is very expensive by the standards of a learning algorithm!
- Storage req.? 🤔 Answer in class.
- Tons of work has gone into algorithms and data structures for efficient nearest neighbors with high dimensions and/or large datasets (see, e.g., approximate K-NN)

Example: Digit Classification

KNN can perform decently on standard image datasets, with some tricks here and there...

- a good *preprocessing* step, e.g., to achieve *invariance to image transformations*, one may tried to *warp* one image to match the other image
- a good similarity measure, e.g., average distance between corresponding points on warped images



Yann LeCunn – MNIST Digit Recognition

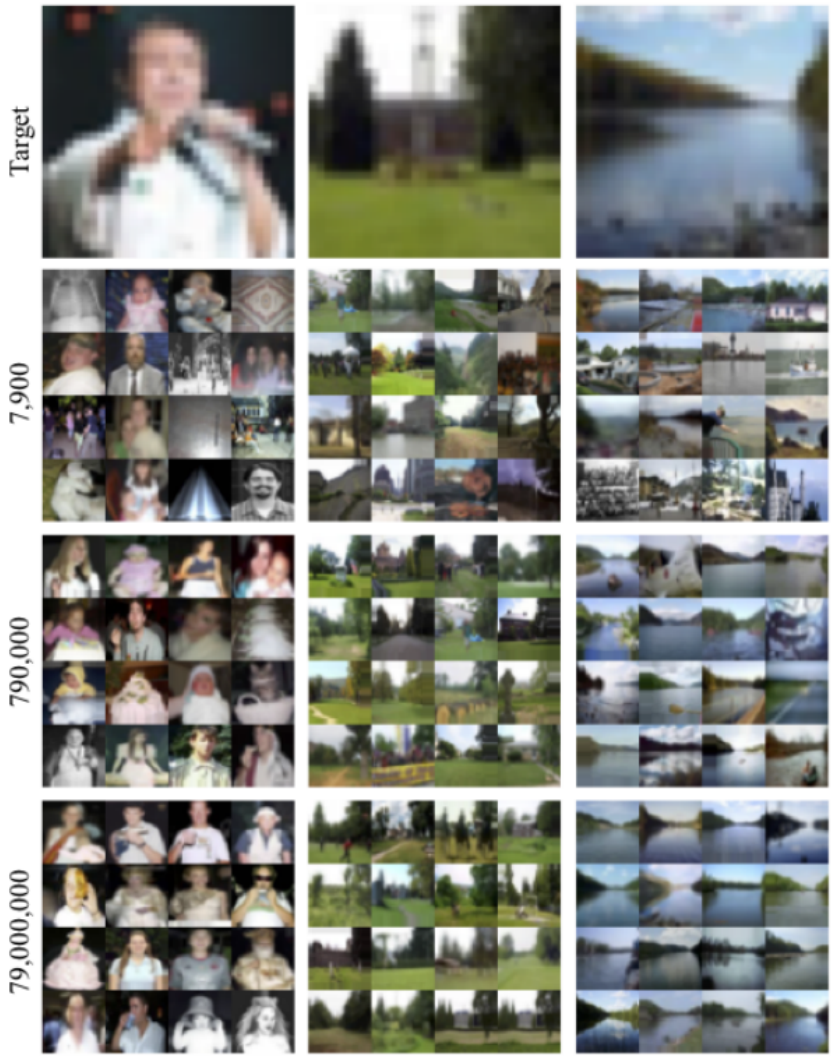
- Handwritten digits
- 28x28 pixel images: $d = 784$
- 60,000 training samples
- 10,000 test samples

Nearest neighbour is competitive

	Test Error Rate (%)
Linear classifier (1-layer NN)	12.0
K-nearest-neighbors, Euclidean	5.0
K-nearest-neighbors, Euclidean, deskewed	2.4
K-NN, Tangent Distance, 16x16	1.1
K-NN, shape context matching	0.67
1000 RBF + linear classifier	3.6
SVM deg 4 polynomial	1.1
2-layer NN, 300 hidden units	4.7
2-layer NN, 300 HU, [deskewing]	1.6
LeNet-5, [distortions]	0.8
Boosted LeNet-4, [distortions]	0.7

Example: 80 Million Tiny Images

- Although more expensive to store and compute, more images in the dataset generally improve semantic performance.
- 80 Million Tiny Images was the first extremely large image dataset. It consisted of color images scaled down to 32×32.
- Note: this required a carefully chosen similarity metric.
- On a workstation (i7-4790, Quadcore + Hyperthreading) how long would a single search take?
  🤔 Answer in class. (Blog)



Dataset collected and then deprecated by Antonio Torralba, Rob Fergus, and Bill Freeman.

Summary

- Simple algorithm that *does all its work at test time*; in a sense, no learning (aka. *lazy learning*)!
- Suffers from several challenges, including
  - the *Curse of Dimensionality*
  - *normalization* challenges, and
  - *high space* and *(test) runtime* requirements
- K-Nearest Neighbors is an example of a *non-parametric model*; can *control the complexity* by varying *hyperparameter* k.
- Next (next) time: parametric models, which learn a compact summary of the data rather than referring back to it at test time.