GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

CROSS-VALIDATION

GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

K-NEAREST NEIGHBORS

I. CLASSIFICATION WITH K-NEAREST NEIGHBORS
II. DISTANCE MEASURES
III. THE CURSE OF DIMENSIONALITY

CLASSIFICATION WITH KNN

	continuous	*categorical*
supervised	???	???
unsupervised	???	???

	continuous	*categorical*
supervised	regression	classification
unsupervised	dimension reduction	clustering

- Sounds scary, but we work with it every day!
- Here is some 5-dimensional data:

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	*I. setosa*
4.9	3.0	1.4	0.2	*I. setosa*
4.7	3.2	1.3	0.2	*I. setosa*
4.6	3.1	1.5	0.2	*I. setosa*
5.0	3.6	1.4	0.2	*I. setosa*
5.4	3.9	1.7	0.4	*I. setosa*
4.6	3.4	1.4	0.3	*I. setosa*
5.0	3.4	1.5	0.2	*I. setosa*

- Sounds scary, but we work with it every day!

- Here are four features with a target we are trying to build a predictor for:

Target

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	*I. setosa*
4.9	3.0	1.4	0.2	*I. setosa*
4.7	3.2	1.3	0.2	*I. setosa*
4.6	3.1	1.5	0.2	*I. setosa*
5.0	3.6	1.4	0.2	*I. setosa*
5.4	3.9	1.7	0.4	*I. setosa*

- Sounds scary, but we work with it every day!

- Here are four features with a target we are trying to build a predictor for:

```
data =
  [((5.1, 3.5, 1.4, 0.2), 0),
   ((4.9, 3.0, 1.4, 0.2), 0),
   ((4.7, 3.2, 1.3, 0.2), 0),
   ...
  ]
```

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	*I. setosa*
4.9	3.0	1.4	0.2	*I. setosa*
4.7	3.2	1.3	0.2	*I. setosa*
4.6	3.1	1.5	0.2	*I. setosa*
5.0	3.6	1.4	0.2	*I. setosa*
5.4	3.9	1.7	0.4	*I. setosa*

Here's (part of) an example dataset:

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	*I. setosa*
4.9	3.0	1.4	0.2	*I. setosa*
4.7	3.2	1.3	0.2	*I. setosa*
4.6	3.1	1.5	0.2	*I. setosa*
5.0	3.6	1.4	0.2	*I. setosa*
5.4	3.9	1.7	0.4	*I. setosa*
4.6	3.4	1.4	0.3	*I. setosa*
5.0	3.4	1.5	0.2	*I. setosa*

independent variables

Here's (part of) an example dataset:

Fisher's *Iris* Data

Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	*I. setosa*
4.9	3.0	1.4	0.2	*I. setosa*
4.7	3.2	1.3	0.2	*I. setosa*
4.6	3.1	1.5	0.2	*I. setosa*
5.0	3.6	1.4	0.2	*I. setosa*
5.4	3.9	1.7	0.4	*I. setosa*
4.6	3.4	1.4	0.3	*I. setosa*
5.0	3.4	1.5	0.2	*I. setosa*

class labels *(categorical)*

independent variables

Q: What does “supervised” mean?

Q: What does "supervised" mean?

A: We know the labels.

```
Welcome to R! Thu Feb 28 13:07:25 2013
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
 Median :5.800   Median :3.000   Median :4.350   Median :1.300  
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
       Species  
 setosa    :50  
 versicolor:50  
 virginica :50  
```

Q: How does a classification problem work?

Q: How does a classification problem work?

A: Data in, predicted labels out.

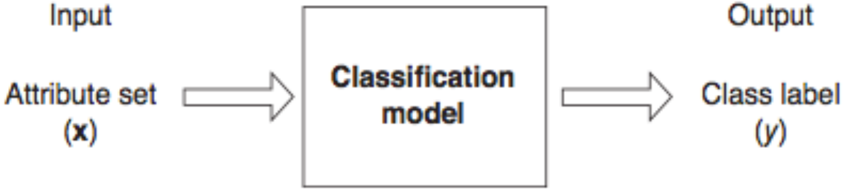
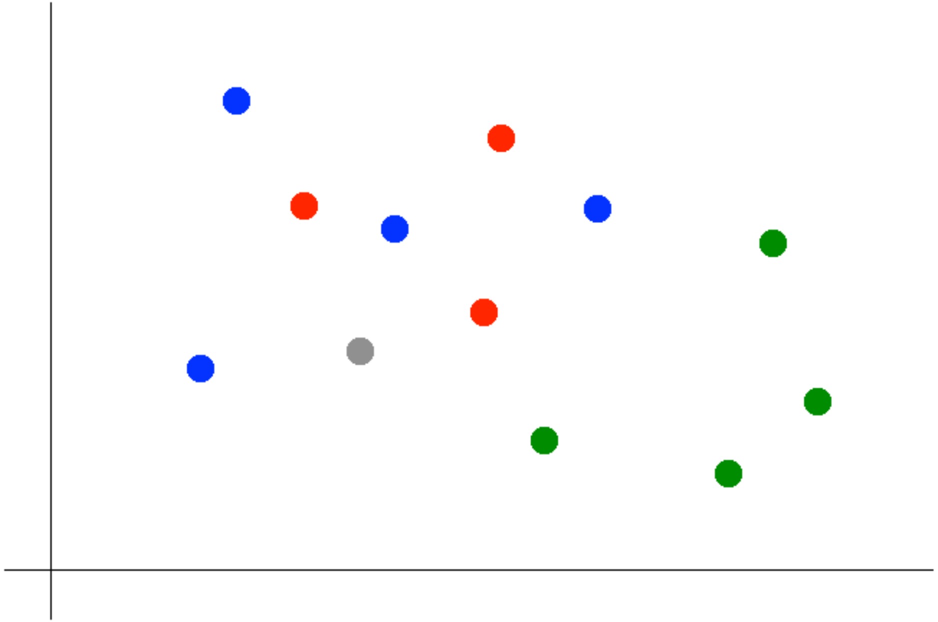


Figure 4.2. Classification as the task of mapping an input attribute set $\mathbf{x}$ into its class label y.

source: http://www-users.cs.umn.edu/~kumar/dmbook/ch4.pdf

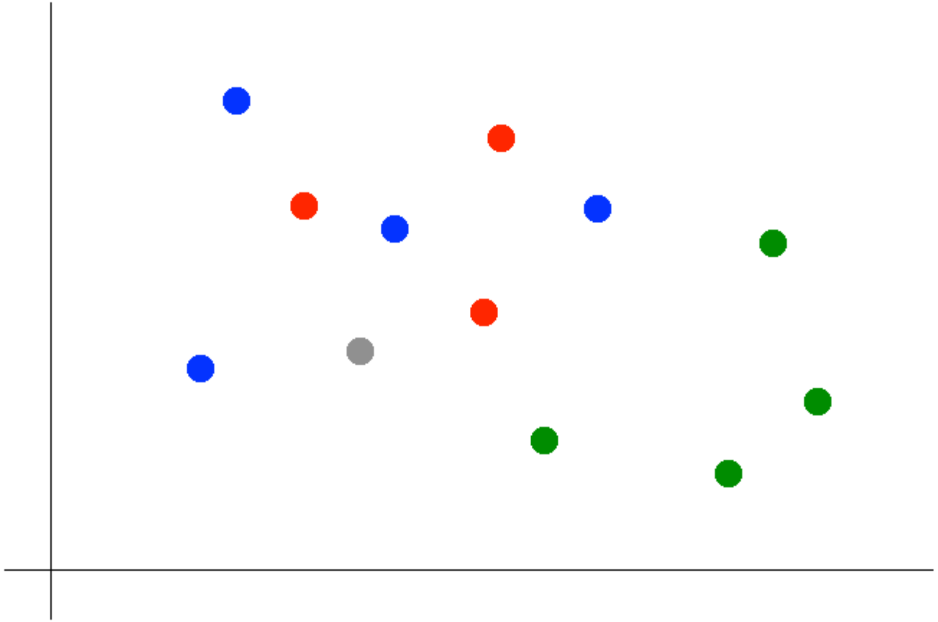
Suppose we want to predict the color of the grey dot.



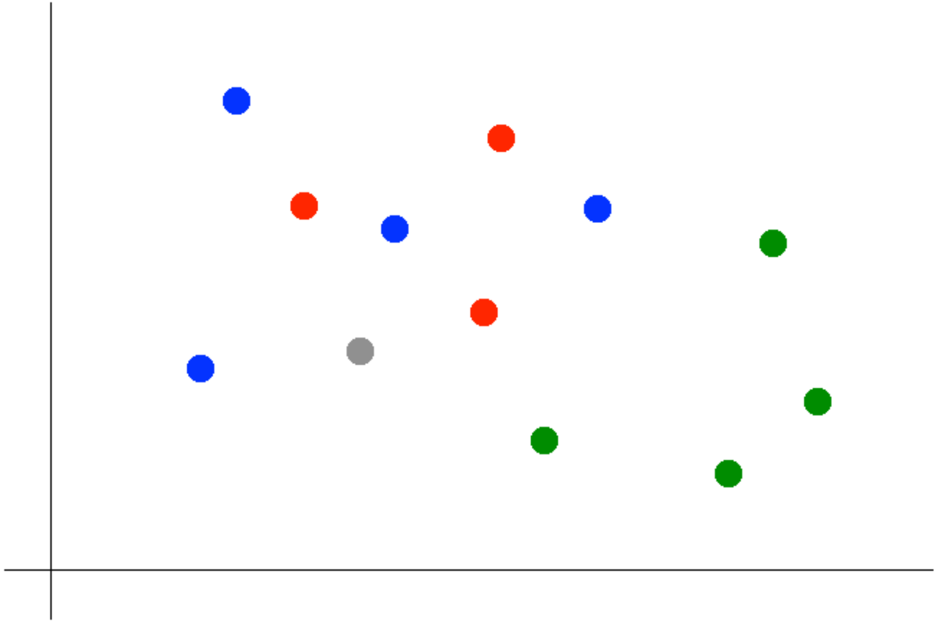
Suppose we want to predict the color of the grey dot.

QUESTION:

What are the features?
What are the labels?

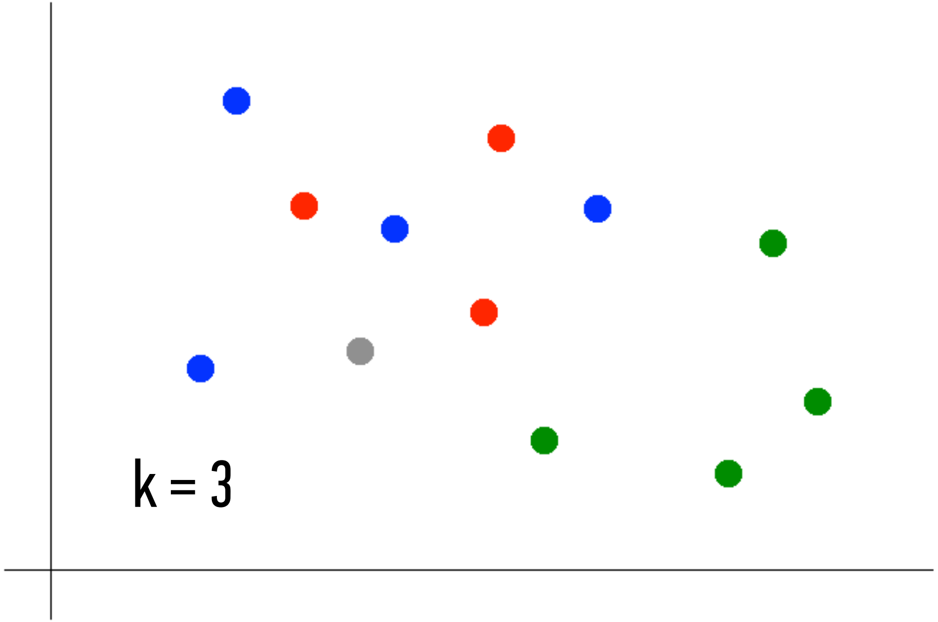


Suppose we want to predict the color of the grey dot.



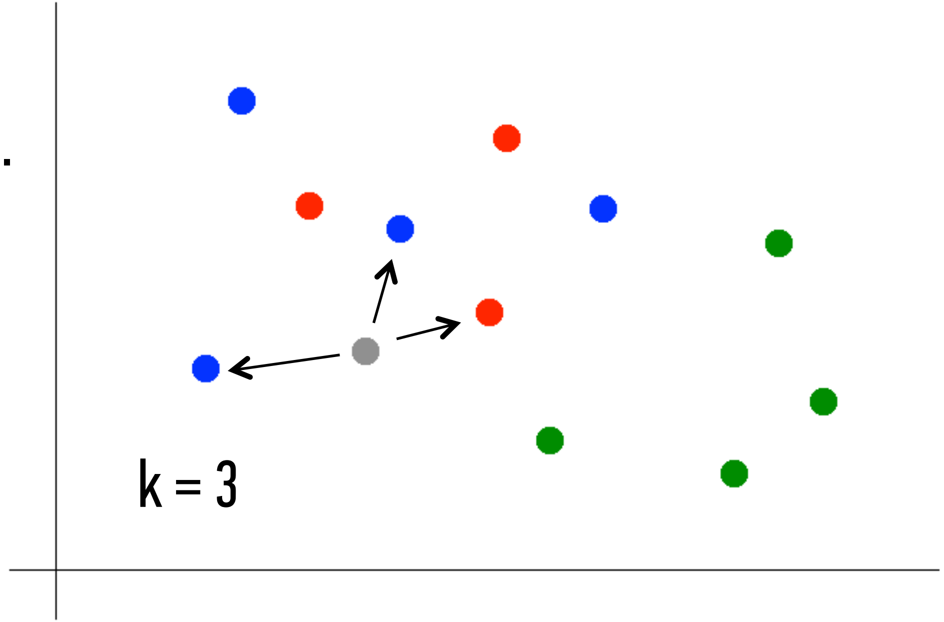
Suppose we want to predict the color of the grey dot.

1) Pick a value for k.



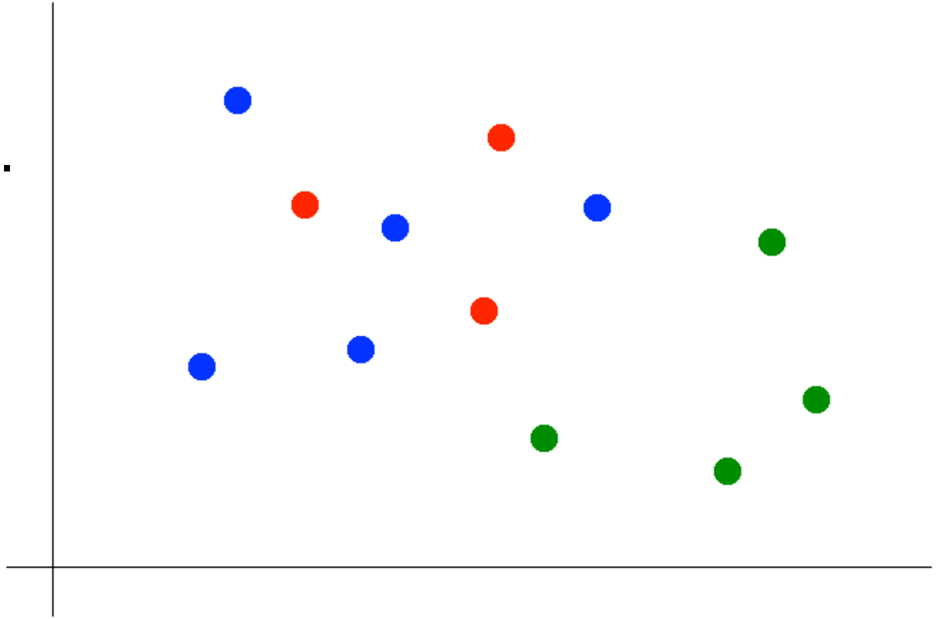
Suppose we want to predict the color of the grey dot.

1) Pick a value for k.
2) Find colors of k nearest neighbors.



Suppose we want to predict the color of the grey dot.

1) Pick a value for k.
2) Find colors of k nearest neighbors.
3) Assign the most common color to the grey dot.

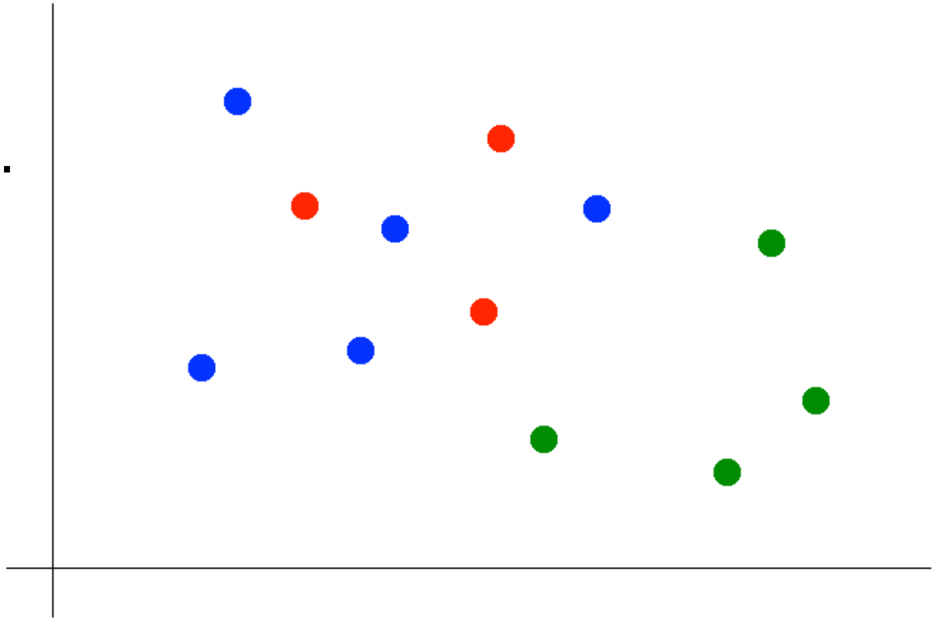


Suppose we want to predict the color of the grey dot.

1) Pick a value for k.
2) Find colors of k nearest neighbors.
3) Assign the most common color to the grey dot.

NOTE:

Our definition of "nearest" implicitly uses the *Euclidean distance function.*



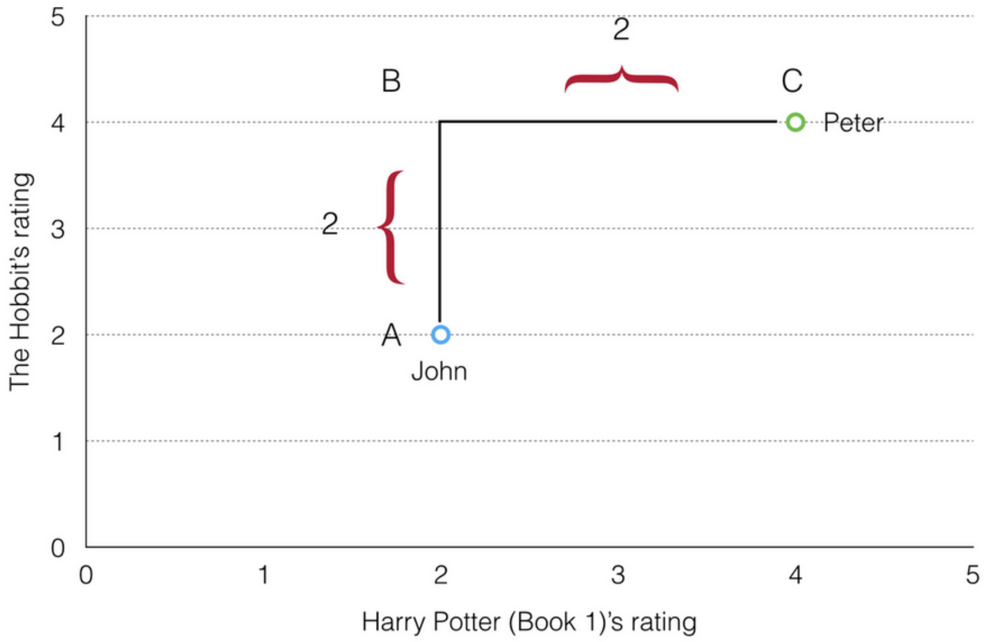
DISTANCE MEASURES

COMMON DISTANCE MEASURES

- Also known as “Taxicab Distance”.
- Here, two vectors X and Y are passed in. Each individual coordinate is compared.

Manhattan Distance

$$d(x,y)=\sum_{k=1}^{n}|x_k-y_k|$$



http://www.mickaellegal.com/blog/2014/1/30/how-to-build-a-recommender

Euclidean Distance

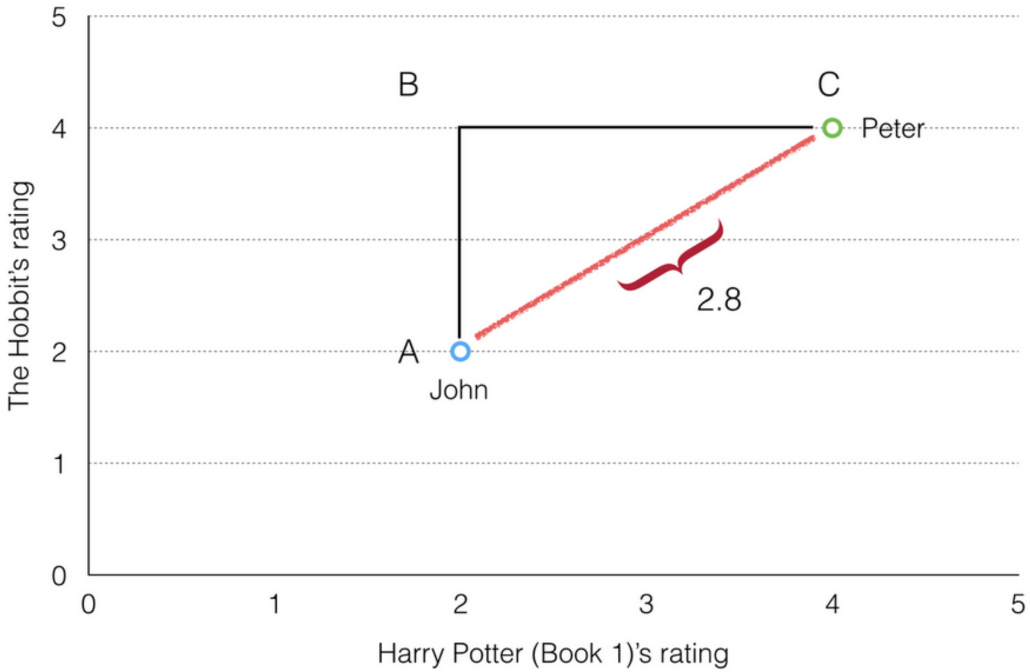
K-NEAREST NEIGHBORS

$$d(x,y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

- Make sure you understand how X and Y are defined:

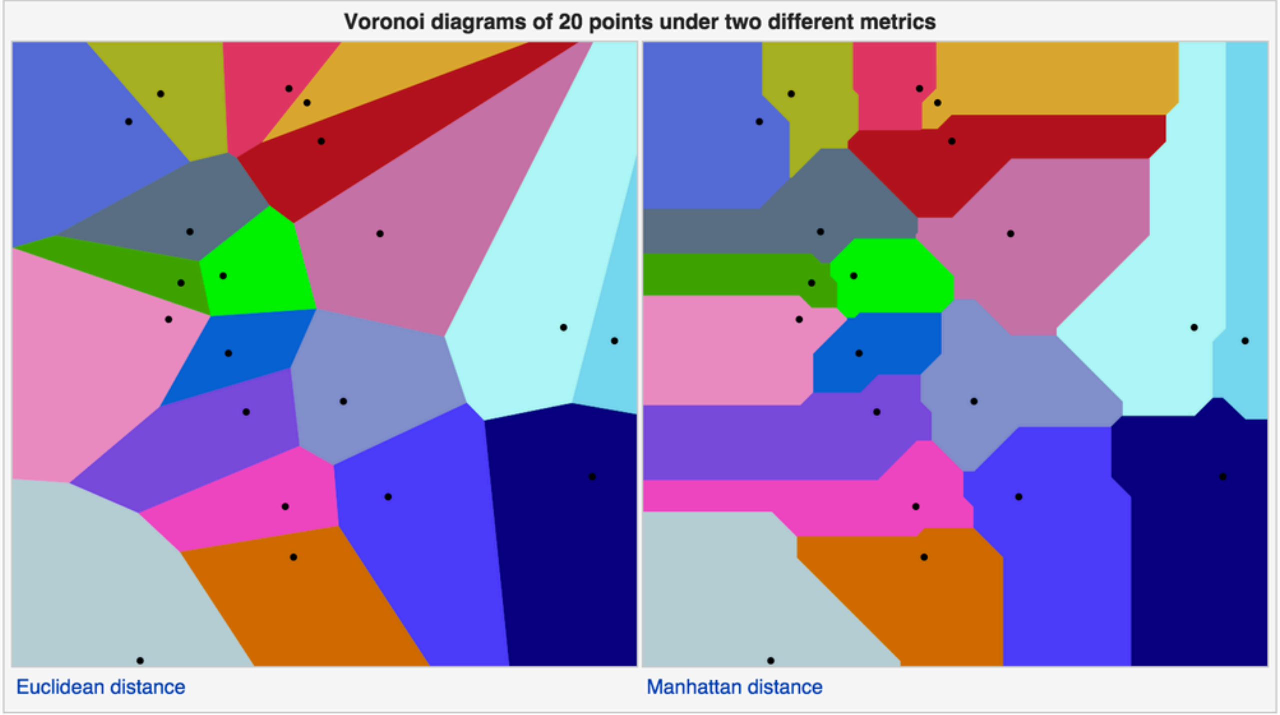
X = (1, 2, 3)
Y = (4, 5, 6)

d(X, Y) =
$(1 - 4)^2 + (2 - 5)^2 + (3 - 6)^2$



http://www.mickaellegal.com/blog/2014/1/30/how-to-build-a-recommender

- The regions that make up k-NN can be visualized using **Voronoi diagrams.**



https://en.wikipedia.org/wiki/Voronoi_diagram

Jaccard Distance

- Suppose A and B are sets of words in two different news articles.
- The Jaccard Distance between the news articles is the proportion of words they do not have in common.
- So, if two articles contained exactly the same words, the distance would be zero.

$$d_J(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}.$$

http://www.mickaellegal.com/blog/2014/1/30/how-to-build-a-recommender

Hamming Distance

- Here, we have two bit strings X and Y.
- Each bit is either 0 or 1.
- So, if the bits are the same: |0-0| = |1–1| = 0.
- If the bits are different: |0-1|=|1-0|= 1.

Hamming Distance

$$D_H = \sum_{i=1}^{k} |x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

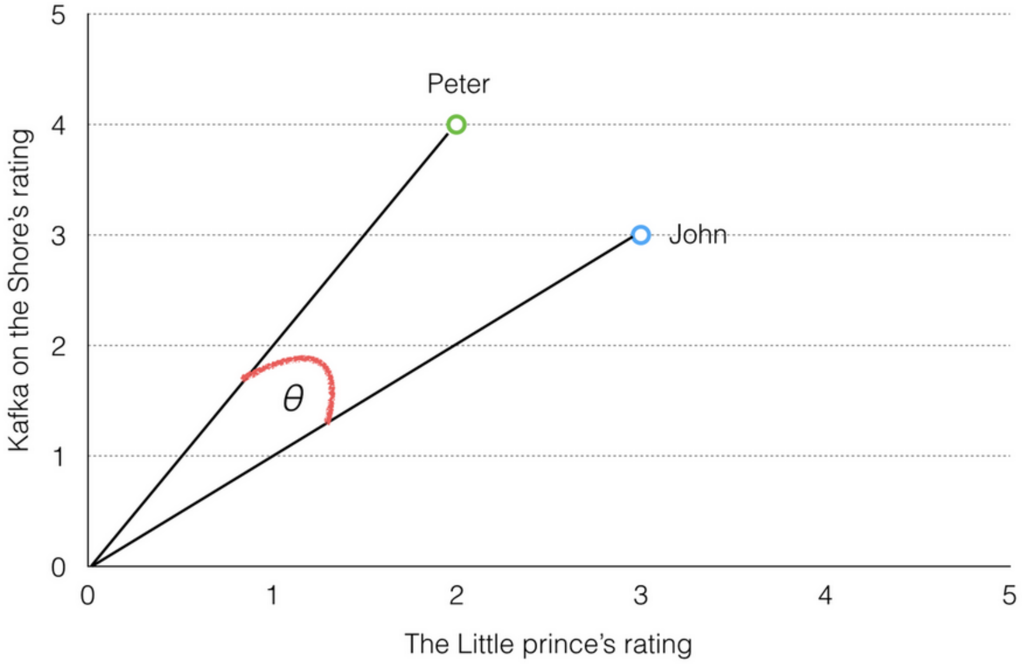
http://www.saedsayad.com/k_nearest_neighbors.htm

DISTANCES THAT SCALE

- Cosine Distance is popular because the magnitude of each vector is irrelevant.
- This is useful since biases in ratings are irrelevant.
- For example, suppose User 1 and User 2 both enjoy two books equally, but User 2 never gives 5. Then, User 1 and User 2 will have the same Cosine Distance.

Cosine Distance

$$cos(\theta) = \frac{x \cdot y}{||x|| * ||y||}$$



http://www.mickaellegal.com/blog/2014/1/30/how-to-build-a-recommender

- For some i, S_i is the standard deviation of the x_i and y_i over the sample set.
- This can alternatively be achieved by normalizing the features first then taking the standard Euclidean distance (subtract the mean and divide by the standard deviation): $\dfrac{x-\mu}{\sigma}$

Normalized Euclidean Distance

$$d(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{N} \frac{(x_i - y_i)^2}{s_i^2}},$$

Also see the Mahalanobis distance: https://en.wikipedia.org/wiki/Mahalanobis_distance

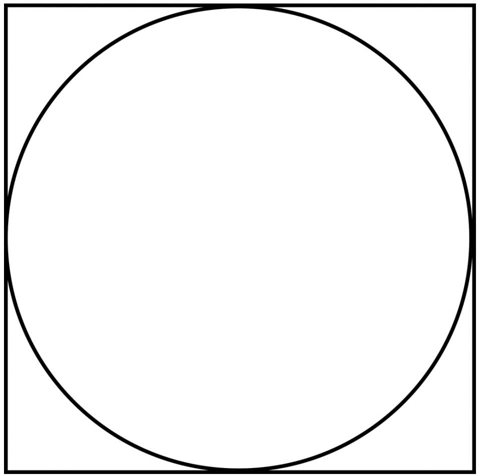
- **Neighbors are far apart with high dimensionality.**
- **Overfitting may easily occur.**
- Some features may require weighting.
- Some particular features may be correlated to others
- Many unknowns may result in poor Jaccard overlap
- Scaling problem with Euclidean distance, particularly with binary values
- Measurement error can affect result in unknown ways
- Calculation cost, since every data point must be compared

http://columbiadatascience.com/2012/10/18/week-7-hunch-com-recommendation-engines-svd-alternating-least-squares-convexity-filter-bubbles/

THE CURSE OF DIMENSIONALITY

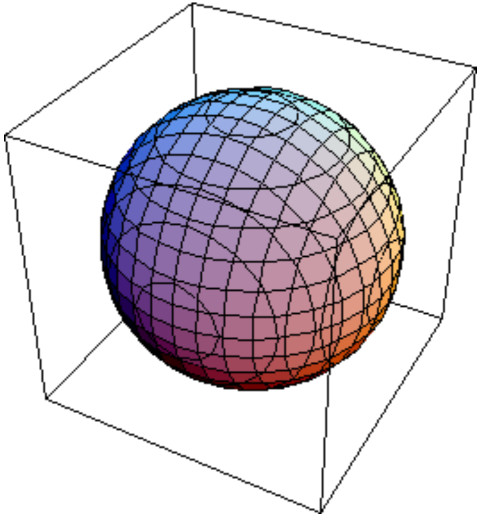
$$x^2 < R^2$$

1-dimensional



$$x^2 + y^2 < R^2$$

2-dimensional



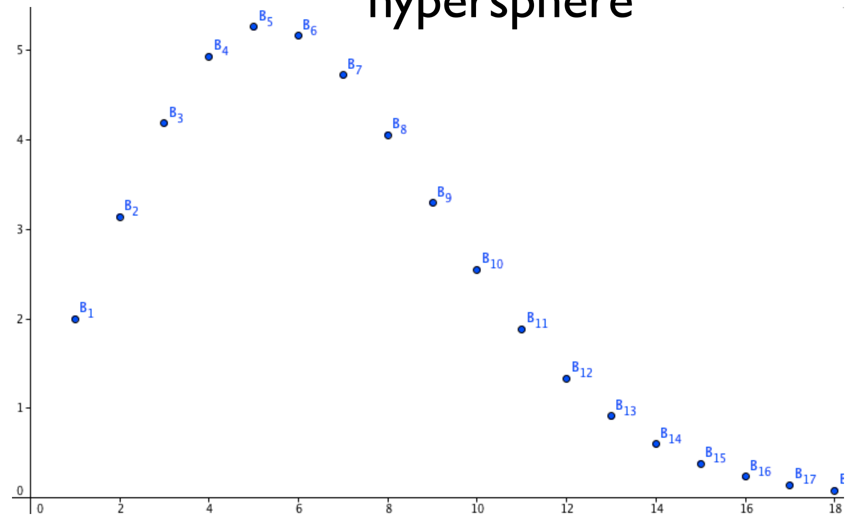
$$x^2 + y^2 + z^2 < R^2$$

3-dimensional

- As the number of dimensions increases with fixed radius, the hypersphere volume approaches zero.

Volume of n-dimensional hypersphere

Volume



Dimensions

$$V_1(R) = 2R,\ V_2(R) = \pi R^2, \text{ and } V_n(R) = \frac{2\pi R^2}{n} V_{n-2}(R),\ \text{for } n \geq 3.$$

Source: http://divisbyzero.com/2010/05/09/volumes-of-n-dimensional-balls/

- You may hear that most volume of an N-dimensional feature space is concentrated at the edges. To see this, define a hypersphere of radius 1:

$$x_1^2 + x_2^2 + \ldots + x_N^2 < 1$$

- Then, for $n = 100$, the point of equilibrium where all x are equal is $x = 0.1$, as follows:

$$N\overline{x}^2 = 1 \qquad \overline{x} = \sqrt{1/N}$$

- Hence, the edge of the "average" hypersphere is far away from the edge of the feature space.

Let's roughly estimate the effect of high dimensionality on k-NN. Doing this, we'll see that to hold a steady error rate, **exponentially more data is needed with each dimension added.**

- Suppose we have n features.
- Then, our "feature space" is n-dimensional.
- We will scale each feature to make it an n-dimensional unit cube, with a total volume of 1.

From Norvig & Russell, "Artificial Intelligence: A Modern Approach"

We will use k-NN with N data points.

- So, each neighborhood will require k/N fraction of the data points.
- On average, this is k/N of the total feature space.

To simplify calculations, suppose each neighborhood is an n-dimensional cube with side length b. Then:

$$b^n = k/N$$

Assuming the total feature space has sides 1, how is b affected by larger feature sets?

From Norvig & Russell, "Artificial Intelligence: A Modern Approach"

Knowing the total n-dimensional feature space has unit sides, how is the side length b of each neighborhood affected by more features?

$$b^n = k/N$$

Suppose $N = 1{,}000{,}000$ data points.
$k = 3$ nearest neighbors.

If $n = 3$ features, then $b \sim 0.01$.
If $n = 100$ features, then $b \sim 0.88$.

To encircle the 3 nearest neighbors, each neighborhood requires:

1% of feature space

88% of feature space!