Supervised Machine Learning

Heather Berginc

Land Belenky, Herbert Yuan, Ryan Henning, Brandon Martin-Anderson, Lori Bordzuk

galvanize

galvanize

Success Criteria

- Explain the purpose of supervised machine learning?
- Identify some key assumptions that should be made before using machine learning.
- Give an example of features and a target for a given dataset.
- Describe the KNN model
- Explain what happens to our KNN model as K increases/decreases
- ID the distance metrics available for KNN
- Explain the curse of dimensionality

galvanize

Machine Learning:

An automated process to discover a relationship between known inputs and known outputs, so that (reasonably) accurate predictions can be made for *unknown* inputs.

galvanize

Key Assumptions

- There exists a relationship between the inputs and the outputs
- The relationship can be discovered by a computer
- The relationship will continue to hold for the immediate future
- The most significant features are represented in the data
- The relevant features can be quantified or classified

Example

The price of a house in a particular neighborhood can be determined by examining the **features** of the house:

- Number of bedrooms
- Number of bathrooms
- Square Feet
- Lot Size
- Age of house
- Whether basement is finished
- Condition of roof
- Type of heating

- Sales Price of House

Predicting Stock Price

The price of a stock can be determined by examining the history of the stock price

- Price yesterday
- Price day before yesterday
- Price last week
- Price last month
- Price last year

- Price tomorrow

galvanize

Predicting Lottery Numbers

Hypothesis: Tomorrow's winning lottery numbers can be deduced from a history of winning and losing lottery numbers:

- Previous Winning Lottery Numbers
- Previous Losing Lottery Numbers

- Whether a particular set of lottery numbers will win or lose.

Conclusion: The assumption that there is a relationship between between the inputs and outputs does not hold. Machine Learning is not possible.

galvanize

Predictive Text

The next word a person is going to type can be anticipated by analyzing what they have typed so far.

Inputs:

- A large corpus of text including common words and phrases arranged in sentences
- The words the user has typed so far

Prediction:

- The next word that they will type

Text Comprehension

Hypothesis:

The identity of the murderer in a mystery novel can be deduced from clues given in the text

Inputs:

- A corpus of murder mystery novels
- The text of this novel

Labels:

- The identity of the murderer in previous novels

Conclusion:
Although the text of the novel contains information that can lead to the identification of the murderer, it seems unlikely (at this time) that this relationship can be reliably discovered by a computer.

‘Black Swan’ Events

Hypothesis: The price of a stock can be reliably determined from historical information about the stock price and the business operations of the company

- Stock price history
- Company Sales history
- Company historical operational costs

A comet strikes the headquarters of the company.

This violates the assumption that the relationship between historical conditions and future conditions will continue to hold.

Example

The sales price of a diamond can be predicted by measuring the diamond:

- Cut (ideal, deep, shallow)
- Color
- Clarity (Flawless, Slightly Included, etc)
- Carat Weight

- Sales price of diamond

Counter Example

The auction price of a painting can be determined by measuring the painting:

- Size
- Shape
- Color
- Weight
- Condition
- Moment of Inertia

Conclusion: Core assumptions do not hold

The most relevant features of the painting are not reflected in the data set.

(Artist, style, subject, composition, historical significance, etc.)

The most relevant features of the painting may not be quantifiable

galvanize

Supervised Learning Has **Labels** or **Targets**

Bedrooms	Bathrooms	Square Feet	Sales Price
2	2	1800	220000
3	2	1950	240000
3	3	2100	270000
4	3	2600	????

Two Types of Supervised Learning:

Regression:

The **Target** is a number:

- Sales price of house
- Number of visitors to a museum
- Inches of snow in December
- Gas mileage of a car

Classification:

The **Target** is a category:

- An email is ‘spam’ or ‘not spam’
- A painting was by Picasso or Van Gogh
- A house will sell or not sell by a date
- A person will vote for a Republican, Democrat, Libertarian or Green.
- A consumer will buy a Ford or Ferrari
- An applicant will be accepted, waitlisted or rejected

Many different approaches

- Linear
  - Individual features contribute to the outcome according to fixed equations
- Non-linear
  - Feature importance may vary under different conditions
  - Feature effects may change from positive to negative
  - Example: House price vs. proximity to public transportation

galvanize

Sklearn Models

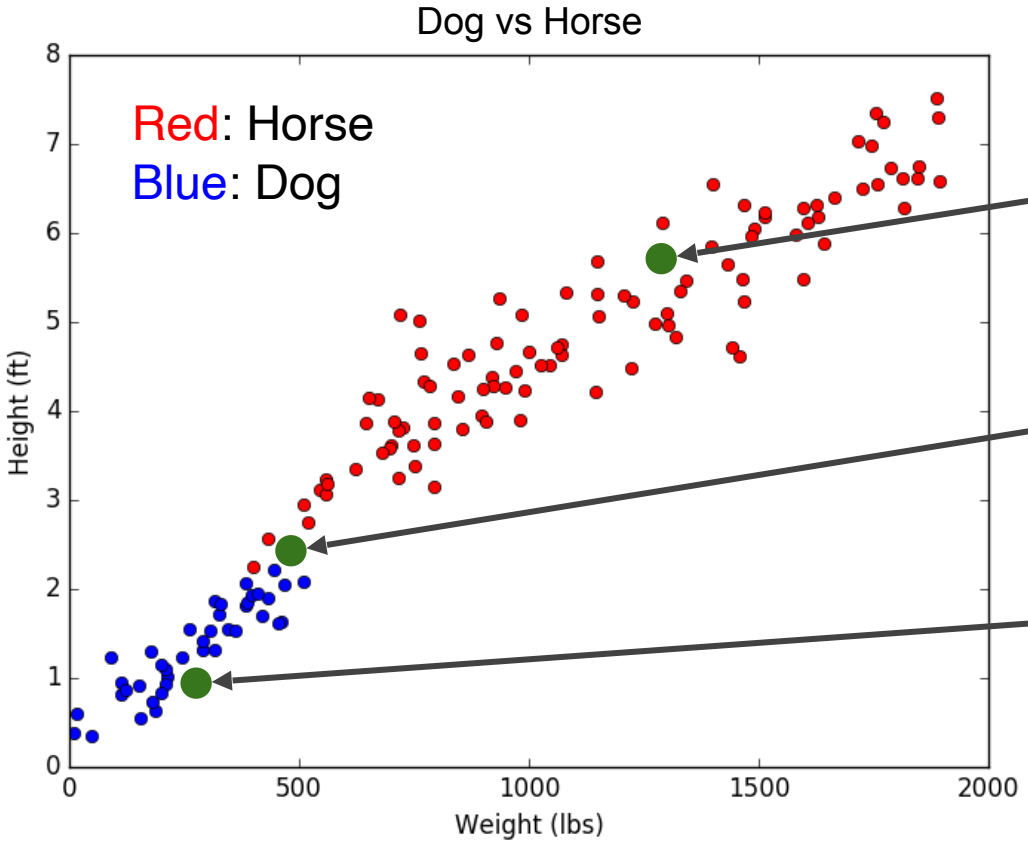
Generally, we will use models that have a common interface with four major parts:

1. The type of algorithm
    a. Linear, KNN, DT, RF, NN, SVM, etc.
2. Hyperparameters
    a. Specific to each model, determine the model's operations
3. .train() method
    a. Built from *known* data
4. .predict() method
    a. Apply to *unknown* data

Bedrooms	Bathrooms	Square Feet	Sales Price
2	2	1800	220000
3	2	1950	240000
3	3	2100	270000
4	3	2600	????

k-Nearest Neighbors
(kNN)

Big dog or small horse?



New datapoint 1:
Is it a dog or a horse?

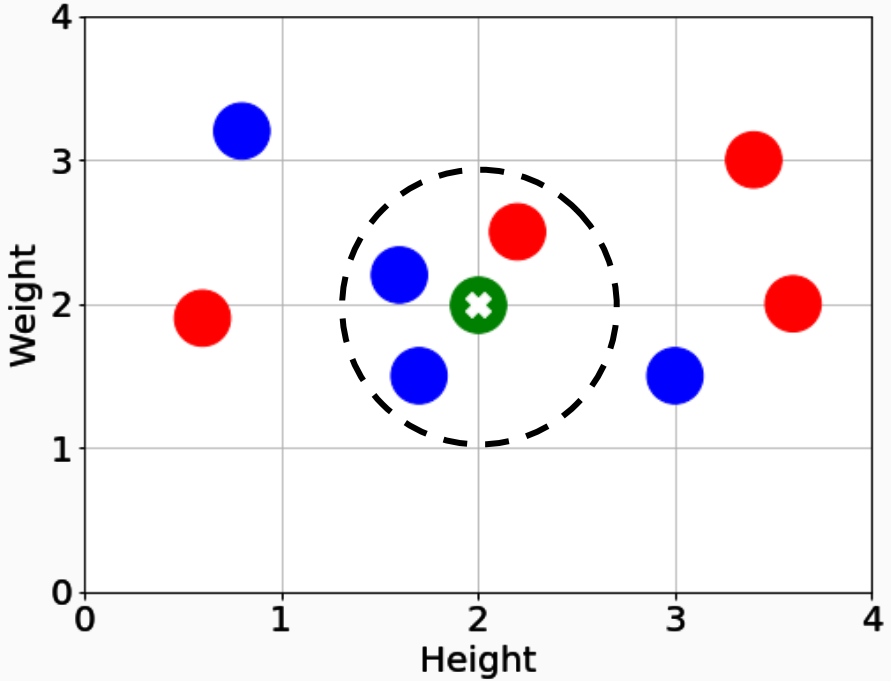
New datapoint 3:
Is it a dog or a horse?

New datapoint 2:
Is it a dog or a horse?

The k-Nearest Neighbors: Classification

For a new input *x*, predict the **most common label** amongst its *k* closest neighbors

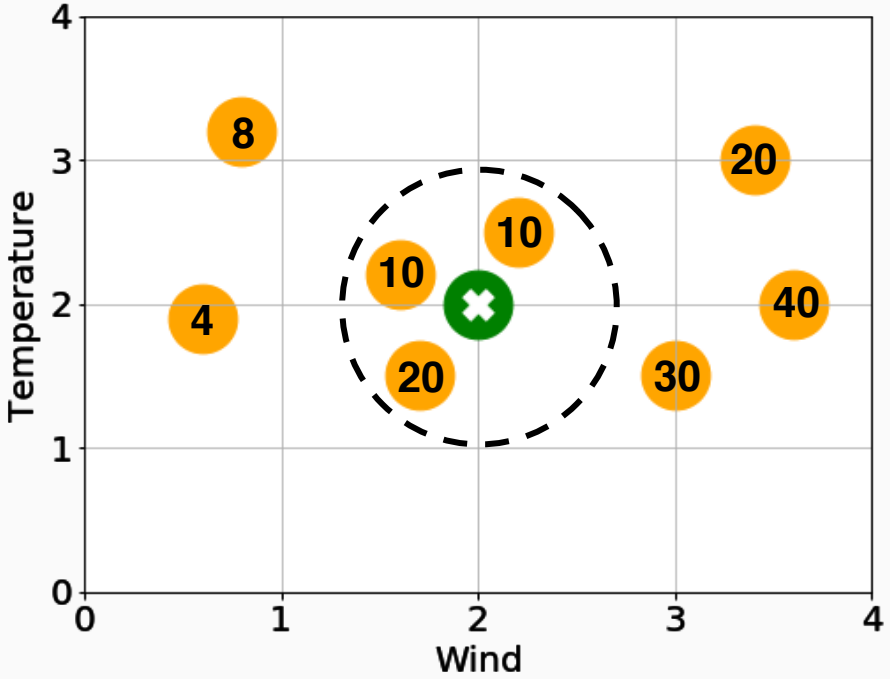
Image on right:
k = 3
Predict **BLUE**



The k-Nearest Neighbors: Regression

For a new input x, predict the **average label** amongst its k closest neighbors

Image on right:
k = 3
Predict **13.3**



galvanize

The k-Nearest Neighbors Algorithm

Training algorithm:

1. Store all the data.

Prediction algorithm (predict the class of a new point x'):

1. Calculate the distance from x' to all points in your dataset.
2. Sort the points in your dataset by increasing distance from x'.
3. Predict the majority label of the *k* closest points.

kNN Hyperparameter: Distance Metrics

Euclidean Distance (L2):

$$\sum_i (a_i - b_i)^2$$

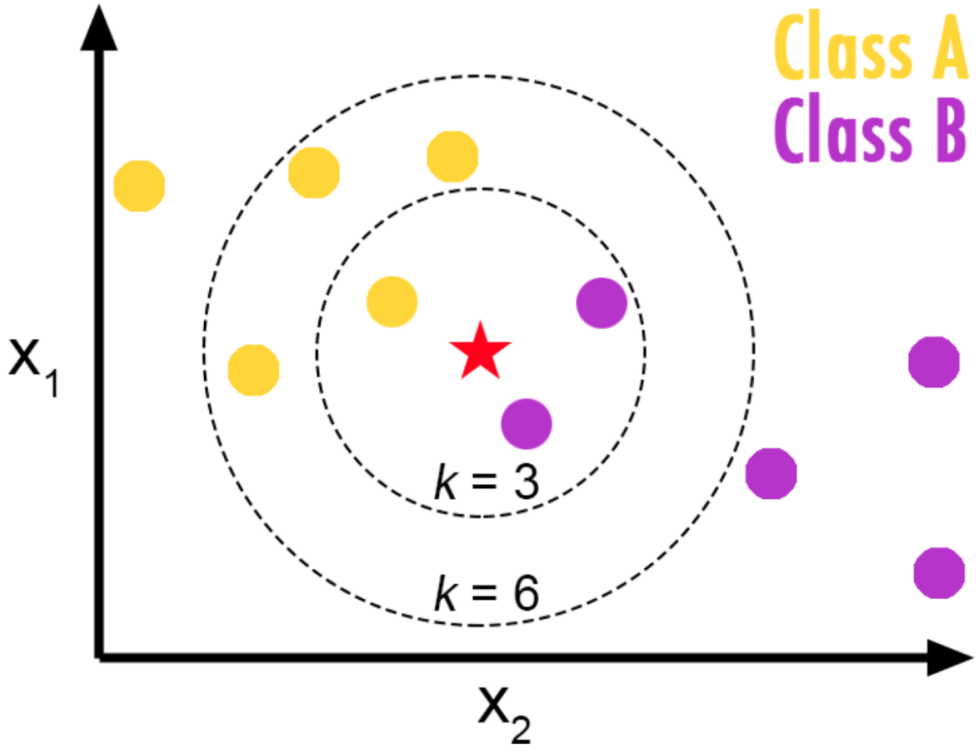
Manhattan Distance (L1):

$$\sum_i |a_i - b_i|$$

Cosine Distance = 1 - Cosine Similarity:

$$1 - \frac{a \cdot b}{||a||||b||}$$

kNN Hyperparameter k



What is the prediction when k=3?

Class B

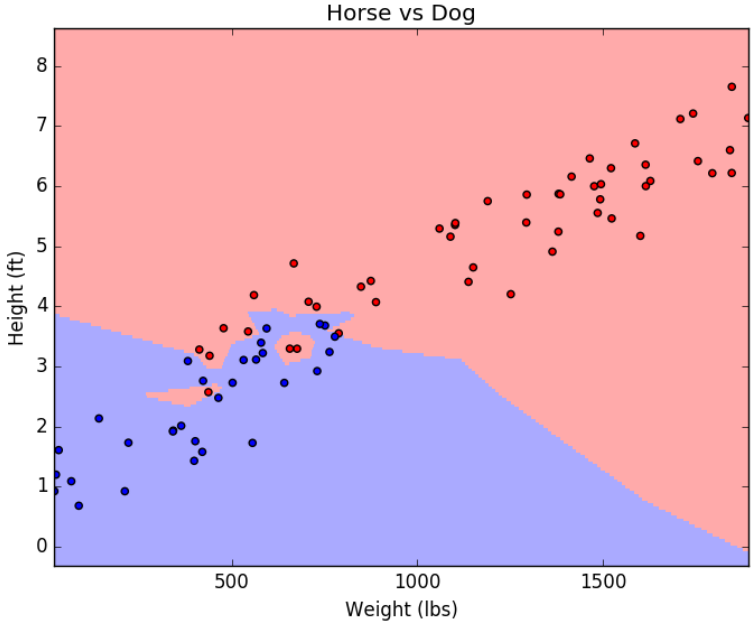
What is the prediction when k=6?

Class A

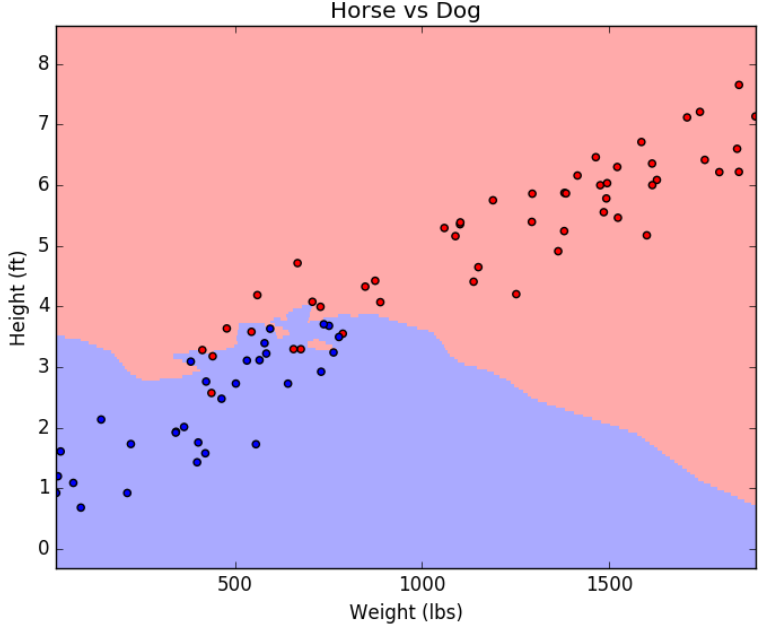
Hyperparameter *k*: the number of nearest neighbors to consider

galvanize

k=1



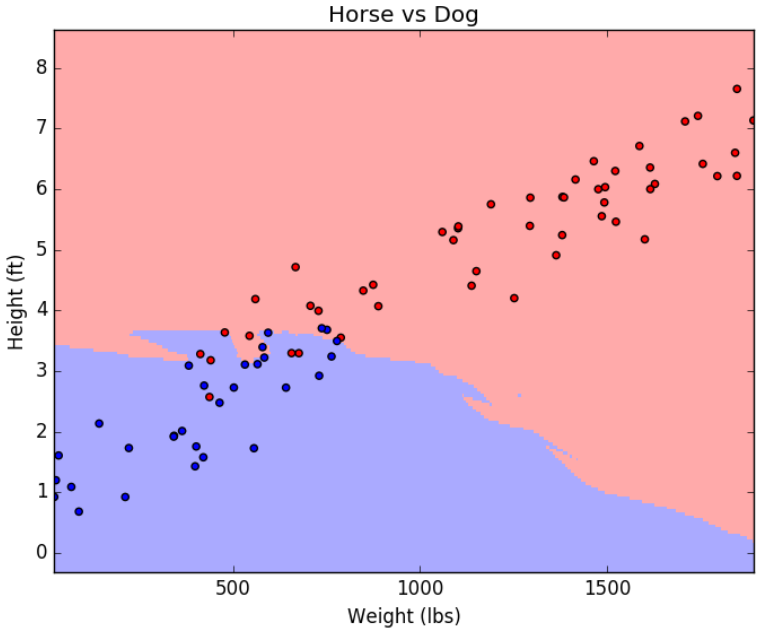
k=5



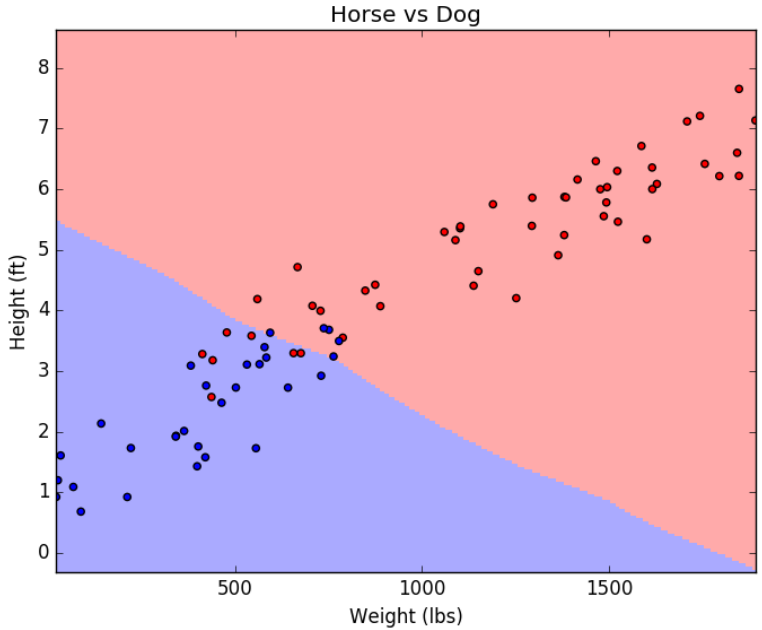
Hyperparameter *k*: the number of nearest neighbors to consider

galvanize

k=10

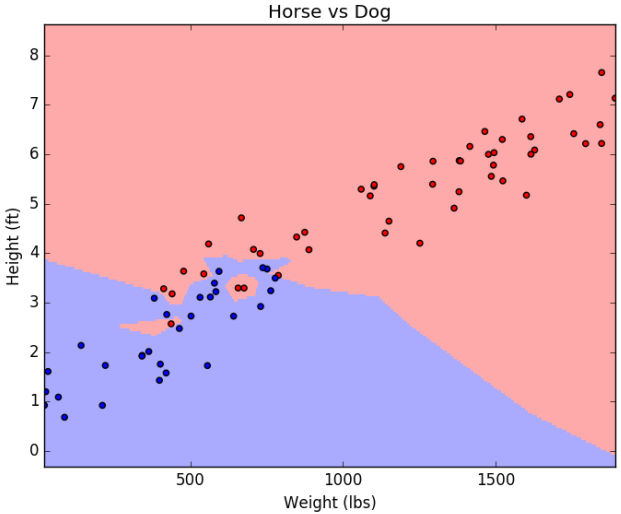


k=50



Which model is overfit?

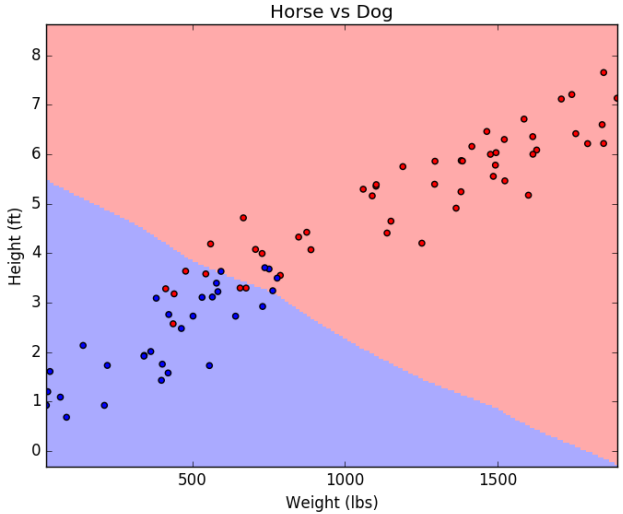
k=1



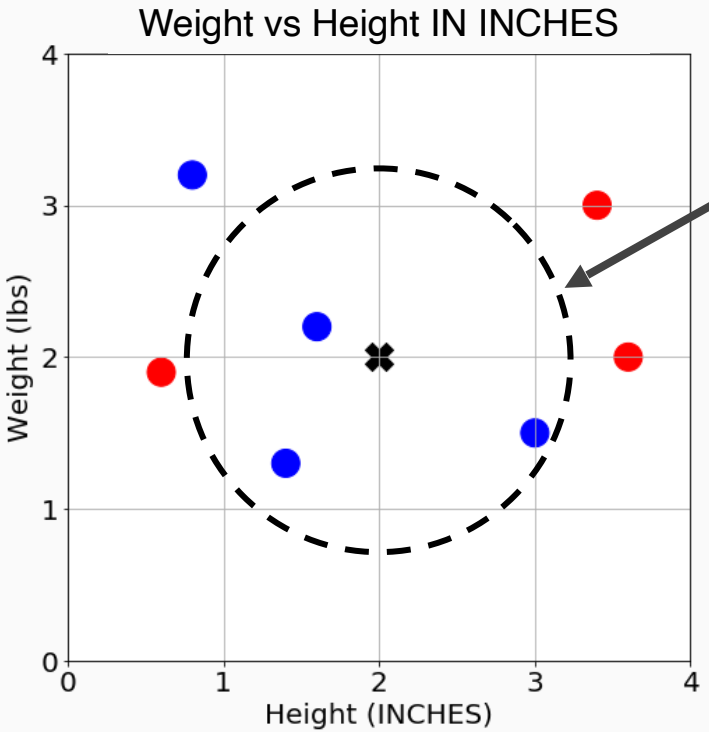
As a general rule, start with:

$$k = \sqrt{n}$$

k=50



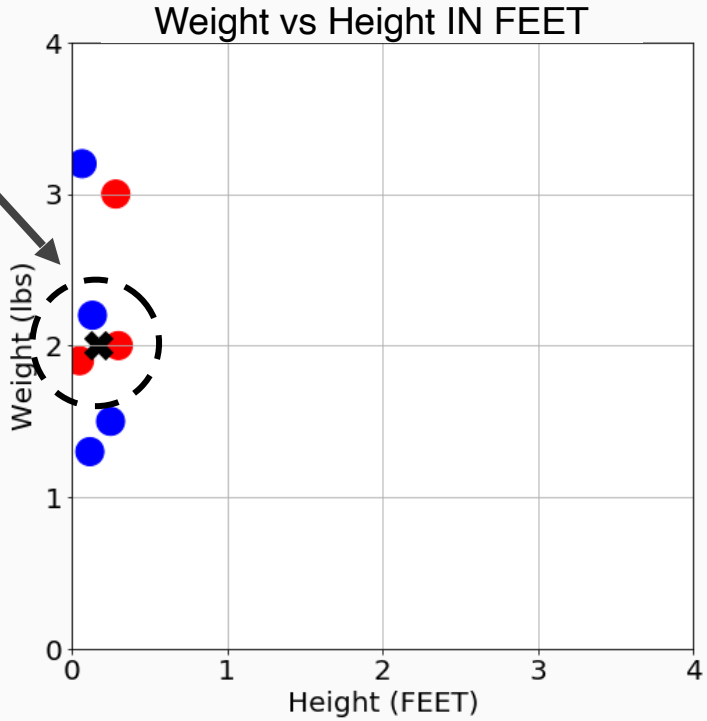
Be careful with the scale of your features!



$k = 3$

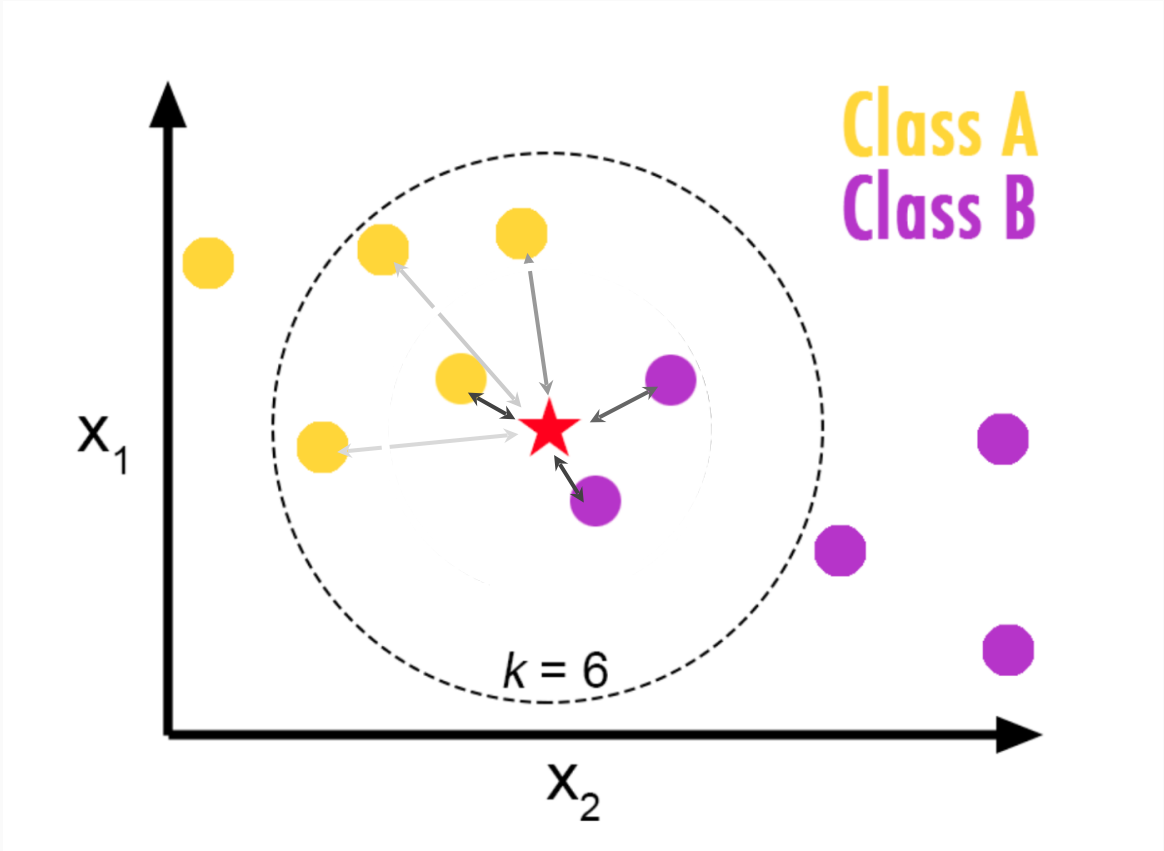
The three "closest neighbors" differ depending on the scale of the feature….

Don't forget to scale your data!!!!



$k = 3$

Point weighting (weighted voting)



Let the *k* nearest points have distances:

$$d_1, d_2, \ldots, d_k$$

The *ith* point votes with a weight of:

$$\frac{1}{d_i}$$

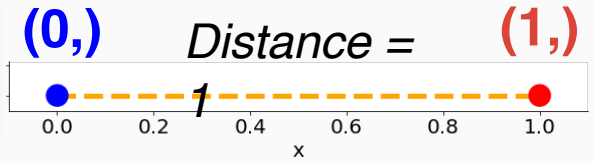
small distances are weighted more!

kNN in high dimensions

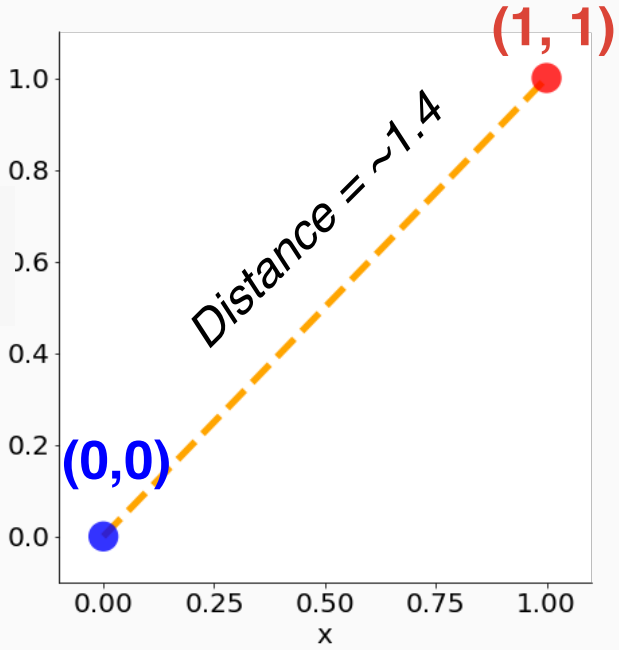
kNN works pretty well (in *general*) for dimensions < 5
but is problematic when used with high dimensional spaces

In high dimensions, the nearest neighbors can be very "far away"

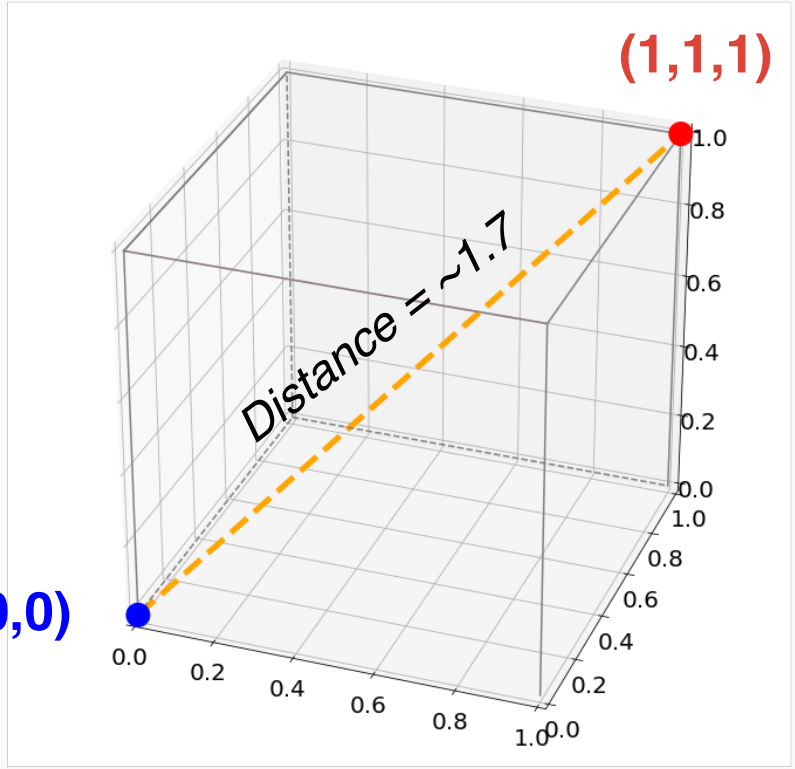
1D:


(0,)
Distance =
1
(1,)
0.0
0.2
0.4
0.6
0.8
1.0
x

2D:


(1, 1)
Distance = ~1.4
(0,0)
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
x

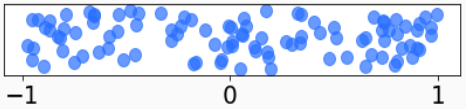
3D:


(1,1,1)
Distance = ~1.7
(0,0,0)
0.0
0.2
0.4
0.6
0.8
1.0

The Curse of Dimensionality **(another perspective)**

galvanize

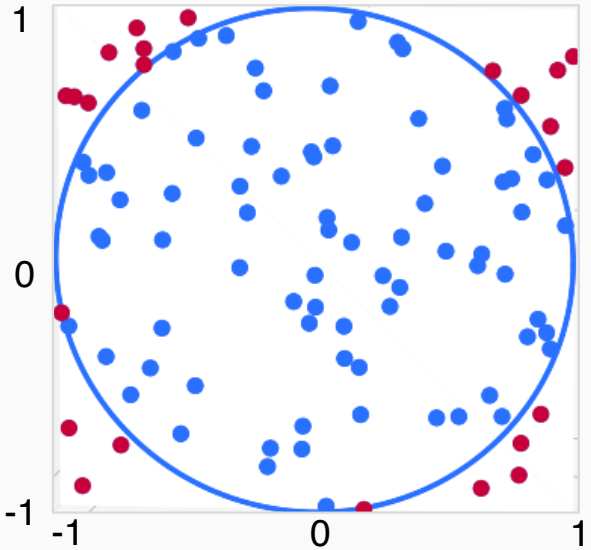
Given 100 (random) sample points….



1D

All 100 pts within 1 unit of center.

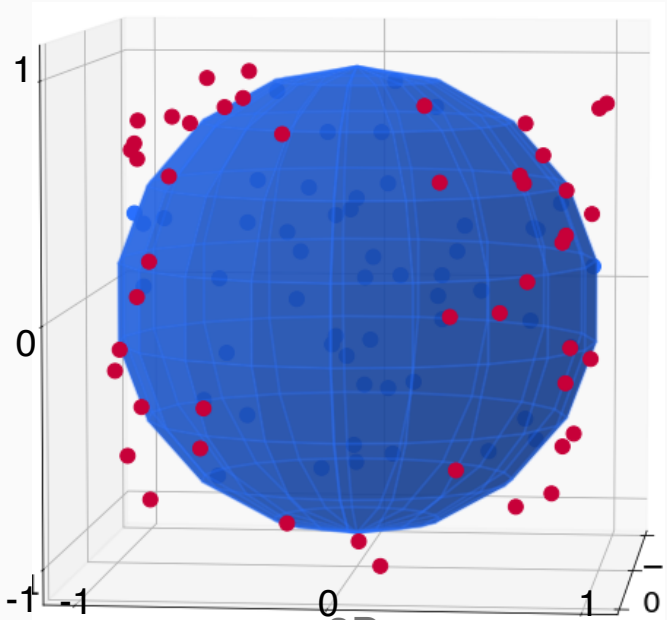
Density = 100



2D

77 pts within 1 unit of center.

Density = 77

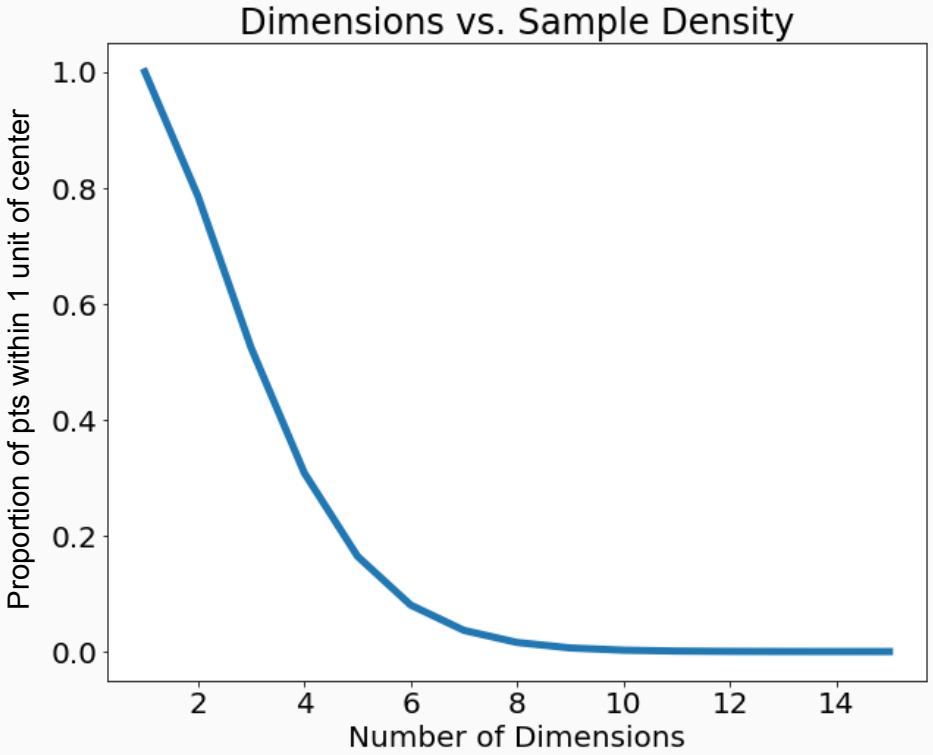


3D

53 pts within 1 unit of center.

Density = 53

The Curse of Dimensionality **(another perspective)**



The **more dimensions** you have, the **more data points** you need to maintain density.

General guideline:

- Given n data points in d_{orig} dimensions...
- If you want to *increase* the total number of dimensions to d_{new} , you now need:
- $n^{\frac{d_{new}}{d_{orig}}}$ data points to maintain density

The Curse of Dimensionality takeaways

- kNN (or any method that relies on distance metrics) will suffer in high dimensions.
  - Nearest neighbors are “far” away in high dimensions (even for d=10).
- High dimensional data tends to be sparse; it’s easy to overfit sparse data.
  - It takes A LOT OF DATA to make up for increased dimensionality.

The Curse of Dimensionality: Remedies

- Get more data
- Get more varied data
- Reduce number of dimensions
  - Identify and eliminate unhelpful features
  - Combine collinear dimensions into principal vectors
- Reduce relative importance of some features
- Standardize Data

galvanize

Summary: kNN

Pros:

- Super simple
- Training is trivial (store the data)
- Works with any number of classes
- Easy to add more data
- Few hyperparameters:
  - *distance metric*
  - *k*

Cons:

- High prediction cost (especially for large datasets)
- Bad with high dimensions
  - you'll learn dimensionality reduction methods later on!
- Categorical features don't work well

galvanize

Review: Today's Success Criteria

- Explain the purpose of supervised machine learning?
- Identify some key assumptions that should be made before using machine learning.
- Give an example of features and a target for a given dataset.
- Describe the KNN model
- Explain what happens to our KNN model as K increases/decreases
- ID the distance metrics available for KNN
- Explain the curse of dimensionality

KNN Assignment: Implement a sklearn-style KNN algorithm