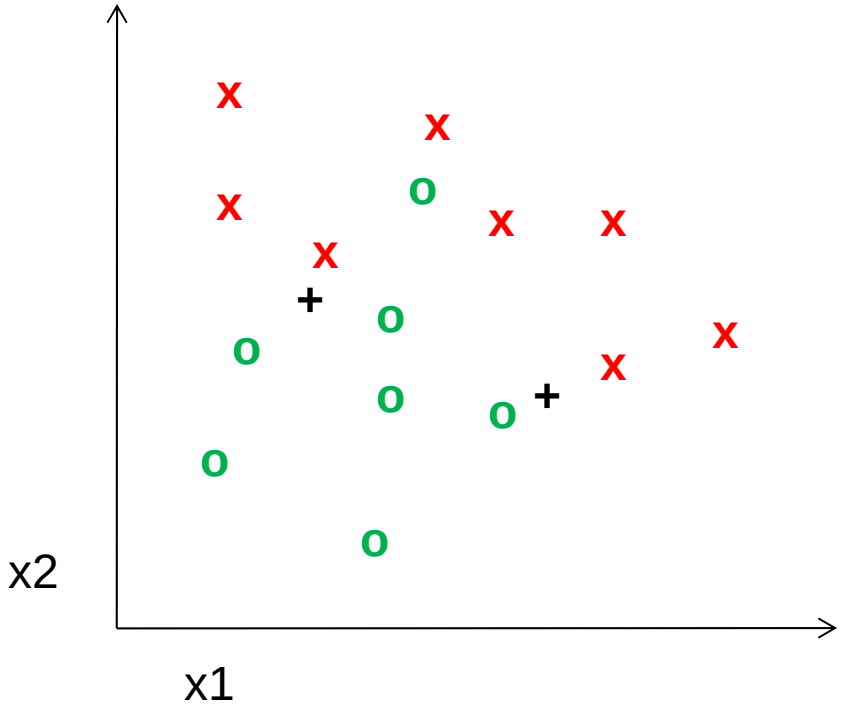
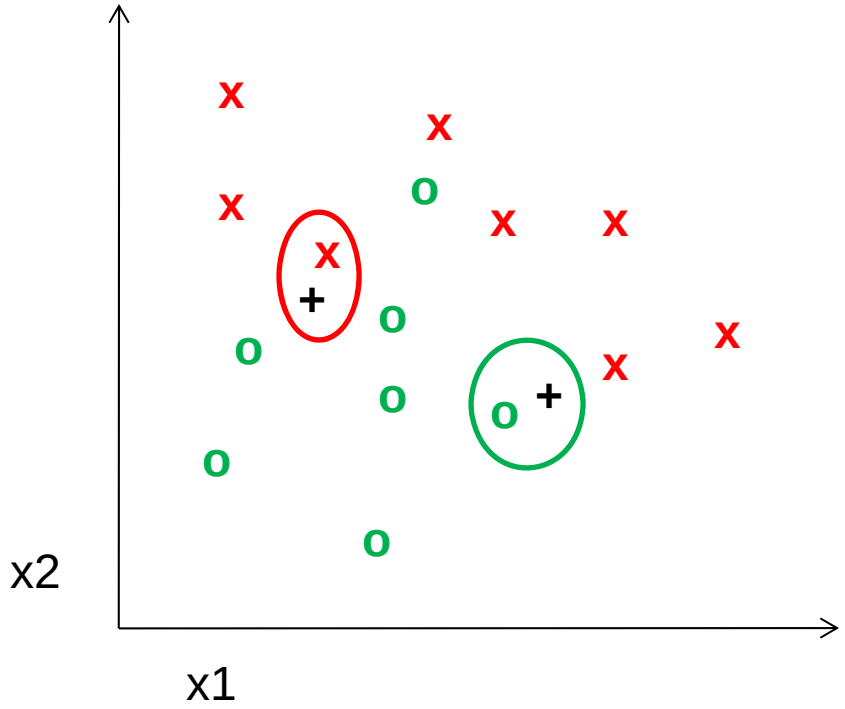
KNN Algorithm

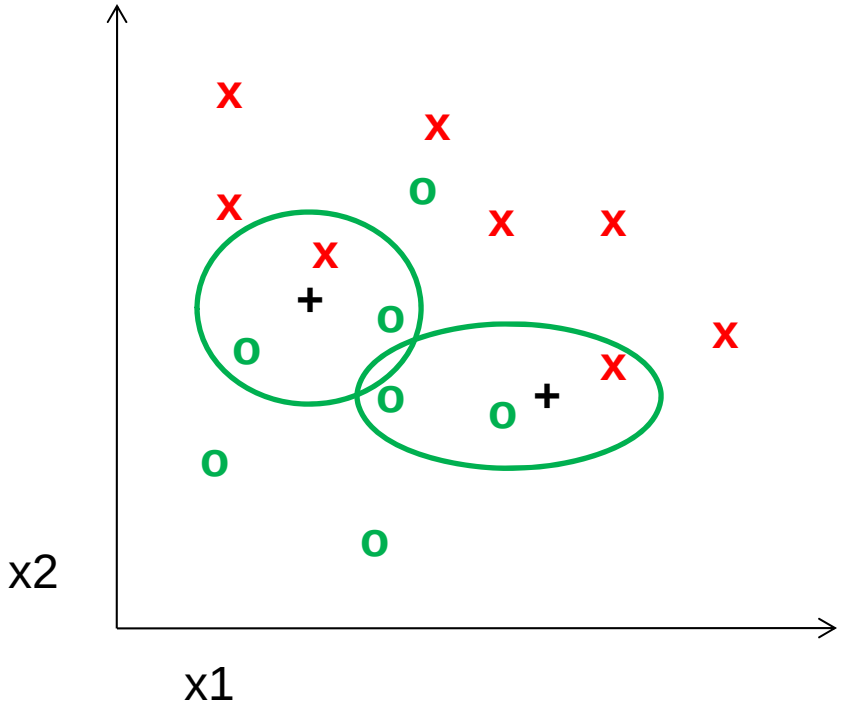
K-nearest neighbor



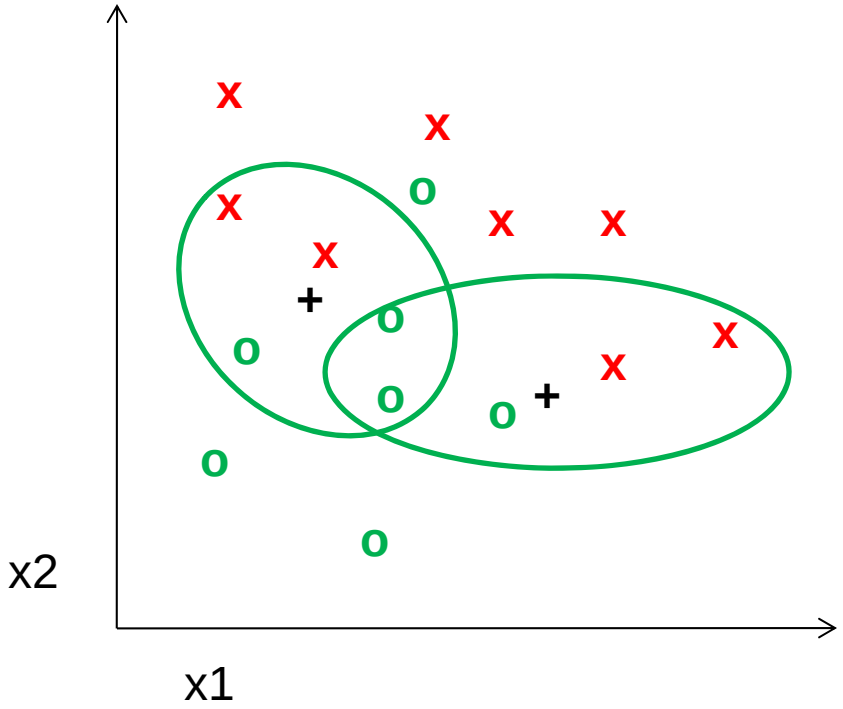
1-nearest neighbor



3-nearest neighbor



5-nearest neighbor

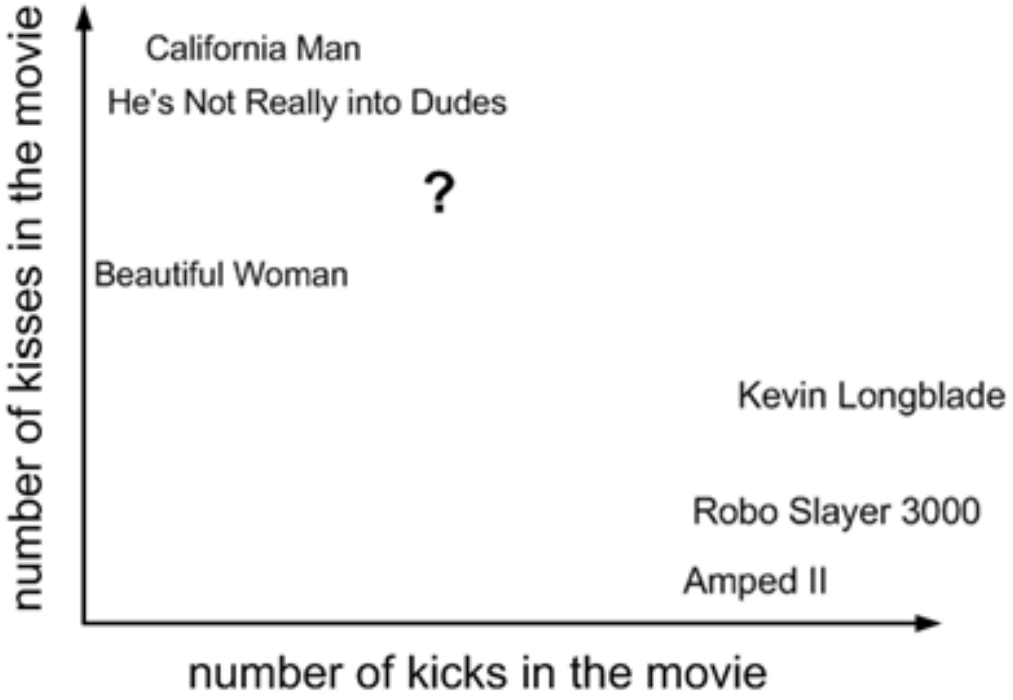


KNN-Introduction

- It assumes we have existing dataset
- Dataset has labels for all the data samples
- Given a new data sample, we compare it to all the data samples in our dataset
- We compute the most similar (i.e., nearest neighbors) data samples and check their labels
- We look at the top **k** similar data samples
- We choose the majority vote from k samples for the label of new data sample

KNN-Example

- We need to classify whether a movie is romance or action movie?



KNN-Example

- Dataset we have:

Movie title	# of kicks	# of kisses	Type of movie
California Man	3	104	Romance
He's Not Really into Dudes	2	100	Romance
Beautiful Woman	1	81	Romance
Kevin Longblade	101	10	Action
Robo Slayer 3000	99	5	Action
Amped II	98	2	Action
?	18	90	Unknown

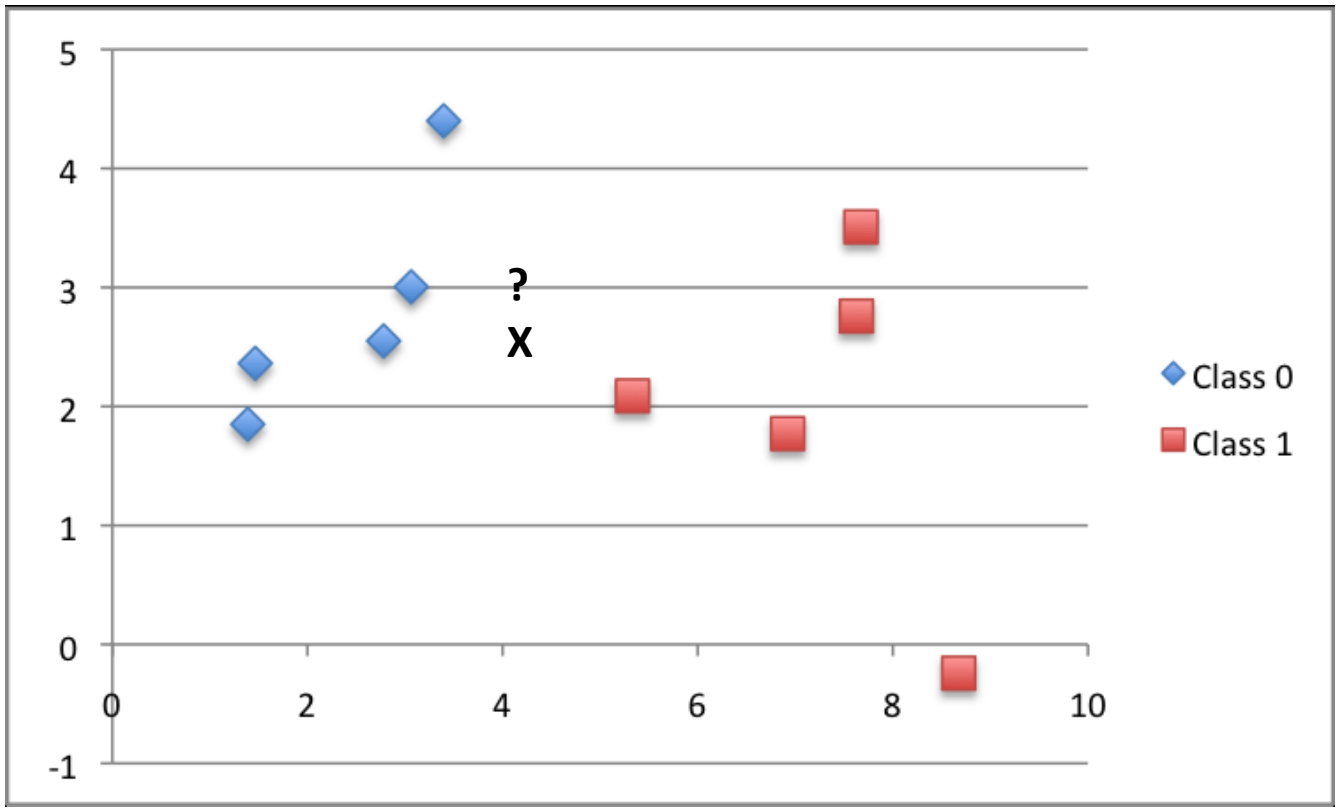
KNN-Example

- Compute distance between given sample and all other samples:

Movie title	Distance to movie "?"
California Man	20.5
He's Not Really into Dudes	18.7
Beautiful Woman	19.2
Kevin Longblade	115.3
Robo Slayer 3000	117.4
Amped II	118.9

- What if k= 3, 4 and 5?

KNN- Another example



Distance between two sample?

- Compute distance between two given sample?
- Distance between two points **x1** and **x2**:

  Euclidean Distance:

$$distance = \sqrt{\sum_{i=1}^{n}(x1_i - x2_i)^2}$$

Distance between two sample?

- **Hamming Distance:** Calculate the distance between binary vectors.
- **Manhattan Distance:** Calculate the distance between real vectors using the sum of their
- absolute difference. Also called City Block Distance.
- **Minkowski Distance:** Generalization of Euclidean and Manhattan distance.

KNN Algorithm

Steps:

1. Determine value of k
2. Compute distance between input data sample and all data sample in dataset
3. Sort (ascending) the data samples based on this distance
4. Choose the class value that has majority vote among the k-data samples

KNN Algorithm

- Use KNN algorithm for:
  - Classification of categorical values
  - Prediction of real values
- Does it have any training model? - No
- How to get k-value? – sqrt(#data_points)/2
- What if the dataset is huge, i.e. >1 million data samples?
- Note: Use k-d tree data structure to improve the look-up operations.

KNN Algorithm

- **Instance-Based Learning**
  - raw training instances are used to make predictions.
- **Lazy Learning**
  - No learning of the model is required and all of the work happens at the time a prediction is requested.
- **Nonparametric**
  - KNN makes no assumptions about the functional form of the problem being solved.

When to consider KNN?

- Simple, a good one to try first
- Less than 20 attributes per instance
- Lots of training data
- Advantages:
  - No training is needed
  - Learn complex target functions
  - Don't lose information
- Disadvantages:
  - Slow at query time
  - Easily fooled by irrelevant attributes

Curse of Dimensionality

- Imagine instances described by 20 attributes, but only 2 are relevant to target function
- **Curse of dimensionality**
  - nearest neighbor is easily mislead with high-dimensional input vector
  - i.e., distance metric is not useful in high dimensions, as all data samples are almost equidistant to the input data

Applications of KNN

- Recommendation systems
- Text mining
- Stock market forecasting