Course	ICS 352 Machine Learning
Term	Spring 2023
Week	
Date	
Chapter. Topic	

K Nearest Neighbors

Siva R Jasthi

Computer Science and Cybersecurity

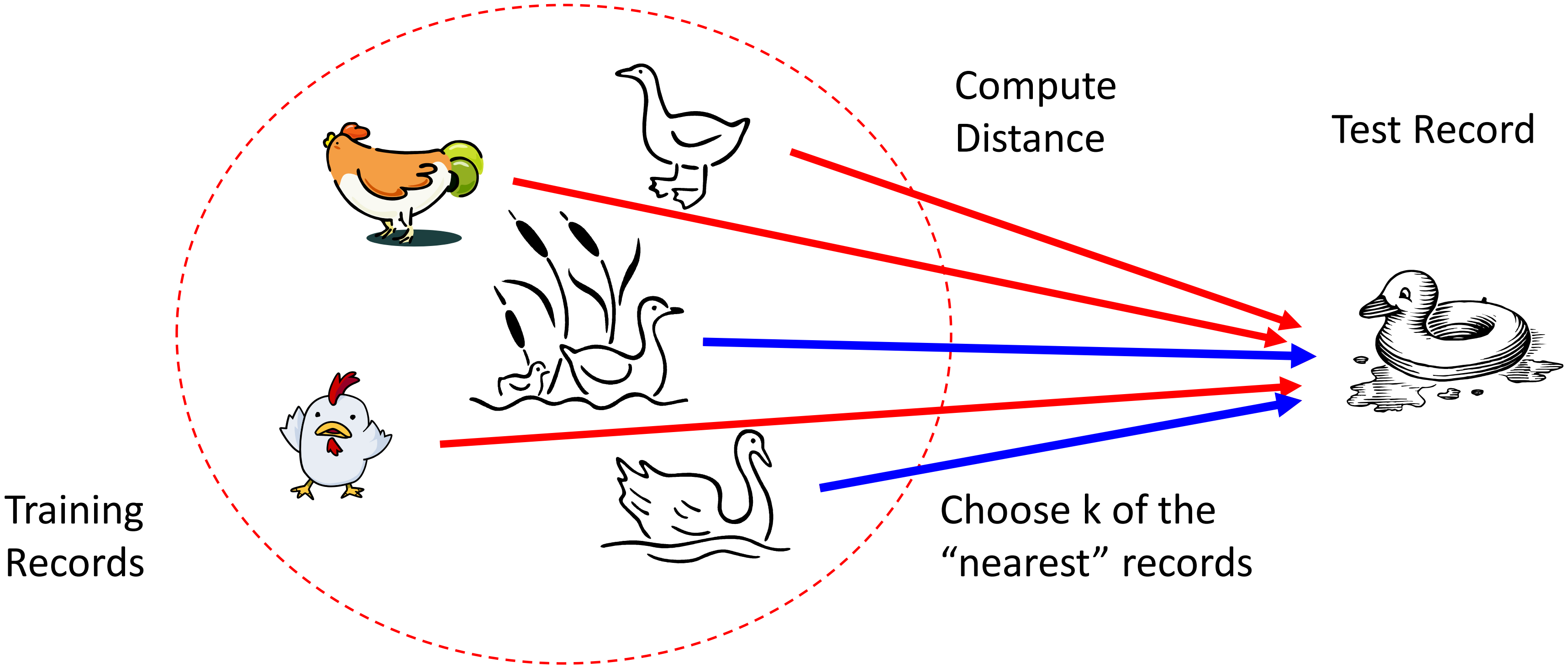
Metropolitan State University

Supervised Learning

- A training set of examples with the correct responses is provided
- Based on this training set
    - The algorithm generalizes to response correctly to all possible inputs
        - Image recognition
- Known as learning from exemplars

Nearest Neighbor Classifier

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



Compute Distance

Test Record

Training Records

Choose k of the “nearest” records

K-NN is based on

- Instance-based Learning
  - Learning=storing all training instances
  - Classification=assigning target function to a new instance

- Also, referred to as
  - Instance Based Learning
  - Lazy Learning
  - Case Based Reasoning
  - Exemplar Based Learning

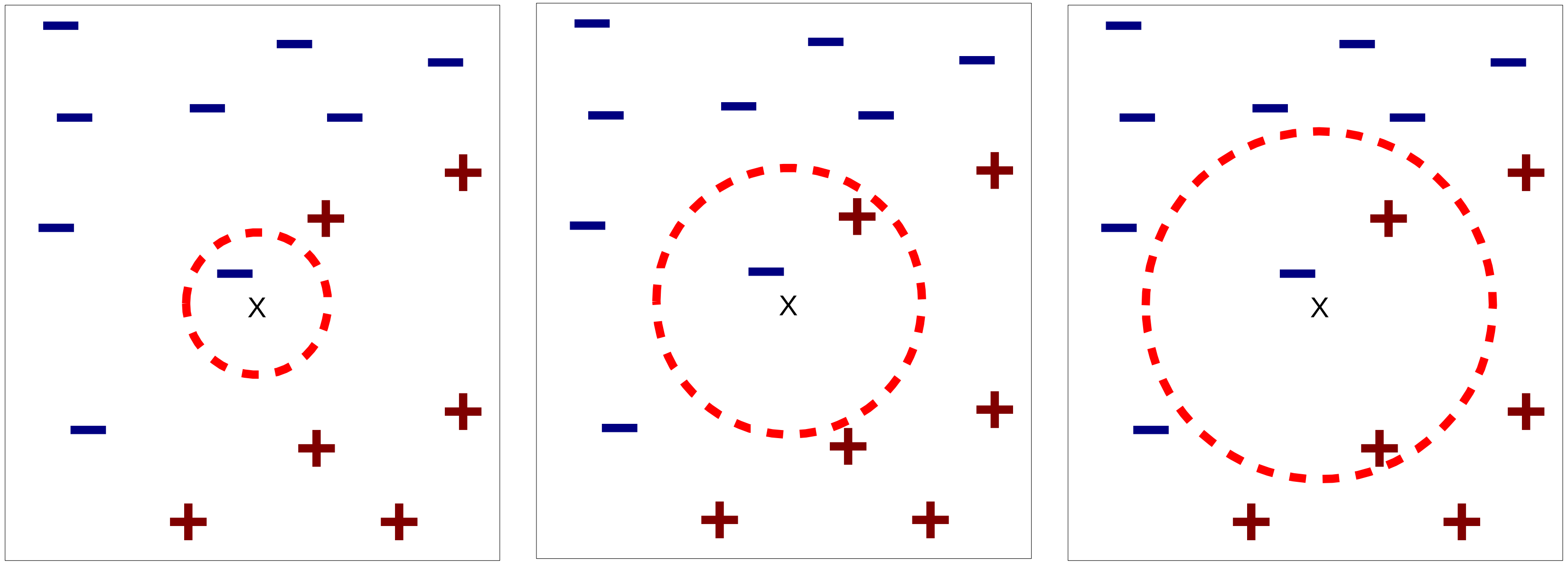
K-NN : Basic classification principle

- *k*-NN classification rule is to assign to a test sample the majority category label of its *k-nearest* training samples

- In practice, *k* is usually chosen to be odd, so as to avoid ties

- The *k* = 1 rule is generally called the nearest-neighbor classification rule

K-NN : Nearest Neighbors? How many?



(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

K-NN for Classification and Regression

- In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small).

- In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

Question: What happens if k = 1?

Question: What happens if k = 2 or 4 or 6?

Applications of KNN

- Classification: KNN is often used for classification tasks, such as image recognition, document classification, and sentiment analysis. In these applications, KNN is trained on labeled data to predict the class of new, unseen data points.
- Regression: KNN can also be used for regression tasks, such as predicting the price of a house based on its features or estimating the demand for a product based on historical sales data.
- Anomaly (Outlier) detection: KNN can be used for anomaly detection, where the goal is to identify unusual or anomalous data points that deviate significantly from the normal behavior of the system.
- Recommender systems: KNN can be used to build personalized recommender systems that suggest items or products based on the user's past behavior or preferences.
- Image retrieval: KNN can be used for content-based image retrieval, where the goal is to find images in a database that are similar to a query image based on their visual features.
- Natural language processing: KNN can be used for tasks such as text classification, document clustering, and word sense disambiguation in natural language processing.
- imputation: KNN can be used for missing value imputation, where missing values in a dataset are replaced with the average of their k-nearest neighbors.

Features of KNN

- Intuitive: KNN is easy to understand and implement. The algorithm is based on the idea that objects that are close to each other are more likely to belong to the same class.
- Non-parametric: KNN is a non-parametric algorithm, which means it does not make any assumptions about the underlying data distribution. This makes it more flexible and able to handle a wide range of data types and structures.
- Versatile: KNN can be used for both classification and regression tasks. For classification, the algorithm assigns a label to a new data point based on the labels of its k-nearest neighbors. For regression, the algorithm predicts a numerical value based on the values of its k-nearest neighbors.
- Handles multi-class problems: KNN can handle multi-class classification problems, where there are more than two classes to predict.
- Robust to noisy data: KNN is robust to noisy data, as outliers and noise tend to have less influence on the final decision due to the averaging effect of the algorithm.

Dealing with non-numerical data

- Preprocess the data to label the non-numerical data

- Use different distance measures that can operate on non-numerical data.

Hyper-parameter for KNN (K)

- K = Number of nearest neighbors
- Which K gives the best performance?

- You can build the model with different K values and find the corresponding Accuracy.

- Even values for K are avoided for obvious reasons.
- We want the majority VOTE from the nearest neighbors.
- So, we need to avoid any possible ties.

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes

Rule of thumb:
K = sqrt(N)
N: number of training points

Hyper-parameter for KNN (Distance)

What distance measure is suitable?

1 1 1 1 1 1 1 1 1 1 1 0

0 1 1 1 1 1 1 1 1 1 1 1

d = 1.4142

vs

1 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 1

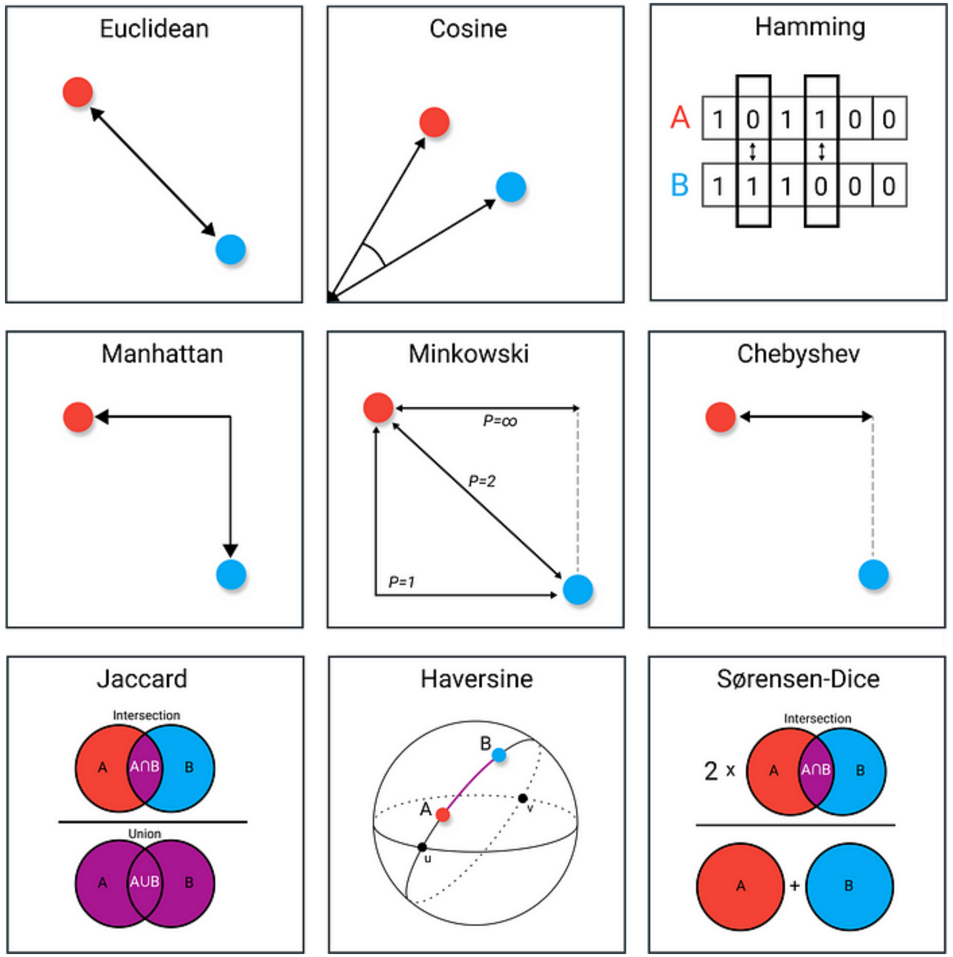
d = 1.4142

Hyper-parameter for KNN (Distance)

- Distance Metric (default = Ecuclidean)
- 'euclidean', 'manhattan', 'minkowski’, ‘jaccard’, ‘hamming’

- The choice of distance metric can have a significant impact on the performance of a KNN model, depending on the structure of the data and the problem bei

- Euclidean distance is sensitive to scale. If you have data in different scales, using Manhattan distance or the Chebyshev distance may be more appropriate.
- For categorical values, using Jaccard or Hamming distances may be more appropriate.



https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa

K-NN : Computational Complexity

- Suppose there are m instances and n features in the dataset
- Nearest neighbor algorithm requires computing m distances
- Each distance computation involves scanning through each feature value
- Running time complexity is proportional to m X n

Summary

- Very simple and intuitive ML algorithm
- Useful for both Classification and Regression problems.
- Two hyper parameters – K and Distance Metric

Any Questions?

Thank You.