Artificial intelligence: A subfield of CS, AI refers to computer programs that can

solve problems humans are good at (e.g. vision, natural language,)

Machine learning: an algorithm to automatically learn from data, or from experience, uncover patterns in data, building autonomous agents

Neural networks: A parametric model used in ML; (very loosely) based on

biological neurons

Deep learning: Neural networks with multiple layers (i.e., processing steps)

Data science: An emerging eld which applies ML techniques to domain-specific

Problems

Supervised learning: have labeled examples of the correct behavior

Semi-supervised learning: utilizes both labeled and unlabeled data

Reinforcement learning: learning system (agent) interacts with the world and learns to maximize a scalar reward signal

Unsupervised learning: no labeled examples – instead, looking for “interesting” patterns in the data

ML Workflow

1. Should I use ML on this problem?

• Is there a pattern to detect?

• Can I solve it analytically?

• Do I have data?

2. Gather and organize data

• Preprocessing, cleaning, visualizing.

3. Establishing a baseline.

4. Choosing a model, loss, regularization, ...

5. Optimization

6. Hyperparameter search.

7. Analyze performance & mistakes, and iterate back to step 4 (or 2).

Nearest Neighbors

• a supervised classification algorithm that predicts the

class of an output feature based on the class of other

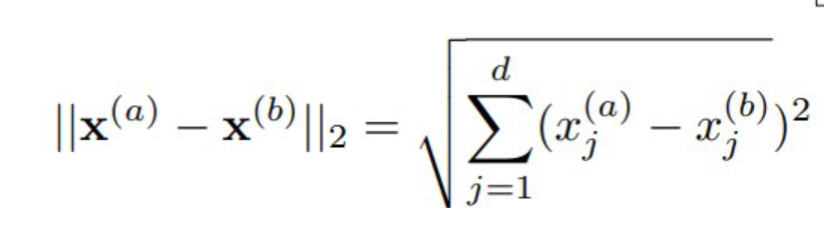
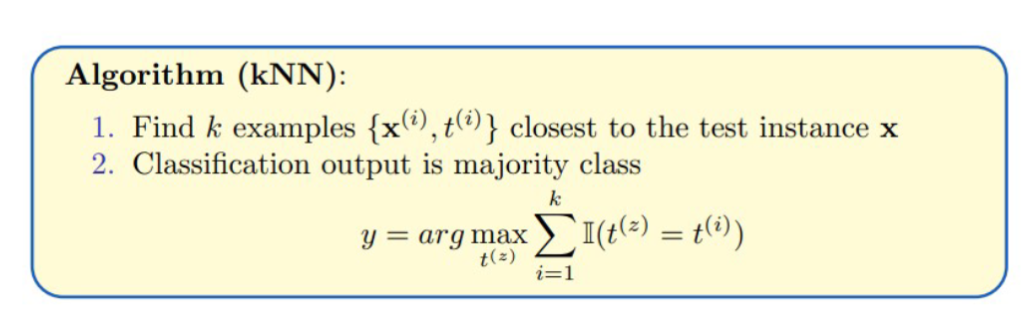
instances with the most similar, or "nearest," input

features

• neighbors, are identified using some distance measure,

and the classes of each neighbor's output feature are

identified

k-Nearest Neighbors

1.Measure the distance: Calculate the distance between the new

data point and all the data points in the dataset.

1.use the Euclidean distance, which is like measuring the straight-line

distance between two points.

2.Find the K nearest neighbors: Identify the K points with the

shortest distances to the new point. These are the K nearest

neighbors.

1.For instance, if K = 3, we select the three points that are closest to our new

point.

3.Majority voting: Among the K nearest neighbors, count how many

points are there for each class type.

1.Whichever type has the majority becomes our prediction for the new point.

1.For example, if two neighbors are class “a” and one is a class “b”, we predict that the

new point is a “a”.

K determines the tradeoff between fitting the data and overfitting the data.

Small k

Good at capturing fine-grained patterns

May overfit, i.e. be sensitive to random idiosyncrasies in the training data

Overfitting happens when a model becomes too complex, capturing noise or irrelevant patterns from the training data.

 model become overly sensitive to local variations in the training data, leading to poor generalization to unseen data.

Large k

Makes stable predictions by averaging over lots of examples

May underfit, i.e. fail to capture important regularities

Underfitting occurs when a model is too simple to capture the underlying patterns in the data

model is too generalized and oversimplifies the underlying patterns in the data.

**How to avoid Overfitting and Underfitting**

Balancing k

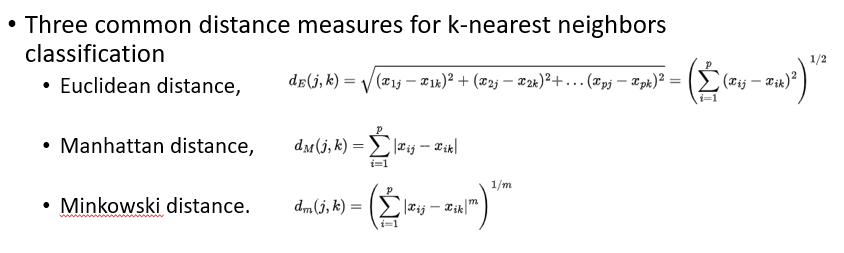
Optimal choice of k depends on number of data points n.

Nice theoretical properties if

Rule of thumb: choose

K=3

using validation set



Simple

No training

Easy to justify classification to customer.

Can easily do multi-class.

Large dataset: lazy learning technique

in training phase KNN doing nothing, so training is fast

but on time of prediction it becomes slow as large dataset come since model has to calculate euclidean distance from given point to all points in the dataset

Curse of Dimensionality:

feature space becomes increasingly sparse as the number of dimensions (features) grows

In high-dimensional spaces, the notion of proximity or similarity becomes less meaningful.

“most” points are approximately the same distance

Imbalanced dataset: the majority class typically has significantly more samples than the minority class.

large number of neighbors from the majority class can overpower the neighbors from the minority class

dominate the decision-making process,

leading to a bias towards the majority class in the predictions.

**Decision Trees**

Simple but powerful learning algorithm

DT learning: Method for learning discrete-valued target functions in which the function to be learned is represented by a decision tree.

Make predictions by splitting on features according to a tree structure.

Continuous Features: Split continuous features by checking whether that feature is greater than or less than some threshold.

Decision boundary is made up of axis-aligned planes.

Internal nodes test a feature

Branching is determined by the feature value

Leaf nodes are outputs (predictions)

Classification tree

discrete output

leaf value ym typically set to the most common value in {t(m1), . . . , t(mk )}