Nearest Neighbor Methods

Shusen Wang

K-Nearest Neighbor (KNN)

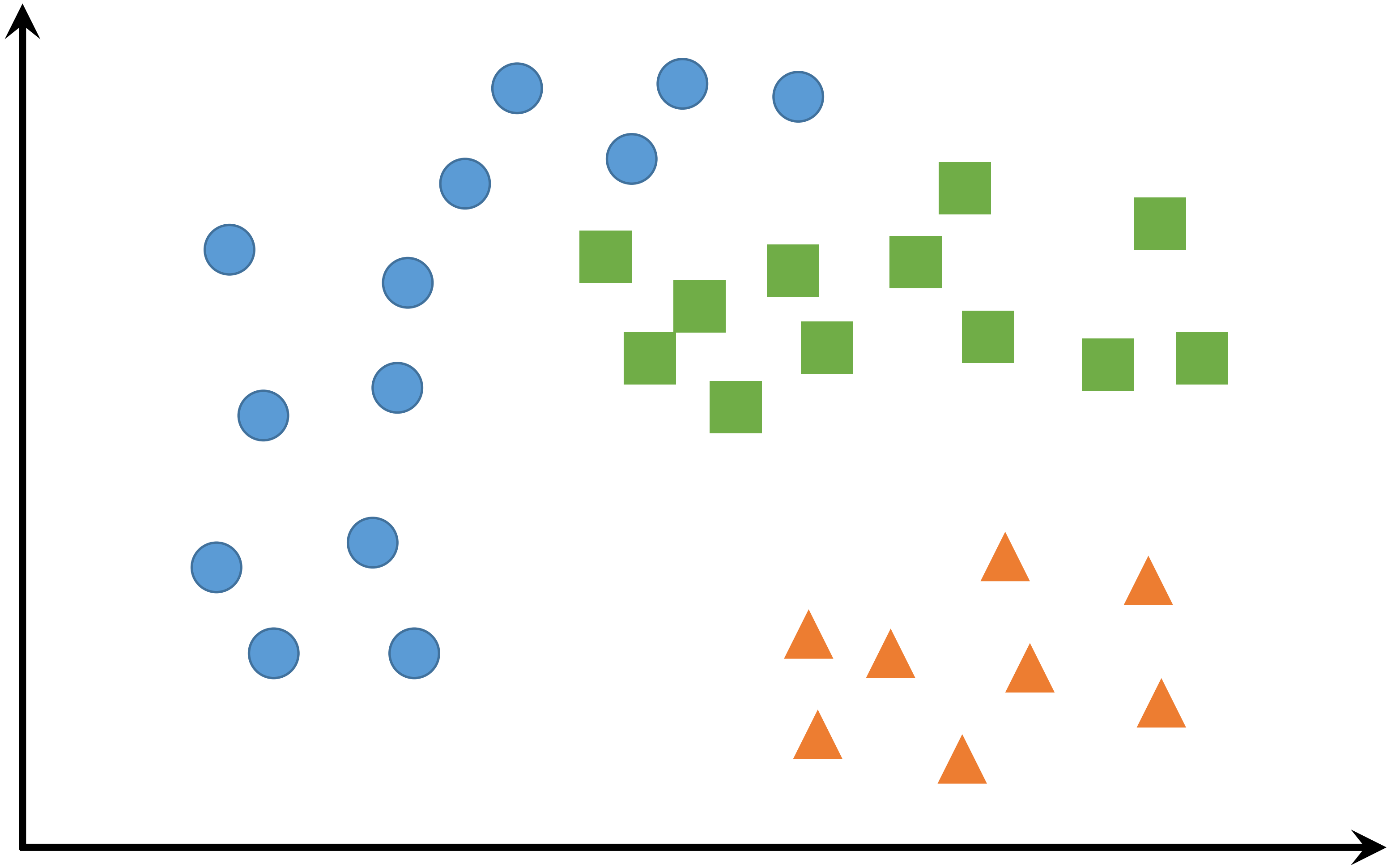
Tasks

Methods

Algorithms

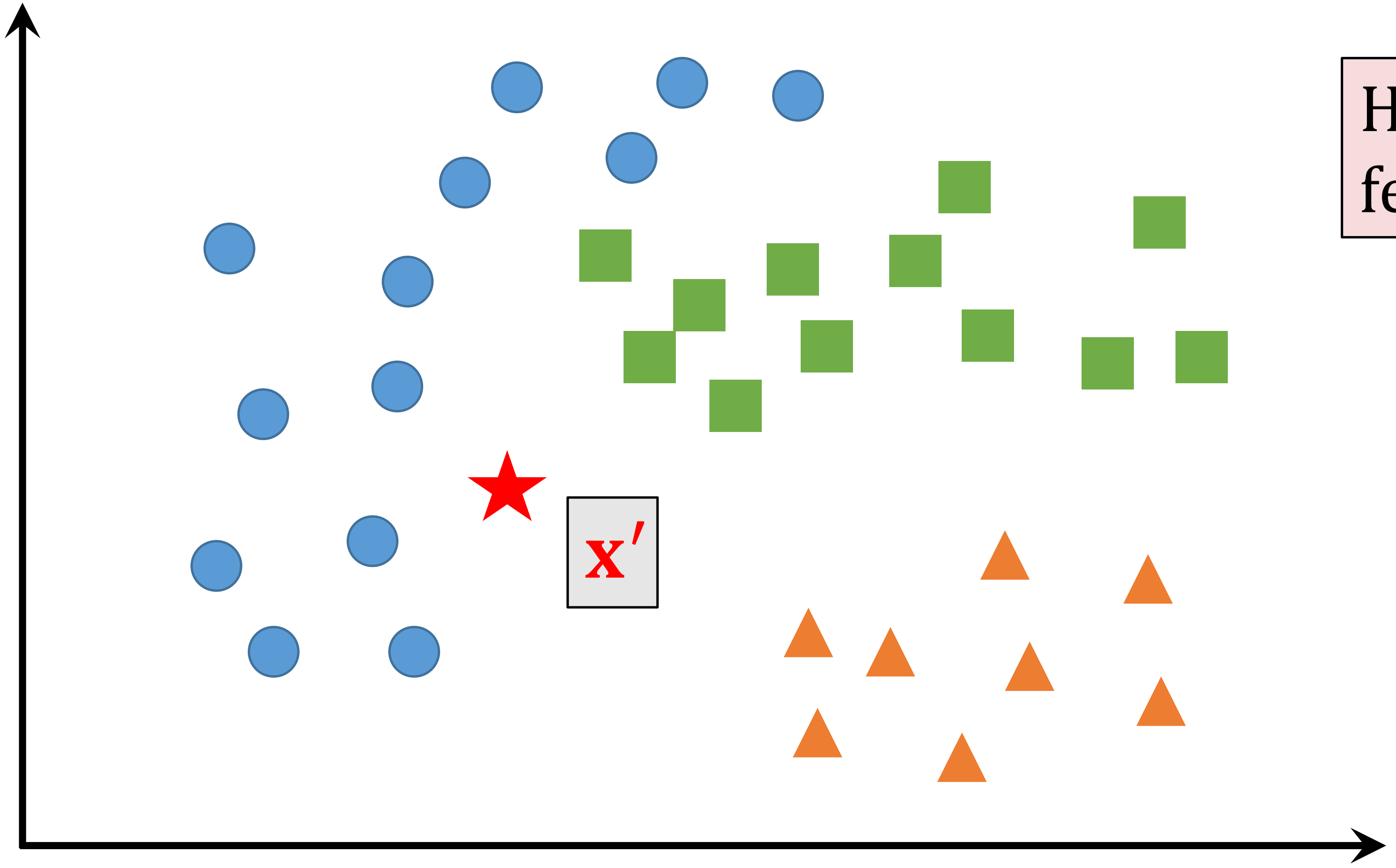
Nearest Neighbor Classifier

Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.



Nearest Neighbor Classifier

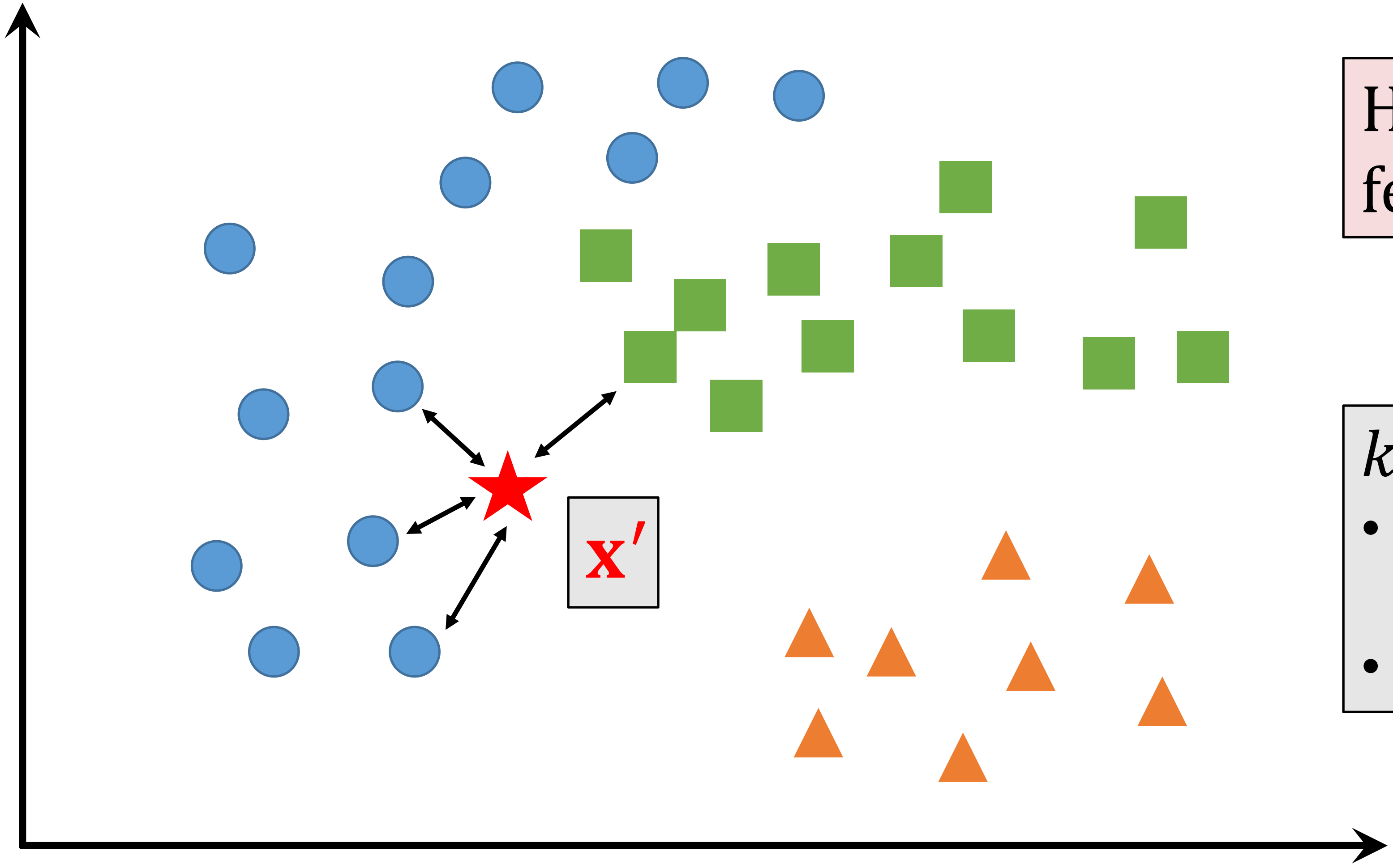
Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.



How to classify an test feature vector $\mathbf{x}'$?

Nearest Neighbor Classifier

Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.



How to classify an test feature vector $\mathbf{x}'$?

k-Nearest Neighbor (KNN):

- Find the k nearest neighbors (NN) of $\mathbf{x}'$.
- Let the k NNs vote.

Nearest Neighbor Classifier

Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.

k**-Nearest Neighbor (KNN) classifier:**

- Find the k nearest neighbors of $\mathbf{x}'$.
- Let the NNs vote.

k: hyper-parameter.

Nearest Neighbor Classifier

Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.

k-**Nearest Neighbor (KNN) classifier**:

- Find the k nearest neighbors of $\mathbf{x}'$.
- Let the NNs vote.

How to define similarity? Examples:

- Cosine similarity: $\mathrm{sim}(\mathbf{x}, \mathbf{x}') = \dfrac{\mathbf{x}^T \mathbf{x}'}{||\mathbf{x}||_2 ||\mathbf{x}'||_2}$.
- Gaussian kernel: $\mathrm{sim}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{\sigma^2} ||\mathbf{x} - \mathbf{x}'||_2^2\right)$.
- Laplacian kernel: $\mathrm{sim}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{\sigma} ||\mathbf{x} - \mathbf{x}'||_1\right)$.

Nearest Neighbor Classifier

Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.

k-**Nearest Neighbor (KNN) classifier**:

- Find the k nearest neighbors of $\mathbf{x}'$.
- Let the NNs vote.

Nearest neighbor of $\mathbf{x}'$:

$$\mathbf{x}_{\text{nearest}} = \underset{\mathbf{x} \in \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}}{\operatorname{argmax}} \ \operatorname{sim}(\mathbf{x}, \mathbf{x}') \,.$$

Nearest Neighbor Classifier

Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.

k-**Nearest Neighbor (KNN) classifier**:
- Find the k nearest neighbors of $\mathbf{x}'$.
- Let the NNs vote.

How to find the k nearest neighbors?

- Naïve algorithm
  - compute all the similarities $\mathrm{sim}(\mathbf{x}_1, \mathbf{x}'), \cdots, \mathrm{sim}(\mathbf{x}_n, \mathbf{x}')$
  - Sort the scores and find the top k.
  - $O(nd)$ time complexity (n: #samples, d: # features).
- Efficient algorithms (to be discussed later).

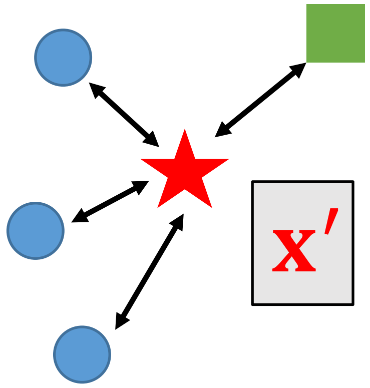
Nearest Neighbor Classifier

Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.

k-**Nearest Neighbor (KNN) classifier**:
- Find the k nearest neighbors of $\mathbf{x}'$.
- Let the NNs vote.

How to vote? Examples:

- Every neighbor has the same weight.



Nearest Neighbor Classifier

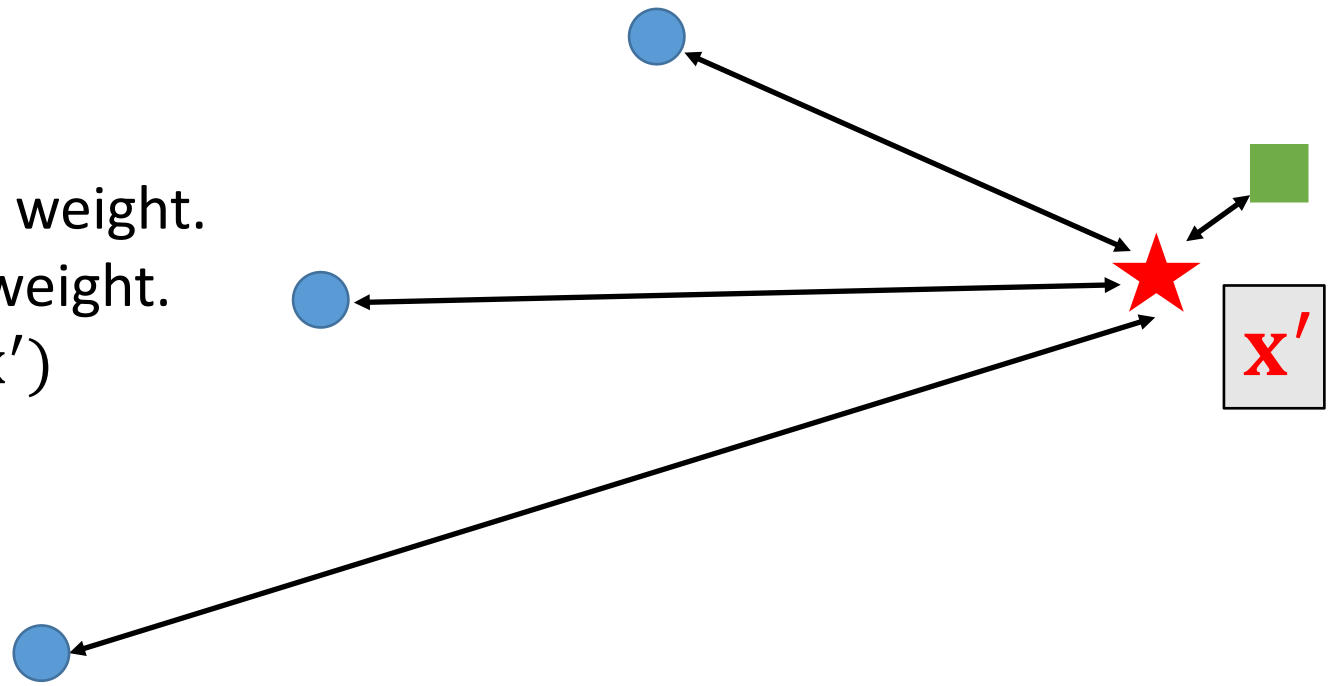
Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.

k-**Nearest Neighbor (KNN) classifier**:

- Find the k nearest neighbors of $\mathbf{x}'$.
- Let the NNs vote.

How to vote? Examples:

- Every neighbor has the same weight.
- Nearer neighbor has higher weight.
  - E.g., $\text{weight}_i = \text{sim}(\mathbf{x}_i, \mathbf{x}')$



Nearest Neighbor Classifier

Tasks

Methods

Algorithms

KNN: Naïve Algorithm

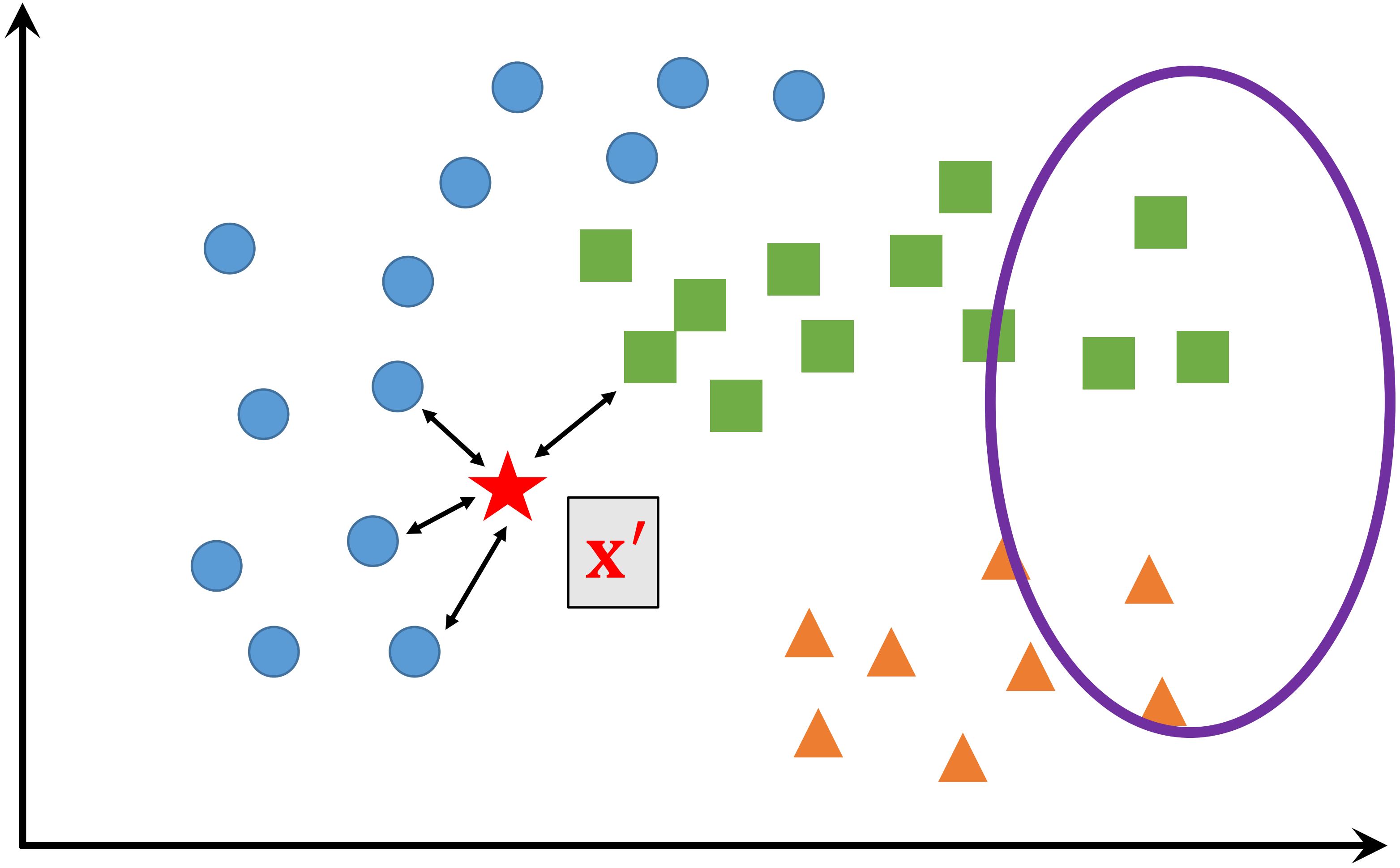
Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.

Algorithm: find the k nearest neighbors to $\mathbf{x}'$.

- Naïve algorithm
  - compute all the similarities $\text{sim}(\mathbf{x}_1, \mathbf{x}'), \cdots, \text{sim}(\mathbf{x}_n, \mathbf{x}')$ and find the top k.
- Training: no training at all.
- Test: for each query, $O(nd)$ time complexity

KNN: Efficient Algorithm

Input: feature vectors $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \cdots, y_n \in \mathbb{N}$.



Does the computer have to look at this region?

Question: find your nearest post office (given longitude & latitude).

Question: find your nearest post office (given longitude & latitude).

Data: $n = 30{,}000$ post offices' latitude and longitude:

- Post office 1: $(\mathrm{lat}_1,\ \mathrm{lon}_1)$
- Post office 2: $(\mathrm{lat}_2,\ \mathrm{lon}_2)$
- Post office 3: $(\mathrm{lat}_3,\ \mathrm{lon}_3)$
- Post office 4: $(\mathrm{lat}_4,\ \mathrm{lon}_4)$

$\vdots$

- Post office n: $(\mathrm{lat}_n,\ \mathrm{lon}_n)$

Query: your own latitude and longitude:

- $(40.74627,\ \ -74.02431)$

Question: find your nearest post office (given longitude & latitude).

Data: $n = 30{,}000$ post offices' latitude and longitude:

Query: your own latitude and longitude:

- Post office 1: $(\mathrm{lat}_1,\ \mathrm{lon}_1)$
- Post office 2: $(\mathrm{lat}_2,\ \mathrm{lon}_2)$
- Post office 3: $(\mathrm{lat}_3,\ \mathrm{lon}_3)$
- Post office 4: $(\mathrm{lat}_4,\ \mathrm{lon}_4)$

$\vdots$

- Post office n: $(\mathrm{lat}_n,\ \mathrm{lon}_n)$

- $(40.74627,\ \ \text{-}74.02431)$

Question: Which is your nearest post office?

Vector Quantization for KNN

Training:

1. Vector quantization (build landmarks)

Vector Quantization for KNN

Training:

1. Vector quantization (build landmarks)
2. Assign each post office to its nearest landmarks.

Vector Quantization for KNN

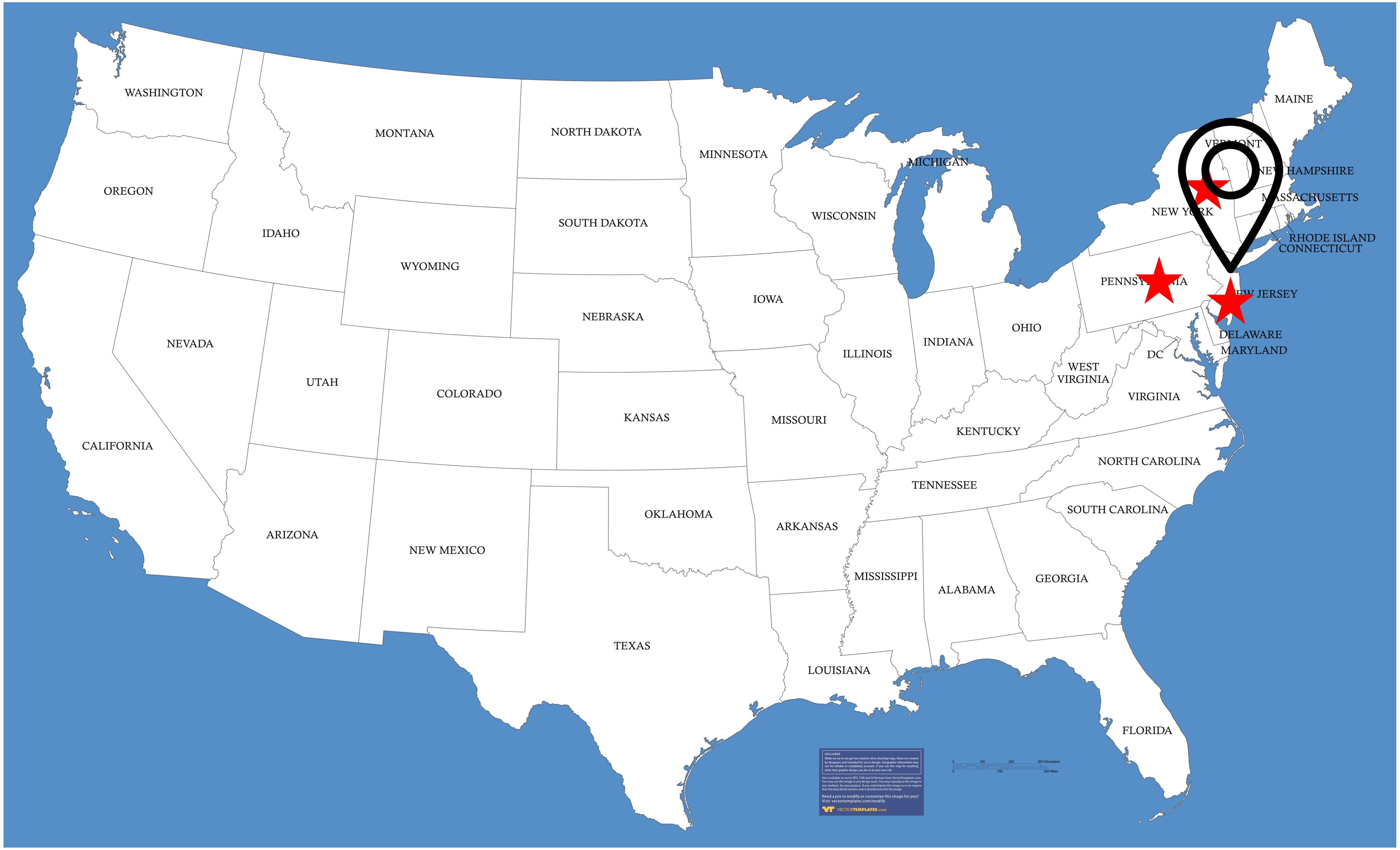
Training:

1. Vector quantization (build landmarks)
2. Assign each post office to its nearest landmarks.

Test

1. Compare your location with all the landmarks and find the nearest landmarks.

Vector Quantization for KNN



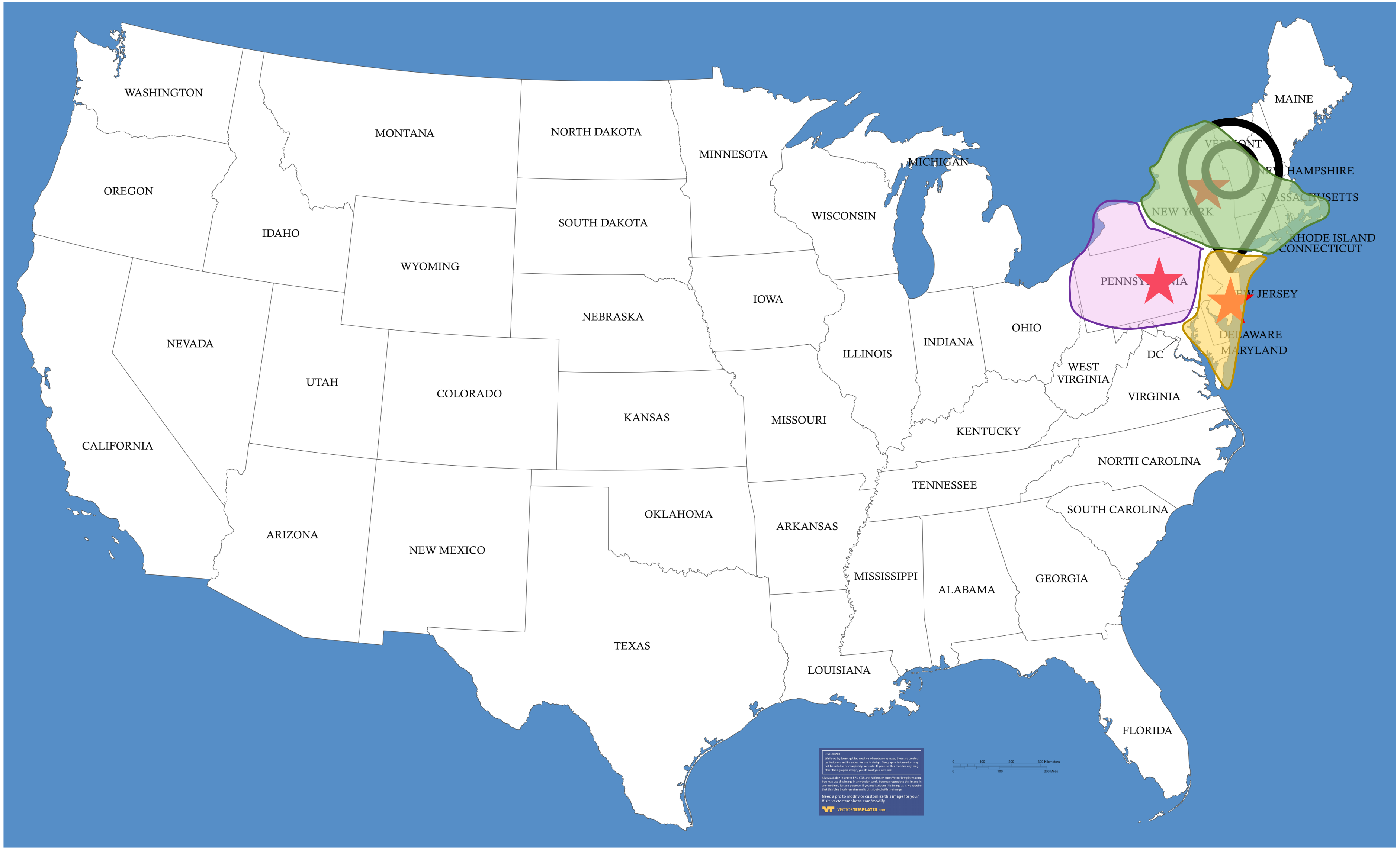
Training:

1. Vector quantization (build landmarks)
2. Assign each post office to its nearest landmarks.

Test

1. Compare your location with all the landmarks and find the nearest landmarks.

Vector Quantization for KNN



Training:

1. Vector quantization (build landmarks)
2. Assign each post office to its nearest landmarks.

Test

1. Compare your location with all the landmarks and find the nearest landmarks.
2. Compare with the postal offices assigned to the landmarks.

KNN: Efficient Algorithms

- Fast algorithms
  - Vector Quantization
  - KD-tree
  - Locality sensitive hashing

- More resources:
  - KNN Search (Wikipedia)

Summary

- KNN method for multi-class classification.
- KNN’s advantage over Softmax classifier:
  - When #class is huge, Softmax classifier is expensive.
  - E.g., in the face recognition problem, #class can be millions.

Summary

- Training: partition the feature space to regions.

- Prediction (for a test feature vector $\mathbf{x}'$):
  1. Find the nearest regions.
  2. Retrieve all the training feature vectors in the regions.
  3. Compare $\mathbf{x}'$ with the retrieved feature vectors (using similarity score) and return the k nearest.
  4. Weighted/unweighted votes by the k nearest neighbors.