УНИВЕРСИТЕТ ИТМО

Lecture 2
Non-parametric and metric methods

Machine Learning Camp

Sergey Muravyov

January 26, 2024

Lecture outline

Similarity-based classification

kNN - Nearest Neighbours method

Kernel smoothing

Non-parametric regression

Other problems

Lecture outline

Similarity-based classification

kNN - Nearest Neighbours method

Kernel smoothing

Non-parametric regression

Other problems

Classification problem (revise)

Given

- X is an object set
- Y is an answer set
- $y : X \to Y$ is an unknown dependency
- $\mathcal{D} = \{(x_i, y_i)\}$ is a set of examples

Problem

- To find an algorithm $a : X \to Y$, approximating y at X.

Data assumptions

Duck test

- If it looks like a duck, swims like a duck, and quacks like a duck, then it probably is a duck.

Hypothesis of compactness

- Objects from one class are similar to each other.

Formalization of "similarity"

- "Similarity" is a distance between objects. We will talk about **metrics**.
- **Metric space** is a set with a metric, defined on it

$$\rho(a, b) : X \times X \to [0; +\infty)$$

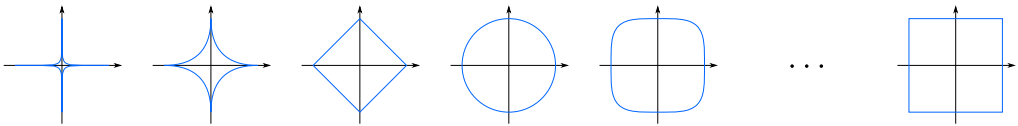
- **Metric axioms**: symmetry, triangle inequality, distinguishability of not identical objects, distinguishability of indetical objects.

Minkowski distance

- General case ($p \in \mathbb{R}_+$):

$$\rho(a,b) = \left(\sum_i |a_i - b_i|^p \right)^{\frac{1}{p}}$$

- In case of $p = 1$ it is the **Manhattan distance**: $\rho(a,b) = \sum_i |a_i - b_i|$
- In case of $p = 2$ it is the **Euclidean distance**: $\rho(a,b) = \sqrt{\sum_i (a_i - b_i)^2}$
- In case of $p = \infty$ it is the **Chebyshev distance**: $\rho(a,b) = \max_i |a_i - b_i|$



Cosine metrics

- **Cosine similarity (NOT distance):**

$$\text{CosineSimilarity}\,(a,b) = \frac{\sum_i a_i \cdot b_i}{\sqrt{\sum_i a_i^2}\sqrt{\sum_i b_i^2}} \in [-1;+1]$$

- **Cosine distance:**

$$\text{CosineDistance}\,(a,b) = 1 - \text{CosineSimilarity} \in [0;2]$$

- If the given data set is normalized in a «wrong» way:

$$\text{CosineSimilarity}\,(a,b) = \sum_i a_i \cdot b_i = \langle a,b \rangle$$

Mahalanobis distance

$$\rho(a,b) = \sqrt{(a-b)^T S^{-1} (a-b)}$$

- S is a covariance matrix of space X $(a, b \in X)$
- S is rated on the training data set
- Metric with built in normalization

Lecture outline

Similarity-based classification

kNN - Nearest Neighbours method

Kernel smoothing

Non-parametric regression

Other problems

1NN - One Nearest Neighbour method

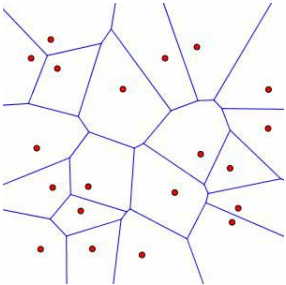
- Assuming $x_{(u,1)}$ is the **nearest neighbour** of object u:

$$x_{(u,1)} = \arg\min_{x \in D_{train}} \rho(u, x).$$

- **(One) nearest neighbour** classifier:

$$a_{1\text{NN}}(u, \mathcal{D}_{train}) = y\left(x_{(u,1)}\right).$$

- **Voronoi diagram**:



1NN analysis

Advantages:

- simplicity;
- lucidity;
- interpretability.

Disadvantages:

- sensibility to noise;
- low efficiency;
- no training parameters (explicitly defined);

kNN - k-Nearest Neighbours method

- Let's choose distance ρ and number of neighbours k.
- Then sort objects:

$$\rho\left(u, x_{(u,1)}\right) \leq \rho\left(u, x_{(u,2)}\right) \leq \cdots \leq \rho\left(u, x_{(u,|\mathcal{D}_{train}|)}\right).$$

- **kNN algorithm:**

$$a_{k\text{NN}}\left(u; \mathcal{D}_{train}\right) = \arg\max_{y \in Y} \sum_{i=1}^{k} \left[y\left(x_{(u,i)}\right) = y\right] = \text{Mode}\left\{y(x_{(u,i)})\right\}_{i=1}^{k}$$

$$a_{k\text{NN}}\left(u; \mathcal{D}_{train}\right) = \arg\max_{y \in Y} \sum_{i=1}^{|\mathcal{D}_{train}|} \left[y\left(x_{(u,i)}\right) = y\right] [i \leq k]$$

- There may be an uncertainty if $|Y| > 2$ or k is even.

Generalized metric classifier

- Assuming that $w_{(i,u)}$ is function representing importance of i-th neighbour of u, a posteriori weight.
- Not to be confused with a priori weights w_i which do not depend on request.
- If we need to take into account a priori and a posteriori weights then $w_{(i,u)} = w_i \cdot w'_{(i,u)}$.
- **Algorithm:**

$$a_{\text{GenDistClassifier}}\left(u; \mathfrak{D}_{train}\right) = \arg\max_{y \in Y} \sum_{i=1}^{|\mathfrak{D}_{train}|} \left[y\left(x_{(u,i)}\right) = y\right] w_{(i,u)}$$

- We can think of $\sum_{i=1}^{|\mathfrak{D}_{train}|} \left[y\left(x_{(u,i)}\right) = y\right] w_{(i,u)}$ as about rate of similarity of object u to class y.

Lecture outline

Similarity-based classification

kNN - Nearest Neighbours method

Kernel smoothing

Non-parametric regression

Other problems

Kernel function

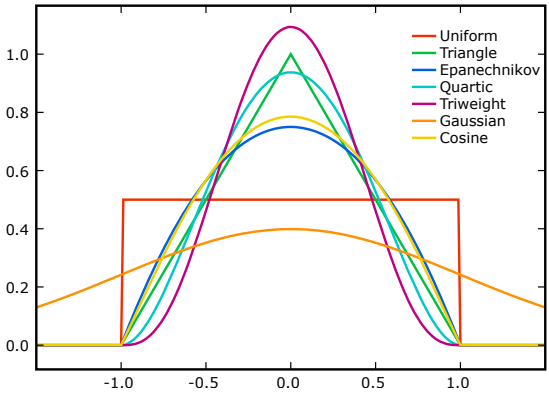
Kernel function $K(x)$ is a symmetric non-negative function, $\int_{-\infty}^{+\infty} K(x)\,dx = 1$.

$$\text{Uniform}(u) = \frac{1}{2} \cdot [|u| < 1]$$

$$\text{Triangular}(u) = (1 - |u|) \cdot [|u| < 1]$$

$$\text{Epanechnikov}(u) = \frac{3}{4}\left(1 - u^2\right) \cdot [|u| < 1]$$

$$\text{Gaussian}(u) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-u^2}{2}\right)$$



One dimension case

If $Pr([a,b])$ if a probability measure on $[a,b]$ then

$$p(x) = \lim_{h \to 0} \frac{1}{2h} Pr([x-h, x+h]).$$

Is an empirical assessment of probability with window of width h

$$\widehat{p_h}(x) = \frac{1}{2nh} \sum_{i=1}^{n} [|x - x_i| < h].$$

Parzen–Rosenblatt window

An empirical assessment of probability with window h:

$$\widehat{p_h}(x) = \frac{1}{2hn} \sum_{i=1}^{n} \left[\frac{|x - x_i|}{h} < 1 \right].$$

Parzen–Rosenblatt estimation with window of width h:

$$\widehat{p_h}(x) = \frac{1}{2hn} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h} \right),$$

where $K(r)$ is some kernel function.
$\widehat{p_h}(x)$ converges to $p(x)$.

Generalizing for n-dimensional case

- If objects are described by m real features $f_j : X \to \mathbb{R}, j = 1, \ldots, m$,

$$\widehat{p_h}(x) = \frac{1}{n} \sum_{i=1}^{n} \prod_{j=1}^{m} \frac{1}{h_j} K\left(\frac{f_j(x) - f_j(x_i)}{h_j}\right).$$

- If X is a metric space with the distance measure $\rho(x, x')$:

$$\widehat{p_h}(x) = \frac{1}{nV(h)} \sum_{i=1}^{n} K\left(\frac{\rho(x, x_i)}{h}\right),$$

where $V(h) = \int_X K\left(\frac{\rho(x,x_i)}{h}\right) dx$ is a normalization factor.

Parzen–Rosenblatt window for classification

With fixed window width:

$$a\left(u; \mathfrak{D}_{train}; h; K\right) = \arg\max_{y \in Y} \sum_{i=1}^{|\mathfrak{D}_{train}|} \left[y\left(x_{(u,i)}\right) = y\right] K\left(\frac{\rho\left(u, x_{(u,i)}\right)}{h}\right)$$

With variable window width:

$$a\left(u; \mathfrak{D}_{train}; k; K\right) = \arg\max_{y \in Y} \sum_{i=1}^{|\mathfrak{D}_{train}|} \left[y\left(x_{(u,i)}\right) = y\right] K\left(\frac{\rho\left(u, x_{(u,i)}\right)}{\rho\left(u, x_{(u,k+1)}\right)}\right)$$

Lecture outline

Similarity-based classification

kNN - Nearest Neighbours method

Kernel smoothing

Non-parametric regression

Other problems

Regression problem (revise)

- X is an object set, Y is an answer set
- $y : X \to Y$ is an unknown dependency, $Y \subseteq \mathbb{R}$
- $\mathcal{D} = \{(x_i, y_i)\}$ is a set of instances.
- **Problem:** to find an algorithm $a : X \to Y$, approximating y at X.
- Hypothesis of compactness changes with hypothesis of continuity.

Nadaraya–Watson formula

Main idea

- Let's think that $a_\theta\left(x, \mathcal{D}_{train}\right) = \theta$ nearby u.
- Let's guess that empirical risk equals to:

$$\mathcal{L}(\theta) = \sum_{i=1}^{|\mathcal{D}_{train}|} w_{(i,u)}(\theta - y(x_i))^2$$

- If we solve the problem of minimisation $\mathcal{L}$ relative to θ analytically, then the optimal solutuion is the weighted average.

Nadaraya–Watson kernel smoothing:

$$a_{\text{NPR}}\left(u, \mathcal{D}_{train}\right) = \frac{\sum_{i=1}^{|\mathcal{D}_{train}|} y_i w_{(i,u)}}{\sum_{i=1}^{|\mathcal{D}_{train}|} w_{(i,u)}} = \frac{\sum_{x_i \in \mathcal{D}_{train}} y_i K\left(\frac{\rho(x_i,u)}{h}\right)}{\sum_{x_i \in \mathcal{D}_{train}} K\left(\frac{\rho(x_i,u)}{h}\right)}.$$

Basic theorem

If

- Sample $\mathfrak{D}_{train}$ is simple, distributed with $p(x, y)$
- $\int_0^\infty K(r)\,dr < \infty$, $\lim_{r\to\infty} rK(r) = 0$;
- $\forall x \in X : E(y^2|x) < \infty$;
- $\lim_{i\to\infty} h_i = 0$, $\lim_{i\to\infty} ih_i = \infty$,

Then

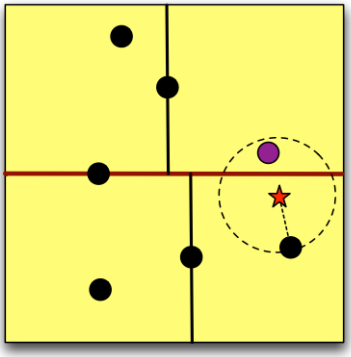
- $a_{\text{NonParamRegh}}(x, \mathfrak{D}_{train}) \xrightarrow{P} E(y|x)$ in any $x \in X$, where $E(y|x)$, $p(x)$, $D(y|x)$ are continuing, $p(x) > 0$.

Classification and regression methods analysis

- Theoretically we do not need a triangle inequality.
- Modality of the method depends on data structure of storing nearest neighbours.
- k-NN is a regression with one-hot encoding.
- Choice of kernel function impacts on smoothness of a loss function.
- Choice of kernel function has small impacton approximation quality.
- Choice of h and k impacts on approximation quality.
- The bigger h or k we take, the smaller model complexity is. If $k = n$ and we use rectangular kernel, then the algorithm is equivalent to the naive one (the answer is always equals to an average value of the data set).

Data structures for storing data

- Complexity of the algorithm is related to complexity of finding nearest neighbours.
- The most common are ***k-d-trees.***



Lecture outline

Similarity-based classification

kNN - Nearest Neighbours method

Kernel smoothing

Non-parametric regression

Other problems

Few-Shot (One-Shot) learning

- Training on a small sample size
- ~~Training from the first attempt~~
- ~~One attempt of training~~
- Bigger number of classes
- Set of classes can change
- Small number of samples for each class
- **Main problem** (classification) is solved using 1NN
- **Sub-problem** (feature extraction): to build a space, where we will be calculation distances.

Non-parametric generation of objects (SMOTE)

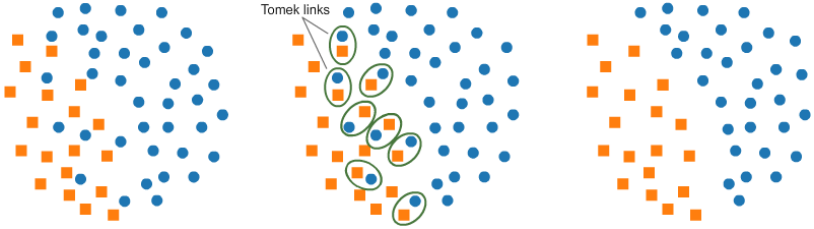
Wrong name: Synthetic Minority Over-sampling Technique

1. Randomly choose point a
2. Choose k nearest points from its class
3. Randomly choose one of them, b
4. Randomly choose a point in the interval (a, b)
5. Add it with the same label as a has

Search of Tomek Links

Tomek Links

- Pairs of objects from different classes that are close to each other.
- Such objects most likely break the compactness hypothesis.



Strategies:

- Remove both objects.
- Remove an object from the majoritarian class.
- Change label of an object from the majoritarian class.

LOWESS

- Instead of removing objects we can take them into account with lower prior (local) weight w_i.

$$w_i = K(y(x_i) - a_{kNN}(x_i, D_{train} \setminus \{x_i\}))$$

- Weighting can be repeated iteratively.
- Kernel function K may differ from kernel function of the a algorithm.
- Theoretically we can use any algorithm a.
- Practically we use kNN, because it allows us to change training set D_{train} effectively.

Thanks for your attention!

ITMOre than a UNIVERSITY