Introduction to machine learning

Victor Kitov

v.v.kitov@yandex.ru

Course information

- Instructor - Victor Vladimirovich Kitov
- Tasks of the course
- Structure:
  - lectures, seminars
  - assignements: theoretical, labs, competitions
  - exam
- Tools
  - python
  - ipython notebook
  - numpy, scipy, pandas
  - matplotlib, seaborn
  - scikit-learn.

Recommended materials

- Лекции К.В.Воронцова (видео-лекции и материалы на machinelearning.ru)
- **The Elements of Statistical Learning: Data Mining, Inference, and Prediction.** Trevor Hastie, Robert Tibshirani, Jerome Friedman, 2nd Edition, Springer, 2009. http://statweb.stanford.edu/~tibs/ElemStatLearn/.
- **Statistical Pattern Recognition.** 3rd Edition, Andrew R. Webb, Keith D. Copsey, John Wiley & Sons Ltd., 2011.

- Any additional public sources:
    - wikipedia, articles, tutorials, video-lectures.
- Practical questions:
    - stackoverflow.com, sklearn documentation, kaggle forums.

Table of Contents

Formal definitions of machine learning

- Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed.
- A computer program is said to learn from **experience E** with respect to some **class of tasks T** and **performance measure P**, if its performance P at tasks in T improves with experience E.
- Examples from text analysis: spell checker, spam filtering, POS tagger.

Major niches of ML

- dealing with huge datasets with many attributes (text categorization)
- hard to formulate explicit rules (image recognition)
- further adaptation to usage conditions is required (voice detection)
- fast adaptation to changing conditions (stock prices prediction)

Examples of ML applications

- WEB
  - Web-page ranking
  - Spam filtering
    - e-mails
    - search results
- Networks monitoring
  - Intrusion detection
  - Anomaly detection
- Business
  - Fraud detection
  - Churn prediction
  - Credit scoring
  - Stock prices / risks forecsting

Examples of ML applications

- Texts
  - Document classification
  - POS tagging, semantic parsing,
  - named entities detection
  - sentimental analysis
  - automatic summarization
- Images
  - Handwriting recognition
  - Face detection, pose detection
  - Person identification
  - Image classification
  - Image segmentation
  - Adding artistic style
- Other
  - Target detection / classification
  - Particle classification

Table of Contents

General problem statement

- Set of objects O
- Each object is described by a vector of known characteristics $\mathbf{x} \in \mathcal{X}$ and predicted characteristics $y \in \mathcal{Y}$.

$$o \in O \longrightarrow (\mathbf{x}, y)$$

- Usually $\mathcal{X} = \mathbb{R}^D$, $\mathcal{Y}$ - a scalar, but they may be any structural descriptors of objects in general.

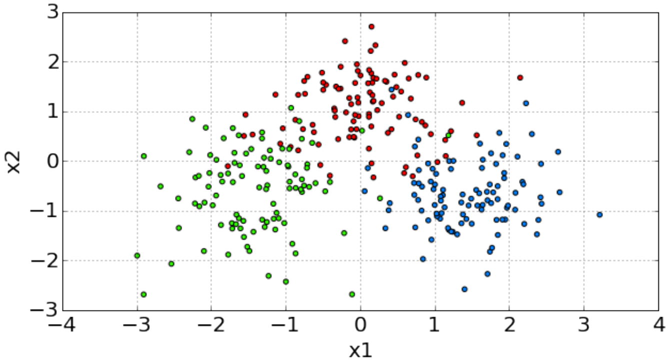
General problem statement

- Task: find a mapping f, which could accurately approximate $\mathcal{X} \to \mathcal{Y}$.
    - using a finite «training» set of objects with known (x, y).
    - to apply on a set of objects of interest
- Questions solved in ML:
    - how to select object descriptors - features
    - in what sense a mapping f should approximate true relationship
    - how to construct f

Variants of problem statement

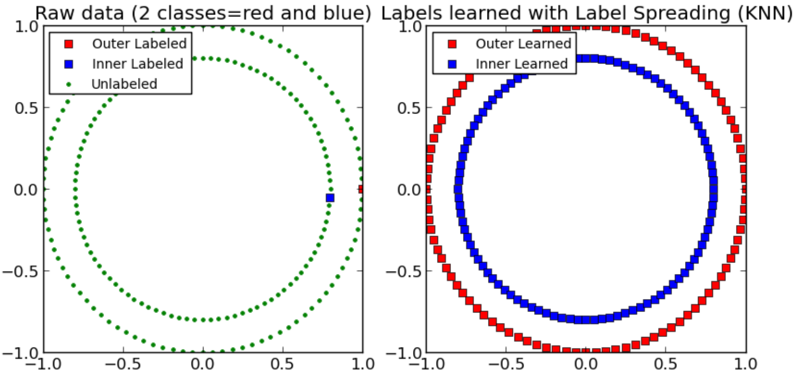
- For each new object x need to associate y.
- What is known:
    - $(x_1, y_1), (x_2, y_2), ...(x_N, y_N)$ - supervised learning:
    - $x_1, x_2, ...x_N$ - unsupervised learning
        - dimensionality reduction
        - clustering
    - $(x_1, y_1), (x_2, y_2), ...(x_N, y_N), x_{N+1} x_{N+2}, ...x_{N+M}$ - semi-supervised learning.
- If predicted objects $x_1', x_2', ...x_K'$ for which y is forecasted, are known in advance, then this is «transductive» learning.

Example of supervised classification



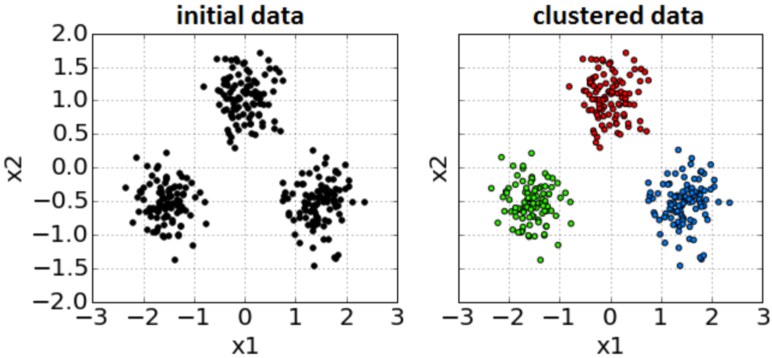
Supervised learning: $x = (x_1, x_2)$, y is shown with color

Example of semi-supervised classification



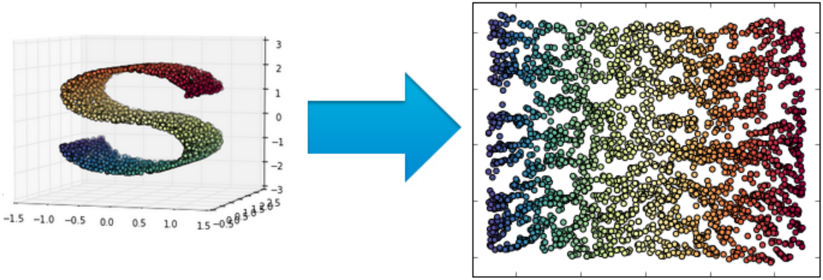
Semi-supervised learning.

Example of clustering (unsupervised)



Unsupervised learning: clustering

Dimensionality reduction (unsupervised)



Unsupervised learning: dimensionality reduction

Generative and discriminative models[1]

Generative model

Full distribution $p(x, y)$ is modeled.

- Can generate new observations (x, y)

$$\begin{aligned}\widehat{y}(x) &= \arg\max_y p(y|x) = \arg\max_y \frac{p(x,y)}{p(x)} = \arg\max_y p(y)p(x|y) \\ &= \arg\max_y \{\log p(y) + \log p(x|y)\}\end{aligned}$$

Discriminative model

- **Discriminative with probability**: only $p(y|x)$ is modeled
- **Reduced discriminative**: only $y = f(x)$ is modeled.

[1] Which is more general problem statement and which - more specific?

Generative and discriminative - discussion

- **Disadvantages of generative models:**
    - Discriminative models are more general
    - $p(x|y)$ may be inaccurate in high dimensional spaces

Generative and discriminative - discussion

- **Disadvantages of generative models:**
  - Discriminative models are more general
  - $p(x|y)$ may be inaccurate in high dimensional spaces
- **Advantages of generative models:**
  - Generative models can be adjusted to varying $p(y)$
  - Naturally adjust to missing features (by marginalization)
  - Easily detect outliers (small $p(x)$)

Types of features

- Full object description $\mathbf{x} \in \mathcal{X}$ consists of individual features $x_i \in \mathcal{X}_i$
- Types of feature:
  - $\mathcal{X}_i = \{0, 1\}$ - binary feature
  - $|\mathcal{X}_i| < \infty$ - discrete (nominal) feature
  - $|\mathcal{X}_i| < \infty$ and $\mathcal{X}_i$ is ordered - ordinal feature
  - $\mathcal{X}_i = \mathbb{R}$ - real feature

Types of target variable

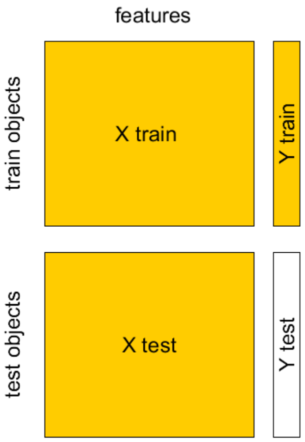
- Types of target variable:
  - $\mathcal{Y} = \mathbb{R}$ - regression (in supervised learning)
  - $\mathcal{Y} = \mathbb{R}^M$ - vector regression (in supervised learning) or feature extraction (in unsupervised learning)
  - $\mathcal{Y} = \{\omega_1, \omega_2, ...\omega_C\}$ - classification (in supervised learning) or clustering (in unsupervised learning).
    - C=2: binary classification, encoding - $\mathcal{Y} = \{+1, -1\}$ or $\mathcal{Y} = \{0, 1\}$.
    - C>2: multiclass classification
  - $\mathcal{Y}$-set of all sets of $\{\omega_1, \omega_2, ...\omega_C\}$ - labeling
    - $\mathcal{Y} = \{y \in \mathbb{R}^C : y_i \in \{0, 1\}\}$, $y_i = 1 \Leftrightarrow$object is associated with ω_i.

Table of Contents

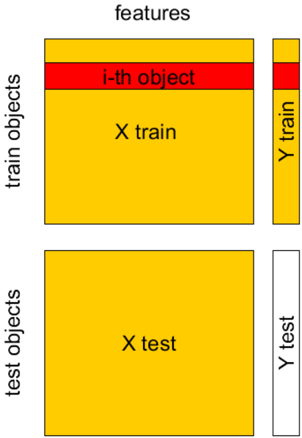
Training set

- **Training set:** $X \in \mathbb{R}^{N x D}$ - **design matrix**, $Y \in \mathbb{R}^{N}$- predicted outputs (target values)
- Using X, Y the task is to estimate unknown parameters $\widehat{\theta}$ of mapping $\widehat{y} = f_{\theta}(x)$ so that it will approximate true relationship $y = y(x)$
- It is assumed that $z_n = (x_n, y_n)$ for $n = 1, 2, ...N$ - are independent and identically distributed random variables (i.i.d).
- Two steps of ML:
    - **training**
    - **application**

Train set, test set

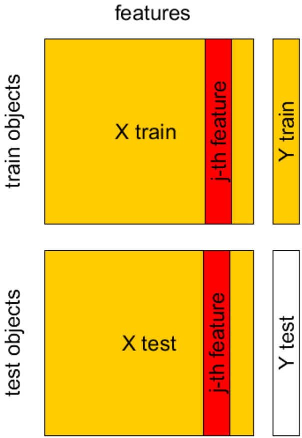


Train set, test set



N - number of objects for which targets (Y) are known.

Train set, test set



D - number of features (advanced case: variable feature count).

Table of Contents

Function class. Linear example.

- **Function class** - parametrized set of functions $F = \{f_\theta, \theta \in \Theta\}$, from which the true relationship $\mathcal{X} \to \mathcal{Y}$ is approximated.

Function class. Linear example.

- **Function class** - parametrized set of functions $F = \{f_\theta, \theta \in \Theta\}$, from which the true relationship $\mathcal{X} \to \mathcal{Y}$ is approximated.
- Examples of linear class functions:
    - regression:

$$f(x) = \theta_0 + \theta_1 x^1 + \theta_2 x^2 + ... + \theta_D x^D$$

Function class. Linear example.

- **Function class** - parametrized set of functions $F = \{f_\theta,\, \theta \in \Theta\}$, from which the true relationship $\mathcal{X} \to \mathcal{Y}$ is approximated.
- Examples of linear class functions:
  - regression:

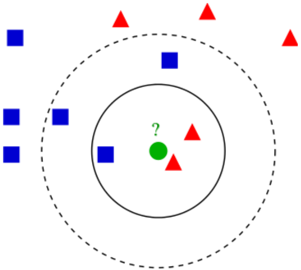
$$f(x) = \theta_0 + \theta_1 x^1 + \theta_2 x^2 + ... + \theta_D x^D$$

  - binary classification $y \in \{+1, -1\}$:

$$f(x) = \text{sign}\{\theta_0 + \theta_1 x^1 + \theta_2 x^2 + ... + \theta_D x^D\},$$

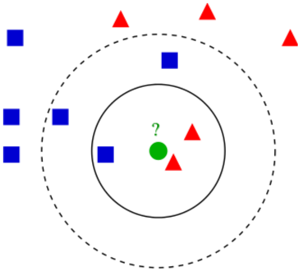
Function class. K-NN example.

Classification:

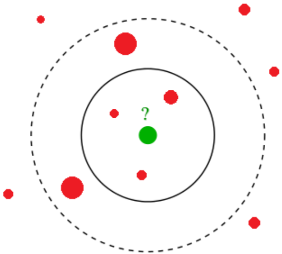


Function class. K-NN example.

Classification:



Regression:



Function class. K-NN example.

- denote for each x:
  - $i(x,k)$ - index of the k-th most close object to x
  - $I(x,K)$ - set of indexes of K nearest neighbours.
- regression:

$$f(x) = \frac{1}{K}\left(y_{i(x,1)} + ... + y_{i(x,K)}\right)$$

- classification:

$$f(x) = argmax\left\{\sum_{i \in I(x,K)} \mathbb{I}[y_i = 1], ... \sum_{i \in I(x,K)} \mathbb{I}[y_i = C],\right\}$$

Model complexity vs. data complexity

Underfitted model

Model that oversimplifies true relationship $\mathcal{X} \to \mathcal{Y}$.

Overfitted model

Model that is too tuned on particular peculiarities (noise) of the training set instead of the true relationship $\mathcal{X} \to \mathcal{Y}$.

Examples of overfitted/underfitted models

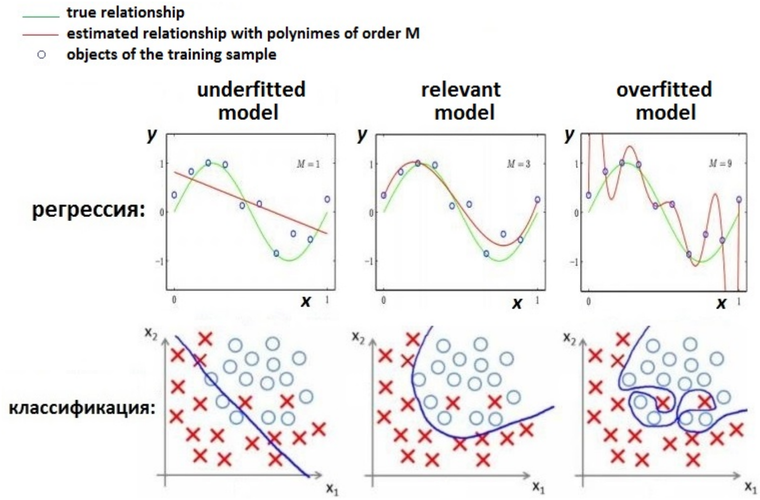


Table of Contents

Score versus loss

- In machine learning predictions, functions, objects can be assigned:
    - **score, rating** - this should be maximized
    - **loss, cost** - this should be minimized[2]

[2] how can one convert score↔loss?

Loss function $\mathcal{L}(\widehat{y}, y)$[3]

- Examples:
    - **classification:**
        - misclassification rate

$$\mathcal{L}(\widehat{y}, y) = \mathbb{I}[\widehat{y} \neq y]$$

    - **regression:**
        - MAE (mean absolute error):

$$\mathcal{L}(\widehat{y}, y) = |\widehat{y} - y|$$

        - MSE (mean squared error):

$$\mathcal{L}(\widehat{y}, y) = (\widehat{y} - y)^2$$

[3] Selecting loss is not trivial. Consider demand forecasting.

Empirical risk

- Want to minimize:

$$\int\int \mathcal{L}(f_\theta(x), y)p(x, y)dxdy \to \min_\theta$$

Empirical risk

- Want to minimize:

$$\int\int \mathcal{L}(f_\theta(x), y)p(x,y)dxdy \to \min_\theta$$

- Empirical risk:

$$L(\theta|X,Y) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{L}(f_\theta(x_n),\, y_n)$$

- Method of empirical risk minimization:

$$\widehat{\theta} = \arg\min_\theta L(\theta|X,Y)$$

Estimation of empirical risk

- Generally it holds that:

$$L(\widehat{\theta}|X, Y) < L(\widehat{\theta}|X', Y')$$

where X, Y is the training sample and X', Y' is the new data.

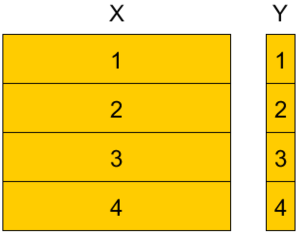
Estimation of empirical risk

- Generally it holds that:

$$L(\widehat{\theta}|X, Y) < L(\widehat{\theta}|X', Y')$$

  where X, Y is the training sample and X', Y' is the new data.
- $L(\widehat{\theta}|X', Y')$ can be estimated using :
  - separate **validation set**
  - **cross-validation**
  - **leave-one-out** method

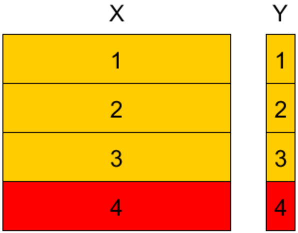
4-fold cross validation example



Divide training set into K parts, referred as «folds» (here $K = 4$).
Variants:

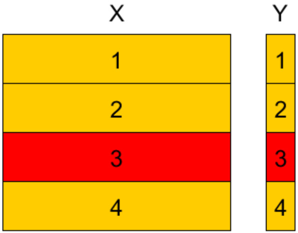
- randomly
- randomly with stratification (w.r.t target value or feature value).

4-fold cross validation example



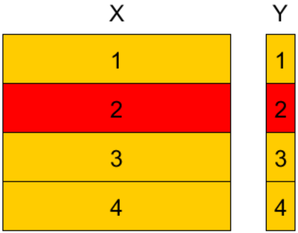
Use folds 1,2,3 for model estimation and fold 4 for model evaluation.

4-fold cross validation example



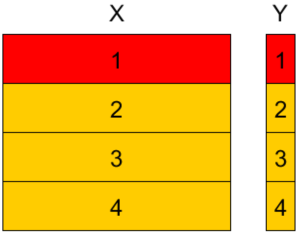
Use folds 1,2,4 for model estimation and fold 3 for model evaluation.

4-fold cross validation example



Use folds 1,3,4 for model estimation and fold 2 for model evaluation.

4-fold cross validation example



Use folds 2,3,4 for model estimation and fold 1 for model evaluation.

4-fold cross validation example

- Denote
    - $k(n)$ - fold to which observation (x_n, y_n) belongs to: $n \in I_k$.
    - $\widehat{\theta}^{-k}$ - parameter estimation using observations from all folds except fold k.

[4] will samples be correlated?

4-fold cross validation example

- Denote
  - $k(n)$ - fold to which observation (x_n, y_n) belongs to: $n \in I_k$.
  - $\widehat{\theta}^{-k}$ - parameter estimation using observations from all folds except fold k.

Cross-validation empirical risk estimation

$$\widehat{L}_{total} = \frac{1}{N}\sum_{n=1}^{N} \mathcal{L}(f_{\widehat{\theta}^{-k(n)}}(x_n),\ y_n)$$

[4] will samples be correlated?

4-fold cross validation example

- Denote
  - $k(n)$ - fold to which observation (x_n, y_n) belongs to: $n \in I_k$.
  - $\widehat{\theta}^{-k}$ - parameter estimation using observations from all folds except fold k.

Cross-validation empirical risk estimation

$$\widehat{L}_{total} = \frac{1}{N}\sum_{n=1}^{N} \mathcal{L}(f_{\widehat{\theta}^{-k(n)}}(x_n),\ y_n)$$

- For K-fold CV we have:
  - K parameters $\widehat{\theta}^{-1}, ...\widehat{\theta}^{-K}$
  - K models $f_{\widehat{\theta}^{-1}}(x), ...f_{\widehat{\theta}^{-K}}(x)$.
    - can use ensembles
  - K estimations of empirical risk: $\widehat{L}_k = \frac{1}{|I_k|}\sum_{n\in I_k} \mathcal{L}(f_{\widehat{\theta}^{-k}}(x_n),\ y_n),\ k = 1, 2, ...K.$
    - can estimate variance & use statistics![4]

[4] will samples be correlated?

Comments on cross-validation

- When number of folds K is equal to number of objects N, this is called **leave-one-out method**.
- Cross-validation uses the i.i.d.[5] property of observations
- Stratification by target helps for imbalanced/rare classes.

[5] i.i.d.=independent and identically distributed

Cross-validation vs. A/B testing

- A/B testing:
  1. divide objects randomly into two groups - A and B.
  2. apply model 1 to A
  3. apply model 2 to B
  4. compare final results

[6] may final business quality be high when forecasting quality is low?

Cross-validation vs. A/B testing

- A/B testing:
  1. divide objects randomly into two groups - A and B.
  2. apply model 1 to A
  3. apply model 2 to B
  4. compare final results

Comparison of cross-validation and A/B test:

cross-validation	A/B test
evaluates forecasting quality	evaluates final business quality[6] (may evaluate forecasting quality as well)
uses available data, only computational costs	requires time and resources for collecting & evaluating feedback from objects of groups A and B

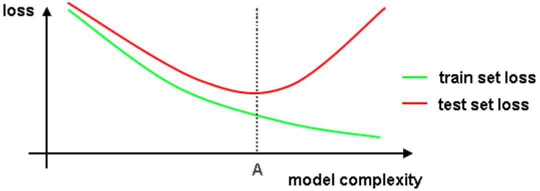
[6] may final business quality be high when forecasting quality is low?

Hyperparameters selection

- Using CV we can select hyperparameters of the model[7]:
  - regression: # of features d, e.g. $x, x^2, ...x^d$
  - K-NN: number of neighbors K

[7] can we use CV loss in this case as estimation for future losses?

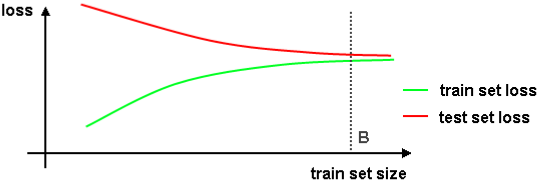
Loss vs. model complexity



Comments:

- expected loss on test set is always higher than on train set.
- left to A: model too simple, underfitting, high bias
- right to A: model too complex, overfitting, high variance

Loss vs. train set size



Comments:

- expected loss on test set is always higher than on train set.
- right to B there is no need to further increase training set size
  - useful to limit training set size when model fitting is time consuming

Cost matrix[8]

For classification in case we output final class predictions $\widehat{y}$ (not probabilities) $\mathcal{L}(y, \widehat{y})$ becomes a matrix:

predicted classes

true classes		$\widehat{y}=1$	$\cdots$	$\widehat{y}=C$
	$y=1$	λ_{11}	$\cdots$	λ_{1C}
	$\cdots$	$\cdots$	$\cdots$	$\cdots$
	$y=C$	λ_{C1}	$\cdots$	λ_{CC}

[8] propose some sample cost matrix for binary classification predicting illness

Table of Contents

Discriminant functions[9]

- Discriminant functions is the most general way to describe each classifier.
- Each classifier implies a particular set of discriminant functions.

Discriminant functions

- a set of C functions $g_y(x)$, $y = 1, 2...C$.
- $g_y(x)$ measures the score of class y, given object x.

Usage

Assign x to class having maximum discriminant function value:

$$\widehat{c} = \arg\max_{c} g_c(x)$$

[9] For fixed classifier are they uniquely defined?

Examples[10]

- K-NN:

$$g_f(x) = \sum_{k=1}^{K} \mathbb{I}[y_{i(k)} = f]$$

- Linear classifier:

$$g_f(x) = \langle w_f, x \rangle$$

- Nearest centroid:

$$g_f(x) = \rho(x, \mu_f)$$

- Maximum posterior probability classifier:

$$g_f(x) = p(y = f|x)$$

[10] Provide discriminant functions for classifier minimizing expected cost according to given cost matrix.

Binary classification

- For two class case $y \in \{-1, +1\}$ we may define a single discriminant function $g(x) = g_1(x) - g_2(x)$ such that

$$\widehat{y}(x) = \begin{cases} +1, & g(x) \geq 0, \\ -1 & g(x) < 0. \end{cases}$$

- Compact notation: $\widehat{y}(x) = sign[g(x)]$
- Boundary between classes:

Binary classification

- For two class case $y \in \{-1, +1\}$ we may define a single discriminant function $g(x) = g_1(x) - g_2(x)$ such that

$$\widehat{y}(x) = \begin{cases} +1, & g(x) \geq 0, \\ -1 & g(x) < 0. \end{cases}$$

- Compact notation: $\widehat{y}(x) = sign[g(x)]$
- Boundary between classes: $\{x : g(x) = 0\}$.
- Linear classifier:
    - $g(x) = \langle w_{+1}, y \rangle - \langle w_{-1}, y \rangle = \langle w, y \rangle$

Binary classification: probability calibration

- $g(x)$ - score of positive class, $p(y=+1|x)$[11]-?

[11] does this apply to K-NN? How to smooth probabilities of K-NN for small K?

Binary classification: probability calibration

- $g(x)$ - score of positive class, $p(y=+1|x)$[11]-?
- Platt scaling: $p(y=+1|x)=\sigma\left(\theta_0+\theta_1 g(x)\right)$,
    - $\sigma(u)=\frac{1}{1+e^{-u}}$

[11] does this apply to K-NN? How to smooth probabilities of K-NN for small K?

Binary classification: probability calibration[12]

- Using the property $1-\sigma(z)=\sigma(-z)$:

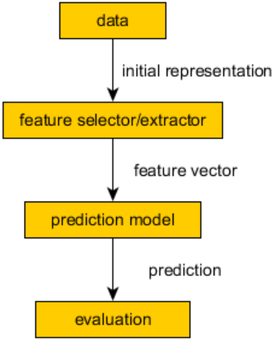
$$p(y=1|x)=\sigma\left(\theta_0+\theta_1 g(x)\right)$$
$$p(y=-1|x)=1-\sigma(\theta_0+\theta_1 g(x))=\sigma\left(-\theta_0-\theta_1 g(x)\right)$$

- Thus $p(y|x)=\sigma\left(y(\theta_0+\theta_1 g(x))\right)$
- Estimate θ_0,θ_1 using maximum likelihood:

$$\prod_{n=1}^{N}\sigma\left(y_n(\theta_0+\theta_1 g(x_n))\right)\to\max_{\theta_0,\theta_1}$$

[12] extend this reasoning to multiclass case

General modelling pipeline



If evaluation gives poor results we may return to each of preceding stages.

Connection of ML with other fields

- Pattern recognition
  - recognize patterns and regularities in the data
- Computer science
- Artificial intelligence
  - create devices capable of intelligent behavior
- Time-series analysis
- Theory of probability, statistics
  - when relies upon probabilistic models
- Optimization methods
- Theory of algorithms

Notation used in the course

- If this corresponds the context and there are no redefinitions, then:
  - x - vector of known input characteristics of an object
  - y - predicted target characteristics of an object specified by x
  - x_i - i-th object of a set, y_i - corresponding target characteristic
  - x^k - k-th feature of object specified by x
  - x_i^k - k-th feature of object specified by x_i
  - D - dimensionality of the feature space: $x \in \mathbb{R}^D$
  - N - the number of objects in the training set
  - X - design matrix, $X \in \mathbb{R}^{NxD}$
  - $Y \in \mathbb{R}^N$ - target characteristics of a training set
  - $\mathcal{L}(\widehat{y}, y)$ - loss function, where y is the true value and $\widehat{y}$ is the predicted value.
  - $\{\omega_1, \omega_2, ...\omega_C\}$ - possible classes, C - total number of classes.
  - $\widehat{z}$ defines an estimate of z, based on the training set: for example, $\widehat{\theta}$ is the estimate of θ, $\widehat{y}$ is the estimate of y, etc.