Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Universität Zürich[UZH]

Machine Learning, Fall Semester 2014

The Cherry-Picking Summary of the lectures in 2014

Diana Ponce-Morado
Joachim Ott
Imanol Studer
Benjamin Ellenberger

Monday 26th January, 2015

svn repository: https://code.google.com/p/eth-machine-learning-summary/
Contact be.ellenberger@gmail.com for further information.

Professor:
Prof. Dr. Joachim Maximilian Buhmann
Institute of Neuroinformatics, UZH

As in Cherry-picking, a term taken from revision control, where it denotes moving only some revisions from one branch to another, we cherry-pick the most understandable explanations and definitions into one summary to summarize the content of the lecture about Machine Learning of Prof. Joachim Buhmann. Of course we tell you the source of the explanation even if we do not properly cite or properly reference, which separates us from a gang of kanging academics.

—

Contents

1. Preliminaries

2. A Refresher on Probabilities

3. Convergence of RVs

Literature

- **Christopher M. Bishop, Pattern Recognition and Machine Learning. Springer Verlag (2006)**
- Richard O. Duda, Peter E. Hart & David G. Stork, *Pattern Classification.* Wiley & Sons (2001)
- Trevor Hastie, Robert Tibshirani & Jerome Friedman, *The Elements of Statistical Learning: Data Mining, Inference and Prediction.* Springer Verlag (2001)
- Luc Devroye, Laslo Györfi & Gabor Lugosi, *A Probabilistic Theory of Pattern Recognition.* Springer Verlag (1996)
- Vladimir N. Vapnik, *Estimation of Dependences Based on Empirical Data.* Springer Verlag (1983)
- Ulf Grenander, General Pattern Theory: A Mathematical Study of Regular Structures. Oxford University Press (1993)
- Andrew Webb, *Statistical Pattern Recognition.* Wiley & Sons, (2002)
- Keinosuke Fukunaga, *Statistical Pattern Recognition.* Academic Press (1990)
- Brian D. Ripley, *Pattern Recognition and Neural Networks.* Cambridge University Press (1996)
- Larry Wasserman, *All of Statistics.* (1st ed. 2004. Corr. 2nd printing, ISBN: 0-387-40272-1) Springer Verlag (2004)

1. Preliminaries

2. A Refresher on Probabilities

3. Convergence of RVs

Probability

- Fraction of times an event occurs.
- Degree of belief about an event.
- Useful as data model, incorporate noise, uncertainty, ...

Random Variables

- A random variable is a "probabilistic" outcome of an experiment, such as a coin flip or the height of a person chosen from a population.
- Notation:

  X Random variable
  $\approx$ a device from which we draw a value.

  x If x is not capital, it denotes a realization of X.
  Pr$\{X = x\}$ denotes the probability for this to occur.

  $\mathcal{X}$ Sample space or domain of X.
  The set of all values a draw from X may result in.

Random Variables

- RVs take on values in a sample space $\mathcal{X}$. This space may be discrete or continuous, and the space may be defined differently for different scenarios.

Types of sample spaces:

1. Discrete sets:
   - Finite: for a coin flip $\mathcal{X} = \{H, T\}$
   - Infinite: $\mathcal{X} = \mathbb{N}, \mathbb{Z}$ etc.
2. Continuous sets: e.g. $\mathcal{X} = \mathbb{R}, \mathbb{R}_+, \mathbb{R}^d, [0,1], [a,b]$

- There is not necessarily one uniquely "correct" sample space for a particular concept.

Probability of Random Variables

Probability distribution function describes how probabilities are distributed over the values of the random variable.

$$\text{Probability(event)} = \frac{\text{The total number of events of interest}}{\text{The total number of events}}$$

Probability of Random Variables

- A discrete distribution assigns a probability to every atom in the sample space of a random variable.
- For example, if X is an (unfair) coin, then the sample space consists of the atomic events $X = H$ and $X = T$, and the discrete distribution might look like:
  Pr$\{X = H\} = 0.7$
  Pr$\{X = T\} = 0.3$
- For any valid discrete distribution, the probabilities over the atomic events must fulfill:
  1. Non-negativity: Pr$\{x\} \geq 0$
  2. Normalization: $\sum_{x \in \mathcal{X}} \Pr\{X = x\} = 1$

Probability of Random Variables

- An event is a subset of atoms (one or more). The probability of an event is the sum of the probabilities of its constituent atoms.
- **Example:** Consider the event of a single die roll (D) is bigger than 3

  The probability of $D > 3$ is equivalent to the probability that the outcome is 4, or the outcome is 5, or the outcome is 6. The probabilities that the die is 4, 5, or 6 are added together:

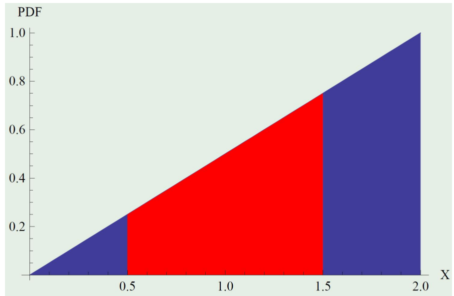
$$\Pr\{D > 3\} = \Pr\{D = 4\} + \Pr\{D = 5\} + \Pr\{D = 6\}$$

Continuous Random Variables

- A continuous random variable can assume any value in an interval or in a collection of intervals.

$$\Pr\{a \le X \le b\} = \int_a^b p(x)dx$$

Example: Find the probability that $0.5 \le X \le 1.5$



Continuous Random Variables

- For continuous probability distributions, we require:
  1. Non-negativity: $p(x) \ge 0$
  2. Normalization: $\int_{\mathcal{X}} p(x)dx = 1$
- **Notation:** We deal with three types of symbols:

$\Pr\{\dots\}$	Probability of an event (inside the curly brackets), such as $\Pr\{X = x\}$.
$P(x)$	Probability mass function.
$p(x)$	Probability density function.

- Density functions are only applicable in the case of continuous sample spaces.

Joint Probabilities

Typically, one considers collections of RVs.
For example, the flipping of 4 coins involves 4 RVs, 1 for each coin.

Joint probability:	The probability for precisely the values x, y to occur together.
Definition:	$P(x, y) := \Pr\{X = x, Y = y\}$

The joint distribution for a flip of each of 4 coins assigns a probability to every outcome in the space of all possible outcomes of the 4 flips.

If all coins are fair:

$$\begin{aligned} P(HHHH) &= 0.0625 \\ P(HHHT) &= 0.0625 \\ P(HHTH) &= 0.0625 \\ &\dots \end{aligned}$$

Conditional Probability

A conditional distribution is the distribution of some random variable given some evidence, such as the value of another random variable.

- **Def.**: Pr$\{X = x|Y = y\}$ is the probability that $X = x$ when $Y = y$.

A conditional distribution gives more information about X than the distribution of $P(X)$ alone.

Conditional Probability

The conditional distribution Pr$\{X = x|Y = y\}$ is a different distribution for each value of y. Such that

$$\sum_x \Pr\{X = x|Y = y\} = 1$$

However, remember that, generally

$$\sum_y \Pr\{X = x|Y = y\} \neq 1.$$

Marginalization

- Given a collection of random variables, we are often interested in only a subset of them. For example, we might want to compute $P(X)$ from a joint distribution $P(X, Y, Z)$.

Def.

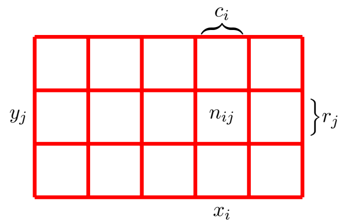
Marginal probability:	The probability for x to occur, regardless of y.
Discrete case:	$P(x) := \sum_{y \in \mathcal{Y}} P(x, y)$
Continuous case:	$p(x) := \int_{\mathcal{Y}} p(x, y) dy$

Marginalization

This property actually derives from the chain rule:

$$\begin{aligned}
\sum_{y \in \mathcal{Y}} P(x, y) &= \textstyle\sum_{y \in \mathcal{Y}} P(x) P(y|x) && \text{by the chain rule} \\
&= P(x) \textstyle\sum_{y \in \mathcal{Y}} P(y|x) && P(x) \text{ doesn't depend on y} \\
&= P(x) && \textstyle\sum_{y \in \mathcal{Y}} P(y|x) = 1
\end{aligned}$$

Conditional, Joint, Marginal



Joint Probability

The entry of both values jointly.

$$\Pr\{X = x_i, Y = y_j\} = \frac{n_{ij}}{N}$$

Marginal Probability

The sum over a row or column.

$$\Pr\{X = x_i\} = \frac{c_i}{N}$$

Conditional Probability

The fraction of a row or column in a particular cell.

$$\Pr\{Y = y_j \mid X = x_i\} = \frac{n_{ij}}{c_i}$$

Simpson's Paradox

- illustrates the difference between marginal and conditional distributions
- men and women under treatment $\longrightarrow$ recovery?
- marginal of frequencies:

		Recovery 0	Recovery 1
Ttreatment	0	180	180
	1	200	200

obviously:
R indep. of T

- conditional tables of frequencies (given the gender):

Men:

		Recovery 0	Recovery 1
Ttreatment	0	120	160
	1	50	100

Women:

		Recovery 0	Recovery 1
Ttreatment	0	60	20
	1	150	100

obviously:
R dep. on T given M/F

The other way around

- men and women under treatment $\longrightarrow$ recovery?
- marginal table of frequencies:

		Recovery 0	Recovery 1
Ttreatment	0	50	40
	1	90	120

obviously:
R dep. on T

- conditional tables of frequencies (given the gender):

Men:

		Recovery 0	Recovery 1
Ttreatment	0	40	20
	1	40	20

Women:

		Recovery 0	Recovery 1
Ttreatment	0	10	20
	1	50	100

obviously:
R indep. of T given M/F

Conditional Probability and Related Concepts

Conditional probability can be defined in terms of the joint and single probability distributions:

$$P(X \mid Y) = \frac{P(X, Y)}{P(Y)}$$

(provided $P(Y) > 0$)

The Chain Rule

The definition of conditional probability leads to the chain rule, which lets us define the joint distribution of two (or more) random variables as a product of conditionals:

The Chain Rule:

$$
\begin{aligned}
P(X,Y) &= \frac{P(X,Y)P(Y)}{P(Y)} \\
&= P(X|Y)P(Y)
\end{aligned}
$$

- The chain rule can be used to derive the $P(X,Y)$ when it is not known.
- The chain rule can be extended to any set of n variables.

Bayes Rule

By the chain rule:

$$
\begin{aligned}
P(X,Y) &= P(X|Y)P(Y) \\
&= P(Y|X)P(X)
\end{aligned}
$$

This is equivalently expressed as Bayes rule:

$$
P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}
$$

posterior $\propto$ likelihood $\times$ prior

Independence

- Random variables are independent if knowing about X tells us nothing about Y. That is,

$$
P(Y|X) = P(Y)
$$

.

- This means that their joint distribution factorizes:

$$
P(X,Y) = P(X)P(Y)
$$

- This factorization is possible because of the chain rule:

$$
\begin{aligned}
P(X,Y) &= P(X)P(Y|X) \\
&= P(X)P(Y)
\end{aligned}
$$

i.i.d.

- i.i.d. $=$ independently, identically distributed
- RVs $X_1, ..., X_n$ are i.i.d. iff
  1. They are mutually statistically independent.
  2. All drawn according to the same distribution.
- Note: If $X_1, ..., X_n$ are i.i.d., then

$$
\begin{aligned}
p(x_1, ..., x_n) &= p_{X_1}(x_1)...p_{X_n}(x_n) \\
&= \prod_{i=1}^{n} p(x_i)
\end{aligned}
$$

Expectation

- Definition:
$$\mu_x := \mathbb{E}[X] := \int_{\mathcal{X}} xp(x)dx$$
The integral is called the first moment of p.
- Note: Expected value $\neq$ Most likely value.
- For a function f:
$$\mathbb{E}[f(X)] := \int_{\mathcal{X}} f(x)p(x)dx$$

Variance

- Definition:
$$\sigma_X^2 := \mathrm{Var}[X] := \int_{\mathcal{X}} (x-\mu_X)^2 p(x)dx$$
$\rightarrow$ second centralized moment of p.
- Always: $\mathrm{Var}[X] \geq 0$
- Definition: The square root $\sigma_X = \sqrt{\mathrm{Var}[X]}$ is called the standard deviation of X.

Multiple Dimensions

- A vector of random variables
$$\mathbf{X} = (X_1, ..., X_n)^\top$$
A draw $\mathbf{x} = (x_1 \dots x_n)^\top$ from $\mathbf{X}$ defines a point in n-dimensional space.
- It is treated just like a list of $1D$ RV's.
- The vector components are not necessarily i.i.d
- We can add RV's to produce a new RV
$$Y := c_1 X_1 + c_2 X_2$$

Multidimensional Moment Statistics

- Expectation: Vector of components expectation
$$\mathrm{E}[\mathbf{X}] := (\mathrm{E}[X_1], ..., \mathrm{E}[X_n])^\top$$
- Variance: Generalized to covariance:
$$\begin{array}{rcl} Cov[X,Y] & := & \int_{\mathcal{X}} \int_{\mathcal{Y}} p(x,y)(x-\mu_X)(y-\mu_Y)dxdy \\ & = & \mathrm{E}_{X,Y}[(x-\mu_X)(y-\mu_Y)] \end{array}$$
- If two RVs have non-zero covariance, we call them correlated
- The covariance is a linear measure statistical dependence

Covariance Behavior

- If X,Y are independent, then $Cov[x, y] = 0$
- Proportional behavior:

$$\begin{aligned} Cov[X, Y] > 0 \quad &\Leftrightarrow \quad X, Y \text{increase together} \\ Cov[X, Y] < 0 \quad &\Leftrightarrow \quad X, Y \text{are anti-proportional} \end{aligned}$$

Covariance Matrix

- For RVs $X_1, ..., X_n$ we use a covariance matrix Σ to describe their mutual covariances:

$$\Sigma_{i,j} := Cov[X_i, X_j] \qquad i, j = 1, ..n$$

- The covariance matrix Σ generalizes the notion of variance to sets of RVs or multiple dimensions.

Covariance Matrix Properties

1. Diagonal entries are RVs variances:

$$\Sigma_{i,j} := Cov[X_i, X_i] = Var[X_i]$$

2. Σ is symmetric:

$$\Sigma_{i,j} = Cov[X_i, X_j] = Cov[X_j, X_i] = \Sigma_{j,i}$$

3. Σ is positive semi-definite

Question: What does a diagonal covariance matrix, Σ mean?

Gaussian Distribution (1D)

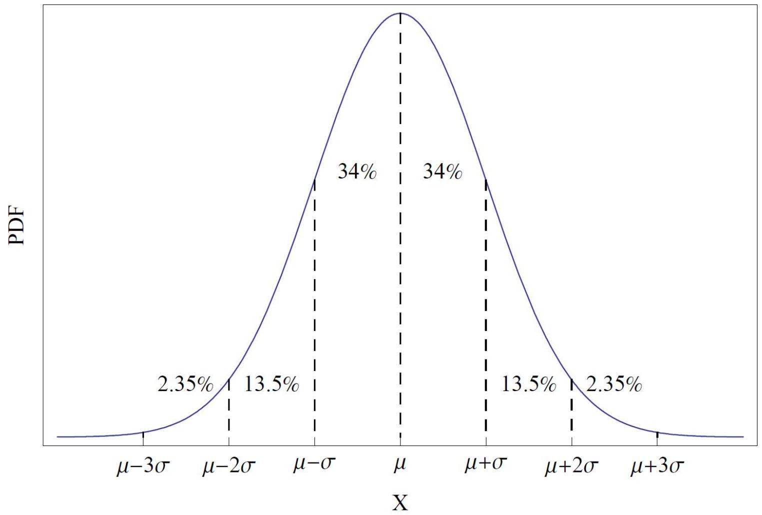
- Sample space $\mathcal{X} = \mathbb{R}$
- Definition: $p(X|\mu, \sigma) := \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(X-\mu)^2}{2\sigma^2})$
- Statistics:

$$\mathrm{E}[X] := \mu, Var[X] := \sigma^2$$

Technically speaking, the Gaussian distribution specifies that the probability density associated with a point x is proportional to the negative exponentiated half-distance to μ scaled by σ^2..

Gaussian Distribution (1D)

Here is a more compelling explanation..



Gaussian Distribution (nD)

- Sample space $\mathcal{X} = \mathbb{R}^n, \mathbf{x} = (x_1, .., x_n)^\top$
- Definition:
  $p(\mathbf{x}|\mu, \Sigma) := \frac{1}{(\sqrt{2\pi})^n |\Sigma|^{\frac{1}{2}}} \exp(-\frac{1}{2}(\mathbf{x} - \mu)^\top \Sigma^{-1}(\mathbf{x} - \mu))$

  where Σ is the covariance matrix and $|\Sigma|$ is its determinant

Gaussian Distribution

- Using only correlation/covariance to describe independence means: Higher-order dependencies are neglected.
- This is what the Gaussian does: Parametrized only by location and covariance.
- Describing dependencies in data by covariance is equivalent to approximation of data distribution by a Gaussian model.

Data vs. Distribution

- Important: Be careful to distinguish between distributions (smooth functions in most examples) and data (point clouds).
- Machine learning:
  - Data = input
  - Distribution = model or assumption
- ML methods usually make some general assumptions about distribution, then try to obtain ("infer") the specifics from the data.

Example 1) Modeling step: Assume a Gaussian as model.
2) Inference step: Estimate Gaussian parameters (μ and σ) from data.

1. Preliminaries

2. A Refresher on Probabilities

3. Convergence of RVs

Empirical distribution

- We try to regard data sample (imagine some point cloud) as a distribution.
- Problem: We only know whether or not a point is there, not how probable that is.
- Simple solution: Assign same probability to each point.

  Def. Let $S = \{x_1, .., x_n\}$ be a sample of the data, we call

  $$P(x) := \frac{1}{n} \cdot \#\{y \in S | y = x\}$$

  the empirical distribution defined by the data.

Large Sample Theory

Basic question: What can we say about the limiting behavior of a sequence of RVs $X_1, X_2, X_3, \ldots$?
In calculus:

- A sequence of real numbers x_n converges to a limit x if, for every $\epsilon > 0, |x_n - x| < \epsilon$ for all large n
- Trivial example: Suppose $x_n = x$ for all n, then trivially $\lim_{n \to \infty} x_n = x$

In probability theory: for continuous distribution a $\Pr\{X = x_0\} = 0$ thus it's difficult to speak of limits in the same sense as in calculus.

Types of Convergence

Let $X_1, X_2, X_3, \ldots$ be a sequence of RVs and X another RV. Let F_n denote the CDF of X_n and F the CDF of X.

1. X_n converges to X in probability, written $X_n \xrightarrow{P} X$, if for every $\epsilon > 0$
   $$\Pr\{|X_n - X| > \epsilon\} \to 0$$
   as $n \to \infty$.
2. X_n converges to X in distribution, written $X_n \rightsquigarrow X$, if
   $$\lim_{n \to \infty} F_n(t) = F(t)$$
   at all t for which F is continuous.

Law of Large Numbers (weak statement)

The weak law of large numbers states that the sample average

$$\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

converges in probability to the expectation

$$\mu = \mathbb{E}(X_i)$$

Question: what conditions are forgotten here for the statement to hold?

Relationships and Transformations

It holds that convergence in

probability $\Rightarrow$ distribution

Some convergence properties are preserved under transformations. Examples:

- If $X_n \xrightarrow{P} X$ and $Y_n \xrightarrow{P} Y$, then $X_n + Y_n \xrightarrow{P} X + Y$.
- The same is not true for convergence in distribution.
- If $X_n \xrightarrow{P} X$, then $g(X_n) \xrightarrow{P} g(X)$.
- If $X_n \rightsquigarrow X$, then $g(X_n) \rightsquigarrow g(X)$.

Note: Expected Error vs Train Error

Recall the lecture. Training error

$$\hat{R}_D(\mathrm{w}) = \frac{1}{|D|}\sum_{(\mathrm{x},y)\in D} (y - \langle \mathrm{w}, \mathrm{x}\rangle)^2.$$

Note: training error is itself a random variable!
It is exactly the weighted sum from the formulation of the Law of Large Numbers! Its justifies, that when the training set grows, then

$$\hat{R}_D(\mathrm{w}) \xrightarrow{P} R(\mathrm{w}), \quad \text{(weak LLN; in lecture — strong LLN)}.$$

Be careful: in most cases the expected value of the training error underestimates the expected error of the training set:

$$\mathbb{E}\,\hat{R}_D^{\text{train}}(\mathrm{w}) \le \mathbb{E}\,\hat{R}_D^{\text{test}}(\mathrm{w}).$$

Reason: training set is used for training!
Can be proven rigorously in case of regression (see in later tutorial).

2 Representations

2.1 Measurements and Data

How to represent data is a crucial part to successful learning. The concrete choice of representation ('features') Learning methods expect standardized representations of data (e.g. Points in vector spaces, nodes in a graph, similarity matrices ...)

Taxonomy of data

Goal: We like to represent objects/items of interest and characterize them according to their typical patterns for detection, classification, abstraction (compression), ...

- Object space: Design/Configuration/ Object space $\mathcal{O}$
- Measurement/Feature: Measurement X maps an object into a domain $X : \mathcal{O}^{(1)} \dots \mathcal{O}^{(R)} \rightarrow \mathbb{K}$ and represents an object in a data space.
- Data
  - **Monadic data** $\mathcal{O} \rightarrow \mathbb{R}^d$: An indivisible, impenetrable unit of data such as temperature, water depth, pressure, intensity etc..
  - **Dyadic data** $\mathcal{O}^{(1)} \times \mathcal{O}^{(2)} \rightarrow \mathbb{R}$: Data consisting of two monadic datasets of different object spaces. This can be $\{users\} \times \{websites\}$ etc..
  - **Polyadic data** $\mathcal{O}^{(1)} \times \mathcal{O}^{(2)} \times \dots \times \mathcal{O}^{(R)} \rightarrow \mathbb{R}$: Data consisting of multiple monadic datasets of different object spaces.
  - **Pairwise data** $\mathcal{O} \times \mathcal{O} \rightarrow \mathbb{R}$: Data that comes in pairs of the same object space such as $\{proteins\} \times \{proteins\}$ comparisons.

The choice of features X is an engineering problem. We must find a tradeoff between underfitting and overfitting the data.

1. Ideally as informative as possible about label Y
2. Must consider cost of acquiring / computing them
3. Poor choice of features $\rightarrow$ no luck with learning!

2.2 Data Types, Scale, Transformation invariances

Data Types and Scale

- **Nominal or categorical scale**: Qualitative, but without quantitative measurements, e.g. binary scale $\mathbb{X} = \{0, 1\}$ (presence or absence of properties). Ordering does not matter.

- **Ordinal scale**: Measurement values are meaningful only with respect to other measurements, i.e., the rank order of measurements carries the information, not the numerical differences *(e.g. information on the ranking of different marathon races)*
- **Quantitative scales**
  - **Interval scale**: The relation of numerical differences carries the information. Invariance w.r.t. translation and scaling *(Fahrenheit scale of temperature)*.
  - **Ratio scale**: Zero value of the scale carries information but not the measurement unit. *(Kelvin scale)*.
  - **Absolute scale**: Absolute values are meaningful. *(grades of final exams)*

Transformation invariances

Importance of invariances: if the measurements are invariant under a set of transformation then the mathematical definition of structure should obey the same invariances. Otherwise, our structure search procedure breaks the symmetry in an a priori (not data-dependent) way.

scale type	transformation invariances
nominal	$T = \{f : \mathbb{R} \to \mathbb{R} \mid \text{f bijective}\}$
ordinal	$T = \{f : \mathbb{R} \to \mathbb{R} \mid f(x_1) < f(x_2), \forall x_1 < x_2\}$
interval	$T = \{f : \mathbb{R} \to \mathbb{R} \mid f(x) = ax + c, a \in \mathbb{R}^+, c \in \mathbb{R}\}$
ratio	$T = \{f : \mathbb{R} \to \mathbb{R} \mid f(x) = ax, a \in \mathbb{R}^+\}$
absolute	$T = \{f : \mathbb{R} \to \mathbb{R} \mid \text{f is identity map}\}$

Table 1: Formal characterisation of different scale types and their invariance properties.

2.3 Readings

1. Duda 2nd ed.

Write down readings from the books covering the topics of the representations section.

3 Regression

In statistics, regression analysis is a statistical process for estimating the relationships among variables (one independent variable Y and one or more independent variable $(X_1, \dots, X_d)$). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed. A model of the relationship is hypothesized, and estimates of the parameter values are used to develop an estimated regression equation. Various tests are then employed to determine if the model is satisfactory.If the model is deemed satisfactory, the estimated regression equation can be used to predict the value of the dependent variable given values for the independent variables. (Source: Wikipedia and Encyclopedia Britannica on Regression)

- Object space $\mathcal{O}$
- Measurement/Feature space $\mathcal{F} = \mathbb{R}^d \times \mathbb{R}$
- Data $\mathcal{Z} = \{(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R} : 1 \geq i \geq n\}$

The problem of regression: What is the optimal estimate of a function $f : \mathbb{R}^d \to \mathbb{R}$ based on noisy data $y_i = f(x_i) + \varepsilon_i$?
Solution: The regression function: $y(x) = \mathbb{E}\{y|X = x\} = \int_\Omega yp(y|X = x)dy$
How many data are required to estimate a model or a regression function given a hypothesis class (set of possible regression functions)?

3.1 Linear Regression

Linear regression comes in a closed form solution and delivers an exact solution to the problem.

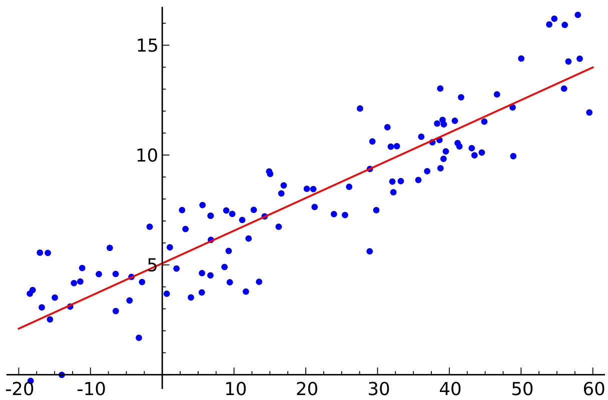


Figure 1: Linear regression of data

Given a vector of inputs $X^T = (X_1, \ldots, X_d)$. The output variable (also called response variable) is predicted via the model

$$Y = \beta_0 + \sum_{j=1}^{d} X_j \beta_j, \quad Y \in \mathbb{R}$$

β_0 is called bias(Machine Learning) or intercept (Statistics). We extend the X vector with an $X_0 = 1$ and we can include the bias β_0 into the vector of βs and can write Y in matrix form: $Y = X^T\beta, \;\; X, \beta \in \mathbb{R}^{d+1}$

Training of the Linear Regressor: Residual Sum of Squares(RSS)

Training of an accurate Predictor = Minimize the residual sum of squares between the actual value y and the predicted $\overline{X}\beta$:

$$RSS(\beta) = \sum_{i=1}^{n} (y_i - x_i^T \beta)^2$$

or matrix notation

$$RSS(\beta) = (y - \overline{X}\beta)^T(y - \overline{X}\beta), \quad X \in \mathbb{R}^n \times \mathbb{R}^{d+1}$$

(Each row of $\overline{X}$ is an input vector X, y is an n-vector of the outputs in the training set)

We find the minimum RSS by differentiating w.r.t. β

$$\begin{aligned}
&\rightarrow X^T(y - X\beta) = 0 \\
&\rightarrow \hat{\beta} = (X^T X))^{-1} X^T y \ \ \text{(Solution for nonsingular} X^T X)
\end{aligned}$$

Prediction via the Trained Linear Regressor

$$\hat{y} = X\hat{\beta} = X(X^T X)^{-1} X^T y$$

The matrix $X(X^T X)^{-1} X^T$ is sometimes called the hat matrix which is an orthogonal projection on the space spanned by the columns of X.

Perturbation of Linear Regressor with Gaussian Noise

Gaussian noise ε with $\mathcal{E}(\varepsilon) = 0, \mathcal{V} = \sigma^2$

Proof

$$\begin{aligned}
Y &= \mathbb{E}(Y|X_1, \dots, X_d) + \varepsilon \\
&= \beta_0 + \sum_{j=1}^{d} X_j \beta_j + \varepsilon \\
&= X\beta + \varepsilon
\end{aligned}$$

Meaning that gaussian noise does not perturb the estimator. $\square$

Optimality of Least Squares Estimate

The least squares estimate of the parameter β has the smallest variance among all linear unbiased estimates.

We want to predict a. Prediction $\hat{\theta}$ is a linear function $c_0^T y = \overbrace{a^T (X^T X)^{-1} X^T}^{c_0} y$ of the response vector y.

Proof $a^T \hat{\beta}$ is unbiased since the expectation $\mathbb{E}$ yields

$$\begin{aligned}
\mathbb{E}(a^T \hat{\beta}) &= \mathbb{E}(a^T (X^T X)^{-1} X^T y) \\
&= a^T (X^T X)^{-1} X^T \mathbb{E}(X\beta + \varepsilon) \\
&= a^T (X^T X)^{-1} X^T (X\beta + \underbrace{\mathbb{E}(\varepsilon)}_{=0}) = a^T \beta
\end{aligned}$$

Variance of $a^T\hat{\beta}$

$$\mathbb{V}(a^T\hat{\beta}) = \mathbb{V}\Big(a^T \underbrace{(X^TX)^{-1}X^T \overbrace{(X\beta+\varepsilon)}^{y}}_{\hat{\beta}}\Big)$$
$$= \underbrace{\mathbb{V}(a^T\beta)}_{=0} + \mathbb{V}\left(a^T(X^TX)^{-1}X^T\varepsilon\varepsilon^TX(X^TX)^{-1}a\right) = \sigma^2 a(X^TX)^{-1}a$$

The cross-correlations are linear in ε and, therefore, they vanish.

Alternative unbiased linear estimator (to show the least squares estimator is the best)

$$\hat{\theta} = c^Ty = a^T\hat{\beta} + a^TDy$$
$$\mathbb{E}(c^Ty) = \mathbb{E}(a^T\hat{\beta}) + \mathbb{E}(a^TDy) = a^T\beta + \mathbb{E}a^TD(X\beta+\varepsilon)$$
$$= a^T\beta + a^TDX\beta + a^TD\underbrace{\mathbb{E}(\varepsilon)}_{=0} = a^T\beta$$

The (unbiasedness) condition $\mathbb{E}(c^Ty) = a^T\beta$ implies $DX = 0$. □

Gauss-Markov Theorem proves $\mathbb{V}(a^T\hat{\beta}) \leq \mathbb{V}(c^Ty)$ (The least squares estimator is the best unbiased linear estimator)

3.2 Ridge Regression

Ridge regression is a regularized version of the linear regression with the same closed form but adds a penalty term depending on the λ parameter to it to restrict the solution. The λ shrinks the regression coefficients by imposing a penalty on their size. This is done via a cost function:

$$\hat{\beta}^{ridge} = \underset{\beta}{\operatorname{argmin}}\left\{\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{d}x_{i,j}\beta_j)^2 + \underbrace{\lambda\sum_{j=1}^{d}\beta^2}_{L_2 \text{ norm penalty}}\right\}$$

$\lambda \leq 0$ controls the amount of shrinkage (Weight decay in Neuronal Network Literature). The penalty term is λ times the L_2 norm (Norms@Glossary), which is the Euclidian length of the β vector.
https://en.wikipedia.org/wiki/Tikhonov_regularization

Solution to Ridge Regression

The method used in Ridge regression is a Gradient descent solution. Shift the coordinate system into the center of mass of the data distribution:

$$x_{i,j} = x_{i,j} - \overline{x}_j, \beta_0 = \sum_{i=1}^{n}\frac{y_i}{n}$$

By this, we reduce the system from d+1 to d dimensions (homogeneous coordinates@Glossary) and get:

$$RSS(\beta) = (y - X\beta)^T(y - X\beta) + \lambda\beta^T\beta$$

The ridge solution is then:

$$\hat{\beta}^{ridge} = \underbrace{(X^T X + \lambda I)^{-1}}_{X^T X \text{ is regularized by } \lambda I} \quad X^T y$$

This can also be solved via Singular value decomposition (SVD@Glossary).

3.3 The LASSO

An alternative regularized version of least squares is LASSO (Least absolute shrinkage and selection operator). It uses the same unconstrained linear regression with the same closed form but adds a penalty term to it to restrict the solution, as it is done in the ridge regression.

$$\hat{\beta}^{LASSO} = \underset{\beta}{\text{argmin}} \left\{ \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{d} x_{i,j}\beta_j)^2 \right\}$$

subject to

$$\sum_{j=1}^{d} |\beta_j| \leq s$$

The lower equation is the penalty term, which is different from the ridge regression and is the L_1 norm penalty, which is the so-called "Manhattan distance" (Norms@Glossary). The LASSO does not have a closed form anymore and must must be solved numerically by using quadratic programming or more general convex optimization methods.
https://en.wikipedia.org/wiki/Least_squares#Lasso_method

Ridge vs. LASSO Estimation

One of the prime differences between LASSO and ridge regression is that in ridge regression, as the penalty is increased, all parameters are reduced while still remaining non-zero, while in LASSO, increasing the penalty will cause more and more of the parameters to be driven to zero. This is an advantage of LASSO over ridge regression, as driving parameters to zero deselects the features from the regression. Thus, Lasso automatically selects more relevant features and discards the others, whereas Ridge regression never fully discards any features.

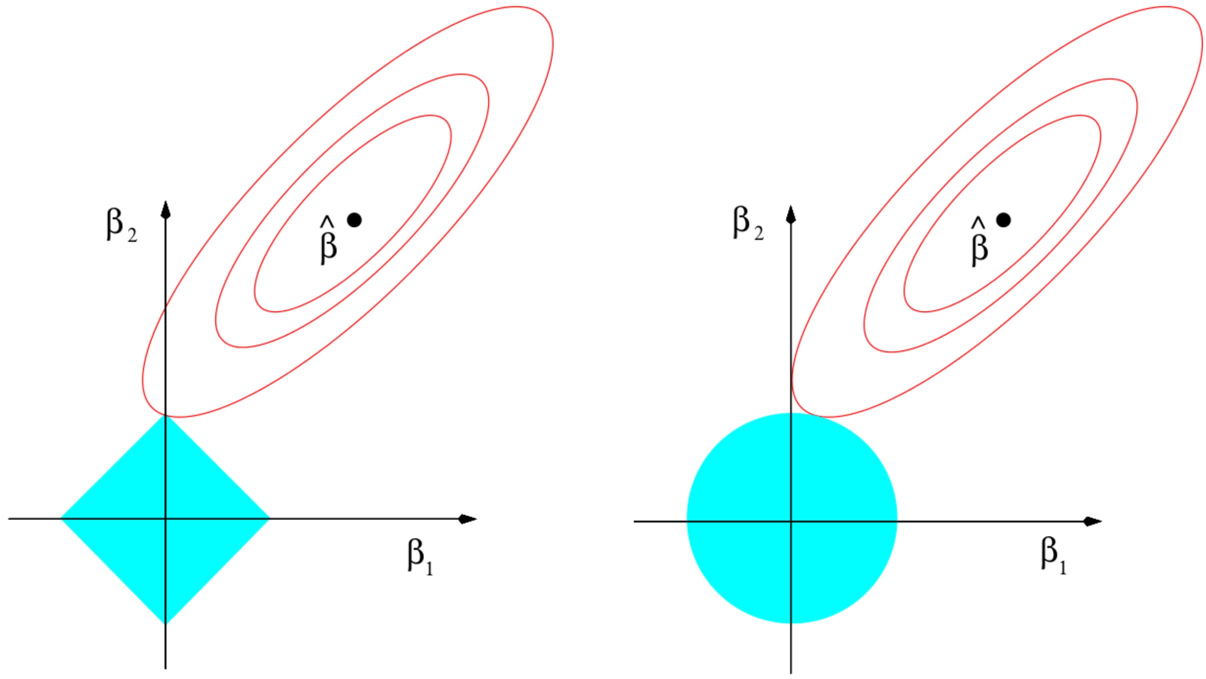


Figure 2: Estimation picture for the LASSO (left) and Ridge regression (right). Shown are contours of the least squares error function (red) and constraint functions $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t_2$ (blue). Therefore LASSO estimates are known to zero-out several coefficients and only keep a few non-zero. The reason is that the LSE error surface often hits the corners of the constraint surface(Source: fig 3.12 of Hastie et al. 2001)

Generalized Ridge Regression

The generalized ridge regression has known closed forms for certain values of p.

$$\hat{\beta}^{ridge} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{d} x_{i,j}\beta_j)^2 + \underbrace{\lambda \sum_{j=1}^{d} |\beta|^p}_{L_p \text{ norm penalty}} \right\}$$

For $p = 2$, we get the usual Ridge Regression with the closed form as described above. For $p = 1$, we get the LASSO Regression as described above and for all other values we get regression with different strengths of shrinkage as shown below.

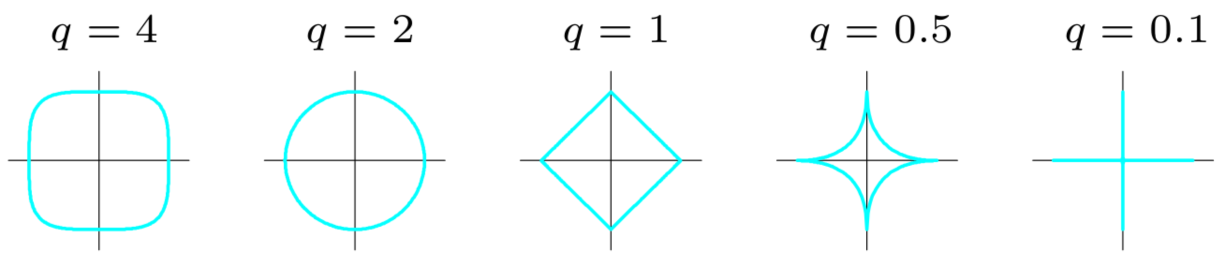


Figure 3: Contours of constraint functions $\sum_j |\beta_j|^p$ for different values of q (Source: HTF'01)

3.4 Nonlinear Regression by basis expansion

Idea: Transform the variables X nonlinearly and fit a linear model into the resulting (feature) space. Transformation: $h_m(X) : \mathbb{R}^d \rightarrow \mathbb{R}, 1 \leq m \leq M$

$$f(X) = \sum_{m=1}^{M} \beta_m h_m(X)$$

f is linear in β but non-linear in X! We use a series of piecewise-cubic polynomials to fit the data, of which the smoothness can be controlled by regularization of the the maximal number of knots.

3.5 Wavelet regression

Regression can also be done with Wavelets. A wavelet is a wave-like oscillation with an amplitude that begins at zero, increases, and then decreases back to zero. It can typically be visualized as a "brief oscillation" like one might see recorded by a seismograph or heart monitor. Generally, wavelets are purposefully crafted to have specific properties that make them useful for signal processing. Wavelets can be combined, using a "reverse, shift, multiply and integrate" technique called convolution, with portions of a known signal to extract information from the unknown signal. (Source: Wikipedia.org)

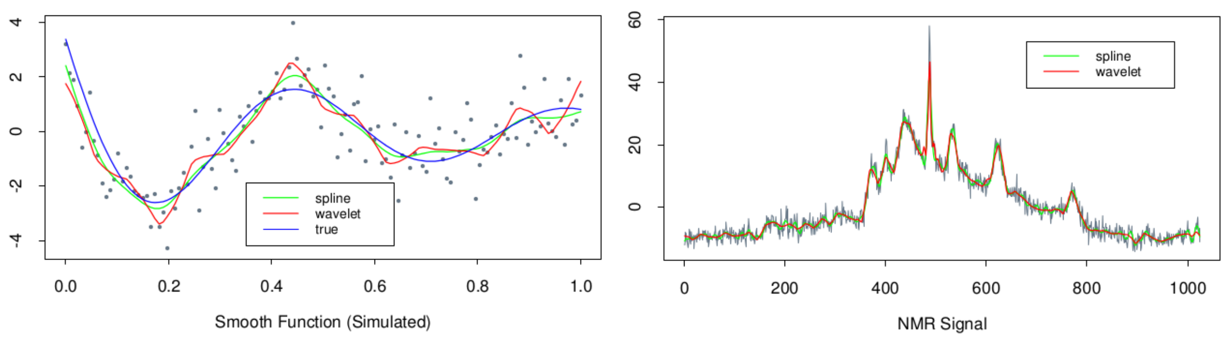


Figure 4: Wavelet smoothing compared with smoothing splines on two examples. Each panel compares the SURE-shrunk wavelet fit to the crossvalidated smoothing spline fit. (Source: HTF'01)

3.6 Bias-variance-Tradeoff in Regression

We have data $D = \{(x_i, y_i)\}_{i=1}^{n}, \;\; x_i \in \mathbb{R}^d, y_i \in \mathbb{R}$ which contains observations of the random variables (X_i, Y_i) which are drawn i.i.d. from the distribution $P(X, Y)$. We have the objective to find a regression function $f \in \mathcal{F}(\mathcal{F}$ is the hypothesis class) such that $\mathbb{E}(Y - f(X))^2$ is minimal. We can find the optimum at $f^*(x) = \mathbb{E}(Y|X = x)$. So we construct an estimator $\hat{f}(X)$ which estimates Y depending on the random variable X and on the data D.

Problems resulting in the Tradeoff

1. Finite training set
2. Complexity of hypothesis class $\mathcal{F}$ is unknown

So we must find the best balance between a complex $\mathcal{F}$ (by overfitting which keeps the variance high) and simple $\mathcal{F}$ (by underfitting which keeps the bias high). This can be done by minimizing the error components:

$$\begin{aligned}\mathbb{E}_D\mathbb{E}_{X,Y}\left(\hat{f}(X) - Y\right)^2 &= \mathbb{E}_D\mathbb{E}_X\left(\hat{f}(X) - \mathbb{E}(Y|X)\right)^2 + \mathbb{E}_{X,Y}\left(Y - \mathbb{E}(Y|X)\right)^2 \\ &= \underbrace{\mathbb{E}_X\mathbb{E}_D\left(\hat{f}(X) - \mathbb{E}_D(\hat{f}(X))\right)^2}_{variance} + \underbrace{\mathbb{E}_X\left(\mathbb{E}_D(\hat{f}(X)) - \mathbb{E}(Y|X)\right)^2}_{\text{bias}^2} + \underbrace{\mathbb{E}_{X,Y}\left(Y - \mathbb{E}(Y|X)\right)^2}_{\text{noise}}\end{aligned}$$

Usually, it is very complex to minimize them both at the same time.

Combining regressors

Bias A set of unbiased estimators that is simply averaged remains unbiased after averaging.

$$bias[f(x)] = \mathbb{E}_D\hat{f}(x) - \mathbb{E}(Y|x) = \frac{1}{B}\sum_{i=1}^{B}\mathbb{E}_D\hat{f}_i(x) - \mathbb{E}(Y|x) = \frac{1}{B}\sum_{i=1}^{B}\left(\mathbb{E}_D\hat{f}_i(x) - \mathbb{E}(Y|x)\right) = \frac{1}{B}\sum_{i=1}^{B} bias[f_i(x)]$$

Variance

$$\begin{aligned}\mathbb{V}\{\hat{f}(x)\} &= \mathbb{E}_D\hat{f}(X) - \mathbb{E}_D\hat{f}(X) = \mathbb{E}_D(\frac{1}{B}\sum_{i=1}^{B}\hat{f}_i(x) - \frac{1}{B}\sum_{i=1}^{B}\mathbb{E}_D\hat{f}_i(x))^2 \\ &= \mathbb{E}_D(\frac{1}{B}\sum_{i=1}^{B}(\hat{f}_i(x) - \mathbb{E}_D\hat{f}_i(x)))^2 = \frac{1}{B^2}\sum_{i=1}^{B}\mathbb{V}\{\hat{f}_i(x)\} + \frac{1}{B^2}\sum_{i\neq j}\sum \mathbb{C}ov\{\hat{f}_i(x), \hat{f}_j(x)\}\end{aligned}$$

Assuming that covariances are small $\mathbb{C}ov\{\hat{f}_i\} \approx 0$ and variances small as well $\mathbb{V}\{\hat{f}(x)\} \approx \sigma^2$ then we can reduce to $\mathbb{V}\{\hat{f}(x)\} \approx \frac{\sigma^2}{B}$. So the variance reduces by $\frac{1}{B}$ if bias remains the unchanged.

3.7 Readings

1. Duda 2nd ed.

Write down readings from the books covering the topics of the Regression section.

4 Numerical Estimation Techniques

The main problem of estimators is that we would like to be able to compute the expectation of some statistic such as the prediction error:

$$\mathcal{R}(c) = \mathbb{E}[\overbrace{\mathbb{I}_{\{c(x)\neq y\}}}^{\text{Indicator function counts errors}}] = \sum_{y=1}^{K} \int \mathbb{I}_{c(x_i)\neq y_i} P(x,y)dx$$

But we can only estimate the empirical mean

$$\hat{\mathcal{R}}(c) = \frac{1}{n} \sum_{i\leq n} \mathbb{I}_{c(x_i)\neq y_i}$$

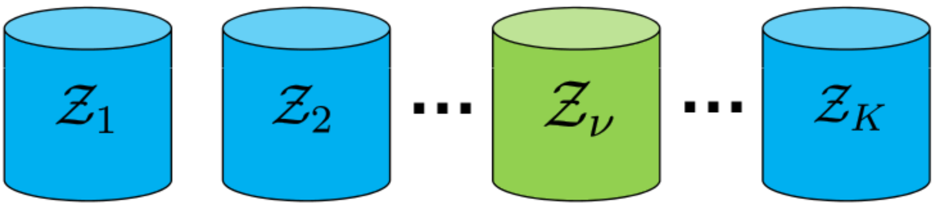
from the data we have.
The question is how far is $\mathcal{R}(c)$ from $\hat{\mathcal{R}}(c)$ and how different are the prediction functions $c^{opt} \in \operatorname{argmin}_{c(x)} \mathcal{R}(c)$ and $\hat{c}(x) \in \operatorname{argmin}_{c(x)} \hat{\mathcal{R}}(c)$?

4.1 Cross-Validation

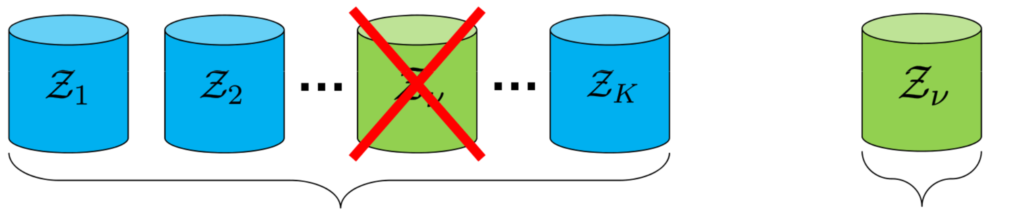
Cross-validation is a model validation technique for assessing how the results of a statistical analysis (here a prediction) will generalize to an independent data set. In a prediction problem, a model is usually given a dataset of known data on which training is run (**training dataset**), and a dataset of unknown data (or first seen data) against which the model is tested (**testing dataset**). The goal of cross validation is to define a dataset to "test" the model in the training phase (**validation dataset**), in order to limit problems like overfitting, give an insight on how the model will generalize to an independent data set. (Wikipedia)

K-fold Cross validation

In K-fold cross-validation, the original data is randomly partitioned into K equal size subsets.



Of the K subsets, a single subset $\mathcal{Z}_v$ is retained as the validation data for testing the model, and the remaining $K-1$ subsets are used as training data with $n\frac{K-1}{K}$ data samples.



Data sets: Training data set to learn $\hat{c}^{-\mathcal{Z}_v}(x)$ Validation data

The cross-validation process is then repeated K times (K-fold), with each of the K subsets used exactly once as the validation data. The K results from the folds can then be averaged (or otherwise combined) to produce a single estimation.

$$\hat{\mathcal{R}}^{cv} = \frac{1}{K} \sum_{v \leq K} \hat{\mathcal{R}}_v = \frac{1}{n} \sum_{i \leq n} \mathbb{I}_{\{y_i \neq \hat{c}^{-\mathcal{Z}_{v(i)}}(x_i)\}}$$

$\hat{c}^{-\mathcal{Z}_{v(i)}}(x_i)$ is the classifier which has been trained by omitting the data $(x_i, y_i) \in \mathcal{Z}_{v(i)}$. The $\mathcal{Z}_{v(i)}$ is then used for the error estimation.

Leave one out method

When $K = n$ (the number of observations), the K-fold cross-validation is exactly the leave-one-out cross-validation.

Engineer's solution to bias-variance problems

Problem: Highly correlated training sets can cause large prediction error variantion.
Solution: Choose K in K-fold cross validation as $min\{\sqrt{n}, 10\}$.

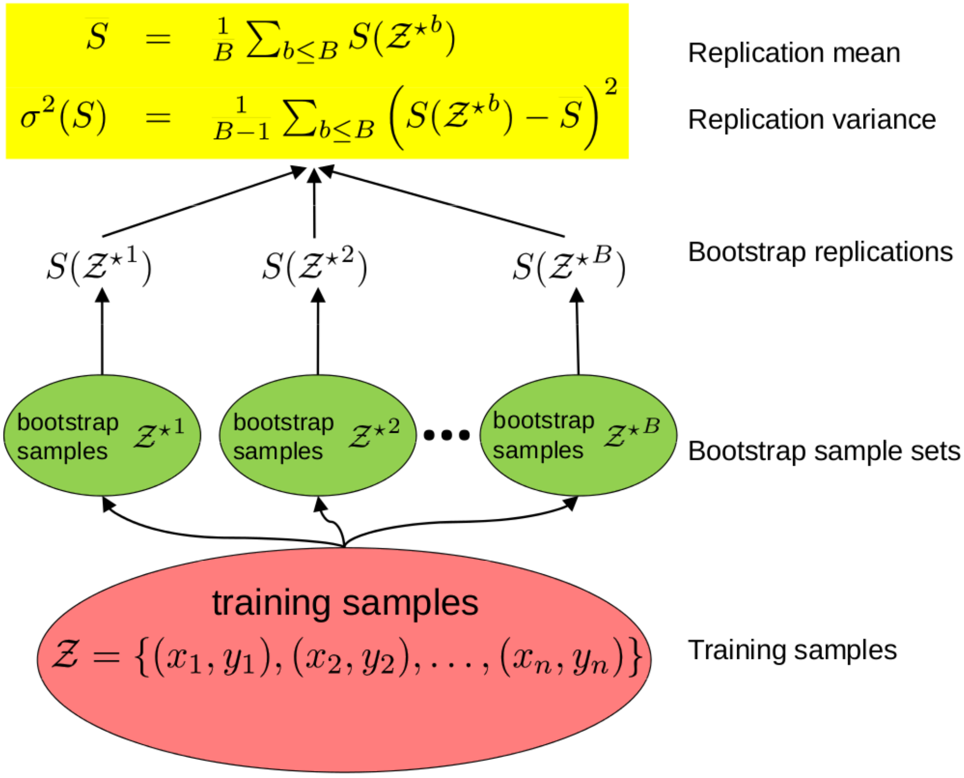
Cross-validation guided Model selection

Adapt feature set by subset selection and control it by minimizing the prediction error. Note that it is important to calculate the final prediction error on data which have not been used in model fitting (training) nor in model selection (testing for parameter adaptation). Therefore you can split your data set again into more subsets for this.

1. Run cross-validation for every value of the hyperparameter $(\lambda, \mathcal{C}, \ldots)$
2. Choose the value corresponding to the lowerst prediction error.
3. However you can not report this error as the generalization error of your algorithm, as the algorithm has seen all the data for model selection when choosing your hyperparameter.
4. So you need a separate dataset to evaluate the generalization error.

4.2 Bootstrapping

Bootstrapping is the practice of estimating properties of an estimator (such as its variance, bias, variance, confidence intervals, prediction error or some other such measure) by measuring those properties when sampling from an approximating distribution. One standard choice for an approximating distribution is the empirical distribution of the observed data. In the case where a set of observations can be assumed to be from an independent and identically distributed population, this can be implemented by constructing a number of resamples with replacement, of the observed dataset (and of equal size to the observed dataset). (Wikipedia)



Goal of Bootstrap Calculate numerically the estimation error of a statistic S(Z), e.g., the probability of error $\hat{\mathcal{R}}(\hat{c}(x))$ for classifier $\hat{c}(x)$ based on the empirical distribution function $\hat{F}_X(x)$. We approximate F_X by the empricial data source $\hat{F}_X$
Consistency of the bootstrap estimate:

$$\lim_{B\to\infty} \frac{1}{B-1} \sum_{b\le B} \left(S(\mathcal{Z}^{*b}) - \frac{1}{B} \sum_{\beta \le B} S(\mathcal{Z}^{*\beta}) \right)^2 = \mathbb{V}_{\hat{F}}[S(\mathcal{Z})]$$

Bootstrap works if the deviation between empirical and bootstrap estimator ($\hat{F}$ being the Bootstrap CDF) converges in probability to the deviation between true parameter value and the empirical estimator, i.e. $\mathcal{R}_n^{str}(\hat{F}, \hat{F}^*) - \mathcal{R}_n^{str}(F, \hat{F}) \xrightarrow{P} 0$, for $n \to \infty$ where $\mathcal{R}_n^{str}$ is one of the following:

- Error distribution: $\mathcal{R}_n^{err}(F,\hat{F}) = P(\sqrt{n}(S(\hat{F}) - S(F)))$
- Bias: $\mathcal{R}_n^{bias}(F,\hat{F}) = \mathbb{E}_F[S(\hat{F})] - S(F)$
- Standard error: $\mathcal{R}_n^{std}(F,\hat{F}) = \sqrt{\mathbb{E}_F[(S(\hat{F}) - S(F))^2]}$

Describe the e_0 bootstrap estimator if that is needed.

4.3 The "Jackknife" Method

In statistics, the jackknife is a resampling technique especially useful for variance and bias estimation. The jackknife predates other common resampling methods such as the bootstrap. The jackknife estimator of a parameter is found by systematically leaving out each observation from a dataset and calculating the estimate and then finding the average of these calculations. Given a sample of size N, the jackknife estimate is found by aggregating the estimates of each $N-1$ estimate in the sample. (Wikipedia)

Estimation

The jackknife estimate of a parameter can be found by estimating the parameter for each subsample omitting the ith observation to obtain an estimate $\bar{\theta}_i$. The overall jackknife estimator is found by averaging each of these subsample estimators.

$$\bar{\theta}_{\text{Jack}} = \frac{1}{n}\sum_{i=1}^{n}(\bar{\theta}_i)$$

Bias estimation and correction

The jackknife technique can be used to estimate the bias of an estimator calculated over the entire sample.

$$\bar{\theta}_{\text{BiasCorrected}} = N\bar{\theta} - (N-1)\bar{\theta}_{\text{Jack}}$$

This reduces bias by an order of magnitude, from $O(N^{-1})$ to $O(N^{-2})$.

This provides an estimated correction of bias due to the estimation method. The jackknife does not correct for a biased sample. Bias corrected estimators can have a considerably larger variance than uncorrected estimators!

Variance estimation

An estimate of the variance of an estimator can be calculated using the jackknife technique.

$$\mathbb{V}(\theta) = \mathbb{V}\left(\frac{\sum_{i=1}^{n}(X_i)}{n}\right) = \frac{\sigma^2}{n} = \frac{n-1}{n}\sum_{i=1}^{n}(\bar{\theta}_i - \bar{\theta}_{\text{Jack}})^2$$

where $\bar{\theta}_i$ is the parameter estimate based on leaving out the ith observation, and $\bar{\theta}_{\text{Jack}}$ is the jackknife estimator based on all of the samples.

Improve the text on the jackknife to fit the quality of the content of the lecture (Expansion not understood by Benjamin)

4.4 Hypothesis Testing

1. Define the null hypothesis $\mathcal{H}_0$ (devil's advocate).
2. Define the alternative $\mathcal{H}_1$ (one sided/two sided).
3. Find the test statistic.
4. Decide on the type I error α that you are willing to take.
5. Compute the Probability of observing the data given the null hypothesis: P-value.
6. Compare the P-value to α; if it is smaller, reject $\mathcal{H}_0$.
7. We define a **Type I error** as the test accepts $\mathcal{H}_0$ if in reality it is false (false negative).
8. We define a **Type II error** as the test rejects $\mathcal{H}_0$ if in reality it is true (false positive).

Neyman-Pearson Test

Selects a decision rules which minimizes the Type I error and fixes the Type II error.

Describe the Neyman-Pearson Test. Not understood by Benjamin

4.5 Readings

1. Duda 2ed

Write down readings from the books covering the topics of the Numerical Estimation Techniques section.

5 Classification

Classification in general is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. An example would be assigning a given email into "spam" or "non-spam" classes or assigning a diagnosis to a given patient as described by observed characteristics of the patient (gender, blood pressure, presence or absence of certain symptoms, etc.).

Classification is considered an instance of supervised learning, i.e. learning where a training set of correctly identified observations is available. The corresponding unsupervised procedure is known as clustering, and involves grouping data into categories based on some measure of inherent similarity or distance.

5.1 The problem of statistical decisions

n objects should be grouped in the classes $1, \dots, k, \mathcal{D}, \mathcal{O}$.

- $\mathcal{D}$: doubt class (more measurements required)
- $\mathcal{O}$: outlier class (definitely none of the classes $1, \dots, k$)

Objects: are characterized by feature vectors $X \in \mathcal{X}$(feature space).
Feature space: $\mathcal{X}(= \mathcal{X}_1 \times \mathcal{X}_2 \times \dots \times \mathcal{X}_d$ with $\mathcal{X}_i \subseteq \mathbb{R}) \subset \mathbb{R}^d$
Feature vector X: $\mathcal{O} \to \mathbb{R}^d \times \underbrace{[1, \dots, k]}_{classes}$

$o \to (X_0, Y_0)$
Data $\mathcal{Z}$: $\{(x_i, y_i) : 1 \le i \le n\}$

Use consistent notation: $\mathcal{O}$ stands for outlier class and for the object space at the same time.

5.2 Bayesian Decision Theory

The methods of statistical inference previously described are often referred to as classical methods. Bayesian methods (so called after the English mathematician Thomas Bayes) provide alternatives that allow one to combine prior information about a population parameter with information contained in a sample to guide the statistical inference process. A prior probability distribution for a parameter of interest is specified first. Sample information is then obtained and combined through an application of Bayes's theorem to provide a posterior probability distribution for the parameter. The posterior distribution provides the basis for statistical inferences concerning the parameter. A key, and somewhat controversial, feature of Bayesian methods is the notion of a probability distribution for a population parameter. According to classical statistics, parameters are constants and cannot be represented as random variables. Bayesian proponents argue that, if a parameter value is unknown, then it makes sense to specify a probability distribution that describes the possible values for the parameter as well as their likelihood. The Bayesian approach permits the use of objective data or subjective opinion in specifying a prior distribution. With the Bayesian approach, different individuals might specify different prior distributions. Classical statisticians argue that for this reason Bayesian methods suffer from a lack of objectivity. Bayesian proponents argue that the classical methods of statistical inference have built-in subjectivity (through the choice of a sampling plan) and that the advantage of the Bayesian approach is that the subjectivity is made explicit.

Bayesian methods have been used extensively in statistical decision theory (see below Decision analysis). In this context, Bayes's theorem provides a mechanism for combining a prior probability distribution for the states of nature with sample information to provide a revised (posterior) probability distribution about the states of nature. These posterior probabilities are then used to make better decisions.

Bayes's theorem Bayes's theorem is in probability theory a means for revising predictions in light of relevant evidence, also known as conditional probability or inverse probability. The theorem was discovered among the papers of the English Presbyterian minister and mathematician Thomas Bayes and published posthumously in 1763. Related to the theorem is Bayesian inference, or Bayesianism, based on the assignment of some a priori distribution of a parameter under investigation. In 1854 the English logician George Boole criticized the subjective character of such assignments, and Bayesianism declined in favour of confidence intervals and hypothesis tests, now basic research methods.

Problem Setting for Bayesian Inference

Bayesian view: Both feature vector X and class label Y of an object O are random variables!
Notation:
π_y : fraction of the samples out of class Y = y (π_y is known)
$p_y(x) := P\{X = x|Y = y\}$: probability distribution / density of the samples out of class Y = y (class conditional density)

1. $p_y(x)$ is unknown $\rightarrow$ density estimation
2. $p_y(x)$ is known up to a parameter $\rightarrow$ parameter estimation

Bayes rule: $\underbrace{P\{model|data\}}_{\text{posterior}} = \dfrac{\overbrace{P\{data|model\}}^{\text{likelihood}}\overbrace{P\{model\}}^{\text{prior}}}{\underbrace{P\{data\}}_{\text{evidence}}}$

- $\overbrace{P\{model\}}^{\text{prior}}$: This indicates one's previous estimate of the probability that a hypothesis is true, before gaining the current evidence.
- $\overbrace{P\{data|model\}}^{\text{likelihood}}$: It indicates the compatibility of the evidence with the given hypothesis.
- $\underbrace{P\{model|data\}}_{\text{posterior}}$: This tells us what we want to know: the probability of a hypothesis given the observed evidence.
- $\underbrace{P\{data\}}_{\text{evidence}}$: It is the probability of the data being observed at all.
- the quotient $\frac{P(model|data)}{P(data)}$ represents the support the data provides for the model.

5.3 Bayesian Classifier

Assume that we know how the features are distributed for the different classes, i.e., the class conditional densities p y (x) and their parameters are known. What is the best classification strategy in this situation?

$$\hat{c} : \mathcal{X} \rightarrow \hat{c}(X) \rightarrow \mathcal{Y} := \{1, \dots, k, \mathcal{D}\}$$

(Mapping of $\mathcal{X}$ into $\mathcal{Y}$. No outlier class)

Classification error

But we must minimize the classification error.

$$\hat{\mathcal{R}}(\hat{c}|\mathcal{Z}) = \sum_{x \in \mathcal{X}} \mathbb{I}_{\{\hat{c}(x) \neq y\}} = \sum_{(x_i, y_i) \in \mathcal{Z}} \mathbb{I}_{\{\hat{c}(x_i) \neq y_i\}}$$

Note that this error count is a random variable! Expected errors are also called the expected risk and they define the quality of a classifier

$$\mathcal{R}(\hat{c}) = \sum_{y \leq k} P(y) \mathbb{E}_{P(x|y)} \mathbb{I}_{\{\hat{c}(x) \neq y\}} |Y = y] + \text{terms from D}$$

Loss function

Weighted mistakes are introduced when classification errors are not equally costly. We introduce a loss function $L(y,z)$ which denotes the loss for the decision z if class y is correct. Classification costs can also be asymmetric, that means $L(y,z) \neq L(z,y)$ (Think of one class having worse outcomes than the other).
Example for Loss functions is the $L^{0-1}(y,z)$ function.

$$L^{0-1}(y,z) = \begin{cases} 0 & \text{if } z = y \\ 1 & \text{if } z \neq y \wedge z \neq D \text{ (wrong decision)} \\ d & \text{if } z = D \text{ (no decision)} \end{cases}$$

with according expected loss:

$$\mathcal{R}(\hat{c}, y) = \mathbb{E}_X[L^{0-1}(y, \hat{c}(x))|Y = y] = \underbrace{P\{\hat{c}(x) = y \wedge \hat{c}(x) = D|Y = y\}}_{\text{pmc(y) probability of misclassification}} + \underbrace{d \cdot P\{\hat{c}(x) = D|Y = y\}}_{\text{pd(y) probability of doubt}}$$

or in an general version

$$\mathcal{R}(\hat{c}) = \sum_{z \leq k} \overbrace{\pi_z}^{=P(Z=z)} pmc(z) + d \cdot \sum_{z \leq k} \pi_z pd(z) = \mathbb{E}_C[\mathcal{R}(\hat{c}, C)]$$

Posterior class probability

$$\overbrace{p(y|x) \equiv P\{Y = y|X = x\}}^{Posterior} = \frac{\overbrace{\pi_y}^{Prior} \overbrace{p_y(x)}^{Likelihood}}{\underbrace{\sum_z \pi_z p_z(x)}_{Evidence}}$$

- Prior: $\pi_y := P(Y = y)$ is the probability of class $Y = y$.
- Likelihood: The class conditional density $p_y(x)$ is the probability of observing data $X = x$ given class $Y = y$ $(= P\{X = x|Y = y\})$.

Bayes classifier

General principle of the Bayes classifier: Select the class y^{MAP} with highest $\pi_y p_y(x)$ value if the costs for not making a decision exceed the loss of this class y MAP , i.e., $\pi_y p_y(x) > (1-d)p(x)$.

Classification rule for L^{0-1} loss is:

$$c(x) = \begin{cases} y & \text{if } p(y|x) = \max_{z \le k} p(z|x) > 1-d, \\ D & \text{if } p(y|x) \le 1 - d \forall y. \end{cases}$$

Generalization to arbitrary loss functions:

$$c(x) = \begin{cases} y & \text{if } \sum_z L(z,y)p(z|x) = \min_{\rho \le k} \sum_z L(z,\rho)p(z|x) \le d \\ D & else. \end{cases}$$

Remark on Outliers: The outlier concept causes conceptual problems and it does not fit to the statistical decision theory since outliers indicate an erroneous or incomplete specification of the statistical model! But for completeness:

$$\pi_{\mathcal{O}} p_{\mathcal{O}}(x) \ge \max\left\{(1-d)p(x), \max_z \pi_z p_z(x)\right\}$$

5.4 Discriminant functions

Discriminant functions take the feature vector as an input and decide if the object belongs to their class. The highest output of a discriminant function g_c makes the object belong to class c: $\forall z \neq y: \ g_c(x) > g_z(x) \Rightarrow$ class c. The classifier is said to assign a feature vector x to class ω_i if

$$g_i(x) > g_j(x) \text{ for all } j = i$$

Thus, the classifier is viewed as a network or machine that computes c discriminant functions and selects the category corresponding to the largest discriminant.

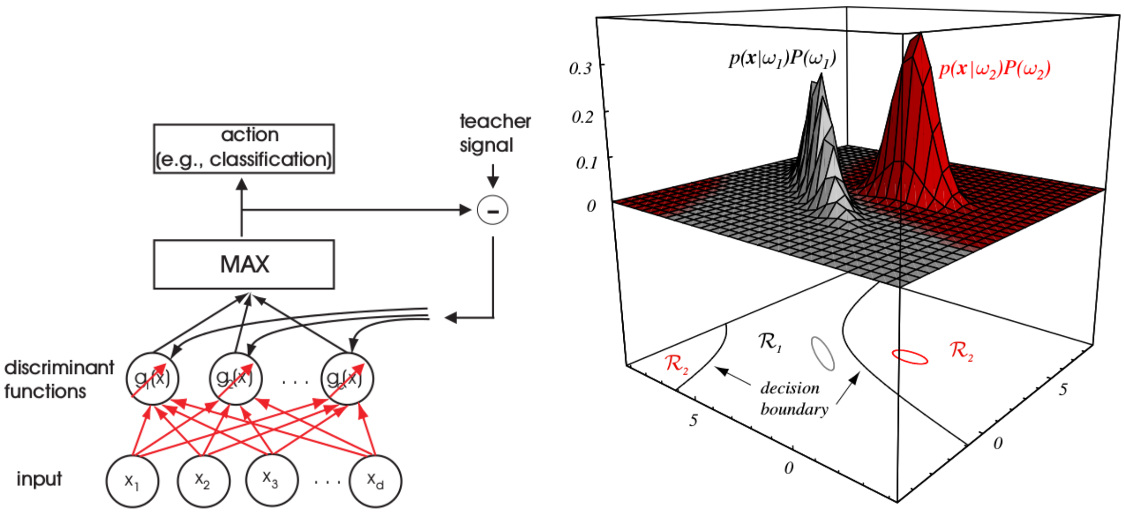


Figure 5: The red connections (weights) are adapted in such a way that the teacher signal is imitated by the discriminant function. Source: Duda

Linear classifiers

We use normal distributions to describe the likelihood of class y:

$$p_y(x) = \frac{1}{\sqrt{2\pi^d | \underbrace{\Sigma_y}_{\text{Covariance matrix for class y}} |}} exp(-\frac{1}{2}(x-\mu_y)^T \Sigma_y^{-1}(x-\mu_y))$$

Linear classifiers have a linear decision rule:

$$w^T(x - x_0) = 0$$
$$w = \mu_{y1} - \mu_{y2}$$
$$x_0 = \frac{1}{2}(\mu_{y1} + \mu_{y2}) - \frac{\sigma^2 w}{||w||^2} \log \frac{\pi_{y1}}{\pi_{y2}}$$

The separating hyperplane is defined by the difference vector w and the offset x_0. The hyperplane is shifted by the log prior relative to the average of the two means.

Generalized linear classifier $\Sigma_y = \Sigma \forall y$

$$w^T(x - x_0) = 0$$
$$w = \Sigma^{-1}(\mu_{y1} - \mu_{y2})$$
$$x_0 = \frac{1}{2}(\mu_{y1} + \mu_{y2}) - \frac{\mu_{y1} - \mu_{y2}}{\underbrace{(\mu_{y1} - \mu_{y2})^T w}_{\text{Mahalanobis distance}}} \log \frac{\pi_{y1}}{\pi_{y2}}$$

How much detail is important here?

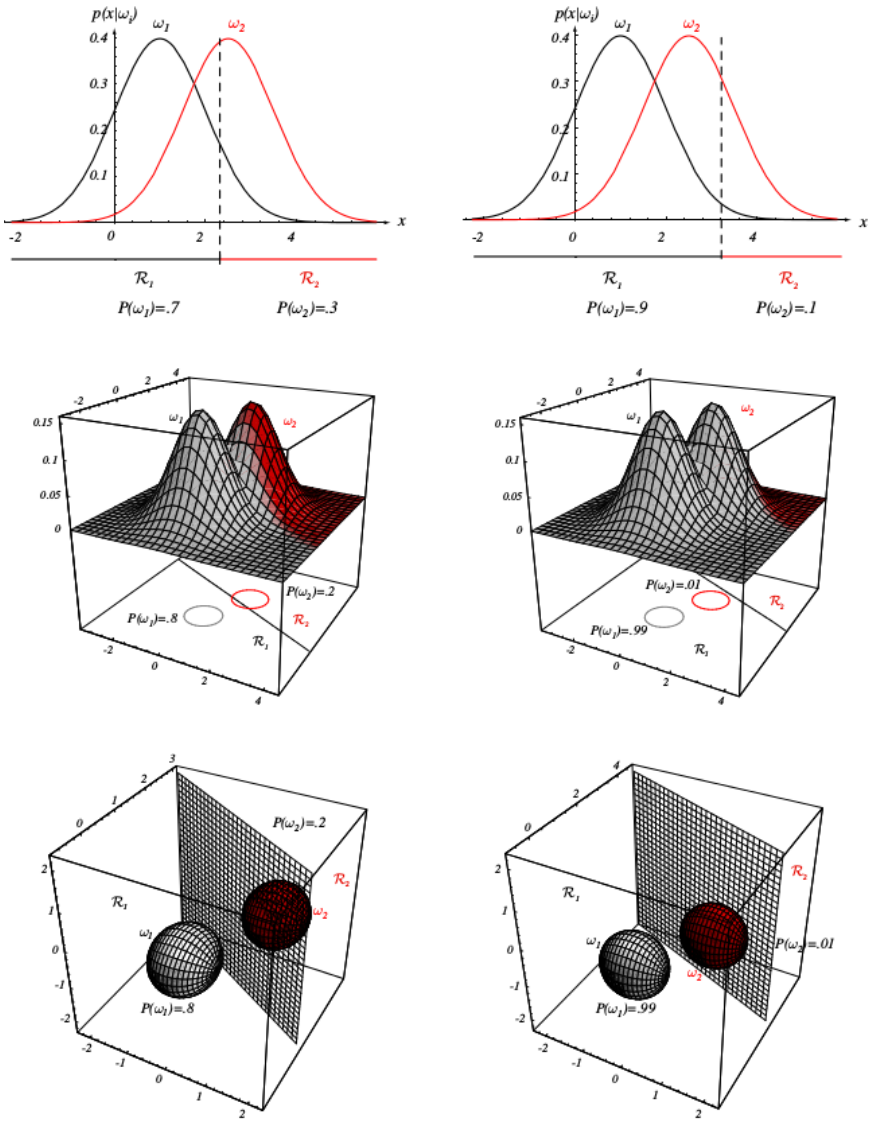


Figure 6: If the covariance matrices for two distributions are equal and proportional to the identity matrix, then the distributions are spherical in d dimensions, and the boundary is a generalized hyperplane of d - 1 dimensions perpendicular to functions the line separating the means. In these one-, two-, and three-dimensional examples, we indicate $p(x|\omega_i)$ and the boundaries for the case $P(\omega_1) = P(\omega_2)$. In the three-dimensional case, the grid plane separates $\mathcal{R}_1$ and $\mathcal{R}_2$. The prior can cause a significant shift of the linear discriminant function in the direction of the less probable mode. Strong priors might even shift the linear discriminant function beyond the mode peak of the alternative class (see right column).

5.5 Readings

1. Duda 2ed., Chapter 2,Bayesian Decision Theory (2.1,2.2,2.3.2,2.4,2.5,2.6)

Write readings separately for all topics of classification.

6 Parametric Models

Unfortunately, in pattern recognition applications we rarely if ever have this kind of complete knowledge about the probabilistic structure of the problem. In a typical case we merely have some vague, general knowledge about the situation, together with a number of design samples or training data - particular representatives of the patterns we want to classify. The problem, then, is to find some way to use this information to design or train the classifier.
In statistics, a parametric model or parametric family or finite-dimensional model is a family of distributions that can be described using a finite number of parameters. These parameters are usually collected together to form a single k-dimensional parameter vector $\theta = (\theta_1, \theta_2, \ldots, \theta_k)$.

If we know the number of parameters in advance and our general knowledge about the problem permits us to parameterize the conditional densities, then the severity of these problems can be reduced significantly. The problem of parameter estimation is a classical one in statistics, and it can be approached in several ways. We shall consider two common and reasonable procedures, maximum likelihood estimation and Bayesian estimation.

6.1 Maximum Likelihood Estimation

Likelihood of the data set: $P(\mathcal{X}|\theta_y) = \prod_{i \leq n_y} p(x_i|\theta_y)$

Estimation principle: Select the parameter $\hat{\theta}_y$ which maximizes the likelihood, i.e., the probability of the data given the parameter:

$$\hat{\theta}_y = \underset{\theta_y}{\operatorname{argmax}} P\{X_y|\theta_y\}$$

The random variable $\hat{\theta}_y(\mathcal{X}_y)$ is called an estimator for the parameter θ.
Procedure: Find the extremum of the log-likelihood function:

$$\Lambda(\theta) \triangleq \nabla_{\theta_y} \log P\{\mathcal{X}|\theta_y\} = \frac{\partial}{\partial \theta_y} \sum_{i \leq n} \log p(x_i|\theta_y) = 0$$

Remarks on Maximum Likelihood estimators

Bias of an estimator $bias(\hat{\theta}_n) = \mathbb{E}[\hat{\theta}_n] - \theta$ The bias measures how much the expected value of the estimator deviates from the true parameter value. The design of unbiased estimators ($\mathbb{E}[\hat{\theta}_n] = \theta$) and bias reduction has been considered as a goal of statistics, although the work of Stein has shown that there exists biased estimators which are better in the least squares sense than any unbiased estimator.

Consistent estimator A point estimator $\hat{\theta}_n$ of a parameter θ is consistent if $\hat{\theta}_n \xrightarrow{P} \theta$, i.e.,

$$\forall \varepsilon P\{|\hat{\theta}_n - \theta| > \varepsilon\} \overset{n \to \infty}{\rightarrow} 0$$

Efficiency ML estimators are asymptotically efficient estimators, i.e.,

$$\lim_{n \to \infty} \left(\mathbb{V}[\hat{\theta}^{ML}(x_1, \ldots, x_n)] I(\theta) \right)^{-1} = 1, \quad I = \mathbb{V}\left[\frac{\partial \log P(x|\theta)}{\partial \theta} \right]$$

Rao-Cramer Inequality

The Rao-Cramer inequality states that the variance of an estimator is bounded from below by the inverse Fisher information. The fisher information is denoted by:

$$I(\theta) = \overbrace{\mathbb{V}\left[\frac{\partial \log P(x;\theta)}{\partial \theta} \right]}^{\text{Variance of the score function}} = \underbrace{\mathbb{E}\left[\left(\frac{\partial \log P(x;\theta)}{\partial \theta} \right)^2 \right]}_{\text{Expected value of the observed information}} = -\mathbb{E}\left[\frac{\partial^2 \log P(x;\theta)}{\partial \theta^2} \right]$$

Let $\hat{\theta}(X_1, \ldots, X_n)$ be an unbiased estimator of θ. then the Rao Cramer inequality holds for the class of unbiased estimators:

$$\int (\theta - \hat{\theta})^2 P(x_1, \ldots, x_n; \theta) dx_1 \ldots dx_n \leq \frac{1}{I(\theta)}$$

Convergence of Maximum Likelihood Estimators

The Maximum Likelihood estimator converges (in distribution; Central Limit Theorem) to the best model θ_0 as $n \to \infty$ (Asymptotic normality).

$$\sqrt{n}(\hat{\theta}_n^{ML} - \theta_0) \overset{n \to \infty}{\rightarrow} \mathcal{N}(0, J^{-1}(\theta_0) I(\theta_0) J^{-1}(\theta_0))$$

$$J(\theta) = -\mathbb{E}\left[\frac{\partial^2 \log P(x|\theta)}{\partial \theta^2} \right]$$

$$I(\theta) = -\mathbb{V}\left[\frac{\partial \log P(x|\theta)}{\partial \theta} \right]$$

Do we need the proof for that?

6.2 Bayesian Learning (batch/online)

Goal: To predict the value of one or more continuous target variables t given d-dim x input vectors.

LINEAR regression : most simple approach

Specific EXAMPLE of Linear regression:

Type of variables : independent $X \in \mathbb{R}^d$

dependent $Y \in \mathbb{R}$

We assume a linear relationship: $y = x^T \beta$

$$y = \sum_{i=1}^{d} x_i \beta_i$$

⇒ FXN: polynomial curve fitting.

* NOTE THIS APPROACH IS A NON-BAYESIAN / Frequentist. Applying the frequentist notion of probability to the random values of the observed variables t_n.

vector form

STATISTICAL MODEL

Generalized setting:

$$y = \beta_0 + \sum_{j=1}^{d} \beta_j x_j \quad , \text{ where } \beta_0 = \text{intercept, bias term}$$

Now introduce homogenous coordinates : $X_0 = 1$
to rewrite generalized equation in a compact form. This way you don't worry about the offset from the origin on the y-axis

$$\Rightarrow \; y = x^T \beta \qquad x, \beta \in \mathbb{R} \rightarrow d+1 \text{ dimensions}$$

Now with noisy observations in data: $(x_i, y_i) \sim p(x, y \mid \beta)$
are distributed according to some β.

You can assume the equation with noise that is iid. in which case, you are also trying to model the noise by predicting β.

$y_i = x_i^T \beta + \varepsilon_i$, where ε_i is noise with iid.

ε_i : $1 \leq i \leq n$ is iid.

So the question changes to; what is the probability of the ~~observed~~ observing a noise ε_i given a β.

$$P(\varepsilon_i \mid \beta) = P(\underbrace{y_i - x_i^T \beta}_{\text{residuals}} \mid \beta)$$

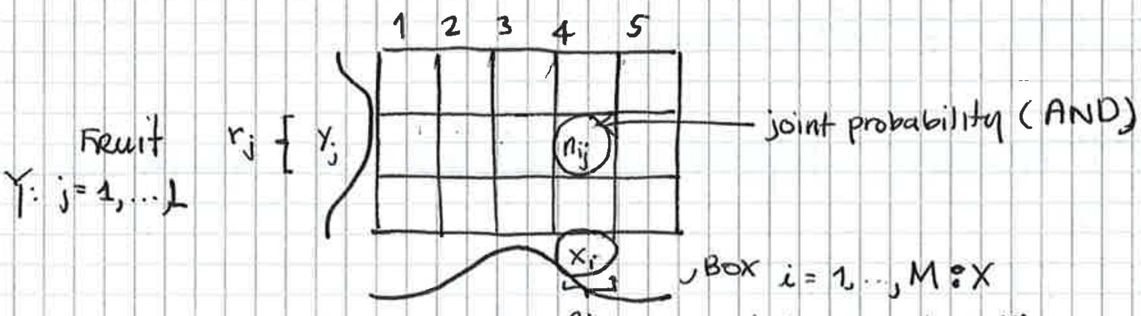
If you know what ε_i, then you would know how $y_i - x_i^T \beta$ is distributed.

y_i = True value

$\hat{y}_i = x_i^T \beta_i$, value of dependent variable/(prediction) predicted by the model.

Estimation problem involves determing β from observations (training data),

$$Z = \{ (x_i, y_i) , \; 1 \leq i \leq n \}$$



x can take any values x_i

Two RV X and Y

total N trials

$P(X = x_i) = \frac{c_i}{N}$ probability that X takes x_i value irrespective of Y, that fall into a particular column.

n_{ij} when we sample from both RV

C_i = # of trails X_i takes irrespective of Y

Joint probability: that X will take x_i value and Y will take y_i value

$$P(X = x_i, Y = y_j) = \frac{n_{ij}}{N}$$

We take a bayesian view-point to quantitavely estimate the uncertainty. Consider β as a RV determined by nature.] Bayesian

Joint probability between X, Y, and β:

$$P((x,y), \beta) = P(x,y \mid \beta) \cdot P(\beta)$$

marginal prob = $P((x,y))$

Joint prob = $P((x,y), \beta)$

observations = $P(x,y)$

We apply bayes formula:

$$P((x,y), \beta) = P(\beta \mid (x,y)) \cdot P((x,y))$$

$$\frac{P((x,y),\beta)}{P((x,y))} = P(\beta \mid (x,y))$$

posterior probability ⇒ the conditional probability that is assigned when the relevant evidence is taken into account.

"posterior" term = relative future tense.

$$P(\beta \mid (x,y)) = \frac{\overbrace{P((x,y) \mid \beta)}^{\text{likelihood fxn}} \cdot \overbrace{P(\beta)}^{\text{prior, previous knowledge}}}{\underbrace{P((x,y))}_{\text{Evidence}}}$$

What is the probability of (x,y) occurring?

$$P((x,y)) = \int_{\Omega} P((x,y)|\beta) \cdot P(\beta)\, d\beta \quad \Omega \text{ (domain of } \beta \text{, possible values of } \beta)$$

Let us assume that ε_i are ϕ mean: $\mathbb{E}_{(x,y)}[\varepsilon] = 0$

Likelihood fxn, The likelihood of observing a set of parameters given observations

$$L(\theta|x) = P(x|\theta)$$

Error of probability: $P(\varepsilon_1, \dots, \varepsilon_n|\beta) = \prod_{i=1}^{n} P(\varepsilon_i|\beta) = \prod_{i=1}^{n} P(y_i - x^T\beta|\beta)$

↑ dependent ↑ independent

Deterministic fxn $Y(x,\beta)$ with additive Gaussian noise

$$y_i = x_i^T\beta + \varepsilon_i \Rightarrow y_i - x_i^T\beta = \varepsilon_i$$

How would you suspect the RV ε_i behaves? IT should be normally distributed.

$y_i = x_i^T\beta + 0$, where is gaussian distributed.

$$y_i - \underbrace{x_i^T\beta}_{\sum_{j=1}^{d} x_j\beta_j} = 0$$

If ε is gaussian and normally distributed, then $\varepsilon_i = y_i - x_i^T\beta$ can be fitted by least squares

$$\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$$

mean $= 0$
variance $= \sigma^2$

$$= \frac{1}{\sqrt{2\pi\sigma^2}} \exp \frac{(\varepsilon_i)^2}{2\sigma^2}$$

Cost/Error fxn

We then take the log likelihood and want to minimize the cost:

$$-\log P(\varepsilon_1 \cdots \varepsilon_n|\beta) = \sum_{i=1}^{n} \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(y_i - x^T\beta)^2}{2\sigma^2}\right)\right]$$

↑ minimize cost ↑ log ↑ likelihood fxn

$$= \sum_{i=1}^{n} \frac{(y_i - x^T\beta)^2}{2\sigma^2} + \frac{N}{2}\log 2\pi\sigma^2$$

Because of sq-root and sum.

We will continue we deriving a linear regression equation built on the MLE method.

We will estimate $\hat{\beta}_{ML}$ that will be used for prediction (iff x^TX is a non-singular matrix).

The final prediction (inference of out $\hat{\beta}$ estimator is):

$$\hat{Y}_{i+1} = x_{i+1}^T \hat{\beta}_{ML} = x_{i+1}^T \underbrace{(x^TX)^{-1} X^TY}_{\text{Comes from what you estimated}}$$

↓ Estimate scalar n-dim vector

↓ MLE estimator

Now we want to fit the data to our chosen statistical model (linear regression) by using residual sum of squares → cost function.

- We show that minimizing a sum-of-squares error fxn could be motivated as the ML solution under an assumed Gaussian noise model.
- You can take an inference method. You will need this step because you have to compute how well you come up with beliefs.

  - Apply "maximum likelihood" method.

Select β such that it maximizes the probability of observing the data given.

$$\hat{\beta} \in \arg\max_{\beta} \{ \underbrace{P(\varepsilon_1, \dots, \varepsilon_n \mid \beta)}_{\text{likelihood}} \} \qquad \hat{\beta} = \text{estimate value}$$

We fit a polynomial fxn to data set by minimizing a sum-of-squares error fxn.
objective: is to find the β that maximizes the objective fxn $P(x_1, \dots, x_n \mid \theta)$

Change to log-likelihood fxn to make computation easier and take the negative to find lowest residual error

$$\text{cost/error fxN} = \arg\min_{\beta} \left[-\log P(\varepsilon_1, \dots, \varepsilon_n \mid \beta) \right]$$

* put a minus sign if you want to look for the min β.

$$= \arg\min_{\beta} \sum_{i=1}^{n} (y_i - x_i^T \beta)^2$$

// cost fxN.
// minimize the residual sum of squares

$$RSS(\beta) = \sum_{i=1}^{n} (y_i - x_i^T \beta)^2$$

Change above expression into matrix notation.

$$= \arg\min_{\beta} (Y - X\beta)^T (Y - X\beta)$$

, where X is $n \times (d+1)$ matrix
Each row is an input vector x.
Y is an $n \times 1$ of the outputs in the training set.

In order to compute the minimum condition ($\hat{\beta}$), we set the equation equal to $\emptyset$ and differentiate to minimize.

Minimizing of: $(Y - \vec{x}\beta)^T (Y - \vec{x}\beta)$

$$0 = (Y - \vec{x}\beta)^T (Y - \vec{x}\beta) \frac{d}{d\beta}$$

$$-2\vec{x}^T (Y - \vec{x}\beta) = 0$$

$$= -2\vec{x}^T Y + 2\vec{x}^T \vec{x}\beta = 0$$

$$-2X^T Y + 2X^T X \beta = 0$$

$$2X^T X \beta = 2X^T Y$$

$$X^T X \beta = X^T Y$$

$$\beta = \frac{X^T Y}{X^T X}$$

$$\frac{((d+1) \times n)(n \times 1)}{((d+1) \times n)(n \times (d+1))} = \frac{((d \times 1) \times 1)}{((d+1) \times (d+1))}$$

↑ matrix needs to be nonsingular!

If so, then

$$\hat{\beta} = (X^T X)^{-1} (X^T Y)$$

CLOSED-FORM SOLUTION!

Find Estimate of $\hat{\beta}_{ML}$:

$$\boxed{\hat{\beta}_{ML} = (X^T X)^{-1} X^T Y}$$

⇒ **prediction:**

$$\boxed{\hat{Y} = X^T \hat{\beta}}$$

X: $(d+1) \times n$ β: $(d+1) \times 1$ Y:

$$(n \times \cancel{(d+1)})(\cancel{(d+1)} \times 1)$$

$$n \times 1$$

Here we introduce another method: Ridge regression

In the bayesian world, you would need to infer β parameter values, which you would then not go for the ML method (frequentist approach).
Instead you would ask what is the most probable, likely β, that is most probable for seeing your data?

$$Z = \{(x_i, y_i) \quad 1 \le i \le n\}$$, Full set of observations

$$= \{z_1, \dots, z_n \quad 1 \le i \le n\}$$

Bayes Formula: $$P(\beta | Z) = \frac{\overbrace{P(Z|\beta)}^{\text{likelihood fxn}} \cdot \overbrace{P(\beta)}^{\text{Prior}}}{\underbrace{P(Z)}_{\text{Evidence}}}$$

Fixed data Z.

Then you might go for: MAXIMUM A POSTERIORI ESTIMATE (MAP)

$$\hat{\beta}_{MAP} \in \arg\max_{\beta} P(\beta | Z)$$ posterior probability $P(\beta|Z)$.

$$= \arg\min_{\beta} \{ \overbrace{-\log P(Z|\beta)}^{\text{likelihood}} - \log P(\beta) \}$$ minimizing ~~[illegible]~~. chosen model

* maximization of a likelihood fxn under conditional Gaussian noise distribution for a linear model is the same to minimize a sum-of-squares error fxn.

REGULARIZATION

Before you look at the data, know that we assumed β to be distributed by nature according to gaussian law.

How to choose $P(\beta)$? prior

β is distributed by nature, $$P(\beta) = \prod_{i=1}^{d} \frac{\exp\left(\frac{-\beta^2}{2\sigma_\beta^2}\right)}{\sqrt{2\pi\sigma_\beta^2}}$$

Take the log of $P(\beta)$.

$$-\log P(\beta) = \sum_{j=1}^{d} \frac{\beta^2}{2\sigma_\beta^2} + \text{constant}$$

d = dim
n = examples

$$= \arg\min_{\beta} \{ \underbrace{\sum_{i=1}^{n} (y_i - \underbrace{x_i^T \beta}_{\text{prediction}})^2}_{\text{error term}} + \underbrace{\lambda \sum_{j=1}^{d} \beta^2}_{\text{penalty regularizer, scalar to penalize max-likelihood fxn}} \}$$ // cost fxn for ridge regression

used to control model complexity, its an isotropic type of penalty term for choosing the number of β's you many want to restrict.

cost fxn depends on β. $$RSS(\beta) = \underbrace{(Y - \vec{X}\beta)^T (Y - X\beta)}_{\text{data term}} + \underbrace{\lambda \|\beta\|^2}_{\text{penalty term}}$$

where λ = lagrange parameter

$\lambda = \frac{1}{2\sigma_\beta^2}$, validated by data and controls solution to fitting parameters of the model.

Another Regularization method: LASSO.

Goal is to choose a sparse β.

$$P_{sparse}(\beta) = \prod_{i=1}^{d} e^{-\lambda|\beta|}$$, penalty prior $P_{sparse}(\beta)$

$$-\log\left(P_{Lasso}(\beta)\right) = \lambda \sum_{i=1}^{d} |\beta| .$$ // LASSO, least absolute solution operation

- Favors β values to be exactly $\emptyset$.

- SPARSENESS penalty, penalty measures when you invested in a beta value.

$$-\log P_{sparse}(\beta) = \underbrace{S}_{\text{penalty term}} \sum_{j=1}^{d} \mathbb{I}_{\{\beta_j \neq 0\}}$$

Now, this leads to a combinatorial optimization problem.

Probably somebody should tex the bayesian part of this at some point.

6.3 Readings

1. Maximum Likelihood Estimators: Duda chp 3.1,3.2
2. Bayesian Estimator: Duda chp 3.3,3.4,3.5

7 Design of Linear Discriminant Functions

Linear discriminant analysis (LDA) as in the related Fisher's linear discriminant are methods used in statistics, pattern recognition and machine learning to find a linear combination of features which characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier, or, more commonly, for dimensionality reduction before later classification. (Wikipedia, Linear Discriminant Analysis https://en.wikipedia.org/wiki/Linear_discriminant_analysis.

7.1 Generalized Linear Discriminant Functions

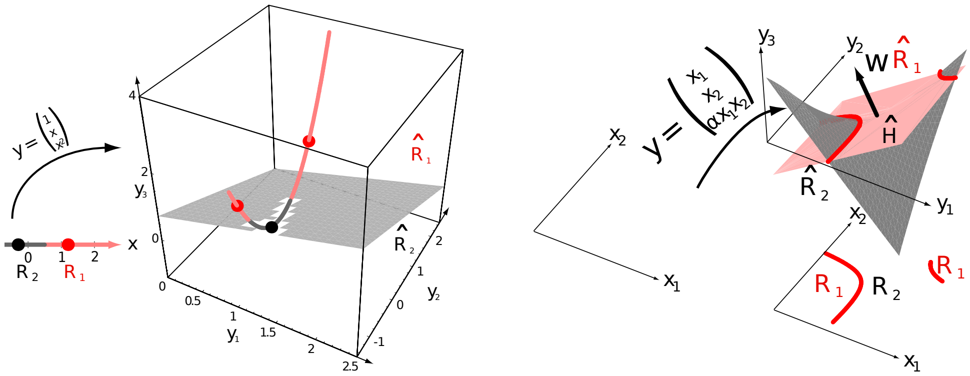
$$g(x) = w_0 + \sum_{i \leq d} w_i x_i = \overbrace{(w_0, w)}^{\text{generalized coords a}} \underbrace{(1, x)^T}_{\text{generalized coords } \tilde{x}} = a^T \tilde{x}$$

Quadratic Discriminant Functions:

$$g(x) = w_0 + \sum_{i \leq d} w_i x_i + \sum_{i \leq d} \sum_{j \leq d} w_{ij} x_i x_j$$

How to learn nonlinear discriminant functions?

Transform the features non-linearly and use the linear classifier in a high dimensional feature space as usual.



Advantage of Homogeneous Coordinates

$$\tilde{x} = (1, x)^T \quad a = (w_0, w)^T$$

Easy transform, but makes analysis much easier.

1. Class 1 if $a^T \tilde{x}_i > 0$
2. Class 2 if $a^T \tilde{x}_i < 0$

Normalization transform $\tilde{x}_i \to -\tilde{x}_i$ iff the object belongs to class 2.

Why would you do that?Maybe to let data from class 2 get properties of class 1...

Linearly separable two class case

Linear separability

$$\exists a \text{ with } \begin{cases} a^T \tilde{x}_i > 0 & \text{for } y_i = 1 \\ a^T \tilde{x}_i < 0 & \text{for } y_i = 2 \end{cases}$$

- Problem: The solution vector is not unique.
- Idea: Introduce a margin b to classify data with a "safe" distance from the decision boundary, i.e. $a^T \tilde{x}_i \geq b > 0 \ \to$ regularization of the classifier!

Why is the solution vector not unique? Because it is more of a region?Then the idea does not make sense.

Gradient descent

$J(a(k))$ is a general cost function for weight vector a. It measures how well a hyperplane orthogonal to a classifies the data. $\eta(k)$ denotes the step size or learning rate at iteration k.

1. init $a, \varepsilon, \eta(\cdot), k = 0$
2. **repeat**
3. $\quad a = a + \eta(k)\nabla J(a)$
4. $\quad k = k + 1$
5. **until** $|\eta(k)\nabla J(a)| < \varepsilon$

Best learning rate is $\eta^{opt} = \frac{||\nabla J||^2}{\nabla J^T \frac{\partial^2 J}{\partial a_i \partial a_j} \nabla J}$
(2nd order taylor expansion of $J(a)$ at $a(k)$ then insert gradient descent rule $a(k+1) - a(k) = -\eta(k)\nabla J(a(k))$)

Newton's Algorithm

Optimality Condition: Choose $a(k+1)$ to minimize the 2^{nd} order Taylor expansion of $J(a(k+1))$, i.e.

$$\frac{\partial}{\partial a(k+1)} J(a(k+1)) = 0$$

$$\nabla J + \frac{\partial^2 J}{\partial a_i \partial a_j} (\underbrace{a(k+1) - a(k)}_{-\eta \nabla J}) = 0$$

$$\eta \nabla J = H^{-1} \nabla J$$

Newton's Descent Rule: $a(k+1) = a(k) - H^{-1}\nabla J$

1. init $a, \varepsilon, \eta(\cdot), k = 0$
2. **repeat**
3. $\quad a = a - \underline{H^{-1}\nabla J}$
4. $\quad k = k + 1$
5. **until** $|\eta(k)\nabla J(a)| < \varepsilon$

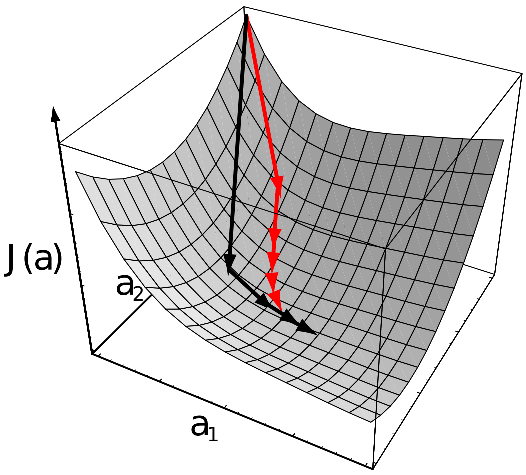


Figure 7: The sequence of weight vectors given by a simple gradient descent (red) and by Newton's (second order) gradient algorithm (black). Newton's rule leads to greater improvement per step, even when using optimal learning methods. However added computational burden of inverting the Hessian matrix.

7.2 Perceptrons

We need a cost function $J(a, \tilde{x_1}, \ldots, \tilde{x_n}$, that qualifies as a good cost function to update the weights and to solve the inequalities $a^T \tilde{x}_i > 0, \ \forall i$.

1. Number of misclassified samples is not a good choice as it is piecewise constant and without gradient.

2. Sum of violating projections

$$J_p(a) = \sum_{\tilde{x} \in \underbrace{\tilde{\mathcal{X}}}_{\text{set of missclassified samples}}} (-a^T \tilde{x})$$

   Perceptron Rule: $a(k+1) = a(k) + \eta(k) \sum_{\tilde{x} \in \tilde{\mathcal{X}}} \tilde{x}$

Batch version with variable learning rate:

1. init $a, \varepsilon, \eta(\cdot), k = 0$
2. **repeat**
3. $\quad a = a + \underline{\sum_{\tilde{x} \in \tilde{\mathcal{X}}} \eta(k) \tilde{x}}$
4. $\quad k = k + 1$
5. **until** $|\eta(k) \sum_{\tilde{x} \in \tilde{\mathcal{X}}}| < \varepsilon$

Fixed-Increment Single Sample Perceptron (moves the solution vector hyperplane towards the misclassified samples to make them switch sides):

1. init $a, \eta(\cdot), k = 0$
2. **repeat**
3. $\quad k = k + 1 \mod n$
4. $\quad$ **if** $\tilde{x}^k$ is misclassified by a **then**
5. $\qquad a = a + \underline{\tilde{x}^k}$
6. $\quad$ **end if**
7. **until** all patterns are correctly classified

Convergence of the perceptron rule

If the training samples are linearly separable, then the sequence of weight vectors $a = a + \tilde{x}^k$ will terminate at a solution vector.

Make proof easy to understand.

Limitations of Single-Layer Perceptron

A single-layer perceptron can not solve the XOR problem.

φ_1	φ_2	y
1	1	0
0	1	1
1	0	1
0	0	0

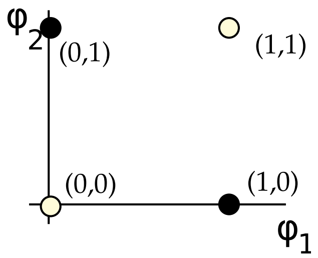


Figure 8: The XOR data set as shown here can not be separated in a 2 dimensional setting. Therefore, the single-layer perceptron can not achieve a proper separation. With a multilayer perceptron, we can get this data set into higher dimensions to make it properly separable.

Variable-Increment Perceptron with Margin

1. init $a, \varepsilon, \eta(\cdot), k = 0$
2. **repeat**
3. $k = k + 1$
4. **if** $a^T \tilde{x}^k \leq b$ **then**
5. $a = a + \underline{\eta(k)\tilde{x}^k}$
6. **end if**
7. **until** $a^T \tilde{x}^k > b, \forall k$

We do not need the thresholdε here!

We can formulate all perceptron algorithms as batch (all-data-at-once) or online (sequential data processing) algorithms. Online variance have a dependency on the sequence but tend to be more robust.

Why are they more robust?

7.3 WINNOW algorithm

Idea: Exponential update of weights
Consider a two class learning problem with many irrelevant dimensions (normalization is not performed).

1. a^+, a^- are weight vectors associated with either class and are corrected iff samples of their class are misclassfied.
2. α is the scaling factor for exponential update.
3. $z_k = \begin{cases} 1 & \text{if } \tilde{x_k} \text{ belongs to class 1} \\ -1 & \text{if } \tilde{x_k} \text{ belongs to class 2} \end{cases}$

Balanced WINNOW

1. init $a^+, a^-, \varepsilon, \eta(\cdot), k = 0, \alpha > 1$
2. **repeat**
3. $k = k + 1 \mod n$
4. **if** $\mathrm{sgn}[a^{+T}, a^{-T}\tilde{x_k}] \neq z_k$ **then** (pattern misclassified)
5. **if** $z_k = +1$ **then** (class 1 error)
6. **for all** $i \leq d$ **do**
7. $a_i^+ = a_i^+ \cdot \alpha^{+\tilde{x_{ki}}}$ (exponentiated update)
8. $a_i^- = a_i^- \cdot \alpha^{-\tilde{x_{ki}}}$
9. **end for**
10. **end if**
11. **if** $z_k = -1$ **then** (class 2 error)
12. **for all** $i \leq d$ **do**
13. $a_i^+ = a_i^+ \cdot \alpha^{-\tilde{x_{ki}}}$ (exponentiated update)

14. $\quad\quad a_i^- = a_i^- \cdot \alpha^{+\tilde{x_{ki}}}$
15. $\quad$ **end for**
16. $\quad$ **end if**
17. **end if**
18. **until** convergence

Minimum squared error procedures

Closed form solution LSE learning is learning with all $\tilde{x}_i$ in $a^T\tilde{x}_i = b_i > 0$ instead of training it with $\tilde{x}_i \in \tilde{\mathcal{X}}$ in $a^T\tilde{x}_i > 0 \;\forall i$ As we have the learning condition as $\tilde{X}a = b$, we can get the error criterion as $e = \tilde{X}a - b$. The squared error is then

$$J_s(a) = ||\tilde{X}a - b||^2 = \sum_{i \leq n}(a^T\tilde{x}_i - b_i)^2$$

from this we can calculate the gradient by deriving it:

$$\nabla J_s = 2\tilde{X}^T(\tilde{X}a - b) = 0 \rightarrow a = (\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T b = \underbrace{\tilde{X}^{\dagger}}_{\text{pseudo inverse of } \tilde{X}} b$$

Iterative solution

1. init a(1)
2. **repeat**
3. $\eta(k) = \frac{\eta(1)}{k}$
4. $a(k+1) = a(k) - \frac{\eta(k)}{2}\nabla J_s \quad (= a(k) - \eta(k)\tilde{X}^T(\tilde{X}a(k) - b))$
5. **until** convergence

LMS Rule

1. init $a, b, \theta, \eta(\cdot)$
2. k = 0
3. **repeat**
4. $k = k + 1 \mod n$
5. $\eta(k) = \frac{\eta(1)}{k}$
6. $a = a + \underline{\eta(k)(b_k a^T\tilde{x}^k)\tilde{x}^k}$
7. **until** $|\eta(k)(b_k - a^T\tilde{x}^k)\tilde{x}^k| < \theta$

7.4 Fisher's linear discriminant analysis

Idea: Project high-dimensional data of two classes to a one-dimensional linear subspace such that the classes are optimally separated.

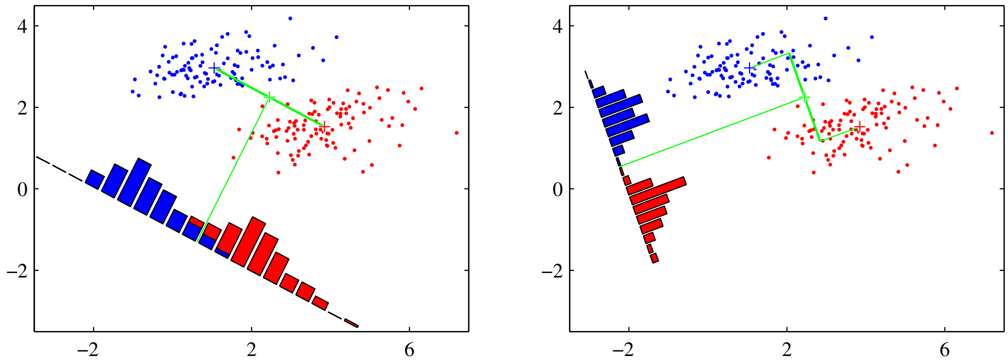


Figure 9: Projection on the left is suboptimal as it does not separate the classes properly, whereas the projection on the right gives rise to proper class separations.

We have samples $\mathcal{Z} = \{(x_1, y_1), \ldots, (x_n, y_n)\} =: \mathcal{X} \times \mathcal{Y}$ that we want to partition into k class specific subsets $\mathcal{Z}_{y_i} = \mathcal{X}_{y_i} \times \mathcal{Y}_{y_i}$, $1 \leq y_i \leq k$, $\forall x_i \in \mathcal{X}_{y_i}$ by a projection $x_i' = w^T x_i$, $1 \leq i \leq n$.

How to we measure the discrimination of the projected points?

Sample averages (class centroids):

$$m_{y_i} = \frac{1}{\underbrace{n_{y_i}}_{|\mathcal{X}_{y_i}|}} \sum_{x \in \mathcal{X}_{y_i}} x$$

Averages of projected points:

$$m_{y_i} = \frac{1}{\underbrace{n_{y_i}}_{|\mathcal{X}_{y_i}|}} \sum_{x \in \mathcal{X}_{y_i}} w^T x = w^T m_{y_i}$$

Distance of the sample averages in the two class case:

$$w^T(m_1 - m_2)$$

Scatter of class y_i:

$$\Sigma_{y_i} = \sum_{x \in \mathcal{X}_{y_i}} (x - m_{y_i})(x - m_{y_i})^T$$

"Within" scatter:

$$\Sigma_w = \Sigma_1 + \Sigma_2$$

Scatter of projected data(average of class specific variance)

$$\begin{aligned}\tilde{\Sigma}_{y_i} &= \sum_{x\in\mathcal{X}_{y_i}} (w^T x - \tilde{m}_{y_i})(w^T x - \tilde{m}_{y_i})^T \\ &= \sum_{x\in\mathcal{X}_{y_i}} w^T (x - m_{y_i})(x - m_{y_i})^T w \\ &= w^T \Sigma_{y_i} w \\ \text{Two classes: } &\rightarrow \tilde{\Sigma}_1 + \tilde{\Sigma}_1 = w^T \Sigma_w w\end{aligned}$$

Remark: To separate the data of different classes as good as possible, the class centroids in the projected space should be as far away as possible and, simultaneously, the variance in the projection space of the class specific data should be as small as possible.

Separation Criterion

Idea: Separate classes as good as possible relative to their variances (maximal J(w) = optimal w)

$$J(w) = \frac{\sigma^2_{between}}{\sigma^2_{within}} = \frac{||\tilde{m}_1 - \tilde{m}_2||^2}{\tilde{\Sigma}_1 + \tilde{\Sigma}_2} = \frac{w^T \overbrace{(m_1 - m_2)(m_1 - m_2)^T} = \Sigma_B w}{w^T \Sigma_W w}$$

So we differentiate $J(w)$:

$$\begin{aligned}\frac{d}{dw} J(w) &= \frac{d}{dw}((w^T \Sigma_W w)^{-1} w^T \Sigma_B w) \\ &= -2\Sigma_W w (w^T \Sigma_W w)^{-2} w^T \Sigma_B w + 2(w^T \Sigma_W w)^{-1} \Sigma_B w = 0 \\ \rightarrow \Sigma_B w &= \Sigma_w w \frac{w^T \Sigma_B w}{w^T \Sigma_W w}\end{aligned}$$

Solution via eigenvalue problem for weight vector

$$\Sigma_W^{-1} \underbrace{\Sigma_B w}_{\text{an unscaled projector making any vector parallel to}(m_1 - m_2)} = \lambda w, \ \lambda = \frac{w^T \Sigma_B w}{w^T \Sigma_W w}$$

Unscaled solution $(\mathcal{O}(d^2 n)$:

$$\tilde{w} = \Sigma_W^{-1}(m_1 - m_2)$$

Multiple Discriminant Analysis

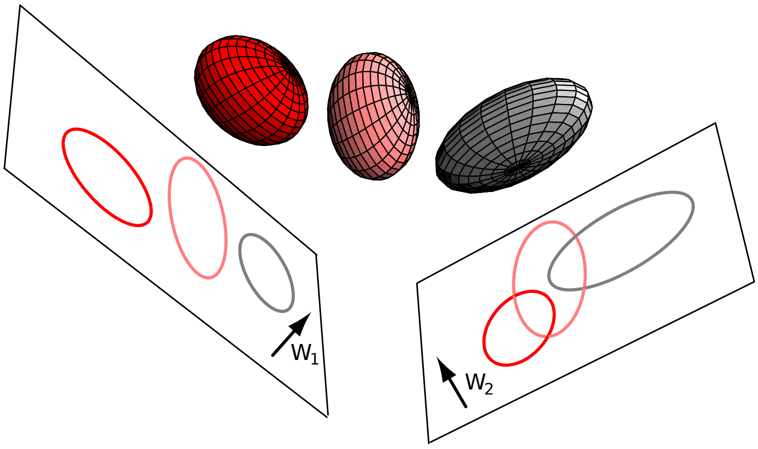


Figure 10: Informally, multiple discriminant methods seek the optimum such subspace, that is, the one with the greatest separation of the projected distributions for a given total within-scatter matrix.

k-Class Problem: Find a $k-1$ dimensional linear subspace, which separates the classes in an optimal fashion $(d \geq k)$.
Within-class scatter:

$$\Sigma_W = \sum_{y_i \leq k} \Sigma_{y_i} = \sum_{y_i \leq k} \sum_{x \in \mathcal{X}_{y_i}} (x - m_{y_i})(x - m_{y_i})^T$$

with class centroids:

$$m_{y_i} = \underbrace{\frac{1}{n_{y_i}}}_{|\mathcal{X}_{y_i}|} \sum_{x \in \mathcal{X}_{y_i}} x$$

Total mean Vector:

$$m = \frac{1}{n} \Sigma_x x = \frac{1}{n} \Sigma_{y_i} n_{y_i} m_{y_i}$$

What is n here?Number of objects in class?

Total Scatter Matrix:

$$
\begin{aligned}
\Sigma_T &= \frac{1}{n}\Sigma_x (x-m)(x-m)^T \\
&= \sum_{y_i} \sum_{x\in\mathcal{X}_{\dagger_\rangle}} (x - m_{y_i} + m_{y_i} - m)(x - m_{y_i} + m_{y_i} - m)^T \\
&= \sum_{y_i} \sum_{x\in\mathcal{X}_{\dagger_\rangle}} (x - m_{y_i})(x - m_{y_i})^T + \sum_{y_i} \sum_{x\in\mathcal{X}_{\dagger_\rangle}} (m_{y_i} - m)(m_{y_i} - m)^T \\
&= \Sigma_W + \sum_{y_i} n_{y_i}(m_{y_i} - m)(m_{y_i} - m)^T = \Sigma_W + \Sigma_B
\end{aligned}
$$

Projection from a d-dimensional feature space to a (k - 1)-dimensional subspace is achieved by (k - 1) discriminant functions $y_i = w_i^T x,\ 1 \leq i \leq k-1$

Fisher's Criterion

$$J(W) = \frac{|W^T \Sigma_B W|}{|W^T \Sigma_W W|}$$

1. Solve the generalized eigenvalue problem $\Sigma_B w_i = \lambda_i \Sigma_W w_i$ for the $(k-1)$ largest eigenvalues by Gram-Schmidt orthonormalization.

2. Σ_B is a sum of k rank one (or less) matrices; at most $(k-1)$ matrices are independent (center of mass constraint) $\rightarrow |\Sigma_B| \leq (k-1)$.

3. Isotropic case: space is spanned by the $(k-1)$ vectors $m_i - m$.

Remarks on Multicategory Linear Discriminants

It is often preferable to reformulate the multiclass problem as $(k-1)$ "class y_i - not class y_i" dichotomies or $k(k-1)/2$ "class α or β" dichotomies.
Problem: Some areas in feature space will be ambiguously classified.

7.5 Readings

1. Perceptrons: chp 5

2. Fisher's Linear Discriminant Analysis: 4.10

3. Both: chp 5.8.2

8 Support Vector Machines

This is the chapter on Support Vector Machines.

8.1 Lagrangian optimization theory

Overview An Optimization Problem is given by a function which we want to minimize and some constraints:

$$\begin{aligned} \textit{minimize } f(w) \\ \textit{subject to } g_i(w) \leq 0, \forall_i \\ \textit{and } h_j(w) = 0, \forall_j \end{aligned}$$

Equality constraints Equality constraints define sub-manifolds in the solution space on which the minimal solution has to be located, which is the same as saying that an equation $h(w) = 0$ is a (D-1)-dimensional surface in the solution space of w. At any point on the constraint surface, $\nabla h(w)$ will be orthogonal to the $h(w) = 0$ surface.
Proof: Consider a point w on the constraint surface and a nearby point $w + \varepsilon$ also on the surface. Making a Taylor Expansion around w, we get

$$h(w + \varepsilon) \simeq h(w) + \varepsilon^T \nabla h(w)$$

Because both points are on the constraint surface, we have $h(w) = h(w + \varepsilon) \Rightarrow \varepsilon^T \nabla h(w) \simeq 0$. In the limit $\parallel \varepsilon \parallel \rightarrow 0$ we have $\varepsilon^T \nabla h(w) = 0$ and because ε is parallel to the surface $h(w)$, $\nabla h(w)$ is normal to the surface.
Next we seek a point w^* so that $f(w)$ is minimized. Such a point must have the property that $\nabla f(w)$ is also orthogonal to the constraint surface, because otherwise we would increase the value of $f(w)$ by moving a short distance along along the surface. Thus ∇f and ∇h are parallel (or antiparallel) vectors and there's a solution for $\nabla f + \beta \nabla h = 0$ where β is a lagrange multiplier.
For equality constraints β may be positive as well as negative. We now introduce Lagrange functions, first for the simple case with one equality constraint:

$$L(w, \beta) = f(w) + \beta h(w)$$

The constrained stationary condition is obtained by setting $\nabla_w L = 0$. Furthermore, the condition $\frac{\partial L}{\partial \beta} = 0$ leads to the constrained equation $h(w) = 0$:

$$\frac{\partial L}{\partial \beta} = h(w) = 0$$

Thus, to find the minimum of $f(w)$ subject to $g(w) = 0$ we take $L(w, \beta)$ and find a stationary point of it with respect to w and β.
With a 2-D w we get three equations with three variables:

$$\begin{aligned} \frac{\partial L}{\partial w_1} = 0 \\ \frac{\partial L}{\partial w_2} = 0 \\ \frac{\partial L}{\partial \beta} = 0 \end{aligned}$$

Solve equations to find w^* and β. If only interested in w^* we can eliminate β from the equations and solve for w^* directly. In this case the lagrange multipliers are also called undetermined multipliers.

Inequality Constraints Now we extend the concept with inequality constraints in the form $g(w) \geq 0$. There are two kinds of solutions possible:

- With an inactive constraint $g(w) > 0$
- With an active constraint $g(w) = 0$

The inactive case corresponds to a solution which ignores the constraint:

$$L(w, \alpha) = f(w) + \alpha g(w) \ with \ \alpha = 0 \Rightarrow \nabla f(w) = 0$$

The active case corresponds to the case of the equality constraint with $\alpha \neq 0$. Now however the sign of the lagrange multiplier (i.e. α) matters because $f(w)$ will only be at its maximum if the gradient $\nabla f(w)$ is oriented away from the region $g(w) > 0$. We therefore have:

$$\nabla f(w) = -\alpha \nabla g(w) \ for \ \alpha > 0$$

In both cases, the active and inactive, the following holds: $\alpha g(w) = 0$
Thus, by optimizing

$$\begin{aligned} L(w, \alpha) = f(w) + \alpha g(w) \\ subject \ to \ g(w) \geq 0 \\ \alpha \geq 0 \\ \alpha g(w) = 0 \end{aligned}$$

we find our solution. The three constraints given here are known as the **Karush-Kuhn-Tucker (KKT) conditions**.

Dual Problem Finally we can come up with an equation containing several constraints of both types:

$$\begin{aligned} L(w, \alpha, \beta) = f(w) + \sum_i \alpha_i g_i(w) + \sum_j \beta_j h_j(w) \\ subject \ to \ \alpha_i \geq 0 \ \forall_i \\ \alpha_i g_i(w) = 0 \ \forall_i \end{aligned}$$

Steps:

1. Find w^* of $L(w, \alpha, \beta)$ by $\frac{\partial L}{\partial w}|_{w=w^*} = 0$
2. Insert w^* into L and find β^* by $\frac{\partial L}{\partial \beta}|_{\beta=\beta^*} = 0$
3. Calculate α by solving the above optimization problem with w^* and β^*

The **Dual Problem** given by

$$\begin{aligned} max \theta(\alpha, \beta) \\ subject \ to \ \alpha_i \geq 0 \ \forall_i \\ where \ \theta(\alpha, \beta) = inf_w L(w, \alpha, \beta) \end{aligned}$$

has an upper bound:

$$\theta(\alpha, \beta) = inf_u L(u, \alpha, \beta) \leq L(w, \alpha, \beta) = f(w) + \sum_i \alpha_i g_i(w) + \sum_j \beta_j h_j(w) \leq f(w)$$

Feasibility of w implies $g_i(w) \leq 0$ and $h_j(w) = 0$.
Feasibility of α and β implies $\alpha_i \geq 0$.
Therefore the first sum is negative or zero and the second one is zero.

Duality Gap The optimal solution of the Dual Problem is also called the **value of the problem**. The value of the dual problem is upper bounded by the value of the primal problem:

$$sup\{\theta(\alpha,\beta) : \alpha \geq 0\} \leq inf\{f(w) : g(w) \leq 0, h(w) = 0\}$$

The difference between the two is called the **duality gap**. The solution (w^*, α^*, β^*) is a saddle point of Lagrangian for the primal problem if and only if its components are optimal solutions of the primal and dual problem and if there's no duality gap.
Given an optimization problem with convex f and convex domain of $w \in \omega \subseteq \mathbb{R}$ and with g_i and h_j that are affine functions, i.e. $g(w) = Aw - b$ for some matrix A and some vector b, then the duality gap is zero. This applies to SVMs.

8.2 Hard margin SVMs

Lets start again with the two-class classification problem described earlier in the form

$$y = w^T\phi(x) + w_0$$

where ϕ is some transformation applied to the features. We assume for now that the training data $\mathcal{Z} = \{(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R} : 1 \geq i \geq n\}$ is linearly separable in feature space. Additionally to other similar algorithms (e.g. perceptron) the SVM minimizes the generalization error by maximizing the distance from the separating hyperplane to the closest points of both classes (i.e. margin).

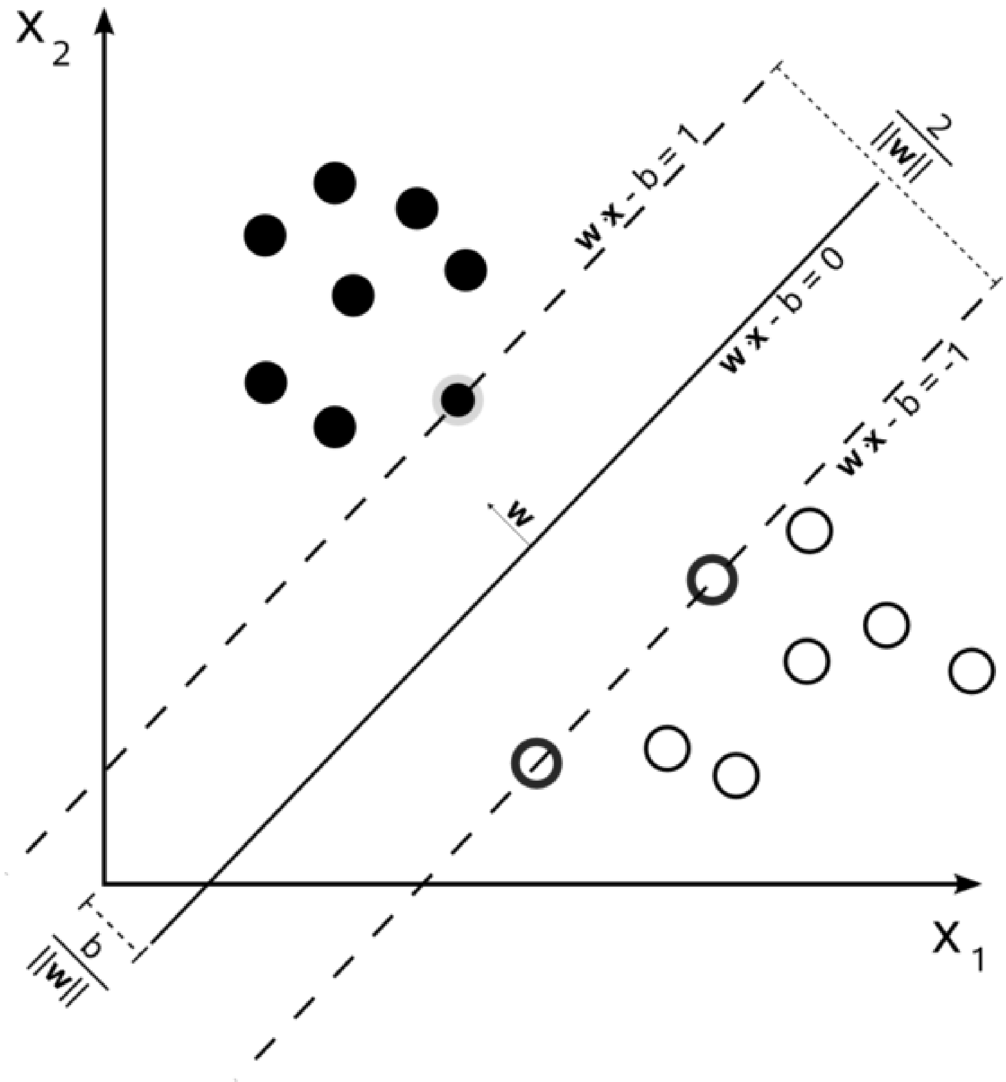


Figure 11: Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors. **b** stands for margin.

The Hyperplane The perpendicular distance of a point x from a hyperplane defined by $f(x) = 0$ where $f(x) = w^T\phi(x) + w_0$ is given by $\frac{|f(x)|}{\|w\|}$. Furthermore we're only interested in a solution where all points are correctly classified. Thus the distance of a point x_i to the decision surface becomes

$$\frac{y_i f(x_i)}{\| w \|} = \frac{y_i(w^T\phi(x_i) + w_0)}{\| w \|}$$

The maximum margin solution is given by

$$\underset{w,w_0}{\operatorname{argmax}}\{\frac{1}{\| w \|}\min_i[y_i(w^T\phi(x_i) + w_0)]\}$$

where we have taken factor $\frac{1}{\|w\|}$ outside of the optimization because w doesn't depend on i. Direct solution of this problem would be very complex, therefore we simplify it. We notice that if we rescale

$w \to \gamma w$ and $w_0 \to \gamma w_0$ then the distance from any x_i to decision surface given by $\frac{y_i f(x_i)}{\|w\|}$ doesn't change. Therefore

$$y_i(w^T\phi(x_i) + w_0) = 1$$

for the point that is closest to the surface. In this case all datapoints will satisfy

$$y_i(w^T\phi(x_i) + w_0) \geq 1$$

This is called the **canonical representation** of the decision hyperplane. In case of points for which the constraints holds, the constraints are called active. Otherwise they're inactive. The optimization problem therefore requires to maximize $\| w \|^{-1}$ which is equivalent to minimizing $\| w \|^2$. The optimization problem becomes:

$$\underset{w,w_0}{\operatorname{argmin}} \frac{1}{2} \| w \|^2$$
$$\textit{subject to } y_i(w^T\phi(x_i) + w_0) \geq 1 \ \forall_i$$

Lagrangian Now we reformulate this problem as a Lagrange problem. The condition needs to be turned into the form $g(w) \geq 0$ and we add multipliers:

$$L(w, w_0, \alpha) = \frac{1}{2} \| w \|^2 - \sum_i^N \alpha_i \{y_i(w^T\phi(x_i) + w_0) - 1\}$$

where α is a vector containing the lagrange multipliers for each training sample $i \in \{1, ..., N\}$. The sign in front of the sum means that we're minimizing with respect to w and w_0 and maximizing with respect to α. Setting the derivatives of $L(w, w_0, \alpha)$ with respect to w and w_0 equal to zero, we obtain the following two conditions:

$$w = \sum_i \alpha_i y_i \phi(x_i)$$
$$0 = \sum_i \alpha_i y_i$$

Dual Problem We can now eliminate w and w_0 from $L(w, w_0, \alpha)$ which gives us the dual representation of the maximum margin problem in which we maximize

$$L(w, w_0, \alpha) = \frac{1}{2} \| w \|^2 - \sum_i^N \alpha_i \{y_i(w^T\phi(x_i) + w_0) - 1\}$$
$$\tilde{L}(\alpha) = \frac{1}{2} \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j k(x_i, x_j) - \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j k(x_i, x_j) + \sum_i^N \alpha_i$$
$$\tilde{L}(\alpha) = \sum_i^N \alpha_i - \frac{1}{2} \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\textit{subject to constraints } \alpha \geq 0 \ \forall_i$$
$$\sum_i^N \alpha_i y_i = 0$$

Here the kernel function is defined by $k(x,x') = \phi(x)^T\phi(x')$. To evaluate new data points we use

$$f(x) = \sum_i^N \alpha_i y_i k(x, x_i) + w_0$$

Since usually only a small minority of data points are support vectors, most $\alpha_i = 0$. For the support vectors holds $y_i f(x_i) = 1$. Therefore the evaluation is computationally simple, because the non-support vectors can be discarded after training.

Evaluating Data Points To calculate w_0 we use the fact that for any support vector $y_i f(x_i) = 1$. Using $f(x) = \sum_i^N \alpha_i y_i k(x, x_i) + w_0$ we obtain:

$$y_i(\sum_{j\in\mathcal{S}} \alpha_j y_j k(x_i, x_j) + w_0) = 1$$

where $\mathcal{S}$ is the set of support vectors. Using $y_i^2 = 1$ we get

$$w_0 = \frac{1}{N_\mathcal{S}} \sum_{i\in\mathcal{S}} (y_i - \sum_{j\in\mathcal{S}} \alpha_j y_j k(x_i, x_j))$$

which is numerically more stable and where $N_\mathcal{S}$ is the total number of support vectors.

8.3 Soft margin SVMs

So far we assumed that the data points are linearly separable. Now we consider the case where the classes overlap, which is usually the case in practice. Therefore we introduce a method to allow some data points to be inside the margin or even on the wrong side of the separating hyperplane. To do this we use slack variables $\xi_i \geq 0$ for each data point which gives a measure of the misclassification of data point i.these are defined by $\xi_i = 0$ for data points on or inside the correct margin boundary and $\xi_i = |y_i - f(x_i)|$ for other points. Thus points on the separating hyperplane will have $\xi_i = 1$ and misclassified datapoints will have $\xi_i > 1$. The exact misclassification constraints are then replaced with

$$y_i f(x_i) \geq 1 - \xi_i \ \forall_i$$

These are called sometimes relaxation constraints and an SVM using these is known as a **Soft margin SVM**. Our goal is now tho maximize the margin while softly penalizing points lying on the wrong side of the margin boundary. We therefore optimize

$$C\sum_i^N \xi_i + \frac{1}{2} \parallel w \parallel^2$$

where $C > 0$ weights the impact of the slack penalties on the final result. We turn this into a Lagrangian:

$$L(w, w_0, \xi, \alpha, \beta) = \frac{1}{2} \parallel w \parallel^2 + C\sum_i^N \xi_i - \sum_i^N \xi_i\{y_i f(x_i) - 1 + \xi_i\} - \sum_i^N \beta_i \xi_i$$

where $\alpha_i \geq 0$ and $\beta_i \geq 0$ are Lagrange multipliers. The corresponding KKT conditions are given by

$$
\begin{aligned}
\alpha_i &\geq 0 \\
y_i f(x_i) - 1 + \xi_i &\geq 0 \\
\alpha_i (y_i f(x_i) - 1 + \xi_i) &= 0 \\
\beta_i &\geq 0 \\
\xi_i &\geq 0 \\
\beta_i \xi_i &= 0
\end{aligned}
$$

We now optimize out w, w_0 and ξ using $f(x) = w^T \phi(x) + w_0$

$$
\begin{aligned}
\frac{\partial L}{\partial w} = 0 &\Rightarrow w = \sum_i^N \alpha_i y_i \phi(x_i) \\
\frac{\partial L}{\partial b} = 0 &\Rightarrow \sum_i^N \alpha_i y_i = 0 \\
\frac{\partial L}{\partial \xi_i} = 0 &\Rightarrow \alpha_i = C - \beta_i
\end{aligned}
$$

using these to eliminate w, w_0 and ξ we obtain the dual Lagrangian in the form

$$L(w, w_0, \xi, \alpha, \beta) = \frac{1}{2} \parallel w \parallel^2 + C \sum_i^N \xi_i - \sum_i^N \alpha_i \{ y_i (w^T \phi(x_i) + w_0) - 1 + \xi_i \} - \sum_i^N \beta_i \xi_i$$

$$L(w, w_0, \xi, \alpha, \beta) = \frac{1}{2} \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j k(x_i, x_j) + C \sum_i^N \xi_i - \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j k(x_i, x_j) + \sum_i^N \alpha_i (1 - \xi_i) - \sum_i^N \beta_i \xi_i$$

$$L(w, w_0, \xi, \alpha, \beta) = \sum_i^N \alpha_i - \frac{1}{2} \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j k(x_i, x_j) + \sum_i^N (C - \alpha_i - \beta_i) \xi_i$$

$$
\begin{aligned}
ignoring : C - \alpha_i - \beta_i &= \frac{\partial L}{\partial \xi_i} = 0 \\
\tilde{L}(\alpha) &= \sum_i^N \alpha_i - \frac{1}{2} \sum_i^N \sum_j^N \alpha_i \alpha_j y_i y_j k(x_i, x_j)
\end{aligned}
$$

We notice that the Lagrangian looks the same as for the Hard Margin case. The difference lies in the constraints that are somewhat different. We note that $\alpha_i \geq 0$ is required because they're Lagrange multipliers. Furthermore $\alpha_i = C - \beta_i$ together with $\beta_i \geq 0$ implies that $\alpha \leq C$. We therefore maximize with respect to α subject to

$$
\begin{aligned}
0 \leq \alpha_i \leq C \\
\sum_i^N \alpha_i y_i = 0
\end{aligned}
$$

where the first kind of constraints are known as the box constraints. Prediction of new data points are made like in the Hard Margin SVM. As before a subset of the data points may have $\alpha_i = 0$ in which case they don't contribute to the predictions. The remaining data points constitute the support vectors.

These have $\alpha_i > 0$ and from $\alpha_i(y_i f(x_i) - 1 + \xi_i) = 0$ must satisfy

$$y_i f(x_i) = 1 - \xi_i$$

If $\alpha_i < C$ then $\alpha_i = C - \beta_i$ implies that $\beta_i > 0$, which according to $\beta_i \xi_i = 0$ requires $\xi_i = 0$ and hence such points lie on the margin. Points with $\alpha = C$ can lie inside the margin and can be either correctly classified if $\xi_i \leq 1$ or misclassified if $\xi_i > 1$. To determine w_0 we note that support vectors for which $0 < \alpha_i < C$ have $\xi_i = 0$ so that $y_i f(x_i) = 1$ and hence will satisfy

$$y_i(\sum_{j \in \mathcal{S}} \alpha_j y_j k(x_i, x_j) + w_0) = 1$$

like in the Hard Margin SVM case. Again, a numerically more stable solution is given by

$$w_0 = \frac{1}{N_{\mathcal{M}}} \sum_{i \in \mathcal{M}} (y_i - \sum_{j \in \mathcal{M}} \alpha_j y_j k(x_i, x_j))$$

where $\mathcal{M}$ is the set of indices of data points satisfying $0 < \alpha_i < C$.

8.4 Readings

1. SVM: Bishop, Chapter 7.1
2. Lagrange multipliers: Bishop, Appendix E
3. Slides from Lecture 8

9 Nonlinear Support Vector Machines

Many linear parametric models (including SVMs) can be re-cast into an equivalent 'dual representation' in which the predictions are also based on linear combinations of a kernel function evaluated at the training data points.

9.1 Kernels

For models which are based on a fixed nonlinear feature space mapping $\phi(x)$, the kernel function is given by the relation

$$k(x, y) = \langle \phi(x), \phi(y) \rangle$$

A kernel is a symmetric function of its arguments so that $k(x, y) = k(y, x)$. The simplest example of a kernel function is obtained for $\phi(x) = x$ (i.e. no transformation), in which case $k(x, y) = \langle x, y \rangle$. This may be called the linear kernel. The concept of a kernel formulated as an inner product in a feature space allows us to build interesting extensions of many well-known algorithms by making use of the **kernel trick**. The idea is that if the vector x enters only as a scalar product into the training algorithm, then we can replace the scalar product with some other kernel.

Kernelized SVM The optimization function of the SVM for linear features is given by

$$\tilde{L}(\alpha) = \sum_{i}^{N} \alpha_i - \frac{1}{2} \sum_{i}^{N} \sum_{j}^{N} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

where $k(x_i, x_j)$ replaces $\langle \phi(x_i), \phi(x_j) \rangle$.

Properties Every Kernel is the scalar product of two vectors in some n-dimensional space. Consider the following kernel, taking two 1-dimensional vectors (i.e. scalars) as arguments:

$$k(x,y) = \langle log(x), log(y) \rangle$$

What this does is, it transforms the inputs into log space and there calculates the scalar product. If you can transform a function taking two vectors as inputs into the following form $\langle \phi(x_i), \phi(x_j) \rangle$, you have proven that it is a kernel. **Kernels are symmetric** $\rightarrow k(x,y) = k(y,x)$.

9.2 Kernelized Linear Regression

Many linear models for regression and classification can be reformulated in a dual representation in which the feature vectors only enter as scalar products or as a **design matrix** (see below). In this case we can apply the kernel trick. Consider the following linear regression model

$$J(w) = \frac{1}{2}\sum_i^N (w^T\phi(x_i) - y_i)^2 + \frac{\lambda}{2} w^T w$$

where $\lambda \geq 0$. By setting the gradient with respect to w equal to zero, we see that the solution for w takes the form of linear combination of the vectors $\phi(x_i)$ with coefficients that are functions of w:

$$w = -\frac{1}{\lambda}\sum_i^N (w^T\phi(x_i) - y_i)\phi(x_i) = \sum_i^N a_i\phi(x_i) = \Phi^T a$$

where Φ is the design matrix, whose n^{th} row is given by $\phi(x_i)^T$. Here the vector $a = (a_1, ..., a_N)^T$, and we have defined

$$a_i = -\frac{1}{\lambda}(w^T\phi(x_i) - y_i)$$

Instead of working with parameter vector w, we can now reformulate the leastsquares algorithm in terms of the parameter vector a, giving rise to the **dual representation**. If we substitute $w = \Phi^T a$ into $J(w)$, we obtain

$$J(a) = \frac{1}{2}a^T\Phi\Phi^T\Phi\Phi^T a - a^T\Phi\Phi^T y + \frac{1}{2}t^T t + \frac{\lambda}{2}\Phi\Phi^T a$$

where $y = (y_1, ..., y_N)^T$. We now define the Gram matrix $G = \Phi\Phi^T$, which is an NxN symmetric matrix with elements

$$G_{ij} = \phi(x_i)^T\phi(x_j) = k(x_i, x_j)$$

where we have introduced the kernel function k(x, y). In terms of the Gram matrix G, the sum-of-squares error function can be written as

$$J(a) = \frac{1}{2}a^T KKa - a^T Ky + \frac{1}{2}y^T y + \frac{\lambda}{2}a^T Ka$$

Using $w = \Phi^T a$ to eliminate w from $a_i = -\frac{1}{\lambda}(w^T\phi(x_i) - y_i)$ and solving for a we obtain

$$a = (K + \lambda I_N)^{-1} y$$

Substituting this back into the linear regression model, we obtain the following prediction for a new input x

$$f(x) = w^T\phi(x) = a^T\Phi\phi(x) = k(x)^T(K + \lambda I_N)^{-1} y$$

where we have defined the vector $k(x)$ with elements $k_n(x) = k(x_n, x)$.

9.3 Readings

1. Add some readings here

10 Ensemble Methods for Classifier Design

Ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms. A machine learning ensemble refers only to a concrete finite set of alternative models, but typically allows for much more flexible structure to exist between those alternatives.
An ensemble is itself a supervised learning algorithm, because it can be trained and then used to make predictions. The trained ensemble, therefore, represents a single hypothesis. This hypothesis, however, is not necessarily contained within the hypothesis space of the models from which it is built. Thus, ensembles can be shown to have more flexibility in the functions they can represent. This flexibility can, in theory, enable them to over-fit the training data more than a single model would, but in practice, some ensemble techniques (especially bagging) tend to reduce problems related to over-fitting of the training data.

Empirically, ensembles tend to yield better results when there is a significant diversity among the models. Many ensemble methods, therefore, seek to promote diversity among the models they combine. Although perhaps non-intuitive, more random algorithms (like random decision trees) can be used to produce a stronger ensemble than very deliberate algorithms (like entropy-reducing decision trees). Using a variety of strong learning algorithms, however, has been shown to be more effective than using techniques that attempt to dumb-down the models in order to promote diversity. (Wikipedia)

10.1 Advantages

Computational: They are easy to train.

Statistical: Data randomization in the spirit of Bootstrap captures statistical dependencies between alternative hypotheses. $\rightarrow$ The classifier ensemble encodes uncertainty intervals in the hypothesis class (output space).

What does this mean?

10.2 Reminders

Bias Combining a set of estimators $\hat{f}_1(x), \hat{f}_2(x), \ldots, \hat{f}_B(x)$ by a simple average $\hat{f}(x) = \frac{1}{B}\sum_{i=1}^{B} \hat{f}_i(x)$ lets unbiased estimators remain unbiased.

Variance Combining a set of B estimators by a simple average reduces the variance by a factor of $\frac{1}{B}$ if bias remains unchanged.

Empirical error

$$\hat{\mathcal{R}}_n(c) = \frac{1}{n}\#\{c(x_i) \neq y_i : 1 \leq i \leq n\}$$

Expected error

$$\mathcal{R}(c) = P\{c(x) \neq Y\}$$

(ε, δ) **criterion**

$$P_{X,Y}\{\mathcal{R}(\hat{c}_n) \leq \mathcal{R}(c^{Bayes}) + \varepsilon\} > 1 - \delta$$

10.3 Empirical Risk Minimization Principle

Select classifier $c \in \mathcal{C}$ with the smallest error on the training data $\mathcal{Z} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$:

$$c_n^* \operatorname*{argmin}_{c \in \mathcal{C}} \hat{\mathcal{R}}_n(c) = \operatorname*{argmin}_{c \in \mathcal{C}} \frac{1}{n}\#\{c(x_i) \neq y_i :\ 1 \leq i \leq n\}$$

Goal: Derive a distribution independent bound for

$$P\{\mathcal{R}(\hat{c}_n^*) - \inf_{c \in \mathcal{C}} \underbrace{\mathcal{R}}_{=P(c(X)) \neq Y}(c) > \varepsilon\} < \delta$$

10.4 Probably Approximately Correct (PAC) Learning

Strong and Weak Learning

Assumption: Restricted classifier $c \in \mathcal{C}$ (hypothesis class). Classification Error for empirically determined classifier $\hat{c}$

$$P\left\{\mathcal{R}(\hat{c}) - \inf_{c \in \mathcal{C}} \mathcal{R}(c) > \varepsilon\right\} < \delta(*)$$

Strong PAC Learning requires to achieve an arbitrarily small error ε with high probability $1 - \delta$.

Weak PAC Learning demands that $(*)$ holds for 'large', but 'better than chance' ε. Weak learning has to achieve a non trivial error rate, i.e., the rate has to be bounded away from the chance ε-value.

10.5 Motivation for Ensemble Methods

Train several sufficiently diverse predictors, i.e., classifiers or regressors and use an aggregation scheme to produce a valid solution with low bias and variance.

Weak Learners used for bagging or boosting

- **Stumps**: Single axis parallel partition of space
- **Decision trees**: Hierarchical partition of space
- **Multi-layer perceptrons**: General non-linear function approximators
- **Radial basis functions**: Non-linear expansions based on kernels

Combining classifiers

Input: A pool of binary classifiers $c_1(x), c_2(x), \ldots, c_B(x)$ Objective: Infer a composite classifier with weights $\alpha_b{}_{b=1}^{B}$

$$\hat{c}_B(x) = \mathrm{sgn}\left(\sum_{b=1}^{B} \alpha_b c_b(x)\right)$$

Question: When can this procedure succeed? It works by adding enough diversity among the classifiers

Diversity:

- Use different subsets of the data for each c_b
- Use different features
- Decorrelate classifiers during training

10.6 Bagging

Bagging or bootstrap aggregation combines several classifiers which have been trained on random subsamples (with replacements) of the data. (Breiman, 1996)

Idea: Generate diversity in classifier pool by training on different, e.g. bootstrapped subsets!
Bootstrap sample: Given $Z = (x_1, y_1), \ldots, (x_n, y_n)$ generate Z by choosing i.i.d. pairs with replacement.

INPUT: data $(x_1, y_1), \ldots, (x_n, y_n)$ OUTPUT: classifier with majority vote of $\{c_1, c_2, \ldots, c_B\}$

1. **for** b = 1 to B **do**
2. $pl\mathcal{Z}^{*b}$ = b-th bootstrap sample from $\mathcal{Z}$
3. Construct classifier c_b based on $\mathcal{Z}^{*b}$
4. **end for**

5. **return** ensemble classifier $\hat{c}_B(x) = \operatorname{sgn}\left(\sum_{b=1}^{B} c_b(x)\right)$

Why does Bagging work?

Covariances small: Due to using different subsets for training

Variances similar: Estimator from each sub-sample behaves similarly (on average)

Biases: Weakly affected

More elaborate explanation - Bühlmann and Yu 1999

Takeaway Message: It is advantageous to reduce dependence between component estimators

10.7 Decision Trees

Decision tree learning is the construction of a decision tree from class-labeled training tuples. A decision tree is a flow-chart-like structure, where each internal (non-leaf) node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf (or terminal) node holds a class label. The topmost node in a tree is the root node.

There are many specific decision-tree algorithms. Notable ones include:

- **ID3** (Iterative Dichotomiser 3)
- **C4.5** (successor of ID3)
- **CART** (Classification And Regression Tree)
- **CHAID** (CHi-squared Automatic Interaction Detector). Performs multi-level splits when computing classification trees. **MARS:** extends decision trees to handle numerical data better.

The term Classification And Regression Tree (CART) analysis is an umbrella term used to refer to both of the following procedures:

- **Classification tree analysis** is when the predicted outcome is the class to which the data belongs.
- **Regression tree analysis** is when the predicted outcome can be considered a real number (e.g. the price of a house, or a patient's length of stay in a hospital).

Trees used for regression and trees used for classification have some similarities - but also some differences, such as the procedure used to determine where to split.

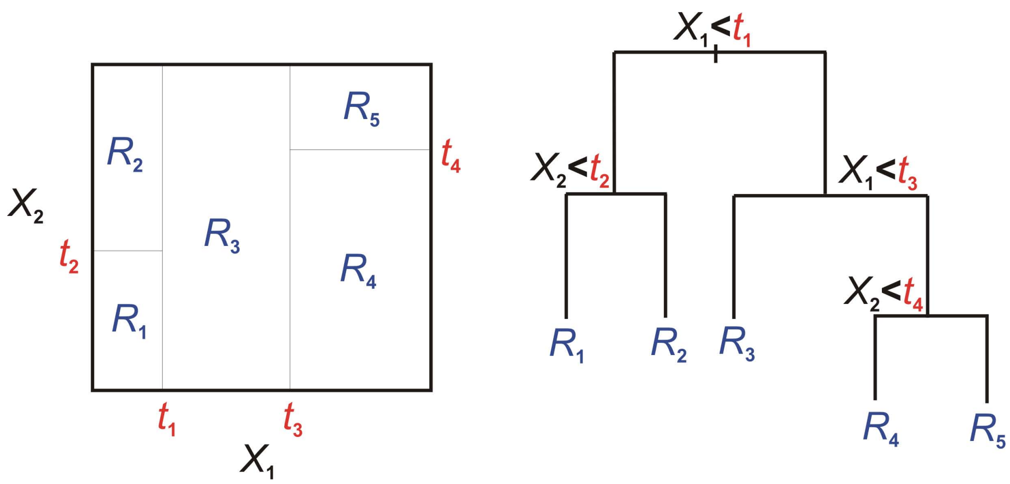


Figure 12: The data space is recursively partitioned into dichotomies. Example for $(X_1, X_2) \in \mathbb{R}^2$ with partitions R_i and thresholds t_j

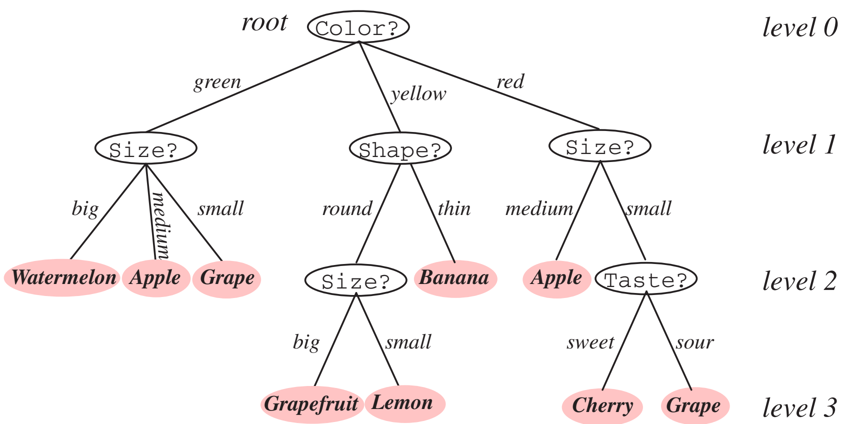


Figure 13: Classification of fruits

10.8 Random Forests

Random forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random forests correct for decision trees' habit of overfitting to their training set.

The algorithm for inducing a random forest combines the "bagging" idea and the random selection of features in order to construct a collection of decision trees with controlled variance.

Strategy for bagging trees Grow a sufficiently deep decision tree to reduce bias $\rightarrow$ noisy classifier $\rightarrow$ average to enhance robustness.

Classification of a new sample Let $\hat{c}_b(x)$ be the class prediction of the b^{th} random forest tree. Then, the random forest classification yields

$$\hat{c}_{rf}^B(x) = \text{majority vote}\{\hat{c}_b(x)\}_{b=1}^B$$

1. **for** b = 1 to B **do**
2. draw a bootstrap sample $\mathcal{Z}^* = \{(X_1^*, Y_1^*, \ldots, (X_n^*, Y_n^*)\}$ of size n from the training data $\mathcal{Z}$;
3. //generatation of a random forest tree
4. grow a random forest tree $\hat{c}_b(x)$ to the bootstrapped data, by recursively repeating
5. // the following steps for each leaf until the minimum node size n_min is reached.
6. **repeat**
7. select m variables at random from p variables
8. $(m \approx \sqrt{p})$
9. pick the best variable/split-point among the m selected variables
10. split the node into two daughter nodes
11. **until** node size n_min is reached
12. **end for**
13. **return** Output the ensemble of trees $\{\hat{c_b}(x)\}_{b=1}^B$

10.9 Boosting

Boosting is an approach to machine learning based on the idea of creating a highly accurate, excellent, complex classifier by making an adaptive combination of many relatively weak and inaccurate rules with sufficient diversity. (R. Shapire, Explaining AdaBoost. Springer, 2013.) It is built from the base class of simple classifiers such as perceptrons, decision stumps, axis parallel splits etc.

Output classifier:

$$\hat{c}_B(x) = \operatorname{sgn}\left(\sum_{b=1}^{B} \alpha_b c_b(x)\right)$$

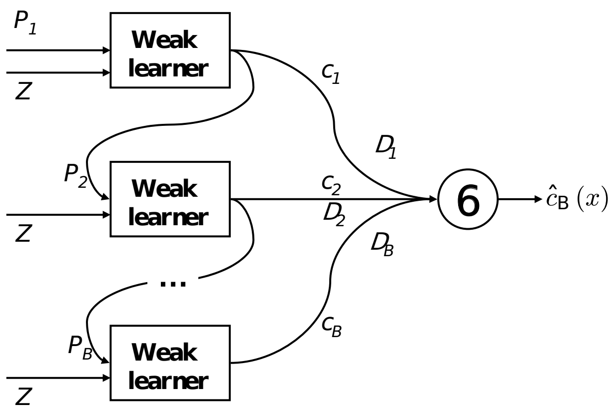


Figure 14: Boosting in form of AdaBoost is an ensemble method with error sensitive reweighting of the classifiers (Freund & Shapire, 1995).

Weak Learner:

$$\varepsilon_b \triangleq \frac{\sum_{i=1}^{n} w_i^{(b)} \mathbb{I}_{\{\underbrace{c_b(x_i)}_{\text{binary}} \neq y_i\}}}{\sum_{i=1}^{n} w_i^{(b)}}$$

Adaboost

AdaBoost fits an additive model in base learners,optimizing the exponential loss function, i.e., AdaBoost.M1 is equivalent to forward stagewise additive modeling using exponential loss.

AdaBoost.M1 Input: data $(x_1, y_1), \dots, (x_n, y_n)$ Output: classifier $\hat{c}_B(x)$

1. Initialize observation weights $w_i = \frac{1}{n}$
2. **for** b = 1 to B **do**
3. Fit a classifier $c_b(x)$ to the training data using weights w_i

4. Compute $\varepsilon_b = \frac{\sum_{i=1}^{n} w_i^{(b)} \mathbb{I}_{\{c_b(x_i) \neq y_i\}}}{\sum_{i=1}^{n} w_i^{(b)}}$

5. Compute $\alpha_b = \log \frac{1-\varepsilon_b}{\varepsilon_b}$

6. // Data Reweighting

7. // Apply weights $w_1^{(b)}, w_2^{(b)}, \ldots, w_n^{(b)}$ to each of the training data $(x_i, y_i), 1 \leq i \leq n$ at b^{th} step.

8. Set $\forall i :\ w_i = w_i \exp(\alpha_b \mathbb{I}_{\{y_i \neq c_b(x_i)\}})$

9. **end for**

10. **return** $\hat{c}_B(x) = sgn(\sum_{b=1}^{B} \alpha_b c_b(x))$

10.10 Arcing

Arcing or adaptive reweighting and combining is an extension of bagging introduced by Breiman (1998)

10.11 Exponential Loss

11 Unsupervised Learning

11.1 Nonparametric Density Estimation

Difference between parametric and nonparametric density estimation: Parametric approach assumes a parametric model distribution and estimates the parameters from the sample (e.g. maximum likelihood). The nonparametric approach, however, estimates the underlying density from a data sample without assuming a parametric model distribution.
Common approaches are

1. Histograms
2. Window estimates
3. Nearest neighbor estimates

Dirac's delta function

Not really a function: In mathematical terms, δ is not a function but a generalized function. Nonetheless, it is commonly referred to as the "delta function"

Definition 1
We introduce a mathematical "object" $\delta(x)$ satisfying the condition

$$\int_{\mathbb{R}} \delta(x) f(x) dx = f(0)$$

for any integrable function f on $\mathbb{R}$, with the following properties:

$$\begin{array}{lll} \int_{\mathbb{R}} \delta(x-x_0)f(x)dx & = bf(x_0) & \text{shifting} \\ \int_{\mathbb{R}} \delta(-x)f(x)dx & = f(0) & \text{symmetry} \\ \int_{\mathbb{R}} \delta(cx)f(x)dx & = \frac{1}{|c|}f(0) & \text{scaling} \end{array}$$

Normalization: The delta function integrates to 1

$$\int_{\mathbb{R}} \delta(x)dx = \int_{\mathbb{R}} \delta(x) \cdot c_1(x)dx = c_1(0) = 1$$

where c1 (x) is the constant function of value 1. Thus, we may regard $\delta(x-x_0)$ as a density with all its probability mass centered at a single point x_0

Definition 2

Let N denote the Gaussian density function.

$$\delta(x-x_0) = \lim_{\sigma \to +\infty} \mathcal{N}(x|x_0, \sigma)$$

Empirical distribution

The empirical distribution is the data sample regarded as a probability distribution. Each sample point is equally probable. Discrete case: Let $\mathcal{S} = x_1, ..., x_n$ be a data sample. We call

$$\mathsf{P}_{\mathsf{emp}}(x) := \frac{1}{n} \cdot \#\{y \in \mathcal{S} | y = y\}$$

the empirical distribution defined by the data.

Continuous case: More difficult: We need to combine probability mass centered at a finite set of points with the notion of a density.

Empirical density

Using the delta function we can write the empirical density of a sample $x_1, ..., x_n$ as

$$p_{\mathsf{emp}}(x) = \sum_{i=1}^{n} \frac{1}{n} \cdot \delta(x-x_i)$$

Expectation value with empirical density

$$\begin{aligned} \mathbb{E}_{p_{emp}(x)}\{f\} &= \int_{\Omega} f(x) p_{\mathsf{emp}}(x)dx \\ &= \frac{1}{n} \sum_{i=1}^{n} \int_{\Omega} \delta(x-x_i)f(x)dx = \frac{1}{n} \sum_{i=1}^{n} f(x_i) \end{aligned}$$

11.2 Histograms

A histogram represents data by counting the number of data points on each of a set of subintervals.

Definition

Let $\mathcal{S} = x_1, ..., x_n$ be a data sample on an interval I.

Divide the interval into k pairwise disjoint subintervals I_j , called the bins. The corresponding histogram of the sample is the tuple $H = (H_1, ..., H_k)$, with $H_j := \#\{x \in \mathcal{S} | x \in I_j\}$

Bin Size

If the number of bins is too large with respect to the sample size, the quality of the estimate deteriorates. For a large sample, choosing a small number of bins results in a loss of information.
Choosing the bin size: Common strategies are

1. Equidistant binning
   All bins have identical size $\frac{|I|}{k}$

2. Adaptive binning
   Use smaller bins in regions with higher data density

Histogram as non-parametric density estimator

A histogram can be turned into a discrete distribution by normalization

$$\hat{H} := \frac{1}{n}(h_1, ..., h_k)$$

We have an empirical density $p_{\text{emp}}(x)$. This is zero for every x that does not exactly coincide with a sample point (zero probability event!).
To turn $p_{\text{emp}}(x)$ into a more useful density, we have to regularize (smooth) it in some way. The histogram regularizes by binning. Choosing the bin size determines the level of regularization.

Alternative approach: The signal can be smoothed by applying a low-pass filter. Low-pass filter density estimators are called **window estimates**.

Multiple dimensions

Given a d-dimensional sample $x_1, ..., x_n$, we have two possible strategies:

Joint histogram
Compute the histogram on the complete domain. The bins are d-dimensional volume elements instead of intervals. A joint histogram corresponds to the underlying joint distribution. Problem: Requires large sample number.

Marginal histogram:
Compute d one-dimensional histograms. Histogram i is computed from the i-th component of the data vectors x_j . A marginal histogram corresponds to a marginal of the underlying distribution. Problem: Co-occurrence information is lost.

Engineer rule-of-thumb:
Reliable estimation requires at least ten sample values per parameter to be estimated.

Curse of dimensionality

Fundamental problem
To obtain a reliable estimate at a given regularity, the required number of samples grows exponentially with the dimension of the sample space.

Example: Joint histogram
Generalize a 20 bin histogram to multiple dimensions. Assumption: Good estimate requires ten sample values per bin.

- Number of possible bins (to ensure acceptable estimate) grows linearly with sample size ($\approx \frac{n}{10}$)
- Number of required bins (to keep the resolution) increases exponentially with dimension ($\sim 20^d$)

11.3 Parzen Estimators

The term "Parzen window estimate" is often used synonymously for "window density estimate".

Window Function

A **window function** is a function which vanishes or quickly decays outside a small interval/volume. Let V_n be the Volume of the d-dimensional hypercube of edge length h_n .

Box window
$\Phi = \begin{cases} 1 & |x_j| \leq \frac{1}{2},\ j = 1, ..., d \\ 0 & \text{else} \end{cases}$
 # Samples in a hypercube

$$k_n = \sum_{i=1}^{n} \underbrace{\Phi\left(\frac{x - x_i}{h_n}\right)}_{\text{Indicator function for } x_i \text{is in the hypercube around } x}$$

In principle, Φ might be any continuous function that vanishes outside a hypercube. To ensure that p_n is a density, we have to require (for $V_n = h_n^d$):

$$\Phi(x) \geq 0$$

$$\int \Phi(x) dx = 1$$

Parzen window density estimate

$$\hat{p}_n(x) := \frac{1}{n} \sum_{i=1}^{n} \frac{1}{V_n} \Phi\left(\frac{x - x_i}{h_n}\right)$$

Convergence

Mean and variance

- $\hat{p}_n(x)$ depends on sample $x_1, ..., x_n$, so $\hat{p}_n(x)$ is a RV
- By taking expectations w. r. t. $x_1, ..., x_n$, we can compute a mean $\bar{p}_n(x)$ and variance $\bar{\sigma}_n^2(x)$ of the estimate

Convergence result
For an "acceptable" estimate, we expect:

$$\lim_{n\to\infty} \bar{p}_n(x) = p(x)$$
$$\lim_{n\to\infty} \bar{\sigma}_n^2(x) = 0$$

These limits hold, provided that the window function $\Phi(x)$ and the volume sequence satisfy convergence conditions.

Convergence conditions

- Exclude pathological cases Φ

$$\sup_u \Phi(u) < \infty$$
$$\lim_{||u||\to\infty} \Phi(u) \prod_{i\leq d} u_i = 0$$

- Ensure convergence of mean and variance

$$\lim_{n\to\infty} V_n = 0$$
$$\lim_{n\to\infty} nV_n = \infty$$

These are satisfied e. g. for $V_n = \frac{V_1}{\sqrt{n}}$ but not for $V_n = \frac{V_1}{n}$.

Window estimates as convolutions

Convolution of empirical density and window function

$$\begin{aligned}\left(p_{\text{emp}}\frac{1}{V_n}\Phi\right)(x) &= \int \frac{1}{n}\sum_i \delta(y-x_i)\frac{1}{V_n}\Phi\left(\frac{x-y}{h_n}\right)dy \\ &= \frac{1}{n}\sum_i \frac{1}{V_n}\int \delta(y-x_i)\Phi\left(\frac{x-y}{h_n}\right)dy \\ &= \hat{p}_n(x)\end{aligned}$$

which is exactly the window estimate of the density.

Interpretation
In the case of a Gaussian shaped window function, this is standard low-pass filtering.

Computational importance
Convolutions are products in the Fourier domain, and can be efficiently evaluated by the FFT. We do not actually have to compute the integral.

Terminology
In applied mathematics, functions applied by convolution are often referred to as kernel functions. Therefore, window density estimators are also called kernel density estimators. This estimation technique is not the same type of kernel approach as used e. g. with SVMs, where the kernel represents a scalar product in a high-dimensional implicit feature space.

Popular kernels

This estimation technique is not the same type of kernel approach as used e. g. with SVMs!
The choice of the kernel can be influenced by a number of factors or requirements, e.g., existence of higher order derivatives (Gaussian k.), compactness of the kernel (box k., Epanechnikov k.), smoothness, heavy tails (Cauchy k.)...
Application knowledge often serves as a guidance to make an informed choice.

Uniform (Box) kernel	$\Phi(x) = \mathbb{I}_{\{\lvert\lvert x\rvert\rvert \leq \frac{1}{2}\}}$
Gaussian kernel	$\Phi(x) = exp\left(-\frac{1}{2}\lvert\lvert x\rvert\rvert^2\right) / \sqrt{2\pi}$
Exponential kernel	$\Phi(x) = exp\left(-\lambda\lvert\lvert x\rvert\rvert\right) / \lambda$
Cauchy kernel	$\Phi(x) = \frac{1}{1+\lvert\lvert x\rvert\rvert^{d+1}}$
Epanechnikov kernel	$\Phi(x) = (1 - \lvert\lvert x\rvert\rvert^2)\mathbb{I}_{\{\lvert\lvert x\rvert\rvert \leq 1\}}$

Parzen window vs. parametric estimates

Observation: A Gaussian window estimate places a Gaussian at the location of each sample point and normalizes. Each Gaussian is a parametric model.

The window estimate is non-parametric: The definition of a parametric model requires the number of parameters to be constant (w. r. t. the sample size). The number of Gaussians (and parameters) used in the window estimate is proportional to the sample size.

Density estimation and classification

- Induced classifier
  Any density estimator $\hat{p}(x)$induces a classification rule: From the training data, estimate the density $\hat{p}_c$ of each class c, then assign a test value x to the class for which $\hat{p}_c(x)$ is largest
- Parzen classifier
  The classifier induced by the Parzen window estimate is called the Parzen window classifier in the literature.

11.4 k-Nearest Neighbor Estimator

No training
Classification results are computed directly from the training data for each test value.

Problems The complexity of computing a classification result increases with ...

1. ... the dimension of the feature space (because we have to compute norms)
2. ... The sample size, because we have to compute differences w. r. t. all sample points and minimize over them. This is a big disadvantage in comparison to other classifiers

Why it is interesting anyway
k-NN rules, especially 3-NN, are known to yield excellent results. Comparison with 3-NN is a good test for newly designed classifiers.

k-Nearest Neighbor vs. Parzen

Parzen estimates: Different strategies to choose volume sequence
$V_n = \frac{V_1}{\sqrt{n}}$, $V_n = \frac{V_1}{logn}$ etc.

Problems with Parzen window estimates:

1. If V_0 is too small: Insufficient smoothing $\rightarrow$ noisy estimate
   If V_0 is too large: Oversmoothing $\rightarrow$ loss of detail.
2. Different behavior of the data distribution may require different strategies in different parts of the feature space, but V_n is data independent.

k Nearest Neighbor estimation

Solution: Choose volume data dependent, i.e. V_n grows until a pre-specified number k of samples is included. This strategy yields small cells for large density and large cells for low density.

k-NN density estimation
The k nearest neighbor density estimator is defined as

$$\hat{p}(x) := \frac{1}{V_k(x)}$$

where $V_k(x)$ = minimal volume around x containing k neighbors.

How do we choose the volume? Two choices:

- Let volume "grow" until it contains k neighbors. $\rightarrow$ This means we have to specify a metric and, thereby, we have implicitly selected a shape for the volume (hypercube, ball etc)
- For each data point x, select the cell of points for which x is the closest data point. (In 2D, this is a Voronoi tesselation.) For k-NN, average the volumes of the cell around x and its k nearest neighbors

k-NN classification

The induced classifier

- Training data $x_1, ..., x_n$, test point x_{n+1}
- Choose the k training points closest to x_{n+1}
- Classify x_{n+1} by majority vote

Ties Two different types of ties may occur

1. Distance ties: Several training data points have exactly the same distance to x_{n+1}
2. Voting ties. (Avoidable in the two-class case by choosing k odd.)

Tie-breaking strategies are necessary in practice, but difficult to handle theoretically.

Terminology

- k-NN rule usually refers to the classifier, rather than the density estimator
- nearest neighbor rule refers to the 1-NN classifier

1-NN

1-NN error rate

Error rate of a classifier = probability of misclassification
Theorem: Consider a classification problem with C classes and training data $x_1, ..., x_n . Let P^*$ denote the error rate of the Bayes rule. The expected error rate of the nearest neighbor rule

1. converges in L_1 for $n \to \infty$ to a value P_1
2. is bounded asymptotically by $P_1 \leq P^* \left(2 - \frac{C}{C-1} P^*\right)$

Remarks

- The expectation is taken with respect to the class labels
- Note that the result is independent of any distributional assumptions. The price we pay for this independence is that the result holds only for asymptotically large samples
- Convergence in L_1 means: The expected error rate is a function (of the training data). It converges in L_1 against the constant function of value P_1 , i. e. the L_1 norm of their difference converges to zero as $n \to \infty$
- Since the Bayes error rate P^* is optimal and $P^* \left(2 - \frac{C}{C-1} P^*\right) = 2P^* - \frac{C}{C-1} (P^*)^2 \leq 2P^*$ the 1-NN error rate for any number of classes is always $P^* \leq P \leq 2P^*$

12 Neural Networks

This is the chapter on Neural Networks.

12.1 Motivation by Computational Neuroscience

Artificial neural networks are inspired by biology. In the human brain we have $10^1 1$ neurons with on average 10^4 synapses, hence a extremely complex computational network. The network is made of over 1000 different cell types. Neurons can act onto each other either in an excitatory or inhibitory fashion.

12.2 Multilayer Perceptrons and Backpropagation

Introdu count-ing/in-dexing prob-lem

Training

MLP training consists of these steps

1. Feed-forward input data to compute output
2. Compute misclassification error
3. Backpropagate error signal and alter weights

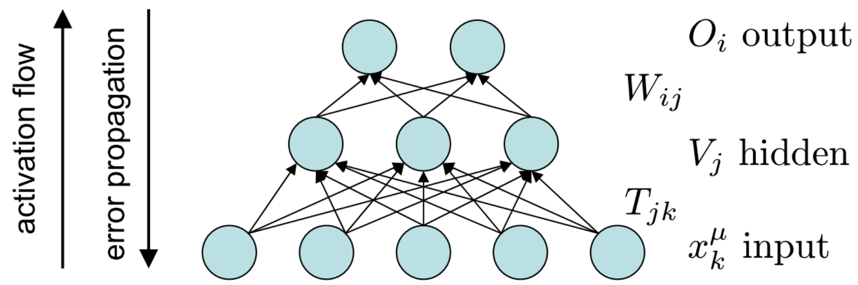


Figure 15: Two layer MLP (input is not counted in modern nomenclature)

Feed Forward

The data is fed into layer 0 (also called input layer). The sample (with index μ) has the form:

$$(x^\mu, y^\mu), 1 \leq \mu \leq n$$

At each node V_j of the first Layer, all the inputs times the weight are summed up, giving us the activation of that node:

$$h_j^\mu = \sum_k T_{jk} x_k^\mu$$

Where k is the index of the input/feature This activation is then transformed using a nonlinear activation function to give

$$V_j^\mu = S(h_j^\mu) = S(\sum_k T_{jk} x_k^\mu)$$

which will act as input for the next layer O. This layer, which in this example is also the output layer, performs again a summation followed by a transformation

$$O_i^\mu = S(\sum_j W_{ij} V_j^\mu) = S(\sum_j W_{ij} S(\sum_k T_{jk} x_k^\mu))$$

Where i is the index of the output node

Error function

$$J(W,T) = \frac{1}{2}\sum_{i,\mu}(y_i^\mu - S(\sum_j W_{ij} S(\sum_k T_{jk} x_k^\mu)))^2 = \frac{1}{2}\sum_{i,\mu}(y_i^\mu - O_i^\mu)^2$$

Backpropagation

There is no way to directly alter weights from all layers based on classification error. We have to propagate the error signal back from the output to the input layer. We use the chain rule to "assign the credit" (blame) for a particular error to specific weights. We start with the connections from the hidden layer to the output, hence the weights W_{ij}:

$$S'(x) = \frac{dS}{dx}$$

$$\Delta W_{ij} = -\eta \frac{\partial J}{\partial W_{ij}} = \eta \sum_\mu (y_i^\mu - O_i^\mu) \cdot S'(\sum_l W_{il} V_l^\mu) \cdot V_j^\mu = \eta \sum_\mu \delta_i^\mu V_j^\mu$$

where

$$\delta_i^\mu \equiv S'(\sum_l W_{il} V_l^\mu)(y_i^\mu - O_i^\mu)$$

Next, we adapt the connections between input and hidden layer:

$$\begin{aligned}
\Delta T_{jk} &= -\eta \frac{\partial J}{\partial T_{jk}} = -\eta \frac{\partial J}{\partial V_j^\mu} \frac{\partial V_j^\mu}{\partial T_{jk}} \\
&= \eta \sum_{\mu i} (y_i^\mu - O_i^\mu) \cdot S'(\sum_l W_{il} V_l^\mu) W_{ij} S'(\sum_l T_{jl} x_l^\mu) \\
&= \eta \sum_{\mu i} \delta_i^\mu W_{ij} S'(\sum_l T_{jl} x_l^\mu) x_k^\mu \\
&= \eta \sum_\mu \tilde{\delta}_j^\mu x_k^\mu
\end{aligned}$$

where

$$\tilde{\delta}_j^\mu \equiv S'(\sum_l T_{jl} x_l^\mu) \sum_l W_l j \delta_l^\mu$$

This leads to the **General learning rule:**

$$\Delta W_p q = \eta \sum_{\text{patterns}} \delta_{\text{output}} V_{\text{input}}$$

Backpropagation Algorithm

1. Initialize the weights W_{ij}^m with small random values
2. Choose a pattern x_k^μ and feed it to the input layer $(m = 0)$, i.e., $V_k^0 = x_k^\mu, \forall k$
3. Propagate the activity through the network where $V_i^m = S(\sum_j W_{ij}^m V_j^{m-1})$ holds for each i and each layer m
4. Calculate the weight corrections for the output layer $\delta_i^M = S'(h_i^M)[y_i^\mu - V_i^M]$ by comparing the actual output V_i^M with the desired output y_i^μ for pattern μ
5. Calculate the weight corrections for the previous layers by sending back the error signal $\delta_i^{m-1} = S'(h_i^{m-1}) \sum_j W_{ji}^m \delta_j^m$ for $m = M, M-1, ..., 2$, until a weight correction has been calculated for each unit.
6. Use $\Delta W_{ij} = \eta \delta_i^m V_j^{m-1}$, to adapt all weights according to $W_{ij}^{\text{new}} = W_{ij}^{\text{old}} + \Delta W_{ij}$
7. Goto 2 and repeat the loop for the next pattern

Remarks

- **A perceptron with one inner layer can approximate every continuous function**
- Backpropagation can be implemented with online or iterative and with batch learning mode $\Delta W_{pq} = \eta \delta_{\text{output}}^\mu V_{\text{input}}^\mu$
- Patterns should be presented in random order
- Learning is sometimes accelerated by adding noise
- The learning rule is local; all information after propagating back the error signal is available at node i
- The learning rule lacks biological plausibility!
- Rule of thumb: 5-10 training examples per independent weight are necessary

Variations on Backpropagation

- Alternative Cost Function
  - $J = \sum_{i,\mu} [y_i^\mu log \frac{y_i^\mu}{O_i^\mu} + (1 - y_i^\mu) log \frac{1-y_i^\mu}{1-O_i^\mu}]$

- O_i^μ is the probability that the hypothesis represented by a neuron is true
- y_i^μ is the probability which is supposed to be learned

- Convergence speedup by using momentum value
  - $m \in \mathbb{R}^+$, e.g. $m = 0.1$
  - $\delta_i^\mu = (S'(\sum_l W_{il} V_l^\mu) + m)(y_i^\mu - O_i^\mu)$

- Alternative minimization techniques
  - Conjugate gradient methods
  - Methods based on the Hessian $\frac{\partial^2 J}{\partial W_{ik}^2}$

- Regularisation techniques to avoid overfitting
  - Weight decay: rarely used weights decay to zero
    $J = J_0 + \gamma \sum_{i,k} \frac{W_{i,k}^2}{1+W_{i,k}^2}$
  - Connections between neurons that strongly fluctuate during training are selectively eliminated

12.3 Deep Neural Networks

Training Deep Networks

- This learning rule minimizes the cost function
  $$R = \sum_{\text{visible nodes}} Q_\alpha log \frac{Q_\alpha}{P_\alpha}(\{w_{ik}\}) \text{ since } \Delta w_{ik} = -\eta \frac{\partial R}{\partial w_{ik}}$$
  The target probability and the derived probability of visible units are denoted by Q_α and $P_\alpha(\{w_{ik}\})$, respectively

- Calculation of averages: Monte-Carlo simulation of a network. Disadvantage: training of a Boltzmann machine is rather time consuming.

12.4 Universal Approximation

Given:

- function $f : \mathbb{R}^n \rightarrow \mathbb{R}^p$ to be learned
- n input neurons, p output neurons

Prove

- uniform approximation of f on a compact support $\mathcal{K} \subset \mathbb{R}^n$
- $\forall \varepsilon > 0 \exists g \in \mathcal{G}$ such that $\|f(x) - g(x)\| < \varepsilon \quad \forall x \in \mathcal{K}$

Theorem

- Every continuous function $f : \mathbb{R}^n \to \mathbb{R}^p$ can be uniformly approximated on the compact support K by functions
  $y_k = S_k(\alpha_k + \sum_{j \to k} w_{jk} S_j(\alpha_j + \sum_{i \to j} w_{ij} x_i))$, $S_k \in \mathcal{NN}$
  with linear output units and logistic $S(x) = \frac{1}{1+e^{-x}}$ hidden units
- The notation $i \to j$ denotes the set of all indices i, where the associated node has an edge to j

12.5 Autoassociation

XOR

12.6 Boltzmann machines

A Boltzmann machine, like a Hopfield network, is a network of units with an "energy" defined for the network. It also has binary units, but unlike Hopfield nets, Boltzmann machine units are stochastic.
Difference to MLP: The multilayer perceptron is a network with a purely feedforward dynamics, meaning the activity is propagated from layer i to layer $i+1$ without feedback. However, a Boltzmann machine is a type of stochastic recurrent neural network.
The states of the units/neurons are updated in an asynchronous stochastic way by using the Metropolis sampler.
Dynamics

- stochastic binary neurons $x_i \in \{0, 1\}$ $1 \leq i \leq n$
- asynchronous update rule $x_i^{(t+1)} = \begin{cases} 1 & \text{if } \sum_{j=1}^{n} w_{ij} x_j^{(t)} \geq \theta_i \\ 0 & \text{otherwise} \end{cases}$

Network Architecture

symmetric couplings

- $w_{ik} = w_{ik} \Rightarrow$ cost function exists
- $w_{ii} = 0 \ \ \forall i$ (No unit has a connection with itself)

costs of the network

$$R(\{x_i\}) = -\frac{1}{2} \sum_{i,k=1}^{n} w_{ik} x_i x_k + \sum_{i=1}^{n} \theta_i x_i$$

The cost function $R(\{x_i^{(t)}\})$ is non-increasing under the deterministic asynchronous update rule:

$$x_i^{(t+1)} = \begin{cases} 1 & \text{if } \sum_{j=1}^{n} w_{ij} x_j^{(t)} \geq \theta_i \\ 0 & \text{otherwise} \end{cases}$$

Since $R(\{x_i^{(t)}\})$ is bounded from below for bounded weights w_{ik} the update dynamics terminates with probability 1 after a finite number of steps.
averages are calculated according to the Gibbs distribution

$$\mathbb{E}[f] = \sum_{\{x_i\}} P(\{x_i\}) f(\{x_i\})$$

with

$$P(\{x_i\}) = \frac{exp(-R(\{x_i\})/T)}{\sum_{\{x_i\}} exp(-R(\{x_i\})/T)}$$

Where $\sum_{\{x_i\}}$ denotes a sum over all possible 2^n configurations of the activities $x_i \in \{0,1\}$

Metropolis Sampler for Boltzmann Machines

MCMC Algorithm: (sequential asynchronous update)
Input: n neurons, cost function $R(\{x_i\})$
Output: activity pattern $x : \{\text{neurons}\} \rightarrow \{\text{activities}\}$

1. initialize $x(i) \in \{0,1\}$ randomly; $t \leftarrow 0$
2. repeat
    3. $t \leftarrow t + 1$
    4. draw $x' \sim Q(x)$ (where $Q(x)$ is the proposal distribution)
    5. $p \leftarrow min\{exp(-(R(x') - R(x))/T), 1\}$
    6. draw $b \sim$Bernoulli(p)
    7. if $b = 1$ then $x \leftarrow x'$
8. until convergence

Variants Gibbs or importance sampling is competitive!(The Metropolis sampler is often used to calculate averages)

Training of a Boltzmann Machine

- let the network evolve under two different conditions
    1. clamped dynamics: fix the activity of the visible nodes to the input/output patterns and measure correlation differences in the activity of two nodes
    2. free dynamics: let the network evolve without constraints
- measure the differences in average activity under the two conditions

- Learning rule for Boltzmann Machines

$$\Delta w_{ik} = \eta \frac{1}{T}\left[\overline{\mathbb{E}[x_i x_k]}_{\text{clamped}} - \mathbb{E}[x_i x_k]_{\text{free}}\right]$$

where $\overline{\mathbb{E}[x_i x_k]}$ denotes an average over all patterns.

13 Mixture Models

This is the chapter on Mixture Models.

13.1 k-Means Algorithm

13.2 Mixture Models

13.3 Expectation Maximization Algorithm

13.4 Convergence Proof of EM Algorithm

13.5 Readings

13.6 Problem Solving Procedures

EM for non-gaussian mixture

Problem setting You have to derive the EM algorithm for a mixture of non-gaussian distributions. You are given the model behind the EM in the form:

$$p(x) = \sum_{j=1}^{n} \pi_j Pr_{\text{non-gaussian}}$$

Solution

1. Write down the log likelihood function, which is
$$L(X, \{\text{parameters of } Pr_{non-gaussian} u_j s\}) = \sum_{i=1}^{n} \log\left(p(x_i)\right)$$
Let us assume that the parameters of $Pr_{non-gaussian}$ are $u_j s$.

2. Write down the $\gamma_j(x_i) = p(z_j = j|x_i)$ which is the probability that x_i was generated from j^{th} distribution of the mixture.
$$\gamma_j(x_i) = p(z_i = j|x_i) = \frac{p(x_i|z_i = j)p(z_i = j)}{p(x_i)} = \frac{p(x_i|z_i = j)p(z_i = j)}{\sum\limits_{k=1}^{n} p(x_i|z_i = k)p(z_i = k)} = \frac{\pi_i L(X, u_j s)}{\sum_{k=1}^{n} \pi_k L(X, u_j s)}$$

3. **To get optimal $\mathbf{u_j}$s**, derive the likelihood function $L(x, u_j s)$ by each parameter u_j. You will find out that it is something like

$$\nabla x_j L(X, u_j s) = \sum_{i=1}^{n} \frac{1}{Pr_{non-gaussian}} \cdot (\nabla u_j Pr_{non-gaussian})$$

(possibly replace by $\gamma_j(x_i)$ (above and below) leaving back some factor around it). Solve for the parameter u_j.

4. **Estimate the π parameters** by doing a Lagrange optimization on the log likelihood function and constraining the optimization by $\lambda \left(\sum_{j=1}^{k} \pi_j - 1 \right)$ as $\sum_{j=1}^{k} \pi_j = 1$

5. Put the λ into the formula and find the formula for π_j.

14 Learning Time Series

This is the chapter on Learning Time Series.

14.1 Stochastic Processes

14.2 Markov Models

14.3 Hidden Markov Models

14.4 Classifying Animal Behavior with HMM

15 Dimension Reduction

This lecture is/was not part of the exam of HS14. This is the chapter on Dimension Reduction.

15.1 Linear Projections

15.2 Locally Linear Embedding

15.3 Multi-Dimensional Scaling

16 Cheat sheet 2014 Benjamin Ellenberger

16.1 Probability

Probability Rules

Sum Rule	$P(X=x_i)=\sum_{j=1}^{J} p(X=x_i,Y=y_i)$
Product rule	$P(X,Y)=P(Y\|X)P(X)$
Independence	$P(X,Y)=P(X)P(Y)$
Bayes' Rule	$P(Y\|X)=\frac{P(X\|Y)P(Y)}{P(X)}=\frac{P(X\|Y)P(Y)}{\sum_{i=1}^{k}P(X\|Y_i)P(Y_i)}$
Conditional independence	$X\perp Y\|Z$
	$P(X,Y\|Z)=P(X\|Z)P(Y\|Z)$
	$P(X\|Z,Y)=P(X\|Z)$

Expectation

$E(X)=\int_{\inf}^{\inf} xp(x)dx$
$\sigma^2(X)=E(x^2)-E(x)^2$
$\sigma^2(X)=\int_x (x-\mu_x)^2 p(x)dx$
$Cov(X,Y)=\int_x\int_y p(x,y)(x-\mu_x)(y-\mu_y)dxdy$

Gaussian

$$p(X|\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$
$$=\frac{1}{\sqrt{2\pi^k|\Sigma|}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

Kernels

Requirements: Symmetric ($k(x,y)=k(y,x)$)
positive semi-definite K.

$k(x,y)=ak_1(x,y)+bk_2(x,y)$
$k(x,y)=k_1(x,y)k_2(x,y)$
$k(x,y)=f(x)f(y)$
$k(x,y)=k_3(\phi(x),\phi(y))$
with $a>0, b>0$

Linear	$k(x,y)=x^\top y$
Polynomial	$k(x,y)=(x^\top y+1)^d$
Gaussian RBF	$k(x,y)=\exp(\frac{-\|x-y\|_2^2}{h^2})$
Sigmoid	$k(x,y)=\tanh(kx^\top y-b)$

16.2 Regression

Linear Regression:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2$$

Closed form solution: $\beta^*=(X^\top X)^{-1}X^\top Y$
Ridge Regression:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2+\lambda\|w\|_2^2$$

Closed form solution: $\beta^*=(X^\top X+\lambda I)^{-1}X^\top Y$
Lasso Regression (sparse), requiring $\sum_j^d|\beta_j|<s$:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2+\lambda\|w\|_1$$

Kernelized Linear Regression:

$$\min_\alpha \|K\alpha-y\|_2^2+\lambda\alpha^\top K\alpha$$

Closed form solution: $\alpha=(K-\lambda I)^{-1}y$
Nonlinear Regression:

$$RSS(f,\lambda)=\sum_{i=1}^{n}(y_i-f(x_i))^2+\lambda\int(f''(x))^2dx$$

$$f(x)=\sum_{m=1}^{M}\beta_m h_m(x) \text{ with } h_m 1,x,(x-\xi_1)^2$$

ξ_1 are knots, the center of a function.
$f''(x)=0$ is required to have smoothness
Bias vs. Variance

$$\mathbb{E}_D\mathbb{E}_{X,Y}\left(\hat f(X)-Y\right)^2=\mathbb{E}_D\mathbb{E}_X\left(\hat f(X)-\mathbb{E}(Y|X)\right)^2$$
$$+\mathbb{E}_{X,Y}\left(Y-\mathbb{E}(Y|X)\right)^2$$
$$=\underbrace{\mathbb{E}_X\mathbb{E}_D\left(\hat f(X)-\mathbb{E}_D(\hat f(X))\right)^2}_{variance}$$
$$+\underbrace{\mathbb{E}_X\left(\mathbb{E}_D(\hat f(X))-\mathbb{E}(Y|X)\right)^2}_{\text{bias}^2}+\underbrace{\mathbb{E}_{X,Y}\left(Y-\mathbb{E}(Y|X)\right)^2}_{\text{noise}}$$

Combining regressions

$$bias[f(x)]=\mathbb{E}_D\hat f(x)-\mathbb{E}(Y|x)=\frac{1}{B}\sum_{i=1}^{B}\mathbb{E}_D\hat f_i(x)-\mathbb{E}(Y|x)$$

$$=\frac{1}{B}\sum_{i=1}^{B}(\mathbb{E}_D\hat f_i(x)-\mathbb{E}(Y|x))=\frac{1}{B}\sum_{i=1}^{B}bias[f_i(x)]$$

$$\mathbb{V}\{\hat f(x)\}\approx\frac{\sigma^2}{B}$$

16.3 Classification

0/1 Loss	$w^*=$	$\operatorname{argmin}_w \sum_{i=1}^{n}[y_i\neq sign(w^\top x_i)]$
Perceptron	$w^*=$	$\operatorname{argmin}_w \sum_{i=1}^{n}[\max(0,y_iw^\top x_i)]$

SVM

Primal, constrained:

$$\min_w w^\top w+C\sum_{i=1}^{n}\xi_i, \text{ s.t. } y_iw^\top x_i\geq 1-\xi_i, \xi_i\geq 0$$

Primal, unconstrained:

$$\min_w w^\top w+C\sum_{i=1}^{n}\max(0,1-y_iw^\top x_i) \text{ (hinge loss)}$$

Dual:

$$\max_\alpha \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i^\top x_j, \text{ s.t. } 0\leq\alpha_i\leq C$$

Dual to primal: $w^*=\sum_{i=1}^{n}\alpha_i^* y_i x_i$, $\alpha_i>0$: support vector.

Kernelized SVM

$$\max_\alpha \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(x_i,x_j), \text{ s.t. } 0\geq\alpha_i\geq C$$

Classify: $y=sign(\sum_{i=1}^{n}\alpha_i y_i k(x_i,x))$

How to find a^T?

$a=\{w_0,w\}$ used along $\tilde x=\{1,x\}$
Gradient Descent: $a(k+1)=a(k)-\eta(k)\nabla J(a(k))$
Newton method: 2nd order Taylor to get $\eta_{opt}=H^{-1}$ with $H=\frac{\partial^2 J}{\partial a_i\partial a_j}$
J is the cost matrix, popular choice is
Perceptron cost: $J_p(a)=\sum(-a^T\tilde x)$

Bootstrapping

Sample creation with replacement. Calculate mean and var of it and average.

$$\bar S=\frac{1}{B}\sum S(Z*)$$
$$\sigma^2(S)=\frac{1}{B-1}(S(Z^*)-\bar S)^2$$

Bootstrapping works if for $n\to\infty$ the error of empirical&bootstrap is the same as real&empiricial
Probability for a sample not to appear in set: $(1-\frac{1}{n})^n$. Goes to $\frac{1}{e}$ for $n\to\infty$
Multiplicity N sample to choose k times with replacement: $\binom{N-1+k}{k}$. In bootstrapping $N=k$

Jackknife

Method for debiasing at the price of variance $\bar\theta_{\text{Jack}}=\frac{1}{n}\sum_{i=1}^{n}(\bar\theta_i)$ (leave out ith sample, estimate and average)

16.4 Misc

Lagrangian: $f(x,y) s.t. g(x,y)=c$

$$\mathcal{L}(x,y,\gamma)=f(x,y)-\gamma(g(x,y)-c)$$

Parametric learning: model is parametrized with a finite set of parameters, like linear regression, linear SVM, etc.
Nonparametric learning: models grow in complexity with quantity of data: kernel SVM, k-NN, etc.
Empirical variance: Look for dense and sparse regions. Regularize so that sparse regions are not contained (decr. variance). Measure by Variance CV of some classifiers.

16.5 Probabilistic Methods

MLE

Least Squares, Gaussian Noise

$$L(w)=-\log(P(y_1...y_n|x_1...x_n,w))$$
$$=\frac{n}{2}\log(2\pi\sigma^2)+\sum_{i=1}^{n}\frac{(y_i-w^\top x_i)^2}{2\sigma^2}$$

$$\operatorname{argmax}_w P(y|x,w)=\operatorname{argmin}_w L(w)$$
$$=\operatorname{argmin}_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2$$

$$\hat\sigma^2=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar x)^2, E\left[\hat\sigma^2\right]=\frac{n-1}{n}\sigma^2$$

MAP

Ridge regression, Gaussian prior on weights

$$\hat y_{MAP}=\operatorname{argmax}_w P(w)\prod_i^n P(y_i|x_i,w)$$
$$=\operatorname{argmin}_w \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-w^\top x_i)+\frac{1}{2\beta^2}\sum_{i=1}^{n}w_i^2$$

$P(w)$ or $P(\theta)$ - conjugate prior (beta, Gaussian) (posterior same class as prior)
$P(y_i|\theta)$ - likelihood function (binomial, multinomial, Gaussian)
Beta distribution: $P(\theta)=Beta(\theta;\alpha_1,\alpha_2)\propto\theta^{\alpha_1-1}(1-\theta)^{\alpha_2-1}$

Bayesian Decision Theory

$$a^*=\operatorname{argmin}_{a\in A} E_y[C(y,a)|x]$$

16.6 Ensemble Methods

Use combination of simple hypotheses (weak learners) to create one strong learner.
strong learners: minimum error is below some $\delta<0.5$

weak learner: maximum error is below 0.5

$$f(x) = \sum_{i=1}^{n} \beta_i h_i(x)$$

Bagging: train weak learners on bootstrapped sets with equal weights.
Boosting: train on all data, but reweigh misclassified samples higher.

Decision Trees

Stumps: partition linearly along 1 axis

$$h(x) = sign(ax_i - t)$$

Decision Tree: recursive tree of stumps, leaves have labels. To train, either label if leaf's data is pure enough, or split data based on score.

Ada Boost

Effectively minimize exponential loss.

$$f^*(x) = \underset{f \in F}{\operatorname{argmin}} \sum_{i=1}^{n} \exp(-y_i f(x_i))$$

Train m weak learners, greedily selecting each one

$$(\beta_i, h_i) = \underset{\beta, h}{\operatorname{argmin}} \sum_{i=1}^{n} \exp(-y_i(f_{i-1}(x_j) + \beta h(x_j)))$$

$$c_b(x) \text{ trained with } w_i$$

$$\varepsilon_b = \sum_i^n \frac{w_i^b}{\sum_i^n w_i^b} I_{c(x_i) \neq y_i}$$

$$\alpha_b = log \frac{1 - \varepsilon_b}{\varepsilon_b}$$

$$w_i^{b+1} = w_i^b \cdot exp(\alpha_b I_{y_i \neq c_b(x_i)})$$

Exponential loss function
Additive logistic regression
Bayesian approached (assumes posteriors)
Newtonlike updates (Gradient Descent)
If previous classifier bad, next has heigh weight

16.7 Generative Methods

Discriminative - estimate $P(y|x)$ - conditional.
Generative - estimate $P(y, x)$ - joint, model data generation.

Naive Bayes

All features independent.

$$P(y|x) = \frac{1}{Z} P(y) P(x|y), Z = \sum_y P(y) P(x|y)$$

$$y = \underset{y'}{\operatorname{argmax}} P(y'|x) = \underset{y'}{\operatorname{argmax}} \hat{P}(y') \prod_{i=1}^{d} \hat{P}(x_i|y')$$

Discriminant Function

$$f(x) = \log(\frac{P(y=1|x)}{P(y==1|x)}), y = sign(f(x))$$

Fischer's Linear Discriminant Analysis (LDA)

Idea: project high dimensional data on one axis.
Complexity: $\mathcal{O}(d^2 n$ with d number of classifiers

$$c = 2, p = 0.5, \hat{\Sigma}_- = \hat{\Sigma}_+ = \hat{\Sigma}$$

$$\begin{aligned} y &= sign(w^\top x + w_0) \\ w &= \hat{\Sigma}^{-1}(\hat{\mu}_+ - \hat{\mu}_-) \\ w_0 &= \frac{1}{2}(\hat{\mu}_-^\top \Sigma^{-1} \hat{\mu}_- - \hat{\mu}_+^\top \Sigma^{-1} \hat{\mu}_+) \end{aligned}$$

16.8 Neural Networks

Backpropagation algorithm

1. Initialize the weights W_{ij}^m with small random values.
2. Choose a pattern x_μ^k and feed it to the input layer (m = 0), i.e., $V_k^0 = x_k^\mu$, $\forall k$.
3. Propagate the activity through the network where

   $$V_i^m = S\left(\sum_j W_{ij}^m V_j^{m-1}\right)$$

   holds for each i and each layer m.
4. Calculate the weight corrections for the output layer

   $$\delta_i^M = S'(h_i^M)[y_i^\mu - V_i^M]$$

   by comparing the actual output V_i^M with the desired output y_i^μ for pattern μ.
5. Calculate the weight corrections for the previous layers by sending back the error signal

   $$\delta_i^{m-1} = S'(h_i^{m-1}) \sum_j W_{ji}^m \delta_j^m$$

   for $m = M, M-1, \ldots, 2$, until a weight correction has been calculated for each unit.
6. Use $\Delta W_{ij} = \eta \delta_i^m V_j^{m-1}$, to adapt all weights according to $W_{ij}^{new} = W_{ij}^{old} + \Delta W_{ij}$
7. Goto 2 and repeat the loop for the next pattern.

16.9 Unsupervised Learning

Parzen

$$\hat{p}_n = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{V_n} \phi(\frac{x - x_i}{h_n})$$

where $\int \phi(x) dx = 1$

K-NN

$$\hat{p}_n = \frac{1}{V_k} \text{ volume with } k \text{ neighbours}$$

K-means

(clustering = classification)

$$L(\mu) = \sum_{i=1}^{n} \min_{j \in \{1 \ldots k\}} \|x_i - \mu_y\|_2^2$$

Lloyd's Heuristic: (1) assign each x_i to closest cluster (2) recalculate means of clusters.
Iteration over (repeated till stable):

Step 1: $\text{argmin}_c ||x - \mu_c||^2$

Step 2: $\mu_\alpha = \frac{1}{n_\alpha} \sum \vec{x}$

Gaussian Mixture Modeling

Same as Bayes, but class label z unobserved.

$$(\mu^*, \Sigma^*, w^*) = \text{argmin} - \sum_i log \sum_{j=1}^{k} w_j \mathcal{N}(x_i | \mu_i, \Sigma_j)$$

EM Algorithm

Problem: sum within log-term of likelihood.
E-step: expectation: pick clusters for points. Calculate $\gamma_j^{(t)}(x_i) = \frac{P(c|\theta^j)(x_i|c,\theta^j)}{\sum P(x_i|\theta)}$ for each i and j
M-Step: maximum likelihood: adjust clusters to best fit points.

$$\text{prior}_j^{(t)} = \pi_j^{(t)} \quad \leftarrow \quad \frac{1}{n} \sum_{i=1}^{n} \gamma_j^{(t)}(x_i)$$

$$\mu_j^{(t)} \quad \leftarrow \quad \frac{\sum_{i=1}^{n} \gamma_j^{(t)}(x_i)(x_i)}{\sum_{i=1}^{n} \gamma^{(t)}(x_i)}$$

$$\Sigma_j^{(t)} \quad \leftarrow \quad \frac{\sum_{i=1}^{n} \gamma_j^{(t)}(x_i)(x_i - \mu_j^{(t)})(x_i - \mu_j^{(t)})^\top}{\sum_{i=1}^{n} \gamma_j^{(t)}(x_i)}$$

EM for non-gaussian mixture

Problem setting Derive the EM algorithm. The model behind the EM in the form: $p(x) = \sum_{j=1}^{n} \pi_j Pr_{\text{non-gaussian}}$

Solution

1. Write log likelihood function:

   $$L(X, \{\text{params} Pr_{non-gaussian} u_j s\}) = \sum_{i=1}^{n} \log(p(x_i))$$

2. Write $\gamma_j(x_i) = p(z_j = j|x_i)$ (prob. that x_i was generated from j^{th} dist. of the mixture).

   $$\gamma_j(x_i) = p(z_i = j|x_i) = \frac{p(x_i|z_i = j)p(z_i = j)}{p(x_i)}$$

   $$= \frac{p(x_i|z_i = j)p(z_i = j)}{\sum_{k=1}^{n} p(x_i|z_i = k)p(z_i = k)} = \frac{\pi_i L(X, u_j s)}{\sum_{k=1}^{n} \pi_k L(X, u_j s)}$$

3. **To get optimal $\mathbf{u_j}$s**, derive $L(x, u_j s)$ by each param u_j. You get:

   $$\nabla x_j L(X, u_j s) = \sum_{i=1}^{n} \frac{1}{Pr_{non-gaussian}} \cdot (\nabla u_j Pr_{non-gaussian})$$

   (possibly replace by $\gamma_j(x_i)$ (above and below) leaving back some factor). Solve for the parameter u_j.

4. **Estimate the π parameters** by Lagrange optimization on log likelihood function and constraint $\lambda\left(\sum_{j=1}^{k} \pi_j - 1\right)$. Put the λ into the formula and find the formula for π_j.

16.10 Hidden-Markov model

State only depends on previous state.
Always given: sequence of symbols $\vec{s} = \{s_1, s_2, \ldots s_n\}$

Evaluation (Forward & Backward)

Known: $a_{ij}, e_k(s_t)$
Wanted: $P(X = x_i | S = s_t)$

$$f_l(s_{t+1}) = e_l(s_{t+1}) \sum f_k(s_t) a_{kl}$$

$$b_l(s_t) = e_l(s_t) \sum b_k(s_{t+1}) a_{lk}$$

$$P(\vec{s}) = \sum_k f_k(s_n) a_k \cdot \text{ end}$$

$$P(x_{l,t}|\vec{s}) = \frac{f_l(s_t) b_l(s_t)}{P(\vec{s})}$$

Complexity in time: $\mathcal{O}(|S|^2 \cdot T)$

Decoding (Viterbi)

Known: $a_{ij}, e_k(s_t)$
Wanted: most likely path $\vec{x} = \{x_1, x_2, \ldots x_n\}$

$$v_l(s_{t+1}) = \max_k \{v_k(s_t) a_{kl}\}$$

$$\text{ptr}_t(l) = \text{argmax}_k \{v_k(s_t) a_{kl}\}$$

Follow pointers back from end to beginning.
Dynamic, because splittable in sub parts (per time step)
Time: $\mathcal{O}(|S|^2 \cdot T)$ Space $\mathcal{O}(|S| \cdot T)$

Learning (Baum-Welch)

Known: only sequence and sequence space Θ
Wanted: $a_{ij}, e_k(s_t)$ & most likely path $\vec{x} = \{x_1, x_2, \ldots x_n\}$
E-step I: $f_k(s_t), b_k(s_t)$ by forward & backward algorithm
E-step II:

$$P(X_t = x_k, X_{t+1} = x_l | \vec{s}, \Theta) = \frac{1}{P(\vec{s})} f_k(s_t) a_{kl} e_l(s_{t+1}) b_l(s_{t+1})$$

$$A_{kl} = \sum_{j=1}^{m} \sum_{t=1}^{n} P(X_t = x_k, X_{t+1} = x_l | \vec{s}, \Theta)$$

M-step :

$$a_{kl} = \frac{A_{kl}}{\sum_i^n A_{ki}} \text{ and } e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

Complexity: $\mathcal{O}(|S|^2)$ in storage (space)

17 Cheat sheet 2014 Joachim Ott

17.1 Probability

Probability Rules

Sum Rule	$P(X = x_i) = \sum_{j=1}^{J} p(X = x_i, Y = y_i)$
Product rule	$P(X,Y) = P(Y\|X)P(X)$
Independence	$P(X,Y) = P(X)P(Y)$
Bayes' Rule	$P(Y\|X) = \frac{P(X\|Y)P(Y)}{P(X)}$
Conditional independence	$X \perp Y \| Z$

$$P(X,Y|Z) = P(X|Z)P(Y|Z)$$
$$P(X|Z,Y) = P(X|Z)$$

Expectation

$E(X) = \int_{\inf}^{\inf} xp(x)dx$
$\sigma^2(X) = E(x^2) - E(x)^2$
$\sigma^2(X) = \int_x (x-\mu_x)^2 p(x)dx$
$Cov(X,Y) = \int_x \int_y p(x,y)(x-\mu_x)(y-\mu_y)dxdy$

Gaussian

$$p(X|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Kernels

Requirements: Symmetric $(k(x,y) = k(y,x))$
positive semi-definite K.

$k(x,y) = ak_1(x,y) + bk_2(x,y)$
$k(x,y) = k_1(x,y)k_2(x,y)$
$k(x,y) = f(x)f(y)$
$k(x,y) = k_3(\phi(x),\phi(y))$

Linear	$k(x,y) = x^\top y$
Polynomial	$k(x,y) = (x^\top y + 1)^d$
Gaussian RBF	$k(x,y) = \exp(\frac{-\|x-y\|_2^2}{h^2})$
Sigmoid	$k(x,y) = \tanh(kx^\top y - b)$

17.2 Regression

Linear Regression:

$$\min_w \sum_{i=1}^{n} (y_i - w^\top x_i)^2$$

Closed form solution: $w^* = (x^\top x)^{-1} x^\top y$
Ridge Regression:

$$\min_w \sum_{i=1}^{n} (y_i - w^\top x_i)^2 + \lambda \|w\|_2^2$$

Closed form solution: $w^* = (x^\top x + \lambda I)^{-1} x^\top y$
Lasso Regression (sparse):

$$\min_w \sum_{i=1}^{n} (y_i - w^\top x_i)^2 + \lambda \|w\|_1$$

Kernelized Linear Regression:

$$\min_\alpha \|K\alpha - y\|_2^2 + \lambda \alpha^\top K \alpha$$

Closed form solution: $\alpha = (K - \lambda I)^{-1} y$

17.3 Classification

$$\text{0/1 Loss} \quad w^* = \operatorname*{argmin}_w \sum_{i=1}^{n} [y_i \neq sign(w^\top x_i)]$$

$$\text{Perceptron} \quad w^* = \operatorname*{argmin}_w \sum_{i=1}^{n} [\max(0, y_i w^\top x_i)]$$

SVM

Primal, constrained:

$$\min_w w^\top w + C \sum_{i=1}^{n} \xi_i, \text{ s.t. } y_i w^\top x_i \geq 1 - \xi_i, \xi_i \geq 0$$

Primal, unconstrained:

$$\min_w w^\top w + C \sum_{i=1}^{n} \max(0, 1 - y_i w^\top x_i) \text{ (hinge loss)}$$

Dual:

$$\max_\alpha \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^\top x_j, \text{ s.t. } 0 \geq \alpha_i \geq C$$

Dual to primal: $w^* = \sum_{i=1}^{n} \alpha_i^* y_i x_i$, $\alpha_i > 0$: support vector.

Kernelized SVM

$$\max_\alpha \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i, x_j), \text{ s.t. } 0 \geq \alpha_i \geq C$$

Classify: $y = sign(\sum_{i=1}^{n} \alpha_i y_i k(x_i, x))$

17.4 Misc

Lagrangian: $f(x,y) s.t. g(x,y) = c$

$$\mathcal{L}(x,y,\gamma) = f(x,y) - \gamma(g(x,y) - c)$$

Parametric learning: model is parameterized with a finite set of parameters, like linear regression, linear SVM, etc.

Nonparametric learning: models grow in complexity with quantity of data: kernel SVM, k-NN, etc.

17.5 Probabilistic Methods:

MLE

Least Squares, Gaussian Noise

$$L(w) = -\log(P(y_1...y_n | x_1...x_n, w))$$
$$= \frac{n}{2}\log(2\pi\sigma^2) + \sum_{i=1}^{n} \frac{(y_i - w^\top x_i)^2}{2\sigma^2}$$

$$\operatorname*{argmax}_w P(y|x,w) = \operatorname*{argmin}_w L(w)$$
$$= \operatorname*{argmin}_w \sum_{i=1}^{n} (y_i - w^\top x_i)^2$$

MAP

Ridge regression, Gaussian prior on weights

$$\operatorname*{argmax}_w P(w) \prod_i^n P(y_i | x_i, w)$$
$$= \operatorname*{argmin}_w \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - w^\top x_i) + \frac{1}{2\beta^2} \sum_{i=1}^{n} w_i^2$$

$P(w)$ or $P(\theta)$ - conjugate prior (beta, Gaussian) (posterior same class as prior)
$P(y_i|\theta)$ - likelihood function (binomial, multinomial, Gaussian)
Beta distribution: $P(\theta) = Beta(\theta; \alpha_1, \alpha_2) \propto \theta^{\alpha_1 - 1}(1-\theta)^{\alpha_2 - 1}$

Bayesian Decision Theory

$$a^* = \operatorname*{argmin}_{a \in A} E_y[C(y,a)|x]$$

17.6 Ensemble Methods

Use combination of simple hypotheses (weak learners) to create one strong learner.

$$f(x) = \sum_{i=1}^{n} \beta_i h_i(x)$$

Bagging: train weak learners on random subsamples with equal weights.
Boosting: train on all data, but reweigh misclassified samples higher.

Decision Trees

Stumps: partition linearly along 1 axis

$$h(x) = sign(ax_i - t)$$

Decision Tree: recursive tree of stumps, leaves have labels. To train, either label if leaf's data is pure enough, or split data based on score.

Ada Boost

Effectively minimize exponential loss.

$$f^*(x) = \operatorname*{argmin}_{f \in F} \sum_{i=1}^{n} \exp(-y_i f(x_i))$$

Train m weak learners, greedily selecting each one

$$(\beta_i, h_i) = \operatorname*{argmin}_{\beta,h} \sum_{i=1}^{n} \exp(-y_i (f_{i-1}(x_j) + \beta h(x_j)))$$

17.7 Generative Methods

Discriminative - estimate $P(y|x)$ - conditional.
Generative - estimate $P(y,x)$ - joint, model data generation.

Naive Bayes

All features independent.

$$P(y|x) = \frac{1}{Z} P(y)P(x|y), Z = \sum_y P(y)P(x|y)$$

$$y = \operatorname*{argmax}_{y'} P(y'|x) = \operatorname*{argmax}_{y'} \hat{P}(y') \prod_{i=1}^{d} \hat{P}(x_i|y')$$

Discriminant Function

$$f(x) = \log\left(\frac{P(y=1|x)}{P(y==1|x)}\right), y = sign(f(x))$$

Fischer's Linear Discriminant Analysis (LDA)

$$c = 2, p = 0.5, \hat{\Sigma}_- = \hat{\Sigma}_+ = \hat{\Sigma}$$

$$\begin{aligned} y &= sign(w^\top x + w_0) \\ w &= \hat{\Sigma}^{-1}(\hat{\mu}_+ - \hat{\mu}_-) \\ w_0 &= \frac{1}{2}(\hat{\mu}_-^\top \Sigma^{-1} \hat{\mu}_- - \hat{\mu}_+^\top \Sigma^{-1} \hat{\mu}_+) \end{aligned}$$

17.8 Unsupervised Learning

K-means

(clustering = classification)

$$L(\mu) = \sum_{i=1}^{n} \min_{j \in \{1...k\}} \|x_i - \mu_y\|_2^2$$

Lloyd's Heuristic: (1) assign each x_i to closest cluster (2) recalculate means of clusters.

Gaussian Mixture Modeling

Same as Bayes, but class label z unobserved.

$$(\mu^*, \Sigma^*, w^*) = \operatorname{argmin} - \sum_i log \sum_{j=1}^{k} w_j \mathcal{N}(x_i|\mu_i, \Sigma_j)$$

EM Algorithm

E-step: expectation: pick clusters for points. Calculate $\gamma_j^{(t)}(x_i)$ for each i and j
M-Step: maximum likelihood: adjust clusters to best fit points.

$$
\begin{aligned}
\omega_j^{(t)} &\leftarrow \frac{1}{n}\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)\\
\mu_j^{(t)} &\leftarrow \frac{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)(x_i)}{\sum_{i=1}^{n}\gamma^{(t)}(x_i)}\\
\Sigma_j^{(t)} &\leftarrow \frac{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)(x_i-\mu_j^{(t)})(x_i-\mu_j^{(t)})^\top}{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)}
\end{aligned}
$$

PCA

(dimensional reduction = regression)

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n} x_i x_j^\top, \quad \mu = \frac{1}{n}\sum_{i=1}^{n} x_i = 0$$

$$(W, z_1, \ldots, z_n) = \operatorname{argmin}\sum_{i=1}^{n}\|Wz_i - x_i\|_2^2$$

W is orthogonal, $W = (v_1|\ldots|v_k)$ and $z_i = w^\top x_i$

$$\Sigma = \sum_{i=1}^{d}\lambda_i v_i v_i^\top \quad \lambda_1 \geq \ldots \geq \lambda_d \geq 0$$

Kernel PCA

$$\alpha_i^* = \arg\max_{\alpha^\top K\alpha = 1} = \alpha^\top K^\top K\alpha$$

$$\alpha^{(i)} = \frac{1}{\sqrt{\lambda_i}}\frac{v_i}{\|v_i\|_2}, \quad K = \sum_{i=1}^{n}\lambda_i v_i v_i^\top$$

18 Cheat sheet 2014 Imanol Studer

18.1 Probability

Probability Rules

Sum Rule	$P(X=x_i)=\sum_{j=1}^{J} p(X=x_i, Y=y_i)$
Product rule	$P(X,Y)=P(Y\|X)P(X)$
Independence	$P(X,Y)=P(X)P(Y)$
Bayes' Rule	$P(Y\|X)=\frac{P(X\|Y)P(Y)}{P(X)}$
Conditional independence	$X \perp Y \| Z$
	$P(X,Y\|Z)=P(X\|Z)P(Y\|Z)$ $P(X\|Z,Y)=P(X\|Z)$

Expectation

$E(X)=\int_{\inf}^{\inf} xp(x)dx$
$\sigma^2(X)=E(x^2)-E(x)^2$
$\sigma^2(X)=\int_x (x-\mu_x)^2 p(x)dx$
$Cov(X,Y)=\int_x \int_y p(x,y)(x-\mu_x)(y-\mu_y)dxdy$

Gaussian

$$p(X|\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Kernels

Requirements: Symmetric $(k(x,y)=k(y,x))$

positive semi-definite K.
$k(x,y)=ak_1(x,y)+bk_2(x,y)$
$k(x,y)=k_1(x,y)k_2(x,y)$
$k(x,y)=f(x)f(y)$
$k(x,y)=k_3(\phi(x),\phi(y))$

Linear	$k(x,y)=x^\top y$
Polynomial	$k(x,y)=(x^\top y+1)^d$
Gaussian RBF	$k(x,y)=\exp(\frac{-\|x-y\|_2^2}{h^2})$
Sigmoid	$k(x,y)=\tanh(kx^\top y-b)$

18.2 Regression

Linear Regression:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2$$

Closed form solution: $w^*=(x^\top x)^{-1}x^\top y$
Ridge Regression:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2+\lambda\|w\|_2^2$$

Closed form solution: $w^*=(x^\top x+\lambda I)^{-1}x^\top y$
Lasso Regression (sparse):

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2+\lambda\|w\|_1$$

Kernelized Linear Regression:

$$\min_\alpha \|K\alpha-y\|_2^2+\lambda\alpha^\top K\alpha$$

Closed form solution: $\alpha=(K-\lambda I)^{-1}y$

18.3 Classification

$$\text{0/1 Loss} \quad w^*=\operatorname{argmin}_w \sum_{i=1}^{n}[y_i\neq sign(w^\top x_i)]$$

$$\text{Perceptron} \quad w^*=\operatorname{argmin}_w \sum_{i=1}^{n}[\max(0,y_i w^\top x_i)]$$

SVM

Primal, constrained:

$$\min_w w^\top w+C\sum_{i=1}^{n}\xi_i,\ \text{s.t. } y_i w^\top x_i\geq 1-\xi_i, \xi_i\geq 0$$

Primal, unconstrained:

$$\min_w w^\top w+C\sum_{i=1}^{n}\max(0,1-y_i w^\top x_i)\ \text{(hinge loss)}$$

Dual:

$$\max_\alpha \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i^\top x_j,\ \text{s.t. } 0\geq\alpha_i\geq C$$

Dual to primal: $w^*=\sum_{i=1}^{n}\alpha_i^* y_i x_i$, $\alpha_i>0$: support vector.

Kernelized SVM

$$\max_\alpha \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(x_i,x_j),\ \text{s.t. } 0\geq\alpha_i\geq C$$

Classify: $y=sign(\sum_{i=1}^{n}\alpha_i y_i k(x_i,x))$

18.4 Misc

Lagrangian: $f(x,y)s.t.g(x,y)=c$

$$\mathcal{L}(x,y,\gamma)=f(x,y)-\gamma(g(x,y)-c)$$

Parametric learning: model is parameterized with a finite set of parameters, like linear regression, linear SVM, etc.

Nonparametric learning: models grow in complexity with quantity of data: kernel SVM, k-NN, etc.

18.5 Probabilistic Methods:

MLE

Least Squares, Gaussian Noise

$$L(w)=-\log(P(y_1...y_n|x_1...x_n,w))$$
$$=\frac{n}{2}\log(2\pi\sigma^2)+\sum_{i=1}^{n}\frac{(y_i-w^\top x_i)^2}{2\sigma^2}$$

$$\operatorname{argmax}_w P(y|x,w)=\operatorname{argmin}_w L(w)$$
$$=\operatorname{argmin}_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2$$

MAP

Ridge regression, Gaussian prior on weights

$$\operatorname{argmax}_w P(w)\prod_i^n P(y_i|x_i,w)$$
$$=\operatorname{argmin}_w \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-w^\top x_i)+\frac{1}{2\beta^2}\sum_{i=1}^{n}w_i^2$$

$P(w)$ or $P(\theta)$ - conjugate prior (beta, Gaussian) (posterior same class as prior)
$P(y_i|\theta)$ - likelihood function (binomial, multinomial, Gaussian)
Beta distribution: $P(\theta)=Beta(\theta;\alpha_1,\alpha_2)\propto\theta^{\alpha_1-1}(1-\theta)^{\alpha_2-1}$

Bayesian Decision Theory

$$a^*=\operatorname{argmin}_{a\in A} E_y[C(y,a)|x]$$

18.6 Ensemble Methods

Use combination of simple hypotheses (weak learners) to create one strong learner.

$$f(x)=\sum_{i=1}^{n}\beta_i h_i(x)$$

Bagging: train weak learners on random subsamples with equal weights.
Boosting: train on all data, but reweigh misclassified samples higher.

Decision Trees

Stumps: partition linearly along 1 axis

$$h(x)=sign(ax_i-t)$$

Decision Tree: recursive tree of stumps, leaves have labels. To train, either label if leaf's data is pure enough, or split data based on score.

Ada Boost

Effectively minimize exponential loss.

$$f^*(x)=\operatorname{argmin}_{f\in F}\sum_{i=1}^{n}\exp(-y_i f(x_i))$$

Train m weak learners, greedily selecting each one

$$(\beta_i,h_i)=\operatorname{argmin}_{\beta,h}\sum_{i=1}^{n}\exp(-y_i(f_{i-1}(x_j)+\beta h(x_j)))$$

18.7 Generative Methods

Discriminative - estimate $P(y|x)$ - conditional.
Generative - estimate $P(y,x)$ - joint, model data generation.

Naive Bayes

All features independent.

$$P(y|x)=\frac{1}{Z}P(y)P(x|y), Z=\sum_y P(y)P(x|y)$$

$$y=\operatorname{argmax}_{y'} P(y'|x)=\operatorname{argmax}_{y'}\hat{P}(y')\prod_{i=1}^{d}\hat{P}(x_i|y')$$

Discriminant Function

$$f(x)=\log\left(\frac{P(y=1|x)}{P(y==1|x)}\right), y=sign(f(x))$$

Fischer's Linear Discriminant Analysis (LDA)

$$c=2, p=0.5, \hat{\Sigma}_-=\hat{\Sigma}_+=\hat{\Sigma}$$
$$\begin{aligned} y &= sign(w^\top x+w_0) \\ w &= \hat{\Sigma}^{-1}(\hat{\mu}_+-\hat{\mu}_-) \\ w_0 &= \frac{1}{2}(\hat{\mu}_-^\top\Sigma^{-1}\hat{\mu}_--\hat{\mu}_+^\top\Sigma^{-1}\hat{\mu}_+) \end{aligned}$$

18.8 Unsupervised Learning

K-means

(clustering = classification)

$$L(\mu)=\sum_{i=1}^{n}\min_{j\in\{1...k\}}\|x_i-\mu_y\|_2^2$$

Lloyd's Heuristic: (1) assign each x_i to closest cluster (2) recalculate means of clusters.

Gaussian Mixture Modeling

Same as Bayes, but class label z unobserved.

$$(\mu^*,\Sigma^*,w^*)=\operatorname{argmin}-\sum_i log\sum_{j=1}^{k}w_j\mathcal{N}(x_i|\mu_i,\Sigma_j)$$

EM Algorithm

E-step: expectation: pick clusters for points. Calculate $\gamma_j^{(t)}(x_i)$ for each i and j
M-Step: maximum likelihood: adjust clusters to best fit points.

$$\omega_j^{(t)} \leftarrow \frac{1}{n}\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)$$

$$\mu_j^{(t)} \leftarrow \frac{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)(x_i)}{\sum_{i=1}^{n}\gamma^{(t)}(x_i)}$$

$$\Sigma_j^{(t)} \leftarrow \frac{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)(x_i-\mu_j^{(t)})(x_i-\mu_j^{(t)})^\top}{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)}$$

PCA

(dimensional reduction = regression)

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n} x_i x_j^\top, \quad \mu = \frac{1}{n}\sum_{i=1}^{n} x_i = 0$$

$$(W, z_1, \dots, z_n) = \operatorname{argmin} \sum_{i=1}^{n} \|Wz_i - x_i\|_2^2$$

W is orthogonal, $W = (v_1|\dots|v_k)$ and $z_i = w^\top x_i$

$$\Sigma = \sum_{i=1}^{d} \lambda_i v_i v_i^\top \quad \lambda_1 \geq \dots \geq \lambda_d \geq 0$$

Kernel PCA

$$\alpha_i^* = \arg\max_{\alpha^\top K\alpha = 1} = \alpha^\top K^\top K \alpha$$

$$\alpha^{(i)} = \frac{1}{\sqrt{\lambda_i}}\frac{v_i}{\|v_i\|_2}, \quad K = \sum_{i=1}^{n} \lambda_i v_i v_i^\top$$

19 Cheat sheet 2014 Corneel Van der Pol

19.1 Probability

Probability Rules

Sum Rule	$P(X=x_i)=\sum_{j=1}^{J} p(X=x_i,Y=y_i)$
Product rule	$P(X,Y)=P(Y\|X)P(X)$
Independence	$P(X,Y)=P(X)P(Y)$
Bayes' Rule	$P(Y\|X)=\frac{P(X\|Y)P(Y)}{P(X)}$
Conditional independence	$X \perp Y \| Z$
	$P(X,Y\|Z)=P(X\|Z)P(Y\|Z)$
	$P(X\|Z,Y)=P(X\|Z)$

Expectation

$E(X)=\int_{\inf}^{\inf} xp(x)dx$
$\sigma^2(X)=E(x^2)-E(x)^2$
$\sigma^2(X)=\int_x (x-\mu_x)^2 p(x)dx$
$Cov(X,Y)=\int_x\int_y p(x,y)(x-\mu_x)(y-\mu_y)dxdy$

Gaussian

$$p(X|\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Kernels

Requirements: Symmetric $(k(x,y)=k(y,x))$ positive semi-definite K.

$k(x,y)=ak_1(x,y)+bk_2(x,y)$
$k(x,y)=k_1(x,y)k_2(x,y)$
$k(x,y)=f(x)f(y)$
$k(x,y)=k_3(\phi(x),\phi(y))$
with $a>0, b>0$

Linear	$k(x,y)=x^\top y$
Polynomial	$k(x,y)=(x^\top y+1)^d$
Gaussian RBF	$k(x,y)=\exp(\frac{-\|x-y\|_2^2}{h^2})$
Sigmoid	$k(x,y)=\tanh(kx^\top y-b)$

19.2 Regression

Linear Regression:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2$$

Closed form solution: $\beta^*=(X^\top X)^{-1}X^\top Y$

Ridge Regression:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2+\lambda\|w\|_2^2$$

Closed form solution: $\beta^*=(X^\top X+\lambda I)^{-1}X^\top Y$

Lasso Regression (sparse), requiring $\sum_j^d|\beta_j|<s$:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2+\lambda\|w\|_1$$

Kernelized Linear Regression:

$$\min_\alpha \|K\alpha-y\|_2^2+\lambda\alpha^\top K\alpha$$

Closed form solution: $\alpha=(K-\lambda I)^{-1}y$

Nonlinear Regression:

$$RSS(f,\lambda)=\sum_{i=1}^{n}(y_i-f(x_i))^2+\lambda\int(f''(x))^2dx$$

$$f(x)=\sum_{m=1}^{M}\beta_m h_m(x) \text{ with } h_m 1,x,(x-\xi_1)^2$$

ξ_1 are knots, the center of a function. $f''(x)=0$ is required to have smoothness

19.3 Classification

0/1 Loss $\quad w^*=\underset{w}{\operatorname{argmin}}\sum_{i=1}^{n}[y_i\neq sign(w^\top x_i)]$

Perceptron $\quad w^*=\underset{w}{\operatorname{argmin}}\sum_{i=1}^{n}[\max(0,y_i w^\top x_i)]$

SVM

Primal, constrained:

$$\min_w w^\top w+C\sum_{i=1}^{n}\xi_i, \text{ s.t. } y_i w^\top x_i\geq 1-\xi_i, \xi_i\geq 0$$

Primal, unconstrained:

$$\min_w w^\top w+C\sum_{i=1}^{n}\max(0,1-y_i w^\top x_i) \text{ (hinge loss)}$$

Dual:

$$\max_\alpha \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i^\top x_j, \text{ s.t. } 0\geq\alpha_i\geq C$$

Dual to primal: $w^*=\sum_{i=1}^{n}\alpha_i^* y_i x_i$, $\alpha_i>0$: support vector.

Kernelized SVM

$$\max_\alpha \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(x_i,x_j), \text{ s.t. } 0\geq\alpha_i\geq C$$

Classify: $y=sign(\sum_{i=1}^{n}\alpha_i y_i k(x_i,x))$

How to find $a^\top$?

$a=\{w_0,w\}$ used along $\tilde{x}=\{1,x\}$
Gradient Descent: $a(k+1)=a(k)-\eta(k)\nabla J(a(k))$
Newton method: 2nd order Taylor to get $\eta_{opt}=H^{-1}$ with $H=\frac{\partial^2 J}{\partial a_i\partial a_j}$
J is the cost matrix, popular choice is
Perceptron cost: $J_p(a)=\sum(-a^T\tilde{x})$

Bootstrapping

Method for reducing variance

$$\bar{S}=\frac{1}{B}\sum S(Z*)$$

$$\sigma^2(S)=\frac{1}{B-1}(S(Z^*)-\bar{S})^2$$

Bootstrapping works if for $n\to\infty$ the error of empirical&bootstrap is the same as real&empiricial
Probability for a sample not to appear in set: $(1-\frac{1}{n})^n$. Goes to $\frac{1}{e}$ for $n\to\infty$
Multiplicity N sample to choose k times with replacement: $\binom{N-1+k}{k}$. In bootstrapping $N=k$

Jackknife

Method for debiasing at the price of variance

19.4 Misc

Lagrangian: $f(x,y) s.t. g(x,y)=c$

$$\mathcal{L}(x,y,\gamma)=f(x,y)-\gamma(g(x,y)-c)$$

Parametric learning: model is parameterized with a finite set of parameters, like linear regression, linear SVM, etc.

Nonparametric learning: models grow in complexity with quantity of data: kernel SVM, k-NN, etc.

19.5 Probabilistic Methods:

MLE

Least Squares, Gaussian Noise

$$L(w)=-\log(P(y_1...y_n|x_1...x_n,w))$$
$$=\frac{n}{2}\log(2\pi\sigma^2)+\sum_{i=1}^{n}\frac{(y_i-w^\top x_i)^2}{2\sigma^2}$$

$$\underset{w}{\operatorname{argmax}} P(y|x,w)=\underset{w}{\operatorname{argmin}} L(w)$$
$$=\underset{w}{\operatorname{argmin}}\sum_{i=1}^{n}(y_i-w^\top x_i)^2$$

$$\hat{\sigma}^2=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2, E\left[\hat{\sigma}^2\right]=\frac{n-1}{n}\sigma^2$$

MAP

Ridge regression, Gaussian prior on weights

$$\hat{y}_{MAP}=\underset{w}{\operatorname{argmax}} P(w)\prod_i P(y_i|x_i,w)$$

$$=\underset{w}{\operatorname{argmin}}\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-w^\top x_i)+\frac{1}{2\beta^2}\sum_{i=1}^{n}w_i^2$$

$P(w)$ or $P(\theta)$ - conjugate prior (beta, Gaussian) (posterior same class as prior)

$P(y_i|\theta)$ - likelihood function (binomial, multinomial, Gaussian)
Beta distribution: $P(\theta) = Beta(\theta;\alpha_1,\alpha_2) \propto \theta^{\alpha_1-1}(1-\theta)^{\alpha_2-1}$

Bayesian Decision Theory

$$a^* = \operatorname*{argmin}_{a\in A} E_y[C(y,a)|x]$$

19.6 Ensemble Methods

Use combination of simple hypotheses (weak learners) to create one strong learner.
strong learners: minimum error is below some $\delta < 0.5$
weak learner: maximum error is below 0.5

$$f(x) = \sum_{i=1}^{n} \beta_i h_i(x)$$

Bagging: train weak learners on bootstrapped sets with equal weights.
Boosting: train on all data, but reweigh misclassified samples higher.

Decision Trees

Stumps: partition linearly along 1 axis

$$h(x) = sign(ax_i - t)$$

Decision Tree: recursive tree of stumps, leaves have labels. To train, either label if leaf's data is pure enough, or split data based on score.

AdaBoost

Effectively minimize exponential loss.

$$f^*(x) = \operatorname*{argmin}_{f\in F} \sum_{i=1}^{n} \exp(-y_i f(x_i))$$

Train m weak learners, greedily selecting each one

$$(\beta_i, h_i) = \operatorname*{argmin}_{\beta,h} \sum_{i=1}^{n} \exp(-y_i(f_{i-1}(x_j) + \beta h(x_j)))$$

$c_b(x)$ trained with w_i

$$\varepsilon_b = \sum_i^n \frac{w_i^b}{\sum_i^n w_i^b} I_{c(x_i)\neq y_i}$$

$$\alpha_b = log\frac{1-\varepsilon_b}{\varepsilon_b}$$

$$w_i = e^{-\alpha_b I_{y_i \neq c_b(x_i)}}$$

Exponential loss function
Additive logistic regression
Bayesian approached (assumes posteriors)
Newtonlike updates (Gradient Descent)
If previous classifier bad, next has heigh weight

19.7 Generative Methods

Discriminative - estimate $P(y|x)$ - conditional.
Generative - estimate $P(y,x)$ - joint, model data generation.

Naive Bayes

All features independent.

$$P(y|x) = \frac{1}{Z}P(y)P(x|y), Z = \sum_y P(y)P(x|y)$$

$$y = \operatorname*{argmax}_{y'} P(y'|x) = \operatorname*{argmax}_{y'} \hat{P}(y')\prod_{i=1}^{d}\hat{P}(x_i|y')$$

Discriminant Function

$$f(x) = \log(\frac{P(y=1|x)}{P(y==1|x)}), y = sign(f(x))$$

Fischer's Linear Discriminant Analysis (LDA)

Idea: project high dimensional data on one axis.
Complexity: $\mathcal{O}(d^2 n$ with d number of classifiers

$$c = 2, p = 0.5, \hat{\Sigma}_- = \hat{\Sigma}_+ = \hat{\Sigma}$$

$$\begin{aligned} y &= sign(w^\top x + w_0) \\ w &= \hat{\Sigma}^{-1}(\hat{\mu}_+ - \hat{\mu}_-) \\ w_0 &= \frac{1}{2}(\hat{\mu}_-^\top \Sigma^{-1}\hat{\mu}_- - \hat{\mu}_+^\top \Sigma^{-1}\hat{\mu}_+) \end{aligned}$$

19.8 Unsupervised Learning

Parzen

$$\hat{p}_n = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}\phi(\frac{x-x_i}{h_n})$$

where $\int \phi(x)dx = 1$

K-NN

$$\hat{p}_n = \frac{1}{V_k} \text{ volume with } k \text{ neighbours}$$

K-means

(clustering = classification)

$$L(\mu) = \sum_{i=1}^{n} \min_{j\in\{1...k\}} \|x_i - \mu_y\|_2^2$$

Lloyd's Heuristic: (1) assign each x_i to closest cluster
(2) recalculate means of clusters.
Iteration over (repeated till stable):

Step 1: $\operatorname{argmin}_c ||x - \mu_c||^2$
Step 2: $\mu_\alpha = \frac{1}{n_\alpha}\sum \vec{x}$

Gaussian Mixture Modeling

Same as Bayes, but class label z unobserved.

$$(\mu^*, \Sigma^*, w^*) = \operatorname{argmin} - \sum_i log \sum_{j=1}^{k} w_j \mathcal{N}(x_i|\mu_i, \Sigma_j)$$

EM Algorithm

Problem: sum within log-term of likelihood.
E-step: expectation: pick clusters for points.
Calculate $\gamma_j^{(t)}(x_i) = \frac{P(c|\theta^j)(x_i|c,\theta^j)}{\sum P(x_i|\theta)}$ for each i and j
M-Step: maximum likelihood: adjust clusters to best fit points.

$$\text{prior}_j^{(t)} = \pi_j^{(t)} \leftarrow \frac{1}{n}\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)$$

$$\mu_j^{(t)} \leftarrow \frac{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)(x_i)}{\sum_{i=1}^{n}\gamma^{(t)}(x_i)}$$

$$\Sigma_j^{(t)} \leftarrow \frac{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)(x_i - \mu_j^{(t)})(x_i - \mu}{\sum_{i=1}^{n}\gamma_j^{(t)}(x_i)}$$

EM for non-gaussian mixture

Problem setting Derive the EM algorithm. The model behind the EM in the form: $p(x) = \sum_{j=1}^{n}\pi_j Pr_{\text{non-gaussian}}$ **Solution**

1. Write log likelihood function:

$$L(X, \{\text{params}Pr_{non-gaussian}u_j s\}) = \sum_{i=1}^{n}\log(p($$

2. Write $\gamma_j(x_i) = p(z_j = j|x_i)$ (prob. that x_i was generated from j^{th} dist. of the mixture).

$$\gamma_j(x_i) = p(z_i = j|x_i) = \frac{p(x_i|z_i = j)p(z_i = j)}{p(x_i)}$$

$$= \frac{p(x_i|z_i = j)p(z_i = j)}{\sum_{k=1}^{n} p(x_i|z_i = k)p(z_i = k)} = \frac{\pi_i L(}{\sum_{k=1}^{n}\pi}$$

3. **To get optimal $\mathbf{u_j}$s**, derive $L(x, u_j s)$ by each param u_j. You get:

$$\nabla x_j L(X, u_j s) = \sum_{i=1}^{n}\frac{1}{Pr_{non-gaussian}} \cdot (\nabla u_j Pr_{non}$$

(possibly replace by $\gamma_j(x_i)$ (above and below) leaving back some factor). Solve for the parameter u_j.

4. **Estimate the π parameters** by Lagrange optimization on log likelihood function and constraint $\lambda\left(\sum_{j=1}^{k}\pi_j - 1\right)$. Put the λ into the formula and find the formula for π_j.

19.9 Hidden-Markov model

State only depends on previous state.
Always given: sequence of symbols $\vec{s} = \{s_1, s_2, \dots s_n\}$

Evaluation (Forward & Backward)

Known: $a_{ij}, e_k(s_t)$
Wanted: $P(X = x_i | S = s_t)$

$$f_l(s_{t+1}) = e_l(s_{t+1}) \sum f_k(s_t) a_{kl}$$
$$b_l(s_t) = e_l(s_t) \sum b_k(s_{t+1}) a_{lk}$$
$$P(\vec{s}) = \sum_k f_k(s_n) a_k \cdot \text{ end}$$
$$P(x_{l,t} | \vec{s}) = \frac{f_l(s_t) b_l(s_t)}{P(\vec{s})}$$

Complexity in time: $\mathcal{O}(|S|^2 \cdot T)$

Decoding (Viterbi)

Known: $a_{ij}, e_k(s_t)$
Wanted: most likely path $\vec{x} = \{x_1, x_2, \dots x_n\}$

$$v_l(s_{t+1}) = \text{max}_k \{v_k(s_t) a_{kl}\}$$
$$\text{ptr}_t(l) = \text{argmax}_k \{v_k(s_t) a_{kl}\}$$

Follow pointers back from end to beginning.
Dynamic, because splittable in sub parts (per time step)
Time: $\mathcal{O}(|S|^2 \cdot T)$ Space $\mathcal{O}(|S| \cdot T)$

Learning (Baum-Welch)

Known: only sequence and sequence space Θ
Wanted: $a_{ij}, e_k(s_t)$ & most likely path $\vec{x} = \{x_1, x_2, \dots x_n\}$

E-step I: $f_k(s_t), b_k(s_t)$ by forward & backward algorithm

E-step II:

$$P(X_t = x_k, X_{t+1} = x_l | \vec{s}, \Theta) = \frac{1}{P(\vec{s})} f_k(s_t) a_{kl} e_l(s_{t+1}) b_l(s_{t+1})$$

$$A_{kl} = \sum_{j=1}^{m} \sum_{t=1}^{n} P(X_t = x_k, X_{t+1} = x_l | \vec{s}, \Theta)$$

M-step :

$$a_{kl} = \frac{A_{kl}}{\sum\limits_{i}^{n} A_{ki}} \text{ and } e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

Complexity: $\mathcal{O}(|S|^2)$ in storage (space)

20 Cheat sheet 2013

20.1 Probability

Probability Rules

Sum Rule	$P(X=x_i)=\sum_{j=1}^{J} p(X=x_i,Y=y_i)$
Product rule	$P(X,Y)=P(Y\|X)P(X)$
Independence	$P(X,Y)=P(X)P(Y)$
Bayes' Rule	$P(Y\|X)=\frac{P(X\|Y)P(Y)}{P(X)}$
Conditional independence	$X \perp Y \| Z$
	$P(X,Y\|Z)=P(X\|Z)P(Y\|Z)$
	$P(X\|Z,Y)=P(X\|Z)$

Expectation

$$\begin{aligned}
E(X) &= \int_{\inf}^{\inf} x p(x) dx \\
\sigma^2(X) &= E(x^2)-E(x)^2 \\
\sigma^2(X) &= \int_x (x-\mu_x)^2 p(x) dx \\
Cov(X,Y) &= \int_x \int_y p(x,y)(x-\mu_x)(y-\mu_y) dx dy
\end{aligned}$$

Gaussian

$$p(X|\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{(x-\mu)^2}{2\sigma^2})$$

Kernels

Requirements: Symmetric ($k(x,y)=k(y,x)$)
positive semi-definite K.

$$\begin{aligned}
k(x,y) &= ak_1(x,y)+bk_2(x,y) \\
k(x,y) &= k_1(x,y)k_2(x,y) \\
k(x,y) &= f(x)f(y) \\
k(x,y) &= k_3(\phi(x),\phi(y))
\end{aligned}$$

Linear	$k(x,y)=$	$x^\top y$
Polynomial	$k(x,y)=$	$(x^\top y+1)^d$
Gaussian RBF	$k(x,y)=$	$\exp(\frac{-\|x-y\|_2^2}{h^2})$
Sigmoid (Neural Net)	$k(x,y)=$	$\tanh(kx^\top y-b)$

20.2 Regression

Linear Regression:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2$$

Closed form solution: $w^*=(x^\top x)^{-1}x^\top y$
Ridge Regression:

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2+\lambda\|w\|_2^2$$

Closed form solution: $w^*=(x^\top x+\lambda I)^{-1}x^\top y$
Lasso Regression (sparse):

$$\min_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2+\lambda\|w\|_1$$

Kernelized Linear Regression:

$$\min_\alpha \|K\alpha-y\|_2^2+\lambda\alpha^\top K\alpha$$

Closed form solution: $\alpha=(K-\lambda I)^{-1}y$

20.3 Classification

0/1 Loss	$w^*=$	$\operatorname{argmin}_w \sum_{i=1}^{n}[y_i \neq sign(w^\top x_i)]$
Perceptron	$w^*=$	$\operatorname{argmin}_w \sum_{i=1}^{n}[\max(0,y_i w^\top x_i)]$

SVM

Primal, constrained:

$$\min_w w^\top w + C\sum_{i=1}^{n}\xi_i, \text{ s.t. } y_i w^\top x_i \geq 1-\xi_i, \xi_i \geq 0$$

Primal, unconstrained:

$$\min_w w^\top w + C\sum_{i=1}^{n}\max(0,1-y_i w^\top x_i) \text{ (hinge loss)}$$

Dual:

$$\max_\alpha \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i^\top x_j, \text{ s.t. } 0 \geq \alpha_i \geq C$$

Dual to primal: $w^*=\sum_{i=1}^{n}\alpha_i^* y_i x_i$, $\alpha_i>0$: support vector.

Kernelized SVM

$$\max_\alpha \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(x_i,x_j), \text{ s.t. } 0 \geq \alpha_i \geq C$$

Classify: $y=sign(\sum_{i=1}^{n}\alpha_i y_i k(x_i,x))$

20.4 Misc

Lagrangian: $f(x,y) s.t. g(x,y)=c$

$$\mathcal{L}(x,y,\gamma)=f(x,y)-\gamma(g(x,y)-c)$$

Parametric learning: model is parameterized with a finite set of parameters, like linear regression, linear SVM, etc.

Nonparametric learning: models grow in complexity with quantity of data: kernel SVM, k-NN, etc.

20.5 Probabilistic Methods:

MLE

Least Squares, Gaussian Noise

$$L(w)=-\log(P(y_1...y_n|x_1...x_n,w))=\frac{n}{2}\log(2\pi\sigma^2)+\sum_{i=1}^{n}\frac{(y_i-w^\top x_i)^2}{2\sigma^2}$$

$$\operatorname{argmax}_w P(y|x,w)=\operatorname{argmin}_w L(w)=\operatorname{argmin}_w \sum_{i=1}^{n}(y_i-w^\top x_i)^2$$

MAP

Ridge regression, Gaussian prior on weights

$$\operatorname{argmax}_w P(w)\prod_i P(y_i|x_i,w)=\operatorname{argmin}_w \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-w^\top x_i)+\frac{1}{2\beta^2}\sum_{i=1}^{n}w_i^2$$

$P(w)$ or $P(\theta)$ - conjugate prior (beta, Gaussian) (posterior same class as prior)
$P(y_i|\theta)$ - likelihood function (binomial, multinomial, Gaussian)
Beta distribution: $P(\theta)=Beta(\theta;\alpha_1,\alpha_2)\propto\theta^{\alpha_1-1}(1-\theta)^{\alpha_2-1}$

Logistic Regression

MLE with Bernoulli noise

$$\begin{aligned}
\text{MLE: } \operatorname{argmin}_w L(w) &= \sum_{i=1}^{n}\log(1+\exp(-y_i w^\top x_i)) \\
\text{MAP: } &+ \left\{\lambda\|w\|_2^2, \lambda\|w\|_1\right\}
\end{aligned}$$

Classification: $P(y|x,\hat{w}=\frac{1}{1+\exp(-y\hat{w}^\top x)}$

Bayesian Decision Theory

$$a^*=\operatorname{argmin}_{a\in A} E_y[C(y,a)|x]$$

Bayesian Model Averaging (BMA)

Ridge regression, but with probabilities

$$\begin{aligned}
P(y|x,D) &= \int P(y|x,w)P(w|D)dw \\
P(w) &= \mathcal{N}(w;0,\sigma_w^2) \\
P(y|w,x) &= \mathcal{N}(y;wx,\sigma_y^2) \\
P(w|x,y) &= \mathcal{N}(w;\mu_{w|y},\sigma_{w|y}^2)
\end{aligned}$$

$$\mu_{w|y} = \frac{xy\sigma_w^2}{x^2\sigma_w^2 + \sigma_y^2}$$

$$\sigma_{w|y}^2 = \frac{\sigma_w^2\sigma_y^2}{x^2\sigma_w^2 + \sigma_y^2}$$

$$\begin{aligned} \text{MAP}P(y'|x', \hat{w}) &= \mathcal{N}(y'; x'\mu_{w|y}, \sigma_y^2) \\ \text{BMAP}P(y'|x', x, y) &= \mathcal{N}(y'; x'\mu_{w|y}, \sigma_y^2 + x'^2\sigma_{w|y}^2) \end{aligned}$$

Bayesian Linear Regression

$$\begin{aligned} \mathcal{N}(x_i, \mu_V, \Sigma_{VV}) &= \frac{1}{\sqrt{(2\pi)^d \Sigma_{VV}}} \exp(-\frac{1}{2}(x-\mu_V)^\top \Sigma_{VV}^{-1}(x-\mu_V)) \\ P(y|x, y_A) &= \mathcal{N}(y; \mu_{y|A}, \sigma_{y|A}^2) \\ \mu_{y|A} &= \sum_{x,A} \Sigma_{AA}^{-1} y_A \\ \Sigma_{VV} &= \beta^2 XX^\top + \sigma^2 I \\ \sigma_{y|A}^2 &= \Sigma_{xx} - \Sigma_{xA}\Sigma_{AA}^{-1}\Sigma_{Ax} \end{aligned}$$

Gaussian Process (Kernelized BLR)

Replace $\Sigma_{VV} = K + \sigma^2 I_n$.

20.6 Active Learning

D-optimality: $x_t = \text{argmax}_{x \in X}\, \sigma_{t-1}^2(x)$ (pick the most uncertain sample)
A-optimality: $x_t = \text{argmax}_{x \in X} \int [\sigma_t^2(x) - \sigma_{t-1}^2(x)]dx$ (pick the sample that'll reduce the variance the most)

20.7 Ensemble Methods

Use combination of simple hypotheses (weak learners) to create one strong learner.

$$f(x) = \sum_{i=1}^n \beta_i h_i(x)$$

Bagging: train weak learners on random subsamples with equal weights.
Boosting: train on all data, but reweigh misclassified samples higher.

Decision Trees

Stumps: partition linearly along 1 axis

$$h(x) = sign(ax_i - t)$$

Decision Tree: recursive tree of stumps, leaves have labels. To train, either label if leaf's data is pure enough, or split data based on score.

Ada Boost

Effectively minimize exponential loss.

$$f^*(x) = \underset{f \in F}{\text{argmin}} \sum_{i=1}^n \exp(-y_i f(x_i))$$

Train m weak learners, greedily selecting each one

$$(\beta_i, h_i) = \underset{\beta, h}{\text{argmin}} \sum_{i=1}^n \exp(-y_i(f_{i-1}(x_j) + \beta h(x_j)))$$

20.8 Generative Methods

Discriminative - estimate $P(y|x)$ - conditional.
Generative - estimate $P(y, x)$ - joint, model data generation.

Naive Bayes

All features independent.

$$P(y|x) = \frac{1}{Z} P(y)P(x|y), Z = \sum_y P(y)P(x|y)$$

$$y = \underset{y'}{\text{argmax}}\, P(y'|x) = \underset{y'}{\text{argmax}}\, \hat{P}(y') \prod_{i=1}^d \hat{P}(x_i|y')$$

Discriminant Function

$$f(x) = \log(\frac{P(y=1|x)}{P(y==1|x)}), y = sign(f(x))$$

Fischer's Linear Discriminant Analysis (LDA)

$$c = 2, p = 0.5, \hat{\Sigma}_- = \hat{\Sigma}_+ = \hat{\Sigma}$$

$$\begin{aligned} y &= sign(w^\top x + w_0) \\ w &= \hat{\Sigma}^{-1}(\hat{\mu}_+ - \hat{\mu}_-) \\ w_0 &= \frac{1}{2}(\hat{\mu}_-^\top \Sigma^{-1}\hat{\mu}_- - \hat{\mu}_+^\top \Sigma^{-1}\hat{\mu}_+) \end{aligned}$$

20.9 Unsupervised Learning

K-means

(clustering = classification)

$$L(\mu) = \sum_{i=1}^n \min_{j \in \{1 \ldots k\}} \|x_i - \mu_y\|_2^2$$

Lloyd's Heuristic: (1) assign each x_i to closest cluster
(2) recalculate means of clusters.

Gaussian Mixture Modeling

Same as Bayes, but class label z unobserved.

$$(\mu^*, \Sigma^*, w^*) = \text{argmin} - \sum_i log \sum_{j=1}^k w_j \mathcal{N}(x_i|\mu_i, \Sigma_j)$$

EM Algorithm

E-step: expectation: pick clusters for points. Calculate $\gamma_j^{(t)}(x_i)$ for each i and j
M-Step: maximum likelihood: adjust clusters to best fit points.

$$\begin{aligned} \omega_j^{(t)} &\leftarrow \frac{1}{n}\sum_{i=1}^n \gamma_j^{(t)}(x_i) \\ \mu_j^{(t)} &\leftarrow \frac{\sum_{i=1}^n \gamma_j^{(t)}(x_i)(x_i)}{\sum_{i=1}^n \gamma^{(t)}(x_i)} \\ \Sigma_j^{(t)} &\leftarrow \frac{\sum_{i=1}^n \gamma_j^{(t)}(x_i)(x_i - \mu_j^{(t)})(x_i - \mu_j^{(t)})^\top}{\sum_{i=1}^n \gamma_j^{(t)}(x_i)} \end{aligned}$$

PCA

(dimensional reduction = regression)

$$\Sigma = \frac{1}{n}\sum_{i=1}^n x_i x_j^\top, \ \ \mu = \frac{1}{n}\sum_{i=1}^n x_i = 0$$

$$(W, z_1, \ldots, z_n) = \text{argmin} \sum_{i=1}^n \|Wz_i - x_i\|_2^2$$

W is orthogonal, $W = (v_1|\ldots|v_k)$ and $z_i = w^\top x_i$

$$\Sigma = \sum_{i=1}^d \lambda_i v_i v_i^\top \quad \lambda_1 \geq \ldots \geq \lambda_d \geq 0$$

Kernel PCA

$$\alpha_i^* = \arg \max_{\alpha^\top K\alpha = 1} = \alpha^\top K^\top K\alpha$$

$$\alpha^{(i)} = \frac{1}{\sqrt{\lambda_i}} \frac{v_i}{\|v_i\|_2}, \ \ K = \sum_{i=1}^n \lambda_i v_i v_i^\top$$

We could use colors in the cheat sheet to improve searchability

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Introduction to Machine Learning

Autumn Semester 2012

Prof. J.M. Buhmann

Final Exam

February 6th, 2013

First and Last name: ______________________________

ETH number: ______________________________

Signature: ______________________________

General Remarks

- You have 2 hours for the exam. There are five sections, each of which is worth 20 points. Scoring 100 points guarantees you a grade of six. In two sections you will find bonus questions, worth together 10 points. The bonus questions are a bit more difficult, we suggest you leave them to the end.
- Write your answers directly on the exam sheets. At the end of the exam you will find supplementary sheets, feel free to seperate them from the exam. If you submit the supplementary sheets, put your name and ETH number on top of each.
- Answer the questions in English. Do not use a pencil or red color pen.
- You may provide at most one valid answer per question. Invalid solutions must be canceled out clearly.

	Topic	Max. Points	Points	Signature
1	Assorted Questions	20		
2	Bayesian Inf., MAP and ML	$20+5$		
3	Supervised Learning	20		
4	Kernelized Ridge Regression	20		
5	Unsupervised Learning	$20+5$		
Total		$100+10$		

Grade: . . . .

Question 1: Assorted Questions (20 pts.)

1. Figure 1 shows 4 times the same binary classification dataset.

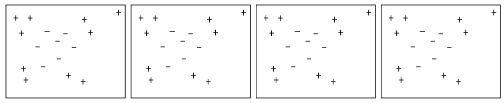


Figure 1: 4 times the same dataset

(a) Cross all of the following algorithms/classifiers, which can achieve zero training error on this dataset.
   - ☐ Perceptron
   - ☐ Decision tree
   - ☐ SVM with Gaussian kernel
   - ☐ Ensemble of linear kernel SVMs

(b) For each of the methods that can achieve zero training error, qualitatively depict a possible decision boundary (having zero error) in one of the plots of the dataset in Figure 1. Indicate which method belongs to which plot.

4 pts.

2. Let $\mathcal{F}$ be an hypothesis class for a binary classification task and f be a randomly chosen prediction function, having a training error of 0.65, on some dataset $\mathcal{S}$. Explain how to use f to obtain $\tilde{f}$, a prediction function which is **guaranteed** to have a smaller training error than f.

2 pts.

3. We consider applying the Viterbi algorithm to estimate a trajectory of an HMM with $|S|$ states over T time points. Assume that the number of states $|S|$ grows as $O(\sqrt{T})$. What is worst-case asymptotic computational complexity of the algorithm (as a function of T)?

3 pts.

4. For each of the following statements, circle the correct answer below.

(a) The number of nodes in a decision tree is bounded by the number of features.
True/False

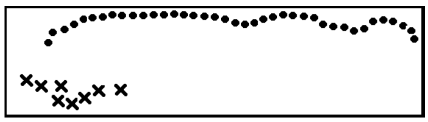
(b) Boosting classifiers can in principle be done in a parallel manner.
True/False

(c) Can the Baum-Welch algorithm be considered a type of Expectation Maximization procedure?
Always/Never/Only Sometimes

4 pts.

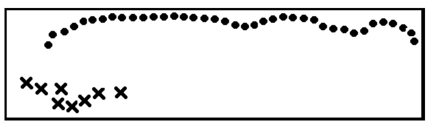
5. The following figures show a dataset of 48 objects from two different sources, represented by different symbols.

(a) Sketch the optimal K-means solution on this dataset, for $K = 2$. Draw the centers as well as the clusters.



3 pts.

(b) Consider reducing the dimensionality of the data to 1 before finding a 2-means solution. Propose an appropriate dimension reduction by drawing a projection line through the estimated center of mass of the data.
Now sketch the optimal 2-means solution on the dimension reduced data.



4 pts.

Question 2: **Bayesian Inference, MAP and ML estimation (20 pts.)**

1. Let $x_1, \dots x_n \in \mathbb{R}$ be a dataset consisting of n samples which are assumed to be drawn iid from a normal distribution $\mathcal{N}(x|\mu, \sigma^2)$ in which the variance σ^2 and the mean μ are unknown.
   Demonstrate that the maximum likelihood estimation of μ can be performed without knowing the maximum likelihood estimate of the variance.

8 pts.

2. Consider the following Maximum a Posteriori estimation task. The likelihood function is the normal distribution with unknown mean μ and variance $\sigma^2 = 1$. Let μ_0 and σ_0^2 respectively denote the mean and variance of the prior, and recall that the posterior has mean and variance respectively given by

$$\begin{aligned} \mu_n &= \frac{\sigma^2}{n\sigma_0^2 + \sigma^2}\mu_0 + \frac{n\sigma_0^2}{n\sigma_0^2 + \sigma^2}\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right), \\ \sigma_n^2 &= \left(\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}\right)^{-1}. \end{aligned} \tag{1}$$

(a) Show how to derive the above posterior formula for μ, from the prior and the likelihood function.

8 pts.

(b) Let $\sigma_0^2 = \pi$ and $x_i = 1$ for $i = 1, \dots, 5$. What is the numerical value of the maximum a posteriori estimate of μ?

4 pts.

3. **Bonus question:** Let μ_{ML} and μ_{MAP} respectively denote the maximum likelihood estimator and the maximum a posteriori estimator for μ. Calculate the following:

$$\lim_{\sigma_0^2 \to \infty} \mathbb{E}[\mu_{\mathsf{MAP}}] - \mathbb{E}[\mu_{\mathsf{ML}}] \stackrel{?}{=}$$

5 pts.

Question 3: **Supervised Learning**

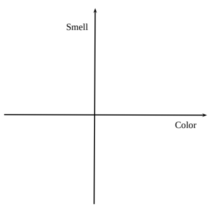
This question is concerned with classification of watermelons into 'good' watermelons $(+1)$ and 'bad' ones (-1). Watermelons can be distinguished based only on their color and smell. Let $\mathcal{H}$ be the class of all *circles* in $\mathbb{R}^2$. We associate a classification rule with each $h \in \mathcal{H}$: the interior of the circle is classified as 'good' and outside of the circle is 'bad'.
Given $\{(\mathbf{x}_i, y_i)_{i=1}^n \, \mathbf{x}_i \in \mathbb{R}^2, y_i \in \{1, -1\}\}$, a labeled sample of watermelons, we used the following criterion for the parameters of $h^* \in \mathcal{H}$:

$$w_1^*, w_2^*, r^* = \underset{w_1, w_2, r}{\operatorname{argmin}} \sum_{i=1}^{n} \exp(-y_i[r^2 - ((x_{i1} - w_1)^2 + (x_{i2} - w_2)^2)]) \tag{2}$$

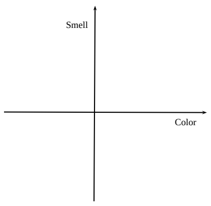
We then sold h^* to Migros as part of a watermelon test kit.
Unfortunately h^* did not meet the expectations, it misclassified a non-negligible proportion of the watermelons used at test time.

1. For each of the following additional assumptions:
   (a) Give a possible explanation for h^* performing poorly
   (b) Draw a training set, the prediction function h^*, and the true distribution (if needed) that demonstrate your explanation.

Additional assumption: h^* had a very low training error



Additional assumption: The sample size was large, and h^* had training error of ~ 0.4



8 pts.

2. Suggest a way to measure the empirical variance of the classifier h^*, given that we are out of budget for obtaining more watermelon samples.

3 pts.

Figure 2 depicts the dataset we had, and h^* that we got using equation (2). To improve h^* we decided to add a regularizing term, the new criterion will be

$$\underset{w_1,w_2,r}{\operatorname{argmin}} \sum_{i=1}^{n} \exp(-y_i[r^2-((x_{i1}-w_1)^2+(x_{i2}-w_2)^2)])+\lambda\Omega(w_1,w_2,r) \tag{3}$$

Where $\Omega(w_1, w_2, r)$ is the regularizer. We ask you to suggest a suitable regularizer.

4. (a) Draw on Figure 2 the regularized solution you envision.
   (b) Write down the mathematical term of the regularizer. Explain your answer.

   $\Omega(w_1, w_2, r) =$

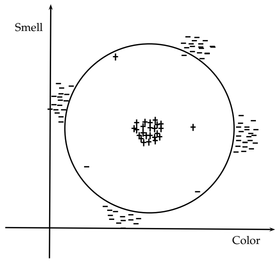


Figure 2: Watermelons dataset and h^*

5 pts.

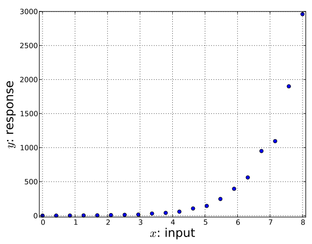
5. Assume that the true distribution of watermelons consists of high density regions visible in Figure 2, plus sparse outliers. Explain what happens to the variance of h^* as we increase λ compared to some starting value $\lambda_0 > 0$.

4 pts.

Question 4: Kernelized Ridge Regression (20 pts.)

Recall the regression setting: Given input vectors $\mathbf{x}_i$, and output (response) variables y_i, the goal is to find a functional relation between them, often expressed with a weight vector $\mathbf{w}$ and bias b.

1. Below is a dataset with one dimensional input variables x, and response variable y. Your task is to find a kernel function $K(x_i, x_j)$, such that you can use a linear regression method in the kernel space.



$K(x_i, x_j) = \ldots$ **3 pts.**

2. You will now derive a kernelized version of ridge regression by introducing a feature transform $\mathbf{\Phi}(\mathbf{x}_i)$. This should allow a non-linear regression solution, for datasets such as the one depicted above.
   Recall the formulation of ridge regression as an optimization problem:

$$\min_{\mathbf{w},b} \quad \sum_i (y_i - \mathbf{w}^\top \mathbf{x}_i - b)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \tag{4}$$

   (a) We replace the inputs $\mathbf{x}_i$ in Equation (4) with the vectors of the features in the kernel space and rewrite the problem as a

constrained optimization problem by introducing the new variables ξ_i. Write down the equality constraint in Equation (6).

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \sum_i \xi_i^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \tag{5}$$

$$\text{s.t.} \qquad \xi_i = \ldots \tag{6}$$

2 pts.

   (b) Write down the Lagrangian of this new optimization problem using $\boldsymbol{\alpha}$ as the dual variable.

   (c) Derive the dual optimization problem.

8 pts.

3. Express the dual problem in terms of the kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$.

3 pts.

4. Given the optimal solution of the dual problem $\boldsymbol{\alpha}^\star$ and a new point $\mathbf{x}_k$, write down the equation to compute y_k.

4 pts.

Question 5: **Unsupervised Learning (20 pts.)**

1. In this section we study non-parametric density estimation of an arbitrary point x. We consider some small region $\mathcal{R}$ containing x. In the class we have seen the following generic formula for density estimation:
$$p(x) = \frac{K}{nV},$$
where K denotes the number of data points falling inside the region $\mathcal{R}$ and V shows the volume of the region. n is the number of data points in the sample set $\mathcal{S} = \{x_1, ..., x_n\}$.

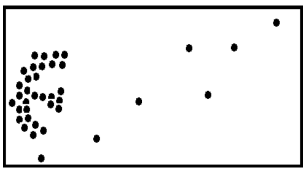
   (a) Consider the following Gaussian distribution to be used as a Parzen window function:
$$\phi(x - x_j) = \frac{1}{(2\pi)^{1/2}} \exp\left(-\frac{(x - x_j)^2}{2}\right). \tag{7}$$
What are K and V for this window function?

4 pts.

   (b) This particular choice of a window function leads to underfitting. Add a parameter to increase the model complexity.

2 pts.

   (c) Consider the following sample set. Which of the density estimation methods would you choose? Window-based (Eq. (7)) or K-nearest neighbor? Explain your answer.



3 pts.

   (d) For a general Parzen window function prove that it provides a probability distribution.

4 pts.

2. We consider a mixture of K poisson distributions and perform the Expectation-Maximization (EM) algorithm to compute the unknown parameters. The log-likelihood function of n independent objects for mixture of K Poisson distribution is defined as:
$$P(x; \lambda) = \sum_{i=1}^{n} \log \sum_{c=1}^{K} \pi_c f(x_i; \lambda_c)$$
where π_c's are the mixture weights and λ_c's are the parameters of K Poisson distributions. $f(x_i; \lambda_c)$ is defined as:
$$f(x; \lambda_c) = \frac{\lambda^x e^{-\lambda_c}}{x!}.$$

   (a) Introduce the latent indicator variables necessary for maximizing the log-likelihood function.

2 pts.

   (b) Calculate the expectation of the latent variables. Provide a Bayesian interpretation for your answer.

5 pts.

(c) **Bonus question:** Assume the expectations of the latent variables are given. Calculate the unknown parameters λ_c's. Write down the details of your calculations.

5 pts.

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Introduction to Machine Learning
Autumn Semester 2012

Prof. J.M. Buhmann

Final Exam
February 6th, 2013

First and Last name: ______________________

ETH number: ______________________

Signature: ______________________

General Remarks

- You have 2 hours for the exam. There are five sections, each of which is worth 20 points. Scoring 100 points guarantees you a grade of six. In two sections you will find bonus questions, worth together 10 points. The bonus questions are a bit more difficult, we suggest you leave them to the end.
- Write your answers directly on the exam sheets. At the end of the exam you will find supplementary sheets, feel free to seperate them from the exam. If you submit the supplementary sheets, put your name and ETH number on top of each.
- Answer the questions in English. Do not use a pencil or red color pen.
- You may provide at most one valid answer per question. Invalid solutions must be canceled out clearly.

	Topic	Max. Points	Points	Signature
1	Assorted Questions	20		
2	Bayesian Inf., MAP and ML	20 + 5		
3	Supervised Learning	20		
4	Kernelized Ridge Regression	20		
5	Unsupervised Learning	20 + 5		
Total		100 + 10		

Grade: ....

Question 1: Assorted Questions (20 pts.)

1. Figure 1 shows 4 times the same binary classification dataset.

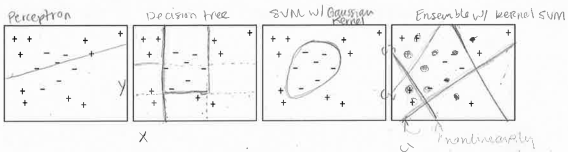


Figure 1: 4 times the same dataset

(a) Cross all of the following algorithms/classifiers, which can achieve zero training error on this dataset.

☐ Perceptron no data needs to be linearly separable.

☒ Decision tree

☒ SVM with Gaussian kernel

☒ Ensemble of linear kernel SVMs

(b) For each of the methods that can achieve zero training error, qualitatively depict a possible decision boundary (having zero error) in one of the plots of the dataset in Figure 1. Indicate which method belongs to which plot.

4 pts.

2. Let $\mathcal{F}$ be an hypothesis class for a binary classification task and f be a randomly chosen prediction function, having a training error of 0.65, on some dataset $\mathcal{S}$. Explain how to use f to obtain $\tilde{f}$, a prediction function which is **guaranteed** to have a smaller training error than f.

prediction 0/1

f : prediction fxn = Train error 0.65

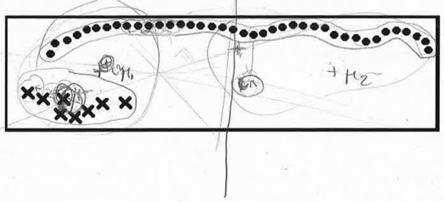
$f_R \to \hat{f}_{pred}$ regularized form of f and perform CV.

2 pts.

f can be evaluated by selecting a cost fxn and performing a cross validation on the number of predicted fxns to choose the best parameters.
OR in ensemble methods update the weight of each classifier such as boosting.

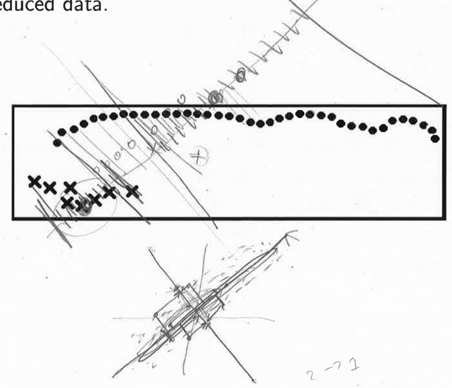
5. The following figures show a dataset of 48 objects from two different sources, represented by different symbols.

(a) Sketch the optimal K-means solution on this dataset, for $K = 2$. Draw the centers as well as the clusters.



3 pts.

(b) Consider reducing the dimensionality of the data to 1 before finding a 2-means solution. Propose an appropriate dimension reduction by drawing a projection line through the estimated center of mass of the data.
Now sketch the optimal 2-means solution on the dimension reduced data.



2 → 1

4 pts.

(a) Show how to derive the above posterior formula for μ, from the prior and the likelihood function.

Defn of MAP $\hat{\mu}_{MAP} = \arg\max_{\mu} P(x_1,\dots,x_n \mid \mu,\sigma)\, p(\mu \mid \mu_0,\sigma_0)$ ← PRIOR

$\div p(x_1,\dots,x_n \mid \sigma, \mu_0,\sigma_0)$ ← can throw away because doesn't depend on μ.

MAP is a linear combination of ML

$\hat{\mu}_{MAP} =$ scale accordingly $(\hat{\mu}_{ML})$ + constant

$= \arg\max_{\mu} p(x_1,\dots,x_n \mid \mu,\sigma)\cdot p(\mu \mid \mu_0,\sigma_0)$

multiplying 2 gaussians together. likelihood

$= \arg\max_{\mu} \left[\prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i-\mu)^2}{2\sigma^2}\right)\right]\cdot \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{(\mu-\mu_0)^2}{2\sigma_0^2}\right)$

scaling doesn't affect max / $\hat{\mu}$ value

$= \arg\max_{\mu} \exp\left(-\frac{1}{2}\left[\left[\sum_i^N \frac{(x_i-\mu)^2}{\sigma^2}\right] + \frac{(\mu-\mu_0)^2}{\sigma_0^2}\right]\right) = \arg\min_{\mu} \left[\sum_i^N \frac{(x_i-\mu)^2}{\sigma^2}\right] + \frac{(\mu-\mu_0)^2}{\sigma_0^2}$

$\hat{\mu}_{MAP}$ such that $\frac{\partial}{\partial \hat{\mu}_{map}}\left(\left[\sum_i^n \frac{(x_i-\mu)^2}{\sigma^2}\right] + \frac{(\hat{\mu}_{ML}-\mu_0)^2}{\sigma_0^2}\right) = 0$

$= 2\left[\sum_1^N -\frac{x_i}{\sigma^2} - \frac{\mu_0}{\sigma_0^2} + \hat{\mu}_{MAP}\left(\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}\right)\right] = 0 \Rightarrow \hat{\mu}_{MAP} = \frac{n\sigma^2}{\frac{n}{\sigma^2}+\frac{1}{\sigma^2}}\left(\frac{1}{n}\sum x_i\right) + \frac{\sigma^2}{n\sigma_0^2+\sigma^2}\mu_0$

8 pts.

(b) Let $\sigma_0^2 = \pi$ and $x_i = 1$ for $i = 1,\dots,5$. What is the numerical value of the maximum a posteriori estimate of μ?

$\hat{\mu}_{MAP} = \frac{5\pi + \mu_0}{5\pi + 1}$

4 pts.

Question 3: Supervised Learning

This question is concerned with classification of watermelons into 'good' watermelons (+1) and 'bad' ones (−1). Watermelons can be distinguished based only on their color and smell. Let $\mathcal{H}$ be the class of all *circles* in $\mathbb{R}^2$. We associate a classification rule with each $h \in \mathcal{H}$: the interior of the circle is classified as 'good' and outside of the circle is 'bad'.

Given $\{(\mathbf{x}_i, y_i)_{i=1}^n \ \mathbf{x}_i \in \mathbb{R}^2, y_i \in \{1, -1\}\}$, a labeled sample of watermelons, we used the following criterion for the parameters of $h^* \in \mathcal{H}$:

cost classifiers

$$w_1^*, w_2^*, r^* = \underset{w_1, w_2, r}{\operatorname{argmin}} \sum_{i=1}^{n} \exp(-y_i[r^2 - ((x_{i1} - w_1)^2 + (x_{i2} - w_2)^2)]) \qquad (2)$$

center radius

$-y_i\,(r^2 - (x_i - w_i)^2)$

Trained classifier

We then sold h^* to Migros as part of a watermelon test kit. Unfortunately h^* did not meet the expectations, it misclassified a non-negligible proportion of the watermelons used at test time.

1. For each of the following additional assumptions:
   (a) Give a possible explanation for h^* performing poorly
   (b) Draw a training set, the prediction function h^*, and the true distribution (if needed) that demonstrate your explanation.

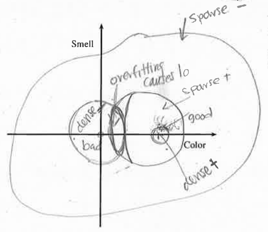
① overfitted h^*

Know that the distribution of the watermelons is not 50/50 and you have to consider the

Additional assumption: h^* had a very low training error

sparse − additional assumption that setting 105/39 and give you a dense solution for good

Added noise to true distribution



4. (a) Draw on Figure 2 the regularized solution you envision.
   (b) Write down the mathematical term of the regularizer. Explain your answer.

$\Omega(w_1, w_2, r) =$ R^2 in L2-norm.

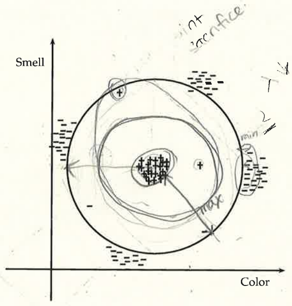


Figure 2: Watermelons dataset and h^*

5 pts.

5. Assume that the true distribution of watermelons consists of high density regions visible in Figure 2, plus sparse outliers. Explain what happens to the variance of h^* as we increase λ compared to some starting value $\lambda_0 > 0$.

The size the circle will decrease.

$\lambda \uparrow$

variance ↓ 150

4 pts.

constrained optimization problem by introducing the new variables ξ_i. Write down the equality constraint in Equation (6).

primal

$$\min_{\mathbf{w}, b, \xi} \quad \sum_i \xi_i^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2 \qquad (5)$$

$$\text{s.t.} \quad \xi_i = \dots \qquad (6)$$

2 pts.

(b) Write down the Lagrangian of this new optimization problem using $\boldsymbol{\alpha}$ as the dual variable.

objective of lagrangian is to minimize the problem.

$\max \ f(\vec{x})$

s.t. $g(\vec{x}) = 0$ $\quad L(\vec{x}, \lambda) = f(\vec{x}) + \lambda g(\vec{x})$

$\vec{x} \in \mathbb{R}^D$

$\nabla g(\vec{x}) \perp \quad g(\vec{x} + \vec{\varepsilon}) \simeq g(\vec{x}) + \vec{\varepsilon}^T \nabla g(x)^T \quad \|\vec{\varepsilon}\| \to 0$

$\nabla f(x^*)$ $\quad \vec{x} + \vec{\varepsilon}$ $\quad \nabla g(\vec{x})$

solve the larangian

$\frac{\partial L}{\partial x} = 0$

$\frac{\partial L}{\partial \lambda} = g(x) = 0$

(c) Derive the dual optimization problem.

objective fxn

$L(\vec{\alpha}, \vec{w}, b, \vec{\xi}) = \sum_i \xi_i^2 + \frac{\lambda}{2} \|\vec{w}\|^2 + \sum_i \alpha_i (y_i - \vec{w}^T \vec{\phi}_i - b - \xi_i)$

$\vec{w} = \binom{b}{\vec{w}} \quad \vec{x} = \binom{1}{\vec{x}}$

$\frac{\partial L}{\partial \vec{w}} = -\sum_i \alpha_i \vec{\phi}_i + \lambda \vec{w} = 0 \Rightarrow \boxed{\vec{w} = \frac{\sum_i \alpha_i \vec{\phi}_i}{\lambda}}$ This is why you derive on α.

$\frac{\partial L}{\partial \xi_i} = 2\xi_i - \alpha_i = 0 \Rightarrow \boxed{\xi_i = \frac{\alpha_i}{2}}$

$\|\vec{w}\|^2 = \frac{\lambda}{2} \cdot \frac{1}{\lambda^2} \|\sum_i \alpha_i \vec{\phi}_i\|^2 = \sum_{i,j} \alpha_i \alpha_j \vec{\phi}_i^T \vec{\phi}_j$

You want to get the objective fxn on alpha.

$L(\vec{\alpha}) = -\frac{1}{4} \sum_i \alpha_i^2 + \sum_i \alpha_i y_i - \frac{1}{2\lambda} \sum_{i,j} \alpha_i \alpha_j \vec{\phi}_i^T \vec{\phi}_j$, $\min_{\vec{\alpha}} L(\vec{\alpha})$

8 pts.

Question 5: Unsupervised Learning (20 pts.)

1. In this section we study non-parametric density estimation of an arbitrary point x. We consider some small region $\mathcal{R}$ containing x. In the class we have seen the following generic formula for density estimation:

$$p(x) = \frac{K}{nV},$$

where K denotes the number of data points falling inside the region $\mathcal{R}$ and V shows the volume of the region. n is the number of data points in the sample set $\mathcal{S} = \{x_1, ..., x_n\}$.

(a) Consider the following Gaussian distribution to be used as a Parzen window function:

$$\phi(x - x_j) = \frac{1}{(2\pi)^{1/2}} \exp\left(-\frac{(x - x_j)^2}{2}\right) \qquad (7)$$

What are K and V for this window function?

$K = \sum_{i=1}^{n} \phi(x - x_j)$

$V = 1$ ← because a probability distribution

4 pts.

(b) This particular choice of a window function leads to underfitting. Add a parameter to increase the model complexity.

Add variance parameter. $= h$, controls smoothness

$\phi\left(\frac{(x - x_j)^2}{h}\right) = \frac{1}{(2\pi)^{\frac{1}{2}} h} \exp\left(-\frac{(x - x_j)^2}{2h^2}\right)$

2 pts.

2. We consider a mixture of K poisson distributions and perform the Expectation-Maximization (EM) algorithm to compute the unknown parameters. The log-likelihood function of n independent objects for mixture of K Poisson distribution is defined as:

$$P(x; \lambda) = \sum_{i=1}^{n} \log \sum_{c=1}^{K} \pi_c f(x_i; \lambda_c)$$

where π_c's are the mixture weights and λ_c's are the parameters of K Poisson distributions. $f(x_i; \lambda_c)$ is defined as:

$1 \cdot k \cdot K$

$$f(x; \lambda_k) = \frac{\lambda_k^x e^{-\lambda_k}}{x!}.$$

(a) Introduce the latent indicator variables necessary for maximizing the log-likelihood function.

$$\log P(x|\vec{\lambda}, \vec{\pi}) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k f(x_n|\lambda_k)$$

$Z \in \mathbb{R}^{N \times K}$, each row is a vector for a point

$$Z_{nk} = \begin{cases} 1 & : \text{component } k \text{ is responsible for generated } x_n \\ 0 & : \text{ow.} \end{cases}$$

2 pts.

(b) Calculate the expectation of the latent variables. Provide a Bayesian interpretation for your answer.

Conditional expectation

E-step: estimate posterior distribution of data

$$p(Z|x, \Theta^{old}) = E_Z(Z|\dots)$$

M-step: $\Theta^{new} = \arg\max_{\Theta} Q(\Theta, \Theta^{old})$

joint distribution

$$Q(\Theta, \Theta^{old}) = \sum_{Z} p(Z|x, \Theta^{old}) \cdot \log P(X, Z|\Theta);$$ complete data log-likelihood

$$E[Z_{nk}] = P(Z_{nk} = 1|\vec{x}_n) * 1 = \frac{\overbrace{P(Z_{nk}=1)}^{=\pi_k} \cdot \overbrace{P(\vec{x}_n | Z_{nk}=1)}^{= p(x_n|\lambda_k)}}{\sum_{Z_1} P(x_n, Z_n)} = \frac{}{\sum_{j=1}^{k} \pi_j \cdot f(x_n|\lambda_j)}$$

Responsibility $\gamma(Z_{nk})$

$$P(\vec{Z}_n|\vec{\pi}) = \prod_{k=1}^{K} \pi_k^{Z_{nk}}$$

$$\Rightarrow P(x_n|Z_n, \lambda_k) = \prod_{k=1}^{K} p(x_n|\lambda_k)^{Z_{nk}}$$

5 pts.

Supplementary Sheet

NO 'fixed "right solution"', as in real life. ⇒ The accuracy is the only goal.

Google solution for insights and do it better ⇒ do not reinvent the wheel.

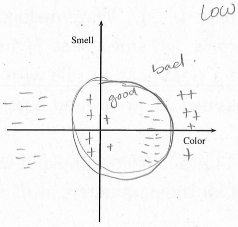
CV scheme is really important

SVM ⇒ assume metric space behind data.

Q/A session before exam.

Additional assumption: The sample size was large, and h^* had training error of ~ 0.4

Low training ERROR



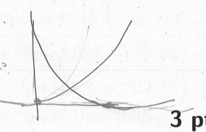
circle would never capture the structure of bimodial data

10 sets of N fruit and 4/10 will be miss classified

8 pts.

2. Suggest a way to measure the empirical variance of the classifier h^*, given that we are out of budget for obtaining more watermelon samples. you want to penalize the size of the circle.

$\Omega(w_1, w_2, r) = r^2$



arg min $(\exp^{-\alpha})$

3 pts.

Figure 2 depicts the dataset we had, and h^* that we got using equation (2). To improve h^* we decided to add a regularizing term, the new criterion will be

$$\operatorname*{argmin}_{w_1,w_2,r} \sum_{i=1}^{n} \exp(-y_i[r^2 - ((x_{i1}-w_1)^2 + (x_{i2}-w_2)^2)]) + \lambda\Omega(w_1, w_2, r) \quad (3)$$

Where $\Omega(w_1, w_2, r)$ is the regularizer. We ask you to suggest a suitable regularizer.

4. (a) Draw on Figure 2 the regularized solution you envision.
   (b) Write down the mathematical term of the regularizer. Explain your answer.

$\Omega(w_1, w_2, r) =$ R^2 in L2-norm.

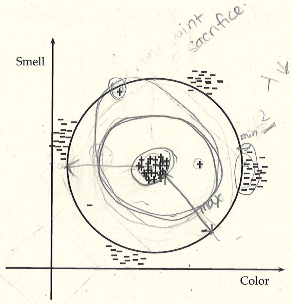


Figure 2: Watermelons dataset and h^*

5 pts.

5. Assume that the true distribution of watermelons consists of high density regions visible in Figure 2, plus sparse outliers. Explain what happens to the variance of h^* as we increase λ compared to some starting value $\lambda_0 > 0$.

The size the circle will decrease. $\lambda\downarrow$

variance $\downarrow$ 150

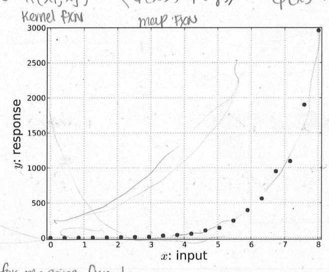
4 pts.

Question 4: **Kernelized Ridge Regression (20 pts.)**

Recall the regression setting: Given input vectors $\mathbf{x}_i$, and output (response) variables y_i, the goal is to find a functional relation between them, often expressed with a weight vector $\mathbf{w}$ and bias b.

1. Below is a dataset with one dimensional input variables x, and response variable y. Your task is to find a kernel function $K(x_i, x_j)$, such that you can use a linear regression method in the kernel space. DEFN: $K(\vec{x}_i,\vec{x}_j) = \langle \phi(\vec{x}), \phi(\vec{y}) \rangle$ $\quad \phi(x) \xrightarrow{map} y$

Kernel fxn map fxn



$\phi(\vec{x}) = \log(\vec{x})$

$K(\cdot) = \log(\vec{x}_i)\cdot\log(\vec{x}_j)$

relationship is exponential, or polynomial

Kernel method used for mapping fxn to make data linearly separable.

$K(x_i, x_j) = \ldots$ **3 pts.**

2. You will now derive a kernelized version of ridge regression by introducing a feature transform $\Phi(\mathbf{x}_i)$. This should allow a non-linear regression solution, for datasets such as the one depicted above.
   Recall the formulation of ridge regression as an optimization problem:

$$\min_{\mathbf{w},b} \sum_i (y_i - \mathbf{w}^\top \mathbf{x}_i - b)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \quad (4)$$

ξ_i^2

   (a) We replace the inputs $\mathbf{x}_i$ in Equation (4) with the vectors of the features in the kernel space and rewrite the problem as a

$\xi_i = y_i - w^T\phi(x_i) - b$ $\quad i = 1 \ldots n$

12 minimization problem.

constrained optimization problem by introducing the new variables ξ_i. Write down the equality constraint in Equation (6).

primal

$$\min_{\mathbf{w},b,\xi} \sum_i \xi_i^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \quad (5)$$

$$\text{s.t.} \qquad \xi_i = \ldots \quad (6)$$

2 pts.

(b) Write down the Lagrangian of this new optimization problem using $\boldsymbol{\alpha}$ as the dual variable.

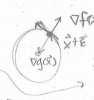
objective of Lagrangian is to minimize the problem.

max $f(x)$

s.t. $g(\vec{x}) = 0$ $\quad L(\vec{x},\lambda) = f(\vec{x}) + \lambda g(\vec{x})$

$\vec{x} \in \mathbb{R}^D$

$\nabla g(\vec{x}) \perp$ $g(\vec{x}+\vec{\epsilon}) \simeq g(\vec{x}) + \vec{\epsilon}^T \nabla g(\vec{x})^T$ $\|\vec{\epsilon}\| \to 0$



solve the larangain

$\frac{\partial L}{\partial x} = 0$

$\frac{\partial L}{\partial \lambda} = g(x) = 0$

(c) Derive the dual optimization problem.

objective fxn

$L(\vec{\alpha}, \vec{w}, b, \vec{\xi}) = \sum_i \xi_i^2 + \frac{\lambda}{2}\|\vec{w}\|^2 + \sum_i \alpha_i (y_i - \vec{w}^T\phi_i - b - \xi_i)$

$\vec{w} = \binom{b}{\vec{w}}$ $\quad \vec{x} = \binom{1}{\vec{x}}$

$\frac{\partial L}{\partial \vec{w}} = -\sum_i \alpha_i \vec{\phi}_i + \lambda \vec{w} = 0 \Rightarrow \boxed{\vec{w} = \frac{\sum_i \alpha_i \vec{\phi}_i}{\lambda}}$ This is why you derive on α.

$\frac{\partial L}{\partial \xi_i} = 2\xi_i - \alpha_i = 0 \Rightarrow \boxed{\xi_i = \frac{\alpha_i}{2}}$

$\|\vec{w}\|^2 = \frac{\lambda}{2}\cdot\frac{1}{\lambda^2}\|\sum_i \alpha_i \vec{\phi}_i\|^2 = \sum_{ij} \alpha_i \alpha_j \vec{\phi}_i^T \vec{\phi}_j$

You want to get the objective fxn on alpha

$L(\vec{\alpha}) = -\frac{1}{4}\sum_i \alpha_i^2 + \sum_i \alpha_i y_i - \frac{1}{2\lambda}\sum_{ij} \alpha_i \alpha_j \vec{\phi}_i^T \vec{\phi}_j$, $\quad \min_{\vec{\alpha}} L(\vec{\alpha})$

8 pts.

3. Express the dual problem in terms of the kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$.

$K(\vec{x}_i, \vec{x}_j) = \Phi_i^T \Phi_j$

3 pts.

4. Given the optimal solution of the dual problem $\boldsymbol{\alpha}^*$ and a new point $\mathbf{x}_k$, write down the equation to compute y_k.

$y_k = \vec{w}^T \vec{\phi}(\vec{x}_k)$

plug in $\vec{w} = \frac{\sum_i \alpha_i \vec{\phi}_i}{\lambda}$

$= \lambda \sum_i \alpha_i^* \vec{\phi}_i^T \vec{\phi}_k = \frac{1}{\lambda} \sum_i \alpha_i^* K(\vec{x}_i, \vec{x}_j)$

4 pts.

Question 5: **Unsupervised Learning (20 pts.)**

1. In this section we study non-parametric density estimation of an arbitrary point x. We consider some small region $\mathcal{R}$ containing x. In the class we have seen the following generic formula for density estimation:

$$p(x) = \frac{K}{nV},$$

where K denotes the number of data points falling inside the region $\mathcal{R}$ and V shows the volume of the region. n is the number of data points in the sample set $\mathcal{S} = \{x_1, ..., x_n\}$.

(a) Consider the following Gaussian distribution to be used as a Parzen window function:

$$\phi(x - x_j) = \frac{1}{(2\pi)^{1/2}} \exp\left(-\frac{(x - x_j)^2}{2}\right) \tag{7}$$

What are K and V for this window function?

$K = \sum_{i=1}^{n} \phi(x - x_j)$

$V = 1$ ← Because a probability distribution

4 pts.

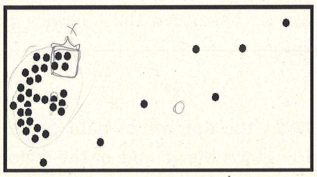
(b) This particular choice of a window function leads to underfitting. Add a parameter to increase the model complexity.

Add variance parameter. $= h$, controls smoothness

$\phi\left(\frac{(x-x_j)^2}{h}\right) = \frac{1}{(2\pi)^{\frac{1}{2}} \cdot h} \exp\left(-\frac{(x-x_j)^2}{2h^2}\right)$

2 pts.

(c) Consider the following sample set. Which of the density estimation methods would you choose? Window-based (Eq. (7)) or K-nearest neighbor? Explain your answer.



Choose K-nearest neighbor. → local based

3 pts.

(d) For a general Parzen window function prove that it provides a probability distribution.

Must satisfy two conditions : ① $\phi(\vec{u}) \geq 0$

close to a probability distribution ② $\int \phi(\vec{u})\, d\vec{u} = 1$

$P(x) = \frac{K}{nV} = \frac{1}{n} \sum_{j=1}^{n} \int_{-\infty}^{\infty} \phi(x - x_j) \Rightarrow \int p(x)\, dx = 1$

$\frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}}$ $\quad \frac{1}{n} \sum^{n} \frac{1}{h^V} \phi\left(\frac{x - x_j}{h}\right)$

$\frac{1}{\sqrt{2\pi\sigma^2}}$

4 pts.

2. We consider a mixture of K poisson distributions and perform the Expectation-Maximization (EM) algorithm to compute the unknown parameters. The log-likelihood function of n independent objects for mixture of K Poisson distribution is defined as:

$$P(x; \lambda) = \sum_{i=1}^{n} \log \sum_{c=1}^{K} \pi_c f(x_i; \lambda_c)$$

where π_c's are the mixture weights and λ_c's are the parameters of K Poisson distributions. $f(x_i; \lambda_c)$ is defined as:

$1 \cdot k \cdot K$

$$f(x; \lambda_k) = \frac{\lambda_k^x e^{-\lambda_k}}{x!}.$$

(a) Introduce the latent indicator variables necessary for maximizing the log-likelihood function.

$\log P(x | \vec{\lambda}, \vec{\pi}) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k f(x_n | \lambda_k)$

$Z \in \mathbb{R}^{N \times K}$, each row is a vector for a point

$Z_{nk} = \begin{cases} 1 & : \text{component } k \text{ is responsible for generated } x_n \\ 0 & : \text{ow} \end{cases}$

2 pts.

Conditional Expectation

(b) Calculate the expectation of the latent variables. Provide a Bayesian interpretation for your answer.

E-step: estimate posterior distribution of data

$p(Z | x, \Theta^{old}) = E_Z(z | \cdots)$

M-step: $\Theta^{new} = \arg\max_{\Theta} Q(\Theta, \Theta^{old})$

joint distribution

$Q(\Theta, \Theta^{old}) = \sum_Z P(Z | x, \Theta^{old}) \cdot \log P(X, Z | \Theta)$; complete data log-likelihood

$E[Z_{nk}] = P(Z_{nk} = 1 | \vec{x}_n) * 1 = \frac{\overbrace{\boxed{P(Z_{nk}=1)}}^{=\pi_k} \cdot \overbrace{\boxed{P(\vec{x}_n | Z_{nk}=1)}}^{= P(x_n|\lambda_k)}}{\boxed{\sum_{Z_1} P(x_n, Z_n)} = \sum_{j=1}^{k} \pi_j \cdot P(x_n|\lambda_j)}$

Responsibility $\lambda(Z_{nk})$

$P(\vec{Z}_n | \vec{\pi}) = \prod_{k=1}^{K} \pi_k^{Z_{nk}}$

$\Rightarrow P(x_n | Z_n, \lambda_k) = \prod_{k=1}^{K} p(x_n | \lambda_k)^{Z_{nk}}$

5 pts.

conditional

(c) **Bonus question:** Assume the expectations of the latent variables are given. Calculate the unknown parameters λ_c's. Write down the details of your calculations.

objective fxn $Q = \underset{z}{E}\left[\log \underbrace{P(x, z|\vec{\lambda}, \vec{\pi})}_{=\left[\prod_{n=1}^{N}\prod_{k}\{\pi_k P(\vec{x}_n|\lambda_k)\}\right]^{z_{nk}}}\right]$

$= \sum_n \sum_k \gamma(z_{nk})\{\log \pi_k + x_n \log \lambda_k - \lambda_k - \log x\}$

pg. 445, 430.

5 pts.

Supplementary Sheet

NO 'fixed "right solution"', as in real life. ⇒ The accuracy is the only goal.

Google solution for insights and do it better ⇒ do not reinvent the wheel.

CV scheme is really important

SVM ⇒ assume metric space behind data.

Q/A session before exam.

24 Exam 2012 solutions

These are the solutions for the exam of 2012. Unfortunately, nobody wrote them yet.

These solutions might be plain wrong since they were not verified by a teaching assistant but were written by a student. Be cautious! We hope that they help you even though.

Write the solutions for exam 2012.

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Introduction to Machine Learning

Autumn Semester 2011

Prof. J.M. Buhmann

Final Exam

February 8th, 2012

First and Last name: ______________________

ETH number: ______________________

Signature: ______________________

	Topic	Max. Points	Points	Signature
1	Bayesian Inference and MLE	20		
2	Linear Classifiers and Kernels	20		
3	Ensemble Methods	20		
4	Regression, Bias and Variance	20		
5	Unsupervised Learning	20		
Total		100		

Grade: ....

Question 1: Bayesian Inference and Maximum Likelihood(20 pts.)

A telecommunication company needs to estimate the rate of the telephone calls in a small town in order to adjust its channel capacity. We model a length of a time-interval between telephone calls as a random variable x. Knowing that a sequence of calls is the realization of the Poisson process, we model the time it takes before the next call using the exponential distribution.

Consider the task of estimating the rate parameter λ of the exponential distribution from n i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$, $x \in \mathbb{R}$

$$Exp\,(x|\lambda) = \lambda \exp\left(-\lambda x\right).$$

a) Write the maximum likelihood estimator for the rate $\widehat{\lambda}_{\mathsf{ML}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$:
(please write the direct closed-form solution and the derivation)

1. $\widehat{\lambda}_{\mathsf{ML}}(\mathcal{X}) = \arg\max p(\mathcal{X}|\lambda) =$ **2 pts.**

2. Where did you use the fact that the observations are i.i.d? **1 pt.**

To find the posterior density $p(\lambda|\mathcal{X})$ we need a prior on λ. We claim that a conjugate prior for the exponential distribution is the gamma distribution

$$Gamma\,(\lambda|\alpha,\beta) = \frac{\Gamma(\alpha)}{\beta^\alpha}\lambda^{\alpha-1}\exp(-\lambda\beta),$$

where $\Gamma(\alpha) = \int_0^\infty \exp^{-t} t^{\alpha-1} dt$ is the gamma function.

b) 1. What does *conjugate prior* mean? **1 pt.**

2. Show that the gamma distribution is the conjugate prior of the exponential distribution. **2 pts.**

c) Given a Gamma prior over the rate λ (prior with parametes α and β), write the maximum a posteriori estimator $\widehat{\lambda}_{\mathsf{MAP}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$:
(please write the direct closed-form solution)

$\widehat{\lambda}_{\mathsf{MAP}}(\mathcal{X}) = \arg\max p(\lambda|\mathcal{X}) =$ **2 pts.**

d) When is the maximum likelihood estimatior (MLE) equal to the maximum a posteriori (MAP) estimator given a set of i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$?

1. If the number of observations is finite. **2 pts.**

2. If the number of observations is infinite $(n \to \infty)$. **2 pts.**

e) Assume that you have a set of the i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$ and that you can not decide which distribution to use for data description: the Gaussian distribution or the Beta distribution.
If you use the Bayesian framework what you can look at? **2 pts.**

Now consider a binary classification task from a set of the i.i.d. observations $\mathcal{X} = \{\mathbf{x}_i, y_i\}_{i=1}^n$, with $\mathbf{x} \in \mathbb{R}^D$. Assume that the likelihood of both classes is Gaussian (assume class prior π_i, mean μ_i, and covariance matrix Σ_i for class y_i, with $i = 1, 2$).

f) Recall that a discriminant function for class y_i is defined as:

$$g_{y_i}(\mathbf{x}) = p\left(y_i|\mathbf{x}\right).$$

How can you find a decision surface in terms of likelihood, prior and evidence? **1 pt.**

g) Assume that

$$\mu_1 = \mu_2 = \mu$$
$$\Sigma_1 = \frac{1}{2\lambda_1}\mathbb{I}, \quad \Sigma_2 = \frac{1}{2\lambda_2}\mathbb{I}$$
$$\lambda_1 > 0, \quad \lambda_2 > 0, \quad \lambda_1 \neq \lambda_2$$

where $\mathbb{I}$ denotes the identity matrix. Write the equation satisfied by the separating decision surface. The equation must be an explicit function of $\mathbf{x}_1$ (the single observation), of the class prior, means and covariance:
(please write the solution in the polinomial form)

1. Decision surface: **3 pts.**

2. In the case described above, is the decision surface linear, parabolic, spherical, cylindrical, or something else?

   linear|parabolic|spherical|cylindrical|other **2 pts.**

Question 2: Linear Classifiers and Kernels (20 pts.)

a) Below is a list of algorithms which given a training set output a prediction function. Cross **all** of the algorithms that necessarily output a linear (in the original space) prediction function.

- ☐ Neural network
- ☐ Perceptron with learning rate $\eta = 1$
- ☐ SVM with radial basis kernel
- ☐ K-nearest neighbor classifier
- ☐ SVM with polynomial kernel with degree 1
- ☐ Ridge regression

3 pts.

b) Recall the SVM problem. As a constrained optimization problem a solution can be obtained through both the primal and the dual form.

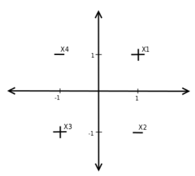
1. Given a primal solution for the SVM, write down the resulting classifier. **1 pt.**

2. Given a dual solution for the SVM, write down the resulting classifier. **1 pt.**

3. In practice, often the dual form of the SVM is solved to obtain a classifier. Provide one advantage of solving the dual SVM instead of the primal. **3 pts.**

c) Let $S = \{\mathbf{x}_i, y_i\}_{i=1}^4$ be the following training set

$\mathbf{x}_1 = (1, 1) \quad y_1 = 1,$
$\mathbf{x}_2 = (1, -1) \quad y_2 = -1,$
$\mathbf{x}_3 = (-1, -1) \quad y_3 = 1,$
$\mathbf{x}_4 = (-1, 1) \quad y_4 = -1$



Suppose that we trained an SVM on S and the resulting classifier $f(x)$ achieved zero training error. We ask you to provide an explicit description of $f(x)$ (a formula with numeric values).
Hint: Think of a suitable kernel function or alternatively a feature map. **5 pts.**

d) Consider a training set $S = \{\mathbf{x}_i, y_i\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^D$ and $y_i \in \{-1, 1\}$

1. Briefly describe a leave one out (LOO) procedure for estimating the error of an SVM classifier on S. **2 pts**

2. What is the LOO error? **1 pts**

3. Suppose that we trained an SVM classifier on the **entire** dataset S, denote by sv the set of support vectors, $sv = \{\mathbf{x}_j | \alpha_j > 0\}$.
   For the same value of C, prove that the LOO error is bounded by $\frac{|sv|}{n}$ i.e.
   $$\text{LOO error} \leq \frac{|sv|}{n}$$
   4 pts.

Question 3: Bagging and Boosting (20 pts.)

a) Answer precisely the following questions.

1. Are bagging and Boosting Bayesian approaches? Why?

1 pt.

2. How is it possible to detect outliers with AdaBoost?

1 pt.

3. From the frequentist perspective, bagging is motivated by the tradeoff between two terms. Which?

1 pt.

4. AdaBoost has an alternative interpretation which is based on the minimization of a certain cost function. Which function?

1 pt.

5. AdaBoost aims at selecting the best approximation to which ratio?

2 pt.

6. Why is the standard form of AdaBoost limited to binary classification?

1 pt.

7. How could one parallelize bagging?

1 pt.

8. Name a design property of the base classifiers of AdaBoost which impacts the overall predictive power.

1 pt.

9. Under which conditions does AdaBoost yield good results even when the base classifiers exhibit an individual performance that is only slightly better than that purely due to chance?

2 pt.

b) Consider *bagging* in the context of binary classification. Let the *target function* be $h(x)$, where $h : \mathbb{R}^d \to \{\pm 1\}$. Let us combine B *individual classifiers* $y_b(x), b = 1 \dots B$ to obtain a $committee$ $model$

$$y_{\mathsf{COM}}(x) = \text{sign}\left[\frac{1}{B}\sum_{b=1}^{B} y_b(x)\right]. \tag{1}$$

1. Write down the pseudocode of bagging for binary classification, from the input (data and model) to the prediction output.

3 pts.

2. The error $\epsilon_b(x) = \exp\{-h(x)y_b(x)\}$ indicates the error of an individual model $y_b(x)$ for a single sample x in terms of the target function $h(x)$ and the output of the individual model $y_b(x)$. Write down E_{AV}, that is the average of the expected errors over the individual classifiers $y_b(x)$, and the expected error E_{COM} made by combined model $y_{\mathsf{COM}}(x)$ as a function of the output of the committee model and of the target function.

$E_{\mathsf{AV}} =$

$E_{\mathsf{COM}} =$

1 pts.

3. Under which conditions is $E_{\mathsf{AV}} < E_{\mathsf{COM}}$?

3 pts.

4. With the same exponential error, write down the error function for each iteration of AdaBoost with weighting coefficients α_b for the B base classifers $y_b(x)$.

$E_{\mathsf{AdaBoost}} =$

1 pt.

5. In this scenario, within AdaBoost the minimization of this error function is performed with respect to two terms, which?

1)

2)

1 pt.

Question 4: Regression, Bias and Variance (20 pts.)

a) Write down the linear regression model (component-wise and in vector notation) for input variable $x = (1, x_1, \ldots, x_D)^\mathsf{T} \in \mathbb{R}^{D+1}$ and output variable y. Formally introduce the model parameter(s).

From now on, assume that the input dataset consists of N samples given by the matrix $\mathbf{X} \in \mathbb{R}^{N\times(D+1)}$ (where the first column is $\mathbf{1}$), and the output, $\mathbf{Y} \in \mathbb{R}^N$. Write down the linear regression model for all observations $\mathbf{Y}$ (in matrix notation).

2 pts.

b) Write down the following cost functions (in a notation of your choice): **2 pts.**

1. Ridge Regression (*RR*):
2. Least Absolute Shrinkage and Selection Operator (*LASSO*). Formulate as a constrained optimization problem:

c) Formulate the objective of **learning** (formula) in the regression setting introduced above with data $(\mathbf{X}, \mathbf{Y})$ i.i.d. from $P(\mathbf{X}, \mathbf{Y})$.

2 pts.

d) Assuming that the observations in $\mathbf{Y}$ are affected by additive Gaussian noise ϵ with $\epsilon \sim \mathcal{N}(0, \sigma^2)$. What do we know about the distribution of the *RSS*-estimator (i.e. $\hat{\beta}^{RSS}$)?

3 pts.

e) Please briefly give another motivation for the *RSS* cost function.

1 pt.

f) Model inference (computing model parameters) for the *RSS* cost function requires the inversion of a matrix: $(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}$.
- Specify a mathematical condition when this inversion is *numerically unstable*.
- Describe in your own words under which circumstance this instability happens during a practical application of regression.
- Are we then in risk of *under-* or *over-fitting*?
- Also comment qualitatively on the bias and variance of the model parameter's estimation in this circumstance.

3.5 pts.

g) We can stabilize the inversion by reducing the model complexity.
- Explain **how** *Ridge Regression* (*RR*) limits the model complexity.
- Provide another approach to limiting the model complexity.
- Demonstrate the stabilization mathematically by writing down the *Ridge Regression* solution and argue with the Eigenvalues of the matrix that needs to be inverted.

4 pts.

h) Comment qualitatively on the bias and variance of *Ridge Regression* as compared to the *RSS* estimator.

1.5 pts.

i) By depicting an appropriate plot (including notation), provide a graphical argument why *LASSO* favors sparse solutions.

1 pt.

Question 5: Unsupervised Learning (20 pts.)

a) For each of the following non-parametric approaches mention the most important parameter that influences the smoothness of the results: **3 pts.**

- histograms

- Parzen window estimates

- nearest neighbor estimates.

b) Determine whether the following statements are true or false. Briefly explain your answer. **2 pts.**

- Non-parametric estimation methods are less sensitive than parametric approaches to model misspecification.

- In histograms, by changing the dimensionality the number of required bins (to keep the resolution) increases linearly with dimension.

c) Consider Hidden Markov Models (HMMs). For each of the following algorithms determine if it solves a supervised or an unsupervised problem. Explain your answer.

4 pts.

- Viterbi algorithm

- Baum-Welch algorithm.

d) In this section we study the k-means clustering method.

1. Mention at least two main differences between k-means and Gaussian Mixture Model (GMM) clustering methods.

   2 pts.

   Consider the k-means cost function defined as:

$$R^{km} = \sum_{n=1}^{N} \sum_{l=1}^{k} r_{nl} ||\mathbf{x}_n - \boldsymbol{\mu}_l||^2. \tag{2}$$

   Here $\boldsymbol{\mu}_l$ denotes the l-th centroid and $r_{nl} \in \{0, 1\}$ indicates the assignment of object $\mathbf{x}_n$ to the l-th cluster.

2. Write down the assignment update step (E-step) for the k-means algorithm.

   2 pts.

3. Derive the centroids update step (M-step). We expect you to write down all intermediate steps of the centroid update derivation.

   4 pts.

4. Show that the k-means algorithm always converges.

   3 pts.

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Introduction to Machine Learning
Autumn Semester 2011

Prof. J.M. Buhmann

Final Exam
February 8th, 2012

First and Last name: ______________________

ETH number: ______________________

Signature: ______________________

	Topic	Max. Points	Points	Signature
1	Bayesian Inference and MLE	20		
2	Linear Classifiers and Kernels	20		
3	Ensemble Methods	20		
4	Regression, Bias and Variance	20		
5	Unsupervised Learning	20		
Total		100		

Grade: ....

Question 1: Bayesian Inference and Maximum Likelihood(20 pts.)

A telecommunication company needs to estimate the rate of the telephone calls in a small town in order to adjust its channel capacity. We model a length of a time-interval between telephone calls as a random variable x. Knowing that a sequence of calls is the realization of the Poisson process, we model the time it takes before the next call using the exponential distribution.

Consider the task of estimating the rate parameter λ of the exponential distribution from n i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$, $x \in \mathbb{R}$

$$Exp\,(x|\lambda) = \lambda \exp\left(-\lambda x\right).$$

a) Write the maximum likelihood estimator for the rate $\widehat{\lambda}_{\mathsf{ML}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$: (please write the direct closed-form solution and the derivation)

1. $\widehat{\lambda}_{\mathsf{ML}}(\mathcal{X}) = \arg\max p(\mathcal{X}|\lambda) = \prod_{i=1}^{n} \lambda \exp(-\lambda x_i)$ **2 pts.**

$\Rightarrow \log\left(\sum_i^n \lambda \exp(-\lambda x_i)\right)$

$\sum_i^n \left(\log(\lambda) - \lambda x_i\right) = n \log \lambda - \lambda \sum_i^n x_i \stackrel{!}{=} 0$

$\frac{d}{d\lambda}\left(n \log\lambda - \lambda \sum_i^n x_i\right) \stackrel{!}{=} 0 \quad n\frac{1}{\lambda} - \sum_i^n x_i = 0 \quad \frac{n}{\lambda} = \sum_i^n x_i$

$\boxed{\hat{\lambda}_{ML} = \frac{n}{\sum_i^n x_i}}$

2. Where did you use the fact that the observations are i.i.d? **1 pt.**

$\prod_{i=1}^{n}$ in the likelihood equation

To find the posterior density $p(\lambda|\mathcal{X})$ we need a prior on λ. We claim that a conjugate prior for the exponential distribution is the gamma distribution

$p(\lambda|x) = p(x|\lambda)\, p(\lambda)$ ← old distribution; ↑ distribution you assumed that x is drawn from

$$Gamma\,(\lambda|\alpha,\beta) = \frac{\Gamma(\alpha)}{\beta^\alpha}\lambda^{\alpha-1}\exp(-\lambda\beta),$$

where $\Gamma(\alpha) = \int_0^\infty \exp^{-t} t^{\alpha-1} dt$ is the gamma function.

b) 1. What does *conjugate prior* mean? **1 pt.**

The prior distribution $P(\lambda)$ is conjugate to the likelihood distribution $p(x|\lambda)$ if multiplying these two distributions together and normalizing results in another distribution of the same form as the prior $P(\lambda)$.

2. Show that the gamma distribution is the conjugate prior of the exponential distribution. **2 pts.**

exp × gamma

$= \left(\lambda \exp(-\lambda x)\right)\left(\frac{\Gamma(\alpha)}{\beta^\alpha}\lambda^{\alpha-1}\exp(-\lambda\beta)\right) = \lambda^1 \lambda^{\alpha-1} \exp^{-\lambda x - \lambda\beta}$

$= \lambda^{1+\alpha-1} \exp^{-\lambda(x+\beta)}$

$\boxed{= \lambda^{(\alpha+1)-1} \exp^{-\lambda(x+\beta)}}$

*c) Given a Gamma prior over the rate λ (prior with parametes α and β), write the maximum a posteriori estimator $\widehat{\lambda}_{\mathsf{MAP}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$: (please write the direct closed-form solution)

$\widehat{\lambda}_{\mathsf{MAP}}(\mathcal{X}) = \arg\max p(\lambda|\mathcal{X}) = \prod_{i=1}^{n}\left[\left(\lambda\exp(-\lambda x_i)\right)\left(\frac{\Gamma(\alpha)}{\beta^\alpha}\lambda^{\alpha-1}\exp(-\lambda\beta)\right)\right]$ **2 pts.**

$= \cancel{\prod_{i=1}^{n}}\left[\lambda^{\alpha} \exp^{-\lambda(x_i+\beta)}\right] \stackrel{\log}{\Rightarrow} \sum_i^n \left(\log(\lambda^\alpha) - \lambda(x_i-\beta)\right) \frac{d}{d\lambda} \stackrel{!}{=} 0 \quad n \cdot \frac{1}{\lambda^\alpha} - \sum_i^n (x_i - \beta) = 0$

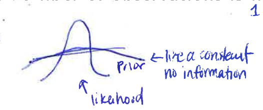
$\frac{1}{\lambda^\alpha} = \frac{\sum_i^n (x_i-\beta)}{n} \Rightarrow \lambda^{\alpha\cdot\frac{1}{\alpha}} = \left(\frac{n}{\sum_i^n x_i - \beta}\right)^{\frac{1}{\alpha}}$ $\boxed{\cancel{\lambda_{MAP} = \left(\frac{n}{\sum_i^n x_i - \beta}\right)^{\frac{1}{\alpha}}}}$

$\hat{\lambda}_{map} = \frac{\alpha - 1 + n}{\beta + \sum x_i}$

d) When is the maximum likelihood estimatior (MLE) equal to the maximum a posteriori (MAP) estimator given a set of i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$?

1. If the number of observations is finite. **2 pts.**

NO. 1 sample



2. If the number of observations is infinite ($n \to \infty$). **2 pts.**

yes Always converges

e) Assume that you have a set of the i.i.d. observations $\mathcal{X} = \{x_i\}_{i=1}^n$ and that you can not decide which distribution to use for data description: the Gaussian distribution or the Beta distribution.

If you use the Bayesian framework what you can look at? **2 pts.**

conjugate prior and likelihood

* Now consider a binary classification task from a set of the i.i.d. observations $\mathcal{X} = \{\mathbf{x}_i, y_i\}_{i=1}^n$, with $\mathbf{x} \in \mathbb{R}^D$. Assume that the likelihood of both classes is Gaussian (assume class prior π_i, mean μ_i, and covariance matrix Σ_i for class y_i, with $i = 1, 2$).

multivariate normal $\frac{1}{(2\pi)^{n/2}|\Sigma|} \exp\left(-\frac{1}{2}(\vec{x}-\vec{\mu})^T \Sigma^{-1}(\vec{x}-\vec{\mu})\right)$ $|\Sigma| \Rightarrow$ Ndet Σ

f) Recall that a discriminant function for class y_i is defined as:

$$g_{y_i}(\mathbf{x}) = p\left(y_i|\mathbf{x}\right).$$

class decision: $g_y(x) > g_z(x)$ $\forall z \neq y$

How can you find a decision surface in terms of likelihood, prior and evidence? **1 pt.**

$p(y_i|x) = p(x|y_i)p(y_i) =$

$\cancel{p(y_i|x) = \frac{\left(\prod_{i=1}^{n} \frac{1}{(2\pi)^{n/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right)\right) \cdot \pi_i}{\int_{-\infty}^{\infty} \left(p(x|y_i)\cdot\pi_i\right) dx_i}}$

L5 Sl 27

Decision surface $g_1 = g_2$

$p(y=1|x)\pi_1 > p(y=2|x)\pi_2$

Decision discriminant fcn = posterior, defines boundary if one is greater

$\rightarrow \frac{p(x|y_1)\,p(y_1)}{p(x)} > \frac{p(x|y_2)\,p(y_2)}{p(x)}$

g) Assume that

$$\mu_1 = \mu_2 = \mu$$
$$\Sigma_1 = \frac{1}{2\lambda_1}\mathbb{I}, \quad \Sigma_2 = \frac{1}{2\lambda_2}\mathbb{I}$$
$$\lambda_1 > 0, \quad \lambda_2 > 0, \quad \lambda_1 \neq \lambda_2$$

where $\mathbb{I}$ denotes the identity matrix. Write the equation satisfied by the separating decision surface. The equation must be an explicit function of $\mathbf{x}_1$ (the single observation), of the class prior, means and covariance: p 36
(please write the solution in the polinomial form)

1. Decision surface: $g_1(x) - g_2(x) = 0$ **3 pts.**

$\ln(P(x_1 | \mu_1)) + \ln P(\omega_1) - \ln(P(x|\omega_2)) - \ln(P(\omega_1))$

$-\frac{1}{2}x^T W x + w^T x + w_{i0}$ $\quad -\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma|$

$W = \Sigma^{-1}$ $\quad w = \Sigma^{-1}\mu$ $\quad \frac{1}{2}\mu^T\Sigma^{-1}\mu$

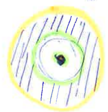
$\Sigma_1 = \Sigma_2 \rightarrow$ linear $\quad \Sigma_1 \neq \Sigma_2$.

2. In the case described above, is the decision surface linear, parabolic, spherical, cylindrical, or something else?

ellipsoidal

linear|parabolic|spherical|cylindrical|other **2 pts.**

Dependent on covariance matrix



Gaussian within a gaussian.

Question 2: Linear Classifiers and Kernels (20 pts.)

a) Below is a list of algorithms which given a training set output a prediction function. Cross **all** of the algorithms that necessarily output a linear (in the original space) prediction function.

inputs, output, step fxn

- ☒ Neural network
- ☒ Perceptron with learning rate $\eta = 1$
- ☐ SVM with radial basis kernel
- ☐ K-nearest neighbor classifier
- ☒ SVM with polynomial kernel with degree 1
- ☒ Ridge regression $w\,\phi(x)$

also works w/ kernels. **3 pts.**

b) Recall the SVM problem. As a constrained optimization problem a solution can be obtained through both the primal and the dual form.

1. Given a primal solution for the SVM, write down the resulting classifier. **1 pt.**

~~$W = \sum \alpha_i y_i x_i$~~ $w^T x + w_0$

$y = \text{sgn}(w^T x + w_0)$

2. Given a dual solution for the SVM, write down the resulting classifier. **1 pt.**

$W = \sum_i \alpha_i y_i x_i \rightarrow w^T x + w_0 = y$.

$\alpha_i \geq 0$

$y = \text{sgn}\left(\sum_i \alpha_i y_i x_i\, X + w_0\right)$

3. In practice, often the dual form of the SVM is solved to obtain a classifier. Provide one advantage of solving the dual SVM instead of the primal. **3 pts.**

Even though the primal form it has been transformed into an

$y = w^T x + w_0 \quad \max_\alpha \left(\sum\sum \alpha_i \alpha_j y_i y_j x_i x_j\right)$

$k(x,y) = \langle\phi(x), \phi(y)\rangle$

$w = \sum \alpha_i y_i x_i$

c) Let $S = \{\mathbf{x}_i, y_i\}_{i=1}^4$ be the following training set

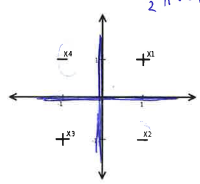
$\min \frac{1}{2}\|w\|^2 + \alpha(\cdots)$

$\mathbf{x}_1 = (1, 1) \quad y_1 = 1,$

$\mathbf{x}_2 = (1, -1) \quad y_2 = -1,$

$\mathbf{x}_3 = (-1, -1) \quad y_3 = 1,$

$\mathbf{x}_4 = (-1, 1) \quad y_4 = -1$



Suppose that we trained an SVM on S and the resulting classifier $f(x)$ achieved zero training error. We ask you to provide an explicit description of $f(x)$ (a formula with numeric values).
Hint: Think of a suitable kernel function or alternatively a feature map. **5 pts.**

~~$w^T x$~~ $\quad \text{sgn}(\langle a, b\rangle)$ scalar product

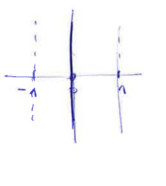
$\phi(x) = \langle\phi(x_1), \phi(x_2)\rangle$ $\quad \prod_i x_i \prod_j x_j$ $\quad K(x_i, x_j) =$

$K(x,y) = \prod_i x_i \prod_j x_j \quad = k(x_1) = 1 \cdot 1 = 1$

$k(x_2) = 1 \cdot -1 = -1$

$k(x_3) = -1 \cdot -1 = 1$

$k(x_4) = -1 \cdot 1 = -1$

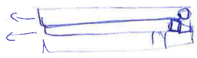


Closure property:

$k(x,y) = f(x) \cdot f(z)$

d) Consider a training set $S = \{\mathbf{x}_i, y_i\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^D$ and $y_i \in \{-1, 1\}$

1. Briefly describe a leave one out (LOO) procedure for estimating the error of an SVM classifier on S. **2 pts**



$R^{CV} = \frac{1}{n}\sum_n \mathbb{I}$ $\quad \mathbb{I}$ indicator fxn

$\frac{\text{number of errors}}{10}$

2. What is the LOO error? **1 pts**

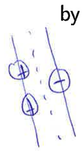
$\Rightarrow R^{CV} = \frac{1}{n}\sum_n \mathbb{I}_{y_i \neq c^{*}(x_i)}$

3. Suppose that we trained an SVM classifier on the **entire** dataset S, denote by sv the set of support vectors, $sv = \{\mathbf{x}_j | \alpha_j > 0\}$.
For the same value of C, prove that the LOO error is bounded by $\frac{|sv|}{n}$ i.e.

$$\text{LOO error} \leq \frac{|sv|}{n}$$

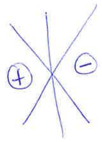
cardinality → number of sv

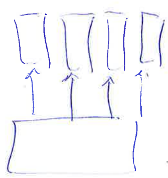
4 pts.



$R^{CV} = \frac{1}{n}\sum_{i=1}^{n} R^{-\mu}$

$= \frac{1}{n}\sum_{\mu \in SV} R^{\mu} \leq \frac{1}{n}|sv| \cdot c$





Question 3: Bagging and Boosting (20 pts.)

a) Answer precisely the following questions.

1. Are bagging and Boosting Bayesian approaches? Why?

Not Bayesian approaches. → pg. 410 Hastie **1 pt.**

655 Bagging construct new parameters ← approximate parameters prior

2. How is it possible to detect outliers with AdaBoost?

super large weights at the end of the training. **1 pt.**
to one classifier will detect

3. From the frequentist perspective, bagging is motivated by the tradeoff between two terms. Which?

Bias, variance **1 pt.**

4. AdaBoost has an alternative interpretation which is based on the minimization of a certain cost function. Which function?

exponential loss **1 pt.**

5. AdaBoost aims at selecting the best approximation to which ratio?

log-odds ratio **2 pt.**

6. Why is the standard form of AdaBoost limited to binary classification?

1 pt.

$\hat{C}_b = \operatorname{sgn}\left(\sum_i^B \alpha_b C_b(x)\right)$

7. How could one parallelize bagging?

sub, sets **1 pt.**

8. Name a design property of the base classifiers of AdaBoost which impacts the overall predictive power.

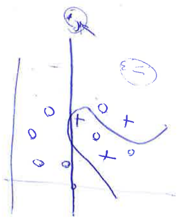
uncorrelated different

weights to classifiers. **1 pt.**

weakness of basic learners
strong and weak learners.

9. Under which conditions does AdaBoost yield good results even when the base classifiers exhibit an individual performance that is only slightly better than that purely due to chance?

when uncorrelated to different classifiers. **2 pt.**
classifiers give different outputs
and answers.
B is much bigger than 1.
and much larger dataset



Supplementary Sheet

Supplementary Sheet

27 Exam 2011 solutions

These are the solutions for the exam of 2011. Unfortunately, nobody wrote them yet.

These solutions might be plain wrong since they were not verified by a teaching assistant but were written by a student. Be cautious! We hope that they help you even though.

Write the solutions for exam 2011.

Prof. J. M. Buhmann

Final Exam

January 29th, 2011

First and Last name: ______________________________

ETH number: ______________________________

Signature: ______________________________

	Topic	Max. Points	Points Achieved	Visum
1	Bayesian Inference	25		
2	Regression	25		
3	Bagging and Boosting	25		
4	SVM	25		
5	Mixture Models	25		
Total		125		

Grade: . . . .

Question 1: Bayesian Inference (25 pts.)

Consider the task of estimating mean μ and variance σ^2 of a Gaussian density from n i.i.d. observations $\mathcal{X} = \{x_1, \dots, x_n\}$, $x \in \mathbb{R}$.

a) Write the maximum likelihood estimator for the mean $\widehat{\mu}_{\mathsf{ML}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_1, \dots, x_n\}$:
(please write the direct closed-form solution)

1. $\widehat{\mu}_{\mathsf{ML}}(\mathcal{X}) = \arg\max p(\mathcal{X}|\mu) =$ **1 pts.**

2. Is this a biased estimator? YES\NO **1 pts.**

b) Write the maximum likelihood estimator for the variance $\widehat{\sigma}^2_{\mathsf{ML}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_1, \dots, x_n\}$:
(please write the direct closed-form solution)

1. $\widehat{\sigma}^2_{\mathsf{ML}}(\mathcal{X}) = \arg\max p(\mathcal{X}|\sigma^2) =$ **1 pts.**

2. Is this a biased estimator? YES\NO **1 pts.**

Now, assume that the variance σ^2 is known.

c) Given a Gaussian prior over the mean (prior with zero mean and variance one), write the posterior density $p(\mu|\mathcal{X})$ as an explicit function of the single observation $\mathcal{X} = \{x_1\}$:
(please write the direct closed-form solution)

1. $p(\mu|\mathcal{X}) = \frac{p(\mathcal{X}|\mu)p(\mu)}{p(\mathcal{X})} =$ **2 pts.**

2. Is the posterior a Gaussian density? YES\NO **1 pts.**
3. Is the posterior a Gaussian density for all priors? YES\NO **1 pts.**

d) Given a Gaussian prior over the mean (prior with zero mean and variance one), write the maximum a posteriori estimator $\widehat{\mu}_{\mathsf{MAP}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_1, \dots, x_n\}$ (recall that $\int e^{-(\alpha^2)} d\alpha = \sqrt{(\pi)}$):
(please write the direct closed-form solution)

1. $\widehat{\mu}_{\mathsf{MAP}}(\mathcal{X}) = \arg\max p(\mu|\mathcal{X}) =$ **2 pts.**

2. Does the maximum a posteriori estimate equal the mean of the posterior? YES\NO **1 pts.**

Now consider a binary classification task from observations $\mathcal{X} = \{x_1, \dots, x_n\}$, with $x \in \mathbb{R}^D$. Assume that the likelihood of both classes is Gaussian (assume class prior π_i, mean μ_i, and covariance matrix Σ_i for class y_i, with $i = 1, 2$).

e) Write the discriminant $g_{y_1}(x) = p(y_1|x)$ as an explicit function of class prior, mean and covariance:
(please write the direct closed-form solution)

$g_{y_1}(x) =$ **3 pts.**

f) Assume that $\Sigma_1 = \Sigma_2 = \sigma^2\mathbb{I}$, where $\mathbb{I}$ denotes the identity matrix. Write the equation satisfied by the separating decision surface. The equation must be an explicit function of x_1 (the single observation), of class prior, means, and covariance:
(please write the direct closed-form solution)

1. $0 = g_{y_1}(x) - g_{y_2}(x) =$ **2 pts.**

2. In this case, is the decision surface constant, linear, quadratic, cubic, or something else? constant\linear\quadratic\cubic\other **2 pts.**
3. Would it be possible to obtain a cylindrical decision surface? YES\NO **1 pts.**

g) In the two dimensional case subject to uniform class prior, write means (μ_1, μ_2) and covariances (Σ_1, Σ_2) such that the decision surface is a hyperplane:
(any numerical instantiation which satisfies this constraint is acceptable)

$\mu_1 =$

$\mu_2 =$

$\Sigma_1 =$

$\Sigma_2 =$ **3 pts.**

h) In the two dimensional case subject to uniform class prior, write means (μ_1, μ_2) and covariances (Σ_1, Σ_2) such that the decision surface is spherical:
(any numerical instantiation which satisfies this constraint is acceptable)

$\mu_1 =$

$\mu_2 =$

$\Sigma_1 =$

$\Sigma_2 =$ **3 pts.**

Question 2: Regression (25 pts.)

Consider the linear regression model expressed in the homogeneous coordinates:

$$y = \boldsymbol{\beta}_{(0)} + \sum_{i=1}^{D} \mathbf{x}_{(i)}\boldsymbol{\beta}_{(i)} = \mathbf{x}^\top \boldsymbol{\beta},$$

where $\mathbf{x} = (1, \mathbf{x}_{(1)}, ..., \mathbf{x}_{(D)}) \in \mathbb{R}^{D+1}$ is the input variable and y is the corresponding target variable.

Assume that the input dataset is given by the matrix $\mathbf{X} \in \mathbb{R}^{N\times(D+1)}$ whose first column is $\mathbf{1}$. Then the linear regression model for all the observations is written as $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}$. Consider this linear regression model and answer the following questions:

a) For this problem, formally define the Residual Sum of Squares (RSS) cost function and write it down in matrix notation.
Answer:

2 pts.

b) Briefly motivate the connection between minimizing the Residual Sum of Squares and maximizing the likelihood of $\mathbf{y}$ given $\mathbf{X}$.
Answer:

2 pts.

c) We minimize the RSS function and infer the model parameters as $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$. Considering the matrix inverse operation, mathematically describe why in practice we are interested in regularized models such as *ridge* regression models rather than the given unregularized model.
Answer:

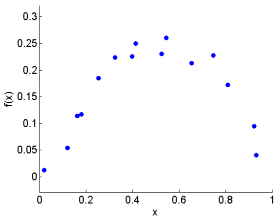
3 pts.

d) Formally define ridge and LASSO regression models. By depicting appropriate plots, demonstrate the difference between these two models in inferring the model parameters $\hat{\boldsymbol{\beta}}$.
Answer:

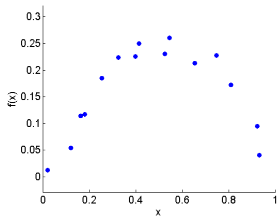
5 pts.

Now, we consider a general form of the problem where the set of observation $\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{x}_i \in \mathbb{R}^D, y_i \in \mathbb{R}$ are drawn i.i.d. from the joint distribution $P(X,Y)$. The goal is to find the regression function $f \in \mathcal{F}$ such that the mean squared error $\mathbb{E}_{XY}[(Y - f(X))^2]$ is minimal. Where the hypothesis class $\mathcal{F}$ contains the set of all polynomial functions.

e) Choosing an inappropriate function f might lead to either underfitting or overfitting. For the depicted given data, draw relevant plots to show each of these situations. For the overfitting case, briefly explain what happens if we have more observations.
Answer:



underfitting



overfitting

4 pts.

We can split the mean squared error $\mathbb{E}_{XY}[(Y - f(X))^2]$ into bias and variance, and find the best tradeoff between them.

f) Write down the mathematical definition of bias and variance.
Answer:

2 pts.

g) Expand the mean squared error $\mathbb{E}_{XY}[(Y - f(X))^2]$ and write it in the form of variance + squared bias.
Answer:

3 pts.

In this section we study how the averaging changes the bias of a set of unbiased estimators. Assume that we are given a set of B estimators $\hat{f}_1(\mathbf{x}), \hat{f}_2(\mathbf{x}), ..., \hat{f}_B(\mathbf{x})$. We take the average estimator by $\hat{f}(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(\mathbf{x})$.

h) Calculate the bias of the average estimator $\hat{f}(\mathbf{x})$ in terms of the bias of the estimators $\hat{f}_1(\mathbf{x}), ..., \hat{f}_B(\mathbf{x})$.
Hint: Start with the mathematical definition of the bias for $\hat{f}(\mathbf{x})$.
Answer:

3 pts.

i) According to the results, briefly explain why unbiased estimators remain unbiased after averaging.
Answer:

1 pts.

Question 3: Bagging and Boosting (25 pts.)

Bagging and *boosting* are two possible approaches to combine multiple models (classifiers or regressors) to achieve a *composite model* with improved performance. Both methods can be used in both a classification and a regression setting.

a) State two essential differences between bagging and boosting.
Answer:

2 pts.

b) Explain in terms of the bias-variance trade-off why the idea of combining models works.
Answer:

2 pts.

c) We now look at the problem of regression and how the combination of individual regression models can give better results. Left-column figures show the individual regression models, right-column figures show the true target function (solid) and the output of averaging the individual models to obtain a composite model (dashed).

1. The individual regression models (in Figure 1 and Figure 3) have been regularized using a regularization parameter λ, i.e. the cost function had the following form

$$RSS_{Ridge}(\boldsymbol{\beta}) = \sum_{i=1}^{n}(t_i - \boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\beta})^2 \;+\; \lambda||\boldsymbol{\beta}||_2^2$$

The parameters used were $\lambda = 0.09$ and $\lambda = 13.5$.
Associate the regularization parameters to the figures in the left column.
Answer:

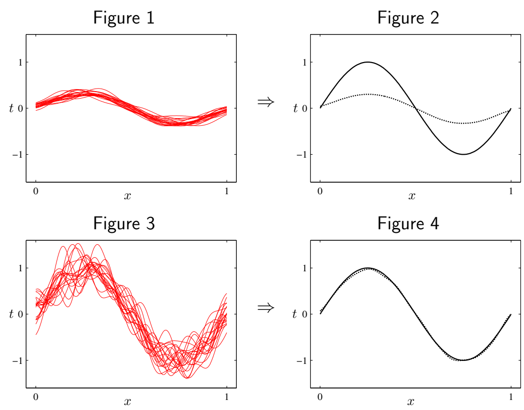
Figure 1: $\lambda = ...$

Figure 3: $\lambda = ...$

2. Please interpret the figures in terms of the bias-variance trade-off
Answer:

Figure 1 and 2:

Figure 3 and 4:



4 pts.

d) We now consider *bagging* in a regression setting. In order to model a *target function* $h(x)$, we combine M *individual models* $y_m(x), m = 1..M$ to obtain a *committee model* $y_{COM}(x)$. We model the *error* of each individual model using $\epsilon_m(x), m = 1..M$. This question's final goal is to show that the error of the committee model is M times smaller than the average error of the individual models.

1. Write down the output of the committee model, which averages the individual model's outputs. Answer:
   $y_{COM}(x) = \quad ...$
2. Write down the error $\epsilon_m(x)$ of an individual model in terms of the target function $h(x)$ and the output of the individual model $y_m(x)$. Answer:
   $\epsilon_m(x) = \quad ... \quad - \quad ...$
3. Write down the expected squared error of an individual model. $\mathbb{E}_x$ denotes the expectation value w.r.t. the distribution of x. Answer:
   $E_m = \mathbb{E}_x\Big[\qquad \qquad \Big]$
4. Write down the average of the expected squared errors made by the individual models
   Answer:
   $E_{AV} = \quad ...$
5. Write down the expected squared error made by the committee model, in terms of the output of the committee model and the target function.
   Answer:
   $E_{COM} = \mathbb{E}_x\Big[\qquad \qquad \Big]$
6. Show that $E_{COM} = \frac{1}{M} E_{AV}$.
   Hint: Use the assumption that the errors of the individual models are uncorrelated, i.e. $\mathbb{E}_x[\epsilon_m(x)\epsilon_l(x)] = 0, m \neq l$, and that the mean error of the individual models is zero, i.e. $\mathbb{E}_x[\epsilon_m(x)] = 0$.
   Answer:

10 pts.

We now turn our attention to *boosting* in a classification setting.

e) Given i.i.d. training data $\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), ..., (\mathbf{x}_n, c_n)\}$, $\mathbf{x}_i \in \mathbb{R}^D$, $c_i \in \{-1, 1\}$, and individual classifiers $y_m(\mathbf{x}), m = 1...M$, we can define the following additive model

$$f_k(\mathbf{x}) = \frac{1}{2} \sum_{j=1}^{k} \alpha_j y_j(\mathbf{x})$$

where α_j are weights, $k \leq M$.
Think of $f_k(\mathbf{x})$ as the *additive model* which uses the first $1..k$ individual classifiers.

1. Define the sum-of-squares error of $f_k(\mathbf{x})$ on the training dataset.
   Answer:
   $Err_k = \quad ...$
2. Assume that the first $1..(k-1)$ classifiers $y_j(\mathbf{x})$ and weights α_j, $j = 1..(k-1)$ have already been determined, i.e. fixed.
   Show that finding the optimal k-th classifier $y_k(\mathbf{x})$ and its weight α_k involves fitting the k-th classifier to the residual errors $f_{k-1}(\mathbf{x}_i) - c_i$ made by the $(k-1)$-th additive model.
   (*Hint: Start from the expression* Err_k *and recognize the terms* $f_{k-1}(\mathbf{x}_i) - c_i$)

7 pts.

Question 4: SVM (25 pts.)

In the following questions choose one answer only.

a) Why does a Support Vector Machine generalizes better than a Perceptron?
1. The selected support vectors tend to be very typical samples.
2. By supporting vectors it is not restricted to scalar input.
3. The requirement of maximal margins reduces the arbitrariness.
4. The support vector machine will have access to more training data.

2 pts.

b) What is the advantage of using kernels in support vector machines?
1. They tend to maximize the margins.
2. They enable non-linear separations.
3. They will increase the number of support vectors.
4. They reduce the risk of getting stuck in local minima.

2 pts.

Explain your answer to the following questions in 1-2 sentences.

c) Suppose that you have a binary SVM classifier with a linear kernel. Consider a vector $\mathbf{x}_i$ for which $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + w_0) > 1 \quad (\mathbf{x}_i \in \mathbb{R}^D, y_i \in \{-1, +1\})$
1. Is $\mathbf{x}_i$ correctly classified? YES\NO
2. If we remove $\mathbf{x}_i$ from the training set and re-train the classifier, will the decision boundary change or stay the same?

Answer:

3 pts.

d) We would like to build a binary SVM classifier for histograms. Let H denote the class of histograms, for each $\mathbf{h} \in H$ the following properties hold:

1. $\mathbf{h} = \{h_1, \ldots, h_m\}, \quad \forall j : h_j \geq 0$
2. $\sum_{j=1}^{m} h_j = L$ for some $L \in \mathbb{N}$.

Consider using the following kernel function:

$$K(\mathbf{h}, \tilde{\mathbf{h}}) = \sum_{j=1}^{m} \min(h_j, \tilde{h_j}) \quad \mathbf{h}, \tilde{\mathbf{h}} \in H$$

What kind of separation is imposed by this kernel? (which points in the space will become close/far using this kernel).
Answer:

3 pts.

e) Let $\mathcal{X}$ be a finite set and consider the following function. For $\mathbf{x}, \bar{\mathbf{x}} \in \mathcal{X}$:

$$K(\mathbf{x}, \bar{\mathbf{x}}) = \begin{cases} 1 & \text{if } \mathbf{x} = \bar{\mathbf{x}} \\ 0 & \text{else} \end{cases}$$

Prove that $K(\mathbf{x}, \bar{\mathbf{x}})$ is a legitimate kernel function.

Hint: One possible way to prove it is to find a feature mapping $\phi(\mathbf{x})$, such that
$K(\mathbf{x}, \bar{\mathbf{x}}) = \langle \phi(\mathbf{x}), \phi(\bar{\mathbf{x}}) \rangle$.
Answer:

3 pts.

$L2$-SVM use the square sum of the slack variables ξ_i in the objective function instead of the linear sum of the slack variables (squaring the hinge loss). Let $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ be a training set of examples and binary labels $y_i \in \{-1, +1\}$. The primal formulation of the $L2$-SVM is as follows

$$\min_{\mathbf{w}, w_0, \xi} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{2}\sum_{i=1}^{n} {\xi_i}^2$$

s.t.

$$\begin{aligned} y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + w_0) &\geqslant 1 - \xi_i \quad i = 1, \ldots, n \\ \xi_i &\geqslant 0 \qquad\qquad i = 1, \ldots, n \end{aligned}$$

f) Show that removing the last set of constraints: $\forall i : \xi_i \geqslant 0$ does not change the optimal solution to the primal problem.
Answer:

4 pts.

Now we would like to derive the dual form of the $L2$-SVM.

g) Write down the Lagrangian of the $L2$-SVM.
Answer:

$L(\qquad) =$

2 pts.

h) Compute the derivatives of the Lagrangian with respect to the appropriate variables.
Answer:

3 pts.

We decided not to bother you with the remaining computation, instead we finished the derivation ourselves.

i) Below are 4 optimization problems, only 1 of which is the dual $L2$-SVM. Circle the optimization problem corresponding to the dual $L2$-SVM.
Explain your choice by either shortly falsifying the other options or by showing the full derivation.
Hint: The full derivation is more time consuming.

$$\max_{\alpha} \quad \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j {\mathbf{x}_i}^T\mathbf{x}_j - \frac{1}{2C}\sum_{i=1}^{n}\alpha_i^2$$

s.t.:

$$\sum_{i=1}^{n}\alpha_i y_i = 0 \qquad \forall i : \alpha_i \geq 0$$

(a)

$$\max_{\alpha} \quad \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j {\mathbf{x}_i}^T\mathbf{x}_j$$

s.t.:

$$\sum_{i=1}^{n}\alpha_i y_i = 0 \qquad \forall i : 0 \leq \alpha_i \leq \frac{\xi_i}{C}$$

(b)

$$\min_{\alpha} \quad \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j {\mathbf{x}_i}^T\mathbf{x}_j - \frac{1}{2C}\sum_{i=1}^{n}\alpha_i^2$$

s.t.:

$$\sum_{i=1}^{n}\alpha_i y_i = 0 \qquad \forall i : \alpha_i \geq 0$$

(c)

$$\max_{\alpha, \xi} \quad \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j {\mathbf{x}_i}^T\mathbf{x}_j + C\sum_{i=1}^{n}\xi_i$$

s.t.:

$$\sum_{i=1}^{n}\alpha_i y_i = 0 \qquad \forall i : \alpha_i \geq 0$$

(d)

Answer:

3 pts.

Question 5: Mixture of Bernoulli models (25 pts.)

Let $\mathbf{x} = (x_1, \ldots, x_D)^T \in \{0, 1\}^D$ be a D-dimensional random binary vector. We assume that every x_i is Bernoulli distributed with parameter μ_i, i.e.

$$p(x_i = 1;\, \mu_i) = \mu_i$$
$$p(x_i = 0;\, \mu_i) = 1 - \mu_i,$$

which can be also written as

$$p(x_i;\, \mu_i) = \mu_i^{x_i}(1 - \mu_i)^{1 - x_i}.$$

Under the assumption of the x_i's being independent, the distribution of the random vector $\mathbf{x} = (x_1, \ldots, x_D)^T$ is

$$p(\mathbf{x};\, \boldsymbol{\mu}) = \prod_{i=1}^{D}\mu_i^{x_i}(1 - \mu_i)^{1 - x_i},$$

with $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_D)^T$.

a) What is $\mathbb{E}(\mathbf{x})$ and $\mathbf{Cov}(\mathbf{x})$ in this setting?
Answer:

3 pts.

b) Consider a Bernoulli mixture model for binary random vectors with K mixture components, i.e.

$$p(\mathbf{x};\, (\boldsymbol{\mu}_k)_k, \mathbf{w}) = \sum_{k=1}^{K} w_k p(\mathbf{x};\, \boldsymbol{\mu}_k)$$

with weights $\mathbf{w} = (w_1, \ldots, w_K)^T$ and Bernoulli parameters $(\boldsymbol{\mu}_k)_k = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K)$.

(Side remark, which is not relevant to solve the question: The number of mixture components K cannot be larger than the dimension D of the random vector to ensure identifiability.)

What are natural constraints on $\mathbf{w}$ to obtain a proper probability distribution $p(\mathbf{x};\, (\boldsymbol{\mu}_k)_k, \mathbf{w})$?
Answer:

2 pts.

c) Show that

$$\mathbb{E}(\mathbf{x}) = \sum_k w_k \boldsymbol{\mu}_k \qquad \text{and}$$

$$(\mathbf{Cov}(\mathbf{x}))_{i,j} = \begin{cases} \sum_k w_k \mu_{ki} - \mathbb{E}(\mathbf{x}_i)^2 & \text{if } i = j \\ \sum_k w_k \mu_{ki}\mu_{kj} - \mathbb{E}(\mathbf{x}_i)\mathbb{E}(\mathbf{x}_j) & \text{if } i \neq j \end{cases}$$

(the elements of the covariance matrix)

where μ_{ki} is the i-th component of $\boldsymbol{\mu}_k$.

Hints:

1. You do not need to include the last terms, $-\mathbb{E}(\mathbf{x}_i)^2$ and $-\mathbb{E}(\mathbf{x}_i)\mathbb{E}(\mathbf{x}_j)$, in your derivation, since these follow from the relation
$$\mathbf{Cov}(\mathbf{x}) = \mathbb{E}(\mathbf{x}\mathbf{x}^T) - \mathbb{E}(\mathbf{x})\mathbb{E}(\mathbf{x})^T.$$
Thus, you need to focus only on $\mathbb{E}(\mathbf{x}\mathbf{x}^T)$.
2. Consider $\mathbb{E}(\mathbf{x}_i\mathbf{x}_j)$ separately for the cases $i = j$ and $i \neq j$. Use the fact that $\mathbb{E}(\mathbf{x}_i\mathbf{x}_j) = P(\mathbf{x}_i = 1 \wedge \mathbf{x}_j = 1)$, since $\mathbf{x}$ is a binary vector.

What is the main (qualitative) difference between the covariance derived in a) and the one for the mixture model?

Answer:

5 pts.

d) In order to derive an EM-method to determine the mixture of Bernoulli paramaters, we introduce a latent binary vector $\mathbf{z} = (z_1, \ldots, z_K)^T \in \{0,1\}^K$ linked with $\mathbf{x}$, such that $\sum_k z_k = 1$ and $z_k = 1$ if $\mathbf{x}$ is generated by the k-th mixture component. Hence, we have

$$p(\mathbf{x} \mid \mathbf{z}) = \prod_{k=1}^{K} p(\mathbf{x};\, \boldsymbol{\mu}_k)^{z_k} \qquad \text{and} \qquad p(\mathbf{z}) = \prod_{k=1}^{K} w_k^{z_k}.$$

Derive $p(\mathbf{x})$ by marginalizing over $\mathbf{z}$. Answer:

3 pts.

e) M-step: let $\mathbf{X} := (\mathbf{x}_n)_n \in \mathbb{R}^{D\times N}$ be a sequence of N i.i.d. samples drawn from the mixture of Bernoulli distribution. We have a corresponding matrix of the latent variables $\mathbf{Z} := (\mathbf{z}_n)_n \in \mathbb{R}^{K\times N}$.
Write down the joint log-likelihood function $\ln p(\mathbf{X}, \mathbf{Z};\, (\boldsymbol{\mu}_k)_k, \mathbf{w})$ and derive the M-step by maximizing the joint log-likelihood over the unknown parameters $(\boldsymbol{\mu}_k)_k$ and $\mathbf{w}$. Hint: don't forget to introduce a Lagrange multiplier for the constraint on $\mathbf{w}$.
Answer:

6 pts.

f) E-step: derive the expectation of the joint log-likelihood function with respect to $\mathbf{Z}$. Hints:

1. Derive $p(\mathbf{Z} \mid \mathbf{X})$ first via $p(\mathbf{z})$ and $p(\mathbf{x} \mid \mathbf{z})$ given above.
2. Consider $\mathbb{E}(z_{nk}) = p(z_{nk} = 1)$, since $z_{nk} \in \{0,1\}$.
3. Observe that z_{nk} appears only linearly in the joint log-likelihood function.

Answer:

6 pts.

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Introduction to Machine Learning
2010

Prof. J. M. Buhmann

Final Exam
January 29th, 2011

First and Last name: ____________________

ETH number: ____________________

Signature: ____________________

	Topic	Max. Points	Points Achieved	Visum
1	Bayesian Inference	25		
2	Regression	25		
3	Bagging and Boosting	25		
4	SVM	25		
5	Mixture Models	25		
Total		125		

Grade: ....

Question 1: Bayesian Inference (25 pts.)

Consider the task of estimating mean μ and variance σ^2 of a Gaussian density from n i.i.d. observations $\mathcal{X} = \{x_1, \ldots, x_n\}$, $x \in \mathbb{R}$.

a) Write the maximum likelihood estimator for the mean $\widehat{\mu}_{\text{ML}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_1, \ldots, x_n\}$:
(please write the direct closed-form solution)

1. $\widehat{\mu}_{\text{ML}}(\mathcal{X}) = \arg\max p(\mathcal{X}|\mu) = \frac{1}{n}\sum_i^n x_i$ **1 pts.**

2. Is this a biased estimator? YES\NO **1 pts.**

b) Write the maximum likelihood estimator for the variance $\widehat{\sigma}^2_{\text{ML}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_1, \ldots, x_n\}$:
(please write the direct closed-form solution)

1. $\widehat{\sigma}^2_{\text{ML}}(\mathcal{X}) = \arg\max p(\mathcal{X}|\sigma^2) = \frac{1}{n}\sum_i^n (x_i - \mu)^2$ **1 pts.**

unbiased estimate of population variance: $\frac{1}{n-1}\sum_i^n (x_i - \mu)^2$

2. Is this a biased estimator? YES\NO **1 pts.**

Now, assume that the variance σ^2 is known.

c) Given a Gaussian prior over the mean (prior with zero mean and variance one), write the posterior density $p(\mu|\mathcal{X})$ as an explicit function of the single observation $\mathcal{X} = \{x_1\}$:
(please write the direct closed-form solution)

bc conjugate pair, write posterior distribution

$P(\mu) = \frac{1}{\sqrt{2\pi}} \exp \frac{-(x_i)^2}{2}$ $\quad = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \frac{-(x_i-\mu)^2}{2\sigma^2}$

1. $p(\mu|\mathcal{X}) = \frac{p(\mathcal{X}|\mu)p(\mu)}{p(\mathcal{X})} = \frac{1}{\sqrt{2\pi}\sqrt{2\pi\sigma^2}} \exp \frac{-(x_i)^2}{2} - \frac{(x_i-\mu)^2}{2\sigma^2}$ **2 pts.**

$= \frac{1}{(2\pi)^{\frac{1}{2}}\sigma} \exp \frac{-(x_i^2\sigma^2 + (x_i-\mu)^2)}{2\sigma^2}$ $\quad -\frac{x_i}{2} - \frac{(x_i-\mu)^2}{2\sigma^2} = \frac{-x_i\sigma^2 - (x_i-\mu)^2}{2\sigma^2}$

2. Is the posterior a Gaussian density? YES\NO **1 pts.**
3. Is the posterior a Gaussian density for all priors? YES\NO **1 pts.**

you have to consider all existing priors, not only gaussians

d) Given a Gaussian prior over the mean (prior with zero mean and variance one), write the maximum a posteriori estimator $\widehat{\mu}_{\text{MAP}}(\mathcal{X})$ as an explicit function of the i.i.d. observations $\mathcal{X} = \{x_1, \ldots, x_n\}$ (recall that $\int e^{-(\alpha^2)} d\alpha = \sqrt{(\pi)}$):
(please write the direct closed-form solution)

Not sure why you want to integrate instead of take derivative

1. $\widehat{\mu}_{\text{MAP}}(\mathcal{X}) = \arg\max p(\mu|\mathcal{X}) = \frac{n}{n+\sigma^2}\hat{\mu} + \frac{\sigma^2}{n+\sigma^2}\mu \sum_i^n x_i$ **2 pts.**

2. Does the maximum a posteriori estimate equal the mean of the posterior? YES\NO $\hat{\mu}_{MAP} = \mu_n$ **1 pts.**

Now consider a binary classification task from observations $\mathcal{X} = \{x_1, \ldots, x_n\}$, with $x \in \mathbb{R}^D$. Assume that the likelihood of both classes is Gaussian (assume class prior π_i, mean μ_i, and covariance matrix Σ_i for class y_i, with $i = 1, 2$).

GMM

e) Write the discriminant $g_{y_1}(x) = p(y_1|x)$ as an explicit function of class prior, mean and covariance:
(please write the direct closed-form solution)

$\sum_{i=1}^{2} \pi_i\, p(x|\mu_i, \Sigma_i)$

$g_{y_1}(x) = \frac{P(x|y_1, \mu_1, \Sigma_1) \cdot P(y_1|\mu_1, \Sigma_1)}{P(x|\mu_1, \Sigma_1)}$ **3 pts.**

$= \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma_1|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right) \cdot \pi_1$

f) Assume that $\Sigma_1 = \Sigma_2 = \sigma^2\mathbb{I}$, where $\mathbb{I}$ denotes the identity matrix. Write the equation satisfied by the separating decision surface. The equation must be an explicit function of x_1 (the single observation), of class prior, means, and covariance:
(please write the direct closed-form solution)

1. $0 = g_{y_1}(x) - g_{y_2}(x) = \left[\frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma_1|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x_1-\mu_1)^T\Sigma_1^{-1}(x_1-\mu_1)\right) \cdot \pi_1\right]$ **2 pts.**

$- \left[\frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma_2|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x_1-\mu_2)^T\Sigma_2^{-1}(x_1-\mu_2)\right) \cdot \pi_2\right]$

every time $\Sigma_i = \Sigma_j$ then $\rightarrow$ linear.

Duda pg 36

2. In this case, is the decision surface constant, linear, quadratic, cubic, or something else? constant\linear\quadratic\cubic\other **2 pts.**
3. Would it be possible to obtain a cylindrical decision surface? YES\NO **1 pts.**

Be linear.

$-\frac{1}{2}x_1^T(\Sigma_1^{-1} - \Sigma_2^{-1})x_1$

$a\, x_1 + $ ______

g) In the two dimensional case subject to uniform class prior, write means $(\overline{\mu}_1, \overline{\mu}_2)$ and covariances (Σ_1, Σ_2) such that the decision surface is a hyper-plane:
(any numerical instantiation which satisfies this constraint is acceptable)

$\mu_1 = \frac{\sum_i x_i P(\mu_1^{old}, \Sigma_1^{old}|x_i)}{\sum_i P(\mu_1^{old}, \Sigma_1^{old}|x_i)}$ $\quad \frac{1}{n}\sum x_i$ $\quad \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$\mu_2 = \frac{\sum_i x_i P(\mu_2^{old}, \Sigma_2^{old}|x_i)}{\sum_i P(\mu_2^{old}, \Sigma_2^{old}|x_i)}$ $\quad \begin{bmatrix} 2 \\ 3 \end{bmatrix}$

$\Sigma_1 = \frac{\sum_i (x_i - \mu_1^{new})^2 \cdot P(\mu_1^{old}, \Sigma_1^{old}|x_i)}{\sum_i P(\mu^{old}, \Sigma^{old}|x_i)}$

equal

$\Sigma_2 = \frac{\sum_i (x_i - \mu_2^{new})^2 \cdot P(\mu_2^{old}, \Sigma_2^{old}|x_i)}{\sum_i P(\mu^{old}, \Sigma^{old}|x_i)}$ **3 pts.**

h) In the two dimensional case subject to uniform class prior, write means (μ_1, μ_2) and covariances (Σ_1, Σ_2) such that the decision surface is spherical:
(any numerical instantiation which satisfies this constraint is acceptable)

$\mu_1 = \begin{bmatrix} \quad \end{bmatrix}$

$\mu_2 = \begin{bmatrix} \quad \end{bmatrix}$

$\Sigma_1 = \begin{bmatrix} \quad \end{bmatrix}$

$\Sigma_2 = \begin{bmatrix} \quad \end{bmatrix}$ **3 pts.**

Question 2: Regression (25 pts.)

Consider the linear regression model expressed in the homogeneous coordinates:

$$y = \beta_{(0)} + \sum_{i=1}^{D} \mathbf{x}_{(i)}\beta_{(i)} = \mathbf{x}^\top \beta,$$

where $\mathbf{x} = (1, \mathbf{x}_{(1)}, ..., \mathbf{x}_{(D)}) \in \mathbb{R}^{D+1}$ is the input variable and y is the corresponding target variable.

Assume that the input dataset is given by the matrix $\mathbf{X} \in \mathbb{R}^{N\times(D+1)}$ whose first column is $\mathbf{1}$. Then the linear regression model for all the observations is written as $\mathbf{y} = \mathbf{X}\beta$. Consider this linear regression model and answer the following questions:

a) For this problem, formally define the Residual Sum of Squares (RSS) cost function and write it down in matrix notation.
Answer:

$RSS = (y_{True} - \hat{y})^2$

$$RSS = (y_{True} - \beta^T x)^T (y_{True} - \beta^T x) \checkmark$$

2 pts.

b) Briefly motivate the connection between minimizing the Residual Sum of Squares and maximizing the likelihood of $\mathbf{y}$ given $\mathbf{X}$. ✓
Answer:

To minimize RSS is to reduce the error between the regressor and data points

maximizing likelihood $\max_\beta P(X|\beta) \Rightarrow$ finding the Betas coefficients of the regressor that will best "fit" the data.

Error of probability $P(\varepsilon_1, ..., \varepsilon_n|\beta) = \prod_{i=1}^{n} P(\varepsilon_i|\beta) = \prod_{i=1}^{n} P(y_i - x^T\beta|\beta)$

2 pts.

$Y = X\beta + \varepsilon$ $\quad \varepsilon_i \sim N(0, \sigma^2)$

$Y \sim N(XB, \sigma^2)$ $\quad \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(Y - XB)}{2\sigma^2}\right)$

$Y [Y_1 ... Y_N]$

c) We minimize the RSS function and infer the model parameters as $\hat{\beta} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$. Considering the matrix inverse operation, mathematically describe why in practice we are interested in regularized models such as *ridge* regression models rather than the given unregularized model.
Answer: $RSS(\beta) = (Y - \beta^T x)^T (Y - \beta^T x)$ $\quad \|\beta\|_R = \left(\sum_i |\beta_i|^2\right)^{\frac{1}{2}}$

$\hat{\beta} = (X^T X + \lambda I)^{-1} X^T Y$

we can directly manipulate size of β.

good for stabilizing matrix

Positive-definite

Full rank.

no corre when parameters > samples

In regularization, you assume β to be distributed by nature according to gaussian law. To choose β values that control complexity of model.

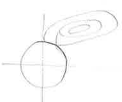
β is distributed by nature, $p(\beta) = \prod_{i=1}^{d} \frac{1}{\sqrt{2\pi\sigma_\beta^2}} \exp\left(\frac{-\beta^2}{2\sigma_\beta^2}\right) \Rightarrow$ TAke log of $P(\beta)$, $L_p = \left(\sum_i |x_i|^p\right)^{\frac{1}{p}}$

then arrive at $-\log p(\beta) = \sum_{j=1}^{d} \frac{\beta^2}{2\sigma_\beta^2} + \text{constant} \Rightarrow \arg\min_\beta \left\{ RSS(\beta) + \lambda \sum_{j=1}^{d} \beta^2 \right\}$

λ = larange multiplier

3 pts.

d) Formally define ridge and LASSO regression models. By depicting appropriate plots, demonstrate the difference between these two models in inferring the model parameters $\hat{\beta}$.
Answer:

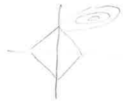


Ridge regression

L2-penalty $\quad \|\hat{\beta}\|_2 = \left(\sum_i |\beta_i|^2\right)^{\frac{1}{2}}$

shrinks the regression coefficients by imposing a penalty on their size.

X



LASSO → L1 penalty $\quad \|\hat{\beta}\|_1 = \left(\sum_i |\beta_i|\right)$

favors β values to be exactly ϕ when λ is large

Spareness penalty model with few coefficients non-vanishing.

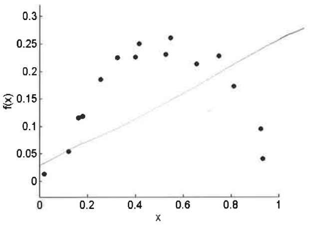
Because the least-square surface often hits the corners of the constraint surface.

$\varphi_1\beta_1 + \varphi_2\beta_2 = \varphi$

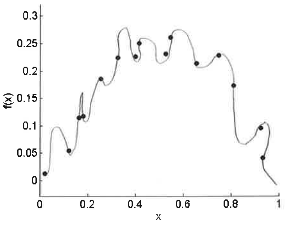
5 pts.

Now, we consider a general form of the problem where the set of observation $\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{x}_i \in \mathbb{R}^D, y_i \in \mathbb{R}$ are drawn i.i.d. from the joint distribution $P(X, Y)$. The goal is to find the regression function $f \in \mathcal{F}$ such that the mean squared error $\mathbb{E}_{XY}[(Y - f(X))^2]$ is minimal. Where the hypothesis class $\mathcal{F}$ contains the set of all polynomial functions.

e) Choosing an inappropriate function f might lead to either underfitting or overfitting. For the depicted given data, draw relevant plots to show each of these situations. For the overfitting case, briefly explain what happens if we have more observations.
Answer:



underfitting



overfitting

4 pts.

We can split the mean squared error $\mathbb{E}_{XY}[(Y - f(X))^2]$ into bias and variance, and find the best tradeoff between them.

f) Write down the mathematical definition of bias and variance.
Answer: MSE = Bias² + variance.

$$MSE = \left(\mathbb{E}[\hat{f}(x)] - f(x)\right)^2 + \mathbb{E}\left[\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right]^2$$

$Bias = \mathbb{E}[\hat{f}(x)] - \underbrace{f(x)}_{Y}$ $\quad$ variance $= \mathbb{E}\left[\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\right]^2$

2 pts.

g) Expand the mean squared error $\mathbb{E}_{XY}[(Y - f(X))^2]$ and write it in the form of variance + squared bias.
Answer:

$\phi \quad \hat{\phi}$

$$\mathbb{E}_{xy}[(Y - f(x))^2] = \mathbb{E}_{x,y}[f(x)^2 - 2Yf(x) + Y^2]$$

$$= \mathbb{E}_{x,y}[f(x)^2]$$

3 pts.

In this section we study how the averaging changes the bias of a set of unbiased estimators. Assume that we are given a set of B estimators $\hat{f}_1(\mathbf{x}), \hat{f}_2(\mathbf{x}), ..., \hat{f}_B(\mathbf{x})$. We take the average estimator by $\hat{f}(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(\mathbf{x})$.

h) Calculate the bias of the average estimator $\hat{f}(\mathbf{x})$ in terms of the bias of the estimators $\hat{f}_1(\mathbf{x}), ..., \hat{f}_B(\mathbf{x})$.
Hint: Start with the mathematical definition of the bias for $\hat{f}(\mathbf{x})$.
Answer:

$$\text{bias}(\hat{f}(x)) = \mathbb{E}[\hat{f}(x)] - \mathbb{E}[y|x] = \mathbb{E}\left[\frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(x)\right] - \mathbb{E}[y|x]$$

$$= \frac{1}{B}\sum_{b=1}^{B} \mathbb{E}[\hat{f}_b(x)] - \mathbb{E}[y|x] = \boxed{\frac{1}{B}\sum_{b=1}^{B} \text{bias}(\hat{f}_b(x))}$$ ✓

3 pts.

i) According to the results, briefly explain why unbiased estimators remain unbiased after averaging. ✓
Answer: Because taking the average of something gives you an unbiased. And in this case, if the estimator is already unbiased and taking the average of the different unbiased estimators again is unbiased.

1 pts.

Question 3: Bagging and Boosting (25 pts.)

Bagging and *boosting* are two possible approaches to combine multiple models (classifiers or regressors) to achieve a *composite model* with improved performance. Both methods can be used in both a classification and a regression setting.

a) State two essential differences between bagging and boosting.
Answer:

	Bagging	Boosting
1)	Bagging varies the training sets using resampling	Trains weak learners with the same training set on every iteration
2)	gives the same importance to every prediction	weights the prediction of every weak classifier according to its accuracy,

2 pts.

b) Explain in terms of the bias-variance trade-off why the idea of combining models works.
Answer:
without combining models, either you are trying to battle with having a high bias and low variance or a low bias and high variance.

when you combine models, you are gaining a diverse group of classifiers that partition the sample space in a complex manner that tends to reduce both the bias and variance.

Bagging → reduce variance

Boosting → reduce bias.
Give more weight to misclassified.

2 pts.

c) We now look at the problem of regression and how the combination of individual regression models can give better results. Left-column figures show the individual regression models, right-column figures show the true target function (solid) and the output of averaging the individual models to obtain a composite model (dashed).

1. The individual regression models (in Figure 1 and Figure 3) have been regularized using a regularization parameter λ, i.e. the cost function had the following form

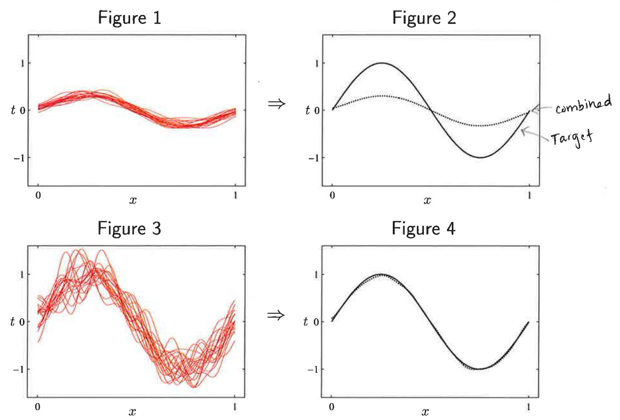
$$RSS_{Ridge}(\boldsymbol{\beta}) = \sum_{i=1}^{n}(t_i - \phi(\mathbf{x}_i)^\top \boldsymbol{\beta})^2 + \lambda||\boldsymbol{\beta}||_2^2$$

The parameters used were $\lambda = 0.09$ and $\lambda = 13.5$.
Associate the regularization parameters to the figures in the left column.
Answer:
Figure 1: $\lambda = ...$ 13.5 → large λ pulls the weight parameters toward zero leading to large bias.
Figure 3: $\lambda = ...$ 0.09 → Allow model become finely tuned to noise → leading to large variance

2. Please interpret the figures in terms of the bias-variance trade-off
Answer:
Figure 1 and 2: Bias is high, variance is low
rigid model.

Figure 3 and 4: Figure 3
Bias is low, variance is high

Figure 4 gives a low bias and variance.
flexible models



4 pts.

d) We now consider *bagging* in a regression setting. In order to model a *target function* $h(x)$, we combine M *individual models* $y_m(x), m = 1..M$ to obtain a *committee model* $y_{COM}(x)$. We model the *error* of each individual model using $\epsilon_m(x), m = 1..M$. This question's final goal is to show that the error of the committee model is M times smaller than the average error of the individual models.

1. Write down the output of the committee model, which averages the individual model's outputs. Answer:
   $y_{COM}(x) = \ldots \frac{1}{M}\sum_{m=1}^{M} y_m(x)$
2. Write down the error $\epsilon_m(x)$ of an individual model in terms of the target function $h(x)$ and the output of the individual model $y_m(x)$. Answer:
   $\epsilon_m(x) = h(x) - y_m(x)$
3. Write down the expected squared error of an individual model. $\mathbb{E}_x$ denotes the expectation value w.r.t. the distribution of x. Answer:
   $E_m = \mathbb{E}_x[\ \epsilon_m(x)^2\ ]$
4. Write down the average of the expected squared errors made by the individual models
   Answer:
   $E_{AV} = \ldots \frac{1}{M}\sum_{m=1}^{M} E_x[\epsilon_m(x)^2]$
5. Write down the expected squared error made by the committee model, in terms of the output of the committee model and the target function.
   Answer:
   $E_{COM} = \mathbb{E}_x[(h(x) - y_{com}(x))^2]$ $\quad E_x\left[\left(\frac{1}{M}\sum_m^M y_m(x) - h(x)\right)^2\right]$
6. Show that $E_{COM} = \frac{1}{M}E_{AV}$.
   Hint: Use the assumption that the errors of the individual models are uncorrelated, i.e. $\mathbb{E}_x[\epsilon_m(x)\epsilon_l(x)] = 0, m \neq l$, and that the mean error of the individual models is zero, i.e. $\mathbb{E}_x[\epsilon_m(x)] = 0$.
   Answer:

$E_{com} = E_x\left[\left(\frac{1}{M}\sum_m^M y_m(x) - h(x)\right)^2\right]$ $\quad \left(\frac{1}{M}\sum_m y_m(x) - h(x)\right)$

$E_{com} = \left(\frac{1}{M}\sum_m E_x[y_m(x) - h(x)]\right)^2$ $\quad \left(\frac{1}{M}\sum y_m\right.$

$= \left(\frac{1}{M}\sum_m E_x[\epsilon_m(x)]\right)^2$

$E_{com} = \frac{1}{M}\sum_m E_{AV}$ $\quad E[y_{com}] = \frac{1}{M^2}\sum_m E_x[e_i(x)^2]$

$\sum_i\sum_j E(e_i(x)e_j(x))$

$E\left[\left(\frac{1}{M}\sum y_m(x) - h(x)\right)\left(\frac{1}{M}\sum y_m(x) - h(x)\right)\right]$ $\quad y_i^2$ $\quad a^2 + 2ab + b^2$ $\quad \frac{1}{M^2}E$

10 pts.

We now turn our attention to *boosting* in a classification setting.

e) Given i.i.d. training data $\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), ..., (\mathbf{x}_n, c_n)\}$, $\mathbf{x}_i \in \mathbb{R}^D$, $c_i \in \{-1, 1\}$, and individual classifiers $y_m(\mathbf{x}), m = 1...M$, we can define the following additive model

$$f_k(\mathbf{x}) = \frac{1}{2}\sum_{j=1}^{k} \alpha_j y_j(\mathbf{x})$$

where α_j are weights, $k \leq M$.
Think of $f_k(\mathbf{x})$ as the *additive model* which uses the first $1..k$ individual classifiers.

1. Define the sum-of-squares error of $f_k(\mathbf{x})$ on the training dataset.
   Answer: here we consider an exponential error fxn
   $Err_k = \ldots \sum_{i=1}^{n} \cancel{\exp(-c_i f_K(x_i))}$
2. Assume that the first $1..(k-1)$ classifiers $y_j(\mathbf{x})$ and weights α_j, $j = 1..(k-1)$ have already been determined, i.e. fixed.

   Show that finding the optimal k-th classifier $y_k(\mathbf{x})$ and its weight α_k involves fitting the k-th classifier to the residual errors $f_{k-1}(\mathbf{x}_i) - c_i$ made by the $(k-1)$-th additive model.

   (*Hint: Start from the expression Err_k and recognize the terms $f_{k-1}(\mathbf{x}_i) - c_i$*) $\quad y_k(x) = ? \quad \alpha_k = ?$

   $\sum_{i=1}^{n} \exp(-c_i f_k(x_i))$ $\quad \alpha = (f_{k-1}(x_i) - c_i)$

   $y_k(x$

   $\boxed{\sum_i^n (c_i - f_k(x_i))^2}$ $\quad \sum_i \left((f_{k-1}(x_i) - c_i) - f_k(x)\right)^2$

7 pts.

Question 4: SVM (25 pts.)

In the following questions choose one answer only.

a) Why does a Support Vector Machine generalizes better than a Perceptron?
1. The selected support vectors tend to be very typical samples. border case.
2. By supporting vectors it is not restricted to scalar input.
3. The requirement of maximal margins reduces the arbitrariness.
4. The support vector machine will have access to more training data.

2 pts.

b) What is the advantage of using kernels in support vector machines?
1. They tend to maximize the margins.
2. They enable non-linear separations.
3. They will increase the number of support vectors.
4. They reduce the risk of getting stuck in local minima.

2 pts.

Explain your answer to the following questions in 1-2 sentences.

c) Suppose that you have a binary SVM classifier with a linear kernel. Consider a vector $\mathbf{x}_i$ for which $y_i(\langle\mathbf{w}, \mathbf{x}_i\rangle + w_0) > 1 \quad (\mathbf{x}_i \in \mathbb{R}^D, y_i \in \{-1, +1\})$

1. Is $\mathbf{x}_i$ correctly classified? YES\NO
2. If we remove $\mathbf{x}_i$ from the training set and re-train the classifier, will the decision boundary change or stay the same?

Answer:
The decision boundary might change depending if the training sample was a support vector in the previous case.

3 pts.

d) We would like to build a binary SVM classifier for histograms. Let H denote the class of histograms, for each $\mathbf{h} \in H$ the following properties hold:

class of histograms

1. $\mathbf{h} = \{h_1, \ldots, h_m\}, \quad \forall j : h_j \geq 0$
2. $\sum_{j=1}^{m} h_j = L \quad$ for some $L \in \mathbb{N}$.

Consider using the following kernel function:

$$K(\mathbf{h}, \tilde{\mathbf{h}}) = \sum_{j=1}^{m} \min(h_j, \tilde{h}_j) \quad \mathbf{h}, \tilde{\mathbf{h}} \in H$$

What kind of separation is imposed by this kernel? (which points in the space will become close/far using this kernel).
Answer:

3 pts.

e) Let $\mathcal{X}$ be a finite set and consider the following function. For $\mathbf{x}, \bar{\mathbf{x}} \in \mathcal{X}$:

$$K(\mathbf{x}, \bar{\mathbf{x}}) = \begin{cases} 1 & \text{if } \mathbf{x} = \bar{\mathbf{x}} \\ 0 & \text{else} \end{cases}$$

Prove that $K(\mathbf{x}, \bar{\mathbf{x}})$ is a legitimate kernel function.

Hint: One possible way to prove it is to find a feature mapping $\phi(\mathbf{x})$, such that
$K(\mathbf{x}, \bar{\mathbf{x}}) = \langle\phi(\mathbf{x}), \phi(\bar{\mathbf{x}})\rangle$. $\quad x = (x_1, x_2)$ $\quad \bar{x} = (\bar{x}_1, \bar{x}_2)$
Answer:

$\langle\phi(x), \phi(\bar{x})\rangle = \phi(x_1, x_2)\cdot\phi(\bar{x}_1, \bar{x}_2)$

3 pts.

$L2$-SVM use the square sum of the slack variables ξ_i in the objective function instead of the linear sum of the slack variables (squaring the hinge loss). Let $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ be a training set of examples and binary labels $y_i \in \{-1, +1\}$. The primal formulation of the $L2$-SVM is as follows

$$\min_{\mathbf{w}, w_0, \xi} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{2}\sum_{i=1}^{n} \xi_i^2 \qquad w = \sum_{1}^{n} \alpha_i$$

s.t.

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + w_0) \geqslant 1 - \xi_i \quad i = 1, \dots, n$$
$$\xi_i \geqslant 0 \quad i = 1, \dots, n$$

$$y_i(\langle w, x_i \rangle + w_0) - 1 + \xi_i$$

f) Show that removing the last set of constraints: $\forall i : \xi_i \geqslant 0$ does not change the optimal solution to the primal problem.
Answer:

$$\lambda y_i w^T x + w_0 - 1$$

$$L(w, w_0, \xi, \lambda_1, \lambda_2) = \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_i^n \xi_i^2 + \lambda_1(y_i(\langle w, x_i \rangle + w_0) - 1 + \xi_i) + \lambda_2(\xi) \stackrel{!}{=} 0$$

$$\frac{d}{d\lambda_1} = y_i(\langle w, x_i \rangle + w_0 - 1 + \xi_i) = 0$$

$$\frac{d}{d\lambda_2} = \xi = 0$$

$$\frac{d}{d\bar{w}} = \bar{w} + \lambda_1 y_i x = 0 \quad \lambda_1 = -\frac{\bar{w}}{y_i x}$$

$$\frac{d}{dw_0} = \lambda_1 = 0$$

$$= \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_i^n \xi_i^2 - \frac{\vec{w}}{y_i x}(y_i(w^T x_i + w_0) - 1 + \xi_i) + \cancel{\lambda_2}(0).$$

4 pts.

w/out $\xi \geq 0$ constraint

$$= \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_i^n \xi_i^2 - \frac{\vec{w}}{y_i x}(y_i(w^T x_i + w_0) - 1 + \xi_i \quad X$$

Now we would like to derive the dual form of the $L2$-SVM.

g) Write down the Lagrangian of the $L2$-SVM.
Answer:

$L(\qquad) =$

2 pts.

h) Compute the derivatives of the Lagrangian with respect to the appropriate variables.
Answer:

3 pts.

We decided not to bother you with the remaining computation, instead we finished the derivation ourselves.

i) Below are 4 optimization problems, only 1 of which is the dual $L2$-SVM. Circle the optimization problem corresponding to the dual $L2$-SVM.
Explain your choice by either shortly falsifying the other options or by showing the full derivation.
Hint: The full derivation is more time consuming.

Derive dual form.

(a)

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j {\mathbf{x}_i}^T \mathbf{x}_j - \frac{1}{2C}\sum_{i=1}^{n} \alpha_i^2$$

s.t.:

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$
$$\forall i : \alpha_i \geq 0$$

(b)

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j {\mathbf{x}_i}^T \mathbf{x}_j$$

s.t.:

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \quad ✓$$
$$\forall i : 0 \leq \alpha_i \leq \frac{\xi_i}{C}$$

(c)

no y's

$$\min_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j {\mathbf{x}_i}^T \mathbf{x}_j - \frac{1}{2C}\sum_{i=1}^{n} \alpha_i^2$$

s.t.:

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$
$$\forall i : \alpha_i \geq 0$$

(d)

$$\max_{\alpha, \xi} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j {\mathbf{x}_i}^T \mathbf{x}_j + C\sum_{i=1}^{n} \xi_i$$

slack

s.t.:

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$
$$\forall i : \alpha_i \geq 0$$

doesn't include constraint for C.

Answer:

IN the primal form you solved for w's and pluged it into dual form $w = \sum_i \alpha_i y_i x_i$.

3 pts.

Question 5: Mixture of Bernoulli models (25 pts.)

Let $\mathbf{x} = (x_1, \dots, x_D)^T \in \{0, 1\}^D$ be a D-dimensional random binary vector. We assume that every x_i is Bernoulli distributed with parameter μ_i, i.e.

$$p(x_i = 1; \mu_i) = \mu_i$$
$$p(x_i = 0; \mu_i) = 1 - \mu_i,$$

which can be also written as

$$p(x_i; \mu_i) = \mu_i^{x_i}(1 - \mu_i)^{1 - x_i}.$$

Under the assumption of the x_i's being independent, the distribution of the random vector $\mathbf{x} = (x_1, \dots, x_D)^T$ is

$$p(\mathbf{x}; \boldsymbol{\mu}) = \prod_{i=1}^{D} \mu_i^{x_i}(1 - \mu_i)^{1 - x_i}, \qquad \mathrm{COV}(X) = E[XX^T] - E[X]E[X]^T$$

with $\boldsymbol{\mu} = (\mu_1, \dots, \mu_D)^T$.

$E[\mu_i^{x_i}$

a) What is $\mathbb{E}(\mathbf{x})$ and $\mathbf{Cov}(\mathbf{x})$ in this setting?
Answer:

$$\mathbb{E}(X) = \mathbb{E}_x\left[\prod_{i=1}^{D} \mu_i^{x_i}(1-\mu_i)^{1-x_i}\right] = \cancel{\log\left[\prod \mu_i^{x_i}(1-\mu_i)^{1-x_i}\right]}$$

$$\boxed{\mathbb{E}[X] = \mu}$$

$$= \log(\mu_i^{x_i}) + \log((1-\mu_i)^{1-x_i}) = x_i \log(\mu_i) + (1-x_i)\log(1-\mu_i)$$

$$= E_x[x_i \log(\mu_i)] + E_x[1 - x_i \log(1-\mu_i)] = \log(\mu_i) E_x[x_i]$$

$\mu \qquad (1-\mu)$

$\mathrm{COV}(X) =$ $\mathrm{Diag}[\mu_i(1-\mu_i)] = I(\mu_i(1-\mu_i))$

3 pts.

b) Consider a Bernoulli mixture model for binary random vectors with K mixture components, i.e.

prior, weights   likelihood

$$p(\mathbf{x}; (\boldsymbol{\mu}_k)_k, \mathbf{w}) = \sum_{k=1}^{K} w_k p(\mathbf{x}; \boldsymbol{\mu}_k)$$

with weights $\mathbf{w} = (w_1, \dots, w_K)^T$ and Bernoulli parameters $(\boldsymbol{\mu}_k)_k = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K)$.

(Side remark, which is not relevant to solve the question: The number of mixture components K cannot be larger than the dimension D of the random vector to ensure identifiability.)

What are natural constraints on $\mathbf{w}$ to obtain a proper probability distribution $p(\mathbf{x};\ (\boldsymbol{\mu}_k)_k, \mathbf{w})$?
Answer:

2 pts.

c) Show that

$$\mathbb{E}(\mathbf{x}) = \sum_k w_k \boldsymbol{\mu}_k \qquad \text{and}$$

$$(\mathbf{Cov}(\mathbf{x}))_{i,j} = \begin{cases} \sum_k w_k \mu_{ki} - \mathbb{E}(\mathbf{x}_i)^2 & \text{if } i = j \\ \sum_k w_k \mu_{ki} \mu_{kj} - \mathbb{E}(\mathbf{x}_i)\mathbb{E}(\mathbf{x}_j) & \text{if } i \neq j \end{cases}$$

(the elements of the covariance matrix)

where μ_{ki} is the i-th component of $\boldsymbol{\mu}_k$.

Hints:

1. You do not need to include the last terms, $-\mathbb{E}(\mathbf{x}_i)^2$ and $-\mathbb{E}(\mathbf{x}_i)\mathbb{E}(\mathbf{x}_j)$, in your derivation, since these follow from the relation

$$\mathbf{Cov}(\mathbf{x}) = \mathbb{E}(\mathbf{x}\mathbf{x}^T) - \mathbb{E}(\mathbf{x})\mathbb{E}(\mathbf{x})^T.$$

   Thus, you need to focus only on $\mathbb{E}(\mathbf{x}\mathbf{x}^T)$.

2. Consider $\mathbb{E}(\mathbf{x}_i\mathbf{x}_j)$ separately for the cases $i = j$ and $i \neq j$. Use the fact that $\mathbb{E}(\mathbf{x}_i\mathbf{x}_j) = P(\mathbf{x}_i = 1 \wedge \mathbf{x}_j = 1)$, since $\mathbf{x}$ is a binary vector.

What is the main (qualitative) difference between the covariance derived in a) and the one for the mixture model?

Answer:

5 pts.

d) In order to derive an EM-method to determine the mixture of Bernoulli paramaters, we introduce a latent binary vector $\mathbf{z} = (z_1, \ldots, z_K)^T \in \{0, 1\}^K$ linked with $\mathbf{x}$, such that $\sum_k z_k = 1$ and $z_k = 1$ if $\mathbf{x}$ is generated by the k-th mixture component. Hence, we have

$$p(\mathbf{x} \mid \mathbf{z}) = \prod_{k=1}^{K} p(\mathbf{x};\ \boldsymbol{\mu}_k)^{z_k} \qquad \text{and} \qquad p(\mathbf{z}) = \prod_{k=1}^{K} w_k^{z_k}.$$

Derive $p(\mathbf{x})$ by marginalizing over $\mathbf{z}$. Answer:

$p(z|x) = \dfrac{P(x|z)^{z_k} \cdot P(z)}{p(x)}$. $\quad p(x) = \int p(x|z)^{z_k} \cdot p(z)$

3 pts.

e) M-step: let $\mathbf{X} := (\mathbf{x}_n)_n \in \mathbb{R}^{D \times N}$ be a sequence of N i.i.d. samples drawn from the mixture of Bernoulli distribution. We have a corresponding matrix of the latent variables $\mathbf{Z} := (\mathbf{z}_n)_n \in \mathbb{R}^{K \times N}$.
Write down the joint log-likelihood function $\ln p(\mathbf{X}, \mathbf{Z};\ (\boldsymbol{\mu}_k)_k, \mathbf{w})$ and derive the M-step by maximizing the joint log-likelihood over the unknown parameters $(\boldsymbol{\mu}_k)_k$ and $\mathbf{w}$. Hint: don't forget to introduce a Lagrange multiplier for the constraint on $\mathbf{w}$.
Answer:

6 pts.

f) E-step: derive the expectation of the joint log-likelihood function with respect to $\mathbf{Z}$. Hints:

1. Derive $p(\mathbf{Z} \mid \mathbf{X})$ first via $p(\mathbf{z})$ and $p(\mathbf{x} \mid \mathbf{z})$ given above.
2. Consider $\mathbb{E}(z_{nk}) = p(z_{nk} = 1)$, since $z_{nk} \in \{0, 1\}$.
3. Observe that z_{nk} appears only linearly in the joint log-likelihood function.

Answer:

$p(z|x) = \dfrac{p(x|z) \cdot p(z)}{p(x)} = \dfrac{\prod_k^K p(x|\mu_k)^{z_k} \cdot \prod_k^k w_k^{z_k}}{p(x)}$

$\prod_k^K p(x|\mu_k)^{z_k}$

log-likelihood fxn:

samples mixtures

$\log p(x, z|\mu_k) = \sum_i^N \sum_j^K z_{ij} \left(\log w_k \right) +$

$\sum_k^D (x_{nk} \ln \mu$

6 pts.

30 Exam 2010 solutions

These are the solutions for the exam of 2010. Unfortunately, nobody wrote them yet.

These solutions might be plain wrong since they were not verified by a teaching assistant but were written by a student. Be cautious! We hope that they help you even though.

Write the solutions for exam 2010.

ETH Eidgenössische Technische Hochschule Zürich Swiss Federal Institute of Technology Zurich

Introduction to Machine Learning

2010

Prof. J. M. Buhmann

Final Exam

February 10th, 2010

First and Last name: ______________________

ETH number: ______________________

Signature: ______________________

General Remarks

- Please check that you have all 21 pages of this exam.
- Remove all material from your desk which is not permitted by the examination regulations.
- Fill in your first and last name and your ETH number and sign the exam. Place your student ID in front of you.
- You have 2 hours for the exam. There are five questions, where you can earn a total of 150 points. Scoring 120 points (equivalent to solving four questions) guarantees you a grade of six.
- Write your answers directly on the exam sheets. If you need more space, put your name and ETH number on top of each supplementary sheet.
- Answer the questions in English. Do not use a pencil or red color pen.
- You may provide at most one valid answer per question. Invalid solutions must be canceled out clearly.

	Topic	Max. Points	Points Achieved	Visum
1	Density Estimation	30		
2	PCA	30		
3	Classifiers	30		
4	Regression	30		
5	Boosting	30		
Total		150		

Grade: ....

Question 1: Parametric Density Estimation (30 pts.)

When estimating densities from data, we assume that the data distribution is well approximated by a parametric model $P(x;\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ denotes the parameters of the model. Estimation of the density then reduces to estimation of the parameter $\boldsymbol{\theta}$.

a) In maximum-likelihood estimation (MLE), we usually start by assuming that the data is given as an i.i.d. sample $x_1, x_2, ..., x_N \sim P(x;\theta)$. Please explain how this assumption simplifies the calculations.
Answer:

2 pts.

b) We model a coin flip as a random variable X, where $X = 1$ denotes the outcome 'heads', and $X = 0$ denotes 'tails'. We model this event using the Bernoulli distribution, parametrized by μ that denotes $P(X = 1)$.

$$\text{Bern}(x|\mu) = \mu^x(1-\mu)^{1-x}.$$

Given a data set $\mathcal{X}_N = \{x_1, ..., x_N\}$ of N i.i.d. observations of X, show us step by step how to compute the MLE for μ. Comment your calculations where necessary.
Answer:

8 pts.

c) We flip our coin three times and obtain the dataset $\mathcal{X}_3 = \{1, 1, 1\}$.
 1. What is the MLE for $\hat{\mu}$ for dataset $\mathcal{X}_3$?
 Answer:
 2. State one problem with this estimate. What is a possible remedy (solution)?
 Answer:

3 pts.

d) Assume that there are a total of m heads in $\mathcal{X}_N$, i.e., $\sum_{i=1}^N x_i = m$.
 1. In terms of μ, N and m only, what is the probability of obtaining this dataset $P(\mathcal{X}_N|\mu)$? Explain the terms of your solution.
 Answer:
 2. What is the expected value of m? (*Hint: For independent random variables, the expectation of a sum is the sum of expectations*)
 Answer:
 $\mathbb{E}[m] =$

5 pts.

We now turn to Bayesian estimation of μ given a data set $\mathcal{X}$, by introducing a prior distribution $P(\mu)$. Here, we use the (conjugate) prior given by the Beta distribution:

$$\text{Beta}(\mu|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\mu^{\alpha-1}(1-\mu)^{\beta-1},$$

where $\Gamma(x)$ denotes the gamma function (not necessary). Some useful properties:

1. Mean $\mathbb{E}(\mu) = \frac{\alpha}{\alpha+\beta}$
2. Variance $\mathbb{V}(\mu) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$

e) With the mean and variance of μ, we can understand how the parameters influence the prior. Select the best (α_0, β_0) pair for the prior that emphasizes the following beliefs.
 1. My coin is biased towards 'tails'.
 ☐$(0.5, 0.5)$ ☐$(1, 4)$ ☐$(4, 1)$ ☐$(2, 2)$
 2. My coin is most probably biased (but I don't know heads or tails)
 ☐$(0.5, 0.5)$ ☐$(1, 4)$ ☐$(4, 1)$ ☐$(2, 2)$

3 pts.

f) Show that if the prior is Beta distributed

$$P(\mu) = \text{Beta}(\alpha_0, \beta_0),$$

then the posterior takes again the same functional dependency on μ as the prior, i.e.,

$$P(\mu|\mathcal{X}_N) \propto \mu^{\alpha_0+m-1}(1-\mu)^{\beta_0+N-m-1}.$$

Answer:

5 pts.

g) In fact, the posterior is again a Beta distribution, and we obtain $P(X=1|\mathcal{X}) = \frac{m+\alpha_0}{N+\alpha_0+\beta_0}$.

1. For $\mathcal{X}_3 = \{1,1,1\}$, use a choice of (α_0, β_0) which assumes a fair coin, and obtain your estimate for $P(X=1|\mathcal{X})$. How does this estimate differ from the MLE?
Answer:

2. What happens to the posterior as $N \to \infty$? How does this relate to the MLE?
Answer:

4 pts.

Question 2: Principal Component Analysis (PCA) (30 pts.)

Principal component analysis is a widely applied method in data analysis and dimensionality reduction. Given a dataset $\mathbf{X} \in \mathbb{R}^{D \times N}$ (observations as columns), where D is the number of dimensions and N is the number of observations, a linear transformation using an orthonormal matrix $\mathbf{P} \in \mathbb{R}^{M \times D}$ is applied to make a *change of basis*, to obtain a (usually) lower-dimensional dataset $\mathbf{Y} \in \mathbb{R}^{M \times N}$.

a) We begin by reviewing the steps of applying PCA to a dataset $\mathbf{X}$. Please complete each step below by providing the appropriate formula to compute the desired quantity.

1. Define the zero-mean dataset $\tilde{\mathbf{X}}$ in terms of the original dataset $\mathbf{X}$. For this purpose, introduce the data mean as a column-vector $\boldsymbol{\mu} \in \mathbb{R}^{D \times 1}$.
Answer:

2. Define the covariance matrix $\mathbf{C}_{\tilde{\mathbf{X}}}$ in terms of the zero-mean dataset $\tilde{\mathbf{X}}$.
Answer:

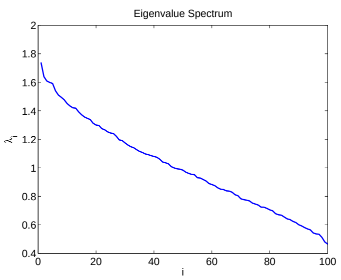
3. Write down the eigen-decomposition of the covariance matrix $\mathbf{C}_{\tilde{\mathbf{X}}}$, in terms of the eigenvector matrix $\mathbf{E}$ (eigenvectors as columns) and the diagonal matrix of eigenvalues $\mathbf{D}$.
Answer:

4. Define the PCA transformation matrix $\mathbf{P}$ in terms of the eigenvector matrix $\mathbf{E}$. (Note: assume we don't want to reduce the dimensionality of the transformed dataset. Further assume that the eigenvectors in $\mathbf{E}$ have already been sorted according to the corresponding eigenvalues, in decreasing order.)
Answer:

5. Define the transformed dataset $\mathbf{Y}$ in terms of the original dataset $\mathbf{X}$ and the transformation matrix $\mathbf{P}$.
Answer:

6 pts.

b) Assume we have applied PCA to some dataset ($D = 100$). We observe the following eigenvalue spectrum of the covariance matrix of the data. (λ_i: eigenvalues)



1. Is the intrinsic dimensionality of this dataset low or high? Why?
Answer:

2. Can this dataset be expressed in few dimensions with low approximation error? Why?
Answer:

3. If yes, which dimensionality (approximately) should be chosen for the transformed dataset and why?
Answer:

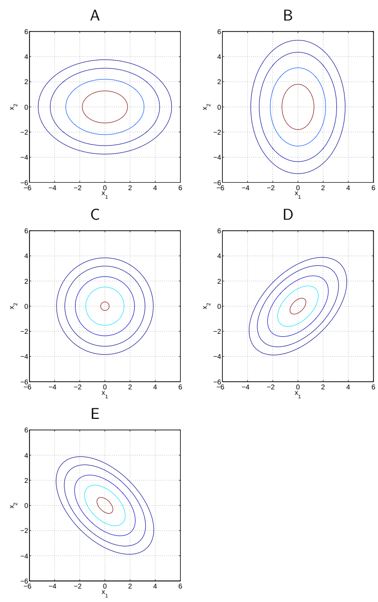
5 pts.

c) Assume you have observed 2D data $\mathbf{X} \in \mathbb{R}^{2 \times N}$ (observations as columns). The first row of $\mathbf{X}$ corresponds to the first dimension x_1, the second row corresponds to x_2. For each of the three covariance matrices $\mathbf{C}_{\mathbf{X}}$ below, please choose the iso-line plot (A-E) corresponding to the covariance matrix. (Note the axis labels on the figures)

1. $\begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$ Answer: ( )

2. $\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}$ Answer: ( )

3. $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ Answer: ( )



3 pts.

d) PCA can be derived as follows: Given a dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\} \in \mathbb{R}^{D \times N}$, find a new set of basis vectors such that the variance of the projected dataset is maximized. We begin by finding the first projection direction $\mathbf{e}_1 \in \mathbb{R}^D$, $||\mathbf{e}_1||_2 = 1$, which satisfies our goal.

1. Please describe in words why we are interested to find a projection direction such that the projected data has large variance.
   Answer:

2. Write down formally the objective function for maximizing the variance of the projected dataset. For this purpose, define the variance of the projected dataset in terms of the original data $\mathbf{x}_n$ and the first projection direction $\mathbf{e}_1$. (Hint: introduce the mean $\boldsymbol{\mu}$ of the data $\mathbf{X}$.)
   Answer:

5 pts.

e) PCA transforms a dataset $\mathbf{X}$ into a dataset $\mathbf{Y}$ by defining a new basis using the eigenvectors of the covariance matrix $\mathbf{C_X}$ of the dataset $\mathbf{X}$. With this particular choice of a new basis, the covariance matrix $\mathbf{C_Y}$ of the transformed dataset $\mathbf{Y}$ is *diagonalized.* (Note: For this exercise, assume a zero-mean dataset.)

1. Please explain in words, why we desire the covariance matrix of the transformed dataset to be diagonal.
   Answer:

2. Show that $\mathbf{C_Y} = \mathbf{P C_X P}^T$, i.e., that the covariance matrix $\mathbf{C_Y}$ of the transformed dataset can be written in terms of the covariance matrix $\mathbf{C_X}$ of the original dataset.
   Answer:

3. Show that the choice $\mathbf{P} = \mathbf{E}^T$ for the PCA transformation matrix, where $\mathbf{E}$ is the matrix of eigenvectors of the covariance matrix $\mathbf{C_X}$, actually diagonalizes the covariance matrix $\mathbf{C_Y}$ of the transformed dataset $\mathbf{Y}$. Use the eigen decomposition of $\mathbf{C_X}$ and the fact that $\mathbf{P}^{-1} = \mathbf{P}^T$.
   Answer:

11 pts.

Question 3: Linear Classifiers (30 pts.)

The simplest representation of a linear discriminant function is obtained by taking a linear function of the input vector $\boldsymbol{x}$ so that

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + b$$

a) Define the term decision boundary. Explain what is the relation between the vector $\boldsymbol{w}$ and the decision boundary.
   Answer:

2 pts.

b) Consider the following set $S \subseteq \mathbb{R}^2 \times \{+1, -1\}$:

$$S = \{((3,3),+1),((1,4),+1),((2,1),-1),((5,1),-1),((6,2),+1),((4,3),+1),((4,1),-1)\}$$

Prove that there exist no homogeneous half-space that separates S with no errors (recall that a half-space is homogeneous if its separating hyperplane passes through the origin). Assume that the classification of a point is given by $y(\boldsymbol{x}) = \mathrm{sign}(\boldsymbol{w}^T \boldsymbol{x} + b)$.
Answer:

5 pts.

The simple form of the online perceptron algorithm is defined as follows:

Initialize by setting $\boldsymbol{w}^{(0)} = \mathbf{0}$.

$$\text{Update } \boldsymbol{w}^{(t+1)} = \begin{cases} \boldsymbol{w}^{(t)} & y_k(\boldsymbol{w}^{(t)T} \boldsymbol{x}_k) > 0 \\ \boldsymbol{w}^{(t)} + y_k \boldsymbol{x}_k & \text{otherwise.} \end{cases}$$

c) We would like to use the perceptron algorithm to find a separating half-space that makes no errors on the set S. Recall that you have just proven that this is impossible using a homogeneous half-space. Make the necessary adjustments and perform 7 iterations of the perceptron algorithm. For each iteration write down the training example and the current vector $\boldsymbol{w}^{(t)}$. Assume that the points are given in the same order as they appear in S.
Answer:

5 pts.

Consider the primal form of the soft margin SVM

$$\begin{aligned} &\min \tfrac{1}{2}\|\boldsymbol{w}\|^2 + C \sum_{i=1}^{n} \xi_i \\ &s.t. \qquad y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) \geq 1 - \xi_i, \qquad i = 1, \ldots, n \end{aligned}$$

d) Indicate which of the following statements hold as we increase the value of the parameter C (relative to any starting value, $C > 0$). Use

D - if the validity of the statement depends on the situation,
T - if the statement is necessary true
N - if the statement is never true

[ ] b will not increase

[ ] $\|\boldsymbol{w}\|$ increases

[ ] $\|\boldsymbol{w}\|$ will not decrease

[ ] more points will be misclassified

[ ] The geometric margin will not increase

10 pts.

The dual soft margin SVM is given below

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$
$$s.t. \qquad 0 \le \alpha_i \le C, \qquad i = 1, \ldots, n$$

We would like to solve the SVM problem using the kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 + (\boldsymbol{x}_i^T \boldsymbol{x}_j)^2$.

e) What is the feature representation $\phi(\boldsymbol{x})$ corresponding to this kernel for $\boldsymbol{x} \in \mathbb{R}^2$?
Answer:

2 pts.

f) Using the dual variables α's and $\phi(\boldsymbol{x})$, give an explicit form for the primal variable $\boldsymbol{w}$.
Answer:

2 pts.

We solved the dual SVM problem with the above kernel. We used the following training set $\boldsymbol{x}_1 = (1, 1), y_1 = +1$, $\boldsymbol{x}_2 = (3, 0), y_2 = -1$, $\boldsymbol{x}_3 = (-2, 1), y_3 = -1$.
Below are the α's and b values
$\alpha_1 = 0.1$, $\alpha_2 = 0$, $\alpha_3 = 0.1$, $b = 1.5$

g) Use these values to compute the kernelized discriminant function $y(\boldsymbol{x})$.
Answer:

4 pts.

Question 4: Regression (30 pts.)

In *univariate regression analysis*, we study the *statistical* dependency of the output variable $y \in \mathbb{R}$ on the input variable $x \in \mathbb{R}$.

a) Complete the equation

$$y = \qquad\qquad + \qquad\qquad \tag{1}$$

where the right hand side consists of two terms. Define each term in the equation.
Note: Be general, don't restrict yourself to linear least-squares regression.
Answer:

3 pts.

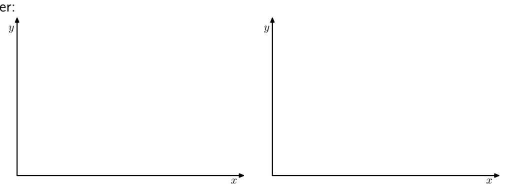
b) Regression analysis is only meaningful if there actually is a true dependency between x and y. Prove that if x and y are *statistically independent*, there is no linear dependency:
1. Assume that x and y are statistically independent.
2. Begin with the definition of *covariance*,
$$\operatorname{cov}(x, y) = \mathbb{E}\left[(x - \mu_x)(y - \mu_y)\right],$$
where $\mathbb{E}$ is the expectation operator and μ_x is the mean of x.
3. Show that x and y are uncorrelated.

Note: Comment your calculations where necessary.
Answer:

5 pts.

c) You have shown that eq. (1) consists of two terms. Illustrate two conceptually different cases where a linear least-squares fit is inappropriate for the true dependency between x and y, one related to the first term in eq. (1), and one related to the second term in eq. (1):
1. Draw a scatter plot of a data sample.
2. Fit the linear least-squares solution into the sample (qualitatively).
3. Explain why the solution does not capture the true dependency between x and y.

Answer:



4 pts.

So far, we considered a single input variable x. In multivariate linear regression, we assume a weighted linear functional dependency

$$y = w_0 + w_1 x_1 + \cdots + w_D x_D \tag{2}$$

between D input variables $x_1, \ldots, x_D \in \mathbb{R}$ and output variable y.

d) Define the input data matrix $\mathbf{X}$, weight vector $\mathbf{w}$ and output data vector $\mathbf{y}$ such that eq. (2) for all N data points of the sample $(x_1^n, x_2^n, \ldots, x_D^n, y^n), n = 1, \ldots, N$ can be written as
$$\mathbf{y} = \mathbf{X}\mathbf{w}.$$
Answer:

$\mathbf{X} :=$ $\qquad$ $\mathbf{w} :=$ $\qquad$ $\mathbf{y} :=$

3 pts.

The linear least-squares prediction $\hat{\mathbf{y}}$ is the *orthogonal projection* of $\mathbf{y}$ onto the space spanned by the columns of $\mathbf{X}$, and can be written as

$$\hat{\mathbf{y}} = \mathbf{P}\mathbf{y}.$$

e) Analytically derive $\mathbf{P}$, where $\hat{\mathbf{y}}$ is the least-squares prediction vector. To achieve this:
1. Derive the optimal weight vector $\hat{\mathbf{w}}$ that minimizes the least-squares objective function
$$\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \tag{3}$$
Note: we have $\frac{\partial}{\partial \mathbf{w}} \mathbf{w}^\top \mathbf{S} \mathbf{w} = 2\mathbf{S}\mathbf{w}$ *for symmetric matrix* $\mathbf{S}$, *and* $\frac{\partial}{\partial \mathbf{w}} \mathbf{w}^\top \mathbf{a} = \mathbf{a}$.
2. Using your result for $\hat{\mathbf{w}}$, give the definition for $\mathbf{P}$.

Answer:

5 pts.

f) Prove that $\mathbf{P}$ is an orthogonal projector, by showing that $\mathbf{P} = \mathbf{P}\mathbf{P}$ and $\mathbf{P} = \mathbf{P}^\top$.
Answer:

2 pts.

g) Computing $\hat{\mathbf{w}}$ involves the inversion of the symmetric matrix $\mathbf{S} := \mathbf{X}^\top \mathbf{X}$, which we assume to be *positive definite*. Give a mathematical condition (in terms of the eigenvalues of $\mathbf{S}$), when this inversion is numerically unstable.
Answer:

2 pts.

h) Ridge regression finds the regularized weight vector $\hat{\mathbf{w}}_{\text{ridge}}$ that minimizes eq. (3) with an additional norm penalty,

$$\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 .$$

Explain why choosing $\lambda > 0$ improves stability of the inversion:

1. Derive the regularized optimal weight vector $\hat{\mathbf{w}}_{\text{ridge}}$.
   Answer:

2 pts.

2. What is the effect of increasing λ on
   (a) the norm $\|\hat{\mathbf{w}}_{\text{ridge}}\|_2$ and
   (b) the numerical stability of computing $\hat{\mathbf{w}}_{\text{ridge}}$?
   Provide **arguments** for your answers.
   Answer:

4 pts.

Question 5: Bagging and AdaBoost (30 pts.)

Bagging and AdaBoost are two ensemble methods used frequently in classification.

a) State two essential differences between bagging and AdaBoost.
Answer:

2 pts.

b) Explain how bagging reduces the variance of an estimator?
Answer:

3 pts.

c) State two factors that may influence the performance of AdaBoost?
Answer:

2 pts.

d) How can we use AdaBoost to detect outliers (mislabeled or ambiguously labeled examples)?
Answer:

2 pts.

For a binary classification problem (± 1), assume we have $c_1, ..., c_B$ successive base classifiers obtained via AdaBoost, and let $\alpha_1, ..., \alpha_B$ be the corresponding weights. The ensemble classifier is $\hat{c}_B(x) = \text{sgn}(\sum_{b=1}^{B} \alpha_b c_b(x))$.

Reusing the same base classifiers $c_1, ..., c_B$, we now train a *linear* classifier, by finding *new* $\tilde{\alpha}_b$'s that minimize the average exponential loss on the training examples, and obtain $\tilde{\hat{c}}_B(x) = \text{sgn}(\sum_{b=1}^{B} \tilde{\alpha}_b c_b(x))$.

e) Is $\tilde{\hat{c}}_B$ equivalent to $\hat{c}_B$? Justify your answer.
Answer:

4 pts.

Consider the following simplified ensemble learning algorithm:

We consider a dataset $\{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$ of N i.i.d. samples, where $y_n \in \{-1, +1\}$. After each iteration b of this algorithm, each training sample x_n has an associated weight $w_{b,n}$. At the beginning, $w_{0,n} = 1$ for $n = 1, ..., N$. Let $0 < \gamma < \frac{1}{2}$ and $\beta = \frac{1+2\gamma}{1-2\gamma}$.

For each iteration $b = 1$ to B:

- The algorithm finds a new weak learner c_b on the weighted training set that is guaranteed to have an error of at most $\frac{1}{2} - \gamma$ on the weighted training set.
- We then update the weights of the training samples:
  - If x_n is misclassified by c_b, i.e., if $c_b(x_n) \neq y_n$: Set $w_{b,n} := \beta w_{b-1,n}$.
  - Otherwise the weight remains unchanged: $w_{b,n} := w_{b-1,n}$.

The ensemble classifier decides by combining the weak learners as follows:

$$\hat{c}_B(x) = \text{sgn}(\sum_{b=1}^{B} c_b(x)).$$

Let $W_b = \sum_{n=1}^{N} w_{b,n}$ be the sum of the weights at the end of iteration b.

f) What is the difference between AdaBoost and this algorithm in the way they combine the weak learners?
Answer:

2 pts.

g) Derive a bound on W_B, the sum of the weights after the final iteration. Given that at each iteration b, the sum of the weights increases by at most a factor of $1 + 2\gamma$:

$$\frac{W_b}{W_{b-1}} \leq 1 + 2\gamma,$$

show that for the final sum of weights it holds that

$$W_B \leq N(1 + 2\gamma)^B.$$

Answer:

4 pts.

h) Prove the following lower bound on the final weight of a training sample, if it is misclassified by the ensemble:

If $\hat{c}_B(x_n) \neq y_n$, then $w_{B,n} \geq \beta^{B/2}$.

Answer:

4 pts.

i) Finally, we will show a bound on the empirical error rate on the training data. Let $M = |\{n : \hat{c}_B(x_n) \neq y_n\}|$ be the number of training samples misclassified by the ensemble classifier.

1. Plugging the two previous bounds together, we have

$$M\beta^{B/2} \leq \sum_{n=1}^{N} w_{n,B} = W_B \leq N(1+2\gamma)^B.$$

Starting from this inequality, show that the empirical error rate on the training data is bounded by the following:

$$\frac{M}{N} \leq e^{-2B\gamma^2}.$$

(*Hint: note that* $(1+a)^c \leq e^{ac}$ *for* $c \geq 0$, *and recall that* $\beta = \frac{1+2\gamma}{1-2\gamma}$)

Answer:

2. Explain what happens to this bound when we
   (a) increase B (the number of training iterations),
   (b) increase γ (have a very good learner).

   Answer:

7 pts.

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Introduction to Machine Learning ~~2010~~ 2009

Prof. J. M. Buhmann

Final Exam

February 10th, 2010

First and Last name: ______________________

ETH number: ______________________

Signature: ______________________

General Remarks

- Please check that you have all 21 pages of this exam.
- Remove all material from your desk which is not permitted by the examination regulations.
- Fill in your first and last name and your ETH number and sign the exam. Place your student ID in front of you.
- You have 2 hours for the exam. There are five questions, where you can earn a total of 150 points. Scoring 120 points (equivalent to solving four questions) guarantees you a grade of six.
- Write your answers directly on the exam sheets. If you need more space, put your name and ETH number on top of each supplementary sheet.
- Answer the questions in English. Do not use a pencil or red color pen.
- You may provide at most one valid answer per question. Invalid solutions must be canceled out clearly.

	Topic	Max. Points	Points Achieved	Visum
1	Density Estimation	30		
2	PCA	30		
3	Classifiers	30		
4	Regression	30		
5	Boosting	30		
Total		150		

Grade: ....

Question 1: Parametric Density Estimation (30 pts.)

When estimating densities from data, we assume that the data distribution is well approximated by a parametric model $P(x;\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ denotes the parameters of the model. Estimation of the density then reduces to estimation of the parameter $\boldsymbol{\theta}$.

a) In maximum-likelihood estimation (MLE), we usually start by assuming that the data is given as an i.i.d. sample $x_1, x_2, ..., x_N \sim P(x;\theta)$. Please explain how this assumption simplifies the calculations.

Answer: ① iid allows us to estimate a probability for each event independent from other giving a product

$$P(X|\theta) = \prod_{i=1}^{N} P(x_i|\theta) = P(x_1|\theta) \cdot P(x_2|\theta) \cdot P(x_3|\theta) \cdot \ldots \cdot P(x_n|\theta)$$

∴ This assumption would simplify to determine a mean and variance for the underlying distribution

2 pts.

b) We model a coin flip as a random variable X, where $X = 1$ denotes the outcome 'heads', and $X = 0$ denotes 'tails'. We model this event using the Bernoulli distribution, parametrized by μ that denotes $P(X = 1)$.

$$\text{Bern}(x|\mu) = \mu^x(1-\mu)^{1-x}.$$

Given a data set $\mathcal{X}_N = \{x_1, ..., x_N\}$ of N i.i.d. observations of X, show us step by step how to compute the MLE for μ. Comment your calculations where necessary.

Answer: Likelihood: $P(X|\theta) = \prod_{i=1}^{N} \text{Bern}(x|\mu) = \prod_{i=1}^{N} \mu^{x_i}(1-\mu)^{1-x_i} \quad \left(= \mu^{\sum x_i}(1-\mu)^{\sum(1-x_i)}\right)$

Take log of likelihood fxn: $\log P(X|\theta) = \log\left(\prod_{i=1}^{N} \text{Bern}(x_i|\mu)\right)$

$$= \sum_{i=1}^{N} \log\left(\mu^{x_i}(1-\mu)^{1-x_i}\right)$$

$$P(\mathcal{X}|\mu) = \sum_{i=1}^{N}\left[\log\left[\mu^{x_i}\right] + \log\left[(1-\mu)^{1-x_i}\right]\right]$$

Take derivative wrt to μ. impose its equal.

$$\nabla_\mu \log P(\mathcal{X}|\mu) = \nabla_\mu\left[\sum_{i=1}^{N} x_i \log\mu + 1 - x_i \log(1-\mu)\right] \stackrel{!}{=} 0$$

$$= \sum_{i=1}^{N} x_i \cdot \frac{1}{\mu} + \frac{1-x_i}{1-\mu} \cdot -1 \Rightarrow \sum_{i=1}^{N} \frac{x_i}{\mu} - \sum_{i=1}^{N} \frac{1-x_i}{1-\mu} = 0$$

8 pts.

$$\sum_{i=1}^{N} \frac{x_i}{\mu} = \sum_{i=1}^{N} \frac{1-x_i}{1-\mu} \Rightarrow \frac{1}{\mu}\sum_{i=1}^{N} x_i = \frac{1}{1-\mu}\sum_{i=1}^{N}(1-x_i)$$

$$\Rightarrow \frac{1}{\mu}\sum_{i=1}^{N} x_i = \frac{1}{1-\mu}\left(\sum_{i=1}^{N} 1 - \sum_{i=1}^{N} x_i\right) \Rightarrow \frac{1}{\mu}\sum_{i=1}^{N} x_i = \frac{1}{1-\mu}\left(N - \sum_{i=1}^{N} x_i\right)$$

$$\Rightarrow \frac{1-\mu}{\mu} = \frac{N - \sum_{i=1}^{N} x_i}{\sum_{i=1}^{N} x_i} \qquad \frac{1}{\mu} - \frac{\mu}{\mu} = \frac{N}{\sum_{i=1}^{N} x_i} - \frac{\sum x_i}{\sum x_i}$$

$$\Rightarrow \frac{1}{\mu} - 1 = \frac{N}{\sum_{i=1}^{N} x_i} - 1 \Rightarrow \frac{1}{\mu} = \frac{N}{\sum_{i=1}^{N} x_i} - 1 + 1$$

$$\Rightarrow \boxed{\hat{\mu} = \frac{\sum_{i=1}^{N} x_i}{N}} \checkmark$$

c) We flip our coin three times and obtain the dataset $\mathcal{X}_3 = \{1, 1, 1\}$.

1. What is the MLE for $\hat{\mu}$ for dataset $\mathcal{X}_3$? $N=3$

   Answer: $\hat{\mu} = \frac{\sum_{i}^{3} x_i}{3} = \frac{1+1+1}{3} = 1$ ✓ $\hat{\mu} = 1$

2. State one problem with this estimate. What is a possible remedy (solution)?

   Answer: Problem: This estimate is not biased ($\hat{\mu} = 1$ versus $\hat{\mu} = \frac{1}{2}$. The MLE gave you a bias of $\hat{\mu} = 1$ unfair coin toss. High variance associated w/ a small population of values.

   Remedy: you more samples to get a better, less unfair estimate unless you take many more trials solution $\hat{\mu}$

3 pts.

d) Assume that there are a total of m heads in $\mathcal{X}_N$, i.e., $\sum_{i=1}^{N} x_i = m$.

1. In terms of μ, N and m only, what is the probability of obtaining this dataset $P(\mathcal{X}_N|\mu)$? Explain the terms of your solution. coming from the same dataset $\mathcal{X}_N$

   Answer: $P(\mathcal{X}_N|\mu) = \prod_{i=1}^{N} \mu^{x_i}(1-\mu)^{1-x_i} = \quad \sum_{i}^{n} \log\left(\mu^{x_i}(1-\mu)^{1-x_i}\right)$

   $\log P(\mathcal{X}_N|\mu) = \sum_{i=1}^{N} \mu^{x_i}(1-\mu)^{\sum(1-x_i)} = \underbrace{\mu^m}_{\text{probability of getting exactly m heads}}\underbrace{(1-\mu)^{N-m}}_{\text{probability of getting exactly N-m tails}}$

   Distributive $\sum x_i = m$ $\sum x_i = N$

2. What is the expected value of m? (*Hint: For independent random variables, the expectation of a sum is the sum of expectations*)

   Answer: $\mathbb{E}[m] = E\left[\sum_{i=1}^{N} x_i\right] = \sum_{i=1}^{N} E[x_i] = \sum_{i=1}^{N} \mu = N \cdot \hat{\mu}$

   $\hat{\mu} = \frac{\sum_{i=1}^{n} x_i}{N} = \frac{m}{N}$ $\quad m = \hat{\mu} \cdot n$ ✓

   $\sum_{i}^{n} x_i = m$ exponential property that. probability

5 pts.

$$\sum_{i}^{n} \cancel{\log}\left(\sum_{i}^{n} \mu^{x_i}(1-\mu)^{1-x_i}\right) = \cancel{\log}\left(\mu^{\sum x_i}(1-\mu)^{\sum(1-x_i)}\right)$$

$$= \cancel{\log}\left(\mu^{m}(1-\mu)^{n-m}\right)$$

We now turn to Bayesian estimation of μ given a data set $\mathcal{X}$, by introducing a prior distribution $P(\mu)$. Here, we use the (conjugate) prior given by the Beta distribution:

$$\text{Beta}(\mu|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\mu^{\alpha-1}(1-\mu)^{\beta-1},$$

where $\Gamma(x)$ denotes the gamma function (not necessary). Some useful properties:

1. Mean $\mathbb{E}(\mu) = \frac{\alpha}{\alpha+\beta}$
2. Variance $\mathbb{V}(\mu) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$

e) With the mean and variance of μ, we can understand how the parameters influence the prior. Select the best (α_0, β_0) pair for the prior that emphasizes the following beliefs.

1. My coin is biased towards 'tails'. more probability of tails. low variance, high bias
   ☐(0.5, 0.5) ☒(1, 4) ☐(4, 1) ☐(2, 2)
2. My coin is most probably biased (but I don't know heads or tails)
   ☒(0.5, 0.5) ☐(1, 4) ☐(4, 1) ☐(2, 2) high variance low bias (not much info about the system)

3 pts.

f) Show that if the prior is Beta distributed

$$P(\mu) = \text{Beta}(\alpha_0, \beta_0),$$

then the posterior takes again the same functional dependency on μ as the prior, i.e.,

$$P(\mu|\mathcal{X}_N) \propto \mu^{\alpha_0+m-1}(1-\mu)^{\beta_0+N-m-1}. \quad P(\mu|\mathcal{X}_N) = P(\mathcal{X}_N|\mu)\,P(\mu)$$

Beta → prior, likelihood → Bern

Answer: $\text{Beta}(\mu|\alpha_0\beta_0) = \frac{\Gamma(\alpha_0+\beta_0)}{\Gamma(\alpha_0)\Gamma(\beta_0)}\mu^{\alpha_0-1}(1-\mu)^{\beta_0-1}$

$\text{Bern}(\mathcal{X}_N|\mu) = \mu^m(1-\mu)^{N-m}$

$$P(\mu|\mathcal{X}_n) = \left(\mu^m(1-\mu)^{N-m}\right)\left(\mu^{\alpha_0-1}(1-\mu)^{\beta_0-1}\right)$$

$$\boxed{P(\mu|\mathcal{X}_n) = \mu^{m+\alpha_0-1}(1-\mu)^{\beta_0+N-m-1}} \checkmark$$

0.5, 0.5   2, 2

mean. $\frac{0.5}{0.5+0.5} = \frac{1}{1} = \frac{1}{2}$ mean $\frac{2}{2+2} = \frac{2}{4} = \frac{1}{2}$

5 pts.

var. $\frac{0.25}{(1)^2(2)} = \frac{0.25}{2} = \frac{\frac{1}{4}}{2} = \frac{1}{8}$ $\frac{4}{(4)^2(5)} = \frac{4}{16.5} = \frac{4}{16.5} = \text{small}$

g) In fact, the posterior is again a Beta distribution, and we obtain $P(X = 1|\mathcal{X}) = \frac{m+\alpha_0}{N+\alpha_0+\beta_0} \stackrel{!}{=} \frac{1}{2}$ prior $= \frac{1}{2}$

1. For $\mathcal{X}_3 = \{1, 1, 1\}$, use a choice of (α_0, β_0) which $\boxed{\text{assumes}}$ a $\boxed{\text{fair coin}}$, and obtain your estimate for $P(X = 1|\mathcal{X})$. How does this estimate differ from the MLE?
Answer:

Fair coin: $\hat{\mu} = \frac{1}{2}$ $\quad \frac{1}{2} = \frac{\alpha}{\alpha+\beta}$ $\quad \alpha+\beta = 2\alpha$ $\quad \beta = 2\alpha - \alpha \Rightarrow \beta = \alpha$ $\quad 1 = 1$

$P(X=1|\mathcal{X}) = \frac{3+\alpha_0}{3+\alpha_0+\beta_0}$

$\alpha_0 = \beta_0 = 1$ $\quad \sum_{i=1}^{N} x_i = m = 3$

$P(X=1|\mathcal{X}) = \frac{m+1}{N+1+1} = \frac{m+1}{N+2} = \frac{3+1}{3+2} = \frac{4}{5} = 0.8$

$1 - 0.8 - 0.5$

MLE posterior prior

MLE $\rightarrow \hat{\mu} = 1$

with the prior information of a fair coin, this tells us our estimate indicates the our MLE estimate is a pretty much biased coin. You can ~~always~~ be sure that the MLE is ~~not~~ unbiased but to know if its truely biased you need a bayesian approach

2. What happens to the posterior as $N \rightarrow \infty$? How does this relate to the MLE?
Answer:

$$\lim_{N\to\infty} \left[P(\mu|\mathcal{X}_n) = \mu^{m + \alpha_0 - 1} (1-\mu)^{\beta_0 + N - m - 1} \right]$$

$\sum x_i$ $\quad N - \sum x_i$

gets larger, constants become insignificant

$\uparrow N$

$\rightsquigarrow \mu^m (1-\mu)^{N-m} = P(\mathcal{X}_n|\mu) \Rightarrow$ MLE

$\hat{\mu} = \frac{m}{N}$

$\uparrow N \mid m \uparrow$

4 pts.

As posterior $N \rightarrow \infty$, then becomes MLE equation

Question 2: Principal Component Analysis (PCA) (30 pts.)

Principal component analysis is a widely applied method in data analysis and dimensionality reduction. Given a dataset $\mathbf{X} \in \mathbb{R}^{D\times N}$ (observations as columns), where D is the number of dimensions and N is the number of observations, a linear transformation using an orthonormal matrix $\mathbf{P} \in \mathbb{R}^{M\times D}$ is applied to make a *change of basis*, to obtain a (usually) lower-dimensional dataset $\mathbf{Y} \in \mathbb{R}^{M\times N}$.

a) We begin by reviewing the steps of applying PCA to a dataset $\mathbf{X}$. Please complete each step below by providing the appropriate formula to compute the desired quantity.

1. Define the zero-mean dataset $\tilde{\mathbf{X}}$ in terms of the original dataset $\mathbf{X}$. For this purpose, introduce the data mean as a column-vector $\boldsymbol{\mu} \in \mathbb{R}^{D\times 1}$.
Answer:

2. Define the covariance matrix $\mathbf{C}_{\tilde{\mathbf{X}}}$ in terms of the zero-mean dataset $\tilde{\mathbf{X}}$.
Answer:

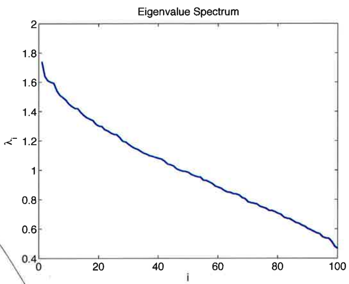
3. Write down the eigen-decomposition of the covariance matrix $\mathbf{C}_{\tilde{\mathbf{X}}}$, in terms of the eigenvector matrix $\mathbf{E}$ (eigenvectors as columns) and the diagonal matrix of eigenvalues $\mathbf{D}$.
Answer:

4. Define the PCA transformation matrix $\mathbf{P}$ in terms of the eigenvector matrix $\mathbf{E}$. (Note: assume we don't want to reduce the dimensionality of the transformed dataset. Further assume that the eigenvectors in $\mathbf{E}$ have already been sorted according to the corresponding eigenvalues, in decreasing order.)
Answer:

5. Define the transformed dataset $\mathbf{Y}$ in terms of the original dataset $\mathbf{X}$ and the transformation matrix $\mathbf{P}$.
Answer:

6 pts.

b) Assume we have applied PCA to some dataset ($D = 100$). We observe the following eigenvalue spectrum of the covariance matrix of the data. (λ_i: eigenvalues)



1. Is the intrinsic dimensionality of this dataset low or high? Why?
Answer:

2. Can this dataset be expressed in few dimensions with low approximation error? Why?
Answer:

3. If yes, which dimensionality (approximately) should be chosen for the transformed dataset and why?
Answer:

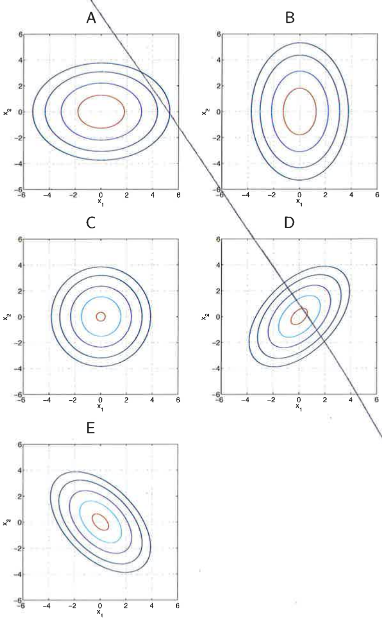
5 pts.

c) Assume you have observed 2D data $\mathbf{X} \in \mathbb{R}^{2\times N}$ (observations as columns). The first row of $\mathbf{X}$ corresponds to the first dimension x_1, the second row corresponds to x_2. For each of the three covariance matrices $\mathbf{C}_\mathbf{X}$ below, please choose the iso-line plot (A-E) corresponding to the covariance matrix. (Note the axis labels on the figures)

1. $\begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$ Answer: ( )

2. $\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}$ Answer: ( )

3. $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ Answer: ( )



3 pts.

d) PCA can be derived as follows: Given a dataset $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\} \in \mathbb{R}^{D \times N}$, find a new set of basis vectors such that the variance of the projected dataset is maximized. We begin by finding the first projection direction $\mathbf{e}_1 \in \mathbb{R}^D$, $||\mathbf{e}_1||_2 = 1$, which satisfies our goal.

1. Please describe in words why we are interested to find a projection direction such that the projected data has large variance.
Answer:

2. Write down formally the objective function for maximizing the variance of the projected dataset. For this purpose, define the variance of the projected dataset in terms of the original data $\mathbf{x}_n$ and the first projection direction $\mathbf{e}_1$. (Hint: introduce the mean $\boldsymbol{\mu}$ of the data $\mathbf{X}$.)
Answer:

5 pts.

e) PCA transforms a dataset $\mathbf{X}$ into a dataset $\mathbf{Y}$ by defining a new basis using the eigenvectors of the covariance matrix $\mathbf{C_X}$ of the dataset $\mathbf{X}$. With this particular choice of a new basis, the covariance matrix $\mathbf{C_Y}$ of the transformed dataset $\mathbf{Y}$ is *diagonalized*. (Note: For this exercise, assume a zero-mean dataset.)

1. Please explain in words, why we desire the covariance matrix of the transformed dataset to be diagonal.
Answer:

2. Show that $\mathbf{C_Y} = \mathbf{P C_X P}^T$, i.e., that the covariance matrix $\mathbf{C_Y}$ of the transformed dataset can be written in terms of the covariance matrix $\mathbf{C_X}$ of the original dataset.
Answer:

3. Show that the choice $\mathbf{P} = \mathbf{E}^T$ for the PCA transformation matrix, where $\mathbf{E}$ is the matrix of eigenvectors of the covariance matrix $\mathbf{C_X}$, actually diagonalizes the covariance matrix $\mathbf{C_Y}$ of the transformed dataset $\mathbf{Y}$. Use the eigen decomposition of $\mathbf{C_X}$ and the fact that $\mathbf{P}^{-1} = \mathbf{P}^T$.
Answer:

11 pts.

Question 3: Linear Classifiers (30 pts.)

The simplest representation of a linear discriminant function is obtained by taking a linear function of the input vector $\boldsymbol{x}$ so that

$$y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$$

a) Define the term decision boundary. Explain what is the relation between the vector $\boldsymbol{w}$ and the decision boundary.
Answer:
Decision boundary: A hyperplane separateng different classes (linearly separaple). points). A decision boundary is always $\perp$ to the $\vec{w}$.

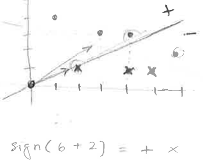
2 pts.

b) Consider the following set $S \subseteq \mathbb{R}^2 \times \{+1, -1\}$:

$$S = \{((3,3),+1),((1,4),+1),((2,1),-1),((5,1),-1),((6,2),+1),((4,3),+1),((4,1),-1)\}$$

Prove that there exist no homogeneous half-space that separates S with no errors (recall that a half-space is homogeneous if its separating hyperplane passes through the origin). Assume that the classification of a point is given by $y(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^T\boldsymbol{x} + b)$.
Answer:

3 points
x_1 ((2,1), -1)
x_2 ((4,3), +1)
x_3 ((6,2), +1)



$y(x_1) = (3,2)\begin{pmatrix}2\\1\end{pmatrix} + 0 = \text{sign}(6+2) = +$ ×

$(3,2)\begin{pmatrix}4\\3\end{pmatrix} + 0 = \text{sign}(12+6) = +$ ✓

5 pts.

The simple form of the online perceptron algorithm is defined as follows:

Initialize by setting $\boldsymbol{w}^{(0)} = \mathbf{0}$.

$$\text{Update } \boldsymbol{w}^{(t+1)} = \begin{cases} \boldsymbol{w}^{(t)} & y_k(\underbrace{\boldsymbol{w}^{(t)T}\boldsymbol{x}_k}_{w^T x}) > 0 \\ \boldsymbol{w}^{(t)} + y_k\boldsymbol{x}_k & \text{otherwise.} \end{cases}$$

c) We would like to use the perceptron algorithm to find a separating half-space that makes no errors on the set S. Recall that you have just proven that this is impossible using a homogeneous half-space. Make the necessary adjustments and perform 7 iterations of the perceptron algorithm. For each iteration write down the training example and the current vector $\boldsymbol{w}^{(t)}$. Assume that the points are given in the same order as they appear in S.
Answer:
necessary adjustment is homogeneous coordinates

	x_1	x_2	x_3 bias	w_1	w_2	w_3	y_k	$\hat{y}_k$	
0	3	3	1	0	0	0	1	0	×
1	1	4	1	3	3	1	1	16	✓ K
2	2	1	1	3	3	1	-1	10	×
3	5	1	1	1	2	0	-1	7	×
4	6	2	1	-1	1	-1	1	-23	×
5	4	3	1	2	3	0	1	17	✓
6	4	1	1	2	3	0	-1	11	×
				-2	2	-1			

$\max \frac{1}{\|w\|} \rightarrow \min \|w\|$

5 pts.

Consider the primal form of the soft margin SVM

$$\min \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.} \quad y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 - \xi_i, \qquad i = 1,\ldots,n$$

Distance of two lines
maximize margin
regularization
regularization term
"Allow error" in classification
mistakes over training examples
"slack" variables
loss fxn

d) Indicate which of the following statements hold as we increase the value of the parameter C (relative to any starting value, $C > 0$). Use

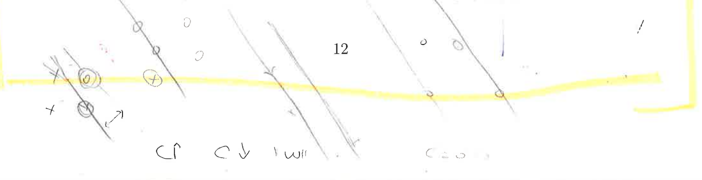
D - if the validity of the statement depends on the situation,
T - if the statement is necessary true
N - if the statement is never true

C=∞ enforces all constraints, hard margin
large C makes constraints hard to ignore → narrows margin
↑ C=1000 → hard-margin SVM
C=10 → wider margin
C=0.1 → widest margin
small C, allows constraints to be easily ignored, large margin

EMAIL TA

D [T] b will not increase Not in the obj fxn only in the constraint which is fixed. D
D [N] $\|\boldsymbol{w}\|$ increases minimizing $\frac{1}{2}\|w\|^2$.
D [N] $\|\boldsymbol{w}\|$ will not decrease
N [N] more points will be misclassified
T? [N] The geometric margin will not increase

T



10 pts.

The dual soft margin SVM is given below

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$
$$\text{s.t.} \quad 0 \le \alpha_i \le C, \qquad i = 1, \ldots, n$$

We would like to solve the SVM problem using the kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 + (\boldsymbol{x}_i^T \boldsymbol{x}_j)^2$.

e) What is the feature representation $\phi(\boldsymbol{x})$ corresponding to this kernel for $\boldsymbol{x} \in \mathbb{R}^2$?
Answer: $K(x, z) = 1 + (x_1 z_1 + x_2 z_2)^2$ $\quad x = (x_1, x_2)$, $z = (z_1, z_2)$, $(a+b)^2 = (a+b)(a+b)$

$= 1 + (x_1 z_1)^2 + 2 x_1 z_1 x_2 z_2 + (x_2 z_2)^2$

$\phi(x)^T \phi(z) = (1, x_1^2, \sqrt{2} x_1 x_2, x_2^2)^T \cdot (1, z_1^2, \sqrt{2} z_1 z_2, z_2^2)$

$\boxed{\phi(x) = (1, x_1^2, \sqrt{2} x_1 x_2, x_2^2)}$ ✓

2 pts.

f) Using the dual variables α's and $\phi(\boldsymbol{x})$, give an explicit form for the primal variable $\boldsymbol{w}$.
Answer:

$w^* = \sum_{i=1}^{n} \alpha_i^* y_i \phi(x_i)$

2 pts.

We solved the dual SVM problem with the above kernel. We used the following training set
$\boldsymbol{x}_1 = (1, 1), y_1 = +1$, $\boldsymbol{x}_2 = (3, 0), y_2 = -1$, $\boldsymbol{x}_3 = (-2, 1), y_3 = -1$.
Below are the α's and b values
$\alpha_1 = 0.1,\ \alpha_2 = 0,\ \alpha_3 = 0.1,\ b = 1.5$

g) Use these values to compute the kernelized discriminant function $y(\boldsymbol{x})$.
Answer: $x \cdot w + b = \sum_{i=1}^{3} \alpha_i y_i (x \cdot x_i) + b$ $\quad K(x, z) = 1 + (x^T z)^2$

$= (\alpha_1 y_1 (x \cdot x_1) + b) + (\alpha_2 y_2 (x \cdot x_2) + b) + (\alpha_3 y_3 (x \cdot x_3) + b)$

$y(x) = \phi(x) w^* + b.$

$\phi(x) = [x_1 x_1,\ x_1 x_2,\ x_1 x_2,\ x_2 x_1,\ x_2 x_2,\ x_2 x_3,\ x_3 x_1,\ x_3 x_2,\ x_3 x_3]$

4 pts.

Question 4: Regression (30 pts.)

In *univariate regression analysis*, we study the *statistical* dependency of the output variable $y \in \mathbb{R}$ on the input variable $x \in \mathbb{R}$.

a) Complete the equation

linear combination of non-linear fcns of input variables $\phi'(x)$ → basis fcns

$$y = \sum_{i=1}^{N} w^T \phi(x) + w_0 \quad \varepsilon \tag{1}$$

nonlinear regression

where the right hand side consists of two terms. Define each term in the equation.
Note: Be general, don't restrict yourself to linear least-squares regression.
Answer: $y = w^T x + w_0 + \varepsilon$

$\phi'(x)$ are basis fcns $\quad N \sim (0, \sigma^2)$

w^T = weights/coefficients

w_0 = offset

$E[w^T x + w_0] + E[\varepsilon] = 0$.

3 pts.

b) Regression analysis is only meaningful if there actually is a true dependency between x and y. Prove that if x and y are *statistically independent*, there is no linear dependency:

1. Assume that x and y are statistically independent.
2. Begin with the definition of *covariance*,

$$\text{cov}(x, y) = \mathbb{E}\left[(x - \mu_x)(y - \mu_y)\right], \qquad f: x \to y$$

where $\mathbb{E}$ is the expectation operator and μ_x is the mean of x.

3. Show that x and y are uncorrelated. $\quad E[x] = \mu_x$ mean of x.

Note: Comment your calculations where necessary.
Answer:

$E[(x - \mu_x)(y - \mu_y)] = E[xy - x\mu_y - y\mu_x + \mu_x \mu_y]$

$= E[xy] - E[x\mu_y] - E[y\mu_x] + E[\mu_x \mu_y]$

independent

$= E[x]E[y] - E[x]\mu_y - E[y]\mu_x + \mu_x \mu_y$

$= E[x]E[y] - \mu_x \mu_y - \mu_y \mu_x + \mu_x \mu_y$

$= \mu_x \mu_y - \mu_x \mu_y$

$= 0$

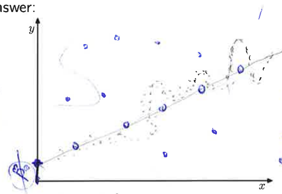
covariance is 0, which means uncorrelated.

5 pts.

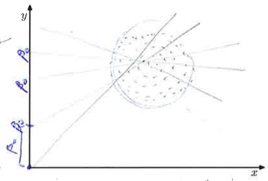
c) You have shown that eq. (1) consists of two terms. Illustrate two conceptually different cases where a linear least-squares fit is inappropriate for the true dependency between x and y, one related to the first term in eq. (1), and one related to the second term in eq. (1):

1. Draw a scatter plot of a data sample.
2. Fit the linear least-squares solution into the sample (qualitatively).
3. Explain why the solution does not capture the true dependency between x and y.

Answer:



The linear fit does not capture the complex relationship between x and y. This classifier is considered to "underfit"



with such a distribution there a multiple ways of determining P_0

✓

4 pts.

So far, we considered a single input variable x. In multivariate linear regression, we assume a weighted linear functional dependency

$$y = w_0 + w_1 x_1 + \cdots + w_D x_D \tag{2}$$

between D input variables $x_1, \ldots, x_D \in \mathbb{R}$ and output variable y.

d) Define the input data matrix $\mathbf{X}$, weight vector $\mathbf{w}$ and output data vector $\mathbf{y}$ such that eq. (2) for all N data points of the sample $(x_1^n, x_2^n, \ldots, x_D^n, y^n), n = 1, \ldots, N$ can be written as
> homogeneous coordinates

$$\mathbf{y} = \mathbf{X}\mathbf{w}.$$

Answer:

$D+1$

$\mathbf{X} := \begin{bmatrix} 1, x_1^1, x_2^1, x_3^1, \ldots, x_D^1 \\ 1, x_1^2, x_2^2, x_3^2, \ldots, x_D^2 \\ \vdots \\ 1, x_1^N, x_2^N, x_3^N, \ldots, x_D^N \end{bmatrix}$ N $\quad N \times (D+1)$

$\mathbf{w} := \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_D \end{bmatrix}$ $\quad D \times 1$

$\mathbf{y} := \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_N \end{bmatrix}$ $\quad N \times 1$ ✓

3 pts.

The linear least-squares prediction $\hat{\mathbf{y}}$ is the *orthogonal projection* of $\mathbf{y}$ onto the space spanned by the columns of $\mathbf{X}$, and can be written as

$$\hat{\mathbf{y}} = \mathbf{P}\mathbf{y}.$$

$\hat{y} = w^T x = \hat{y} = w^T x \Rightarrow w = (x^T x)^{-1} x y$

e) Analytically derive $\mathbf{P}$, where $\hat{\mathbf{y}}$ is the least-squares prediction vector. To achieve this:

1. Derive the optimal weight vector $\hat{\mathbf{w}}$ that minimizes the least-squares objective function

$$\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 \tag{3}$$

Note: we have $\frac{\partial}{\partial \mathbf{w}} \mathbf{w}^\top \mathbf{S} \mathbf{w} = 2\mathbf{S}\mathbf{w}$ *for symmetric matrix* $\mathbf{S}$*, and* $\frac{\partial}{\partial \mathbf{w}} \mathbf{w}^\top \mathbf{a} = \mathbf{a}$.

2. Using your result for $\hat{\mathbf{w}}$, give the definition for $\mathbf{P}$.

$x = n \times d$
$w = d \times 1$
$y = n \times 1$

Answer: $\arg\min_{w} \|xw - y\|_2^2 = (xw - y)^T (xw - y)$

$= w^T x^T x w - w^T x^T y - y^T x w - y^T y \quad \frac{d}{dw} \quad x^T x = S$

$0 = 2 x^T x w - x^T y - y^T x$

$0 = 2 x^T x w - 2 x^T y$

$x^T x w = x^T y$

$w = (x^T x)^{-1} x^T y \quad \boxed{P = x(x^T x)^{-1} x^T} \ y$

$y = x^T P$

5 pts.

f) Prove that $\mathbf{P}$ is an orthogonal projector, by showing that $\mathbf{P} = \mathbf{P}\mathbf{P}$ and $\mathbf{P} = \mathbf{P}^\top$.
Answer:

$(x(x^T x)^{-1} x^T)(x(x^T x)^{-1} x^T) \quad (A^{-1})^T = (A^T)^{-1}$

$P^T = (x(x^T x)^{-1} x^T)^T = x^T($

2 pts.

g) Computing $\hat{\mathbf{w}}$ involves the inversion of the symmetric matrix $\mathbf{S} := \mathbf{X}^\top\mathbf{X}$, which we assume to be *positive definite*. Give a mathematical condition (in terms of the eigenvalues of $\mathbf{S}$), when this inversion is numerically unstable.

Answer: Defn of a positive definite matrix → eigendecomposition All diagonal elements, eigenvalues $\lambda_i > 0$ positive and real. if not then inversion is unstable.

All its eigenvalues are positive

2 pts.

h) Ridge regression finds the regularized weight vector $\hat{\mathbf{w}}_{\text{ridge}}$ that minimizes eq. (3) with an additional norm penalty,

$$\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 .$$

Explain why choosing $\lambda > 0$ improves stability of the inversion: choosing a $\lambda > 0$ will shrink w's to not grow so big.

1. Derive the regularized optimal weight vector $\hat{\mathbf{w}}_{\text{ridge}}$.
   Answer:

$$\arg\min_{w} \|xw - y\|_2^2 + \lambda\|w\|_2^2 \Rightarrow \left(w^T x^T x w - w^T x^T y - y^T x w - y^T y + \lambda w^T w\right)\frac{d}{dw}$$

$$0 = 2x^T x w - 2x^T y + \lambda w$$

$$2x^T y = 2x^T x w + \lambda w$$

$$2x^T y = w(2x^T x + \lambda)$$

$$w_{Ridge} = \frac{2x^T y}{2x^T x + \lambda} = \frac{x^T y}{x^T x + \frac{1}{2}\lambda}$$

$$\boxed{w_{Ridge} = \left(x^T x + \frac{\lambda}{2}\right)^{-1} x^T y}$$

2 pts.

2. What is the effect of increasing λ on
   (a) the norm $\|\hat{\mathbf{w}}_{\text{ridge}}\|_2$ and
   (b) the numerical stability of computing $\hat{\mathbf{w}}_{\text{ridge}}$?

   Provide **arguments** for your answers.
   Answer:

(a) increasing λ will shrink w coefficients to not grow so big and prevent overfitting

(b)

4 pts.

Question 5: Bagging and AdaBoost (30 pts.)

Bagging and AdaBoost are two ensemble methods used frequently in classification.

a) State two essential differences between bagging and AdaBoost.
   Answer:

	Bagging	AdaBoost
choice of data ①	varies the training sets using resampling to train weak learners	Trains weak learners with the same training set on every iteration
choice of weights ②	gives the same importance to every prediction	weights the prediction of every weak classifier according to its accuracy

2 pts.

b) Explain how bagging reduces the variance of an estimator?
   Answer: bagging uses the method of boot strap to create different training sets from the original distribution. in order to reduce the variance, this simulates increasing sample size.

3 pts.

c) State two factors that may influence the performance of AdaBoost?
   Answer:
   1) if the number of outliers is large, then the emphasis placed on these outliers can become large.

2 pts.

d) How can we use AdaBoost to detect outliers (mislabeled or ambiguously labeled examples)?
   Answer: By finding the classifier with the highest weight

2 pts.

For a binary classification problem (± 1), assume we have $c_1, ..., c_B$ successive base classifiers obtained via AdaBoost, and let $\alpha_1, ..., \alpha_B$ be the corresponding weights. The ensemble classifier is $\hat{c}_B(x) = \text{sgn}(\sum_{b=1}^{B} \alpha_b c_b(x))$.

Reusing the same base classifiers $c_1, ..., c_B$, we now train a *linear* classifier, by finding *new* $\tilde{\alpha}_b$'s that minimize the average exponential loss on the training examples, and obtain $\tilde{\hat{c}}_B(x) = \text{sgn}(\sum_{b=1}^{B} \tilde{\alpha}_b c_b(x))$.

e) Is $\tilde{\hat{c}}_B$ equivalent to $\hat{c}_B$? Justify your answer.
   Answer:

$$\hat{C}_B(x) = \text{sgn}\left(\sum_{b=1}^{B} \alpha_b c_b(x)\right) \Rightarrow \tilde{\hat{C}}_B(x) = \text{sgn}\left(\sum_{b=1}^{B} \tilde{\alpha}_b c_b(x)\right)$$

Adaboost = linear classifier

yes, it is equivalent because the sum of classifiers given, weights is a linear fn αc

4 pts.

Consider the following simplified ensemble learning algorithm:

We consider a dataset $\{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$ of N i.i.d. samples, where $y_n \in \{-1, +1\}$. After each iteration b of this algorithm, each training sample x_n has an associated weight $w_{b,n}$. At the beginning, $w_{0,n} = 1$ for $n = 1, ..., N$. Let $0 < \gamma < \frac{1}{2}$ and $\beta = \frac{1+2\gamma}{1-2\gamma}$.

For each iteration $b = 1$ to B:

- The algorithm finds a new weak learner c_b on the weighted training set that is guaranteed to have an error of at most $\frac{1}{2} - \gamma$ on the weighted training set.
- We then update the weights of the training samples:
  - If x_n is misclassified by c_b, i.e., if $c_b(x_n) \neq y_n$: Set $w_{b,n} := \beta w_{b-1,n}$.
  - Otherwise the weight remains unchanged: $w_{b,n} := w_{b-1,n}$.

The ensemble classifier decides by combining the weak learners as follows:

$$\hat{c}_B(x) = \text{sgn}(\sum_{b=1}^{B} c_b(x)).$$

Let $W_b = \sum_{n=1}^{N} w_{b,n}$ be the sum of the weights at the end of iteration b.

f) What is the difference between AdaBoost and this algorithm in the way they combine the weak learners?
   Answer: The don't directly place coefficients with each weak learner, but instead place weights on training samples

   Adaboost assigns a weight to each classifier based on performance.

2 pts.

g) Derive a bound on W_B, the sum of the weights after the final iteration. Given that at each iteration b, the sum of the weights increases by at most a factor of $1 + 2\gamma$:

$$\frac{W_b}{W_{b-1}} \leq 1 + 2\gamma,$$

show that for the final sum of weights it holds that

$$W_B \leq N(1 + 2\gamma)^B.$$

Answer: $W_B \leq (1+2\gamma)\, W_{b-1}$

4 pts.

h) Prove the following lower bound on the final weight of a training sample, if it is misclassified by the ensemble:

If $\hat{c}_B(x_n) \neq y_n$, then $w_{B,n} \geq \beta^{B/2}$.

Answer:

4 pts.

i) Finally, we will show a bound on the empirical error rate on the training data. Let $M = |\{n : \hat{c}_B(x_n) \neq y_n\}|$ be the number of training samples misclassified by the ensemble classifier.

1. Plugging the two previous bounds together, we have

$$M\beta^{B/2} \leq \sum_{n=1}^{N} w_{n,B} = W_B \leq N(1+2\gamma)^B.$$

Starting from this inequality, show that the empirical error rate on the training data is bounded by the following:

$$\frac{M}{N} \leq e^{-2B\gamma^2}.$$

(*Hint: note that* $(1+a)^c \leq e^{ac}$ *for* $c \geq 0$*, and recall that* $\beta = \frac{1+2\gamma}{1-2\gamma}$)

Answer:

2. Explain what happens to this bound when we
   (a) increase B (the number of training iterations),
   (b) increase γ (have a very good learner).

   Answer:

7 pts.

33 Exam 2009 solutions

These are the solutions for the exam of 2009. Unfortunately, nobody wrote them yet.

These solutions might be plain wrong since they were not verified by a teaching assistant but were written by a student. Be cautious! We hope that they help you even though.

Write the solutions for exam 2009.

34 Glossary

Term	Brief explanation of the term
A very important term	The much longer explanations breaking over several lines here and more text to see the line breaking again and again and again and again.
Classification	
Regression	
Structured Prediction	
Supervised Learning	
Unsupervised Learning	'Learning without labels' Usual goals: 1. Compact representation of data sets (e.g. via clustering / quantization) 2. Dimension reduction (e.g.; very useful for visualization) 3. Anomaly detection of "unusual" data points
Clustering	
Dimension reduction	
Anomaly detection	
Feature	
Model selection	
Model validation	
Nominal or categorical scale	Qualitative, but without quantitative measurements, e.g. binary scale $\mathbb{X} = \{0, 1\}$ (presence or absence of properties). Ordering does not matter.
Ordinal scale	Measurement values are meaningful only with respect to other measurements, i.e., the rank order of measurements carries the information, not the numerical differences *(e.g. information on the ranking of different marathon races)*
Interval scale	The relation of numerical differences carries the information. Invariance w.r.t. translation and scaling *(Fahrenheit scale of temperature)*.
Ratio scale	Zero value of the scale carries information but not the measurement unit. *(Kelvin scale)*.

Term	Brief explanation of the term
Absolute scale	Absolute values are meaningful. *(grades of final exams)*
Data Whitening	Normalize the values of a feature vector by the standard deviation (or another scale quantity) in this component. Thereby, differences in dynamic range (e.g., by different measurement units) are eliminated.
Homogeneous coordinates	Suppose we have a point (x,y) in the Euclidean plane. To represent this same point in the projective plane, we simply add a third coordinate of 1 at the end: (x, y, 1) Overall scaling is unimportant, so the point (x,y,1) is the same as the point $(\alpha x, \alpha y, \alpha)$, for any nonzero α. In other words, $$(X, Y, W) = (\alpha X, \alpha Y, \alpha W)$$ for any $\alpha \neq 0$ (Thus the point (0,0,0) is disallowed). Because scaling is unimportant, the coordinates (X,Y,W) are called the homogeneous coordinates of the point. (Stanford Robotics: http://robotics.stanford.edu/~birch/projective/node4.html)
Singular value decomposition	The singular value decomposition (SVD) is a factorization of a matrix. The singular value decomposition of an $m \times n$ matrix X is a factorization of the form $X = UDV^T$, where U is an $m \times m$ orthogonal matrix, D is an $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal, and V^T is an $n \times n$ orthogonal matrix. The diagonal entries D_i are known as the singular values of X. The m columns of U and the n columns of V are called the left-singular vectors and right-singular vectors of X, respectively. (SVD MIT Open courseware/ Computing the Singular Value Decomposition: http://youtu.be/cOUTpqlX-Xs)
Norms (L_1, L_2 norm)	Norms determine the length of a vector in a certain space. In the Euclidian space we use the L_2 norm which is $$\|x\|_2 = \sqrt{(x_1^2 + x_2^2 + \cdots + x_n^2)}$$ whereas if we want to measure the grid distance, we use the L_1 norm or "Manhattan distance" which is $$\|x\|_1 = (\lvert x_1\rvert + \lvert x_2\rvert + \cdots + \lvert x_n\rvert)$$
Regularization	Regularization, in mathematics and statistics and particularly in the fields of machine learning and inverse problems, refers to a process of introducing additional information in order to solve an ill-posed problem or to prevent overfitting. This information is usually of the form of a penalty for complexity, such as restrictions for smoothness or bounds on the vector space norm.

Term	Brief explanation of the term
Likelihood function $\log P(\theta; X)$	In statistics, a likelihood function (often simply the likelihood) is a function of the parameters of a statistical model. Likelihood functions play a key role in statistical inference, especially methods of estimating a parameter from a set of statistics. In informal contexts, "likelihood" is often used as a synonym for "probability." But in statistical usage, a distinction is made depending on the roles of the outcome or parameter. Probability is used when describing a function of the outcome given a fixed parameter value. For example, if a coin is flipped 10 times and it is a fair coin, what is the probability of it landing heads-up every time? Likelihood is used when describing a function of a parameter given an outcome. For example, if a coin is flipped 10 times and it has landed heads-up 10 times, what is the likelihood that the coin is fair? (Wikipedia Likelihood function: https://en.wikipedia.org/wiki/Likelihood_function)
Score function $\Lambda = \frac{\partial \log P(x;\theta)}{\partial \theta}$	The score function indicates how sensitively a likelihood function $\log P(\theta; X)$ depends on its parameter θ. Explicitly, the score for θ is the gradient of the log-likelihood with respect to θ.

Add more terms to the glossary. Link them to the word they describe in the text by using \hyperref[label name]{labelID} to make a link to the position in the text marked by \label{labelID}.

35 TODO

This is the chapter on what still has to be improved in the summary. Please update this list by writing the todos directly into the summary where necessary by writing:

```
\todo{Something that still has to be done}
```

For a missing figure, please write:

```
\missingfigure{A missing figure}
```

Everyone that picks up the work on this summary will thank you for keeping this up-to-date properly.

Todo list