MACHINE LEARNING

Linear and Weighted Regression
Support Vector Regression
Classification (reminder)

Maps $N$-dimensions input $x \in \mathbb{R}^N$ to discrete values $y$

E.g.: $x = \text{[Length, Color]}$  “Banana” or “Apple”

How to estimate a **continuous** output value $y$?
Regression: introduction

Maps N-dimensions input $x \in \mathbb{R}^N$ to continuous values $y \in \mathbb{R}$

Income (GDP) Continuous value of life satisfaction

Life satisfaction

Income: GDP 2003 (log scale)
Regression: introduction

Maps N-dimensions input $x \in \mathbb{R}^N$ to continuous values $y \in \mathbb{R}$

Income (GDP)  Continuous value of life satisfaction

Estimation of life satisfaction = 6.5

Query point: Russia
GDP = 30 000
Example of Use of Regressive Methods

Predict the number of diplomas that will be awarded in the next ten years across the two EPF → the number of diploma follow a non-linear curve as a function of time.
Example of Use of Regressive Methods

Predict the velocity of the robot given its position. \( \dot{x} = f(x) \)
Example of Use of Regressive Methods

Robustness to perturbations
Linear Regression

Linear regression searches a linear mapping between input $x$ and output $y$, parametrized by the slope vector $w$ and intercept $b$.

$$y = f(x; w, b) = w^T x + b$$
Linear Regression

Linear regression searches a linear mapping between input $x$ and output $y$, parametrized by the slope vector $w$ and intercept $b$.

$$y = f(x; w, b) = w^T x + b$$

One can omit the intercept by centering the data:

- $y' = y - \bar{y}$ and $x' = x - \bar{x}$, \hspace{1em} $\bar{x}, \bar{y}$: mean on $x$ and $y$
- $y' = w^T x' + b'$
- with $b' = b + w^T \bar{x} - \bar{y}$
- Least-square estimate of $(b')^* = \bar{y}' - w^T \bar{x}' = 0$
- $\Rightarrow y' = w^T x'$. 
Linear Regression

Linear regression searches a linear mapping between input $x$ and output $y$, parametrized by the slope vector $w$.

$$y = f(x; w) = w^T x$$
Linear Regression

Pair of $M$ training points $X = [x^1 \ x^2 \ ... \ x^M]$ and $y = [y^1 \ y^2 \ ... \ y^M]$ 

$x^i \in \mathbb{R}^N$, $y^i \in \mathbb{R}$.

Find the optimal parameter $w$ through least-square regression:

$$w^* = \min_w \left( \sum_{i=1}^{M} \frac{1}{2} \left( w^T x^i - y^i \right)^2 \right)$$

Finds an analytical solution through partial differentiation:

$$w^* = \left( X^T X \right)^{-1} X^T y$$
Weighted Linear Regression

Regression through \textit{weighted} Least Square

\[
    w^* = \min_w \left( \sum_{i=1}^{M} \frac{1}{2} \beta_i (w^T x^i - y^i)^2 \right), \quad \beta_i \in \mathbb{R} \& \beta_1 = \beta_2 \cdots = \beta_M
\]

\(\Rightarrow\) Standard linear regression

All points have equal weight.
Weighted Linear Regression

Regression through \textit{weighted} Least Square

\[
 w^* = \min_w \left( \sum_{i=1}^{M} \frac{1}{2} \beta_i \left( w^T x^i - y^i \right)^2 \right), \quad \beta_i \in \mathbb{R}
\]
Weighted Linear Regression

Regression through \textit{weighted} Least Square

\[ w^* = \min_w \left( \sum_{i=1}^{M} \frac{1}{2} \beta_i \left( w^T x^i - y^i \right)^2 \right), \quad \beta_i \in \mathbb{R} \]

Points in red have large weights.
Weighted Linear Regression

Assuming a set of weights \( \beta_i \) for all datapoints, we set \( B \) a diagonal matrix

\[
B = \begin{bmatrix}
\beta_1 \\
\beta_2 \\
\vdots \\
\beta_M
\end{bmatrix}
\]

with entries \( \beta_i \),

\[
\text{Change of variable: } Z = BX^T \quad \text{and} \quad v = By.
\]

Minimizing for MSE, one gets an estimator for \( y \) at the query point:

\[
\hat{y} = x^T w = x^T \left( Z^T Z \right)^{-1} Z^T v
\]

Contrast to the solution for un-weighted linear regression

\[
w^* = \left( X^T X \right)^{-1} X^T y
\]
Limitations of Linear Regression

Regression through **weighted** Least Square

\[ w^* = \min_w \left( \sum_{i=1}^{M} \frac{1}{2} \beta_i \left( w^T x^i - y^i \right)^2 \right), \quad \beta_i \in \mathbb{R}: \text{constant weights} \]

assumes that a single linear dependency applies everywhere.
Not true for data sets with local dependencies.
Limitations of Linear Regression

Regression through **weighted** Least Square

\[ w^* = \min_{w} \left( \sum_{i=1}^{M} \frac{1}{2} \beta_i \left( w^T x^i - y^i \right)^2 \right), \quad \beta_i \in \mathbb{R}: \text{constant weights} \]

assumes that **a single linear dependency** applies everywhere.

Not true for data sets with **local dependencies**.

\[ \rightarrow \text{It would be useful to design a regression method that estimates best the linear dependencies locally.} \]
Locally Weighted Regression

Estimate is determined through local influence of each group of datapoints

\[ \hat{y}(x) = \sum_{i=1}^{M} \beta_i(x) y_i / \sum_{j=1}^{M} \beta_j(x) \]  

\[ \beta_i(x) = \frac{1}{\sqrt{K(d(x', x))}}, \quad \text{with} \quad K(d(x', x)) = e^{-d(x', x)}, \quad d(x', x) = \|x' - x\|. \]

\[ x : \text{query point} \]

Weighting Kernel:
Locally Weighted Regression

Estimate is determined through local influence of each group of datapoints

\[ \hat{y}(x) = \sum_{i=1}^{M} \beta_i(x) y^i / \sum_{j=1}^{M} \beta_j(x) \quad \beta_i(x) \in \mathbb{R}: \text{weights function of } x \]

\[ \beta_i(x) = \sqrt{K(d(x^i, x))}, \quad \text{with } K(d(x^i, x)) = e^{-d(x^i, x)}, \quad d(x^i, x) = \|x^i - x\|. \]

Generates a smooth function \( y(x) \)
Locally Weighted Regression

Estimate is determined through local influence of each group of datapoints

\[
\hat{y}(x) = \frac{\sum_{i=1}^{M} \beta_i(x) y_i}{\sum_{j=1}^{M} \beta_j(x)}
\]

\(\beta_i(x) \in \mathbb{R}\): weights function of \(x\)

Model-free regression!
No longer explicit model of the form \(y = w^T x\)

Regression computed at each query point.
Depends on training points.
Locally Weighted Regression

Estimate is determined through local influence of each group of datapoints

\[ \hat{y}(x) = \sum_{i=1}^{M} \beta_i(x) y^i / \sum_{j=1}^{M} \beta_j(x) \quad \beta_i(x) \in \mathbb{R}: \text{weights function of } x \]

\[ \beta_i(x) = \sqrt{K(d(x^i, x))} \]

Optimal solution to the local cost function:

\[ \min J(x) = \min \sum_{i=1}^{M} \left( \hat{y} - y^i \right)^2 K\left(d(x^i, x)\right) \]

Local cost function at \( x \), the query point.
Locally Weighted Regression

Estimate is determined through local influence of each group of datapoints

\[ \hat{y}(x) = \sum_{i=1}^{M} \beta_i(x) y^i \sum_{j=1}^{M} \beta_j(x) \beta_i(x) \in \mathbb{R}: \text{weights function of } x \]

\[ \beta_i(x) = \sqrt{K(d(x^i, x))} \]

Which training points? Which kernel?
Exercise Session Part I
Data-driven Regression

Good prediction depends on the choice of datapoints.

Blue: true function
Red: estimated function
Data-driven Regression

Good prediction depends on the choice of datapoints.

The more datapoints, the better the fit.

Computational costs increase dramatically with number of datapoints.
Data-driven Regression

Several methods in ML for performing non-linear regression. Differ in the objective function, in the amount of parameters.

_Gaussian Process Regression (GPR) uses all datapoints (model-free)_

_Gaussian Process Regression not covered in class! Not examined in the final exam!_
Data-driven Regression

Several methods in ML for performing non-linear regression.

Differ in the objective function, in the amount of parameters.

*Gaussian Process Regression (GPR) uses all datapoints (model-free)*

*Support Vector Regression (SVR) picks a subset of datapoints (support vectors)*

Blue: true function
Red: estimated function
Data-driven Regression

Several methods in ML for performing non-linear regression.

Differ in the objective function, in the amount of parameters.

*Gaussian Process Regression (GPR) uses all datapoints (model-free)*
*Support Vector Regression (SVR) picks a subset of datapoints (support vectors)*
*Gaussian Mixture Regression (GMR) generates a new set of datapoints (centers of Gaussian functions)*
Data-driven Regression

Estimate of the noise is important to measure goodness of fit.
Support Vector Regression (SVR) assumes an estimate of the noise model ($\varepsilon$-tube) and then compute $f$ directly within a noise-tolerance.

Estimate of the noise is important to measure goodness of fit.
Data-driven Regression

Estimate of the noise is important to measure goodness of fit.

Gaussian Mixture Regression (GMR) builds a local estimate of the noise model through the variance of the system.
Support Vector Regression
Support Vector Regression

Assume a nonlinear mapping $f$, s.t. $y = f(x)$.

How to estimate $f$ to best predict the pair of training points $\{x^i, y^i\}_{i=1,...,M}$?

How to generalize the support vector machine framework for classification to estimate continuous functions?

1. Assume a non-linear mapping through feature space and then perform linear regression in feature space

2. Supervised learning – minimizes an error function.

$\rightarrow$ First determine a way to measure error on testing set in the linear case!
Assume a linear mapping $f$, s.t. $y = f(x) = w^T x + b$. 

Measure the error on prediction $b$ is estimated as in SVR through least-square regression on support vectors; hence we ignore it for the rest of the developments.

How to estimate $w$ and $b$ to best predict the pair of training points $\{x^i, y^i\}_{i=1,...,M}$?
Support Vector Regression

Set an upper bound on the error $\varepsilon$ and consider as correctly classified all points such that $f(x) - y \leq \varepsilon$.

Penalize only datapoints that are not contained in the $\varepsilon$-tube.

\[ \hat{y} = f(x) = w^T x + b \]
Support Vector Regression

The $\varepsilon$-margin is a measure of the width of the $\varepsilon$-insensitive tube. It is a measure of the precision of the regression.

A small $||w||$ corresponds to a small slope for $f$. In the linear case, $f$ is more horizontal.
Support Vector Regression

A large $|\|w\||$ corresponds to a large slope for $f$. In the linear case, $f$ is more vertical.

The flatter the slope of the function $f$, the larger the $\varepsilon$-margin.

➔ To maximize the margin, we must minimize the norm of $w$. 

$\varepsilon$-margin
Support Vector Regression

This can be rephrased as a constraint-based optimization problem of the form:

\[
\minimize \frac{1}{2} \|w\|^2 \\
\text{subject to } \begin{cases} 
\langle w, x^i \rangle + b - y^i \leq \varepsilon \\
y^i - \langle w, x^i \rangle - b \leq \varepsilon 
\end{cases} \quad \forall i = 1, \ldots, M
\]

Set an upper bound on the error \( \varepsilon \) and consider as correctly classified all points such that \( f(x) - y \leq \varepsilon \).

Need to penalize points outside the \( \varepsilon \)-insensitive tube.
Support Vector Regression

Introduce slack variables $\xi_i, \xi_i^*, C \geq 0$:

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \|w\|^2 + \frac{C}{M} \sum_{i=1}^{M} (\xi_i + \xi_i^*) \\
\text{subject to} & \quad \langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi_i \\
& \quad y_i - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i^* \\
& \quad \xi_i \geq 0, \quad \xi_i^* \geq 0
\end{align*}
\]

Need to penalize points outside the $\varepsilon$-insensitive tube.
Support Vector Regression

Introduce slack variables $\xi_i, \xi_i^*, C \geq 0$:

\[
\begin{align*}
\text{minimize} \quad & \frac{1}{2} \|w\|^2 + \frac{C}{M} \sum_{i=1}^{M} (\xi_i + \xi_i^*) \\
\text{subject to} \quad & \langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi_i \\
& y_i - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i^* \\
& \xi_i \geq 0, \quad \xi_i^* \geq 0
\end{align*}
\]

We now have the solution to the linear regression problem.

How to generalize this to the nonlinear case?
Lift $x$ into feature space and then perform linear regression in feature space.

**Linear Case:**

$$y = f(x) = \langle w, x \rangle + b$$

**Non-Linear Case:**

$$x \rightarrow \phi(x)$$

$$y = f(\phi(x)) = \langle w, \phi(x) \rangle + b$$

$w$ lives in feature space!
Support Vector Regression

In feature space, we obtain the same constrained optimization problem:

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2}\|w\|^2 + \frac{C}{M} \sum_{i=1}^{M} (\xi_i + \xi_i^*) \\
\text{subject to} & \quad \left\lfloor \begin{array}{l}
\langle w, \phi(x^i) \rangle + b - y^i \leq \varepsilon + \xi_i \\
y^i - \langle w, \phi(x^i) \rangle - b \leq \varepsilon + \xi_i^* \\
\xi_i \geq 0, \quad \xi_i^* \geq 0
\end{array} \right.
\end{align*}
\]
Support Vector Regression

Again, we can solve this quadratic problem by introducing sets of Lagrange multipliers and writing the Lagrangian:

\[
\text{Lagrangian} = \text{Objective function} + \lambda \times \text{constraints}
\]

\[
L(w, \xi, \xi^*, b) = \frac{1}{2} \|w\|^2 + \frac{C}{M} \sum_{i=1}^{M} (\xi_i + \xi_i^*) - \frac{C}{M} \sum_{i=1}^{M} (\eta_i \xi_i + \eta_i^* \xi_i^*)
\]

\[
- \sum_{i=1}^{M} \alpha_i \left( \varepsilon + \xi_i + y_i - \left\langle w, \phi(x_i) \right\rangle - b \right)
\]

\[
- \sum_{i=1}^{M} \alpha_i^* \left( \varepsilon + \xi_i^* - y_i + \left\langle w, \phi(x_i) \right\rangle + b \right)
\]
Support Vector Regression

\( \alpha_i^* \text{ & } \alpha_i \neq 0 \) for all points that do not satisfy the constraints
→ points outside the \( \varepsilon \)-tube

\( \alpha_i > 0 \)

Constraints on points lying on either side of the \( \varepsilon \)-tube

\[
- \sum_{i=1}^{M} \alpha_i \left( \varepsilon + \xi_i + y^i - \left\langle w, \phi(x^i) \right\rangle - b \right)
\]

\[
- \sum_{i=1}^{M} \alpha_i^* \left( \varepsilon + \xi_i^* - y^i + \left\langle w, \phi(x^i) \right\rangle + b \right)
\]
Support Vector Regression

\( \alpha_i^* \neq 0 \) for all points that do not satisfy the constraints

→ points outside the \( \varepsilon \)-tube

\[ \alpha_i > 0 \]

Requiring that the partial derivatives are all zero:

\[
\frac{\partial L}{\partial b} = \sum_{i=1}^{M} (\alpha_i - \alpha_i^*) = 0.
\]

\[
\Rightarrow \sum_{i=1}^{M} \alpha_i = \sum_{i=1}^{M} \alpha_i^*
\]

\[
\frac{\partial L}{\partial w} = w - \sum_{i=1}^{M} (\alpha_i^* - \alpha_i) \phi(x^i) = 0.
\]

\[
\Rightarrow w = \sum_{i=1}^{M} (\alpha_i^* - \alpha_i) \phi(x^i).
\]

Rebalancing the effect of the support vectors on both sides of the \( \varepsilon \)-tube

Linear combination of support vectors
Support Vector Regression

And replacing in the primal Lagrangian, we get the Dual optimization problem:

$$
\begin{align*}
\max_{\alpha, \alpha^*} & \quad -\frac{1}{2} \sum_{i, j=1}^{M} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) \cdot \langle \phi(x^i), \phi(x^j) \rangle \\
& \quad - \varepsilon \sum_{i=1}^{M} (\alpha_i^* + \alpha_i) + y^i \left( \alpha_i^* + \alpha_i \right) \\
\text{subject to} & \quad \sum_{i=1}^{M} (\alpha_i^* - \alpha_i) = 0 \quad \text{and} \quad \alpha_i^*, \alpha_i \in \left[ 0, \frac{C}{M} \right]
\end{align*}
$$

Kernel Trick

$$
k(x^i, x^j) = \langle \phi(x^i), \phi(x^j) \rangle$$
Support Vector Regression

The solution is given by:

\[ y = f(x) = \langle w, x \rangle + b = \sum_{i=1}^{M} (\alpha_i^* - \alpha_i) \langle \phi(x^i), \phi(x) \rangle + b \]

If one uses RBF Kernel, \( M \) un-normalized isotropic Gaussians centered on each training datapoint.

Kernel Trick:

\[ k(x^i, x^j) = \langle \phi(x^i), \phi(x^j) \rangle \]

Linear Coefficients
(Lagrange multipliers for each constraint).
Support Vector Regression

The solution is given by:

\[ y = f(x) = \sum_{i=1}^{M} (\alpha_i^* - \alpha_i) k(x^i, x) + b \]

Kernel places a Gauss function on each SV
The solution is given by:

\[ y = f(x) = \sum_{i=1}^{M} (\alpha_i - \alpha_i^*) k(x^i, x) + b \]

The Lagrange multipliers define the importance of each Gaussian function.

Converges to b when SV effect vanishes.
Support Vector Regression

SVR gives the following estimate for each pair of datapoints \( \{ y^j, x^j \} \)

\[
y^j = \sum_{i=1}^{M} (\alpha_i - \alpha_i^*) k(x^j, x^i) + b, \quad i = 1 \ldots M
\]

An estimate of \( b \) can be computed using the above:

\[
\Rightarrow b = \frac{1}{M} \sum_{j=1}^{M} \left( y^j - \sum_{i=1}^{M} (\alpha_i - \alpha_i^*) k(x^j, x^i) \right)
\]
The solution to SVR we just saw is referred to as $\varepsilon$–SVR

Two Hyperparameters

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \|w\|^2 + \frac{C}{M} \sum_{i=1}^{M} (\xi_i + \xi_i^*) \\
\text{subject to} & \quad \langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi_i \\
& \quad y_i - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i^* \\
& \quad \xi_i \geq 0, \quad \xi_i^* \geq 0
\end{align*}
\]

$C$ controls the penalty term on poor fit
$\varepsilon$ determines the minimal required precision
ε–SVR: Effect of Hyperparameters

Effect of the RBF kernel width on the fit. Here fit using C=100, ε=0.1, kernel width=0.01.
ε-SVR: Effect of Hyperparameters

Effect of the RBF kernel width on the fit. Here fit using C=100, ε=0.01, kernel width=0.01 → Overfitting
ε–SVR: Effect of Hyperparameters

Effect of the RBF kernel width on the fit. Here fit using $C=100$, $\varepsilon=0.05$, kernel width=0.01. Reduction of the effect of the kernel width on the fit by choosing appropriate hyperparameters.
$\varepsilon$-SVR: Effect of Hyperparameters

Mldemos does not display the support vectors if there is more than one point for the same x!
Summary

Linear regression can be solved through Least-Mean-Square estimation and yields an optimal analytical solution.

Weighted regression offers the possibility to perform a local regression and yields also an optimal analytical solution. The estimate is no longer global and is computed around each group of data point!

Support Vector Regression: performs regression on a non-linear function. Determines automatically the important points. The estimate is globally optimal.
Examples of Applications of SVR Next
Model Object’s Dynamics

Build model of dynamics using Support Vector Regression

\[ \ddot{x} = \sum_{i=1}^{M} \alpha_i k \left( [x^i \dot{x}^i]^T, [x \dot{x}]^T \right) + b \]

Compute derivative (closed form)

Use model in Extended Kalman Filter for real-time tracking
Application of SVR: Mapping Eyes to Gaze

- Designed for children from 1 year of age
- Fruit of 3 years of development
  Lorenzo Piccardi, Jean-Baptiste Keller, Martin Duvanel, Olivier Barbey, Karim Benmachiche, Dario Poggiali, Dave Bergomi, Basilio Noris
- 2 cameras, 2 microphones, 1 mirror
  96° x 96° field of view, 25Hz / 50Hz, 180g

www.pomelo-technologies.com
Application of SVR: Mapping Eyes to Gaze

Application of SVR: Mapping Eyes to Gaze
Application of SVR: Mapping Eyes to Gaze

Use Support Vector Regression to learn the mapping from eyes appearance to gaze coordinates.
Application of SVR: Mapping Eyes to Gaze

We normalize the image through high-pass filtering.

We collect images of the eyes and directions of the gaze.

Learn mapping Eye Image $\rightarrow$ Position in Image through Support Vector Regression (SVR)
Application of SVR: Mapping Eyes to Gaze

Different elements give different cues

Pupil, Iris and Cornea

Wrinkles, Eyelids and Eyelashes

Support Vector Regression
Application of SVR: Mapping Eyes to Gaze
From object recognition using Eye tracking

To reconstructing path in Shop

www.pomelo-technologies.com
Monitoring Consumers’ Visual Behavior

- Gaze tracking using SVR
- Object detection using SVM
Msc Projects in Industry

Detection of product purchases from shelves in unconstrained, uncalibrated, heavily cluttered environments

Our work requires us to track changes within shelves in retail environments, corresponding to shoppers picking up products and buying them or putting them back on the shelves. Our methodology involves analysing video from one or more cameras recording up to 16th of store activity.

The task is complex as high numbers of shoppers move and occlude the videos when approaching the shelves. Viewing angles are sometimes drifting throughout the day and there is no opportunity for calibration of the recording equipment when deploying in store.

The goal of the project is to leverage on the existing background subtraction system to integrate shopper occlusion and gradual camera movements to facilitate the detection process.

Knowledge requirements: Very good knowledge of C++ and fundamentals of machine learning/computer vision (knowledge of OpenCV is a welcome plus).

Interested candidates should send an email to aude.billard@epfl.ch with a copy of their CV and grades.

Project: Master Project in Industry
Position: EL IN MT
Season(s): 01.09.2016 - 01.08.2017
Knowledge(s): Machine Learning, Programming in C++
Subject(s): Machine Learning
Responsible(s): Sascha Nentis
URL: Click here

Extraction of profiles of shopping and purchases patterns

In the scope of our shopper behaviour studies, we are confronted with the task of segmenting profiles of shoppers according to the type of purchases they make (categories of products, departments visited, ...)

The goal of this project is to investigate the structure of shopping profiles in terms of clustering of space, high-dimensional data from hundreds of ’shopping acts’, and to extract the main purchasing profiles and product category “proximities” according to the shopping behaviour rather than their relative location in store.

Knowledge requirements: Good understanding of Machine Learning concepts and tools (Clustering, Dimensional Reduction / Scaling, ...). Good knowledge of at least one Machine Learning-related programming language (e.g. Matlab, R, Python, C++)

Interested candidates should send an email to aude.billard@epfl.ch with a copy of their CV and grades.

Project: Master Project in Industry
Position: EL IN MT
Season(s): 01.09.2016 - 01.08.2017
Knowledge(s): Machine Learning, Programming in C++ / Python / Matlab / R