MATERIAL FOR LINEAR MODELS
MACHINE LEARNING E20

Mathias Ravn Tversted
Department of Computer Science
Aarhus University
Åbogade 34, 8200 Aarhus N
`Tversted@post.au.dk`

December 27, 2020

Contents

1 What are linear models? Classification

A good way to explain linear models is to look at the problem of *linear classification*

IINDSÆT PLOT HER AF HVAD DU VIL BRUGE SOM EKSEMPEL

In the above example, we are looking to draw a *line* which serves as a way to *classify data*. This is an example of a linear model.

2 (Pocket) Perceptron Learning Algorithm

A way to nicely do *linear classification* is to use the *Perceptron Learning Algorithm*. Let $X = \{1\} \times \mathbb{R}^d$ be the input space. Let $Y \in \{+1, -1\}$ be the output space. The coordinates of the x-vector corresponds to features of the input. We are looking to "train" weights and bias $w \in R^{d+1}$ such that $\sum_{i=0}^{d} (\text{sign}(w_i x_i) \neq y_i) = 0$ or, the data is perfectly linearly seperated. This is only possible if there exists some hypothesis that can actually do this.

2.1 The Perceptron Learning Algorithm

1. Pick an x s.t $y(t) \neq \text{sign}(w^T(t)x(t))$.
2. *Move the line in that direction* with update rule: $w(t+1) = w(t) + y(t)x(t)$

This algorithm is *guaranteed to terminate* given the data is linearly seperable. INDSÆT HER HVAD ER KØRETIDEN FOR PLA?

2.2 Quick note: Pocket Perceptron Learning Algorithm

If the data is not linearly seperable, we can use the *Pocket Perceptron Learning Algorithm*. We want to find the w that minimises $E_{in}(w) = \frac{1}{n} \sum_{i=1}^{n} 1_{sign(x_i^T w) \neq y_i}$.

Run for set number of epochs, keep track of best. Return best solution after those epochs.

3 Non-linear transforms

If the data is not linearly seperable, then we can transform the data to take advantage of potentially linearly seperable properties.

A *non-linear feature transform* is some $\Phi : R^d \to R^{d'}$ applied to all feature vectors to get some data-matrix $X' \in Mat_{n \times d'}$.

An obvious example of this is *distance to origin* $\Phi(x_1, x_2) = (x_1^2 + x_2^2)$.

INDSÆT TEGNING HER

Learning theory tells us that simple hypotheses generalise better, and this is worth keeping in mind. INDSÆT MERE OM DET HER.

4 Regression

HERE WE EXPLAIN THE PROBLEM(S) Evt. de forskellige slags regression

- Linear Regression
- Logistic Regression

4.1 Linear Regression

If instead of *classification* we want to find *values*, we can use *regression*. Here we are also drawing some hyperplane, where we have a loss function (MSE) to be minimised $L(w) = \frac{1}{N}(Xw - y)^T(Xw - y)$.

Let $X \in \mathbb{R}^{N \times (d+1)}$ be the matrix of inputs, and $y \in \mathbb{R}^n$ be the target vector.

The in-sample error is MSE

$$E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N}(w^T x_n - y_2)^2 \tag{1}$$

$$= \frac{1}{N}||Xw - y||^2 \tag{2}$$

And we are looking to solve the following optimisation problem: $w_{lin} = \text{argmin}_{w \in \mathbb{R}^{d+1}} E_{in}(w)$

We would like the gradient to be $\nabla E_{in}(w) = 0$. This is a column vector where $[\nabla E_{in}(w)]_i = \frac{\partial}{\partial w_i}E_{in}(w)$.

First we expand our loss function

$$\frac{1}{N}(Xw - y)^T(Xw - y) \tag{3}$$

$$= \frac{1}{N}(w^T X^T X w + y^T y - 2X^T w^T y) \tag{4}$$

Det er kvadratsætningen. Husk at det ikke er $X^T w^T$ da vi bytter rundt på rækkefølgen pga. transponering.

Then we compute the derivative. We have the following identies that can help us:

$$\nabla_w(w^T \cdot A \cdot w) = (A + A^T) \cdot w \qquad \text{Identity 1} \tag{5}$$

$$\nabla_w(w^T * b) = b \qquad \text{Identity 2} \tag{6}$$

$$\nabla E_{in}(w) = \nabla \frac{1}{N}(w^T X^T X w + y^T y - 2X^T w^T y) \tag{7}$$

$$= \frac{2}{N}(X^T X w - X^T y) \tag{8}$$

From this we can see that $\nabla E_{in}(w) = 0$ if $X^T X w = X^T y$.

Page 87, *Linear Regression Algorithm*:

1. Compute *Pseudo-Inverse* $X^\dagger$.
2. If $X^T X$ is invertible, then $X^\dagger = (X^T X)^{-1} X^T$.
3. Return $w_{lin} = X^\dagger y$

Here we have a nice *analytic solution*. This gives us the ability to "predict" future values.

4.2 Logistic Regression

In *logistic regression*, we are looking at probabilities and not values. Output values are thus bounded to the interval $[0, 1]$. We will use the *logistic function* $\theta(s) = \frac{e^s}{1+e^s}$. As in our hypothesis h is $h = \theta(w^T x)$. This output is then interpreted as the *probability for a binary event*, where we trying to learn a target function $f(x) = P[y = +1|x]$, with us trying to find

$$P(y|x) = \begin{cases} f(x) & \text{for } y = +1 \\ 1 - f(x) & \text{for } y = -1 \end{cases} \tag{9}$$

Because of potential noise.

4.2.1 Error measure

: We look at *likelihood*. We want to find a hypothesis h that maximises

$$\Pi_{n=1}^{N} P(y_n|x_n) \tag{10}$$

We prefer minimisation problems, and we want to be able to have an error measure, so we can transform this as follows:

$$-\frac{1}{N}\ln\left(\Pi_{n=1}^{N} P(y_n|x_n)\right) = \frac{1}{N}\sum_{n=1}^{N} ln(\frac{1}{P(y_n|x_n)}) \tag{11}$$

but the probabilities should be our θ function, thus giving us:

$$E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N} ln(\frac{1}{\theta(y_n w^T x_n)}) = \frac{1}{N}\sum_{n=1}^{N} ln(1 + e^{-y_n w^T x_n}) \tag{12}$$

We can not expect to find a nice analytic solution to this problem, so we wil need an algorithm such as $\cdots$

5 (Stochastic) Gradient Descent

Gradient descent allows us to minimise a twice-differentiable function. It allows us to think of $E_{in}(w)$ as a surface, where we can step in the direction of valleys. If the function is *convex*, then it will find the global minimum, otherwise it will find a local minimum.

Fixed learning rate gradient descent

1. Initialize the weights at step $t = 0$ to $w(0)$
2. For $t = 0, 1, 2, \cdots$ **do**
3. Compute gradient $g_t = \nabla E_{in}(w(t))$
4. Set move direction $v_t = -g_t$
5. Update weights $w(t+1) = w(t) + \eta v_t$
6. Iterate until IT'S TIME TO STOP

For logistic regression $\nabla E_{in}(w(t))$ equals $-\frac{1}{N}\sum_{n=1}^{N} \frac{y_n x_n}{1+e^{y_n w^T(t) x_n}}$

Initialization and termination: Good in practice: Choose each weight independently from normal distribution (zero mean) and small variance. INDSÆT HVORNÅR MAN SKAL STOPPE

Learning rate: 0.1 is a decent learning rate in practice.

INDSÆT MERE PRAKTITKS JUNK HER

5.1 Stochastic Gradient descent

We simply look at 1 point for each epoch instead of all the data. It is a much cheaper evaluation (factor N), but it is more jumpy. Eventually these erratic jumps cancel out.