50.059 – Theory and Practice of Deep Learning

Alex

Week 01: The linear model for regression and classification

Learning goals

- See how the linear model can be used for regression and classification
- be able to reproduce the explicit solution for linear regression and ridge regression
- be able to explain how the set of constant points for $\{x : f(x) = w \cdot x = c\}$ looks like for a linear model depending on w and b as parameters
- l2-loss, 0-1-loss, hinge-loss
- a first insight into generalization as one goal of training machine learning model

1 Ordinary Least Squares (Linear regression)

1.1 I. Input and output space?

Some examples:
A. `http://archive.ics.uci.edu/ml/datasets/Real+estate+valuation+data+set`

$X1 =$ the transaction date

$X2 =$ the house age (unit: year)

$X3 =$ the distance to the nearest MRT station (unit: meter)

$X4$ = the number of convenience stores in the living circle on foot (integer)

$X5$ = the geographic coordinate, latitude. (unit: degree)

$X6$ = the geographic coordinate, longitude. (unit: degree)

- The output is as follows Y = house price per unit of area

B. `http://archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data`

$X1$ = Dew Point

$X2$ = Temperature

$X3$ = Pressure

$X4$ = Combined wind direction

$X5$ = Cumulated wind speed

$X6$ = Cumulated hours of snow

$X7$ = Cumulated hours of rain

- The output is as follows Y = PM2.5 concentration ($\mu g/m^3$)

C. `http://archive.ics.uci.edu/ml/datasets/Communities+and+Crime`
(`http://archive.ics.uci.edu/ml/datasets/Concrete+Slump+Test` `https://en.wikipedia.org/wiki/Concrete_slump_test`)
(`http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset`)

Input and output space in regression problems?

$\mathcal{X} = ?, \mathcal{Y} = ?$

1.2 II. the prediction mapping?

Linear function without/with a bias

$$f_w(x) = x \cdot w = \sum_{d=1}^{D} x_d w_d \quad , \; w \in \mathbb{R}^{D\times 1}$$

$$g_{w,b}(x) = x \cdot w + b = \sum_{d=1}^{D} x_d w_d + b \quad , \; w \in \mathbb{R}^{D\times 1}, b \in \mathbb{R}^1$$

weighted sum of features x_d: $X5$ = (Cumulated wind speed) $\leftrightarrow x_5$

1.3 III. Loss function for regression

Loss function to measure quality of prediction for a data sample, here a pair (x, y):

$$\ell(f(x), y) = (f(x) - y)^2$$

Reasons:

- capture deviations on both sides of the real ground truth value y
- simple derivative

1.4 IV. the goal in regression (and machine learning in general!)

Find parameters w, b which generalize well to new, unseen data points.

Lets give here an informal explanation (more formal derivations will be given in a later lecture, here one needs to see that when we do linear regression, there is a principled agenda behind it).
We need a way to model *new, unseen data points*

Assumption 1: We can draw test data sets T_n from your data source. A test data set consists of n pairs (x_i, y_i):

$$T_n = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\} = \{(x_i, y_i), i = 1, \dots, n\}$$

x_i is the input feature, y_i is the ground truth label to it (here: the regression value which x_i is expected to have.)

Assumption 2: We cannot predict which samples (x_i, y_i) we will obtain from your data source as new, unseen data points. Therefore, we model the uncertainty by drawing from a probability for the (x_i, y_i).

$$(x_i, y_i) \sim P_{test}.$$

An Example for such a generating probability
for input samples $x \in [0, 1] \times [0, 1] \subset \mathbb{R}^2$:

$$\begin{aligned}
p(x) &= Unif([0, 1] \times [0, 1]) \\
p(y|x) &= Normal(\mu(x), \sigma^2 = 0.1) \\
\mu(x) &= 2x_1 - 3x_2 \\
P_{test}(x, y) &= p(x)p(y|x)
\end{aligned}$$

Drawing in practice:

- draw $x \sim p(x)$
- draw $y \sim p(y|x)$
- this implies: have $(x, y) \sim p(x)p(y|x) = P_{test}(x, y)$

By drawing n times in this way, one can obtain a training dataset $D_n = \{(x_i, y_i), i = 1, \ldots, n\}$ or a test dataset T_n of independent samples.

Generalization for regression with l2-loss (short form)

From an intuitive view, a mapping generalizes well, if it gives low prediction errors on unseen data samples.

Generalize well means in short form:

- we find parameters (w^*, b^*) defining a mapping f which for most draws of test datasets T_n produces predictions with low errors on them:

$$\begin{aligned}\hat{L}(f, T_n) &= \frac{1}{n} \sum_{(x_i, y_i) \in T_n} \ell(f(x_i), y_i) \\ &= \frac{1}{n} \sum_{i=1}^{n} (f(x_i) - y_i)^2 \to \text{low}\end{aligned}$$

We do not specify here what "low" means. From a practitioners perspective it would be predictions, with errors such that the effort to correct them is low enough, so that using this predictor is productive.

In practice, the search for such parameters is done on a training dataset D_n, and involves minimizing a training loss $\hat{L}(f, D_n)$ computed on the training dataset. Usually the loss is more than just a square difference $(f(x) - y)^2$ and involves regularizer terms.

$$\begin{aligned}(w^*, b^*) &= \text{argmin}_{(w,b)} \hat{L}(f, D_n) = \text{argmin}_{(w,b)} \frac{1}{n} \sum_{(x_i, y_i) \in D_n} \ell(f(x_i), y_i) \\ &= \text{argmin}_{(w,b)} \frac{1}{n} \sum_{(x_i, y_i) \in D_n} (f(x_i) - y_i)^2\end{aligned}$$

Question: Why the goal is not: to have low loss on one test dataset $T_{50} = \{(x_i, y_i), i = 1, \ldots, 50\}$?
That is important to understand, because often errors are measured only on a single test dataset!

1.5 V. What does a linear mapping represent?

Goal: understand what the mapping does.

in class thinking task

What is the set of points $x = (x_1, x_2) \in \mathbb{R}^2$ such that

$$3x_1 - 2x_2 + 3 = 0 \ ?$$

What is the set of points $x = (x_1, x_2, x_3) \in \mathbb{R}^3$ such that

$$x_1 - x_2 - 2x_3 + 2 = 0 \ ?$$

A. What is the set of points x: $g_{w,b}(x) = 0$?

B. What is the set of points x the prediction is a constant c, that is $g_{w,b}(x) = c$?

1.5.1 What is the set of points x: $g_{w,b}(x) = 0$?

$$\begin{aligned} g_{w,b}(x) &= 0 \\ \Leftrightarrow x \cdot w = -b \end{aligned}$$

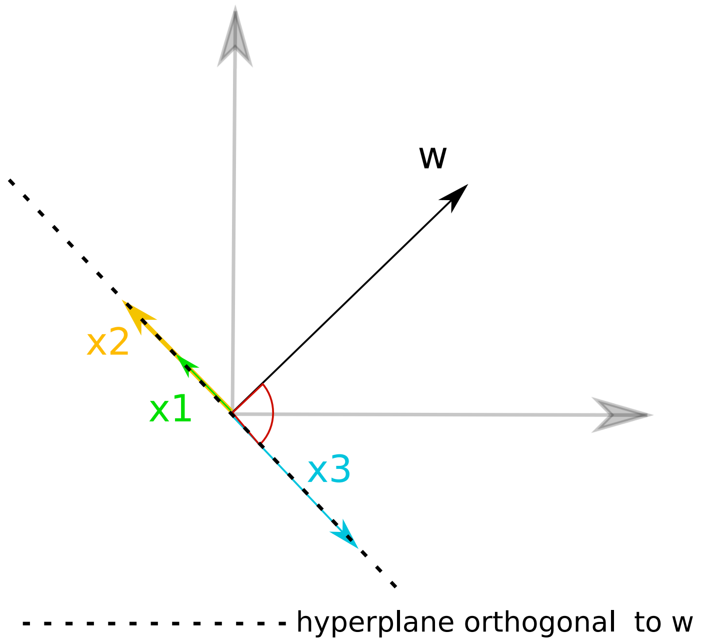
In order to understand how the bias b influences the zero set, lets consider three cases:

- $b = 0$
- $b > 0$
- $b < 0$

The case $b = 0$ We know that for $b = 0$: $g_{w,0}(x) = w \cdot x = 0$ holds for the zero vector $x = 0$.

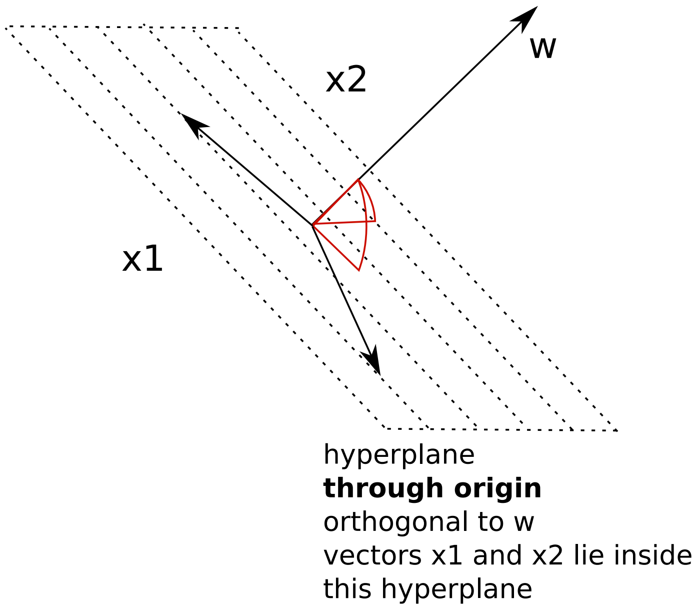
$$x \cdot w = 0$$

The set of points x such that the inner product $x \cdot w = 0$ is in 2 dims a one-dimensional line, which goes through the origin $(x_1, x_2) = (0, 0)$, and which is orthogonal to w.



x_1,x_2,x_3 are all orthogonal to the vector w

The analogy also holds for 3 or more dimensions. So for 3 dims it is a two-dimensional plane, which goes through the origin $(x_1, x_2, x_3) = (0, 0, 0)$.

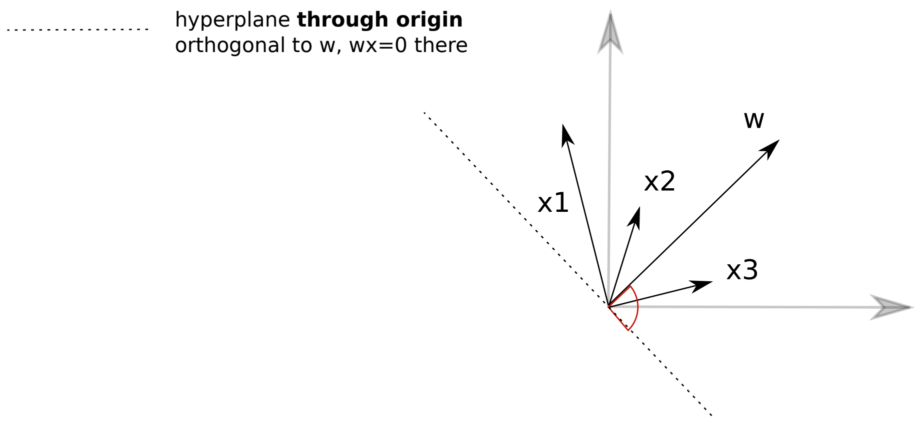


For n dimensions the plane of orthogonal vectors has $n-1$ dimensions (+ goes through the origin), but is still a hyperplane (that is if $x_1 \in P, x_2 \in P \Rightarrow a_1 x_1 + a_2 x_2 \in P$, and P can be represented as a linear combination of $n-1$ basis vectors).

The case $b < 0$

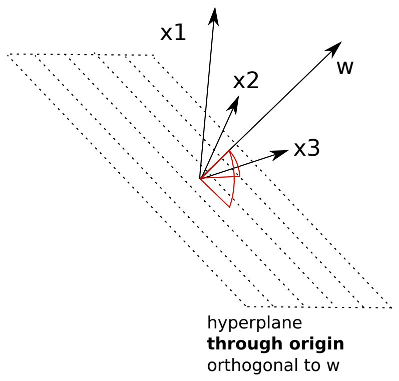
$$b < 0, \ w \cdot x + b = 0 \Rightarrow w \cdot x = -b > 0$$

We know: $w \cdot x > 0$ for all points x that are on that side of the **hyperplane through the origin**, in which w points to.



x_1,x_2,x_3 all have $w \cdot x_i > 0$ because relative to the hyperplane orthogonal to w which goes through the origin, they all point into the direction of w

The same also holds for 3 or more dimensions.



x_1,x_2,x_3 all have $w \cdot x_i > 0$ because relative to the hyperplane orthogonal to w which goes through the origin, they all point into the direction of w

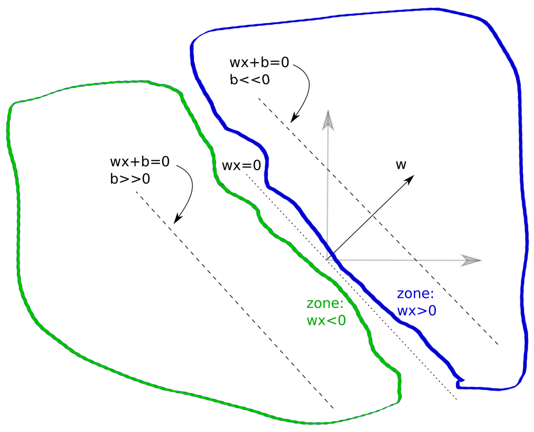
What does that mean? All the above vectors solve $wx + b = 0$ for some bias $b < 0$!

The case $b > 0$
The analogous thought results in the understanding, that set of points x such that

$$\{x : \ wx + b = 0\}$$

is a hyperplane which is parallel to the hyperplane $\{x : x \cdot u = 0\}$ orthogonal to w going through the origin, and which is shifted opposite to the direction of w

As the next graphic shows, the bias b shifts the zero set corresponding to $w \cdot x + b = 0$ anti-parallel to the direction of w.



hyperplane dependency on bias b

- Negative $b < 0$ shift the hyperplane $\{x : wx + b = 0\}$ into the direction of w,
- positive $b > 0$ shift the hyperplane $\{x : wx + b = 0\}$ against the direction of w.
- Large values of $|b|$ shift it away quite far.

hyperplane explicit

As a summary: the linear mapping $g(x) = w \cdot x + b$ has a zero set which is the plane of points

$$\{x : x = u + -b\frac{w}{\|w\|^2}, u \text{ such that } w \cdot u = 0\}$$

In this representation:

- u such that $w \cdot u = 0$ is the hyperplane of vectors u orthogonal to w.
- the vector $-b\frac{w}{\|w\|^2}$ shifts the hyperplane antiparallel to direction of w.

That holds because

$$\begin{aligned} w \cdot x + b &= w \cdot (u + -b\frac{w}{\|w\|^2}) + b \\ &= w \cdot u + w \cdot (-b\frac{w}{\|w\|^2}) + b \\ &= 0 + -b\frac{w \cdot w}{\|w\|^2} + b \\ &= 0 + -b\frac{\|w\|^2}{\|w\|^2} + b = 0 \end{aligned}$$

1.5.2 What is the set of points x where the prediction is a constant, that is $g_{w,b}(x) = c$?

This can be answered by reducing it to a zero set:

$$\begin{aligned} g_{w,b}(x) = wx + b &= c \\ wx + (b - c) &= 0 \end{aligned}$$

So the set of points x such that $g_{w,b}(x) = c$ is just the zero-set of $g_{w,b-c}(x)$.

1.6 VI. a method to select a prediction mapping $g_{w,b}$ / or its parameters from a dataset

First of all, we assume here no bias for this lecture (to get an explicit solution). Parameters are w.
We want to find parameters which minimize the loss on this dataset, that is:

$$(w^*) = \operatorname{argmin}_w \sum_{i=1}^{n} L(f(x_i), y_i) = \operatorname{argmin}_w \sum_{i=1}^{n} (x_i \cdot w - y_i)^2$$

for a given dataset $D_n = \{(x_i, y_i)\}$. Then $f_{w^*}(x) = x \cdot w^*$ is the selected mapping.

Rare case: Can be solved explicitly for w. Write in matrix form:

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} x_1^{(1)}, \dots, x_1^{(D)} \\ x_2^{(1)}, \dots, x_2^{(D)} \\ \vdots \\ x_n^{(1)}, \dots, x_n^{(D)} \end{pmatrix} \in \mathbb{R}^{n \times D}$$

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

then:

$$\sum_{i=1}^{n} (x_i \cdot w - y_i)^2 = (X \cdot w - Y)^T \cdot (X \cdot w - Y)$$

Solve the minimization problem by computing the gradient for w and setting it to zero.

$$\begin{aligned} D_w((X \cdot w - Y)^T \cdot (X \cdot w - Y)) &= 2X^T \cdot (X \cdot w - Y) = 0 \\ \Rightarrow (X^T \cdot X) \cdot w &= X^T \cdot Y \\ w &= (X^T \cdot X)^{-1} X^T \cdot Y \end{aligned}$$

if the matrix inverse $(X^T \cdot X)^{-1}$ exists.

solution without bias

Let X be the training data matrix.

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} x_1^{(1)}, \ldots, x_1^{(D)} \\ x_2^{(1)}, \ldots, x_2^{(D)} \\ \vdots \\ x_n^{(1)}, \ldots, x_n^{(D)} \end{pmatrix} \in \mathbb{R}^{n \times D}$$

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^{n \times 1}$$

The model is $f(x) = w \cdot x$. Then a solution is given as

$$w = (X^T \cdot X)^{-1} X^T \cdot Y$$

if the matrix inverse $(X^T \cdot X)^{-1}$ exists.

How to extend this to a bias?

Extend each feature by an additional dimension set to 1:

$$x = \left(x^{(1)}, \ldots, x^{(D)}\right) \rightarrow \hat{x} = \left(x^{(1)}, \ldots, x^{(D)}, 1\right)$$

Then the parameter w also gets an additional dimension

$$\hat{w} = \left(w^{(1)}, \ldots, w^{(D)}, w^{(D+1)}\right)$$

Demo: `t1()` shows a solution. You have `t1()` also in your code for playing with it!

1.7 VII. From Linear to Ridge regression

Overfitting: low training error, high test error. Reason: during learning one picks up too much of the noise in the training data, and learns weights that listen to noise signals.

One way to deal with it: avoiding weights w getting too large: add a penalty on the euclidean length of w

$$\text{argmin}_w \sum_{i=1}^{n} (x_i \cdot w - y_i)^2 + \lambda \|w\|^2$$

Ridge regression

Ridge regression is Linear regression with added penalty term on weights

$$\lambda \|w\|^2$$

λ is a hyperparameter in this approach. The solution changes to

$$w = (X^T \cdot X + \lambda I)^{-1} X^T \cdot Y$$

In practice, one needs to find a good value for the hyperparameter λ on a validation set, before measuring the performance on the test set. The effect of this regularization will be discussed later.

2 Linear model for classification (hingeloss)

2.1 I. Input and output space

Input and output space in classification problems?

2 classes: $\mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{0, 1\}$ or $\mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{-1, 1\}$

C classes: $\mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{0, \ldots, C-1\}$

2.2 II. the prediction mapping?

Consider the case of two classes. Linear function without/with a bias. Thresholded by a sign

$$\begin{aligned} f_w(x) = x \cdot w &= \sum_{d=1}^{D} x_d w_d && , \; w \in \mathbb{R}^{D \times 1} \\ g_{w,b}(x) = x \cdot w + b &= \sum_{d=1}^{D} x_d w_d + b && , \; w \in \mathbb{R}^{D \times 1}, b \in \mathbb{R}^1 \end{aligned}$$

$$h(x) = sign g_{w,b}(x)$$

2.3 III. Loss function for classification

First loss function for a pair (x, y):

0-1-loss

$$\ell(f(x), y) = 1[\text{sign}(g_{w,b}(x)) \neq y]$$

Problem: sign is unsuitable for gradient optimization. better:

hinge-loss

$$\ell(f(x), y) = \max(0, 1 - g_{w,b}(x)y)$$

The hinge-loss is an upper bound on the zero-one-loss. Therefore, if the upper bound gets minimized, then the 0-1-loss would get minimized as well.

2.4 IV. the goal in classification

Find parameters w, b which generalize well to new, unseen data points. Generalize well means here again the same as for regression:

generalization for classification

From an intuitive view, a mapping generalizes well, if it gives low prediction errors on unseen data samples.

Generalize well means in short form:

- we find parameters (w^*, b^*) defining a mapping f which for most draws of test datasets T_n produces predictions with low errors on them:

$$\begin{aligned}\hat{L}(f, T_n) &= \frac{1}{n} \sum_{(x_i,y_i) \in T_n} \ell(f(x_i), y_i) \\ &= \frac{1}{n} \sum_{i=1}^{n} 1[f(x_i) \neq y_i] \to \text{low}\end{aligned}$$

Take away I: generalization

Generalization is the same idea for many different setups: learn parameters on training data, so that the loss on newly drawn test datasets will be low on average – for most draws of such datasets from a data source.

The difference between the regression part and the classification part is only the choice of loss function.

Take away II

See also that the linear model is suitable for classification or regression – the loss function makes what the linear model (and any model in general, be it tree-classifiers or deep neural nets!) will predict after training.

2.5 V. a method to select a prediction mapping $g_{w,b}$ / or its parameters from a dataset

deferred to the gradient lecture !

2.6 VI. from classification with hingeloss to SVM

now add again a quadratic penalty on the weights when learning parameters over a training set

$$\text{argmin}_{(w,b)} \frac{1}{n} \sum_{(x_i,y_i)\in D_n} \max(0, 1 - g_{w,b}(x_i)y_i) + \lambda\|w\|^2$$

SVM:

- averaged hinge loss $\max(0, 1 - f(x_i)y_i)$
- with a linear/affine model $f(x_i) = w \cdot x_i + b$
- with quadratic penalty $\lambda\|w\|^2$ on the weights.

A different way to arrive at an SVM (remember $\frac{1}{\|w\|}$ is the margin to be maximized)