BLG 561 E FALL 2021

Deep Learning

12.10.2021

Gözde ÜNAL

Machine Learning:

Traditional ML.

Hard-crafted features → prior knowledge.
expert "

Raw data → Feature Extractor → Trainable Classifier → Output.

Input

Trainable

New ML/DL:

Input → Low-Level Feature Extractor → Mid Level Feature Extractor → High Level Feature Extractor → Trainable Classifier → output

Trainable Trainable Trainable

Deep Learning: Many layers → many many parameters
many

In Linear Algebra

$$\frac{\#\ \text{unknowns}}{\#\ \text{parameters}} \approx \le \#\ \text{of data points.}$$

Deep Learning: Over-parameterization!!!

$\#$ parameters $\gg$ $\#$ data points

Input

Text / Emails. → Input Data → Spam Classifier → output (spam / Not spam)

NLP

Vision

pictures

1 2 3

1 2

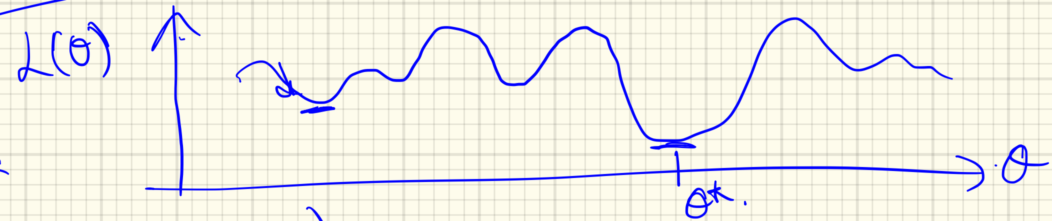
1 2

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

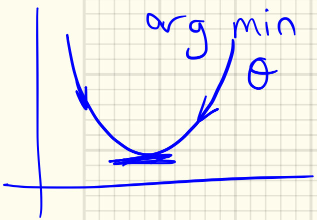
→ Conjectures:

Recall: To get a global extremum in an optimiz. → we need a convex obj fn

non-convex

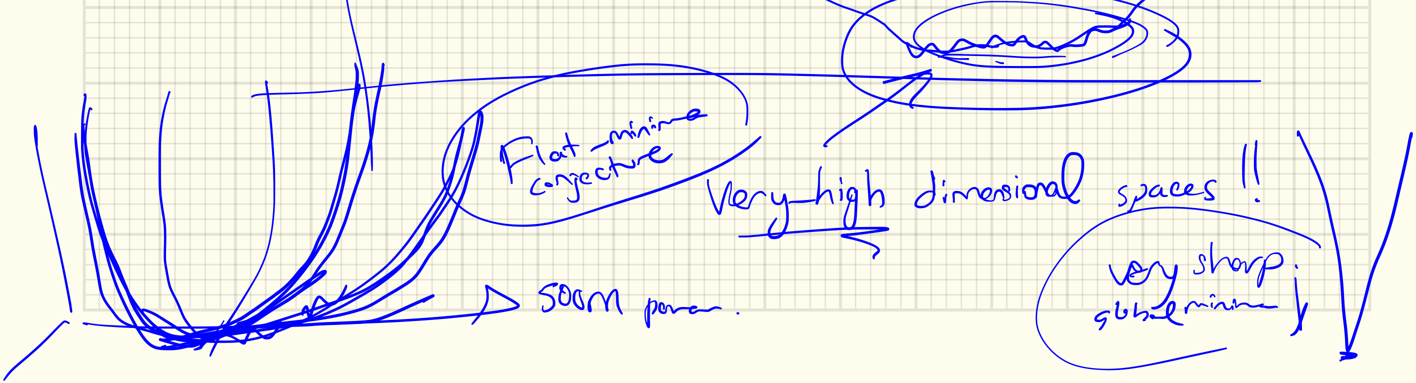



↓ convex



$\arg\min_{\theta} \mathcal{L}(\theta)$

↓ regularized SGD.



## ML Review:

**Supervised Learning:**

$X \longleftrightarrow Y$

$\{x^i\}_{i=1}^m \longleftrightarrow \{y^i\}_{i=1}^m$

| x | y |
|---|---|
| 1 | "1" |
| 1 | "1" |
| 1 | "1" |
| 2 | "2" |
| 0 | "0" |
| ⋮ | |

Example: Handwritten Digit Recognition

| | | | | | |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | ⋯ |
| ⋮ | | | | | |
| | | | 0 | 9 | |

$m$ : # input data samples in the Training Set

$\theta$ : parameters of the algorithm

$$\{x^i\}_{i=1}^m \longrightarrow \boxed{\text{ML algo}} \longrightarrow \hat{y}^i$$

$x_i \longleftrightarrow y^i$ : true label given. $\forall i$

$$\hat{y}^i = h_\theta(x^i) \text{ : hypothesis fn.}$$

$$h : X \longrightarrow Y$$

$$x^i \longrightarrow y^i$$

$$x^i \in X, \quad y^i \in Y$$

Loss function:

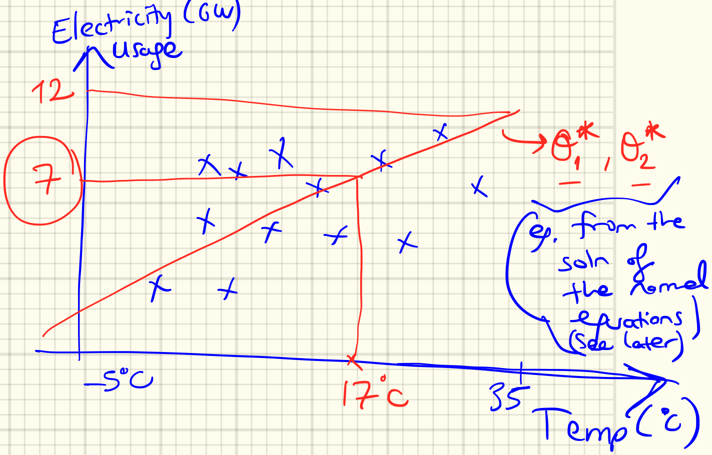
$$\ell(h_\theta(x), y)$$

We minimize this loss fn.: distance btw $h_\theta$ & $y$ to get $\theta$.

→ Linear Regression (Supervised ML).

Q. Predict electricity demand in istanbul tomorrow?

| Dates | Temperature (°c) | Used Electricity (GW) |
|---|---|---|
| 01.06.2017 | 20° | 8.05 |
| 02.06.2017 | 25° | 3.06 |
| ⋮ | | ⋮ |
| 01.06.2021 | 30° | |




ML Notation: Input features $x^{(i)} \in \mathbb{R}^n$, $i = 1, \dots, n$

$$\text{Electricity Demand} \approx \theta_1 \cdot \text{Temp}^{(i)} + \theta_2 \; ; \quad \text{eg. } \underline{x}^i = \begin{bmatrix} T^i \\ 1 \end{bmatrix}$$

Input features

Output $y^i \in \mathbb{R}$ → b/c Regression task

ML Algo: Framework

$\{x^i \in \mathbb{R}^n, y^i \in \mathbb{R}\}_{i=1}^m$ : training data (electricity usage & its predictors) from before.

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

Model Parameters : $\underline{\theta} \in \mathbb{R}^{n+1}$ : same size as the input features

Define the following:

1 Hypothesis Function:

eg. $\boxed{h_\theta(x) = \underline{\theta}^T \underline{x}} = \sum_{j=1}^{n} \theta_j x_j$

tells us how confident we are in the mapping.

$$h : \mathcal{X} \longrightarrow \mathcal{Y}$$

2 Loss Function: measures how "good" our hypothesis function is.

$$\ell(\overbrace{h_\theta(x)}^{\text{prediction}}, \overset{GT}{y}) \quad , \quad h_\theta(x^{(i)}) \approx y^{(i)} \longleftarrow \text{"Ground Truth" (GT)}$$

True response.

or Reference Labels

we should define this loss fn.

$$\longrightarrow \quad \ell : \mathcal{Y} \times \mathcal{Y} \longrightarrow \mathbb{R}^+$$

eg. here $\boxed{\ell : \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}^+}$ for the simple regression task here.

→Q. What is a common loss function for a regression task?

A: $\ell(h_\theta(x), y) = (h_\theta(x) - y)^2$ (MSE) loss.

All Machine Learning (ML) algorithms require the following.

We define the general ML (algorithm) problem:

Given $(x^i, y^i)\big|_{i=1}^{m}$, a set of input samples (#: m of them) & outputs

training set

Goal: Find the parameters that minimize the sum of the losses:

$$\arg\min_\theta \sum_{i=1}^{m} \underbrace{\ell(h_\theta(x^i), y^{(i)})}_{\ell^{(i)}}$$

To solve this, all ML algorithms should specify:

1. What is the hypothesis function?
2. What is the loss function?
3. How do you solve the given optimization problem?

→ $y \in \mathbb{R}$ in linear regression

$y \in \{0,1\}$ in binary classification.

$y \in \{1, \dots, k\}$ multi-class classification.

→ Ex: Linear Regression problem:

① $h_\theta(x) = \underline{\theta}^T \underline{x}$

② $\ell \left( h_\theta(x) - y \right)^2$ ; MSE.

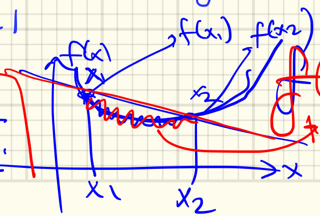
③ With 1 & 2 lead to ML optimization problem:

$$\arg\min_{\underline{\theta}} \sum_{i=}^{m} \ell\left(h_\theta(x^i), y^i\right) = \sum_{i=1}^{m} \left(\theta^T x^i - y^i\right)^2 = \mathcal{L}(\underline{\theta})$$ : overall loss fn.

→ Is $\mathcal{L}(\underline{\theta})$ a convex fn ?

Yes. here.

Recall: Convex fn.




$\alpha \in (0,1)$

$$f\left(\alpha x_1 + (1-\alpha) x_2\right) \leq \alpha . f(x_1) + (1-\alpha) f(x_2)$$

→ For a convex problem; $\exists$ many convex solvers; python e.g. cvxpy.

Generally: Calculate the gradient of our loss (objective) fn.

$$\Rightarrow \nabla_{\underline{\theta}} \Big( \underbrace{\sum_{i=1}^{m} (\theta^T x^i - y^i)^2}_{} \Big) = \sum_{i=1}^{m} \nabla_{\theta} (\underline{\theta}^T \underline{x}^i - y^i)^2$$

$$\nabla_{\theta} \mathcal{L}(\theta) = \sum_{i=1}^{m} 2(\underline{\theta}^T \underline{x}^i - y^i)\, x^i$$

Soln:

1 Iterative Optimization: For example; Gradient Descent (GD)

GD update rule: $\boxed{\theta^0}$ : initial values, $k=0$
k: iteration index (superscript)

$$\underline{\theta}^{k+1} \leftarrow \underline{\theta}^k - \alpha \underbrace{\nabla_{\underline{\theta}} \mathcal{L}(\underline{\theta})}_{}$$

$$\theta_1^k \leftarrow \theta_1^{k-1} - \alpha \nabla_{\theta_1} \mathcal{L}(\underline{\theta})$$
$$\theta_2^k \leftarrow \theta_2^{k-1} - \alpha \nabla_{\theta_2} \mathcal{L}(\underline{\theta})$$

2 For this ex; we get a Closed Form soln:

Introduce Vector matrix Notation

$$\underline{\underline{X}} = \begin{bmatrix} \underline{x}^{(1)T} \\ \underline{x}^{(2)T} \\ \vdots \\ \underline{x}^{(m)T} \end{bmatrix} \in \mathbb{R}^{m \times n}, \quad \underline{\theta} \in \mathbb{R}^{n}$$

matrix (=)

$$\underline{y} = \begin{bmatrix} y^1 \\ \vdots \\ y^m \end{bmatrix} \in \mathbb{R}^m$$

Rewrite the LS objective : $\sum_{i=1}^{m} (\theta^T x^i - y^i)^2 = \| \underline{\underline{X}}\, \underline{\theta} - \underline{y} \|_2^2$

scalar

$$\mathcal{L}(\underline{\theta}) = (\underline{\underline{X}}\,\underline{\theta} - \underline{y})^T (\underline{\underline{X}}\,\underline{\theta} - \underline{y})$$

$$\mathcal{L}(\underline{\theta}) = \underline{\theta}^T \underline{\underline{X}}^T \underline{\underline{X}}\, \underline{\theta} - 2\underline{\theta}^T \underline{\underline{X}}^T \underline{y} + \underline{y}^T \underline{y}$$

take the derivative w.r.t. $\theta$

$$\nabla_{\underline{\theta}} \mathcal{L}(\underline{\theta}) = 2\underline{\underline{X}}^T \underline{\underline{X}}\, \underline{\theta} - 2\underline{\underline{X}}^T \underline{y} \rightarrow \text{set this to } \underline{0}$$

$$\nabla_{\underline{\theta}} \mathcal{L}(\underline{\theta}) = 0 \rightarrow \underbrace{\underset{n\times m}{\underline{\underline{X}}^T}\, \underset{m\times n}{\underline{\underline{X}}}}_{n\times n \text{ matrix}}\, \underline{\theta} = \underline{\underline{X}}^T \underline{y}$$

$$\rightarrow \boxed{\underline{\theta}^* = (\underline{\underline{X}}^T \underline{\underline{X}})^{-1} \underline{\underline{X}}^T \underline{y}}$$

"NORMAL EQUATIONS"

a closed form solution for the minimization of sum of squared losses

e.g. the electricity demand prediction problem, we solve the normal equations, we get $\theta_1^*, \theta_2^*$ $\rightarrow$ these define the Red Line in the plot (from before)

A new ML problem:

Q: Alternative loss function? Absolute Loss function ("MAE: Mean Absolute Error")

(1) Linear hypothesis ✓

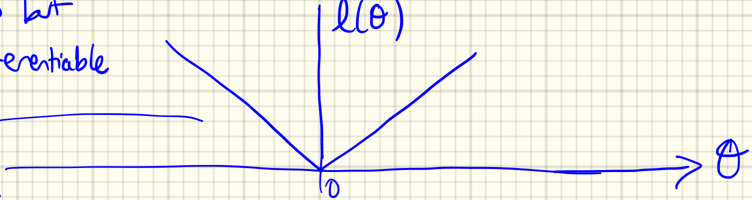
(2) Loss fn. defined ✓

$$\mathcal{L}(\underline{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \underbrace{\ell\left(h_\theta(x^i) - y^i\right)}_{\ell^i:\ \text{loss fn. for 1 sample}} = \frac{1}{m}\sum_{i=1}^{m} \left|h_\theta(x^i) - y^i\right| = \|h_\theta(x) - y\|_1$$

$$\begin{pmatrix} h_\theta(x^1) - y^1 \\ h_\theta(x^2) - y^2 \\ \vdots \\ h_\theta(x^m) - y^m \end{pmatrix}$$

$L^1$-norm

continuous but not differentiable




Note: No closed form solution exists!

We can use (sub)gradients

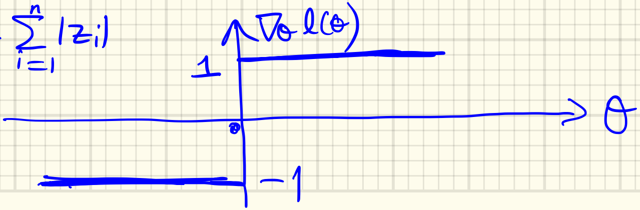
$$\nabla_\theta \|h_\theta(x) - y\|_1 = \underline{X}^T \cdot \text{sign}\big(\underbrace{\underline{X}\,\underline{\theta} - \underline{y}}_{\text{is a vector}}\big)$$

$$\|\underline{X}\,\underline{\theta} - \underline{y}\|_1$$

Now, we have the gradient of the loss fn.

$L_1$-norm:

Note: $\|\underline{z}\|_1 = \sum_{i=1}^{n} |z_i|$




(3) GD:

$$\underline{\theta} \leftarrow \underline{\theta} - \alpha\left(\nabla_{\underline{\theta}} \mathcal{L}(\underline{\theta})\right)$$

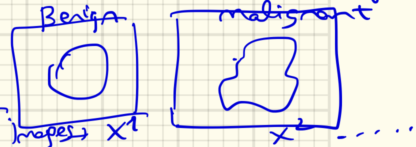
$\alpha$ LR: learning rate.

Q: Define the ML algorithm for the **Binary Classification**: $y \in \{+1, -1\}$

e.g. Spam Classification. / NLP

e.g. Benign / malignant tissue classification

Tissue (breast) images histology




$$\underline{x}^i = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \\ 1 \end{bmatrix} \begin{array}{l} \rightarrow \text{mean area} \\ \rightarrow \text{concavity} \\ \\ \end{array}$$

Choose

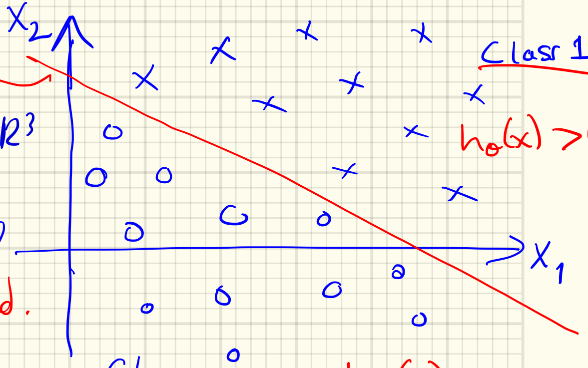
① Linear Hypothesis:

$$h_\theta(x) = \underline{\theta}^T \underline{x}.$$

**Prediction:**

$$\hat{y} = \text{sign}\left(h_\theta(x)\right)$$

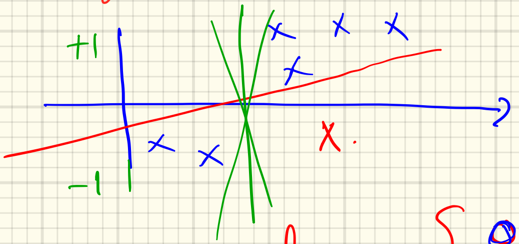
$h_\theta(x) = 0$ ∴ the line is a separating hyperplane (e.g. a line here) the two classes.




Model parameters are: $\underline{\theta} \in \mathbb{R}^{n+1}$, here $\underline{\theta} \in \mathbb{R}^3$

② Loss function ?

What if we use a LS loss? ✗ not desired.




→ **0/1 Loss**:

$$l_{0/1} = \begin{cases} 0, & \text{if } \overbrace{\text{sign}(h_\theta(x))}^{\hat{y}} = y^{\leftarrow GT.} \\ 1, & \text{o/w} \end{cases} \quad \mathbb{1}\{y \cdot \underbrace{h_\theta(x)}_{\theta^T x} \leq 0\} \implies$$

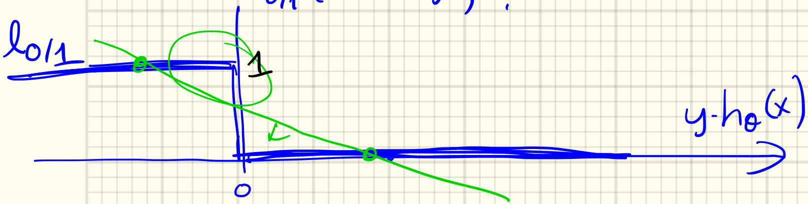
indicator fn.

$\rightarrow \mathbb{1}\{ y \cdot h_\theta(x) \leq 0 \} = \ell(\theta) = \begin{cases} 1, & y \cdot h_\theta(x) \leq 0 \leftarrow \\ 0, & y \cdot h_\theta(x) > 0 \leftarrow \end{cases}$

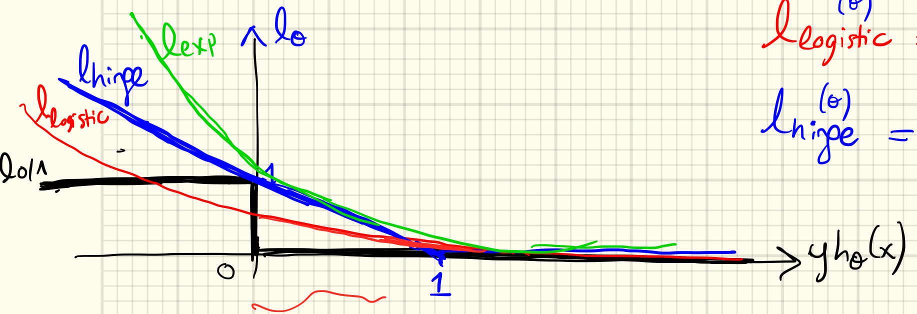
when GT & $h_\theta$ have different signs → we incur a loss.

$\ell_{0/1}(h_\theta(x), y)$ : Is this a convex loss fn.? No!

Not convex, harder to optimize




∴ → Alternative loss functions are used in classification.




$$\ell^{(\theta)}_{logistic} = \log(1 + \exp(-y \cdot h_\theta(x)))$$

$$\ell^{(\theta)}_{hinge} = \max(\underbrace{1 - y \cdot h_\theta(x)}_{\geq 0 \text{ when } y \cdot h_\theta(x) \leq 1}, 0)$$

$$\ell^{(\theta)}_{exp} = \exp(-y \cdot h_\theta(x))$$

$\ell_{logistic}$, $\ell_{hinge}$, $\ell_{exp}$ : convex losses ∴ we typically prefer them over 0/1 loss.

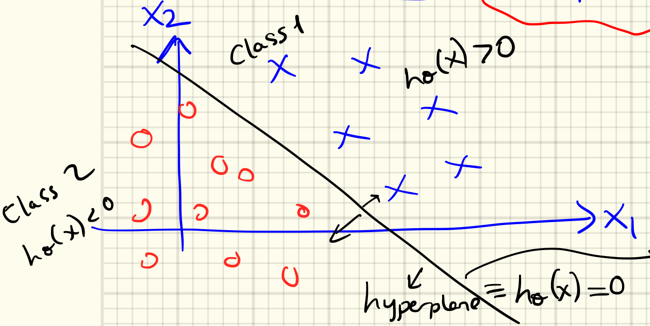
∃ convex optimization tools (eg. cvxpy) to solve such convex optim. problems.

→ Recall: $\boxed{\text{SVM}}$ : Linear SVM solves the ML optimization problem w/ Hinge loss & linear hypothesis function.

$$\underline{\theta}^* = \arg\min_{\underline{\theta}} \underbrace{\sum_{i=1}^{m} \max\{1 - y^i h_\theta(x^i), 0\}}_{l_{data}\ :\ \text{data term.}} + \underbrace{\lambda \cdot \|\underline{\theta}\|_2^2}_{\text{add a Regularizer term}}$$

Unknowns: $\underline{\theta}$

$\lambda$ : a regularization parameter.




→ SVM Classification boundary

$\boxed{\text{Logistic Regression}}$ Using linear hypothesis fn & logistic loss:

ML problem:

$$\underline{\theta}^* = \arg\min_{\underline{\theta}} \sum_{i=1}^{m} \log\left(1 + \exp(-y^i \underline{\theta}^T \underline{x}^i)\right) + \lambda \|\theta\|_2^2$$

Loss fn. is convex & smooth (Note: hinge loss was not differentiable at $y \cdot h_\theta = 1$)

$\Longrightarrow$

$$\Rightarrow \mathcal{L}_{\text{logistic}}(\theta) \equiv \max_{\theta} \sum \log \frac{1}{1+e^{-(y^i \theta^T x^i)}} \rightarrow$$

we see logistic or sigmoid fn. in this loss.

③ Q: Devise an optimization based on GD to solve logistic regression.

## Multi_Class Classification:

Output is in $\{1, ..., k\}$

e.g. Digit classification $\{0, 1, 2, ..., 9\}$

10-class problem

Approach 1: One vs All method:

Build $k$ different binary classifiers: $h_{\theta_{i=1:k}}$

Goal: Predict class $i$ vs. others

$$\hat{y} = \arg\max_{i} h_{\theta_i}(x)$$

$\hookrightarrow$ each binary classifier model.

Approach 2 : Next time .