ECE 1513: Introduction to Machine Learning

Lecture 4: Linear Regression and Classification

Ali Bereyhi

ali.bereyhi@utoronto.ca

Department of Electrical and Computer Engineering

University of Toronto

Winter 2025

Quick Recap: *ML General Recipe*

We defined ML as

> *the set of data-driven approaches that help us understand the environment and its behavior, and generalize it!*

Any learning task is accomplished by ML through *three major steps*

- Collect data
- Specify a model *that captures the pattern*
- Develop a learning algorithm

Quick Recap: *Unsupervised vs Supervised Learning*

In Unsupervised Learning, samples are unlabeled

- *Data* $\rightsquigarrow$ *Collection of samples* $\mathbb{D} = \{\boldsymbol{x}_n : n = 1, \ldots, N\}$
- *Model* $\rightsquigarrow$ *Captures a pattern observed in data, e.g., fitting into clusters*
- *Learning algorithm* $\rightsquigarrow$ *It takes* $\mathbb{D}$ *and returns a* good *model*

In Supervised Learning, samples are labeled

- *Data* $\rightsquigarrow$ *Collection of samples* $\mathbb{D} = \{(\boldsymbol{x}_n, \boldsymbol{v}_n) : n = 1, \ldots, N\}$
- *Model* $\rightsquigarrow$ *Captures relation between data samples and their* labels
- *Learning algorithm* $\rightsquigarrow$ *It takes* $\mathbb{D}$ *and returns a* good *model*

Quick Recap: *Unsupervised Learning*

We studied three major unsupervised learning tasks

- *Clustering*
  - ↳ *Data*
  - ↳ *Model: K-centroid*
  - ↳ *Learning algorithm: K-means clustering*
- *Distribution Learning*
  - ↳ *Data*
  - ↳ *Model: a distribution with unknowns*
  - ↳ *Learning algorithm: maximum likelihood*
- *Dimensionality Reduction*
  - ↳ *Data*
  - ↳ *Model: projection*
  - ↳ *Learning algorithm: PCA*

Today's Agenda: *Supervised Learning via Linear Models*

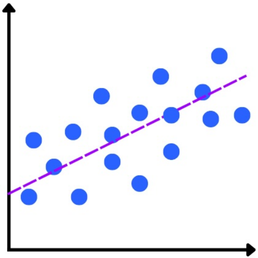
In today's lecture, we start with supervised learning and look into

linear models

through the following steps

- *Formulating supervised learning*
- *Linear Regression*
  - ↳ *We review some notions in functional analysis*
- *Linear Classification*
  - ↳ *Logistic Regression*

Supervised Learning: *Regression*



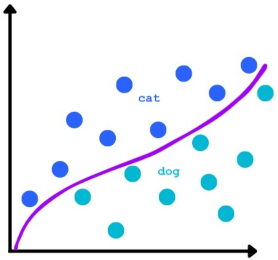
We see

$$\mathbb{D} = \{(\boldsymbol{x}_n, \boldsymbol{v}_n) : n = 1, \ldots, N\}$$

and look for

$$\boldsymbol{v}_n = f(\boldsymbol{x}_n)$$

Supervised Learning: *Classification*



We see

$$\mathbb{D} = \{(\boldsymbol{x}_n, \text{class}_n) : n = 1, \ldots, N\}$$

and look for

$$\text{class}_n = f(\boldsymbol{x}_n)$$

Supervised Learning: *Generic Formulation*

We again have three components

- *labeled dataset*

$$\mathbb{D} = \{(\boldsymbol{x}_n \in \mathbb{X}, \boldsymbol{v}_n \in \mathbb{V}) : n = 1, \ldots, N\}$$

- *Model that relates data samples and their labels*

$$f : \mathbb{X} \mapsto \mathbb{V}$$

↳ *We assume $f \in \mathbb{H}$ with $\mathbb{H}$ being the hypothesis*
↳ ***Example:*** *$\mathbb{H}$ contains all linear function, i.e.,*

$$\mathbb{H} = \{f(x) = wx \text{ for all } w \in \mathbb{R}\}$$

- *Learning algorithm finds optimal model within hypothesis set*

$$\mathcal{A} : \mathbb{D} \mapsto f^{\star} \in \mathbb{H}$$

Supervised Learning: *Generic Formulation*

In regression labels are continuous predictions

Example

(x_n, v_n) *represent weight and height of people:* $\mathbb{V} = [54, 272] \subset \mathbb{R}$

In classification labels are distinct classes

$$\mathbb{V} = \{\text{class}_1, \ldots, \text{class}_K\}$$

for some integer number of classes K

Example

(x_n, v_n) *represent weight and type of pets:* $\mathbb{V} = \{\textit{cat}, \textit{dog}\}$

Supervised Learning: *Learning Algorithm*

? *How can we find the optimal model?*

Let for a sample $(\boldsymbol{x}, \boldsymbol{v})$

$$\boldsymbol{y} = f(\boldsymbol{y})$$

Then, the loss function determines how $\boldsymbol{y}$ and $\boldsymbol{v}$ are different

Loss Function

Loss function computes the difference between model output and true label

$$\mathcal{L}(\boldsymbol{y}, \boldsymbol{v}) \in \mathbb{R}$$

Example: *Euclidean distance between* $\boldsymbol{y}$ *and* $\boldsymbol{v}$, *i.e.,*

$$\mathcal{L}(\boldsymbol{y}, \boldsymbol{v}) = \|\boldsymbol{y} - \boldsymbol{v}\|^2$$

Supervised Learning: *Risk Minimization*

? *How can we find the optimal model?*

! *We search for minimal expected loss*

Risk $\equiv$ Expected Loss

Risk of a model is defined as the expectation of loss w.r.t. data distribution

$$\mathbb{E}_{\boldsymbol{x},\boldsymbol{v}}\left\{\mathcal{L}\left(\boldsymbol{y},\boldsymbol{v}\right)\right\} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{v}}\left\{\mathcal{L}\left(f\left(\boldsymbol{x}\right),\boldsymbol{v}\right)\right\} = R\left(f\right)$$

Optimal model is the one which minimizes the risk

$$f^{\star} = \operatorname*{argmin}_{f \in \mathbb{H}} R\left(f\right)$$

Supervised Learning: *Empirical Risk*

- **?** *We don't know data distribution in general!*
- **!** *We approximate it with the arithmetic average $\equiv$ law of large numbers*

Empirical Risk $\equiv$ Estimate of Risk

Say $\boldsymbol{y}_n = f(\boldsymbol{x}_n)$ for every $(\boldsymbol{x}_n, \boldsymbol{v}_n) \in \mathbb{D}$; then, the empirical risk is

$$\hat{R}(f) = \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}(\boldsymbol{y}_n, \boldsymbol{v}_n)$$

If N is very large and samples are i.i.d.

$$\hat{R}(f) = \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}(\boldsymbol{y}_n, \boldsymbol{v}_n) \approx \mathbb{E}_{\boldsymbol{x},\boldsymbol{v}} \{\mathcal{L}(\boldsymbol{y}, \boldsymbol{v})\} = R(f)$$

Supervised Learning: *General Learning Algorithm*

? *How can we find the optimal model?*

! *We search for minimal empirical risk*

Optimal model is the one which minimizes the empirical risk

$$\begin{aligned} f^\star &= \operatorname*{argmin}_{f \in \mathbb{H}} \hat{R}(f) \\ &= \operatorname*{argmin}_{f \in \mathbb{H}} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}(\boldsymbol{y}_n, \boldsymbol{v}_n) \end{aligned}$$

Regression Example: *Fitting Polynomial*

- *Labeled dataset*

$$\mathbb{D} = \{(\textit{weight}, \textit{price}) = (4, 2), (5, 1.5), (4.8, 3), (6, 4)\}$$

- *Model is a function taking weight and returning price*

$$\mathbb{H} = \left\{ f(x) = w_0 + w_1 x + \ldots + w_P x^P : w_0, \ldots, w_P \in \mathbb{R} \right\}$$

↳ *Hypothesis: v and x are related via a polynomial function*

$$v \approx y = f(x) = \sum_{i=1}^{P} w_i x^i$$

- *Learning algorithm $\equiv$ empirical risk minimization*

Regression Example: *Fitting Polynomial*

Optimal model is the one which minimizes the empirical risk

$$f^\star = \operatorname*{argmin}_{f \in \mathbb{H}} \sum_{n=1}^{N} \frac{1}{N} \mathcal{L}\left(f\left(x_n\right), v_n\right)$$

Say $\mathcal{L}\left(y, v\right) = \left(y - v\right)^2$: *the optimal model is*

$$f^\star\left(x\right) = \sum_{i=1}^{P} w_i^\star x^i$$

for $w_0^\star, \dots, w_P^\star$ *that are*

$$w_0^\star, \dots, w_P^\star = \operatorname*{argmin}_{w_0, \dots, w_P} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}\left(w_0 + w_1 x_n + \dots + w_P x_n^P - v_n\right)^2$$

Regression Example: *Fitting Line*

Let's restrict the hypothesis to a line

$$f(x) = wx$$

The empirical risk computed on

$$\mathbb{D} = \{(\text{weight}, \text{price}) = (4, 2), (5, 1.5), (4.8, 3), (6, 4)\}$$

is written as

$$\hat{R}(f) = \frac{1}{4}\left[(4w - 2)^2 + (5w - 1.5)^2 + (4.8w - 3)^2 + (6w - 4)^2\right]$$

So, we have

$$w^\star = \underset{w}{\operatorname{argmin}}\, \hat{R}(f) \rightsquigarrow f^\star(x) = w^\star x = 0.53x$$

Regression Example: *Fitting Line*

A Nice Thought Practice

Assume you did know that

$$(x, v) \sim P(x, v)$$

Think about the exact risk and how you could determine it!

Recall that

$$R(f) = \mathbb{E}_{x,v}\left\{(wx - v)^2\right\}$$

Classification Example: *Separation by Wight*

- *Labeled dataset*

$$\mathbb{D} = \{(\textit{weight}, \textit{type of pet}) = (5.5, \textit{cat}), (4.5, \textit{cat}), (12.5, \textit{dog}), (9.5, \textit{dog}), (7.5, \textit{cat})\}$$

- *Model is a function taking weight and returning type*

$$\mathbb{H} = \{f(x) = \operatorname{sign}(x - b) \in \{-1, 1\}\} \rightsquigarrow y = \begin{cases} \textit{cat} & f(x) = -1 \\ \textit{dog} & f(x) = 1 \end{cases}$$

↳ *Hypothesis: cat and dog are separated in weight by thresholding*

- *Learning algorithm* $\equiv$ *empirical risk minimization*

Classification Example: *Separation by Wight*

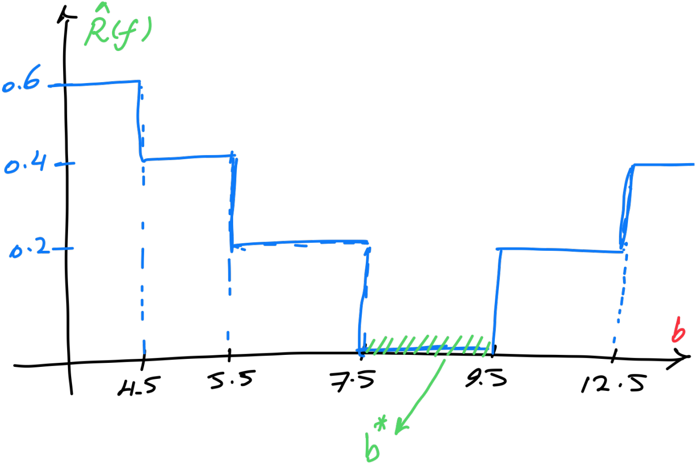
Optimal model is the one which minimizes the empirical risk

$$f^\star = \operatorname*{argmin}_{f \in \mathbb{H}} \sum_{n=1}^{N} \frac{1}{N} \mathcal{L}\left(f\left(x_n\right), v_n\right)$$

Say the loss is

$$\mathcal{L}\left(y, v\right) = \begin{cases} 1 & y \neq v \\ 0 & y = v \end{cases}$$

Classification Example: *Separation by Wight*



Classification Example: *Separation by Wight*

A Nice Thought Practice

Assume you did know that

$$(x, v) \sim P(x, v)$$

Think about the exact risk and how you could determine it!

Note that in this case

$$R(f) = \mathbb{E}_{x,v}\left\{\mathcal{L}\left(\mathrm{sign}(x-b), v\right)\right\} = \Pr\left\{\mathrm{sign}(x-b) \neq v\right\}$$

Regression via Linear Models

Let's think of

$$\mathbb{D} = \left\{ \left(\boldsymbol{x}_n \in \mathbb{R}^d, v_n \in \mathbb{R} \right) : n = 1, \ldots, N \right\}$$

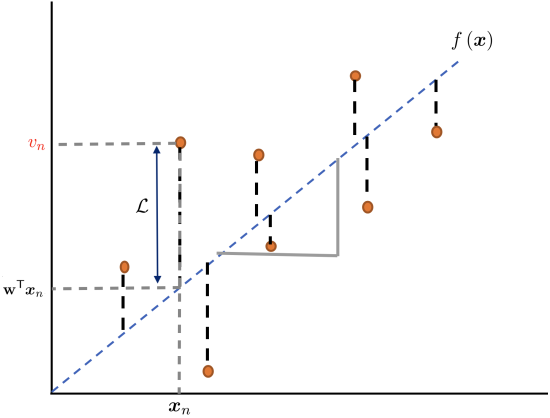
We focus on *linear models*

$$\mathbb{H} = \left\{ f\left(\boldsymbol{x}\right) = \mathbf{w}^{\mathsf{T}} \boldsymbol{x} \text{ for all } \mathbf{w} \in \mathbb{R}^d \right\}$$

Optimal linear model is

$$\mathbf{w}^{\star} = \operatorname*{argmin}_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}\left(\mathbf{w}^{\mathsf{T}} \boldsymbol{x}_n, v_n \right)$$

Regression via Linear Models: *Visualization*



Linear Regression: *Empirical Risk*

Typical choice of the loss function

$$\mathcal{L}\left(y_n, v_n\right) = \left(y_n - v_n\right)^2$$

So the empirical risk is

$$\hat{R}\left(\mathbf{w}\right) = \frac{1}{N} \sum_{n=1}^{N} \left(\mathbf{w}^{\mathsf{T}} \boldsymbol{x}_n - v_n\right)^2$$

Attention

All vectors in these slides are column vectors. We use transpose to make them row vectors: column $\equiv \boldsymbol{x}_n \rightsquigarrow \boldsymbol{x}_n^{\mathsf{T}} \equiv$ *row*

Linear Regression: *Vectorized Empirical Risk*

We can collect the dataset into a matrix

$$\mathbf{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_N]$$

So we have

$$\begin{aligned}\mathbf{w}^{\mathsf{T}}\mathbf{X} &= \mathbf{w}^{\mathsf{T}} [\boldsymbol{x}_1, \dots, \boldsymbol{x}_N] \\ &= \left[\mathbf{w}^{\mathsf{T}}\boldsymbol{x}_1, \dots, \mathbf{w}^{\mathsf{T}}\boldsymbol{x}_N\right] \\ &= [y_1, \dots, y_N] = \boldsymbol{y}^{\mathsf{T}}\end{aligned}$$

So, we have

$$\begin{aligned}\hat{R}(\mathbf{w}) &= \frac{1}{N}\|\boldsymbol{y} - \boldsymbol{v}\|^2 \\ &= \frac{1}{N}\|\mathbf{X}^{\mathsf{T}}\mathbf{w} - \boldsymbol{v}\|^2\end{aligned}$$

Linear Regression: *Empirical Risk Minimization*

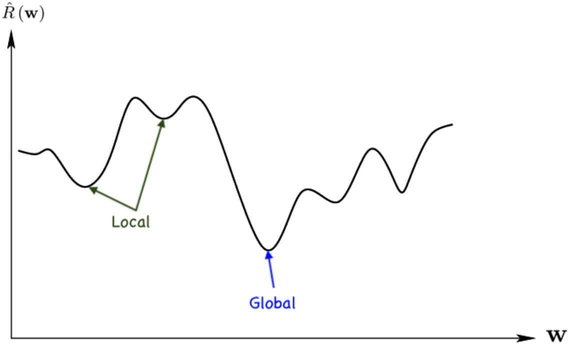
Optimal linear model is the $\mathbf{w}$ that minimized the empirical risk

$$\mathbf{w}^\star = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N} \|\mathbf{X}^\mathsf{T}\mathbf{w} - \boldsymbol{v}\|^2$$

? *How to optimize it?*

Global and Local Minimum

Empirical risk can have local and global minima



Note!

In this figure, we think of $\mathbf{w}$ to be a scalar!

Stationary Points

- ? *How can we find those points?*
- ! *They are again stationary points!*

Stationary Points

The points at which derivative of the function is zero

$$\frac{\mathrm{d}\hat{R}}{\mathrm{d}\mathbf{w}} = 0$$

Stationary point is where the slope is zero

- *Minimum*
- *Maximum*
- *Inflection*

Multivariate Functions: *Gradient*

? *But we have a multivariate function!*

$$\nabla \hat{R} = \begin{bmatrix} \frac{\partial \hat{R}}{\partial w_1} \\ \vdots \\ \frac{\partial \hat{R}}{\partial w_d} \end{bmatrix}$$

Stationary Points

At stationary points the gradient of the function is vector of zero $\nabla \hat{R} = \mathbf{0}$

- *Minimum*
- *Maximum*
- *Saddle Point*

Stationary Points

? *How can we find the global minimum?*

Naive Algorithm $\equiv$ *Exhaustive Search*

1. *Find all stationary points*
2. *Select those that are minimum*
3. *Take the smallest one*

! *Not really easy to find stationary points with a very complex function*
   ↳ *We will see this in next lectures!*

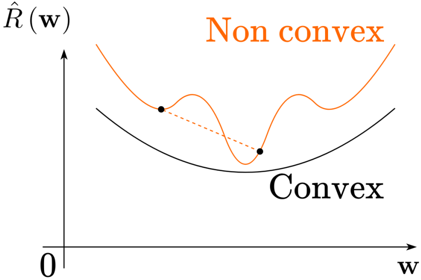
! *The number of such points can grow exponentially with dimension*
   ↳ *This is usually computationally infeasible!*

Convex Optimization

In linear regression, we are lucky since

$$\hat{R}(\mathbf{w}) = \frac{1}{N} \|\mathbf{X}^\mathsf{T} \mathbf{w} - \boldsymbol{v}\|^2$$

is convex: *it has only one minimum which is both local and global*



Linear Regression: *Empirical Risk Minimization*

Since we know that it's convex, we can find the optimal model by

$$\nabla \hat{R} = \frac{1}{N} \nabla \|\mathbf{X}^\mathsf{T} \mathbf{w} - \boldsymbol{v}\|^2 = \mathbf{0}$$

With a bit of computation, we can see

$$\nabla \|\mathbf{X}^\mathsf{T} \mathbf{w} - \boldsymbol{v}\|^2 = 2 \left(\mathbf{X}\mathbf{X}^\mathsf{T} \mathbf{w} - \mathbf{X}\boldsymbol{v} \right)$$

So, we should find

$$\mathbf{X}\mathbf{X}^\mathsf{T} \mathbf{w}^\star = \mathbf{X}\boldsymbol{v}$$

Note!

Replace in the empirical risk and you see it makes sense in special cases!

Linear Regression: *Empirical Risk Minimization*

This is a linear system of d equations with d unknowns

$$\mathbf{X}\mathbf{X}^\mathsf{T}\mathbf{w}^\star = \mathbf{X}\boldsymbol{v}$$

- *We can solve it if equations are linearly independent, i.e.,* $\det \mathbf{X}\mathbf{X}^\mathsf{T} \neq 0$
  - ↳ *Usually the case when* $N \geqslant d$
- *If multiple equations are linearly dependent: we have no unique solution*
  - ↳ *Always the case when* $N < d$

When $\det \mathbf{X}\mathbf{X}^\mathsf{T} \neq 0$, we can write

$$\mathbf{w}^\star = \left(\mathbf{X}\mathbf{X}^\mathsf{T}\right)^{-1}\mathbf{X}\boldsymbol{v} = \mathbf{X}^\dagger\boldsymbol{v}$$

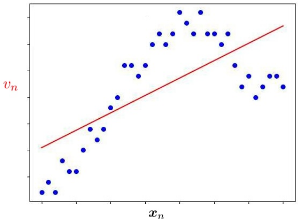
$\mathbf{X}^\dagger$ is the pseudo-inverse of $\mathbf{X}$

Linear Regression: *Optimal Model*

So, the optimal model is given by

$$f^\star(\boldsymbol{x}) = \underbrace{\boldsymbol{x}^\mathsf{T}}_{\text{new sample out of } \mathbb{D}} \quad \underbrace{\mathbf{X}^\dagger \boldsymbol{v}}_{\text{computed from } \mathbb{D}}$$

- *We can check its generalization by testing it on a test set $\mathbb{T}$*
  - ↳ *More about it later*
- *It's only a linear model and is not guaranteed to generalize well*
  - ↳ *If data shows exteme nonlinearity, it doesn't work well!*



Linear Regression with *Affine Models*

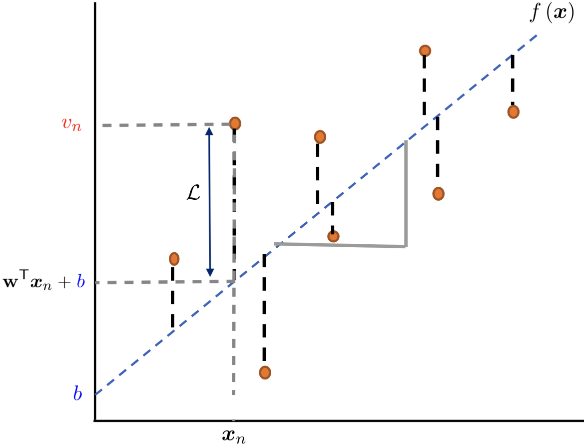
We may also include a bias in the linear model

$$\mathbb{H} = \left\{ f\left(\boldsymbol{x}\right) = \mathbf{w}^{\mathsf{T}}\boldsymbol{x} + b \text{ for all } \mathbf{w} \in \mathbb{R}^{d} \text{ and } b \in \mathbb{R} \right\}$$

Optimal linear model is

$$\mathbf{w}^{\star}, b^{\star} = \underset{\mathbf{w}\in\mathbb{R}^{d}, b\in\mathbb{R}}{\operatorname{argmin}} \frac{1}{N} \sum_{n=1}^{N} \left(\mathbf{w}^{\mathsf{T}}\boldsymbol{x}_{n} + b - v_{n} \right)^{2}$$

Regression via Affine: *Visualization*



Linear Regression with *Affine Models*

We can interpret it as a linear model again

$$\mathbf{w}^\mathsf{T} \boldsymbol{x} + b = \begin{bmatrix} b & \mathbf{w}^\mathsf{T} \end{bmatrix} \begin{bmatrix} 1 \\ \boldsymbol{x} \end{bmatrix}$$

So, we can make a new dataset matrix as

$$\mathbf{X} = \begin{bmatrix} 1 & \dots & 1 \\ \boldsymbol{x}_1 & \dots & \boldsymbol{x}_N \end{bmatrix}$$

and everything goes as before

$$\begin{bmatrix} b^\star \\ \mathbf{w}^\star \end{bmatrix} = \left(\mathbf{X}\mathbf{X}^\mathsf{T} \right)^{-1} \mathbf{X}\boldsymbol{v} = \mathbf{X}^\dagger \boldsymbol{v}$$

Linear Regression with *Vector Labels*

We could also have

$$\mathbb{D} = \left\{ \left(\boldsymbol{x}_n \in \mathbb{R}^d, \boldsymbol{v}_n \in \mathbb{R}^\ell \right) : n = 1, \ldots, N \right\}$$

We could extend everything to higher dimensions

$$\mathbb{H} = \left\{ f\left(\boldsymbol{x}\right) = \mathbf{W}^\mathsf{T} \boldsymbol{x} + \mathbf{b} \text{ for all } \mathbf{W} \in \mathbb{R}^{d \times \ell} \text{ and } \mathbf{b} \in \mathbb{R}^\ell \right\}$$

Everything in this case is again as in scalar case

A Nice Practice

Write the optimal model in this case. You just need to adjust dimensions!

Complexity of Computing Optimal Model

Finding optimal model can be computationally expensive with bid datasets

- *Finding* $\mathbf{X}\mathbf{X}^{\mathsf{T}}$ *needs* $\mathcal{O}\left(d^2 N\right)$ *computations*
- *Finding* $\left(\mathbf{X}\mathbf{X}^{\mathsf{T}}\right)^{-1}$ *needs between* $\mathcal{O}\left(d^{2.4}\right)$ *and* $\mathcal{O}\left(d^3\right)$ *computations*
- *Finding* $\left(\mathbf{X}\mathbf{X}^{\mathsf{T}}\right)^{-1}\mathbf{X}$ *needs* $\mathcal{O}\left(d^2 N\right)$ *computations*
- *We need at least in order of* d *data samples*

So we have around $\mathcal{O}\left(d^3\right)$ complexity!

Not Good with Large-dimensional Data!

If data is very big, i.e., d *extremely large, it's not a good approach!*

Further Read

- Bishop
  ↳ Chapter 3: *Sections 3.1 – 3.3* ***Linear Regression***
- ESL
  ↳ Chapter 3: *Section 3.1 – 3.2* ***Linear Regression***
- Goodfellow
  ↳ Chapter 5: *Section 5.7* ***Supervised Learning***

Binary Classification via Linear Models

Let's think of

$$\mathbb{D} = \left\{ \left(\boldsymbol{x}_n \in \mathbb{R}^d, v_n \in \{0, 1\} \right) : n = 1, \ldots, N \right\}$$

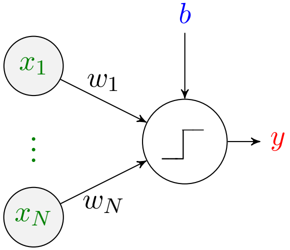
We focus on *linear models*

$$\mathbb{H} = \left\{ f\left(\boldsymbol{x}\right) = \begin{cases} 1 & \mathbf{w}^{\mathsf{T}} \boldsymbol{x} \geqslant 0 \\ 0 & \mathbf{w}^{\mathsf{T}} \boldsymbol{x} < 0 \end{cases} \text{for all } \mathbf{w} \in \mathbb{R}^d \right\}$$

Optimal linear model is

$$\mathbf{w}^\star = \underset{\mathbf{w} \in \mathbb{R}^d}{\operatorname{argmin}} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}\left(\mathbf{w}^{\mathsf{T}} \boldsymbol{x}_n, v_n \right)$$

Linear Classification: *Visualization*



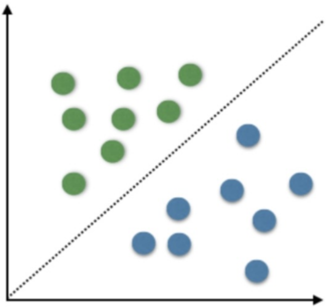
The model is a linear transform followed by a decision-making operation

$$f(\boldsymbol{x}) = \begin{cases} 1 & \mathbf{w}^{\mathsf{T}}\boldsymbol{x} \geqslant 0 \\ 0 & \mathbf{w}^{\mathsf{T}}\boldsymbol{x} < 0 \end{cases} = s\left(\mathbf{w}^{\mathsf{T}}\boldsymbol{x}\right)$$

Linear Classification: *Visualization*

Assume data is in two dimensions

$$f(\boldsymbol{x}) = \begin{cases} 1 & w_1 x_1 + w_2 x_2 \geqslant 0 \\ 0 & w_1 x_1 + w_2 x_2 < 0 \end{cases}$$



Linear Classification: *Empirical Risk*

Typical choice of the loss function

$$\mathcal{L}(y_n, v_n) = \mathbf{1}\{y_n \neq v_n\}$$

So the empirical risk is

$$\begin{aligned}\hat{R}(\mathbf{w}) &= \frac{1}{N}\sum_{n=1}^{N} \mathbf{1}\{y_n \neq v_n\} \\ &= \textit{Error Rate}\end{aligned}$$

Linear Classification: *Empirical Risk Minimization*

Optimal linear model is

$$\mathbf{w}^\star = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N}\sum_{n=1}^{N} \mathbf{1}\left\{ s\left(\mathbf{w}^\mathsf{T}\boldsymbol{x}\right) \neq v_n \right\}$$

This is extremely *non-smooth*!

Note

We cannot compute gradient! So, we cannot follow the approach in linear regression

Old Solution: *Perceptron Algorithm*

1: Start with $\mathbf{w} = \mathbf{0}$ or some small random initial value
2: **while** $\hat{R}(\mathbf{w}) \neq 0$ **do**
3: **for** $i = 1 : I$ **do**
4: Compute $z_i = \mathbf{w}^\mathsf{T} \boldsymbol{x}_i$ and $\hat{y}_i = s(z_i)$ `# pass through perceptron`
5: **if** $\hat{y}_i \neq y_i$ **then**
6: $\mathbf{w} \leftarrow \mathbf{w} - \mathrm{sign}(z_i)\, \boldsymbol{x}_i$
7: **end if**
8: **end for**
9: **end while**

Convergence

You can show that this algorithm converges, if

a line separating the two classes exists

Alternative Solution: *Treating as Regression*

Another solution had been to treat it as a regression!

$$\hat{R}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} \left(\mathbf{w}^{\mathsf{T}} \boldsymbol{x}_n - \tilde{v}_n \right)^2$$

- If $v_n = 1$: set $\tilde{v}_n = 1$
  - Then, we have $\mathbf{w}^{\mathsf{T}} \boldsymbol{x}_n \approx 1$

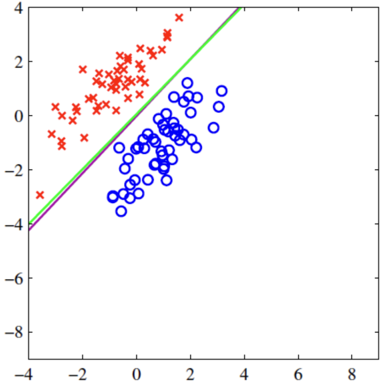
$$f(\boldsymbol{x}) = s\left(\mathbf{w}^{\mathsf{T}} \boldsymbol{x}\right) = 1$$

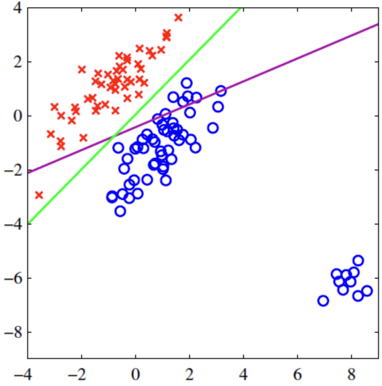
- If $v_n = 0$: set $\tilde{v}_n = -1$
  - Then, we have $\mathbf{w}^{\mathsf{T}} \boldsymbol{x}_n \approx -1$

$$f(\boldsymbol{x}) = s\left(\mathbf{w}^{\mathsf{T}} \boldsymbol{x}\right) = 0$$

Alternative Solution: *Treating as Regression*

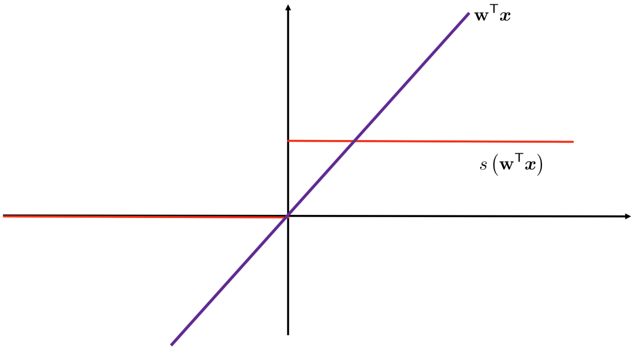
The drawback is though that it is not robust to outliers



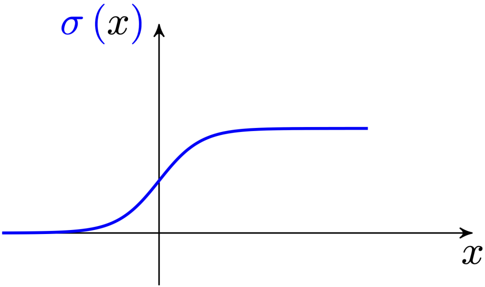


Better Solution: *Thresholding by Sigmoid*

? *Can we do anything in between?*



Better Solution: *Thresholding by Sigmoid*



We can use the sigmoid function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Classification by Sigmoid: *Empirical Risk Minimization*

The empirical risk is then

$$\hat{R}(\mathbf{w}) \frac{1}{N} \sum_{n=1}^{N} \left(\sigma \left(\mathbf{w}^{\mathsf{T}} \boldsymbol{x} \right) - v_n \right)^2$$

We can now compute gradient, and look for

$$\nabla \hat{R} = \mathbf{0}$$

Key Issue

We cannot analytically find it in linear regression!

Thresholding by Sigmoid: *A Call for Alternative Viewpoint*

Say, we found optimal model

$$\mathbf{w}^\star = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} \frac{1}{N}\sum_{n=1}^{N}\left(\sigma\left(\mathbf{w}^\mathsf{T}\boldsymbol{x}\right) - v_n\right)^2$$

- ? *How can we use this model on new data?*

For inference, we compute

$$y = \sigma\left(\boldsymbol{x}^\mathsf{T}\mathbf{w}^\star\right) \rightsquigarrow \begin{cases} \hat{v} = 1 & g \geqslant 0.5 \\ \hat{v} = 0 & g < 0.5 \end{cases}$$

Soft Output

Our model does not compute label. It computes its probability!

Further Read

- Bishop
  - ↳ Chapter 4: *Sections 4.1 – 4.2* ***Linear Classification***
- ESL
  - ↳ Chapter 4: *Section 4.1 – 4.2* ***Linear Classification***