

Convex Optimization and Applications

11 - Application to Data Science

Guillaume Sagnol



Framework

Data: A training set consisting of

- observations or labels $y_1, \dots, y_m \in \mathbb{R}$
- feature vectors $x_1, \dots, x_m \in \mathbb{R}^n$

Assumption: The features $x_i = (x_{i,1}, \dots, x_{i,n})$ carry information that *explain* the observed value y_i .

Ultimate goal: Find a *model* which

- explains the relationship between the x_i 's and the y_i 's.
- can map a new, unseen feature x_* to some *reasonable prediction* of the label y_* .

Framework

Data: A training set consisting of

- observations or labels $y_1, \dots, y_m \in \mathbb{R}$
- feature vectors $x_1, \dots, x_m \in \mathbb{R}^n$

Assumption: The features $x_i = (x_{i,1}, \dots, x_{i,n})$ carry information that *explain* the observed value y_i .

Ultimate goal: Find a *model* which

- explains the relationship between the x_i 's and the y_i 's.
- can map a new, unseen feature x_* to some *reasonable prediction* of the label y_* .

Disclaimer: This lecture is not a course in machine learning; it only focuses on convex optimization problems arising in this field. For more background, see e.g. [Shalev-Schwartz, S. & Ben-David, S., Understanding Machine Learning: From Theory to Algorithms (2014)], (**Book available online**).

Outline

- 1 Linear regression
- 2 Beyond Least squares
- 3 Classification problems
- 4 The Kernel Trick
- 5 Design of Experiments

Linear Regression

Assumed model: $\exists \boldsymbol{\theta} \in \mathbb{R}^n : y_i \simeq \mathbf{x}_i^T \boldsymbol{\theta}, \forall i \in [m]$.

In vector notation,

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\epsilon},$$

where $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^T \in \mathbb{R}^{m \times n}$.

Linear Regression

Assumed model: $\exists \boldsymbol{\theta} \in \mathbb{R}^n : y_i \simeq \mathbf{x}_i^T \boldsymbol{\theta}, \forall i \in [m]$.

In vector notation,

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\epsilon},$$

where $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^T \in \mathbb{R}^{m \times n}$.

Non-linear embedding

- A linear model can deal with non-linear relationships: After change of variable $\mathbf{x}'_i \leftarrow \phi(\mathbf{x}_i)$, we can learn a relationship of the form

$$y_i \simeq \phi(\mathbf{x}_i)^T \boldsymbol{\theta}.$$

Linear Regression

Assumed model: $\exists \boldsymbol{\theta} \in \mathbb{R}^n : y_i \simeq \mathbf{x}_i^T \boldsymbol{\theta}, \forall i \in [m]$.

In vector notation,

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\epsilon},$$

where $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^T \in \mathbb{R}^{m \times n}$.

Non-linear embedding

- A linear model can deal with non-linear relationships: After change of variable $\mathbf{x}'_i \leftarrow \phi(\mathbf{x}_i)$, we can learn a relationship of the form

$$y_i \simeq \phi(\mathbf{x}_i)^T \boldsymbol{\theta}.$$

- For an affine model $y_i \simeq \theta_0 + \boldsymbol{\theta}^T \mathbf{x}_i$:

$$\mathbf{x}'_i \leftarrow [1, \mathbf{x}_i^T]^T \in \mathbb{R}^{n+1}.$$

Linear Regression

Assumed model: $\exists \boldsymbol{\theta} \in \mathbb{R}^n : y_i \simeq \mathbf{x}_i^T \boldsymbol{\theta}, \forall i \in [m]$.

In vector notation,

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\epsilon},$$

where $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^T \in \mathbb{R}^{m \times n}$.

Non-linear embedding

- A linear model can deal with non-linear relationships: After change of variable $\mathbf{x}'_i \leftarrow \phi(\mathbf{x}_i)$, we can learn a relationship of the form

$$y_i \simeq \phi(\mathbf{x}_i)^T \boldsymbol{\theta}.$$

- For an affine model $y_i \simeq \theta_0 + \boldsymbol{\theta}^T \mathbf{x}_i$:

$$\mathbf{x}'_i \leftarrow [1, \mathbf{x}_i^T]^T \in \mathbb{R}^{n+1}.$$

- For a quadratic model $y_i \simeq \theta_0 + \boldsymbol{\theta}^T \mathbf{x} + \mathbf{x}^T \Theta \mathbf{x}$:

$$\mathbf{x}'_i \leftarrow [1, \mathbf{x}_i^T, (\mathbf{x}_{ik}\mathbf{x}_{i\ell})_{1 \leq k \leq \ell \leq n}]^T \in \mathbb{R}^{1+n+n(n+1)/2}.$$

Least-squares estimator

Choose θ so as to minimize $\|\epsilon\|^2 = \epsilon_1^2 + \dots + \epsilon_m^2$, i.e.,

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \|X\theta - y\|^2$$

If X has *full-column* rank, then $X^T X$ is invertible, and the least-squares problem has a unique minimizer, called (*ordinary*) *least squares estimate* of θ :

$$\hat{\theta}_{LS} = (X^T X)^{-1} X^T y.$$

Proof

$$\begin{aligned} f(\theta) &= \|X\theta - y\|^2 = \theta^T X^T X \theta - 2y^T X \theta + y^T y \\ \nabla f(\theta) &= 2(XX^T \theta - X^T y) \end{aligned}$$

$\hat{\theta}_{LS}$ is found by setting $\nabla f(\theta) = \mathbf{0}$.

Least-squares estimator

Choose θ so as to minimize $\|\epsilon\|^2 = \epsilon_1^2 + \dots + \epsilon_m^2$, i.e.,

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \|X\theta - y\|^2$$

But why do we minimize $\|\epsilon\|^2$, and not another function of ϵ ?

Least-squares estimator

Choose θ so as to minimize $\|\epsilon\|^2 = \epsilon_1^2 + \dots + \epsilon_m^2$, i.e.,

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \|X\theta - y\|^2$$

But why do we minimize $\|\epsilon\|^2$, and not another function of ϵ ?

Reasons

- The penalty $\epsilon \mapsto \|\epsilon\|^2$ yields a simple analytical formula...

Least-squares estimator

Choose θ so as to minimize $\|\epsilon\|^2 = \epsilon_1^2 + \dots + \epsilon_m^2$, i.e.,

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \|X\theta - y\|^2$$

But why do we minimize $\|\epsilon\|^2$, and not another function of ϵ ?

Reasons

- The penalty $\epsilon \mapsto \|\epsilon\|^2$ yields a simple analytical formula...
- If we assume that the errors are normally distributed, $\hat{\theta}_{LS}$ coincides with the *maximum likelihood estimator* for θ

Least-squares estimator

Choose θ so as to minimize $\|\epsilon\|^2 = \epsilon_1^2 + \dots + \epsilon_m^2$, i.e.,

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \|X\theta - y\|^2$$

But why do we minimize $\|\epsilon\|^2$, and not another function of ϵ ?

Reasons

- The penalty $\epsilon \mapsto \|\epsilon\|^2$ yields a simple analytical formula...
- If we assume that the errors are normally distributed, $\hat{\theta}_{LS}$ coincides with the *maximum likelihood estimator* for θ
- Gauss-Markov theorem: $\hat{\theta}_{LS}$ is a BLUE (best linear unbiased estimator) !

Maximum Likelihood estimation

Assume the ϵ'_i 's are i.i.d. (independently and identically distributed) normal random variables: $\epsilon_i \stackrel{iid}{\sim} \mathcal{N}(0, \sigma^2)$.

Then, $y_i = \mathbf{x}_i^T \boldsymbol{\theta} + \epsilon_i$ is the realization of a random variable

$$Y_i \sim \mathcal{N}(\mathbf{x}_i^T \boldsymbol{\theta}, \sigma^2).$$

The probability density of (Y_1, \dots, Y_m) at (y_1, \dots, y_m) is a function of the unknown $\boldsymbol{\theta}$, called *likelihood*:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta} | \mathbf{y}, \mathbf{x}_1, \dots, \mathbf{x}_m) &= \prod_{i=1}^m f_{Y_i}(y_i) \\ &= \prod_{i=1}^m C \cdot \exp\left(-\frac{1}{2\sigma^2}(y_i - \mathbf{x}_i^T \boldsymbol{\theta})^2\right) \end{aligned}$$

Taking the log, we see that maximizing the likelihood is equivalent to minimizing $\sum_i (y_i - \mathbf{x}_i^T \boldsymbol{\theta})^2 \implies \hat{\boldsymbol{\theta}}_{MLE} = \hat{\boldsymbol{\theta}}_{LS}$

Gauss-Markov theorem

More generally, we still assume that the ϵ_i 's are i.i.d. (but not necessarily normal), with $\mathbb{E}[\epsilon_i] = 0$, $\mathbb{V}[\epsilon_i] = \sigma^2$. As on previous slide, y is a realization of the random variable $Y = X\theta + \epsilon$.

Gauss-Markov theorem

More generally, we still assume that the ϵ_i 's are i.i.d. (but not necessarily normal), with $\mathbb{E}[\epsilon_i] = 0$, $\mathbb{V}[\epsilon_i] = \sigma^2$. As on previous slide, y is a realization of the random variable $Y = X\theta + \epsilon$.

Consider a *linear estimator* $\hat{\theta} = LY$ for θ , where $L \in \mathbb{R}^{n \times m}$. Note that $\hat{\theta}$ is also a random variable, such that

$$\mathbb{E}[\hat{\theta}] = L\mathbb{E}[Y] = LX\theta, \quad \mathbb{V}[\hat{\theta}] = L\mathbb{V}[Y]L^T = \sigma^2 LL^T.$$

Gauss-Markov theorem

More generally, we still assume that the ϵ_i 's are i.i.d. (but not necessarily normal), with $\mathbb{E}[\epsilon_i] = 0$, $\mathbb{V}[\epsilon_i] = \sigma^2$. As on previous slide, y is a realization of the random variable $Y = X\theta + \epsilon$.

Consider a *linear estimator* $\hat{\theta} = LY$ for θ , where $L \in \mathbb{R}^{n \times m}$. Note that $\hat{\theta}$ is also a random variable, such that

$$\mathbb{E}[\hat{\theta}] = L\mathbb{E}[Y] = LX\theta, \quad \mathbb{V}[\hat{\theta}] = L\mathbb{V}[Y]L^T = \sigma^2 LL^T.$$

The estimator $\hat{\theta}$ is called *unbiased* iff

$$\mathbb{E}[\hat{\theta}] = \theta, \quad \forall \theta \in \mathbb{R}^n \quad \iff \quad LX = I$$

Gauss-Markov theorem

More generally, we still assume that the ϵ_i 's are i.i.d. (but not necessarily normal), with $\mathbb{E}[\epsilon_i] = 0$, $\mathbb{V}[\epsilon_i] = \sigma^2$. As on previous slide, y is a realization of the random variable $Y = X\theta + \epsilon$.

Consider a *linear estimator* $\hat{\theta} = LY$ for θ , where $L \in \mathbb{R}^{n \times m}$. Note that $\hat{\theta}$ is also a random variable, such that

$$\mathbb{E}[\hat{\theta}] = L\mathbb{E}[Y] = LX\theta, \quad \mathbb{V}[\hat{\theta}] = L\mathbb{V}[Y]L^T = \sigma^2 LL^T.$$

The estimator $\hat{\theta}$ is called *unbiased* iff

$$\mathbb{E}[\hat{\theta}] = \theta, \quad \forall \theta \in \mathbb{R}^n \quad \iff \quad LX = I$$

Theorem (Gauss-Markov).

In the class of all linear unbiased estimators, $\hat{\theta}_{LS}$ has the “ \preceq -minimum” variance-covariance matrix.

$$(\hat{\theta} = LY, LX = I) \implies \mathbb{V}[\hat{\theta}] = \sigma^2 LL^T \succeq \mathbb{V}[\hat{\theta}_{LS}] = \sigma^2 (X^T X)^{-1}$$

Gauss-Markov Theorem

Theorem (Gauss-Markov).

In the class of all linear unbiased estimators, $\hat{\theta}_{LS}$ has the “ \preceq -minimum” variance-covariance matrix.

$$(\hat{\theta} = LY, LX = I) \implies \mathbb{V}[\hat{\theta}] = \sigma^2 LL^T \succeq \mathbb{V}[\hat{\theta}_{LS}] = \sigma^2 (X^T X)^{-1}$$

Proof

1. We first check that $\hat{\theta}_{LS}$ is a linear unbiased estimate:

$$\hat{\theta}_{LS} = (X^T X)^{-1} X^T Y$$

$$\mathbb{E}[\hat{\theta}_{LS}] = (X^T X)^{-1} X^T \mathbb{E}[Y] = (X^T X)^{-1} X^T X \theta = \theta.$$

Its variance is

$$\mathbb{V}[\hat{\theta}_{LS}] = (X^T X)^{-1} X^T \underbrace{\mathbb{V}[Y]}_{=\sigma^2 I} X (X^T X)^{-1} = \sigma^2 (X^T X)^{-1}.$$

Gauss-Markov Theorem

Theorem (Gauss-Markov).

In the class of all linear unbiased estimators, $\hat{\theta}_{LS}$ has the “ \preceq -minimum” variance-covariance matrix.

$$(\hat{\theta} = LY, LX = I) \implies \mathbb{V}[\hat{\theta}] = \sigma^2 LL^T \succeq \mathbb{V}[\hat{\theta}_{LS}] = \sigma^2 (X^T X)^{-1}$$

Proof

2. To show that $\hat{\theta}_{LS}$ is the BLUE, we consider another unbiased linear estimator for θ : $\hat{\theta} = LY, LX = I$.

$$\text{Then, } \begin{pmatrix} X^T X & I_n \\ I_n & LL^T \end{pmatrix} = \begin{pmatrix} X^T \\ L \end{pmatrix} \begin{pmatrix} X^T \\ L \end{pmatrix}^T \succeq 0.$$

With our assumption that X has full-column rank, $X^T X \succ 0$, so we can use the Schur-complement lemma:

$$LL^T \succeq (X^T X)^{-1}.$$

Outline

- 1 Linear regression
- 2 Beyond Least squares**
- 3 Classification problems
- 4 The Kernel Trick
- 5 Design of Experiments

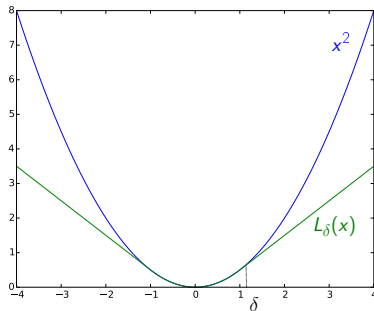
Huber regression

Minimizing $\|\epsilon\|^2$ is not always a good idea, because the assumption that the ϵ_i 's are i.i.d. is not always realistic.

In particular, real-data often contains *outliers* (erroneous data). To cope with this issue, it has been proposed to minimize $\sum_i L_\delta(\epsilon_i)$ instead of $\sum \epsilon_i^2$, where

$$L_\delta(x) := \begin{cases} x^2 & \text{for } |x| \leq \delta \\ \delta(2|x| - \delta), & \text{otherwise} \end{cases}$$

gives less weight to *large deviations*.



Huber regression

Proposition

The Huber loss $x \mapsto L_\delta(x)$ is convex and SOC-representable:

$$L_\delta(x) \leq t \iff \exists a, b \in \mathbb{R} : \begin{cases} a^2 + 2\delta|b| \leq t \\ a + b = x. \end{cases}$$

$$\iff \exists a, b, u \in \mathbb{R} : \begin{cases} \left\| \begin{bmatrix} 2a \\ u - 1 \end{bmatrix} \right\| \leq u + 1 \\ 2\delta|b| \leq t - u \\ a + b = x \end{cases}$$

Consequence: The huber regression problem,

minimize $\sum_{i=1}^m L_\delta(x_i^T \theta - y_i)$, can be formulated as an SOCP.

Ridge regression

The matrix $X^T X$ can be nearly singular/badly conditioned \implies small perturbations in some x_{ik} can cause large variations of $\hat{\theta}_{LS}$.

To fix this issue, in ridge-regression, we add a penalty term

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \|X\theta - y\|^2 + \lambda \|\theta\|^2.$$

This can be recast as a standard least-squares problem:

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \left\| \begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix} \theta - \begin{bmatrix} y \\ \mathbf{0} \end{bmatrix} \right\|^2,$$

with a better-conditioned information matrix:

$$\begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix} = X^T X + \lambda I$$

Ridge regression

Denote by $\hat{\theta}_{\text{ridge}}^{\lambda}$ the ridge-regression estimator with penalty parameter $\lambda \geq 0$, which solves

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \|X\theta - \mathbf{y}\|^2 + \lambda \|\theta\|^2.$$

We have: $\hat{\theta}_{\text{ridge}}^{\lambda} = (X^T X + \lambda I)^{-1} X^T \mathbf{y}$, and $\hat{\theta}_{\text{ridge}}^0 = \hat{\theta}_{LS}$.

Unlike $\hat{\theta}_{LS}$, the estimator $\hat{\theta}_{\text{ridge}}^{\lambda}$ is *biased* for $\lambda > 0$, but it has a smaller variance:

$$\lambda \geq \lambda' \geq 0 \implies \mathbb{V}[\hat{\theta}_{\text{ridge}}^{\lambda}] \preceq \mathbb{V}[\hat{\theta}_{\text{ridge}}^{\lambda'}] \preceq \mathbb{V}[\hat{\theta}_{LS}],$$

where

$$\mathbb{V}[\hat{\theta}_{\text{ridge}}^{\lambda}] := \sigma^2 (X^T X + \lambda I)^{-1} X^T X (X^T X + \lambda I)^{-1}.$$

Lasso regression

When the number of features is very large, we want to find a *sparse model*

$$y_i \simeq \sum_{k \in I} \theta_k x_{ik},$$

where I is a small subset of $[n]$.

Lasso regression

When the number of features is very large, we want to find a *sparse model*

$$y_i \simeq \sum_{k \in I} \theta_k x_{ik},$$

where I is a small subset of $[n]$.

One way to do so would be to minimize the *nonconvex* function

$$\|X\boldsymbol{\theta} - \mathbf{y}\|^2 + \lambda \|\boldsymbol{\theta}\|_0,$$

where the “*0-norm*” counts the number of nonzero entries:

$$\|\boldsymbol{\theta}\|_0 := |\{i \in [n] : \theta_i \neq 0\}|.$$

Lasso regression

When the number of features is very large, we want to find a *sparse model*

$$y_i \simeq \sum_{k \in I} \theta_k x_{ik},$$

where I is a small subset of $[n]$.

One way to do so would be to minimize the *nonconvex* function

$$\|X\boldsymbol{\theta} - \mathbf{y}\|^2 + \lambda \|\boldsymbol{\theta}\|_0,$$

where the “ 0 -norm” counts the number of nonzero entries:

$$\|\boldsymbol{\theta}\|_0 := |\{i \in [n] : \theta_i \neq 0\}|.$$

The Lasso regression replaces $\|\cdot\|_0$ by the ℓ_1 -norm, which is, in some sense, the best convex approximation of the problem:

$$\underset{\boldsymbol{\theta} \in \mathbb{R}^n}{\text{minimize}} \quad \|X\boldsymbol{\theta} - \mathbf{y}\|^2 + \lambda \|\boldsymbol{\theta}\|_1.$$

Outline

- 1 Linear regression
- 2 Beyond Least squares
- 3 Classification problems**
- 4 The Kernel Trick
- 5 Design of Experiments

Logistic regression

- Assume that the labels are binary, $y_i \in \{0, 1\}$.
- Indicates membership of samples in one of two classes.

In logistic regression, we fit a probabilistic model of the form

$$\mathbb{P}[y_i = 0] = \frac{1}{1 + e^{\mathbf{x}_i^T \boldsymbol{\theta}}}, \quad \mathbb{P}[y_i = 1] = \frac{e^{\mathbf{x}_i^T \boldsymbol{\theta}}}{1 + e^{\mathbf{x}_i^T \boldsymbol{\theta}}}.$$

by maximum-likelihood.

Under the assumed model, the likelihood of $\boldsymbol{\theta} \in \mathbb{R}^n$ reads

$$\mathcal{L}(\boldsymbol{\theta} | X, \mathbf{y}) = \prod_{\substack{i \in [m] \\ y_i = 1}} \mathbb{P}[y_i = 1] \cdot \prod_{\substack{i \in [m] \\ y_i = 0}} \mathbb{P}[y_i = 0] = \prod_{i=1}^m \frac{\exp(y_i \cdot \mathbf{x}_i^T \boldsymbol{\theta})}{1 + \exp(\mathbf{x}_i^T \boldsymbol{\theta})}.$$

Logistic regression

$$\mathcal{L}(\theta|X, y) = \prod_{\substack{i \in [m] \\ y_i = 1}} \mathbb{P}[y_i = 1] \cdot \prod_{\substack{i \in [m] \\ y_i = 0}} \mathbb{P}[y_i = 0] = \prod_{i=1}^m \frac{\exp(y_i \cdot \mathbf{x}_i^T \theta)}{1 + \exp(\mathbf{x}_i^T \theta)}.$$

Taking the log, the maximum-likelihood problem can be rewritten as

$$\underset{\theta \in \mathbb{R}^n}{\text{maximize}} \sum_{i=1}^m y_i \mathbf{x}_i^T \theta - \log(1 + \exp(\mathbf{x}_i^T \theta)),$$

which is a convex optimization problem involving the log-sum-exp function:

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \sum_{i=1}^m \log(e^0 + e^{\mathbf{x}_i^T \theta}) - y_i \mathbf{x}_i^T \theta.$$

\implies We can reformulate this problem as a conic program over K_{exp} .

Support vector machines

- Logistic regression provides a *probabilistic classifier*.
- Another approach is to use a deterministic classifier:

$$f_{\theta} : \mathbb{R}^n \rightarrow \{-1, 1\}.$$

(we rename the label “0” to “-1” to simplify the notation)

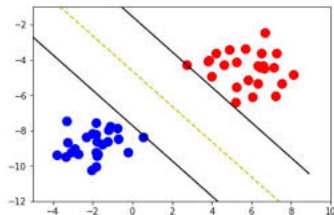
- One of the simplest approaches is to search for a hyperplane separating the samples with label $y_i = -1$ on one side, and the samples labelled $y_i = 1$ on the other side.
- Formally, such a separating hyperplane can be described by $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that

$$y_i(\mathbf{w}^T \mathbf{x}_i - b) > 0, \quad \forall i \in [m].$$

- If we set $\theta = (\mathbf{w}, b) \in \mathbb{R}^n \times \mathbb{R}$, this means that we search a classifier of the form $f_{\theta}(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} - b)$.

Support vector machines

A (hard-margin) *support vector machine* asks to find the separating hyperplane with the widest possible margin.



Of course, it is not always possible to separate the data linearly (i.e., to find (\mathbf{w}, b) s.t. $y_i(\mathbf{w}^T \mathbf{x}_i - b) > 0, \forall i \in [m]$).

Theorem

The data $X \in \mathbb{R}^{m \times n}$, $\mathbf{y} \in \{-1, 1\}^m$ is linearly separable iff the sets $\text{conv} \{x_i : y_i = -1\}$ and $\text{conv} \{x_j : y_j = 1\}$ don't intersect.

Hard margin separation

If the data is linearly separable, we can search for $\mathbf{w} \in \mathbb{R}^n$, $b \in \mathbb{R}$ such that

$$\begin{cases} \mathbf{w}^T \mathbf{x}_i < b - 1 & \text{if } y_i = -1 \\ \mathbf{w}^T \mathbf{x}_i > b + 1 & \text{if } y_i = 1. \end{cases}$$

The width of the stripe between the two hyperplanes $\{\mathbf{x} : \mathbf{w}^T \mathbf{x} \leq b - 1\}$ and $\{\mathbf{x} : \mathbf{w}^T \mathbf{x} \geq b + 1\}$ is $\delta = \frac{2}{\|\mathbf{w}\|}$.

The hard-margin SVM is hence an SOCP:

$$\begin{aligned} & \underset{\mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}}{\text{minimize}} && \|\mathbf{w}\| \\ & \text{s.t.} && y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1, \quad \forall i \in [m]. \end{aligned}$$

Soft-margin separation

- In practice, the data is rarely linearly separable, so the previous problem may be infeasible.
- To cope with this issue, we replace the hard constraint " $y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1$ " by a penalty for points that are on the wrong side of the margin:

$$\begin{aligned}\phi_{\mathbf{w},b}(\mathbf{x}_i, y_i) &= \begin{cases} 0 & \text{if } y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1; \\ 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b) & \text{otherwise.} \end{cases} \\ &= \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b)).\end{aligned}$$

- The soft margin SVM minimizes a linear combination of the two objectives: $\|\mathbf{w}\|^2$ (so the margin is wide) and $\sum_i \phi_{\mathbf{w},b}(\mathbf{x}_i, y_i)$ (so the \mathbf{x}_i 's lie on the correct side):

$$\underset{\mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}}{\text{minimize}} \quad \sum_{i=1}^m \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b)) + \lambda \|\mathbf{w}\|^2$$

Soft-margin separation

- The soft margin SVM minimizes a linear combination of the two objectives: $\|\mathbf{w}\|^2$ (so the margin is wide) and $\sum_i \phi_{\mathbf{w},b}(\mathbf{x}_i, y_i)$ (so the \mathbf{x}_i 's lie on the correct side):

$$\underset{\mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}}{\text{minimize}} \quad \sum_{i=1}^m \max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b)) + \lambda \|\mathbf{w}\|^2$$

- This problem can be reformulated as a QP, or as the following SOCP:

$$\begin{aligned} \underset{\mathbf{w}, \zeta \in \mathbb{R}^n, b, t \in \mathbb{R}}{\text{minimize}} \quad & \sum_{i=1}^m \zeta_i + \lambda t \\ \text{s.t.} \quad & y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1 - \zeta_i, \quad \forall i \in [m] \\ & \zeta_i \geq 0, \quad \forall i \in [m]. \\ & \|\mathbf{w}\|^2 \leq t. \end{aligned}$$

Outline

- 1 Linear regression
- 2 Beyond Least squares
- 3 Classification problems
- 4 The Kernel Trick**
- 5 Design of Experiments

Nonlinear embedding

- In all problems seen so far, we minimize a function $F_{X,y}(\theta)$ where the dependence in $X = [x_1, \dots, x_m]^T$ only occurs through scalar products $x_i^T \theta$:

$$F(\theta) = \sum_i f(x_i^T \theta, y_i) + g(\theta)$$

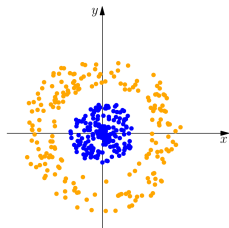
- For ridge regression, $f(u, y) = (u - y)^2$, $g(\theta) = \lambda \|\theta\|^2$.
- For lasso regression, $f(u, y) = (u - y)^2$, $g(\theta) = \lambda \|\theta\|_1$.
- For logistic regression, $f(u, y) = yu - \log(1 + e^u)$, $g = 0$.
- For soft-margin SVM, with $x'_i = [x_i^T, 1]^T$, $\theta = [w^T, -b]^T$:

$$f(u, y) = \max(0, 1 - yu), \quad g(\theta) = \lambda \|w\|^2.$$

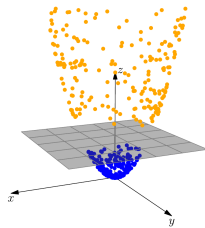
- If we replace x_i by $\varphi(x_i)$ for a non-linear map φ , we can *learn* a more efficient predictor.

Nonlinear embedding

Example:



Points x_i in original feature space.



Points $x'_i = \varphi(x_i)$ in augmented feature space.

- The original data $(x_i, y_i)_{i=1, \dots, m}$ is clearly not separable
- However, the augmented data $(\varphi(x_i), y_i)_{i=1, \dots, m}$ is perfectly separable for

$$\varphi(\mathbf{x}) := [x_1, x_2, x_1^2 + x_2^2]^T.$$

Nonlinear embedding

- If we replace x_i by $\varphi(x_i)$ for a non-linear map φ , we can *learn* a more efficient predictor.
- But this raises two issues:
 - How to select the map φ ?
 - If $\varphi : \mathbb{R}^n \rightarrow \mathbb{R}^{n'}$ maps \mathbf{x} to an augmented feature vector of very high dimension ($n' \gg n$), the learning problem might become prohibitively hard.
- In some situations, the *kernel trick* can overcome these issues.

Positive semidefinite kernel

Definition (Positive semidefinite kernel).

A function $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is called a *positive semidefinite kernel*, if for all $N \in \mathbb{N}$ and for all $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathbb{R}^n$, the matrix

$$K[\mathbf{x}_1, \dots, \mathbf{x}_N] := \{K(\mathbf{x}_i, \mathbf{x}_j)\}_{1 \leq i, j \leq N}$$

is symmetric positive semidefinite.

Intuition: An “infinite” matrix, with entries indexed by $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n \times \mathbb{R}^n$.

■ Counterpart of “ $X \succeq 0 \iff \mathbf{u}^T X \mathbf{u} \geq 0, \forall \mathbf{u}$ ”:

K positive semidefinite kernel

$$\iff \int_{\mathbf{x} \in \mathbb{R}^n} \int_{\mathbf{y} \in \mathbb{R}^n} K(\mathbf{x}, \mathbf{y}) f(\mathbf{x}) f(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0, \forall f \in L^2(\mathbb{R}^n).$$

Mercer's theorem

Theorem (informal version of Mercer's theorem).

Let K be a positive semidefinite kernel. Then, there exists a “function” φ such that

$$K(\mathbf{x}, \mathbf{y}) = \langle \varphi(\mathbf{x}), \varphi(\mathbf{y}) \rangle.$$

Intuition: As for p.s.d. matrices, p.s.d. kernels have an eigendecomposition

$$K(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{\infty} \lambda_i \psi_i(\mathbf{x}) \psi_i(\mathbf{y}),$$

and the functions ψ_i are called the *eigenfunctions* of K . The “function” φ of the theorem is the (possibly infinite) sequence $\varphi(\mathbf{x}) = (\sqrt{\lambda_i} \psi_i(\mathbf{x}))_{i \in \mathbb{N}}$.

Examples of kernels

■ Polynomial kernel

$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^d$ is a p.s.d. kernel over \mathbb{R}^n , $\forall d \in \mathbb{N}$.

The associated feature vector $\varphi(\mathbf{x})$ has dimension $\binom{n+d}{d}$, its coordinates form a basis of $\mathbb{R}_d[x_1, \dots, x_n]$ (space of polynomials of degree d on n variables).

■ Gaussian kernel

$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x} - \mathbf{y}\|^2\right)$ is a p.s.d. kernel over \mathbb{R}^n .

Its associated feature vector $\varphi(\mathbf{x})$ is in fact an infinite sequence.

How do p.s.d. kernels help us?

- Many regression or classification tasks can be solved by an algorithm which only needs to access the value of the scalar products $x_i^T x_j$ ($i, j \in [m]$).
- To run the the learning algorithm on the *augmented feature factor* $x'_i = \varphi(x_i)$, simply replace each $x_i^T x_j$ by $\langle \varphi(x_i), \varphi(x_j) \rangle$ in the algorithm.
- If φ comes from a kernel K , this means that we can replace each $x_i^T x_j$ by $K(x_i, x_j)$
Note: we don't need to know the function φ to do so !

How do p.s.d. kernels help us?

- Many regression or classification tasks can be solved by an algorithm which only needs to access the value of the scalar products $x_i^T x_j$ ($i, j \in [m]$).
- To run the the learning algorithm on the *augmented feature factor* $x'_i = \varphi(x_i)$, simply replace each $x_i^T x_j$ by $\langle \varphi(x_i), \varphi(x_j) \rangle$ in the algorithm.
- If φ comes from a kernel K , this means that we can replace each $x_i^T x_j$ by $K(x_i, x_j)$

Note: we don't need to know the function φ to do so !

- The resulting predictor (in the original space) also needs to be written as a function $\hat{y}_* = f(x_*)$ that only depends on the sample x_* through the scalar products $x_*^T x_i$.

Then, prediction with the “kernelized” algorithm is done by replacing each $x_*^T x_i$ by $K(x_*, x_i)$ in the expression of f .

Example: Kernel ridge regression

- Ridge regression estimator: $\theta^* = (X^T X + \lambda I)^{-1} X^T y$.

Example: Kernel ridge regression

- Ridge regression estimator: $\theta^* = (X^T X + \lambda I)^{-1} X^T y$.
- Prediction at a new sample x_* :

$$\hat{y}_* = x_*^T \theta^* = x_*^T (X^T X + \lambda I_n)^{-1} X^T y.$$

Example: Kernel ridge regression

- Ridge regression estimator: $\theta^* = (X^T X + \lambda I)^{-1} X^T y$.
- Prediction at a new sample x_* :

$$\hat{y}_* = x_*^T \theta^* = x_*^T (X^T X + \lambda I_n)^{-1} X^T y.$$

- After some (nontrivial) tricks of linear algebra,

$$\begin{aligned}\hat{y}_* &= x_*^T X^T (X X^T + \lambda I_m)^{-1} y \\ &= k^T (Q + \lambda I_m)^{-1} y,\end{aligned}$$

where $k := [x_*^T x_1, \dots, x_*^T x_m]^T$ and $Q = \{x_i^T x_j\}_{1 \leq i, j \leq m}$.

Example: Kernel ridge regression

- Ridge regression estimator: $\theta^* = (X^T X + \lambda I)^{-1} X^T y$.
- Prediction at a new sample x_* :

$$\hat{y}_* = x_*^T \theta^* = x_*^T (X^T X + \lambda I_n)^{-1} X^T y.$$

- After some (nontrivial) tricks of linear algebra,

$$\begin{aligned}\hat{y}_* &= x_*^T X^T (X X^T + \lambda I_m)^{-1} y \\ &= k^T (Q + \lambda I_m)^{-1} y,\end{aligned}$$

where $k := [x_*^T x_1, \dots, x_*^T x_m]^T$ and $Q = \{x_i^T x_j\}_{1 \leq i, j \leq m}$.

- Kernel trick: given p.s.d. kernel function K ,
 - **Training**: compute the vector $z = (Q' + \lambda I_m)^{-1} y$, where $Q'_{ij} := K(x_i, x_j)$
 - **Prediction** at x_* : returns $\hat{y}'_* := k(x_*)^T z$, where $k(x_*) := [K(x_*, x_1), \dots, K(x_*, x_m)]^T$.

Example: Kernel ridge regression

- Ridge regression estimator: $\theta^* = (X^T X + \lambda I)^{-1} X^T y$.

- Prediction at a new sample x_* :

$$\hat{y}_* = x_*^T \theta^* = x_*^T (X^T X + \lambda I_n)^{-1} X^T y.$$

- After some (nontrivial) tricks of linear algebra,

$$\begin{aligned}\hat{y}_* &= x_*^T X^T (X X^T + \lambda I_m)^{-1} y \\ &= k^T (Q + \lambda I_m)^{-1} y,\end{aligned}$$

where $k := [x_*^T x_1, \dots, x_*^T x_m]^T$ and $Q = \{x_i^T x_j\}_{1 \leq i, j \leq m}$.

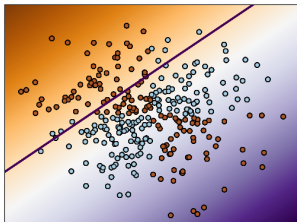
- Kernel trick: given p.s.d. kernel function K ,

- **Training**: compute the vector $z = (Q' + \lambda I_m)^{-1} y$, where $Q'_{ij} := K(x_i, x_j)$

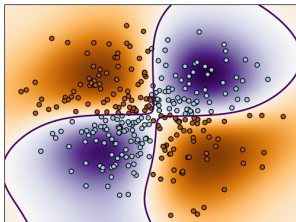
- **Prediction** at x_* : returns $\hat{y}'_* := k(x_*)^T z$, where $k(x_*) := [K(x_*, x_1), \dots, K(x_*, x_m)]^T$.

- **Warning** The kernel trick does not come for free: we must invert a system of dimension m instead of dimension n in the original formulation.

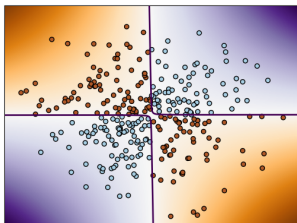
Example: kernel SVM



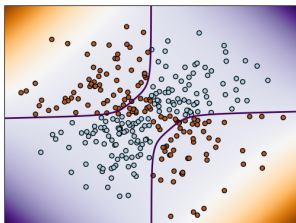
linear: $K(\mathbf{x}, \mathbf{y}) = 1 + \mathbf{x}^T \mathbf{y}$



gaussian: $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2\right)$



quadratic: $K(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x}^T \mathbf{y})^2$



quartic: $K(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x}^T \mathbf{y})^4$

Outline

- 1 Linear regression
- 2 Beyond Least squares
- 3 Classification problems
- 4 The Kernel Trick
- 5 Design of Experiments**

Design of Experiment

Assume a linear model

$$y_i = \mathbf{x}_i^T \boldsymbol{\theta} + \epsilon.$$

- In many problems, collecting samples (\mathbf{x}_i, y_i) is *costly*

Design of Experiment

Assume a linear model

$$y_i = \mathbf{x}_i^T \boldsymbol{\theta} + \epsilon.$$

- In many problems, collecting samples (x_i, y_i) is *costly*
- Given a set of candidate *design points* x_1, \dots, x_m , for which $i \in [m]$ should we collect an observation y_i ?

Design of Experiment

Assume a linear model

$$y_i = \mathbf{x}_i^T \boldsymbol{\theta} + \epsilon.$$

- In many problems, collecting samples (x_i, y_i) is *costly*
- Given a set of candidate *design points* x_1, \dots, x_m , for which $i \in [m]$ should we collect an observation y_i ?
- More generally, we have a budget of N *experimental trials*. The goal is to decide how many times an observation will be collected at the i th design point.

Design of Experiment

Assume a linear model

$$y_i = \mathbf{x}_i^T \boldsymbol{\theta} + \epsilon.$$

- In many problems, collecting samples (\mathbf{x}_i, y_i) is costly
- Given a set of candidate *design points* $\mathbf{x}_1, \dots, \mathbf{x}_m$, for which $i \in [m]$ should we collect an observation y_i ?
- More generally, we have a budget of N *experimental trials*. The goal is to decide how many times an observation will be collected at the i th design point.
- If the i th trial is selected n_i times (with $\sum_{i=1}^m n_i = N$), then we obtain a linear model of the form

$$\mathbf{y} = \mathbf{X}^T \boldsymbol{\theta} + \boldsymbol{\epsilon},$$

where

$$\mathbf{X} = \underbrace{[\mathbf{x}_1, \dots, \mathbf{x}_1]}_{n_1 \text{ times}} \underbrace{[\mathbf{x}_2, \dots, \mathbf{x}_2]}_{n_2 \text{ times}} \dots \underbrace{[\mathbf{x}_m, \dots, \mathbf{x}_m]}_{n_m \text{ times}}^T \in \mathbb{R}^{N \times n}.$$

Information matrix of a design

$$y = X\theta + \epsilon, \quad X = \underbrace{[\mathbf{x}_1, \dots, \mathbf{x}_1]}_{n_1 \text{ times}}, \underbrace{[\mathbf{x}_2, \dots, \mathbf{x}_2]}_{n_2 \text{ times}}, \dots, \underbrace{[\mathbf{x}_m, \dots, \mathbf{x}_m]}_{n_m \text{ times}}^T.$$

- We call a vector $\mathbf{n} \in \mathbb{Z}_{\geq 0}^m$ such that $\sum_{i=1}^m n_i = N$ an *N-exact design*.
- From the Gauss-Markov theorem, we know that the BLUE $\hat{\theta}_{LS}$ for θ has variance proportional to

$$N(X^T X)^{-1} = \left(\sum_{i=1}^m \frac{n_i}{N} n_i \mathbf{x}_i \mathbf{x}_i^T \right)^{-1} =: M(\mathbf{n})^{-1}.$$

- Ideally, we would like to select a design that somehow “minimizes the variance” of $\hat{\theta}_{LS}$, i.e., that “maximizes the information matrix $M(\mathbf{n})$ ”.

Optimality criteria

$$M(\mathbf{n}) = N(\mathbf{X}^T \mathbf{X})^{-1} = \left(\sum_{i=1}^m \frac{n_i}{N} n_i \mathbf{x}_i \mathbf{x}_i^T \right)^{-1}$$

- We cannot maximize $M(\mathbf{n})$ for the $\preceq_{\mathbb{S}_n^+}$ -ordering, because it is not a total order.
- So we will maximize a scalar optimization criterion $\Phi(M(\mathbf{n}))$, such that Φ is concave over \mathbb{S}_{++}^n and monotone w.r.t. the $\preceq_{\mathbb{S}_n^+}$ -ordering.

Optimality criteria

$$M(\mathbf{n}) = N(\mathbf{X}^T \mathbf{X})^{-1} = \left(\sum_{i=1}^m \frac{n_i}{N} n_i \mathbf{x}_i \mathbf{x}_i^T \right)^{-1}$$

- We cannot maximize $M(\mathbf{n})$ for the $\preceq_{\mathbb{S}_n^+}$ -ordering, because it is not a total order.
- So we will maximize a scalar optimization criterion $\Phi(M(\mathbf{n}))$, such that Φ is concave over \mathbb{S}_{++}^n and monotone w.r.t. the $\preceq_{\mathbb{S}_n^+}$ -ordering.
- Most popular criteria:
 - $\Phi_D: M \mapsto \det^{\frac{1}{n}}(M)$;
 - $\Phi_A: M \mapsto (\text{trace } M^{-1})^{-1}$;
 - $\Phi_E: M \mapsto \lambda_{\min}(M)$.

Interpretation of optimality criteria

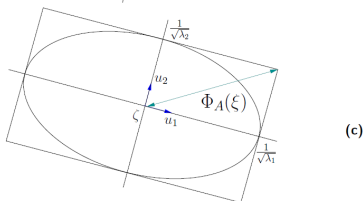
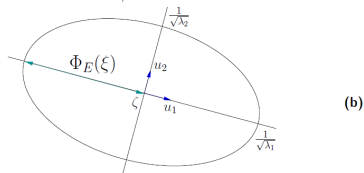
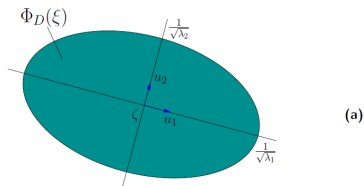
- Under normality assumption of ϵ , with high probability, the BLUE $\hat{\theta}_{LS}$ lies in an ellipsoid centered at θ :

$$\mathcal{E} = \{ \zeta \in \mathbb{R}^n : (\zeta - \theta)^T M(\mathbf{n})^{-1} (\zeta - \theta) \leq \kappa \}.$$

- The semi-axis of this ellipsoid have length proportional to $\frac{1}{\sqrt{\lambda_i}}$, where λ_i 's are the eigenvalues of $M(\mathbf{n})$.

- Hence,

- Φ_D minimizes the *volume* of \mathcal{E} .
- Φ_E minimizes the longest semi-axis of \mathcal{E} .
- Φ_A minimizes the diagonal of the bounding box of \mathcal{E} .



Theory of approximate designs

- Maximizing $\Phi(\sum_{i=1}^m \frac{n_i}{N} x_i x_i^T)$ for a concave criterion Φ over the set of N -exact designs is a hard combinatorial problem.
- Instead, let $w_i := \frac{n_i}{N}$ represent the *fraction* of the experimental effort to spend at x_i .
- w lies in the probability simplex $\Delta = \{w \geq 0, \sum_{i=1}^m w_i = 1\}$.
- The *approximate* design problem is obtained by ignoring the integrality constraints $Nw_i \in \mathbb{Z}$:

$$\underset{w \in \Delta}{\text{maximize}} \quad \Phi\left(\sum_{i=1}^m w_i x_i x_i^T\right)$$

- The criterions Φ_A , Φ_D and Φ_E are SDP-representable. (It can even be shown that Φ_A and Φ_D are SOCP-representable.)

Example: A-optimal design

- By using the Schur complement lemma, we can show that Φ_A has the following SDP-representation:

$$\text{trace } M(\mathbf{w})^{-1} \leq t \iff \exists Y \in \mathbb{S}^n : \begin{cases} \text{trace } Y \leq t \\ \left(\begin{array}{cc} \sum_{i=1}^m w_i \mathbf{x}_i \mathbf{x}_i^T & I_n \\ I_n & Y \end{array} \right) \succeq 0. \end{cases}$$

- Then, the A-optimal design problem can be formulated as

$$\begin{aligned} & \underset{\mathbf{w} \in \mathbb{R}^m, Y \in \mathbb{S}^n}{\text{minimize}} && \text{trace } Y \\ & \text{s.t.} && \left(\begin{array}{cc} \sum_{i=1}^m w_i \mathbf{x}_i \mathbf{x}_i^T & I_n \\ I_n & Y \end{array} \right) \succeq 0 \\ & && \sum_{i=1}^m w_i = 1 \\ & && \mathbf{w} \geq \mathbf{0}. \end{aligned}$$