# Convex Optimization and Applications

## 10 - Application to Data Analysis

Guillaume Sagnol

# Framework

Data: A training set consisting of

- *observations* or *labels* $y_1, \dots, y_m \in \mathbb{R}$
- *feature vectors* $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m \in \mathbb{R}^n$

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

Ultimate goal: Find a *model* which

- explains the relationship between the $\boldsymbol{x}_i$'s and the $y_i$'s.
- can map a new, unseen feature $\boldsymbol{x}_*$ 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* $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m \in \mathbb{R}^n$

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

Ultimate goal: Find a *model* which

- explains the relationship between the $\boldsymbol{x}_i$'s and the $y_i$'s.
- can map a new, unseen feature $\boldsymbol{x}_*$ 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

# Linear Regression

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

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

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

# Linear Regression

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

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

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

## Non-linear embedding

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

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

# Linear Regression

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

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

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

## Non-linear embedding

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

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

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

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

# Linear Regression

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

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

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

## Non-linear embedding

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

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

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

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

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

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

# Least-squares estimator

Choose $\boldsymbol{\theta}$ so as to minimize $\|\boldsymbol{\epsilon}\|^2 = \epsilon_1^2 + \ldots + \epsilon_m^2$, i.e.,

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

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 $\boldsymbol{\theta}$:

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

**Proof**

$$f(\boldsymbol{\theta}) = \|X\boldsymbol{\theta} - \boldsymbol{y}\|^2 = \boldsymbol{\theta}^T X^T X \boldsymbol{\theta} - 2\boldsymbol{y}^T X \boldsymbol{\theta} + \boldsymbol{y}^T \boldsymbol{y}$$
$$\nabla f(\boldsymbol{\theta}) = 2(XX^T \boldsymbol{\theta} - X^T \boldsymbol{y})$$

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

# Least-squares estimator

Choose $\boldsymbol{\theta}$ so as to minimize $\|\boldsymbol{\epsilon}\|^2 = \epsilon_1^2 + \ldots + \epsilon_m^2$, i.e.,

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

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

# Least-squares estimator

Choose $\boldsymbol{\theta}$ so as to minimize $\|\boldsymbol{\epsilon}\|^2 = \epsilon_1^2 + \ldots + \epsilon_m^2$, i.e.,

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

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

## Reasons

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

# Least-squares estimator

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

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

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

## 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 $\theta$

# Least-squares estimator

Choose $\boldsymbol{\theta}$ so as to minimize $\|\boldsymbol{\epsilon}\|^2 = \epsilon_1^2 + \ldots + \epsilon_m^2$, i.e.,

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

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

## Reasons

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

# Maximum Likelihood estimation

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

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

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

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

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

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

# Gauss-Markov theorem

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

# Gauss-Markov theorem

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

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

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

# Gauss-Markov theorem

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

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

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

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

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

# Gauss-Markov theorem

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

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

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

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

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

## Theorem (Gauss-Markov).

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

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

# Gauss-Markov Theorem

## Theorem (Gauss-Markov).

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

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

**Proof**

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

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

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

Its variance is

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

# 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 $\theta$: $\hat{\theta} = LY,\ LX = I$.

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

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

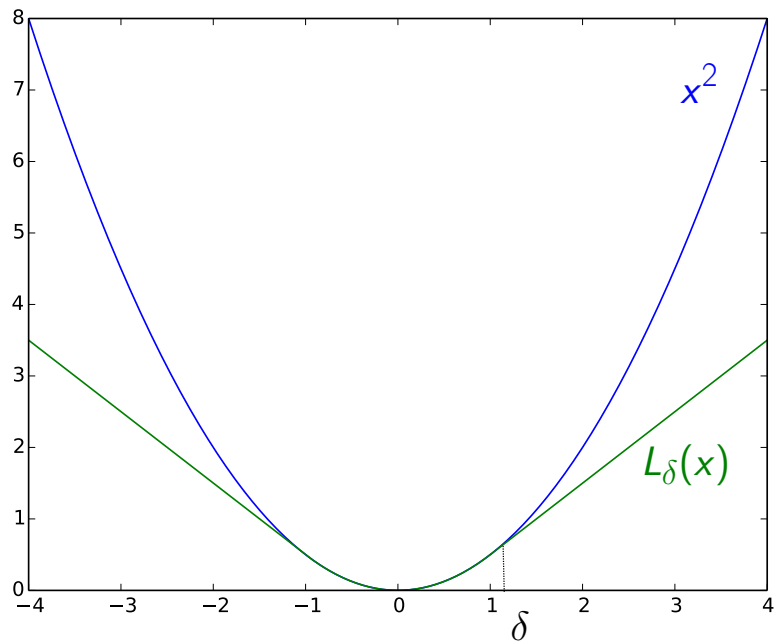
# Huber regression

Minimizing $\|\epsilon\|^2$ is not always a good idea, because the assumption that the $\epsilon_i$'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,

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

# Ridge regression

The matrix $X^T X$ can be nearly singular/badly conditionned $\Longrightarrow$ 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

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

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

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

with a better-conditionned 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 $\hat{\boldsymbol{\theta}}^{\lambda}_{\text{ridge}}$ the ridge-regression estimator with penalty parameter $\lambda \geq 0$, which solves

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

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

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

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

where

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

# 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

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

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

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

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 $\ell_1-$norm, which is, in some sense, the best convex approximation of the problem:

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

# Outline

# 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

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

by maximum-likelihood.

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

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

# Logistic regression

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

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

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

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

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

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

# Support vector machines

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

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

(we rename the label "0" to "$-1$" to simplify the notation)

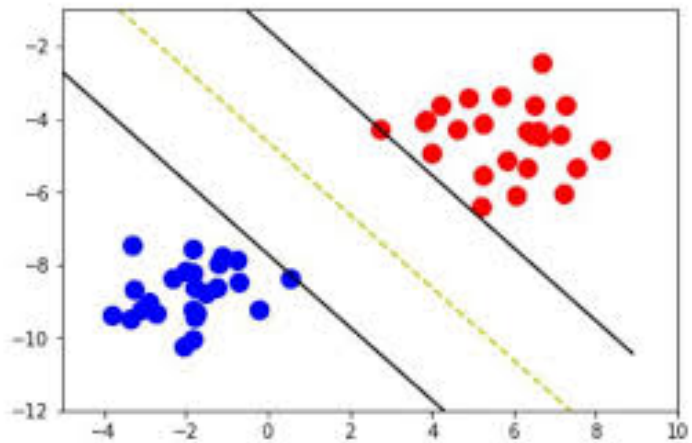
- One of the simplest approaches is to search for a hyperplane separating the samples with label $y_i = -1$ on one side, and the samples labelled $y_i = 1$ on the other side.
- Formally, such a separating hyperplane can be described by $\boldsymbol{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that

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

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

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

## Theorem

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

# Hard margin separation

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

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

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

The hard-margin SVM is hence an SOCP:

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

# 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 "$y_i(\boldsymbol{w}^T \boldsymbol{x}_i - b) \geq 1$" by a penalty for points that are on the wrong side of the margin:

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

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

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

# Soft-margin separation

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

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

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

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

# Outline

# Nonlinear embedding

- In all problems seen so far, we minimize a function $F_{X,\boldsymbol{y}}(\boldsymbol{\theta})$ where the dependence in $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m]^T$ only occurs through scalar products $\boldsymbol{x}_i^T \boldsymbol{\theta}$:

$$F(\boldsymbol{\theta}) = \sum_i f(\boldsymbol{x}_i^T \boldsymbol{\theta}, y_i) + g(\boldsymbol{\theta})$$

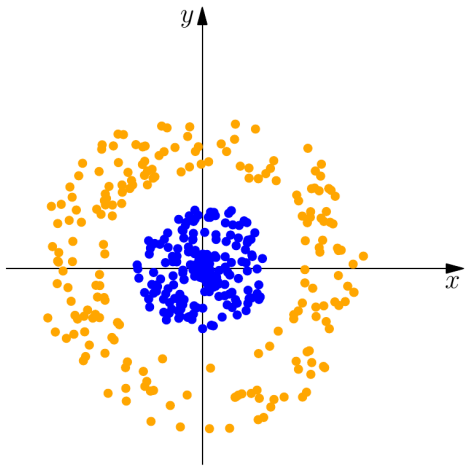
  - For ridge regression, $f(u, y) = (u - y)^2,\ g(\boldsymbol{\theta}) = \lambda \|\boldsymbol{\theta}\|^2$.
  - For lasso regression, $f(u, y) = (u - y)^2,\ g(\boldsymbol{\theta}) = \lambda \|\boldsymbol{\theta}\|_1$.
  - For logistic regression, $f(u, y) = yu - \log(1 + e^u),\ g = 0$.
  - For soft-margin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

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

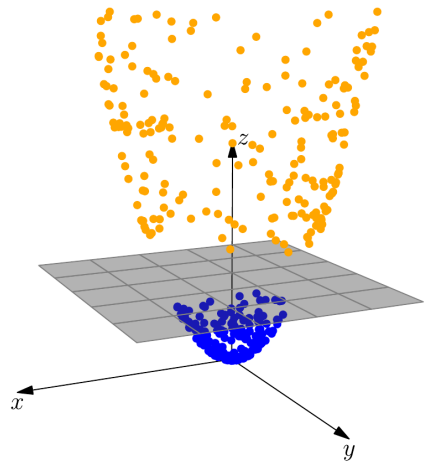
- If we replace $\boldsymbol{x}_i$ by $\varphi(\boldsymbol{x}_i)$ for a non-linear map $\varphi$, we can *learn* a more efficient predictor.

# 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

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

# Nonlinear embedding

- If we replace $\boldsymbol{x}_i$ by $\varphi(\boldsymbol{x}_i)$ for a non-linear map $\varphi$, we can *learn* a more efficient predictor.
- But this raises two issues:
  - How to select the map $\varphi$ ?
  - If $\varphi : \mathbb{R}^n \to \mathbb{R}^{n'}$ maps $\boldsymbol{x}$ 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 $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is called a *positive semidefinite kernel*, if for all $N \in \mathbb{N}$ and for all $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \in \mathbb{R}^n$, the matrix

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

is symmetric positive semidefinite.

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

- Counterpart of "$X \succeq 0 \iff \boldsymbol{u}^T X \boldsymbol{u} \geq 0, \forall \boldsymbol{u}$:

  $K$ positive semidefinite kernel

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

# Mercer's theorem

## Theorem (informal version of Mercer's theorem).

Let $K$ be a positive semidefinite kernel. Then, there exists a "function" $\varphi$ such that

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

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

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

and the functions $\psi_i$ are called the *eigenfunctions* of $K$. The "function" $\varphi$ of the theorem is the (possibly infinite) sequence $\varphi(\boldsymbol{x}) = \left(\sqrt{\lambda_i}\psi_i(\boldsymbol{x})\right)_{i \in \mathbb{N}}$.

# Examples of kernels

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

  The associated feature vector $\varphi(\boldsymbol{x})$ has dimension $\binom{n+d}{d}$, its coordinates form a basis of $\mathbb{R}_d[x_1, \ldots, x_n]$ (space of polynomials of degree $d$ on $n$ variables).
- Gaussian kernel
  $K(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\frac{1}{2\sigma^2}\|\boldsymbol{x} - \boldsymbol{y}\|^2)$ is a p.s.d. kernel over $\mathbb{R}^n$.
  Its associated feature vector $\varphi(\boldsymbol{x})$ 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 need 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 fector* $x_i' = \varphi(x_i)$, simply replace each $x_i^T x_j$ by $\langle \varphi(x_i), \varphi(x_j) \rangle$ in the algorithm.
- If $\varphi$ *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 $\varphi$ to do so !

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

- Many regression or classification tasks can be solved by an algorithm which only need 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 fector* $x_i' = \varphi(x_i)$, simply replace each $x_i^T x_j$ by $\langle \varphi(x_i), \varphi(x_j) \rangle$ in the algorithm.
- If $\varphi$ *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 $\varphi$ 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: $\boldsymbol{\theta}^* = (X^T X + \lambda I)^{-1} X^T \boldsymbol{y}$.

# Example: Kernel ridge regression

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

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

# Example: Kernel ridge regression

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

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

- After some (nontrivial) tricks of linear algebra,

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

  where $\boldsymbol{k} := [\boldsymbol{x}_*^T \boldsymbol{x}_1, \ldots, \boldsymbol{x}_*^T \boldsymbol{x}_m]^T$ and $Q = \{\boldsymbol{x}_i^T \boldsymbol{x}_j\}_{1 \leq i,j \leq m}$.

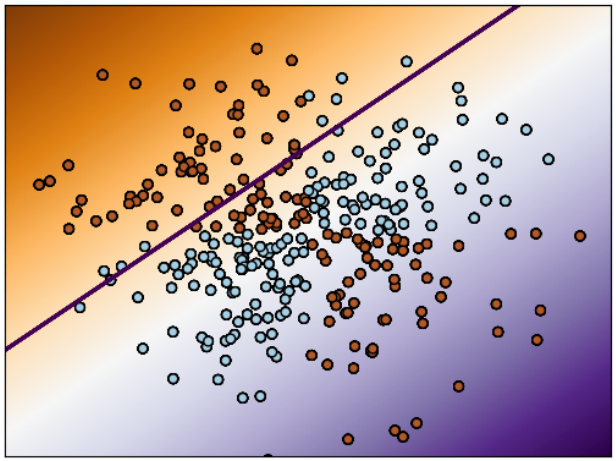
# Example: Kernel ridge regression

- Ridge regression estimator: $\boldsymbol{\theta}^* = (X^T X + \lambda I)^{-1} X^T \boldsymbol{y}$.
- Prediction at a new sample $\boldsymbol{x}_*$:
$$\hat{y}_* = \boldsymbol{x}_*^T \boldsymbol{\theta}^* = \boldsymbol{x}_*^T (X^T X + \lambda I_n)^{-1} X^T \boldsymbol{y}.$$
- After some (nontrivial) tricks of linear algebra,
$$\begin{aligned} \hat{y}_* &= \boldsymbol{x}_*^T X^T (X X^T + \lambda I_m)^{-1} \boldsymbol{y} \\ &= \boldsymbol{k}^T (Q + \lambda I_m)^{-1} \boldsymbol{y}, \end{aligned}$$
where $\boldsymbol{k} := [\boldsymbol{x}_*^T \boldsymbol{x}_1, \ldots, \boldsymbol{x}_*^T \boldsymbol{x}_m]^T$ and $Q = \{\boldsymbol{x}_i^T \boldsymbol{x}_j\}_{1 \leq i,j \leq m}$.
- Kernel trick: given p.s.d. kernel function $K$,
  - Training: compute the vector $\boldsymbol{z} = (Q' + \lambda I_m)^{-1} \boldsymbol{y}$, where $Q'_{ij} := K(\boldsymbol{x}_i, \boldsymbol{x}_j)$
  - Prediction at $\boldsymbol{x}_*$: returns $\hat{y}'_* := \boldsymbol{k}(\boldsymbol{x}_*)^T \boldsymbol{z}$, where $\boldsymbol{k}(\boldsymbol{x}_*) := [K(\boldsymbol{x}_*, \boldsymbol{x}_1), \ldots, K(\boldsymbol{x}_*, \boldsymbol{x}_m)]^T$.

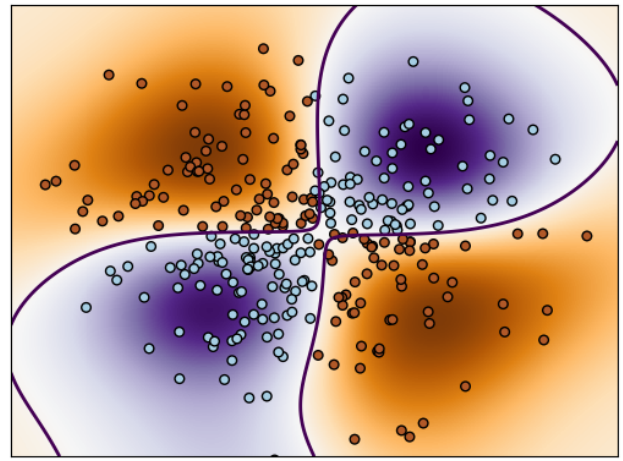
# Example: Kernel ridge regression

- Ridge regression estimator: $\boldsymbol{\theta}^* = (X^T X + \lambda I)^{-1} X^T \boldsymbol{y}$.
- Prediction at a new sample $\boldsymbol{x}_*$:
$$\hat{y}_* = \boldsymbol{x}_*^T \boldsymbol{\theta}^* = \boldsymbol{x}_*^T (X^T X + \lambda I_n)^{-1} X^T \boldsymbol{y}.$$
- After some (nontrivial) tricks of linear algebra,
$$\begin{aligned} \hat{y}_* &= \boldsymbol{x}_*^T X^T (X X^T + \lambda I_m)^{-1} \boldsymbol{y} \\ &= \boldsymbol{k}^T (Q + \lambda I_m)^{-1} \boldsymbol{y}, \end{aligned}$$
where $\boldsymbol{k} := [\boldsymbol{x}_*^T \boldsymbol{x}_1, \ldots, \boldsymbol{x}_*^T \boldsymbol{x}_m]^T$ and $Q = \{\boldsymbol{x}_i^T \boldsymbol{x}_j\}_{1 \leq i,j \leq m}$.
- Kernel trick: given p.s.d. kernel function $K$,
  - Training: compute the vector $\boldsymbol{z} = (Q' + \lambda I_m)^{-1} \boldsymbol{y}$, where $Q'_{ij} := K(\boldsymbol{x}_i, \boldsymbol{x}_j)$
  - Prediction at $\boldsymbol{x}_*$: returns $\hat{y}'_* := \boldsymbol{k}(\boldsymbol{x}_*)^T \boldsymbol{z}$, where $\boldsymbol{k}(\boldsymbol{x}_*) := [K(\boldsymbol{x}_*, \boldsymbol{x}_1), \ldots, K(\boldsymbol{x}_*, \boldsymbol{x}_m)]^T$.
- **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.

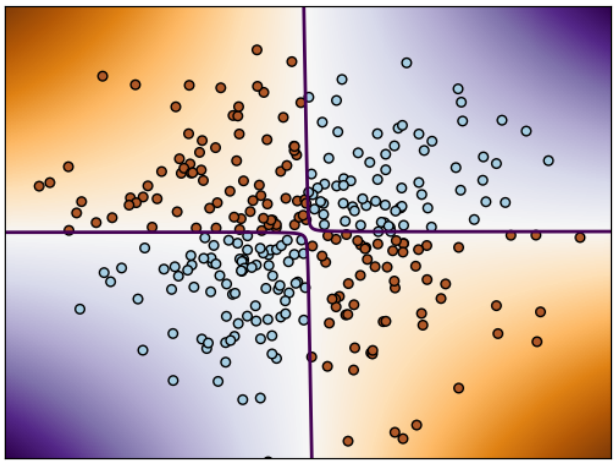
# Example: kernel SVM



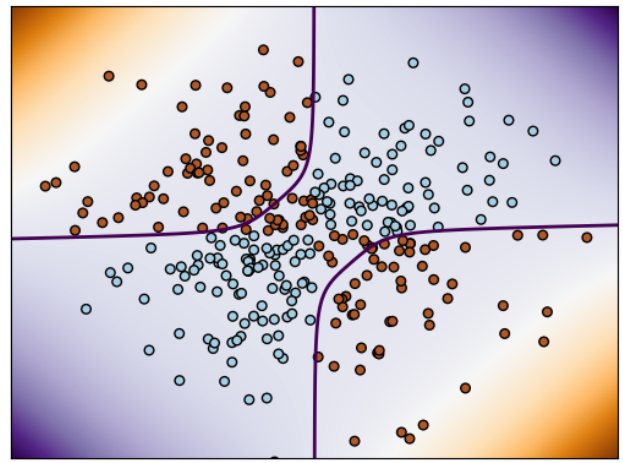
linear: $K(\boldsymbol{x}, \boldsymbol{y}) = 1 + \boldsymbol{x}^T \boldsymbol{y}$



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



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



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

# Outline

# Design of Experiment

Assume a linear model

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

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

# Design of Experiment

Assume a linear model

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

- In many problems, collecting samples $(\boldsymbol{x}_i, y_i)$ is *costly*
- Given a set of candidate *design points* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$, for which $i \in [m]$ should we collect an observation $y_i$ ?

# Design of Experiment

Assume a linear model

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

- In many problems, collecting samples $(\boldsymbol{x}_i, y_i)$ is *costly*
- Given a set of candidate *design points* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$, 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

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

- In many problems, collecting samples $(\boldsymbol{x}_i, y_i)$ is *costly*
- Given a set of candidate *design points* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$, 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

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

where

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

# Information matrix of a design

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

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

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

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

# Optimality criterions

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

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

# Optimality criterions

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

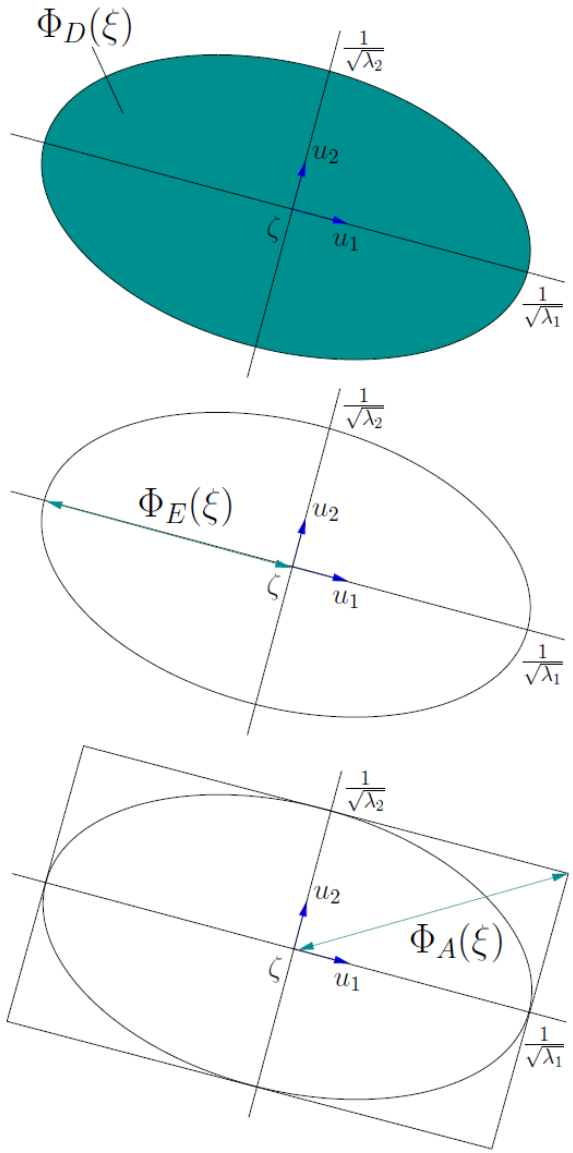
- We cannot maximize $M(\boldsymbol{n})$ for the $\preceq_{\mathbb{S}_n^+}$-ordering, because it is not a total order.
- So we will maximize a scalar optimization criterion $\Phi\big(M(\boldsymbol{n})\big)$, such that $\Phi$ is concave over $\mathbb{S}_{++}^n$ and monotone w.r.t. the $\preceq_{\mathbb{S}_n^+}$-ordering.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

# Interpretation of optimality criterions

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

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

- The semi-axis of this ellipsoid have length proportional to $\frac{1}{\sqrt{\lambda_i}}$, where $\lambda_i$'s are the eigenvalues of $M(n)$.
- Hence,
  - $\Phi_D$ minimizes the *volume* of $\mathcal{E}$.
  - $\Phi_E$ minimizes the longest semi-axis of $\mathcal{E}$.
  - $\Phi_A$ minimizes the diagonal of the bounding box of $\mathcal{E}$.



# Theory of approximate designs

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

$$\underset{\boldsymbol{w} \in \Delta}{\textbf{maximize}} \quad \Phi(\sum_{i=1}^{m} w_i \boldsymbol{x}_i \boldsymbol{x}_i^T)$$

- The criterions $\Phi_A$, $\Phi_D$ and $\Phi_E$ are SDP-representable. (It can even be shown that $\Phi_A$ and $\Phi_D$ are SOCP-representable.)

# Example: A-optimal design

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

$$\textbf{trace}\, M(\boldsymbol{w})^{-1} \leq t \iff \exists Y \in \mathbb{S}^n : \begin{cases} \textbf{trace}\, Y \leq t \\ \begin{pmatrix} \sum_{i=1}^m w_i \boldsymbol{x}_i \boldsymbol{x}_i^T & I_n \\ I_n & Y \end{pmatrix} \succeq 0. \end{cases}$$

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

$$\begin{aligned} \underset{\boldsymbol{w} \in \mathbb{R}^m, Y \in \mathbb{S}^n}{\textbf{minimize}} \quad & \textbf{trace}\, Y \\ s.t. \quad & \begin{pmatrix} \sum_{i=1}^m w_i \boldsymbol{x}_i \boldsymbol{x}_i^T & I_n \\ I_n & Y \end{pmatrix} \succeq 0 \\ & \sum_{i=1}^m w_i = 1 \\ & \boldsymbol{w} \geq \boldsymbol{0}. \end{aligned}$$