Machine Learning

A Bayesian and Optimization Perspective

Academic Press, 2015

Sergios Theodoridis[1]

[1]Dept. of Informatics and Telecommunications, National and Kapodistrian University of Athens, Athens, Greece.

Spring, 2015

Chapter 3
Learning in Parametric Modeling: Basic Concepts and Directions

Version I

Learning in Parametric Modeling

Parameter Estimation: The Deterministic Case

- The task of estimating the value of an unknown parameter vector, $\boldsymbol{\theta}$, is at the center of interest in a number of scientific disciplines. Curve fitting is a typical task. Given a set of data points, the aim is to draw a curve or a surface that "fits" the data.
- The usual path to follow is to adopt a functional form, e.g., a linear function or a quadratic one, and try to estimate the associated unknown coefficients so that the graph of the function "passes through" the data and follows their deployment in space as close as possible.
- The data are given in sets of output-input pairs of points, $(y_n, \boldsymbol{x}_n) \in \mathbb{R} \times \mathbb{R}^l,\ n = 1, 2, \ldots, N$. In a more general setting, the output variables could also be vectors, i.e., $\boldsymbol{y} \in \mathbb{R}^k$.

Learning in Parametric Modeling

Parameter Estimation: The Deterministic Case

- The task of estimating the value of an unknown parameter vector, $\boldsymbol{\theta}$, is at the center of interest in a number of scientific disciplines. Curve fitting is a typical task. Given a set of data points, the aim is to draw a curve or a surface that "fits" the data.
- The usual path to follow is to adopt a functional form, e.g., a linear function or a quadratic one, and try to estimate the associated unknown coefficients so that the graph of the function "passes through" the data and follows their deployment in space as close as possible.
- The data are given in sets of output-input pairs of points, $(y_n, \boldsymbol{x}_n) \in \mathbb{R} \times \mathbb{R}^l,\ n = 1, 2, \ldots, N$. In a more general setting, the output variables could also be vectors, i.e., $\boldsymbol{y} \in \mathbb{R}^k$.

Learning in Parametric Modeling

Parameter Estimation: The Deterministic Case

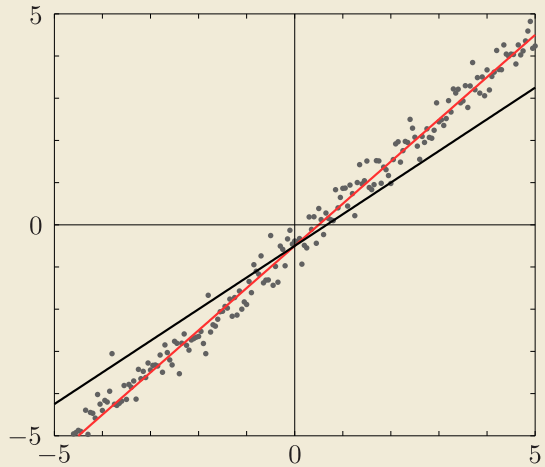
- The task of estimating the value of an unknown parameter vector, $\boldsymbol{\theta}$, is at the center of interest in a number of scientific disciplines. Curve fitting is a typical task. Given a set of data points, the aim is to draw a curve or a surface that "fits" the data.
- The usual path to follow is to adopt a functional form, e.g., a linear function or a quadratic one, and try to estimate the associated unknown coefficients so that the graph of the function "passes through" the data and follows their deployment in space as close as possible.
- The data are given in sets of output-input pairs of points, $(y_n, \boldsymbol{x}_n) \in \mathbb{R} \times \mathbb{R}^l,\ n = 1, 2, \ldots, N$. In a more general setting, the output variables could also be vectors, i.e., $\boldsymbol{y} \in \mathbb{R}^k$.

Curve Fitting

- The parameter estimation task, in the curve fitting context, is demonstrated below via the two examples.

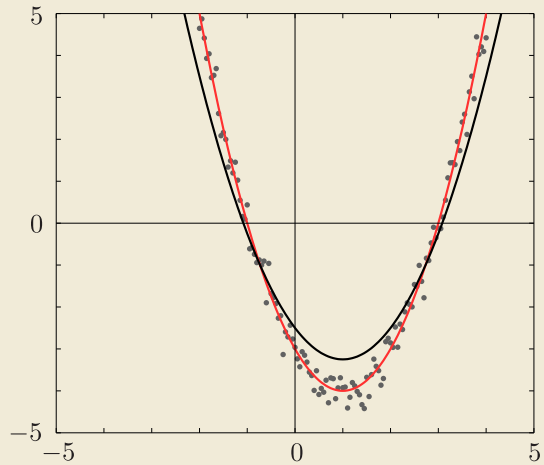
Curve Fitting

- The parameter estimation task, in the curve fitting context, is demonstrated below via the two examples.



$$y = f_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x$$

The choice of the two parameters for the red curve provides a much better fit.

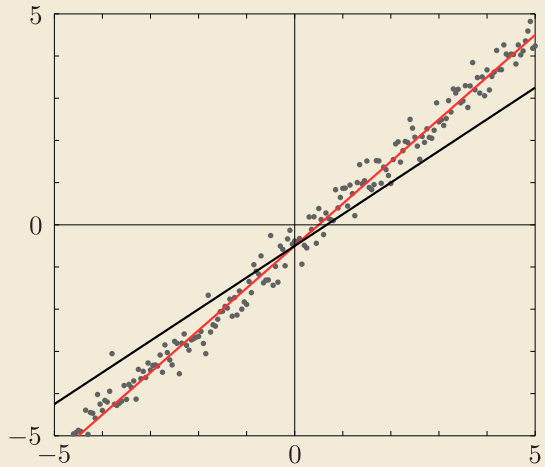


$$y = f_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x + \theta_2 x^2$$

The choice of the three parameters for the red curve provides a much better fit.

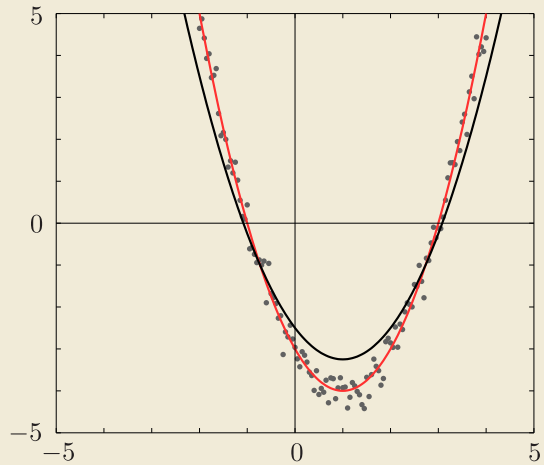
Curve Fitting

- The parameter estimation task, in the curve fitting context, is demonstrated below via the two examples.



$$y = f_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x$$

The choice of the two parameters for the red curve provides a much better fit.



$$y = f_{\boldsymbol{\theta}}(x) = \theta_0 + \theta_1 x + \theta_2 x^2$$

The choice of the three parameters for the red curve provides a much better fit.

- The task comprises two steps: a) Choose a specific parametric functional form and b) select the parameters to provide a "good" fit.

Choice of the Parametric Family of Functions

- In a more formal way, the parameter estimation task is cast as: Given a set of data points, $(y_n, \boldsymbol{x}_n)$, $y_n \in \mathbb{R}$, $\boldsymbol{x}_n \in \mathbb{R}^l$, $n = 1, 2, \ldots, N$, and a parametric set of functions,

$$\mathcal{F} := \left\{ f_{\boldsymbol{\theta}}(\cdot) : \ \boldsymbol{\theta} \in \mathcal{A} \subseteq \mathbb{R}^K \right\},$$

  find a function in $\mathcal{F}$, which will be denoted as $f(\cdot) := f_{\boldsymbol{\theta}_*}(\cdot)$, such that given a value of $\boldsymbol{x} \in \mathbb{R}^l$, $f(\boldsymbol{x})$ best approximates the corresponding value $y \in \mathbb{R}$.

- To reach a decision, with respect to the choice of $\mathcal{F}$, is not an easy task. In practice, one has to use as much a-priori information as possible, concerning the physical mechanism that underlies the generation of the data, and most often to use different families of functions and finally to keep the one that results in the best performance, according to a preselected criterion.

Choice of the Parametric Family of Functions

- In a more formal way, the parameter estimation task is cast as: Given a set of data points, $(y_n, \boldsymbol{x}_n)$, $y_n \in \mathbb{R}$, $\boldsymbol{x}_n \in \mathbb{R}^l$, $n = 1, 2, \ldots, N$, and a parametric set of functions,

$$\mathcal{F} := \left\{ f_{\boldsymbol{\theta}}(\cdot) : \ \boldsymbol{\theta} \in \mathcal{A} \subseteq \mathbb{R}^K \right\},$$

  find a function in $\mathcal{F}$, which will be denoted as $f(\cdot) := f_{\boldsymbol{\theta}_*}(\cdot)$, such that given a value of $\boldsymbol{x} \in \mathbb{R}^l$, $f(\boldsymbol{x})$ best approximates the corresponding value $y \in \mathbb{R}$.

- To reach a decision, with respect to the choice of $\mathcal{F}$, is not an easy task. In practice, one has to use as much a-priori information as possible, concerning the physical mechanism that underlies the generation of the data, and most often to use different families of functions and finally to keep the one that results in the best performance, according to a preselected criterion.

The Loss and Cost Functions

- Having adopted a parametric family of functions, $\mathcal{F}$, one has to get an estimate for the unknown set of parameters. To this end, a measure of fitness is adopted, which is expressed in terms of a loss function.
- The loss function quantifies the deviation/error between the measured value of y and that which is predicted, using the corresponding measurement x, i.e., $f_{\theta}(x)$.
- In a more formal way, we first adopt a nonnegative (loss) function,

$$\mathcal{L}(\cdot,\cdot) : \mathbb{R} \times \mathbb{R} \longmapsto [0, \infty) .$$

- Then, θ_* is computed so that to minimize the total loss, or as we say the cost, over all the data points, i.e.,

$$f(\cdot) := f_{\theta_*}(\cdot) : \ \theta_* = \arg\min_{\theta \in \mathcal{A}} J(\theta), \quad J(\theta) := \sum_{n=1}^{N} \mathcal{L}\big(y_n, f_{\theta}(x_n)\big),$$

assuming that a minimum exists. Note that, in general, there may be more than one optimal values θ_*, depending on the shape of $J(\theta)$.

The Loss and Cost Functions

- Having adopted a parametric family of functions, $\mathcal{F}$, one has to get an estimate for the unknown set of parameters. To this end, a measure of fitness is adopted, which is expressed in terms of a loss function.
- The loss function quantifies the deviation/error between the measured value of y and that which is predicted, using the corresponding measurement $\boldsymbol{x}$, i.e., $f_{\boldsymbol{\theta}}(\boldsymbol{x})$.
- In a more formal way, we first adopt a nonnegative (loss) function,

$$\mathcal{L}(\cdot,\cdot) : \mathbb{R} \times \mathbb{R} \longmapsto [0, \infty) .$$

- Then, $\boldsymbol{\theta}_*$ is computed so that to minimize the total loss, or as we say the cost, over all the data points, i.e.,

$$f(\cdot) := f_{\boldsymbol{\theta}_*}(\cdot) : \ \boldsymbol{\theta}_* = \arg\min_{\boldsymbol{\theta}\in\mathcal{A}} J(\boldsymbol{\theta}), \quad J(\boldsymbol{\theta}) := \sum_{n=1}^{N} \mathcal{L}\big(y_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\big),$$

assuming that a minimum exists. Note that, in general, there may be more than one optimal values $\boldsymbol{\theta}_*$, depending on the shape of $J(\boldsymbol{\theta})$.

The Loss and Cost Functions

- Having adopted a parametric family of functions, $\mathcal{F}$, one has to get an estimate for the unknown set of parameters. To this end, a measure of fitness is adopted, which is expressed in terms of a loss function.
- The loss function quantifies the deviation/error between the measured value of y and that which is predicted, using the corresponding measurement $\boldsymbol{x}$, i.e., $f_{\boldsymbol{\theta}}(\boldsymbol{x})$.
- In a more formal way, we first adopt a nonnegative (loss) function,

$$\mathcal{L}(\cdot,\cdot) : \mathbb{R} \times \mathbb{R} \longmapsto [0, \infty) .$$

- Then, $\boldsymbol{\theta}_*$ is computed so that to minimize the total loss, or as we say the cost, over all the data points, i.e.,

$$f(\cdot) := f_{\boldsymbol{\theta}_*}(\cdot) : \ \boldsymbol{\theta}_* = \arg\min_{\boldsymbol{\theta}\in\mathcal{A}} J(\boldsymbol{\theta}), \quad J(\boldsymbol{\theta}) := \sum_{n=1}^{N} \mathcal{L}\big(y_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\big),$$

assuming that a minimum exists. Note that, in general, there may be more than one optimal values $\boldsymbol{\theta}_*$, depending on the shape of $J(\boldsymbol{\theta})$.

The Loss and Cost Functions

- Having adopted a parametric family of functions, $\mathcal{F}$, one has to get an estimate for the unknown set of parameters. To this end, a measure of fitness is adopted, which is expressed in terms of a loss function.
- The loss function quantifies the deviation/error between the measured value of y and that which is predicted, using the corresponding measurement $\boldsymbol{x}$, i.e., $f_{\boldsymbol{\theta}}(\boldsymbol{x})$.
- In a more formal way, we first adopt a nonnegative (loss) function,

$$\mathcal{L}(\cdot,\cdot) : \mathbb{R} \times \mathbb{R} \longmapsto [0, \infty) \, .$$

- Then, $\boldsymbol{\theta}_*$ is computed so that to minimize the total loss, or as we say the cost, over all the data points, i.e.,

$$f(\cdot) := f_{\boldsymbol{\theta}_*}(\cdot) : \; \boldsymbol{\theta}_* = \arg\min_{\boldsymbol{\theta} \in \mathcal{A}} J(\boldsymbol{\theta}), \quad J(\boldsymbol{\theta}) := \sum_{n=1}^{N} \mathcal{L}\big(y_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\big),$$

assuming that a minimum exists. Note that, in general, there may be more than one optimal values $\boldsymbol{\theta}_*$, depending on the shape of $J(\boldsymbol{\theta})$.

The Squared Error Loss and the Least-Squares Method

- The squared error loss function is defined as

$$\mathcal{L}\big(y, f_{\boldsymbol{\theta}}(\boldsymbol{x})\big) = \big(y - f_{\boldsymbol{\theta}}(\boldsymbol{x})\big)^2,$$

  and it gives rise to the cost function corresponding to the total (over all data points) squared-error loss

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \big(y_n - f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\big)^2.$$

- Minimizing the previous cost function is known as the Least-Squares method (LS), which was first introduced and used by Gauss.
- The use of the LS method together with linear models has a number of computational advantages that makes the method one among, if not the most, popular techniques in machine learning. More specifically:
  - The minimization leads to a unique solution in the parameters' space.
  - The optimal set of the unknown parameters is given by the solution of a linear system of equations.

The Squared Error Loss and the Least-Squares Method

- The squared error loss function is defined as

$$\mathcal{L}\big(y, f_{\boldsymbol{\theta}}(\boldsymbol{x})\big) = \big(y - f_{\boldsymbol{\theta}}(\boldsymbol{x})\big)^2,$$

  and it gives rise to the cost function corresponding to the total (over all data points) squared-error loss

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \big(y_n - f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\big)^2.$$

- Minimizing the previous cost function is known as the Least-Squares method (LS), which was first introduced and used by Gauss.
- The use of the LS method together with linear models has a number of computational advantages that makes the method one among, if not the most, popular techniques in machine learning. More specifically:
  - The minimization leads to a unique solution in the parameters' space.
  - The optimal set of the unknown parameters is given by the solution of a linear system of equations.

The Squared Error Loss and the Least-Squares Method

- The squared error loss function is defined as

$$\mathcal{L}\big(y, f_{\boldsymbol{\theta}}(\boldsymbol{x})\big) = \big(y - f_{\boldsymbol{\theta}}(\boldsymbol{x})\big)^2,$$

  and it gives rise to the cost function corresponding to the total (over all data points) squared-error loss

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \big(y_n - f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\big)^2.$$

- Minimizing the previous cost function is known as the Least-Squares method (LS), which was first introduced and used by Gauss.
- The use of the LS method together with linear models has a number of computational advantages that makes the method one among, if not the most, popular techniques in machine learning. More specifically:
  - The minimization leads to a unique solution in the parameters' space.
  - The optimal set of the unknown parameters is given by the solution of a linear system of equations.

The Squared Error Loss and the Least-Squares Method

- The squared error loss function is defined as

$$\mathcal{L}\big(y, f_{\boldsymbol{\theta}}(\boldsymbol{x})\big) = \big(y - f_{\boldsymbol{\theta}}(\boldsymbol{x})\big)^2,$$

  and it gives rise to the cost function corresponding to the total (over all data points) squared-error loss

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \big(y_n - f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\big)^2.$$

- Minimizing the previous cost function is known as the Least-Squares method (LS), which was first introduced and used by Gauss.
- The use of the LS method together with linear models has a number of computational advantages that makes the method one among, if not the most, popular techniques in machine learning. More specifically:
  - The minimization leads to a unique solution in the parameters' space.
  - The optimal set of the unknown parameters is given by the solution of a linear system of equations.

The Squared Error Loss and the Least-Squares Method

- The squared error loss function is defined as

$$\mathcal{L}\big(y, f_{\boldsymbol{\theta}}(\boldsymbol{x})\big) = \big(y - f_{\boldsymbol{\theta}}(\boldsymbol{x})\big)^2,$$

  and it gives rise to the cost function corresponding to the total (over all data points) squared-error loss

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \big(y_n - f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)\big)^2.$$

- Minimizing the previous cost function is known as the Least-Squares method (LS), which was first introduced and used by Gauss.
- The use of the LS method together with linear models has a number of computational advantages that makes the method one among, if not the most, popular techniques in machine learning. More specifically:
  - The minimization leads to a unique solution in the parameters' space.
  - The optimal set of the unknown parameters is given by the solution of a linear system of equations.

Linear Regression

- Regression is the task of modeling the relationship of a dependent random variable, y, which is considered to be the response of a system, when this is activated by a set of random variables, $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_l$. The latter will be represented as the components of a random vector, $\mathbf{x}$. The relationship is modeled via an additive disturbance or noise term, η. The noise variable, η, is an unobserved random variable.
- The goal of the regression task is to estimate the parameter vector, $\boldsymbol{\theta}$, given a set of measurements, $(y_n, \boldsymbol{x}_n)$, $n = 1, 2, \ldots, N$, that we have at our disposal. This set is known as the training data set, or the observations. The dependent variable is usually known as the output variable and the vector $\mathbf{x}$ as the input vector or the regressor.
- For a linear model, we have that

$$\mathrm{y} = \theta_0 + \theta_1 \mathrm{x}_1 + \ldots + \theta_l \mathrm{x}_l + \eta = \theta_0 + \boldsymbol{\theta}^T \mathbf{x} + \eta = [\boldsymbol{\theta}^T, \theta_0] \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} + \eta,$$

or in short

$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta,$$

where $\boldsymbol{\theta}$ has absorbed θ_0 and $\mathbf{x}$ has been extended by 1. The parameter θ_0 is known as the bias or the intercept.

Linear Regression

- Regression is the task of modeling the relationship of a dependent random variable, y, which is considered to be the response of a system, when this is activated by a set of random variables, $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_l$. The latter will be represented as the components of a random vector, $\mathbf{x}$. The relationship is modeled via an additive disturbance or noise term, η. The noise variable, η, is an unobserved random variable.
- The goal of the regression task is to estimate the parameter vector, $\boldsymbol{\theta}$, given a set of measurements, $(y_n, \boldsymbol{x}_n)$, $n = 1, 2, \ldots, N$, that we have at our disposal. This set is known as the training data set, or the observations. The dependent variable is usually known as the output variable and the vector $\mathbf{x}$ as the input vector or the regressor.
- For a linear model, we have that

$$\mathrm{y} = \theta_0 + \theta_1 \mathrm{x}_1 + \ldots + \theta_l \mathrm{x}_l + \eta = \theta_0 + \boldsymbol{\theta}^T \mathbf{x} + \eta = [\boldsymbol{\theta}^T, \theta_0] \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} + \eta,$$

or in short

$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta,$$

where $\boldsymbol{\theta}$ has absorbed θ_0 and $\mathbf{x}$ has been extended by 1. The parameter θ_0 is known as the bias or the intercept.

Linear Regression

- Regression is the task of modeling the relationship of a dependent random variable, y, which is considered to be the response of a system, when this is activated by a set of random variables, $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_l$. The latter will be represented as the components of a random vector, $\mathbf{x}$. The relationship is modeled via an additive disturbance or noise term, η. The noise variable, η, is an unobserved random variable.
- The goal of the regression task is to estimate the parameter vector, $\boldsymbol{\theta}$, given a set of measurements, $(y_n, \boldsymbol{x}_n)$, $n = 1, 2, \ldots, N$, that we have at our disposal. This set is known as the training data set, or the observations. The dependent variable is usually known as the output variable and the vector $\mathbf{x}$ as the input vector or the regressor.
- For a linear model, we have that

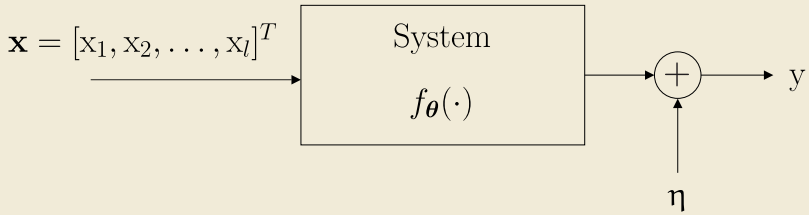
$$\mathrm{y} = \theta_0 + \theta_1 \mathrm{x}_1 + \ldots + \theta_l \mathrm{x}_l + \eta = \theta_0 + \boldsymbol{\theta}^T \mathbf{x} + \eta = [\boldsymbol{\theta}^T, \theta_0] \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} + \eta,$$

or in short

$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta,$$

where $\boldsymbol{\theta}$ has absorbed θ_0 and $\mathbf{x}$ has been extended by 1. The parameter θ_0 is known as the bias or the intercept.

Linear Regression



$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta.$$

- Prediction model for linear regression: The following model is adopted

$$\hat{y} = \hat{\theta}_0 + \hat{\theta}_1 x_1 + \ldots + \hat{\theta}_l x_l := \hat{\boldsymbol{\theta}}^T \boldsymbol{x}.$$

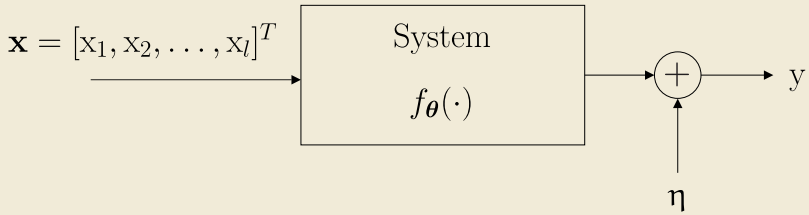
- Using the squared error loss function, the estimate $\hat{\theta}$ is set equal to θ_*, which minimizes the cost function

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2.$$

- Taking the derivative (gradient) with respect to θ and equating to the zero vector, $\mathbf{0}$, we obtain

$$\left(\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T \right) \hat{\boldsymbol{\theta}} = \sum_{n=1}^{N} \boldsymbol{x}_n y_n.$$

Linear Regression



$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta.$$

- Prediction model for linear regression: The following model is adopted

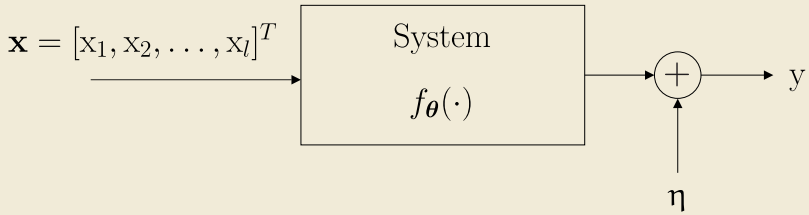
$$\hat{y} = \hat{\theta}_0 + \hat{\theta}_1 x_1 + \ldots + \hat{\theta}_l x_l := \hat{\boldsymbol{\theta}}^T \boldsymbol{x}.$$

- Using the squared error loss function, the estimate $\hat{\theta}$ is set equal to θ_*, which minimizes the cost function

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2.$$

- Taking the derivative (gradient) with respect to θ and equating to the zero vector, $\mathbf{0}$, we obtain

$$\left(\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T \right) \hat{\boldsymbol{\theta}} = \sum_{n=1}^{N} \boldsymbol{x}_n y_n.$$



$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta.$$

- Prediction model for linear regression: The following model is adopted

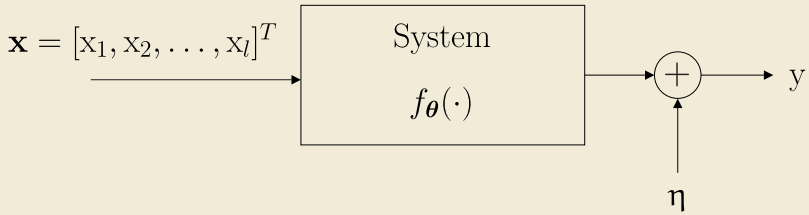
$$\hat{y} = \hat{\theta}_0 + \hat{\theta}_1 x_1 + \ldots + \hat{\theta}_l x_l := \hat{\boldsymbol{\theta}}^T \boldsymbol{x}.$$

- Using the squared error loss function, the estimate $\hat{\boldsymbol{\theta}}$ is set equal to $\boldsymbol{\theta}_*$, which minimizes the cost function

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2.$$

- Taking the derivative (gradient) with respect to θ and equating to the zero vector, 0, we obtain

$$\left(\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T \right) \hat{\boldsymbol{\theta}} = \sum_{n=1}^{N} \boldsymbol{x}_n y_n.$$



$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta.$$

- Prediction model for linear regression: The following model is adopted

$$\hat{y} = \hat{\theta}_0 + \hat{\theta}_1 x_1 + \ldots + \hat{\theta}_l x_l := \hat{\boldsymbol{\theta}}^T \boldsymbol{x}.$$

- Using the squared error loss function, the estimate $\hat{\boldsymbol{\theta}}$ is set equal to $\boldsymbol{\theta}_*$, which minimizes the cost function

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2.$$

- Taking the derivative (gradient) with respect to $\boldsymbol{\theta}$ and equating to the zero vector, $\mathbf{0}$, we obtain

$$\left(\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T \right) \hat{\boldsymbol{\theta}} = \sum_{n=1}^{N} \boldsymbol{x}_n y_n.$$

Linear Regression

- Another way to write the previously obtained relation is via the so-called input matrix, X, defined as the $N \times (l+1)$ matrix, which has as rows the (extended) regressor vectors, $\boldsymbol{x}_n^T$, $n = 1, 2, \ldots, N$, i.e.,

$$X := \begin{bmatrix} \boldsymbol{x}_1^T \\ \boldsymbol{x}_2^T \\ \vdots \\ \boldsymbol{x}_N^T \end{bmatrix} = \begin{bmatrix} x_{11} & \ldots & x_{1l} & 1 \\ x_{21} & \ldots & x_{2l} & 1 \\ \vdots & \ddots & \vdots & \\ x_{N1} & \ldots & x_{Nl} & 1 \end{bmatrix}.$$

Then, it is straightforward to see that the linear system, that provides the Least-Squares solution, can be written as

$$\left(X^T X\right) \hat{\boldsymbol{\theta}} = X^T \boldsymbol{y}, \text{ where } \boldsymbol{y} := [y_1, y_2, \ldots, y_N]^T.$$

- Thus, the LS estimate of the unknown set of parameters, describing to the linear regression model, is given by

$$\hat{\boldsymbol{\theta}} = \left(X^T X\right)^{-1} X^T \boldsymbol{y},$$

assuming, of course, that $\left(X^T X\right)^{-1}$ exists.

Linear Regression

- Another way to write the previously obtained relation is via the so-called input matrix, X, defined as the $N \times (l+1)$ matrix, which has as rows the (extended) regressor vectors, $\boldsymbol{x}_n^T$, $n = 1, 2, \ldots, N$, i.e.,

$$X := \begin{bmatrix} \boldsymbol{x}_1^T \\ \boldsymbol{x}_2^T \\ \vdots \\ \boldsymbol{x}_N^T \end{bmatrix} = \begin{bmatrix} x_{11} & \ldots & x_{1l} & 1 \\ x_{21} & \ldots & x_{2l} & 1 \\ \vdots & \ddots & \vdots & \\ x_{N1} & \ldots & x_{Nl} & 1 \end{bmatrix}.$$

  Then, it is straightforward to see that the linear system, that provides the Least-Squares solution, can be written as

$$\left(X^T X\right) \hat{\boldsymbol{\theta}} = X^T \boldsymbol{y}, \text{ where } \boldsymbol{y} := [y_1, y_2, \ldots, y_N]^T.$$

- Thus, the LS estimate of the unknown set of parameters, describing to the linear regression model, is given by

$$\hat{\boldsymbol{\theta}} = \left(X^T X\right)^{-1} X^T \boldsymbol{y},$$

  assuming, of course, that $\left(X^T X\right)^{-1}$ exists.

Linear Regression

- Another way to write the previously obtained relation is via the so-called input matrix, X, defined as the $N \times (l+1)$ matrix, which has as rows the (extended) regressor vectors, $\boldsymbol{x}_n^T$, $n = 1, 2, \ldots, N$, i.e.,

$$X := \begin{bmatrix} \boldsymbol{x}_1^T \\ \boldsymbol{x}_2^T \\ \vdots \\ \boldsymbol{x}_N^T \end{bmatrix} = \begin{bmatrix} x_{11} & \ldots & x_{1l} & 1 \\ x_{21} & \ldots & x_{2l} & 1 \\ \vdots & \ddots & \vdots & \\ x_{N1} & \ldots & x_{Nl} & 1 \end{bmatrix}.$$

  Then, it is straightforward to see that the linear system, that provides the Least-Squares solution, can be written as

$$\left(X^T X\right) \hat{\boldsymbol{\theta}} = X^T \boldsymbol{y}, \text{ where } \boldsymbol{y} := [y_1, y_2, \ldots, y_N]^T.$$

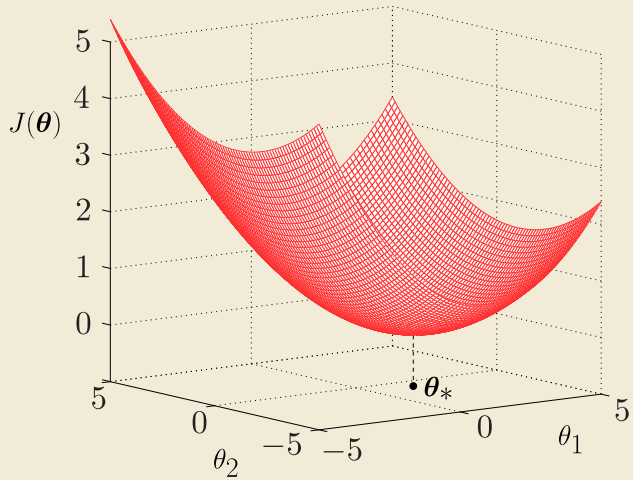
- Thus, the LS estimate of the unknown set of parameters, describing to the linear regression model, is given by

$$\hat{\boldsymbol{\theta}} = \left(X^T X\right)^{-1} X^T \boldsymbol{y},$$

  assuming, of course, that $\left(X^T X\right)^{-1}$ exists.

Linear Regression

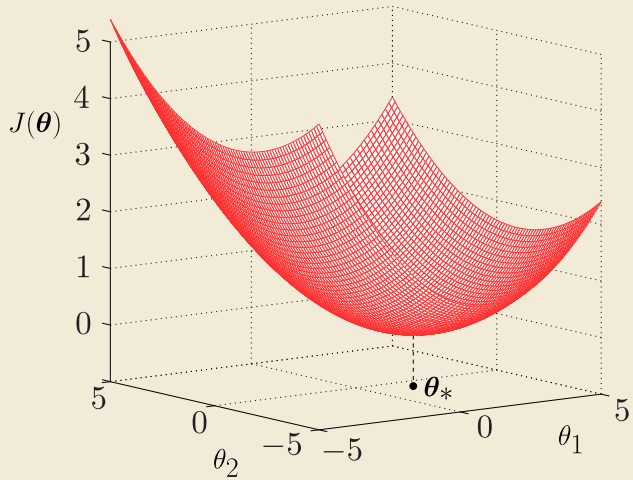
- The LS solution for the linear regression model is unique.



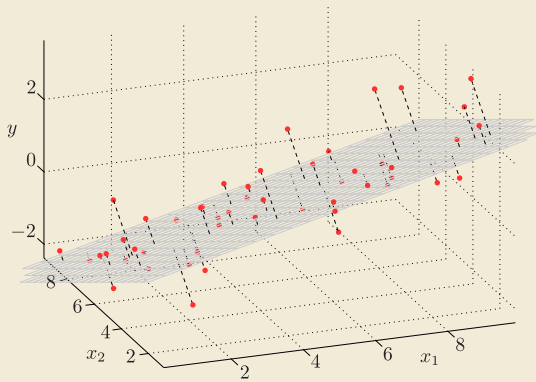
- Assuming the model to be correct, the quality of the fit depends on the noise variance.

Linear Regression

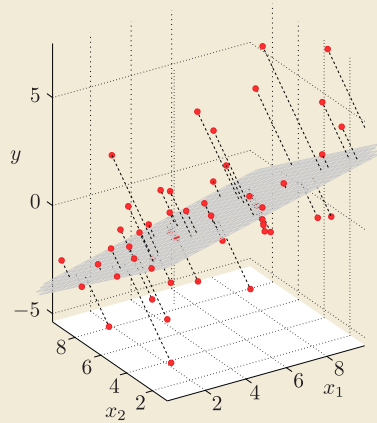
- The LS solution for the linear regression model is unique.



- Assuming the model to be correct, the quality of the fit depends on the noise variance.



Small noise variance



Large noise variance

Classification

- Classification is the task of predicting the class to which an object, known as pattern, belongs. The pattern is assumed to belong to one and only one among a number of a-priori known classes. Each pattern is uniquely represented by a set of measurements, known as features.
- One of the early stages, in designing a classification system, is to select an appropriate set of feature variables. These should "encode" as much class-discriminatory information. Selecting the appropriate, for each problem, set of features is not an easy task and it comprises one of the most important areas within the field of Pattern Recognition.
- Having selected, say, l feature (random) variables, $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_l$, we stack them as the components of the so called feature vector, $\mathbf{x} \in \mathbb{R}^l$.
- The goal is to design a classifier, i.e., a function $f(\boldsymbol{x})$, so that given the values in a feature vector, $\boldsymbol{x}$, which corresponds to a pattern, to be able to predict the class to which the pattern belongs. Equivalently, the classifier defines a decision surface, $f(\boldsymbol{x}) = 0$, in $\mathbb{R}^l$, which partitions the input space into regions. The pattern is classified to a class, according to which region $\boldsymbol{x}$ lies. In the more general setting, a set of functions need to be designed and partition the input space accordingly.

Classification

- Classification is the task of predicting the class to which an object, known as pattern, belongs. The pattern is assumed to belong to one and only one among a number of a-priori known classes. Each pattern is uniquely represented by a set of measurements, known as features.
- One of the early stages, in designing a classification system, is to select an appropriate set of feature variables. These should "encode" as much class-discriminatory information. Selecting the appropriate, for each problem, set of features is not an easy task and it comprises one of the most important areas within the field of Pattern Recognition.
- Having selected, say, l feature (random) variables, $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_l$, we stack them as the components of the so called feature vector, $\mathbf{x} \in \mathbb{R}^l$.
- The goal is to design a classifier, i.e., a function $f(\boldsymbol{x})$, so that given the values in a feature vector, $\boldsymbol{x}$, which corresponds to a pattern, to be able to predict the class to which the pattern belongs. Equivalently, the classifier defines a decision surface, $f(\boldsymbol{x}) = 0$, in $\mathbb{R}^l$, which partitions the input space into regions. The pattern is classified to a class, according to which region $\boldsymbol{x}$ lies. In the more general setting, a set of functions need to be designed and partition the input space accordingly.

Classification

- Classification is the task of predicting the class to which an object, known as pattern, belongs. The pattern is assumed to belong to one and only one among a number of a-priori known classes. Each pattern is uniquely represented by a set of measurements, known as features.
- One of the early stages, in designing a classification system, is to select an appropriate set of feature variables. These should "encode" as much class-discriminatory information. Selecting the appropriate, for each problem, set of features is not an easy task and it comprises one of the most important areas within the field of Pattern Recognition.
- Having selected, say, l feature (random) variables, $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_l$, we stack them as the components of the so called feature vector, $\mathbf{x} \in \mathbb{R}^l$.
- The goal is to design a classifier, i.e., a function $f(\boldsymbol{x})$, so that given the values in a feature vector, $\boldsymbol{x}$, which corresponds to a pattern, to be able to predict the class to which the pattern belongs. Equivalently, the classifier defines a decision surface, $f(\boldsymbol{x}) = 0$, in $\mathbb{R}^l$, which partitions the input space into regions. The pattern is classified to a class, according to which region $\boldsymbol{x}$ lies. In the more general setting, a set of functions need to be designed and partition the input space accordingly.

Classification

- Classification is the task of predicting the class to which an object, known as pattern, belongs. The pattern is assumed to belong to one and only one among a number of a-priori known classes. Each pattern is uniquely represented by a set of measurements, known as features.
- One of the early stages, in designing a classification system, is to select an appropriate set of feature variables. These should "encode" as much class-discriminatory information. Selecting the appropriate, for each problem, set of features is not an easy task and it comprises one of the most important areas within the field of Pattern Recognition.
- Having selected, say, l feature (random) variables, $\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_l$, we stack them as the components of the so called feature vector, $\mathbf{x} \in \mathbb{R}^l$.
- The goal is to design a classifier, i.e., a function $f(\boldsymbol{x})$, so that given the values in a feature vector, $\boldsymbol{x}$, which corresponds to a pattern, to be able to predict the class to which the pattern belongs. Equivalently, the classifier defines a decision surface, $f(\boldsymbol{x}) = 0$, in $\mathbb{R}^l$, which partitions the input space into regions. The pattern is classified to a class, according to which region $\boldsymbol{x}$ lies. In the more general setting, a set of functions need to be designed and partition the input space accordingly.

Classification

- To formulate the task in mathematical terms, each class is represented by the class label variable, y. For the simple two-class classification task, this can take either of two values, depending on the class, e.g, $1, -1$, or $1, 0$, etc. Then, given the value of $\boldsymbol{x}$, corresponding to a specific pattern, its class label is predicted according to the rule,

$$\hat{y} = \phi\big(f(\boldsymbol{x})\big),$$

where ϕ is a non-linear function that indicates on which side of the decision surface, $f(\boldsymbol{x}) = 0$, $\boldsymbol{x}$ lies.

- For example, if the class labels are ± 1, the non-linear function is chosen to be the sign function, i.e., $\phi(\cdot) = \mathrm{sgn}(\cdot)$. The goal is to estimate a function f. Function f is selected so as to belong in a specific parametric class of functions, $\mathcal{F}$. The parameters are obtained so that the deviation between the true class labels, y_n, and the predicted ones, $\hat{y}_n$, to be minimum according to a preselected cost, defined over the training set.

Classification

- To formulate the task in mathematical terms, each class is represented by the class label variable, y. For the simple two-class classification task, this can take either of two values, depending on the class, e.g, $1, -1$, or $1, 0$, etc. Then, given the value of $\boldsymbol{x}$, corresponding to a specific pattern, its class label is predicted according to the rule,

$$\hat{y} = \phi\big(f(\boldsymbol{x})\big),$$

  where ϕ is a non-linear function that indicates on which side of the decision surface, $f(\boldsymbol{x}) = 0$, $\boldsymbol{x}$ lies.

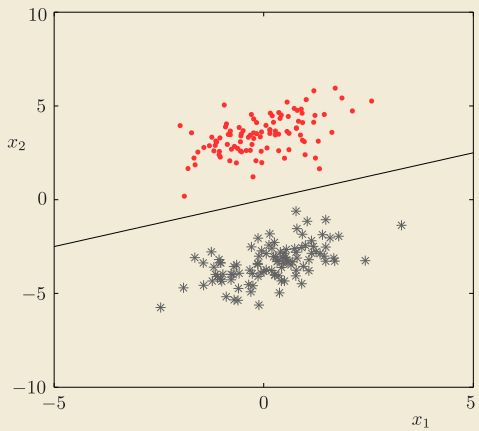
- For example, if the class labels are ± 1, the non-linear function is chosen to be the sign function, i.e., $\phi(\cdot) = \mathrm{sgn}(\cdot)$. The goal is to estimate a function f. Function f is selected so as to belong in a specific parametric class of functions, $\mathcal{F}$. The parameters are obtained so that the deviation between the true class labels, y_n, and the predicted ones, $\hat{y}_n$, to be minimum according to a preselected cost, defined over the training set.

Classification

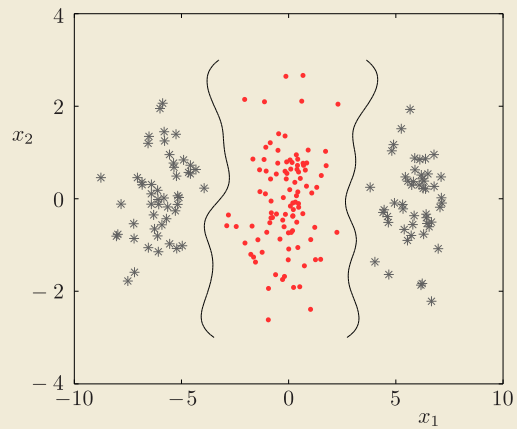
- The training set of points in a classification task is of the form $(y_n, \boldsymbol{x}_n) \in D \times \mathbb{R}^l$, $n = 1, 2, \ldots, N$, where D is the discrete set in which y lies. This is a main difference with regression, where the dependent variable, y, can lie anywhere in an interval of the real axis interval.
- The goal in regression is to estimate a function that follows the deployment of the data in the (y, x) space, while in classification the goal is to partition the space into regions and associate each region with a specific class.

Classification

- The training set of points in a classification task is of the form $(y_n, \boldsymbol{x}_n) \in D \times \mathbb{R}^l$, $n = 1, 2, \ldots, N$, where D is the discrete set in which y lies. This is a main difference with regression, where the dependent variable, y, can lie anywhere in an interval of the real axis interval.
- The goal in regression is to estimate a function that follows the deployment of the data in the $(y, \boldsymbol{x})$ space, while in classification the goal is to partition the space into regions and associate each region with a specific class.



Linear classifier



Nonlinear classifier

Example: Classification via the LS cost

- The LS cost can be used for estimating the parameters of a linear classifier. We set the labels of the training points, that originate from one class, say ω_1, equal to $y = +1$ and the labels of the points originating from the other class, ω_2 (for a two class classification task) equal to $y = -1$. Then, obtain the parameters that define the linear function

$$f(x) := \theta_0 + \theta_1 x_1 + \ldots + \theta_l x_l = \boldsymbol{\theta}^T \boldsymbol{x},$$

so that to minimize the LS cost

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2$$

or

$$J(\boldsymbol{\theta}) = \sum_{n:\boldsymbol{x}_n \in \omega_1} \left(1 - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2 + \sum_{n:\boldsymbol{x}_n \in \omega_2} \left(-1 - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2.$$

Example: Classification via the LS cost

- The LS cost can be used for estimating the parameters of a linear classifier. We set the labels of the training points, that originate from one class, say ω_1, equal to $y = +1$ and the labels of the points originating from the other class, ω_2 (for a two class classification task) equal to $y = -1$. Then, obtain the parameters that define the linear function

$$f(x) := \theta_0 + \theta_1 x_1 + \ldots + \theta_l x_l = \boldsymbol{\theta}^T \boldsymbol{x},$$

so that to minimize the LS cost

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2$$

or

$$J(\boldsymbol{\theta}) = \sum_{n:\boldsymbol{x}_n \in \omega_1} \left(1 - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2 + \sum_{n:\boldsymbol{x}_n \in \omega_2} \left(-1 - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2.$$

Example: Classification via the LS cost

- The LS cost can be used for estimating the parameters of a linear classifier. We set the labels of the training points, that originate from one class, say ω_1, equal to $y = +1$ and the labels of the points originating from the other class, ω_2 (for a two class classification task) equal to $y = -1$. Then, obtain the parameters that define the linear function

$$f(x) := \theta_0 + \theta_1 x_1 + \ldots + \theta_l x_l = \boldsymbol{\theta}^T \boldsymbol{x},$$

so that to minimize the LS cost

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2$$

or

$$J(\boldsymbol{\theta}) = \sum_{n:\boldsymbol{x}_n \in \omega_1} \left(1 - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2 + \sum_{n:\boldsymbol{x}_n \in \omega_2} \left(-1 - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2.$$

Example: Classification via the LS cost

- The LS cost can be used for estimating the parameters of a linear classifier. We set the labels of the training points, that originate from one class, say ω_1, equal to $y = +1$ and the labels of the points originating from the other class, ω_2 (for a two class classification task) equal to $y = -1$. Then, obtain the parameters that define the linear function

$$f(x) := \theta_0 + \theta_1 x_1 + \ldots + \theta_l x_l = \boldsymbol{\theta}^T \boldsymbol{x},$$

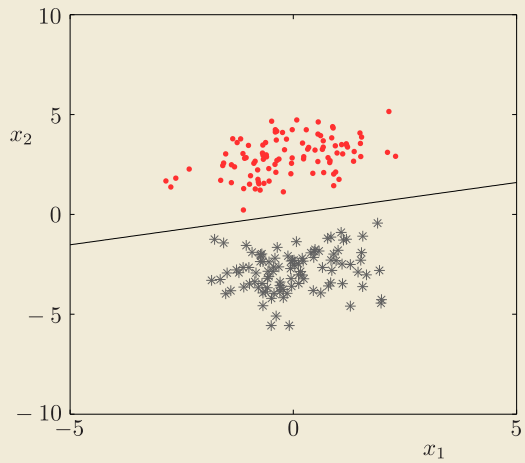
so that to minimize the LS cost

$$J(\boldsymbol{\theta}) = \sum_{n=1}^{N} (y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2$$

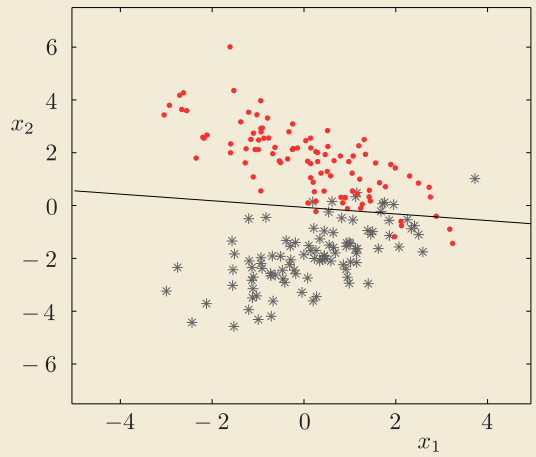
or

$$J(\boldsymbol{\theta}) = \sum_{n:\boldsymbol{x}_n \in \omega_1} \left(1 - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2 + \sum_{n:\boldsymbol{x}_n \in \omega_2} \left(-1 - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2.$$

Example: Classification via the LS cost



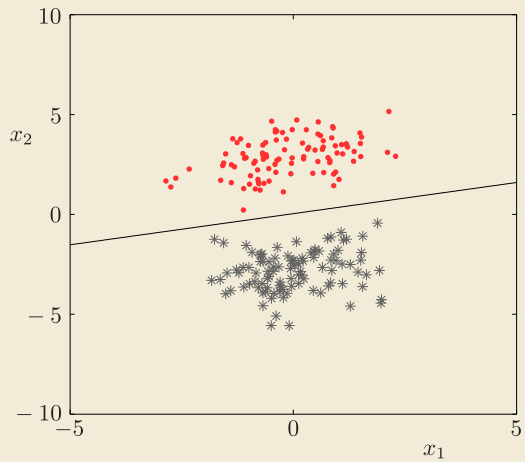
Linearly separable classes



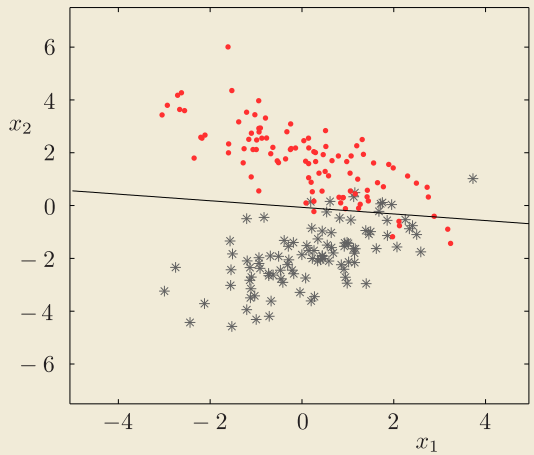
Nonseparable classes

- Due to the discrete nature of the dependent variable (label), y, the LS cost is not well suited for classification tasks. For example, $(y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2$ may be large and contribute to the error, yet, as long as $y_n \boldsymbol{\theta}^T \boldsymbol{x}_n > 0$, the pattern is classified in the correct class and should not be counted as an error. Other, more appropriate loss functions will be considered and used later on, such as the probability of correct classification.

Example: Classification via the LS cost



Linearly separable classes



Nonseparable classes

- Due to the discrete nature of the dependent variable (label), y, the LS cost is not well suited for classification tasks. For example, $(y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n)^2$ may be large and contribute to the error, yet, as long as $y_n \boldsymbol{\theta}^T \boldsymbol{x}_n > 0$, the pattern is classified in the correct class and should not be counted as an error. Other, more appropriate loss functions will be considered and used later on, such as the probability of correct classification.

Discriminative vs Generative Classification Learning

- The path, that we have followed for classification, so far, belongs to the family of methods known as discriminative learning. A functional form of the dependence of the label variable, y, on the input variables, $\mathbf{x}$, was established directly. The statistical nature that ties these two sets of variables, as expressed by their joint distribution, was not taken into account.
- Another form that the discriminative learning can take is to model the conditional $P(y|\boldsymbol{x})$ directly, which also bypasses the need to model the joint distribution. Note that the latter, includes much more information, since it takes into account the statistical nature of the input variables, as well.
- From a statistical point of view, discriminative learning is justified as follows: Recall that

$$p(y, \boldsymbol{x}) = P(y|\boldsymbol{x})p(\boldsymbol{x}).$$

Thus, only the first of the two terms in the product is considered. The distribution of the input data is ignored. The advantage is that simpler models can be used, especially if the input data are described by pdfs of a complex form. The disadvantage is that important information, concerning the input data, is ignored.

Discriminative vs Generative Classification Learning

- The path, that we have followed for classification, so far, belongs to the family of methods known as discriminative learning. A functional form of the dependence of the label variable, y, on the input variables, $\mathbf{x}$, was established directly. The statistical nature that ties these two sets of variables, as expressed by their joint distribution, was not taken into account.
- Another form that the discriminative learning can take is to model the conditional $P(y|\boldsymbol{x})$ directly, which also bypasses the need to model the joint distribution. Note that the latter, includes much more information, since it takes into account the statistical nature of the input variables, as well.
- From a statistical point of view, discriminative learning is justified as follows: Recall that

$$p(y, \boldsymbol{x}) = P(y|\boldsymbol{x})p(\boldsymbol{x}).$$

Thus, only the first of the two terms in the product is considered. The distribution of the input data is ignored. The advantage is that simpler models can be used, especially if the input data are described by pdfs of a complex form. The disadvantage is that important information, concerning the input data, is ignored.

Discriminative vs Generative Classification Learning

- The path, that we have followed for classification, so far, belongs to the family of methods known as discriminative learning. A functional form of the dependence of the label variable, y, on the input variables, $\mathbf{x}$, was established directly. The statistical nature that ties these two sets of variables, as expressed by their joint distribution, was not taken into account.
- Another form that the discriminative learning can take is to model the conditional $P(y|\boldsymbol{x})$ directly, which also bypasses the need to model the joint distribution. Note that the latter, includes much more information, since it takes into account the statistical nature of the input variables, as well.
- From a statistical point of view, discriminative learning is justified as follows: Recall that

$$p(y, \boldsymbol{x}) = P(y|\boldsymbol{x})p(\boldsymbol{x}).$$

Thus, only the first of the two terms in the product is considered. The distribution of the input data is ignored. The advantage is that simpler models can be used, especially if the input data are described by pdfs of a complex form. The disadvantage is that important information, concerning the input data, is ignored.

Discriminative vs Generative Classification Learning

- The path, that we have followed for classification, so far, belongs to the family of methods known as discriminative learning. A functional form of the dependence of the label variable, y, on the input variables, $\mathbf{x}$, was established directly. The statistical nature that ties these two sets of variables, as expressed by their joint distribution, was not taken into account.
- Another form that the discriminative learning can take is to model the conditional $P(y|\boldsymbol{x})$ directly, which also bypasses the need to model the joint distribution. Note that the latter, includes much more information, since it takes into account the statistical nature of the input variables, as well.
- From a statistical point of view, discriminative learning is justified as follows: Recall that

$$p(y, \boldsymbol{x}) = P(y|\boldsymbol{x})p(\boldsymbol{x}).$$

Thus, only the first of the two terms in the product is considered. The distribution of the input data is ignored. The advantage is that simpler models can be used, especially if the input data are described by pdfs of a complex form. The disadvantage is that important information, concerning the input data, is ignored.

Discriminative vs Generative Classification Learning

- In contrast, the alternative path, known as generative learning, exploits the input data distribution, too. Once more, employing the product rule, we have

$$p(y, \boldsymbol{x}) = p(\boldsymbol{x}|y)P(y).$$

  $P(y)$ is the probability concerning the classes and $p(\boldsymbol{x}|y)$ is the conditional distribution of the input **given** the class label.

- For such an approach, we end up with one distribution per class, which has to be learned. In parametric modelling, a set of parameters is associated with each one of these conditional distributions. Once the joint distribution has been learned, the prediction of the class label of an unknown pattern, $\boldsymbol{x}$, is performed based on the a-posteriori probability,

$$P(y|\boldsymbol{x}) = \frac{p(y, \boldsymbol{x})}{p(\boldsymbol{x})} = \frac{p(y, \boldsymbol{x})}{\sum_y p(y, \boldsymbol{x})}.$$

  This is also known as the Bayesian classification rule, and we will focus on such techniques later on.

Supervised, Semisupervised and Unsupervised Learning

- The way our learning tasks have been introduced relied on the availability of a training data set. For this reason, such tasks belong to the family of problems known as supervised learning. However, there are learning problems where the dependent variable is not known, or it may be known for a small percentage of the available training data. In such cases, we refer to clustering and semisupervised learning, respectively. Clustering is also known as unsupervised learning.
- In this series of lectures, most of the emphasis will be on supervised learning. Clustering and semi-supervised learning are treated in detail in

  S. Theodoridis, K. Koutroumbas "Pattern Recognition", 4th Ed., Academic Press, 2009.

Supervised, Semisupervised and Unsupervised Learning

- The way our learning tasks have been introduced relied on the availability of a training data set. For this reason, such tasks belong to the family of problems known as supervised learning. However, there are learning problems where the dependent variable is not known, or it may be known for a small percentage of the available training data. In such cases, we refer to clustering and semisupervised learning, respectively. Clustering is also known as unsupervised learning.
- In this series of lectures, most of the emphasis will be on supervised learning. Clustering and semi-supervised learning are treated in detail in

  S. Theodoridis, K. Koutroumbas "Pattern Recognition", 4th Ed., Academic Press, 2009.

Estimates and Estimators

- In supervised learning, we are given a set of training points, $(y_n, \boldsymbol{x}_n)$, $n = 1, 2, \ldots, N$, and an estimate of the unknown parameter, say $\hat{\boldsymbol{\theta}}$, is returned. However, the training points themselves are random variables. If we are given another set of N observations of the same random variables, these are going to have different values, and obviously the resulting estimate will also be different. In other words, by changing our training data different estimates result.
- Hence, the resulting estimate, of a fixed yet unknown parameter, is itself a random variable. This, in turn, poses questions on how good an estimate is. Each time, the obtained estimate is optimal with respect to the adopted loss function and the specific training set used. However, who guarantees that the resulting estimates are "close" to the true value, assuming that there is one?

Estimates and Estimators

- In supervised learning, we are given a set of training points, $(y_n, \boldsymbol{x}_n)$, $n = 1, 2, \ldots, N$, and an estimate of the unknown parameter, say $\hat{\boldsymbol{\theta}}$, is returned. However, the training points themselves are random variables. If we are given another set of N observations of the same random variables, these are going to have different values, and obviously the resulting estimate will also be different. In other words, by changing our training data different estimates result.
- Hence, the resulting estimate, of a fixed yet unknown parameter, is itself a random variable. This, in turn, poses questions on how good an estimate is. Each time, the obtained estimate is optimal with respect to the adopted loss function and the specific training set used. However, who guarantees that the resulting estimates are "close" to the true value, assuming that there is one?

Estimates and Estimators

- An estimate, e.g., $\hat{\theta} \in \mathbb{R}$, has a specific value, which is the result of a function acting on a specific set of observations, on which our chosen estimate depends, see, e.g., the equations providing the LS estimate. In general, we can write that

$$\hat{\theta} = f(\boldsymbol{y}, X).$$

- However, as the set of observations changes, the estimate becomes itself a random variable, and we write the previous equation in terms of the corresponding random variables,

$$\hat{\theta} = f(\mathbf{y}, \mathrm{X}).$$

This functional dependence is known as the estimator of the corresponding unknown variable θ.

Estimates and Estimators

- An estimate, e.g., $\hat{\theta} \in \mathbb{R}$, has a specific value, which is the result of a function acting on a specific set of observations, on which our chosen estimate depends, see, e.g., the equations providing the LS estimate. In general, we can write that

$$\hat{\theta} = f(\boldsymbol{y}, X).$$

- However, as the set of observations changes, the estimate becomes itself a random variable, and we write the previous equation in terms of the corresponding random variables,

$$\hat{\theta} = f(\mathbf{y}, \mathrm{X}).$$

  This functional dependence is known as the estimator of the corresponding unknown variable θ.

Bias and Variance of an Estimator

- Adopting the squared error loss function to quantify deviations, a reasonable criterion to measure the performance of an estimator, with respect to the true value, denoted here as θ_o, assuming that one exists, is the mean-square error,

$$\text{MSE} = \mathbb{E}\left[(\hat{\theta} - \theta_o)^2\right],$$

where the mean $\mathbb{E}$ is taken over all possible training data sets of size N. If the MSE is small, then we expect that, on average, the resulting estimates to be close to the true value.

- Adding and subtracting in the above the expected value $\mathbb{E}[\hat{\theta}]$, we get

$$\begin{aligned} \text{MSE} &= \mathbb{E}\left[\Big((\hat{\theta} - \mathbb{E}[\hat{\theta}]) + (\mathbb{E}[\hat{\theta}] - \theta_o)\Big)^2\right] \\ &= \underbrace{\mathbb{E}\left[\Big(\hat{\theta} - \mathbb{E}[\hat{\theta}]\Big)^2\right]}_{\text{Variance}} + \underbrace{\Big(\mathbb{E}[\hat{\theta}] - \theta_o\Big)^2}_{\text{Bias}^2}. \end{aligned} \tag{1}$$

- The second equality results if we take into account that the mean value of the product of the two involved terms is zero. Thus, the mean-square error consists of two terms. One is the variance around the mean value and the second one is due to the bias; that is, the deviation of the mean value from the true one.

Bias and Variance of an Estimator

- Adopting the squared error loss function to quantify deviations, a reasonable criterion to measure the performance of an estimator, with respect to the true value, denoted here as θ_o, assuming that one exists, is the mean-square error,

$$\text{MSE} = \mathbb{E}\left[(\hat{\theta} - \theta_o)^2\right],$$

where the mean $\mathbb{E}$ is taken over all possible training data sets of size N. If the MSE is small, then we expect that, on average, the resulting estimates to be close to the true value.

- Adding and subtracting in the above the expected value $\mathbb{E}[\hat{\theta}]$, we get

$$\begin{aligned} \text{MSE} &= \mathbb{E}\left[\left((\hat{\theta} - \mathbb{E}[\hat{\theta}]) + (\mathbb{E}[\hat{\theta}] - \theta_o)\right)^2\right] \\ &= \underbrace{\mathbb{E}\left[\left(\hat{\theta} - \mathbb{E}[\hat{\theta}]\right)^2\right]}_{\text{Variance}} + \underbrace{\left(\mathbb{E}[\hat{\theta}] - \theta_o\right)^2}_{\text{Bias}^2}. \end{aligned} \tag{1}$$

- The second equality results if we take into account that the mean value of the product of the two involved terms is zero. Thus, the mean-square error consists of two terms. One is the variance around the mean value and the second one is due to the bias; that is, the deviation of the mean value from the true one.

Bias and Variance of an Estimator

- Adopting the squared error loss function to quantify deviations, a reasonable criterion to measure the performance of an estimator, with respect to the true value, denoted here as θ_o, assuming that one exists, is the mean-square error,

$$\text{MSE} = \mathbb{E}\left[(\hat{\theta} - \theta_o)^2\right],$$

where the mean $\mathbb{E}$ is taken over all possible training data sets of size N. If the MSE is small, then we expect that, on average, the resulting estimates to be close to the true value.

- Adding and subtracting in the above the expected value $\mathbb{E}[\hat{\theta}]$, we get

$$\begin{aligned} \text{MSE} &= \mathbb{E}\left[\left((\hat{\theta} - \mathbb{E}[\hat{\theta}]) + (\mathbb{E}[\hat{\theta}] - \theta_o)\right)^2\right] \\ &= \underbrace{\mathbb{E}\left[\left(\hat{\theta} - \mathbb{E}[\hat{\theta}]\right)^2\right]}_{\text{Variance}} + \underbrace{\left(\mathbb{E}[\hat{\theta}] - \theta_o\right)^2}_{\text{Bias}^2}. \end{aligned} \tag{1}$$

- The second equality results if we take into account that the mean value of the product of the two involved terms is zero. Thus, the mean-square error consists of two terms. One is the variance around the mean value and the second one is due to the bias; that is, the deviation of the mean value from the true one.

Bias or Unbiased Estimators

- After a first naive look, one may think that an unbiased estimator, i.e., $\mathbb{E}[\hat{\theta}] = \theta_o$, is better than a biased one. However, this is not what the last equation suggests. A good estimator is the one that results in small MSE. Making the last term zero, does **not** mean that MSE becomes necessarily small.
- As a matter fact, the opposite is in general true. Let us make our goal to obtain an estimator that corresponds to the minimum MSE. Then, since the minimum of a constrained task can never become smaller that that of an unconstrained one, we can write

$$\min_{\theta} \text{MSE}(\theta) \leq \min_{\theta:\ \mathbb{E}[\theta]=\theta_o} \text{MSE}(\theta),$$

- Thus, in general, we expect that an optimal biased estimator cannot do worse compared to the unbiased one.
- The unbiased estimator, that results in minimum MSE (variance), is known as the Minimum Variance Unbiased Estimator (MVUE). If such an estimator exists, then it is unique. An MVUE does not always exists.

Bias or Unbiased Estimators

- After a first naive look, one may think that an unbiased estimator, i.e., $\mathbb{E}[\hat{\theta}] = \theta_o$, is better than a biased one. However, this is not what the last equation suggests. A good estimator is the one that results in small MSE. Making the last term zero, does **not** mean that MSE becomes necessarily small.
- As a matter fact, the opposite is in general true. Let us make our goal to obtain an estimator that corresponds to the minimum MSE. Then, since the minimum of a constrained task can never become smaller that that of an unconstrained one, we can write

$$\min_{\theta} \text{MSE}(\theta) \leq \min_{\theta:\ \mathbb{E}[\theta]=\theta_o} \text{MSE}(\theta),$$

- Thus, in general, we expect that an optimal biased estimator cannot do worse compared to the unbiased one.
- The unbiased estimator, that results in minimum MSE (variance), is known as the Minimum Variance Unbiased Estimator (MVUE). If such an estimator exists, then it is unique. An MVUE does not always exists.

Bias or Unbiased Estimators

- After a first naive look, one may think that an unbiased estimator, i.e., $\mathbb{E}[\hat{\theta}] = \theta_o$, is better than a biased one. However, this is not what the last equation suggests. A good estimator is the one that results in small MSE. Making the last term zero, does **not** mean that MSE becomes necessarily small.
- As a matter fact, the opposite is in general true. Let us make our goal to obtain an estimator that corresponds to the minimum MSE. Then, since the minimum of a constrained task can never become smaller that that of an unconstrained one, we can write

$$\min_{\theta} \text{MSE}(\theta) \leq \min_{\theta:\ \mathbb{E}[\theta]=\theta_o} \text{MSE}(\theta),$$

- Thus, in general, we expect that an optimal biased estimator cannot do worse compared to the unbiased one.
- The unbiased estimator, that results in minimum MSE (variance), is known as the Minimum Variance Unbiased Estimator (MVUE). If such an estimator exists, then it is unique. An MVUE does not always exists.

Example Of A Biased Estimator That Does Better Than The MVUE

- The goal is to search for a biased estimator, $\hat{\theta}_b$, which results in a smaller MSE, compared to the unbiased one, assuming that it exists.
- Let us limit our search for $\hat{\theta}_b$, within the class of scalar multiples of $\hat{\theta}_{\text{MVU}}$, i.e.,

$$\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}},$$

where $\alpha \in \mathbb{R}$ is a free parameter.

- Notice that

$$\mathbb{E}[\hat{\theta}_b] = (1+\alpha)\theta_o,$$

where θ_o is the unknown true one.

- Substituting in (1) and after some simple algebra we obtain

$$\text{MSE}(\hat{\theta}_b) = (1+\alpha)^2\text{MSE}(\hat{\theta}_{\text{MVU}}) + \alpha^2\theta_o^2.$$

- In order to get $\text{MSE}(\hat{\theta}_b) < \text{MSE}(\hat{\theta}_{\text{MVU}})$, α must be in the range

$$-\frac{2\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} < \alpha < 0.$$

- The previous range implies that $|1+\alpha| < 1$. Hence,

$$|\hat{\theta}_b| = |(1+\alpha)\hat{\theta}_{\text{MVU}}| < |\hat{\theta}_{\text{MVU}}|.$$

Example Of A Biased Estimator That Does Better Than The MVUE

- The goal is to search for a biased estimator, $\hat{\theta}_b$, which results in a smaller MSE, compared to the unbiased one, assuming that it exists.
- Let us limit our search for $\hat{\theta}_b$, within the class of scalar multiples of $\hat{\theta}_{\text{MVU}}$, i.e.,

$$\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}},$$

where $\alpha \in \mathbb{R}$ is a free parameter.

- Notice that

$$\mathbb{E}[\hat{\theta}_b] = (1+\alpha)\theta_o,$$

where θ_o is the unknown true one.

- Substituting in (1) and after some simple algebra we obtain

$$\text{MSE}(\hat{\theta}_b) = (1+\alpha)^2\text{MSE}(\hat{\theta}_{\text{MVU}}) + \alpha^2\theta_o^2.$$

- In order to get $\text{MSE}(\hat{\theta}_b) < \text{MSE}(\hat{\theta}_{\text{MVU}})$, α must be in the range

$$-\frac{2\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} < \alpha < 0.$$

- The previous range implies that $|1+\alpha| < 1$. Hence,

$$|\hat{\theta}_b| = |(1+\alpha)\hat{\theta}_{\text{MVU}}| < |\hat{\theta}_{\text{MVU}}|.$$

Example Of A Biased Estimator That Does Better Than The MVUE

- The goal is to search for a biased estimator, $\hat{\theta}_b$, which results in a smaller MSE, compared to the unbiased one, assuming that it exists.
- Let us limit our search for $\hat{\theta}_b$, within the class of scalar multiples of $\hat{\theta}_{\text{MVU}}$, i.e.,
$$\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}},$$
where $\alpha \in \mathbb{R}$ is a free parameter.
- Notice that
$$\mathbb{E}[\hat{\theta}_b] = (1+\alpha)\theta_o,$$
where θ_o is the unknown true one.
- Substituting in (1) and after some simple algebra we obtain
$$\text{MSE}(\hat{\theta}_b) = (1+\alpha)^2\text{MSE}(\hat{\theta}_{\text{MVU}}) + \alpha^2\theta_o^2.$$
- In order to get $\text{MSE}(\hat{\theta}_b) < \text{MSE}(\hat{\theta}_{\text{MVU}})$, α must be in the range
$$-\frac{2\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} < \alpha < 0.$$
- The previous range implies that $|1+\alpha| < 1$. Hence,
$$|\hat{\theta}_b| = |(1+\alpha)\hat{\theta}_{\text{MVU}}| < |\hat{\theta}_{\text{MVU}}|.$$

Example Of A Biased Estimator That Does Better Than The MVUE

- The goal is to search for a biased estimator, $\hat{\theta}_b$, which results in a smaller MSE, compared to the unbiased one, assuming that it exists.
- Let us limit our search for $\hat{\theta}_b$, within the class of scalar multiples of $\hat{\theta}_{\text{MVU}}$, i.e.,
$$\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}},$$
where $\alpha \in \mathbb{R}$ is a free parameter.
- Notice that
$$\mathbb{E}[\hat{\theta}_b] = (1+\alpha)\theta_o,$$
where θ_o is the unknown true one.
- Substituting in (1) and after some simple algebra we obtain
$$\text{MSE}(\hat{\theta}_b) = (1+\alpha)^2\text{MSE}(\hat{\theta}_{\text{MVU}}) + \alpha^2\theta_o^2.$$
- In order to get $\text{MSE}(\hat{\theta}_b) < \text{MSE}(\hat{\theta}_{\text{MVU}})$, α must be in the range
$$-\frac{2\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} < \alpha < 0.$$
- The previous range implies that $|1+\alpha| < 1$. Hence,
$$|\hat{\theta}_b| = |(1+\alpha)\hat{\theta}_{\text{MVU}}| < |\hat{\theta}_{\text{MVU}}|.$$

Example Of A Biased Estimator That Does Better Than The MVUE

- The goal is to search for a biased estimator, $\hat{\theta}_b$, which results in a smaller MSE, compared to the unbiased one, assuming that it exists.
- Let us limit our search for $\hat{\theta}_b$, within the class of scalar multiples of $\hat{\theta}_{\text{MVU}}$, i.e.,
$$\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}},$$
where $\alpha \in \mathbb{R}$ is a free parameter.
- Notice that
$$\mathbb{E}[\hat{\theta}_b] = (1+\alpha)\theta_o,$$
where θ_o is the unknown true one.
- Substituting in (1) and after some simple algebra we obtain
$$\text{MSE}(\hat{\theta}_b) = (1+\alpha)^2\text{MSE}(\hat{\theta}_{\text{MVU}}) + \alpha^2\theta_o^2.$$
- In order to get $\text{MSE}(\hat{\theta}_b) < \text{MSE}(\hat{\theta}_{\text{MVU}})$, α must be in the range
$$-\frac{2\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} < \alpha < 0.$$
- The previous range implies that $|1+\alpha| < 1$. Hence,
$$|\hat{\theta}_b| = |(1+\alpha)\hat{\theta}_{\text{MVU}}| < |\hat{\theta}_{\text{MVU}}|.$$

Example Of A Biased Estimator That Does Better Than The MVUE

- The goal is to search for a biased estimator, $\hat{\theta}_b$, which results in a smaller MSE, compared to the unbiased one, assuming that it exists.
- Let us limit our search for $\hat{\theta}_b$, within the class of scalar multiples of $\hat{\theta}_{\text{MVU}}$, i.e.,
$$\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}},$$
where $\alpha \in \mathbb{R}$ is a free parameter.
- Notice that
$$\mathbb{E}[\hat{\theta}_b] = (1+\alpha)\theta_o,$$
where θ_o is the unknown true one.
- Substituting in (1) and after some simple algebra we obtain
$$\text{MSE}(\hat{\theta}_b) = (1+\alpha)^2\text{MSE}(\hat{\theta}_{\text{MVU}}) + \alpha^2\theta_o^2.$$
- In order to get $\text{MSE}(\hat{\theta}_b) < \text{MSE}(\hat{\theta}_{\text{MVU}})$, α must be in the range
$$-\frac{2\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} < \alpha < 0.$$
- The previous range implies that $|1+\alpha| < 1$. Hence,
$$|\hat{\theta}_b| = |(1+\alpha)\hat{\theta}_{\text{MVU}}| < |\hat{\theta}_{\text{MVU}}|.$$

Example Of A Biased Estimator That Does Better Than The MVUE

- We can go a step further and try to compute the optimum value of α, which corresponds to the minimum MSE. By taking the derivative of $\text{MSE}(\hat{\theta}_b)$ with respect to α, it turns out that this occurs for

$$\alpha_* = -\frac{\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} = -\frac{1}{1 + \frac{\theta_o^2}{\text{MSE}(\hat{\theta}_{\text{MVU}})}}.$$

  Therefore, we have found a way to obtain the optimum estimator, among those in the set $\{\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}} : \alpha \in \mathbb{R}\}$, which results in minimum MSE.

- This is true, but as many nice things in life, this is not, in general, realizable. The optimal value for α is given in terms of the unknown, θ_o!
- However, as far as we are concerned, it says something very important. If we want to do better than the MVUE, then, a possible way is to shrink the norm of the MVU estimator. Shrinking the norm is a way of introducing bias into an estimator. We will discuss ways on how to achieve this soon, in the context of regularization.

Example Of A Biased Estimator That Does Better Than The MVUE

- We can go a step further and try to compute the optimum value of α, which corresponds to the minimum MSE. By taking the derivative of $\text{MSE}(\hat{\theta}_b)$ with respect to α, it turns out that this occurs for

$$\alpha_* = -\frac{\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} = -\frac{1}{1 + \frac{\theta_o^2}{\text{MSE}(\hat{\theta}_{\text{MVU}})}}.$$

  Therefore, we have found a way to obtain the optimum estimator, among those in the set $\{\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}} : \alpha \in \mathbb{R}\}$, which results in minimum MSE.
- This is true, but as many nice things in life, this is not, in general, realizable. The optimal value for α is given in terms of the unknown, θ_o!
- However, as far as we are concerned, it says something very important. If we want to do better than the MVUE, then, a possible way is to shrink the norm of the MVU estimator. Shrinking the norm is a way of introducing bias into an estimator. We will discuss ways on how to achieve this soon, in the context of regularization.

Example Of A Biased Estimator That Does Better Than The MVUE

- We can go a step further and try to compute the optimum value of α, which corresponds to the minimum MSE. By taking the derivative of $\text{MSE}(\hat{\theta}_b)$ with respect to α, it turns out that this occurs for

$$\alpha_* = -\frac{\text{MSE}(\hat{\theta}_{\text{MVU}})}{\text{MSE}(\hat{\theta}_{\text{MVU}}) + \theta_o^2} = -\frac{1}{1 + \frac{\theta_o^2}{\text{MSE}(\hat{\theta}_{\text{MVU}})}}.$$

  Therefore, we have found a way to obtain the optimum estimator, among those in the set $\{\hat{\theta}_b = (1+\alpha)\hat{\theta}_{\text{MVU}} : \alpha \in \mathbb{R}\}$, which results in minimum MSE.
- This is true, but as many nice things in life, this is not, in general, realizable. The optimal value for α is given in terms of the unknown, θ_o!
- However, as far as we are concerned, it says something very important. If we want to do better than the MVUE, then, a possible way is to shrink the norm of the MVU estimator. Shrinking the norm is a way of introducing bias into an estimator. We will discuss ways on how to achieve this soon, in the context of regularization.

MSE for Parameter Vectors

- Note that what we said, so far, is readily generalized to parameter vectors. An unbiased parameter vector satisfies

$$\mathbb{E}[\hat{\boldsymbol{\theta}}] = \boldsymbol{\theta}_o.$$

- The MSE around the true value, $\boldsymbol{\theta}_o$, is defined as

$$\text{MSE} = \mathbb{E}\big[(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o)^T(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o)\big].$$

- Looking carefully at the previous definition reveals that the MSE for a parameter vector is the sum of the MSEs of the components, $\hat{\theta}_i$, $i = 1, 2 \ldots, l$, around the corresponding true values θ_{oi}.

MSE for Parameter Vectors

- Note that what we said, so far, is readily generalized to parameter vectors. An unbiased parameter vector satisfies

$$\mathbb{E}[\hat{\boldsymbol{\theta}}] = \boldsymbol{\theta}_o.$$

- The MSE around the true value, $\boldsymbol{\theta}_o$, is defined as

$$\text{MSE} = \mathbb{E}\big[(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o)^T(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o)\big].$$

- Looking carefully at the previous definition reveals that the MSE for a parameter vector is the sum of the MSEs of the components, $\hat{\theta}_i$, $i = 1, 2 \ldots, l$, around the corresponding true values θ_{oi}.

MSE for Parameter Vectors

- Note that what we said, so far, is readily generalized to parameter vectors. An unbiased parameter vector satisfies

$$\mathbb{E}[\hat{\boldsymbol{\theta}}] = \boldsymbol{\theta}_o.$$

- The MSE around the true value, $\boldsymbol{\theta}_o$, is defined as

$$\text{MSE} = \mathbb{E}\big[(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o)^T(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o)\big].$$

- Looking carefully at the previous definition reveals that the MSE for a parameter vector is the sum of the MSEs of the components, $\hat{\theta}_i$, $i = 1, 2 \ldots, l$, around the corresponding true values θ_{oi}.

The Cramér-Rao Lower Bound

- The Cramér-Rao theorem provides a lower bound for the variance of any unbiased estimator, and it is one among the most well known theorems in Statistics.
- Given the set, $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, of the observations, let $p(\mathcal{X}; \theta)$ be the joint distribution describing the observations, which depend on an unknown scalar parameter, $\theta \in \mathbb{R}$. Then, the variance, $\sigma^2_{\hat{\theta}}$, of any unbiased estimator of the corresponding true value of θ, is lower bounded as

$$\sigma^2_{\hat{\theta}} \geq \frac{1}{I(\theta)}, \quad I(\theta) := -\mathbb{E}\Big[\frac{\partial^2 \ln p(\mathcal{X}; \theta)}{\partial \theta^2}\Big].$$

- The necessary and sufficient condition for obtaining an unbiased estimator, which attains the bound, is the existence of a function $g(\cdot)$ such that for all possible values of θ,

$$\frac{\partial \ln p(\mathcal{X}; \theta)}{\partial \theta} = I(\theta)\big(g(\mathcal{X}) - \theta\big).$$

- The MVU estimate is then given by

$$\hat{\theta} = g(\mathcal{X}) := g(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N),$$

and the variance of the respective estimator is equal to $1/I(\theta)$.

The Cramér-Rao Lower Bound

- The Cramér-Rao theorem provides a lower bound for the variance of any unbiased estimator, and it is one among the most well known theorems in Statistics.
- Given the set, $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, of the observations, let $p(\mathcal{X}; \theta)$ be the joint distribution describing the observations, which depend on an unknown scalar parameter, $\theta \in \mathbb{R}$. Then, the variance, $\sigma^2_{\hat{\theta}}$, of any unbiased estimator of the corresponding true value of θ, is lower bounded as

$$\sigma^2_{\hat{\theta}} \geq \frac{1}{I(\theta)}, \quad I(\theta) := -\mathbb{E}\Big[\frac{\partial^2 \ln p(\mathcal{X}; \theta)}{\partial \theta^2}\Big].$$

- The necessary and sufficient condition for obtaining an unbiased estimator, which attains the bound, is the existence of a function $g(\cdot)$ such that for all possible values of θ,

$$\frac{\partial \ln p(\mathcal{X}; \theta)}{\partial \theta} = I(\theta)\big(g(\mathcal{X}) - \theta\big).$$

- The MVU estimate is then given by

$$\hat{\theta} = g(\mathcal{X}) := g(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N),$$

and the variance of the respective estimator is equal to $1/I(\theta)$.

The Cramér-Rao Lower Bound

- The Cramér-Rao theorem provides a lower bound for the variance of any unbiased estimator, and it is one among the most well known theorems in Statistics.
- Given the set, $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, of the observations, let $p(\mathcal{X}; \theta)$ be the joint distribution describing the observations, which depend on an unknown scalar parameter, $\theta \in \mathbb{R}$. Then, the variance, $\sigma^2_{\hat{\theta}}$, of any unbiased estimator of the corresponding true value of θ, is lower bounded as

$$\sigma^2_{\hat{\theta}} \geq \frac{1}{I(\theta)}, \quad I(\theta) := -\mathbb{E}\Big[\frac{\partial^2 \ln p(\mathcal{X}; \theta)}{\partial \theta^2}\Big].$$

- The necessary and sufficient condition for obtaining an unbiased estimator, which attains the bound, is the existence of a function $g(\cdot)$ such that for all possible values of θ,

$$\frac{\partial \ln p(\mathcal{X}; \theta)}{\partial \theta} = I(\theta)\big(g(\mathcal{X}) - \theta\big).$$

- The MVU estimate is then given by

$$\hat{\theta} = g(\mathcal{X}) := g(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N),$$

and the variance of the respective estimator is equal to $1/I(\theta)$.

Example Of An Efficient Estimator

- Assume that we are given a set of noisy measurments/observations of an unknown parameter, θ, i.e.,

$$y_n = \theta + \eta_n, \; n = 1, 2, \ldots N,$$

and the goal is to obtain an estimate of the unknown parameter. Note that this is a special type of a regression task. It is assumed that the noise samples are i.i.d drawn from a Gaussian random variable of zero mean and variance σ_η^2.

- We will show that the mean value of the observations,

$$\bar{y} := \frac{1}{N} \sum_{n=1}^{N} y_n,$$

defines an unbiased estimator that attains the Cramér-Rao lower bound. That is, it is an efficient estimator. Note that (and it can easily be seen), $\hat{\theta} = \bar{y}$ is the LS estimate for this specific regression task.

- First, we show that the corresponding estimator is unbiased. Indeed

$$\mathbb{E}[\bar{\mathrm{y}}] = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}[\mathrm{y}_n] = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}[\theta + \eta_n] = \theta.$$

Example Of An Efficient Estimator

- Assume that we are given a set of noisy measurments/observations of an unknown parameter, θ, i.e.,

$$y_n = \theta + \eta_n, \; n = 1, 2, \dots N,$$

and the goal is to obtain an estimate of the unknown parameter. Note that this is a special type of a regression task. It is assumed that the noise samples are i.i.d drawn from a Gaussian random variable of zero mean and variance σ_η^2.

- We will show that the mean value of the observations,

$$\bar{y} := \frac{1}{N} \sum_{n=1}^{N} y_n,$$

defines an unbiased estimator that attains the Cramér-Rao lower bound. That is, it is an efficient estimator. Note that (and it can easily be seen), $\hat{\theta} = \bar{y}$ is the LS estimate for this specific regression task.

- First, we show that the corresponding estimator is unbiased. Indeed

$$\mathbb{E}[\bar{\mathrm{y}}] = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}[\mathrm{y}_n] = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}[\theta + \eta_n] = \theta.$$

Example Of An Efficient Estimator

- Assume that we are given a set of noisy measurments/observations of an unknown parameter, θ, i.e.,

$$y_n = \theta + \eta_n, \; n = 1, 2, \ldots N,$$

and the goal is to obtain an estimate of the unknown parameter. Note that this is a special type of a regression task. It is assumed that the noise samples are i.i.d drawn from a Gaussian random variable of zero mean and variance σ_η^2.

- We will show that the mean value of the observations,

$$\bar{y} := \frac{1}{N} \sum_{n=1}^{N} y_n,$$

defines an unbiased estimator that attains the Cramér-Rao lower bound. That is, it is an efficient estimator. Note that (and it can easily be seen), $\hat{\theta} = \bar{y}$ is the LS estimate for this specific regression task.

- First, we show that the corresponding estimator is unbiased. Indeed

$$\mathbb{E}[\bar{\mathrm{y}}] = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}[\mathrm{y}_n] = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}[\theta + \eta_n] = \theta.$$

Example Of An Efficient Estimator

- The joint pdf of the observations is given by

$$p(\boldsymbol{y};\theta)=\prod_{n=1}^{N}\frac{1}{\sqrt{2\pi\sigma_\eta^2}}\exp\left(-\frac{(y_n-\theta)^2}{2\sigma_\eta^2}\right),$$

or

$$\ln p(\boldsymbol{y};\theta)=-\frac{N}{2}(2\pi\sigma_\eta^2)-\frac{1}{2\sigma_\eta^2}\sum_{n=1}^{N}(y_n-\theta)^2.$$

- Taking the derivative, we obtain

$$\frac{\partial \ln p(\boldsymbol{y};\theta)}{\partial\theta}=\frac{1}{\sigma_\eta^2}\sum_{n=1}^{N}(y_n-\theta)=\frac{N}{\sigma_\eta^2}(\bar{y}-\theta),\quad \bar{y}:=\frac{1}{N}\sum_{n=1}^{N}y_n.$$

- The second derivative, as required by the theorem, is given by

$$\frac{\partial^2 \ln p(\boldsymbol{y};\theta)}{\partial\theta^2}=-\frac{N}{\sigma_\eta^2},\ \text{hence } I(\theta)=\frac{N}{\sigma_\eta^2}.$$

- Thus, according to the theorem, $\bar{\mathrm{y}}$ is an unbiased estimator that attains the minimum variance error bound with variance equal to $\frac{\sigma_\eta^2}{N}$.

Example Of An Efficient Estimator

- The joint pdf of the observations is given by

$$p(\boldsymbol{y};\theta)=\prod_{n=1}^{N}\frac{1}{\sqrt{2\pi\sigma_\eta^2}}\exp\left(-\frac{(y_n-\theta)^2}{2\sigma_\eta^2}\right),$$

or

$$\ln p(\boldsymbol{y};\theta)=-\frac{N}{2}(2\pi\sigma_\eta^2)-\frac{1}{2\sigma_\eta^2}\sum_{n=1}^{N}(y_n-\theta)^2.$$

- Taking the derivative, we obtain

$$\frac{\partial \ln p(\boldsymbol{y};\theta)}{\partial\theta}=\frac{1}{\sigma_\eta^2}\sum_{n=1}^{N}(y_n-\theta)=\frac{N}{\sigma_\eta^2}(\bar{y}-\theta),\quad \bar{y}:=\frac{1}{N}\sum_{n=1}^{N}y_n.$$

- The second derivative, as required by the theorem, is given by

$$\frac{\partial^2 \ln p(\boldsymbol{y};\theta)}{\partial\theta^2}=-\frac{N}{\sigma_\eta^2},\ \text{hence } I(\theta)=\frac{N}{\sigma_\eta^2}.$$

- Thus, according to the theorem, $\bar{y}$ is an unbiased estimator that attains the minimum variance error bound with variance equal to $\frac{\sigma_\eta^2}{N}$.

Example Of An Efficient Estimator

- The joint pdf of the observations is given by

$$p(\boldsymbol{y};\theta)=\prod_{n=1}^{N}\frac{1}{\sqrt{2\pi\sigma_\eta^2}}\exp\left(-\frac{(y_n-\theta)^2}{2\sigma_\eta^2}\right),$$

or

$$\ln p(\boldsymbol{y};\theta)=-\frac{N}{2}(2\pi\sigma_\eta^2)-\frac{1}{2\sigma_\eta^2}\sum_{n=1}^{N}(y_n-\theta)^2.$$

- Taking the derivative, we obtain

$$\frac{\partial \ln p(\boldsymbol{y};\theta)}{\partial\theta}=\frac{1}{\sigma_\eta^2}\sum_{n=1}^{N}(y_n-\theta)=\frac{N}{\sigma_\eta^2}(\bar{y}-\theta),\quad \bar{y}:=\frac{1}{N}\sum_{n=1}^{N}y_n.$$

- The second derivative, as required by the theorem, is given by

$$\frac{\partial^2 \ln p(\boldsymbol{y};\theta)}{\partial\theta^2}=-\frac{N}{\sigma_\eta^2},\ \text{hence } I(\theta)=\frac{N}{\sigma_\eta^2}.$$

- Thus, according to the theorem, $\bar{\mathrm{y}}$ is an unbiased estimator that attains the minimum variance error bound with variance equal to $\frac{\sigma_\eta^2}{N}$.

The LS Estimator And The Cramér-Rao Bound

- It can easily be shown, following similar arguments as before, that the LS estimator,

$$\hat{\theta} = (X^T X)^{-1} X^T \mathbf{y},$$

in the linear regression problem

$$y_n = \boldsymbol{\theta}^T \boldsymbol{x} + \eta_n,$$

when the noise samples are i.i.d drawn from a zero mean Gaussian distribution, $\mathcal{N}(0, \sigma_\eta^2)$, is an unbiased efficient estimator. This is not true, however, if the noise is not Gaussian or if successive noise samples are correlated; that is, if $\Sigma_\eta \neq \sigma_\eta^2 I$.

Regularization

- We have already seen that the LS estimator is a minimum variance unbiased estimator, under the assumptions of linearity of the regression model and in the presence of a Gaussian white noise source. We also know that one can improve the MSE performance of an estimator by shrinking the norm of the MVU estimator.
- Regularization is a mathematical tool to impose a-priori information on the structure of the solution, which comes as the outcome of an optimization task. Regularization can also be considered as a way to impose bias on an estimator. However, its use in Machine learning can be justified by more general arguments, as it will become apparent soon.
- In the context of the LS regression task, and in order to shrink the norm of the parameter vector estimate, the method of regularization reformulates the LS task as

$$\text{minimize} \quad J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \left(y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2,$$

$$\text{subject to} \quad \|\boldsymbol{\theta}\|^2 \leq \rho,$$

where $\|\cdot\|$ stands for the Euclidean norm of a vector.

Regularization

- We have already seen that the LS estimator is a minimum variance unbiased estimator, under the assumptions of linearity of the regression model and in the presence of a Gaussian white noise source. We also know that one can improve the MSE performance of an estimator by shrinking the norm of the MVU estimator.
- Regularization is a mathematical tool to impose a-priori information on the structure of the solution, which comes as the outcome of an optimization task. Regularization can also be considered as a way to impose bias on an estimator. However, its use in Machine learning can be justified by more general arguments, as it will become apparent soon.
- In the context of the LS regression task, and in order to shrink the norm of the parameter vector estimate, the method of regularization reformulates the LS task as

$$\text{minimize} \quad J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \left(y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2,$$

$$\text{subject to} \quad \|\boldsymbol{\theta}\|^2 \leq \rho,$$

where $\|\cdot\|$ stands for the Euclidean norm of a vector.

Regularization

- We have already seen that the LS estimator is a minimum variance unbiased estimator, under the assumptions of linearity of the regression model and in the presence of a Gaussian white noise source. We also know that one can improve the MSE performance of an estimator by shrinking the norm of the MVU estimator.
- Regularization is a mathematical tool to impose a-priori information on the structure of the solution, which comes as the outcome of an optimization task. Regularization can also be considered as a way to impose bias on an estimator. However, its use in Machine learning can be justified by more general arguments, as it will become apparent soon.
- In the context of the LS regression task, and in order to shrink the norm of the parameter vector estimate, the method of regularization reformulates the LS task as

$$\begin{aligned} \text{minimize} \quad & J(\boldsymbol{\theta}) = \sum_{n=1}^{N} \left(y_n - \boldsymbol{\theta}^T \boldsymbol{x}_n\right)^2, \\ \text{subject to} \quad & \|\boldsymbol{\theta}\|^2 \leq \rho, \end{aligned}$$

where $\|\cdot\|$ stands for the Euclidean norm of a vector.

Ridge Regression

- Constraining the norm of the parameter vector, we do not allow the LS criterion to be completely "free" to reach a solution, but we limit the space in which to search for it.
- For the LS loss function and the previous constraint, the optimization task can also be written as

$$\text{minimize} \qquad L(\boldsymbol{\theta}, \lambda) = \sum_{n=1}^{N} \left(y_n - \boldsymbol{\theta}^T \boldsymbol{x}\right)^2 + \lambda \|\boldsymbol{\theta}\|^2.$$

- It turns out that, for specific choices of $\lambda \geq 0$ and ρ, the two tasks are equivalent. Note that this new cost function, $L(\boldsymbol{\theta}, \lambda)$, involves one term that measures the model misfit and a second one that quantifies the size of the norm of the parameter vector.
- Taking the gradient of L with respect to $\boldsymbol{\theta}$ and equating to zero, we obtain the regularized LS solution for the linear regression task, i.e.,

$$\left(\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T + \lambda I\right) \hat{\boldsymbol{\theta}} = \sum_{n=1}^{N} y_n \boldsymbol{x}_n,$$

where I is the identity matrix of appropriate dimensions. The presence of λ biases the new solution away from that which would have been obtained from the unregularized LS formulation.

Ridge Regression

- Constraining the norm of the parameter vector, we do not allow the LS criterion to be completely "free" to reach a solution, but we limit the space in which to search for it.
- For the LS loss function and the previous constraint, the optimization task can also be written as

$$\text{minimize} \qquad L(\boldsymbol{\theta}, \lambda) = \sum_{n=1}^{N} \left(y_n - \boldsymbol{\theta}^T \boldsymbol{x}\right)^2 + \lambda \|\boldsymbol{\theta}\|^2.$$

- It turns out that, for specific choices of $\lambda \geq 0$ and ρ, the two tasks are equivalent. Note that this new cost function, $L(\theta, \lambda)$, involves one term that measures the model misfit and a second one that quantifies the size of the norm of the parameter vector.
- Taking the gradient of L with respect to θ and equating to zero, we obtain the regularized LS solution for the linear regression task, i.e.,

$$\left(\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T + \lambda I\right) \hat{\boldsymbol{\theta}} = \sum_{n=1}^{N} y_n \boldsymbol{x}_n,$$

where I is the identity matrix of appropriate dimensions. The presence of λ biases the new solution away from that which would have been obtained from the unregularized LS formulation.

Ridge Regression

- Constraining the norm of the parameter vector, we do not allow the LS criterion to be completely "free" to reach a solution, but we limit the space in which to search for it.
- For the LS loss function and the previous constraint, the optimization task can also be written as

$$\text{minimize} \qquad L(\boldsymbol{\theta}, \lambda) = \sum_{n=1}^{N} \left(y_n - \boldsymbol{\theta}^T \boldsymbol{x}\right)^2 + \lambda \|\boldsymbol{\theta}\|^2.$$

- It turns out that, for specific choices of $\lambda \geq 0$ and ρ, the two tasks are equivalent. Note that this new cost function, $L(\boldsymbol{\theta}, \lambda)$, involves one term that measures the model misfit and a second one that quantifies the size of the norm of the parameter vector.
- Taking the gradient of L with respect to θ and equating to zero, we obtain the regularized LS solution for the linear regression task, i.e.,

$$\left(\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T + \lambda I\right) \hat{\boldsymbol{\theta}} = \sum_{n=1}^{N} y_n \boldsymbol{x}_n,$$

where I is the identity matrix of appropriate dimensions. The presence of λ biases the new solution away from that which would have been obtained from the unregularized LS formulation.

Ridge Regression

- Constraining the norm of the parameter vector, we do not allow the LS criterion to be completely "free" to reach a solution, but we limit the space in which to search for it.
- For the LS loss function and the previous constraint, the optimization task can also be written as

$$\text{minimize} \qquad L(\boldsymbol{\theta}, \lambda) = \sum_{n=1}^{N} \left(y_n - \boldsymbol{\theta}^T \boldsymbol{x}\right)^2 + \lambda \|\boldsymbol{\theta}\|^2.$$

- It turns out that, for specific choices of $\lambda \geq 0$ and ρ, the two tasks are equivalent. Note that this new cost function, $L(\boldsymbol{\theta}, \lambda)$, involves one term that measures the model misfit and a second one that quantifies the size of the norm of the parameter vector.
- Taking the gradient of L with respect to $\boldsymbol{\theta}$ and equating to zero, we obtain the regularized LS solution for the linear regression task, i.e.,

$$\left(\sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^T + \lambda I\right) \hat{\boldsymbol{\theta}} = \sum_{n=1}^{N} y_n \boldsymbol{x}_n,$$

where I is the identity matrix of appropriate dimensions. The presence of λ biases the new solution away from that which would have been obtained from the unregularized LS formulation.

Example of Ridge Regression

- The goal of this example is to demonstrate that the obtained via the ridge regression estimate can score a better MSE performance compared to the unconstrained LS solution. The following simple regression model is adopted,

$$y_n = \theta_o + \eta_n, \quad n = 1, 2, \ldots, N,$$

where, for simplicity, we have assumed that the regressors $x_n \equiv 1$, and η_n, $n = 1, 2, \ldots, N$, are i.i.d. samples drawn from a zero-mean Gaussian distribution of variance σ_η^2.

- We have already seen that the LS estimate of the unknown parameter corresponds to the MVU estimator and it is the sample mean, $\hat{\theta}_{\text{MVU}} = \frac{1}{N}\sum_{n=1}^{N} y_n$. Moreover, this solution scores an MSE of σ_η^2/N.
- It can be readily verified that the solution of the corresponding ridge regression task is given by

$$\hat{\theta}_b(\lambda) = \frac{1}{N+\lambda}\sum_{n=1}^{N} y_n = \frac{N}{N+\lambda}\hat{\theta}_{\text{MVU}},$$

where we have explicitly expressed the dependence of the estimate $\hat{\theta}_b$ on the regularization parameter λ. Notice that for the associated estimator, we have, $\mathbb{E}[\hat{\theta}_b(\lambda)] = \frac{N}{N+\lambda}\theta_o$.

Example of Ridge Regression

- The goal of this example is to demonstrate that the obtained via the ridge regression estimate can score a better MSE performance compared to the unconstrained LS solution. The following simple regression model is adopted,

$$y_n = \theta_o + \eta_n, \quad n = 1, 2, \ldots, N,$$

  where, for simplicity, we have assumed that the regressors $x_n \equiv 1$, and η_n, $n = 1, 2, \ldots, N$, are i.i.d. samples drawn from a zero-mean Gaussian distribution of variance σ_η^2.
- We have already seen that the LS estimate of the unknown parameter corresponds to the MVU estimator and it is the sample mean, $\hat{\theta}_{\text{MVU}} = \frac{1}{N}\sum_{n=1}^{N} y_n$. Moreover, this solution scores an MSE of σ_η^2/N.
- It can be readily verified that the solution of the corresponding ridge regression task is given by

$$\hat{\theta}_b(\lambda) = \frac{1}{N+\lambda}\sum_{n=1}^{N} y_n = \frac{N}{N+\lambda}\hat{\theta}_{\text{MVU}},$$

  where we have explicitly expressed the dependence of the estimate $\hat{\theta}_b$ on the regularization parameter λ. Notice that for the associated estimator, we have, $\mathbb{E}[\hat{\theta}_b(\lambda)] = \frac{N}{N+\lambda}\theta_o$.

Example of Ridge Regression

- The goal of this example is to demonstrate that the obtained via the ridge regression estimate can score a better MSE performance compared to the unconstrained LS solution. The following simple regression model is adopted,

$$y_n = \theta_o + \eta_n, \quad n = 1, 2, \ldots, N,$$

  where, for simplicity, we have assumed that the regressors $x_n \equiv 1$, and η_n, $n = 1, 2, \ldots, N$, are i.i.d. samples drawn from a zero-mean Gaussian distribution of variance σ_η^2.
- We have already seen that the LS estimate of the unknown parameter corresponds to the MVU estimator and it is the sample mean, $\hat{\theta}_{\text{MVU}} = \frac{1}{N}\sum_{n=1}^{N} y_n$. Moreover, this solution scores an MSE of σ_η^2/N.
- It can be readily verified that the solution of the corresponding ridge regression task is given by

$$\hat{\theta}_b(\lambda) = \frac{1}{N+\lambda}\sum_{n=1}^{N} y_n = \frac{N}{N+\lambda}\hat{\theta}_{\text{MVU}},$$

  where we have explicitly expressed the dependence of the estimate $\hat{\theta}_b$ on the regularization parameter λ. Notice that for the associated estimator, we have, $\mathbb{E}[\hat{\theta}_b(\lambda)] = \frac{N}{N+\lambda}\theta_o$.

Example of Ridge Regression

- Taking into account the definition of the MSE and taking the derivative with respect to λ, it turns out that that the minimum value of $\text{MSE}(\hat{\theta}_b)$ is

$$\text{MSE}(\hat{\theta}_b(\lambda_*)) = \frac{\frac{\sigma_\eta^2}{N}}{1 + \frac{\sigma_\eta^2}{N\theta_o^2}} < \frac{\sigma_\eta^2}{N} = \text{MSE}(\hat{\theta}_{\text{MVU}}),$$

  and it is attained at $\lambda_* = \sigma_\eta^2/\theta_o^2$.

- Thus, the ridge regression estimator can offer an improvement to the MSE performance. As a matter of fact, there exists always a $\lambda > 0$, such that the ridge regression estimate gives an MSE lower than the one corresponding to the MVU one.
- The following table shows the experimental values, for a specific scenario, by averaging out different realizations to obtain values of the involved MSE estimates. For this case, $\text{MSE}(\hat{\theta}_{\text{MVU}}) \approx 10^{-3}$.

λ	$\text{MSE}(\hat{\theta}_b(\lambda))$
0.1	9.99082×10^{-4}
1.0	9.79790×10^{-4}
100.0	2.74811×10^{-4}
$\lambda_* = 10^3$	9.09671×10^{-5}

Example of Ridge Regression

- Taking into account the definition of the MSE and taking the derivative with respect to λ, it turns out that that the minimum value of $\text{MSE}(\hat{\theta}_b)$ is

$$\text{MSE}(\hat{\theta}_b(\lambda_*)) = \frac{\frac{\sigma_\eta^2}{N}}{1+\frac{\sigma_\eta^2}{N\theta_o^2}} < \frac{\sigma_\eta^2}{N} = \text{MSE}(\hat{\theta}_{\text{MVU}}),$$

  and it is attained at $\lambda_* = \sigma_\eta^2/\theta_o^2$.
- Thus, the ridge regression estimator can offer an improvement to the MSE performance. As a matter of fact, there exists always a $\lambda > 0$, such that the ridge regression estimate gives an MSE lower than the one corresponding to the MVU one.
- The following table shows the experimental values, for a specific scenario, by averaging out different realizations to obtain values of the involved MSE estimates. For this case, $\text{MSE}(\hat{\theta}_{\text{MVU}}) \approx 10^{-3}$.

λ	$\text{MSE}(\hat{\theta}_b(\lambda))$
0.1	9.99082×10^{-4}
1.0	9.79790×10^{-4}
100.0	2.74811×10^{-4}
$\lambda_* = 10^3$	9.09671×10^{-5}

Example of Ridge Regression

- Taking into account the definition of the MSE and taking the derivative with respect to λ, it turns out that that the minimum value of $\text{MSE}(\hat{\theta}_b)$ is

$$\text{MSE}(\hat{\theta}_b(\lambda_*)) = \frac{\frac{\sigma_\eta^2}{N}}{1+\frac{\sigma_\eta^2}{N\theta_o^2}} < \frac{\sigma_\eta^2}{N} = \text{MSE}(\hat{\theta}_{\text{MVU}}),$$

  and it is attained at $\lambda_* = \sigma_\eta^2/\theta_o^2$.
- Thus, the ridge regression estimator can offer an improvement to the MSE performance. As a matter of fact, there exists always a $\lambda > 0$, such that the ridge regression estimate gives an MSE lower than the one corresponding to the MVU one.
- The following table shows the experimental values, for a specific scenario, by averaging out different realizations to obtain values of the involved MSE estimates. For this case, $\text{MSE}(\hat{\theta}_{\text{MVU}}) \approx 10^{-3}$.

λ	$\text{MSE}(\hat{\theta}_b(\lambda))$
0.1	9.99082×10^{-4}
1.0	9.79790×10^{-4}
100.0	2.74811×10^{-4}
$\lambda_* = 10^3$	9.09671×10^{-5}

Inverse Problems: Ill-conditioning and Overfitting

- Most tasks in Machine Learning belong to the so called inverse problems. The latter term encompasses all the problems where one has to infer/ predict/ estimate the values of a model based on a set of available output/input observations-training data. In a less mathematical terminology, in inverse problems one has to unravel unknown causes from known effects; in other words, to reverse the cause-effect relations.
- Inverse problems are typically ill-posed, as opposed to the well-posed ones. Well-posed problems are characterized by: a) the existence of a solution, b) the uniqueness of the solution and c) the stability of the solution. The latter condition is usually violated in machine learning problems. This means that the obtained solution may be very sensitive to changes of the training set. Ill conditioning is another term used to describe this sensitivity.
- The reason for this behavior is that the model used to describe the data can be complex, in the sense that the number of the unknown free parameters is large, with respect to the number of data points. The "face" with which this problem manifests itself in machine learning is known as overfitting.

Inverse Problems: Ill-conditioning and Overfitting

- Most tasks in Machine Learning belong to the so called inverse problems. The latter term encompasses all the problems where one has to infer/ predict/ estimate the values of a model based on a set of available output/input observations-training data. In a less mathematical terminology, in inverse problems one has to unravel unknown causes from known effects; in other words, to reverse the cause-effect relations.
- Inverse problems are typically ill-posed, as opposed to the well-posed ones. Well-posed problems are characterized by: a) the existence of a solution, b) the uniqueness of the solution and c) the stability of the solution. The latter condition is usually violated in machine learning problems. This means that the obtained solution may be very sensitive to changes of the training set. Ill conditioning is another term used to describe this sensitivity.
- The reason for this behavior is that the model used to describe the data can be complex, in the sense that the number of the unknown free parameters is large, with respect to the number of data points. The "face" with which this problem manifests itself in machine learning is known as overfitting.

Inverse Problems: Ill-conditioning and Overfitting

- Most tasks in Machine Learning belong to the so called inverse problems. The latter term encompasses all the problems where one has to infer/ predict/ estimate the values of a model based on a set of available output/input observations-training data. In a less mathematical terminology, in inverse problems one has to unravel unknown causes from known effects; in other words, to reverse the cause-effect relations.
- Inverse problems are typically ill-posed, as opposed to the well-posed ones. Well-posed problems are characterized by: a) the existence of a solution, b) the uniqueness of the solution and c) the stability of the solution. The latter condition is usually violated in machine learning problems. This means that the obtained solution may be very sensitive to changes of the training set. Ill conditioning is another term used to describe this sensitivity.
- The reason for this behavior is that the model used to describe the data can be complex, in the sense that the number of the unknown free parameters is large, with respect to the number of data points. The "face" with which this problem manifests itself in machine learning is known as overfitting.

Inverse Problems: Ill-conditioning and Overfitting

- Overfitting occurs if the estimated parameters of the unknown model learn too much about the idiosyncrasies of the specific training data set, and the model performs badly when it deals with another set of data, other than that used for the training. As a matter of fact, the MSE criterion defined before attempts to quantify exactly this data-dependence of the task; that is, the mean deviation of the obtained estimates from the true value by **changing the training sets**.
- When the number of training samples is small with respect to the number of the unknown parameters, the available information is not enough to "reveal" a sufficiently good model, which fits the data, and it can be misleading due to the presence of the noise and possible outliers. Regularization is an elegant and efficient tool to cope with the complexity of the model; that is, to make it less complex, more smooth. There are different ways to achieve this. One way is by constraining the norm of the unknown parameter, as ridge regression does. When dealing with more complex, compared to linear, models, one can use constraints on the smoothness of the involved non-linear function, e.g., by involving derivatives of the model function in the regularization term.

Inverse Problems: Ill-conditioning and Overfitting

- Overfitting occurs if the estimated parameters of the unknown model learn too much about the idiosyncrasies of the specific training data set, and the model performs badly when it deals with another set of data, other than that used for the training. As a matter of fact, the MSE criterion defined before attempts to quantify exactly this data-dependence of the task; that is, the mean deviation of the obtained estimates from the true value by **changing the training sets**.
- When the number of training samples is small with respect to the number of the unknown parameters, the available information is not enough to "reveal" a sufficiently good model, which fits the data, and it can be misleading due to the presence of the noise and possible outliers. Regularization is an elegant and efficient tool to cope with the complexity of the model; that is, to make it less complex, more smooth. There are different ways to achieve this. One way is by constraining the norm of the unknown parameter, as ridge regression does. When dealing with more complex, compared to linear, models, one can use constraints on the smoothness of the involved non-linear function, e.g., by involving derivatives of the model function in the regularization term.

Mean-Square Error Estimation

- In the same way, that we have already elaborated on the MSE performance of a parameter estimator, we will turn our attention to the task of regression. The more general nonlinear regression task will be considered,

$$\mathrm{y} = g(\mathbf{x}) + \eta.$$

- As a first step, the MSE optimal estimate, $\hat{y}$, given a value of $\boldsymbol{x}$ will be obtained. This is, in general, a nonlinear function of $\boldsymbol{x}$. Let, $p(y, \boldsymbol{x})$ be the corresponding joint distribution. Then, given a set of observations, $\mathbf{x} = \boldsymbol{x} \in \mathbb{R}^l$, the task is to obtain a function $\hat{y} := \hat{g}(\boldsymbol{x}) \in \mathbb{R}$, such that

$$\hat{g}(\boldsymbol{x}) = \arg\min_{f:\mathbb{R}^l \to \mathbb{R}} \mathbb{E}\left[\left(\mathrm{y} - f(\boldsymbol{x})\right)^2\right],$$

  where the expectation is taken with respect to the conditional probability of y given the value of $\boldsymbol{x}$, i.e., $p(y|\boldsymbol{x})$.

- We will show that

$$\hat{g}(\boldsymbol{x}) = \mathbb{E}\left[\mathrm{y}|\boldsymbol{x}\right] := \int_{-\infty}^{+\infty} y p(y|\boldsymbol{x}) dy.$$

Mean-Square Error Estimation

- In the same way, that we have already elaborated on the MSE performance of a parameter estimator, we will turn our attention to the task of regression. The more general nonlinear regression task will be considered,

$$\mathrm{y} = g(\mathbf{x}) + \eta.$$

- As a first step, the MSE optimal estimate, $\hat{y}$, given a value of $\boldsymbol{x}$ will be obtained. This is, in general, a nonlinear function of $\boldsymbol{x}$. Let, $p(y, \boldsymbol{x})$ be the corresponding joint distribution. Then, given a set of observations, $\mathbf{x} = \boldsymbol{x} \in \mathbb{R}^l$, the task is to obtain a function $\hat{y} := \hat{g}(\boldsymbol{x}) \in \mathbb{R}$, such that

$$\hat{g}(\boldsymbol{x}) = \arg\min_{f:\mathbb{R}^l \to \mathbb{R}} \mathbb{E}\left[\left(\mathrm{y} - f(\boldsymbol{x})\right)^2\right],$$

  where the expectation is taken with respect to the conditional probability of y given the value of $\boldsymbol{x}$, i.e., $p(y|\boldsymbol{x})$.

- We will show that

$$\hat{g}(\boldsymbol{x}) = \mathbb{E}\left[\mathrm{y}|\boldsymbol{x}\right] := \int_{-\infty}^{+\infty} y p(y|\boldsymbol{x}) dy.$$

Mean-Square Error Estimation

- In the same way, that we have already elaborated on the MSE performance of a parameter estimator, we will turn our attention to the task of regression. The more general nonlinear regression task will be considered,

$$\mathrm{y} = g(\mathbf{x}) + \eta.$$

- As a first step, the MSE optimal estimate, $\hat{y}$, given a value of $\boldsymbol{x}$ will be obtained. This is, in general, a nonlinear function of $\boldsymbol{x}$. Let, $p(y, \boldsymbol{x})$ be the corresponding joint distribution. Then, given a set of observations, $\mathbf{x} = \boldsymbol{x} \in \mathbb{R}^l$, the task is to obtain a function $\hat{y} := \hat{g}(\boldsymbol{x}) \in \mathbb{R}$, such that

$$\hat{g}(\boldsymbol{x}) = \arg\min_{f:\mathbb{R}^l \to \mathbb{R}} \mathbb{E}\left[\left(\mathrm{y} - f(\boldsymbol{x})\right)^2\right],$$

  where the expectation is taken with respect to the conditional probability of y given the value of $\boldsymbol{x}$, i.e., $p(y|\boldsymbol{x})$.

- We will show that

$$\hat{g}(\boldsymbol{x}) = \mathbb{E}\left[\mathrm{y}|\boldsymbol{x}\right] := \int_{-\infty}^{+\infty} y p(y|\boldsymbol{x}) dy.$$

Mean-Square Error Estimation

- In the same way, that we have already elaborated on the MSE performance of a parameter estimator, we will turn our attention to the task of regression. The more general nonlinear regression task will be considered,

$$\mathrm{y} = g(\mathbf{x}) + \eta.$$

- As a first step, the MSE optimal estimate, $\hat{y}$, given a value of $\boldsymbol{x}$ will be obtained. This is, in general, a nonlinear function of $\boldsymbol{x}$. Let, $p(y, \boldsymbol{x})$ be the corresponding joint distribution. Then, given a set of observations, $\mathbf{x} = \boldsymbol{x} \in \mathbb{R}^l$, the task is to obtain a function $\hat{y} := \hat{g}(\boldsymbol{x}) \in \mathbb{R}$, such that

$$\hat{g}(\boldsymbol{x}) = \arg\min_{f:\mathbb{R}^l \to \mathbb{R}} \mathbb{E}\left[\left(\mathrm{y} - f(\boldsymbol{x})\right)^2\right],$$

  where the expectation is taken with respect to the conditional probability of y given the value of $\boldsymbol{x}$, i.e., $p(y|\boldsymbol{x})$.

- We will show that

$$\hat{g}(\boldsymbol{x}) = \mathbb{E}\left[\mathrm{y}|\boldsymbol{x}\right] := \int_{-\infty}^{+\infty} y p(y|\boldsymbol{x}) dy.$$

Mean-Square Error Estimation

- In the same way, that we have already elaborated on the MSE performance of a parameter estimator, we will turn our attention to the task of regression. The more general nonlinear regression task will be considered,

$$\mathrm{y} = g(\mathbf{x}) + \eta.$$

- As a first step, the MSE optimal estimate, $\hat{y}$, given a value of $\boldsymbol{x}$ will be obtained. This is, in general, a nonlinear function of $\boldsymbol{x}$. Let, $p(y, \boldsymbol{x})$ be the corresponding joint distribution. Then, given a set of observations, $\mathbf{x} = \boldsymbol{x} \in \mathbb{R}^l$, the task is to obtain a function $\hat{y} := \hat{g}(\boldsymbol{x}) \in \mathbb{R}$, such that

$$\hat{g}(\boldsymbol{x}) = \arg\min_{f:\mathbb{R}^l \to \mathbb{R}} \mathbb{E}\left[\left(\mathrm{y} - f(\boldsymbol{x})\right)^2\right],$$

  where the expectation is taken with respect to the conditional probability of y given the value of $\boldsymbol{x}$, i.e., $p(y|\boldsymbol{x})$.

- We will show that

$$\hat{g}(\boldsymbol{x}) = \mathbb{E}\left[\mathrm{y}|\boldsymbol{x}\right] := \int_{-\infty}^{+\infty} y p(y|\boldsymbol{x}) dy.$$

Mean-Square Error Estimation

- **Proof**: We have that

$$\begin{aligned}\mathbb{E}\left[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2\right] &= \mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]+\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2\right] \\ &= \mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right]+\mathbb{E}\left[\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2\right] \\ &\quad +2\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)\right],\end{aligned}$$

  where the dependence of the expectation on $\boldsymbol{x}$ has been suppressed for notational convenience.

- It is readily seen that the last (product) term on the right hand side is zero, hence we are left with the following

$$\mathbb{E}\left[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2\right]=\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right]+\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2,$$

  where we have taken into account that, for fixed $\boldsymbol{x}$, the terms $\mathbb{E}[y|\boldsymbol{x}]$ and $f(\boldsymbol{x})$ are not random variables.

- Thus, we finally obtain our claim, i.e.,

$$\mathbb{E}[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2]\geq\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right].$$

Mean-Square Error Estimation

- **Proof**: We have that

$$\begin{aligned}\mathbb{E}\left[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2\right] &= \mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]+\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2\right] \\ &= \mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right]+\mathbb{E}\left[\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2\right] \\ &\quad +2\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)\right],\end{aligned}$$

  where the dependence of the expectation on $\boldsymbol{x}$ has been suppressed for notational convenience.

- It is readily seen that the last (product) term on the right hand side is zero, hence we are left with the following

$$\mathbb{E}\left[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2\right]=\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right]+\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2,$$

  where we have taken into account that, for fixed $\boldsymbol{x}$, the terms $\mathbb{E}[y|\boldsymbol{x}]$ and $f(\boldsymbol{x})$ are not random variables.

- Thus, we finally obtain our claim, i.e.,

$$\mathbb{E}[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2]\geq\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right].$$

Mean-Square Error Estimation

- **Proof**: We have that

$$\begin{aligned}\mathbb{E}\left[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2\right] &= \mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]+\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2\right] \\ &= \mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right]+\mathbb{E}\left[\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2\right] \\ &\quad +2\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)\right],\end{aligned}$$

  where the dependence of the expectation on $\boldsymbol{x}$ has been suppressed for notational convenience.
- It is readily seen that the last (product) term on the right hand side is zero, hence we are left with the following

$$\mathbb{E}\left[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2\right]=\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right]+\left(\mathbb{E}[\mathrm{y}|\boldsymbol{x}]-f(\boldsymbol{x})\right)^2,$$

  where we have taken into account that, for fixed $\boldsymbol{x}$, the terms $\mathbb{E}[y|\boldsymbol{x}]$ and $f(\boldsymbol{x})$ are not random variables.
- Thus, we finally obtain our claim, i.e.,

$$\mathbb{E}[\left(\mathrm{y}-f(\boldsymbol{x})\right)^2]\geq\mathbb{E}\left[\left(\mathrm{y}-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right].$$

Mean-Square Error Estimation

- The previous is a very elegant result. The optimal, in the MSE sense, estimate of the unknown function is $\hat{g}(\boldsymbol{x}) = \mathbb{E}[\mathrm{y}|\boldsymbol{x}]$. Sometimes, the latter is also known as the regression of y conditioned on $\mathbf{x} = \boldsymbol{x}$. This is, in general, a nonlinear function.
- It can be shown that if $(\mathrm{y}, \mathbf{x})$ take values in $\mathbb{R} \times \mathbb{R}^l$ and are jointly Gaussian, then the optimal MSE estimate $\mathbb{E}[\mathrm{y}|\boldsymbol{x}]$ is a linear (affine) function of $\boldsymbol{x}$.
- The previous results generalize to the case where $\mathbf{y}$ is a random vector that takes values in $\mathbb{R}^k$. The optimal MSE estimate, given the values of $\mathbf{x} = \boldsymbol{x}$, is equal to

$$\hat{\boldsymbol{g}}(\boldsymbol{x}) = \mathbb{E}[\mathbf{y}|\boldsymbol{x}],$$

where now $\hat{\boldsymbol{g}}(\boldsymbol{x}) \in \mathbb{R}^k$. Moreover, if $(\mathbf{y}, \mathbf{x})$ are jointly Gaussian random vectors, the MSE optimal estimate is also an affine function of $\boldsymbol{x}$.

Mean-Square Error Estimation

- The previous is a very elegant result. The optimal, in the MSE sense, estimate of the unknown function is $\hat{g}(\boldsymbol{x}) = \mathbb{E}[\mathrm{y}|\boldsymbol{x}]$. Sometimes, the latter is also known as the regression of y conditioned on $\mathbf{x} = \boldsymbol{x}$. This is, in general, a nonlinear function.
- It can be shown that if $(\mathrm{y}, \mathbf{x})$ take values in $\mathbb{R} \times \mathbb{R}^l$ and are jointly Gaussian, then the optimal MSE estimate $\mathbb{E}[\mathrm{y}|\boldsymbol{x}]$ is a linear (affine) function of $\boldsymbol{x}$.
- The previous results generalize to the case where $\mathbf{y}$ is a random vector that takes values in $\mathbb{R}^k$. The optimal MSE estimate, given the values of $\mathbf{x} = \boldsymbol{x}$, is equal to

$$\hat{\boldsymbol{g}}(\boldsymbol{x}) = \mathbb{E}[\mathbf{y}|\boldsymbol{x}],$$

where now $\hat{\boldsymbol{g}}(\boldsymbol{x}) \in \mathbb{R}^k$. Moreover, if $(\mathbf{y}, \mathbf{x})$ are jointly Gaussian random vectors, the MSE optimal estimate is also an affine function of $\boldsymbol{x}$.

Mean-Square Error Estimation

- The previous is a very elegant result. The optimal, in the MSE sense, estimate of the unknown function is $\hat{g}(\boldsymbol{x}) = \mathbb{E}[\mathrm{y}|\boldsymbol{x}]$. Sometimes, the latter is also known as the regression of y conditioned on $\mathbf{x} = \boldsymbol{x}$. This is, in general, a nonlinear function.
- It can be shown that if $(\mathrm{y}, \mathbf{x})$ take values in $\mathbb{R} \times \mathbb{R}^l$ and are jointly Gaussian, then the optimal MSE estimate $\mathbb{E}[\mathrm{y}|\boldsymbol{x}]$ is a linear (affine) function of $\boldsymbol{x}$.
- The previous results generalize to the case where $\mathbf{y}$ is a random vector that takes values in $\mathbb{R}^k$. The optimal MSE estimate, given the values of $\mathbf{x} = \boldsymbol{x}$, is equal to

$$\hat{\boldsymbol{g}}(\boldsymbol{x}) = \mathbb{E}[\mathbf{y}|\boldsymbol{x}],$$

where now $\hat{\boldsymbol{g}}(\boldsymbol{x}) \in \mathbb{R}^k$. Moreover, if $(\mathbf{y}, \mathbf{x})$ are jointly Gaussian random vectors, the MSE optimal estimate is also an affine function of $\boldsymbol{x}$.

Mean-Square Error Estimation

- The previous findings can be fully justified by physical reasoning. Assume, for simplicity, that the noise variable is of zero mean. Then, for a fixed value $\mathbf{x} = \boldsymbol{x}$, we have that $\mathbb{E}[\mathrm{y}|\boldsymbol{x}] = g(\boldsymbol{x})$ and the respective MSE is equal to

$$\mathrm{MSE} = \mathbb{E}\left[\left(\mathrm{y} - \mathbb{E}[\mathrm{y}|\boldsymbol{x}]\right)^2\right] = \sigma_\eta^2.$$

  No other function of $\boldsymbol{x}$ can do better, since the optimal one achieves an MSE equal to the noise variance, which is irreducible; it represents the intrinsic uncertainty of the system. Any other function, $f(\boldsymbol{x})$, will result in an MSE larger by the factor $(\mathbb{E}[\mathrm{y}|\boldsymbol{x}] - f(\boldsymbol{x}))^2$, which corresponds to the deviation of the MSE from the optimal one.

Bias-Variance Tradeoff

- The optimal, in the MSE sense, estimate of the dependent variable in a regression task is given by the conditional expectation $\mathbb{E}[\mathrm{y}|\boldsymbol{x}]$.
- In practice, any estimator is computed based on a specific training data set, say $\mathcal{D}$. Let us make the dependence on the training set explicit and express the estimate as a function of $\boldsymbol{x}$ parameterized on $\mathcal{D}$, i.e., $f(\boldsymbol{x};\mathcal{D})$. A reasonable measure to quantify the performance of an estimator is its mean-square deviation from the optimal one, i.e., $\mathbb{E}_{\mathcal{D}}[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\big)^2]$, where the mean is taken with respect to all possible training sets, since each one results in a different estimate.
- Adding and subtracting the mean, as we did before for the case of a single parameter, the following elegant formula is obtained

$$\mathbb{E}_{\mathcal{D}}\left[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\big)^2\right] = \underbrace{\mathbb{E}_{\mathcal{D}}\left[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x};\mathcal{D})]\big)^2\right]}_{\text{Variance}} + \underbrace{\Big(\mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x};\mathcal{D})]-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\Big)^2}_{\text{Bias}^2}.$$

Bias-Variance Tradeoff

- The optimal, in the MSE sense, estimate of the dependent variable in a regression task is given by the conditional expectation $\mathbb{E}[\mathrm{y}|\boldsymbol{x}]$.
- In practice, any estimator is computed based on a specific training data set, say $\mathcal{D}$. Let us make the dependence on the training set explicit and express the estimate as a function of $\boldsymbol{x}$ parameterized on $\mathcal{D}$, i.e., $f(\boldsymbol{x};\mathcal{D})$. A reasonable measure to quantify the performance of an estimator is its mean-square deviation from the optimal one, i.e., $\mathbb{E}_{\mathcal{D}}[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\big)^2]$, where the mean is taken with respect to all possible training sets, since each one results in a different estimate.
- Adding and subtracting the mean, as we did before for the case of a single parameter, the following elegant formula is obtained

$$\mathbb{E}_{\mathcal{D}}\left[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\big)^2\right] = \underbrace{\mathbb{E}_{\mathcal{D}}\left[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x};\mathcal{D})]\big)^2\right]}_{\text{Variance}} + \underbrace{\Big(\mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x};\mathcal{D})]-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\Big)^2}_{\text{Bias}^2}.$$

Bias-Variance Tradeoff

- The optimal, in the MSE sense, estimate of the dependent variable in a regression task is given by the conditional expectation $\mathbb{E}[\mathrm{y}|\boldsymbol{x}]$.
- In practice, any estimator is computed based on a specific training data set, say $\mathcal{D}$. Let us make the dependence on the training set explicit and express the estimate as a function of $\boldsymbol{x}$ parameterized on $\mathcal{D}$, i.e., $f(\boldsymbol{x};\mathcal{D})$. A reasonable measure to quantify the performance of an estimator is its mean-square deviation from the optimal one, i.e., $\mathbb{E}_{\mathcal{D}}[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\big)^2]$, where the mean is taken with respect to all possible training sets, since each one results in a different estimate.
- Adding and subtracting the mean, as we did before for the case of a single parameter, the following elegant formula is obtained

$$\mathbb{E}_{\mathcal{D}}\left[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\big)^2\right] = \underbrace{\mathbb{E}_{\mathcal{D}}\left[\big(f(\boldsymbol{x};\mathcal{D})-\mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x};\mathcal{D})]\big)^2\right]}_{\text{Variance}} + \underbrace{\Big(\mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x};\mathcal{D})]-\mathbb{E}[\mathrm{y}|\boldsymbol{x}]\Big)^2}_{\text{Bias}^2}.$$

Bias-Variance Tradeoff

- As it was the case for the MSE parameter estimation task when changing from one training set to another, the mean-square deviation from the optimal estimate comprises two terms. The first one is contributed by the variance of the estimator around its own mean value and the second one by the difference of the mean from the optimal estimate, i.e., the bias.
- It turns out that one cannot make **both** terms small **simultaneously**. For a fixed number of training points, N, in the data sets $\mathcal{D}$, trying to minimize the variance term results in an increase of the bias term and vice versa.
- This is because, in order to reduce the bias term, one has to increase the complexity (more free parameters) of the adopted estimator $f(\cdot; \mathcal{D})$. This, in turn, results in higher variance as we change the training sets. This is a manifestation of the overfitting issue that we have already discussed.
- The only way to reduce both terms simultaneously is to increase the number of the training data points, N, and at the same time to increase the complexity of the model **carefully**, so that to achieve the aforementioned goal. This is known as the bias-variance dilemma or tradeoff.

Bias-Variance Tradeoff

- As it was the case for the MSE parameter estimation task when changing from one training set to another, the mean-square deviation from the optimal estimate comprises two terms. The first one is contributed by the variance of the estimator around its own mean value and the second one by the difference of the mean from the optimal estimate, i.e., the bias.
- It turns out that one cannot make **both** terms small **simultaneously**. For a fixed number of training points, N, in the data sets $\mathcal{D}$, trying to minimize the variance term results in an increase of the bias term and vice versa.
- This is because, in order to reduce the bias term, one has to increase the complexity (more free parameters) of the adopted estimator $f(\cdot; \mathcal{D})$. This, in turn, results in higher variance as we change the training sets. This is a manifestation of the overfitting issue that we have already discussed.
- The only way to reduce both terms simultaneously is to increase the number of the training data points, N, and at the same time to increase the complexity of the model **carefully**, so that to achieve the aforementioned goal. This is known as the bias-variance dilemma or tradeoff.

Bias-Variance Tradeoff

- As it was the case for the MSE parameter estimation task when changing from one training set to another, the mean-square deviation from the optimal estimate comprises two terms. The first one is contributed by the variance of the estimator around its own mean value and the second one by the difference of the mean from the optimal estimate, i.e., the bias.
- It turns out that one cannot make **both** terms small **simultaneously**. For a fixed number of training points, N, in the data sets $\mathcal{D}$, trying to minimize the variance term results in an increase of the bias term and vice versa.
- This is because, in order to reduce the bias term, one has to increase the complexity (more free parameters) of the adopted estimator $f(\cdot;\mathcal{D})$. This, in turn, results in higher variance as we change the training sets. This is a manifestation of the overfitting issue that we have already discussed.
- The only way to reduce both terms simultaneously is to increase the number of the training data points, N, and at the same time to increase the complexity of the model **carefully**, so that to achieve the aforementioned goal. This is known as the bias-variance dilemma or tradeoff.

Bias-Variance Tradeoff

- As it was the case for the MSE parameter estimation task when changing from one training set to another, the mean-square deviation from the optimal estimate comprises two terms. The first one is contributed by the variance of the estimator around its own mean value and the second one by the difference of the mean from the optimal estimate, i.e., the bias.
- It turns out that one cannot make **both** terms small **simultaneously**. For a fixed number of training points, N, in the data sets $\mathcal{D}$, trying to minimize the variance term results in an increase of the bias term and vice versa.
- This is because, in order to reduce the bias term, one has to increase the complexity (more free parameters) of the adopted estimator $f(\cdot;\mathcal{D})$. This, in turn, results in higher variance as we change the training sets. This is a manifestation of the overfitting issue that we have already discussed.
- The only way to reduce both terms simultaneously is to increase the number of the training data points, N, and at the same time to increase the complexity of the model **carefully**, so that to achieve the aforementioned goal. This is known as the bias-variance dilemma or tradeoff.

Occam's Razor Rule

- The bias-variance dilemma is a manifestation of a more general statement in machine learning/inverse problem tasks, known as the Occam's razor rule:

  Plurality must never be posited without necessity

- The great physicist Paul Dirac expressed the same statement from an aesthetics point of view, which underlies mathematical theories: A theory with a mathematical beauty is more likely to be correct than an ugly one that fits the data. In our context of model selection, this is understood that one has to select the simplest model that can **explain** the data. Although this is not a scientifically proven result, it underlies the rationale behind a number of developed model selection techniques.

Occam's Razor Rule

- The bias-variance dilemma is a manifestation of a more general statement in machine learning/inverse problem tasks, known as the Occam's razor rule:

  Plurality must never be posited without necessity

- The great physicist Paul Dirac expressed the same statement from an aesthetics point of view, which underlies mathematical theories: A theory with a mathematical beauty is more likely to be correct than an ugly one that fits the data. In our context of model selection, this is understood that one has to select the simplest model that can **explain** the data. Although this is not a scientifically proven result, it underlies the rationale behind a number of developed model selection techniques.

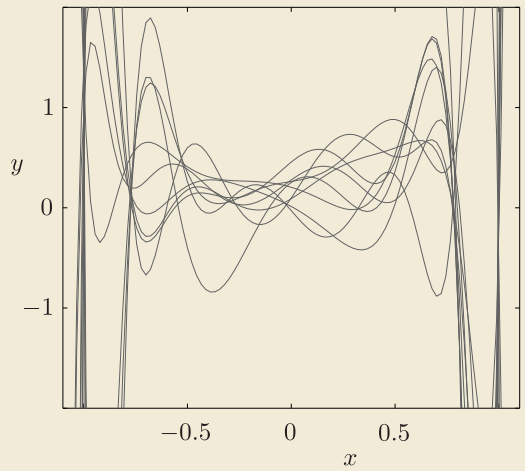
Occam's Razor Rule

- The bias-variance dilemma is a manifestation of a more general statement in machine learning/inverse problem tasks, known as the Occam's razor rule:

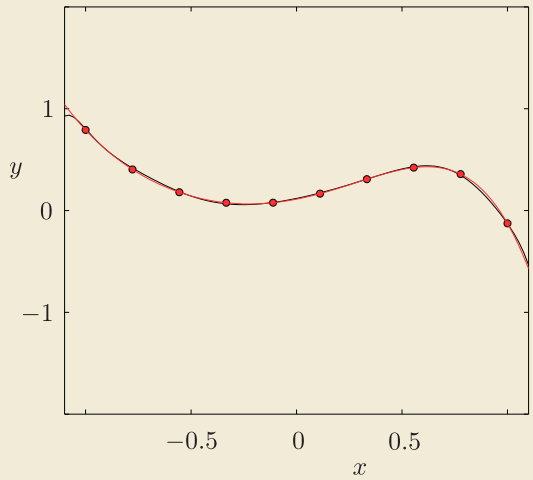
  Plurality must never be posited without necessity

- The great physicist Paul Dirac expressed the same statement from an aesthetics point of view, which underlies mathematical theories: A theory with a mathematical beauty is more likely to be correct than an ugly one that fits the data. In our context of model selection, this is understood that one has to select the simplest model that can **explain** the data. Although this is not a scientifically proven result, it underlies the rationale behind a number of developed model selection techniques.

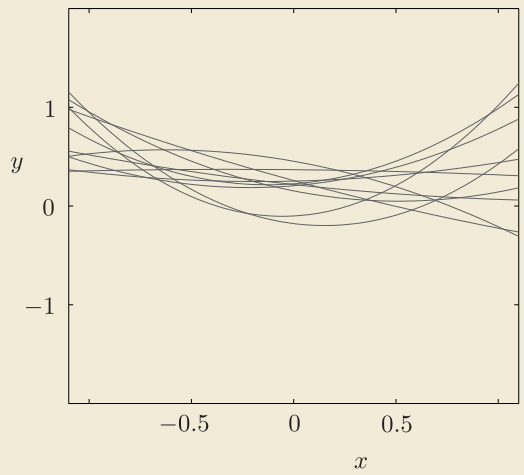
Bias-Variance Dilemma Example



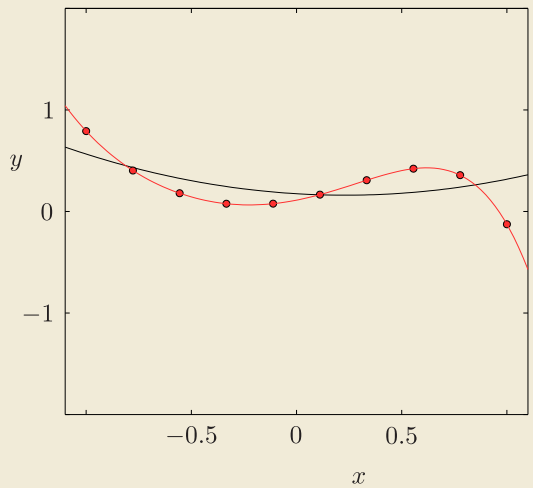
(a)



(b)



(c)



(d)

(a) Ten of the resulting curves from fitting a high order polynomial and (b) the corresponding average over 1000 different experiments, together with the (red) curve of the unknown polynomial. The dots indicate the points that give birth to the training data, as described in the text. (c) and (d) illustrate the results from fitting a low order polynomial. Observe the bias-variance tradeoff as a function of the complexity of the fitted model.

Maximum Likelihood Method

- From now on, we are going to bring into the parameter estimation task information related to the statistical nature of the training data set. We will first formulate the method in a general parameter estimation framework. In the sequel, we are going to apply the methods to specific machine learning related tasks.
- We are given a set of say, N, observations, $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, drawn from a probability distribution. We assume that the joint pdf of these N observations is of a known parametric functional type, denoted as $p(\mathcal{X}; \boldsymbol{\theta})$, where the parameter $\boldsymbol{\theta} \in \mathbb{R}^K$ is unknown and the task is to estimate its value. This joint pdf is known as the likelihood function of $\boldsymbol{\theta}$ with respect to the given set of observations, $\mathcal{X}$. According to the maximum likelihood method, the estimate is provided by:

$$\hat{\boldsymbol{\theta}}_{\text{ML}} := \arg\max_{\boldsymbol{\theta}} p(\mathcal{X}; \boldsymbol{\theta}).$$

- Since the logarithmic function, $\ln(\cdot)$, is monotone and increasing, one can instead search for the maximum of the log-likelihood function, i.e.,

$$\left.\frac{\partial \ln p(\mathcal{X}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{\text{ML}}} = \mathbf{0}.$$

Maximum Likelihood Method

- From now on, we are going to bring into the parameter estimation task information related to the statistical nature of the training data set. We will first formulate the method in a general parameter estimation framework. In the sequel, we are going to apply the methods to specific machine learning related tasks.
- We are given a set of say, N, observations, $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, drawn from a probability distribution. We assume that the joint pdf of these N observations is of a known parametric functional type, denoted as $p(\mathcal{X}; \boldsymbol{\theta})$, where the parameter $\boldsymbol{\theta} \in \mathbb{R}^K$ is unknown and the task is to estimate its value. This joint pdf is known as the likelihood function of $\boldsymbol{\theta}$ with respect to the given set of observations, $\mathcal{X}$. According to the maximum likelihood method, the estimate is provided by:

$$\hat{\boldsymbol{\theta}}_{\text{ML}} := \arg\max_{\boldsymbol{\theta}} p(\mathcal{X}; \boldsymbol{\theta}).$$

- Since the logarithmic function, $\ln(\cdot)$, is monotone and increasing, one can instead search for the maximum of the log-likelihood function, i.e.,

$$\left.\frac{\partial \ln p(\mathcal{X}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{\text{ML}}} = \mathbf{0}.$$

Maximum Likelihood Method

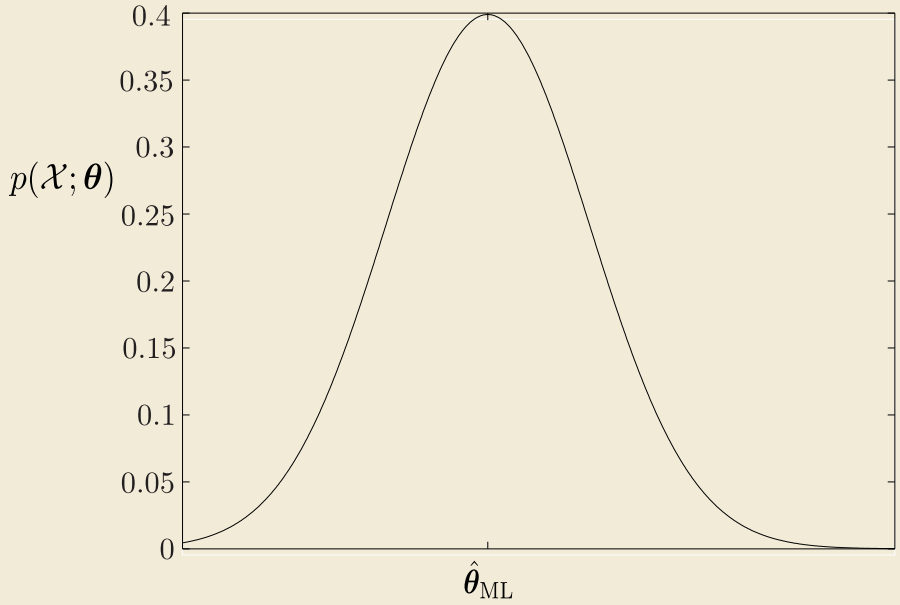
- From now on, we are going to bring into the parameter estimation task information related to the statistical nature of the training data set. We will first formulate the method in a general parameter estimation framework. In the sequel, we are going to apply the methods to specific machine learning related tasks.
- We are given a set of say, N, observations, $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, drawn from a probability distribution. We assume that the joint pdf of these N observations is of a known parametric functional type, denoted as $p(\mathcal{X}; \boldsymbol{\theta})$, where the parameter $\boldsymbol{\theta} \in \mathbb{R}^K$ is unknown and the task is to estimate its value. This joint pdf is known as the likelihood function of $\boldsymbol{\theta}$ with respect to the given set of observations, $\mathcal{X}$. According to the maximum likelihood method, the estimate is provided by:

$$\hat{\boldsymbol{\theta}}_{\text{ML}} := \arg\max_{\boldsymbol{\theta}} p(\mathcal{X}; \boldsymbol{\theta}).$$

- Since the logarithmic function, $\ln(\cdot)$, is monotone and increasing, one can instead search for the maximum of the log-likelihood function, i.e.,

$$\left.\frac{\partial \ln p(\mathcal{X}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{\text{ML}}} = \mathbf{0}.$$

Maximum Likelihood Method



- The ML estimator is **asymptotically** unbiased; that is, assuming that the model of the pdf, which we have adopted, is correct and there exists a true parameter $\boldsymbol{\theta}_o$, then

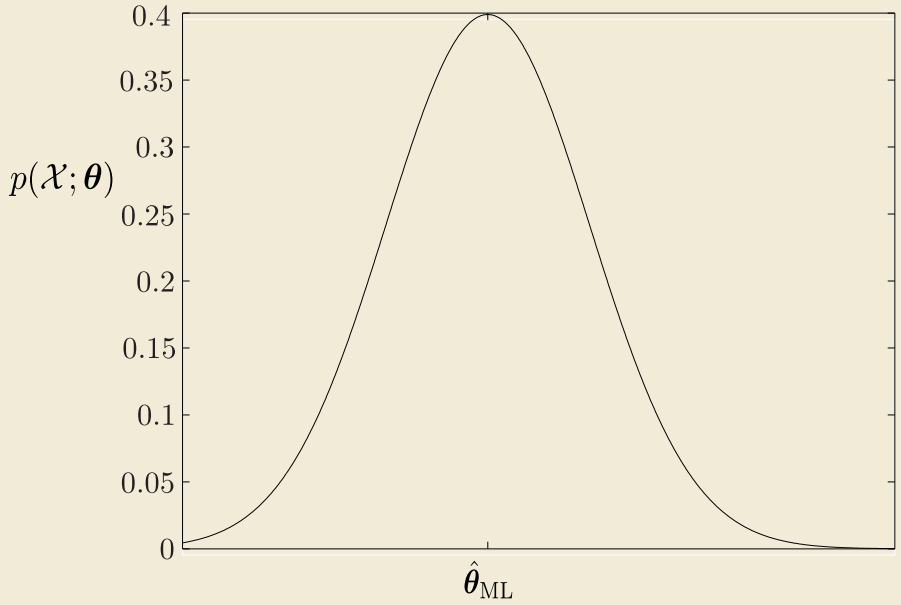
$$\lim_{N\to\infty} \mathbb{E}[\hat{\theta}_{ML}] = \boldsymbol{\theta}_o.$$

- The ML estimate is **asymptotically** consistent; that is, given any value of $\epsilon > 0$,

$$\lim_{N\to\infty} \text{Prob}\left\{\left|\hat{\theta}_{\text{ML}} - \boldsymbol{\theta}_o\right| > \epsilon\right\} = 0,$$

- The ML estimator is **asymptotically** efficient; that is, it achieves the Cramér-Rao lower bound.

Maximum Likelihood Method



- The ML estimator is **asymptotically** unbiased; that is, assuming that the model of the pdf, which we have adopted, is correct and there exists a true parameter $\boldsymbol{\theta}_o$, then

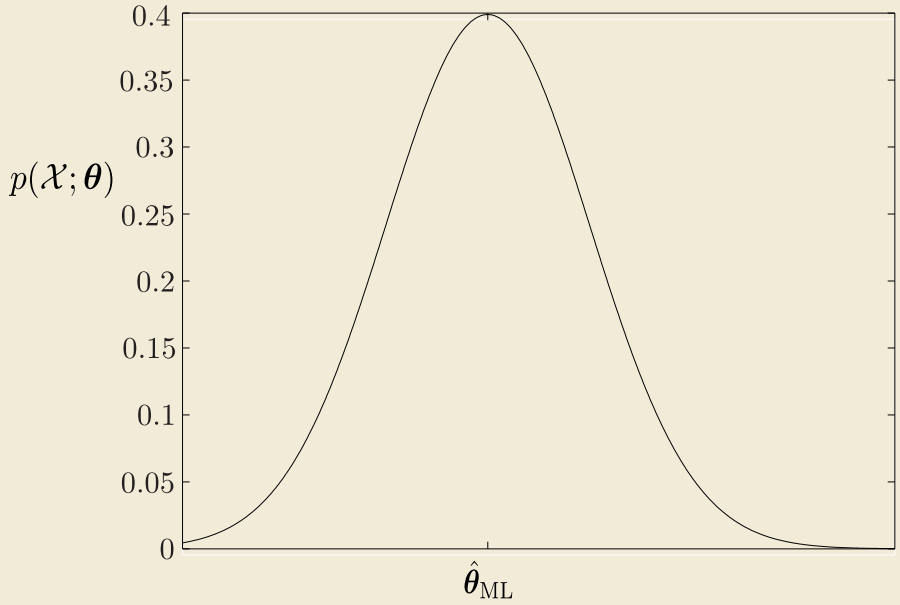
$$\lim_{N\to\infty} \mathbb{E}[\hat{\theta}_{ML}] = \boldsymbol{\theta}_o.$$

- The ML estimate is **asymptotically** consistent; that is, given any value of $\epsilon > 0$,

$$\lim_{N\to\infty} \text{Prob}\left\{\left|\hat{\theta}_{\text{ML}} - \boldsymbol{\theta}_o\right| > \epsilon\right\} = 0,$$

- The ML estimator is **asymptotically** efficient; that is, it achieves the Cramér-Rao lower bound.

Maximum Likelihood Method



- The ML estimator is **asymptotically** unbiased; that is, assuming that the model of the pdf, which we have adopted, is correct and there exists a true parameter $\boldsymbol{\theta}_o$, then

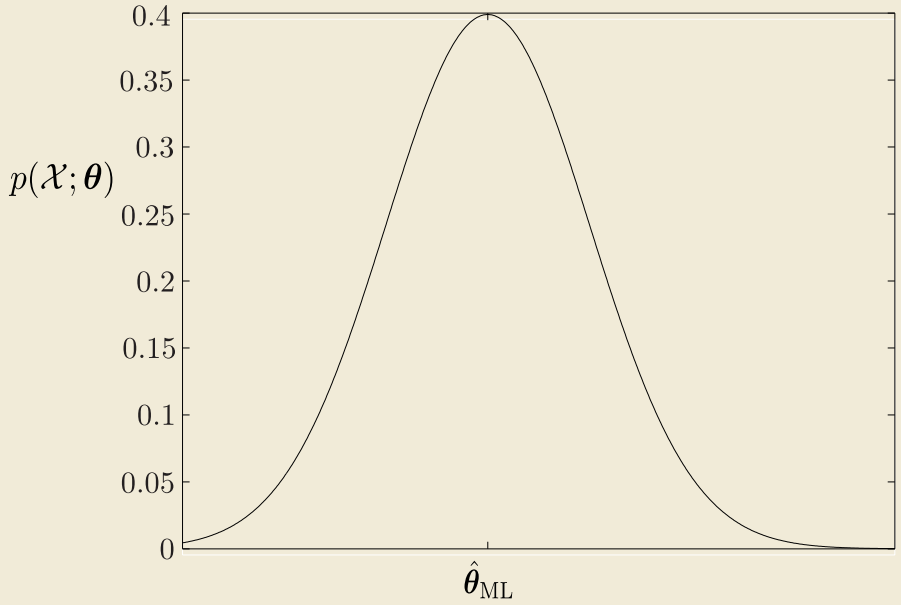
$$\lim_{N\to\infty} \mathbb{E}[\hat{\boldsymbol{\theta}}_{ML}] = \boldsymbol{\theta}_o.$$

- The ML estimate is **asymptotically** consistent; that is, given any value of $\epsilon > 0$,

$$\lim_{N\to\infty} \text{Prob}\left\{\left|\hat{\boldsymbol{\theta}}_{\text{ML}} - \boldsymbol{\theta}_o\right| > \epsilon\right\} = 0,$$

- The ML estimator is **asymptotically** efficient; that is, it achieves the Cramér-Rao lower bound.

Maximum Likelihood Method



- The ML estimator is **asymptotically** unbiased; that is, assuming that the model of the pdf, which we have adopted, is correct and there exists a true parameter $\boldsymbol{\theta}_o$, then

$$\lim_{N\to\infty} \mathbb{E}[\hat{\boldsymbol{\theta}}_{ML}] = \boldsymbol{\theta}_o.$$

- The ML estimate is **asymptotically** consistent; that is, given any value of $\epsilon > 0$,

$$\lim_{N\to\infty} \text{Prob}\left\{\left|\hat{\boldsymbol{\theta}}_{\text{ML}} - \boldsymbol{\theta}_o\right| > \epsilon\right\} = 0,$$

- The ML estimator is **asymptotically** efficient; that is, it achieves the Cramér-Rao lower bound.

ML Estimation of the The Mean Value

- Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ be the observation vectors i.i.d drawn from a normal distribution with known covariance matrix and unknown mean, that is,

$$p(\boldsymbol{x}_n; \boldsymbol{\mu}) = \frac{1}{(2\pi)^{l/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu})^T \Sigma^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu})\right).$$

- The joint log-likelihood function is given by

$$L(\boldsymbol{\mu}) = \ln \prod_{n=1}^{N} p(\boldsymbol{x}_n; \boldsymbol{\mu}) = -\frac{N}{2} \ln\left((2\pi)^l |\Sigma|\right) - \frac{1}{2} \sum_{n=1}^{N} (\boldsymbol{x}_n - \boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu})$$

- Taking the gradient with respect to $\boldsymbol{\mu}$, we obtain

$$\frac{\partial L(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}} := \begin{bmatrix} \frac{\partial L}{\partial \mu_1} \\ \frac{\partial L}{\partial \mu_2} \\ \vdots \\ \frac{\partial L}{\partial \mu_l} \end{bmatrix} = \sum_{n=1}^{N} \Sigma^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}),$$

and equating to $\mathbf{0}$ leads to

$$\hat{\boldsymbol{\mu}}_{\mathrm{ML}} = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n.$$

ML Estimation of the The Mean Value

- Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ be the observation vectors i.i.d drawn from a normal distribution with known covariance matrix and unknown mean, that is,

$$p(\boldsymbol{x}_n; \boldsymbol{\mu}) = \frac{1}{(2\pi)^{l/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu})^T \Sigma^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu})\right).$$

- The joint log-likelihood function is given by

$$L(\boldsymbol{\mu}) = \ln \prod_{n=1}^{N} p(\boldsymbol{x}_n; \boldsymbol{\mu}) = -\frac{N}{2} \ln\left((2\pi)^l |\Sigma|\right) - \frac{1}{2} \sum_{n=1}^{N} (\boldsymbol{x}_n - \boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu})$$

- Taking the gradient with respect to $\boldsymbol{\mu}$, we obtain

$$\frac{\partial L(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}} := \begin{bmatrix} \frac{\partial L}{\partial \mu_1} \\ \frac{\partial L}{\partial \mu_2} \\ \vdots \\ \frac{\partial L}{\partial \mu_l} \end{bmatrix} = \sum_{n=1}^{N} \Sigma^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}),$$

and equating to $\mathbf{0}$ leads to

$$\hat{\boldsymbol{\mu}}_{\text{ML}} = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n.$$

ML Estimation of the The Mean Value

- Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ be the observation vectors i.i.d drawn from a normal distribution with known covariance matrix and unknown mean, that is,

$$p(\boldsymbol{x}_n; \boldsymbol{\mu}) = \frac{1}{(2\pi)^{l/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_n - \boldsymbol{\mu})^T \Sigma^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu})\right).$$

- The joint log-likelihood function is given by

$$L(\boldsymbol{\mu}) = \ln \prod_{n=1}^{N} p(\boldsymbol{x}_n; \boldsymbol{\mu}) = -\frac{N}{2} \ln\left((2\pi)^l |\Sigma|\right) - \frac{1}{2} \sum_{n=1}^{N} (\boldsymbol{x}_n - \boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu})$$

- Taking the gradient with respect to $\boldsymbol{\mu}$, we obtain

$$\frac{\partial L(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}} := \begin{bmatrix} \frac{\partial L}{\partial \mu_1} \\ \frac{\partial L}{\partial \mu_2} \\ \vdots \\ \frac{\partial L}{\partial \mu_l} \end{bmatrix} = \sum_{n=1}^{N} \Sigma^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}),$$

and equating to $\mathbf{0}$ leads to

$$\hat{\boldsymbol{\mu}}_{\mathsf{ML}} = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n.$$

ML and Linear Regression: The Non-white Gaussian Noise Case

- Consider the linear regression model

$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta.$$

We are given N training data points $(y_n, \boldsymbol{x}_n)$, $n = 1, 2 \ldots, N$. The noise samples, η_n, $n = 1, \ldots, N$, originate from a jointly Gaussian distribution with zero mean and covariance matrix Σ_η. Our goal is to obtain the ML estimate of $\boldsymbol{\theta}$.

- The joint log-likelihood function of $\boldsymbol{\theta}$, with respect to the training set, is given by

$$L(\boldsymbol{\theta}) = -\frac{N}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_\eta| - \frac{1}{2}(\boldsymbol{y} - X\boldsymbol{\theta})^T \Sigma_\eta^{-1}(\boldsymbol{y} - X\boldsymbol{\theta}),$$

where $\boldsymbol{y} := [y_1, y_2, \ldots, y_N]^T$, and $X := [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^T$ stands for the input matrix.

- Taking the gradient with respect to $\boldsymbol{\theta}$, we get

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = X^T \Sigma_\eta^{-1}(\boldsymbol{y} - X\boldsymbol{\theta}),$$

and equating to the zero vector, we obtain

$$\hat{\boldsymbol{\theta}}_{\mathrm{ML}} = \left(X^T \Sigma_\eta^{-1} X\right)^{-1} X^T \Sigma_\eta^{-1} \boldsymbol{y}.$$

ML and Linear Regression: The Non-white Gaussian Noise Case

- Consider the linear regression model

$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta.$$

We are given N training data points $(y_n, \boldsymbol{x}_n)$, $n = 1, 2 \ldots, N$. The noise samples, η_n, $n = 1, \ldots, N$, originate from a jointly Gaussian distribution with zero mean and covariance matrix Σ_η. Our goal is to obtain the ML estimate of $\boldsymbol{\theta}$.

- The joint log-likelihood function of $\boldsymbol{\theta}$, with respect to the training set, is given by

$$L(\boldsymbol{\theta}) = -\frac{N}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_\eta| - \frac{1}{2}(\boldsymbol{y} - X\boldsymbol{\theta})^T \Sigma_\eta^{-1}(\boldsymbol{y} - X\boldsymbol{\theta}),$$

where $\boldsymbol{y} := [y_1, y_2, \ldots, y_N]^T$, and $X := [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^T$ stands for the input matrix.

- Taking the gradient with respect to $\boldsymbol{\theta}$, we get

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = X^T \Sigma_\eta^{-1}(\boldsymbol{y} - X\boldsymbol{\theta}),$$

and equating to the zero vector, we obtain

$$\hat{\boldsymbol{\theta}}_{\mathrm{ML}} = \left(X^T \Sigma_\eta^{-1} X\right)^{-1} X^T \Sigma_\eta^{-1} \boldsymbol{y}.$$

ML and Linear Regression: The Non-white Gaussian Noise Case

- Consider the linear regression model

$$\mathrm{y} = \boldsymbol{\theta}^T \mathbf{x} + \eta.$$

We are given N training data points $(y_n, \boldsymbol{x}_n)$, $n = 1, 2 \ldots, N$. The noise samples, η_n, $n = 1, \ldots, N$, originate from a jointly Gaussian distribution with zero mean and covariance matrix Σ_η. Our goal is to obtain the ML estimate of $\boldsymbol{\theta}$.

- The joint log-likelihood function of $\boldsymbol{\theta}$, with respect to the training set, is given by

$$L(\boldsymbol{\theta}) = -\frac{N}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_\eta| - \frac{1}{2}(\boldsymbol{y} - X\boldsymbol{\theta})^T \Sigma_\eta^{-1}(\boldsymbol{y} - X\boldsymbol{\theta}),$$

where $\boldsymbol{y} := [y_1, y_2, \ldots, y_N]^T$, and $X := [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^T$ stands for the input matrix.

- Taking the gradient with respect to $\boldsymbol{\theta}$, we get

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = X^T \Sigma_\eta^{-1}(\boldsymbol{y} - X\boldsymbol{\theta}),$$

and equating to the zero vector, we obtain

$$\hat{\boldsymbol{\theta}}_{\mathsf{ML}} = \left(X^T \Sigma_\eta^{-1} X\right)^{-1} X^T \Sigma_\eta^{-1} \boldsymbol{y}.$$

ML and Linear Regression: The Non-white Gaussian Noise Case

- Compare the previously derived ML with the LS solution. They are different, unless the covariance matrix of the successive noise samples, Σ_η, is diagonal and of the form $\sigma_\eta^2 I$; that is, if the noise is Gaussian as well as white. In this case, the LS and the ML solutions coincide. However, if the noise sequence is non-white, the two estimates differ. Moreover, it can be shown that, in the case of colored Gaussian noise, the ML estimate is an efficient one and it attains the Cramér-Rao bound, even if N is finite.

Bayesian Inference

- In our discussion, so far, we have assumed that the parameters associated with the functional form of the adopted model are deterministic constants, whose values are unknown to us. Now, we turn our attention to a different rationale. The unknown parameters will be treated as random variables. Hence, whenever our goal is to estimate their values, this is conceived as an effort to estimate the values of a specific realization that corresponds to the observed data. The heart of the method beats around the celebrated Bayes theorem.
- Given two jointly distributed random vectors, say, $\mathbf{x}$, θ, Bayes theorem states that
$$p(\boldsymbol{x}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{x})p(\boldsymbol{x}).$$
- Assume that $\mathbf{x}, \theta$ are two statistically dependent random vectors. Let $\mathcal{X} = \{\boldsymbol{x}_n \in \mathbb{R}^l, n = 1, 2, \ldots, N\}$, be the set of the observations resulting from N successive experiments. Then, Bayes theorem gives
$$p(\boldsymbol{\theta}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{X})} = \frac{p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}.$$
- Obviously, if the observations are i.i.d., then we can write
$$p(\mathcal{X}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\theta}).$$

Bayesian Inference

- In our discussion, so far, we have assumed that the parameters associated with the functional form of the adopted model are deterministic constants, whose values are unknown to us. Now, we turn our attention to a different rationale. The unknown parameters will be treated as random variables. Hence, whenever our goal is to estimate their values, this is conceived as an effort to estimate the values of a specific realization that corresponds to the observed data. The heart of the method beats around the celebrated Bayes theorem.
- Given two jointly distributed random vectors, say, $\mathbf{x}$, $\boldsymbol{\theta}$, Bayes theorem states that

$$p(\boldsymbol{x}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{x})p(\boldsymbol{x}).$$

- Assume that $\mathbf{x}, \boldsymbol{\theta}$ are two statistically dependent random vectors. Let $\mathcal{X} = \{\boldsymbol{x}_n \in \mathbb{R}^l, n = 1, 2, \ldots, N\}$, be the set of the observations resulting from N successive experiments. Then, Bayes theorem gives

$$p(\boldsymbol{\theta}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{X})} = \frac{p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}.$$

- Obviously, if the observations are i.i.d., then we can write

$$p(\mathcal{X}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\theta}).$$

Bayesian Inference

- In our discussion, so far, we have assumed that the parameters associated with the functional form of the adopted model are deterministic constants, whose values are unknown to us. Now, we turn our attention to a different rationale. The unknown parameters will be treated as random variables. Hence, whenever our goal is to estimate their values, this is conceived as an effort to estimate the values of a specific realization that corresponds to the observed data. The heart of the method beats around the celebrated Bayes theorem.
- Given two jointly distributed random vectors, say, $\mathbf{x}$, $\boldsymbol{\theta}$, Bayes theorem states that
$$p(\boldsymbol{x}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{x})p(\boldsymbol{x}).$$
- Assume that $\mathbf{x}, \boldsymbol{\theta}$ are two statistically dependent random vectors. Let $\mathcal{X} = \{\boldsymbol{x}_n \in \mathbb{R}^l, n = 1, 2, \ldots, N\}$, be the set of the observations resulting from N successive experiments. Then, Bayes theorem gives
$$p(\boldsymbol{\theta}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{X})} = \frac{p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}.$$
- Obviously, if the observations are i.i.d., then we can write
$$p(\mathcal{X}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\theta}).$$

Bayesian Inference

- In our discussion, so far, we have assumed that the parameters associated with the functional form of the adopted model are deterministic constants, whose values are unknown to us. Now, we turn our attention to a different rationale. The unknown parameters will be treated as random variables. Hence, whenever our goal is to estimate their values, this is conceived as an effort to estimate the values of a specific realization that corresponds to the observed data. The heart of the method beats around the celebrated Bayes theorem.
- Given two jointly distributed random vectors, say, $\mathbf{x}$, θ, Bayes theorem states that
$$p(\boldsymbol{x}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{x})p(\boldsymbol{x}).$$
- Assume that $\mathbf{x}, \theta$ are two statistically dependent random vectors. Let $\mathcal{X} = \{\boldsymbol{x}_n \in \mathbb{R}^l, n = 1, 2, \ldots, N\}$, be the set of the observations resulting from N successive experiments. Then, Bayes theorem gives
$$p(\boldsymbol{\theta}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{X})} = \frac{p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}.$$
- Obviously, if the observations are i.i.d., then we can write
$$p(\mathcal{X}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\theta}).$$

Bayesian Inference

- In the previous formulas, $p(\boldsymbol{\theta})$ is the a-priori pdf concerning the statistical distribution of θ, and $p(\boldsymbol{\theta}|\mathcal{X})$ is the conditional or a-posteriori pdf, formed after the set of N observations has been obtained. The prior probability density, $p(\boldsymbol{\theta})$, can be considered as a constraint that encapsulates our prior knowledge about θ. No doubt, our uncertainty about θ is modified after the observations have been received, since more information is now disclosed to us.
- We will refer to the process of approximating the pdf of a random quantity, based on a set of training data, as inference, to differentiate it from the process of estimation, that returns a single value for each parameter/variable. So, according to the inference approach, one attempts to draw conclusions about the nature of the randomness that underlies the variables of interest. This information, can in turn be used to make predictions and/or to take decisions.

Bayesian Inference

- In the previous formulas, $p(\boldsymbol{\theta})$ is the a-priori pdf concerning the statistical distribution of θ, and $p(\boldsymbol{\theta}|\mathcal{X})$ is the conditional or a-posteriori pdf, formed after the set of N observations has been obtained. The prior probability density, $p(\boldsymbol{\theta})$, can be considered as a constraint that encapsulates our prior knowledge about θ. No doubt, our uncertainty about θ is modified after the observations have been received, since more information is now disclosed to us.
- We will refer to the process of approximating the pdf of a random quantity, based on a set of training data, as inference, to differentiate it from the process of estimation, that returns a single value for each parameter/variable. So, according to the inference approach, one attempts to draw conclusions about the nature of the randomness that underlies the variables of interest. This information, can in turn be used to make predictions and/or to take decisions.

Bayesian Inference

- A first path to exploit the derived posterior pdf is to obtain a single estimate concerning the unknown parameter vector. One possibility is to make use of what we already know. Since $\mathbf{x}$ and $\boldsymbol{\theta}$ are two statistically dependent random vectors, the MSE optimal estimate of the value of $\boldsymbol{\theta}$, given $\mathcal{X}$, is

$$\hat{\boldsymbol{\theta}} = \mathbb{E}[\boldsymbol{\theta}|\mathcal{X}] = \int \boldsymbol{\theta} p(\boldsymbol{\theta}|\mathcal{X}) d\boldsymbol{\theta}.$$

- Another direction is to obtain an estimate of the pdf of $\mathbf{x}$ given the observations $\mathcal{X}$. This can be done by marginalizing over a distribution, i.e.,

$$p(\boldsymbol{x}|\mathcal{X}) = \int p(\boldsymbol{x}|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{X}) d\boldsymbol{\theta},$$

where the conditional independence of $\mathbf{x}$ on $\mathcal{X}$, given the value $\boldsymbol{\theta} = \boldsymbol{\theta}$, i.e., $p\big((\boldsymbol{x}|\mathcal{X}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})\big)$ has been used. The last equation provides an estimate of the unknown pdf, by exploiting the information that resides in the obtained observations as well as in the adopted functional dependence on the parameters $\boldsymbol{\theta}$.

Bayesian Inference

- A first path to exploit the derived posterior pdf is to obtain a single estimate concerning the unknown parameter vector. One possibility is to make use of what we already know. Since $\mathbf{x}$ and $\boldsymbol{\theta}$ are two statistically dependent random vectors, the MSE optimal estimate of the value of $\boldsymbol{\theta}$, given $\mathcal{X}$, is

$$\hat{\boldsymbol{\theta}} = \mathbb{E}[\boldsymbol{\theta}|\mathcal{X}] = \int \boldsymbol{\theta} p(\boldsymbol{\theta}|\mathcal{X}) d\boldsymbol{\theta}.$$

- Another direction is to obtain an estimate of the pdf of $\mathbf{x}$ given the observations $\mathcal{X}$. This can be done by marginalizing over a distribution, i.e.,

$$p(\boldsymbol{x}|\mathcal{X}) = \int p(\boldsymbol{x}|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{X}) d\boldsymbol{\theta},$$

  where the conditional independence of $\mathbf{x}$ on $\mathcal{X}$, given the value $\boldsymbol{\theta} = \boldsymbol{\theta}$, i.e., $p\big((\boldsymbol{x}|\mathcal{X}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})\big)$ has been used. The last equation provides an estimate of the unknown pdf, by exploiting the information that resides in the obtained observations as well as in the adopted functional dependence on the parameters $\boldsymbol{\theta}$.

Bayesian Inference Example

- Consider the simplified linear regression task

$$\mathrm{y} = \theta + \eta.$$

Assume that the noise samples are i.i.d. drawn from a Gaussian process of zero mean and variance σ_η^2. We impose our a-priori knowledge concerning the unknown θ, via the prior distribution

$$p(\theta) = \mathcal{N}(\theta_0, \sigma_0^2).$$

That is, we assume that we know that the values of θ lie around θ_0, and σ_0^2 quantifies our degree of uncertainty about this prior knowledge. Our goals are: a) to obtain the a-posteriori pdf, given the set of measurements $\boldsymbol{y} = [y_1, \ldots, y_N]^T$, and b) to obtain $\mathbb{E}[\theta|\boldsymbol{y}]$.

- We have that

$$p(\theta|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\theta)p(\theta)}{p(\boldsymbol{y})} = \frac{1}{p(\boldsymbol{y})}\left(\prod_{n=1}^{N} p(y_n|\theta)\right)p(\theta)$$

$$= \frac{1}{p(\boldsymbol{y})}\left(\prod_{n=1}^{N}\frac{1}{\sqrt{2\pi}\sigma_\eta}\exp\left(-\frac{(y_n-\theta)^2}{2\sigma_\eta^2}\right)\right)\times$$

$$\frac{1}{\sqrt{2\pi}\sigma_0}\exp\left(-\frac{(\theta-\theta_0)^2}{2\sigma_0^2}\right).$$

Bayesian Inference Example

- Consider the simplified linear regression task

$$\mathrm{y} = \theta + \eta.$$

Assume that the noise samples are i.i.d. drawn from a Gaussian process of zero mean and variance σ_η^2. We impose our a-priori knowledge concerning the unknown θ, via the prior distribution

$$p(\theta) = \mathcal{N}(\theta_0, \sigma_0^2).$$

That is, we assume that we know that the values of θ lie around θ_0, and σ_0^2 quantifies our degree of uncertainty about this prior knowledge. Our goals are: a) to obtain the a-posteriori pdf, given the set of measurements $\boldsymbol{y} = [y_1, \ldots, y_N]^T$, and b) to obtain $\mathbb{E}[\theta|\boldsymbol{y}]$.

- We have that

$$p(\theta|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\theta)p(\theta)}{p(\boldsymbol{y})} = \frac{1}{p(\boldsymbol{y})}\left(\prod_{n=1}^{N} p(y_n|\theta)\right)p(\theta)$$

$$= \frac{1}{p(\boldsymbol{y})}\left(\prod_{n=1}^{N}\frac{1}{\sqrt{2\pi}\sigma_\eta}\exp\left(-\frac{(y_n-\theta)^2}{2\sigma_\eta^2}\right)\right)\times$$

$$\frac{1}{\sqrt{2\pi}\sigma_0}\exp\left(-\frac{(\theta-\theta_0)^2}{2\sigma_0^2}\right).$$

Bayesian Inference Example

- Consider the simplified linear regression task

$$\mathrm{y} = \theta + \eta.$$

Assume that the noise samples are i.i.d. drawn from a Gaussian process of zero mean and variance σ_η^2. We impose our a-priori knowledge concerning the unknown θ, via the prior distribution

$$p(\theta) = \mathcal{N}(\theta_0, \sigma_0^2).$$

That is, we assume that we know that the values of θ lie around θ_0, and σ_0^2 quantifies our degree of uncertainty about this prior knowledge. Our goals are: a) to obtain the a-posteriori pdf, given the set of measurements $\boldsymbol{y} = [y_1, \ldots, y_N]^T$, and b) to obtain $\mathbb{E}[\theta|\boldsymbol{y}]$.

- We have that

$$\begin{aligned} p(\theta|\boldsymbol{y}) &= \frac{p(\boldsymbol{y}|\theta)p(\theta)}{p(\boldsymbol{y})} = \frac{1}{p(\boldsymbol{y})}\left(\prod_{n=1}^{N} p(y_n|\theta)\right)p(\theta) \\ &= \frac{1}{p(\boldsymbol{y})}\left(\prod_{n=1}^{N}\frac{1}{\sqrt{2\pi}\sigma_\eta}\exp\left(-\frac{(y_n-\theta)^2}{2\sigma_\eta^2}\right)\right)\times \\ &\quad \frac{1}{\sqrt{2\pi}\sigma_0}\exp\left(-\frac{(\theta-\theta_0)^2}{2\sigma_0^2}\right). \end{aligned}$$

Bayesian Inference Example

- After some algebraic manipulations, one ends up in the following

$$p(\theta|\boldsymbol{y}) = \frac{1}{\sqrt{2\pi}\sigma_N} \exp\left(-\frac{(\theta-\bar{\theta}_N)^2}{2\sigma_N^2}\right),$$

where

$$\bar{\theta}_N = \frac{N\sigma_0^2\bar{y}_N + \sigma_\eta^2\theta_0}{N\sigma_0^2 + \sigma_\eta^2},$$

with $\bar{y}_N = \frac{1}{N}\sum_{n=1}^{N} y_n$ being the sample mean of the observations and

$$\sigma_N^2 = \frac{\sigma_\eta^2\sigma_0^2}{N\sigma_0^2 + \sigma_\eta^2}.$$

In words, if the prior and the conditional pdfs are Gaussians, then the posterior is also Gaussian.

- Observe that as the number of observations increases, $\bar{\theta}_N$ tends to the sample mean, $\bar{y}_N$, of the observations; recall that the latter is the estimate that results from the ML method. Also, note that the variance keeps decreasing as the number of observations increases; which is in line to common sense, since more observations reduce uncertainty. These findings are illustrated by the following figure.

Bayesian Inference Example

- After some algebraic manipulations, one ends up in the following

$$p(\theta|\boldsymbol{y}) = \frac{1}{\sqrt{2\pi}\sigma_N} \exp\left(-\frac{(\theta-\bar{\theta}_N)^2}{2\sigma_N^2}\right),$$

where

$$\bar{\theta}_N = \frac{N\sigma_0^2\bar{y}_N + \sigma_\eta^2\theta_0}{N\sigma_0^2 + \sigma_\eta^2},$$

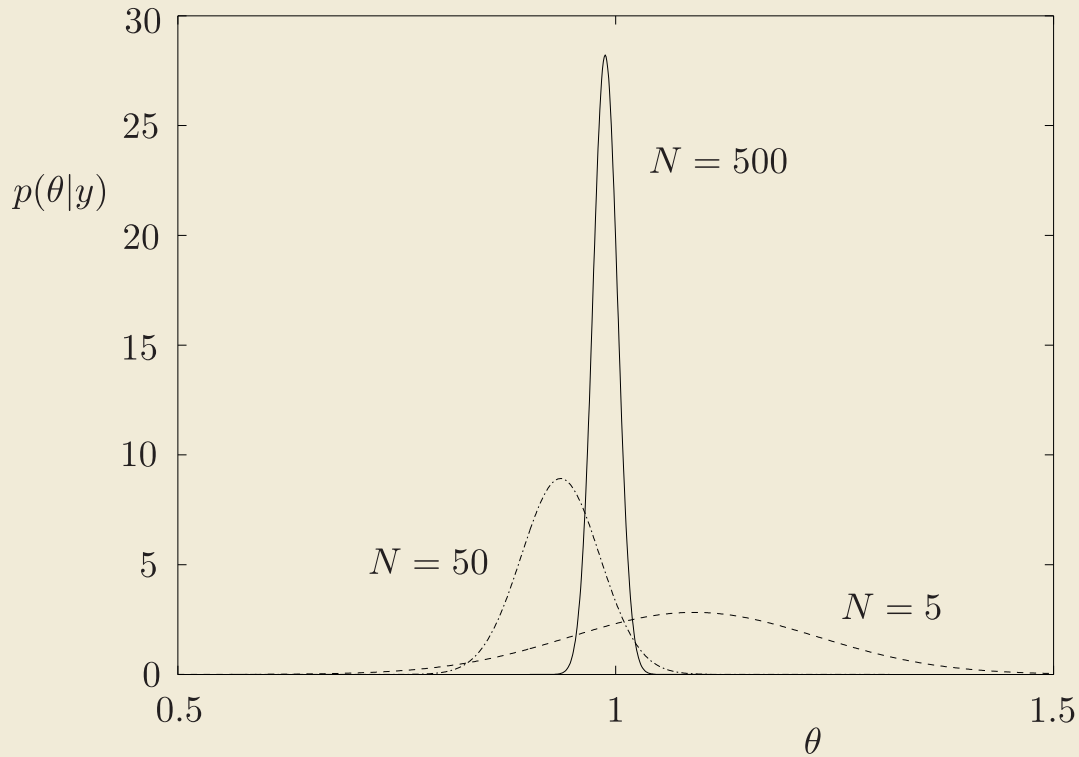
with $\bar{y}_N = \frac{1}{N}\sum_{n=1}^{N} y_n$ being the sample mean of the observations and

$$\sigma_N^2 = \frac{\sigma_\eta^2\sigma_0^2}{N\sigma_0^2 + \sigma_\eta^2}.$$

In words, if the prior and the conditional pdfs are Gaussians, then the posterior is also Gaussian.

- Observe that as the number of observations increases, $\bar{\theta}_N$ tends to the sample mean, $\bar{y}_N$, of the observations; recall that the latter is the estimate that results from the ML method. Also, note that the variance keeps decreasing as the number of observations increases; which is in line to common sense, since more observations reduce uncertainty. These findings are illustrated by the following figure.

Bayesian Inference Example



In the Bayesian inference approach, note that as the number of observations increases, our uncertainty about the true value of the unknown parameter is reduced and the mean of the posterior pdf tends to the true value (in this case equal to 1) and the variance tends to zero.

Maximum A-Posteriori Probability Estimation Method

- The Maximum A-Posteriori Probability Estimation technique, usually denoted as MAP, is based on the Bayesian theorem, but it does not go as far as the Bayesian philosophy allows to. The goal becomes that of obtaining an estimate by maximizing

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} := \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{\theta}).p(\boldsymbol{\theta})}{p(\mathcal{X})}$$

- Because $p(\mathcal{X})$ is independent of θ, this leads to

$$\begin{aligned}\hat{\theta}_{\text{MAP}} &= \arg\max_{\boldsymbol{\theta}} p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \left\{ \ln p(\mathcal{X}|\boldsymbol{\theta}) + \ln p(\boldsymbol{\theta}) \right\}.\end{aligned}$$

- For the case of the estimation of the parameter hidden in noise $(\mathrm{y} = \theta + \eta)$, using as prior the Gaussian $\mathcal{N}(\theta_0, \sigma_0^2)$, one can readily obtain,

$$\hat{\theta}_{\text{MAP}} = \frac{N\bar{y}_N + \frac{\sigma_\eta^2}{\sigma_0^2}\theta_0}{N + \frac{\sigma_\eta^2}{\sigma_0^2}} = \bar{\theta}_N,$$

where $\bar{\theta}_N$ is the mean value of the posterior obtained via the Bayesian inference method. Note that even for this very simple case, the Bayesian inference approach provides an extra piece of information; that is, the variance around $\bar{\theta}_N$.

Maximum A-Posteriori Probability Estimation Method

- The Maximum A-Posteriori Probability Estimation technique, usually denoted as MAP, is based on the Bayesian theorem, but it does not go as far as the Bayesian philosophy allows to. The goal becomes that of obtaining an estimate by maximizing

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} := \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{\theta}).p(\boldsymbol{\theta})}{p(\mathcal{X})}$$

- Because $p(\mathcal{X})$ is independent of $\boldsymbol{\theta}$, this leads to

$$\begin{aligned}\hat{\boldsymbol{\theta}}_{\text{MAP}} &= \arg\max_{\boldsymbol{\theta}} p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \left\{ \ln p(\mathcal{X}|\boldsymbol{\theta}) + \ln p(\boldsymbol{\theta}) \right\}.\end{aligned}$$

- For the case of the estimation of the parameter hidden in noise ($y = \theta + \eta$), using as prior the Gaussian $\mathcal{N}(\theta_0, \sigma_0^2)$, one can readily obtain,

$$\hat{\theta}_{\text{MAP}} = \frac{N\bar{y}_N + \frac{\sigma_\eta^2}{\sigma_0^2}\theta_0}{N + \frac{\sigma_\eta^2}{\sigma_0^2}} = \bar{\theta}_N,$$

where $\bar{\theta}_N$ is the mean value of the posterior obtained via the Bayesian inference method. Note that even for this very simple case, the Bayesian inference approach provides an extra piece of information; that is, the variance around $\bar{\theta}_N$.

Maximum A-Posteriori Probability Estimation Method

- The Maximum A-Posteriori Probability Estimation technique, usually denoted as MAP, is based on the Bayesian theorem, but it does not go as far as the Bayesian philosophy allows to. The goal becomes that of obtaining an estimate by maximizing

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} := \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{\theta}).p(\boldsymbol{\theta})}{p(\mathcal{X})}$$

- Because $p(\mathcal{X})$ is independent of $\boldsymbol{\theta}$, this leads to

$$\begin{aligned}\hat{\boldsymbol{\theta}}_{\text{MAP}} &= \arg\max_{\boldsymbol{\theta}} p(\mathcal{X}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \left\{ \ln p(\mathcal{X}|\boldsymbol{\theta}) + \ln p(\boldsymbol{\theta}) \right\}.\end{aligned}$$

- For the case of the estimation of the parameter hidden in noise ($\mathrm{y} = \theta + \eta$), using as prior the Gaussian $\mathcal{N}(\theta_0, \sigma_0^2)$, one can readily obtain,

$$\hat{\theta}_{\text{MAP}} = \frac{N\bar{y}_N + \frac{\sigma_\eta^2}{\sigma_0^2}\theta_0}{N + \frac{\sigma_\eta^2}{\sigma_0^2}} = \bar{\theta}_N,$$

  where $\bar{\theta}_N$ is the mean value of the posterior obtained via the Bayesian inference method. Note that even for this very simple case, the Bayesian inference approach provides an extra piece of information; that is, the variance around $\bar{\theta}_N$.

When Ridge Regression, MAP and Bayesian Inference Meet

- Let us focus, for simplicity, to the model $(\mathrm{y} = \theta + \eta)$, of estimating a parameter via its noisy observations, $y_1, \ldots, y_N$.
- Let us assume that we have prior knowledge concerning the unknown parameter that is located close to a value, θ_0. We will "embed" this information in the LS criterion in the form of a constraint. This can be done by modifying the constraint used in the ridge regression, such that

$$(\theta - \theta_0)^2 \leq \rho,$$

which leads to the minimization of the following Lagrangian

$$\text{minimize} \quad L(\theta, \lambda) = \sum_{n=1}^{N} (y_n - \theta)^2 + \lambda \left((\theta - \theta_0)^2 - \rho \right).$$

- Taking the derivative with respect to θ and equating to zero, we obtain

$$\hat{\theta} = \frac{N\bar{y}_N + \lambda\theta_0}{N + \lambda}.$$

- Note that this is exactly the same like the MAP estimate, if one sets the regularization parameter $\lambda = \sigma_\eta^2/\sigma_0^2$. In other words, the effect of a prior in MAP and the Bayesian inference is equivalent to the use of a regularizer in the LS (and not only) method. The estimates are the same only for the special case of white Gaussian noise and Gaussian priors.

When Ridge Regression, MAP and Bayesian Inference Meet

- Let us focus, for simplicity, to the model $(\mathrm{y} = \theta + \eta)$, of estimating a parameter via its noisy observations, $y_1, \ldots, y_N$.
- Let us assume that we have prior knowledge concerning the unknown parameter that is located close to a value, θ_0. We will "embed" this information in the LS criterion in the form of a constraint. This can be done by modifying the constraint used in the ridge regression, such that

$$(\theta - \theta_0)^2 \leq \rho,$$

which leads to the minimization of the following Lagrangian

$$\text{minimize} \quad L(\theta, \lambda) = \sum_{n=1}^{N} (y_n - \theta)^2 + \lambda \left((\theta - \theta_0)^2 - \rho \right).$$

- Taking the derivative with respect to θ and equating to zero, we obtain

$$\hat{\theta} = \frac{N\bar{y}_N + \lambda\theta_0}{N + \lambda}.$$

- Note that this is exactly the same like the MAP estimate, if one sets the regularization parameter $\lambda = \sigma_\eta^2/\sigma_0^2$. In other words, the effect of a prior in MAP and the Bayesian inference is equivalent to the use of a regularizer in the LS (and not only) method. The estimates are the same only for the special case of white Gaussian noise and Gaussian priors.

When Ridge Regression, MAP and Bayesian Inference Meet

- Let us focus, for simplicity, to the model $(\mathrm{y} = \theta + \eta)$, of estimating a parameter via its noisy observations, $y_1, \ldots, y_N$.
- Let us assume that we have prior knowledge concerning the unknown parameter that is located close to a value, θ_0. We will "embed" this information in the LS criterion in the form of a constraint. This can be done by modifying the constraint used in the ridge regression, such that

$$(\theta - \theta_0)^2 \leq \rho,$$

which leads to the minimization of the following Lagrangian

$$\text{minimize} \quad L(\theta, \lambda) = \sum_{n=1}^{N} (y_n - \theta)^2 + \lambda \left((\theta - \theta_0)^2 - \rho \right).$$

- Taking the derivative with respect to θ and equating to zero, we obtain

$$\hat{\theta} = \frac{N\bar{y}_N + \lambda\theta_0}{N + \lambda}.$$

- Note that this is exactly the same like the MAP estimate, if one sets the regularization parameter $\lambda = \sigma_\eta^2/\sigma_0^2$. In other words, the effect of a prior in MAP and the Bayesian inference is equivalent to the use of a regularizer in the LS (and not only) method. The estimates are the same only for the special case of white Gaussian noise and Gaussian priors.

When Ridge Regression, MAP and Bayesian Inference Meet

- Let us focus, for simplicity, to the model $(\mathrm{y} = \theta + \eta)$, of estimating a parameter via its noisy observations, $y_1, \ldots, y_N$.
- Let us assume that we have prior knowledge concerning the unknown parameter that is located close to a value, θ_0. We will "embed" this information in the LS criterion in the form of a constraint. This can be done by modifying the constraint used in the ridge regression, such that

$$(\theta - \theta_0)^2 \leq \rho,$$

which leads to the minimization of the following Lagrangian

$$\text{minimize} \quad L(\theta, \lambda) = \sum_{n=1}^{N} (y_n - \theta)^2 + \lambda \left((\theta - \theta_0)^2 - \rho \right).$$

- Taking the derivative with respect to θ and equating to zero, we obtain

$$\hat{\theta} = \frac{N\bar{y}_N + \lambda\theta_0}{N + \lambda}.$$

- Note that this is exactly the same like the MAP estimate, if one sets the regularization parameter $\lambda = \sigma_\eta^2/\sigma_0^2$. In other words, the effect of a prior in MAP and the Bayesian inference is equivalent to the use of a regularizer in the LS (and not only) method. The estimates are the same only for the special case of white Gaussian noise and Gaussian priors.

Curse Of Dimensionality

- In a number of places, we mentioned the need of having a large number of training points. While talking for the bias-variance tradeoff, it was stated that in order to end up with a low overall MSE, the complexity (number of parameters) of the model should be small enough with respect to the number of training points. Also, overfitting was discussed and it was pointed out that, if the number of training points is small with respect to the number of parameters, overfitting occurs.
- The question that is now raised is how big a data set should be. The answer to the previous question depends largely on the dimensionality of the input space. It turns out that, the larger the dimension of the input space is the more data points are needed. This is related to the so-called curse of dimensionality.
- Let us assume that we are given the same number of points, N, thrown randomly in a unit cube (hypercube) in two different spaces, one being of low and the other of very high dimension. Then, the average distance of the points in the latter case will be much larger than that in the low-dimensional space case. As a matter of fact, the average distance shows a dependence that is analogous to the exponential term $(N^{-1/l})$, where l is the dimensionality of the space.

Curse Of Dimensionality

- In a number of places, we mentioned the need of having a large number of training points. While talking for the bias-variance tradeoff, it was stated that in order to end up with a low overall MSE, the complexity (number of parameters) of the model should be small enough with respect to the number of training points. Also, overfitting was discussed and it was pointed out that, if the number of training points is small with respect to the number of parameters, overfitting occurs.
- The question that is now raised is how big a data set should be. The answer to the previous question depends largely on the dimensionality of the input space. It turns out that, the larger the dimension of the input space is the more data points are needed. This is related to the so-called curse of dimensionality.
- Let us assume that we are given the same number of points, N, thrown randomly in a unit cube (hypercube) in two different spaces, one being of low and the other of very high dimension. Then, the average distance of the points in the latter case will be much larger than that in the low-dimensional space case. As a matter of fact, the average distance shows a dependence that is analogous to the exponential term $(N^{-1/l})$, where l is the dimensionality of the space.

Curse Of Dimensionality

- In a number of places, we mentioned the need of having a large number of training points. While talking for the bias-variance tradeoff, it was stated that in order to end up with a low overall MSE, the complexity (number of parameters) of the model should be small enough with respect to the number of training points. Also, overfitting was discussed and it was pointed out that, if the number of training points is small with respect to the number of parameters, overfitting occurs.
- The question that is now raised is how big a data set should be. The answer to the previous question depends largely on the dimensionality of the input space. It turns out that, the larger the dimension of the input space is the more data points are needed. This is related to the so-called curse of dimensionality.
- Let us assume that we are given the same number of points, N, thrown randomly in a unit cube (hypercube) in two different spaces, one being of low and the other of very high dimension. Then, the average distance of the points in the latter case will be much larger than that in the low-dimensional space case. As a matter of fact, the average distance shows a dependence that is analogous to the exponential term $(N^{-1/l})$, where l is the dimensionality of the space.

Curse Of Dimensionality

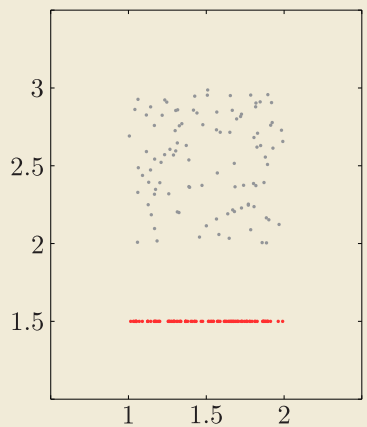
- For example, the average distance of 10^{10} points in the 2-dimensional space is 10^{-5} and in the 40-dimensional space is equal to 1.83.
- The figure below shows two cases, each one consisting of 100 points. The red points lie on a (one-dimensional) line segment of length equal to one and were generated according to the uniform distribution. Gray points cover a (two-dimensional) square region of unit area, which were also generated by a two-dimensional uniform distribution. The square area is more sparsely populated compared to the line segment.
- This is the general trend and high dimensional spaces are sparsely populated; thus, many more data points are needed in order to fill in the space with enough data. Fitting a model in a parameter space, one must have enough data covering sufficiently well all regions in the space, in order to be able to learn well enough the input-output functional dependence.

Curse Of Dimensionality

- For example, the average distance of 10^{10} points in the 2-dimensional space is 10^{-5} and in the 40-dimensional space is equal to 1.83.
- The figure below shows two cases, each one consisting of 100 points. The red points lie on a (one-dimensional) line segment of length equal to one and were generated according to the uniform distribution. Gray points cover a (two-dimensional) square region of unit area, which were also generated by a two-dimensional uniform distribution. The square area is more sparsely populated compared to the line segment.
- This is the general trend and high dimensional spaces are sparsely populated; thus, many more data points are needed in order to fill in the space with enough data. Fitting a model in a parameter space, one must have enough data covering sufficiently well all regions in the space, in order to be able to learn well enough the input-output functional dependence.

Curse Of Dimensionality

- For example, the average distance of 10^{10} points in the 2-dimensional space is 10^{-5} and in the 40-dimensional space is equal to 1.83.
- The figure below shows two cases, each one consisting of 100 points. The red points lie on a (one-dimensional) line segment of length equal to one and were generated according to the uniform distribution. Gray points cover a (two-dimensional) square region of unit area, which were also generated by a two-dimensional uniform distribution. The square area is more sparsely populated compared to the line segment.
- This is the general trend and high dimensional spaces are sparsely populated; thus, many more data points are needed in order to fill in the space with enough data. Fitting a model in a parameter space, one must have enough data covering sufficiently well all regions in the space, in order to be able to learn well enough the input-output functional dependence.



Curse Of Dimensionality

- There are various ways to cope with the curse dimensionality and try to exploit the available data set in the best possible way. A popular direction is to resort to suboptimal solutions by projecting the input/feature vectors in a lower dimensional subspace or manifold. Very often, such an approach leads to small performance losses, since the original training data, although they are generated in a high dimensional space, in fact they may "live" in a lower dimensional subspace or manifold, due to physical dependencies that restrict the number of free parameters. The challenge, now, becomes that of learning the subspace/manifold onto which to project.
- Dimensionality of the input space may not be always the crucial issue. In pattern recognition, it has been shown that the critical factor is the so-called VC-dimension of a classifier. In a number of classifiers, such as (generalized) linear classifiers or neural networks, the VC-dimension is directly related to the dimensionality of the input space. However, one can design classifiers, such as the Support Vector Machines, whose performance is not directly related to the input space and they can be efficiently designed in spaces of very high (of even infinite) dimensionality.

Curse Of Dimensionality

- There are various ways to cope with the curse dimensionality and try to exploit the available data set in the best possible way. A popular direction is to resort to suboptimal solutions by projecting the input/feature vectors in a lower dimensional subspace or manifold. Very often, such an approach leads to small performance losses, since the original training data, although they are generated in a high dimensional space, in fact they may "live" in a lower dimensional subspace or manifold, due to physical dependencies that restrict the number of free parameters. The challenge, now, becomes that of learning the subspace/manifold onto which to project.
- Dimensionality of the input space may not be always the crucial issue. In pattern recognition, it has been shown that the critical factor is the so-called VC-dimension of a classifier. In a number of classifiers, such as (generalized) linear classifiers or neural networks, the VC-dimension is directly related to the dimensionality of the input space. However, one can design classifiers, such as the Support Vector Machines, whose performance is not directly related to the input space and they can be efficiently designed in spaces of very high (of even infinite) dimensionality.

Validation

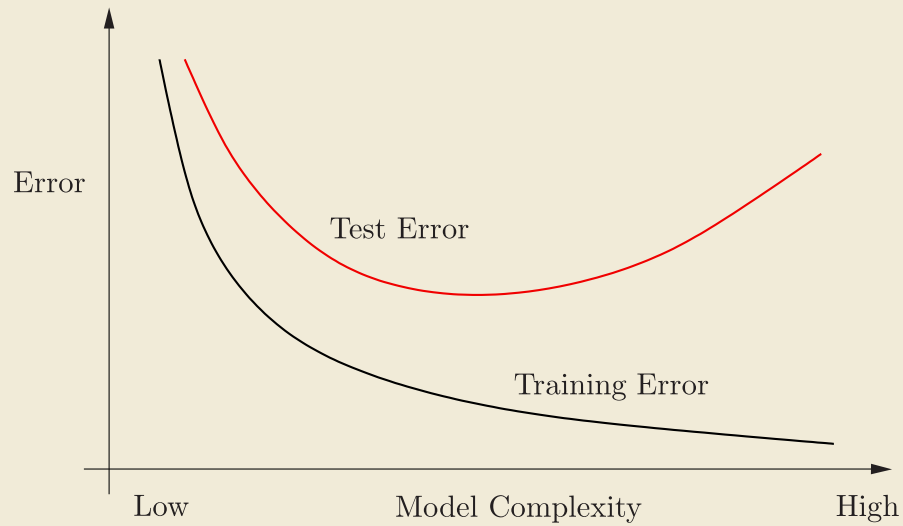
- A major phase, in any machine learning task, is to quantify/predict the performance that the designed (prediction) model is expected to exhibit in practice. Evaluating the performance against the training data set would lead to an optimistic value of the performance index, since this is computed on the same set on which the estimator was optimized.
- For example, if the model is complex enough, with a large number of free parameters, the training error may even become zero, since a perfect fit to the data can be achieved. What is more meaningful and fair is to look for the so-called generalization performance of an estimator; that is, its average performance computed over different data sets, which did not participate in the training.

Validation

- A major phase, in any machine learning task, is to quantify/predict the performance that the designed (prediction) model is expected to exhibit in practice. Evaluating the performance against the training data set would lead to an optimistic value of the performance index, since this is computed on the same set on which the estimator was optimized.
- For example, if the model is complex enough, with a large number of free parameters, the training error may even become zero, since a perfect fit to the data can be achieved. What is more meaningful and fair is to look for the so-called generalization performance of an estimator; that is, its average performance computed over different data sets, which did not participate in the training.

Validation

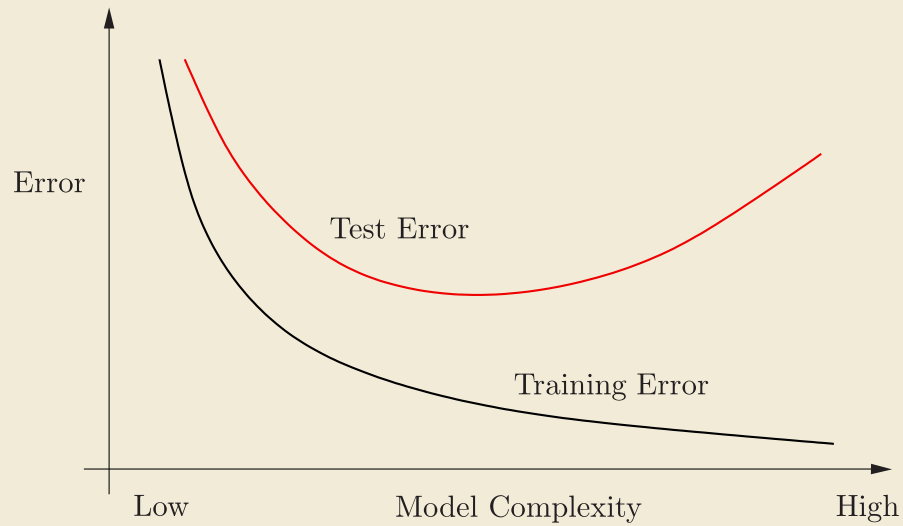
- The figure below shows a typical performance trend, that is expected to result in practice. The error measured on the (single) training data set is shown together with the (average) test/generalization error as the model complexity varies.



- The training error tends to zero as the model complexity increases; for complex enough models with a large number of free parameters, a perfect fit to the training data is possible. However, the test error initially decreases, since more complex models "learn" the data better, till a point. After that point of complexity, the test error increases.

Validation

- The figure below shows a typical performance trend, that is expected to result in practice. The error measured on the (single) training data set is shown together with the (average) test/generalization error as the model complexity varies.



- The training error tends to zero as the model complexity increases; for complex enough models with a large number of free parameters, a perfect fit to the training data is possible. However, the test error initially decreases, since more complex models "learn" the data better, till a point. After that point of complexity, the test error increases.

Cross-Validation

- Cross-validation is a very common technique that is usually employed in practice. According to this method, the data set is split into, say K, roughly equal-sized, parts. We repeat training K times, each time selecting one (different each time) part of the data for testing and the remaining $K-1$ parts for training. This gives us the advantage of testing with a part of the data that has not been involved in the training, hence it can be considered as being independent, and at the same time using, eventually, **all** the data **both** for training and testing.
- Once we finish, we can a) combine the obtained K estimates, e.g., by averaging or via another more advanced way and b) combine the test errors to get a better estimate of the generalization error that our estimator is expected to exhibit in real life applications. The method is known as K-fold cross-validation.
- An extreme case is when we use $K = N$; that is, each time one sample is left for testing. This is sometimes referred to as the Leave-One-Out (LOO) cross-validation method. The price one pays for K-fold cross-validation is the complexity of training K times. In practice, the value of K depends very much on the application, but typical values are of the order of 5 to 10.

Cross-Validation

- Cross-validation is a very common technique that is usually employed in practice. According to this method, the data set is split into, say K, roughly equal-sized, parts. We repeat training K times, each time selecting one (different each time) part of the data for testing and the remaining $K-1$ parts for training. This gives us the advantage of testing with a part of the data that has not been involved in the training, hence it can be considered as being independent, and at the same time using, eventually, **all** the data **both** for training and testing.
- Once we finish, we can a) combine the obtained K estimates, e.g., by averaging or via another more advanced way and b) combine the test errors to get a better estimate of the generalization error that our estimator is expected to exhibit in real life applications. The method is known as K-fold cross-validation.
- An extreme case is when we use $K = N$; that is, each time one sample is left for testing. This is sometimes referred to as the Leave-One-Out (LOO) cross-validation method. The price one pays for K-fold cross-validation is the complexity of training K times. In practice, the value of K depends very much on the application, but typical values are of the order of 5 to 10.

Cross-Validation

- Cross-validation is a very common technique that is usually employed in practice. According to this method, the data set is split into, say K, roughly equal-sized, parts. We repeat training K times, each time selecting one (different each time) part of the data for testing and the remaining $K-1$ parts for training. This gives us the advantage of testing with a part of the data that has not been involved in the training, hence it can be considered as being independent, and at the same time using, eventually, **all** the data **both** for training and testing.
- Once we finish, we can a) combine the obtained K estimates, e.g., by averaging or via another more advanced way and b) combine the test errors to get a better estimate of the generalization error that our estimator is expected to exhibit in real life applications. The method is known as K-fold cross-validation.
- An extreme case is when we use $K = N$; that is, each time one sample is left for testing. This is sometimes referred to as the Leave-One-Out (LOO) cross-validation method. The price one pays for K-fold cross-validation is the complexity of training K times. In practice, the value of K depends very much on the application, but typical values are of the order of 5 to 10.