https://goo.gl/kfxwEg

KYOTO UNIVERSITY

Statistical Learning Theory
- Regression -

Hisashi Kashima

DEPARTMENT OF INTELLIGENCE SCIENCE AND TECHNOLOGY

Linear Regression

Regression: Supervised learning for predicting real values

- Regression learning is one of supervised learning problem settings with wide applications
- Goal: Obtain a function $f: \mathcal{X} \to \Re$ ($\Re$: real value)
  - E.g. $x \in \mathcal{X}$ is a house and $y \in \Re$ is its price (housing dataset in UCI Machine Learning Repository)
  - Usually, $\mathcal{X}$ is a D-dimensional vector space
- Training dataset: N pairs of an input and an output $\{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})\}$

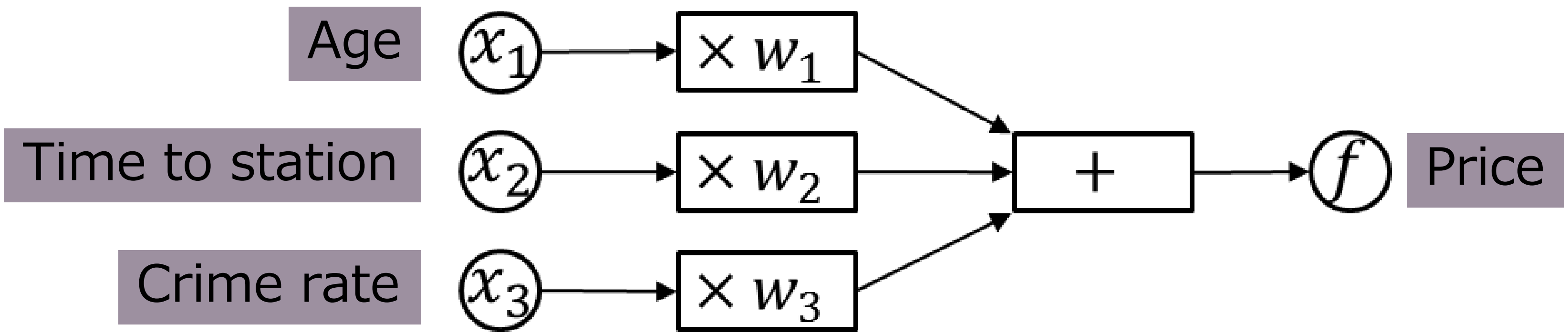
Some applications of regression:
From marketing prediction to chemoinformatics

- Some applications:
  - Price prediction: Predict the price y of a product x
  - Demand prediction: Predict the demanded amount y of a product x
  - Sales prediction: Predict the sales amount y of a product x
  - Chemical activity: Predict the activity level y of a compound x
- Other applications:
  - Time series prediction: Predict the value y at the next time step given the past measurements x
  - Classification

Model:
Linear regression model

- Model: How does y depend on $\mathbf{x}$?
- We consider the simplest choices: Liner regression model

$$y = \mathbf{w}^\top \mathbf{x} = w_1 x_1 + w_2 x_2 + \cdots + w_D x_D$$



Handling discrete features: Dummy variables

- We assume input $\mathbf{x}$ is a real vector
  - In the house price prediction example, features can be age, walk time to the nearest station, crime rate in the area, …
- Discrete features are handled as real values
  - Binary features: $\{\text{Male}, \text{Female}\}$ are encoded as $\{0,1\}$
  - One-hot encoding: $\{\text{Kyoto}, \text{Osaka}, \text{Tokyo}\}$ are encoded with $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$

Objective function of training: Squared loss

- Objective function (to minimize): Disagreement measure of the model to the training dataset
  - Loss function: $\ell^{(i)}\left(y^{(i)}, \mathbf{w}^\top \mathbf{x}^{(i)};\ \mathbf{w}\right)$ for the i-th instance
  - Objective function: $L(\mathbf{w}) = \sum_{i=1}^{N} \ell^{(i)}\left(y^{(i)}, \mathbf{w}^\top \mathbf{x}^{(i)};\ \mathbf{w}\right)$
- Squared loss function:

$$\ell^{(i)}\left(y^{(i)}, \mathbf{w}^\top \mathbf{x}^{(i)};\ \mathbf{w}\right) = \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)}\right)^2$$

  - Absolute loss, Huber loss: more robust choices
- Optimal parameter $\mathbf{w}^*$ is the one that minimizes $L(\mathbf{w})$:

$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} L(\mathbf{w})$$

Important assumption on data: Identically and independently distributed

- We assume data are *identically and independently distributed*:
  - Data instances are generated from the same data generation mechanism (or probability distribution)
    - Past data (training data) and future data (test data) have the same property
  - Data instances are independent of each other

Solution of linear regression: One dimensional case

- Let us start with a case where inputs and outputs are both one-dimensional
- Objective function to minimize:

$$L(w) = \sum_{i=1}^{N} \left(y^{(i)} - wx^{(i)}\right)^2$$

- Solution: $w^* = \frac{\sum_{i=1}^{N} y^{(i)} x^{(i)}}{\sum_{i=1}^{N} {x^{(i)}}^2} = \frac{\mathrm{Cov}(x,y)}{\mathrm{Var}(x)}$

Solution of linear regression: General case

- Matrix and vector notations:
  - Design matrix $\boldsymbol{X} = \left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}\right)^\top$
  - Target vector $\mathbf{y} = \left(y^{(1)}, y^{(2)}, \dots, y^{(N)}\right)^\top$
- Objective function:

$$L(\mathbf{w}) = \sum_{i=1}^{N} \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)}\right)^2 = \|\mathbf{y} - \boldsymbol{X}\mathbf{w}\|_2^2$$

$$= (\mathbf{y} - \boldsymbol{X}\mathbf{w})^\top (\mathbf{y} - \boldsymbol{X}\mathbf{w})$$

- Solution: $\mathbf{w}^* = \mathrm{argmin}_{\mathbf{w}} L(\mathbf{w}) = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \mathbf{y}$

Regularization

Ridge regression: Include penalty on the norm of $\mathbf{w}$ to avoid instability

- Existence of the solution $\mathbf{w}^* = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}\mathbf{y}$ requires that $\boldsymbol{X}^\top \boldsymbol{X}$ is non-singular, i.e. full-rank
  - This is often secured when the number of data instances N is much larger than the number of dimensions D
- Regularization: Adding some constant $\lambda > 0$ to the diagonals of $\boldsymbol{X}^\top \boldsymbol{X}$ for numerical stability
  - New solution: $\mathbf{w}^* = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \mathbf{y}$
- Back to its objective function,

$$L(\mathbf{w}) = \|\mathbf{y} - \boldsymbol{X}\mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_2^2$$

Overfitting:
Degradation of predictive performance for future data

- Previously, we introduced the regularization to avoid numerical stability
- Overfitting to the training data:
  - Our goal is to make correct predictions for future data, not for the training data
  - Too much adaptation to the training data degrades predictive performance on future data
- When the number of data instances N is less than the number of dimensions D, the solution is not unique
  - Arbitrary number of solutions exist

Occam's razor: Adopt the simplest model

- What is the "good" model among the models equally fitting to the training data?
- Occam's razor: Take the simplest model
  - We will discuss why the simple model is good later in the statistical learning theory
- What is the measure of simplicity? For example, number of features = the number of non-zero elements in $\mathbf{w}$

0-norm regularization: Reduce the number of non-zero elements in $\mathbf{w}$

- Number of non-zero elements in $\mathbf{w}$ = 0-norm of $\mathbf{w}$
- Use 0-norm constraint:

$$\text{minimize}_{\mathbf{w}} \|\mathbf{y} - \boldsymbol{X}\mathbf{w}\|_2^2 \ \text{s.t.}\ \|\mathbf{w}\|_0 \leq \eta$$

Number of features used in the model

or 0-norm penalty:

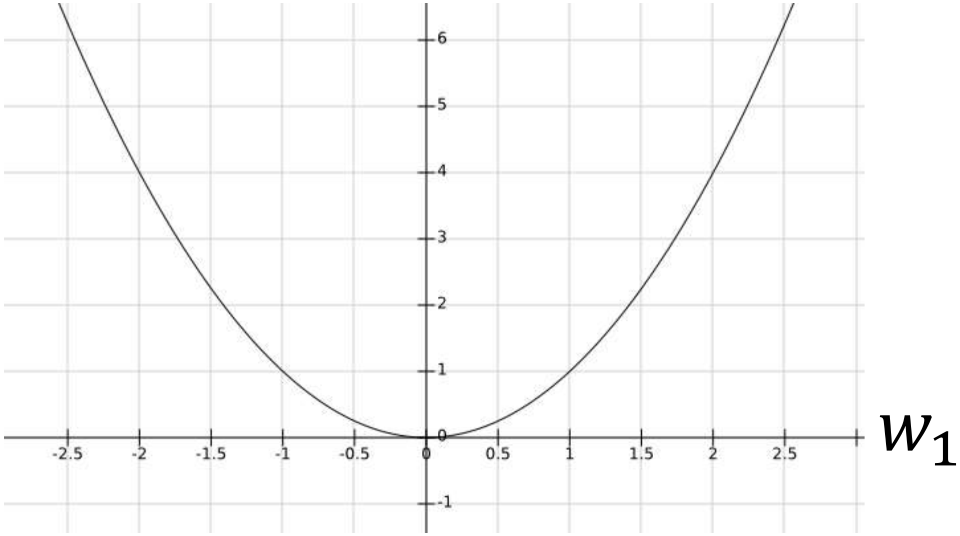
$$\text{minimize}_{\mathbf{w}} \|\mathbf{y} - \boldsymbol{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_0$$

– There is some one-to-one correspondence between η and λ

- However, this is non-convex optimization problems …

Convex surrogate for 0-norm :
2-norm regularization in ridge regression

- Instead of the zero-norm $\|\mathbf{w}\|_0$, we use 2-norm $\|\mathbf{w}\|_2^2$



- Ridge regression: $L(\mathbf{w}) = \|\mathbf{y} - \boldsymbol{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$

Convex ☺

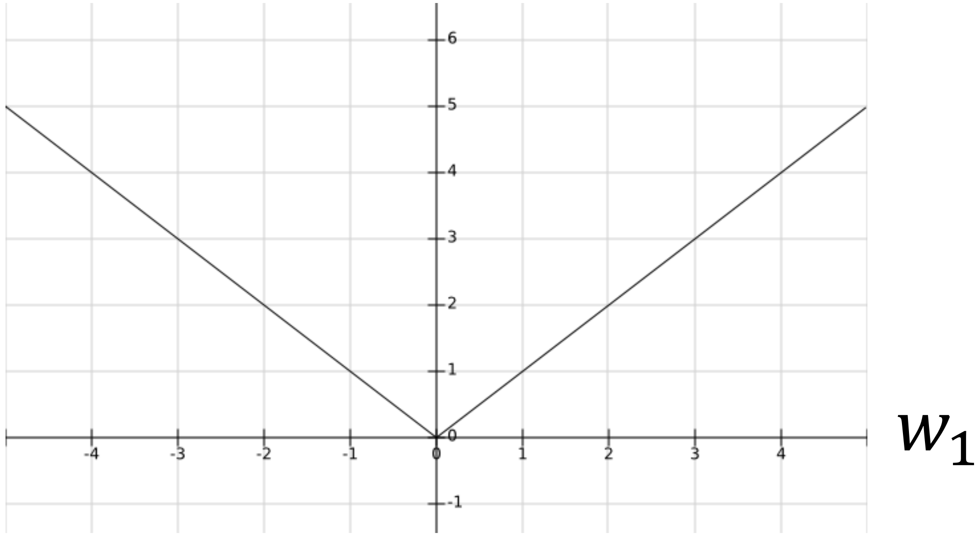
  – Can be seen as a relaxed version of

$$L(\mathbf{w}) = \|\mathbf{y} - \boldsymbol{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_0$$

Non-convex ☹

Another convex surrogate for 0-norm : 1-norm regularization in lasso induces sparsity

- Instead, we can use 1-norm $\|\mathbf{w}\|_1 = |w_1| + |w_2| + \cdots + |w_D|$



- Lasso: $L(\mathbf{w}) = \|\mathbf{y} - \boldsymbol{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_1$
  - Convex optimization, but no closed form solution
- Sparsity inducing norm: 1-norm induces sparse $\mathbf{w}^*$

Statistical Interpretation

Interpretation as statistical inference : Regression as maximum likelihood estimation

- So far we have formulated the regression problem in loss minimization framework
  - Function (prediction model) $f: \mathcal{X} \rightarrow \Re$ is deterministic
  - Least squares: Minimization of the sum of squared losses
- We have not considered any statistical inference
- Actually, we can interpret the previous formulation in a statistical inference framework, namely, maximum likelihood estimation

Maximum likelihood estimation (MLE): Find the parameter that best reproduces training data

- We consider f as a conditional distribution $f(y|\mathbf{x}, \mathbf{w})$

Conditional probability

- Maximum likelihood estimation (MLE):
  - Find $\mathbf{w}$ that maximizes the likelihood function:
    $L(\mathbf{w}) = \prod_{i=1}^{N} f(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{w})$
    - Likelihood function: Probability that the training data is reproduced by the model
    - Note that we assume i.i.d.
  - It is often convenient to use log likelihood instead:

$$L(\mathbf{w}) = \sum_{i=1}^{N} \log f(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{w})$$

Probabilistic version of the linear regression model: Gaussian distribution

- Probabilistic version of the linear regression model $y = \mathbf{w}^\top \mathbf{x}$
- $y \sim \mathcal{N}(\mathbf{w}^\top \mathbf{x}, \sigma^2)$: Gaussian distribution with mean $\mathbf{w}^\top \mathbf{x}$ and variance σ^2

$$f(y|\mathbf{x}, \mathbf{w}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y - \mathbf{w}^\top \mathbf{x})^2}{2\sigma^2}\right)$$

Linear regression model

Relation between least squares and MLE: Maximum likelihood is equivalent to least squares

- Log-likelihood function:

$$L(\mathbf{w}) = \sum_{i=1}^{N} \log f(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{w})$$

$$= \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{\left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)}\right)^2}{2\sigma^2}\right)$$

$$= -\frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)}\right)^2 + \text{const.}$$

- Maximization of $L(\mathbf{w})$ is equivalent to minimization of $\sum_{i=1}^{N} \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)}\right)^2$

Some More Applications

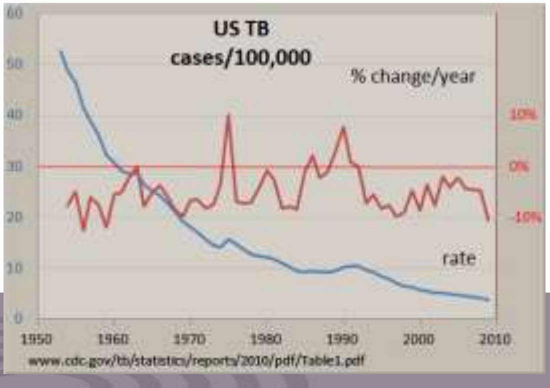
Time series prediction: Auto regressive (AR) model

- Time series data: A sequence of real valued data $x_1, x_2, \dots, x_t, \dots \in \Re$ associated with time stamps $t = 1,2,\dots$
- Time series prediction: Given $x_1, x_2, \dots, x_{t-1}$, predict x_t
- Auto regressive (AR) model:

$$x_t = w_1 x_{t-1} + w_2 x_{t-2} + \cdots + w_D x_{t-D}$$

  – x_t is determined by the recent D data instances

- AR model as a linear regression model $y = \mathbf{w}^\top \mathbf{x}$:
  – $\mathbf{w} = (w_1, w_2, \dots, w_D)^\top$
  – $\mathbf{x} = (x_{t-1}, x_{t-2}, \dots, x_{t-D})^\top$



Classification as regression:
Regression is also applicable to classification

- Classification: $y \in \{+1, -1\}$
- Apply regression to predict $y \in \{+1, -1\}$
- Rigorously, such application is not valid
  - Since an output is either $+1$ or -1, the Gaussian noise assumption does not hold
  - However, since solution of regression is often easier than that of classification, this application can be compromise
- Fisher discriminant: Instead of $\{+1, -1\}$, use $\left\{+\frac{1}{N^+}, -\frac{1}{N^-}\right\}$
  - $N^+(N^-)$ is the number of positive (negative) data

Nonlinear Regression

Nonlinear regression: Introducing nonlinearity in linear models

- So far we have considered only linear models
- How to introduce non-linearity in the models?
- Introduce nonlinear basis functions:
  - Transformed features: e.g. $x \to \log x$
  - Cross terms: e.g. $x_1, x_2 \to x_1 x_2$
  - Kernels: $\mathbf{x} \to \boldsymbol{\phi}(\mathbf{x})$ (some nonlinear mapping to a high-dimensional space)

Nonlinear transformation of features: Simplest way to introduce nonlinearity in linear models

- Nonlinear basis function: $x \to \log x, e^x, x^2, \frac{1}{x}, \ldots$
  - Sometimes used for converting the range
    - E.g. $\log: \Re^+ \to \Re$, $\exp: \Re \to \Re^+$
- Interpretations of log transformation:

	y	$\log y$
x	$y = \beta x + \alpha$ Increase of x by 1 will increase y by β	$\log y = \beta x + \alpha$ Increase of x by 1 will multiply y by $1 + \beta$
$\log x$	$y = \beta \log x + \alpha$ Doubling x will increase y by β	$\log y = \beta \log x + \alpha$ Doubling x will multiply y by $1 + \beta$

Cross terms:
Can include synergetic effects among different features

- Not only the original features $x_1, x_2, \ldots, x_D$, use their cross terms products $\{x_d x_{d'}\}_{d,d'}$

- Model has a matrix parameter $\boldsymbol{W}$:

$$y = \mathrm{Trace}\left(\begin{bmatrix} w_{1,1} & \cdots & w_{1,D} \\ \vdots & \ddots & \vdots \\ w_{D,1} & \cdots & w_{D,D} \end{bmatrix}^\top \begin{bmatrix} x_1^2 & x_1 x_2 & \cdots & x_1 x_D \\ x_2 x_1 & x_2^2 & & x_2 x_D \\ \vdots & & \ddots & \vdots \\ x_D x_1 & x_D x_2 & \cdots & x_D^2 \end{bmatrix} \right)$$

$$= \mathbf{x}^\top \boldsymbol{W}^\top \mathbf{x}$$

- $L(\boldsymbol{W}) = \sum_{i=1}^{N} \left(y^{(i)} - \mathbf{x}^{(i)\top} \boldsymbol{W}^\top \mathbf{x}^{(i)} \right)^2 + \lambda \|\boldsymbol{W}\|_{\mathrm{F}}^2$

Kernels: Linear model in a high-dimensional feature space

- High dimensional non-linear mapping: $\mathbf{x} \to \boldsymbol{\phi}(\mathbf{x})$
  - $\boldsymbol{\phi}: \Re^D \to \Re^{\bar{D}}$ is some nonlinear mapping from D-dimensional space to a $\bar{D}$-dimensional space ($D \ll \bar{D}$)
- Linear model $y = \bar{\mathbf{w}}^\top \boldsymbol{\phi}(\mathbf{x})$
- Kernel regression model: $y = \sum_{i=1}^{N} \alpha^{(i)} k\left(\mathbf{x}^{(i)}, \mathbf{x}\right)$
  - Kernel function $k\left(\mathbf{x}^{(i)}, \mathbf{x}\right) = \langle \mathbf{x}^{(i)}, \mathbf{x} \rangle$: inner product
  - Kernel trick: Instead of working in the $\bar{D}$-dimensional space, we use an equivalent form in an N -dimensional space

Bayesian Statistical Interpretation

Bayesian interpretation of regression: Ridge regression as MAP estimation

- Posterior distribution of parameters
- Maximum A Posteriori (MAP) estimation
- Ridge regression as MAP estimation

Bayesian modeling: Posterior, instead of likelihood

- In maximum likelihood estimation (MLE), we obtain $\mathbf{w}$ that maximizes data *likelihood*:

$$P(\mathbf{y} \mid \boldsymbol{X}, \mathbf{w}) = \prod_{i=1}^{N} f(y^{(i)} | \mathbf{x}^{(i)}, \mathbf{w})$$

or $\log P(\mathbf{y} \mid \boldsymbol{X}, \mathbf{w}) = \sum_{i=1}^{N} \log f(y^{(i)} | \mathbf{x}^{(i)}, \mathbf{w})$

  – The probability of the data reproduced with the parameter: $P(\text{Data} \mid \text{Parameters})$

- In Bayesian modeling, we consider the *posterior distribution* $P(\text{ Parameters} \mid \text{Data })$

  – Posterior distribution is the distribution over model parameters given data

Posterior distribution:
Posterior = likelihood + prior

- Posterior distribution:

$$P(\text{ Parameters} \mid \text{Data }) = \frac{P(\text{ Data} \mid \text{Parameters })P(\text{Parameters})}{P(\text{Data})}$$

(Bayes' formula)

- Log posterior:

$$\log P(\text{ Parameters} \mid \text{Data })$$
$$\propto \underbrace{\log P(\text{ Data} \mid \text{Parameters })}_{\text{Likelihood}} + \underbrace{\log P(\text{Parameters})}_{\text{Prior}}$$

Maximum a posteriori (MAP) estimation: Find parameter that maximizes the posterior

- Log posterior:

$$\log P(\text{Parameters} \mid \text{Data}) =$$
$$\propto \log P(\text{Data} \mid \text{Parameters}) + \log P(\text{Parameters})$$

- Maximum a posteriori (MAP) estimation finds the parameter that maximizes the posterior:
  $\text{Parameters}^* = \text{argmax}_{\text{Parameters}} \log P(\text{Parameters} \mid \text{Data})$
  - MLE considers only $\log P(\text{Data} \mid \text{Parameters})$ part
  - Additional term (Prior) : $\log P(\text{Parameters})$

Ridge regression as MAP estimation: Find parameter that maximizes the posterior

- Log posterior:

$$\log P(\text{ Parameters} \mid \text{Data}\,) =$$
$$\propto \log P(\text{ Data} \mid \text{Parameters}\,) + \log P(\text{Parameters})$$

- Ridge regression:

$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)}\right)^2 + \frac{1}{2\sigma'^2} \|\mathbf{w}\|_2^2$$

  - Log-likelihood: $\sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi}\sigma'} \exp\left(-\frac{\left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)}\right)^2}{2\sigma'^2}\right)$
  - Prior $P(\mathbf{w}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\mathbf{w}^\top \mathbf{w}}{2\sigma^2}\right)$