Supervised Learning

Basic Assumption:

D is a random sample from the underlying distribution *D* defined by the probability density function p(**x**,y).

Example:

The following Data Generating Process (DGP) completely describes a particular joint distribution p(**x**,y). This DGP is very popular and is commonly used in Machine Learning.

$$x \sim N(\mu, \Sigma),$$

$$y = f(x;\theta) + \varepsilon, \quad \varepsilon \sim N(0, \sigma^2) \quad \Rightarrow \quad y \mid x \sim N(f(x;\theta), \sigma^2)$$

Homework question: derive expression for p(**x**,y) (Hint: (**x**,y) is an M+1 dimensional multivariate Gaussian random variable, so you only need to calculate the mean vector and the covariance matrix)

Most General Objective:

Given data set **D** = $\{(\mathbf{x}_i, y_i), i=1, 2, \ldots N\}$ estimate the underlying probability density function p(**x**,y). (Note: the objective is extremely difficult for all but the simple uni-variable scenarios)

Example:

Assuming that the functional form *f* is given, we need to estimate (μ, Σ, θ, σ) from **D**

More Focused Objective for Supervised Learning (probabilistic prediction):

Given data set **D** = $\{(\mathbf{x}_i, y_i), i=1, 2, \ldots N\}$ estimate conditional probability p(y|**x**). (Note1: p(**x**,y) = p(y|**x**)p(**x**); Note2: the objective is still difficult)

Example:

We only need to estimate (θ, σ)

Practical Objective (point prediction):

- Given D learn the mapping f: **X**→Y such that y is "accurately" predicted from f(**x**), for any example taken from the underlying distribution.
- Since we only have a sample **D** from the underlying distribution, the objective translates to: Given D learn the mapping f: **X**→Y such that y is "accurately" predicted from f(**x**), for all i = 1, 2, …N.
- Learning the mapping f is called *the supervised learning*.

Open Questions in Supervised Learning:

- What does "accurately" mean in practice, i.e. what is the accuracy criterion?
- What is the appropriate choice of the functional form *f* ?

Standard Accuracy Measures

In order to assess accuracy of a predictor, we need to define a **loss function** $L(y, f(\mathbf{x}))$, where y is true output, and $f(\mathbf{x})$ is the prediction. Loss function is sometimes denoted with R (risk function).

Given a **predictor** $f(\mathbf{x})$ its **accuracy** is defined as $E_D[L(y, f(\mathbf{x}))]$, where E is expectation and D is the distribution. In words, $E_D[L(y, f(\mathbf{x}))]$ is an expected loss over an example randomly taken from an underlying distribution.

Having the notion of an accuracy measure, we can refine the goal of supervised learning – the goal is to **minimize prediction loss**. Accuracy measures for regression and classification are different.

Regression

Most popular loss function for regression is $L(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$, known as **squared loss**. Accuracy of a predictor in that case is called **Mean Squared Error** (MSE),

$$MSE = E_D[(y - f(\mathbf{x}))^2] = \int_D (y - f(\mathbf{x}))^2 \cdot p(\mathbf{x}, y) \cdot d\mathbf{x} \cdot dy$$

Math note: For a scalar x and any function f, $E[f(x)] = \int_{x \in R} f(x) \cdot p(x) \cdot dx$, where $p(x)$ is the probability density function (pdf).

In general, a loss function can be arbitrarily defined, although typically loss is zero for correct prediction, and lager than zero if the prediction was not correct.

In practice, the available data set is finite. Assuming there are N examples MSE can be estimated as

$$\hat{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y - f(\mathbf{x}))^2 \text{ , note that } MSE = \lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N}(y - f(\mathbf{x}))^2$$

Classification

In general, $L(y, f(\mathbf{x})) = l_{i,j}$, $y \in C_i$, $f(\mathbf{x}) \in C_j$, assuming that we have several distinct types of objects, i.e., K possible classes: $y \in \{C_1, C_2, C_3, \cdots, C_K\}$

In practice, **0-1 (zero-one) loss** is typically used: $L(y, f(\mathbf{x})) = \begin{cases} 1, & y \neq f(\mathbf{x}) \\ 0, & y = f(\mathbf{x}) \end{cases}$

Examples: Suppose we are testing a blood sample for a certain disease. Thus, there are two classes, $y \in \{0, 1\}$, where $y = 1$ means that a person has the disease, and $f(\mathbf{x}) = 1$ means that we predicted that the person has the disease. The 0-1 loss function is defined as follows:

		$f(\mathbf{x})$	
		0	1
y	0	0	1
	1	1	0

What is the price of a mistake? How much would be the cost of a specific kind of a mistake? We may wish to penalize specific kind of mistakes differently:

		$f(\mathbf{x})$	
		0	1
y	0	0	100
	1	10^6	0

If the cost of a mistake is high, we might want to design a very conservative predictor which predicts disease if there is a slightest chance for it.

Optimal Predictors

Regression

The optimal prediction function $f(\mathbf{x})$ minimizes MSE:

$$MSE = \int_D (y - f(\mathbf{x}))^2 \cdot p(\mathbf{x}, y) \cdot d\mathbf{x} \cdot dy = \int_X \left(\int_{Y|X} (y - f(\mathbf{x}))^2 \cdot p(y \mid \mathbf{x}) \cdot dy \right) \cdot p(\mathbf{x}) \cdot dx$$

Therefore, for any given input $\mathbf{x}$ we want to minimize

$$g(A) = \int (y - A)^2 \cdot p(y \mid \mathbf{x}) \cdot dy ,$$

where with A= $f(\mathbf{x})$ (remember that in the inner integral $\mathbf{x}$ is fixed, so it can be considered as a constant).

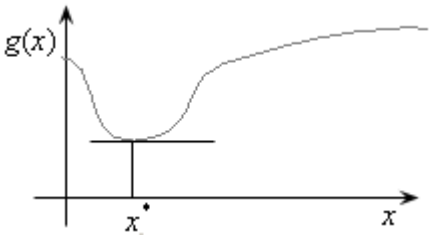
Regression problem is now reduced to:

Find an optimal value of A *that minimizes the integral g*(A).

Math Background:

Unconstrained optimization

Problem: Given a function *g*(**x**) find its minimum (i.e. minimize *g*(**x**))



$$x^* = \arg\min_{x \in R} g(x)$$

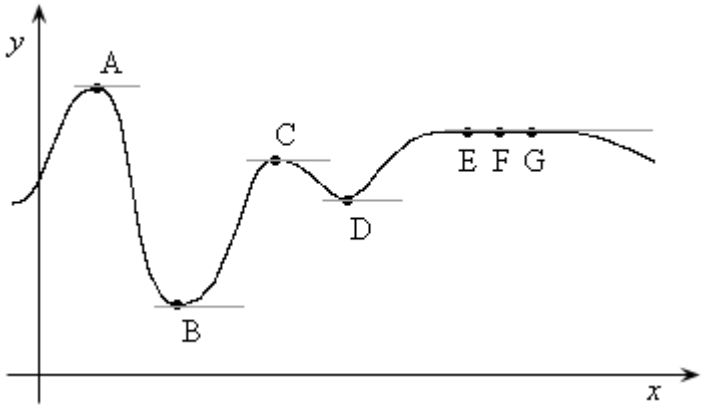
Necessary condition for a data point **x*** to be a minimum:

$\frac{\partial g(x^*)}{\partial x} = 0$, for scalars, $\nabla g(\mathbf{x}^*) = 0$, for vectors (Jacobian, J)

Note: if $\mathbf{x} = [x_1, x_2, \cdots, x_n]$, then $\nabla g(\mathbf{x}) = \begin{bmatrix} \frac{\partial g(\mathbf{x})}{\partial x_1} \\ \vdots \\ \frac{\partial g(\mathbf{x})}{\partial x_n} \end{bmatrix}$

What is the **sufficient** condition for a data point x* to be a minimum?

Example:



If we observe the figure we notice that several points satisfy the necessary condition $\nabla g(\mathbf{x}^*) = 0$: point A is the global maximum, point B is the global minimum, point C is a local maximum, point D is a local minimum, points E, F, G are saddle points.

Sufficient condition for a data point **x*** to be a minimum:

$\frac{\partial^2 g(x)}{\partial x^2} > 0$, for scalars, $\nabla^2 g(\mathbf{x})$ is a positive definite, for vectors (Hessian, H)

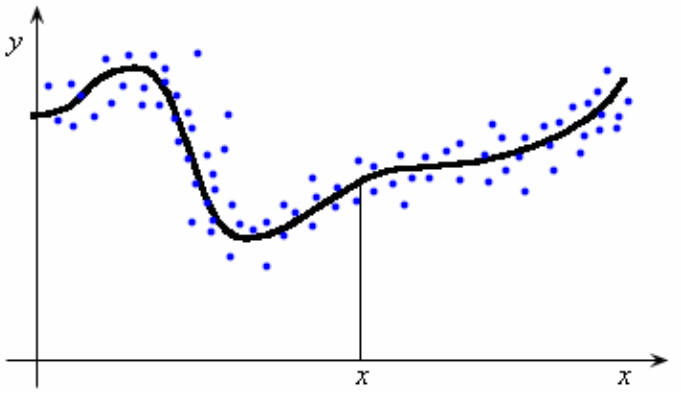
Note: Hessian is defined as H= $\{h_{ij}\}$, $h_{ij} = \frac{\partial^2 g(\mathbf{x})}{\partial x_i \cdot \partial x_j}$

Back to Regression

To find minimum of g(A):

$$\frac{\partial g(A)}{\partial A} = 0 \Rightarrow \int y \cdot p(y \mid \mathbf{x}) \cdot dy = \int A \cdot p(y \mid \mathbf{x}) \cdot dy \Rightarrow E[y \mid \mathbf{x}] = A$$

Therefore, the optimal regression function that minimizes MSE is: $f^*(\mathbf{x}) = E[y \mid \mathbf{x}]$



$E[y \mid \mathbf{x}]$, the optimal function that minimizes the error, is called the **regression function** (hence the term regression).

Note: The optimal solution is a theoretical result. We still have to learn how to estimate $E[y \mid \mathbf{x}]$ as good as possible given a finite data set D.

Classification

The optimal prediction function $f(\mathbf{x})$ minimizes the classification loss:

$$\text{Loss} = \int L(y, f(\mathbf{x})) \cdot p(y \mid \mathbf{x}) \cdot p(\mathbf{x}) \cdot d\mathbf{x} \cdot dy = \int_X \left(\int_{Y|X} L(y, f(\mathbf{x})) \cdot p(y \mid \mathbf{x}) \cdot dy \right) \cdot p(\mathbf{x}) \cdot d\mathbf{x}$$

Therefore, to minimize Loss we need to minimize the inner integral for any given $\mathbf{x}$.

Note that for classification y is a discrete variable. Therefore, the inner integral is actually a sum

$$\sum_{i=1}^{K} L(y \in C_i, f(x)) \cdot P(y \in C_i \mid \mathbf{x}).$$

Notation: $P(y \in C_i \mid \mathbf{x})$ is called **posterior or conditional class probability,** $P(y \in C_i)$ is called **prior class probability**

Therefore, given an input **x**, and by denoting A= $f(\mathbf{x})$, the optimal classification is class A that minimizes $g(A) = \sum_{i=1}^{K} L(y \in C_i, A) \cdot P(y \in C_i \mid \mathbf{x})$, i.e.,

$$A = f^*(\mathbf{x}) = \arg\min_A \sum_{i=1}^{K} L(y \in C_i, A) \cdot P(y \in C_i \mid \mathbf{x}) \quad \text{(Eq.1)}$$

In a special case with 0-1 loss:

$$f^*(\mathbf{x}) = \arg\max_{C_i} P(y \in C_i \mid \mathbf{x}). \text{ (Eq. 2)}$$

Therefore, the optimal decision is the class with the highest posterior!!

Homework question: Prove that Eq. 2 is correct. Please consider a general case with K classes.

Conclusion: If posteriors $P(y \in C_i \mid \mathbf{x}), \quad j = 1, 2, 3, \cdots, K$, are known then the optimal classification is easily calculated

Terminology: The optimal classification function is called the **Bayes classifier**, and the (Eq.1) is called the **Bayes classification (decision) rule**.