Lecture 4

Classification

ISLR 4, ESL 4

Jilles Vreeken
Aleksandar Bojchevski

UNIVERSITÄT DES SAARLANDES

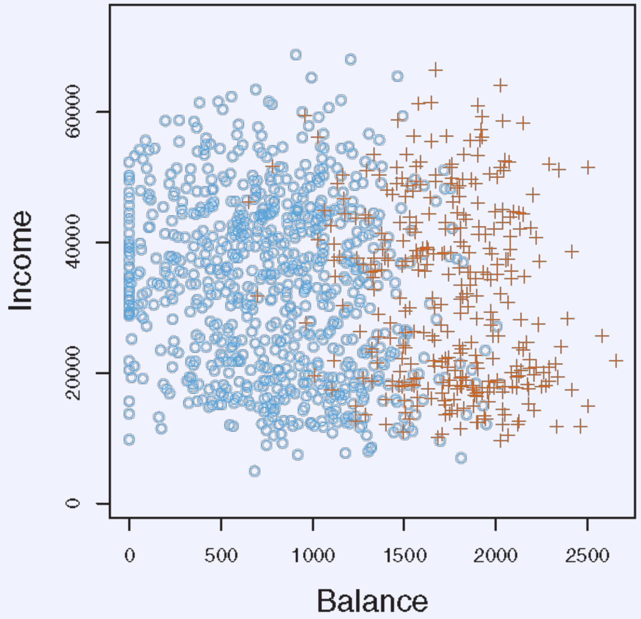
CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Classification Overview

In classification, we want to predict categorical outputs

Example will someone pay back their loan? yes or no?

- inputs: annual **income**, monthly **balance**, **student** status



Classification Overview

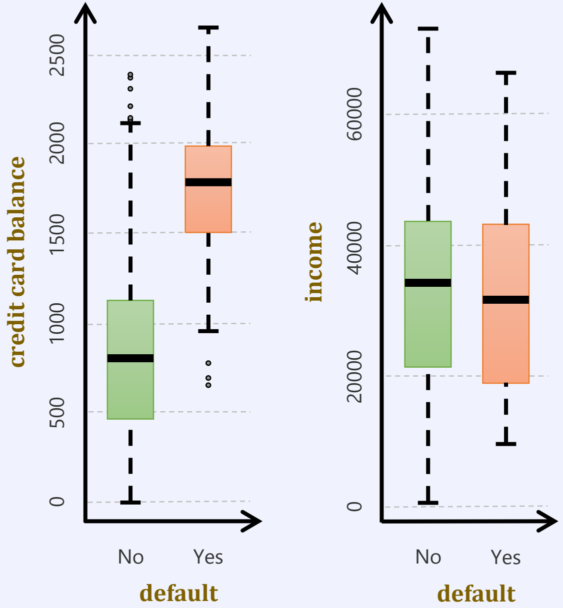
In classification, we want to predict categorical outputs

Example will someone pay back their loan? yes or no?

- inputs: annual **income**, monthly **balance**, **student** status

More examples

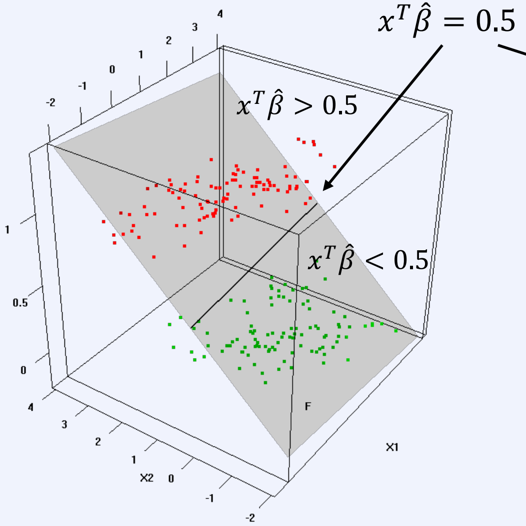
- classify which out of k diseases a patient has given symptoms
- decide whether a transaction is fraudulent based on transaction history, location, IP, DNS, etc.
- identify disease-causing mutations based on DNA sequences from patients with and without a given disease (feature selection)

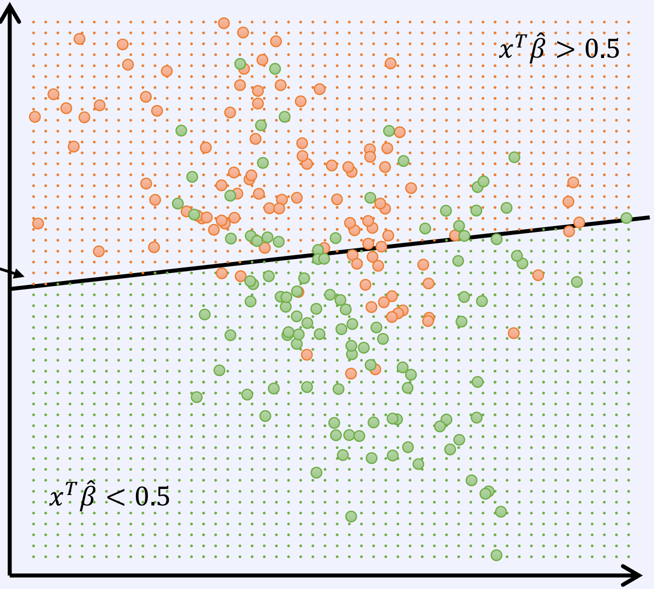


Why not just do linear regression?

Linear regression **can actually work** for binary classification

- simply code $Y = \begin{cases} 0 & \text{if } \textbf{green} \\ 1 & \text{if } \textbf{red} \end{cases}$





Why not just do linear regression?

Linear regression does not generalize to more than two classes

For example, which coding when we have three classes?

- $Y = \begin{cases} 0 & \text{if } \textbf{green} \\ 1 & \text{if } \textbf{red} \\ 2 & \text{if } \textbf{blue} \end{cases}$ or $Y = \begin{cases} 0 & \text{if } \textbf{red} \\ 1 & \text{if } \textbf{blue} \\ 2 & \text{if } \textbf{green} \end{cases}$ or $Y = \begin{cases} 0 & \text{if } \textbf{red} \\ 3 & \text{if } \textbf{blue} \\ 9 & \text{if } \textbf{green} \end{cases}$?
- each imposes a different ordering, and different distances between classes

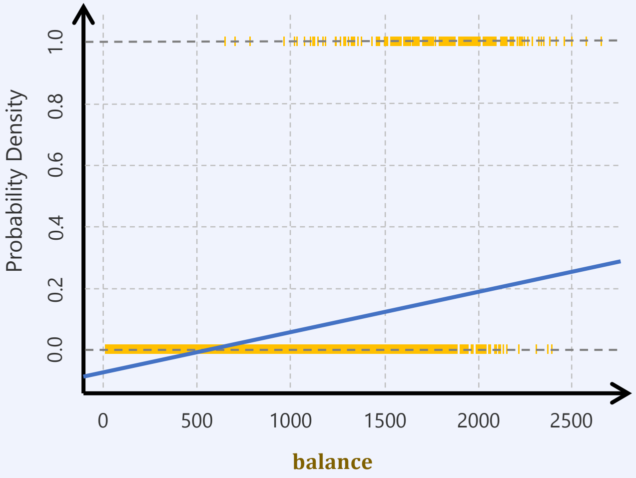
A regression model tries to respect the ordering and numbers representing the classes

- unless we **know** that the labels are metric, we should not impose one as this introduces undue bias
- also, for more than two classes linear-regression has a problem called masking (ESL page 105)

Logistic Regression

Example Credit **default** data

- univariate model, e.g.
$$\Pr(\textbf{default} = \text{yes} \mid \textbf{balance})$$
- simple linear regression models this as
$$f(X) = \beta_0 + \beta_1 X_1$$
- which leads to values outside $[0,1]$



Logistic Regression

Example Credit **default** data

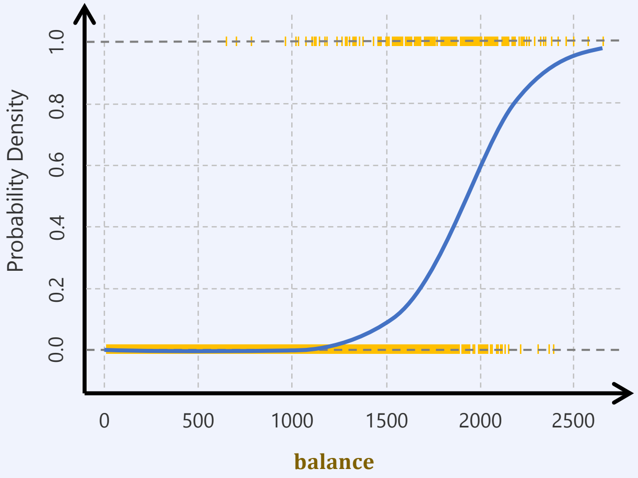
- univariate model, e.g.

$$\Pr(\textbf{default} = \text{yes} \mid \textbf{balance})$$

- simple linear regression models this as

$$f(X) = \beta_0 + \beta_1 X_1$$

- which leads to values outside $[0,1]$



We can map these into $[0,1]$ using the logistic function

$$p(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

probability that
$Y = \textbf{yes} = 1$

Logistic Regression

Example Credit **default** data

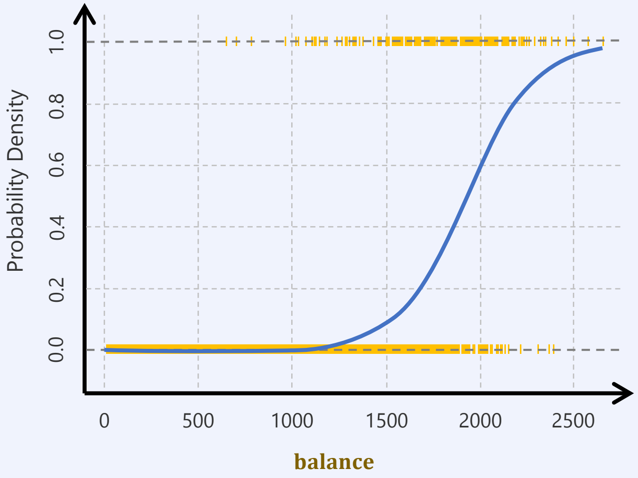
- univariate model, e.g.

$$\Pr(\textbf{default} = \text{yes} \mid \textbf{balance})$$

- simple linear regression models this as

$$f(X) = \beta_0 + \beta_1 X_1$$

- which leads to values outside $[0,1]$



We can map these into $[0,1]$ using the logistic function

probability that $Y = \textbf{yes} = 1$ $\longrightarrow$ $$p(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

- not only are all values now sensible, we also have the

odds ratio as $\frac{p(X)}{1-p(X)} = e^{\beta_0 + \beta_1 X}$, and the log-odds (logit) as $\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$

Interpreting a Logistic Model

If we increasing X by one unit, we

- add β_1 to the log-odds
- multiply the odds by e^{β_1}

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$

Effect on $p(X)$ is non-linear

- if $\beta_1 > 0$, adding X increases $p(X)$
- if $\beta_1 < 0$, adding X decreases $p(X)$

$$\frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}$$

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$

Estimating the Coefficients of Logistic Regression

Maximum Likelihood

- generative approach to find the model that is least surprised to see the given data
- the likelihood function to maximize is

$$p(\boldsymbol{x} \mid \beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1 - p(x_i))$$

- equivalent, but often more practical, is to maximize the log-likelihood

$$\ell(\beta_0, \beta_1) = \sum_{i:y_i=1} \log p(x_i) + \sum_{i:y_i=0} \log(1 - p(x_i))$$

- equivalent, is to minimize the negative log-likelihood (NLL)

We can maximize the likelihood function using nonlinear gradient-descent (Newton-Raphson)

- the intercept only adjusts the average of the fitted probabilities to the proportions of 1s in the data
- in each step we do linear regression, and can hence apply all types of linear model analysis we know

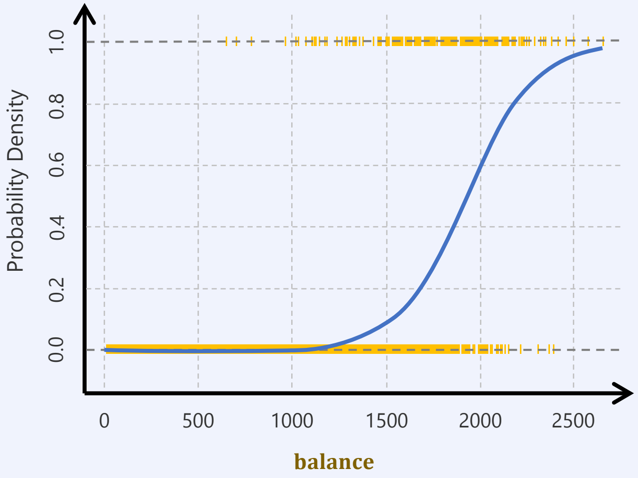
Example Single Continuous Predictor

Probabilities of **default** given **balance**

$$\hat{p}(1000) = \frac{e^{-10.6513+0.0055\times1000}}{1+e^{-10.6513+0.0055\times1000}} = 0.00576$$

$$\hat{p}(2000) = \frac{e^{-10.6513+0.0055\times2000}}{1+e^{-10.6513+0.0055\times2000}} = 0.586$$

- if we increase **balance** by 1 EUR, this
- increases the log odds of defaulting by $\mathbf{0.0055}$
- multiplies the odds of defaulting by $e^{0.0055} = 1.0055\%$



	Coefficient	Std. error	Z-statistic	p-value
intercept	-10.653	0.3612	-29.5	<0.0001
balance	0.0055	0.0002	24.9	<0.0001

Example Single Binary Predictor

Probabilities of **default** given **student**

$$\hat{p}(\textbf{student} = \text{yes}) = \frac{e^{-3.5041+0.40409\times 1}}{1 + e^{-3.5041+0.40409\times 1}} = 0.00431$$

$$\hat{p}(\textbf{student} = \text{no}) = \frac{e^{-3.5041+0.40409\times 0}}{1 + e^{-3.5041+0.40409\times 0}} = 0.00292$$

	Coefficient	Std. error	Z-statistic	p-value
intercept	-3.5041	0.0707	-49.55	<0.0001
student	0.4049	0.1150	3.52	0.0004

Multiple Logistic Regression

The multivariate logistic regression model is defined as

- $\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X + \cdots \beta_p X_p$ with $p(X) = \frac{e^{\beta_0+\beta_1 X+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X+\cdots+\beta_p X_p}}$

Example predicting **default** based on **balance**, **income**, and **student**

$$\hat{p}(\textbf{student} = \text{yes}, \textbf{balance} = 1{,}500, \textbf{income} = 40) = \frac{e^{-10.869+0.00574\times 1{,}500+0.003\times 40-0.6468\times 1}}{1 + e^{-10.869+0.00574\times 1{,}500+0.003\times 40-0.6468\times 1}} = 0.058$$

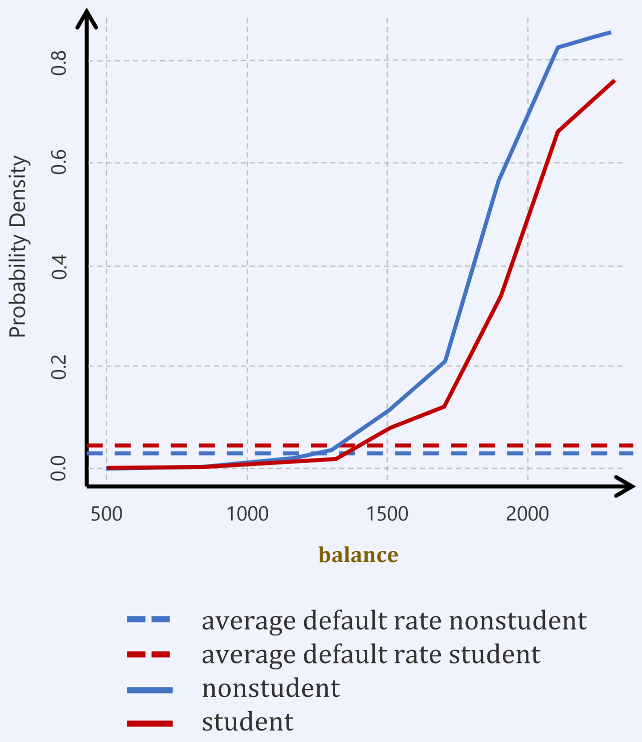
$$\hat{p}(\textbf{student} = \text{no}, \textbf{balance} = 1{,}500, \textbf{income} = 40) = \frac{e^{-10.869+0.00574\times 1{,}500+0.003\times 40-0.6468\times 0}}{1 + e^{-10.869+0.00574\times 1{,}500+0.003\times 40-0.6468\times 0}} = 0.105$$

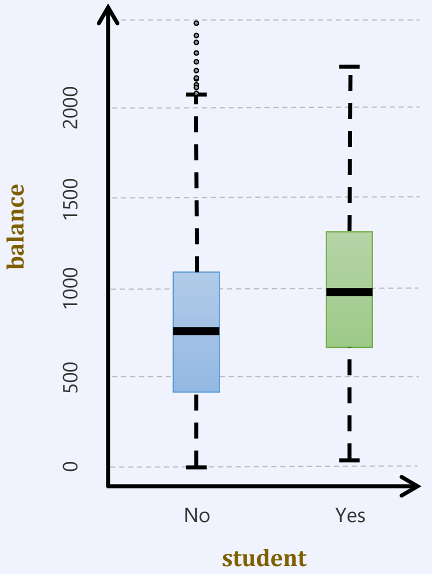
	Coefficient	Std. error	Z-statistic	p-value
intercept	-10.8690	0.4923	-22.08	<0.0001
balance	0.0057	0.0002	24.74	<0.0001
income	0.0030	0.0082	0.37	0.7115
student [yes]	-0.6468	0.2362	-2.74	0.0062

Example Confounding in Logistic Regression

Why is the **student** coefficient positive in the univariate and negative in the multivariate model?

- confounding!
- students have higher **balance**
- students **default** at higher **balance**
- for a fixed value of **balance** and **income**, a **student** is less likely to default than a nonstudent!





Fitting Logistic Regression Models

We usually fit a logistic regression model by maximum likelihood

- log-likelihood function $\ell(\theta) = \sum_{i=1}^{n} \log p_{g_i}(x_i;\theta)$ and density function $p_k(x_i,\theta) = \Pr(\, G = k \mid X = x_i; \theta\,)$
- for a binary problem, class coding $y_i = \begin{cases} 1 \mid g_i = 1 \\ 0 \mid g_i = 0 \end{cases}$ gives us $p_1(x;\theta) = p(x;\theta)$ and $p_2(x;\theta) = 1 - p(x;\theta)$

The log-likelihood then becomes

$$\ell(\beta) = \sum_{i=1}^{n} \{ y_i \log p(x_i;\theta) + (1 - y_i) \log\big(1 - p(x_i;\theta)\big) \} = \sum_{i=1}^{n} \left\{ y_i \beta^T x_i - \log\left(1 + e^{\beta^T x_i}\right) \right\}$$

- where $\beta = \{\beta_0, \beta_1, \dots\}$ and x_i a vector of the input values padded with a constant term $X_0 = 1$

(ESL 4.4.1)

Side calculation

$$\ell(\beta) = \sum_{i=1}^{n} \{y_i \log p(x_i;\theta) + (1-y_i)\log(1-p(x_i;\theta))\}$$

$$= \sum_{i=1}^{n} \left\{ y_i \log \frac{e^{\beta_{10}+\beta_1^T x_i}}{1+e^{\beta_{10}+\beta_1^T x_i}} + (1-y_i)\log \frac{1}{1+e^{\beta_{10}+\beta_1^T x_i}} \right\}$$ (definition of $p(x_i;\theta)$)

$$= \sum_{i=1}^{n} \left\{ y_i \left[\left(\beta_{10}+\beta_1^T x_i\right) - \log\left(1+e^{\beta_{10}+\beta_1^T x_i}\right)\right] - (1-y_i)\log(1+e^{\beta_{10}+\beta_1^T x_i}) \right\}$$ ($\log a/b \; = \; \log a - \log b$)

$$= \sum_{i=1}^{n} \left\{ y_i \beta^T x_i - \log\left(1+e^{\beta^T x_i}\right) \right\}$$ (simplify)

(ESL 4.4.1)

Fitting Logistic Regression Models

We find the β that achieves maximum likelihood by setting the derivative to zero

- this yields the score equations

$$\frac{\partial \ell(\beta)}{\partial \beta} = \sum_{i=1}^{n} x_i \big(y_i - p(x_i;\beta)\big) = 0$$

- these can be broken down to $p + 1$ equations that are nonlinear in β
- because the first value of x_i is 1, the first equation takes the shape

$$\sum_{i=1}^{n} y_i = \sum_{i=1}^{n} p(x_i;\beta)$$

- the expected number of class-1 assignments is the number class-1 we observed

(ESL 4.4.1)

Fitting Logistic Regression Models

We can solve the score equations numerically using Newton-Raphson

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 \ell(\beta^{old})}{\partial\beta\partial\beta^T}\right)^{-1} \frac{\partial \ell(\beta^{old})}{\partial\beta}$$

- i.e. adjust coefficients proportionally to second derivative in the opposite direction of first derivative
- repeat until convergence
- note that $\frac{\partial^2 \ell(\beta^{old})}{\partial\beta\partial\beta^T} = -\sum_{i=1}^{n} x_i x_i^T p(x_i;\beta)\big(1 - p(x_i;\beta)\big)$ is our old friend, the Hessian matrix!

Log-likelihood is concave

- single starting point suffices, $\boldsymbol{\beta} = \mathbf{0}$ is fine
- typically converges, but overshooting can occur
- diagonal of the Hessian matrix contains the squared standard deviations of outputs in the training set

(ESL 4.4.1)

Fitting Logistic Regression Models

In matrix notation we have

$$\frac{\partial \ell(\beta)}{\partial \beta} = \mathbf{X}^T(\mathbf{y} - \mathbf{p}) \qquad \frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^T} = -\mathbf{X}^T\mathbf{W}\mathbf{X}$$

- where $\mathbf{W}$ is a diagonal matrix with elements $w_{ii} = -p(x_i; \beta^{old})\left(1 - p(x_i; \beta^{old})\right)$

A single Newton-Raphson step is

$$\beta^{new} = \beta^{old} - \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}^T(\mathbf{y} - \mathbf{p}) = \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{W}\left(\mathbf{X}\beta^{old} - \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p})\right) = \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z}$$

$$\mathbf{z} = \mathbf{X}\beta^{old} - \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p})$$

- a linear least-squares problem with output $\mathbf{z}$ weighted by diagonal matrix $\mathbf{W}$

$$\beta^{new} = \arg\min_{\beta}(\mathbf{z} - \mathbf{X}\beta)^T\mathbf{W}(\mathbf{z} - \mathbf{X}\beta)$$

(ESL 4.4.1)

Classification Discriminative vs. Generative

	Discriminative	Generative
Output for an input x	estimate $\hat{g}(x)$ of class $g(x)$	probability distribution $\{p_g(x) \mid g \in G\}$, $p_g(x)$ is the probability that x belongs to class g
Main idea	the classifier returns an estimate of the output, which discriminates between different classes	the classifier generates the output with some probability
Performance measure	loss function that measures the deviation between estimate and output, e.g. 0-1 loss	(log-)likelihood of the estimator generating the output $\sum_{i=1}^{n} \log p_{g_i}(x)$
Optimization problem	Minimize the loss function	Maximize the likelihood