Classification Methods

Alexander Jung

Assistant Professor

Department of Computer Science

Aalto University

Machine Learning With Python

FITech NETWORK UNIVERSITY

Aalto-yliopisto

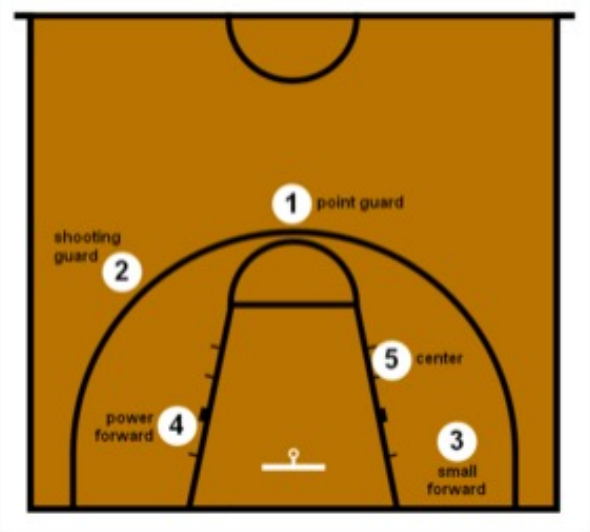
Classification Problems

labels take on values from a finite set

features: height, weight, passing stat., scoring statistics, rebound stat.

label: position "Shooting Guard"

Finite Set of Label Values



Guards	1. Point Guard	Combo Guard (PG/SG)
	2. **Shooting Guard**	Swingman (SG/SF)
Forwards	3. Small Forward	Point Forward (SF/PG, PF/PG)
	4. Power Forward	Combo Forward (PF/SF)
Center	5. Center	Forward-Center (PF/C)

label can take on 10 different values

There is No (Obvious) Distance!

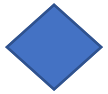
classifier 1 : $\hat{y}$ =“Dog”

classifier 2 : $\hat{y}$ =“Human”

true label y = “Fish”

which classifier is “closer” to true label ?

Binary Classification Problems

- label can take on only two different values
- for notational convenience we use values +1 and -1
- we could also use 0 and 1 as the two label values
- or we could use  and  as two label values

A Binary Classification Problem

Insurance Claim ????
Name:
Date:
Case:

→ maybe fraudulent

Insurance Claim ????
Name:
Date:
Case:

→ most likely not fraudulent

Classifiers

- data point with features x and label y
- we want to learn a classifier map h(x)
- h(x) reads in features x and outputs a prediction for label
- choose or learn classifier which best matches true labels

Classifiers and Classification

- allow classifier map h(x) to be real-valued
- obtain classification by thresholding

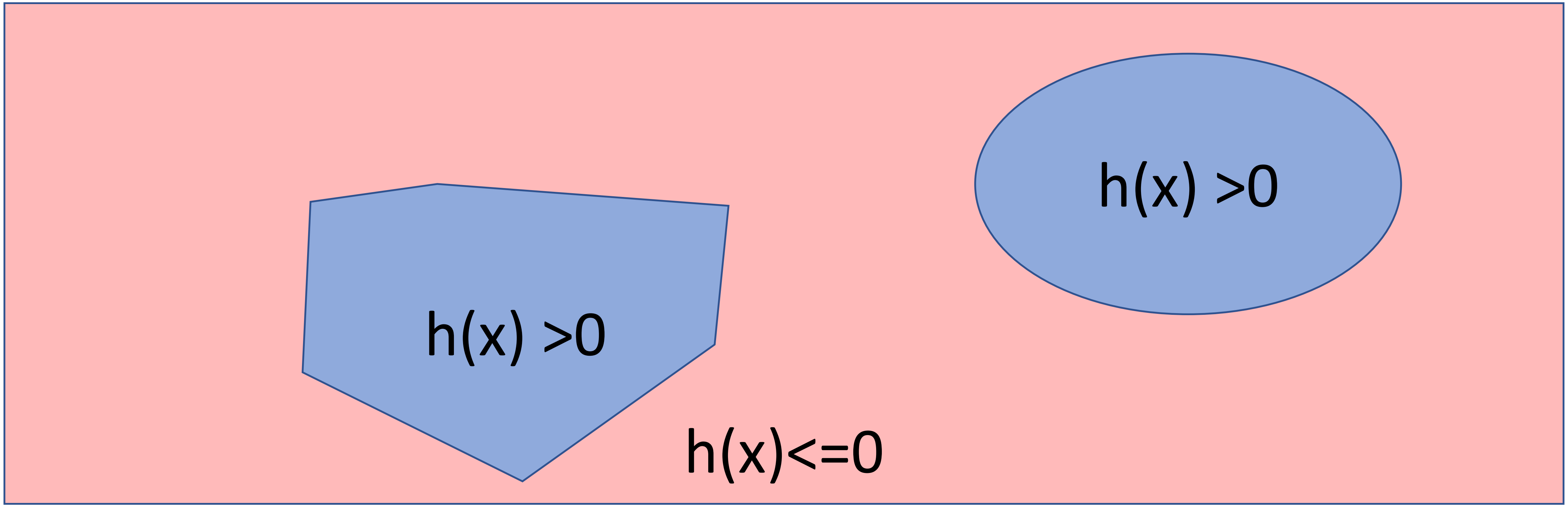
$$\hat{y} = \begin{cases} 1 \; if \; h(x) > 0 \\ -1 \; otherwise. \end{cases}$$

- absolute value |h(x)| is confidence/trust in classification

Decision Regions

- consider data point with two features $\boldsymbol{x} = (x_1, x_2)$

- classifier divides feature space into two components



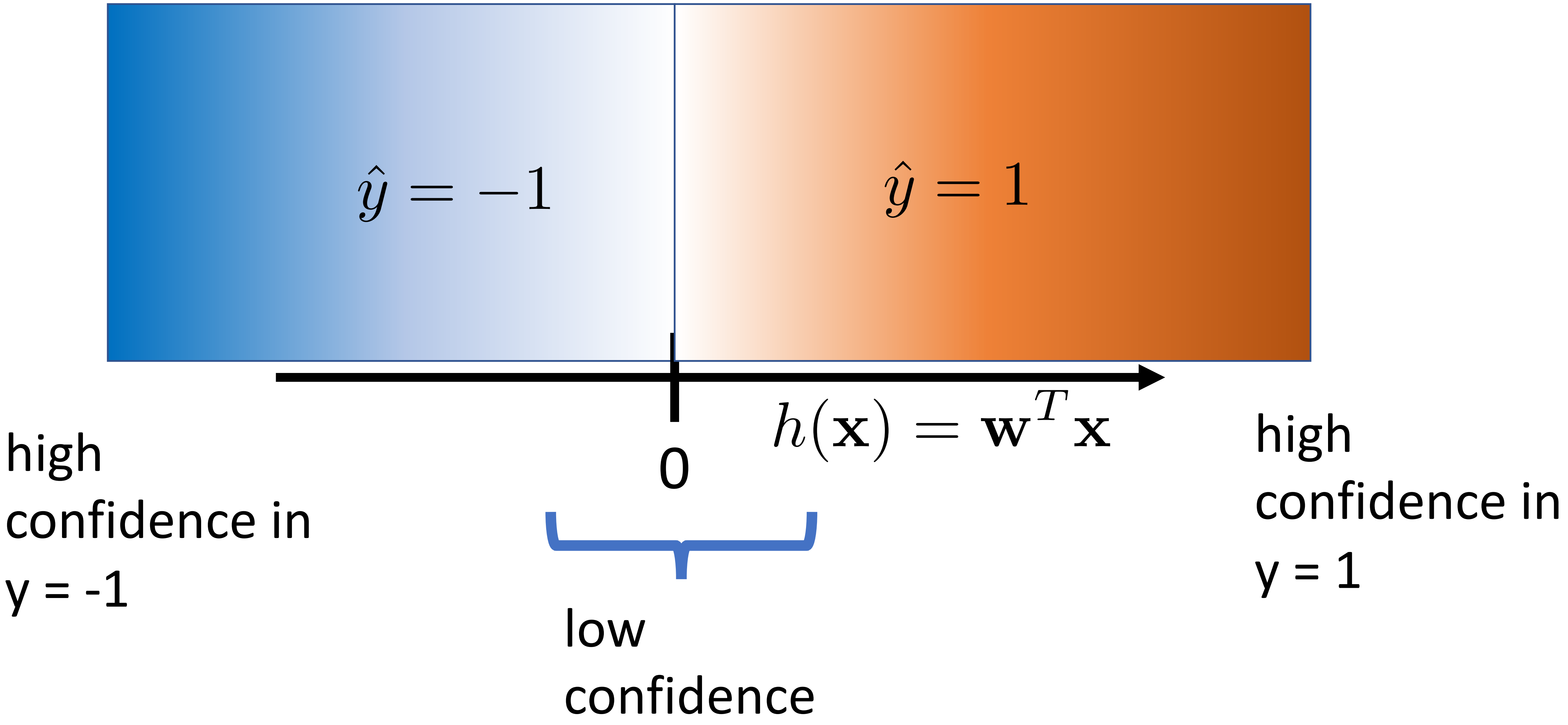
Logistic Regression

- data points with numeric features $\mathbf{x} = \left(x_1, \ldots, x_n\right)^T \in \mathbb{R}^n$
- binary label $y \in \{-1, 1\}$
- linear predictor map $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x} = \sum_{i=1}^{n} w_i x_i$
- predictor defined by weights $\mathbf{w} = \left(w_1, \ldots, w_n\right)^T$

Logistic Regression – The Classifier

- linear predictor h(x) can be any real number
- what if h(x)= 0.333 ? what if h(x)=-100.44 ?
- we use sign of h(x) to classify: $\hat{y} = \begin{cases} 1 & \text{if } h(\mathbf{x}) \geq 0 \\ -1 & \text{if } h(\mathbf{x}) < 0 \end{cases}$
- we use magnitude |h(x)| as reliability (confidence) measure

Logistic Regression – The Classifier



Probabilistic Machine Learning

- interpret label y as realization of random variable
- fraction of data points with y=1 is probability Prob(y=1)
- probability Prob(y=1) depends on features x !
- logistic regression aims to predict (estimate) Prob(y=1)

Logistic Regression – Probabilistic Model

- interpret label y as realization of random variable

- define confidence in y=1 via "log-odds" $\log \frac{\text{Prob}\{y=1\}}{\underbrace{(1-\text{Prob}\{y=1\})}_{\text{Prob}\{y=-1\}}}$

- since we predict confidence using h(x), we obtain

$$\text{Prob}\{y=1\} = \frac{1}{1+\exp(-h(\mathbf{x}))} = \frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x})}$$

- probability distribution parametrized by weights w of predictor!

Logistic Regression – Towards a Loss Function

- consider m labeled data points $\left(\mathbf{x}^{(i)}, y^{(i)}\right),\ i=1,\ldots,m$

- labels are realizations of independent random variables

$$\text{Prob}\{y^{(i)}=1\}=\frac{1}{1+\exp(-\mathbf{w}^T\mathbf{x}^{(i)})},\ \text{and}\ \text{Prob}\{y^{(i)}=-1\}=\frac{1}{1+\exp(\mathbf{w}^T\mathbf{x}^{(i)})}$$

- probability of actually observing the labels

$$\text{Prob}\left\{y^{(1)},\ldots,y^{(m)}\right\} = \prod_{i=1}^{m}\text{Prob}\{y^{(i)}\}$$

$$= \prod_{i=1}^{m}\frac{1}{1+\exp(-y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)})}$$

Logistic Regression – Maximum Likelihood

- learn weights by maximizing probability of data !

$$\widehat{\mathbf{w}} = \text{argmax}_{\mathbf{w}\in\mathbb{R}^n} \prod_{i=1}^{m} \frac{1}{1+\exp(-y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)})}$$

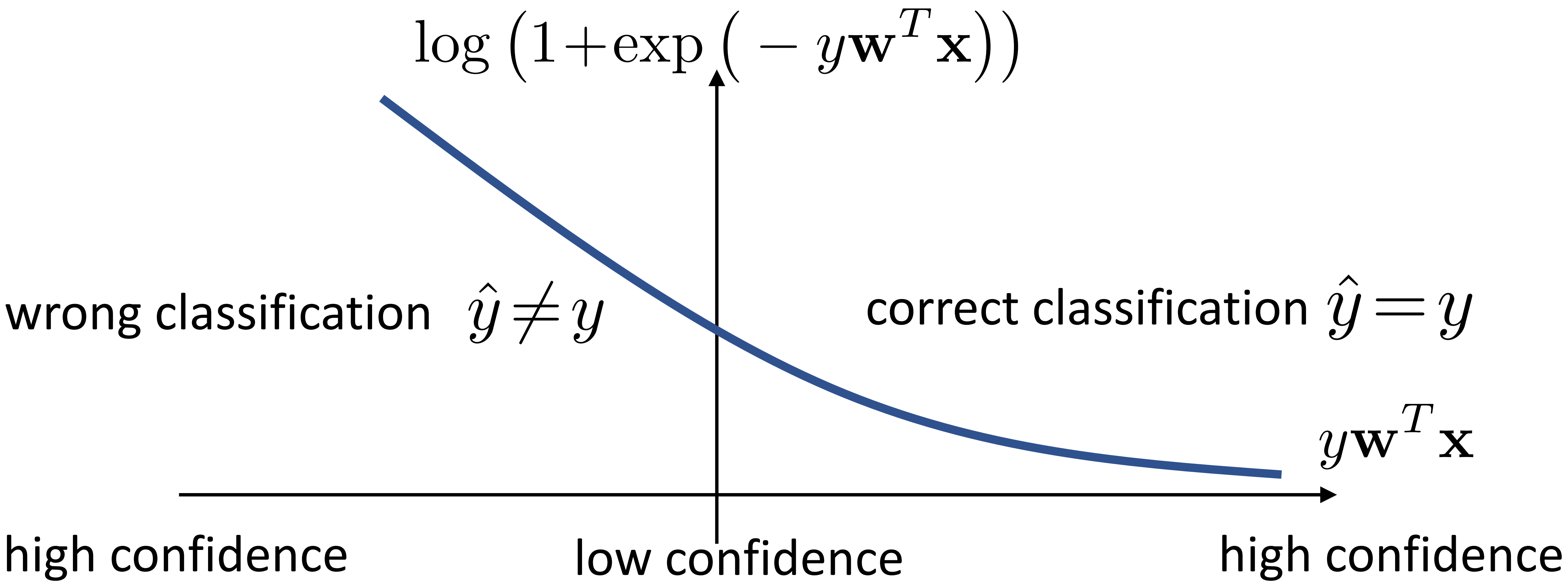
$$= \text{argmax}_{\mathbf{w}\in\mathbb{R}^n} \log \prod_{i=1}^{m} \frac{1}{1+\exp(-y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)})}$$

$$= \text{argmin}_{\mathbf{w}\in\mathbb{R}^n} -\log \prod_{i=1}^{m} \frac{1}{1+\exp(-y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)})}$$

$$= \text{argmin}_{\mathbf{w}\in\mathbb{R}^n} \sum_{i=1}^{m} \log\left(1+\exp(-y^{(i)}\mathbf{w}^T\mathbf{x}^{(i)})\right)$$

Logistic Regression – Logistic Loss

max. probability of data = minimizing the average logistic loss



ID-Card of Logistic Regression

sklearn.linear_model.LogisticRegression

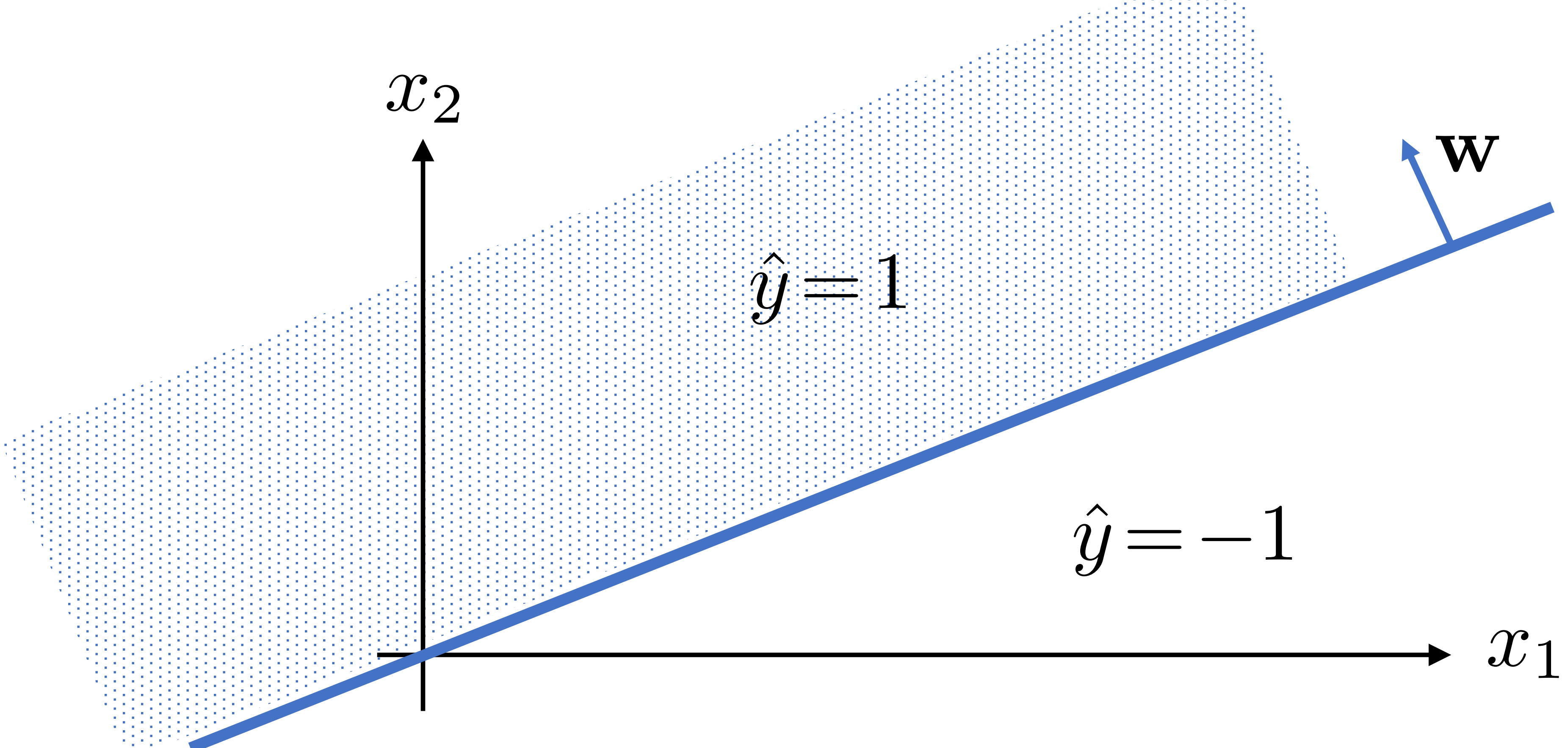
linear_model.**LogisticRegression**(*penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercep*
g=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='au

- features: real numbers
- labels: two different values (categories)
- hypothesis space: linear predictor maps
- loss: logistic loss
- instance of a linear classifier

Linear Classifiers

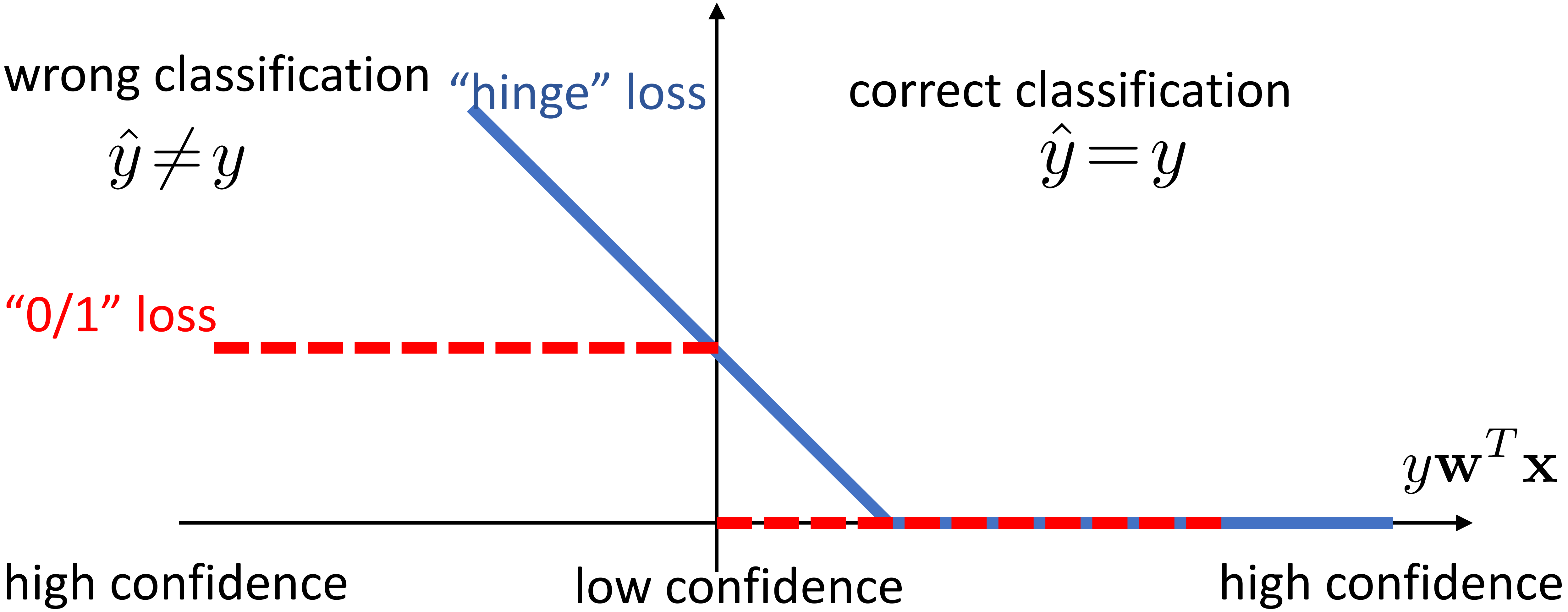
- linear classifiers predict label y using $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$

- linear classifiers differ in how to learn weights w

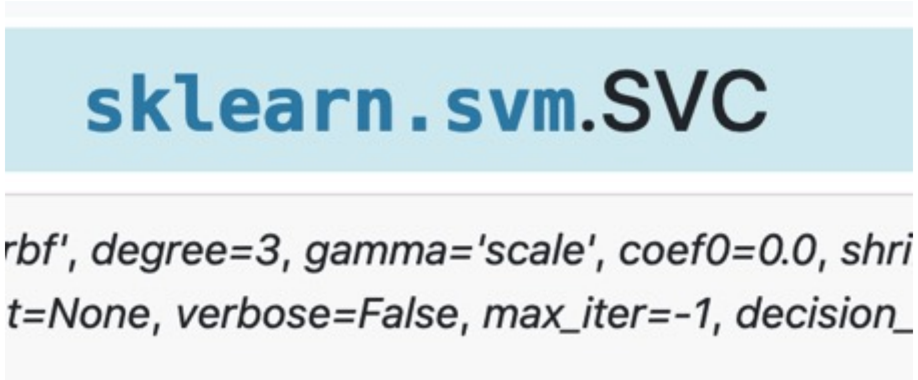


Loss Functions for Linear Classifiers

learn linear predictors using different loss functions



ID-Card of Support Vector Classifier



- features: real numbers
- labels: two different values (categories)
- hypothesis space: linear predictor maps
- loss: hinge loss
- instance of a linear classifier

ID-Card of Naive Bayes Classifier

sklearn.naive_bayes.GaussianNB

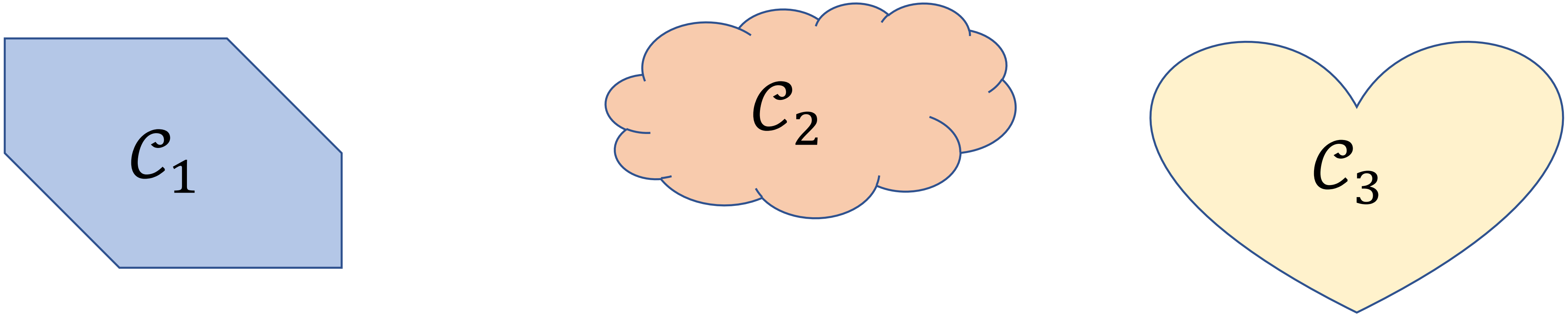
`e_bayes.GaussianNB(priors=None, var_smoothing=1e-09)`

- features: real numbers
- labels: two different values (categories)
- hypothesis space: linear predictor maps
- loss: 0/1 loss
- instance of a linear classifier

You Can Do Anything with Linear Classifiers!

- data points with two numeric features z_1 and z_2
- some (complicated) regions $\mathcal{C}_j$, j=1,…,n, of feature space
- construct new features of data point by

$$x_j = \begin{cases} 1 & when \quad (z_1, z_2) \epsilon \mathcal{C}_j \\ 0 & otherwise. \end{cases}$$



So What?

- logistic regression uses linear predictors
- predictors are interpreted as "log-odds"
- predictor weights parametrize the probability of y=1
- learning weights is a statistical inference problem
- inference problem equivalent to minimizing logistic loss

So What?

- logistic regression one instance of linear classifier
- other linear classifiers obtained from different loss
- support vector classifier uses hinge loss
- Naïve Bayes classifier based on 0/1 loss
- loss functions differ in statistical and computational properties

Thank You !

Machine Learning
With Python

FITech
NETWORK UNIVERSITY