

Classification

Alex(ander) Jung

Assistant Professor for Machine Learning

Department of Computer Science

Aalto University

Reading.

- Ch. 2.3, 3.6 of MLBook

Classification

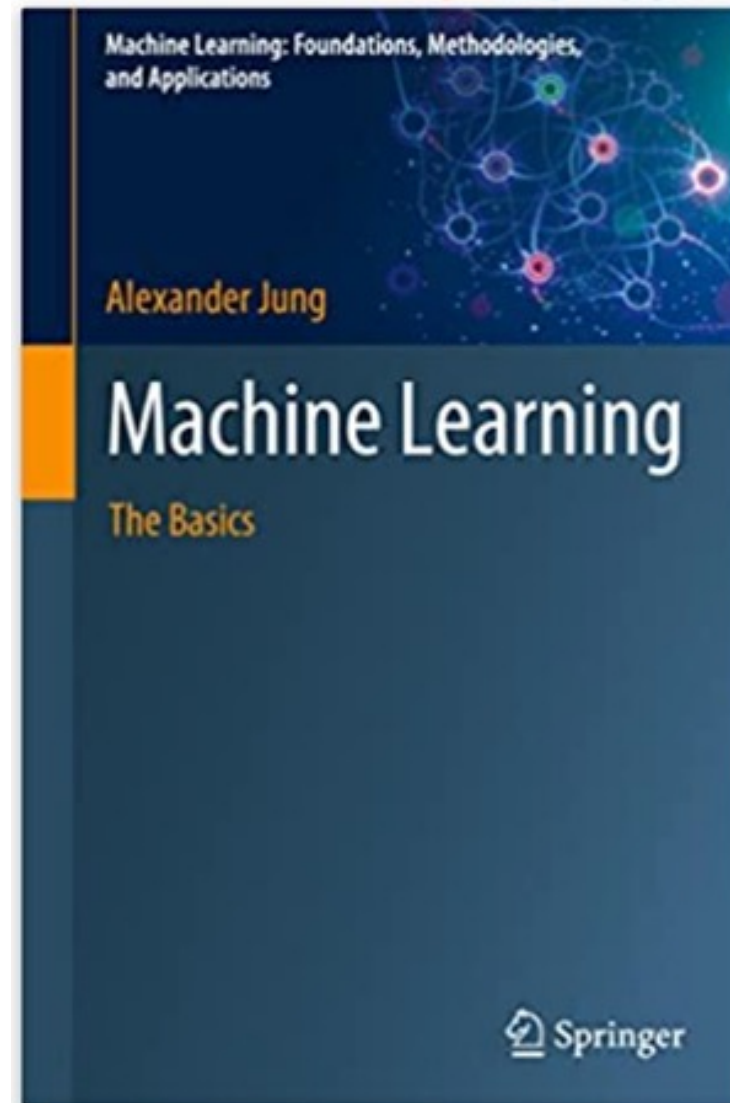
Identifying which category an object belongs to.

Applications: Spam detection, image recognition.

Algorithms: SVM, nearest neighbors, random forest, and more...



Examples



<https://scikit-learn.org/stable/index.html>

Learning Goals:

- be able to recognize classification problems
- know **binary**, **multi-class** and **multi-label** problems
- know design choices of basic classif. methods
- know **stat./comp. trade-offs** in classif. methods

What is ML About ?

fit **models** to **data** to make
predictions or forecasts !

Data. Model. Loss.

data: set of datapoints (x,y)

model: set of hypothesis maps $h(.)$

loss: quality measure $L((x,y),h)$

Machine Learning.

find hypothesis in model that incurs
smallest loss when predicting label of
any datapoint

Expected Loss or Risk

$$\mathbb{E}\{L((\mathbf{x}, y), h)\} := \int_{\mathbf{x}, y} L((\mathbf{x}, y), h) dp(\mathbf{x}, y). \quad (2.14)$$

note: to compute this expectation
we need to know the probability distribution
 $p(\mathbf{x}, y)$ of datapoints (\mathbf{x}, y)

Empirical Risk

IDEA: approximate expected loss by average loss on some datapoints (training set)

$$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(m)}, y^{(m)})\}.$$

$$\mathbb{E}\{L((\mathbf{x}, y), h)\} \approx (1/m) \sum_{i=1}^m L((\mathbf{x}^{(i)}, y^{(i)}), h) \text{ for sufficiently large sample size } m. \quad (2.17)$$

with the average loss or **empirical risk**

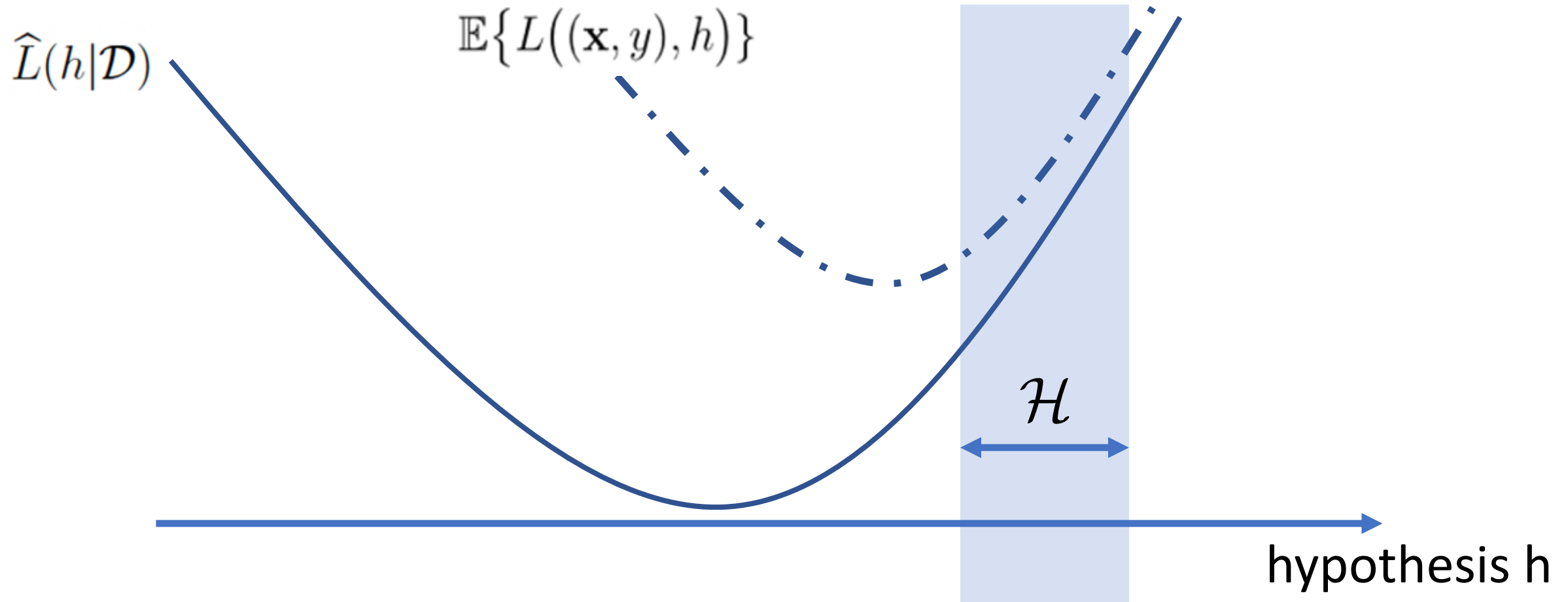
$$\hat{L}(h|\mathcal{D}) = (1/m) \sum_{i=1}^m L((\mathbf{x}^{(i)}, y^{(i)}), h). \quad (2.16)$$

Empirical Risk Minimization

$$\hat{h} \in \operatorname{argmin}_{h \in \mathcal{H}} \hat{L}(h|\mathcal{D})$$

$$\stackrel{(2.16)}{=} \operatorname{argmin}_{h \in \mathcal{H}} (1/m) \sum_{i=1}^m L((\mathbf{x}^{(i)}, y^{(i)}), h).$$

Empirical Risk Minimization





ERM for Parametrized Models

learnt (optimal) parameter vector


$$\hat{\mathbf{w}} = \operatorname{argmin}_{\mathbf{w} \in \mathbb{R}^n} f(\mathbf{w})$$

loss incurred by $h(\cdot)$
for i -th data point

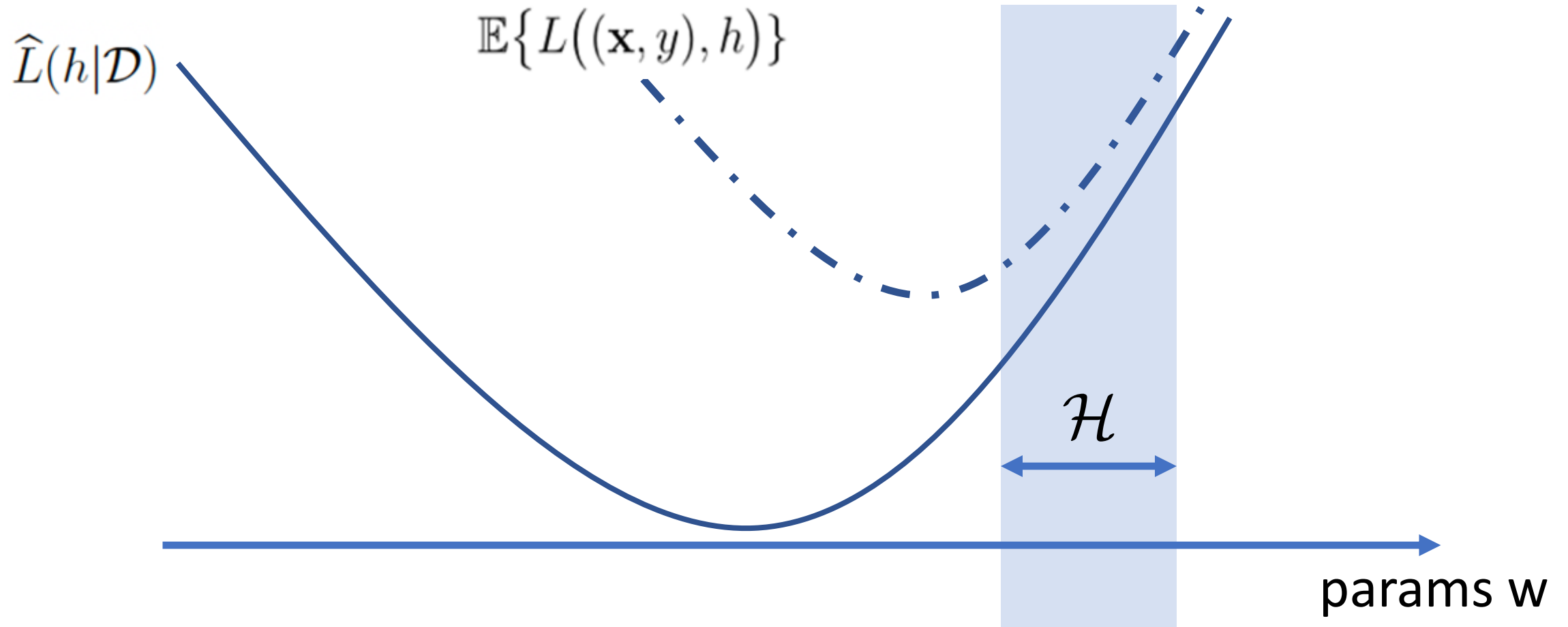

$$\text{with } f(\mathbf{w}) := (1/m) \sum_{i=1}^m L((\mathbf{x}^{(i)}, y^{(i)}), h^{(\mathbf{w})}) .$$


$$\hat{L}(h^{(\mathbf{w})} | \mathcal{D})$$

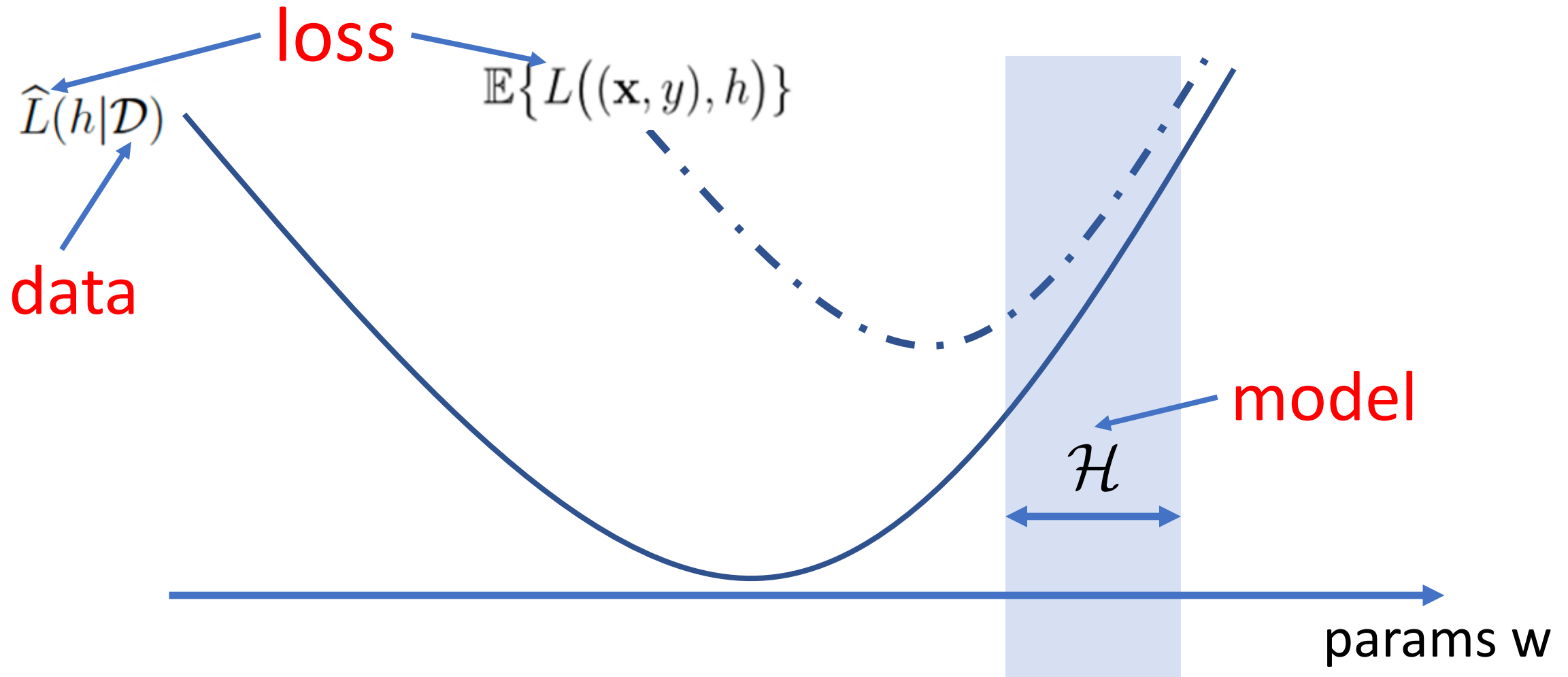


average loss or
empirical risk

ERM for Param. Models



Design Choices in ERM



yesterday (“Regression”): numeric labels, loss functions obtained from distance between numbers

today (“Classification”): discrete-valued labels, loss functions obtained from “confidence” measures

Logistic Regression

[Sec. 3.6., MLBook]

LogReg vs. LinReg

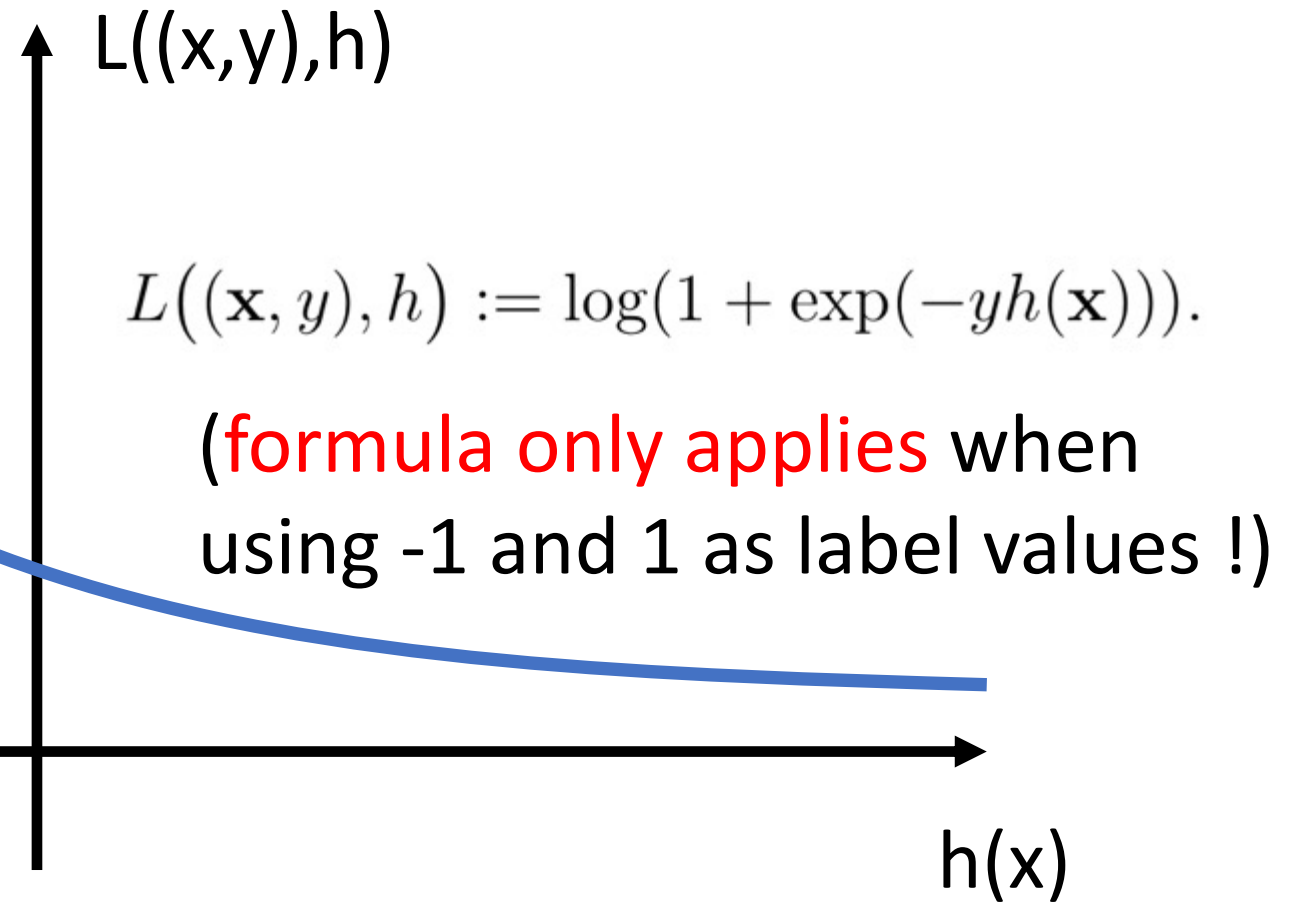
- **data** points with numeric features (same as lin.reg.)
- binary label values, e.g., $y=-1$ vs. $y=1$ (diff. from lin.reg.)
- **model** = space of linear maps (same as lin.reg!)
- logistic **loss** (different from lin.reg!)

Linear Classifier

- log.reg. uses linear hypothesis $h(x) = w'x$
- sign of $h(x)$ used for label prediction
- $|h(x)|$ used as confidence measure
- $h(x) = 100000$ means very confident in $\hat{y}=1$
- $h(x) = -100000$ very confident in $\hat{y}=0$

Logistic Loss

differentiable and
convex as function of $h(x)$
and, in turn, of weight w
for linear $h(x) = w' x$



Logistic Loss Formulas

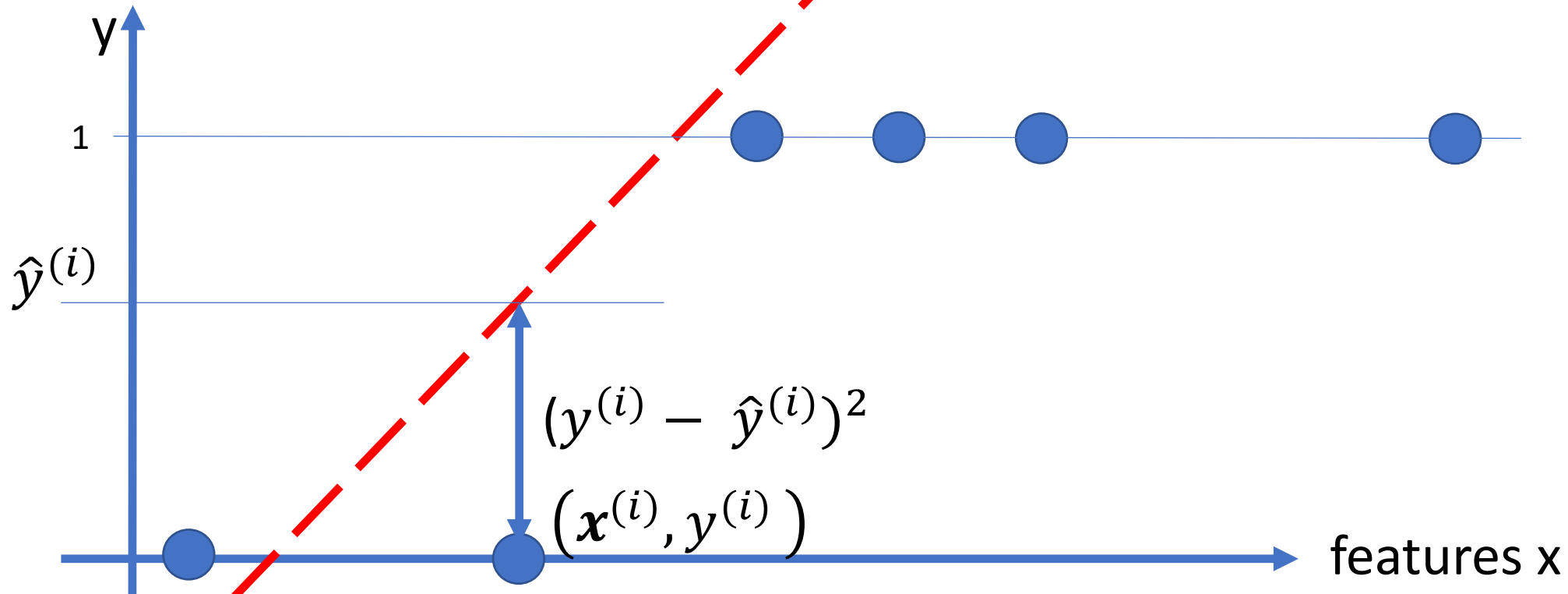
-1 and 1

0 and 1

"A" and "B"

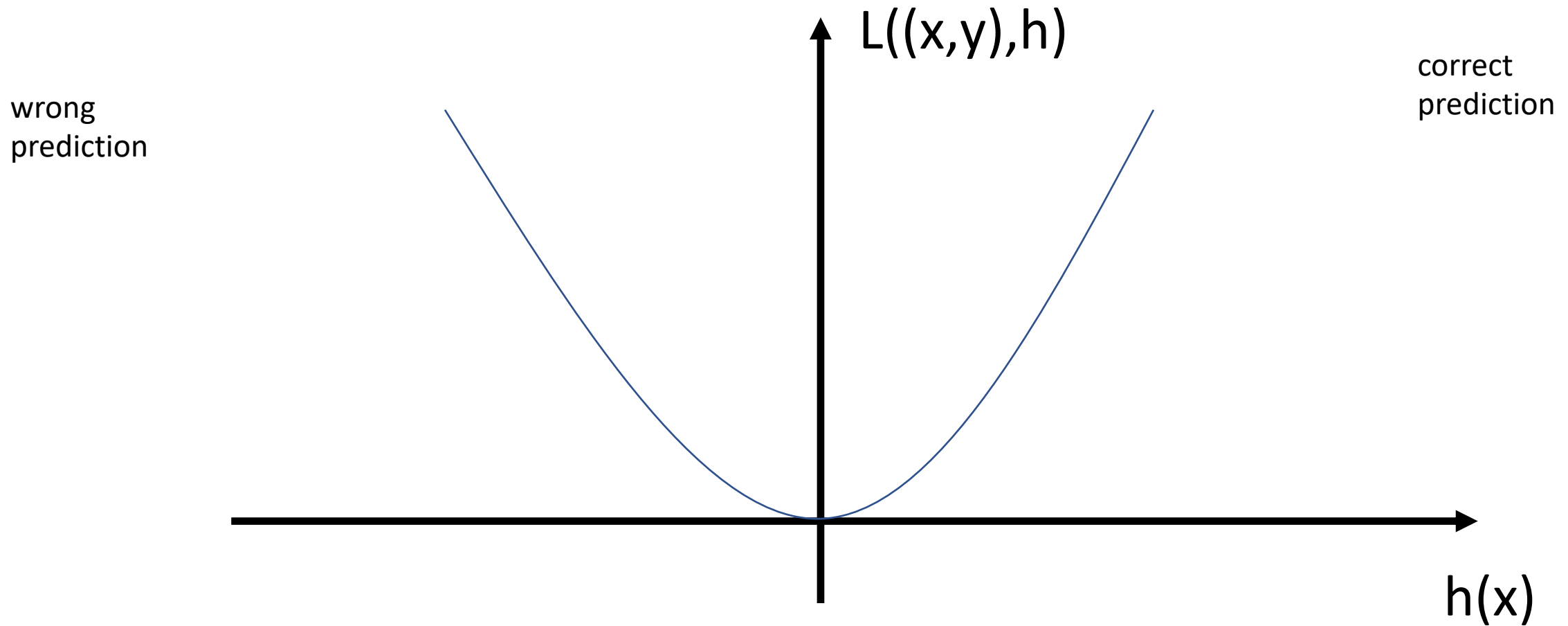
$$h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$$

Why not Squared Loss?



choose parameter/weight vector \mathbf{w} to
minimize average squared error loss

Squared Error Loss



Classif. Loss Functions

[Sec 2.3.3, MLBook]

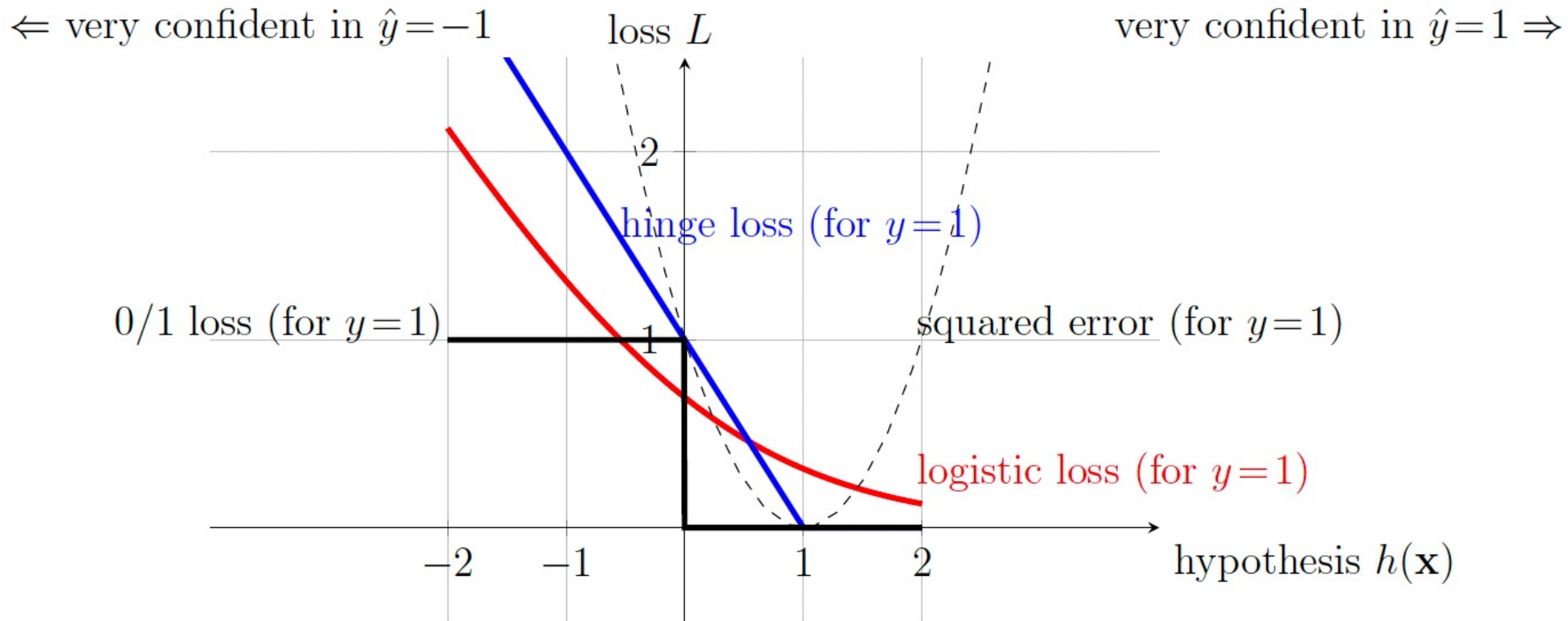


Figure 2.15: The solid curves depict three widely-used loss functions for binary classification.

LogReg. Probabilistic Interpretation

interpret label of data point as realization of binary RV with prob.

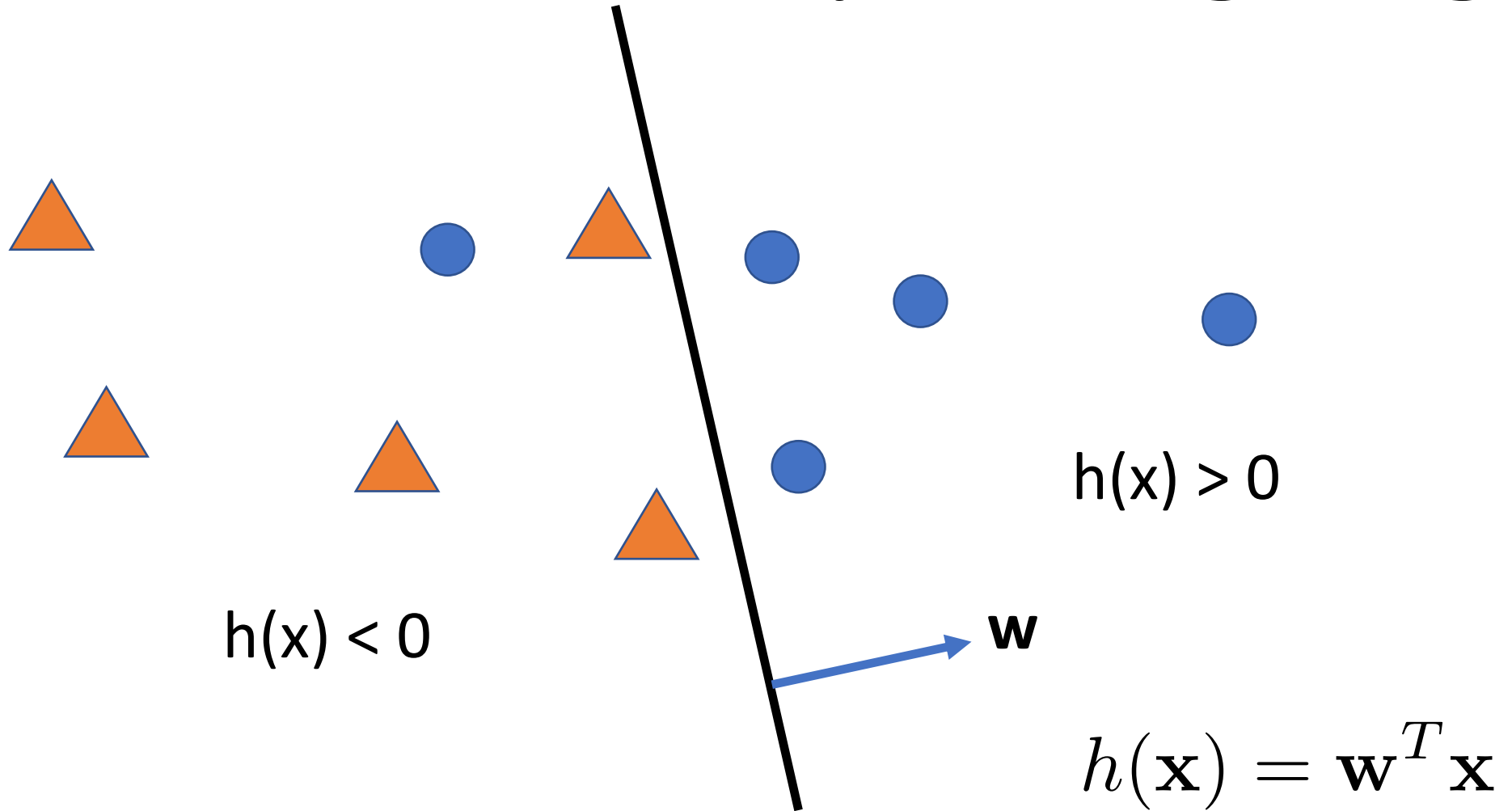
$$p(y = 1; \mathbf{w}) = 1/(1 + \exp(-\mathbf{w}^T \mathbf{x}))$$

$$\underset{=}{h^{(\mathbf{w})}(\mathbf{x})} = \mathbf{w}^T \mathbf{x} \quad 1/(1 + \exp(-h^{(\mathbf{w})}(\mathbf{x})))$$

max. likelihood est. for \mathbf{w} equivalent to log.reg.

see Sec. 3.6 of MLBook

Decision Boundary of Log.Reg.



Logistic Regression in Python

`sklearn.linear_model.LogisticRegression`

```
class sklearn.linear_model.LogisticRegression(penalty='l2', *, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None)
```

[\[source\]](#)

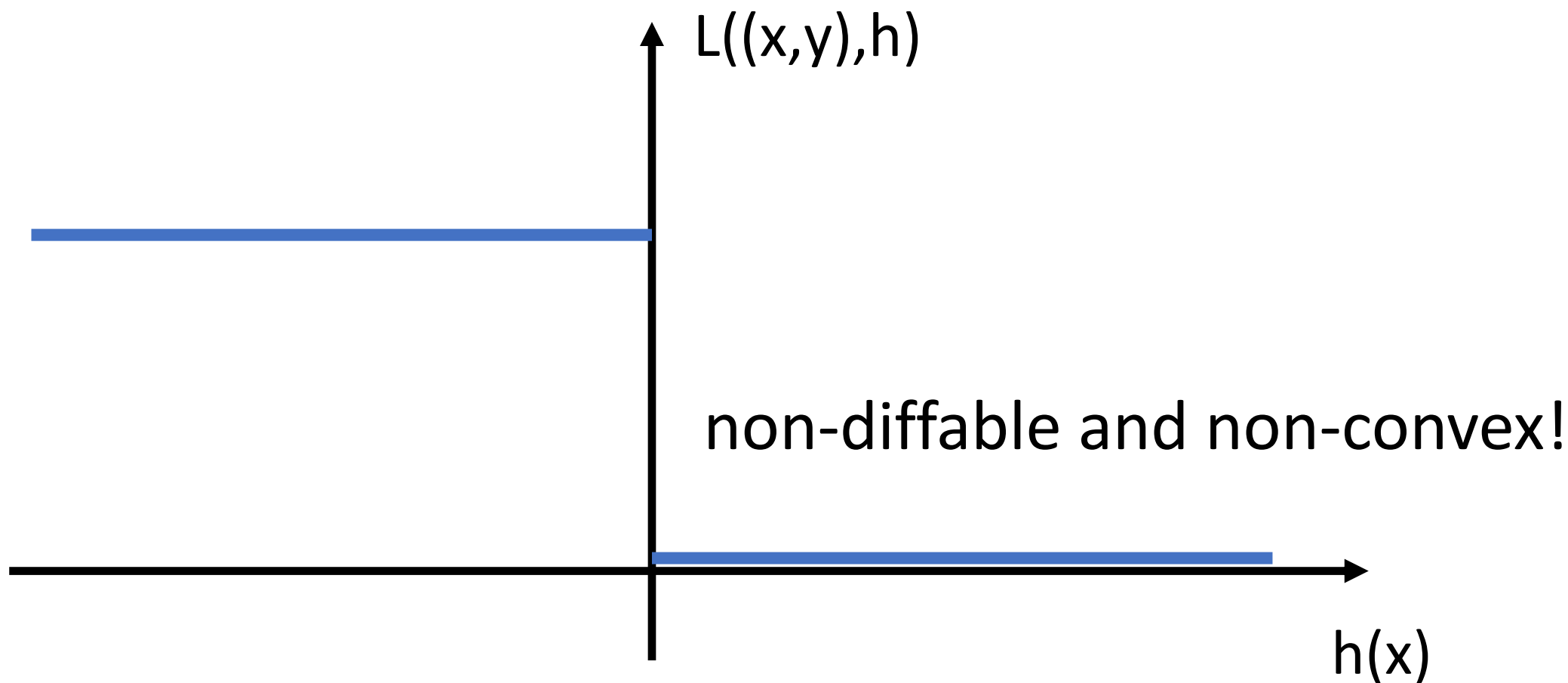
Logistic Regression (aka logit, MaxEnt) classifier.

Naïve Bayes' Classifier (NBClass)

NBClass. – Design Choices

- **data** points with numeric features (same as log.reg.)
- binary label values, e.g., $y=-1$ vs. $y=1$ (same as log.reg)
- **model** = space of linear maps (same as log.reg!)
- 0/1 **loss** (different from log.reg!)

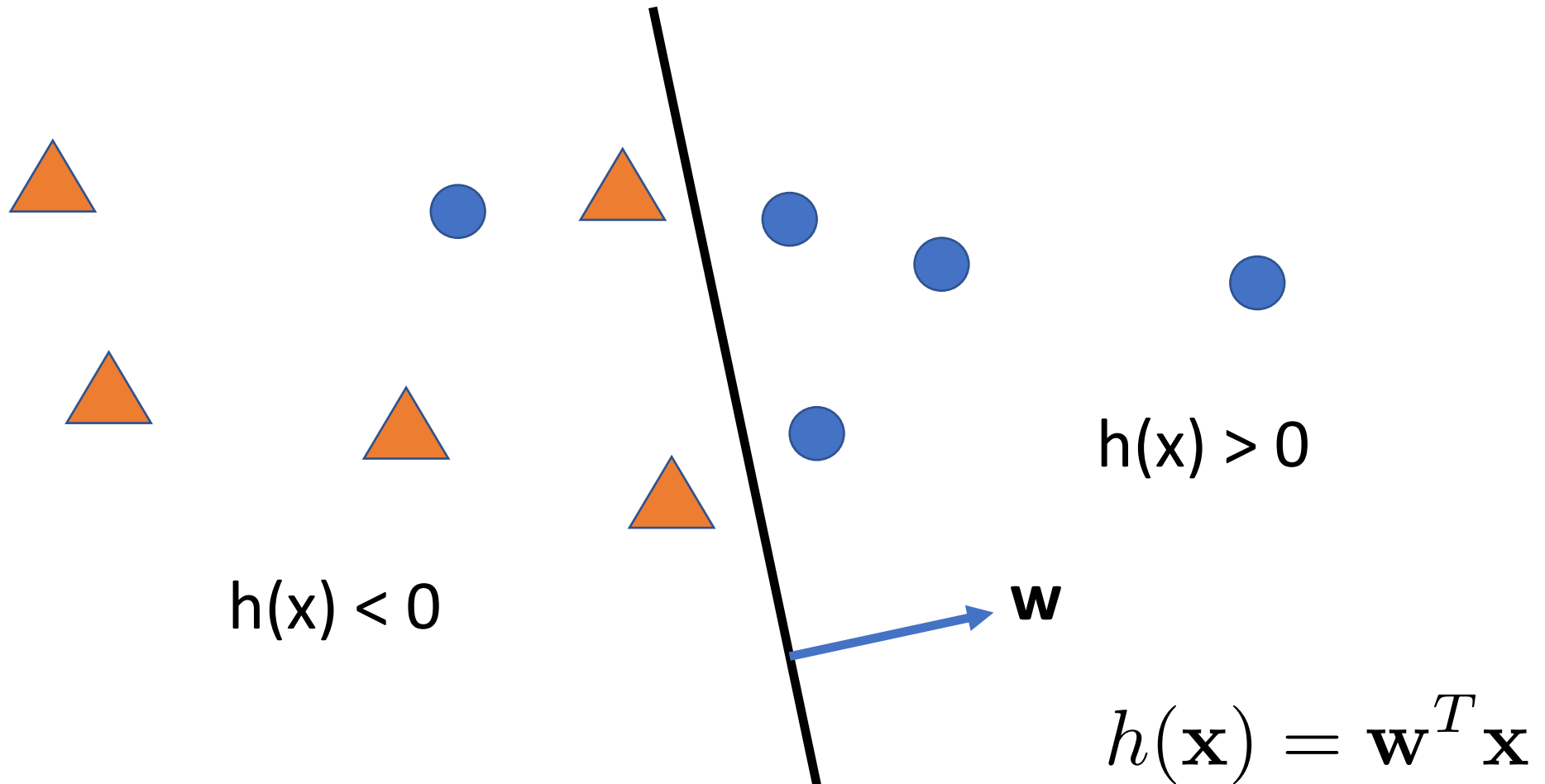
0/1 Loss



NBClass.

- ERM with 0/1 loss difficult for gradient methods
- ERM aims at minimizing error prob. $P(\hat{y} \neq y)$
- Bayes' rule tells that this is equiv. to $\max_y P(\hat{y}|x)$
- assume x is Gaussian with mean depending on y
- $P(\hat{y}|x)$ maximized by thresholding linear map !

Naïve Bayes' Classifier



Logistic Loss vs. 0/1 Loss

- logistic loss nice for optimization/solving ERM
- log. loss is not very interpretable
- what does $\text{log.loss} = 0.3$ mean ?
- average 0/1 loss (error rate) is **more tangible**
- $\text{accuracy} = 1 - \text{average 0/1 loss}$

NBClassifier in Python

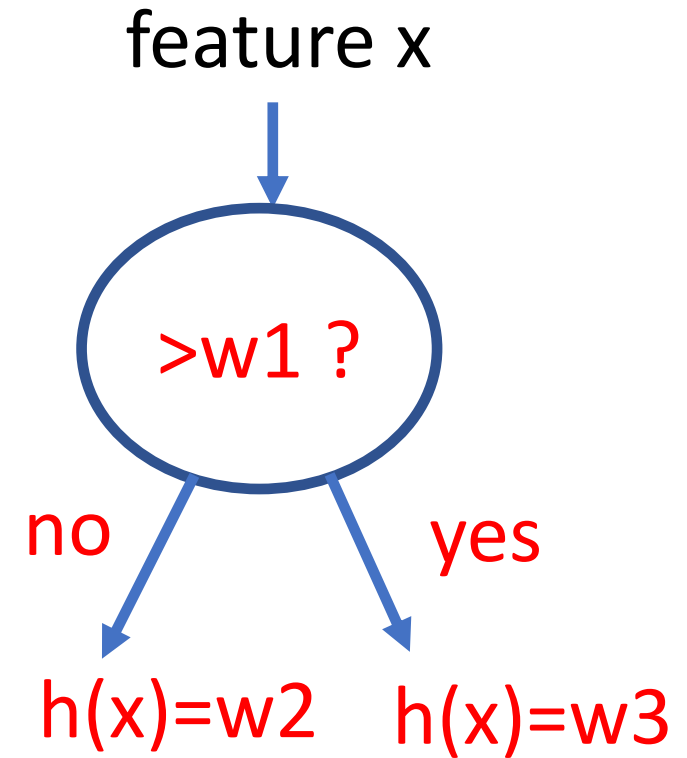
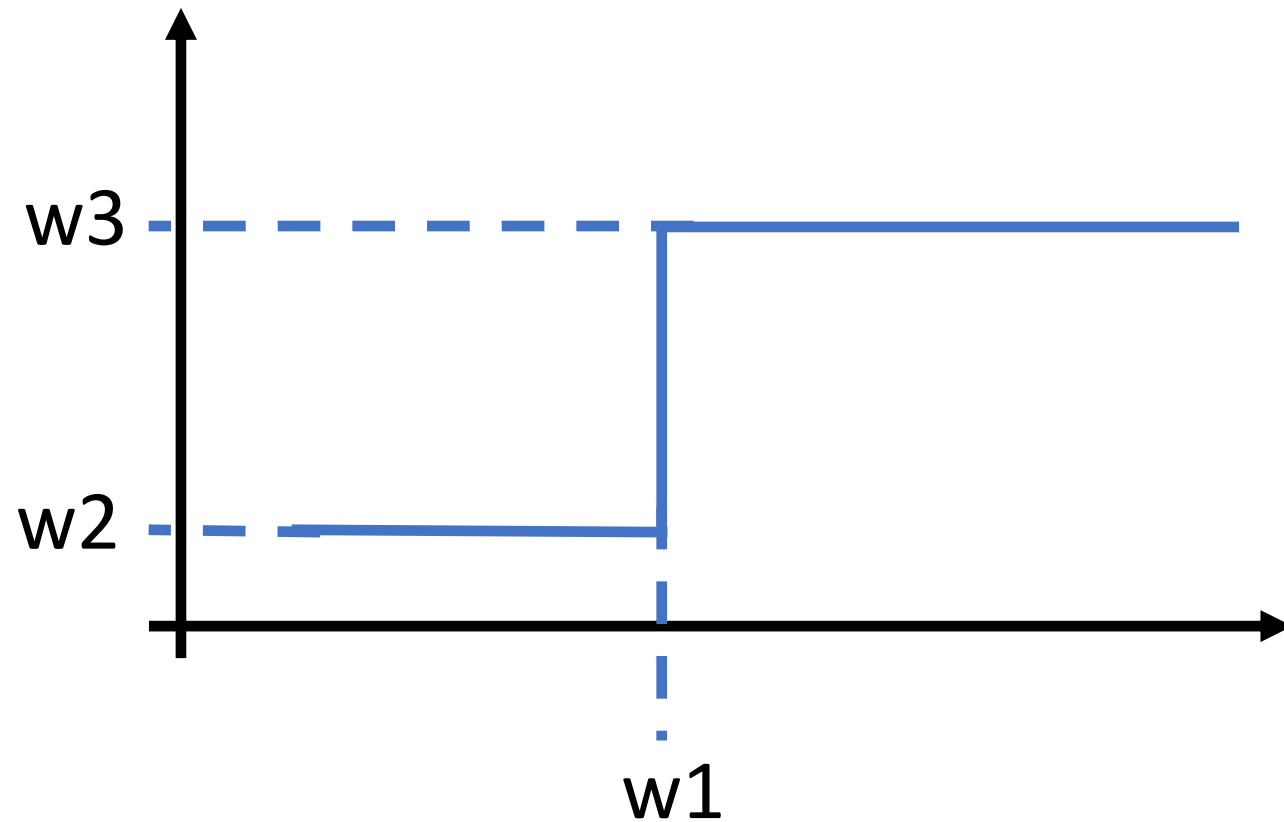
```
>>> from sklearn.datasets import load_iris
>>> from sklearn.model_selection import train_test_split
>>> from sklearn.naive_bayes import GaussianNB
>>> X, y = load_iris(return_X_y=True)
>>> X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state=0)
>>> gnb = GaussianNB()
>>> y_pred = gnb.fit(X_train, y_train).predict(X_test)
>>> print("Number of mislabeled points out of a total %d points : %d"
...       % (X_test.shape[0], (y_test != y_pred).sum()))
Number of mislabeled points out of a total 75 points : 4
```


Decision Tree (DT) Classifier

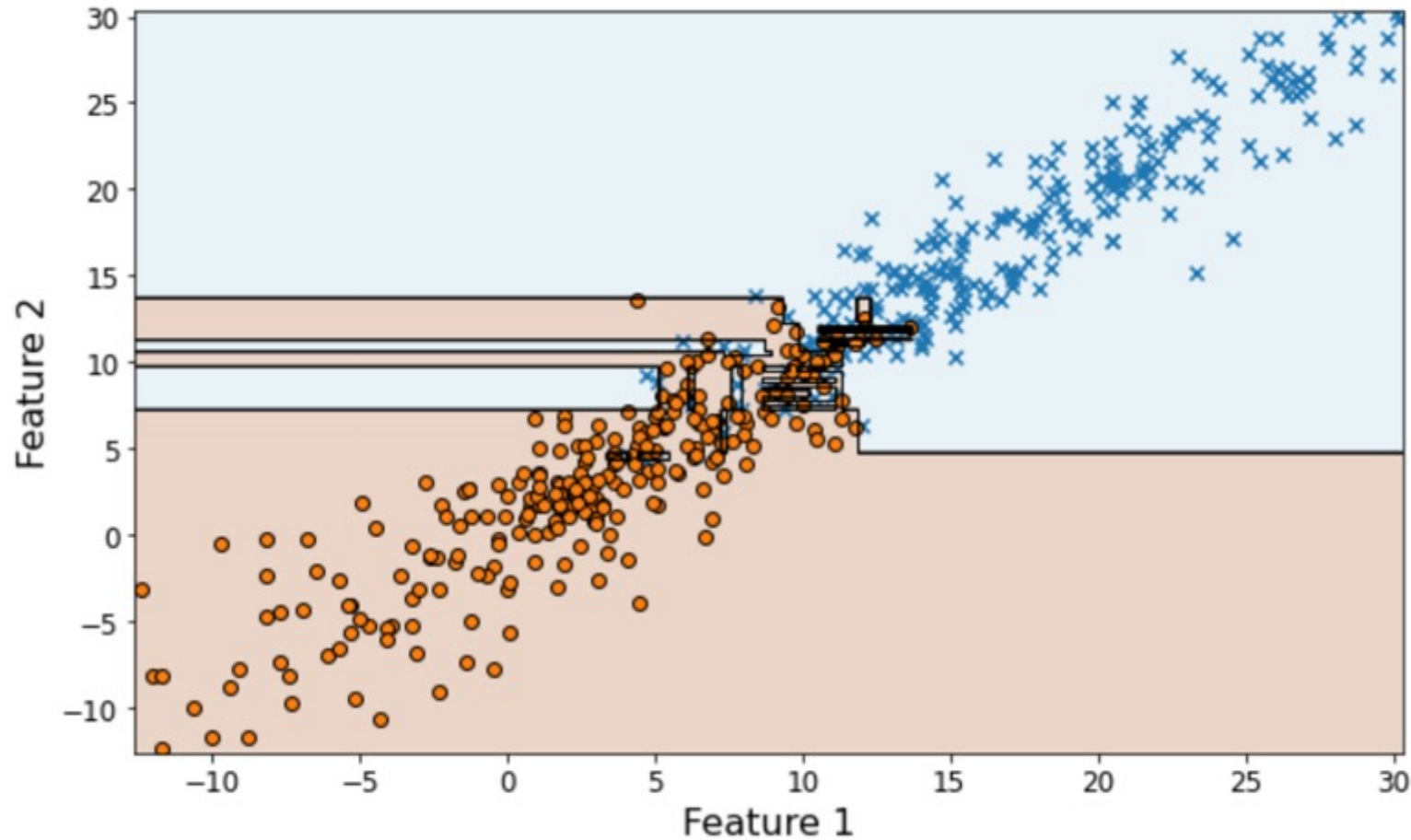
Design Choices

- **data** points with numeric and categ. features
- label values arbitrary, we use $y=-1$ vs. $y=1$
- **model** = **piece-wise constant** maps represented by flow chart (“decision tree”)
- different options for **loss** function

Parametrized DT

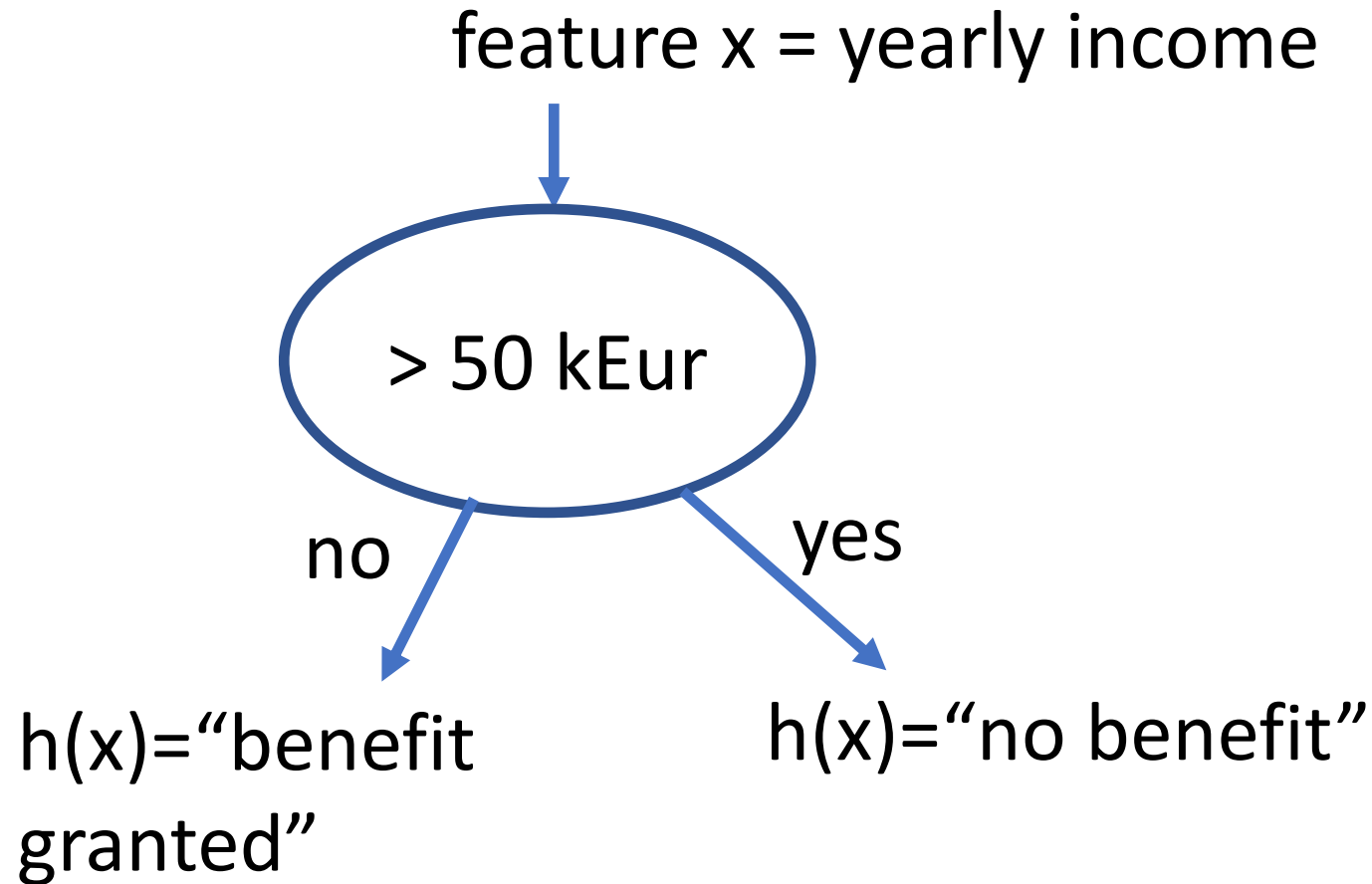


DT - Decision Boundary



piece-wise constant !

DT - Interpretability



DT Pro/Con

- allows for non-linear decision boundary
- computationally expensive
- shallow DT considered interpretable

DT in Python

`sklearn.tree`.DecisionTreeClassifier

```
class sklearn.tree.DecisionTreeClassifier(*, criterion='gini', splitter='best', max_depth=None, min_samples_split=2,  
min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None,  
min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0)
```

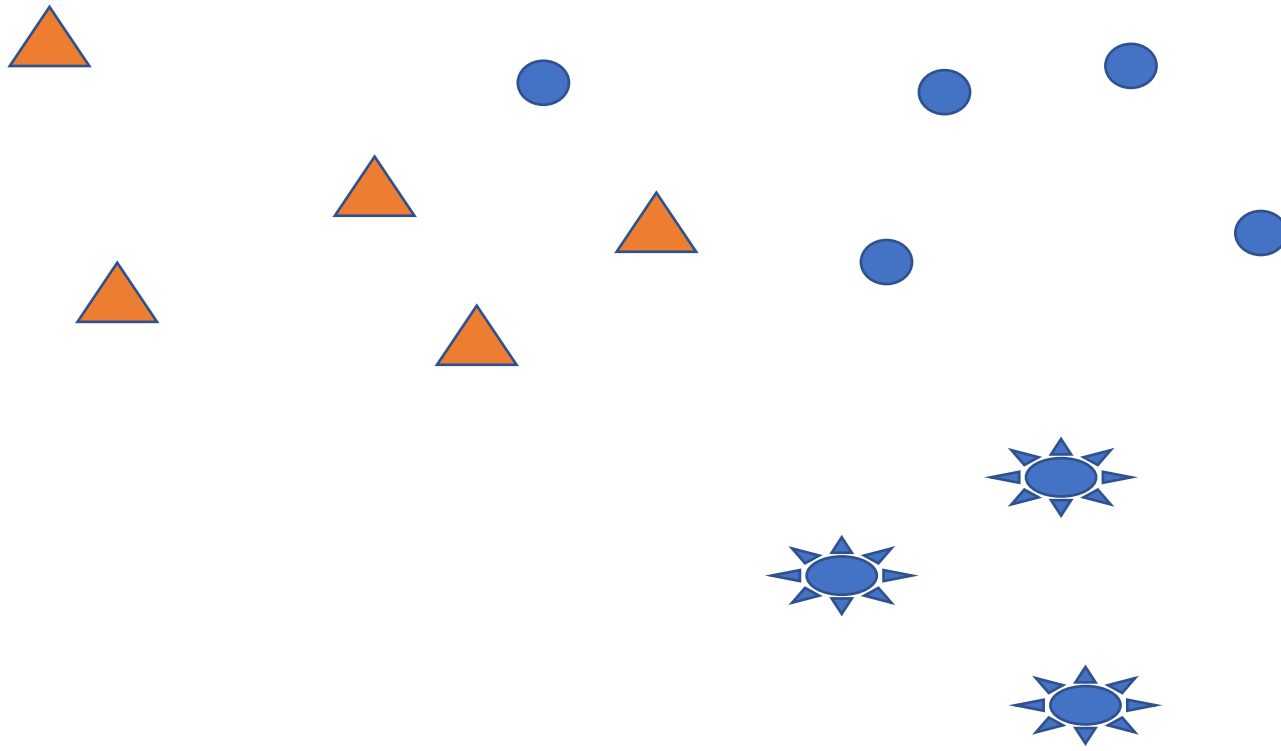
[\[source\]](#)

Multi-Class Classification

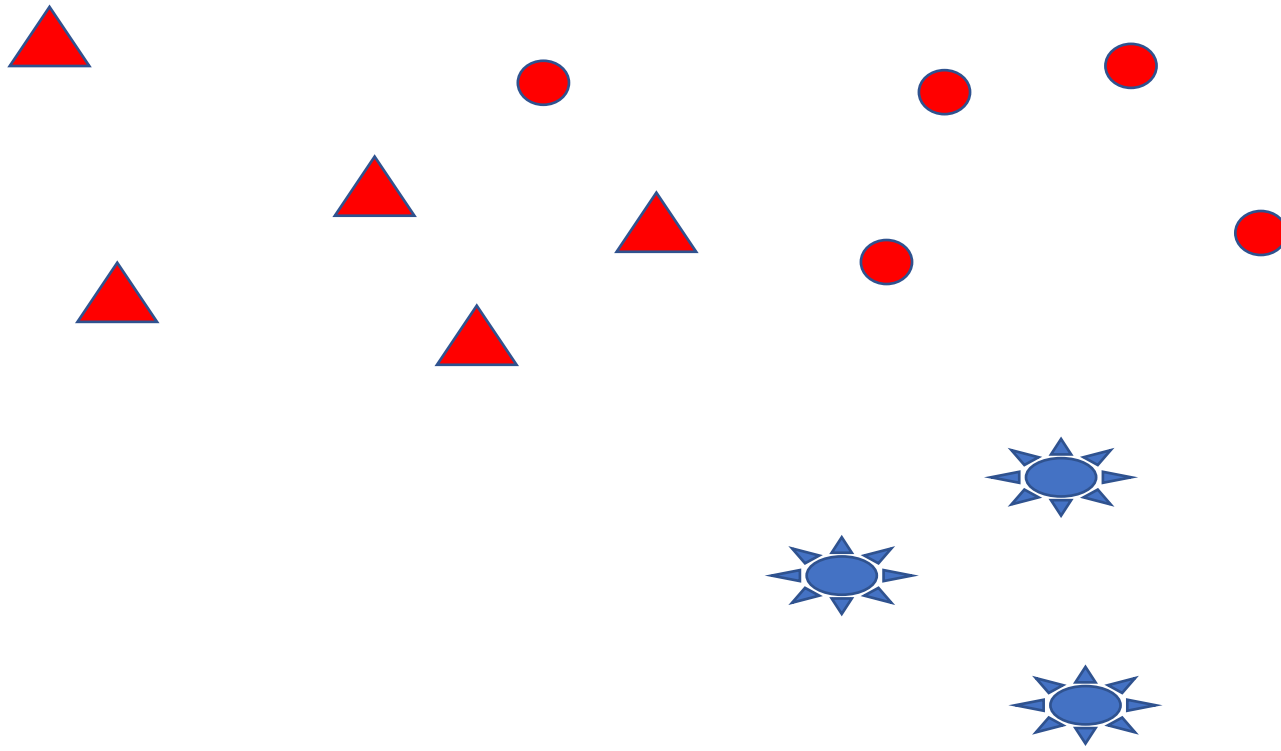
One-vs-Rest Trick

- data points with label values "1", "2", "3"
- break into 3 binary class. problems
 - Problem 1: label values "1", "either 2 or 3"
 - Problem 2: label values "2", "either 1 or 3"
 - Problem 3: label values "3", "either 2 or 3"

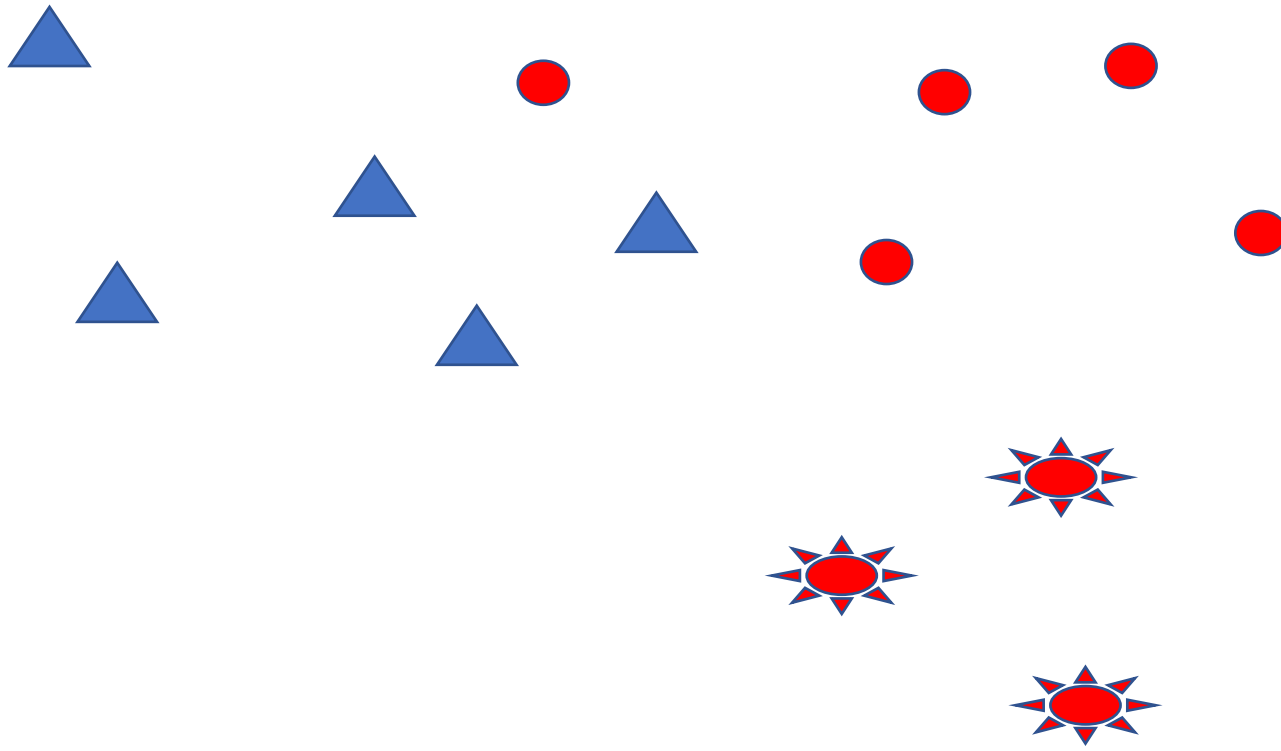
One-vs-Rest Trick



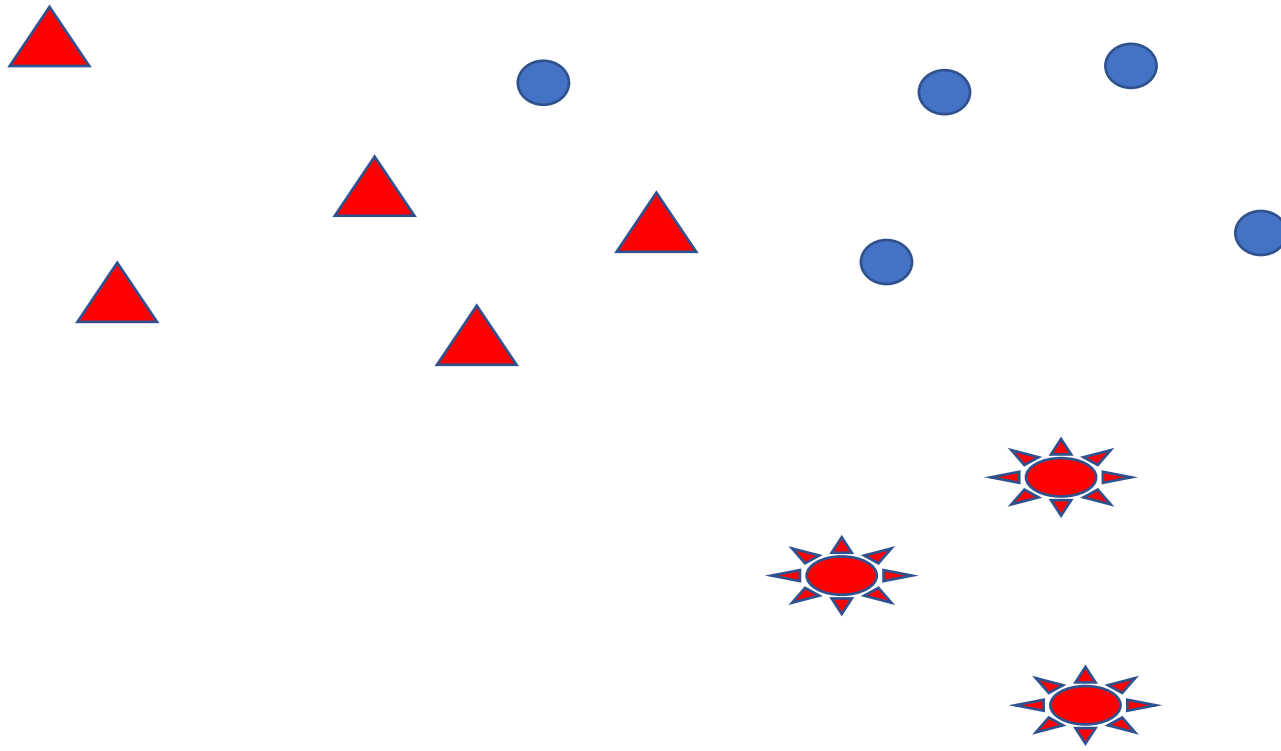
Sub-Problem 1 (red/blue)



Sub-Problem 2 (red/blue)



Sub-Problem 3 (red/blue)



One-vs-Rest Summary

- one sub-problem for each label value c
- sub-problem c : “label = c or not”
- learn hypothesis h_c for each sub-problem
- predict c with highest confidence/prob. $|h_c|$

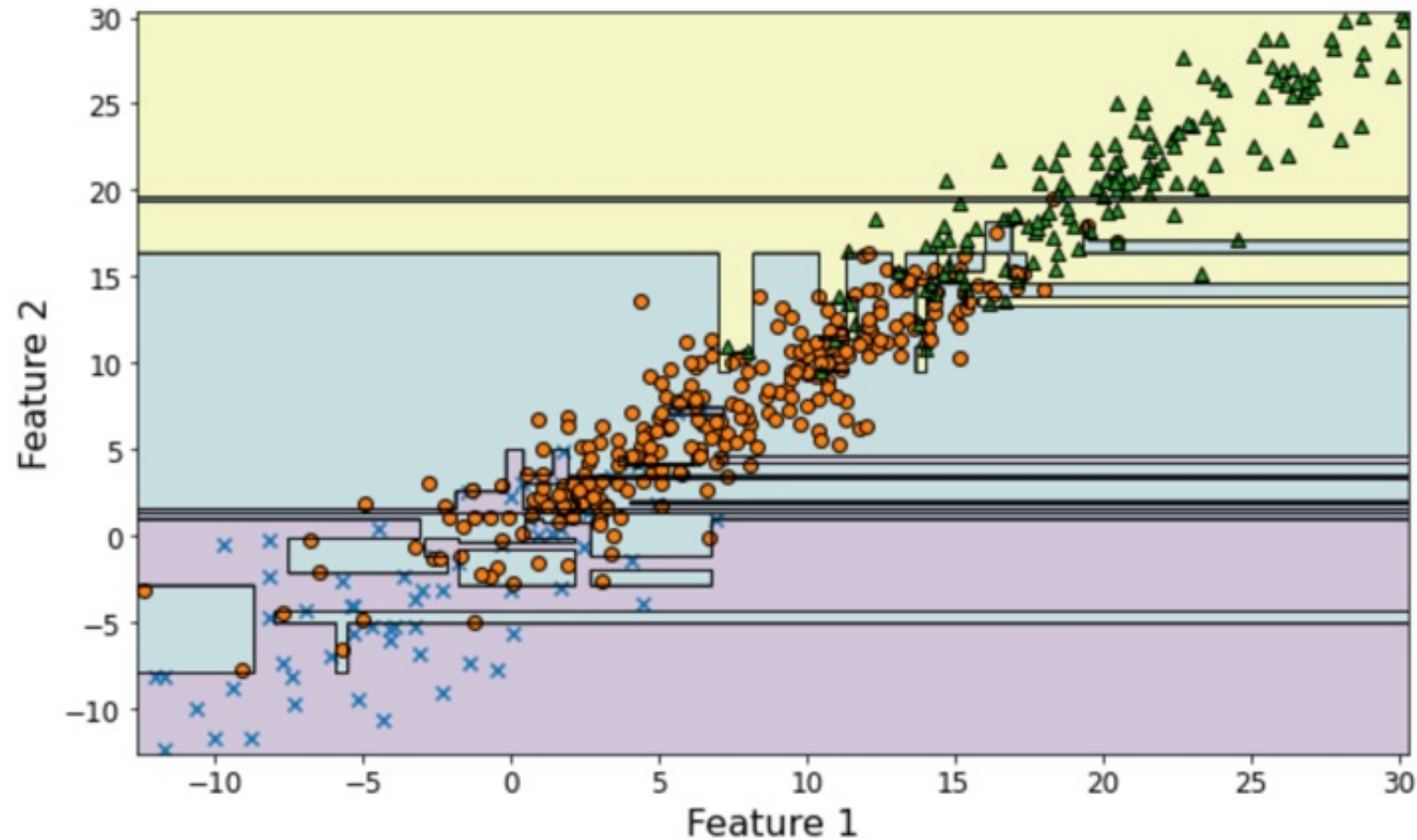
Multi-Class LogReg

- **specific loss functions** for multi-class data
- 0/1 loss also works for > 2 label values (classes)
- but how to encode confidence in predictions?
- **soft-max** (extending log-loss to more classes):

$$P(y = j \mid \mathbf{x}) = \frac{e^{\mathbf{x}^\top \mathbf{w}_j}}{\sum_{k=1}^K e^{\mathbf{x}^\top \mathbf{w}_k}}$$

source: https://en.wikipedia.org/wiki/Softmax_function

DT Multi-Class



Multi-Label Classification



label y1 = contains tree ? yes/no
label y2 = contains house ? yes/no
label y3 = taken during leisure? yes/no
label y4 = taken during office? yes/no
label y5 = location in Finland? yes/no
label y6 = location in Sweden? yes/no

Bonsai - Diverse and Shallow Trees for Extreme Multi-label Classification

Sujay Khandagale¹, Han Xiao² and Rohit Babbar²

¹Indian Institute of Technology Mandi, India

²Aalto University, Helsinki, Finland

“...benchmark Amazon-3M dataset with 3 million labels,..

Ignorant Approach

- consider each label separately
- solve binary/multi-class problem for each label
- **ignores correlations** among different labels



label y1 = contains tree ? yes/no



label y2 = contains house ? yes/no



label y6 = location in Sweden? yes/no

Multi-Class Approach

- each combination of label values defines category
- obtain a multi-class problem with many classes
- **huge number** of resulting **categories**

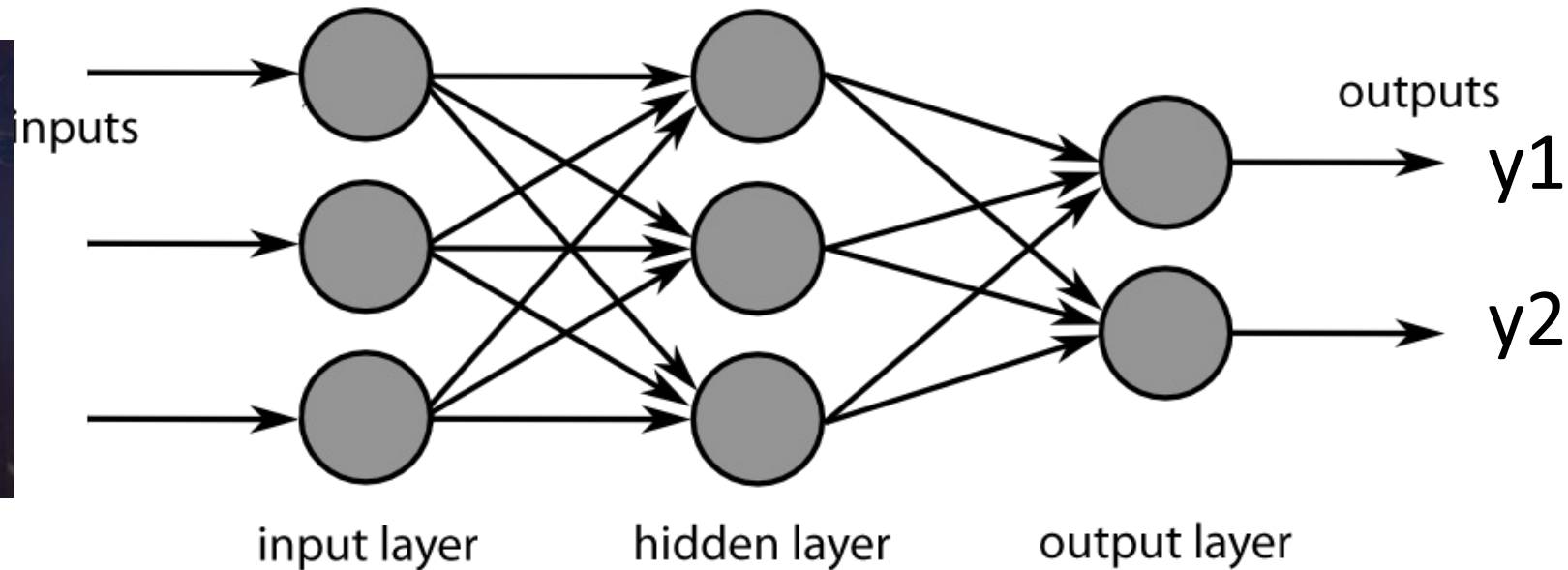
	y1	y2
cat. 1	0	0
cat. 2	0	1
cat. 3	1	0
cat. 4	1	1

Multi-Task Learning

- each individual label results in separate learning task
- use similarities between learning tasks
- similarities inform regularization techniques
- more in Lecture “Regularization”

Y. Huang, W. Wang, L. Wang and T. Tan, "Multi-task deep neural network for multi-label learning," *2013 IEEE International Conference on Image Processing*, 2013, pp. 2897-2900, doi: 10.1109/ICIP.2013.6738596.

Multi-Head Deep Net



share lower-level feature maps/layers across hypotheses

source: https://commons.wikimedia.org/wiki/File:MultiLayerNeuralNetwork_english.png

Summary

- classif. methods are instances of ERM
- binary, multi-class and multi-label classif.
- linear vs. non-linear classifiers (DT)
- different loss functions for classif.
- trade-offs between comput./statist./interpretability

What's Next ?

- lecture “Model Validation and Selection”, tmrw morning
- lecture “SVM” [MLBook, Sec. 3.7.], tmrw afternoon