CS-C3240 - Machine Learning

Model Regularization

Alexander Jung

What I want to teach you today:

- recap of model training, validation and selection
- basic idea of regularization
- basic idea of data augmentation
- equivalence between regularization and data aug.

Empirical Risk Minimization

learn hypothesis out of model that incurs minimum loss when predicting labels of datapoints based on their features

$$\hat{h} \in \underset{h\in\mathcal{H}}{\operatorname{argmin}} \widehat{L}(h|\mathcal{D}) \overset{(\boxed{2.16})}{=} \underset{h\in\mathcal{H}}{\operatorname{argmin}} (1/m)\sum_{i=1}^{m} L((\mathbf{x}^{(i)}, y^{(i)}), h).$$

hypothesis

label of i-th datapoint

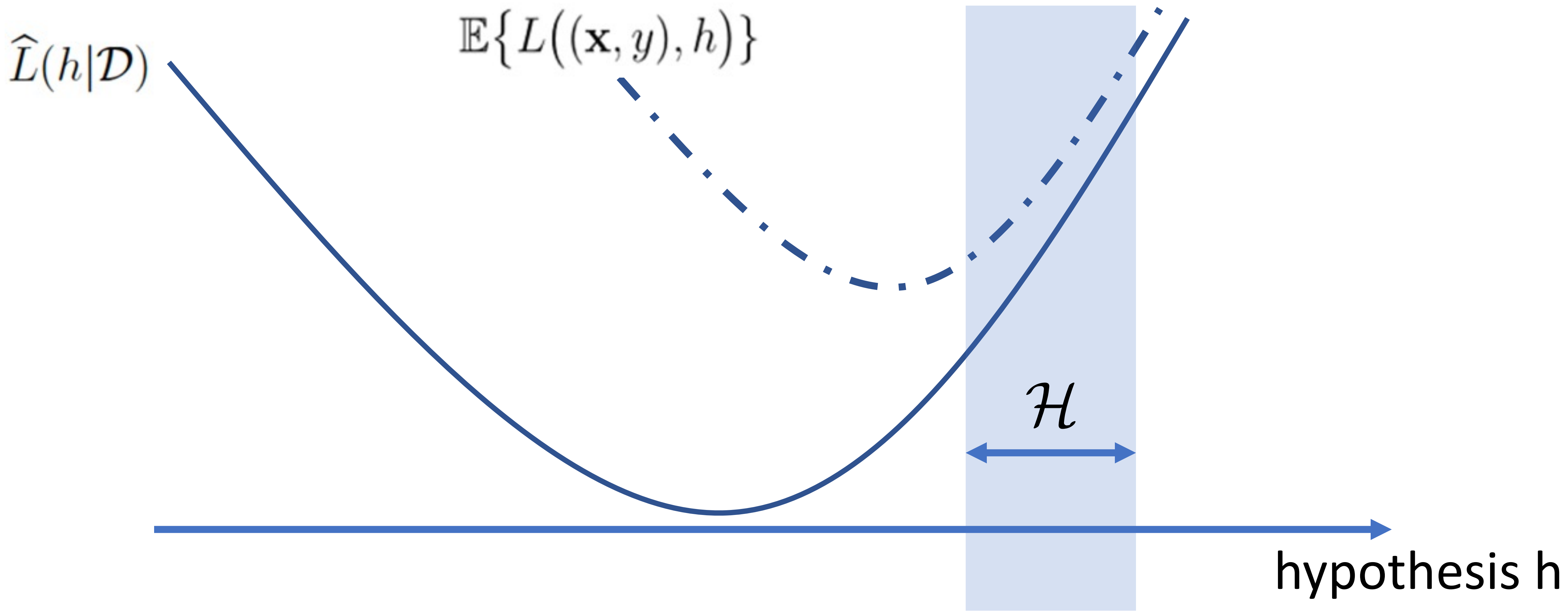
model

loss function

features of i-th datapoint

see Ch. 4.1 of mlbook.cs.aalto.fi

ERM is only Approximation!

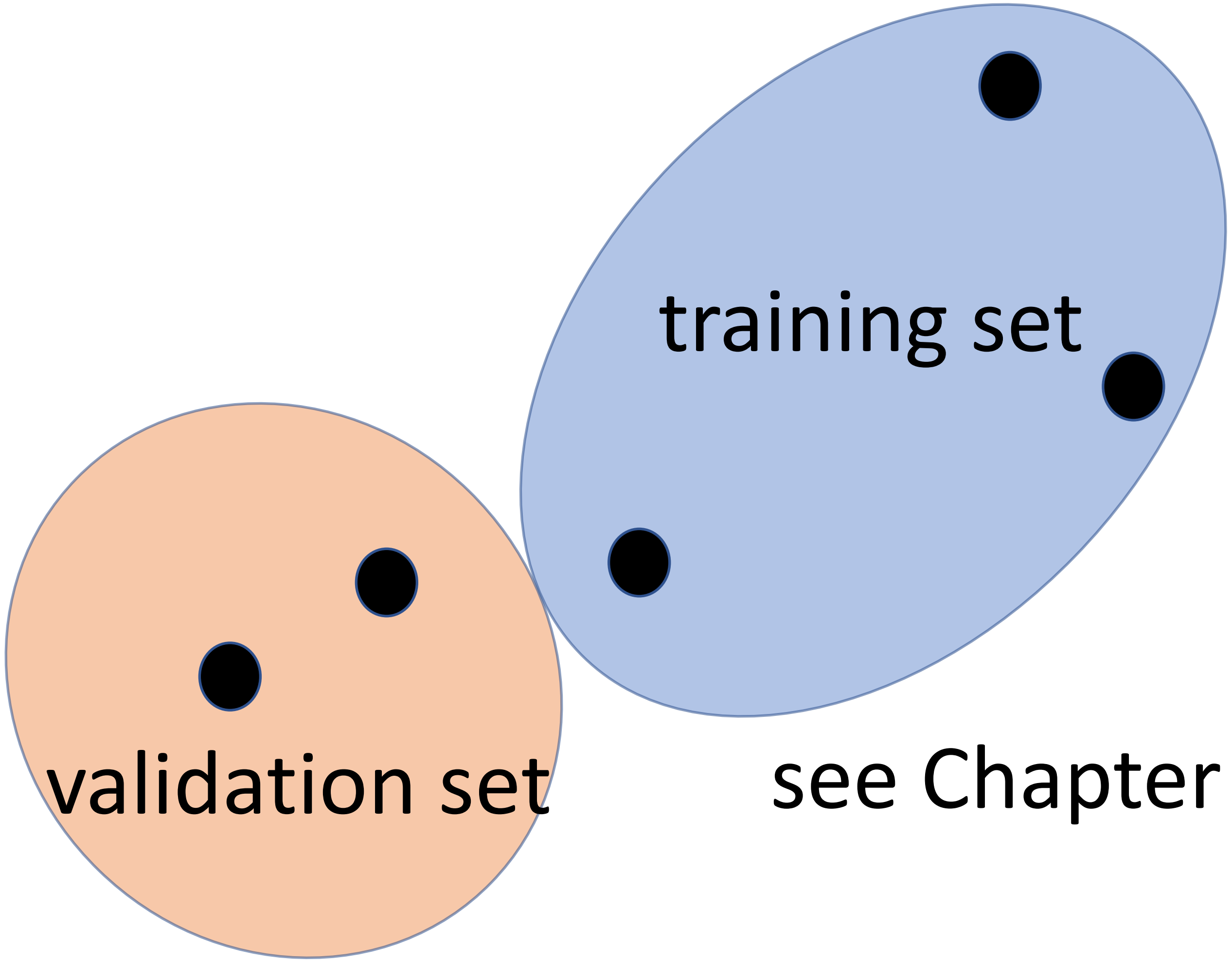


Model Validation and Selection

Learn and Validate!

- divide datapoints into two subsets
- training and validation set
- train.set: used to learn $\hat{h} \in \mathcal{H}$
- val.set: used to probe $\hat{h}$ outside trainset

Split Data into Training and Validation Set !



see Chapter 6.2 of mlbook.cs.aalto.fi

Split into Train and Val Set in Python

labels of datapoints in trainset

labels of datapoints

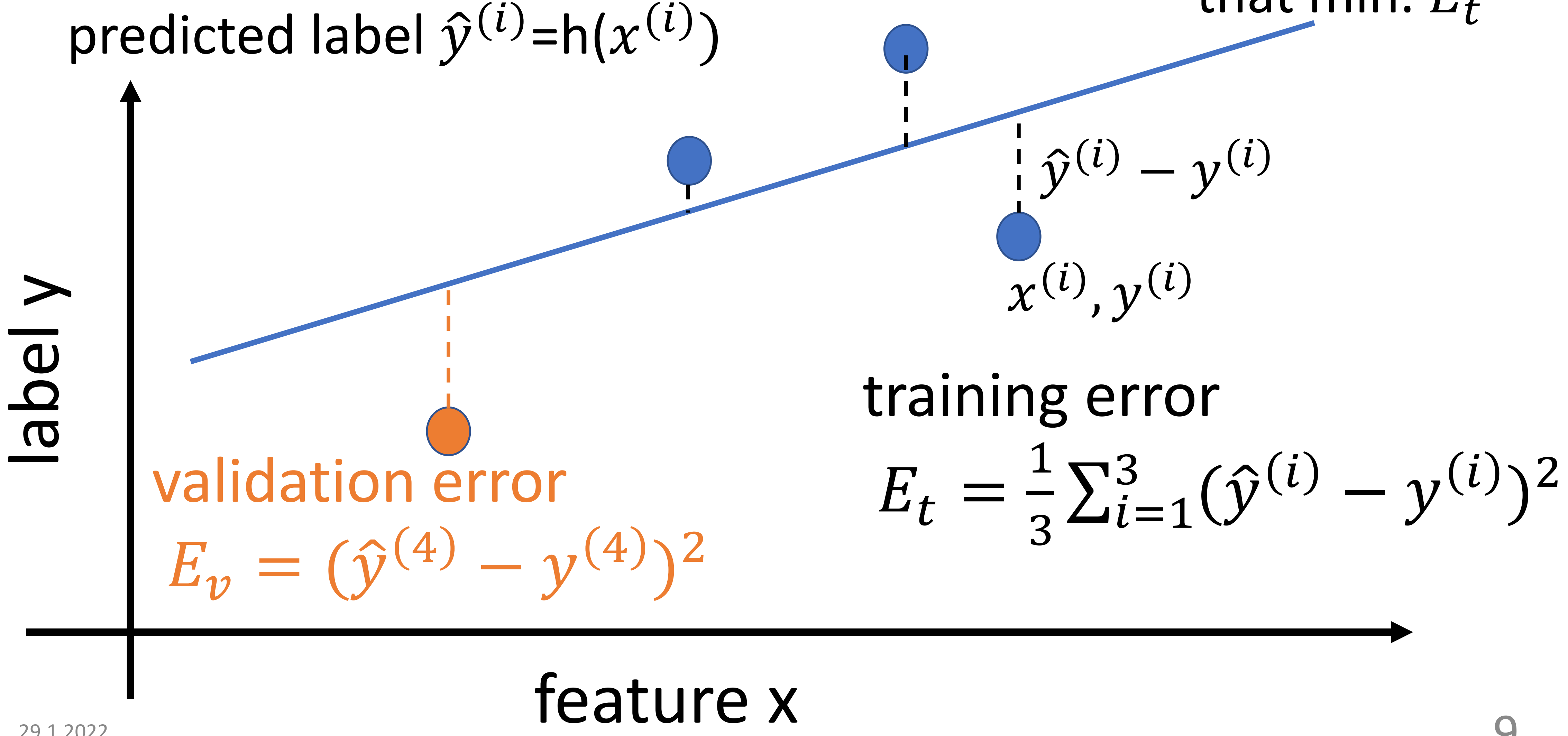
```python
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.4, random_state=42)
```

feature vectors of datapoints in trainset

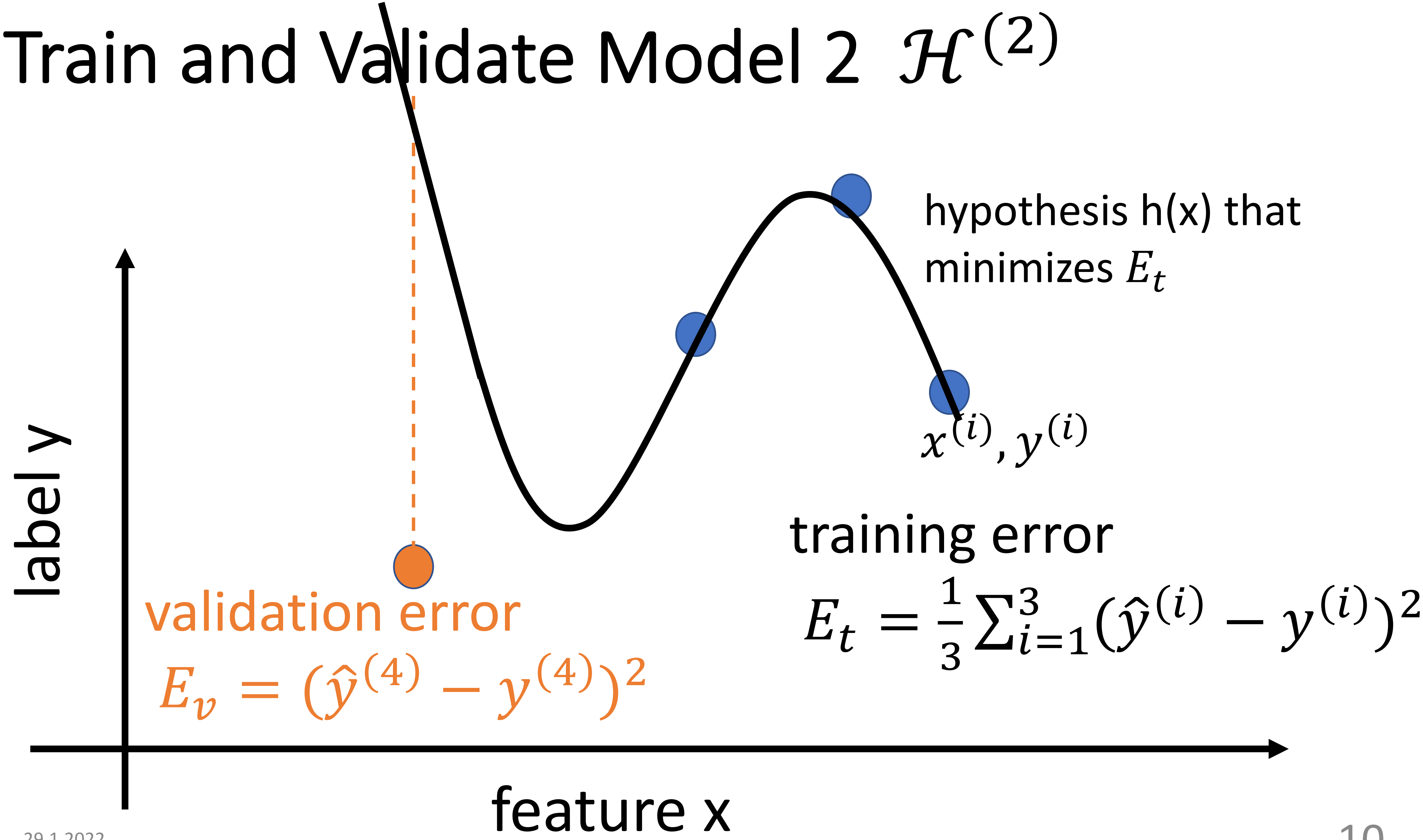
feature vectors of datapoints

Train and Validate Model 1 $\mathcal{H}^{(1)}$

hypothesis h(x) that min. E_t



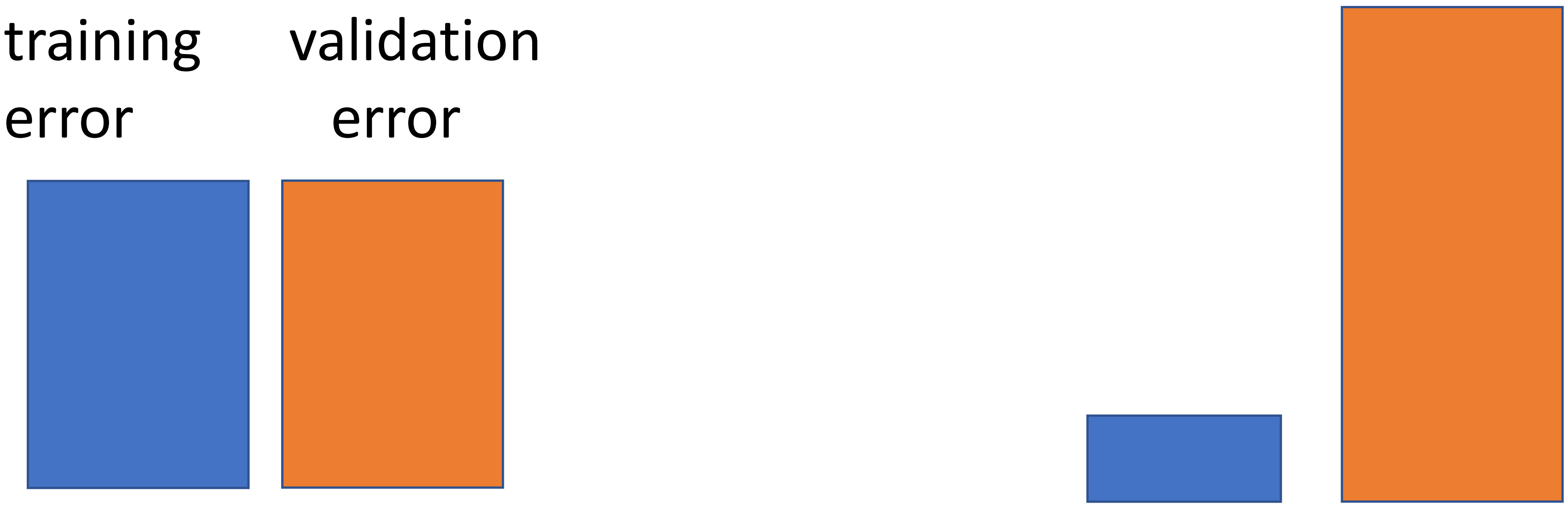
Train and Validate Model 2 $\mathcal{H}^{(2)}$



Basic Idea of Model Selection

- choose model with smallest validation error!

training error   validation error



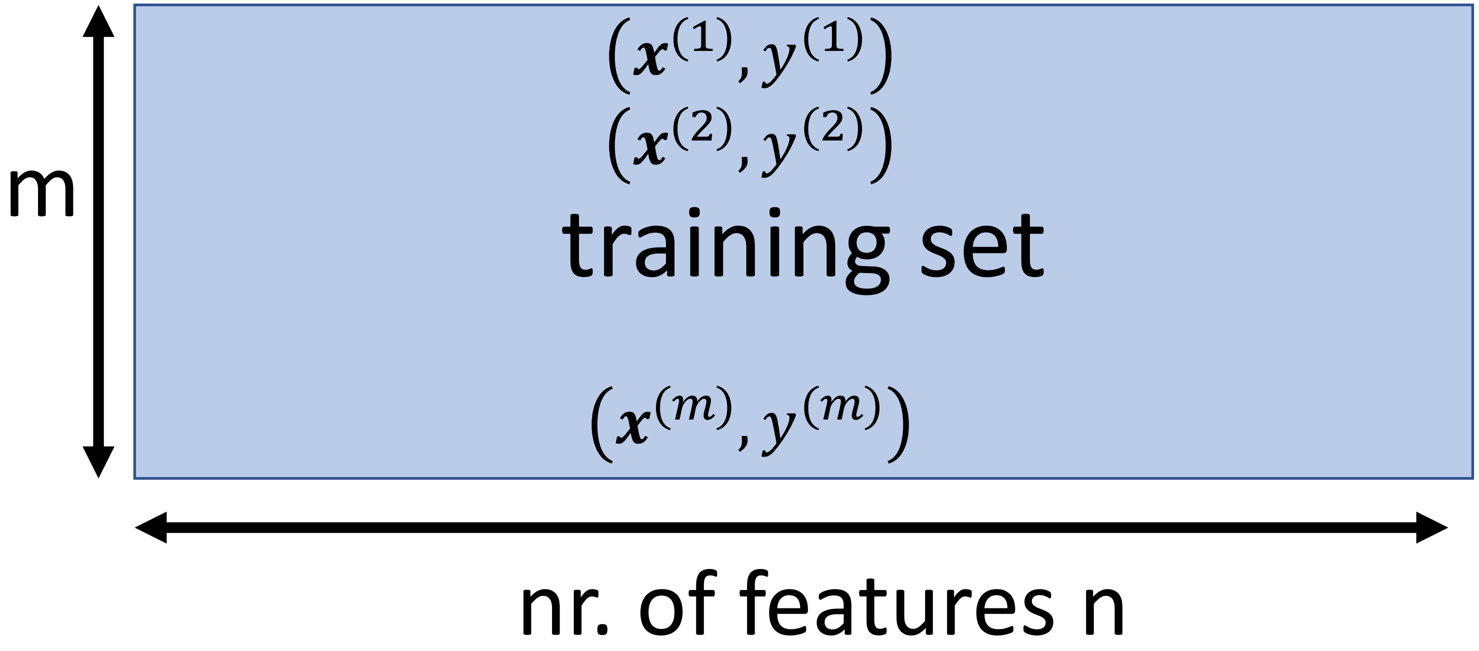
model 1
degree 1 polyn.

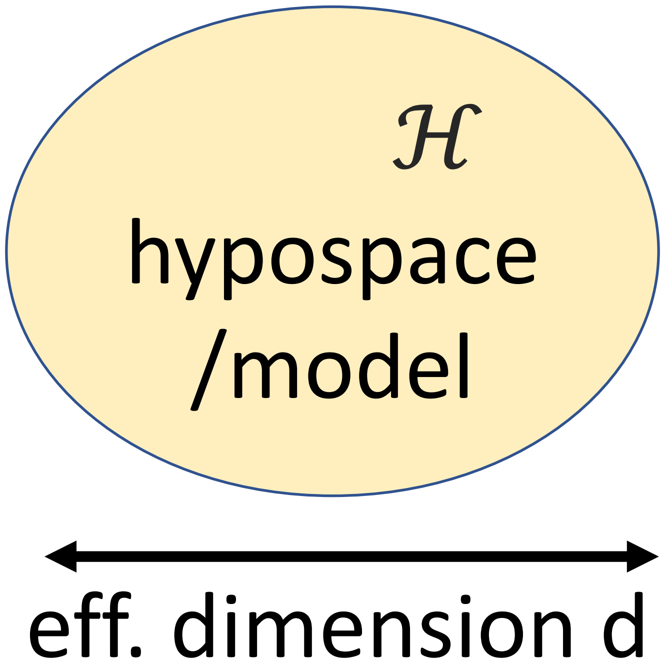
model 2:
degree 3 polyn.

Use Different Loss for Train and Val

- we can use different loss for training and validation
- this enables the comparison of different ML methods
- logistic regression uses log loss to learn hypothesis h1(x)
- SVM uses hinge loss to learn hypothesis h2(x)
- compare h1, h2 by their average 0/1 loss (“accuracy”) on val. set

Data and Model Size





crucial parameter is the ratio d/m

Effective Dim. Linear Maps

- linear map can perfectly fit m data points with n features, as soon as $n \geq m$ [Ch 6.1, mlbook.cs.aalto.fi]

- eff.dim. of linear maps = nr. of features

- $d = n$

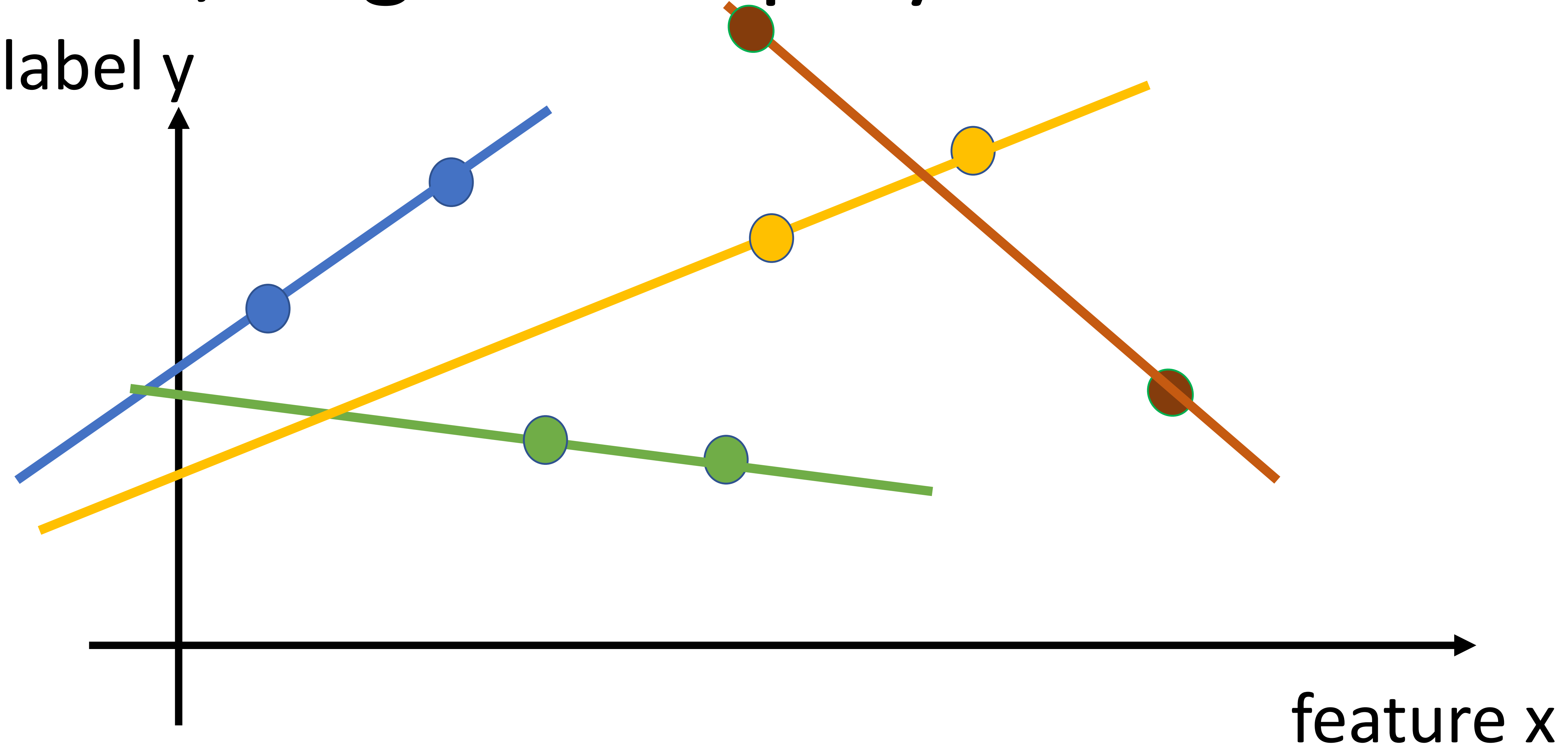
Effective Dim. Polyn. Reg.

perfectly fit (almost) any m data points using polynomials of max degree r as soon as

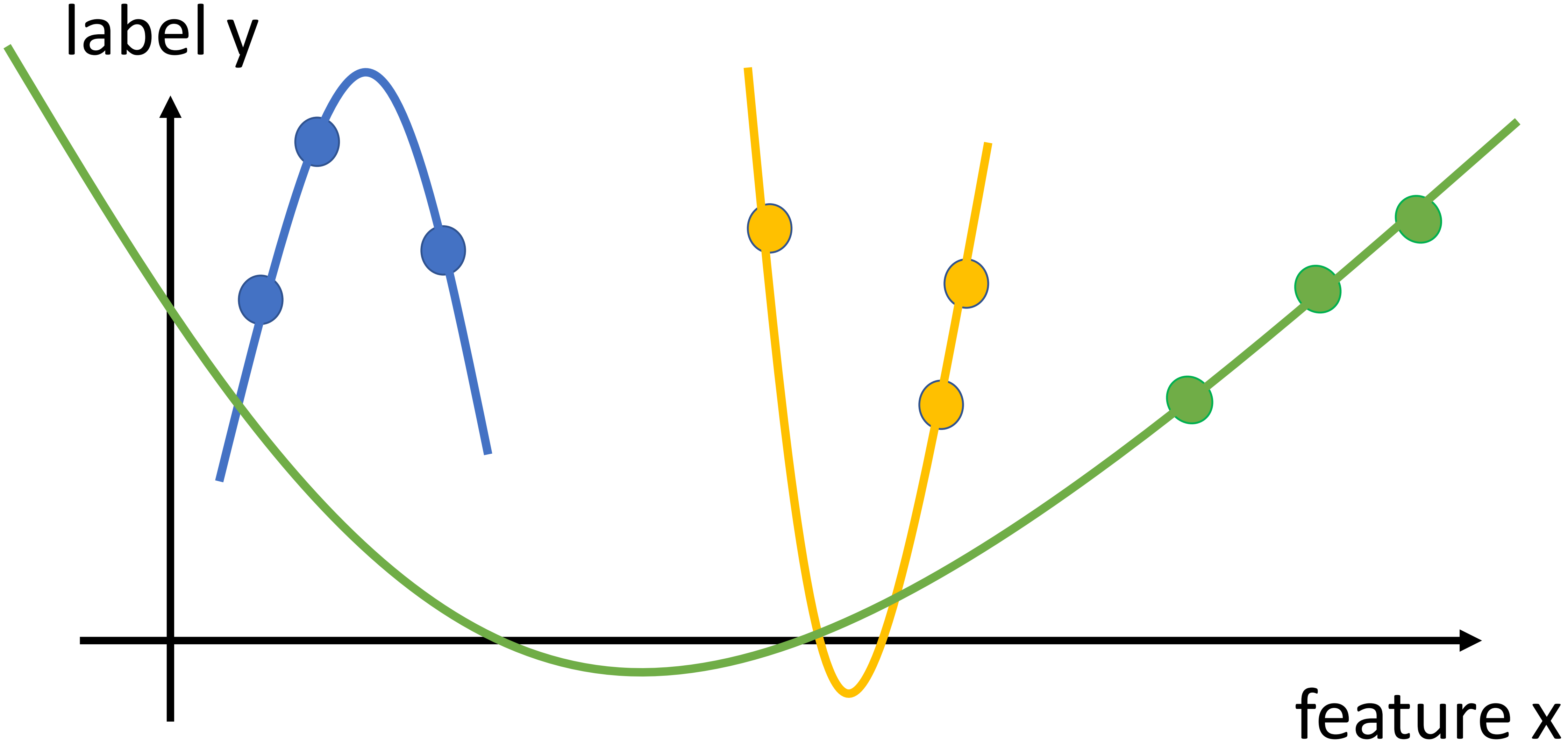
$$r+1 \geq m$$

-> $d = r+1$ (effective dim. of polyn. regression equals the max. polyn. degree plus one!)

m=2, degree r=1 polynomial

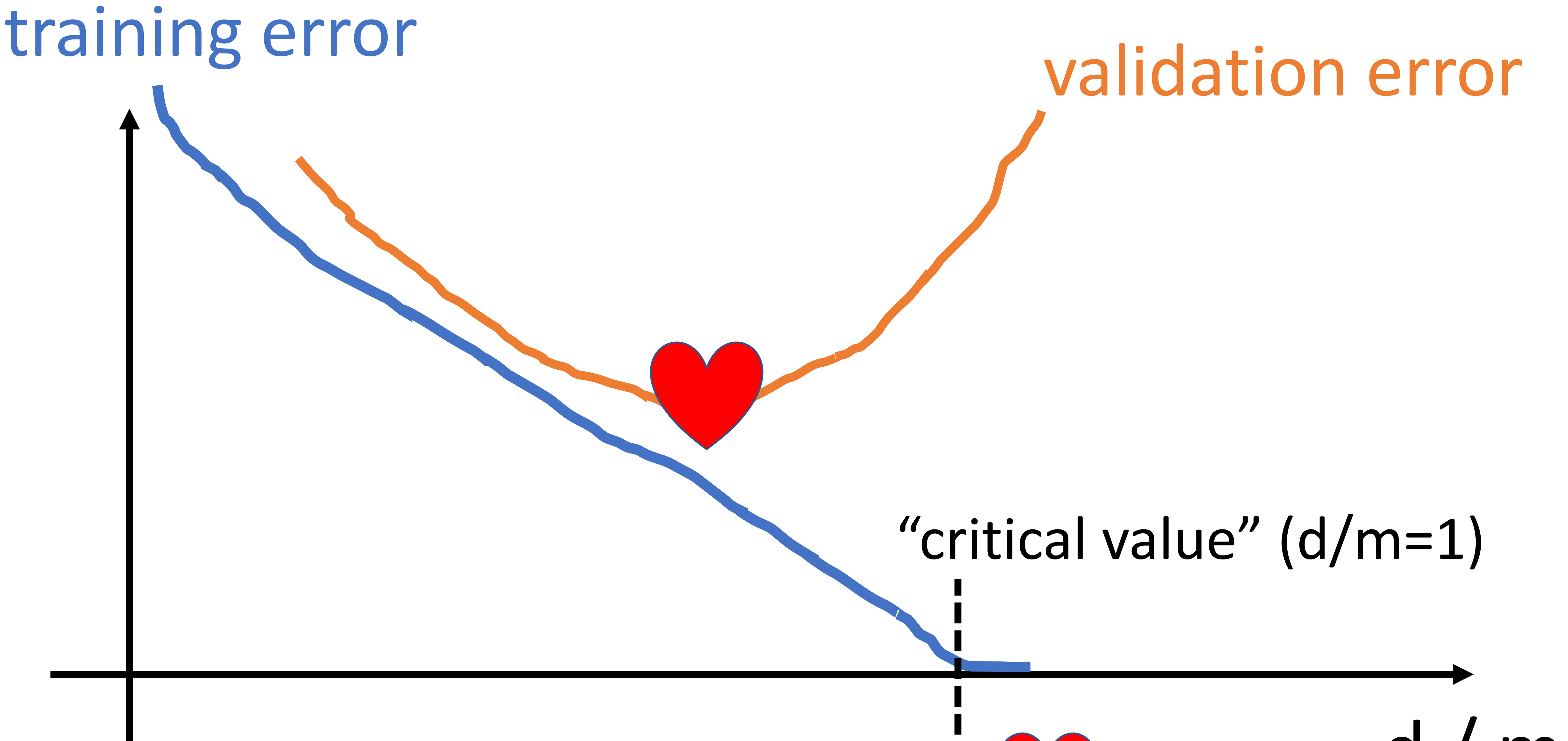


m=3, degree r=2 polynomial



Data Hungry ML Methods

- millions of features for datapoints (e.g. megapixel image)
- eff.dim. d of linear maps is also millions
- eff.dim d of deep nets is millions … billions
- can perfectly fit any set of 100000s (!) of datapoints
- training error will be zero (overfitting!)



adjust model and/or data to reach ♥

how to bring d/m below critical value?

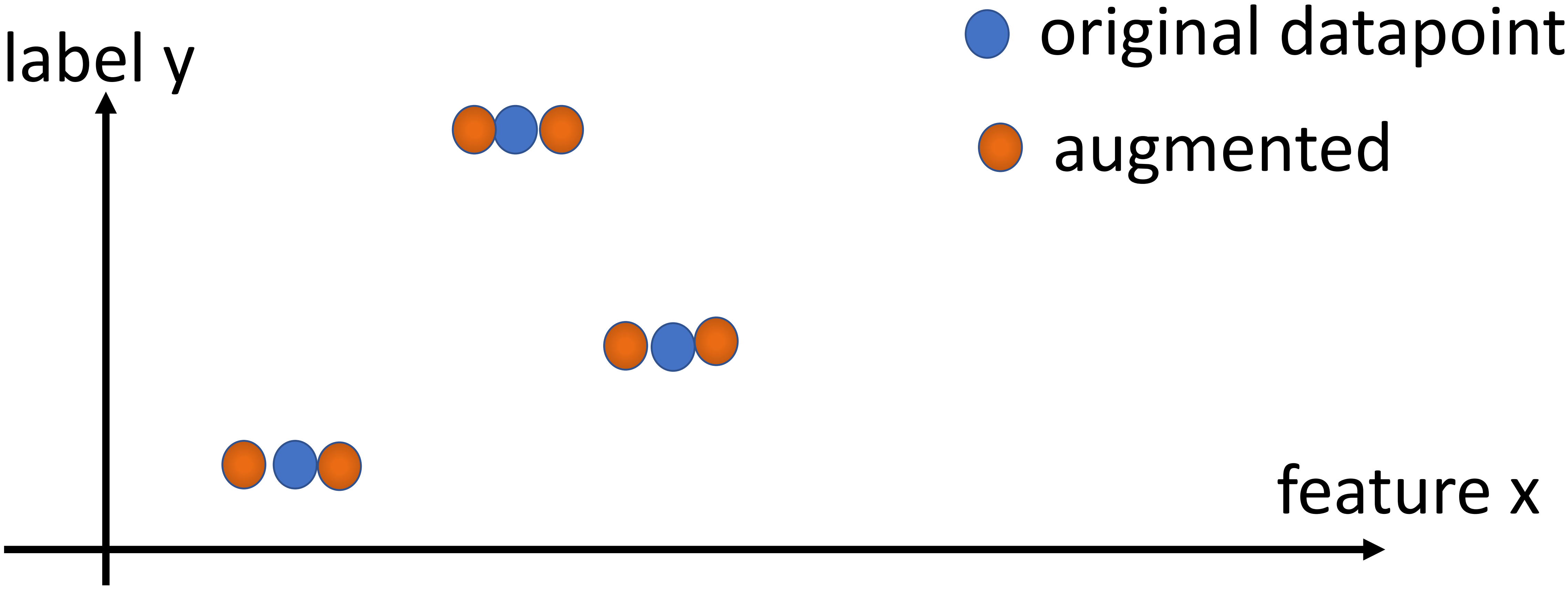
- increase m by using more training data
- decrease d by using smaller hypothesis space

how to bring d/m below critical value?

- increase m by using more training data
- decrease d by using smaller hypothesis space

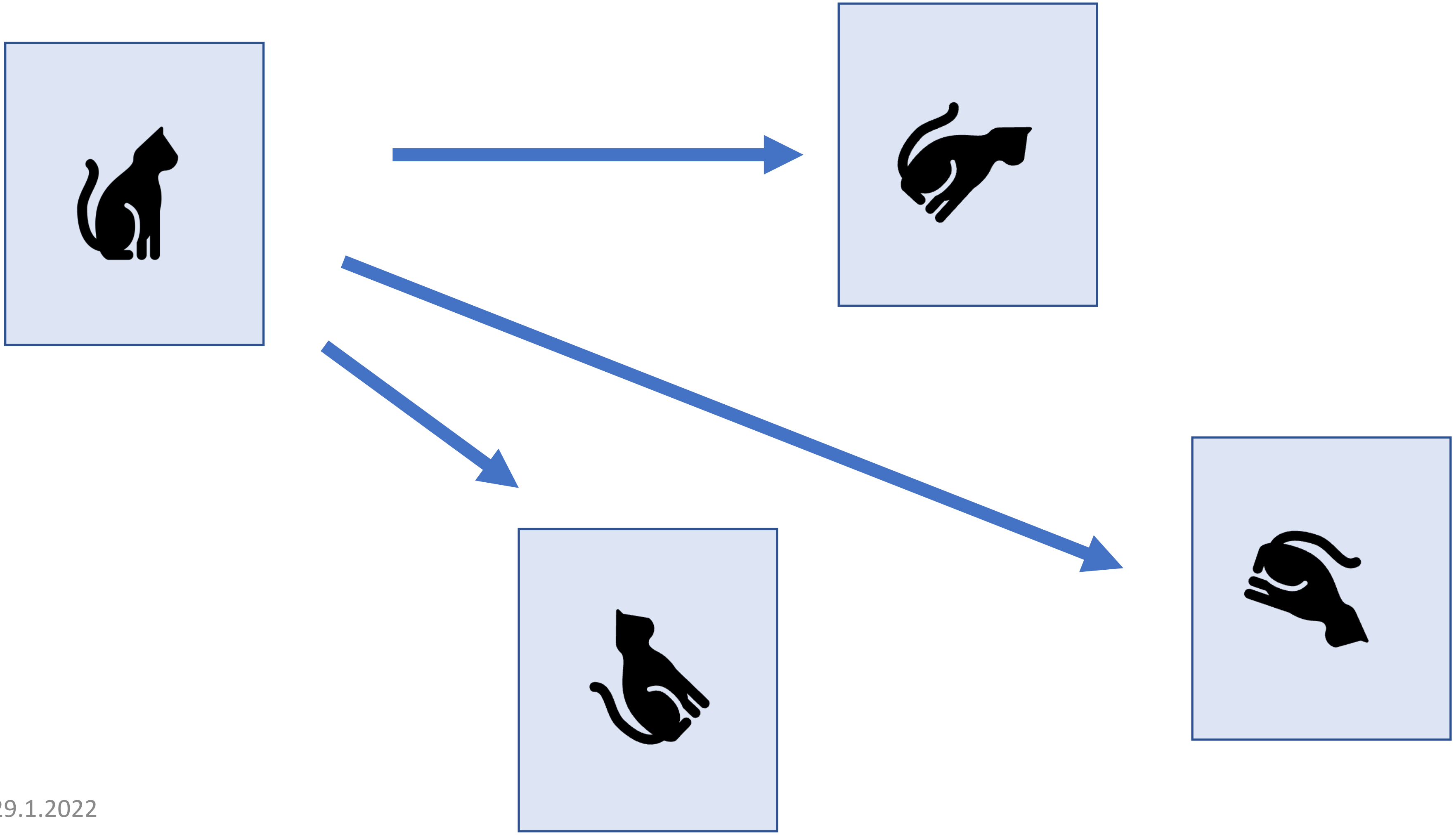
Data Augmentation

add a bit of noise to features

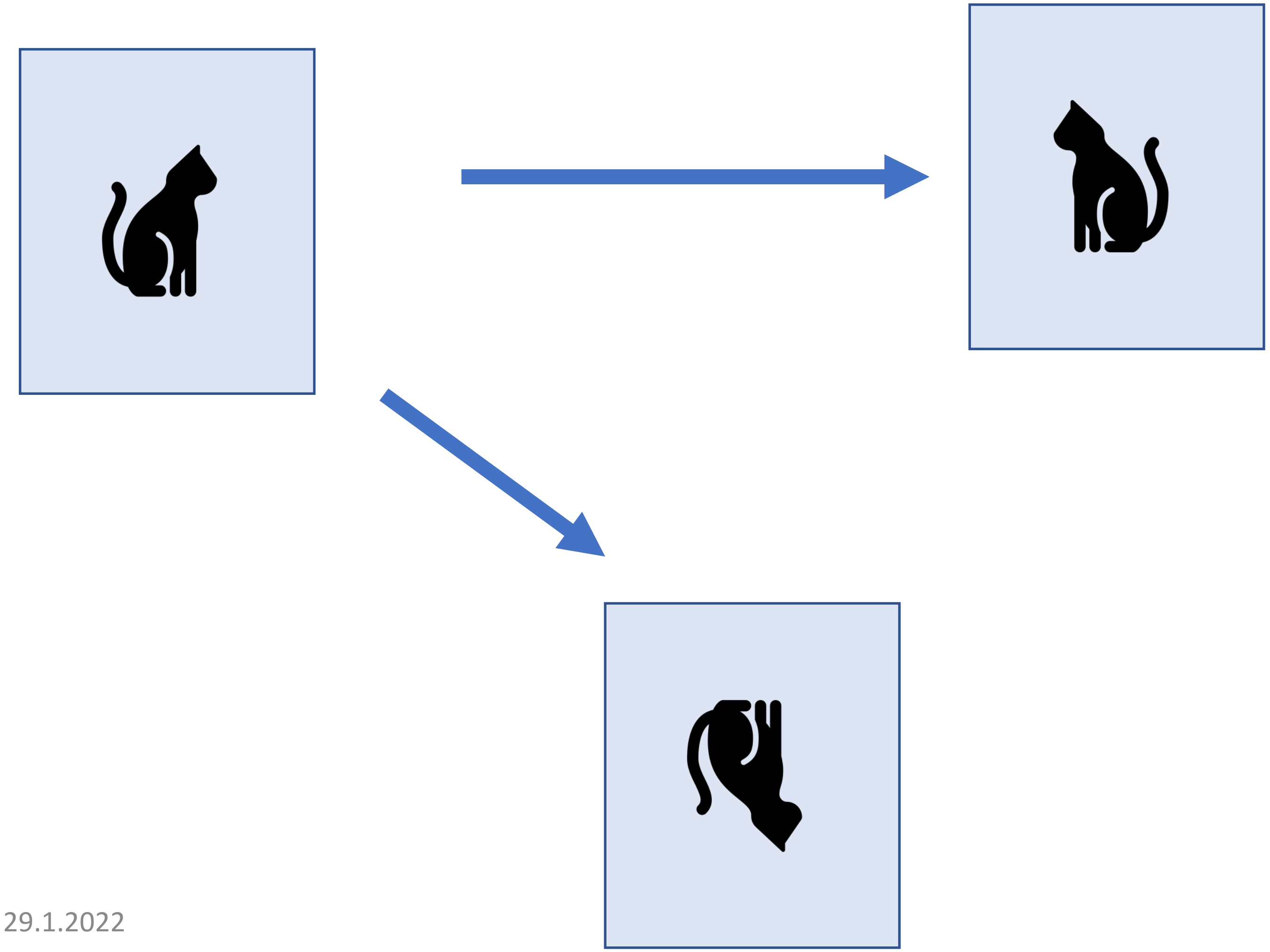


we have increased the dataset by factor 3 !

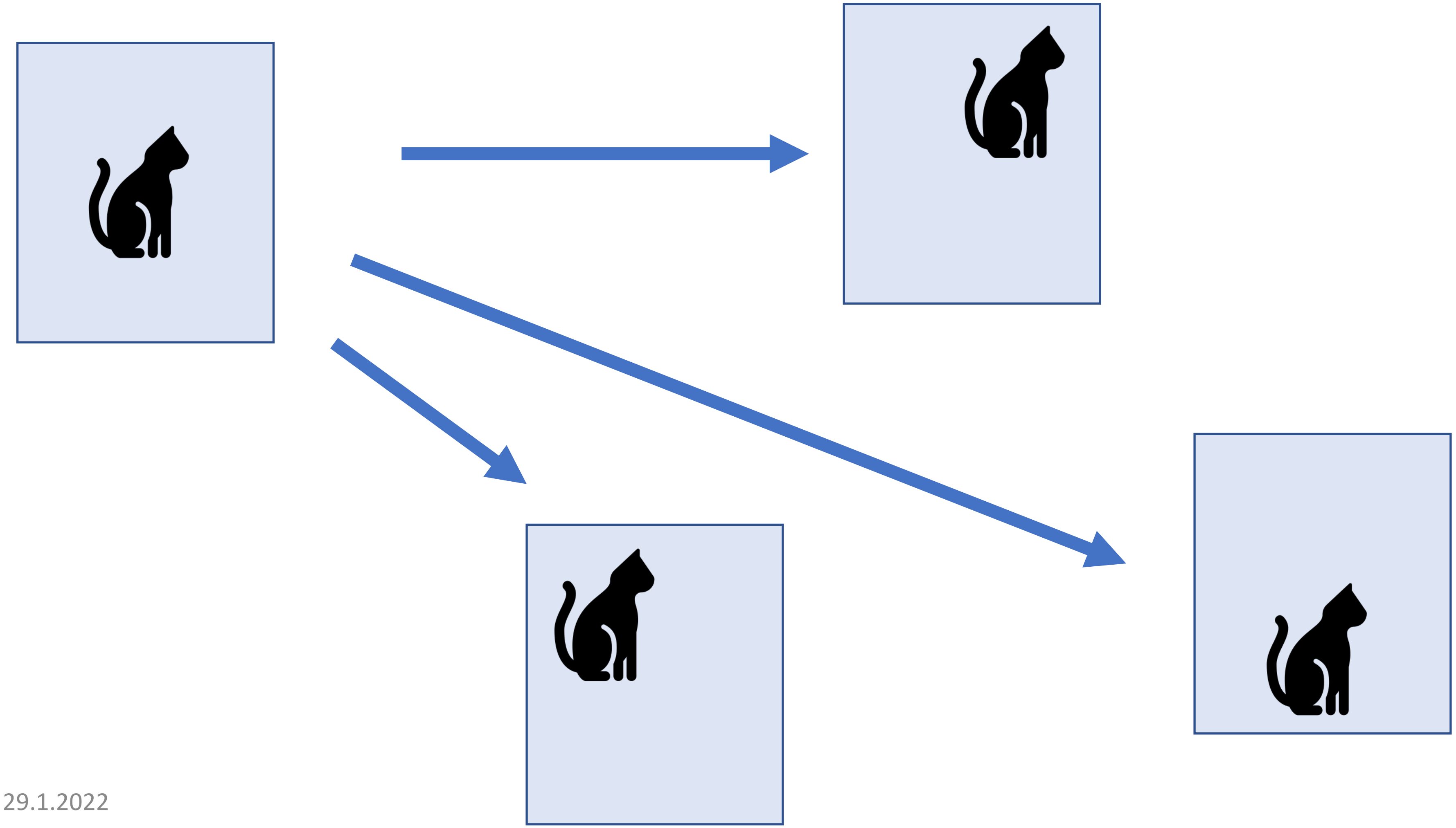
rotated cat image is still cat image



flipped cat image is still cat image



shifted cat image is still cat image



how to bring d/m below critical value?

- increase m by using more training data
- decrease d by using smaller hypothesis space

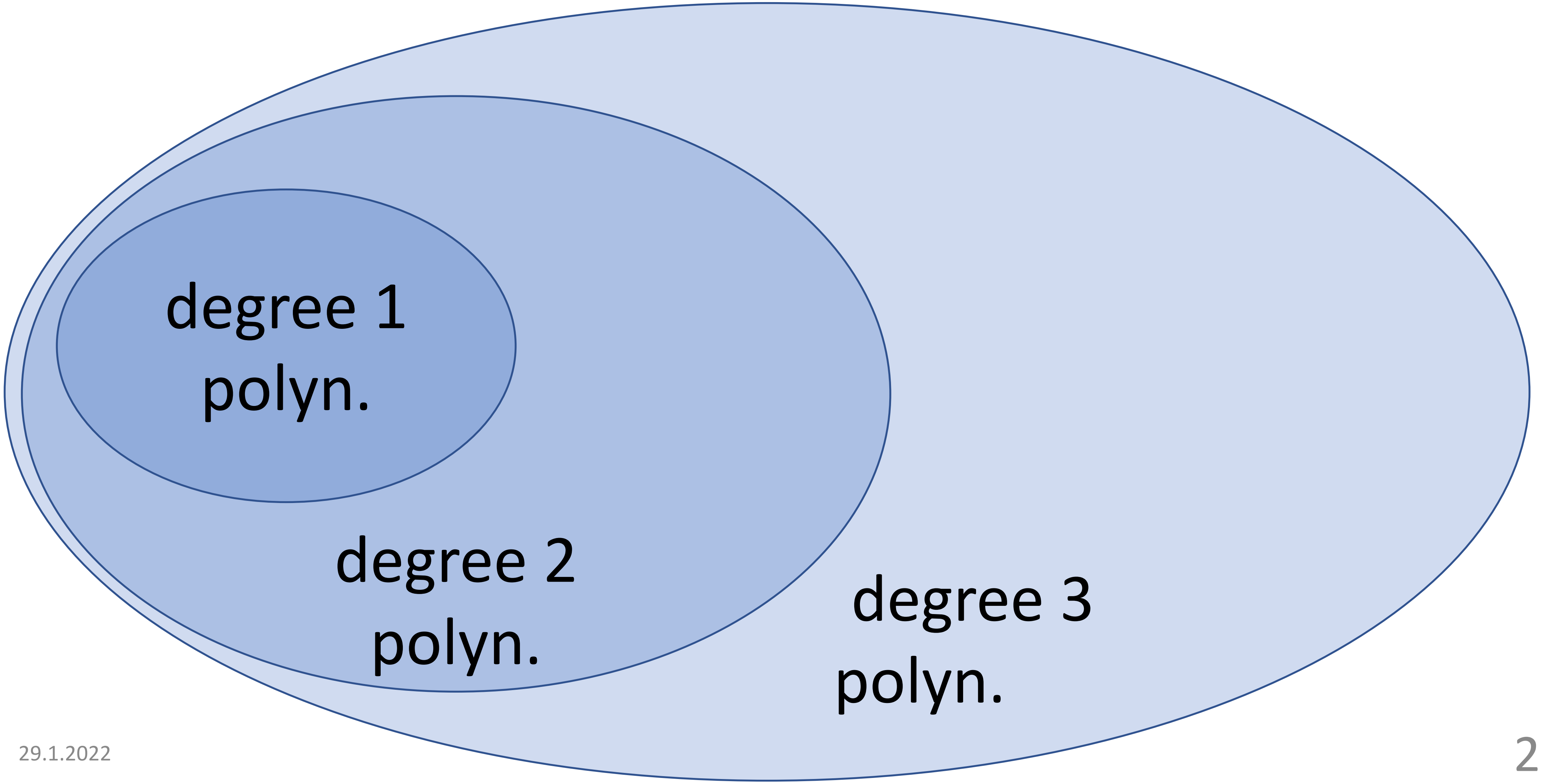
replace original ERM

$$\min_{h \epsilon \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} L\left(\left(x^{(i)}, y^{(i)}\right), h\right)$$

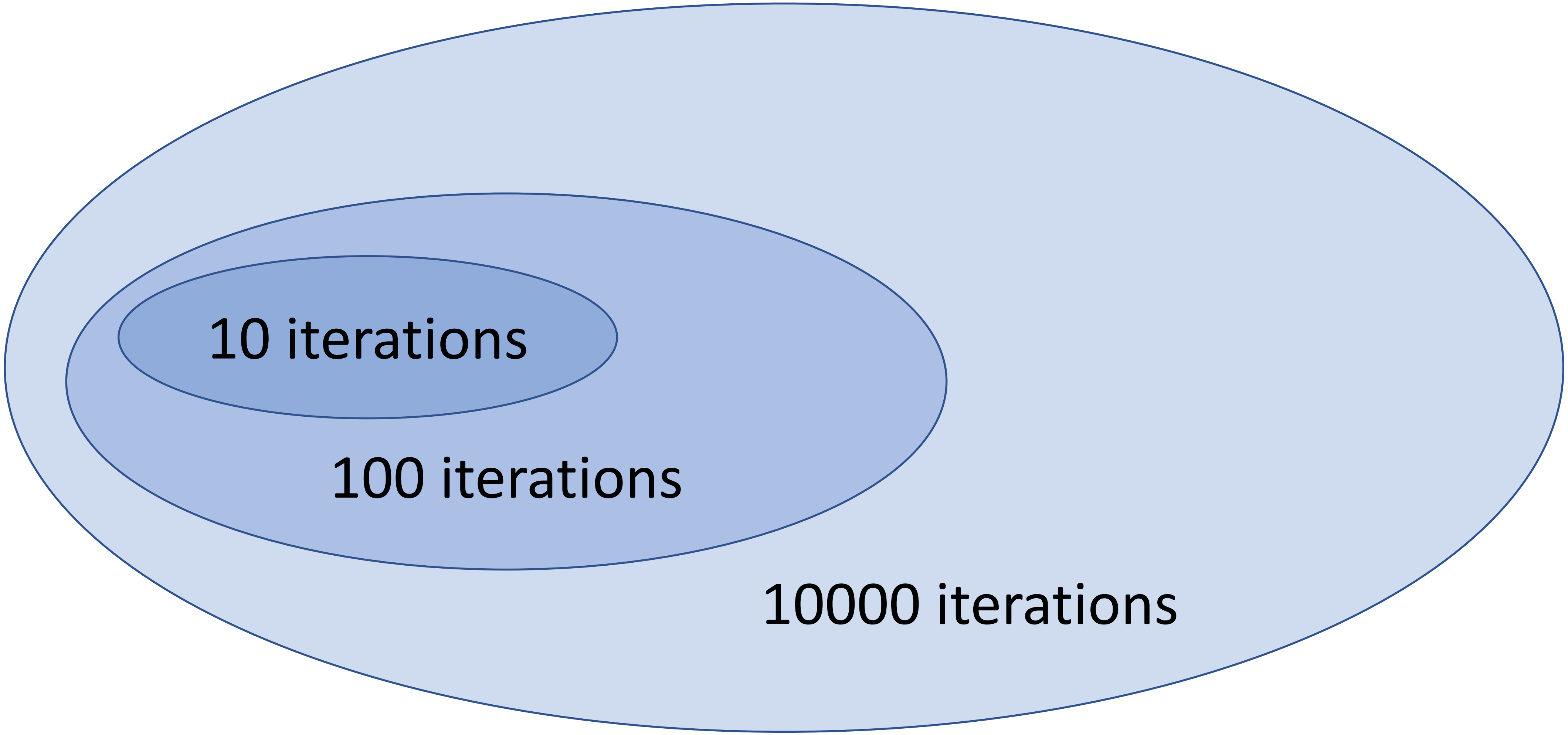
with ERM on smaller $\widehat{\mathcal{H}} \subset \mathcal{H}$

$$\min_{h \epsilon \widehat{\mathcal{H}}} \frac{1}{m} \sum_{i=1}^{m} L\left(\left(x^{(i)}, y^{(i)}\right), h\right)$$

Nested Models



Prune Hypospace by Early Stopping



Soft Model Pruning via Regularization

Regularized ERM

learn hypothesis h out of
model (hypospace) $\mathcal{H}$ by minimizing

$$\underbrace{\frac{1}{m}\sum_{i=1}^{m}\mathcal{L}\big((x^{(i)},y^{(i)}),h\big)}_{\text{average loss on training set (empirical risk of h)}} + \underbrace{\lambda\mathcal{R}(h)}_{\text{loss increase for datapoints outside training set}}$$

average loss on training set
(empirical risk of h)

loss increase for datapoints
outside training set

Regularized Linear Regression

- squared error loss
- linear hypothesis map $h(x) = w^T x = w_1 x_1 + \cdots + w_n x_n$

$$\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - w^T x^{(i)}\right)^2 + \lambda\mathcal{R}(w)$$

- ridge regression uses $\mathcal{R}(w) = \|w\|_2^2 = w_1^2 + \cdots + w_n^2$
- Lasso uses $\mathcal{R}(w) = \|w\|_1 = |w_1| + \cdots + |w_n|$

Regularization = Implicit Pruning!

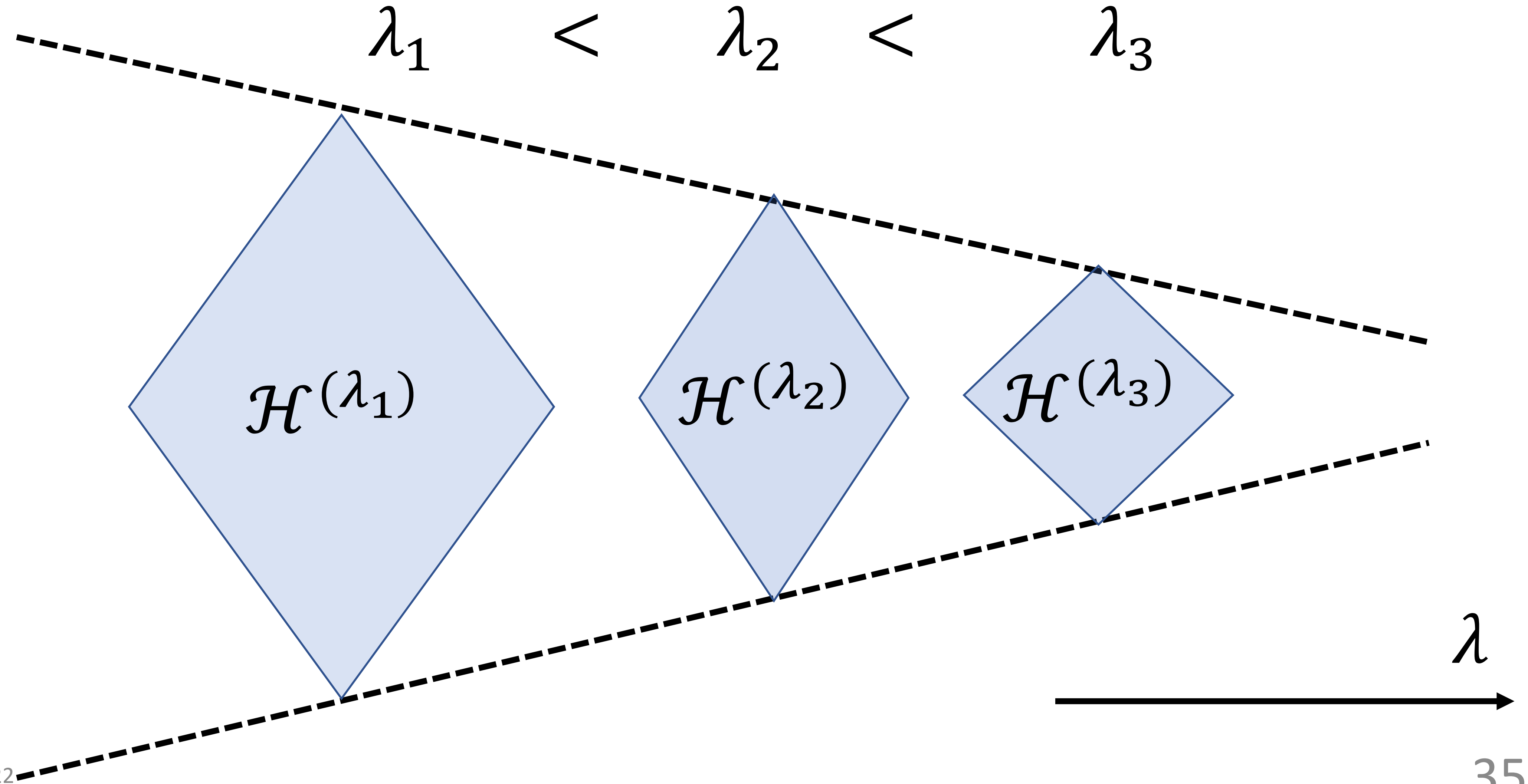
$$\min_{h \epsilon \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(\left(x^{(i)}, y^{(i)}\right), h\right) + \lambda \mathcal{R}(h)$$

equivalent to

$$\min_{h \epsilon \mathcal{H}^{(\lambda)}} \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(\left(x^{(i)}, y^{(i)}\right), h\right)$$

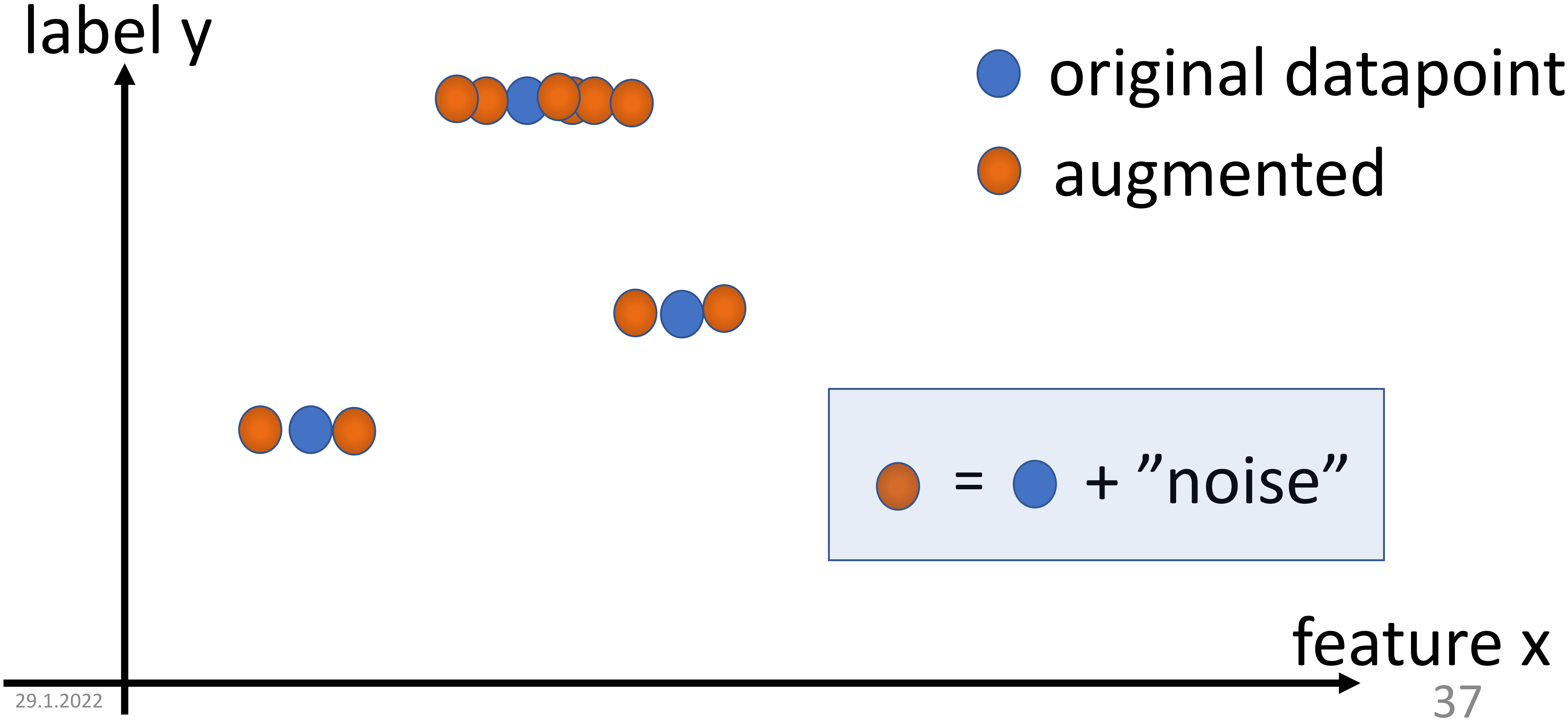
with pruned model $\mathcal{H}^{(\lambda)} \subset \mathcal{H}$

Regularization = “Soft” Model Selection

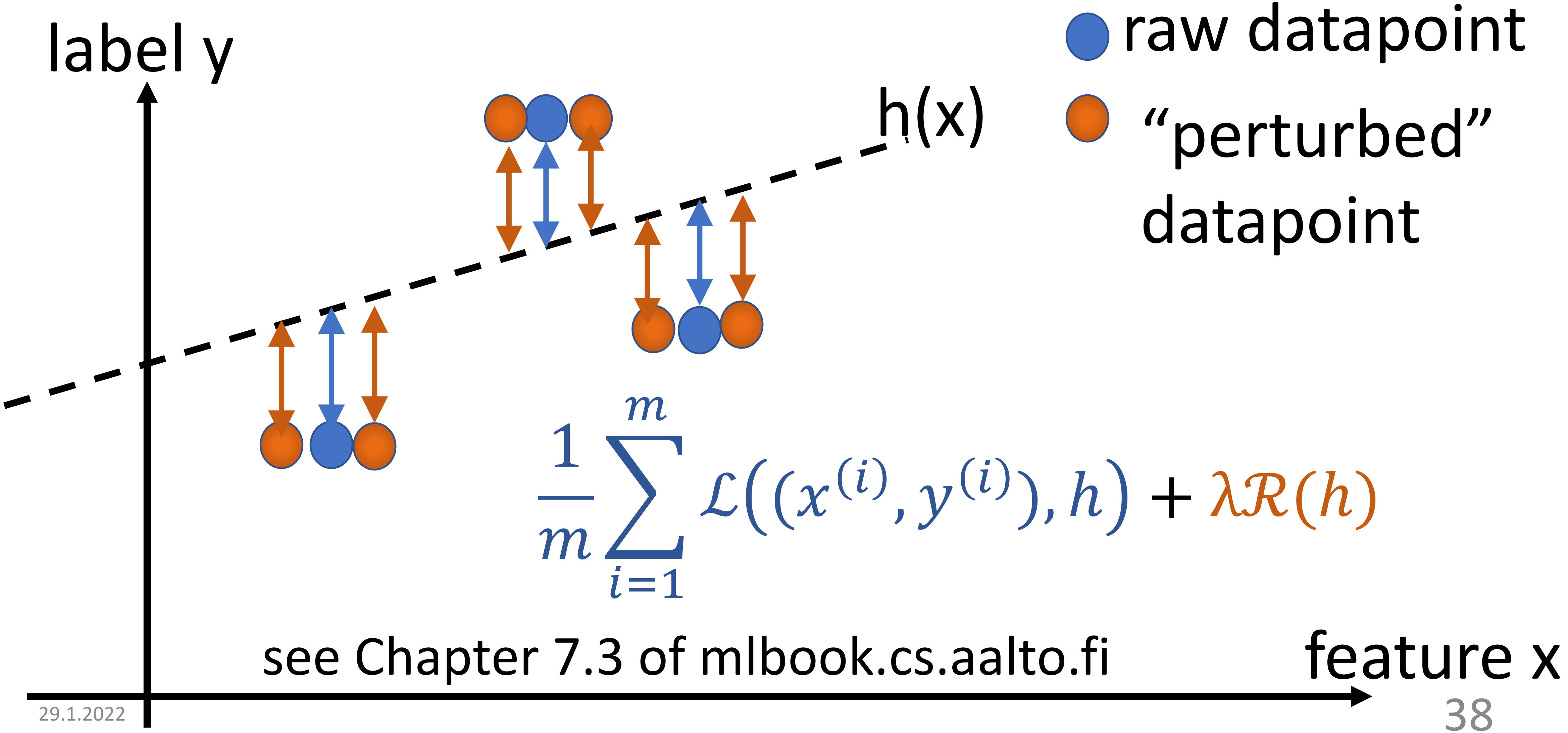


Regularization does implicit Data Augmentation

augment with (infinitely many) realizations of RV!



Regularization =Implicit Data Aug.



$$\frac{1}{m}\sum_{i=1}^{m}\mathcal{L}\big((x^{(i)},y^{(i)}),h\big)+\lambda\mathcal{R}(h)$$

see Chapter 7.3 of mlbook.cs.aalto.fi

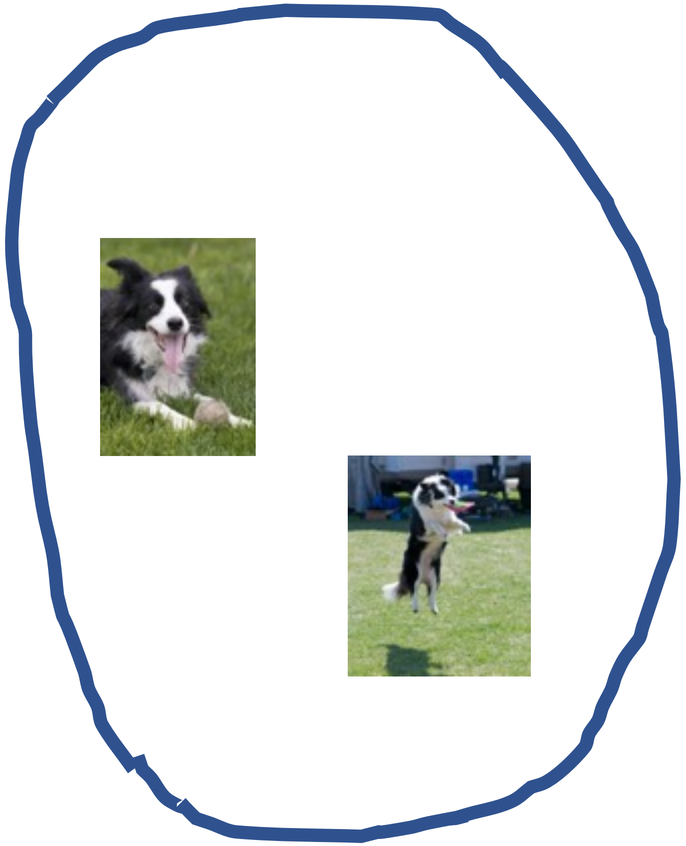
To sum up,

- large ratio d/m leads to overfitting
- reduce d by using smaller model (“pruning”)
- increase m by using more data points
- regularization is a soft model pruning
- regularization does implicit data augmentation

Questions ?

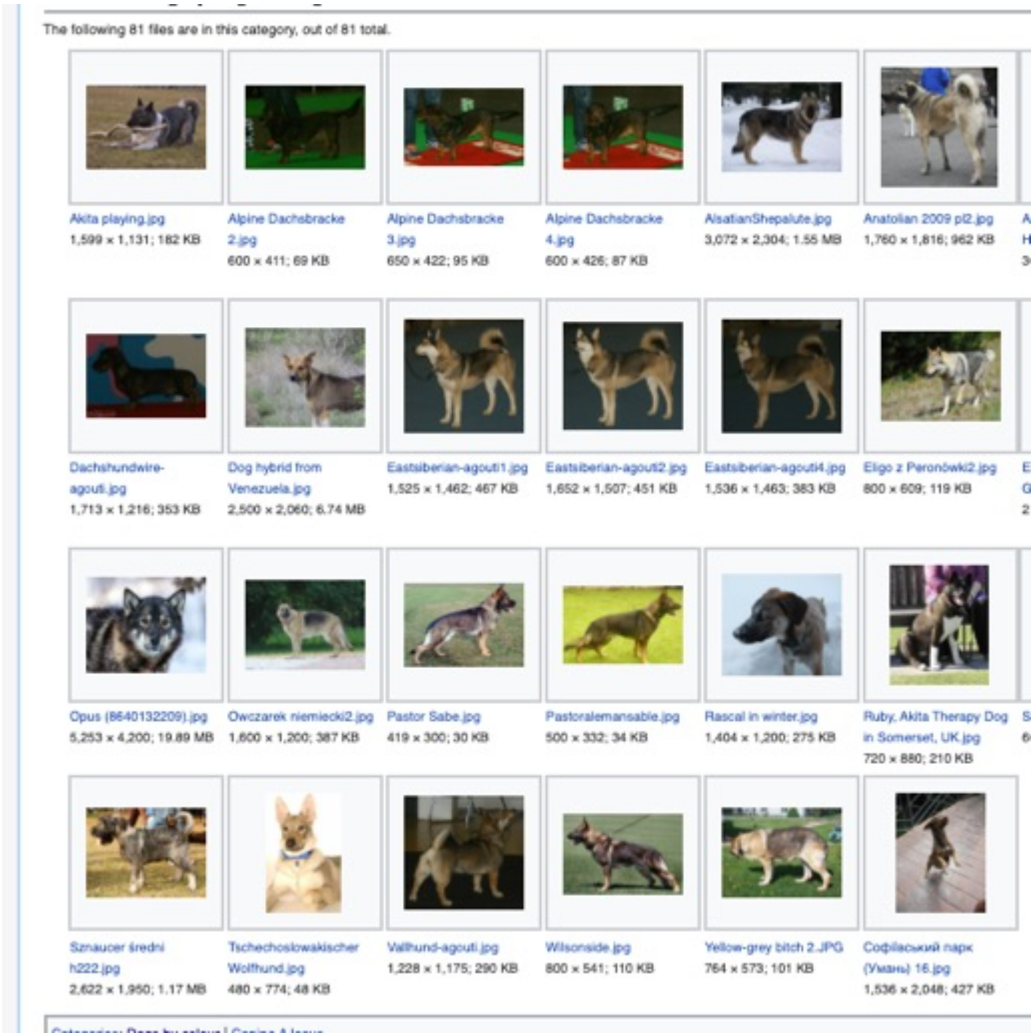
Transfer Learning via Regularization

- Problem I: classify image as “shows border collie” vs. “not”
- Problem II: classify image as “shows a dog” vs. “not”
- ML Problem I is our main interest
- only little training data $\mathcal{D}^{(1)}$ for Problem I
- much more labeled data $\mathcal{D}^{(2)}$ for Problem II
- pre-train a hypothesis on $\mathcal{D}^{(2)}$, fine-tune on $\mathcal{D}^{(1)}$



$\mathcal{D}^{(1)}$

learn h by fine-tuning $\hat{h}$



$\mathcal{D}^{(2)}$

pre-train hypothesis $\hat{h}$

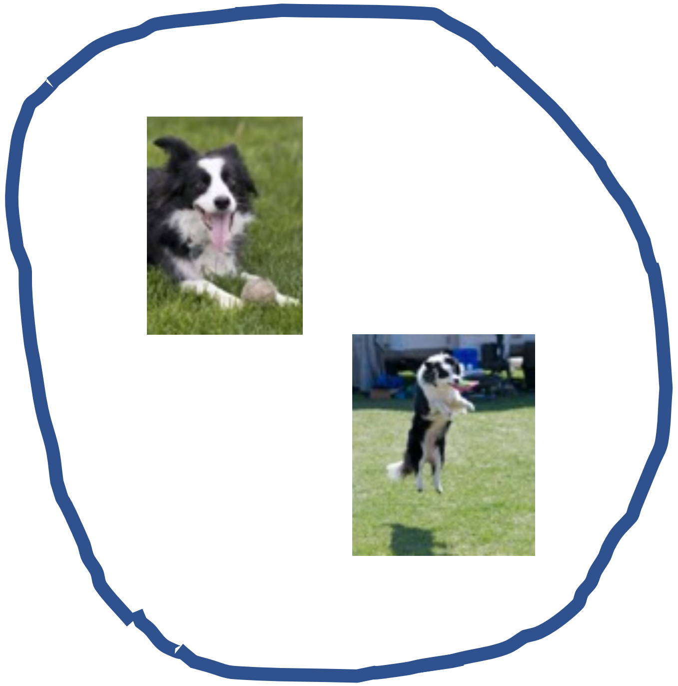
$$\underbrace{\min_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(\left(x^{(i)}, y^{(i)}\right), h\right)}_{\text{fine tuning on } \mathcal{D}^{(1)}} + \underbrace{\lambda d(h, \hat{h})}_{\text{distance to hypothesis } \hat{h} \text{ which is pre-trained on } \mathcal{D}^{(2)}}$$

fine tuning on $\mathcal{D}^{(1)}$

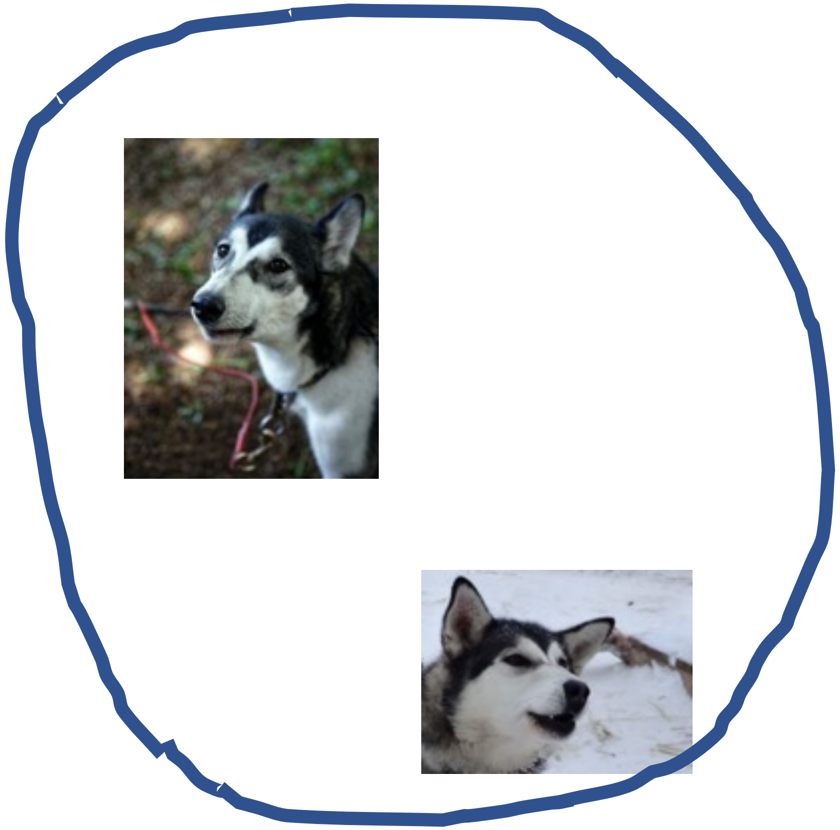
distance to hypothesis $\hat{h}$ which is pre-trained on $\mathcal{D}^{(2)}$

Multi-Task Learning via Regularization

- Problem I: classify image as “shows border colly” vs. “not”
- Problem II: classify image as “shows husky” vs. “not”
- training data $\mathcal{D}^{(1)}$ for Problem I and $\mathcal{D}^{(2)}$ for Problem II
- jointly learn hypothesis $h^{(1)}$ on $\mathcal{D}^{(1)}$ and $h^{(2)}$ on $\mathcal{D}^{(2)}$
- require $h^{(1)}$ to be “similar” to $h^{(2)}$



$\mathcal{D}^{(1)}$



$\mathcal{D}^{(2)}$

jointly learn similar
$h^{(1)}$ and $h^{(2)}$ for each dataset

training error of $h^{(1)}$

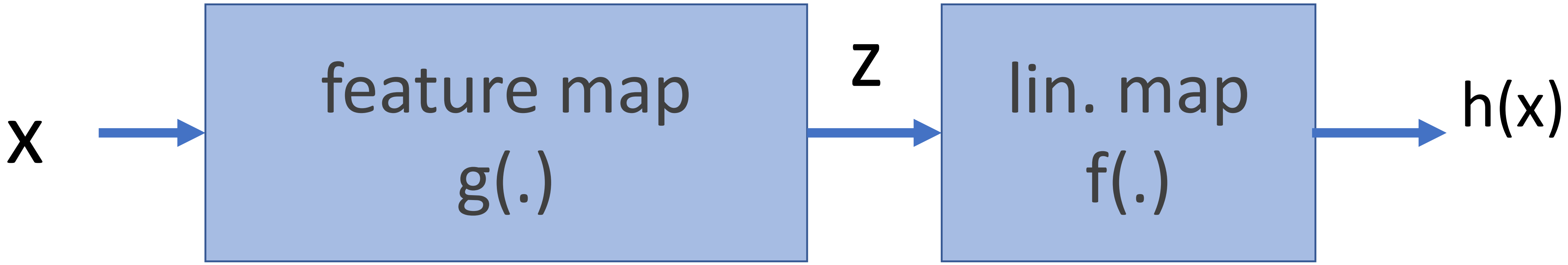
training error of $h^{(2)}$

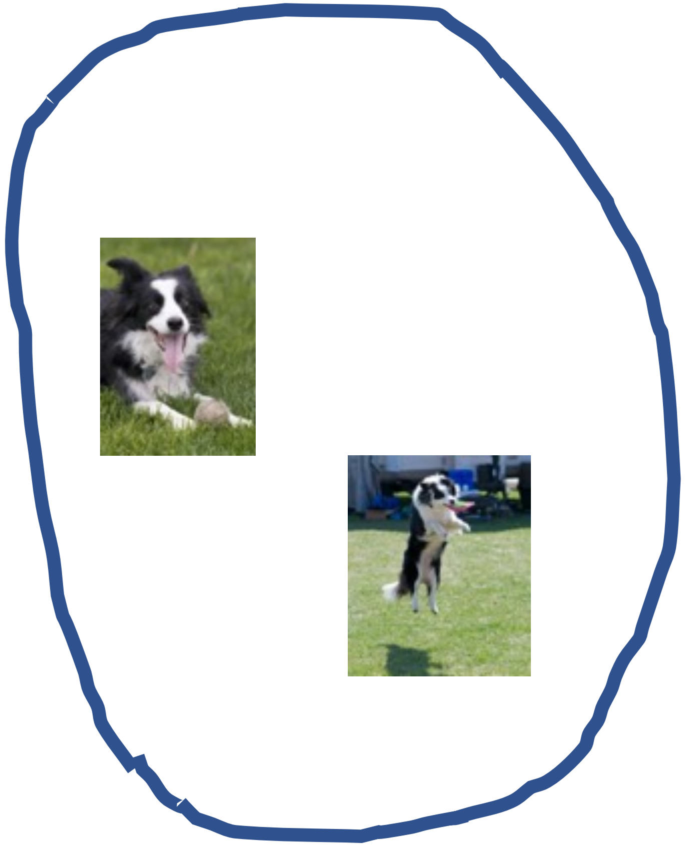
$$\min_{h^{(1)}, h^{(2)}} \mathcal{E}\left(h^{(1)} \middle| \mathcal{D}^{(1)}\right) + \mathcal{E}\left(h^{(2)} \middle| \mathcal{D}^{(2)}\right) + \lambda d\left(h^{(1)}, h^{(2)}\right)$$

“distance” between $h^{(1)}$ and $h^{(2)}$

Semi-Supervised Learning via Regularization

- classify image as “shows border colly” vs. “not”
- small labeled dataset $\mathcal{D}^{(1)}$
- massive image database $\mathcal{D}^{(2)}$ with unlabeled images
- train hypothesis h(.) on $\mathcal{D}^{(1)}$ with following structure:





$\mathcal{D}^{(1)}$

learn linear classifier f(.)

$\mathcal{D}^{(2)}$

learn feature map g(.)

$$\underbrace{\min_{h\epsilon\mathcal{H}} \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}\left(\left(x^{(i)}, y^{(i)}\right), h\right)}_{\text{use training error to fine tune f(.)}} + \lambda\, \underbrace{\mathcal{E}\left(g \middle| \mathcal{D}^{(2)}\right)}_{\text{learn feature map g(.) using large unlabeled database } \mathcal{D}^{(2)}}$$