REFERENCES:

- BISHOP: PATTERN RECOGNITION AND MACHINE LEARNING
- MY PHD THESIS
- MITCHELL: MACHINE LEARNING

DEFINITION:

LEARNING = IMPROVING WITH EXPERIENCE AT SOME TASK

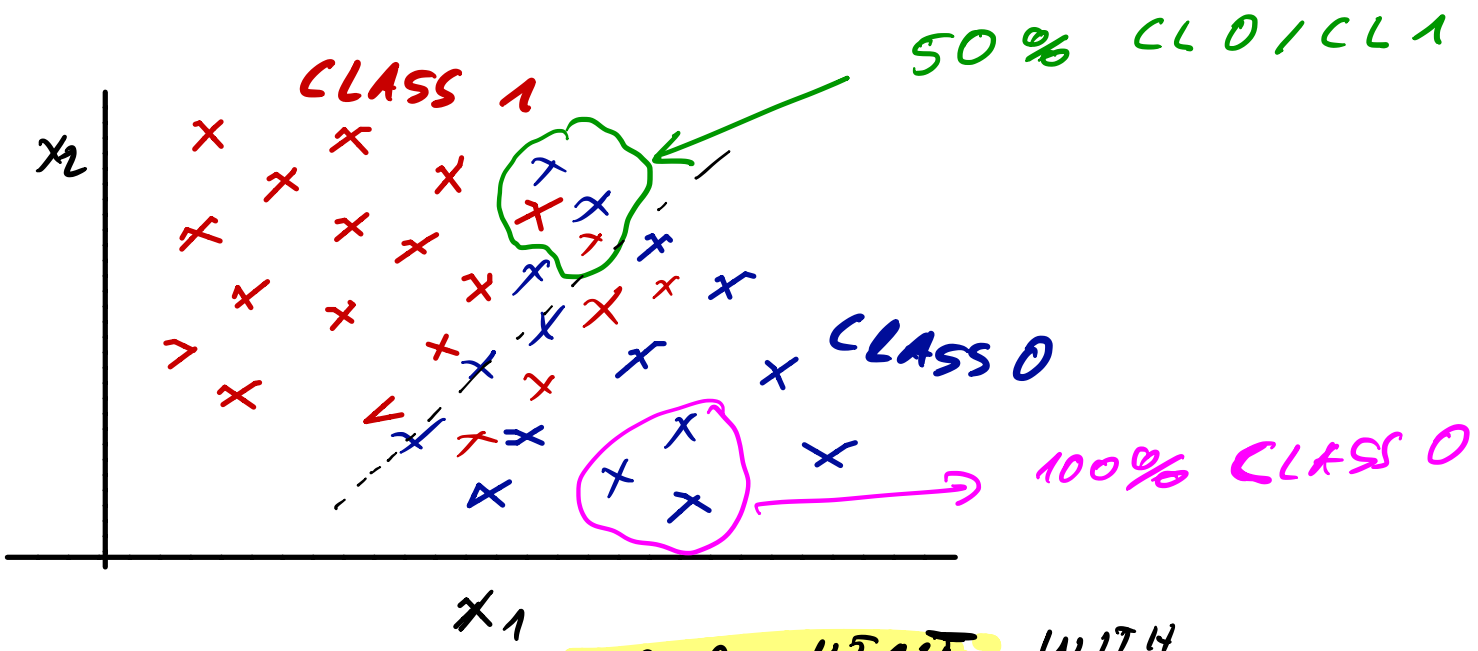
EXPERIENCE: „DATA"

TASK: REGRESSION / CLASSIFICATION / CLUSTERING

BASIC IDEA:

1. FORMULATE LIKELIHOOD FOR PROBLEM (CLASSIFICATION / REGRESSION)
2. FIND SUITABLE FUNCTIONS TO PARAMETRIZE STATISTICS OF LIKELIHOOD
3. FIT DATA

CLASSIFICATION



" COIN-FLIP EXPERIMENT WITH SUCCESS PROBABILITY THAT DEPENDS ON POSITION IN INPUT SPACE "

EXPERIMENT: COIN FLIP WITH STATES:

$$S = \{0, 1\}$$

$$p(x=0) = \mu \qquad p(x=1) = 1-\mu$$

$$P(x=s) = \mu^s (1-\mu)^{1-s}$$

LIKELIHOOD:

$$p(D|\mu) = \prod_{i=1}^{N} \mu^{x_i} (1-\mu)^{1-x_i}$$

LOG LIKELIHOOD:

$$\log(p(D|\mu)) = \sum_{i=1}^{N} x_i \log(\mu) + (1-x_i) \log(1-\mu)$$

" CROSS-ENTROPY COST FUNKT "

MINIMUM OF LOG LIKELIHOOD:

$$\mu_{ML} = \frac{1}{N} \sum_i x_i$$

INTRODUCE DEPENDENCE ON INPUTS

$$p(X=0) = \mu \longrightarrow p(X=0 \mid \vec{x}) = \frac{1}{1+\exp(-f(x))}$$

CONDITIONAL ON POSITION IN INPUT SPACE

$$p(X=0 \mid \vec{x}) = \frac{1}{1+\exp(-f(x))}$$

⇒ INTERPRET AS PROBABILITY SINCE BETWEEN [0,1]. NOTE: OTHER FUNKTS. POSSIBLE

$f(x)$ IS SOME FLEXIBLE FUNKT:

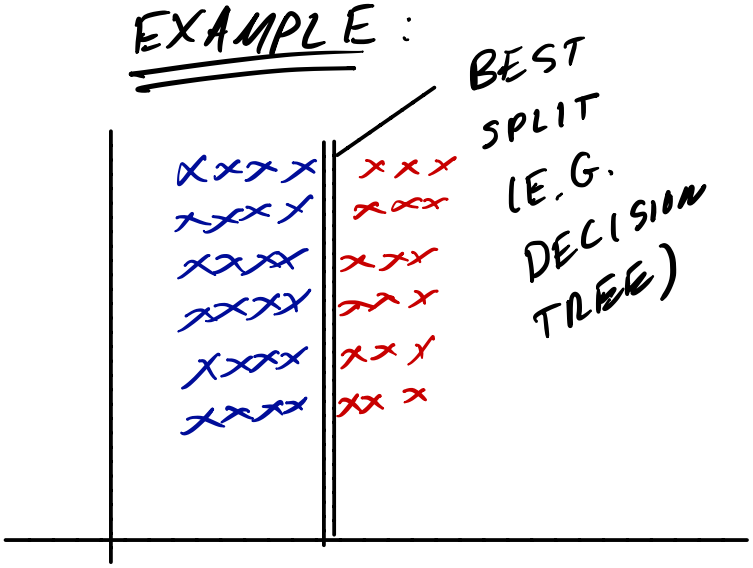
× LINEAR BASIS FUNKT: $f(\vec{x}) = \sum_i w_i \phi(\vec{x})$

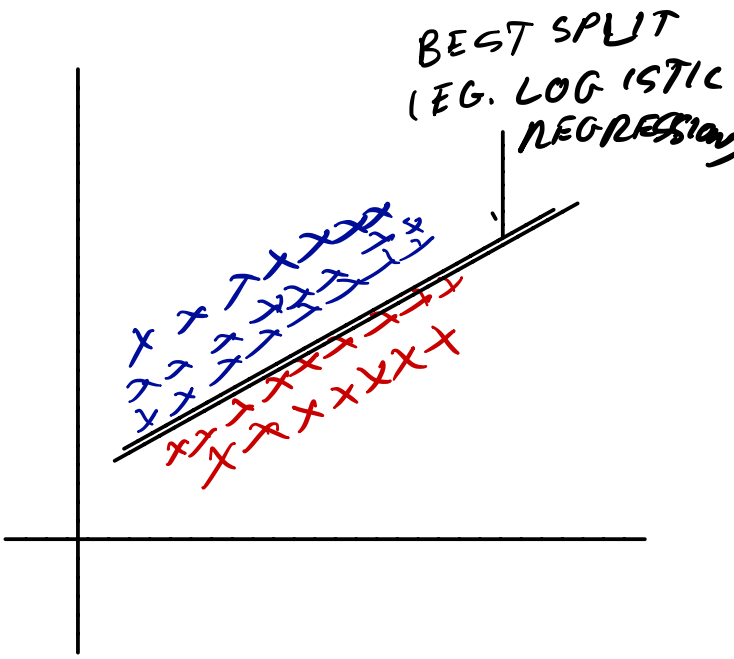
× NEURAL NETWORK: $f(\vec{x}) = \sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i \right)$

h: SIGMOID / TANH.

NOTE: DIFFERENT ML MODELS IMPOSE DIFFERENT $f(\vec{x})$

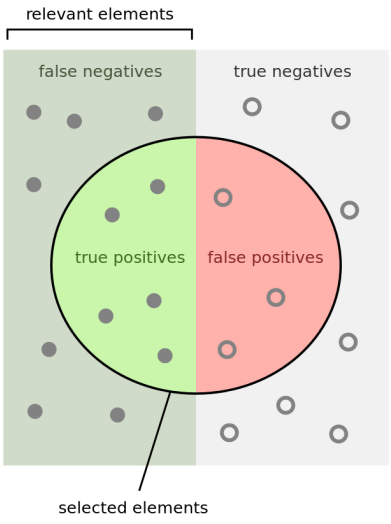
EXAMPLE:

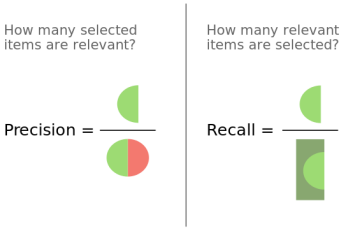




DIFFERENT DATA ⇒ DIFFERENT MODEL PERFORMS BES

HOW TO EVALUATE CLASSIFIERS?





FROM WIKIPEDIA

PROBLEM: IMBALANCED DATASET: 1% LABEL 1
99% " 2

⇒ "CLASSIFIER" THAT "ALWAYS PREDICTS 2 HAS 99% ACCURACY

⇒ F1 SCORE:

$$F1 = \frac{2}{\frac{1}{\text{RECALL}} + \frac{1}{\text{PRECISION}}}$$

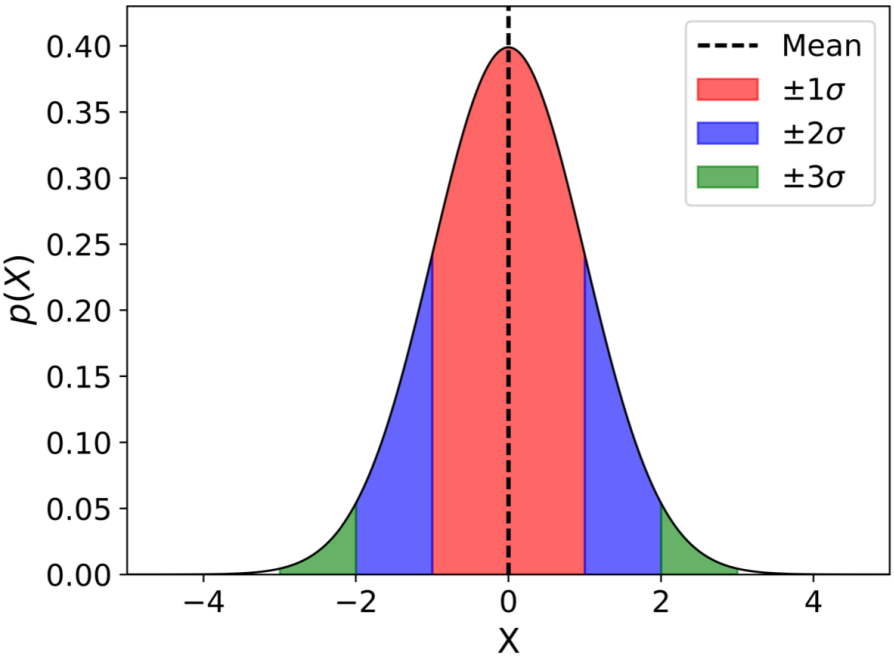
$$= 2\,\frac{\text{PREC} \cdot \text{RECALL}}{\text{PREC} + \text{RECALL}}$$

REGRESSION

GAUSSIAN DISTRIBUTION: 2 PARAMETERS:

μ: MEAN

σ: STANDARD DEVIATION



LIKELIHOOD:

$$p(D|\mu,\sigma) = \prod_{i=1}^{N} \mathcal{N}(x_i|\mu,\sigma)$$

$$\log(p(D|\mu,\sigma)) = -\frac{1}{2\sigma^2}\sum_{i=1}^{N}(x_i-\mu)^2 - \frac{N}{2}\log(\sigma^2) - \frac{N}{2}\log(2\pi)$$

$$\mu_{ML} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma_{ML} = \frac{1}{N}\sum_{i=1}^{N}(x_i-\mu_{ML})^2$$

← MEAN SQUARED ERROR

INTRODUCE DEPENDENCE ON INPUT

$p(y) \longrightarrow p(y|x)$

$\mu \Rightarrow f(x)$ ⇒ MINIMIZE THE MSE

USE "SAME" FUNCTS. AS FOR CLASSIFICATION!

× HOW COMPLEX SHOULD THE FUNCT. BE?

× WHAT IF $p(y|x)$ IS NOT GAUSSIAN?

MODEL COMPLEXITY // BIAS-VARIANCE TRADE OFF

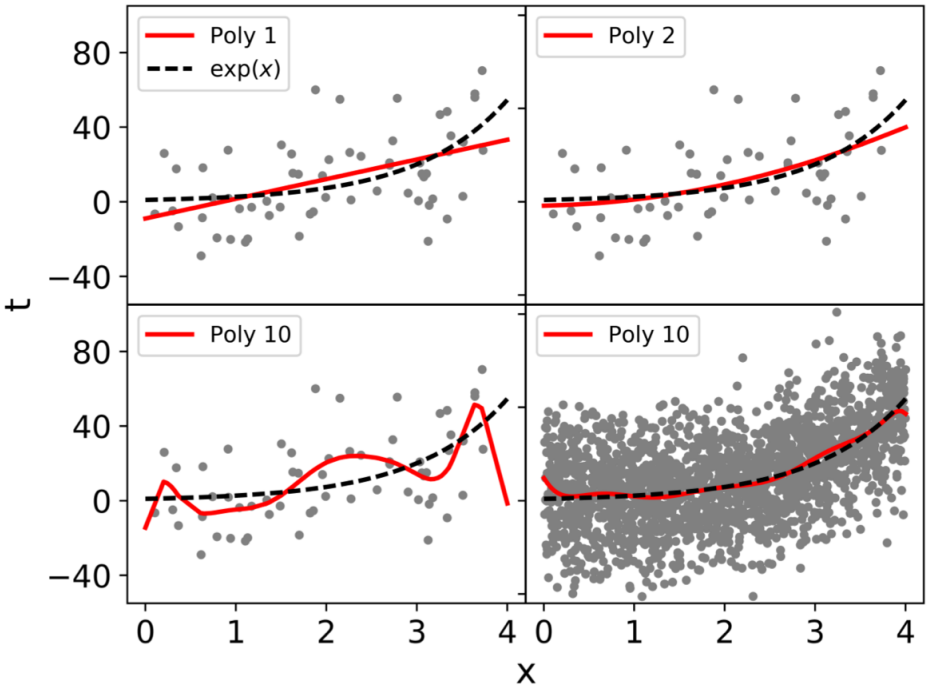


Figure 2.4: Illustration of the impact of model complexity on the accuracy of Machine Learning models. The black dashed line shows the exponential function from which $N = 60$ grey data points have been generated by adding zero mean Gaussian noise with $\sigma = 20$. The red curves show the polynomial models of order M that have been fitted to this data set. The bottom right panel shows the performance of a polynomial model with $M = 10$ fitted on a larger sample with $N = 2000$.

$$E[L] = \iint [y(\vec{x}) - t]^2 p(\vec{x}, t)\, d\vec{x}\, dt$$

$\vec{x}$: INPUT

t : KNOWN TRAINING DATA RESPONSE

$y(x)$: FNKT. TO BE FITTED

$$\frac{\delta E[L]}{\delta y(\vec{x})} = 2 \iint [y(\vec{x}) - t]\, p(\vec{x}, t)\, d\vec{x}\, dt = 0$$

$$\Rightarrow \underbrace{y(\vec{x}) = \int t\, p(t|\vec{x})\, dt}_{\text{CONDITIONAL MEAN}}$$

IRRESPECTIVE OF FORM OF $p(t|\vec{x})$!

FNKT. THAT MINIMIZES MSE

$\Rightarrow$ ESTIMATES CONDITIONAL MEAN

ASSUME WE HAVE SUCH AN ESTIMATE

$\hat{g}(x)$

$$E[L] = \int\int [\hat{g}(x) - t]^2 p(x,t)\, d\vec{x}\, dt$$

EXPAND AROUND OPTIMAL $y(x)$

$$E[L] = \int (\hat{g}(\vec{x}) - g(\vec{x}))^2 p(\vec{x})\, d\vec{x} \quad (A)$$

$$+ \int\int (g(\vec{x}) - t)^2 p(\vec{x}, t)\, d\vec{x}\, dt \quad (B)$$

(A): HOW CLOSE IS $\hat{g}(x)$ to optimal?

(B): WHAT IS THE „INTRINSIC" ERROR IN THE DATA

DECOMPOSE (A) FURTHER:

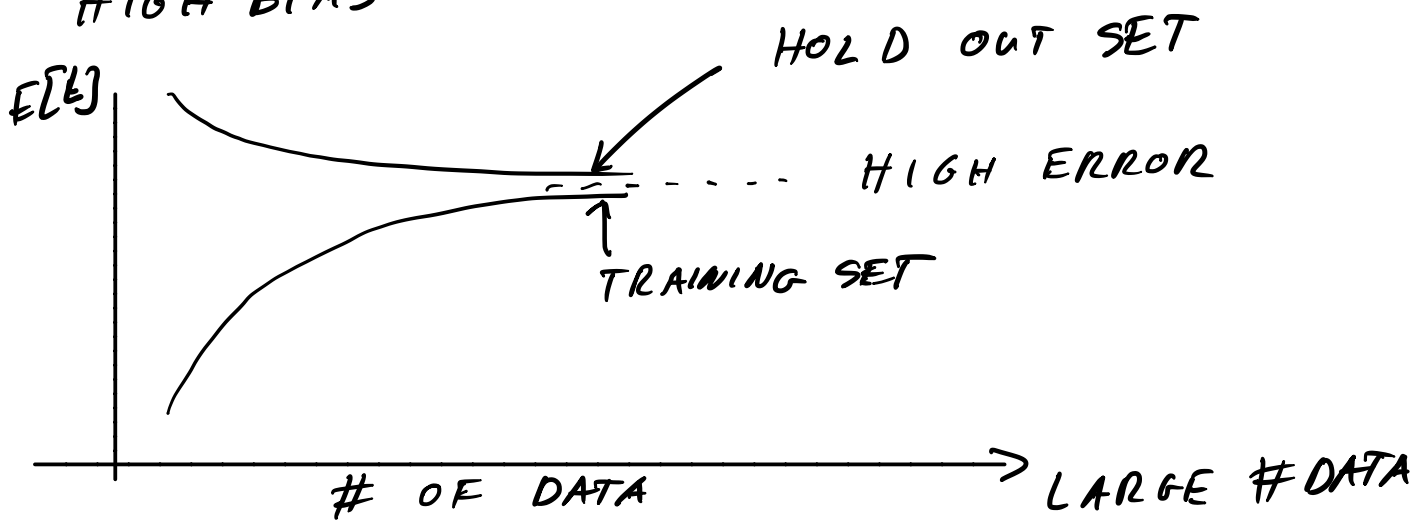
$$E[L] = (\text{BIAS})^2 + \text{VARIANCE} + \underbrace{\text{NOISE}}_{(B)}$$

$$(\text{BIAS})^2 = \int (E_D[\hat{g}(\vec{x}; D)] - g(\vec{x}))^2 p(\vec{x})\, dx$$

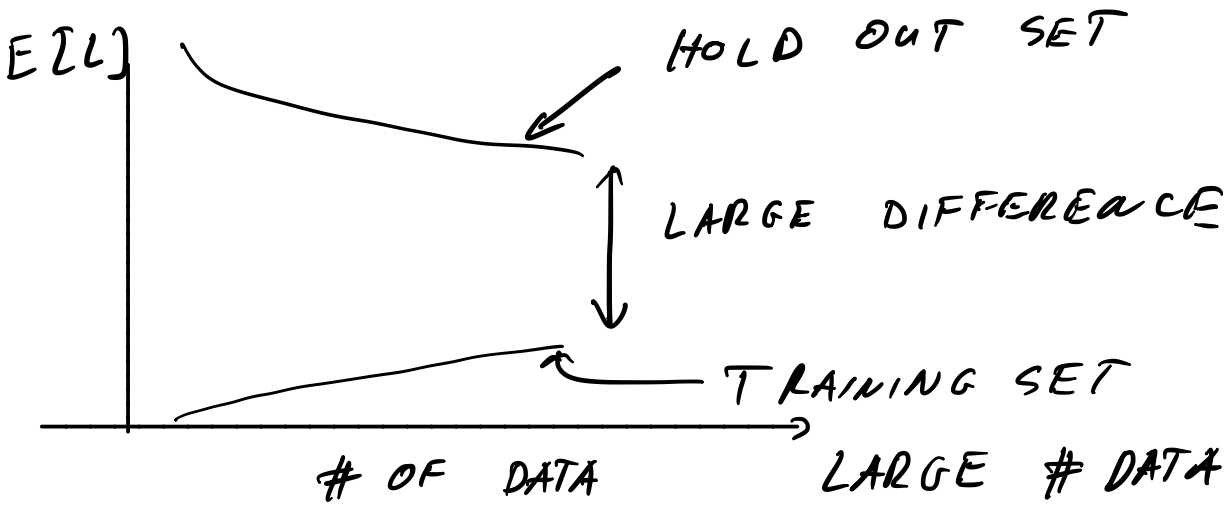
$$\text{VARIANCE} = \int E_D\left[(\hat{g}(x; D) - E_D(\hat{g}(x; D))]^2\right] p(x)\, dx$$

2 SCENARIOS:

HIGH BIAS



⇒ MAKE MODEL MORE COMPLEX
(FEATURE ENGINEERING, MORE LAYERS, HIGHER ORDER POLYNOMIALS....)



⇒ ADD MORE DATA, MAKE MODEL LESS COMPLEX;
„REGULARIZATION", LESS WEIGHTS, ...