Machine Intelligence:: Deep Learning

Beate Sick, Lilach Goren, Pascal Bühler

Institut für Datenanalyse und Prozessdesign
Zürcher Hochschule für Angewandte Wissenschaften

Most of the slides were developed by Oliver Dürr & Beate Sick

Outline of the DL Module (tentative)

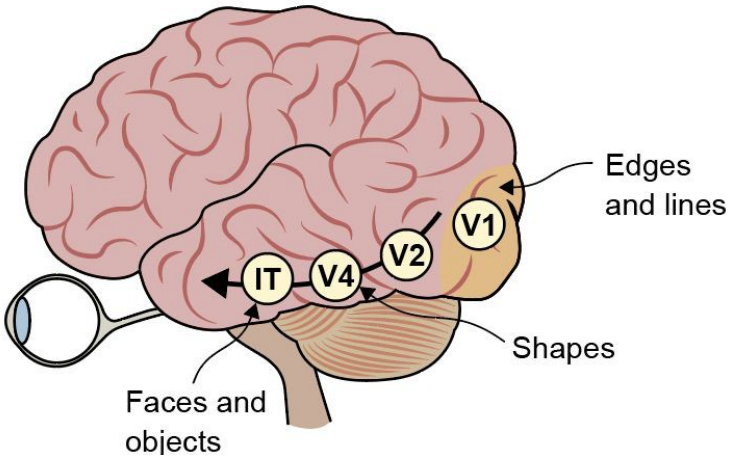
The course is split in 8 sessions, each 4 lectures long. Topics might be adapted during the course

Day	Date	Time	Topic
1	15.04.2025	09:00-12:30	Introduction to Deep Learning & Keras, first NNs
-	21.04.2025	-	FRÜHLINGS-FERIEN
-	28.04.2025	-	FRÜHLINGS-FERIEN
2	06.05.2025	09:00-12:30	Loss, Optimization, Regression, Classification
3	13.05.2025	09:00-12:30	Computer vision, CNN-archictecture
4	20.05.2025	09:00-12:30	DL in practice, pretrained (foundation) models
5	27.05.2025	09:00-12:30	Model evaluation, baselines, xAI, troubleshooting
6	03.06.2025	09:00-12:30	Generative Models, Transformer-architecture
7	10.06.2025	09:00-12:30	Vision Transformer
8	17.06.2025	09:00-12:30	Projects, deep Ensembling

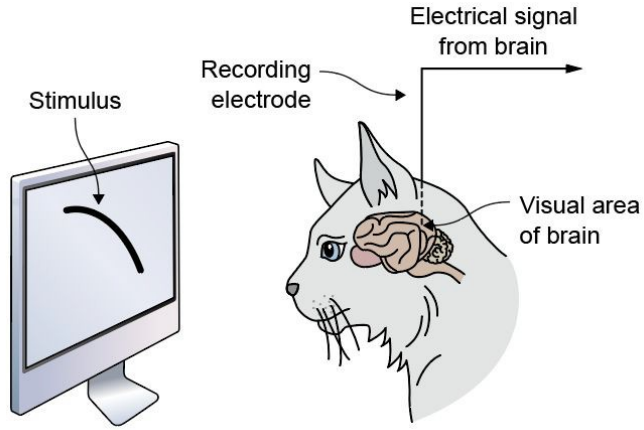
Topics

- Some similarities of the brain and NNs
- xAI: Understand where the NN has looked at to make the prediction
- Model evaluation: Prediction Performance & Calibration
- Projects

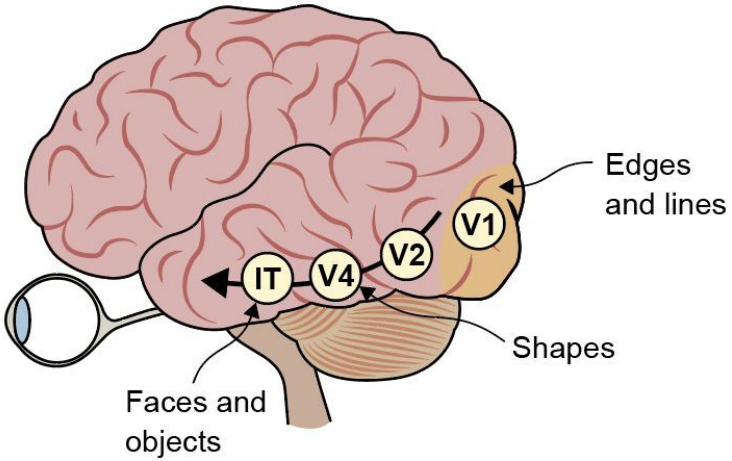
Biological and Artificial NNs

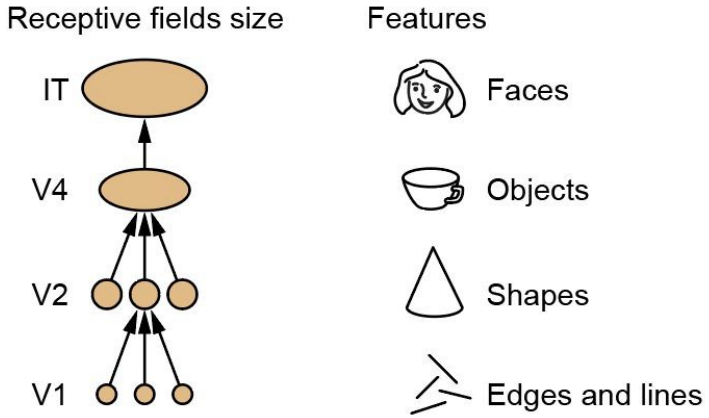


How does the brain respond to visually recieved stimuli?



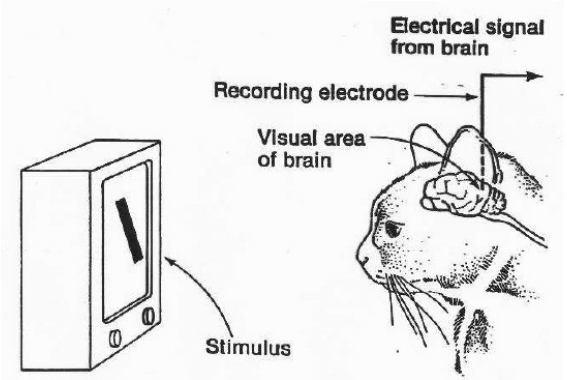
Setup of the experiment of Hubel and Wiesel in late 1950s in which they discovered neurons in the visual cortex that responded when moving **edges** were shown to the cat.

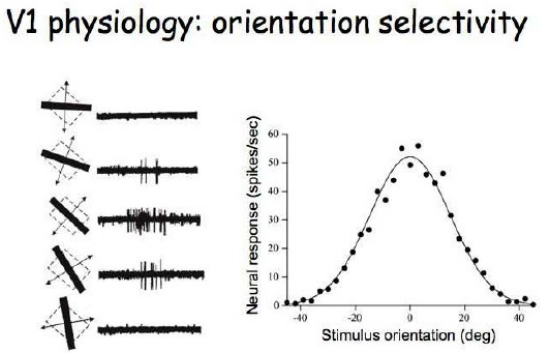


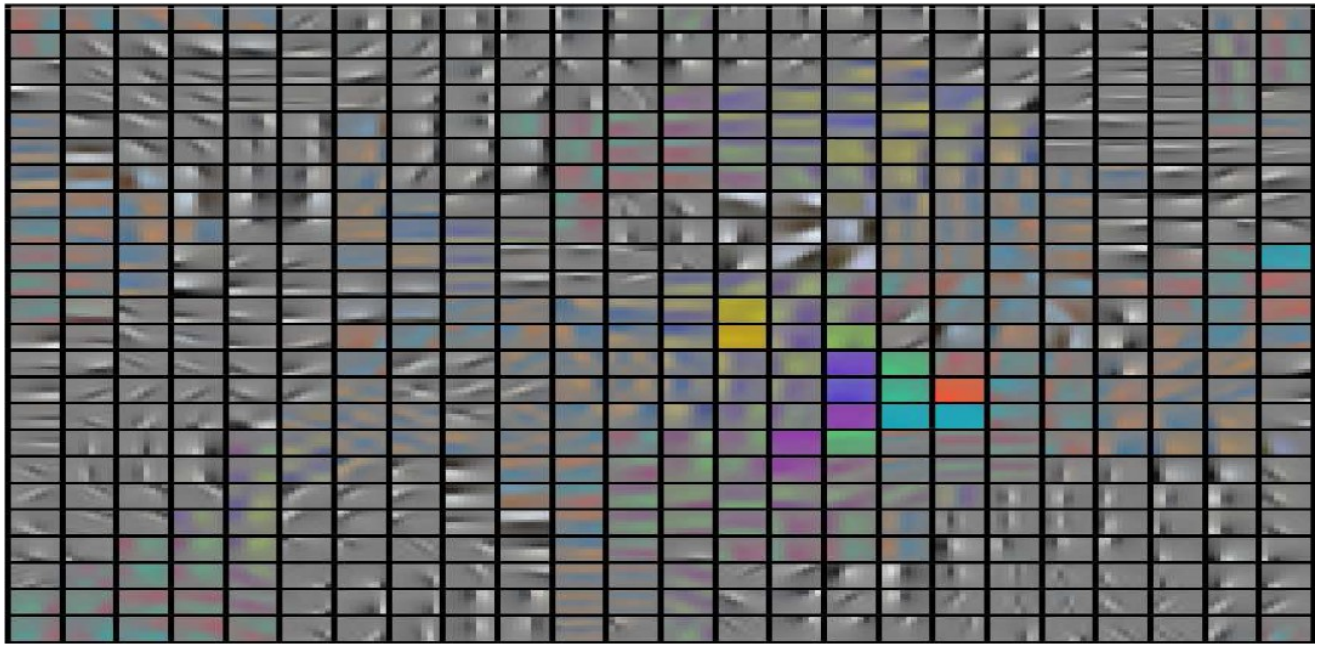


Organization of the visual cortex in a brain, where neurons in different regions respond to more and more complex stimuli

Compare neurons in brain region V1 in first layer of a CNN





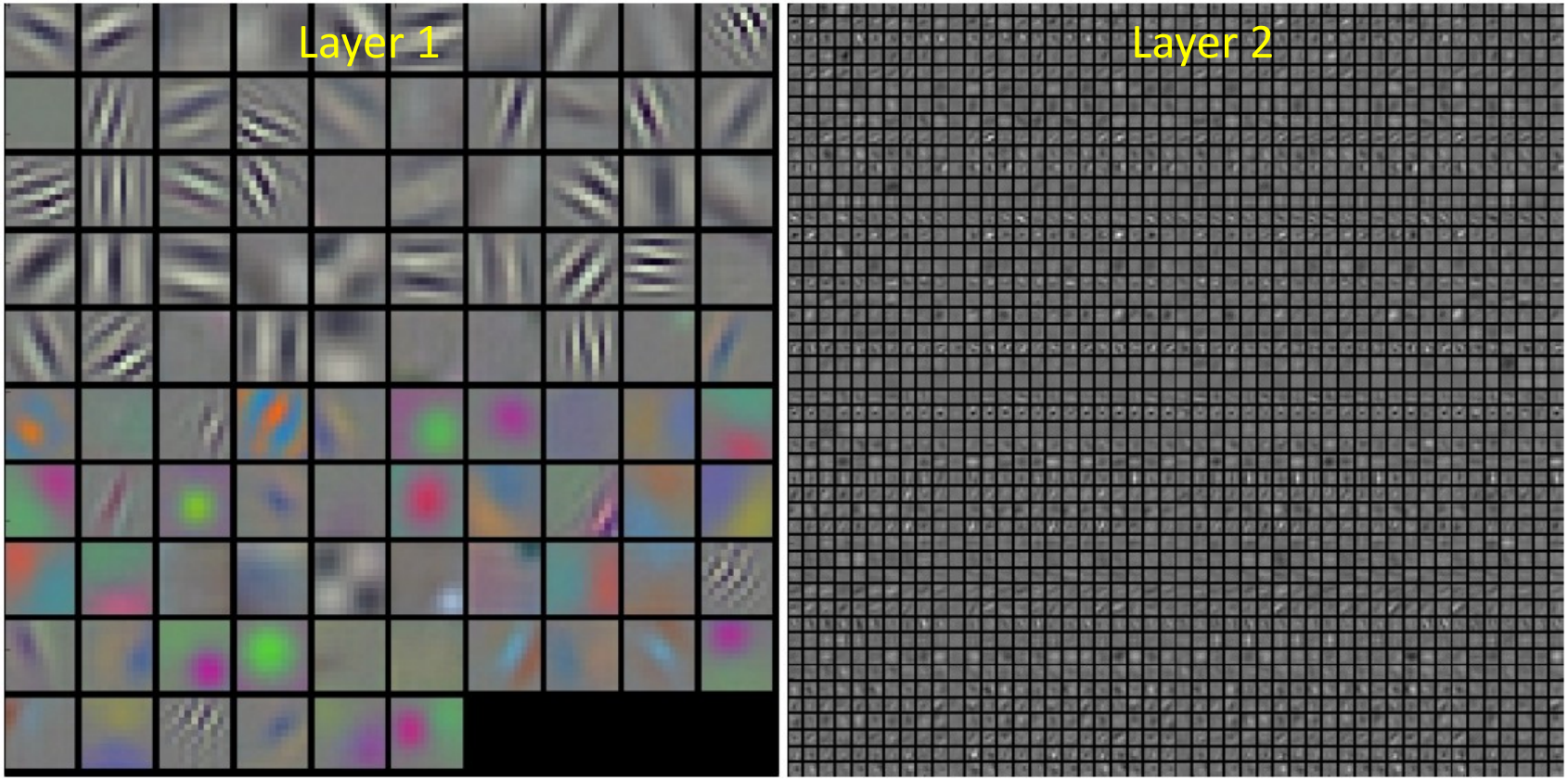


Neurons in brain region V1 and neurons in 1. layer of a CNN respond to similar patterns

Slide (modified) credits: Nando de Freitas

Visualize the weights used in filters

Filter weights from a trained Alex Net

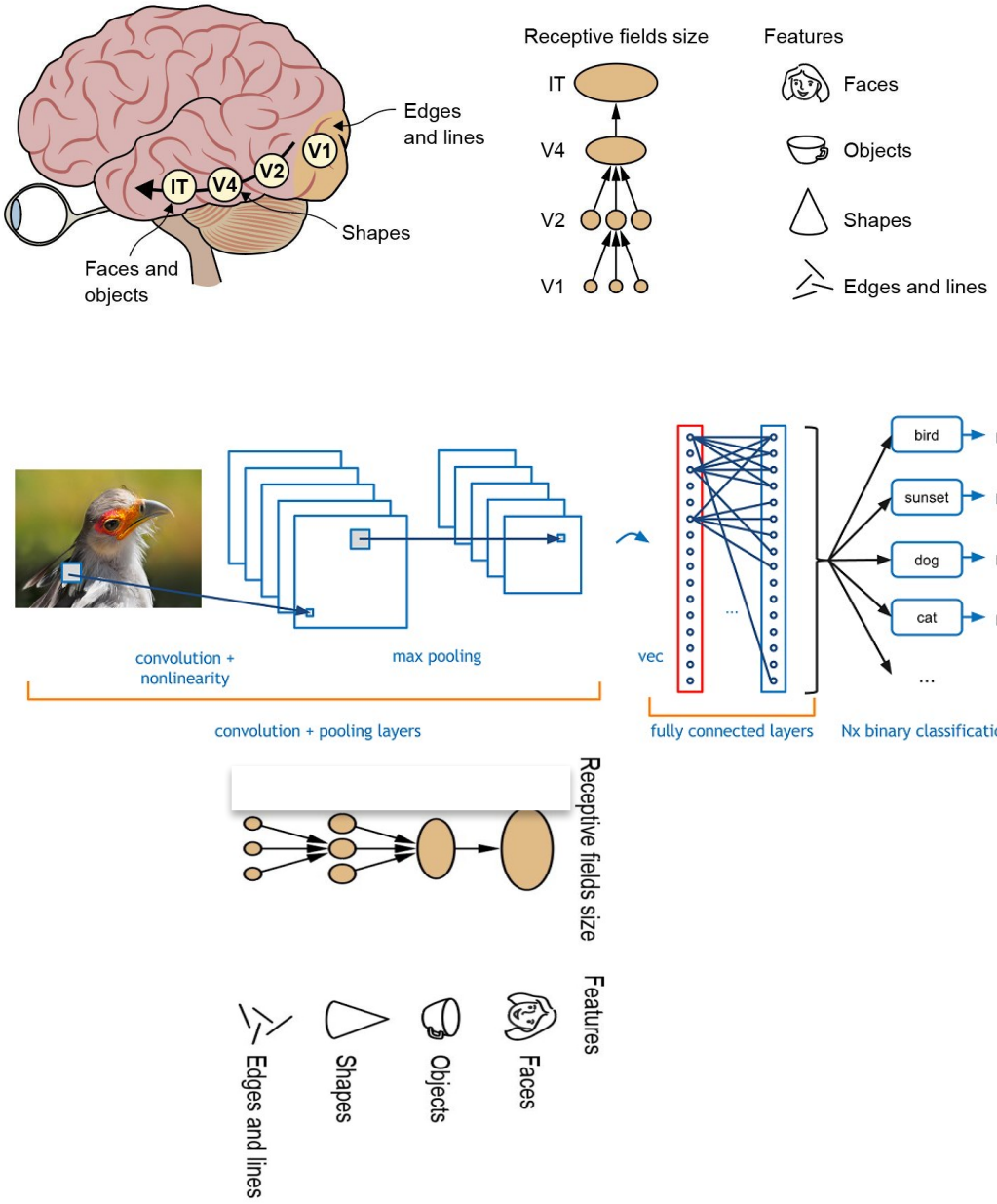


Only in layer 1 the filter pattern correspond to extracted patterns in the image.

In higher layers we can only check if patterns look noisy, which would indicate that the network that hasn't been trained for long enough, or possibly with a too low regularization strength that may have led to overfitting.

Image credits: http://cs231n.github.io/understanding-cnn/

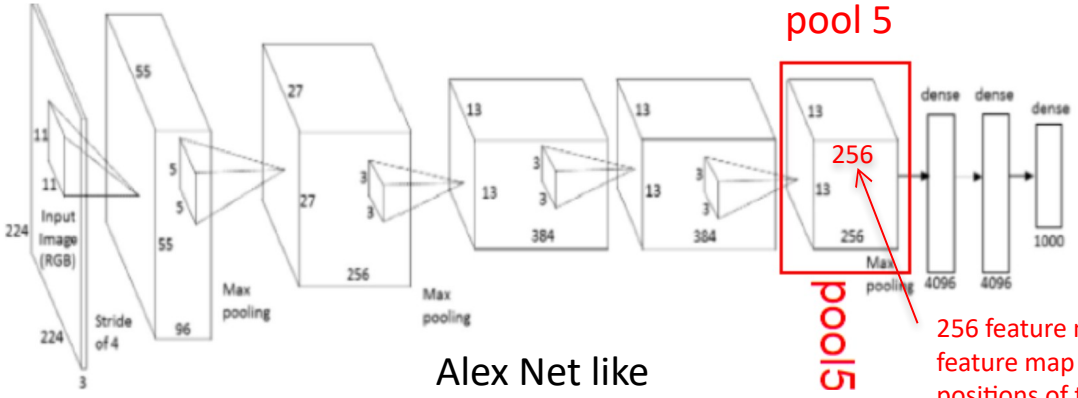
Weak analogies between brain and CNNs architecture



xAI = explainable AI
Where does a CNN look at?

Visualize patches yielding high values in activation maps

Figure 4: Top regions for six pool$_5$ units. Receptive fields and activation values are drawn in white. Some units are aligned to concepts, such as people (row 1) or text (4). Other units capture texture and material properties, such as dot arrays (2) and specular reflections (6).



Alex Net like

http://cs231n.github.io/understanding-cnn/

Here we show image patches that activate maps in layer 5 most.

256 feature maps generated by 256 different 3x3 filters. Each feature map consists of equivalent neurons looking at different positions of the input volume.

What kind of image (patches) excites a certain neuron corresponding to a large activation in a feature map?

10 images from data set leading to high signals 6 feature maps of **conv6**

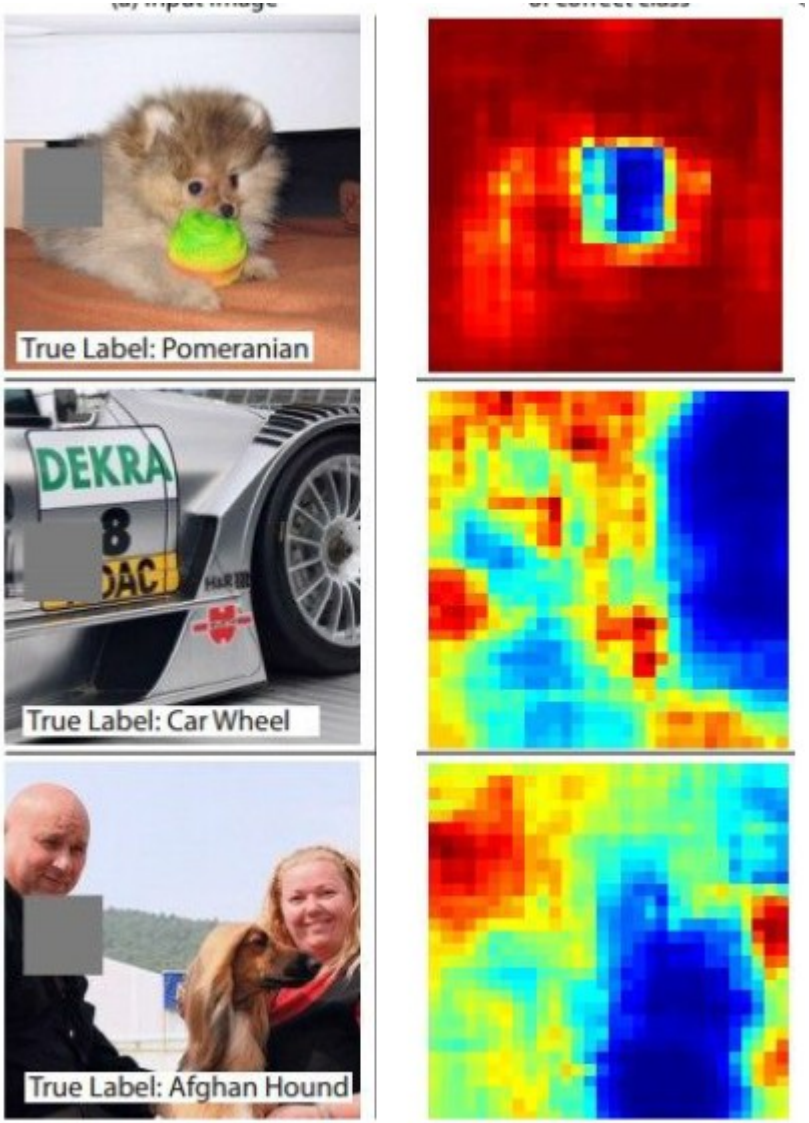
10 images from data set leading to high signals 6 feature maps of **conv9**

image credit: cs231n

Which pixels are important for the classification?
Occlusion experiments

Occlude part of the image with a mask and check for each position of the mask how strongly the score for the correct class is changing.

Warning:
Usefulness depends on application…



Occlusion experiments [Zeiler & Fergus 2013]

image credit: cs231n

Which pixels are important for the classification?
LIME: Local Interpretable Model-agnostic Explanations

Idea:

1) perturb interpretable features of the instance – e.g. randomly delete super-pixels in an image and track as perturbation vector such as (0,1,1,0,…,1)=x.
2) Classify perturbed instance by your model, here a CNN, and track the achieved classification-score
3) Identify for which features/super-pixels the presence in the perturbed input version are important to get a high classification score (use RF or lasso for y~x)

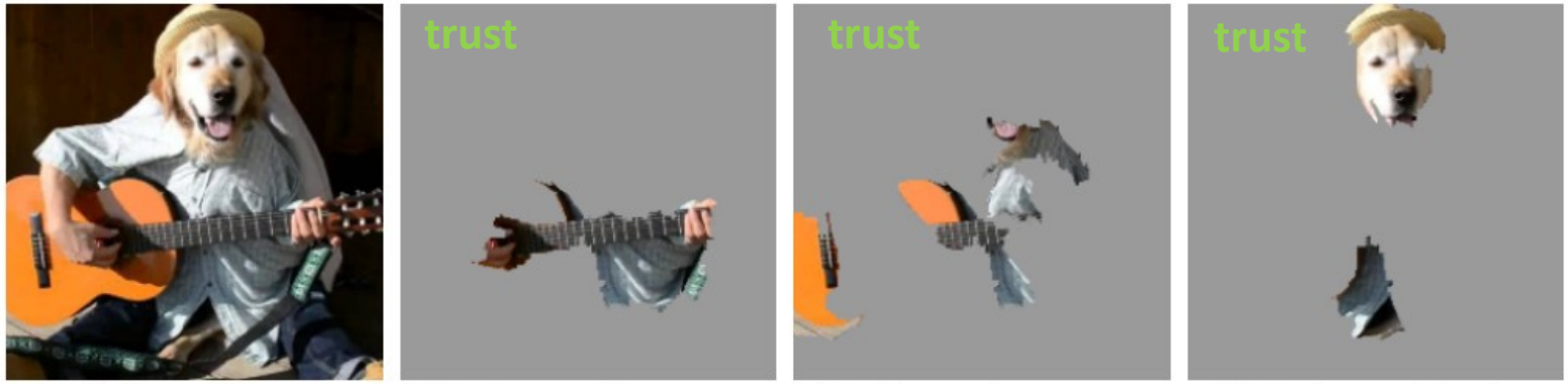
https://arxiv.org/abs/1602.04938

(a) Husky classified as wolf (b) Explanation

-> presence of snow was used to distinguish wolf and husky

-> Explain the CNN classification by showing instance-specific important features

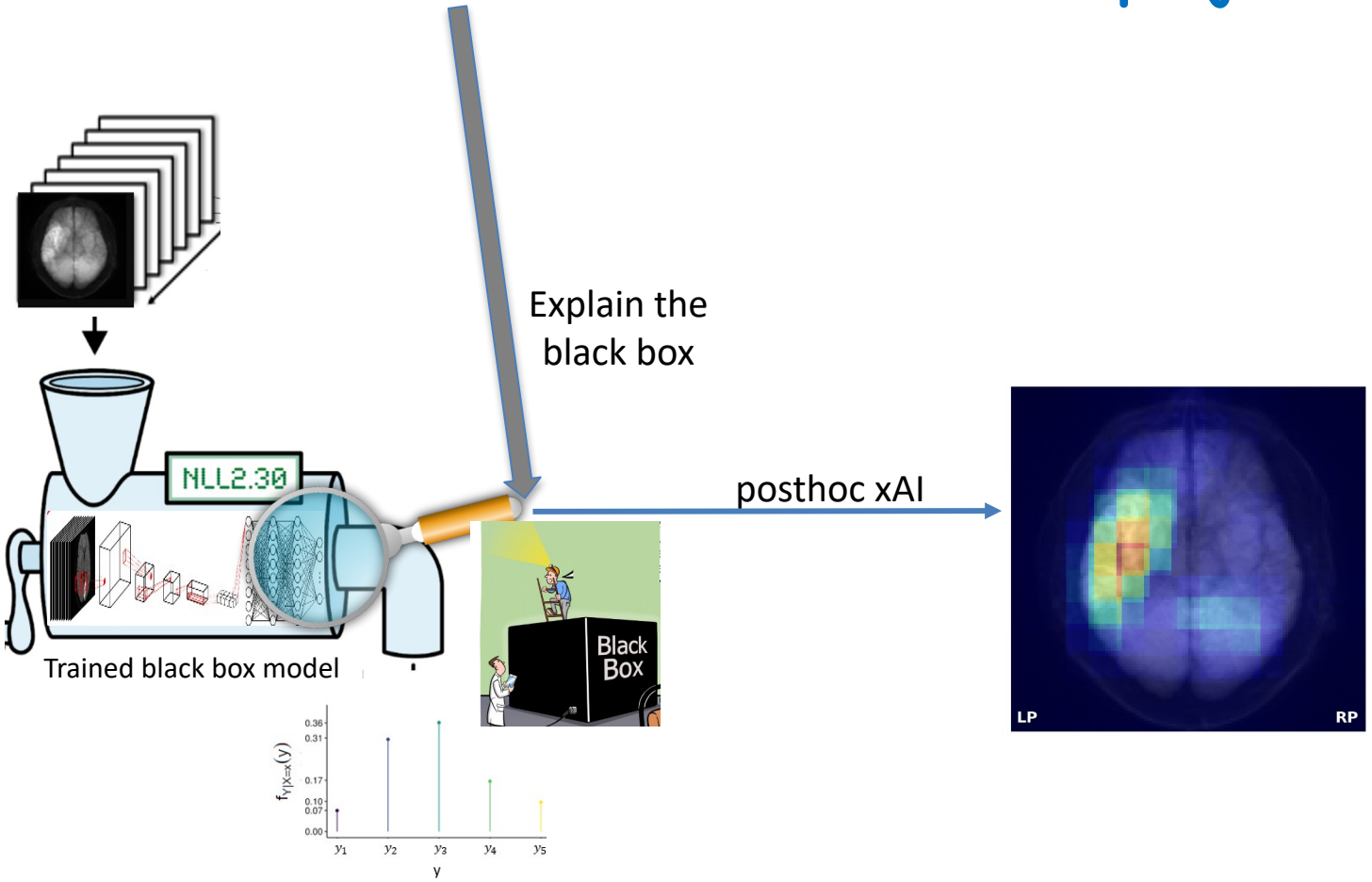
visualize important feature allows to judge the individual classification



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Tf, python code: https://github.com/marcotcr/lime -> image tutorial
https://github.com/marcotcr/lime/blob/master/doc/notebooks/Tutorial%20-%20Image%20Classification%20Keras.ipynb

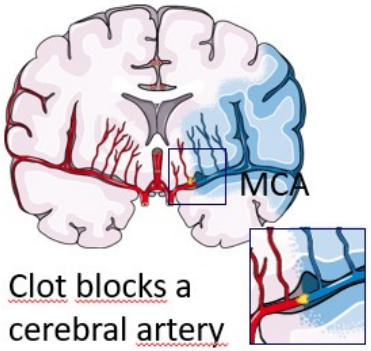
xAI in a medical research project



xAI delivers an «explanation» for the prediction by highlighting a region in the input image, that was identified (posthoc) to be responsible for the predichtion.

Stroke application: Motivation

Ischemic stroke (88%)



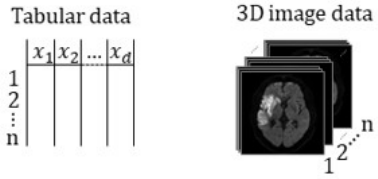
Clot blocks a cerebral artery

Large vessel occlusion stroke

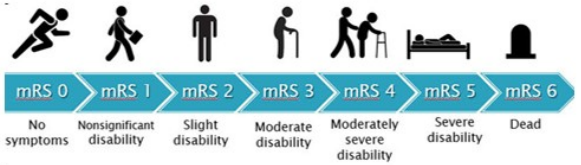
1.9 million neurons/min.

Time is brain!

Semi-structured patient data ($n = 407$)

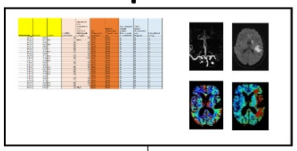


ordinal outcome

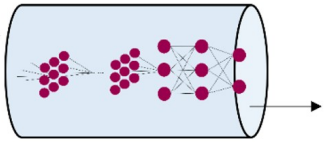


binary outcome

Input

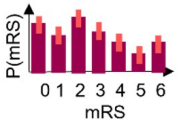


Model

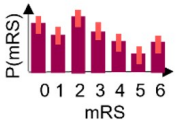


Output

Thrombectomy



Thrombolysis

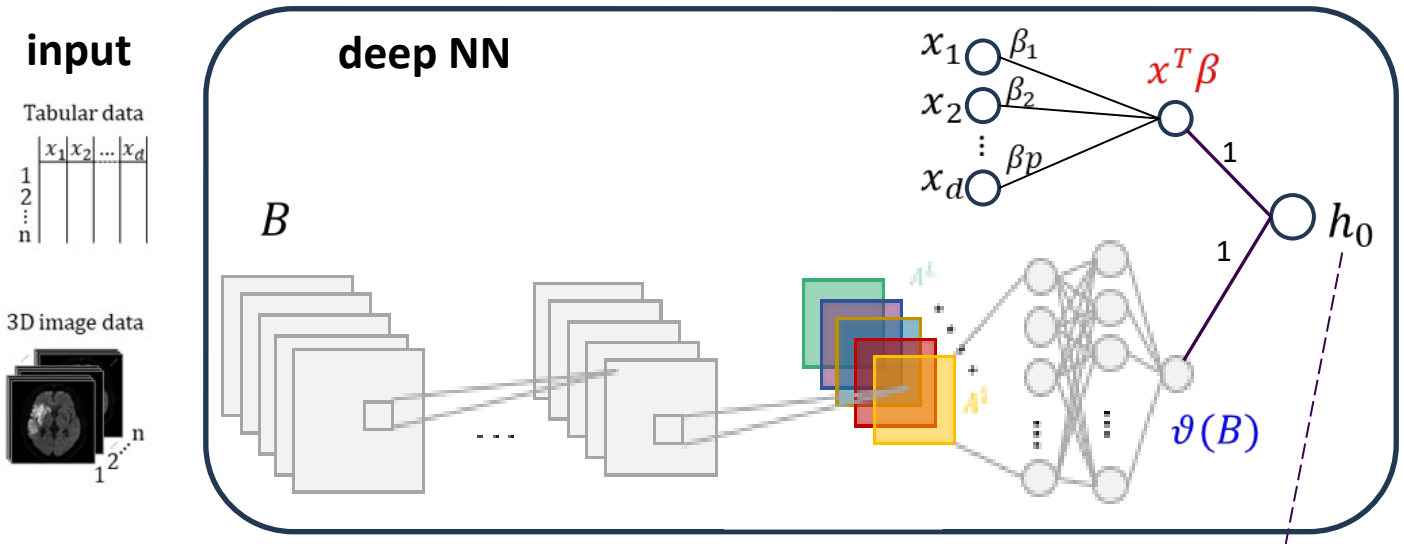


Treat or not?

Why?

Is it certain?

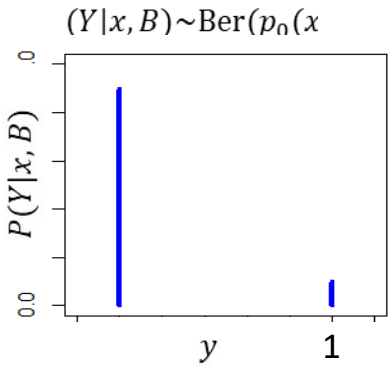
Architecture of a multi-modal NN for a binary Y

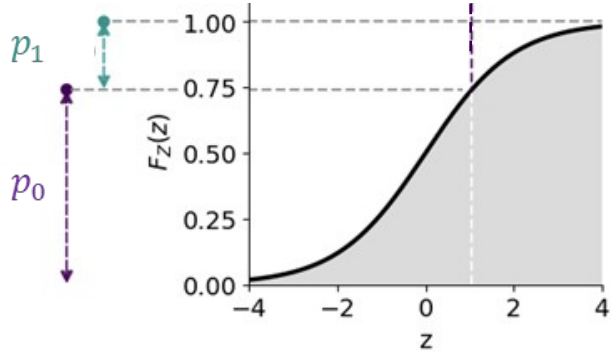


$p_0 = P(\text{favorable}|x, B) = \sigma(h_0) = 0.8$

$p_1 = 1 - p_0 = P(\text{unfavorable}|x, B) = 0.2$

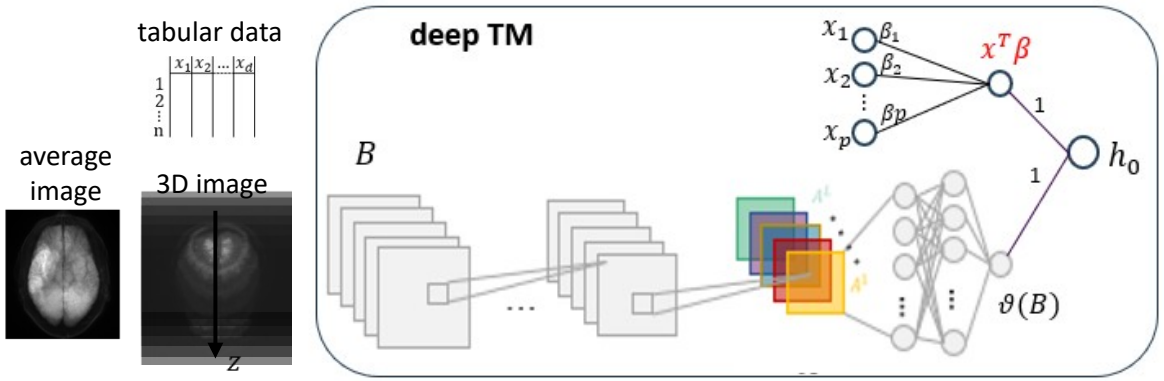
$z_0 = h_0 = h(Y = 0|B, x) = \vartheta(B) - x^T\beta$





Occlussion for multi-modal NN in stroke application

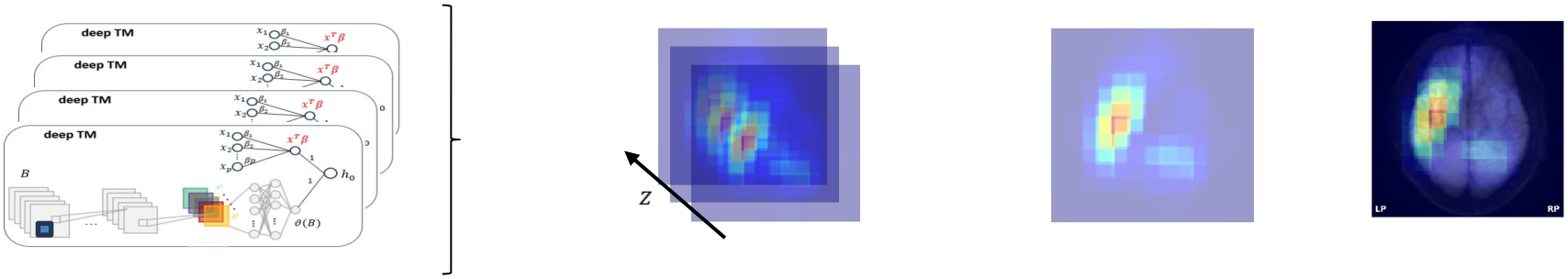
Trained deep TM



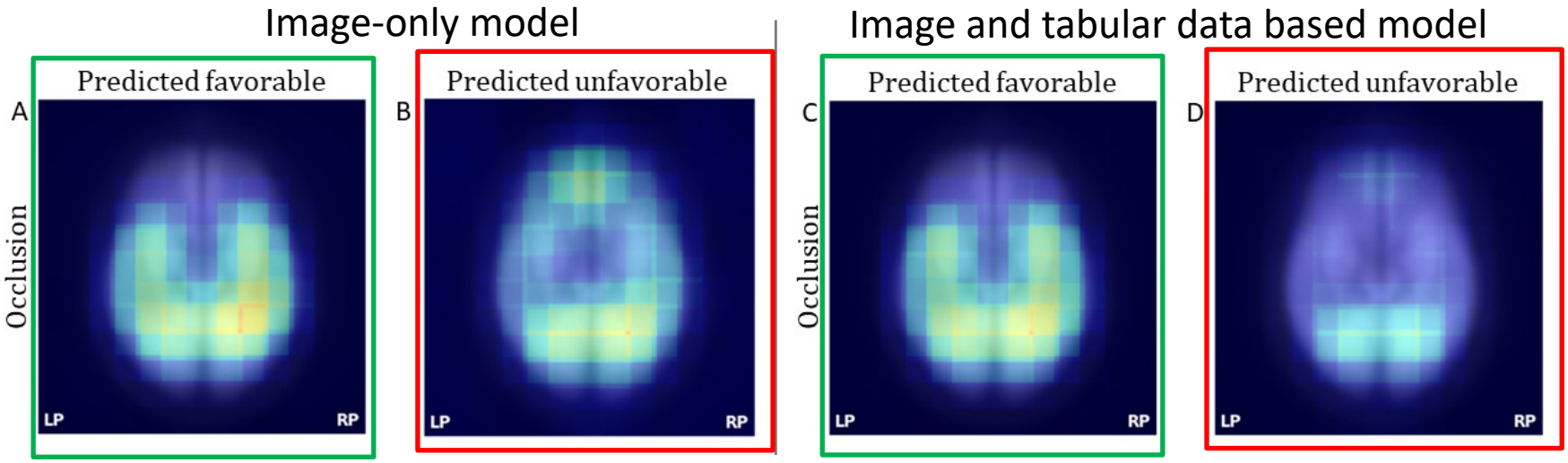
$$p_0 = F_z(h_0) = 0.2$$
$$\boldsymbol{p_1 = 1 - p_0 = 0.8}$$

Occlusion

Predict probabilities with different occluding masks → Determine reduction of probability of predicted class due to occlusion per voxel → Averaging along z-direction → Overlay



Explanation maps for patients predicted as un/favorable

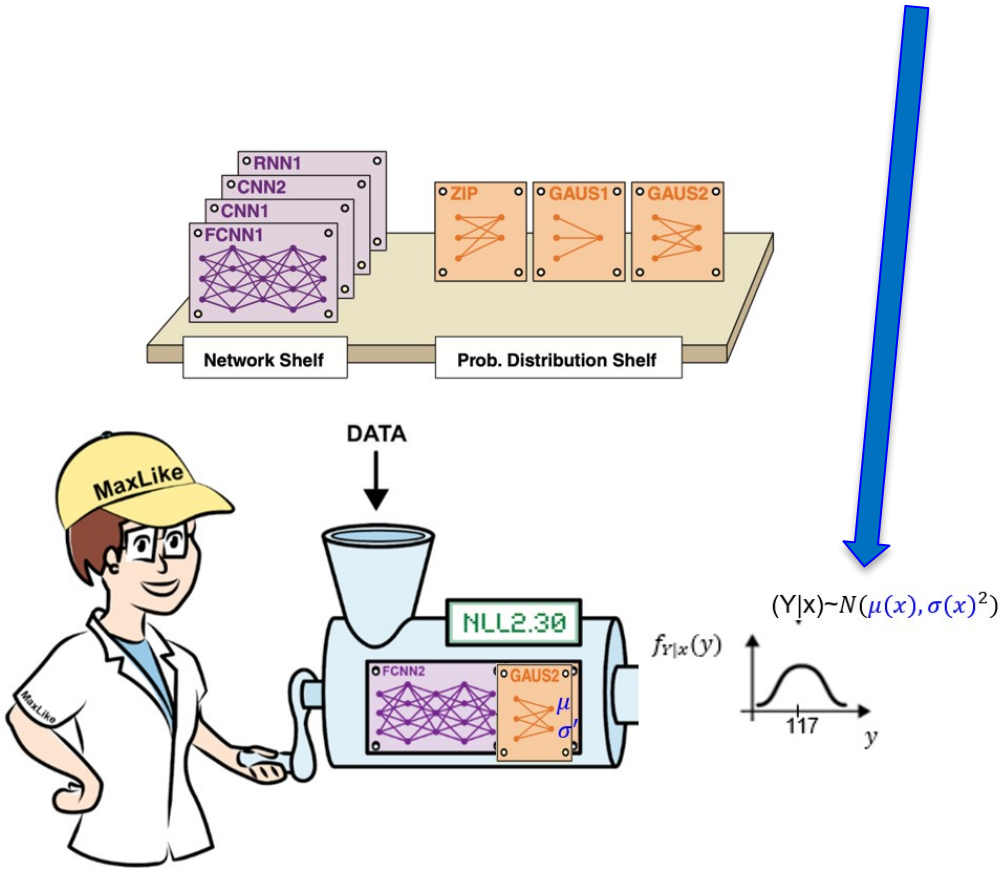


→ xAI for image only based model highlights frontal lobe if bad outcome is predicted

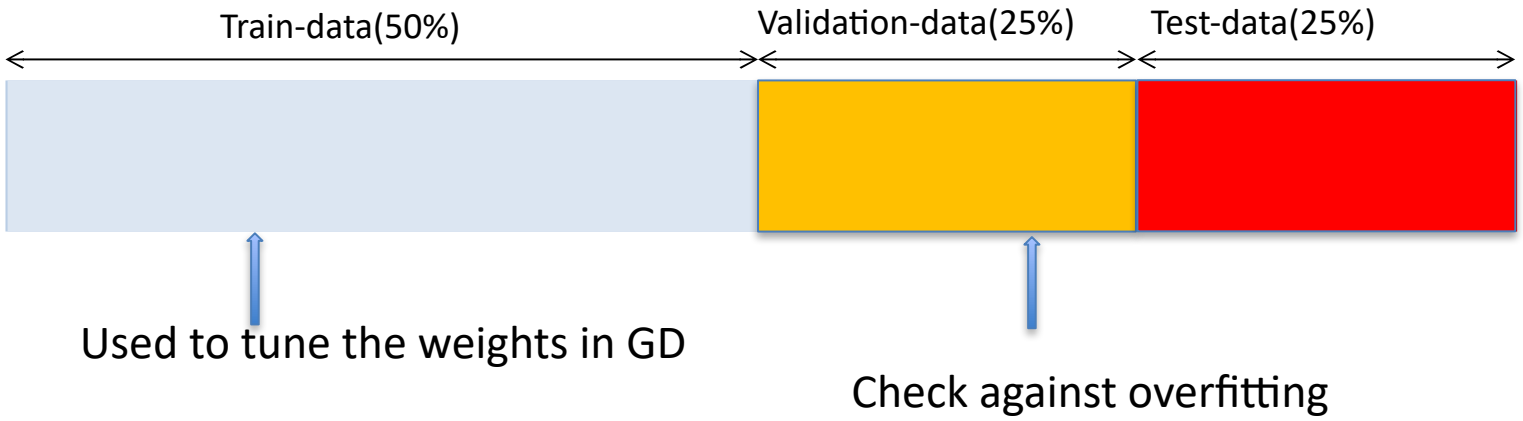
→ Neurologist know that the frontal lobe indicates the age of the patient

→ When providing age as tabular data in addition to the image to the data, the frontal lobe is not anymore hightlighted

Evaluating the quality of the probabilistic predictions



Model evaluation is done on Validation and Test Set



Best practice: Lock an extra test data set away, and use it only at the very end, to evaluate the chosen model, that performed best on your validation set.
Reason: **When trying many models, you probably overfit on the validation set**.

Determine performance metrics, such as NLL, Brier score, AUC or MSE, to evaluate the predictions **on new validation or test data**

Evaluation of probabilistic models

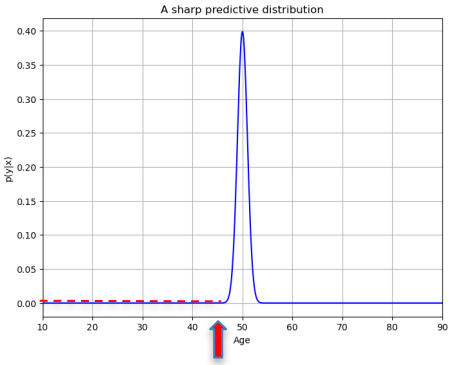
Which model is better / evaluating probabilistic models

Meryl Streep, 41y

Model 1

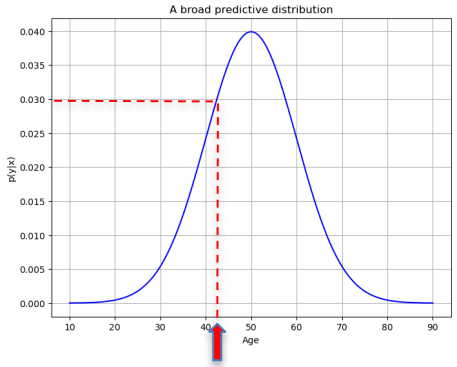
Model 2

Predicted age distribution



$p(y|x)$

$L_{\text{streep}} = 0.0001$



$p(y|x)$

$L_{\text{streep}} = 0.03$

Which model is better?

For the evaluation, we don't care if $p(y|x)$ is from a NN or from another model.
Evaluation of $p(y|x)$ has long tradition, e.g. in metrology (Good 1952).

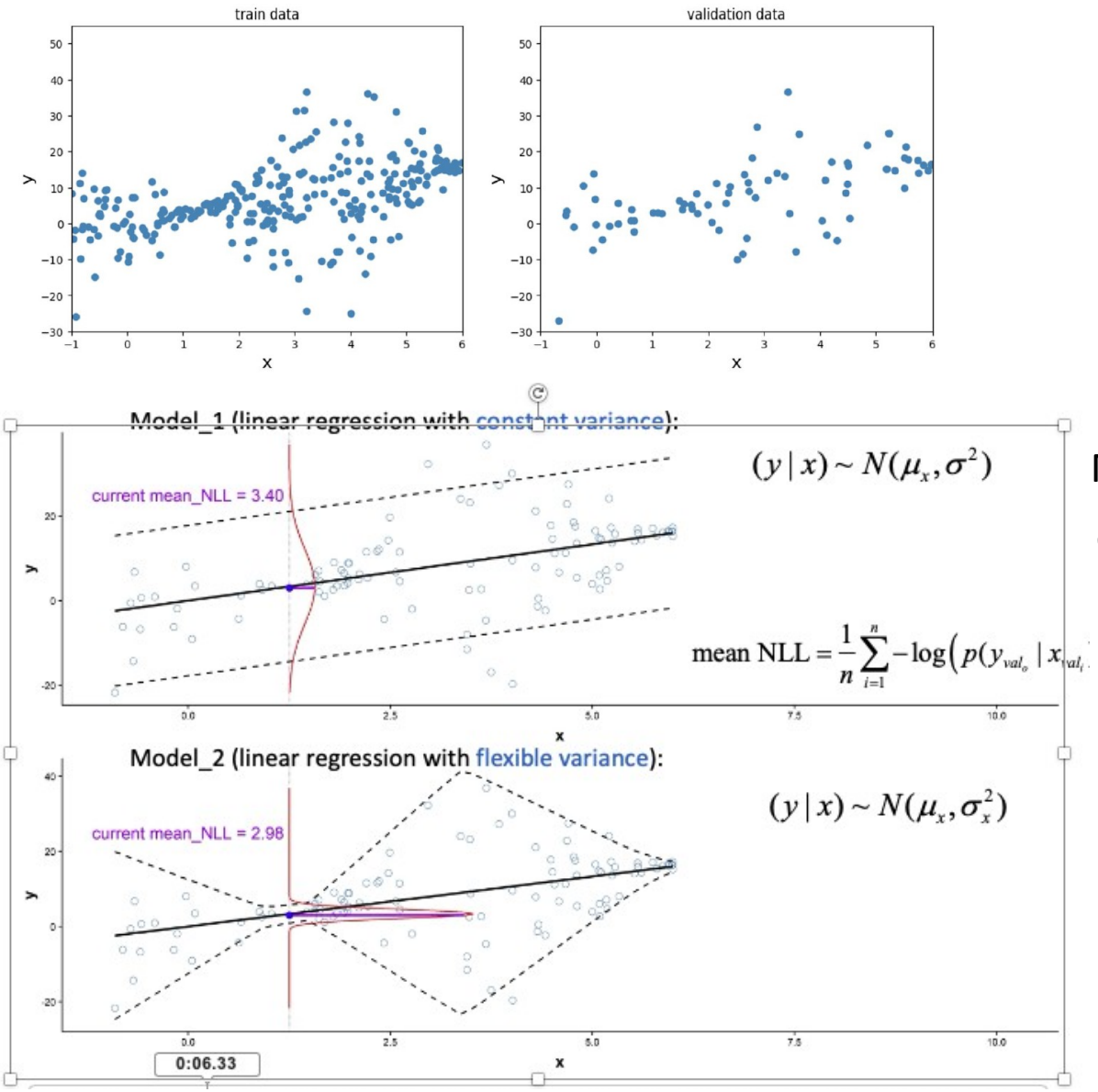
Is the test Likelihood or NLL suited for evaluation?

- The best measures for evaluation of a probabilistic model:
  - test **NLL** (negative-log-likelihood) for continuous and discrete outcomes
  - test log-likelihood (aka log-score) for continuous and discrete outcomes
  - … some few other "proper" scores you will see later

- Why?
  - **the objective in the training was to minimize the NLL**
    → the better the NLL the better the model!!
    Otherwise we should have optimized another measure ;)
  - The NLL (or log-score) is "strictly proper" and the only smooth and local score which gets only optimal if the predicted distribution matches the data generating distribution.

- Why not?
  - People like to see accuracy, AUC, MSE… → provide these measures additionally even though they are not "proper" and the model was not optimized for this purpose.

Which of the two probabilistic models is better?



Model 1 (constant variance)

$$(y \mid x) \sim N(\mu_x, \sigma^2)$$

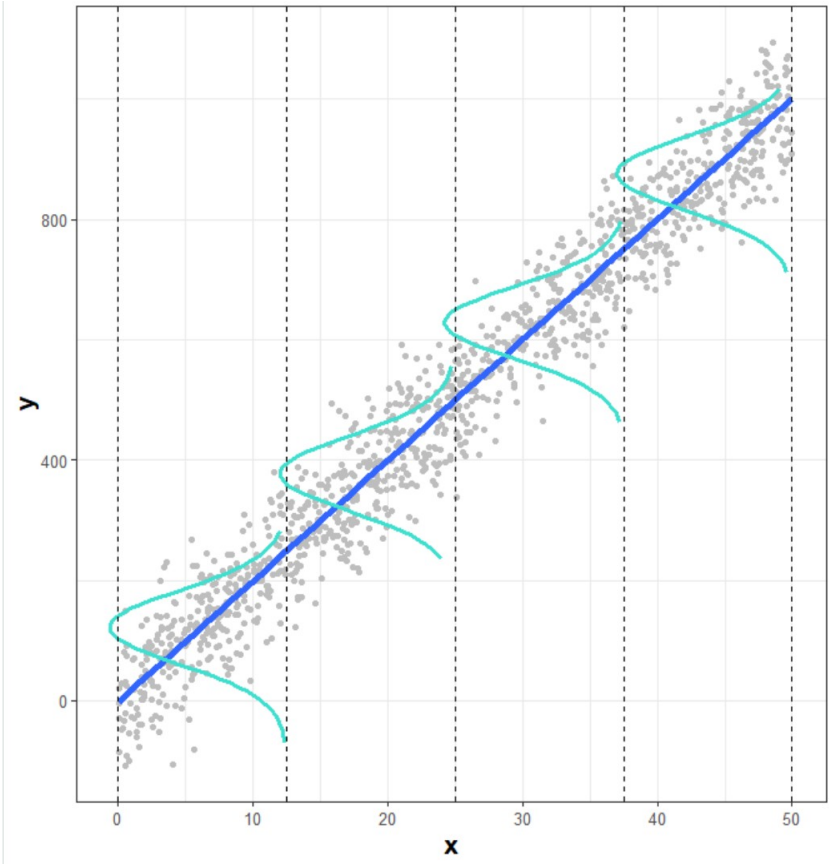
Model 2 (flexible variance)

$$(y \mid x) \sim N(\mu_x, \sigma_x^2)$$

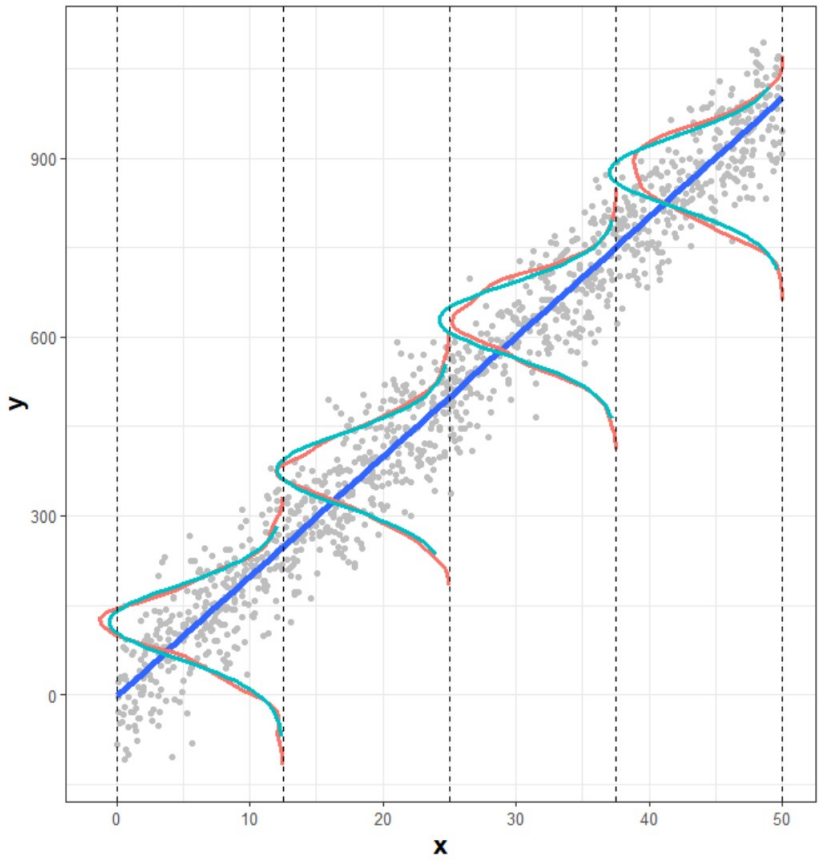
https://www.youtube.com/watch?v=QeWd0g_UzZY

Do predicted and observed outcome distribution match?

Validation data along with predicted outcome distribution (Gauss with const σ)



Validation data along with predicted and observed outcome distribution



A large validation data set is needed to ensure underlying assumption:
observed distribution = distribution of data generating distribution

Prominent Scores for binary classifiers

Definition 9.9 (Scoring rules for binary predictions) Let $Y \sim \mathrm{B}(\pi)$ be the predictive distribution for a binary event, i.e.

$$f(y) = \begin{cases} \pi & \text{for } y = 1, \\ 1 - \pi & \text{for } y = 0. \end{cases}$$

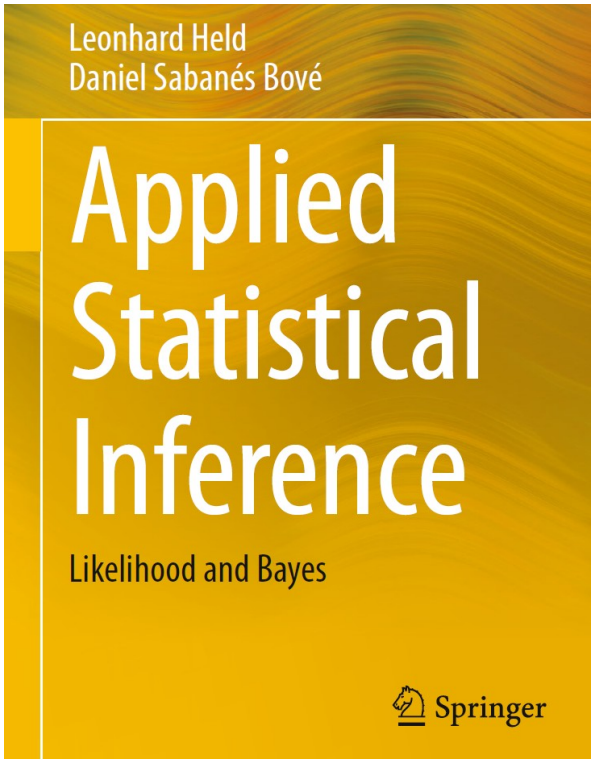
The *Brier score* BS, the *absolute score* AS and the *logarithmic score* LS are defined as

Strictly proper: $\mathrm{BS}\big(f(y), y_o\big) = (y_o - \pi)^2,$

Not proper: $\mathrm{AS}\big(f(y), y_o\big) = |y_o - \pi|$ and

Strictly proper: $\mathrm{LS}\big(f(y), y_o\big) = -\log f(y_o),$

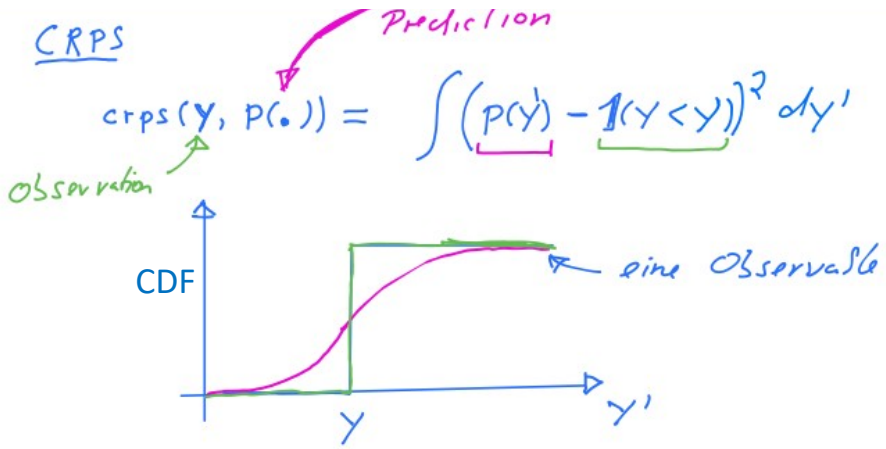
respectively.



Remark: For binary classification, the log score is not the only strictly proper score.

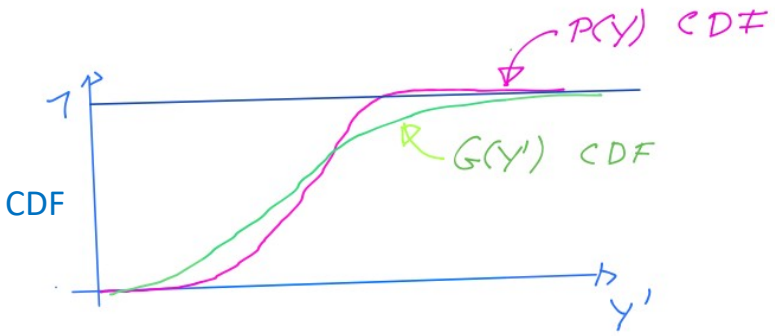
Other strictly (but non-local) proper scoring rules

In forecasting communities (power consumption / weather / …)
Continuous Ranked Probability Score is often used CPRS. $P(y')$ is CDF



Hard to compute (integral) in general

- Associated Score divergence (**expectations / many variables**)*
  - $D_{CPRS}(P, G) = \int \left(P(y) - G(y)\right)^2 dy$



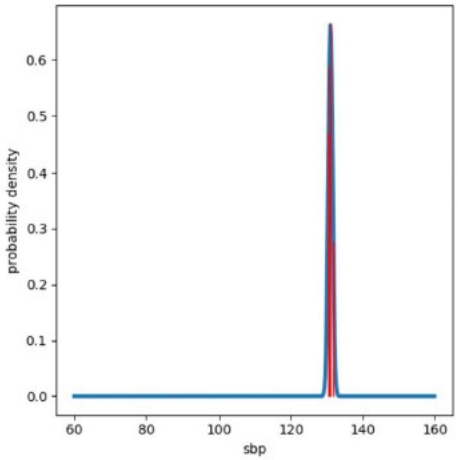
*See e.g. https://www.stat.berkeley.edu/~ryantibs/statlearn-s23/lectures/calibration.pdf for the calculation

Non-proper scoring rules

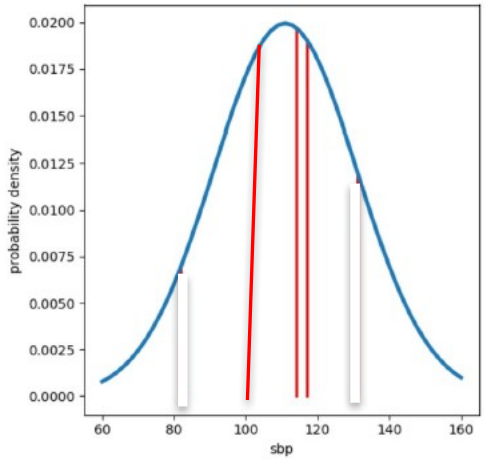
The following performance measures are very popular but not proper:

- Accuracy – for categorical outcome
- Kappa – for categorical outcome
- Area under the curve (AUC) - for binary outcome
- Mean Square Error (MSE) – for continuous outcome
- Mean Absolut Error (MAE) – for continuous outcome
- …

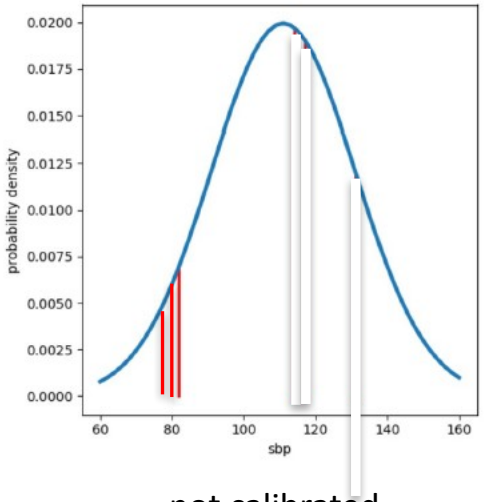
Calibration & Sharpness



calibrated
&
sharp



calibrated
&
not sharp

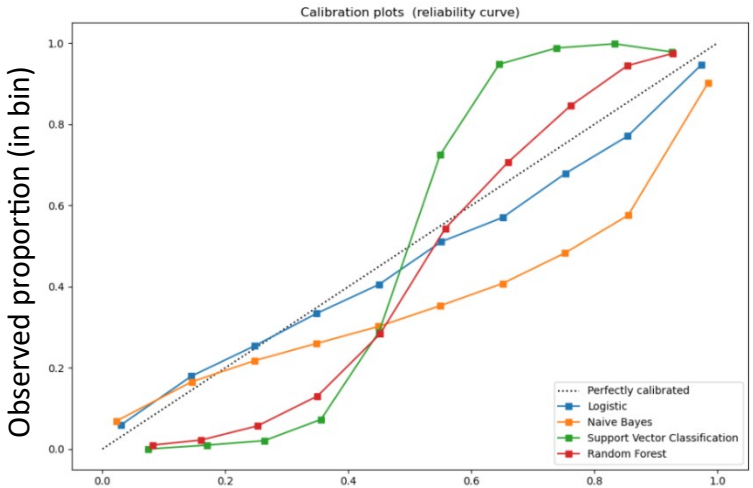


not calibrated
&
Not sharp

Calibration

- Consider the classification of tomorrows weather 
  - Y: Weather ∈ {Sunny, Mixed, Rainy}
- Example
  - Classifier $p_{\text{pred}}(Y = \text{sunny}) = 0.7$
  - In exactly 70% of those cases, it should be sunny
  - Check: Take all days with $p_{\text{pred}}(Y = \text{sunny}) = 0.7$ and check if on 70% of these days the weather was sunny. Do this which each probability bin.
- Calibration plot

$$\frac{\#(y_{\text{obs}} = \text{"rain"})}{\#y_{\text{obs}}}$$

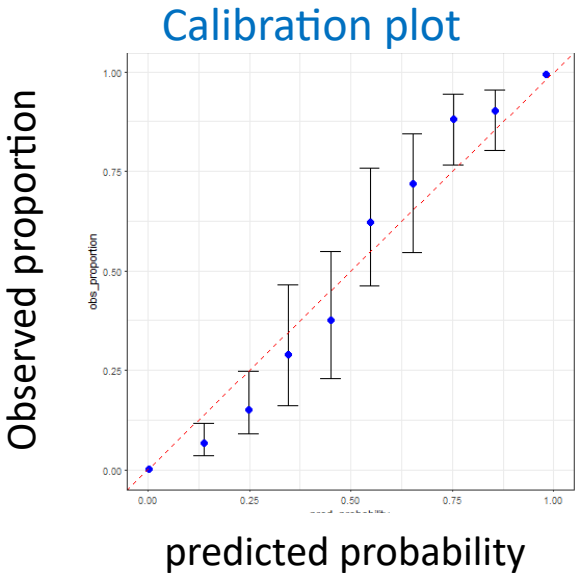


$p_{\text{pred}}(Y = \text{"rain"})$

Do NN produce calibrated probabilities?

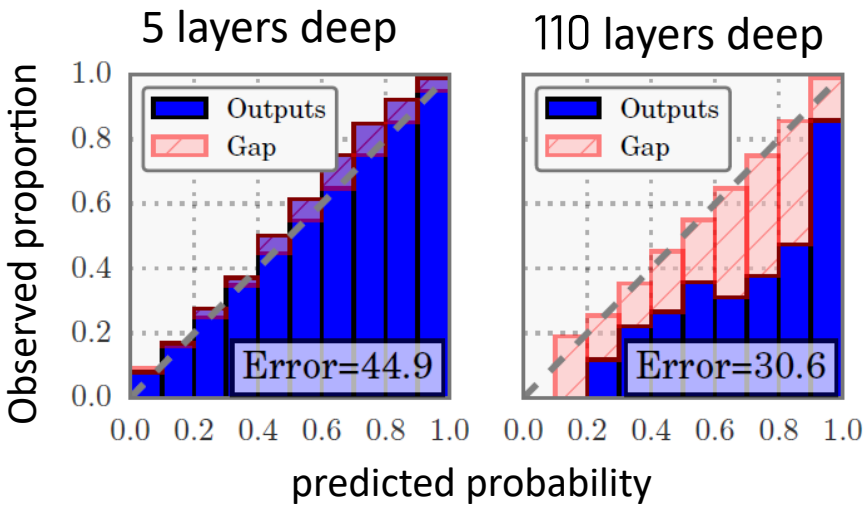
A probabilistic Classifier is calibrated, i.e. unbiased, if the predicted probabilities are on average correct and not systematically too small or too large.

To assess calibration one uses calibration plots.



Guo et al. (2017)
[On Calibration of Modern NN]

The deeper CNNs get
- the fewer miss-classifications
- the less well calibrated they get



Guo et all https://arxiv.org/pdf/1706.04599.pdf

Good news:
deep NN can be “recalibrated” and then we get calibrated probabilities.

Calibration vs Sharpness

- E.g. in Bern, the rain probability for the next day is 20% (over a year)
- A Classifier that predicts $p_{rain} = 0.2$ for each day is well calibrated
- A good classifier should make sharp classifications (predict probability of rain close to 0 for non-rainy days and close to 1 for rainy das) that is still calibrated (= unbiased, i.e. the predicted probabilities are not systematically too small or too large)

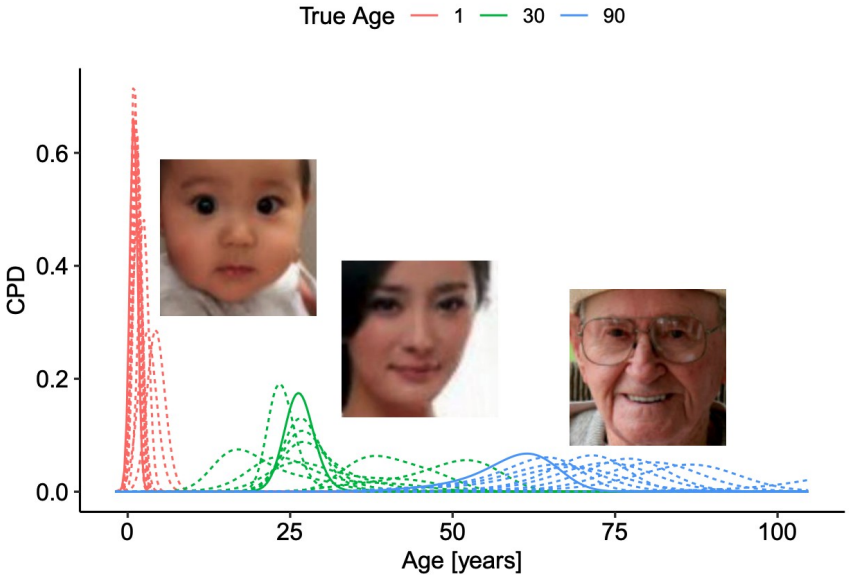
Example

	Good Classifier Prediction p_{rain}	Const Classifier Prediction p_{rain}	Actual Rain
Day 1	0.03	0.2	No
Day 2	0.05	0.2	No
Day 3	0.04	0.2	No
Day 4	0.03	0.2	No
Day 5	0.95	0.2	Yes

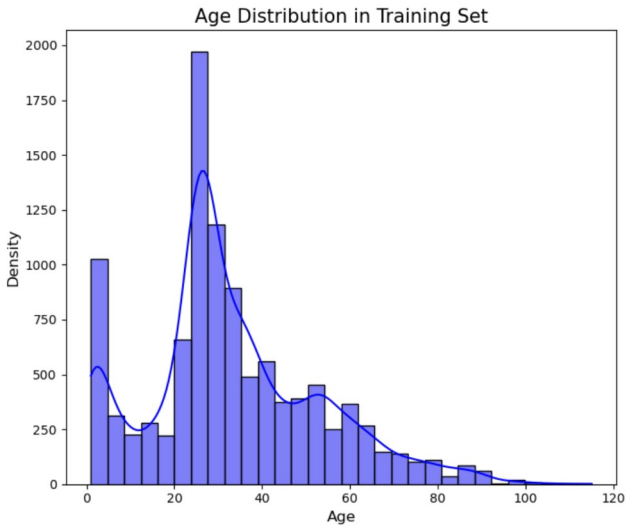
	Good Classifier Prediction p_{rain}	Const Classifier Prediction p_{rain}	Actual Rain
Day 1	0.03	0.2	No
Day 2	0.05	0.2	No
Day 3	0.04	0.2	No
Day 4	0.03	0.2	No
Day 5	0.95	0.2	Yes

	Actual Rain
Day 1	No
Day 2	No
Day 3	No
Day 4	No
Day 5	Yes

Calibration vs Sharpness $p(Y|x)$



Sharper predictor depends on x.
But is it calibrated?



A very uninformative but calibrated forecaster would just predict the population distribution (for every) x.

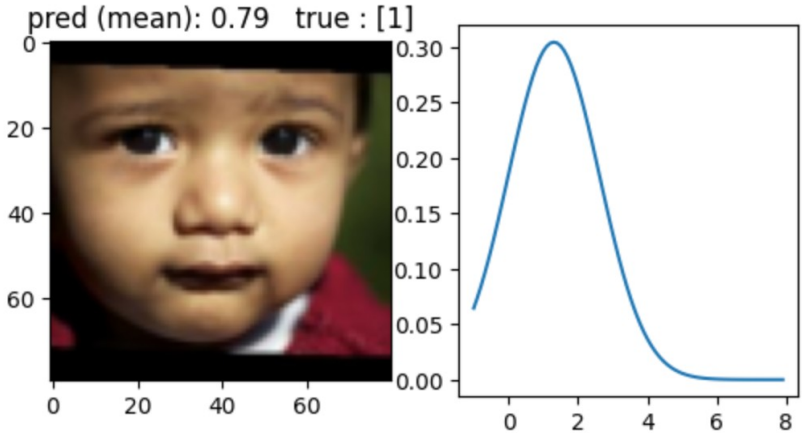
Is there a score that takes into account calibration and sharpness?

Yes, there is: the **NLL** ☺

Exercise Probabilitic Age Prediction

Notebook: 03

15_faces_regression_keras_torch.ipynb



NLL and MAE on the test set

```
# Since the NLL is also the loss function, we can also use the evaluate function of keras.
```

NLL as general cure-all in probabilistic modeling

- Maximize likelihood ↔ minimize negative log-likelihood (NLL)
- The log-score (NLL) is strictly proper score for regression.
- The log-score (NLL) is also strictly proper for classification models.
- To train a probabilistic model: minimize NLL!
- To evaluate or compare probabilistic models: use the validation NLL!

Benchmark your model

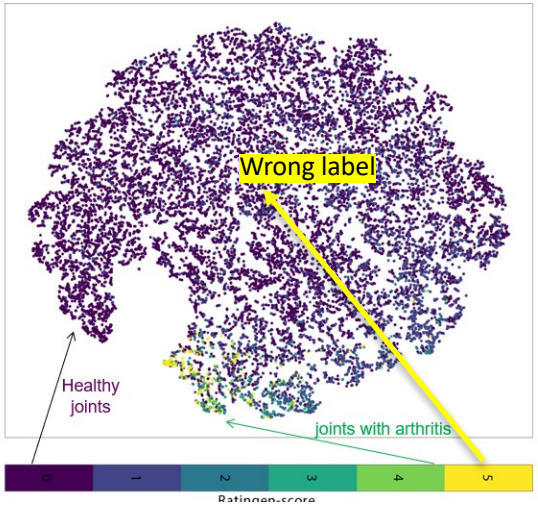
- Apply your model to benchmark data sets for which the performance of other SoA models are known from literature

- Apply beside your model also baseline models to your data and compare the performances
  - Null model that predicts always the train marginal outcome distribution
  - Random Forest (or other simple shallow model)
    - On raw input features (e.g. image pixel values)
    - On constructed features (e.g. extracted from pretrained model)

Evaluate your model and perform an error analysis

- Check test performance (test NLL, AUC, MSE)
- Check calibration (calibration and residual plots)
- Visualize/investigate wrongly classified instance
- Check grouping of wrongly classified instance
- Check for labeling errors
- Perform xAI, check interpretability
- Check intermediate results in debugging mode

Not cute?
Why?
Cage?



outcome