

BBS: Ordinal Transformation Models

Ordinal Neural Network Transformation Models: Deep and interpretable regression models for ordinal outcomes

Lucas Kook^{1,2*}, Lisa Herzog^{1,2*}, Torsten Hothorn¹, Oliver Dürr³, Beate Sick^{1,2†}

¹ University of Zurich, Switzerland

² Zurich University of Applied Sciences, Switzerland

³ Konstanz University of Applied Sciences, Germany

Abstract

Outcomes with a natural order commonly occur in prediction tasks and oftentimes the available input data are a mixture of complex data, like images, and tabular predictors. Although deep Learning (DL) methods have shown outstanding performance on image classification, most models treat ordered outcomes as unordered and lack interpretability. In contrast, classical ordinal regression models yield interpretable predictor effects but are limited to tabular input data. Here, we present the highly modular class of ordinal neural network transformation models (ONTRAMS). Transformation models use a parametric transformation function and a simple distribution to trade off flexibility and interpretability of individual model components. In ONTRAMS, this trade-off is achieved by additively decomposing the transformation function into terms for the tabular and image data using a set of jointly trained neural networks. We show that the most flexible ONTRAMS achieve on-par performance with DL classifiers while outperforming them in training speed. We discuss how to interpret components of ONTRAMS in general and in the case of correlated tabular and image data. Taken together, ONTRAMS join benefits of DL and distributional regression to create interpretable prediction models for ordinal outcomes.

- Goal: Combine Deep Learning with statistical models for interpretation
- Here: logistic- and ordinal regression
- <https://arxiv.org/abs/2010.08376>

Presented by Oliver

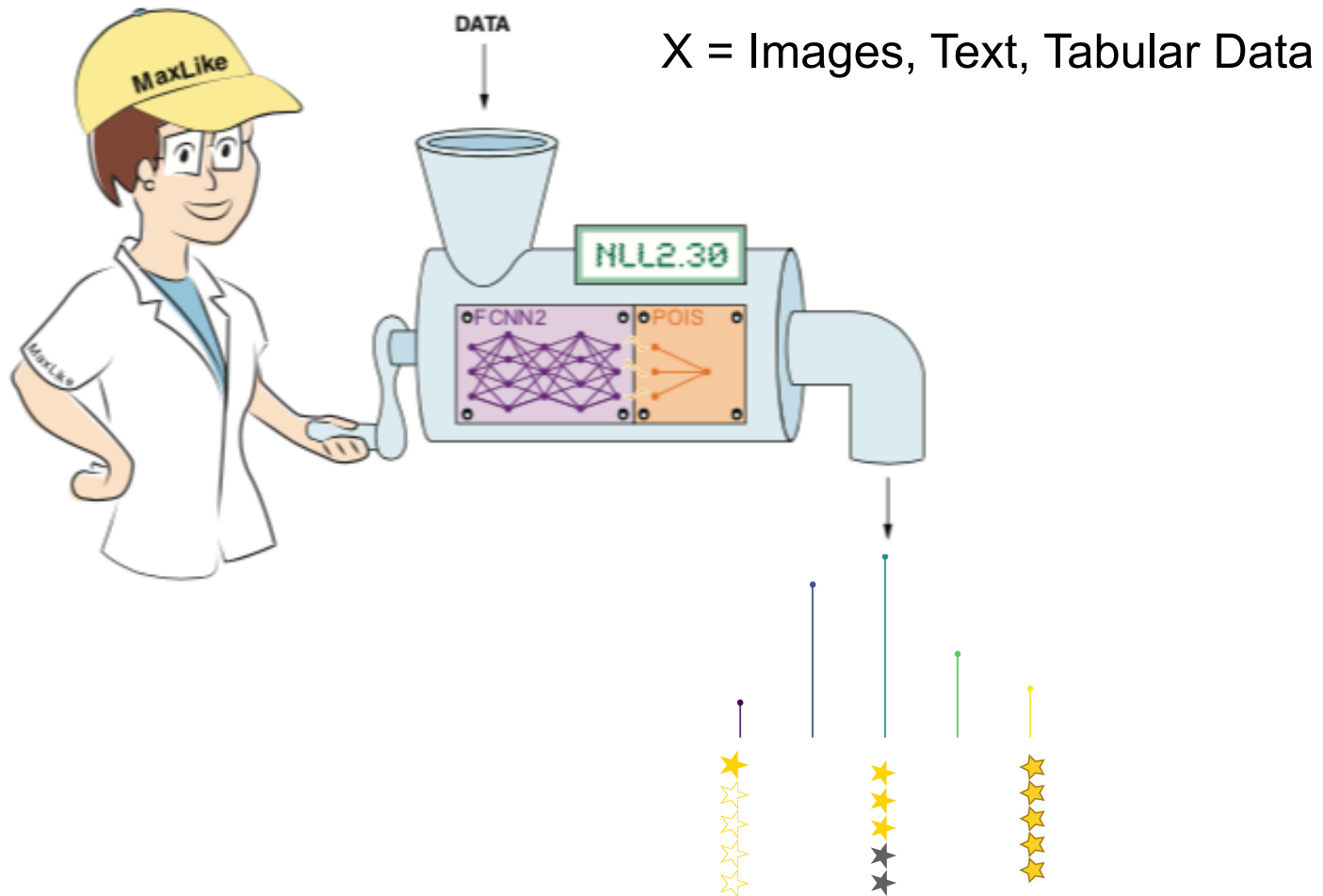
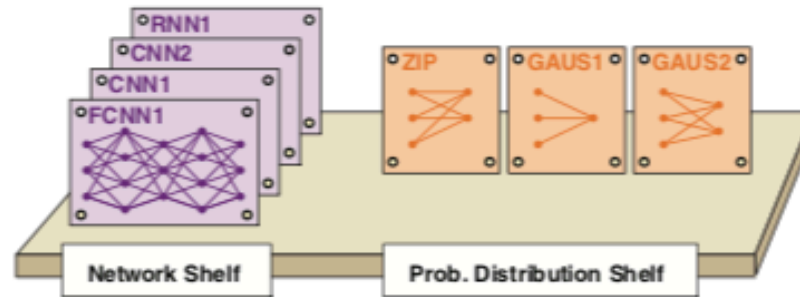
Ordinal Data (Examples)

- Amazon Ratings ★★★★★
- Medical ratings
 - Reopatiy Score
 - Neuroscore
- Wine quality (score between 0 and 10)
 - UCI-Data* set (N=4898) from (Cortez et. al. 2009) with 11 covariates like:
 - fixed acidity, residual sugar, sulphates, ..., alcohol
- Age group (UKTFace)
 - Images as high dimensional features



«Less than numerical data, more than just classes / categories»

Modeling Distributions with Neural Networks



Aim: Interpretation

Use NN for complex data (e.g. images) combined with a statistical model which allows for interpretable coefficients

Today

- Look again at logistic regression as latent variable model
- Interpretable Ordinal Regression model
- Interpretable Ordinal Regression Models with Neural Networks
- Formulation as transformation model

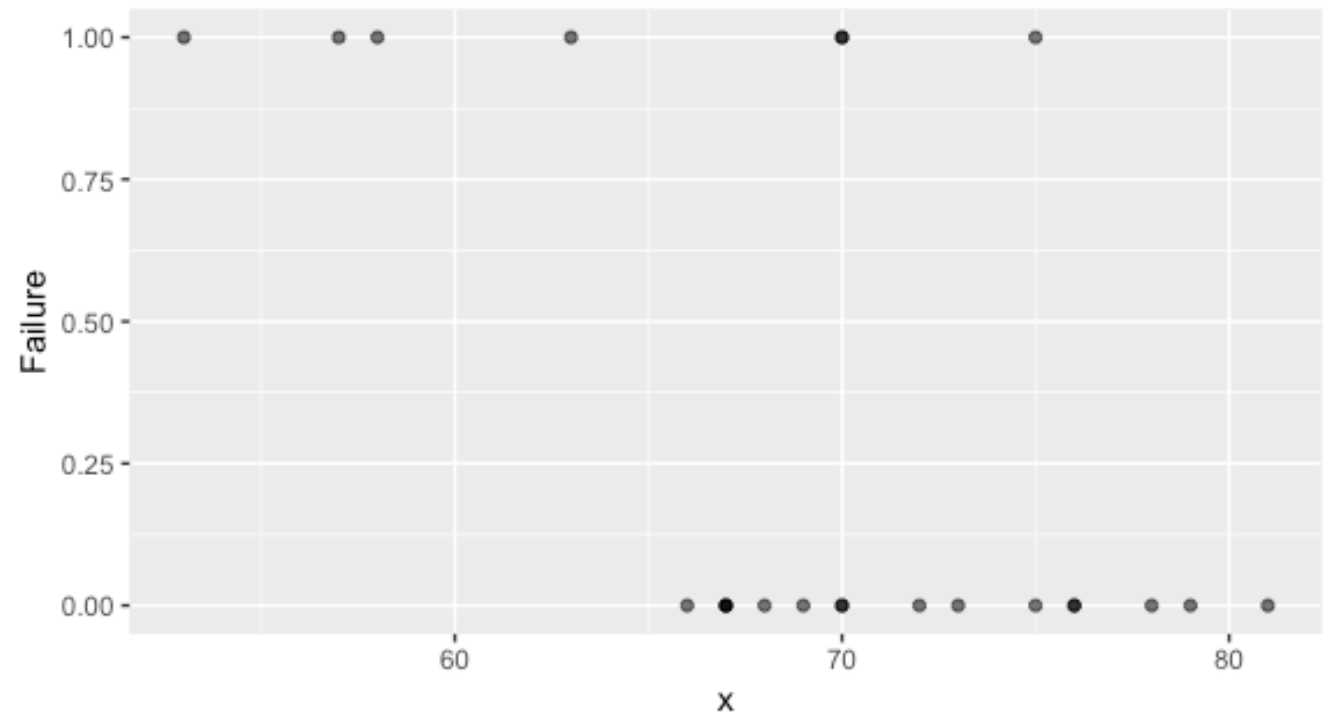
Logistic regression as latent variable model

Recap: Modelling with logistic regression

Zero / one classification

Want: $p(x) = Pr(Y = 1|x)$

Prob. for a O-ring to be defect $Y=1$ at a given temperature X

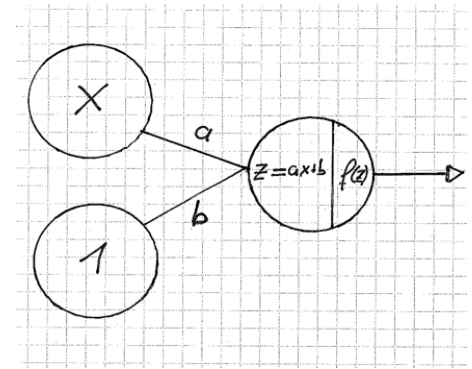


Question:

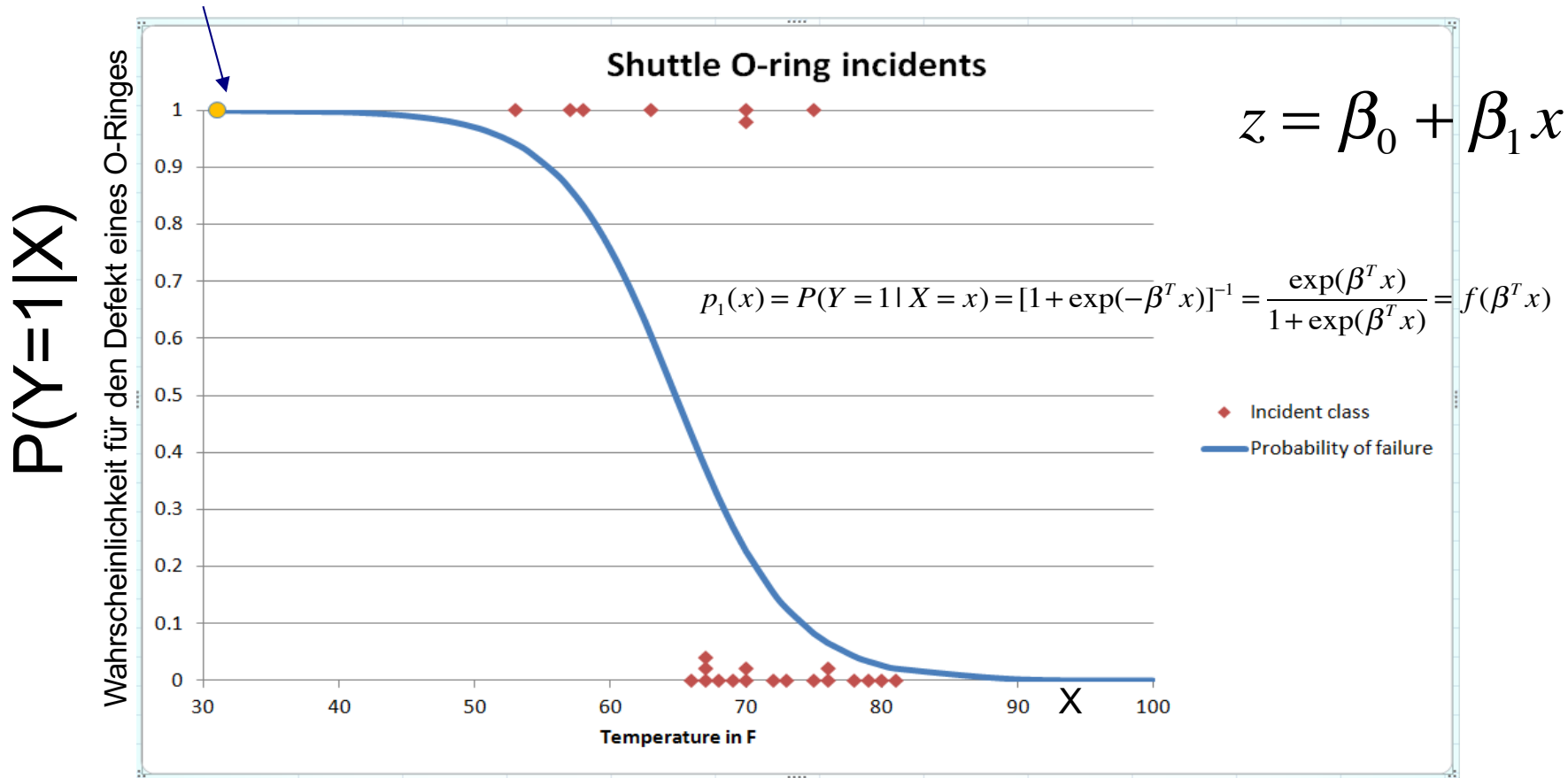
- Guess curve?
- Why is $p(X) = b + a X$ (linear regression) wrong?

Recap: Logistic Regression

Predict if O-Ring is broken, depending on temperature



Challenger launch @31 F
 Prob. of a failure=0.9997




Determination of the parameters with MaxLike Principle

Interpretation with odds

- Probability for event

- $p(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}}$
- $odds(x) := \frac{p(x)}{1 - p(x)} = e^{\beta_0 + \beta_1 x} \in [-\infty, +\infty]$
- $\log(odds(x)) = \beta_0 + \beta_1 x$



win.comparator		ODDS		RESULTS	
MENU		OFFERS	BETTING SITES	FOOTBALL	
ROGER FEDERER		2 - 1		RAFAEL NADAL	
		ODDS			
		1X2			
FULL-TIME		1	2	Bonus	
bet365	1.83	1.83		Bonus	
1XBET	1.86	1.92		\$100	
betway	1.8	2		£30	
388	1.8	1.95		£30	
bwin	1.8	1.95		£10	
betway MILL	1.8	1.91		Bonus	
bet-at-home	1.78	1.94		£20	
UNIBET	1.8	1.95		£40	
BETFRED	1.8	2		£30	

<https://www.wincomparator.com/roger-federer-id2930-rafael-nadal-id2905/>

- Interpretation (odds):

- Examples

- Odds : „Prob for winning“ / „Prob for not winning“
 - Federer against Nadal 2:1 Two times more likely that Federe wins
 - Horse Clever Hans wins 3:1 Three times more likely that Hans wins (vs. not winning)
- Odds : „Probability of broken o-ring“ vs. Non-broken
 - 8:1

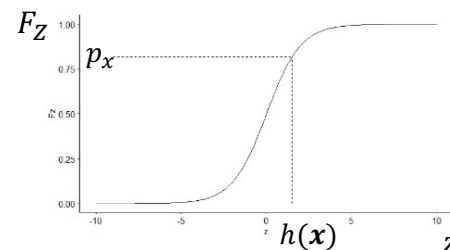
Odds ratios

- Q: How does the odds change with x
- Example Tennis (Federer against Nadal)
 - $x = 0$ (not grass) $odds(x = 0) = \exp(\beta_0) = 1.2$
 - $x = 1$ (grass) $odds(x = 1) = \exp(\beta_0 + \beta_1 \cdot 1) = 2$
 - Odds Ratio: $OR_{x=0 \rightarrow x=1} = \frac{odds(x=1)}{odds(x=0)} = \exp(\beta_1 \cdot 1) = 2/1.2$
 - $\log(OR_{x=0 \rightarrow x=1}) = \beta_1 = 2/1.2$
- Works also for continuous x important is just Δx (not absolute value)

More than one variable

Model for cut-point:

$$\log(\text{odds}(x)) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$$



The **coefficient β_k** as the **log-odds-ratio** for $Y = 1$ when comparing a situation where x_k is increases by 1 unit (while fixing all other variables) with the situation before increasing x_k

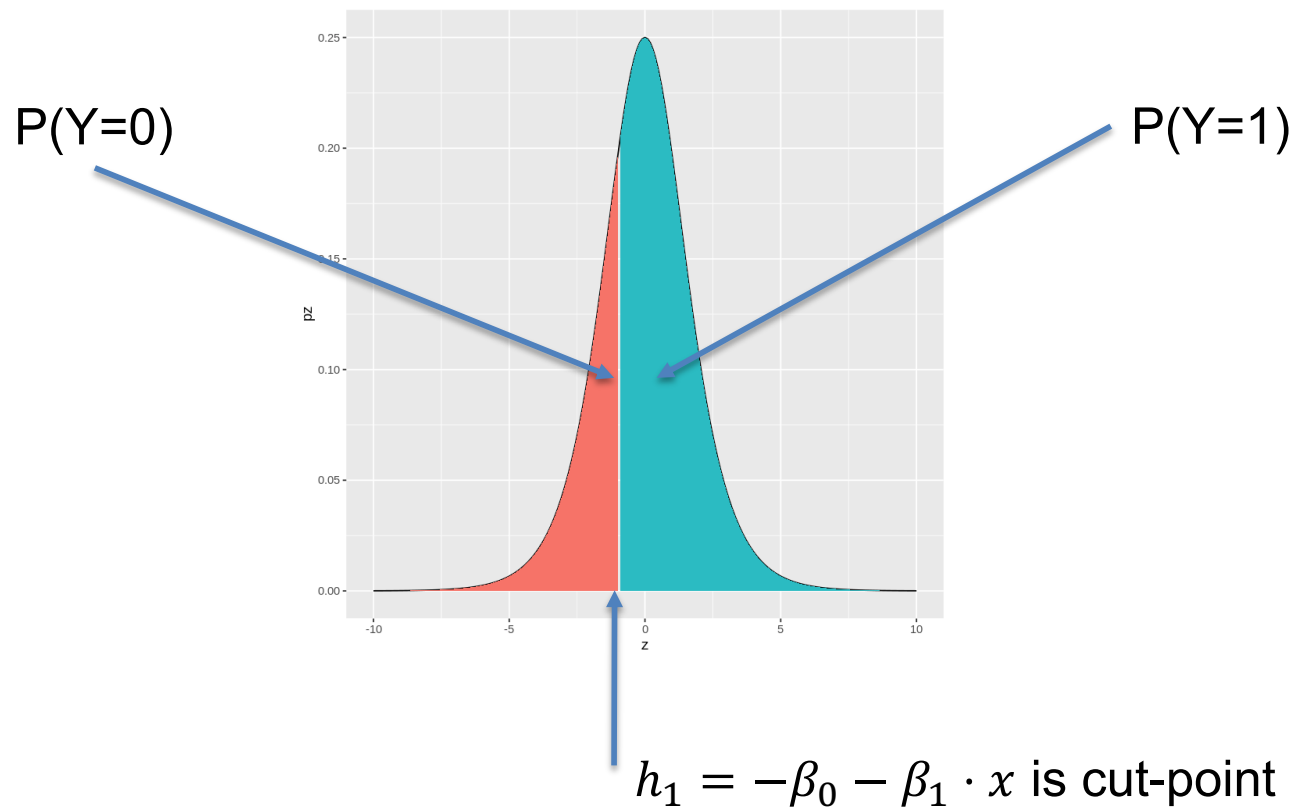
$$\log(\text{OR}_k) = \log\left(\frac{\text{odds}(x_1, \dots, x_k+1, \dots, x_p)}{\text{odds}(x_1, \dots, x_k, \dots, x_p)}\right) = \log\left(\frac{e^{\beta_0} \cdot e^{\beta_1 x_1} \cdot \dots \cdot e^{\beta_k(x_k+1)} \cdot \dots \cdot e^{\beta_p x_p}}{e^{\beta_0} \cdot e^{\beta_1 x_1} \cdot \dots \cdot e^{\beta_k x_k} \cdot \dots \cdot e^{\beta_p x_p}}\right) = \log(e^{\beta_k}) = \beta_k$$

$$\Rightarrow e^{\beta_k} = \text{OR}_{x_k \rightarrow x_k+1}$$

Logistic Regression as latent variable model

Idea:

A continuous latent (unobserved) variable z determines the probability to observe $Y = 1$

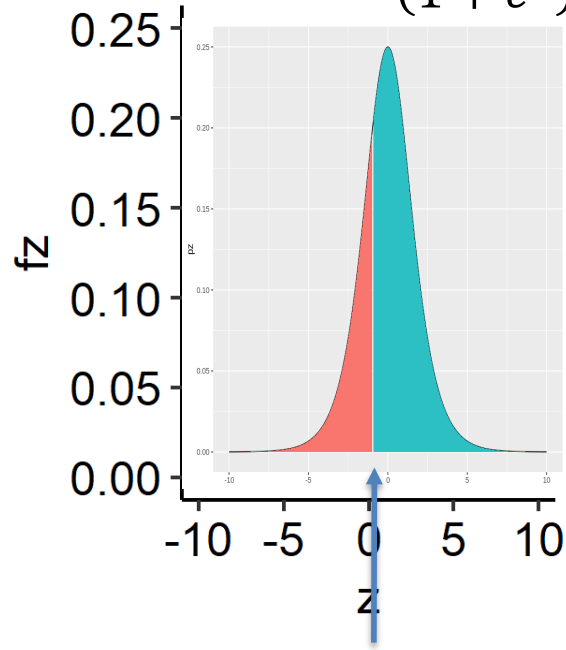


If the distribution is the logistic distribution, then this latent variable model is the same logistic regression.

Logistic Distribution

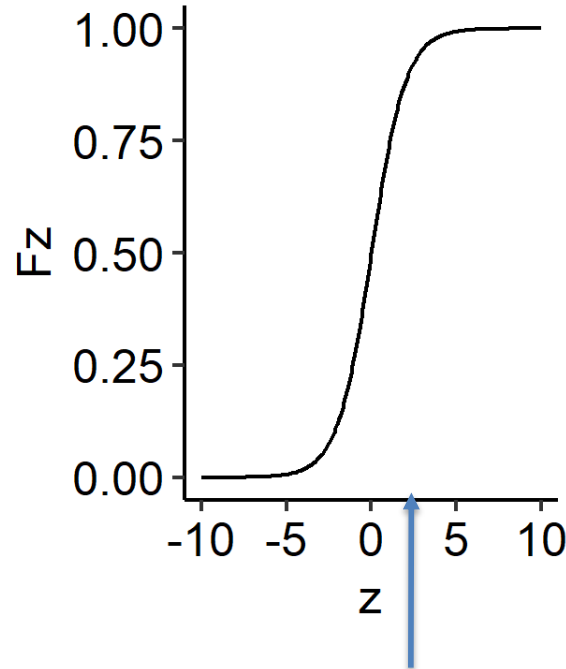
PDF (f_z)

$$f_z(z) = \frac{e^z}{(1 + e^z)^2}$$



CDF (F_Z)

$$F_Z(z) = \frac{1}{1 + e^{-z}}$$



$h_1 = -\beta_0 - \beta_1 \cdot x$ is cut-point

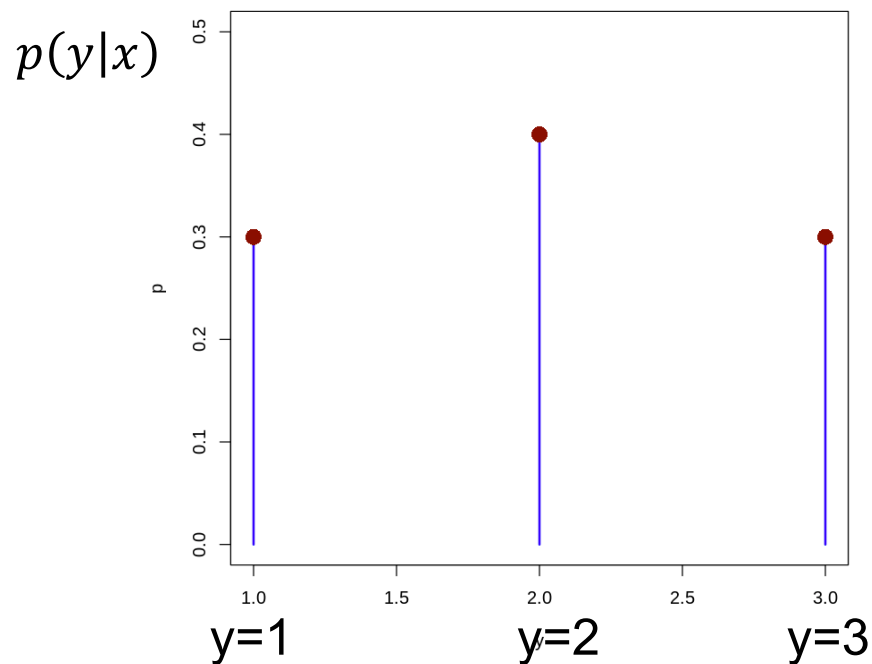
$$p_1(x) = 1 - F_Z(h_1) = 1 - \frac{1}{1 + e^{-h_1}} = \frac{1 + e^{-h_1} - 1}{1 + e^{-h_1}} = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}} = \sigma(\beta_0 + \beta_1 x)$$

Same as logistic regression

Ordinal regression (more than two levels)

Goal of ordinal regression

- Get the conditional probability distribution CPD for a given x .
 - $p(y|x)$
- Simple example
 - x alcohol content in wine
 - $y=1,2,3$ corresponds to bad, medium, good



Distribution for fixed x

Proportional odds-model*

- Latent variable model can be extended for more than 2 levels



Cut points move same amount fixed β .

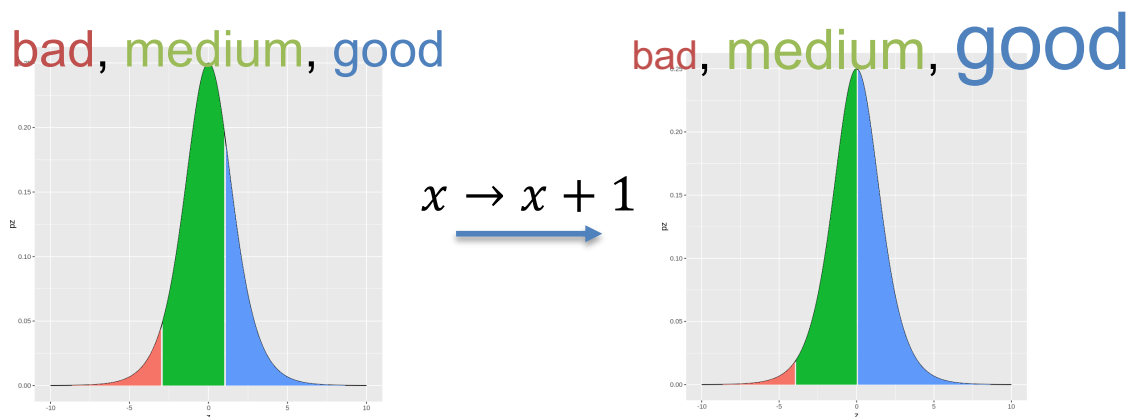
More than a single x
 $x \cdot \beta \rightarrow x^T \beta$

$$h_1 = \vartheta_1 - x \cdot \beta \quad h_2 = \vartheta_2 - x \cdot \beta$$

- In general cut-points modeled via: $h_i = \vartheta_i - x \cdot \beta$

Proportional odds model (interpretation)

- Since we have more levels, odds are now defined as
 - $odds(Y > y_k | x) = \frac{P(Y > y_k | x)}{P(Y \leq y_k | x)}$
 - Example $k = 2$ odds prob for good / prob less than good
- Changes in odds with x again with odds ratio
 - $OR_{x \rightarrow x+1} = \frac{odds(Y > y_k | x+1)}{odds(Y > y_k | x)} = e^\beta$ (not depending on k)
- Example $x = alcohol$

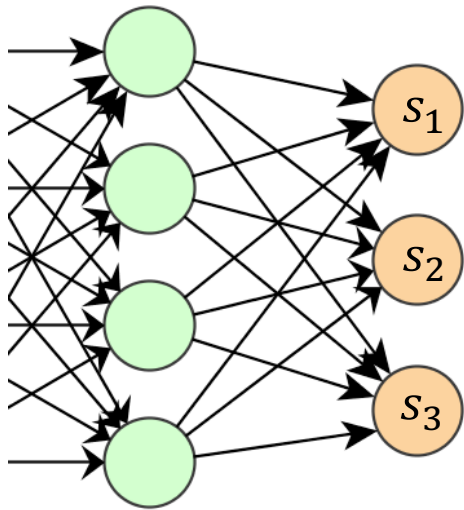


- **Questions:** Does alcohol make the wine taste better?
- What is β in $h_i = \vartheta_i - x \cdot \beta$

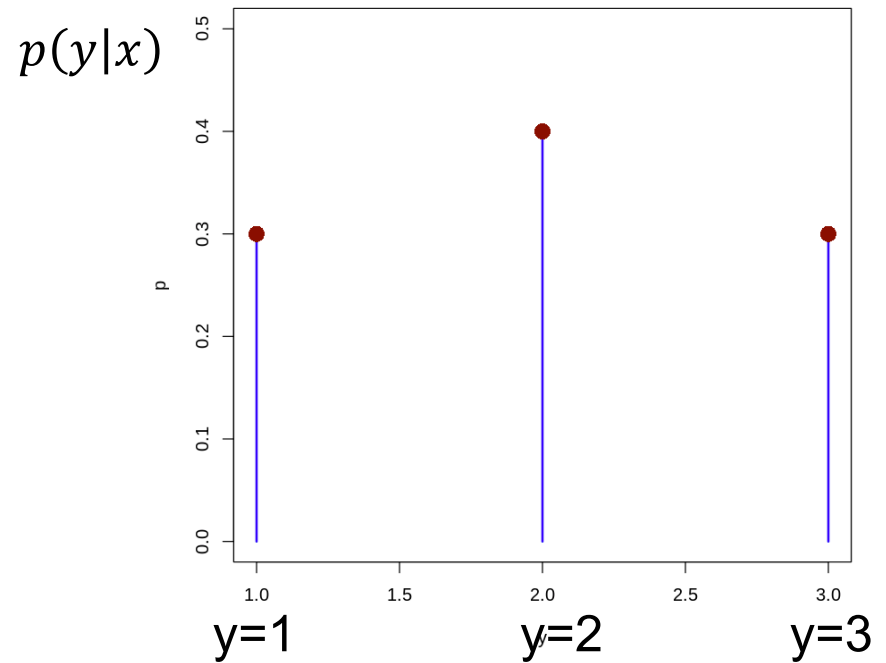
Modeling ordinal data with neural networks

Multi-Class classification (MCC)

- Classical deep learner “X-Entropy”



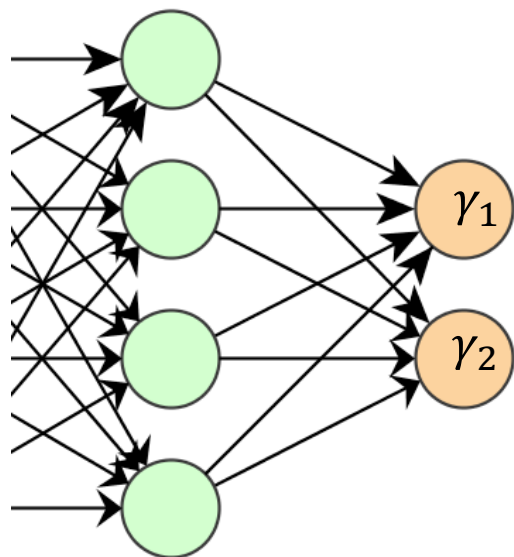
$$p_k = p(y|x) = \text{softmax}(s_k(x)) \propto e^{s_k(x)}$$



$$\text{cross-entropy} = - \sum_{i=1}^n \sum_{k=1}^K y_{ki} \cdot \log(P(Y = y_{ki}|x_i)) = - \sum_{i=1}^n \log L_i$$

Proportional odds modeling

- Output of the network controls cut-points

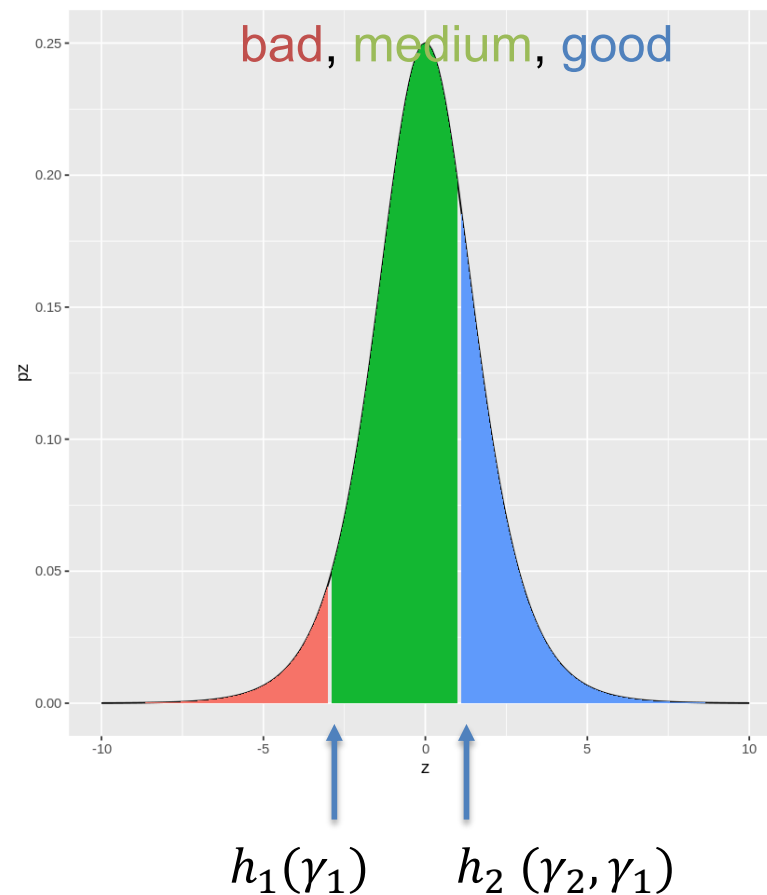


Cut-points need to be ordered

$$h_1 = \gamma_1$$
$$h_2 = h_1 + e^{\gamma_2}$$

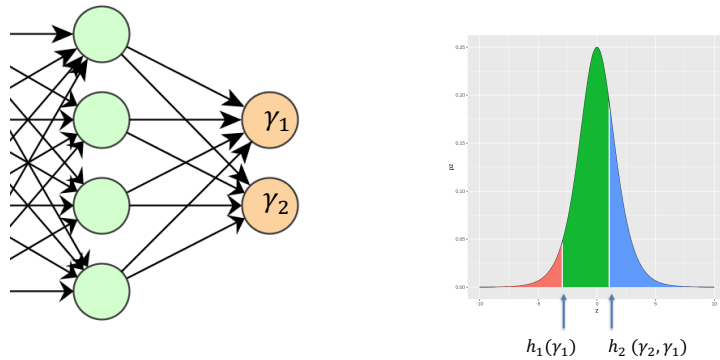
In general:

$$h_k = h_1 + \sum_{i=2}^k e^{\gamma_i}$$

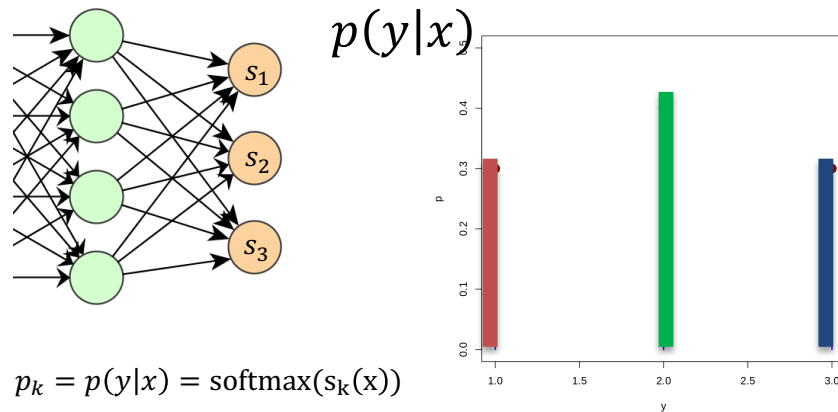


Training with NLL

- Training data $(y^{(i)}, x^{(i)})$, with say level $y^{(i)}=2$



NLL contribution: *green area*



NLL contribution: *green bar*

With a flexible NN, both models can produce the same likelihood.

Learning speed

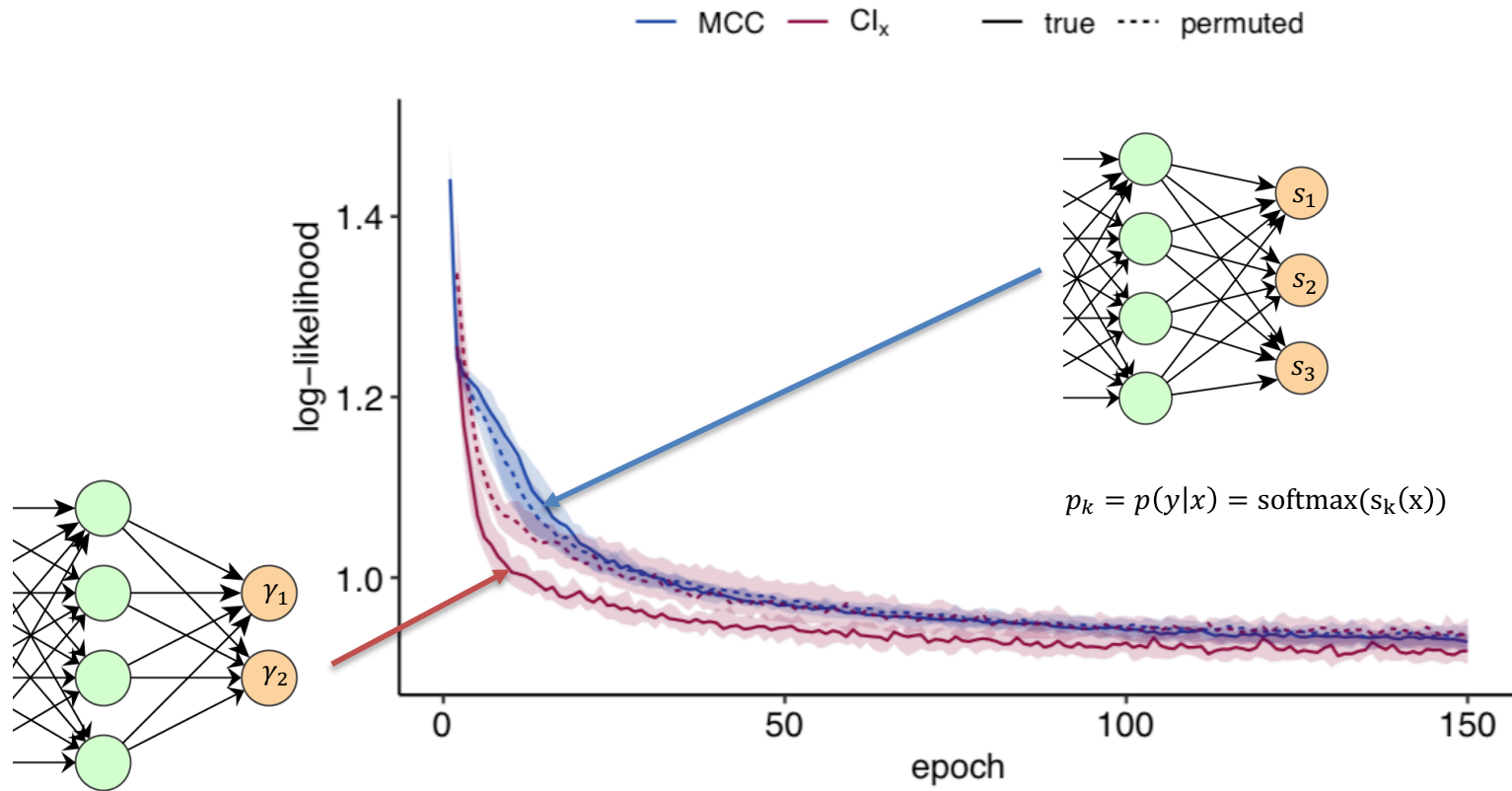
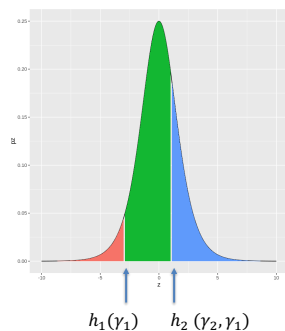


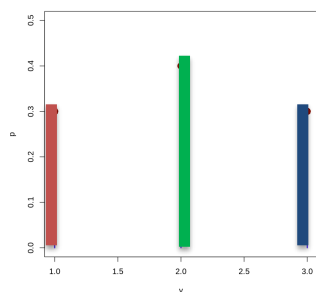
Figure A1: Test learning curves comparing a MCC against a complex intercept model (CI_x) for the true and a permuted ordering of the classes in the wine quality dataset. Depicted are the median test losses (thick line) together with the empirical 20th and 80th percentile (shaded regions) over 20 runs.

A temptation

- NLL is *local*, i.e. only the observed (“true”) class enters the NLL.



$$NLL := -\frac{1}{n} \sum_{i=1}^n \ell(h; y_i, \mathbf{x}_i, \mathbf{B}_i) = -\frac{1}{n} \sum_{i=1}^n \log (F_Z(h(y_{ki} | \mathbf{x}_i, \mathbf{B}_i)) - F_Z(h(y_{(k-1)i} | \mathbf{x}_i, \mathbf{B}_i)))$$

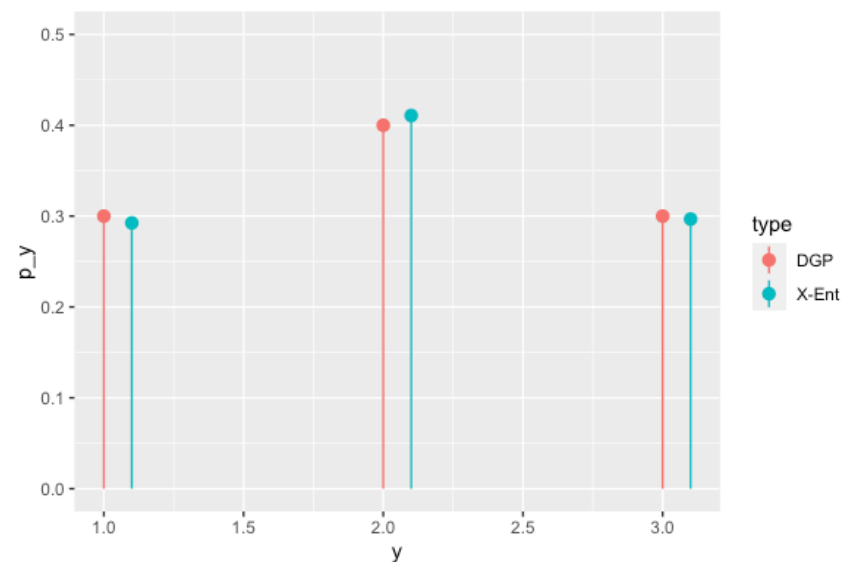


$$NLL = -\frac{1}{N} \sum \log(p(y_i | x_i))$$

- Say, we have ten classes. True class is 5
- Prediction 6 is as bad as 10 for NLL!
- Is the fair?
- Quadratic Weighted Kappa Loss (QWK)
 - De La Torre et al., 2018

Simple example

- Data Generating Process (DGP)
 - Data all $x=1$ (unconditional)
 - $y=1,2,3$ with probability $p_y=c(0.3,0.4,0.3)$
- Network (FC-NN with some hidden layers)
- QWK does not yield* DGP
 - This is a consequence that QWK is not a proper score



Use loss functions which are proper scoring rules, if you want to reproduce the DGP.

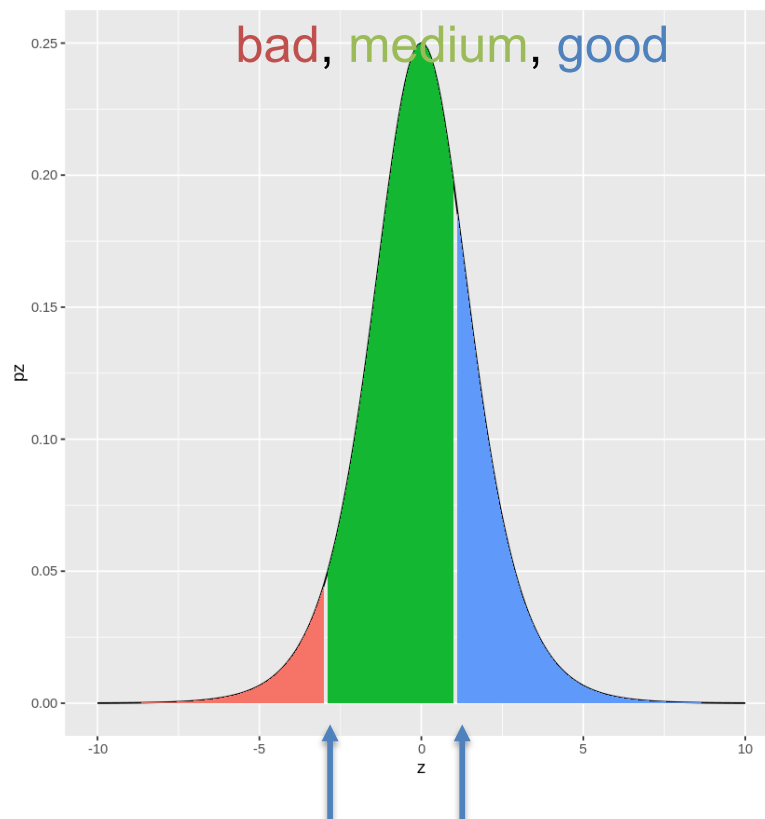
Standard Deep Learning Classification and the ONTRAM models do reproduce DGP

*A distribution of 0.1, 0.8, 0.1 has lower QWK-loss

Intermediate Summary

- Shoehorning the loss function, yield non-probabilistic models
- Introduced neural ordinal regression models, yield similar likelihood compared to standard deep learning approach (maybe a bit better training speed)
- So why use them at all?

Mixing flexibility with interpretability



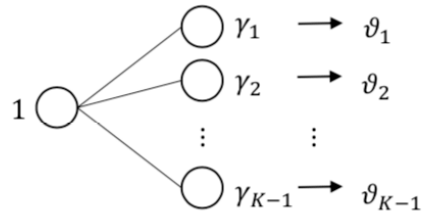
$$h_1 = \vartheta_1(B) - x \cdot \beta$$

$$h_2 = \vartheta_2(B) - x \cdot \beta$$

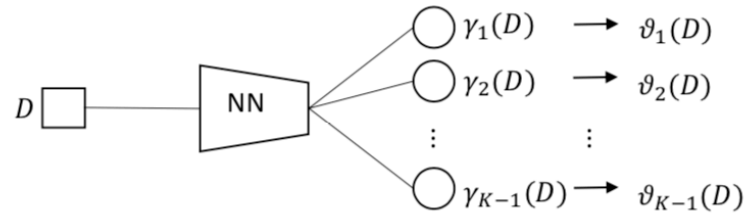
The main contribution of ONTRAM is to combine tabular interpretable data x (like #number of cigarettes) with complex data B like images (chest X-ray) to predict ordered y (the severity of cancer after 3 month)

Additive components to the cut-points

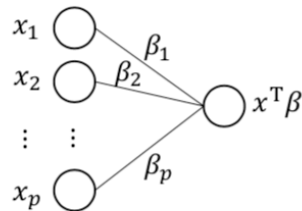
Simple intercept: ϑ



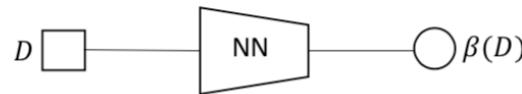
Complex intercept: $\vartheta(D)$, $D \in \{x, B\}$



Linear shift: $x^T \beta$



Complex shift: $\beta(D)$, $D \in \{x, B\}$



Shift of the cutpoints h_k

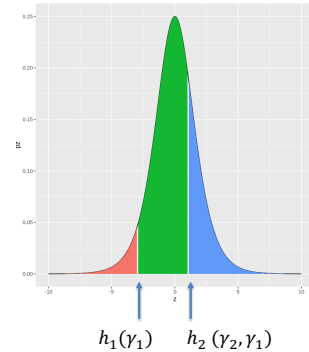
Complex intercept	CI_B	$\vartheta_k(B)$
Complex intercept + tabular	CI_B-LS_x	$\vartheta_k(B) - x^T \beta$
Simple intercept + complex shift	$SI-CS_B$	$\vartheta_k - \eta(B)$
Simple intercept + complex shift + tabular	$SI-CS_B-LS_x$	$\vartheta_k - \eta(B) - x^T \beta$
Simple intercept + tabular	$SI-LS_x$	$\vartheta_k - x^T \beta$

Betas can be interpreted as log-odds ratios

Results

Wine Dataset

- 8 levels of wine quality
- 11 predictors (acidity, sugar content,...)

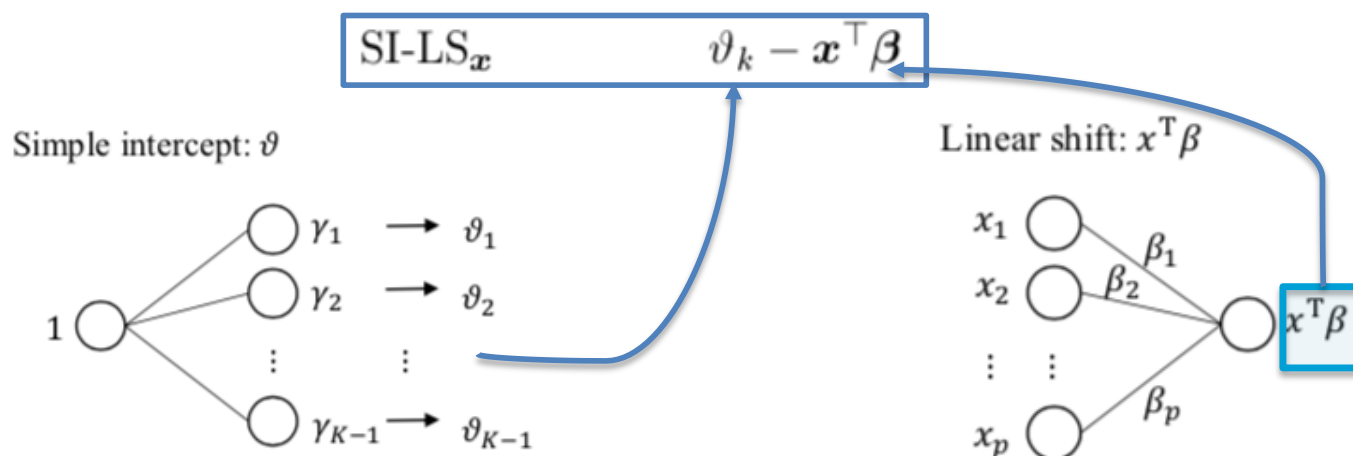
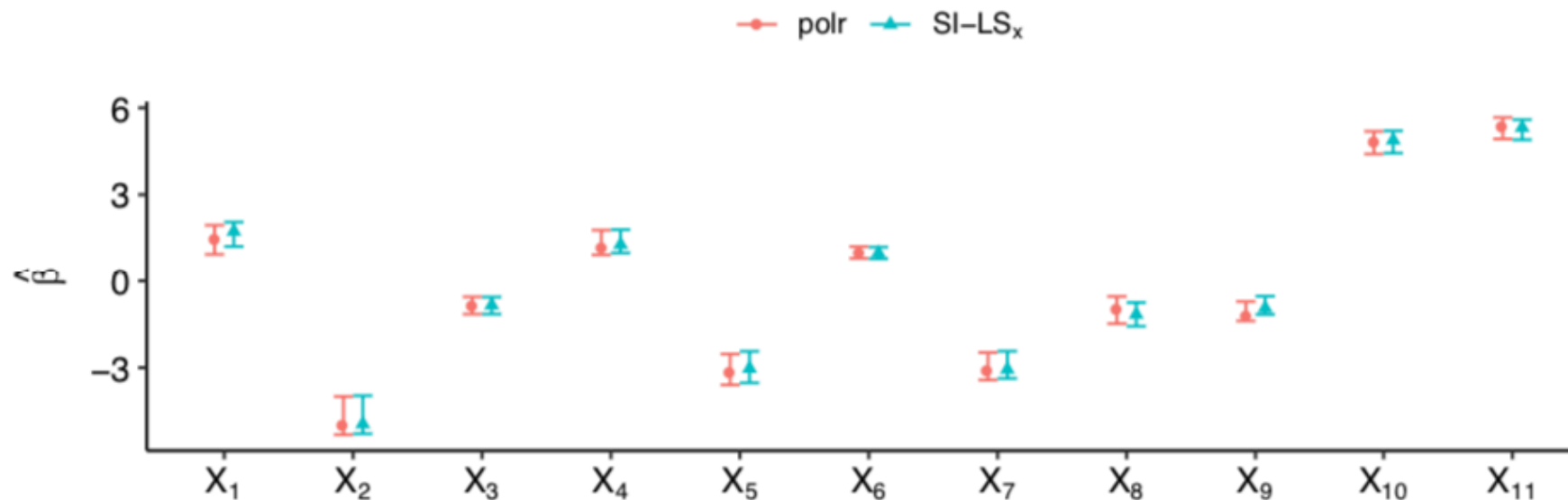


Models

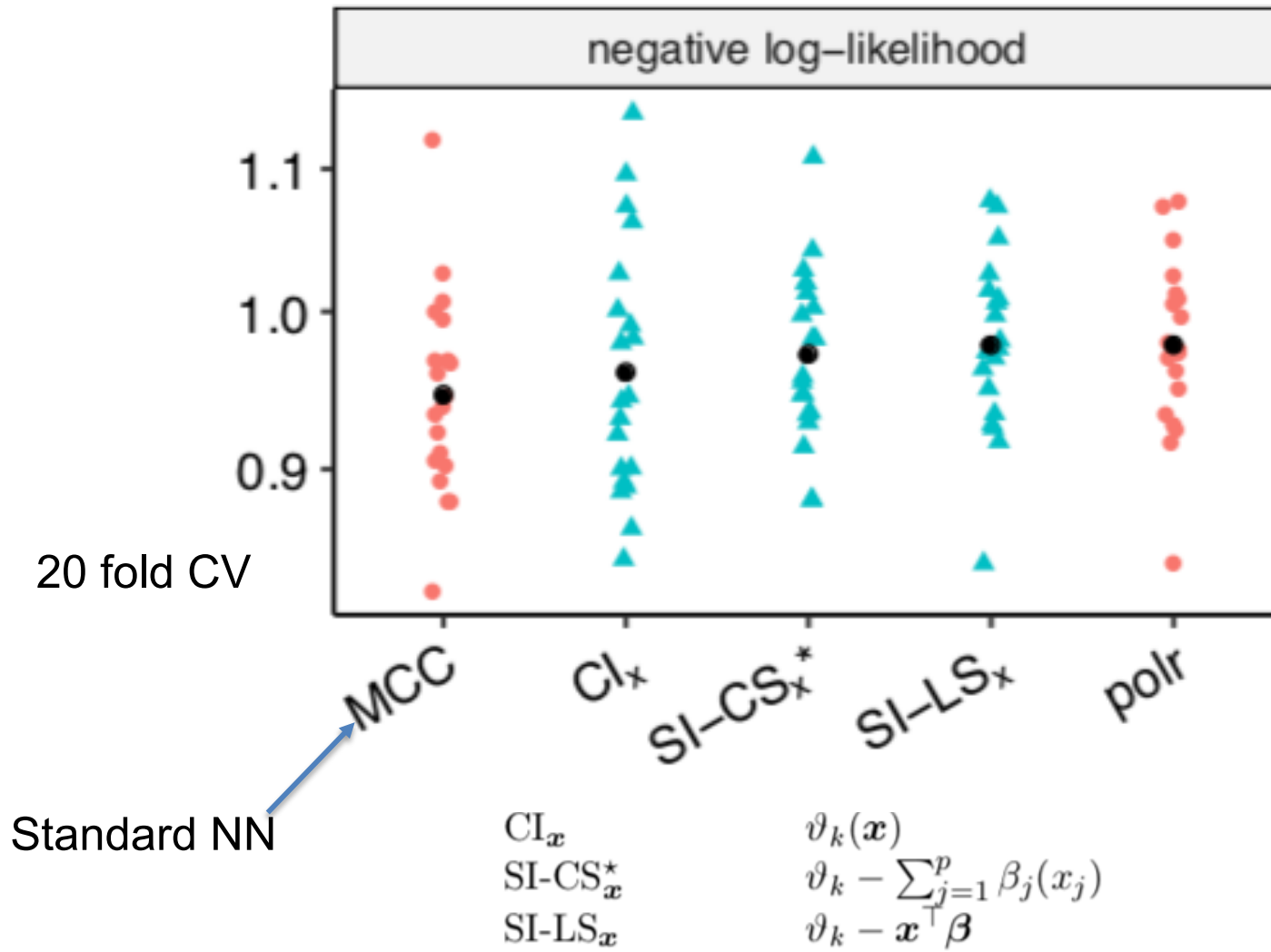
Dataset	Model name	Abbreviation	Trafo $h(y_k D)$
Wine	Multi-class classification	MCC	
	Generalized additive model	GAM	$\vartheta_k - \sum_{j=1}^p \beta_j(x_j)$
	Proportional odds logistic regression	polr	$\vartheta_k - \mathbf{x}^\top \boldsymbol{\beta}$
	Complex intercept	$CI_{\mathbf{x}}$	$\vartheta_k(\mathbf{x})$
	Simple intercept + GAM complex shift	$SI-CS_{\mathbf{x}}^*$	$\vartheta_k - \sum_{j=1}^p \beta_j(x_j)$
	Simple intercept + linear shift	$SI-LS_{\mathbf{x}}$	$\vartheta_k - \mathbf{x}^\top \boldsymbol{\beta}$

Wine Results: I (Comparison with classical methods)

Polr() in R (tram package)

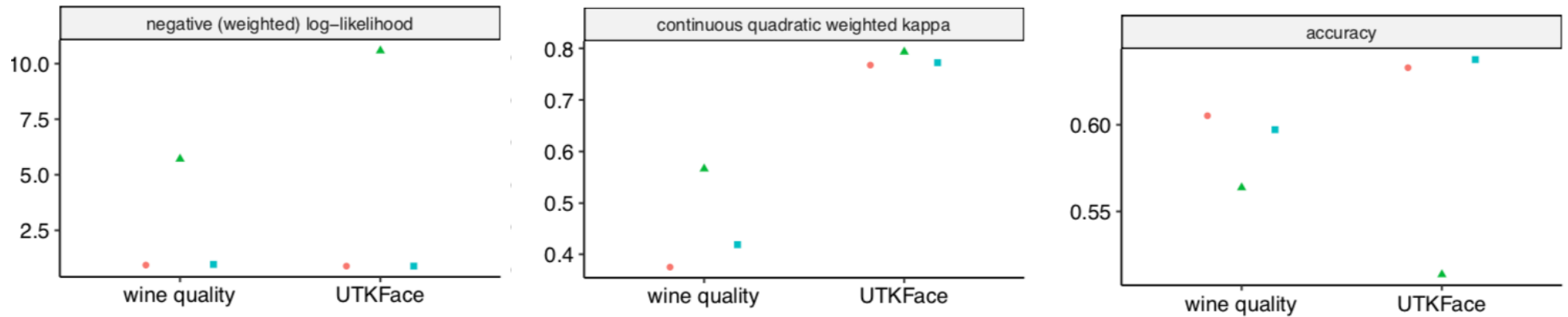


Wine Results: II (Comparison Classical vs. Deep Learning)



Wine Results III (Classification vs. Probabilistic models)

● MCC ▲ QWK ■ CI



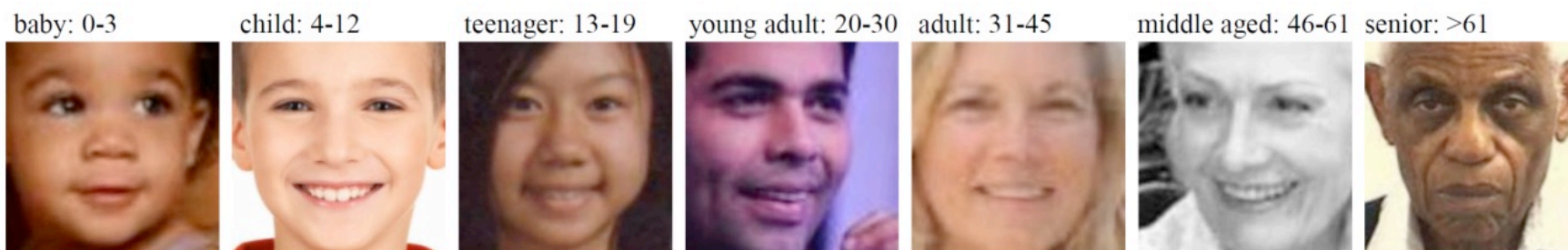
- The probabilistic models have similar NLL and are better than hand-crafted QWK models for NLL and accuracy.
- QWK better on corresponding loss

UKT-Face

Ordinal neural network regression to model ordinal outcome is the **age-category**.

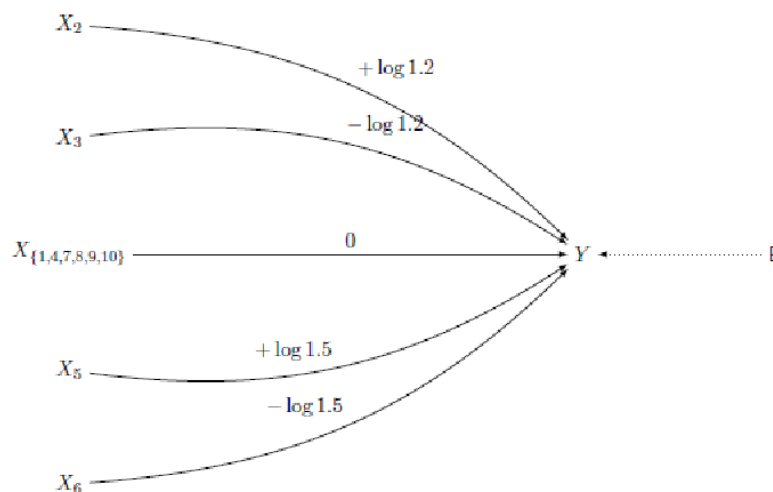
Modeled transformation function: $h(y_k|x, \mathbf{B}) = \vartheta_k - x^\top \beta - \eta(\mathbf{B})$

Image input Data (one random example for each of the 7 age-categories):

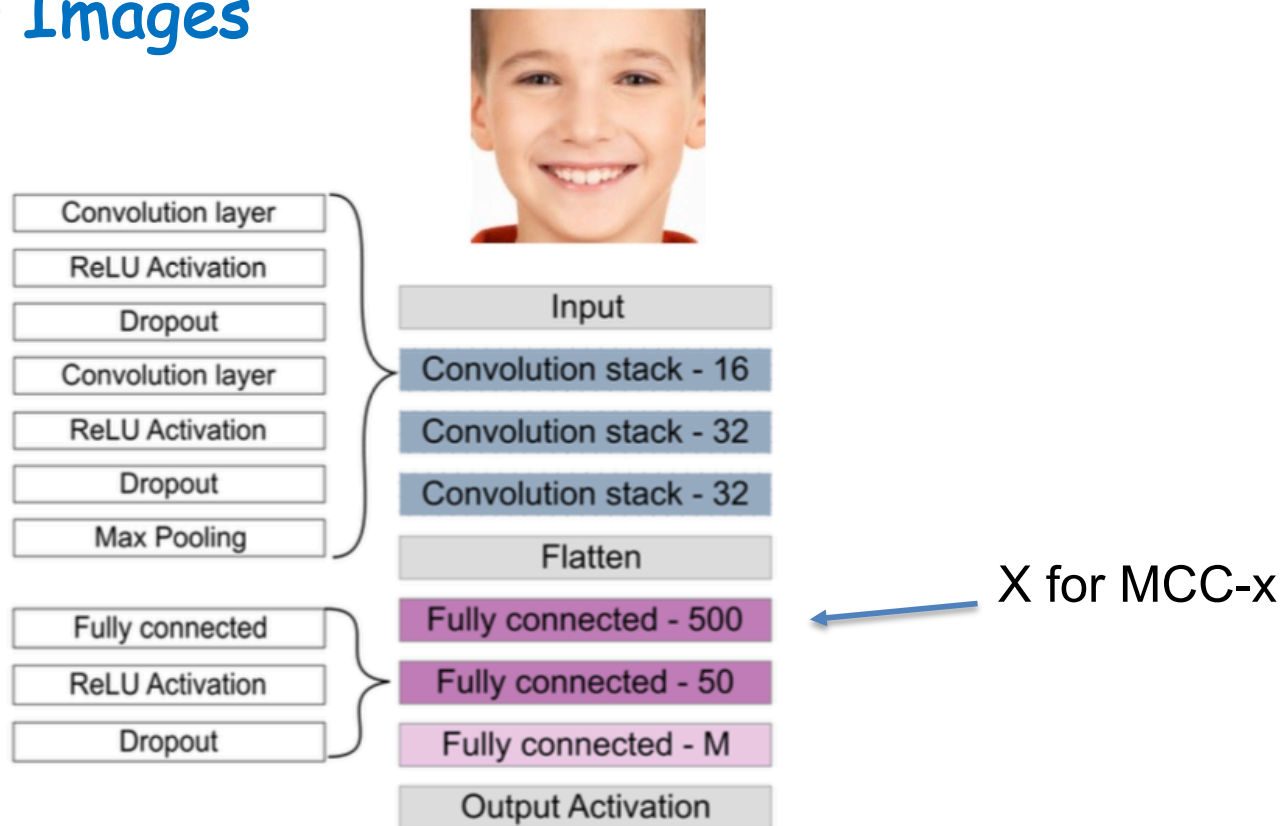


Tabular Co-Variates:

- Gender
- Ethnicity
- 10 simulated covariates with known effect sizes



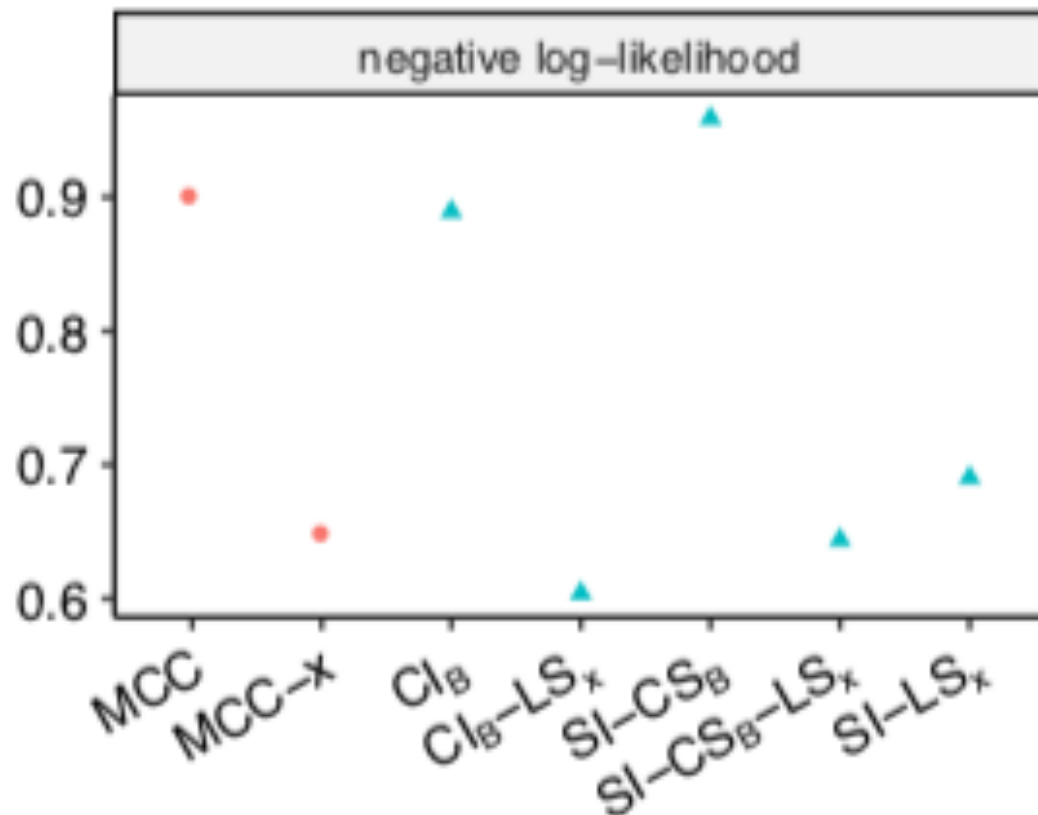
Models for Images



Class-probabilities (MCC), $\vartheta(B), \eta(B)$

MCC	
MCC- \mathbf{x}	
CI_B	$\vartheta_k(B)$
CI_{B-LS_x}	$\vartheta_k(B) - \mathbf{x}^\top \beta$
$SI-CS_B$	$\vartheta_k - \eta(B)$
$SI-CS_{B-LS_x}$	$\vartheta_k - \eta(B) - \mathbf{x}^\top \beta$
$SI-LS_x$	$\vartheta_k - \mathbf{x}^\top \beta$

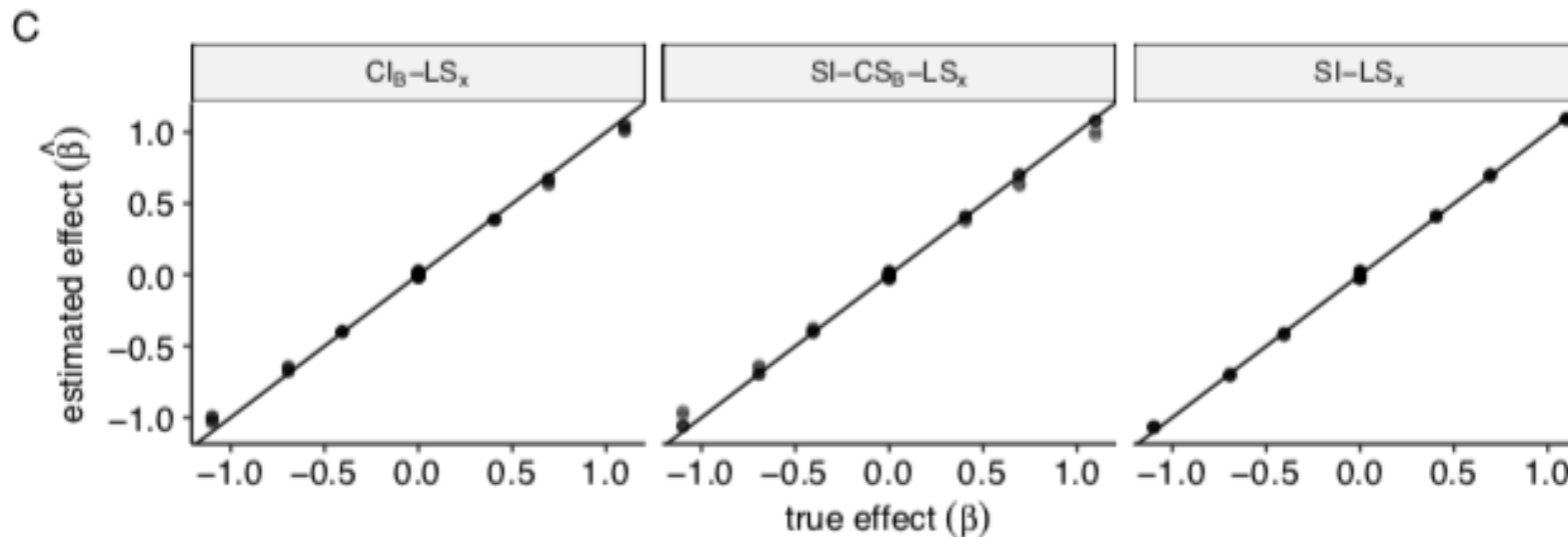
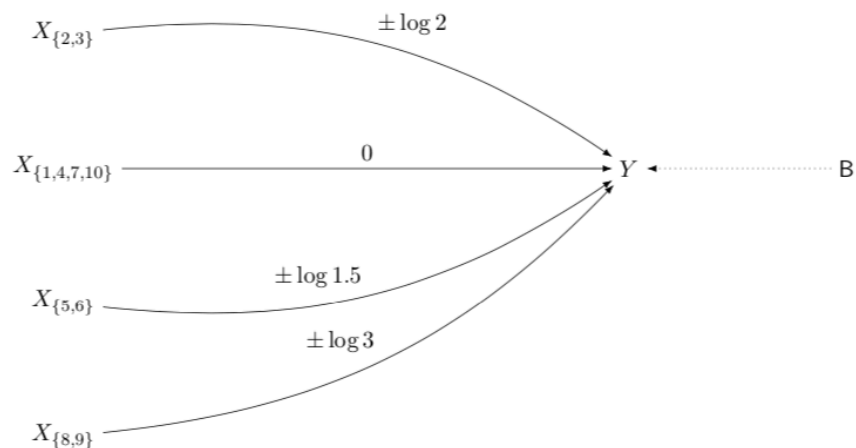
UKT-Face: Prediction Performance



MCC	
MCC- \mathbf{x}	
CI _B	$\vartheta_k(\mathbf{B})$
CI _B -LS _x	$\vartheta_k(\mathbf{B}) - \mathbf{x}^\top \boldsymbol{\beta}$
SI-CS _B	$\vartheta_k - \eta(\mathbf{B})$
SI-CS _B -LS _x	$\vartheta_k - \eta(\mathbf{B}) - \mathbf{x}^\top \boldsymbol{\beta}$
SI-LS _x	$\vartheta_k - \mathbf{x}^\top \boldsymbol{\beta}$

Similar performance as standard Multiclass Neural Networks.
But we can recover effects...

UKT-Face: Can, we recover simulated tabular data?



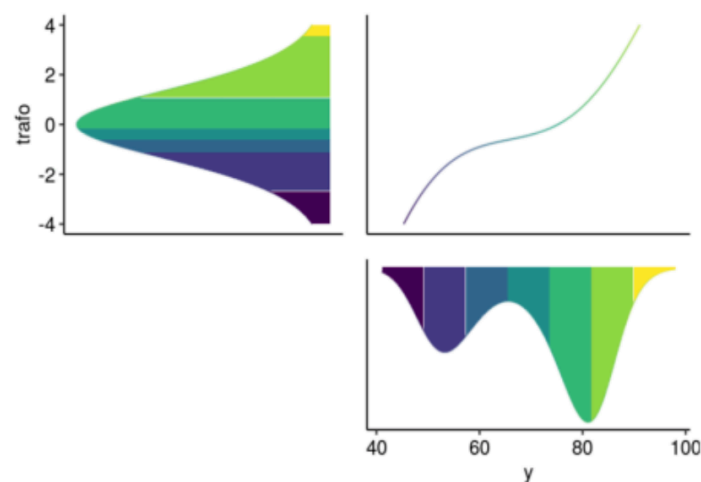
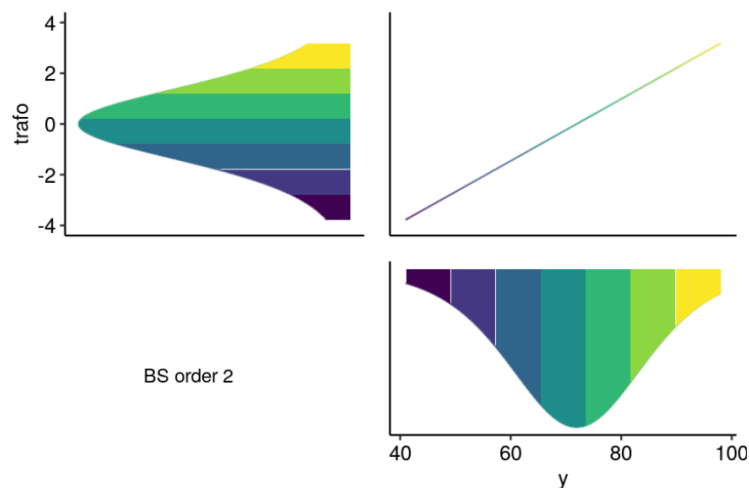
Recovering the simulated effects.
Practical example. Ongoing...

Transformers



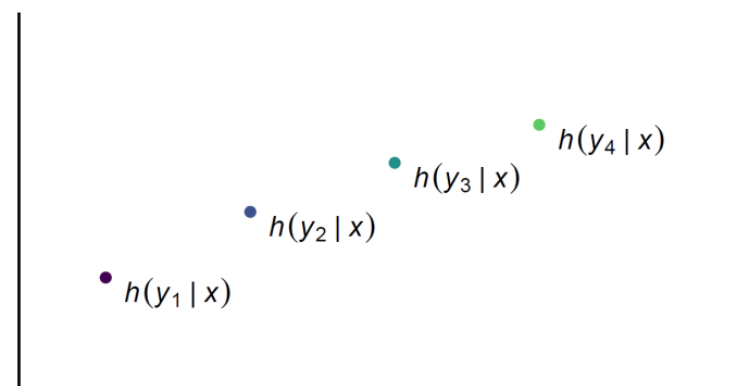
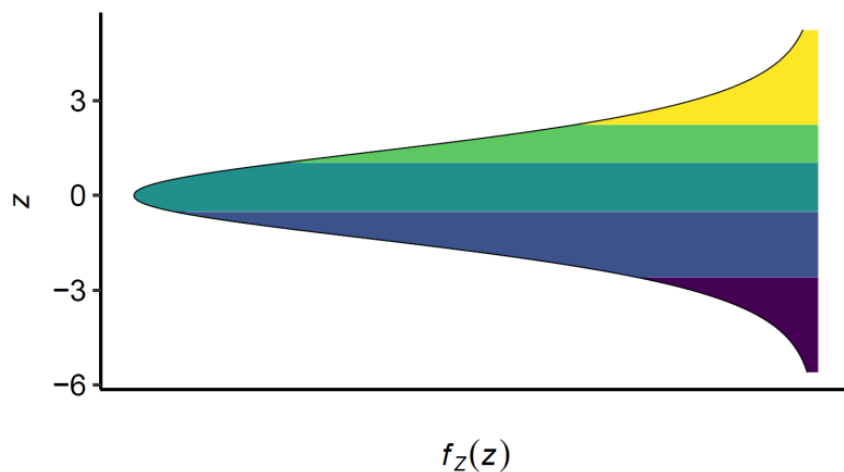
Transformation models

- Powerful tools to bring many statistical approaches in a unifying framework.
 - Numerous examples (tram package and friends)
- Main Idea, model complex distributions via transformations
- See Lucas' and Beate's talks on Transformation models
<https://tensorchiefs.github.io/bbs/>

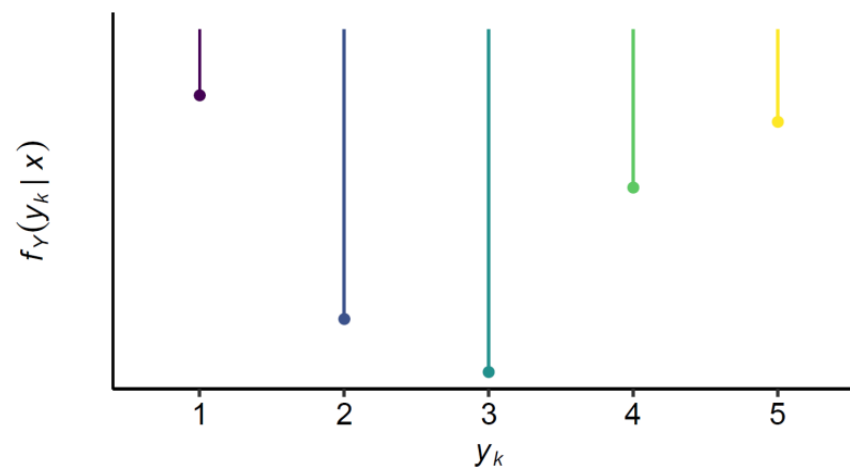


Ordinal regression as transformation model

Transform $(Y = y_k|x)$ to cut-points $h(y_k|x)$ of a latent variable Z .



Ordinal regression fits in the framework



Summary on ordinal neural transformation network (ontram)

- ONTRAM allows to work with image data and tabular predictors
- ONTRAM has the high prediction performance of DL models
- ONTRAM allows for the same interpretability than statistical ordinal regression
 - $F_Z = \text{logistic}$ and $h(y_k|x) = \vartheta_k(B) + \sum_{l=1}^p \beta_l \cdot x_l$ leads to log-odds interpretation of β
- Principles
 - The transformation function yields the cutpoints $h(y_k|x)$ in the latent variable Z
 - The additive parts for the transformation function are determined by NNs
 - All NN are jointly trained by minimizing the NLL = $\sum_{i=1}^n \log(L_i)$

Thank you!

- **Generalizations**
 - **Gompertz (instead of logistics)**
 - For Survival Data β log-hazard rate
 - **Gaussian (instead of logistics)**
 - β cannot be interpreted on the original scale