Boston

Beate and Oliver

1/16/2020

Goal of this script

We want to implement linear transformation models in NN and compare the achieved NLL and estimated ceofficients with the MLT results.

We fit a transformation function $h : (y|x) \to (z|x)$ with the property $(z|x) = h(y|x) \sim N(0, 1)$

In a **linear** transformation model the transformation function has the special form: $h_Y(y) - \sum_i \beta_i x_i$

Then we know, that.

- $F_{Y|X=x}(y) = F_z(h_Y(y) - \sum_i \beta_i x_i)$

Importing the required packages

```
library(MASS)
library(ggplot2)
library(mlt)
```

```
## Loading required package: basefun
```

```
## Loading required package: variables
```

```
##
## Attaching package: 'variables'
```

```
## The following object is masked from 'package:ggplot2':
##
##     unit
```

```
library(basefun)
library(keras)
library(tensorflow)
library(tfprobability)
```

```
## Warning: package 'tfprobability' was built under R version 3.6.2
```

```
T_STEPS = 2000
```

Source functions h and h_dash in w and w/o batch magic

```
# source("mlt_utils.R")  # eg scaling fct
# # preparing eval_h an eval_h_dash, fct implemented in tfp
# source("mlt_utils_keras_v2.R")  # causes error when knittering
#source('https://raw.githubusercontent.com/tensorchiefs/dl_playr/master/mlt/bern_utils.R')
#source('~/Documents/workspace/dl_playr/mlt/bern_utils.R')
```

```
source('bern_utils.R')

source('data.R')
```

Loading the data

We scale the y-varible to [0,1]

```
xy_dat = get_data_boston()
```

```
##  [1] "Names in X : crim"    "Names in X : zn"      "Names in X : indus"
##  [4] "Names in X : chas"    "Names in X : nox"     "Names in X : rm"
##  [7] "Names in X : age"     "Names in X : dis"     "Names in X : rad"
## [10] "Names in X : tax"     "Names in X : ptratio" "Names in X : b"
## [13] "Names in X : lstat"
```

```
dat = xy_dat$dat
sum(dat$y**2) # 299626.3 to compare with BH data in paper
```

```
## [1] 97.90634
```

```
dat$y_obs = dat$y
dat$y = NULL
y_range = xy_dat$scale
dat$y_scale = dat$y_obs
dat$y_obs = NULL
x = xy_dat$x
y = xy_dat$y
```

Defining the model

We set up the formula for the model:

```
fm_large = (y_scale ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax + ptratio + b + lstat)
#fm_small = (y_scale ~  rm + lstat) #lm log lik 346
#fm_uni = (y_scale ~  rm)
(fm = fm_large)
```

```
## y_scale ~ crim + zn + indus + chas + nox + rm + age + dis + rad +
##     tax + ptratio + b + lstat
```

```
is_univariate = TRUE
sum(dat$rm**2)  # 20234.6 to compare with BH data in paper
```

```
## [1] 20234.6
```

Baseline Linear Model

```
fit_lm = lm(fm, data=dat)
fit_lm$coef
```

```
##   (Intercept)          crim            zn         indus          chas
##  6.990997e-01 -2.400252e-03  1.031566e-03  4.568584e-04  5.970520e-02
##           nox            rm           age           dis           rad
```

```
## -3.948136e-01  8.466367e-02  1.538277e-05 -3.279037e-02  6.801100e-03
##           tax       ptratio             b         lstat
## -2.741021e-04 -2.117216e-02  2.069263e-04 -1.166130e-02
```

```
(logLik_lm=logLik(fit_lm) )/nrow(dat) + log(y_range)# the larger the better
```

```
## 'log Lik.' 4.651261 (df=15)
```

MLT fit and results

Variable and Model definition and fit

```
nb = 1  # order defining the Number of Bernstein fct in polynom
len_theta = nb+1
# specify a numeric variable with data in [0,1] and principle bounds [0,Inf]
var_y <- numeric_var("y_scale", support = c(0, 1), bounds = c(-Inf, Inf), add = c(0,0))
# what is done with the bound information (default bounds c(-INF, INF)

# set up monoton increasing polynomial of order nb with Bernstein basis function
bb <- Bernstein_basis(var_y, order=nb, ui="increasing")

# set up grid in interval supp+add -> gives data.frame with col y_scale
y_grid <- as.data.frame(mkgrid(bb, n = 500))

# set up model for mlt
ctm = ctm(bb, shift=fm[-2L], data=dat, todistr="Normal")
#~-1 + crim
#ctm = ctm(bb, shift = ~ b + crim - 1, data=dat, todistr="Normal")
# fm[-2L] defnes the basis function for the shift term h_y(y) in h(y|x)=h_y(y)+h_x(x)
# the intercept is included in the baseline-trafo h_y(y) (not in linear predictor h_x(x))
```

Fit of the model:

```
# fit the mlt model
mlt_fit <- mlt(ctm, data = dat, verbose=TRUE)
```

logLik with MLT

```
(logLik_mlt = logLik(mlt_fit))  #  df = nr-theta + nr-beta
```

```
## 'log Lik.' 427.3669 (df=15)
```

```
# compare to logLik of the baseline model - the larger the better
NLL_MLT = -logLik_mlt / nrow(dat) + log(y_range)
```

Estimated coefficients with MLT

Get the coefficients of the trafo h from the mlt fit:

```
( mlt_fit$coef )
```

```
##  Bs1(y_scale)  Bs2(y_scale)          crim            zn         indus
## -6.7233579933  2.8936981829  0.0230834359 -0.0099206139 -0.0043939933
```

```
##          chas           nox            rm           age           dis
## -0.5741817181  3.7970202703 -0.8142129737 -0.0001480286  0.3153481499
##           rad           tax       ptratio             b         lstat
## -0.0654068642  0.0026360531  0.2036151913 -0.0019900164  0.1121474952
```

```
( theta = mlt_fit$coef[1:(nb+1)] )
```

```
## Bs1(y_scale) Bs2(y_scale)
##    -6.723358     2.893698
```

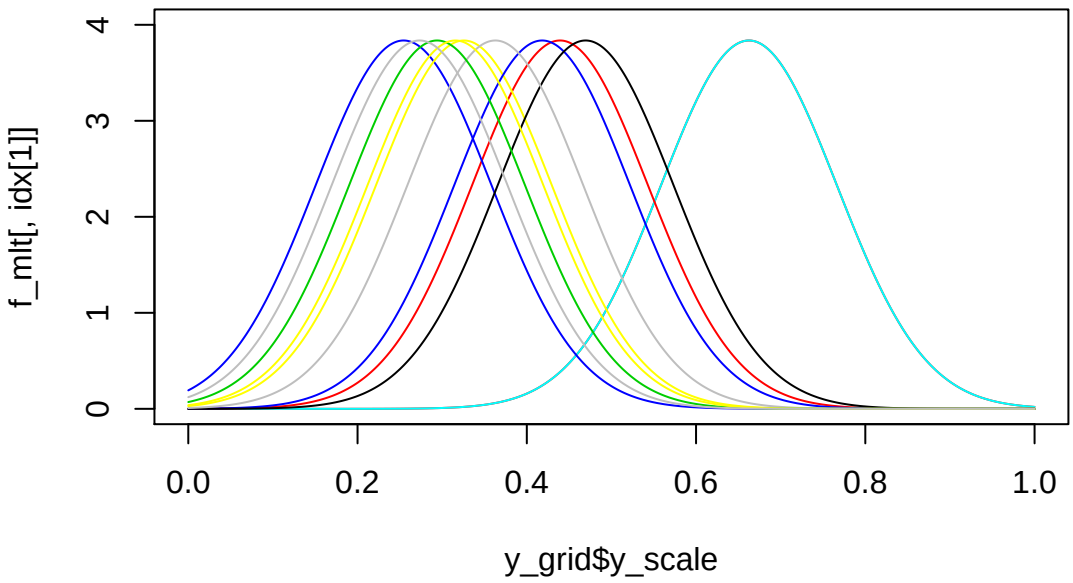
```
( beta = mlt_fit$coef[(nb+2):length(mlt_fit$coef)] )
```

```
##          crim            zn         indus          chas           nox
##  0.0230834359 -0.0099206139 -0.0043939933 -0.5741817181  3.7970202703
##            rm           age           dis           rad           tax
## -0.8142129737 -0.0001480286  0.3153481499 -0.0654068642  0.0026360531
##       ptratio             b         lstat
##  0.2036151913 -0.0019900164  0.1121474952
```

The conditional PDF for some observations

```
f_mlt = predict(mlt_fit, newdata=dat,  q=y_grid$y_scale, type='density')

q_mlt = predict(mlt_fit, newdata=dat,
                prob=c(0.025,0.25,0.5, 0.75,0.975), type='quantile')
q_mlt = t(q_mlt)
#q_mlt = matrix(q_mlt$exact, ncol = 5, byrow = TRUE)
set.seed(3)
idx = sample(1:ncol(f_mlt))[1:10]
m = max(f_mlt[,idx])
plot(y_grid$y_scale, f_mlt[,idx[1]], type='l',col='red', ylim=c(0,4),
     main="mlt-predicted CPD for some picked predictors")
for (i in idx){
  lines(y_grid$y_scale, f_mlt[,i], col=i)
}
```


mlt−predicted CPD for some picked predictors
f_mlt[, idx[1]]
0
1
2
3
4
0.0
0.2
0.4
0.6
0.8
1.0
y_grid$y_scale

NN

NN approach for a linear shift model, modeled with NN

Fitting means to find the *nb* coefficients *theta* for the Bernsteinpolynom which approximaties the transformation function with *nb* being set to:

```
nb
```

```
## [1] 1
```

Preparing input and output

```
y = tf$Variable(as.matrix(dat$y_scale)[,drop=FALSE], dtype='float32')
y$shape # has to be (#y,1)
```

```
## (506, 1)
```

```
# conditional - we give the rm-variables as input to the NN
#x = tf$Variable(as.matrix(dat$rm)[,drop=FALSE], dtype='float32')

#x = tf$Variable(as.matrix(dat[,c('rm','lstat'),drop=FALSE]), dtype='float32')

#dat$chas = as.numeric(as.character(dat$chas))
x = tf$Variable(x, dtype='float32') #all
x$shape  # has to be (#y,1) for a univariate model
```

```
## (506, 13)
```

```
source('model_3.R')
source('bern_utils.R')
source("model_utils.R")
x_dim = as.integer(dim(x)[2])
model_3 = new_model_3(len_theta = as.integer(len_theta), x_dim = x_dim, y_range=y_range)
```

```
## Error in on_load() :
##   TensorFlow Probability has to be used with the TensorFlow Keras implementation.
```

```
T_OUT = 100
run = 1
history = model_train(model_3, make_hist(), x_train = x, y_train = y,
                      x_test = x, y_test = y, T_STEPS=15000)
```

```
## [1] "100 model_3: likelihood (in optimize)  6.85580444335938 likelihood (in test)  6.85331916809082"
## [1] "200 model_3: likelihood (in optimize)  6.61996984481812 likelihood (in test)  6.61773586273193"
## [1] "300 model_3: likelihood (in optimize)  6.40778350830078 likelihood (in test)  6.40577125549316"
## [1] "400 model_3: likelihood (in optimize)  6.21647644042969 likelihood (in test)  6.21466016769409"
## [1] "500 model_3: likelihood (in optimize)  6.04364013671875 likelihood (in test)  6.0419979095459"
## [1] "600 model_3: likelihood (in optimize)  5.88717174530029 likelihood (in test)  5.8856840133667"
## [1] "700 model_3: likelihood (in optimize)  5.74523019790649 likelihood (in test)  5.74387836456299"
## [1] "800 model_3: likelihood (in optimize)  5.61618804931641 likelihood (in test)  5.61495780944824"
## [1] "900 model_3: likelihood (in optimize)  5.49860525131226 likelihood (in test)  5.49748277664185"
## [1] "1000 model_3: likelihood (in optimize)  5.39119958877563 likelihood (in test)  5.39017295837402
## [1] "1100 model_3: likelihood (in optimize)  5.29282379150391 likelihood (in test)  5.29188251495361
## [1] "1200 model_3: likelihood (in optimize)  5.20244646072388 likelihood (in test)  5.20158004760742
## [1] "1300 model_3: likelihood (in optimize)  5.11913824081421 likelihood (in test)  5.11833763122559
```

```
## [1] "1400 model_3: likelihood (in optimize)  5.0420618057251 likelihood (in test)  5.04131937026978"
## [1] "1500 model_3: likelihood (in optimize)  4.97045850753784 likelihood (in test)  4.96976804733276"
## [1] "1600 model_3: likelihood (in optimize)  4.90364933013916 likelihood (in test)  4.90300321578979"
## [1] "1700 model_3: likelihood (in optimize)  4.84102392196655 likelihood (in test)  4.84041690826416"
## [1] "1800 model_3: likelihood (in optimize)  4.78204107284546 likelihood (in test)  4.78146839141846"
## [1] "1900 model_3: likelihood (in optimize)  4.72622537612915 likelihood (in test)  4.72568225860596"
## [1] "2000 model_3: likelihood (in optimize)  4.6731653213501 likelihood (in test)  4.67264747619629"
## [1] "2100 model_3: likelihood (in optimize)  4.62250995635986 likelihood (in test)  4.62201452255249"
## [1] "2200 model_3: likelihood (in optimize)  4.5739631652832 likelihood (in test)  4.57348775863647"
## [1] "2300 model_3: likelihood (in optimize)  4.52728271484375 likelihood (in test)  4.52682447433472"
## [1] "2400 model_3: likelihood (in optimize)  4.4822678565979 likelihood (in test)  4.48182535171509"
## [1] "2500 model_3: likelihood (in optimize)  4.43875885009766 likelihood (in test)  4.43833065032959"
## [1] "2600 model_3: likelihood (in optimize)  4.39662551879883 likelihood (in test)  4.39621067047119"
## [1] "2700 model_3: likelihood (in optimize)  4.35576486587524 likelihood (in test)  4.35536241531372"
## [1] "2800 model_3: likelihood (in optimize)  4.31609392166138 likelihood (in test)  4.31570291519165"
## [1] "2900 model_3: likelihood (in optimize)  4.27754497528076 likelihood (in test)  4.27716493606567"
## [1] "3000 model_3: likelihood (in optimize)  4.24006366729736 likelihood (in test)  4.2396936416626"
## [1] "3100 model_3: likelihood (in optimize)  4.20360469818115 likelihood (in test)  4.20324516296387"
## [1] "3200 model_3: likelihood (in optimize)  4.16813087463379 likelihood (in test)  4.16778087615967"
## [1] "3300 model_3: likelihood (in optimize)  4.1336088180542 likelihood (in test)  4.13326835632324"
## [1] "3400 model_3: likelihood (in optimize)  4.10001039505005 likelihood (in test)  4.0996789932251"
## [1] "3500 model_3: likelihood (in optimize)  4.06731033325195 likelihood (in test)  4.06698751449585"
## [1] "3600 model_3: likelihood (in optimize)  4.03548336029053 likelihood (in test)  4.03516960144043"
## [1] "3700 model_3: likelihood (in optimize)  4.00450706481934 likelihood (in test)  4.00420141220093"
## [1] "3800 model_3: likelihood (in optimize)  3.97435808181763 likelihood (in test)  3.97406077384949"
## [1] "3900 model_3: likelihood (in optimize)  3.94501352310181 likelihood (in test)  3.94472408294678"
## [1] "4000 model_3: likelihood (in optimize)  3.91644978523254 likelihood (in test)  3.91616797447205"
## [1] "4100 model_3: likelihood (in optimize)  3.88864207267761 likelihood (in test)  3.88836765289307"
## [1] "4200 model_3: likelihood (in optimize)  3.86156582832336 likelihood (in test)  3.86129856109619"
## [1] "4300 model_3: likelihood (in optimize)  3.83519554138184 likelihood (in test)  3.83493494987488"
## [1] "4400 model_3: likelihood (in optimize)  3.80950450897217 likelihood (in test)  3.809250831604"
## [1] "4500 model_3: likelihood (in optimize)  3.7844672203064 likelihood (in test)  3.78421998023987"
## [1] "4600 model_3: likelihood (in optimize)  3.76005744934082 likelihood (in test)  3.75981640815735"
## [1] "4700 model_3: likelihood (in optimize)  3.73625040054321 likelihood (in test)  3.73601531982422"
## [1] "4800 model_3: likelihood (in optimize)  3.71302223205566 likelihood (in test)  3.71279287338257"
## [1] "4900 model_3: likelihood (in optimize)  3.69035077095032 likelihood (in test)  3.69012689590454"
## [1] "5000 model_3: likelihood (in optimize)  3.66821575164795 likelihood (in test)  3.66799688339233"
## [1] "5100 model_3: likelihood (in optimize)  3.64659810066223 likelihood (in test)  3.64638447761536"
## [1] "5200 model_3: likelihood (in optimize)  3.62548208236694 likelihood (in test)  3.62527346611023"
## [1] "5300 model_3: likelihood (in optimize)  3.6048526763916 likelihood (in test)  3.60464859008789"
## [1] "5400 model_3: likelihood (in optimize)  3.58469676971436 likelihood (in test)  3.58449745178223"
## [1] "5500 model_3: likelihood (in optimize)  3.56500267982483 likelihood (in test)  3.5648078918457"
## [1] "5600 model_3: likelihood (in optimize)  3.54575991630554 likelihood (in test)  3.54556965827942"
## [1] "5700 model_3: likelihood (in optimize)  3.52695870399475 likelihood (in test)  3.52677273750305"
## [1] "5800 model_3: likelihood (in optimize)  3.50859045982361 likelihood (in test)  3.50840902328491"
## [1] "5900 model_3: likelihood (in optimize)  3.49064660072327 likelihood (in test)  3.49046921730042"
## [1] "6000 model_3: likelihood (in optimize)  3.47311973571777 likelihood (in test)  3.47294640541077"
## [1] "6100 model_3: likelihood (in optimize)  3.4560022354126 likelihood (in test)  3.45583295822144"
## [1] "6200 model_3: likelihood (in optimize)  3.43928742408752 likelihood (in test)  3.43912220001221"
## [1] "6300 model_3: likelihood (in optimize)  3.4229679107666 likelihood (in test)  3.42280673980713"
## [1] "6400 model_3: likelihood (in optimize)  3.40703749656677 likelihood (in test)  3.40688037872314"
## [1] "6500 model_3: likelihood (in optimize)  3.3914897441864 likelihood (in test)  3.39133620262146"
## [1] "6600 model_3: likelihood (in optimize)  3.37631797790527 likelihood (in test)  3.3761682510376"
## [1] "6700 model_3: likelihood (in optimize)  3.36151647567749 likelihood (in test)  3.36137008666992"
```

```
## [1] "6800 model_3: likelihood (in optimize)  3.34707880020142 likelihood (in test)  3.34693622589111"
## [1] "6900 model_3: likelihood (in optimize)  3.33299922943115 likelihood (in test)  3.33286046981812"
## [1] "7000 model_3: likelihood (in optimize)  3.31927275657654 likelihood (in test)  3.31913733482361"
## [1] "7100 model_3: likelihood (in optimize)  3.3058934211731 likelihood (in test)  3.30576157569885"
## [1] "7200 model_3: likelihood (in optimize)  3.29285645484924 likelihood (in test)  3.29272747039795"
## [1] "7300 model_3: likelihood (in optimize)  3.28015685081482 likelihood (in test)  3.28003144264221"
## [1] "7400 model_3: likelihood (in optimize)  3.26779007911682 likelihood (in test)  3.2676682472229"
## [1] "7500 model_3: likelihood (in optimize)  3.2557520866394 likelihood (in test)  3.25563335418701"
## [1] "7600 model_3: likelihood (in optimize)  3.24403858184814 likelihood (in test)  3.24392294883728"
## [1] "7700 model_3: likelihood (in optimize)  3.23264575004578 likelihood (in test)  3.23253345489502"
## [1] "7800 model_3: likelihood (in optimize)  3.22157049179077 likelihood (in test)  3.22146129608154"
## [1] "7900 model_3: likelihood (in optimize)  3.21080899238586 likelihood (in test)  3.21070289611816"
## [1] "8000 model_3: likelihood (in optimize)  3.20035815238953 likelihood (in test)  3.20025515556335"
## [1] "8100 model_3: likelihood (in optimize)  3.19021439552307 likelihood (in test)  3.19011449813843"
## [1] "8200 model_3: likelihood (in optimize)  3.18037486076355 likelihood (in test)  3.18027806282043"
## [1] "8300 model_3: likelihood (in optimize)  3.17083692550659 likelihood (in test)  3.17074298858643"
## [1] "8400 model_3: likelihood (in optimize)  3.16159677505493 likelihood (in test)  3.16150617599487"
## [1] "8500 model_3: likelihood (in optimize)  3.1526517868042 likelihood (in test)  3.15256404876709"
## [1] "8600 model_3: likelihood (in optimize)  3.14399838447571 likelihood (in test)  3.14391326904297"
## [1] "8700 model_3: likelihood (in optimize)  3.13563346862793 likelihood (in test)  3.13555145263672"
## [1] "8800 model_3: likelihood (in optimize)  3.12755370140076 likelihood (in test)  3.12747430801392"
## [1] "8900 model_3: likelihood (in optimize)  3.11975574493408 likelihood (in test)  3.11967897415161"
## [1] "9000 model_3: likelihood (in optimize)  3.11223554611206 likelihood (in test)  3.11216163635254"
## [1] "9100 model_3: likelihood (in optimize)  3.10498952865601 likelihood (in test)  3.10491847991943"
## [1] "9200 model_3: likelihood (in optimize)  3.09801435470581 likelihood (in test)  3.09794592857361"
## [1] "9300 model_3: likelihood (in optimize)  3.09130525588989 likelihood (in test)  3.09123945236206"
## [1] "9400 model_3: likelihood (in optimize)  3.08485865592957 likelihood (in test)  3.0847954750061"
## [1] "9500 model_3: likelihood (in optimize)  3.07867002487183 likelihood (in test)  3.07860946655273"
## [1] "9600 model_3: likelihood (in optimize)  3.07273483276367 likelihood (in test)  3.07267665863037"
## [1] "9700 model_3: likelihood (in optimize)  3.06704831123352 likelihood (in test)  3.06699275970459"
## [1] "9800 model_3: likelihood (in optimize)  3.06160593032837 likelihood (in test)  3.06155276298523"
## [1] "9900 model_3: likelihood (in optimize)  3.05640268325806 likelihood (in test)  3.05635166168213"
## [1] "10000 model_3: likelihood (in optimize)  3.05143260955811 likelihood (in test)  3.0513842105865"
## [1] "10100 model_3: likelihood (in optimize)  3.04669141769409 likelihood (in test)  3.0466446876525"
## [1] "10200 model_3: likelihood (in optimize)  3.04217219352722 likelihood (in test)  3.0421280860900"
## [1] "10300 model_3: likelihood (in optimize)  3.03786969184875 likelihood (in test)  3.0378277301788"
## [1] "10400 model_3: likelihood (in optimize)  3.03377747535706 likelihood (in test)  3.0337374210357"
## [1] "10500 model_3: likelihood (in optimize)  3.02988910675049 likelihood (in test)  3.0298511981964"
## [1] "10600 model_3: likelihood (in optimize)  3.02619862556458 likelihood (in test)  3.0261626243591"
## [1] "10700 model_3: likelihood (in optimize)  3.02269864082336 likelihood (in test)  3.0226645469665"
## [1] "10800 model_3: likelihood (in optimize)  3.01938223838806 likelihood (in test)  3.0193500518798"
## [1] "10900 model_3: likelihood (in optimize)  3.01624250411987 likelihood (in test)  3.0162119865417"
## [1] "11000 model_3: likelihood (in optimize)  3.01327228546143 likelihood (in test)  3.0132431983947"
## [1] "11100 model_3: likelihood (in optimize)  3.01046371459961 likelihood (in test)  3.0104365348815"
## [1] "11200 model_3: likelihood (in optimize)  3.00780963897705 likelihood (in test)  3.0077841281890"
## [1] "11300 model_3: likelihood (in optimize)  3.0053026676178 likelihood (in test)  3.00527834892273"
## [1] "11400 model_3: likelihood (in optimize)  3.00293517112732 likelihood (in test)  3.0029120445251"
## [1] "11500 model_3: likelihood (in optimize)  3.00069952011108 likelihood (in test)  3.0006775856018"
## [1] "11600 model_3: likelihood (in optimize)  2.99858808517456 likelihood (in test)  2.9985675811767"
## [1] "11700 model_3: likelihood (in optimize)  2.99659419059753 likelihood (in test)  2.9965748786926"
## [1] "11800 model_3: likelihood (in optimize)  2.99471044540405 likelihood (in test)  2.9946920871734"
## [1] "11900 model_3: likelihood (in optimize)  2.99292993545532 likelihood (in test)  2.9929125308990"
## [1] "12000 model_3: likelihood (in optimize)  2.99124598503113 likelihood (in test)  2.9912292957305"
## [1] "12100 model_3: likelihood (in optimize)  2.98965215682983 likelihood (in test)  2.9896364212036"
```

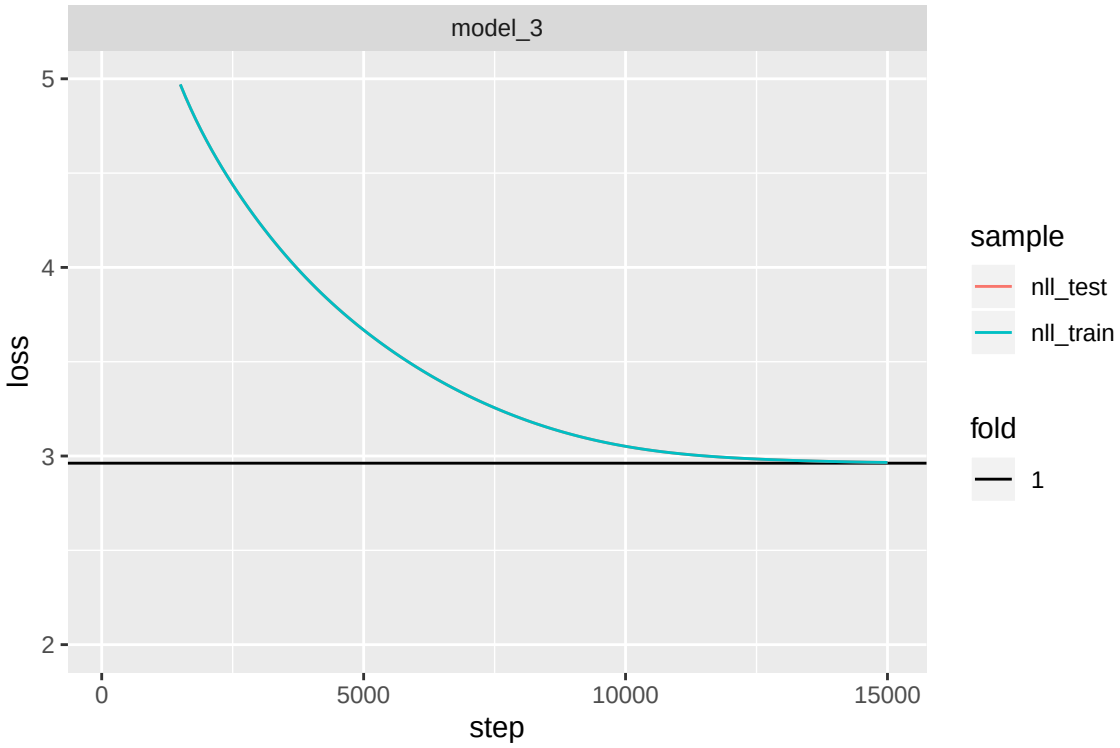
```
## [1] "12200 model_3: likelihood (in optimize)  2.98814249038696 likelihood (in test)  2.9881279468536
## [1] "12300 model_3: likelihood (in optimize)  2.98671126365662 likelihood (in test)  2.9866974353790
## [1] "12400 model_3: likelihood (in optimize)  2.98535346984863 likelihood (in test)  2.9853401184082
## [1] "12500 model_3: likelihood (in optimize)  2.98406386375427 likelihood (in test)  2.9840512275695
## [1] "12600 model_3: likelihood (in optimize)  2.98283815383911 likelihood (in test)  2.9828262329101
## [1] "12700 model_3: likelihood (in optimize)  2.98167157173157 likelihood (in test)  2.9816601276397
## [1] "12800 model_3: likelihood (in optimize)  2.98056077957153 likelihood (in test)  2.9805500507354
## [1] "12900 model_3: likelihood (in optimize)  2.9795024394989 likelihood (in test)  2.9794921875"
## [1] "13000 model_3: likelihood (in optimize)  2.97849321365356 likelihood (in test)  2.9784832000732
## [1] "13100 model_3: likelihood (in optimize)  2.97753024101257 likelihood (in test)  2.9775209426879
## [1] "13200 model_3: likelihood (in optimize)  2.97661089897156 likelihood (in test)  2.9766020774841
## [1] "13300 model_3: likelihood (in optimize)  2.97573351860046 likelihood (in test)  2.9757251739502
## [1] "13400 model_3: likelihood (in optimize)  2.97489547729492 likelihood (in test)  2.9748873710632
## [1] "13500 model_3: likelihood (in optimize)  2.97409582138062 likelihood (in test)  2.9740877151489
## [1] "13600 model_3: likelihood (in optimize)  2.97333168983459 likelihood (in test)  2.9733242988586
## [1] "13700 model_3: likelihood (in optimize)  2.9726026058197 likelihood (in test)  2.97259545326233
## [1] "13800 model_3: likelihood (in optimize)  2.97190713882446 likelihood (in test)  2.9719002246856
## [1] "13900 model_3: likelihood (in optimize)  2.9712438583374 likelihood (in test)  2.97123718261719
## [1] "14000 model_3: likelihood (in optimize)  2.9706118106842 likelihood (in test)  2.97060585021973
## [1] "14100 model_3: likelihood (in optimize)  2.97001051902771 likelihood (in test)  2.9700045585632
## [1] "14200 model_3: likelihood (in optimize)  2.96943831443787 likelihood (in test)  2.9694325923919
## [1] "14300 model_3: likelihood (in optimize)  2.96889448165894 likelihood (in test)  2.9688892364502
## [1] "14400 model_3: likelihood (in optimize)  2.96837902069092 likelihood (in test)  2.9683737754821
## [1] "14500 model_3: likelihood (in optimize)  2.9678897857666 likelihood (in test)  2.9678852558136"
## [1] "14600 model_3: likelihood (in optimize)  2.96742749214172 likelihood (in test)  2.9674227237701
## [1] "14700 model_3: likelihood (in optimize)  2.96699023246765 likelihood (in test)  2.9669861793518
## [1] "14800 model_3: likelihood (in optimize)  2.96657776832581 likelihood (in test)  2.9665737152099
## [1] "14900 model_3: likelihood (in optimize)  2.96618962287903 likelihood (in test)  2.9661860466003
## [1] "15000 model_3: likelihood (in optimize)  2.96582508087158 likelihood (in test)  2.9658212661743
```

```
history$step = as.integer(history$step)
history$fold = as.integer(history$fold)
history$nll_train = as.numeric(history$nll_train)
history$nll_test = as.numeric(history$nll_test)
history$OK = NULL# = as.numeric(history$OK)

library(tidyr)
h = gather(history, 'sample', 'loss', nll_train:nll_test)
h$loss = as.numeric(h$loss)
h$sample = as.factor(h$sample)
h$fold = as.factor(h$fold)
hh =h[!is.na(h$loss),]

ggplot(hh, aes(x=step,y=loss, color=sample, linetype=fold)) +
ylim(2,5) + geom_hline(yintercept=NLL_MLT)+  geom_line() + facet_grid(. ~ method)
```

```
## Warning: Removed 28 rows containing missing values (geom_path).
```



Compare NN model to MLT model

Get beta coefficients

```
( beta_nn = as.numeric( model_3$model_beta$get_weights()[[1]]) )
```

```
##  [1] -0.19048834  0.21082620 -0.01328137  0.15270863 -0.42524067
##  [6]  0.61688751 -0.04010275 -0.65562105  0.40925869 -0.29911718
## [11] -0.42399636  0.18546553 -0.67663097
```

```
 ( beta_mlt = mlt_fit$coef[(len_theta+1):(len_theta+ncol(x))] )
```

```
##          crim            zn         indus          chas           nox
##  0.0230834359 -0.0099206139 -0.0043939933 -0.5741817181  3.7970202703
##            rm           age           dis           rad           tax
## -0.8142129737 -0.0001480286  0.3153481499 -0.0654068642  0.0026360531
##       ptratio             b         lstat
##  0.2036151913 -0.0019900164  0.1121474952
```

```
  beta_nn/beta_mlt
```

```
##         crim           zn        indus         chas          nox
##   -8.2521657  -21.2513263    3.0226186   -0.2659587   -0.1119933
##           rm          age          dis          rad          tax
##   -0.7576488  270.9121351   -2.0790388   -6.2571215 -113.4716052
##      ptratio            b        lstat
##   -2.0823415  -93.1979889   -6.0334025
```

Get theta coefficients

```
  one = tf$ones(shape = c(1,1))
  ( theta_nn = to_theta(model_3$model_hy(one)) )
```

```
## tf.Tensor([[-3.6681843  5.7466593]], shape=(1, 2), dtype=float32)
```

```
  ( theta_mlt = mlt_fit$coef[1:len_theta] )
```

```
## Bs1(y_scale) Bs2(y_scale)
##    -6.723358     2.893698
```

```
  theta_nn$numpy()/theta_mlt
```

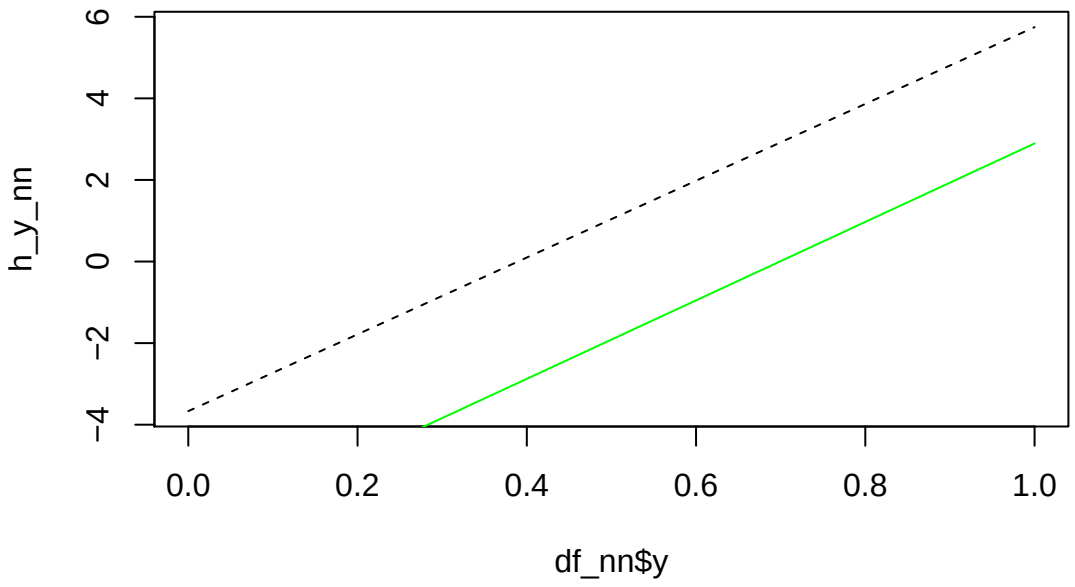
```
##           [,1]     [,2]
## [1,] 0.5455881 1.985922
```

predict baseline trafo (first part of trafo w/o shift)

```
#nn
out_row = model_3$model_hy(one) #Pick row and compute CPD
df_nn = bernp.p_y_h(model_3$bernp, out_row, from = 0, to = 1, length.out = length(y_grid$y_scale))
h_y_nn = df_nn$h

# mlt
h_y_mlt = predict(bb, newdata = y_grid, coef = theta_mlt, type='trafo')

plot(df_nn$y, h_y_nn,  type='l', lty=2)
lines(y_grid$y_scale, h_y_mlt, type='l',col='green')
```

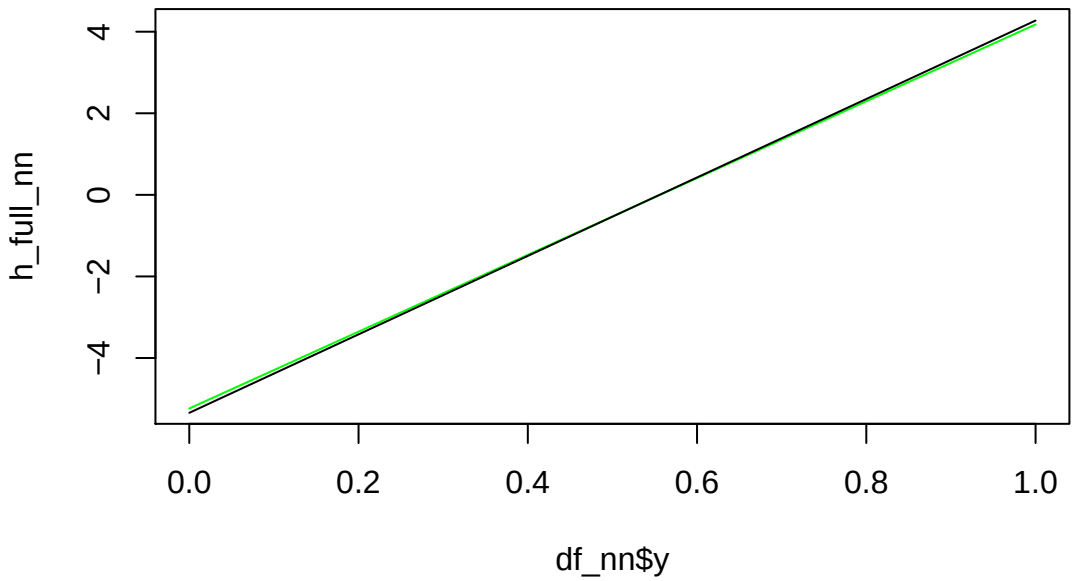


Predict full trafo (all parts of trafo inclusive shift) for picked observation

```
# nn
pick_idx = 1
shift = beta_nn %*% x[pick_idx,1:length(beta_nn)]$numpy()
out_row = model_3$model_hy(one) #Pick row and compute CPD
df_nn = bernp.p_y_h(model_3$bernp, out_row, from = 0, to = 1, length.out = 100, out_eta = shift)
h_full_nn = df_nn$h

# mlt:
h_full_mlt = predict(mlt_fit, newdata = dat[pick_idx,], q=y_grid$y_scale, type='trafo')

plot(df_nn$y, h_full_nn, type='l', col='green')
lines(y_grid$y_scale, h_full_mlt)
```



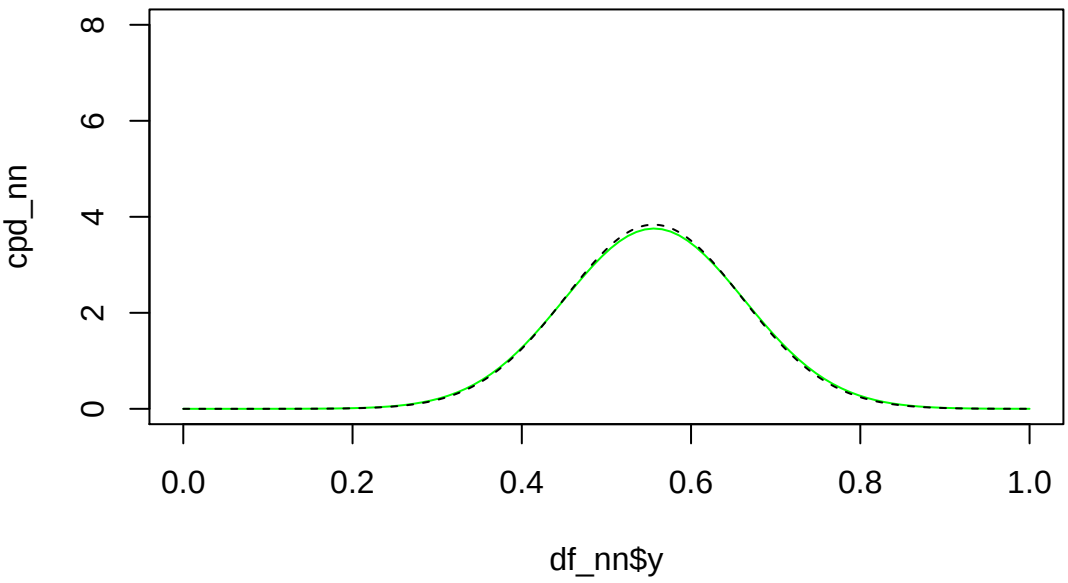
Predict CPD for picked observations

```
# NN

cpd_nn = df_nn$p_y

# mlt:
cpd_mlt = predict(mlt_fit, newdata = dat[pick_idx,], q=y_grid$y_scale, type='density')


plot(df_nn$y, cpd_nn, ylim=c(0,8), type="l", col='green')
lines(y_grid$y_scale, cpd_mlt, lty=2)
```


cpd_nn
0
2
4
6
8
0.0
0.2
0.4
0.6
0.8
1.0
df_nn$y