PROJECT

BLAL BLA BLA

anonymous

1 Introduction

- Background
- Problem formulation/scope
- Main modeling idea
- Some picture of the data?

2 Data Desctription

- Dataset:

```
Number of variables:  6

Number of observations:  3504
```

sex	age	cad_dur	choleste	sigdz	tvdlm
0	73	132	268	1	1
0	68	85	120	1	1
0	54	45	NA	1	0
1	58	86	245	0	0
1	56	7	269	0	0
0	64	0	NA	1	0

Histograms and barplot of the explanatory variables:

Barplots of the responsive variables:

Explanation on data preprocessing (removing NA rows, scaling variables, removing tvdlm etc.)

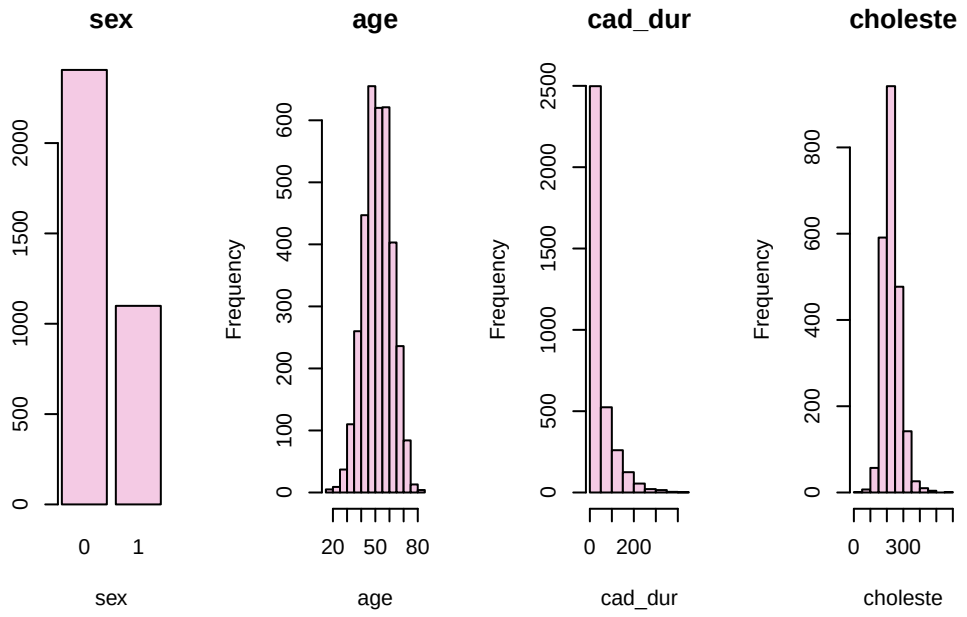


Figure 1: TBD 1

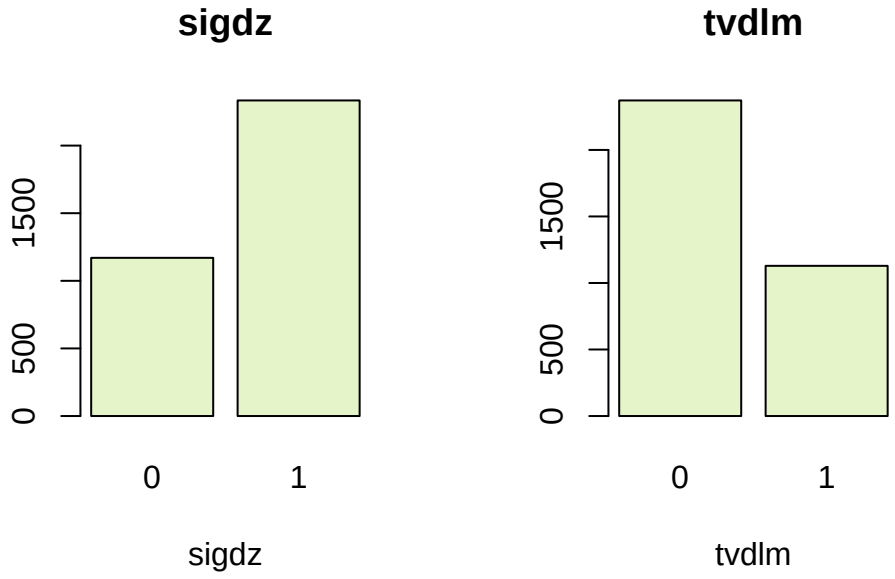


Figure 2: TBD 2

```
After preprocessing we have:

Number of predictive variables:  4

Number of observations:  2258
```

- Histograms
- Correlations
- Has it been used in previous studies?

Bivariate correlations:

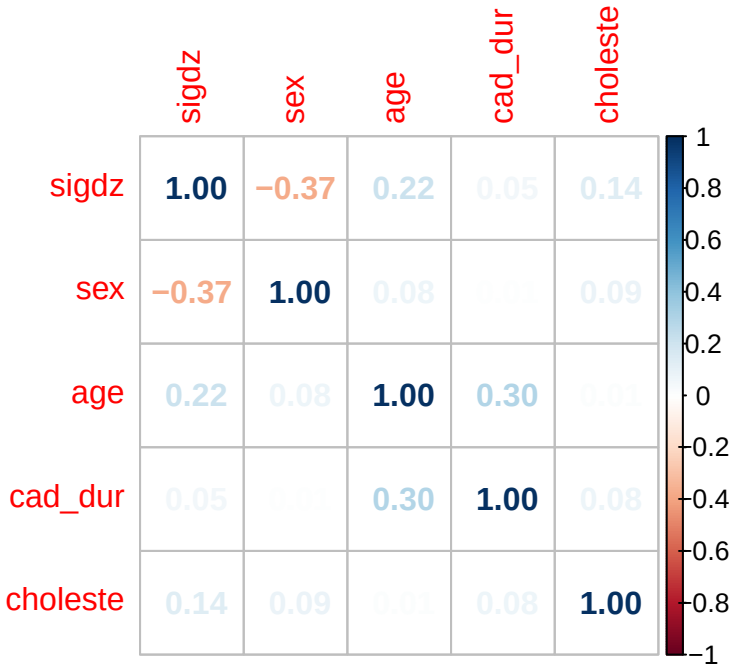


Figure 3: TBD 3

Seems that sex and age have the most significant bivariate correlation to the responsive variable sigdz.

3 Mathematical Model

- Bayesian logistic regression
- Likelihood
- Priors

- Posteriors

Motivation: - Logistic regression gives the estimate of event probability. - Bayesian logistic regression gives also estimate on the *uncertainty* of the predicted probability OR the model itself.

Probability of success: $P(y = 1) = Sigmoid(w^T x + b)$ - Sigmoid maps the output to interval [0, 1]

Bayesian approach: - Instead of learning the weights, we learn the *distribution of the weights*: $P(y = 1|w) = Sigmoid(w^T x + b)$

For M = 'r expl_var' variables and N = 'r n_obs'. Then:

For each variable $(j = 1, ..., M)$, the **prior** is $w_j \sim prior(w_j)$ $\rightarrow$ We define the prior to the **weights** of each variable

For each datapoint x_i and outcome y_i $(i = 1, ..., N)$: $\epsilon_i = Sigmoid(w^T x_i + b)$ $y_i \sim Bernoulli(\epsilon_i)$

4 Model Definitions and Implementation

- Tie mathematical model to our implementation
- Ndraws, warmup, etc.

4.1 Linear model

```
# Make response variable a factor
cath$sigdz <- as.factor(cath$sigdz)

x <- model.matrix(sigdz ~ . - 1, data=cath)
y <- cath$sigdz

# Formula
formula_linear <- formula(sigdz ~ sex + age + cad_dur + choleste)

# Prior
prior_linear <- student_t(df = 3, location = 0, scale = 2.5)

# The model
# model_linear <- stan_glm(formula_linear, data = cath,
#                  family = binomial(link = "logit"),
```

```
#                   prior = prior_linear, prior_intercept = prior_linear, QR=TRUE,
#                   refresh=0)

# saveRDS(model_linear, file = "./additional_files/model_linear.rds")
model_linear <- readRDS("./additional_files/model_linear.rds")
```

4.2 Nonlinear model

```
# Formula
formula_nonlinear <- formula(sigdz ~ sex + s(age) + s(cad_dur) + s(choleste))

# Model definition
# model_nonlinear <- stan_gamm4(
#   formula_nonlinear, data = cath,
#   family = binomial(link = "logit",
#   refresh = 0)
# )

# saveRDS(model_nonlinear, file = "./additional_files/model_nonlinear.rds")
model_nonlinear <- readRDS("./additional_files/model_nonlinear.rds")
```

5 Model Evaluation

- Rhat, ESS, HMC divergences, pp_check(), loo_compare(), classification accuracy
- Prior sensitivity analysis

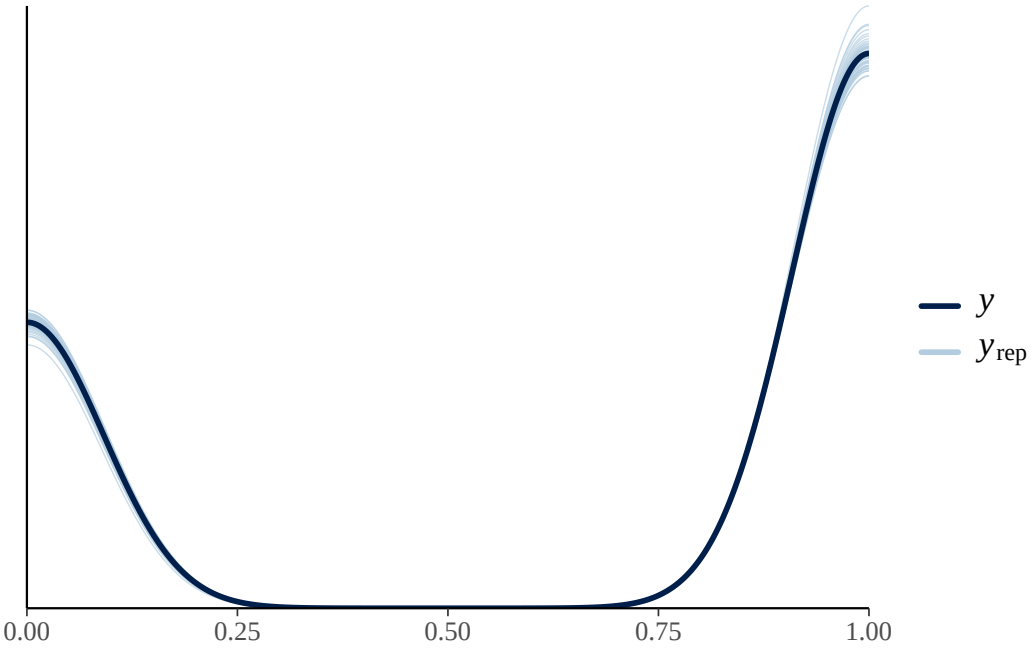
5.1 Convergence diagnostics & posterior predictive checks

```
summary(model_linear)
```

```
Model Info:
 function:     stan_glm
 family:       binomial [logit]
 formula:      sigdz ~ sex + age + cad_dur + choleste
 algorithm:    sampling
 sample:       4000 (posterior sample size)
```

```
 priors:       see help('prior_summary')
 observations: 2258
 predictors:   5

Estimates:
              mean   sd   10%   50%   90%
(Intercept)  0.8    0.1  0.7   0.8   0.9
sex         -1.0    0.1 -1.0  -1.0  -0.9
age          0.7    0.1  0.6   0.7   0.8
cad_dur     -0.1    0.1 -0.2  -0.1   0.0
choleste     0.5    0.1  0.4   0.5   0.5

Fit Diagnostics:
           mean   sd   10%   50%   90%
mean_PPD 0.7    0.0  0.6   0.7   0.7

The mean_ppd is the sample average posterior predictive distribution of the outcome variable

MCMC diagnostics
              mcse Rhat n_eff
(Intercept)   0.0  1.0  4895
sex           0.0  1.0  4687
age           0.0  1.0  4094
cad_dur       0.0  1.0  5440
choleste      0.0  1.0  4964
mean_PPD      0.0  1.0  4773
log-posterior 0.0  1.0  1897

For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective
```

```
pp_check(model_linear)
```



Nonlinear model:

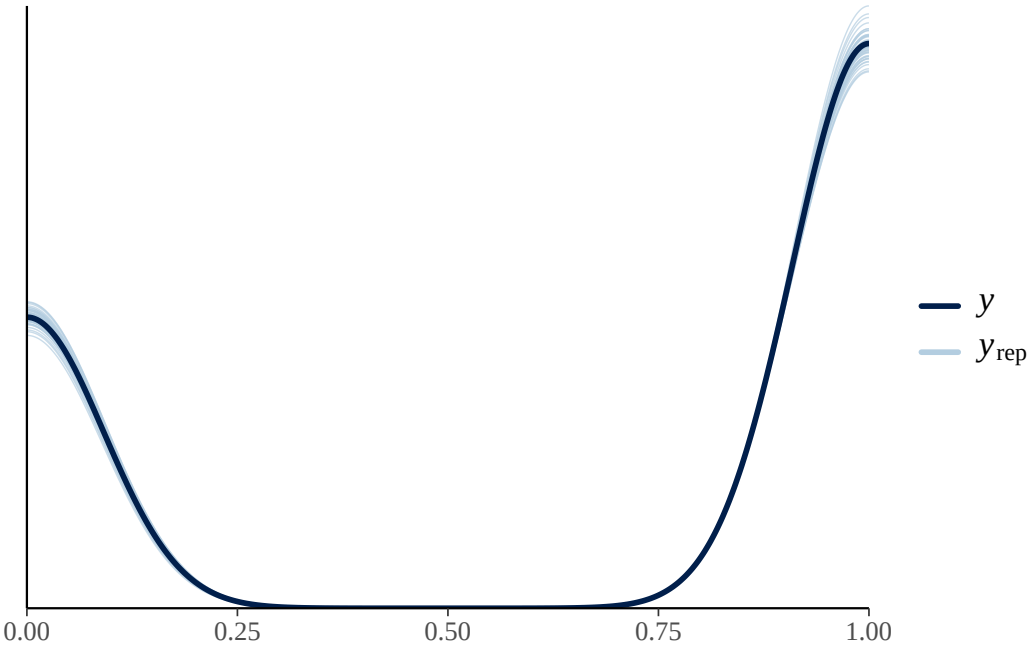
```
summary(model_nonlinear)
```

```
Model Info:
 function:     stan_gamm4
 family:       binomial [logit]
 formula:      sigdz ~ sex + s(age) + s(cad_dur) + s(choleste)
 algorithm:    sampling
 sample:       4000 (posterior sample size)
 priors:       see help('prior_summary')
 observations: 2258

Estimates:
                        mean   sd   10%   50%   90%
(Intercept)              0.8    0.1  0.7   0.8   0.9
sex                     -1.0    0.1 -1.0  -1.0  -0.9
s(age).1                -0.8    2.3 -3.8  -0.4   1.7
s(age).2                -0.9    2.6 -4.2  -0.4   1.8
s(age).3                -2.3    2.5 -5.7  -1.8   0.3
s(age).4                 1.1    2.0 -1.1   0.8   3.8
```

```
s(age).5                 3.3    2.2  0.1   3.3   6.2
s(age).6                -0.6    1.0 -1.9  -0.4   0.5
s(age).7                 1.1    0.9 -0.1   1.3   2.1
s(age).8                -1.3    2.3 -4.2  -0.8   1.0
s(age).9                 1.9    2.8 -0.4   0.6   6.5
s(cad_dur).1            -0.2    1.4 -1.8  -0.1   1.1
s(cad_dur).2             0.2    1.4 -1.2   0.1   1.8
s(cad_dur).3             0.4    1.2 -0.8   0.3   1.8
s(cad_dur).4            -0.4    1.4 -2.0  -0.2   0.9
s(cad_dur).5            -0.9    1.1 -2.3  -0.7   0.2
s(cad_dur).6             0.8    1.1 -0.2   0.7   2.2
s(cad_dur).7             0.5    1.0 -0.6   0.4   1.7
s(cad_dur).8            -0.1    0.8 -1.0   0.0   0.8
s(cad_dur).9            -0.2    1.0 -1.4  -0.1   0.6
s(choleste).1            0.1    2.2 -2.5   0.1   2.7
s(choleste).2           -0.3    2.3 -3.0  -0.2   2.4
s(choleste).3           -1.0    2.0 -3.5  -1.0   1.3
s(choleste).4           -1.2    2.1 -3.9  -1.0   1.2
s(choleste).5           -0.1    1.8 -2.3  -0.1   2.1
s(choleste).6            0.4    1.2 -1.1   0.4   1.9
s(choleste).7            3.5    1.1  2.2   3.4   4.9
s(choleste).8            0.4    1.9 -1.9   0.2   2.8
s(choleste).9            0.2    1.3 -1.0   0.0   1.5
smooth_sd[s(age)1]       2.2    1.3  0.4   2.2   3.9
smooth_sd[s(age)2]       1.7    1.6  0.2   1.2   4.0
smooth_sd[s(cad_dur)1]   1.2    0.7  0.4   1.0   2.1
smooth_sd[s(cad_dur)2]   0.9    0.9  0.1   0.7   2.1
smooth_sd[s(choleste)1]  2.2    0.8  1.3   2.0   3.2
smooth_sd[s(choleste)2]  1.0    1.0  0.1   0.7   2.3

Fit Diagnostics:
           mean   sd   10%   50%   90%
mean_PPD 0.7    0.0  0.6   0.7   0.7

The mean_ppd is the sample average posterior predictive distribution of the outcome variable

MCMC diagnostics
                       mcse Rhat n_eff
(Intercept)            0.0  1.0  7248
sex                    0.0  1.0  7722
s(age).1               0.0  1.0  3902
s(age).2               0.0  1.0  3967
s(age).3               0.1  1.0  2117
```

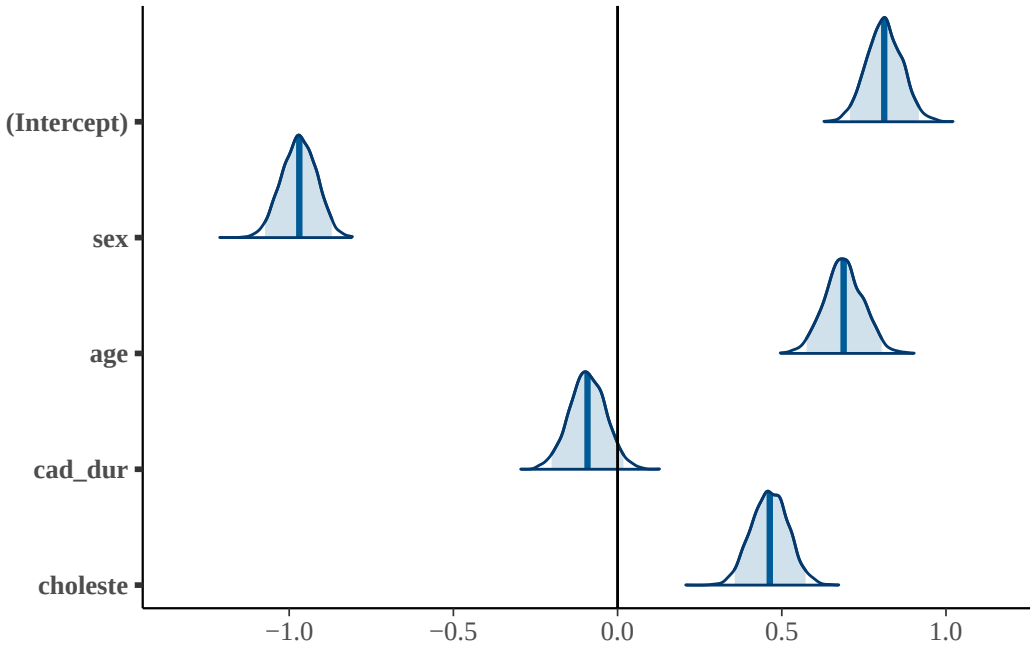
```
s(age).4                0.0  1.0  4963
s(age).5                0.1  1.0   876
s(age).6                0.0  1.0  2884
s(age).7                0.0  1.0   910
s(age).8                0.0  1.0  2604
s(age).9                0.1  1.0   666
s(cad_dur).1            0.0  1.0  4580
s(cad_dur).2            0.0  1.0  5469
s(cad_dur).3            0.0  1.0  5324
s(cad_dur).4            0.0  1.0  4652
s(cad_dur).5            0.0  1.0  3506
s(cad_dur).6            0.0  1.0  4001
s(cad_dur).7            0.0  1.0  3872
s(cad_dur).8            0.0  1.0  4147
s(cad_dur).9            0.0  1.0  3176
s(choleste).1           0.0  1.0  5721
s(choleste).2           0.0  1.0  5747
s(choleste).3           0.0  1.0  5685
s(choleste).4           0.0  1.0  5028
s(choleste).5           0.0  1.0  6365
s(choleste).6           0.0  1.0  4747
s(choleste).7           0.0  1.0  3920
s(choleste).8           0.0  1.0  4384
s(choleste).9           0.0  1.0  2378
smooth_sd[s(age)1]      0.0  1.0   822
smooth_sd[s(age)2]      0.1  1.0   976
smooth_sd[s(cad_dur)1]  0.0  1.0  2453
smooth_sd[s(cad_dur)2]  0.0  1.0  5184
smooth_sd[s(choleste)1] 0.0  1.0  2983
smooth_sd[s(choleste)2] 0.0  1.0  5062
mean_PPD                0.0  1.0  5011
log-posterior           0.1  1.0  1138

For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective
```

```
pp_check(model_nonlinear)
```

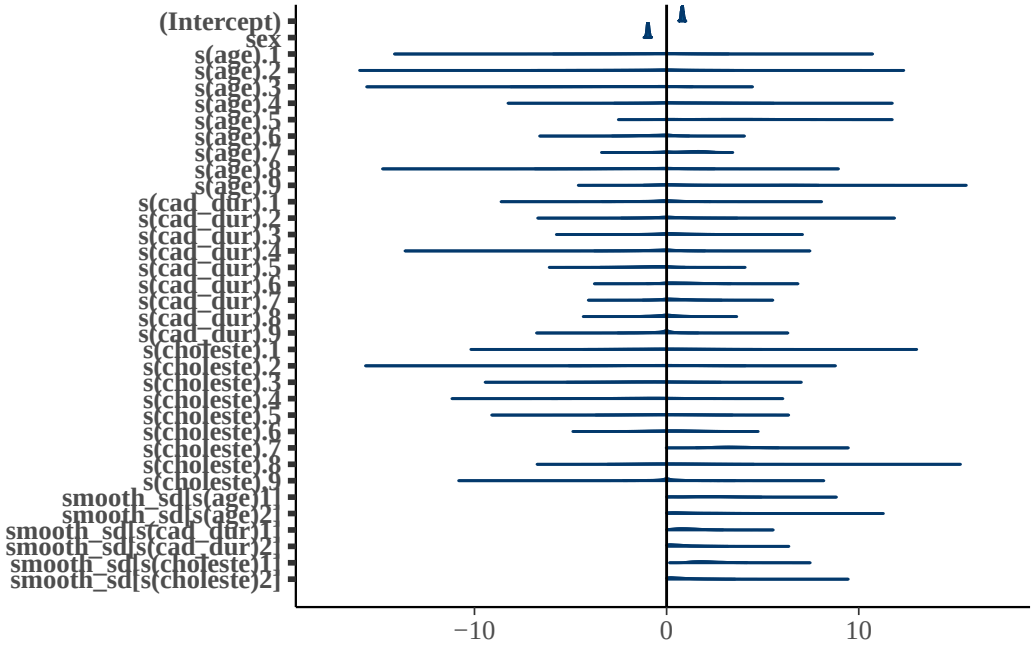


Both:

```
pplot_linear<-plot(model_linear, "areas", prob = 0.95, prob_outer = 1)
pplot_linear + geom_vline(xintercept = 0)
```




```
pplot_nonlinear<-plot(model_nonlinear, "areas", prob = 0.95, prob_outer = 1)
pplot_nonlinear + geom_vline(xintercept = 0)
```



5.2 Model comparison using LOO-CV

```r
# LOO of our models
loo_linear <- loo(model_linear, save_psis = TRUE)
loo_nonlinear <- loo(model_nonlinear, save_psis = TRUE)
print(loo_linear)
```

```
Computed from 4000 by 2258 log-likelihood matrix

         Estimate   SE
elpd_loo  -1177.6 24.4
p_loo         5.4  0.2
looic      2355.2 48.7
------
Monte Carlo SE of elpd_loo is 0.0.

All Pareto k estimates are good (k < 0.5).
See help('pareto-k-diagnostic') for details.
```

```r
print(loo_nonlinear)
```

```
Computed from 4000 by 2258 log-likelihood matrix

         Estimate   SE
elpd_loo  -1170.4 24.3
p_loo        11.6  0.7
looic      2340.8 48.7
------
Monte Carlo SE of elpd_loo is 0.1.

All Pareto k estimates are good (k < 0.5).
See help('pareto-k-diagnostic') for details.
```

```r
# Baseline model of the linear model
model_baseline <- update(model_linear, formula = sigdz ~ 1, QR = FALSE)

# LOO of the baseline model
```

```
loo_baseline <- loo(model_baseline, save_psis = TRUE)
```

Comparing the LOOs:

```
loo_compare(loo_linear, loo_nonlinear, loo_baseline)
```

```
               elpd_diff se_diff
model_nonlinear    0.0       0.0
model_linear      -7.2       3.4
model_baseline  -278.3      21.8
```

The nonlinear model seems to be the best!

5.3 Posterior predictive performance: Classification accuracy

5.3.1 Linear model:

```
## Predicted probabilities
linpred_linear <- posterior_linpred(model_linear)
preds_linear <- posterior_epred(model_linear)
pred_linear <- colMeans(preds_linear)
pr_linear <- as.integer(pred_linear >= 0.5)

## posterior classification accuracy
round(mean(xor(pr_linear,as.integer(y==0))),2)
```

```
[1] 0.75
```

```
# posterior balanced classification accuracy
round((mean(xor(pr_linear[y==0]>0.5,as.integer(y[y==0])))+mean(xor(pr_linear[y==1]<0.5,as.
```

```
[1] 0.69
```

LOO balanced (Better estimate, maybe let's include only these?):

```
# LOO predictive probabilities
ploo_linear=E_loo(preds_linear, loo_linear$psis_object, type="mean", log_ratios = -log_lik
```

```
# LOO classification accuracy
round(mean(xor(ploo_linear>0.5,as.integer(y==0))),2)
```

```
[1] 0.75
```

```
# LOO balanced classification accuracy
round((mean(xor(ploo_linear[y==0]>0.5,as.integer(y[y==0])))+mean(xor(ploo_linear[y==1]<0.5
```

```
[1] 0.69
```

5.3.2 Nonlinear model

```
## Predicted probabilities
linpred_nonlinear <- posterior_linpred(model_nonlinear)
preds_nonlinear <- posterior_epred(model_nonlinear)
pred_nonlinear <- colMeans(preds_nonlinear)
pr_nonlinear <- as.integer(pred_nonlinear >= 0.5)

## posterior classification accuracy
round(mean(xor(pr_nonlinear,as.integer(y==0))),2)
```

```
[1] 0.76
```

```
# posterior balanced classification accuracy
round((mean(xor(pr_linear[y==0]>0.5,as.integer(y[y==0])))+mean(xor(pr_nonlinear[y==1]<0.5,
```

```
[1] 0.69
```

LOO balanced (Better estimate, maybe let's include only these?):

```
# LOO predictive probabilities
ploo_nonlinear=E_loo(preds_nonlinear, loo_nonlinear$psis_object, type="mean", log_ratios =

# LOO classification accuracy
round(mean(xor(ploo_nonlinear>0.5,as.integer(y==0))),2)
```

```
[1] 0.76
```

```
# LOO balanced classification accuracy
round((mean(xor(ploo_nonlinear[y==0]>0.5,as.integer(y[y==0])))+mean(xor(ploo_nonlinear[y==
```

```
[1] 0.69
```

5.3.3 Calibration curves

```
cplot_linear <- ggplot(data = data.frame(pred=pred_linear,loopred=ploo_linear,
  y=as.numeric(y)-1), aes(x=loopred, y=y)) +
  stat_smooth(method='glm', formula = y ~ ns(x, 5), fullrange=TRUE, color="deeppink") +
  geom_abline(linetype = 'dashed') +
  labs(x = "Predicted (LOO)",y = "Observed",title = "Linear model - Calibration plot") +
  geom_jitter(height=0.02, width=0, alpha=0.1) +
  scale_y_continuous(breaks=seq(0,1,by=0.1)) +
  xlim(c(0,1)) +
  theme_minimal()

cplot_noninear <- ggplot(data = data.frame(pred=pred_nonlinear,loopred=ploo_nonlinear,
  y=as.numeric(y)-1), aes(x=loopred, y=y)) +
  stat_smooth(method='glm', formula = y ~ ns(x, 5), fullrange=TRUE, color="deeppink") +
  geom_abline(linetype = 'dashed') +
  labs(x = "Predicted (LOO)",y = "Observed",title = "Nonlinear model - Calibration plot")
  geom_jitter(height=0.02, width=0, alpha=0.1) +
  scale_y_continuous(breaks=seq(0,1,by=0.1)) +
  xlim(c(0,1)) +
  theme_minimal()

grid.arrange(cplot_linear, cplot_noninear, ncol=2)
```

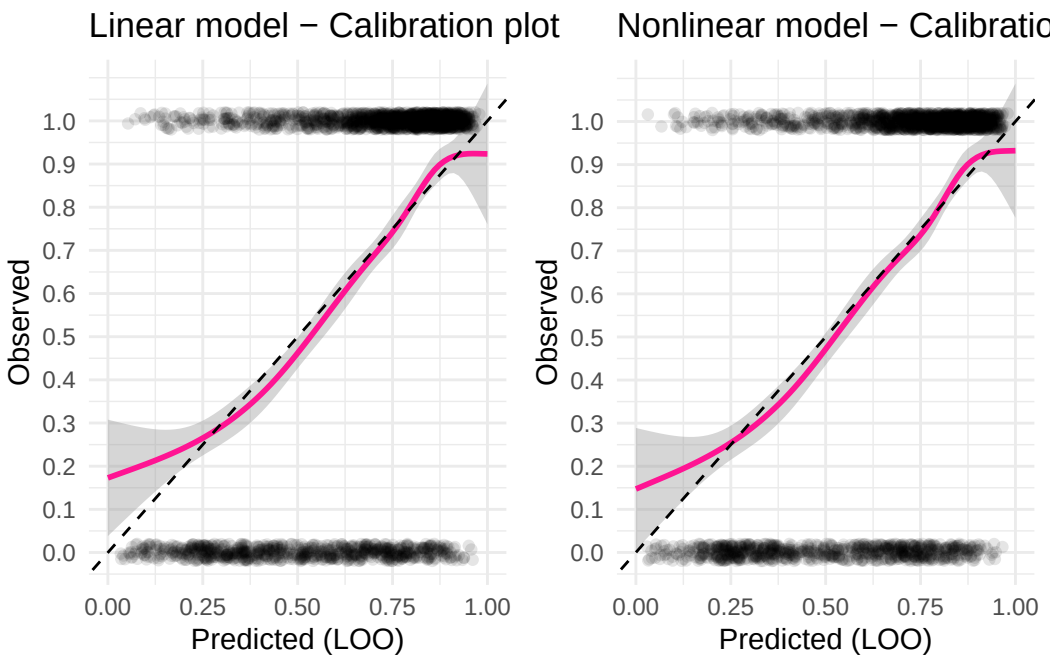


Figure 4: TBD X

6 Discussion

- Problems, potential improvements?
- Conclusion of analysis

7 Lessons Learned

8 References