Exam

2025-05-24

Task 1a: Bootstrap histogram for volatility

```
library(boot)
library(ISLR)
library(insuranceData)

market <- Smarket
str(market)
```

```
## 'data.frame':    1250 obs. of  9 variables:
##  $ Year     : num  2001 2001 2001 2001 2001 ...
##  $ Lag1     : num  0.381 0.959 1.032 -0.623 0.614 ...
##  $ Lag2     : num  -0.192 0.381 0.959 1.032 -0.623 ...
##  $ Lag3     : num  -2.624 -0.192 0.381 0.959 1.032 ...
##  $ Lag4     : num  -1.055 -2.624 -0.192 0.381 0.959 ...
##  $ Lag5     : num  5.01 -1.055 -2.624 -0.192 0.381 ...
##  $ Volume   : num  1.19 1.3 1.41 1.28 1.21 ...
##  $ Today    : num  0.959 1.032 -0.623 0.614 0.213 ...
##  $ Direction: Factor w/ 2 levels "Down","Up": 2 2 1 2 2 2 1 2 2 2 ...
```

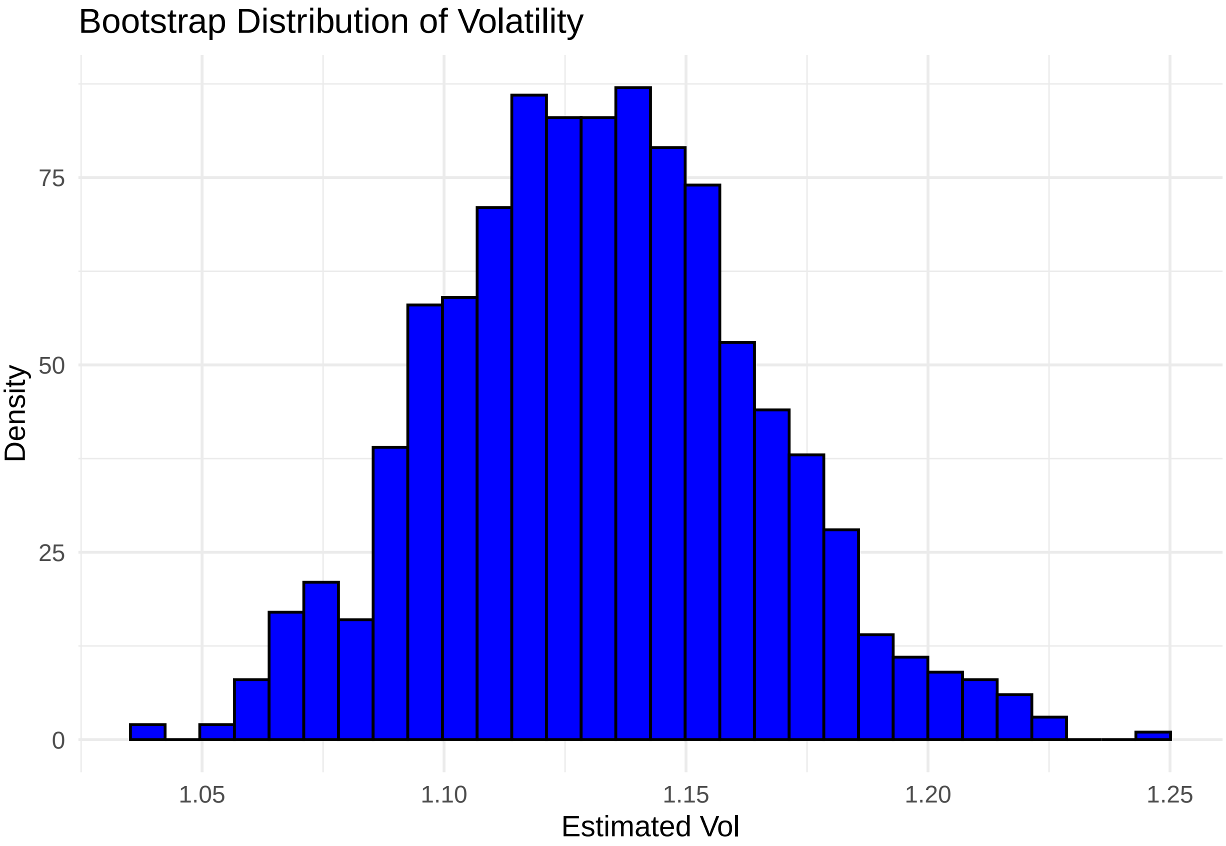
```
set.seed(1)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggplot2)

?boot

#First make a function of what you want to achieve like here we want sd
calculate_sd_Smarket <- function(data_vector, index){
  return(sd(data_vector[index]))
}

## This is why
#sd(market[,"Today"])
```

```
#Then choose the column we want to bootstrap with func.
sd_market_boot<- boot(market$Today, calculate_sd_Smarket, R=1000)

sd_market_boot$t %>%
  #Rename . to t_stat
  data_frame(t_stat=.) %>%
  ggplot(aes(x =t_stat))+
  geom_histogram(color="black", fill="blue", bins=30)+
  theme_minimal()+
  labs(title="Bootstrap Distribution of Volatility", x= "Estimated Vol", y= "Density")
```

```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## i Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Task 1b

```
?boot.ci
conf_intervals <- boot.ci(sd_market_boot, conf = 0.95 , type="norm")
print(conf_intervals)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
```

```
## boot.ci(boot.out = sd_market_boot, conf = 0.95, type = "norm")
##
## Intervals :
## Level      Normal
## 95%   ( 1.077,  1.204 )
## Calculations and Intervals on Original Scale
```

Task 1c

```
conf_intervals_perc<- boot.ci(sd_market_boot, conf = 0.95 , type="perc")
print(conf_intervals_perc)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = sd_market_boot, conf = 0.95, type = "perc")
##
## Intervals :
## Level     Percentile
## 95%   ( 1.069,  1.201 )
## Calculations and Intervals on Original Scale
```

Task 1d

```
model_sq_returns_Smarket <- lm(Today^2 ~ Lag1^2, data = market)

summary(model_sq_returns_Smarket)
```

```
##
## Call:
## lm(formula = Today^2 ~ Lag1^2, data = market)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9098 -1.1772 -0.8766  0.0127 31.0519
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.29098    0.07626  16.928  < 2e-16 ***
## Lag1        -0.19408    0.06714  -2.891  0.00391 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.696 on 1248 degrees of freedom
## Multiple R-squared:  0.006651,   Adjusted R-squared:  0.005855
## F-statistic: 8.356 on 1 and 1248 DF,  p-value: 0.003912
```

Task 1e

```
## Task 1e - Beregn bootstrap-standardfeil for koeffisienten til (Lag1)^2

library(boot)
```

```r
# Definer funksjon som returnerer koeffisienten til (Lag1)^2 i modellen
bootstrap_lag1_squared <- function(data, index) {
  sampled_data <- data[index, ]
  model <- lm(I(Today^2) ~ I(Lag1^2), data = sampled_data)
  return(coef(model)[["I(Lag1^2)"]])  # Navngitt tilgang til koeffisient
}

# Sett frø for reproduserbarhet
set.seed(1)

# Kjør bootstrap med 1000 replikasjoner
boot_result <- boot(
  data = market,
  statistic = bootstrap_lag1_squared,
  R = 1000
)

# Beregn standardavviket (standardfeilen) til koeffisienten
boot_se <- sd(boot_result$t)
cat("Bootstrap-estimert standardfeil:", round(boot_se, 5), "\n")
```

```
## Bootstrap-estimert standardfeil: 0.04953
```

```r
# Sammenlign med OLS-resultatet fra Task 1d
ols_model <- summary(model_sq_returns_Smarket)
cat("OLS standardfeil for koeffisienten:", round(ols_model$coefficients["Lag1", "Std. Error"], 5), "\n"
```

```
## OLS standardfeil for koeffisienten: 0.06714
```

Task 2a

```r
library(insuranceData)
data("dataCar")

str(dataCar)
```

```
## 'data.frame':    67856 obs. of  11 variables:
##  $ veh_value: num  1.06 1.03 3.26 4.14 0.72 2.01 1.6 1.47 0.52 0.38 ...
##  $ exposure : num  0.304 0.649 0.569 0.318 0.649 ...
##  $ clm      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ numclaims: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ claimcst0: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ veh_body : Factor w/ 13 levels "BUS","CONVT",..: 4 4 13 11 4 5 8 4 4 4 ...
##  $ veh_age  : int  3 2 2 2 4 3 3 2 4 4 ...
##  $ gender   : Factor w/ 2 levels "F","M": 1 1 1 1 1 2 2 2 1 1 ...
##  $ area     : Factor w/ 6 levels "A","B","C","D",..: 3 1 5 4 3 3 1 2 1 2 ...
##  $ agecat   : int  2 4 2 2 2 4 4 6 3 4 ...
##  $ X_OBSTAT_: Factor w/ 1 level "01101    0    0    0": 1 1 1 1 1 1 1 1 1 1 ...
```

```r
task2a<- dataCar %>%
  filter(clm!= 0) %>%
  select(-X_OBSTAT_)

str(task2a)
```

```
## 'data.frame':    4624 obs. of  10 variables:
##  $ veh_value: num  1.66 1.51 0.76 1.89 4.06 1.39 2.66 0.5 1.16 3.56 ...
##  $ exposure : num  0.485 0.994 0.539 0.654 0.851 ...
##  $ clm      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ numclaims: int  1 1 1 2 1 1 1 1 2 1 ...
##  $ claimcst0: num  670 807 402 1812 5434 ...
##  $ veh_body : Factor w/ 13 levels "BUS","CONVT",..: 10 10 4 11 11 4 11 4 11 6 ...
##  $ veh_age  : int  3 3 3 3 2 3 1 4 4 3 ...
##  $ gender   : Factor w/ 2 levels "F","M": 2 1 2 2 2 1 1 1 1 2 ...
##  $ area     : Factor w/ 6 levels "A","B","C","D",..: 2 6 3 6 6 1 6 1 2 6 ...
##  $ agecat   : int  6 4 4 2 3 4 5 5 2 4 ...
```

Task 2b

```r
library(ggplot2)

task2b<- task2a %>%
  select(-clm) %>%
  mutate(
    veh_age = as_factor(veh_age),
    agecat = as_factor(veh_age),
  )

  #Find that it has 3 values
  #ggplot(aes(x=numclaims))+ geom_histogram()

  str(task2b)
```

```
## 'data.frame':    4624 obs. of  9 variables:
##  $ veh_value: num  1.66 1.51 0.76 1.89 4.06 1.39 2.66 0.5 1.16 3.56 ...
##  $ exposure : num  0.485 0.994 0.539 0.654 0.851 ...
##  $ numclaims: int  1 1 1 2 1 1 1 1 2 1 ...
##  $ claimcst0: num  670 807 402 1812 5434 ...
##  $ veh_body : Factor w/ 13 levels "BUS","CONVT",..: 10 10 4 11 11 4 11 4 11 6 ...
##  $ veh_age  : Factor w/ 4 levels "1","2","3","4": 3 3 3 3 2 3 1 4 4 3 ...
##  $ gender   : Factor w/ 2 levels "F","M": 2 1 2 2 2 1 1 1 1 2 ...
##  $ area     : Factor w/ 6 levels "A","B","C","D",..: 2 6 3 6 6 1 6 1 2 6 ...
##  $ agecat   : Factor w/ 4 levels "1","2","3","4": 3 3 3 3 2 3 1 4 4 3 ...
```

Task 2c

```r
# Load the tidymodels meta-package for modeling and resampling
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 1.2.0 --

## v broom        1.0.7     v rsample      1.2.1
## v dials        1.3.0     v tune         1.2.1
## v infer        1.0.7     v workflows    1.1.4
## v modeldata    1.4.0     v workflowsets 1.1.0
## v parsnip      1.2.1     v yardstick    1.3.1
## v recipes      1.1.0

## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Use suppressPackageStartupMessages() to eliminate package startup messages
```

```
# Set seed for reproducibility
set.seed(123)

# Create 10-fold cross-validation splits from the task2b dataset
folds <- vfold_cv(task2b, v = 10)

# Specify a linear regression model using the "lm" engine
linmod <- linear_reg() %>% set_engine("lm")

# ---- FULL MODEL ----

# Build a workflow:
# Add the linear model and a recipe using all predictors (.) to predict claimcst0
full_wf <- workflow() %>%
  add_model(linmod) %>%
  add_recipe(recipe(claimcst0 ~ ., data = task2b))

# Perform resampling (cross-validation) using the full model
full_res <- fit_resamples(full_wf, resamples = folds)
```

```
## > A | warning: prediction from rank-deficient fit; consider predict(., rankdeficient="NA")
```

```
## There were issues with some computations   A: x1There were issues with some computations   A: x2There
```

```
# Extract RMSE metric, and label it as "Fullmodell"
full_rmse <- collect_metrics(full_res) %>%
  filter(.metric == "rmse") %>%
  mutate(modell = "Fullmodell")

# ---- INTERCEPT-ONLY MODEL ----

# Build a workflow:
# Only include intercept (no predictors)
int_wf <- workflow() %>%
  add_model(linmod) %>%
  add_recipe(recipe(claimcst0 ~ 1, data = task2b))

# Perform resampling using the intercept-only model
int_res <- fit_resamples(int_wf, resamples = folds)
```

```
## > A | warning: A correlation computation is required, but `estimate` is constant and has 0
##                 standard deviation, resulting in a divide by 0 error. `NA` will be returned.
## There were issues with some computations   A: x1There were issues with some computations   A: x4There
```

```
# Extract RMSE and label it as "Intercept-only"
int_rmse <- collect_metrics(int_res) %>%
  filter(.metric == "rmse") %>%
  mutate(modell = "Intercept-only")

# ---- SIMPLE MODELS: ONE PREDICTOR AT A TIME ----
```

```r
# Extract all predictor names except the response variable
vars <- names(task2b)[names(task2b) != "claimcst0"]

# Loop through each predictor and:
# 1) Build a workflow with just that predictor
# 2) Run CV
# 3) Collect RMSE and attach the variable name
simple_rmses <- map_dfr(vars, function(v) {
  wf <- workflow() %>%
    add_model(linmod) %>%
    add_recipe(recipe(as.formula(paste("claimcst0 ~", v)), data = task2b))
  res <- fit_resamples(wf, resamples = folds)
  collect_metrics(res) %>%
    filter(.metric == "rmse") %>%
    mutate(modell = v)
})

# ---- COMBINE RESULTS ----

# Combine RMSE results from full model, intercept-only, and simple models
all_rmse <- bind_rows(full_rmse, int_rmse, simple_rmses)

# Print a nicely formatted table with RMSE, sorted by mean error
all_rmse %>%
  arrange(mean) %>%
  select(modell, mean, std_err) %>%
  knitr::kable(digits = 3, caption = "RMSE fra 10-fold kryssvalidering")
```

```
## Warning in attr(x, "align"): 'xfun::attr()' is deprecated.
## Use 'xfun::attr2()' instead.
## See help("Deprecated")

## Warning in attr(x, "format"): 'xfun::attr()' is deprecated.
## Use 'xfun::attr2()' instead.
## See help("Deprecated")
```

Table 1: RMSE fra 10-fold kryssvalidering

modell	mean	std_err
Fullmodell	3454.775	194.451
exposure	3470.855	195.319
numclaims	3487.410	202.952
area	3493.028	200.781
gender	3493.695	201.330
Intercept-only	3497.574	201.872
veh_age	3498.478	201.911
agecat	3498.478	201.911
veh_value	3498.591	201.870
veh_body	3501.790	201.948

```r
# Output the full data frame to view all columns if needed
print(all_rmse)
```

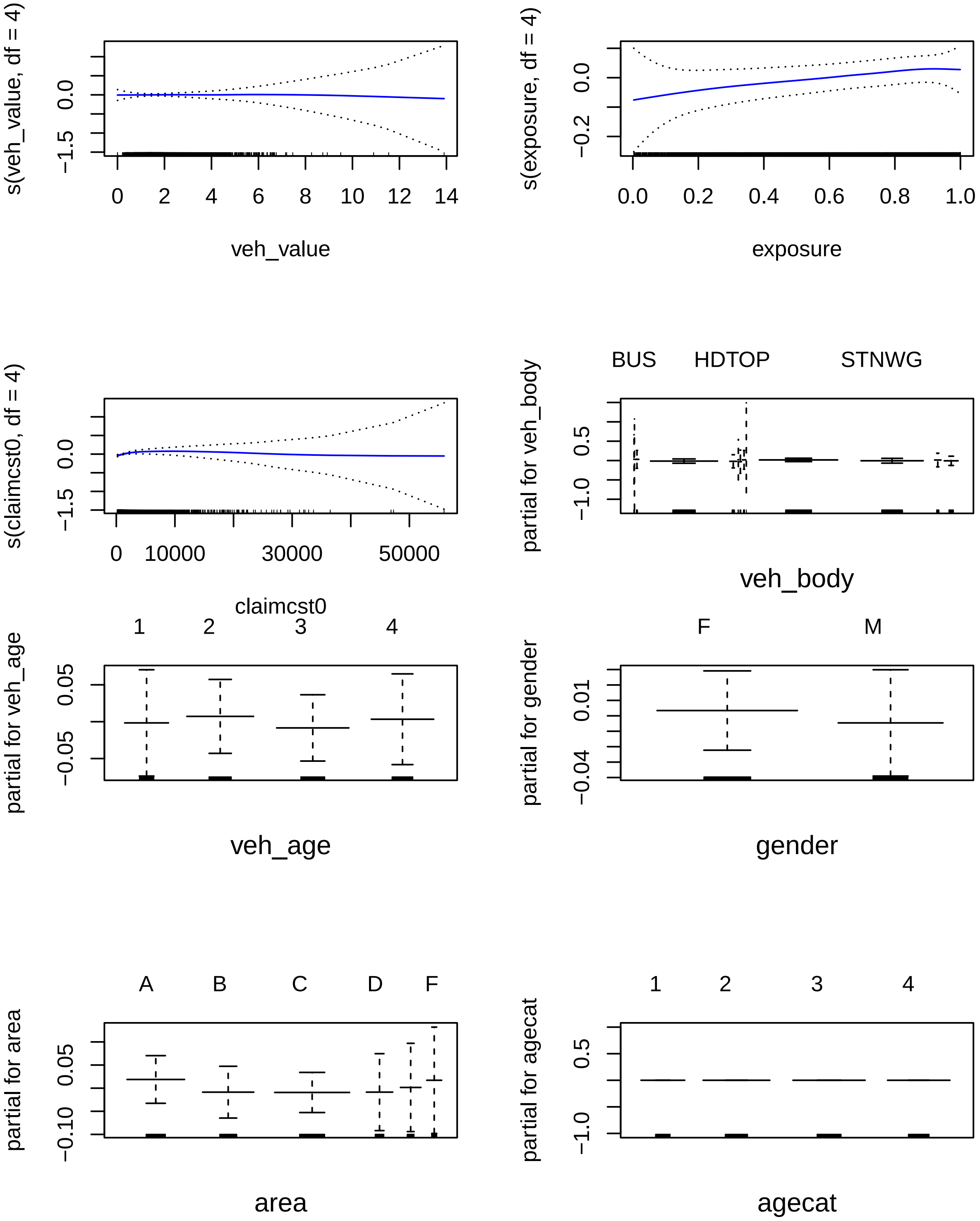
```
## # A tibble: 10 x 7
##    .metric .estimator  mean     n std_err .config              modell
##    <chr>   <chr>      <dbl> <int>   <dbl> <chr>                <chr>
##  1 rmse    standard   3455.    10    194. Preprocessor1_Model1 Fullmodell
##  2 rmse    standard   3498.    10    202. Preprocessor1_Model1 Intercept-only
##  3 rmse    standard   3499.    10    202. Preprocessor1_Model1 veh_value
##  4 rmse    standard   3471.    10    195. Preprocessor1_Model1 exposure
##  5 rmse    standard   3487.    10    203. Preprocessor1_Model1 numclaims
##  6 rmse    standard   3502.    10    202. Preprocessor1_Model1 veh_body
##  7 rmse    standard   3498.    10    202. Preprocessor1_Model1 veh_age
##  8 rmse    standard   3494.    10    201. Preprocessor1_Model1 gender
##  9 rmse    standard   3493.    10    201. Preprocessor1_Model1 area
## 10 rmse    standard   3498.    10    202. Preprocessor1_Model1 agecat
```

Task 2d

```r
library(gam)
```

```
## Loading required package: splines

## Loading required package: foreach

##
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
##
##     accumulate, when

## Loaded gam 1.22-5
```

```r
# Modell: antall krav som funksjon av bilverdi, eksponering og skadebeløp (numeriske) + faktorer
model <- gam(numclaims ~
               s(veh_value, df=4) +
               s(exposure, df=4) +
               s(claimcst0, df=4) +
               veh_body + veh_age + gender + area + agecat,
             family = poisson, data = task2b)

#summary(model)

par(mfrow=c(2,2))
plot(model, se=TRUE, col="blue")
```


s(veh_value, df = 4)
−1.5
0.0
0
2
4
6
8
10
12
14
veh_value
s(exposure, df = 4)
−0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
exposure
s(claimcst0, df = 4)
−1.5
0.0
0
10000
30000
50000
claimcst0
partial for veh_body
BUS
HDTOP
STNWG
−1.0
0.5
veh_body
partial for veh_age
1
2
3
4
−0.05
0.05
veh_age
partial for gender
F
M
−0.04
0.01
gender
partial for area
A
B
C
D
F
−0.10
0.05
area
partial for agecat
1
2
3
4
−1.0
0.5
agecat