Algorithm

```
library(tidyverse)
library(haven)
library(palmerpenguins)
library(gtsummary)
library(caret)
library(finalfit)
library(ranger)
library(kernlab)
```

Import dataset

```
#PR <- read_spss("BDPR7RFL.SAV")
hr <- read_sav("BDHR7RFL.SAV")

#PR_df <-PR |>
# select(HV226, HV206, HV208, HV243A, HV221, HV209, HV242, HV025, HV220, HV219, HV106,
# rename(fuel= HV226, Electricity = HV206,
      #   Television = HV208, Mobile.phone = HV243A, Landline = HV221,
       #  Refrigerator = HV209, separate.kitchen = HV242, residence = HV025, age = HV220,
        # sex = HV219, education = HV106, marital.status = HV115, work.status = SH13,
         #mutate(Cooking.fuel = cut(fuel,
         #                          breaks = c(1,5,10),
         #                          labels = c("Clean Fuel", "Not Clean"),
         #                         right = TRUE))


hr_df <- hr |>
  select(HV226, HV206, HV208, HV243A, HV221, HV209, HV242,HV241, HV025, HV220,
         HV219, `HV106$01`, HV024, `HV115$01`, `SH13$01`, HV270, HV009) |>
  ## Renaming Variable
  rename(fuel= HV226, Electricity = HV206, Television = HV208,
         Mobile.phone = HV243A, Landline = HV221, Refrigerator = HV209,
         separate.kitchen = HV242,  Kitchen = HV241, residence = HV025,
         age = HV220, Division = HV024,
         sex = HV219, education = `HV106$01`, marital.status = `HV115$01`,
         work.status = `SH13$01`, Wealth.index = HV270, Family.size = HV009) |>
```

```
        mutate(cooking.fuel = case_when(fuel <= 5 ~ 1,
                                        ## Categories fuel into two categories
                                        fuel == 6 ~ 0,
                                        ## 1 = Clean, 0 = Unclean
                                        fuel == 7 ~ 0,
                                        fuel == 8 ~ 0,
                                        fuel == 9 ~ 0,
                                        fuel == 10 ~ 0,
                                        fuel == 11 ~ 0,
                                        TRUE ~ NA),
               sex = case_when(sex == 2 ~ 0,
                               sex == 1 ~ 1),
               residence = case_when(residence == 1 ~ 1,
                                     # 1 = Urban 0 = Rural
                                     residence ==2 ~ 0),
               marital.status = case_when(marital.status == 1 ~ 1,
                                          marital.status == 2 ~ 1,
                                          # 1 = Yes
                                          marital.status == 0 ~ 0,
                                          marital.status == 3 ~ 0,
                                          marital.status == 4 ~ 0,
                                          marital.status == 5 ~ 0),
                                          # 0 = No

               separate.kitchen = if_else(separate.kitchen == 1, 1, 0, missing = 1),
                                          # 0 = No 1 = Yes

               tele.communication = case_when(Landline == 1 | Mobile.phone == 1 ~ 1,
                                              TRUE ~ 0)
                                          # 1 = Yes 0 = NO
        )

table(hr_df$separate.kitchen)
```

```
  0     1
387 19070
```

```
table(hr_df$tele.communication)
```

```
   0     1
1042 18415
```

```
head(hr_df)
```

```
# A tibble: 6 x 19
  fuel                Electricity Television Mobile.phone Landline Refrigerator
  <dbl+lbl>           <dbl+lbl>   <dbl+lbl>  <dbl+lbl>    <dbl+lb> <dbl+lbl>
1  8 [Wood]           0 [No]      0 [No]     1 [Yes]      0 [No]   0 [No]
2  8 [Wood]           0 [No]      0 [No]     1 [Yes]      0 [No]   0 [No]
3 11 [Animal dung]    0 [No]      0 [No]     1 [Yes]      0 [No]   0 [No]
4  8 [Wood]           0 [No]      0 [No]     1 [Yes]      0 [No]   0 [No]
5  8 [Wood]           0 [No]      0 [No]     1 [Yes]      0 [No]   0 [No]
6 10 [Agricultural cr~ 0 [No]      0 [No]     1 [Yes]      0 [No]   0 [No]
# i 13 more variables: separate.kitchen <dbl>, Kitchen <dbl+lbl>,
#   residence <dbl>, age <dbl+lbl>, sex <dbl>, education <dbl+lbl>,
#   Division <dbl+lbl>, marital.status <dbl>, work.status <dbl+lbl>,
#   Wealth.index <dbl+lbl>, Family.size <dbl>, cooking.fuel <dbl>,
#   tele.communication <dbl>
```

Multidimentional Energy Poverty Index:

```r
hr_ep <-  hr_df |>
  select(cooking.fuel, Electricity, Television, tele.communication ,
         Refrigerator, separate.kitchen,residence, age, sex, education,
         marital.status, work.status, Wealth.index, Family.size, Division) |>
  mutate(cooking.fuel = case_when( cooking.fuel == 0 ~ 1,
                                   cooking.fuel == 1 ~ 0,
                                   # 1 = Do not use clean fuel
                                   TRUE ~ NA),
         Electricity = case_when( Electricity == 0 ~ 1,
                                  # 1 = Do not have Electricity
                                  Electricity == 1 ~ 0),

         Television = case_when( Television == 0 ~ 1,
                                 # 1 = Do not have Television
                                 Television == 1 ~ 0),

         tele.communication = case_when( tele.communication == 0 ~ 1,
                                 # 1 = Do not have a landline or mobile phone
                                        tele.communication == 1 ~ 0),

         Refrigerator = case_when( Refrigerator == 0 ~ 1,
                                   # 1 = Do not have Refrigerator
                                   Refrigerator == 1 ~ 0),

         separate.kitchen = case_when( separate.kitchen == 0 ~ 1,
                                       # 1 = Do not have separate.kitchen
                                       separate.kitchen == 1 ~ 0),

         ) |>
  na.omit()
```

```
head(hr_ep)
```

```
# A tibble: 6 x 15
  cooking.fuel Electricity Television tele.communication Refrigerator
         <dbl>       <dbl>      <dbl>              <dbl>        <dbl>
1            1           1          1                  0            1
2            1           1          1                  0            1
3            1           1          1                  0            1
4            1           1          1                  0            1
5            1           1          1                  0            1
6            1           1          1                  0            1
# i 10 more variables: separate.kitchen <dbl>, residence <dbl>, age <dbl+lbl>,
#   sex <dbl>, education <dbl+lbl>, marital.status <dbl>,
#   work.status <dbl+lbl>, Wealth.index <dbl+lbl>, Family.size <dbl>,
#   Division <dbl+lbl>
```

```
table(hr_ep$cooking.fuel)
```

```

    0     1
 3983 15434
```

```
table(hr_df$cooking.fuel)
```

```

    0     1
15435  3983
```

```
table(hr_ep$Electricity)
```

```

    0     1
15779  3638
```

```
table(hr_df$Electricity)
```

```

    0     1
 3643 15814
```

```
table(hr_ep$Television)
```

```

    0     1
 9218 10199
```

```r
table(hr_df$Television)
```

```
    0     1
10223  9234
```

```r
table(hr_ep$tele.communication)
```

```
    0     1
18379  1038
```

```r
table(hr_df$tele.communication)
```

```
    0     1
 1042 18415
```

```r
table(hr_ep$Refrigerator)
```

```
    0     1
 5734 13683
```

```r
table(hr_df$Refrigerator)
```

```
    0     1
13711  5746
```

```r
table(hr_ep$separate.kitchen)
```

```
    0     1
19031   386
```

```r
table(hr_df$separate.kitchen)
```

```
    0     1
  387 19070
```

```r
w <-c(0.2, 0.2, 0.15, 0.15, 0.15, 0.15)
```

```r
y1 = as.matrix(hr_ep$cooking.fuel)*(w[1])
y2 = as.matrix(hr_ep$Electricity)*w[2]
y3 = as.matrix(hr_ep$Television)*w[3]
y4 = as.matrix(hr_ep$tele.communication)*w[4]
y5 = as.matrix(hr_ep$Refrigerator)*w[5]
y6 = as.matrix(hr_ep$separate.kitchen)*w[6]

head(y6)
```

```
     [,1]
[1,]    0
[2,]    0
[3,]    0
[4,]    0
[5,]    0
[6,]    0
```

```r
Y = as.matrix(cbind(y1,y2,y3,y4,y5,y6),ncol = 6)



head(Y)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]  0.2  0.2 0.15    0 0.15    0
[2,]  0.2  0.2 0.15    0 0.15    0
[3,]  0.2  0.2 0.15    0 0.15    0
[4,]  0.2  0.2 0.15    0 0.15    0
[5,]  0.2  0.2 0.15    0 0.15    0
[6,]  0.2  0.2 0.15    0 0.15    0
```

```r
#C = Y * as.vector(w)
C = Y

Energy <- C %>% as_tibble() %>%
   mutate(deprivation_score = rowSums(across(where(is.numeric))),

          deprived = case_when( deprivation_score >= 0.35 ~ deprivation_score,
                                deprivation_score < 0.35 ~ 0),

          energy_poor = case_when(deprived == 0 ~ 0,
                                  TRUE ~ 1)
```

```
            )
```

```
Warning: The `x` argument of `as_tibble.matrix()` must have unique column names if
`.name_repair` is omitted as of tibble 2.0.0.
i Using compatibility `.name_repair`.
```

```
  dim(Y)
```

```
[1] 19417     6
```

```
  dim(t(w))
```

```
[1] 1 6
```

```
  head(C)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]  0.2  0.2 0.15    0 0.15    0
[2,]  0.2  0.2 0.15    0 0.15    0
[3,]  0.2  0.2 0.15    0 0.15    0
[4,]  0.2  0.2 0.15    0 0.15    0
[5,]  0.2  0.2 0.15    0 0.15    0
[6,]  0.2  0.2 0.15    0 0.15    0
```

```
  head(Y)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]  0.2  0.2 0.15    0 0.15    0
[2,]  0.2  0.2 0.15    0 0.15    0
[3,]  0.2  0.2 0.15    0 0.15    0
[4,]  0.2  0.2 0.15    0 0.15    0
[5,]  0.2  0.2 0.15    0 0.15    0
[6,]  0.2  0.2 0.15    0 0.15    0
```

```
  head(Energy)
```

```
# A tibble: 6 x 9
     V1    V2    V3    V4    V5    V6 deprivation_score deprived energy_poor
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>             <dbl>    <dbl>       <dbl>
1   0.2   0.2  0.15     0  0.15     0               0.7      0.7           1
2   0.2   0.2  0.15     0  0.15     0               0.7      0.7           1
3   0.2   0.2  0.15     0  0.15     0               0.7      0.7           1
4   0.2   0.2  0.15     0  0.15     0               0.7      0.7           1
5   0.2   0.2  0.15     0  0.15     0               0.7      0.7           1
6   0.2   0.2  0.15     0  0.15     0               0.7      0.7           1
```

```
head = 13113/19417 ;head
```

```
[1] 0.675336
```

```
intensity = sum(Energy$deprived)/13113;intensity
```

```
[1] 0.5199115
```

```
MEPI = head * intensity;MEPI
```

```
[1] 0.351115
```

```
table(Energy$energy_poor)
```

```
   0    1
6304 13113
```

#Building Model

```
hr_ml <-hr_ep |>
  select(Family.size, age, residence, sex, education, marital.status, work.status, Wealth.index, Div
  na.omit()

hr_ml$poverty <- cbind(Energy$energy_poor)

head(hr_ml)
```

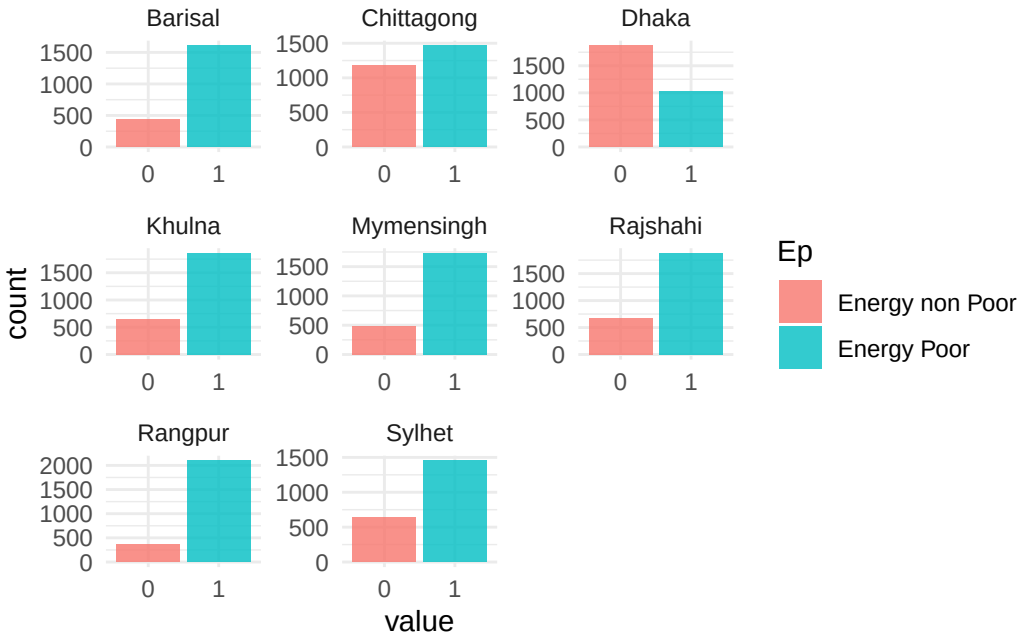
```
# A tibble: 6 x 10
  Family.size age       residence   sex education   marital.status work.status
        <dbl> <dbl+lbl>     <dbl> <dbl> <dbl+lbl>            <dbl> <dbl+lbl>
1           5 45                0     1 1 [Primary]              1 1 [Yes]
2           6 65                0     1 1 [Primary]              1 1 [Yes]
3           5 65                0     1 1 [Primary]              1 1 [Yes]
4           6 68                0     1 1 [Primary]              1 1 [Yes]
5           4 42                0     0 1 [Primary]              1 1 [Yes]
6           5 42                0     1 1 [Primary]              1 1 [Yes]
# i 3 more variables: Wealth.index <dbl+lbl>, Division <dbl+lbl>,
#   poverty <dbl[,1]>
```

```r
table(hr_ml$poverty)
```

```
   0     1
6304 13113
```

```r
hrml1 <- hr_ml |>
  mutate_at(factor, .vars = vars(residence:poverty))
```

```r
hrml1 |>
  pivot_longer(poverty) |>
  mutate(Ep = case_when( value == 1 ~ "Energy Poor",
                         TRUE ~ "Energy non Poor"),
         Division = case_when(Division == 1 ~ "Barisal",
                              Division == 2 ~  "Chittagong",
                              Division == 3 ~ "Dhaka",
                              Division == 4 ~ "Khulna",
                              Division == 5 ~ "Mymensingh",
                              Division == 6 ~ "Rajshahi",
                              Division == 7 ~ "Rangpur",
                              Division == 8 ~ "Sylhet"
                              )) |>
  ggplot(aes(x = value)) +
  geom_bar(aes(fill= Ep),alpha = 0.8) +
  facet_wrap(~Division,scales = "free") +
  theme_minimal()
```




```
hrml1 |>
   tbl_summary(by = poverty) |>
  add_p()
```

Characteristic	**0**, N = 6,304	**1**, N = 13,113	**p-value**
Number of household members	4.00 (3.00, 6.00)	4.00 (3.00, 6.00)	<0.001
Age of head of household	44 (35, 55)	45 (35, 56)	0.001
residence			<0.001
0	2,154 (34%)	10,179 (78%)	
1	4,150 (66%)	2,934 (22%)	
sex			0.5
0	986 (16%)	1,997 (15%)	
1	5,318 (84%)	11,116 (85%)	
education			<0.001
0	858 (14%)	4,550 (35%)	
1	1,461 (23%)	4,842 (37%)	
2	2,064 (33%)	2,932 (22%)	
3	1,915 (30%)	779 (5.9%)	
8	6 (<0.1%)	10 (<0.1%)	
marital.status			<0.001
0	492 (7.8%)	1,325 (10%)	
1	5,812 (92%)	11,788 (90%)	
work.status			<0.001
0	1,181 (19%)	1,633 (12%)	
1	5,123 (81%)	11,480 (88%)	
Wealth.index			<0.001
1	0 (0%)	4,072 (31%)	
2	16 (0.3%)	3,817 (29%)	

Characteristic	0, N = 6,304	1, N = 13,113	p-value
3	582 (9.2%)	3,046 (23%)	
4	1,930 (31%)	1,859 (14%)	
5	3,776 (60%)	319 (2.4%)	
Division			<0.001
1	436 (6.9%)	1,606 (12%)	
2	1,177 (19%)	1,464 (11%)	
3	1,870 (30%)	1,034 (7.9%)	
4	653 (10%)	1,858 (14%)	
5	484 (7.7%)	1,728 (13%)	
6	684 (11%)	1,877 (14%)	
7	368 (5.8%)	2,095 (16%)	
8	632 (10%)	1,451 (11%)	

```
names(hrml1)
```

```
[1] "Family.size"    "age"            "residence"      "sex"
[5] "education"      "marital.status" "work.status"    "Wealth.index"
[9] "Division"       "poverty"
```

```
dependent = names(hrml1)[10]
explanatory = names(hrml1)[-c(10)]

T <-hrml1 |>
  finalfit(dependent,explanatory, metrics = FALSE,p = TRUE,estimate_name = "Odds ratio",digits = c(3


knitr::kable(T,
             caption = "Logistic regression results predicting likelihood of Energy Poverty")
```

Table 2: Logistic regression results predicting likelihood of Energy Poverty

	Dependent: poverty		0	1	Odds ratio (univariable)	Odds ratio (multivariable)
15	Number of household members	Mean (SD)	4.8 (2.3)	4.6 (2.0)	0.956 (0.943 to 0.970, TRUE<0.0001)	0.949 (0.926 to 0.972, TRUE<0.0001)
1	Age of head of household	Mean (SD)	45.1 (13.7)	46.0 (14.6)	1.004 (1.002 to 1.006, TRUE=0.0001)	1.009 (1.005 to 1.013, TRUE<0.0001)
18	residence	0	2154 (17.5)	10179 (82.5)	-	-
19		1	4150 (58.6)	2934 (41.4)	0.150 (0.140 to 0.160, TRUE<0.0001)	0.550 (0.496 to 0.611, TRUE<0.0001)
20	sex	0	986 (33.1)	1997 (66.9)	-	-
21		1	5318 (32.4)	11116 (67.6)	1.032 (0.950 to 1.121, TRUE=0.4563)	1.163 (0.973 to 1.390, TRUE=0.0969)

	Dependent: poverty		0	1	Odds ratio (univariable)	Odds ratio (multivariable)
10	education	0	858 (15.9)	4550 (84.1)	-	-
11		1	1461 (23.2)	4842 (76.8)	0.625 (0.569 to 0.686, TRUE<0.0001)	0.901 (0.777 to 1.044, TRUE=0.1654)
12		2	2064 (41.3)	2932 (58.7)	0.268 (0.244 to 0.294, TRUE<0.0001)	0.799 (0.687 to 0.928, TRUE=0.0033)
13		3	1915 (71.1)	779 (28.9)	0.077 (0.069 to 0.086, TRUE<0.0001)	0.567 (0.473 to 0.679, TRUE<0.0001)
14		8	6 (37.5)	10 (62.5)	0.314 (0.116 to 0.926, TRUE=0.0254)	0.568 (0.036 to 3.809, TRUE=0.6112)
16	marital.status	0	492 (27.1)	1325 (72.9)	-	-
17		1	5812 (33.0)	11788 (67.0)	0.753 (0.675 to 0.839, TRUE<0.0001)	0.765 (0.627 to 0.933, TRUE=0.0083)
27	work.status	0	1181 (42.0)	1633 (58.0)	-	-
28		1	5123 (30.9)	11480 (69.1)	1.621 (1.493 to 1.759, TRUE<0.0001)	1.158 (0.987 to 1.358, TRUE=0.0717)
22	Wealth.index	1	0 (0.0)	4072 (100.0)	-	-
23		2	16 (0.4)	3817 (99.6)	0.000 (0.000 to 0.000, TRUE=0.9334)	0.000 (0.000 to 0.000, TRUE=0.9320)
24		3	582 (16.0)	3046 (84.0)	0.000 (0.000 to 0.000, TRUE=0.9154)	0.000 (0.000 to 0.000, TRUE=0.9140)
25		4	1930 (50.9)	1859 (49.1)	0.000 (0.000 to 0.000, TRUE=0.9074)	0.000 (0.000 to 0.000, TRUE=0.9064)
26		5	3776 (92.2)	319 (7.8)	0.000 (0.000 to 0.000, TRUE=0.8960)	0.000 (0.000 to 0.000, TRUE=0.8952)
2	Division	1	436 (21.4)	1606 (78.6)	-	-
3		2	1177 (44.6)	1464 (55.4)	0.338 (0.296 to 0.385, TRUE<0.0001)	0.543 (0.442 to 0.667, TRUE<0.0001)
4		3	1870 (64.4)	1034 (35.6)	0.150 (0.132 to 0.171, TRUE<0.0001)	0.278 (0.226 to 0.341, TRUE<0.0001)
5		4	653 (26.0)	1858 (74.0)	0.772 (0.672 to 0.887, TRUE=0.0003)	2.096 (1.696 to 2.592, TRUE<0.0001)
6		5	484 (21.9)	1728 (78.1)	0.969 (0.837 to 1.122, TRUE=0.6754)	0.789 (0.629 to 0.991, TRUE=0.0415)
7		6	684 (26.7)	1877 (73.3)	0.745 (0.649 to 0.854, TRUE<0.0001)	0.949 (0.769 to 1.171, TRUE=0.6252)
8		7	368 (14.9)	2095 (85.1)	1.546 (1.326 to 1.802, TRUE<0.0001)	2.291 (1.800 to 2.921, TRUE<0.0001)
9		8	632 (30.3)	1451 (69.7)	0.623 (0.541 to 0.718, TRUE<0.0001)	1.355 (1.081 to 1.698, TRUE=0.0083)

Model Building

```
split <-  createDataPartition(hrml1$poverty, p=3/4, list = FALSE)
training <- hrml1[split,]
testing <- hrml1[-split,]
```

```
set.seed(12345)
model <- train(poverty ~ ., data =training,
method = "svmLinear",
na.action = na.omit,
preProcess = c("scale","center"),
trControl = trainControl(method = "none"),
tune_grid = data.frame(degree=1, scale = 1, C=1))

model.cv <- train(poverty ~ ., data = training,
method = "svmLinear",
na.action = na.omit,
preProcess = c("scale","center"),
trControl = trainControl(method = "cv",number = 10),
tuneGrid = expand.grid(C = c(0,0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2,5)),
tuneLength = 10)
```

```
Warning: model fit failed for Fold01: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning: model fit failed for Fold02: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning: model fit failed for Fold03: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning: model fit failed for Fold04: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning: model fit failed for Fold05: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning: model fit failed for Fold06: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning: model fit failed for Fold07: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning: model fit failed for Fold08: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters
```

```
Warning: model fit failed for Fold09: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning: model fit failed for Fold10: C=0.00 Error in .local(x, ...) :
  No Support Vectors found. You may want to change your parameters

Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
: There were missing values in resampled performance measures.

Warning in train.default(x, y, weights = w, ...): missing values found in
aggregated results
```

```
model.Rf <- train(poverty ~ ., data = training,
method = 'ranger',
na.action = na.omit,
preProcess = c("scale","center"),
trControl = trainControl(method = "cv",number = 10))
```

```
Growing trees.. Progress: 52%. Estimated remaining time: 3 minutes, 18 seconds.
```

```
model.knn <- train(poverty ~ ., data = training,
method = "knn",
na.action = na.omit,
preProcess = c("scale","center"),
trControl = trainControl(method = "cv",number = 10))

model.glm <- train(poverty ~ ., data = training,
method = "glm",
na.action = na.omit,
preProcess = c("scale","center"),
trControl = trainControl(method = "cv",number = 10))
```

Apply model for prediction

```
model.train <- predict(model, training)
model.test <- predict(model, testing)
model.cross <- predict(model.cv,training)
model.cross.test <-  predict(model.cv, testing)
model.random.forest <- predict(model.Rf,training)
model.random.forest.test <- predict(model.Rf, testing)
model.kNN <- predict(model.knn,training)
model.kNN.test <-  predict(model.knn, testing)
model.lr <- predict(model.glm,training)
model.lr.test <-  predict(model.glm, testing)
```

Display confusion matrix

```
model.train.confusion <- confusionMatrix(model.train, training$poverty)
print(model.train.confusion)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 3996  973
         1  732 8862

               Accuracy : 0.8829
                 95% CI : (0.8776, 0.8881)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7365

 Mcnemar's Test P-Value : 6.162e-09

            Sensitivity : 0.8452
            Specificity : 0.9011
         Pos Pred Value : 0.8042
         Neg Pred Value : 0.9237
             Prevalence : 0.3247
         Detection Rate : 0.2744
   Detection Prevalence : 0.3412
      Balanced Accuracy : 0.8731

       'Positive' Class : 0
```

```
model.test.confusion <- confusionMatrix(model.test, testing$poverty)
print(model.test.confusion)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1317  320
         1  259 2958

               Accuracy : 0.8807
                 95% CI : (0.8713, 0.8897)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2e-16
```

```
                Kappa : 0.7307

 Mcnemar's Test P-Value : 0.01265

            Sensitivity : 0.8357
            Specificity : 0.9024
         Pos Pred Value : 0.8045
         Neg Pred Value : 0.9195
             Prevalence : 0.3247
         Detection Rate : 0.2713
   Detection Prevalence : 0.3372
      Balanced Accuracy : 0.8690

       'Positive' Class : 0
```

```
model.cv.confusion <- confusionMatrix(model.cross, training$poverty)
model.cv.confusion1 <- confusionMatrix(model.cross.test, testing$poverty)
print(model.cv.confusion)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 3784  727
         1  944 9108

               Accuracy : 0.8853
                 95% CI : (0.88, 0.8904)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7352

 Mcnemar's Test P-Value : 1.264e-07

            Sensitivity : 0.8003
            Specificity : 0.9261
         Pos Pred Value : 0.8388
         Neg Pred Value : 0.9061
             Prevalence : 0.3247
         Detection Rate : 0.2598
   Detection Prevalence : 0.3098
      Balanced Accuracy : 0.8632

       'Positive' Class : 0
```

```r
print(model.cv.confusion1)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1250  234
         1  326 3044

               Accuracy : 0.8846
                 95% CI : (0.8753, 0.8935)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7329

 Mcnemar's Test P-Value : 0.0001203

            Sensitivity : 0.7931
            Specificity : 0.9286
         Pos Pred Value : 0.8423
         Neg Pred Value : 0.9033
             Prevalence : 0.3247
         Detection Rate : 0.2575
   Detection Prevalence : 0.3057
      Balanced Accuracy : 0.8609

       'Positive' Class : 0
```

```r
model.rf.confusion <- confusionMatrix(model.random.forest,training$poverty)
model.rf.confusion1 <- confusionMatrix(model.random.forest.test, testing$poverty)
print(model.rf.confusion)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 3473  405
         1 1255 9430

               Accuracy : 0.886
                 95% CI : (0.8807, 0.8911)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7273
```

```
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.7346
            Specificity : 0.9588
         Pos Pred Value : 0.8956
         Neg Pred Value : 0.8825
             Prevalence : 0.3247
         Detection Rate : 0.2385
   Detection Prevalence : 0.2663
      Balanced Accuracy : 0.8467

       'Positive' Class : 0
```

```
print(model.rf.confusion1)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1133  144
         1  443 3134

               Accuracy : 0.8791
                 95% CI : (0.8696, 0.8881)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7099

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.7189
            Specificity : 0.9561
         Pos Pred Value : 0.8872
         Neg Pred Value : 0.8762
             Prevalence : 0.3247
         Detection Rate : 0.2334
   Detection Prevalence : 0.2631
      Balanced Accuracy : 0.8375

       'Positive' Class : 0
```

```
model.knn.confusion <- confusionMatrix(model.kNN, training$poverty)
model.knn.confusion1 <- confusionMatrix(model.kNN.test, testing$poverty)
```

```r
print(model.knn.confusion)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 3838  640
         1  890 9195

               Accuracy : 0.8949
                 95% CI : (0.8898, 0.8999)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7571

 Mcnemar's Test P-Value : 1.943e-10

            Sensitivity : 0.8118
            Specificity : 0.9349
         Pos Pred Value : 0.8571
         Neg Pred Value : 0.9118
             Prevalence : 0.3247
         Detection Rate : 0.2635
   Detection Prevalence : 0.3075
      Balanced Accuracy : 0.8733

       'Positive' Class : 0
```

```r
print(model.knn.confusion1)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1213  263
         1  363 3015

               Accuracy : 0.871
                 95% CI : (0.8613, 0.8803)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.701

 Mcnemar's Test P-Value : 7.595e-05
```

```
            Sensitivity : 0.7697
            Specificity : 0.9198
         Pos Pred Value : 0.8218
         Neg Pred Value : 0.8925
             Prevalence : 0.3247
         Detection Rate : 0.2499
   Detection Prevalence : 0.3041
      Balanced Accuracy : 0.8447

       'Positive' Class : 0
```

```
model.glm.confusion <- confusionMatrix(model.lr,training$poverty)
model.glm.confusion1 <- confusionMatrix(model.lr.test,testing$poverty)
print(model.glm.confusion)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 3785  717
         1  943 9118

               Accuracy : 0.886
                 95% CI : (0.8807, 0.8911)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7368

 Mcnemar's Test P-Value : 3.344e-08

            Sensitivity : 0.8005
            Specificity : 0.9271
         Pos Pred Value : 0.8407
         Neg Pred Value : 0.9063
             Prevalence : 0.3247
         Detection Rate : 0.2599
   Detection Prevalence : 0.3091
      Balanced Accuracy : 0.8638

       'Positive' Class : 0
```

```
print(model.glm.confusion1)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1243  244
         1  333 3034

               Accuracy : 0.8811
                 95% CI : (0.8717, 0.8901)
    No Information Rate : 0.6753
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7249

 Mcnemar's Test P-Value : 0.0002488

            Sensitivity : 0.7887
            Specificity : 0.9256
         Pos Pred Value : 0.8359
         Neg Pred Value : 0.9011
             Prevalence : 0.3247
         Detection Rate : 0.2561
   Detection Prevalence : 0.3063
      Balanced Accuracy : 0.8571

       'Positive' Class : 0
```

Univariate Analysis

```
hr_a <- hr_df |>
  select(cooking.fuel,Electricity, Television, Mobile.phone, Landline, Refrigerator, separate.kitche
  mutate_all(as.numeric, as.factor) |>
  mutate(across(1:7,as.factor)) |>
  tbl_summary()


skimr::skim(hrml1) %>%
select(-c( n_missing,complete_rate)) %>%
filter(skim_variable != "poverty")
```

Table 3: Data summary

Name	hrml1
Number of rows	19417
Number of columns	10
Column type frequency:	

factor	7
numeric	2
Group variables	None

Variable type: factor

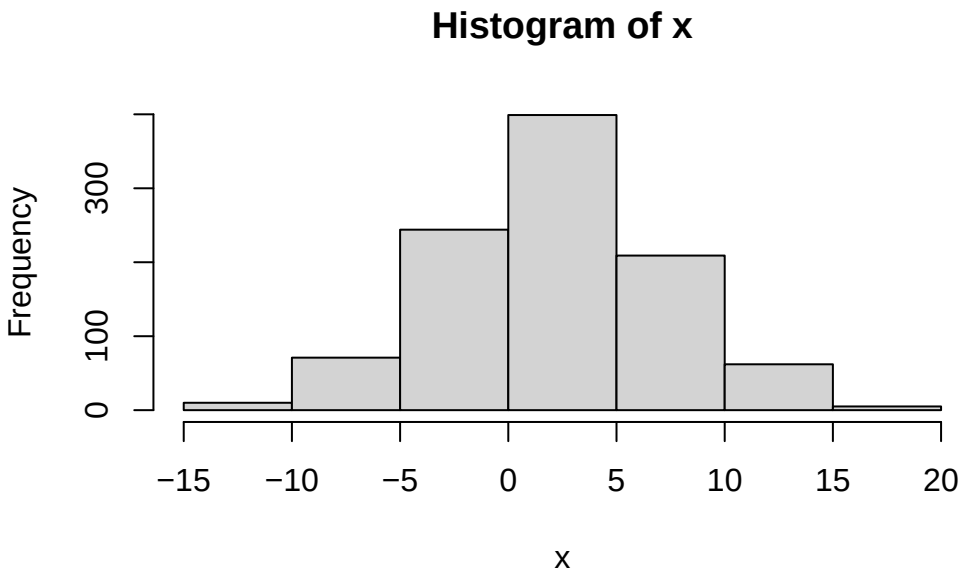
skim_variable	ordered	n_unique	top_counts
residence	FALSE	2	0: 12333, 1: 7084
sex	FALSE	2	1: 16434, 0: 2983
education	FALSE	5	1: 6303, 0: 5408, 2: 4996, 3: 2694
marital.status	FALSE	2	1: 17600, 0: 1817
work.status	FALSE	2	1: 16603, 0: 2814
Wealth.index	FALSE	5	5: 4095, 1: 4072, 2: 3833, 4: 3789
Division	FALSE	8	3: 2904, 2: 2641, 6: 2561, 4: 2511

Variable type: numeric

skim_variable	mean	sd	p0	p25	p50	p75	p100	hist
Family.size	4.62	2.11	1	3	4	6	30	
age	45.72	14.33	15	35	45	55	95	

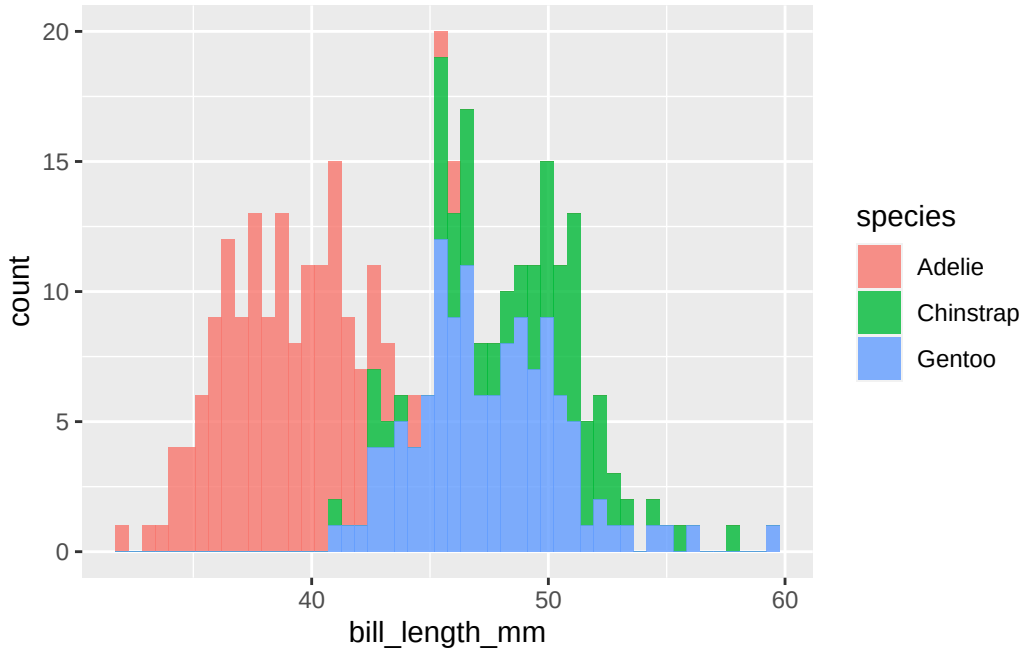
Generate data from Normal Distribution

```
x <- rnorm(1000,2,5)
hist(x)
```



```
penguins |>
  ggplot(aes(x= bill_length_mm, fill = species))+
  geom_histogram(bins = 50, alpha=0.8)
```

```
Warning: Removed 2 rows containing non-finite values (`stat_bin()`).
```

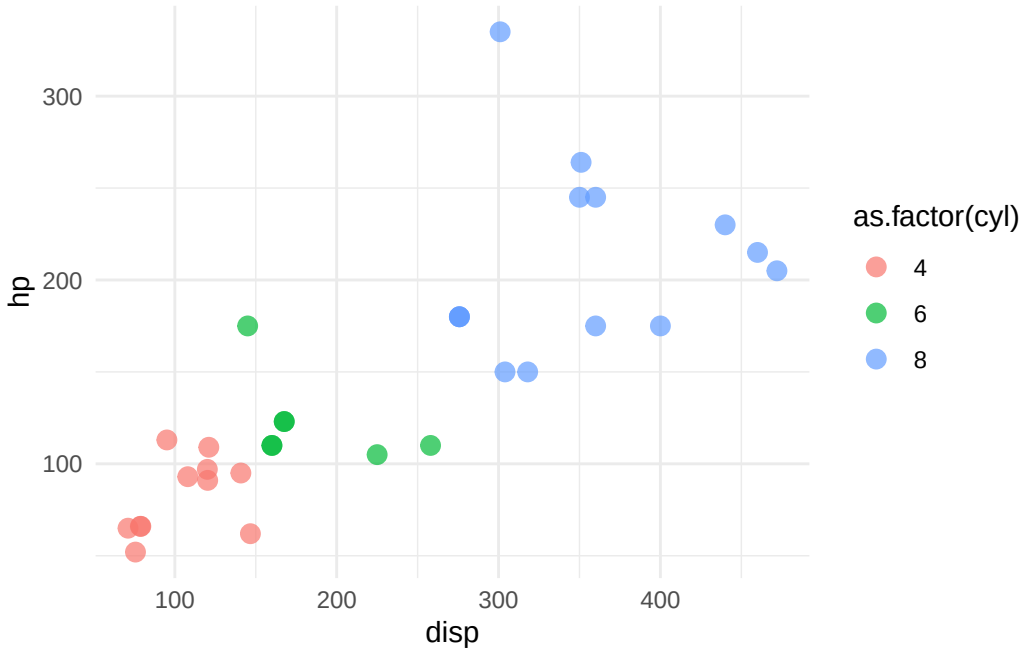


Data Cleaning

```
head(mtcars)
```

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

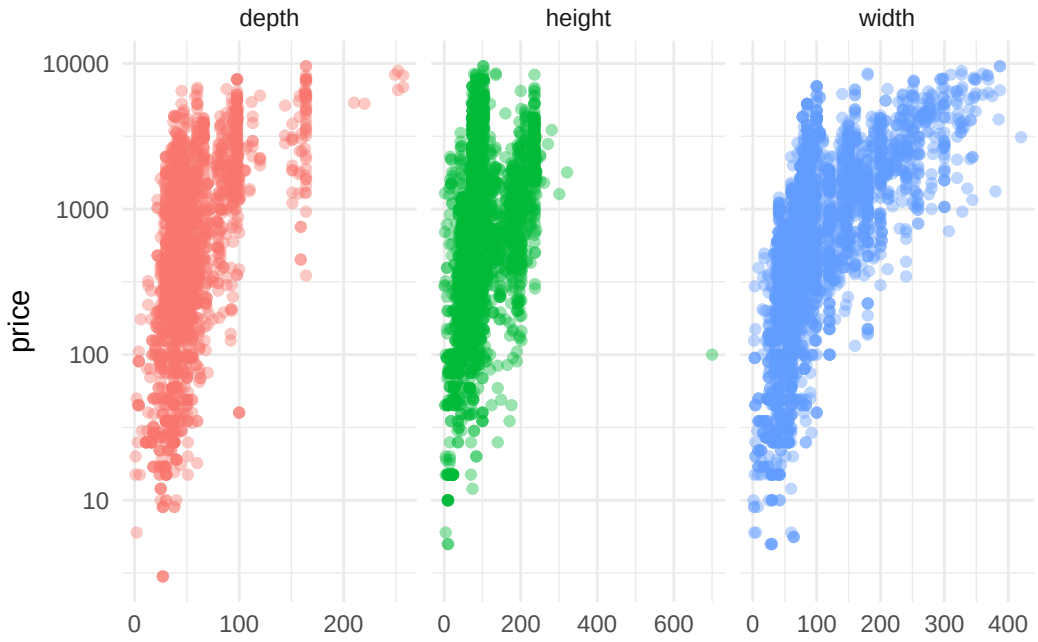
```
ggplot(mtcars,aes(x= disp,y=hp,col=as.factor(cyl)))+
  geom_point(alpha=0.7,size=3)+
  theme_minimal()
```




```
library(tidyverse)
ikea <- read_csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2020/202
```

```
New names:
Rows: 3694 Columns: 14
-- Column specification
-------------------------------------------------------- Delimiter: "," chr
(7): name, category, old_price, link, other_colors, short_description, d... dbl
(6): ...1, item_id, price, depth, height, width lgl (1): sellable_online
i Use `spec()` to retrieve the full column specification for this data. i
Specify the column types or set `show_col_types = FALSE` to quiet this message.
* `` -> `...1`
```

```
ikea <- rename(ikea, id = ...1)

ikea %>%
  select(id, price, depth:width) %>%
  pivot_longer(depth:width, names_to = "dim") %>%
  ggplot(aes(value, price, color = dim)) +
  geom_point(alpha = 0.4, show.legend = FALSE) +
  scale_y_log10() +
  facet_wrap(~dim, scales = "free_x") +
  labs(x = NULL) +
  theme_minimal()
```




```
ikea_df <- ikea %>%
  select(price, name, category, depth, height, width) %>%
  mutate(price = log10(price)) %>%
  mutate_if(is.character, factor)

ikea_df
```

```
# A tibble: 3,694 x 6
   price name                  category      depth height width
   <dbl> <fct>                 <fct>         <dbl>  <dbl> <dbl>
 1  2.42 FREKVENS              Bar furniture    NA     99    51
 2  3.00 NORDVIKEN             Bar furniture    NA    105    80
 3  3.32 NORDVIKEN / NORDVIKEN Bar furniture    NA     NA    NA
 4  1.84 STIG                  Bar furniture    50    100    60
 5  2.35 NORBERG               Bar furniture    60     43    74
 6  2.54 INGOLF                Bar furniture    45     91    40
 7  2.11 FRANKLIN              Bar furniture    44     95    50
 8  2.29 DALFRED               Bar furniture    50     NA    50
 9  2.11 FRANKLIN              Bar furniture    44     95    50
10  3.34 EKEDALEN / EKEDALEN   Bar furniture    NA     NA    NA
# i 3,684 more rows
```

#Building Model

```
## Build Model


library(tidymodels)
```

```
-- Attaching packages ------------------------------------- tidymodels 1.1.1 --

v broom        1.0.5     v rsample      1.2.0
v dials        1.2.0     v tune         1.1.2
v infer        1.0.5     v workflows    1.1.3
v modeldata    1.2.0     v workflowsets 1.0.1
v parsnip      1.1.1     v yardstick    1.2.0
v recipes      1.0.8

-- Conflicts ---------------------------------------- tidymodels_conflicts() --
x recipes::all_double()    masks gtsummary::all_double()
x recipes::all_factor()    masks gtsummary::all_factor()
x recipes::all_integer()   masks gtsummary::all_integer()
x recipes::all_logical()   masks gtsummary::all_logical()
x recipes::all_numeric()   masks gtsummary::all_numeric()
x scales::alpha()          masks kernlab::alpha(), ggplot2::alpha()
x kernlab::cross()         masks purrr::cross()
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 caret::lift()            masks purrr::lift()
x yardstick::precision()   masks caret::precision()
x yardstick::recall()      masks caret::recall()
x yardstick::sensitivity() masks caret::sensitivity()
x yardstick::spec()        masks readr::spec()
x yardstick::specificity() masks caret::specificity()
x recipes::step()          masks stats::step()
* Learn how to get started at https://www.tidymodels.org/start/
```

```r
set.seed(123)
ikea_split <- initial_split(ikea_df, strata = price)
ikea_train <- training(ikea_split)
ikea_test <- testing(ikea_split)

set.seed(234)
ikea_folds <- bootstraps(ikea_train, strata = price)
ikea_folds
```

```
# Bootstrap sampling using stratification
# A tibble: 25 x 2
   splits             id
   <list>             <chr>
 1 <split [2770/994]>  Bootstrap01
 2 <split [2770/1003]> Bootstrap02
 3 <split [2770/1037]> Bootstrap03
 4 <split [2770/1010]> Bootstrap04
```

```
 5 <split [2770/1014]> Bootstrap05
 6 <split [2770/1007]> Bootstrap06
 7 <split [2770/1036]> Bootstrap07
 8 <split [2770/1016]> Bootstrap08
 9 <split [2770/1021]> Bootstrap09
10 <split [2770/1043]> Bootstrap10
# i 15 more rows
```

```
  library(usemodels)
  use_ranger(price ~ ., data = ikea_train)
```

```
ranger_recipe <-
  recipe(formula = price ~ ., data = ikea_train)

ranger_spec <-
  rand_forest(mtry = tune(), min_n = tune(), trees = 1000) %>%
  set_mode("classification") %>%
  set_engine("ranger")

ranger_workflow <-
  workflow() %>%
  add_recipe(ranger_recipe) %>%
  add_model(ranger_spec)

set.seed(67013)
ranger_tune <-
  tune_grid(ranger_workflow, resamples = stop("add your rsample object"), grid = stop("add number of c
```

```
  ## lots of options, like use_xgboost, use_glmnet, etc


  library(textrecipes)
  ranger_recipe <-
    recipe(formula = price ~ ., data = ikea_train) %>%
    step_other(name, category, threshold = 0.01) %>%
    step_clean_levels(name, category) %>%
    step_impute_knn(depth, height, width)

  ranger_spec <-
    rand_forest(mtry = tune(), min_n = tune(), trees = 1000) %>%
    set_mode("regression") %>%
    set_engine("ranger")

  ranger_workflow <-
    workflow() %>%
    add_recipe(ranger_recipe) %>%
    add_model(ranger_spec)
```

```r
set.seed(8577)
doParallel::registerDoParallel()
ranger_tune <-
  tune_grid(ranger_workflow,
    resamples = ikea_folds,
    grid = 11
  )
```

```
i Creating pre-processing data to finalize unknown parameter: mtry
```

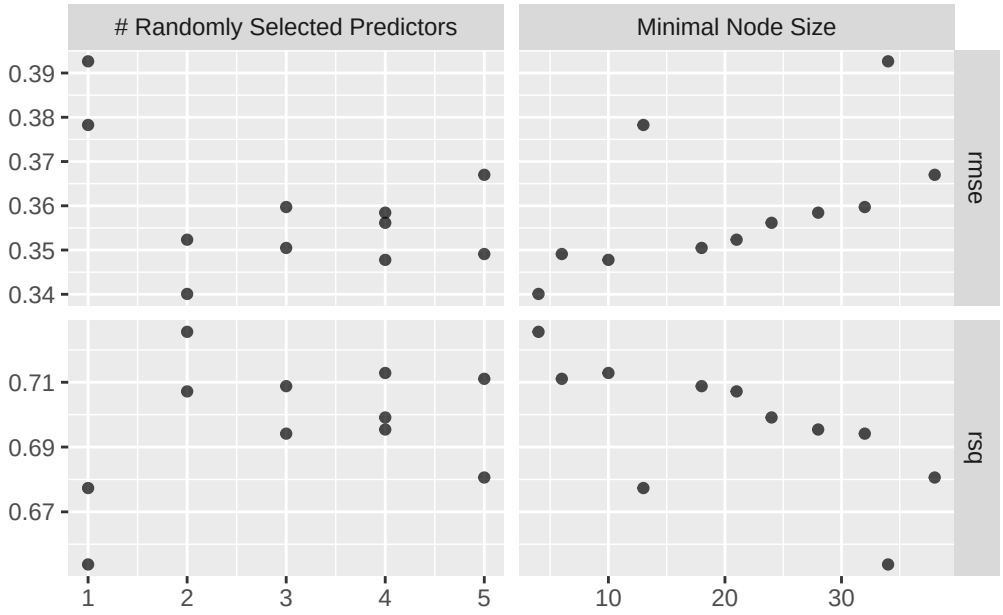
```r
show_best(ranger_tune, metric = "rmse")
```

```
# A tibble: 5 x 8
   mtry min_n .metric .estimator  mean     n std_err .config
  <int> <int> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1     2     4 rmse    standard   0.340    25 0.00203 Preprocessor1_Model10
2     4    10 rmse    standard   0.348    25 0.00226 Preprocessor1_Model05
3     5     6 rmse    standard   0.349    25 0.00235 Preprocessor1_Model06
4     3    18 rmse    standard   0.350    25 0.00218 Preprocessor1_Model01
5     2    21 rmse    standard   0.352    25 0.00200 Preprocessor1_Model08
```

```r
show_best(ranger_tune, metric = "rsq")
```

```
# A tibble: 5 x 8
   mtry min_n .metric .estimator  mean     n std_err .config
  <int> <int> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1     2     4 rsq     standard   0.726    25 0.00332 Preprocessor1_Model10
2     4    10 rsq     standard   0.713    25 0.00372 Preprocessor1_Model05
3     5     6 rsq     standard   0.711    25 0.00385 Preprocessor1_Model06
4     3    18 rsq     standard   0.709    25 0.00368 Preprocessor1_Model01
5     2    21 rsq     standard   0.707    25 0.00347 Preprocessor1_Model08
```

```r
autoplot(ranger_tune)
```




```
final_rf <- ranger_workflow %>%
  finalize_workflow(select_best(ranger_tune))
```

```
Warning: No value of `metric` was given; metric 'rmse' will be used.
```

```
final_rf
```

```
== Workflow ====================================================================
Preprocessor: Recipe
Model: rand_forest()

-- Preprocessor ----------------------------------------------------------------
3 Recipe Steps

* step_other()
* step_clean_levels()
* step_impute_knn()

-- Model -----------------------------------------------------------------------
Random Forest Model Specification (regression)

Main Arguments:
  mtry = 2
  trees = 1000
  min_n = 4

Computational engine: ranger
```

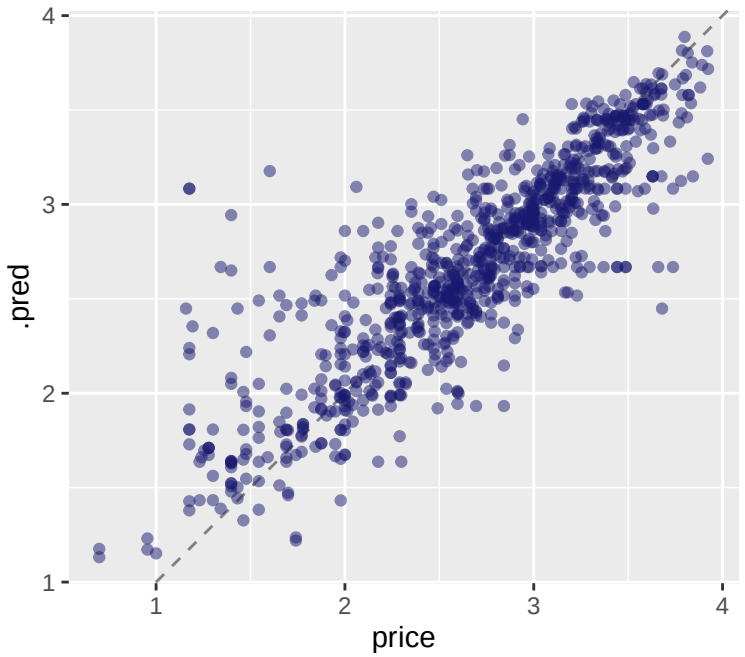
```r
ikea_fit <- last_fit(final_rf, ikea_split)
ikea_fit
```

```
# Resampling results
# Manual resampling
# A tibble: 1 x 6
  splits             id               .metrics .notes   .predictions .workflow
  <list>             <chr>            <list>   <list>   <list>       <list>
1 <split [2770/924]> train/test split <tibble> <tibble> <tibble>     <workflow>
```

```r
collect_metrics(ikea_fit)
```

```
# A tibble: 2 x 4
  .metric .estimator .estimate .config
  <chr>   <chr>          <dbl> <chr>
1 rmse    standard       0.318 Preprocessor1_Model1
2 rsq     standard       0.753 Preprocessor1_Model1
```

```r
collect_predictions(ikea_fit) %>%
  ggplot(aes(price, .pred)) +
  geom_abline(lty = 2, color = "gray50") +
  geom_point(alpha = 0.5, color = "midnightblue") +
  coord_fixed()
```




```r
predict(ikea_fit$.workflow[[1]], ikea_test[15, ])
```

```
# A tibble: 1 x 1
  .pred
  <dbl>
1  2.42
```

```
library(vip)
```

```
Attaching package: 'vip'
```

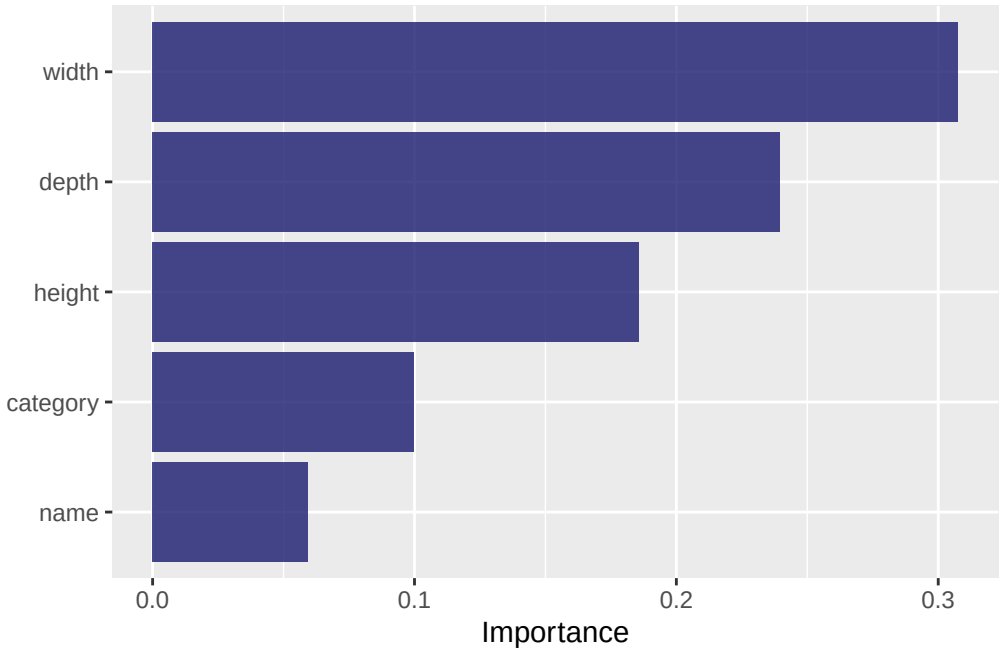
```
The following object is masked from 'package:utils':

    vi
```

```
imp_spec <- ranger_spec %>%
  finalize_model(select_best(ranger_tune)) %>%
  set_engine("ranger", importance = "permutation")
```

```
Warning: No value of `metric` was given; metric 'rmse' will be used.
```

```
workflow() %>%
  add_recipe(ranger_recipe) %>%
  add_model(imp_spec) %>%
  fit(ikea_train) %>%
  pull_workflow_fit() %>%
  vip(aesthetics = list(alpha = 0.8, fill = "midnightblue"))
```

```
Warning: `pull_workflow_fit()` was deprecated in workflows 0.2.3.
i Please use `extract_fit_parsnip()` instead.
```


width
depth
height
category
name
0.0
0.1
0.2
0.3
Importance