Maturity

Javier Martinez-Arribas (javimartinezarribas@gmail.com)

01 diciembre 2022

Contents

Age-Length Keys

As idades dos peixes são dados importantes para a compreensão a dinâmica das populações de peixes. Estimando a idade para um grande número de peixes, no entanto, é trabalhoso. Felizmente, geralmente há uma forte relação entre o comprimento e a idade, e medindo o comprimento de um grande número de peixes é relativamente fácil.

Assim, a estrutura etária para uma grande amostra de peixes pode ser estimado de forma confiável resumindo a relação entre idade e comprimento para uma subamostra de peixes e, em seguida, aplicar este resumo a toda a amostra. O resumo da relação entre idade e comprimento é uma chave de comprimento de idade (ALK).

A tabela de probabilidades condicionais, $p_{j|i} = \frac{n_{ij}}{n_i}$ derivadas da amostra envelhecida é o ALK.

No estudo seguinte tentaremos descobrir a relação entre a idade do peixe e um conjunto diferente de variáveis explicativas como o tamanho, o sexo ou o ano em que a amostra foi recolhida. Tentaremos conhecer esta relação através de vários tipos de modelos, tais como:

- Modelo de Regressão Multinomial
- Modelo de Random Forest
- Modelo de matriz ALK

Nosso primeiro passo será carregar as bibliotecas que vamos usar e depois vamos nos conectar ao banco de dados para carregar os dados relacionados ao peso-tamanho do peixe.

```
load.libraries <- c('DBI','tidyverse','tidymodels','modeltime','timetk','lubridate',
                    'imputeTS','plm','tsibble','fable','FSA','missMethods','nnet',
                    'effects','ranger','vip','tune','randomForest','NeuralNetTools',
                    'mice','data.table')
install.lib <- load.libraries[!load.libraries %in% installed.packages()]

for(libs in install.lib) install.packages(libs, dependences = TRUE)
sapply(load.libraries, require, character = TRUE)
```

```
##            DBI      tidyverse     tidymodels      modeltime          timetk
##           TRUE           TRUE           TRUE           TRUE           TRUE
##      lubridate       imputeTS            plm        tsibble          fable
##           TRUE           TRUE           TRUE           TRUE           TRUE
##            FSA    missMethods           nnet        effects         ranger
##           TRUE           TRUE           TRUE           TRUE           TRUE
##            vip           tune   randomForest NeuralNetTools           mice
##           TRUE           TRUE           TRUE           TRUE           TRUE
##     data.table
##           TRUE
```

e carregar os dados...

```
dbicon <-  DBI::dbConnect(RPostgres::Postgres(),
                          db = "PESCAz",
                          host = "localhost",
                          port = "5432",
                          user = "postgres",
                          password = "1234")
dbListTables(dbicon)
```

```
## [1] "tblmaturity"
```

```
sql <- 'SELECT * FROM "tblmaturity"'
df_maturity <- dbGetQuery(dbicon, sql)


dbDisconnect(dbicon)
```

Em seguida, selecionamos os preditores que vamos utilizar na análise e calculamos o sexo e a idade na primeira maturação.

```
maturity_tbl <- df_maturity %>%
  select(Data, Ano, Mes, Trimestre, Matura, Sexo, MaturaH1, SexoH1, Complf, Especie) %>%
  set_names(c("date", "ano", "mes", "trimestre", "matura", "sexo", "maturaH1",
              "sexoH1", "comprimento", "specie"))

# Criando coluna de sexo reunindo o sexo da primeira matura para M,F ou hermafroditas
maturity_tbl <- maturity_tbl %>% mutate(sexo = ifelse(sexoH1 %in% "", sexo, sexoH1))
# Criando coluna de matura reunindo a matura da primeira matura para M,F ou hermafroditas
maturity_tbl <- maturity_tbl %>% mutate(matura = ifelse(maturaH1 %in% "",
                                                        substr(matura,start=1,stop=1),
                                                        maturaH1))
maturity_tbl$day <- gsub("(.*)[-]", "", maturity_tbl$date)

maturity_tbl <- maturity_tbl %>%
  select(date, ano, mes, trimestre, day, comprimento, sexo, matura, specie)

# Alterar alguns formatos
maturity_tbl <- maturity_tbl %>%
                    mutate(ano = as.numeric(ano),
                           day = as.numeric(day),
                           mes = as.numeric(mes),
```

```
                            trimestre = as.numeric(trimestre),
                            matura = as.factor(matura),
                            specie = as.factor(trimws(specie)))
```

Filtramos as observações com valores da variável sexo ('IND', 'H' e '') e observações com comprimento=0 e matura vazia, uma vez que serão mais difíceis de imputar.

```
print("Frequência de sexo=('', 'IND' ou 'H') em os tipos de matura")
```

```
## [1] "Frequência de sexo=(’’, ’IND’ ou ’H’) em os tipos de matura"
```

```
with(maturity_tbl[maturity_tbl$sexo %in% c('','IND','H'),], table(matura))
```

```
## matura
##       0   1   2   3   4   5   I
## 321 286   0   0   0   1   3   3
```

```
maturity_tbl <- maturity_tbl %>%
                  filter(!sexo %in% c('IND','H',''))
print("Frequência de comprimento=0 em os tipos de matura")
```

```
## [1] "Frequência de comprimento=0 em os tipos de matura"
```

```
with(maturity_tbl[maturity_tbl$comprimento==0,], table(matura))
```

```
## matura
##         0    1    2    3    4    5    I
##   13  723 3516 1585  105   32  332    0
```

```
maturity_tbl <- maturity_tbl %>%
                  filter(!(comprimento==0 & matura==''))


maturity_tbl$matura[maturity_tbl$matura %in% c('I','')] <- NA

# Descartamos os níveis que não são adequados no conjunto de dados:
maturity_tbl$matura <- droplevels(maturity_tbl$matura)
print("Removemos os níveis matura = (' ','I'). Obs frequência com compressão = 0")
```

```
## [1] "Removemos os níveis matura = (’ ’,’I’). Obs frequência com compressão = 0"
```

```
with(maturity_tbl[maturity_tbl$comprimento==0,], table(matura))
```

```
## matura
##    0    1    2    3    4    5
##  723 3516 1585  105   32  332
```

Criamos o preditor de intervalo e selecionamos as observações para a espécie GOR.

```
maturity_tbl <- maturity_tbl %>% mutate(lcat10=lencat(comprimento,w=10))

#Filtramos as observações de la especie "GOR" e a variável "specie"
maturity_tbl <- maturity_tbl %>% filter(specie=='GOR') %>% select(-specie)
```

Separamos as observações com matura=NA. Com esses registros testaremos o modelo que melhor se ajusta aos nossos dados de treinamento e teste:

```
maturity_pred <- maturity_tbl %>%
                  rowid_to_column() %>%
                  filter(is.na(matura))
maturity_train <- maturity_tbl[-as.numeric(maturity_pred$rowid),]
```

Dividimos o conjunto de dados sem NAs atribuídos à variável madura em duas subamostras.

Um será usado para ajuste, train set, e o outro para obter uma avaliação mais realista do ajuste, o test set. O método utilizado será a amostragem estratificada pela variável madura pois parece que há muita diferença entre o número de registros em cada tipo de matura.

75% train / 25% test

```
data_split <- initial_split(maturity_train,
                            prop = 3/4,
                            strata = matura)
train <- training(data_split)
test <- testing(data_split)

#Verificamos a proporção de casos das diferentes maturas
print("Frequência relativa dos diferentes valores para matura no train set")
```

```
## [1] "Frequência relativa dos diferentes valores para matura no train set"
```

```
train%>% select(matura) %>% table()/dim(train)[1]
```

```
## .
##          0          1          2          3          4          5
## 0.08904446 0.42704495 0.31920904 0.02788013 0.04544338 0.09137804
```

```
print("Frequência relativa dos diferentes valores para matura no test set")
```

```
## [1] "Frequência relativa dos diferentes valores para matura no test set"
```

```
test%>% select(matura) %>% table()/dim(test)[1]
```

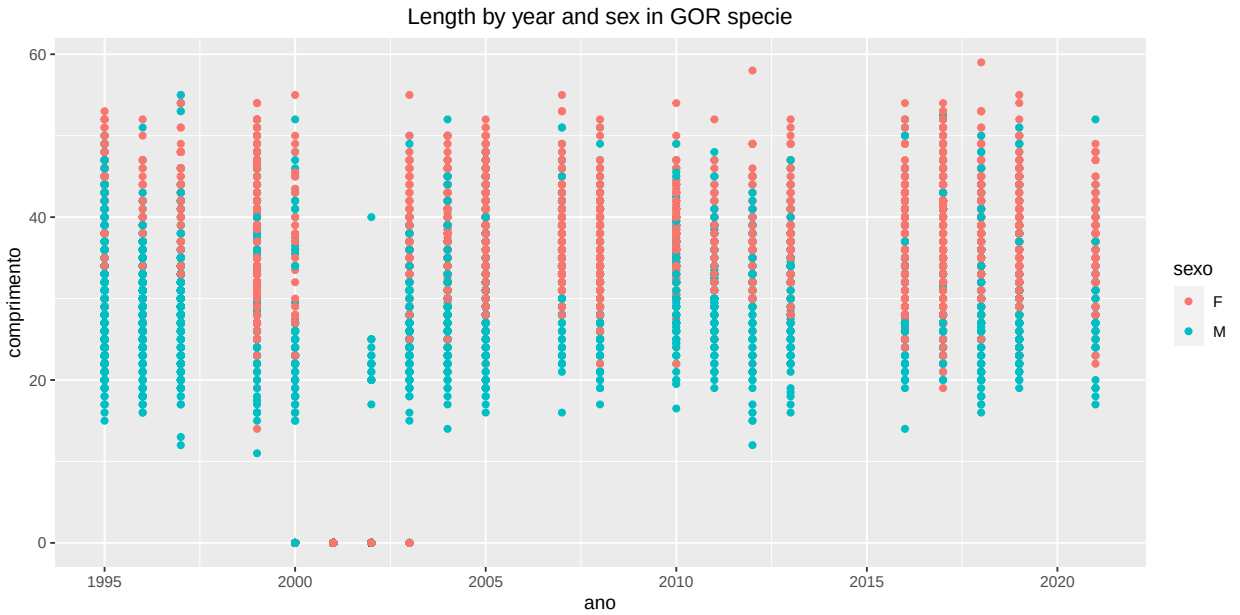
```
## .
##          0          1          2          3          4          5
## 0.08173785 0.42488954 0.33468336 0.02798233 0.04086892 0.08983800
```

Análise exploratória de dados

Primeiramente vamos tentar conhecer um pouco melhor as distribuições de nossas variáveis através dos gráficos a seguir.

Comprimento ao longo dos anos por sexo:

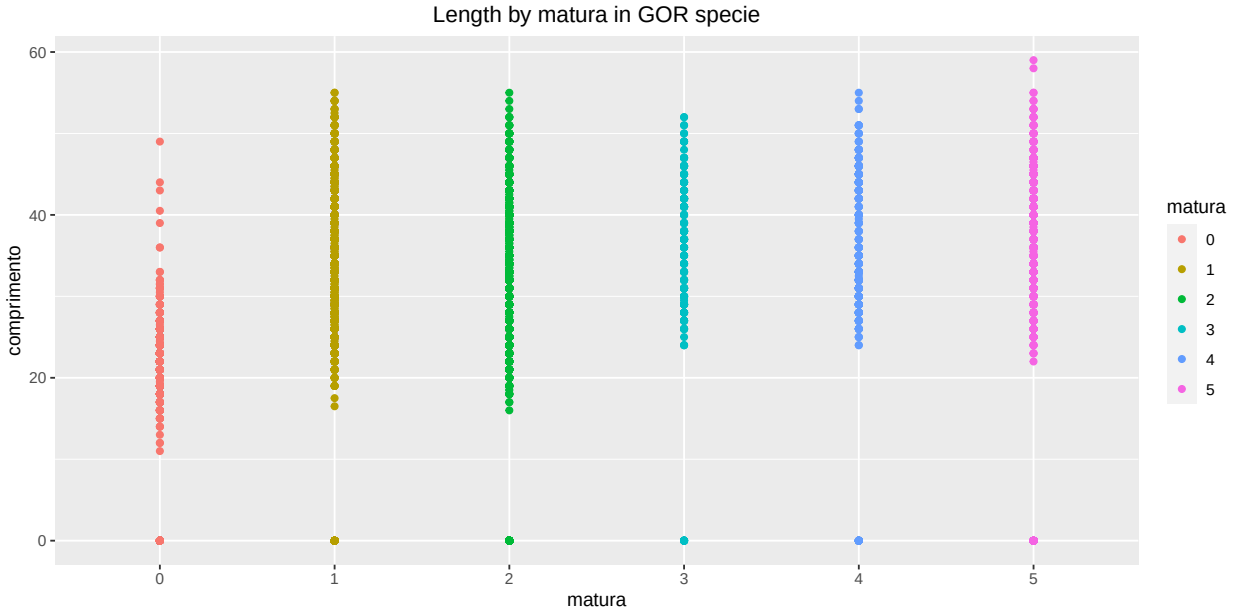
```
ggplot(train, aes(x=ano, y=comprimento, col=sexo)) +
  geom_point() +
  ggtitle("Length by year and sex in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```



Parece que o tamanho das fêmeas é maior que o dos machos em quase todos os anos.

Comprimento por matura:

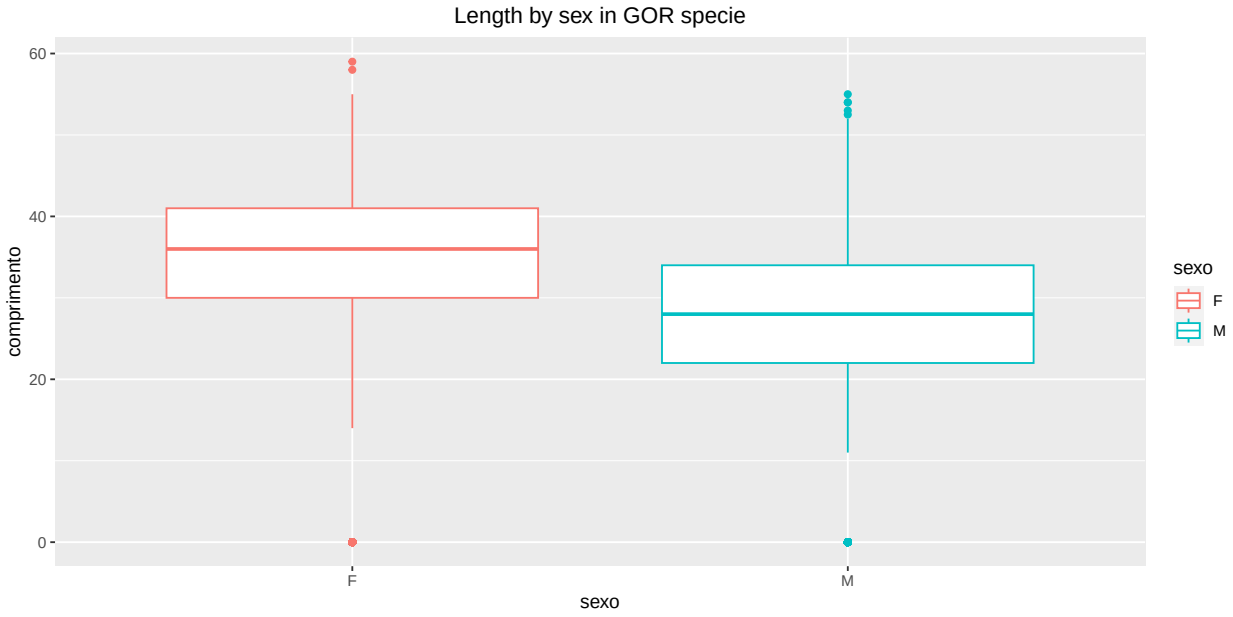
```
ggplot(train, aes(x=matura, y=comprimento, col=matura, group=matura)) +
  geom_point() +
  ggtitle("Length by matura in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```



Como é lógico, o tamanho parece aumentar com a idade, embora pareça estabilizar um pouco após o segundo ano de maturidade.

Comprimento por sexo:

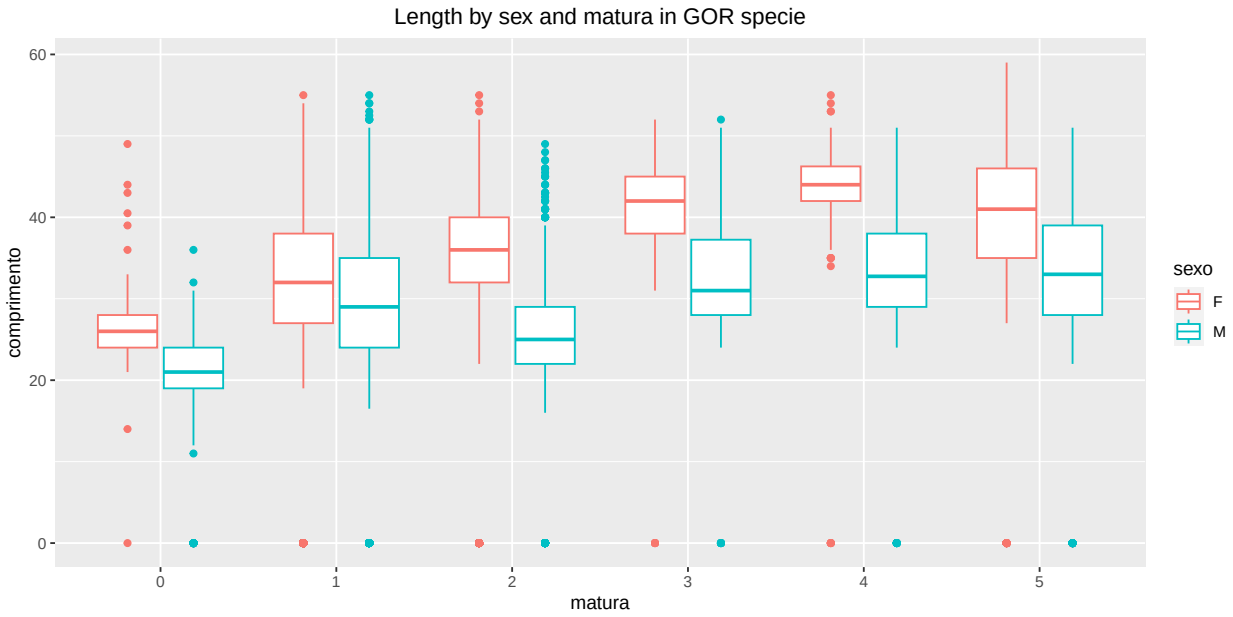
```
ggplot(train, aes(x=sexo, y=comprimento, color=sexo)) +
  geom_boxplot() +
  ggtitle("Length by sex in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```



Mais uma vez parece que a distribuição de tamanho é diferente entre os dois sexos. Sendo maior no sexo feminino.

Comprimento por sexo e matura

```
ggplot(train, aes(x=matura, y=comprimento, color=sexo)) +
  geom_boxplot() +
  ggtitle("Length by sex and matura in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```

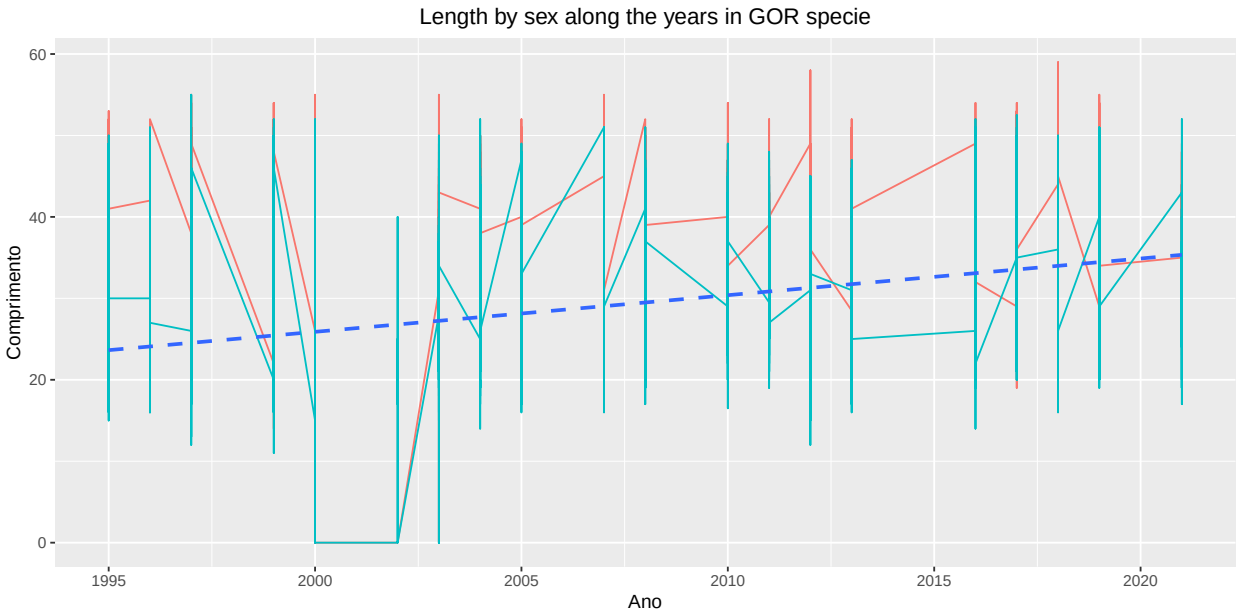


Talvez nas maturas 2 e 4 seja onde há uma maior diferença de tamanho entre os sexos.

Comprimento médio por ano e sexo

```
ggplot(data = arrange(train,date), aes(x = ano, y = comprimento)) +
  geom_line(aes(colour = as.factor(sexo))) +
  geom_smooth(method = "lm", se = F, lty = "dashed") +
  labs(x = "Ano",  y = "Comprimento") +
  theme(legend.position = "none") +
  ggtitle("Length by sex along the years in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```

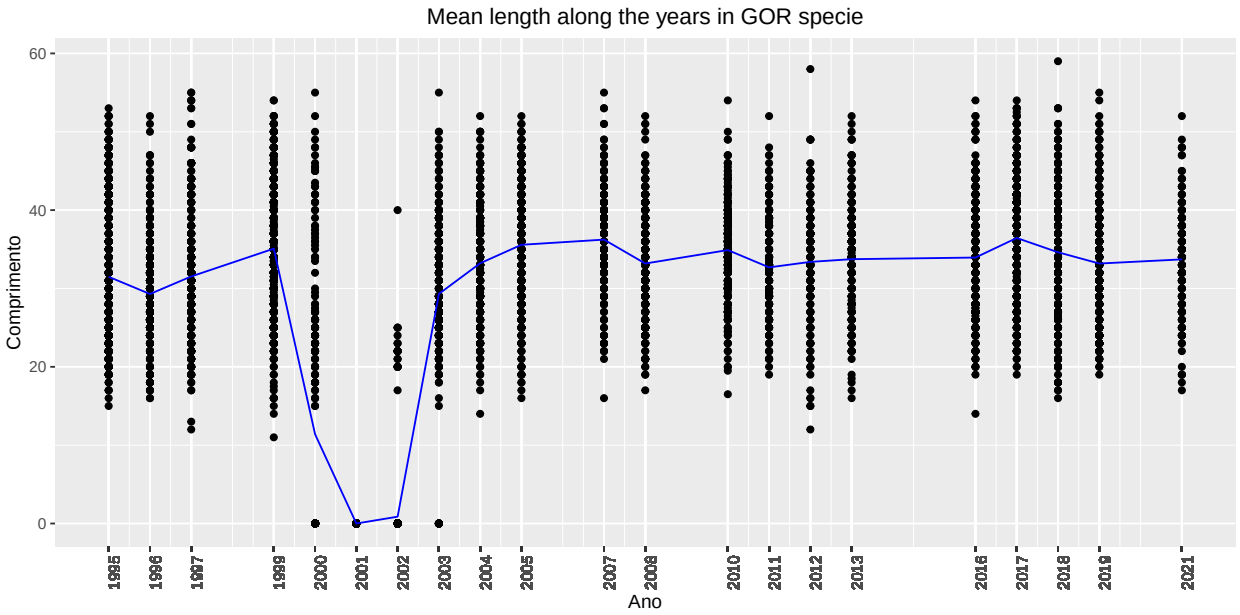
```
## `geom_smooth()` using formula = 'y ~ x'
```



O gráfico mostra um aumento do tamanho médio ao longo dos anos, embora não esteja claro se os machos ou as fêmeas são os principais responsáveis por esse aumento.

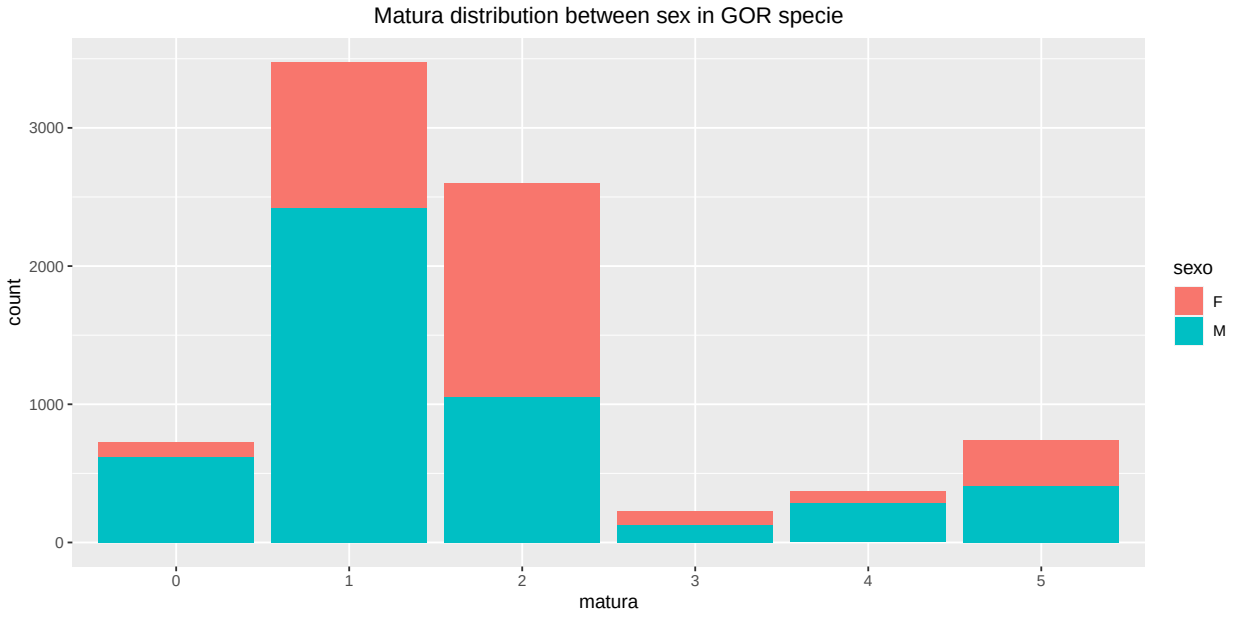
Heterogeneidade ao longo do tempo:

```
#Heterogeneity along the time
train %>%
  group_by(ano) %>%
  summarise(comp_mean = mean(comprimento)) %>%
  left_join(train,by="ano") %>%
  ggplot(data = .,
         aes(x = ano, y = comprimento)) +
  geom_point() +
  geom_line(aes(x = ano, y = comp_mean), col = "blue") +
  scale_x_continuous(labels = as.character(train$ano),
                     breaks = train$ano) +
  labs(x = "Ano", y = "Comprimento") +
  theme(axis.text.x = element_text(angle = 90)) +
  ggtitle("Mean length along the years in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```



Distribuição da matura entre os sexos:

```
ggplot(data=train, aes(x=matura,  fill=sexo)) +
  geom_bar(stat="count") +
  ggtitle("Matura distribution between sex in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```

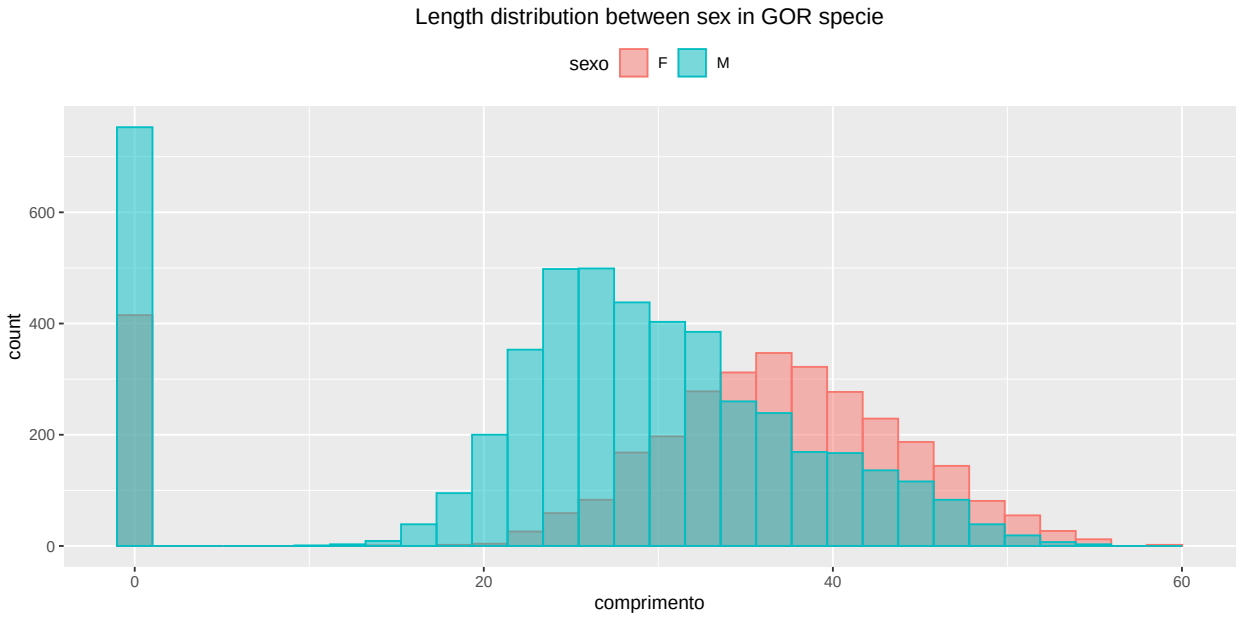


Distribuição do comprimento entre os sexos:

```
# comprimento by sexo distribution
ggplot(data=train, aes(x=comprimento,  fill= sexo, color=sexo)) +
  geom_histogram(position="identity",alpha=0.5)+
  theme(legend.position="top") +
```

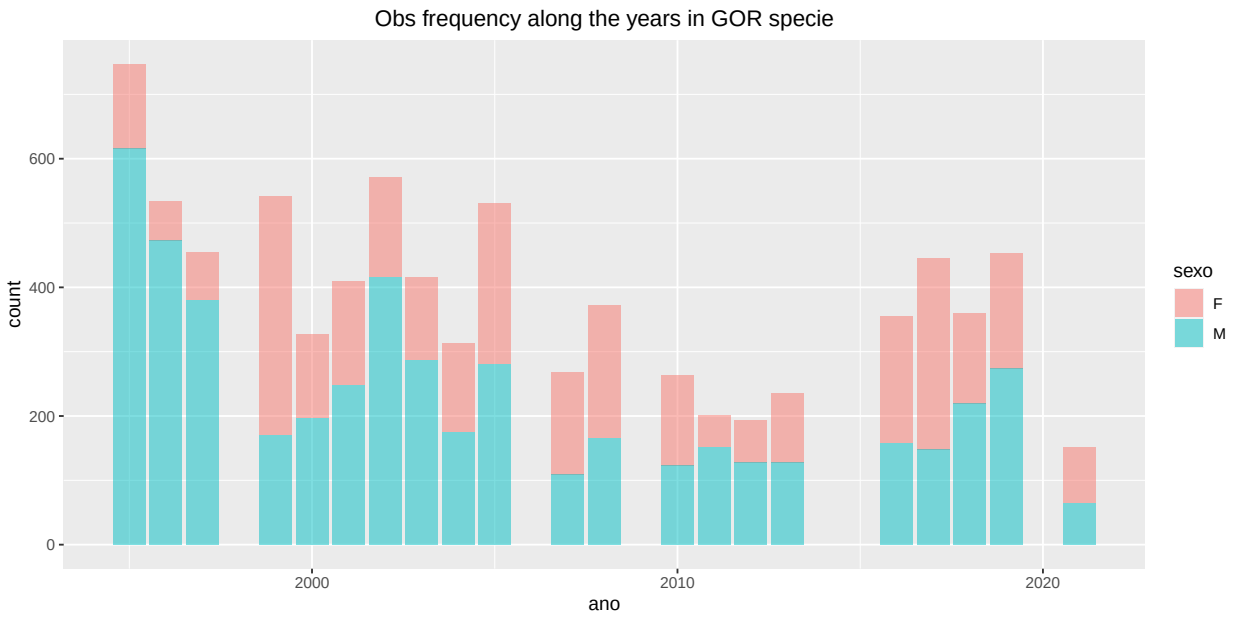
```
  ggtitle("Length distribution between sex in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



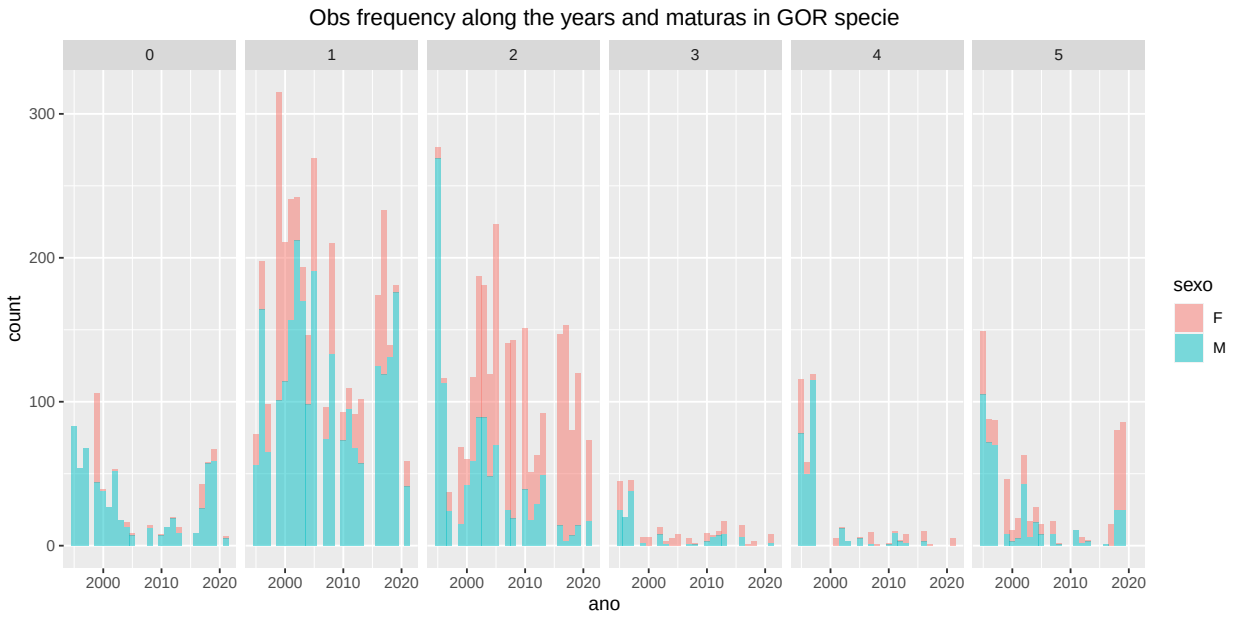
Distribuição do número de observações ao longo do tempo e entre os dois sexos:

```
# Number obs along the time by sexo
ggplot(data = train, aes(x = ano, fill = sexo)) +
  geom_bar(stat="count",alpha=0.5) +
  ggtitle("Obs frequency along the years in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```



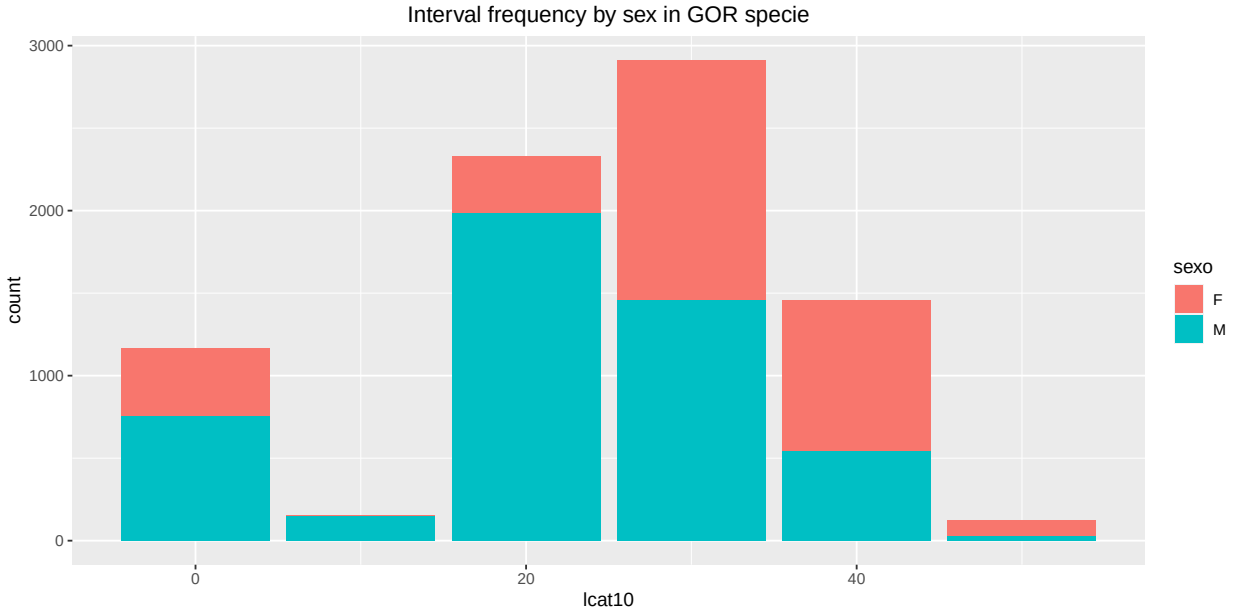
Embora realmente se olharmos para as amostras das diferentes maturas, vemos que há muito poucas maturas amostradas para matura = 0,3 e 4

```
# Number obs along the time by sexo and matura
ggplot(data = train, aes(x = ano, fill = sexo)) +
  geom_bar(stat="count",alpha=0.5) +
  facet_grid(.~matura) +
  ggtitle("Obs frequency along the years and maturas in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```

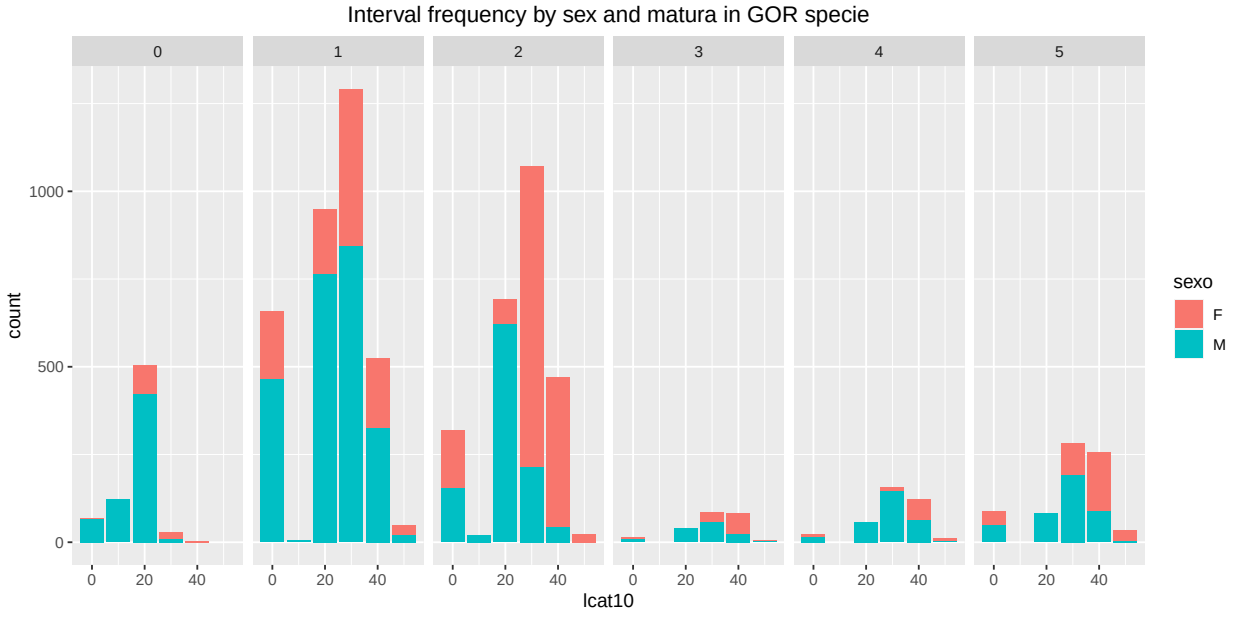


Distribuição dos intervalos entre os sexos:

```
# lcat10 distribution
ggplot(data=train, aes(x=lcat10,  fill=sexo)) +
  geom_bar(stat="count") +
  ggtitle("Interval frequency by sex in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
# lcat10 distribution
ggplot(data=train, aes(x=lcat10,  fill=sexo)) +
  geom_bar(stat="count") +
  facet_grid(.~matura) +
  ggtitle("Interval frequency by sex and matura in GOR specie") +
  theme(plot.title = element_text(hjust = 0.5))
```



Modeling

Em seguida, vamos estudar 3 modelos diferentes de regressão multinomial:

A matura em relação ao intervalo de comprimento:

matura ~ lcat10

A matura em relação ao intervalo de comprimento, sexo e a interação entre esses:

matura ~ lcat10 + sexo + lcat10:sexo

A matura em relação ao intervalo de comprimento, sexo, ano e a interação entre esses:

matura ~ lcat10 + sexo + ano + lcat10:sexo + lcat10:ano + sexo:ano

E veremos se eles são significativamente diferentes um do outro.

Tanto no train set quanto no test set temos muitos registros com compressão=0, algo que não pode acontecer e que pode ter ocorrido por não ter coletado os dados ou não ter salvado.

Usaremos um modelo de regressão multinomial através do pacote tidymodels. Para trabalhar com esses registros vamos imputar um valor para comprimento com a ajuda do algoritmo K-Nearest Neighbourhood com 3 vizinhos.

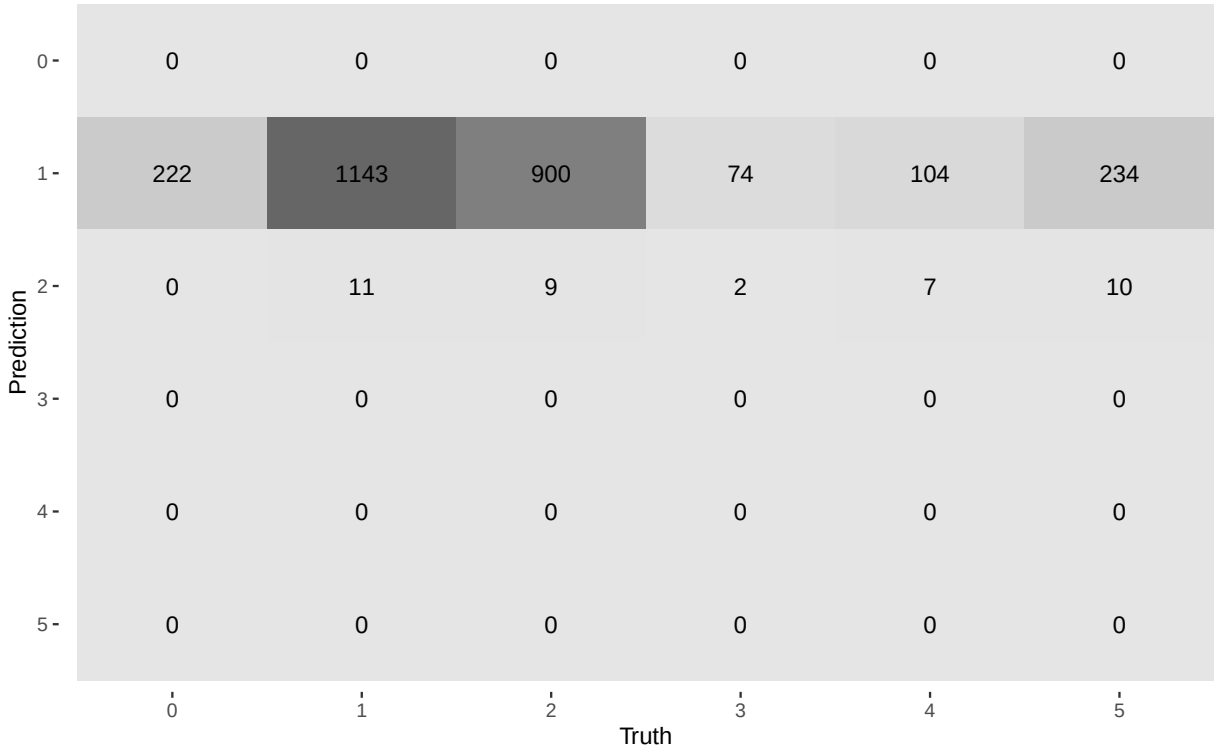
Da mesma forma, vamos tentar escolher o valor ótimo para o hiperparâmetro de "penalty" por meio de validação cruzada. Uma vez conhecido o valor de "penalty" que nos dá o melhor ajuste, com a função last_fit, ajustaremos este modelo ao conjunto de dados completo (train+validation) para posteriormente avaliar seu desempenho no test set.

```
# Nós definimos a fórmula e omitimos qualquer NA que possa existir.
ml1_rec <-
  recipe(matura ~  lcat10,
         data = train) %>%
  step_naomit(everything(), skip = TRUE)

# Definimos o método de treino, K-fold cross-validation com 10 folds.
# Com 9 folds treinamos e com 1 validamos. Repetimos este esquema 10 vezes.
set.seed(1234)
cv_folds <- vfold_cv(train, v = 10)

# Nós definimos o modelo, a biblioteca que contém o algoritmo e o parâmetro que
# queremos otimizar, "penalty".
ml_spec <- multinom_reg(
  mode = "classification",
  engine = "nnet",
  penalty = tune(),
  mixture = NULL
)

# Unimos as especificações anteriores em um fluxo de trabalho (workflow).
ml1_wflow <-
  workflow() %>%
  add_recipe(ml1_rec) %>%
  add_model(ml_spec)

# Lançamos a formação no método de treino previamente definido.
random_tune1 <-
  ml1_wflow %>%
  tune_grid(
    resamples = cv_folds, grid = 5
  )

# Criamos um fluxo de trabalho final com o parâmetro de "penalty" ideal no
```

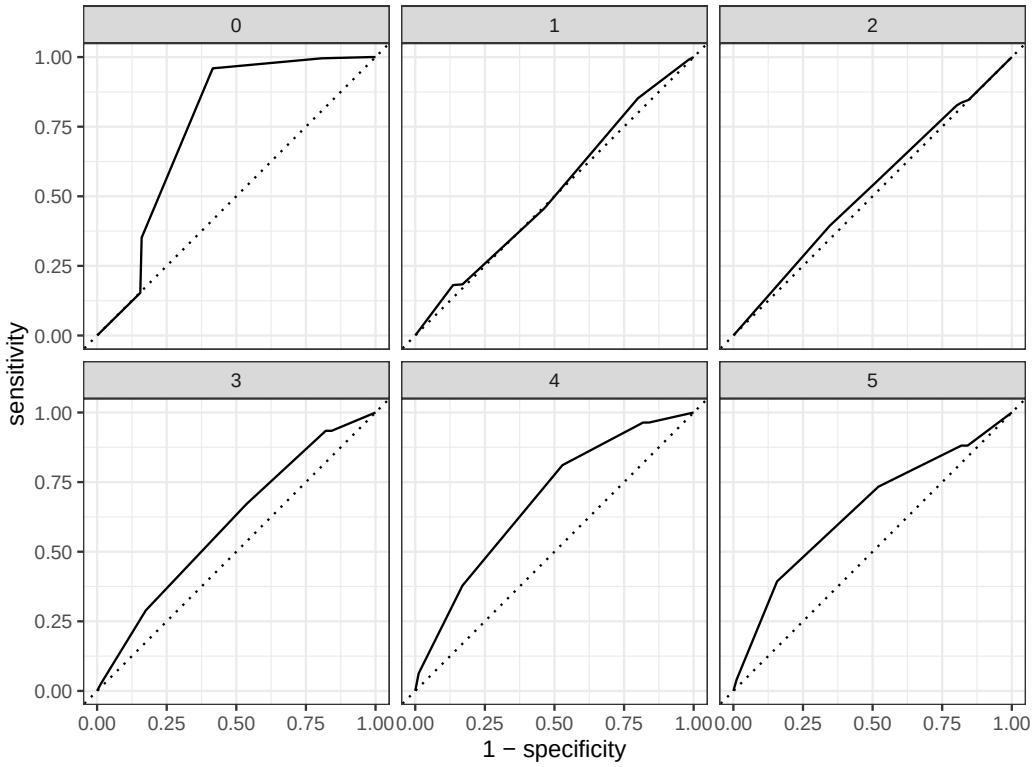
```r
# conjunto de dados do trem total, sem validação cruzada.
random_final1 <-
  finalize_workflow(ml1_wflow, select_best(random_tune1)) %>%
  fit(train)

# Lançamos o treino mas desta vez vamos avaliar no conjunto de teste e não nos
# 10 folds de validação anterior. Usaremos um conjunto de métricas para a avaliação.
last_fit_ml1 <- last_fit(random_final1,
                         split = data_split,
                         metrics = metric_set(
                           precision,roc_auc,
                           sens, spec, f_meas)
)


# Mostramos o resultado das métricas de avaliação.
last_fit_ml1 %>%
  collect_metrics()
```

```
## # A tibble: 5 x 4
##   .metric   .estimator .estimate .config
##   <chr>     <chr>          <dbl> <chr>
## 1 precision macro          0.329 Preprocessor1_Model1
## 2 sens      macro          0.167 Preprocessor1_Model1
## 3 spec      macro          0.834 Preprocessor1_Model1
## 4 f_meas    macro          0.308 Preprocessor1_Model1
## 5 roc_auc   hand_till      0.627 Preprocessor1_Model1
```

```r
# Mostramos a matriz de confusão.
last_fit_ml1 %>%
  collect_predictions() %>%
  conf_mat(matura, .pred_class) %>%
  autoplot(type = "heatmap")
```




```
# E mostramos as curvas ROC. Quanto mais perto do canto superior esquerdo, melhor.
last_fit_ml1 %>%
  collect_predictions() %>%
  roc_curve(matura,'.pred_0','.pred_1','.pred_2','.pred_3','.pred_4','.pred_5') %>%
  autoplot()
```




```
# Ajustamos o modelo final em todo o conjunto de dados. Train + Test sets
final_model1 <-
  random_final1 %>%
  fit(data = maturity_train)
```

O ajuste não é muito bom como podemos ver nas curvas ROC.

Podemos ver que o modelo só prevê matura 1 (42% dos dados), que é para os valores que temos mais registros.

Se calcularmos o expoente dos coeficientes, podemos ver como eles influenciam os odds de matura:

```
exp(coefficients(final_model1$fit$fit$fit))
```

```
##   (Intercept)   lcat10
## 1  2.27199951 1.038969
## 2  1.29632043 1.051176
## 3  0.03738428 1.093571
## 4  0.04507296 1.103471
## 5  0.14059039 1.088352
```

Conclusões:

Aumentar o valor do intervalo de comprimento um nível aumenta em 10% a probabilidade de que a idade do peixe seja 4 em vez de 0

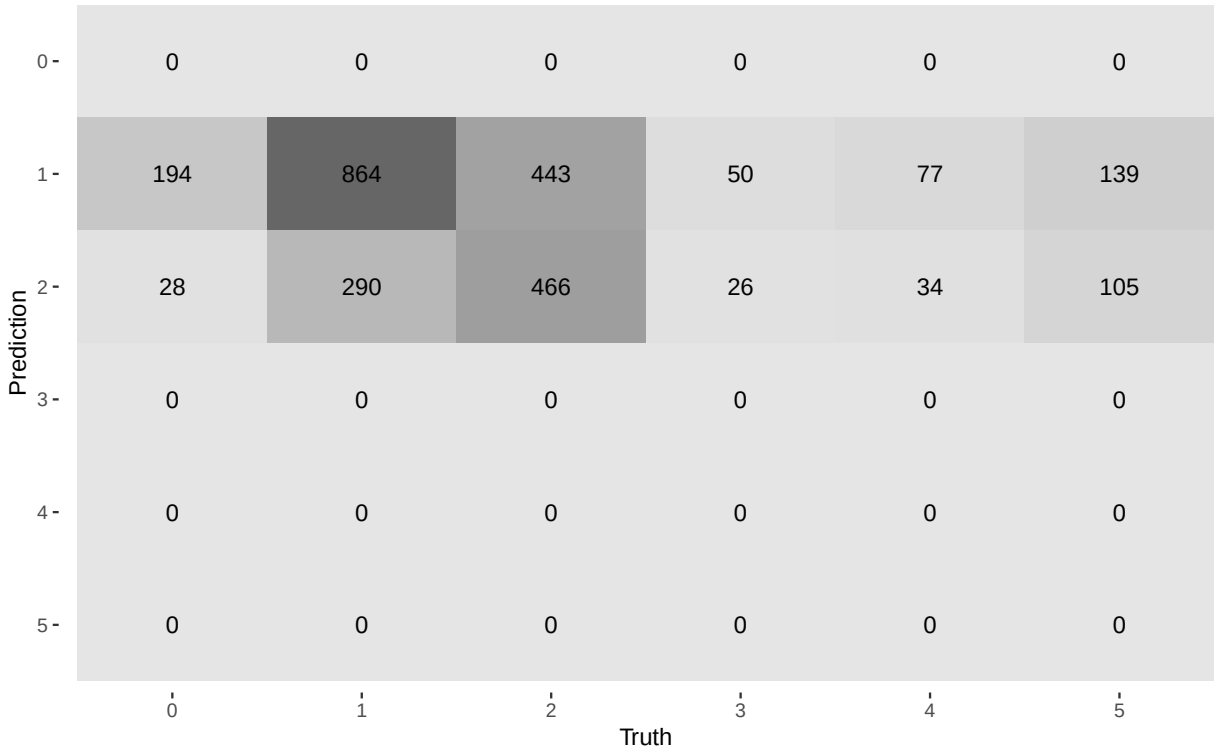
Vejamos o que acontece quando adicionamos a variável sexo e suas iterações com a variável interval lcat10:

```r
ml2_rec <-
  recipe(matura ~  lcat10 + sexo,
         data = train) %>%
  step_naomit(everything(), skip = TRUE) %>%
  step_novel(sexo) %>%
  step_interact(terms = ~ lcat10:sexo)

# K-fold cross-validation
set.seed(1234)
cv_folds <- vfold_cv(train, v = 10)


# ml_spec permanece o mesmo. Modelo de regressão multinomial

ml2_wflow <-
  workflow() %>%
  add_recipe(ml2_rec) %>%
  add_model(ml_spec)


set.seed(100)
random_tune2 <-
  ml2_wflow %>%
  tune_grid(
    resamples = cv_folds, grid = 5
  )

random_final2 <-
  finalize_workflow(ml2_wflow, select_best(random_tune2)) %>%
  fit(train)

last_fit_ml2 <- last_fit(random_final2,
                         split = data_split,
                         metrics = metric_set(
                           precision,roc_auc,
                           sens, spec, f_meas)
)


last_fit_ml2 %>%
  collect_metrics()
```

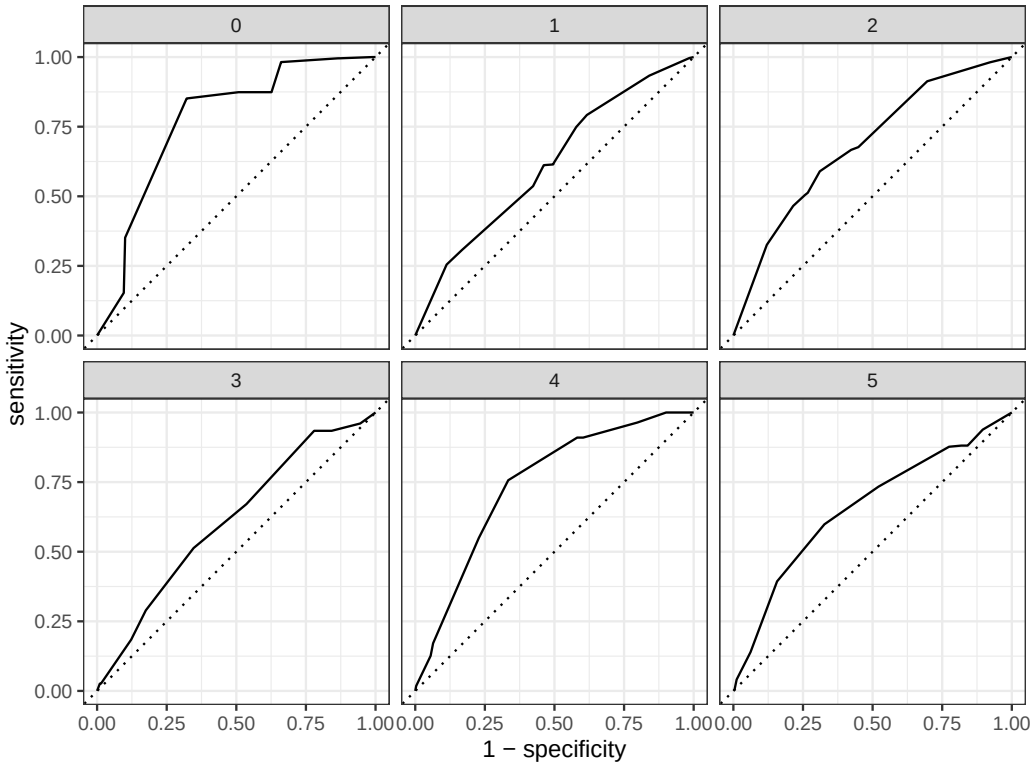
```
## # A tibble: 5 x 4
##   .metric   .estimator .estimate .config
##   <chr>     <chr>          <dbl> <chr>
## 1 precision macro          0.490 Preprocessor1_Model1
## 2 sens      macro          0.210 Preprocessor1_Model1
## 3 spec      macro          0.859 Preprocessor1_Model1
## 4 f_meas    macro          0.547 Preprocessor1_Model1
## 5 roc_auc   hand_till      0.666 Preprocessor1_Model1
```

```r
last_fit_ml2 %>%
  collect_predictions() %>%
```

```
  conf_mat(matura, .pred_class) %>%
  autoplot(type = "heatmap")
```




```
last_fit_ml2 %>%
  collect_predictions() %>%
  roc_curve(matura,'.pred_0','.pred_1','.pred_2','.pred_3','.pred_4','.pred_5') %>%
  autoplot()
```




```
# Ajustamos o modelo final em todo o conjunto de dados.
final_model2 <-
  random_final2 %>%
  fit(data = maturity_train)
```

Melhor do que antes, mas o ajuste também não parece muito exato. Agora conseguimos que o modelo também preveja matura 1 ou 2, que ainda são as duas classes mais abundantes nos dados.

Se calcularmos o expoente dos coeficientes, podemos ver como eles influenciam os odds de matura:

```
exp(coefficients(final_model2$fit$fit$fit))
```

```
##   (Intercept)   lcat10     sexoM sexonew lcat10_x_sexoM lcat10_x_sexonew
## 1 7.457834917 1.014884 0.2097217       1      1.0341585                 1
## 2 6.026233215 1.037671 0.1855853       1      0.9885742                 1
## 3 0.071842178 1.092206 0.4520430       1      0.9989214                 1
## 4 0.008514901 1.156910 4.7725325       1      0.9599819                 1
## 5 0.426651311 1.075451 0.2565203       1      1.0109335                 1
```

Conclusões:

Ao aumentar lcat10 um nível aumentará a probabilidade de ter um matura 4 em vez de um 0 um 14%.

Parece que um matura 4 é quase 3 vezes mais provável de ocorrer em peixes machos do que nas fêmeas.

O aumento de lcat10 em um nível parece afetar a maturidade da mesma forma em ambos os sexos M como em F.

Um teste de razão de verossimilhança para identificar uma diferença entre dois grupos requer o ajuste de dois modelos multinomiais. O primeiro modelo mais simples tem intervalo de comprimento como a única

variável explicativa. O segundo modelo mais complexo tem intervalo de comprimento, a variável fator que identifica os grupos, e a interação entre essas duas variáveis como variáveis explicativas.

```
anova(final_model1$fit$fit$fit,final_model2$fit$fit$fit)
```

```
## Likelihood ratio tests of Multinomial Models
##
## Response: ..y
##                                                 Model Resid. df Resid. Dev   Test
## 1                                              lcat10     54280   29555.41
## 2 lcat10 + sexo + lcat10_x_sexoM + lcat10_x_sexonew     54270   28380.80 1 vs 2
##      Df LR stat. Pr(Chi)
## 1
## 2    10 1174.608       0
```

Os dois modelos são estatisticamente diferentes, então a distribuição do tamanhos dentro de cada idade é diferente para cada sexo.

Se agora adicionarmos o ano:

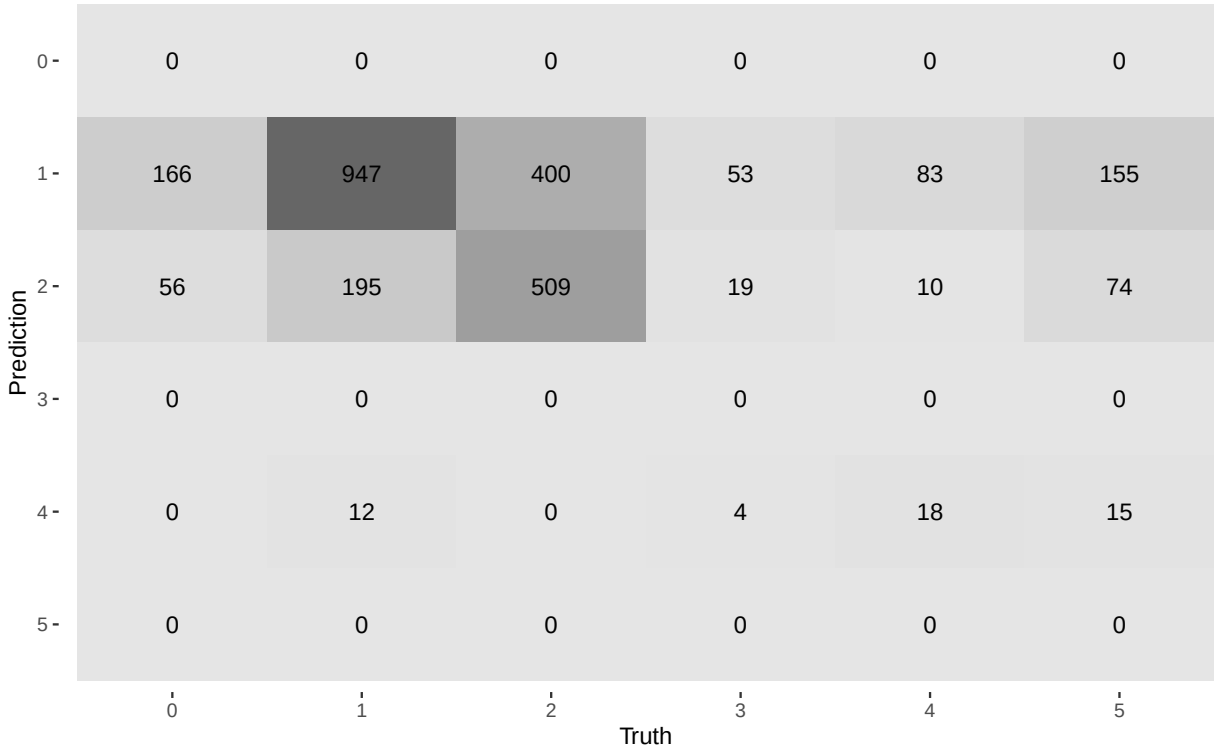
Comprimento de primeira maturação por sexo e por ano bem como todas as iterações de variáveis de dois por dois que podem ser obtidas:

```
ml3_rec <-
  recipe(matura ~  lcat10 + sexo + ano,
         data = train) %>%
  step_naomit(everything(), skip = TRUE) %>%
  step_novel(sexo) %>%
  step_interact(~all_predictors() * all_predictors())


# ml_spec permanece o mesmo. Modelo de regressão multinomial
ml3_wflow <-
  workflow() %>%
  add_recipe(ml3_rec) %>%
  add_model(ml_spec)

random_tune3 <-
  ml3_wflow %>%
  tune_grid(
    resamples = cv_folds, grid = 5
  )

random_final3 <-
  finalize_workflow(ml3_wflow, select_best(random_tune3)) %>%
  fit(train)

last_fit_ml3 <- last_fit(random_final3,
                         split = data_split,
                         metrics = metric_set(
                           precision,roc_auc,
                           sens, spec, f_meas)
)
```

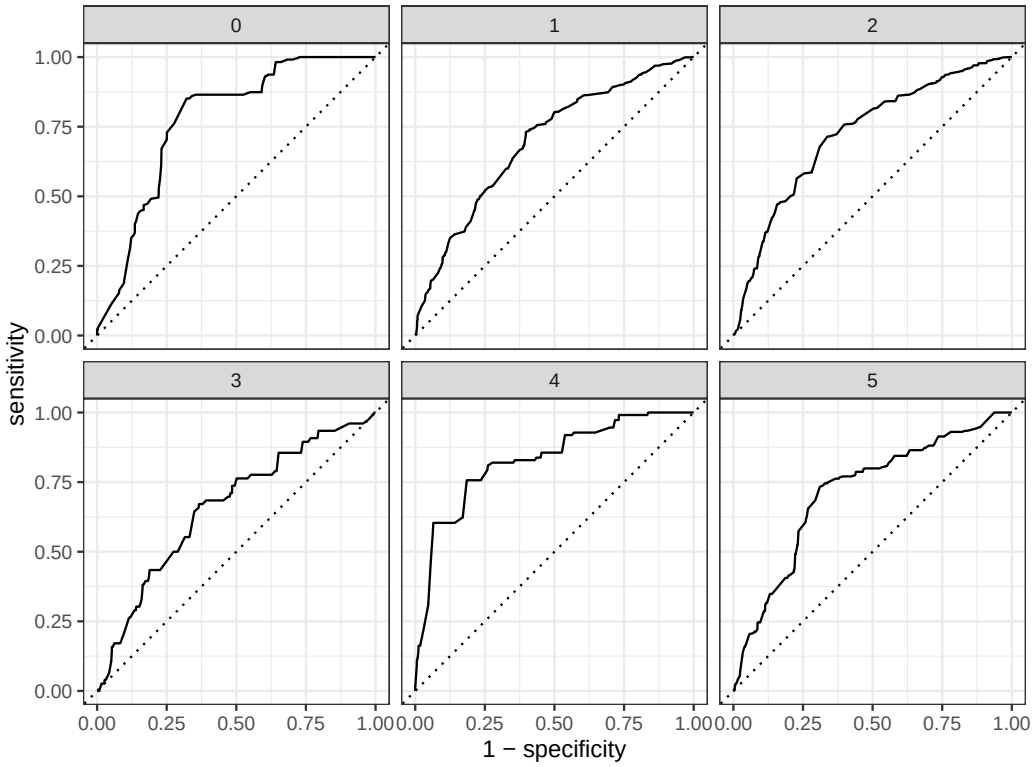
```
last_fit_ml3 %>%
  collect_metrics()
```

```
## # A tibble: 5 x 4
##   .metric   .estimator .estimate .config
##   <chr>     <chr>          <dbl> <chr>
## 1 precision macro          0.494 Preprocessor1_Model1
## 2 sens      macro          0.257 Preprocessor1_Model1
## 3 spec      macro          0.874 Preprocessor1_Model1
## 4 f_meas    macro          0.480 Preprocessor1_Model1
## 5 roc_auc   hand_till      0.710 Preprocessor1_Model1
```

```
last_fit_ml3 %>%
  collect_predictions() %>%
  conf_mat(matura, .pred_class) %>%
  autoplot(type = "heatmap")
```




```
last_fit_ml3 %>%
  collect_predictions() %>%
  roc_curve(matura,'.pred_0','.pred_1','.pred_2','.pred_3','.pred_4','.pred_5') %>%
  autoplot()
```




```
# Ajustamos o modelo final em todo o conjunto de dados ee estudamos os coeficientes:
final_model3 <-
  random_final3 %>%
  fit(data = maturity_train)
```

```
exp(coefficients(final_model3$fit$fit$fit))
```

```
##     (Intercept)       lcat10         sexoM sexonew       ano lcat10_x_sexoM
## 1  3.564817e+92 3.161522e-04  2.412633e-32       1 0.8999022      1.0264457
## 2 6.345121e+113 6.096598e-07  1.259214e+88       1 0.8780712      1.0074626
## 3 4.064925e+117 6.855343e-05  1.318453e+54       1 0.8723398      1.0085969
## 4  1.590440e+26 1.523829e+00 1.845549e+117       1 0.9683169      0.9765676
## 5 9.969427e+210 3.331732e-08  5.845640e+75       1 0.7842423      1.0325826
##   lcat10_x_sexonew lcat10_x_ano sexoM_x_ano sexonew_x_ano
## 1                1     1.004041   1.0362610             1
## 2                1     1.007185   0.9028665             1
## 3                1     1.004844   0.9392018             1
## 4                1     0.999854   0.8741529             1
## 5                1     1.008670   0.9156959             1
```

Conclusões:

- O ajuste melhora um pouco, talvez um pouco mais para peixes de maturidade 4. A chance de ser matura 4 aumenta em mais de 50% ao mudar de um nível de lcat10 para outro.

Um teste de razão de verossimilhança:

```
anova(final_model2$fit$fit$fit,final_model3$fit$fit$fit)
```

```
## Likelihood ratio tests of Multinomial Models
##
## Response: ..y
##                                                                                                    Mode
## 1                                                     lcat10 + sexo + lcat10_x_sexoM + lcat10_x_sexone
## 2 lcat10 + sexo + ano + lcat10_x_sexoM + lcat10_x_sexonew + lcat10_x_ano + sexoM_x_ano + sexonew_x_an
##   Resid. df Resid. Dev   Test    Df LR stat. Pr(Chi)
## 1     54270   28380.80
## 2     54255   26789.61 1 vs 2    15 1591.189       0
```

O terceiro modelo é estatisticamente diferente do segundo, então podemos dizer que a distribuição de tamanhos dentro de cada idade e sexo é diferente ao longo os anos.

Se finalmente compararmos os 3 modelos para ver qual deles tem um valor menor para o critério de Akaike, veremos que:

```
AIC(final_model1$fit$fit$fit,final_model2$fit$fit$fit,final_model3$fit$fit$fit)
```

```
##                          df      AIC
## final_model1$fit$fit$fit 10 29575.41
## final_model2$fit$fit$fit 20 28420.80
## final_model3$fit$fit$fit 35 26859.61
```

Podemos afirmar que conseguimos um melhor ajuste com o terceiro modelo, inclua a variável ano e as iterações duas a duas entre todas as variáveis.

Vamos tentar agora um modelo Random Forest, conjunto de árvores de classificação.

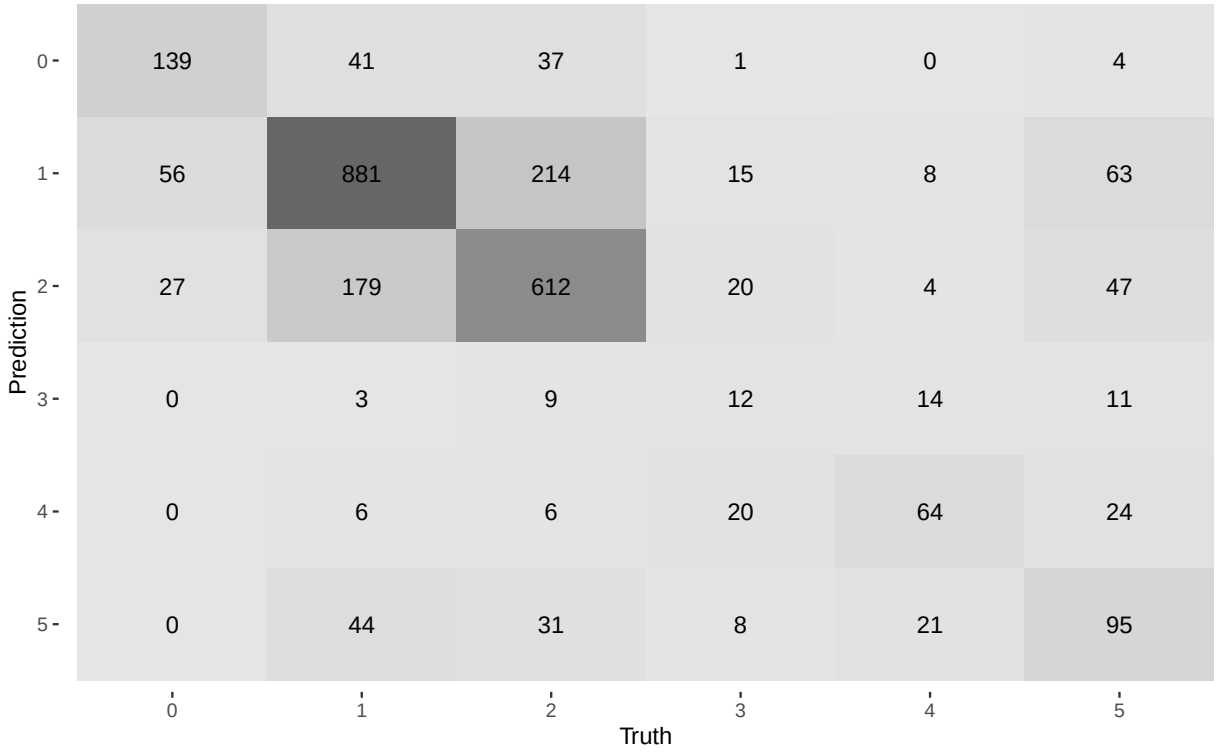
Construímos o modelo seguindo os mesmos passos do caso do modelo de regressão multinomial:

```
rf_rec <-
  recipe(matura ~  comprimento + sexo + ano + mes + trimestre + day,
         data = train) %>%
  step_naomit(everything(), skip = TRUE) %>%
  step_novel(sexo) %>%
  step_impute_knn(comprimento, neighbors = 3)

# K-fold cross-validation
set.seed(1234)
cv_folds <- vfold_cv(train, v = 10)

rf_spec <-
  rand_forest(
    mode = "classification",
    mtry = tune(),
    trees = tune()
  ) %>%
  set_engine("randomForest")

rf_wflow <-
  workflow() %>%
```

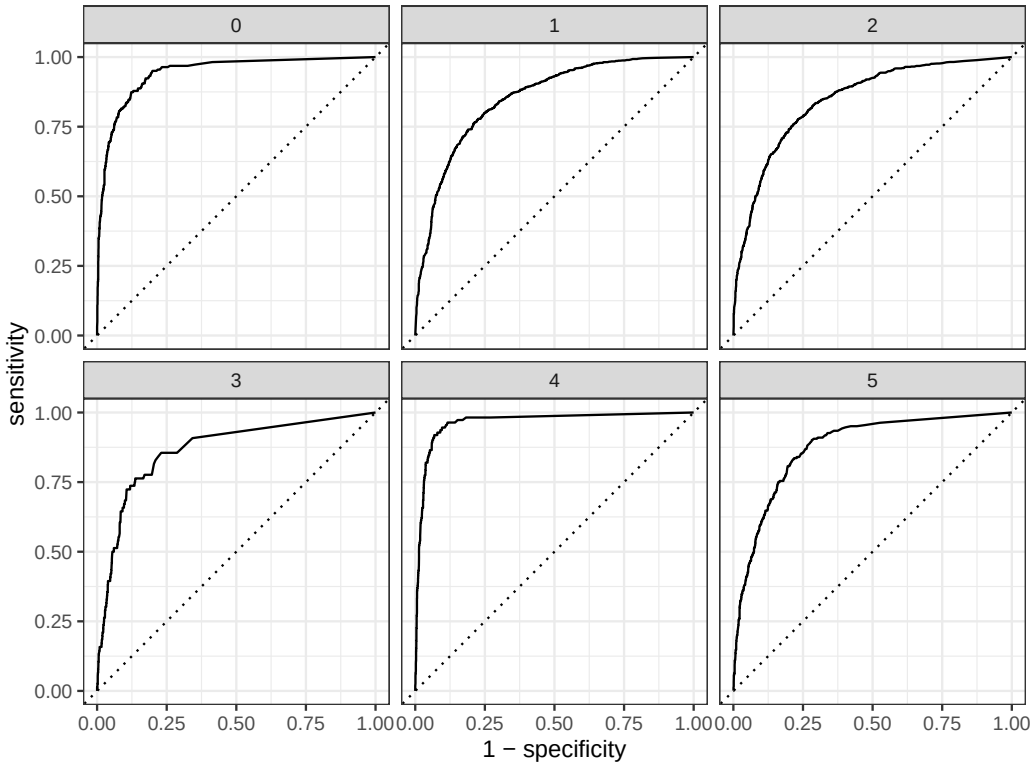
```
  add_recipe(rf_rec) %>%
  add_model(rf_spec)

random_tune_rf <-
  rf_wflow %>%
  tune_grid(
    resamples = cv_folds, grid = 5
  )

random_final_rf <-
  finalize_workflow(rf_wflow, select_best(random_tune_rf)) %>%
  fit(train)

last_fit_rf <- last_fit(random_final_rf,
                        split = data_split,
                        metrics = metric_set(
                           precision,roc_auc,
                           sens, spec, f_meas)
)

last_fit_rf %>%
  collect_metrics()
```

```
## # A tibble: 5 x 4
##   .metric   .estimator .estimate .config
##   <chr>     <chr>          <dbl> <chr>
## 1 precision macro          0.547 Preprocessor1_Model1
## 2 sens      macro          0.531 Preprocessor1_Model1
## 3 spec      macro          0.918 Preprocessor1_Model1
## 4 f_meas    macro          0.536 Preprocessor1_Model1
## 5 roc_auc   hand_till      0.873 Preprocessor1_Model1
```

```
last_fit_rf %>%
  collect_predictions() %>%
  conf_mat(matura, .pred_class) %>%
  autoplot(type = "heatmap")
```



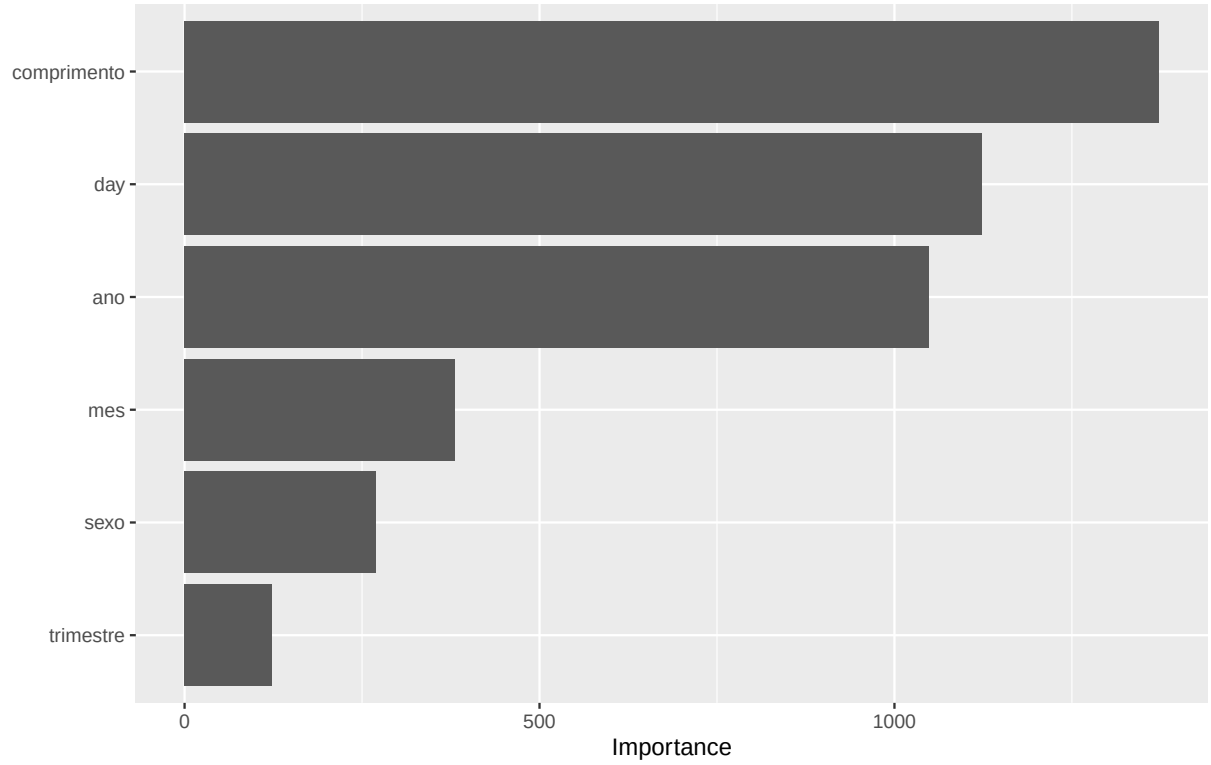

```
last_fit_rf %>%
  collect_predictions() %>%
  roc_curve(matura,'.pred_0','.pred_1','.pred_2','.pred_3','.pred_4','.pred_5') %>%
  autoplot()
```



A precisão aumentou um pouco mais, sendo a melhor dos 5 modelos. As curvas ROC também melhoraram nas maturas onde os modelos anteriores não se encaixavam tão bem.

Podemos também estudar a importância que o modelo dá a cada variável no ajuste:

```
last_fit_rf %>%
  pluck(".workflow", 1) %>%
  extract_fit_parsnip() %>%
  vip(num_features = 10)
```



O comprimento, o ano e o dia de pesca parecem ser as variáveis que melhor discriminam as maturas.

Finalmente vamos trabalhar com as probabilidades condicionais através da matriz ALK.

Se calculássemos a maturação da subamostra com valores de NAs por meio da matriz ALK, obteríamos as seguintes previsões:

Nós fazemos a matriz ALK:

```
mALK <- round(prop.table(table(test$lcat10,test$matura),1),2)
mALK
```

```
##
##         0    1    2    3    4    5
##   0  0.08 0.50 0.33 0.01 0.01 0.07
##   10 0.79 0.05 0.16 0.00 0.00 0.00
##   20 0.17 0.41 0.32 0.03 0.02 0.05
##   30 0.01 0.46 0.36 0.03 0.05 0.08
##   40 0.00 0.36 0.32 0.04 0.08 0.19
##   50 0.00 0.28 0.23 0.05 0.18 0.26
```

```
mALK2 <- round(prop.table(table(train$lcat10,train$matura),1),2)
mALK2
```

```
##
##         0    1    2    3    4    5
##   0  0.06 0.56 0.27 0.01 0.02 0.08
##   10 0.81 0.05 0.14 0.00 0.00 0.00
##   20 0.22 0.41 0.30 0.02 0.02 0.04
```

```
##   30 0.01 0.44 0.37 0.03 0.05 0.10
##   40 0.00 0.36 0.32 0.06 0.08 0.18
##   50 0.00 0.38 0.19 0.05 0.10 0.27
```

```
df_ALK <- data.frame(apply(mALK,1,function(row) colnames(mALK)[which.max(row)]))
colnames(df_ALK) <- c('matura')
df_ALK$matura <- as.factor(df_ALK$matura)
df_ALK <- rownames_to_column(df_ALK, "lcat10")
df_ALK$lcat10 <- as.numeric(df_ALK$lcat10)

ALK_test <- left_join(test, df_ALK, by="lcat10")
setnames(ALK_test,old = c("matura.x","matura.y"), new = c("matura.observed",
                                                          "matura.predicted"))
ALK_test$matura.predicted <- factor(ALK_test$matura.predicted, levels=c("0","1","2","3","4","5"))

#Confussion Matrix and statistics
cfm <- ALK_test %>% conf_mat(matura.observed,matura.predicted)
summary(cfm)
```

```
## # A tibble: 13 x 3
##    .metric              .estimator .estimate
##    <chr>                <chr>          <dbl>
##  1 accuracy             multiclass    0.440
##  2 kap                  multiclass    0.0381
##  3 sens                 macro         0.199
##  4 spec                 macro         0.838
##  5 ppv                  macro        NA
##  6 npv                  macro         0.898
##  7 mcc                  multiclass    0.133
##  8 j_index              macro         0.0375
##  9 bal_accuracy         macro         0.519
## 10 detection_prevalence macro         0.167
## 11 precision            macro         0.609
## 12 recall               macro         0.199
## 13 f_meas               macro         0.460
```

Se juntarmos as previsões de todos os modelos, podemos ter uma ideia de quais serão mais enviesados com os dados que temos.

```
df <- as.data.frame(cbind(predict(random_final1, test),predict(random_final2, test),
                          predict(random_final3, test),predict(random_final_rf,test),
                          ALK_test$matura.predicted,ALK_test$matura.observed))
colnames(df) <- c("model_1","model_2","model_3","model_rf","model_ALK","observed")

table(df$model_1)
```

```
##
##    0    1    2    3    4    5
##    0 2677   39    0    0    0
```

```
table(df$model_2)
```

```
##
##    0    1    2    3    4    5
##    0 1767  949    0    0    0
```

```
table(df$model_3)
```

```
##
##    0    1    2    3    4    5
##    0 1804  863    0   49    0
```

```
table(df$model_rf)
```

```
##
##    0    1    2    3    4    5
##  222 1238  890   40  125  201
```

```
table(df$model_ALK)
```

```
##
##    0    1    2    3    4    5
##   56 2660    0    0    0    0
```

```
table(df$observed)
```

```
##
##    0    1    2    3    4    5
##  222 1154  909   76  111  244
```

Por fim, usaremos o melhor dos cinco modelos, modelo Random Forest, para prever a maturidade do conjunto de dados com NAs.

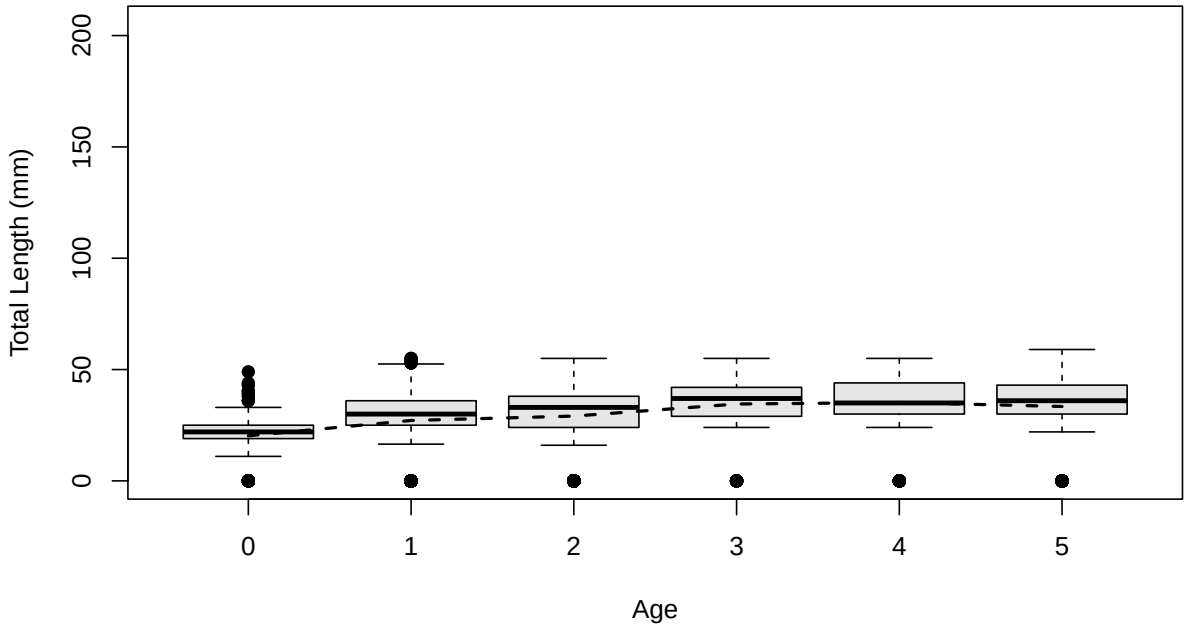
Predicting

Finalmente, podemos fazer as previsões com o modelo mais recente das amostras com NAs atribuídas na matura para poder reconstruir a série original.

```
pred <- random_final_rf %>%
  predict(maturity_pred) %>%
  bind_cols(maturity_pred)

maturity_pred <- maturity_pred %>%
  mutate(matura=pred$.pred_class)


maturity_pred <- maturity_pred %>% select(-rowid)
cc.fnl <- rbind(maturity_train,maturity_pred)

cc.sumlen <- cc.fnl %>% group_by(matura) %>%
  summarize(n_obs=validn(comprimento),mean_comprimento=mean(comprimento,na.rm=TRUE),
            standard_deviation=sd(comprimento,na.rm=TRUE),standard_error=se(comprimento,na.rm=TRUE)) %>%
  as.data.frame()
cc.sumlen
```

```
##   matura n_obs mean_comprimento standard_deviation standard_error
## 1      0   948         20.24367           7.976219      0.2590556
## 2      1  4657         27.13472          14.353930      0.2103380
## 3      2  3511         29.09433          13.201005      0.2227879
## 4      3   303         34.51815          11.236051      0.6454943
## 5      4   481         35.19127          10.889110      0.4965007
## 6      5   988         33.41194          14.057158      0.4472178
```

```
plot(comprimento~matura,data=cc.fnl,pch=19,col=rgb(0,0,0,1/10),
     xlab="Age",ylab="Total Length (mm)",ylim=c(0,205))
lines(mean_comprimento~matura,data=cc.sumlen,lwd=2,lty=2)
```



Como possíveis melhorias no futuro, poderíamos sobreamostrar os conjuntos de dados com matura 0,3,4 e 5 que tiveram poucas observações.