PSTAT 131 Project

Alaina Liu

2023-05-31

```r
movies1 <- read.csv("/Users/alainaliu/Downloads/archive (1) 2/Best Movies Netflix.csv")
shows1 <- read.csv("/Users/alainaliu/Downloads/archive (1) 2/Best Shows Netflix.csv")
head(movies1)[,-1]; head(shows1)[,-1] # remove the index column
```

```
##                                      TITLE RELEASE_YEAR SCORE NUMBER_OF_VOTES
## 1 David Attenborough: A Life on Our Planet         2020   9.0           31180
## 2                                Inception         2010   8.8         2268288
## 3                             Forrest Gump         1994   8.8         1994599
## 4                               Anbe Sivam         2003   8.7           20595
## 5                       Bo Burnham: Inside         2021   8.7           44074
## 6                      Saving Private Ryan         1998   8.6         1346020
##   DURATION  MAIN_GENRE MAIN_PRODUCTION
## 1       83 documentary              GB
## 2      148       scifi              GB
## 3      142       drama              US
## 4      160      comedy              IN
## 5       87      comedy              US
## 6      169       drama              US
```

```
##                         TITLE RELEASE_YEAR SCORE NUMBER_OF_VOTES DURATION
## 1                Breaking Bad         2008   9.5         1727694       48
## 2 Avatar: The Last Airbender         2005   9.3          297336       24
## 3                  Our Planet         2019   9.3           41386       50
## 4                Kota Factory         2019   9.3           66985       42
## 5              The Last Dance         2020   9.1          108321       50
## 6                      Arcane         2021   9.1          175412       41
##   NUMBER_OF_SEASONS  MAIN_GENRE MAIN_PRODUCTION
## 1                 5       drama              US
## 2                 3       scifi              US
## 3                 1 documentary              GB
## 4                 2       drama              IN
## 5                 1 documentary              US
## 6                 1      action              US
```

Categories:

- Release Year
- IMDb Score
- Number of Votes

- Duration (in minutes)
- Main Genre
- Main Production (Country Code)

```
movies1 %>%
  group_by(MAIN_GENRE) %>%
  summarize(count=n())
```

```
## # A tibble: 15 x 2
##    MAIN_GENRE  count
##    <chr>       <int>
##  1 action          5
##  2 animation       3
##  3 comedy         58
##  4 crime          21
##  5 documentary    20
##  6 drama         151
##  7 fantasy        19
##  8 horror          8
##  9 musical         4
## 10 romance        21
## 11 scifi           7
## 12 sports          1
## 13 thriller       59
## 14 war             3
## 15 western         7
```

```
movies1 %>%
  group_by(MAIN_PRODUCTION) %>%
  summarize(count=n())
```

```
## # A tibble: 35 x 2
##    MAIN_PRODUCTION count
##    <chr>           <int>
##  1 AR                  1
##  2 AU                  3
##  3 BE                  2
##  4 BR                  1
##  5 CA                  4
##  6 CD                  1
##  7 CN                  3
##  8 DE                  9
##  9 DK                  1
## 10 ES                  6
## # i 25 more rows
```

```
shows1 %>%
  group_by(MAIN_GENRE) %>%
  summarize(count=n())
```

```
## # A tibble: 12 x 2
##    MAIN_GENRE  count
##    <chr>       <int>
##  1 action         28
##  2 animation       4
##  3 comedy         43
##  4 crime          20
##  5 documentary     7
##  6 drama          82
##  7 reality         2
##  8 romance         1
##  9 scifi          45
## 10 thriller        4
## 11 war             8
## 12 western         2
```

```
shows1 %>%
  group_by(MAIN_PRODUCTION) %>%
  summarize(count=n())
```

```
## # A tibble: 19 x 2
##    MAIN_PRODUCTION count
##    <chr>           <int>
##  1 AU                  1
##  2 BE                  2
##  3 BR                  1
##  4 CA                 13
##  5 DE                  5
##  6 DK                  2
##  7 ES                  4
##  8 FI                  1
##  9 FR                  5
## 10 GB                 27
## 11 IL                  1
## 12 IN                  3
## 13 IT                  1
## 14 JP                 26
## 15 KR                  9
## 16 NO                  4
## 17 SE                  3
## 18 TR                  4
## 19 US                134
```

```
levels(movies1$RELEASE_YEAR)
```

```
## NULL
```

```
levels(shows1$RELEASE_YEAR)
```

```
## NULL
```

As we can see, there are 35 different countries for movies and 19 different countries for shows. We will group them into regions to make the data easier to work with.

```
# only looking at movies and shows released in the 21st century
movies2 <- subset(movies1, MAIN_PRODUCTION!="XX" & RELEASE_YEAR >= 2000)
movies <- movies2 %>%
  mutate(REGION = forcats::fct_collapse(MAIN_PRODUCTION,
                                        AsiaOceania = c("CN", "HK", "ID", "IN", "JP",
                                                 "KH", "KR", "TH", "AU", "NZ"),
                                        AfricaME = c("CD", "MW", "ZA", "PS", "TR"),
                                        NSAmerica = c("CA", "US", "AR", "BR", "MX"),
                                        Europe = c("BE", "DE", "DK", "ES", "FR",
                                                   "GB", "HU", "IE", "IT", "LT",
                                                   "NL", "NO", "PL", "UA"))) %>%
  select(-MAIN_PRODUCTION)
movies$RELEASE_YEAR <- factor(movies$RELEASE_YEAR, ordered=TRUE)

shows2 <- subset(shows1, RELEASE_YEAR >= 2000)
shows <- shows2 %>%
  mutate(REGION = forcats::fct_collapse(MAIN_PRODUCTION,
                                        AsiaOceania = c("IN", "JP", "KR", "AU"),
                                        NSAmerica = c("CA", "US", "BR"),
                                        EuropeME = c("BE", "DE", "DK", "ES", "FI",
                                                   "FR", "GB", "IT", "NO", "SE",
                                                   "TR", "IL"))) %>%
  select(-MAIN_PRODUCTION)
shows$RELEASE_YEAR <- factor(shows$RELEASE_YEAR, ordered=TRUE)
```

```
movies %>%
  group_by(MAIN_GENRE) %>%
  summarize(count=n())
```

```
## # A tibble: 15 x 2
##    MAIN_GENRE  count
##    <chr>       <int>
##  1 action          4
##  2 animation       3
##  3 comedy         49
##  4 crime          20
##  5 documentary    20
##  6 drama         133
##  7 fantasy        19
##  8 horror          5
##  9 musical         4
## 10 romance        17
## 11 scifi           5
## 12 sports          1
## 13 thriller       54
## 14 war             2
## 15 western         5
```

```
movies %>%
  group_by(REGION) %>%
  summarize(count=n())
```

```
## # A tibble: 4 x 2
##   REGION      count
##   <fct>       <int>
## 1 NSAmerica     132
## 2 AsiaOceania   134
## 3 Europe         64
## 4 AfricaME       11
```

```
shows %>%
  group_by(MAIN_GENRE) %>%
  summarize(count=n())
```

```
## # A tibble: 12 x 2
##    MAIN_GENRE  count
##    <chr>       <int>
##  1 action         27
##  2 animation       4
##  3 comedy         41
##  4 crime          20
##  5 documentary     7
##  6 drama          82
##  7 reality         2
##  8 romance         1
##  9 scifi          41
## 10 thriller        4
## 11 war             8
## 12 western         1
```

```
shows %>%
  group_by(REGION) %>%
  summarize(count=n())
```

```
## # A tibble: 3 x 2
##   REGION      count
##   <fct>       <int>
## 1 AsiaOceania    36
## 2 EuropeME       58
## 3 NSAmerica     144
```

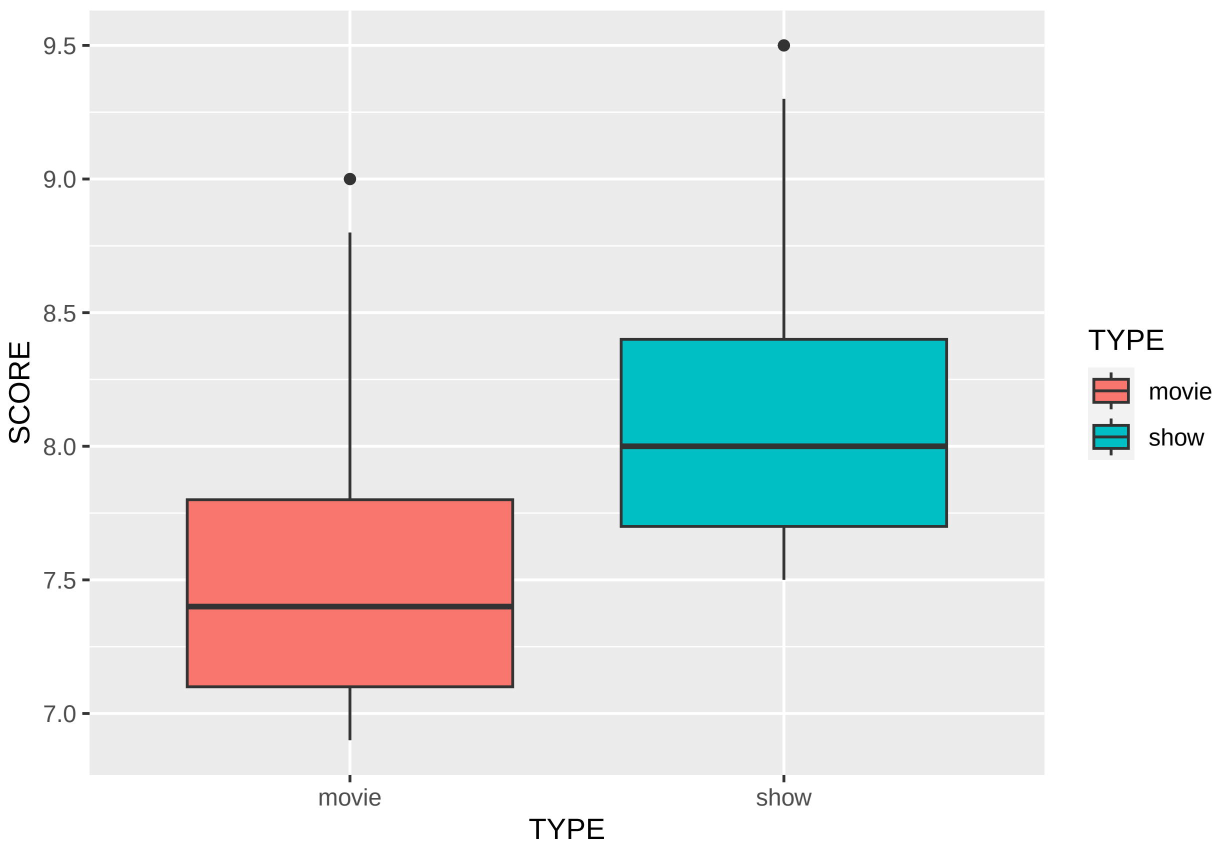
```
levels(movies$RELEASE_YEAR)
```

```
##  [1] "2000" "2001" "2002" "2003" "2004" "2005" "2006" "2007" "2008" "2009"
## [11] "2010" "2011" "2012" "2013" "2014" "2015" "2016" "2017" "2018" "2019"
## [21] "2020" "2021" "2022"
```

```
levels(shows$RELEASE_YEAR)
```

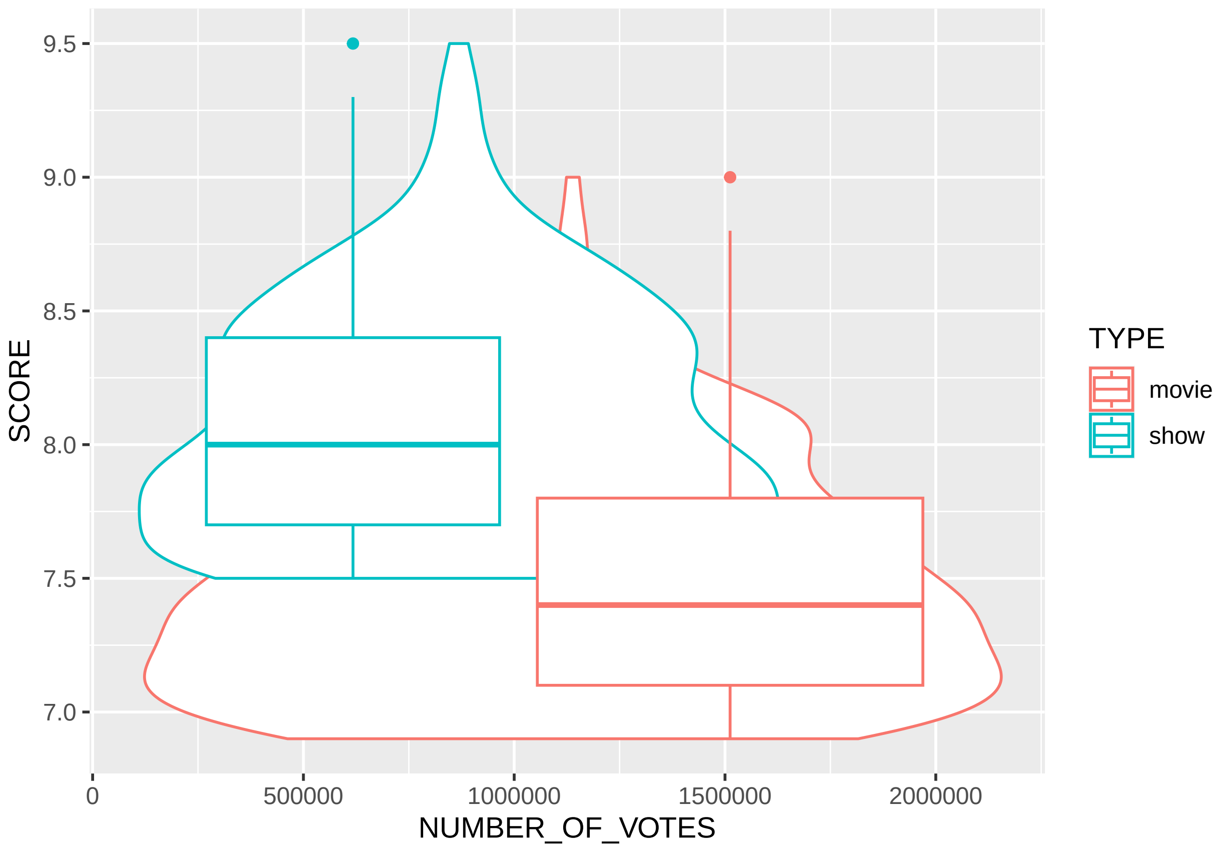
```
##  [1] "2000" "2001" "2002" "2003" "2004" "2005" "2006" "2007" "2008" "2009"
## [11] "2010" "2011" "2012" "2013" "2014" "2015" "2016" "2017" "2018" "2019"
## [21] "2020" "2021" "2022"
```

```r
# graphs
movies$TYPE <- rep("movie", nrow(movies))
shows$TYPE <- rep("show", nrow(shows))
netflix_combined <- dplyr::bind_rows(movies[c(3:9)], shows[c(3:6,8:10)])

netflix_combined %>%
  ggplot(aes(x=TYPE, y=SCORE, fill=TYPE)) +
  geom_boxplot()
```

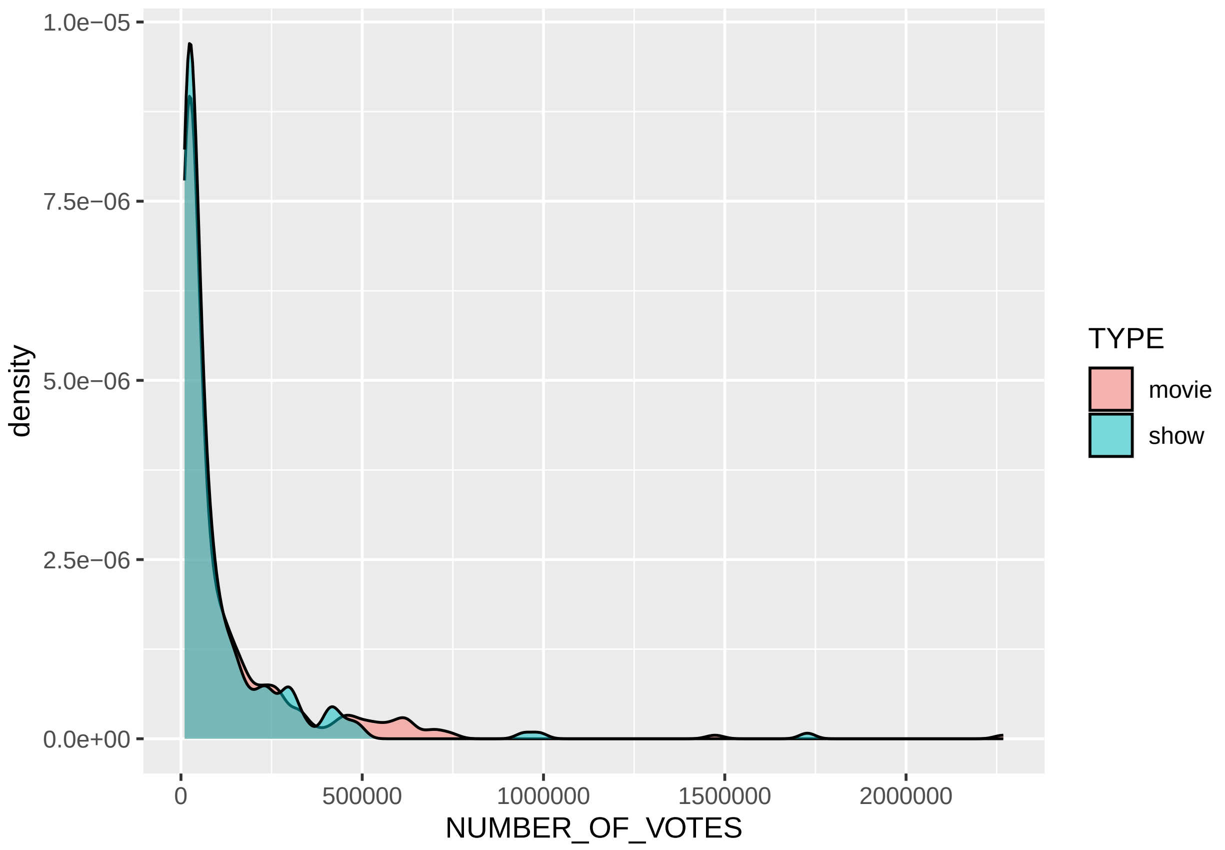


```r
netflix_combined %>%
  ggplot(aes(x=NUMBER_OF_VOTES, y=SCORE, color=TYPE)) +
  geom_violin() +
  geom_boxplot()
```

```
## Warning: ‘position_dodge()‘ requires non-overlapping x intervals
```

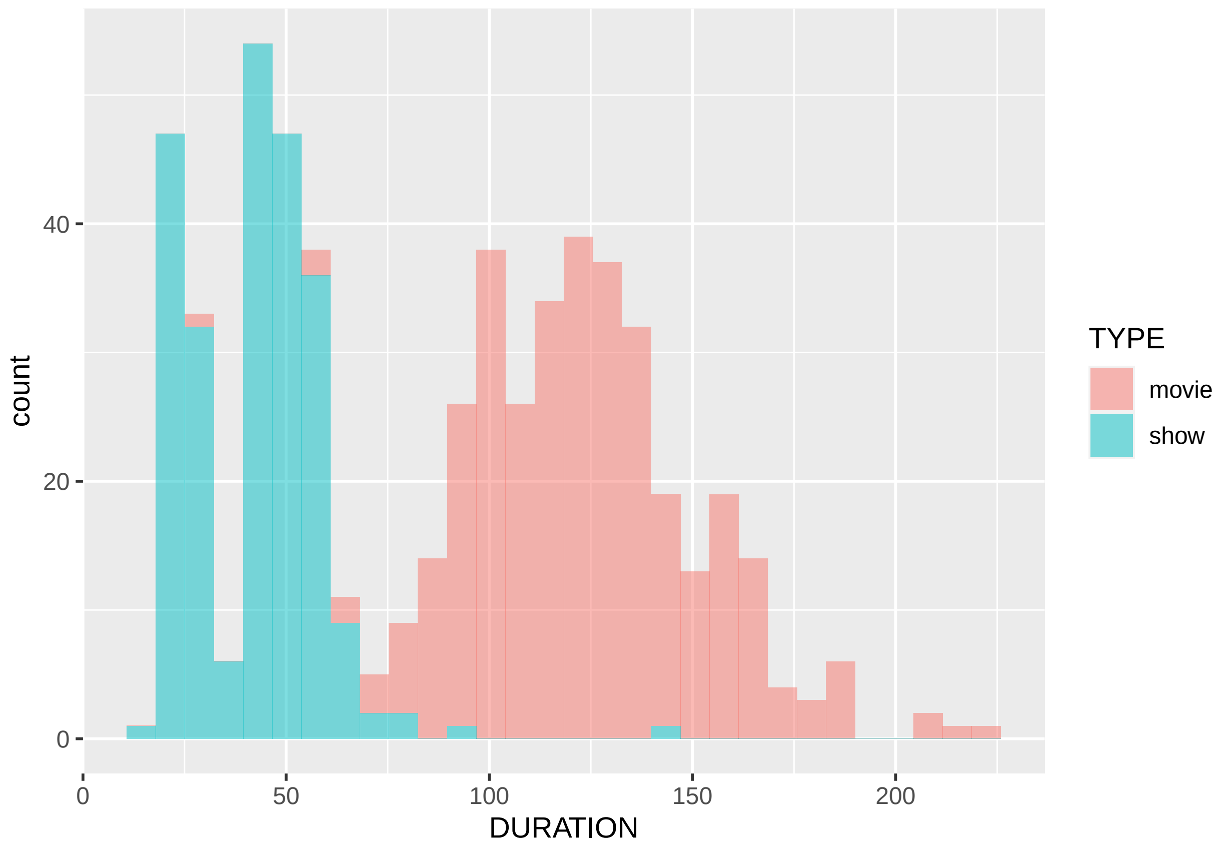


```
netflix_combined %>%
  ggplot(aes(x=NUMBER_OF_VOTES, fill=TYPE)) +
  geom_density(alpha=0.5)
```

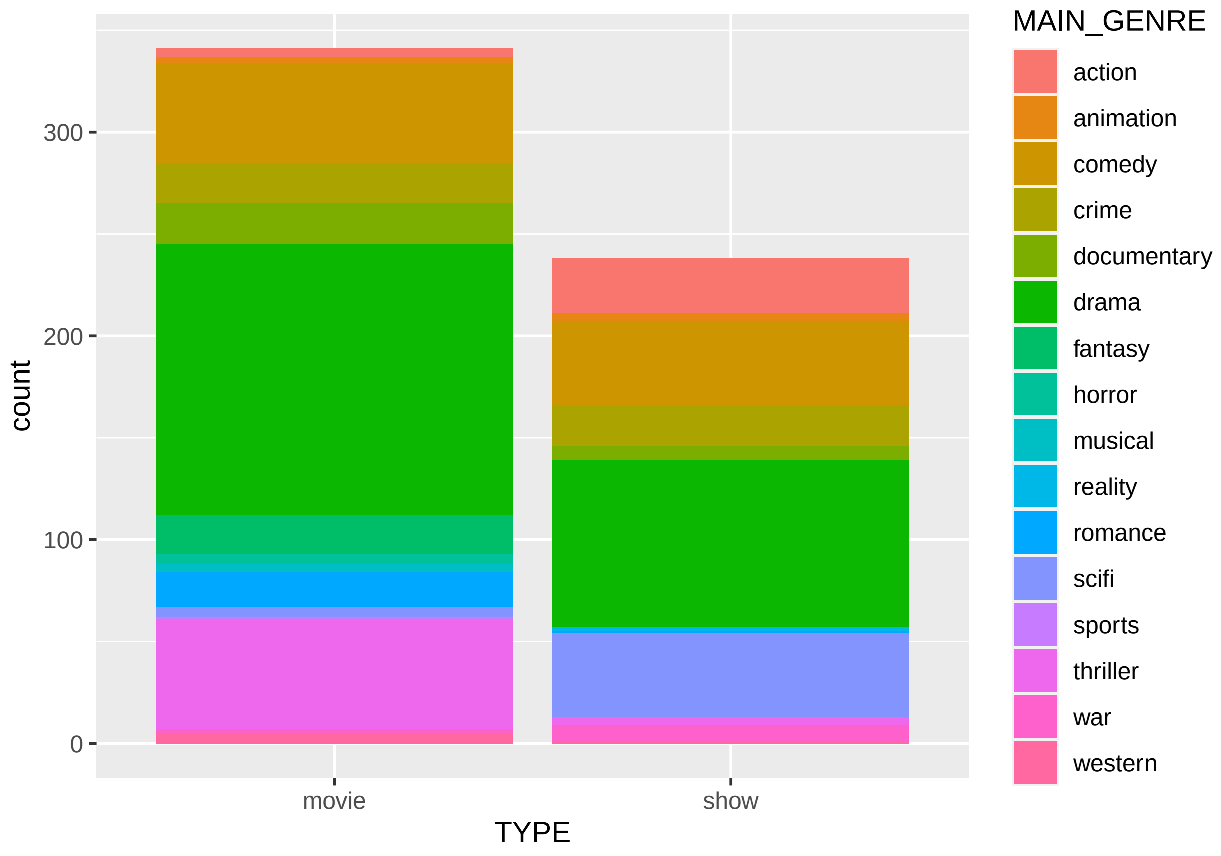


```
netflix_combined %>%
  ggplot(aes(x=DURATION, fill=TYPE)) +
  geom_histogram(alpha=0.5)
```

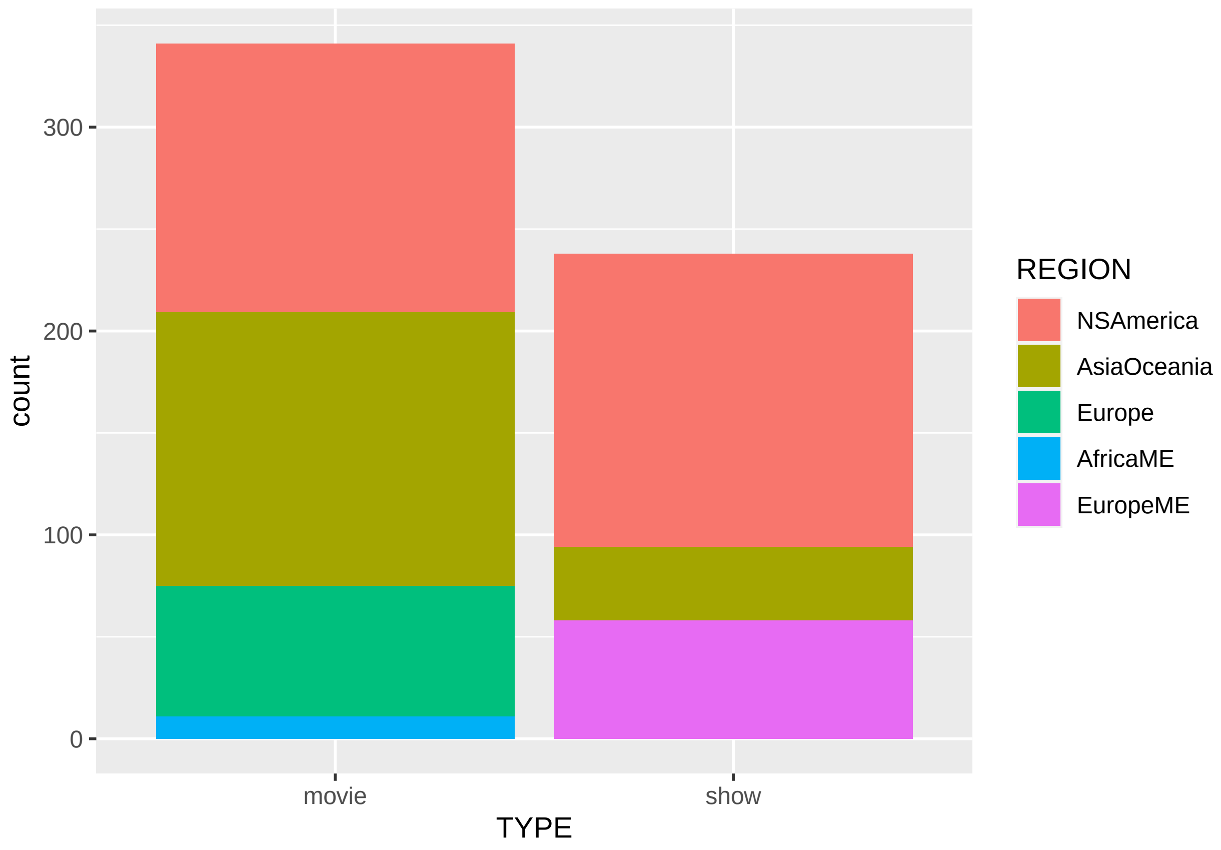
```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



```
netflix_combined %>%
  ggplot(aes(x=TYPE, fill=MAIN_GENRE)) +
  geom_bar()
```

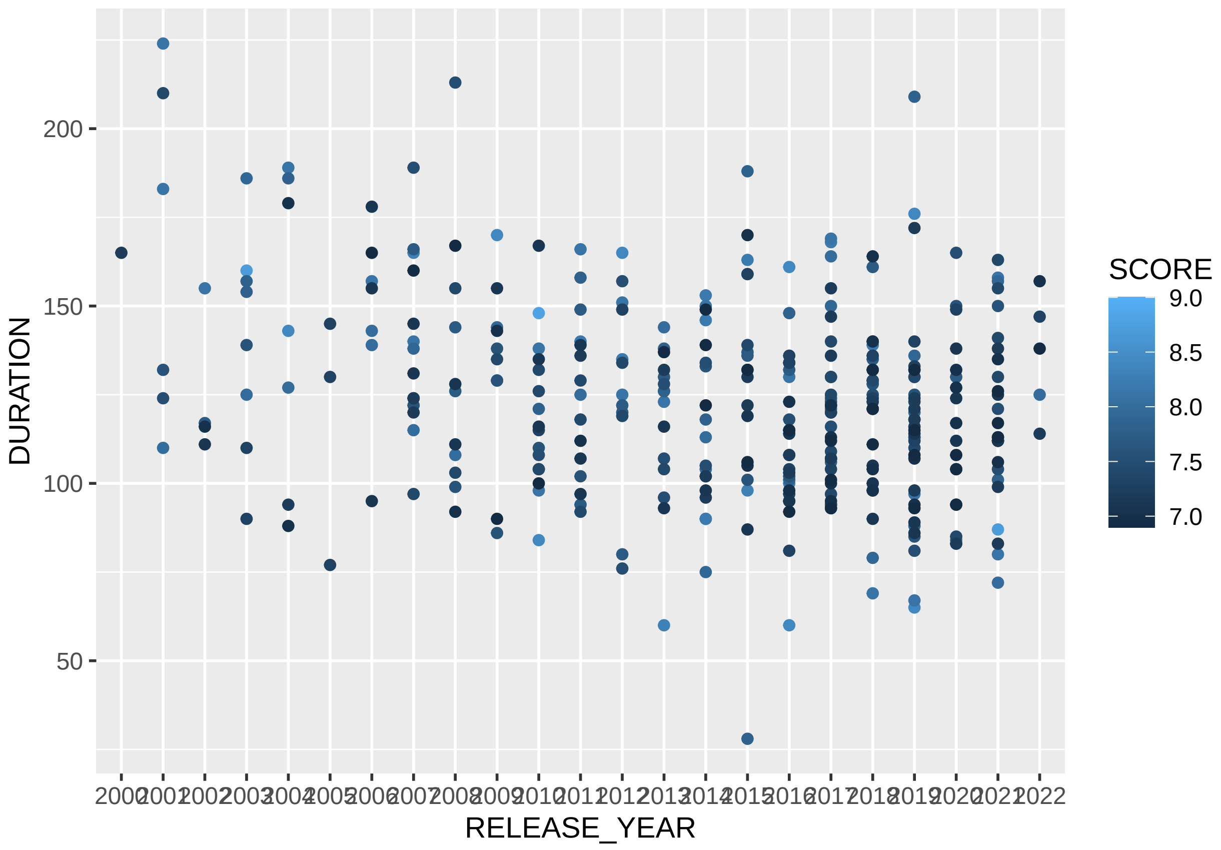


```
netflix_combined %>%
  ggplot(aes(x=TYPE, fill=REGION)) +
  geom_bar()
```



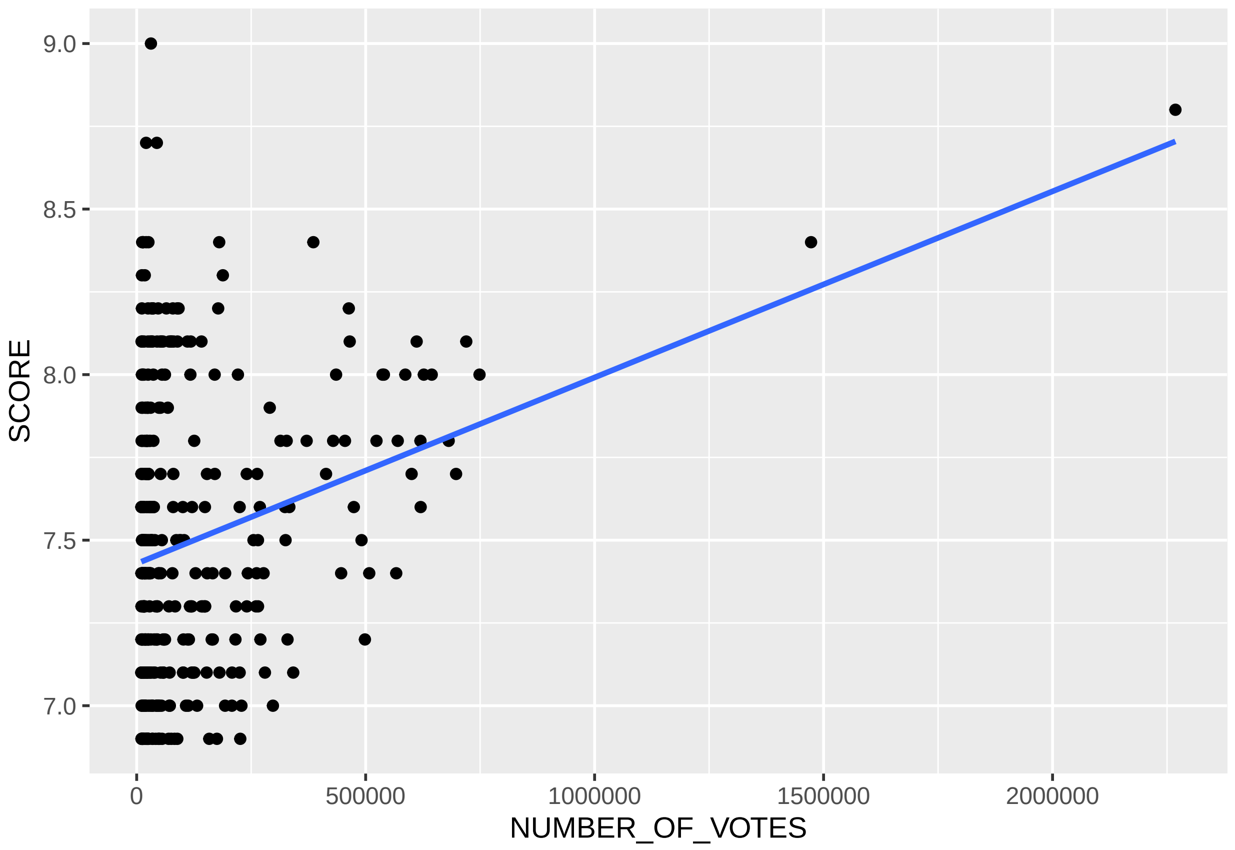
From the dataset, movie scores range from 6.9 to 9.0 and show scores range from 7.5 to 9.5.

```
movies %>%
  ggplot(aes(x=RELEASE_YEAR, y=DURATION, color=SCORE)) +
  geom_point()
```

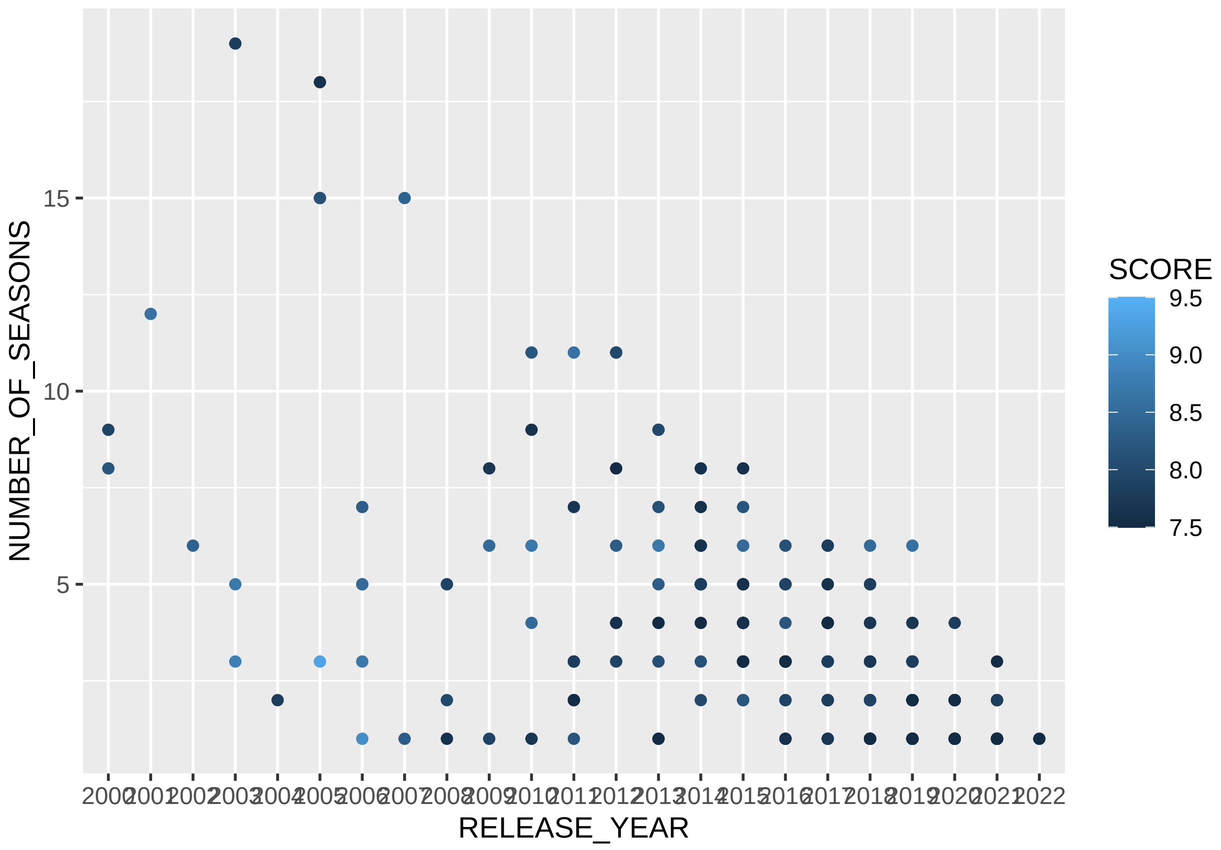


```
movies %>%
  ggplot(aes(x=NUMBER_OF_VOTES, y=SCORE)) +
  geom_point() + geom_smooth(method="lm", se=FALSE)
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```

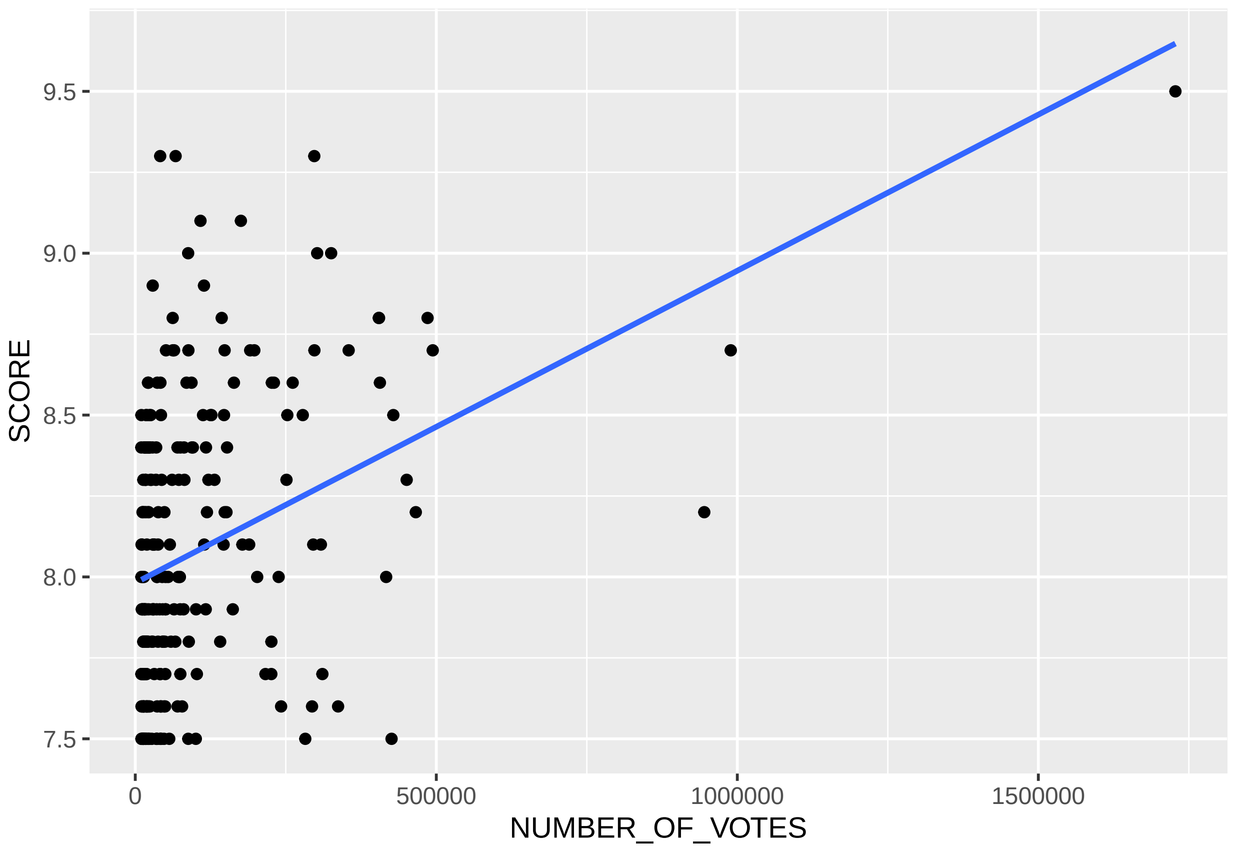


```
shows %>%
  ggplot(aes(x=RELEASE_YEAR, y=NUMBER_OF_SEASONS, color=SCORE)) +
  geom_point()
```



```
shows %>%
  ggplot(aes(x=NUMBER_OF_VOTES, y=SCORE)) +
  geom_point() + geom_smooth(method="lm", se=FALSE)
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



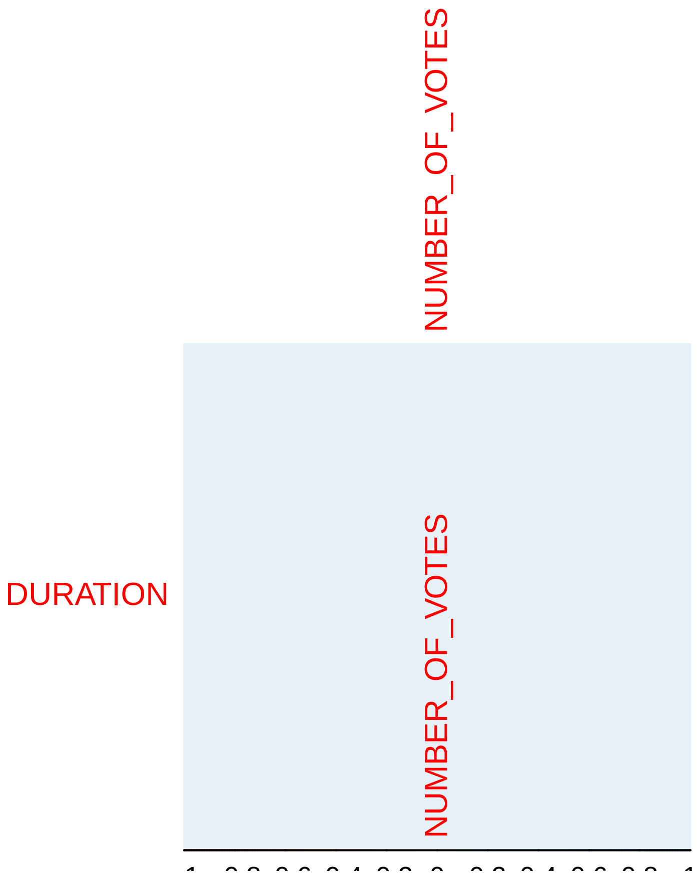
```r
set.seed(131)
movies_split <- initial_split(movies, prop=0.75, strata=SCORE)
movies_train <- training(movies_split)
movies_test <- testing(movies_split)

shows_split <- initial_split(shows, prop=0.75, strata=SCORE)
shows_train <- training(shows_split)
shows_test <- testing(shows_split)

movies_folds <- vfold_cv(movies_train, v=5);movies_folds
```

```
## #  5-fold cross-validation
## # A tibble: 5 x 2
##   splits           id
##   <list>           <chr>
## 1 <split [203/51]> Fold1
## 2 <split [203/51]> Fold2
## 3 <split [203/51]> Fold3
## 4 <split [203/51]> Fold4
## 5 <split [204/50]> Fold5
```

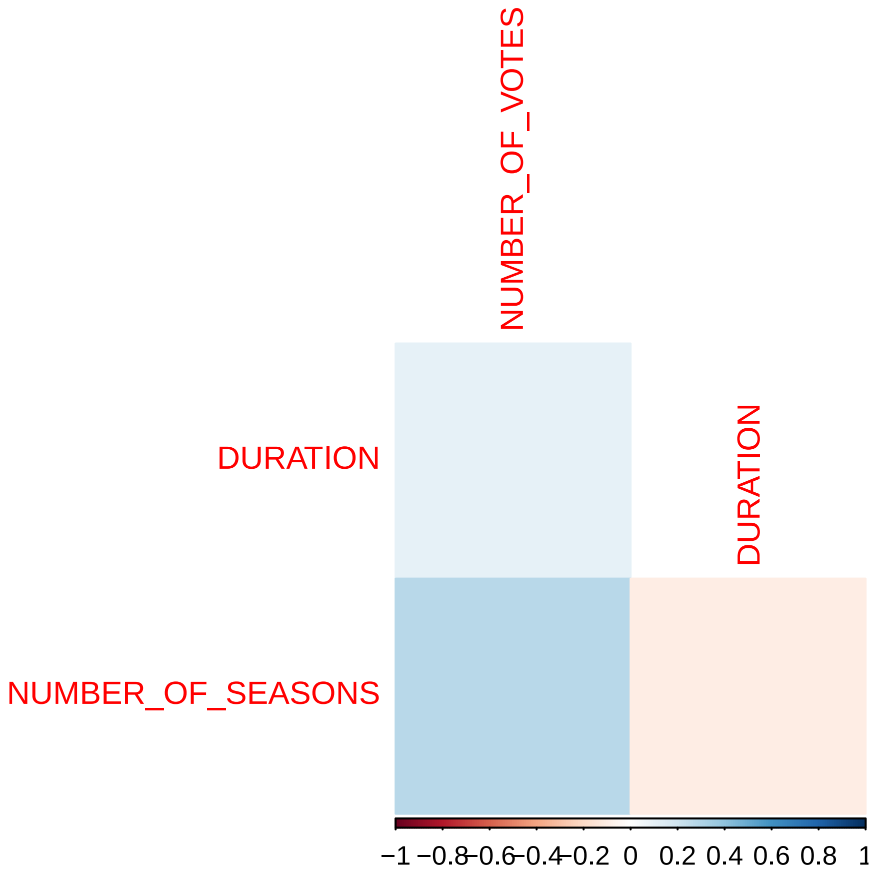
```r
shows_folds <- vfold_cv(shows_train, v=5);shows_folds
```

```
## #  5-fold cross-validation
## # A tibble: 5 x 2
##   splits          id
##   <list>          <chr>
## 1 <split [141/36]> Fold1
## 2 <split [141/36]> Fold2
## 3 <split [142/35]> Fold3
## 4 <split [142/35]> Fold4
## 5 <split [142/35]> Fold5
```

```
movies_train %>%
  dplyr::select(NUMBER_OF_VOTES, DURATION) %>%
  cor() %>%
  corrplot(type="lower", method="color", diag=FALSE)
```



```
shows_train %>%
  dplyr::select(NUMBER_OF_VOTES, DURATION, NUMBER_OF_SEASONS) %>%
  cor() %>%
  corrplot(type="lower", method="color", diag=FALSE)
```



```
movies_recipe <- movies_train %>%
  recipe(SCORE ~ RELEASE_YEAR + NUMBER_OF_VOTES + DURATION + MAIN_GENRE + REGION) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_interact(terms = ~ starts_with("REGION"):starts_with("MAIN_GENRE") + NUMBER_OF_VOTES:starts_with
  step_normalize(NUMBER_OF_VOTES, DURATION)
movies_recipe
```

```
##

## -- Recipe ----------------------------------------------------------------------

##

## -- Inputs

## Number of variables by role

## outcome:   1
## predictor: 5

##

## -- Operations
## * Dummy variables from: all_nominal_predictors()

## * Interactions with: starts_with("REGION"):starts_with("MAIN_GENRE") +
##   NUMBER_OF_VOTES:starts_with("MAIN_GENRE")

## * Centering and scaling for: NUMBER_OF_VOTES, DURATION
```

```
movies_recipe %>%
  prep() %>%
  bake(new_data = movies_train) %>%
  head()
```

```
## # A tibble: 6 x 98
##   NUMBER_OF_VOTES DURATION SCORE RELEASE_YEAR_01 RELEASE_YEAR_02 RELEASE_YEAR_03
##             <dbl>    <dbl> <dbl>           <dbl>           <dbl>           <dbl>
## 1          -0.445 -0.00897   7.1          0.283            0.197          0.0165
## 2          -0.507 -1.34      7.1          0.251            0.106         -0.110
## 3          -0.431 -0.430     7.1          0.283            0.197          0.0165
## 4           0.687 -0.465     7.1         -0.0943          -0.186          0.193
## 5          -0.385 -0.289     7.1          0.0629          -0.213         -0.138
## 6          -0.502 -1.24      7.1          0.251            0.106         -0.110
## # i 92 more variables: RELEASE_YEAR_04 <dbl>, RELEASE_YEAR_05 <dbl>,
## #   RELEASE_YEAR_06 <dbl>, RELEASE_YEAR_07 <dbl>, RELEASE_YEAR_08 <dbl>,
## #   RELEASE_YEAR_09 <dbl>, RELEASE_YEAR_10 <dbl>, RELEASE_YEAR_11 <dbl>,
## #   RELEASE_YEAR_12 <dbl>, RELEASE_YEAR_13 <dbl>, RELEASE_YEAR_14 <dbl>,
## #   RELEASE_YEAR_15 <dbl>, RELEASE_YEAR_16 <dbl>, RELEASE_YEAR_17 <dbl>,
## #   RELEASE_YEAR_18 <dbl>, RELEASE_YEAR_19 <dbl>, RELEASE_YEAR_20 <dbl>,
## #   RELEASE_YEAR_21 <dbl>, RELEASE_YEAR_22 <dbl>, ...
```

```
shows_recipe <- shows_train %>%
  recipe(SCORE ~ RELEASE_YEAR + NUMBER_OF_VOTES + DURATION + NUMBER_OF_SEASONS + MAIN_GENRE + REGION) %
  step_dummy(all_nominal_predictors()) %>%
  step_interact(terms = ~ NUMBER_OF_SEASONS:NUMBER_OF_VOTES + starts_with("REGION"):starts_with("MAIN_G
  step_normalize(NUMBER_OF_VOTES, NUMBER_OF_SEASONS, DURATION)
shows_recipe
```

```
##

## -- Recipe ------------------------------------------------------------------

##

## -- Inputs

## Number of variables by role

## outcome:   1
## predictor: 6

##
```

```
## -- Operations

## * Dummy variables from: all_nominal_predictors()

## * Interactions with: NUMBER_OF_SEASONS:NUMBER_OF_VOTES +
##   starts_with("REGION"):starts_with("MAIN_GENRE") +
##   NUMBER_OF_VOTES:starts_with("MAIN_GENRE")

## * Centering and scaling for: NUMBER_OF_VOTES, NUMBER_OF_SEASONS, DURATION
```

```
shows_recipe %>%
  prep() %>%
  bake(new_data = shows_train) %>%
  head()
```

```
## # A tibble: 6 x 73
##   NUMBER_OF_VOTES DURATION NUMBER_OF_SEASONS SCORE RELEASE_YEAR_01
##             <dbl>    <dbl>             <dbl> <dbl>           <dbl>
## 1          -0.472 -1.04               -0.889   7.7         -0.0314
## 2          -0.488 -0.0579              0.160   7.7          0.251
## 3           1.10   0.00737             1.56    7.7         -0.0629
## 4          -0.379  0.138              -0.889   7.7          0.189
## 5           0.651  0.334              -0.190   7.7          0.220
## 6          -0.450 -0.776              -0.190   7.7          0.157
## # i 68 more variables: RELEASE_YEAR_02 <dbl>, RELEASE_YEAR_03 <dbl>,
## #   RELEASE_YEAR_04 <dbl>, RELEASE_YEAR_05 <dbl>, RELEASE_YEAR_06 <dbl>,
## #   RELEASE_YEAR_07 <dbl>, RELEASE_YEAR_08 <dbl>, RELEASE_YEAR_09 <dbl>,
## #   RELEASE_YEAR_10 <dbl>, RELEASE_YEAR_11 <dbl>, RELEASE_YEAR_12 <dbl>,
## #   RELEASE_YEAR_13 <dbl>, RELEASE_YEAR_14 <dbl>, RELEASE_YEAR_15 <dbl>,
## #   RELEASE_YEAR_16 <dbl>, RELEASE_YEAR_17 <dbl>, RELEASE_YEAR_18 <dbl>,
## #   RELEASE_YEAR_19 <dbl>, RELEASE_YEAR_20 <dbl>, RELEASE_YEAR_21 <dbl>, ...
```

Setting up workflows:

```
# linear regression
movies_lm <- linear_reg() %>%
  set_engine("lm")
movies_lm
```

```
## Linear Regression Model Specification (regression)
##
## Computational engine: lm
```

```
movies_lm_workflow <- workflow() %>%
  add_model(movies_lm) %>%
  add_recipe(movies_recipe)

# knn
movies_knn <- nearest_neighbor(neighbors=tune()) %>%
  set_engine("kknn") %>%
  set_mode("regression")
movies_knn
```

```
## K-Nearest Neighbor Model Specification (regression)
##
## Main Arguments:
##   neighbors = tune()
##
## Computational engine: kknn
```

```
movies_knn_workflow <- workflow() %>%
  add_model(movies_knn) %>%
  add_recipe(movies_recipe)

# elastic net linear regression
movies_en <- linear_reg(mixture = tune(), penalty = tune()) %>%
  set_mode("regression") %>%
  set_engine("glmnet")

movies_en_workflow <- workflow() %>%
  add_model(movies_en) %>%
  add_recipe(movies_recipe)

# pruned decision trees
movies_tree <- decision_tree(cost_complexity = tune()) %>%
  set_engine("rpart") %>%
  set_mode("regression")

movies_tree_workflow <- workflow() %>%
  add_model(movies_tree) %>%
  add_recipe(movies_recipe)

# random forest
movies_forest <- rand_forest(mtry = tune(),
                             trees = tune(),
                             min_n = tune()) %>%
  set_engine("ranger", importance = "impurity") %>%
  set_mode("regression")

movies_forest_workflow <- workflow() %>%
  add_model(movies_forest) %>%
  add_recipe(movies_recipe)

# gradient-boosted trees
movies_bt <- boost_tree(mtry = tune(),
                        trees = tune(),
                        learn_rate = tune()) %>%
  set_engine("xgboost") %>%
  set_mode("regression")

movies_bt_workflow <- workflow() %>%
  add_model(movies_bt) %>%
  add_recipe(movies_recipe)
```

```
# linear regression
shows_lm <- linear_reg() %>%
  set_engine("lm")
```

```r
shows_lm
```

```
## Linear Regression Model Specification (regression)
##
## Computational engine: lm
```

```r
shows_lm_workflow <- workflow() %>%
  add_model(shows_lm) %>%
  add_recipe(shows_recipe)

# knn
shows_knn <- nearest_neighbor(neighbors=tune()) %>%
  set_engine("kknn") %>%
  set_mode("regression")
shows_knn
```

```
## K-Nearest Neighbor Model Specification (regression)
##
## Main Arguments:
##   neighbors = tune()
##
## Computational engine: kknn
```

```r
shows_knn_workflow <- workflow() %>%
  add_model(shows_knn) %>%
  add_recipe(shows_recipe)

# elastic net linear regression
shows_en <- linear_reg(mixture = tune(), penalty = tune()) %>%
  set_mode("regression") %>%
  set_engine("glmnet")

shows_en_workflow <- workflow() %>%
  add_model(shows_en) %>%
  add_recipe(shows_recipe)
```

Setting up grids

```r
# knn
movies_knn_grid <- grid_regular(neighbors(range=c(1, 10)), levels=10)
movies_knn_grid %>% kable()
```

neighbors
1
2
3
4
5
6
7
8
9
10

```r
# en
movies_en_grid <- grid_regular(penalty(range=c(0,1),
                               trans = identity_trans()),
                               mixture(range=c(0, 1)), levels=10)
movies_en_grid %>% head() %>% kable()
```

penalty	mixture
0.0000000	0
0.1111111	0
0.2222222	0
0.3333333	0
0.4444444	0
0.5555556	0

```r
# pruned decision trees
movies_tree_grid <- grid_regular(cost_complexity(range = c(-3, -1)), levels = 10)
movies_tree_grid %>% kable()
```

cost_complexity
0.0010000
0.0016681
0.0027826
0.0046416
0.0077426
0.0129155
0.0215443
0.0359381
0.0599484
0.1000000

```r
# random forest
movies_forest_grid <- grid_regular(mtry(range = c(1, 6)),
                        trees(range = c(200, 600)),
                        min_n(range = c(10, 20)),
                        levels = 5)

# gradient-boosted trees
movies_bt_grid <- grid_regular(mtry(range = c(1, 6)),
                        trees(range = c(200, 600)),
                        learn_rate(range = c(-10, -1)),
                        levels = 5)

shows_knn_grid <- movies_knn_grid
shows_en_grid <- movies_en_grid
```

Fitting our models

```r
#linear regression
movies_lm_tune <- tune_grid(
  object = movies_lm_workflow,
  resamples = movies_folds
)
```

```
## Warning: No tuning parameters have been detected, performance will be evaluated
## using the resamples with no tuning. Did you want to [tune()] parameters?

## > A | warning: prediction from a rank-deficient fit may be misleading

## There were issues with some computations   A: x1                                              > B
## There were issues with some computations   A: x1There were issues with some computations   A: x1   B
## There were issues with some computations   A: x1   B: x1There were issues with some computations   A
## There were issues with some computations   A: x2   B: x1   C: x1There were issues with some computati
```

```
# knn
movies_knn_tune <- tune_grid(
  object = movies_knn_workflow,
  resamples = movies_folds,
  grid = movies_knn_grid,
)
```

```
## > A | warning: There are new levels in a factor: horror
## There were issues with some computations   A: x1                                              > B
##                  x Existing data has 49 rows.
##                  x Assigned data has 51 rows.
##                  i Only vectors of size 1 are recycled.
##                  Caused by error in `vectbl_recycle_rhs_rows()`:
##                  ! Can't recycle input of size 51 to size 49.
## There were issues with some computations   A: x1There were issues with some computations   A: x1   B
## There were issues with some computations   A: x1   B: x1There were issues with some computations   A
##                  x Existing data has 50 rows.
##                  x Assigned data has 51 rows.
##                  i Only vectors of size 1 are recycled.
##                  Caused by error in `vectbl_recycle_rhs_rows()`:
##                  ! Can't recycle input of size 51 to size 50.
## There were issues with some computations   A: x1   B: x1   C: x1There were issues with some computati
## There were issues with some computations   A: x1   B: x1   C: x1   D: x1There were issues with some c
##                  x Existing data has 49 rows.
##                  x Assigned data has 50 rows.
##                  i Only vectors of size 1 are recycled.
##                  Caused by error in `vectbl_recycle_rhs_rows()`:
##                  ! Can't recycle input of size 50 to size 49.
## There were issues with some computations   A: x1   B: x1   C: x1   D: x1   E: x1There were issues wi
```

```
# elastic net linear regression
movies_en_tune <- tune_grid(
  object = movies_en_workflow,
  resamples = movies_folds,
  grid = movies_en_grid
)
```

```
## > A | warning: A correlation computation is required, but `estimate` is constant and has 0 standard 
## There were issues with some computations   A: x1                                              > B
## There were issues with some computations   A: x1There were issues with some computations   A: x1   B
## There were issues with some computations   A: x2   B: x10There were issues with some computations   
## There were issues with some computations   A: x4   B: x10   C: x10There were issues with some computa
```

```r
# pruned decision tree
movies_tree_tune <- tune_grid(
  object = movies_tree_workflow,
  resamples = movies_folds,
  grid = movies_tree_grid
)
```

```
## > A | warning: There are new levels in a factor: horror
## There were issues with some computations   A: x1There were issues with some computations   A: x2There
## There were issues with some computations   A: x10                                                    >
## There were issues with some computations   A: x10There were issues with some computations   A: x10
## There were issues with some computations   A: x10   B: x3   C: x10There were issues with some computa
```

```r
# random forest
movies_forest_tune <- tune_grid(
  object = movies_forest_workflow,
  resamples = movies_folds,
  grid = movies_forest_grid
)
```

```
## > A | warning: There are new levels in a factor: horror
## There were issues with some computations   A: x1                                                   > B
## There were issues with some computations   A: x1There were issues with some computations   A: x1   B
## There were issues with some computations   A: x125   B: x124There were issues with some computations
## There were issues with some computations   A: x125   B: x125   C: x1There were issues with some compu
## There were issues with some computations   A: x125   B: x125   C: x125   D: x124There were issues wi
## There were issues with some computations   A: x125   B: x125   C: x125   D: x...There were issues wi
```

```r
# gradient-boosted trees
movies_bt_tune <- tune_grid(
  object = movies_bt_workflow,
  resamples = movies_folds,
  grid = movies_bt_grid
)
```

```
## > A | warning: A correlation computation is required, but ‘estimate‘ is constant and has 0 standard 
## There were issues with some computations   A: x1                                                   > B
## There were issues with some computations   A: x1There were issues with some computations   A: x1   B
## There were issues with some computations   A: x1   B: x25There were issues with some computations   
## There were issues with some computations   A: x4   B: x25   C: x25There were issues with some computa
```

```r
save(movies_lm_tune, file="movies_lm_results.rda")
save(movies_knn_tune, file="movies_knn_results.rda")
save(movies_en_tune, file="movies_en_results.rda")
save(movies_tree_tune, file="movies_tree_results.rda")
save(movies_forest_tune, file="movies_forest_results.rda")
save(movies_bt_tune, file="movies_bt_results.rda")
movies_lm_tune
```

```
## # Tuning results
## # 5-fold cross-validation
## # A tibble: 5 x 4
##   splits           id    .metrics         .notes
##   <list>           <chr> <list>           <list>
## 1 <split [203/51]> Fold1 <tibble [2 x 4]> <tibble [1 x 3]>
## 2 <split [203/51]> Fold2 <tibble [2 x 4]> <tibble [1 x 3]>
## 3 <split [203/51]> Fold3 <tibble [2 x 4]> <tibble [1 x 3]>
## 4 <split [203/51]> Fold4 <tibble [2 x 4]> <tibble [1 x 3]>
## 5 <split [204/50]> Fold5 <tibble [2 x 4]> <tibble [1 x 3]>
##
## There were issues with some computations:
##
##   - Warning(s) x1: There are new levels in a factor: horror, prediction from a rank-...
##   - Warning(s) x1: There are new levels in a factor: sports, prediction from a rank-...
##   - Warning(s) x1: There are new levels in a factor: war, prediction from a rank-def...
##   - Warning(s) x2: prediction from a rank-deficient fit may be misleading
##
## Run `show_notes(.Last.tune.result)` for more information.
```

```
movies_knn_tune
```

```
## # Tuning results
## # 5-fold cross-validation
## # A tibble: 5 x 4
##   splits           id    .metrics          .notes
##   <list>           <chr> <list>            <list>
## 1 <split [203/51]> Fold1 <tibble [20 x 5]> <tibble [0 x 3]>
## 2 <split [203/51]> Fold2 <tibble [0 x 5]>  <tibble [2 x 3]>
## 3 <split [203/51]> Fold3 <tibble [0 x 5]>  <tibble [2 x 3]>
## 4 <split [203/51]> Fold4 <tibble [20 x 5]> <tibble [0 x 3]>
## 5 <split [204/50]> Fold5 <tibble [0 x 5]>  <tibble [2 x 3]>
##
## There were issues with some computations:
##
##   - Error(s) x1: Assigned data `orig_rows` must be compatible with existing data. ...
##   - Error(s) x1: Assigned data `orig_rows` must be compatible with existing data. ...
##   - Error(s) x1: Assigned data `orig_rows` must be compatible with existing data. ...
##   - Warning(s) x1: There are new levels in a factor: horror
##   - Warning(s) x1: There are new levels in a factor: sports
##   - Warning(s) x1: There are new levels in a factor: war
##
## Run `show_notes(.Last.tune.result)` for more information.
```

```
movies_en_tune
```

```
## # Tuning results
## # 5-fold cross-validation
## # A tibble: 5 x 4
##   splits           id    .metrics           .notes
##   <list>           <chr> <list>             <list>
## 1 <split [203/51]> Fold1 <tibble [200 x 6]> <tibble [1 x 3]>
## 2 <split [203/51]> Fold2 <tibble [200 x 6]> <tibble [11 x 3]>
## 3 <split [203/51]> Fold3 <tibble [200 x 6]> <tibble [11 x 3]>
## 4 <split [203/51]> Fold4 <tibble [200 x 6]> <tibble [1 x 3]>
## 5 <split [204/50]> Fold5 <tibble [200 x 6]> <tibble [11 x 3]>
##
## There were issues with some computations:
##
##   - Warning(s) x5: A correlation computation is required, but `estimate` is constant...
##   - Warning(s) x10: There are new levels in a factor: horror
##   - Warning(s) x10: There are new levels in a factor: sports
##   - Warning(s) x10: There are new levels in a factor: war
##
## Run `show_notes(.Last.tune.result)` for more information.
```

```
movies_tree_tune
```

```
## # Tuning results
## # 5-fold cross-validation
## # A tibble: 5 x 4
##   splits           id    .metrics          .notes
##   <list>           <chr> <list>            <list>
## 1 <split [203/51]> Fold1 <tibble [20 x 5]> <tibble [0 x 3]>
## 2 <split [203/51]> Fold2 <tibble [20 x 5]> <tibble [11 x 3]>
## 3 <split [203/51]> Fold3 <tibble [20 x 5]> <tibble [11 x 3]>
## 4 <split [203/51]> Fold4 <tibble [20 x 5]> <tibble [1 x 3]>
## 5 <split [204/50]> Fold5 <tibble [20 x 5]> <tibble [11 x 3]>
##
## There were issues with some computations:
##
##   - Warning(s) x4: A correlation computation is required, but `estimate` is constant...
##   - Warning(s) x10: There are new levels in a factor: horror
##   - Warning(s) x10: There are new levels in a factor: sports
##   - Warning(s) x10: There are new levels in a factor: war
##
## Run `show_notes(.Last.tune.result)` for more information.
```

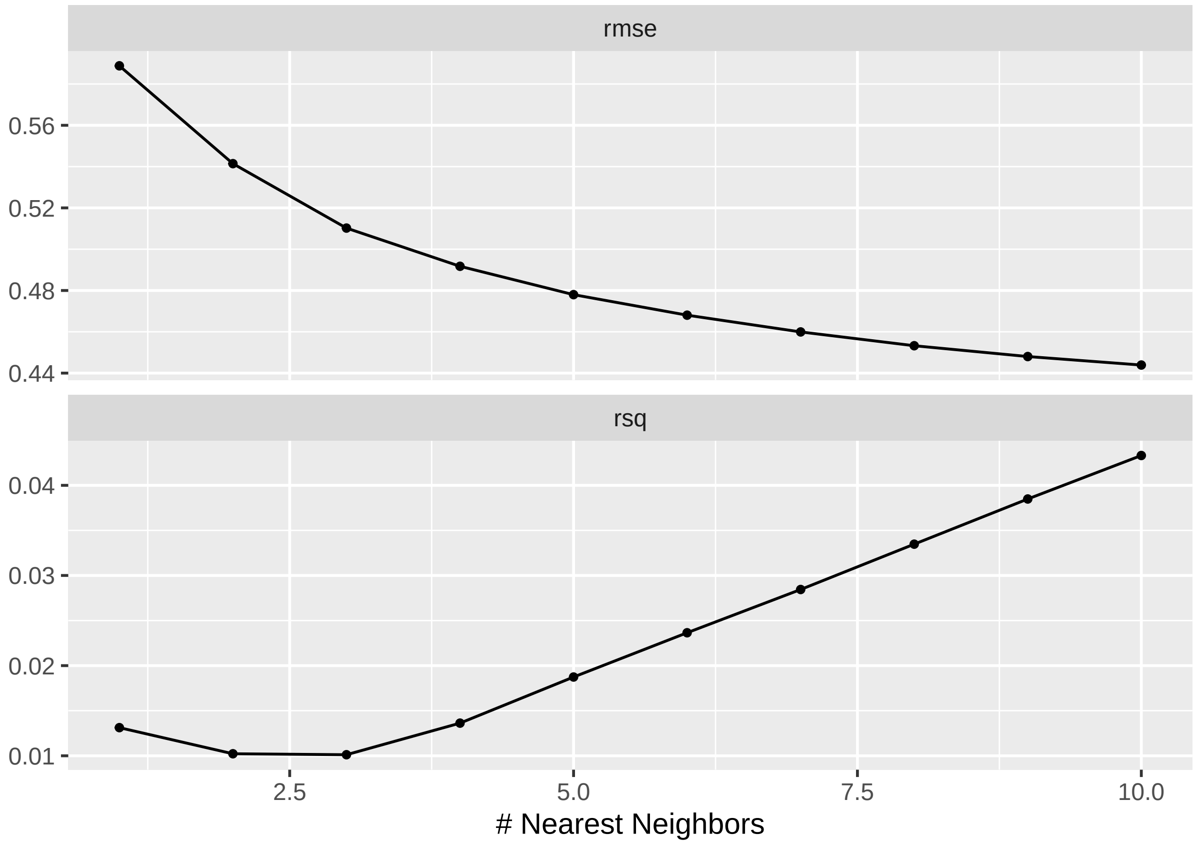
```
movies_forest_tune
```

```
## # Tuning results
## # 5-fold cross-validation
## # A tibble: 5 x 4
##   splits           id    .metrics           .notes
##   <list>           <chr> <list>             <list>
## 1 <split [203/51]> Fold1 <tibble [250 x 7]> <tibble [0 x 3]>
## 2 <split [203/51]> Fold2 <tibble [0 x 7]>   <tibble [250 x 3]>
## 3 <split [203/51]> Fold3 <tibble [0 x 7]>   <tibble [250 x 3]>
## 4 <split [203/51]> Fold4 <tibble [250 x 7]> <tibble [0 x 3]>
## 5 <split [204/50]> Fold5 <tibble [0 x 7]>   <tibble [250 x 3]>
##
## There were issues with some computations:
##
##   - Error(s) x125: Missing data in columns: MAIN_GENRE_animation, MAIN_GENRE_comedy,...
##   - Error(s) x125: Missing data in columns: MAIN_GENRE_animation, MAIN_GENRE_comedy,...
##   - Error(s) x125: Missing data in columns: MAIN_GENRE_animation, MAIN_GENRE_comedy,...
##   - Warning(s) x125: There are new levels in a factor: horror
##   - Warning(s) x125: There are new levels in a factor: sports
##   - Warning(s) x125: There are new levels in a factor: war
##
## Run ‘show_notes(.Last.tune.result)‘ for more information.
```

```
movies_bt_tune
```

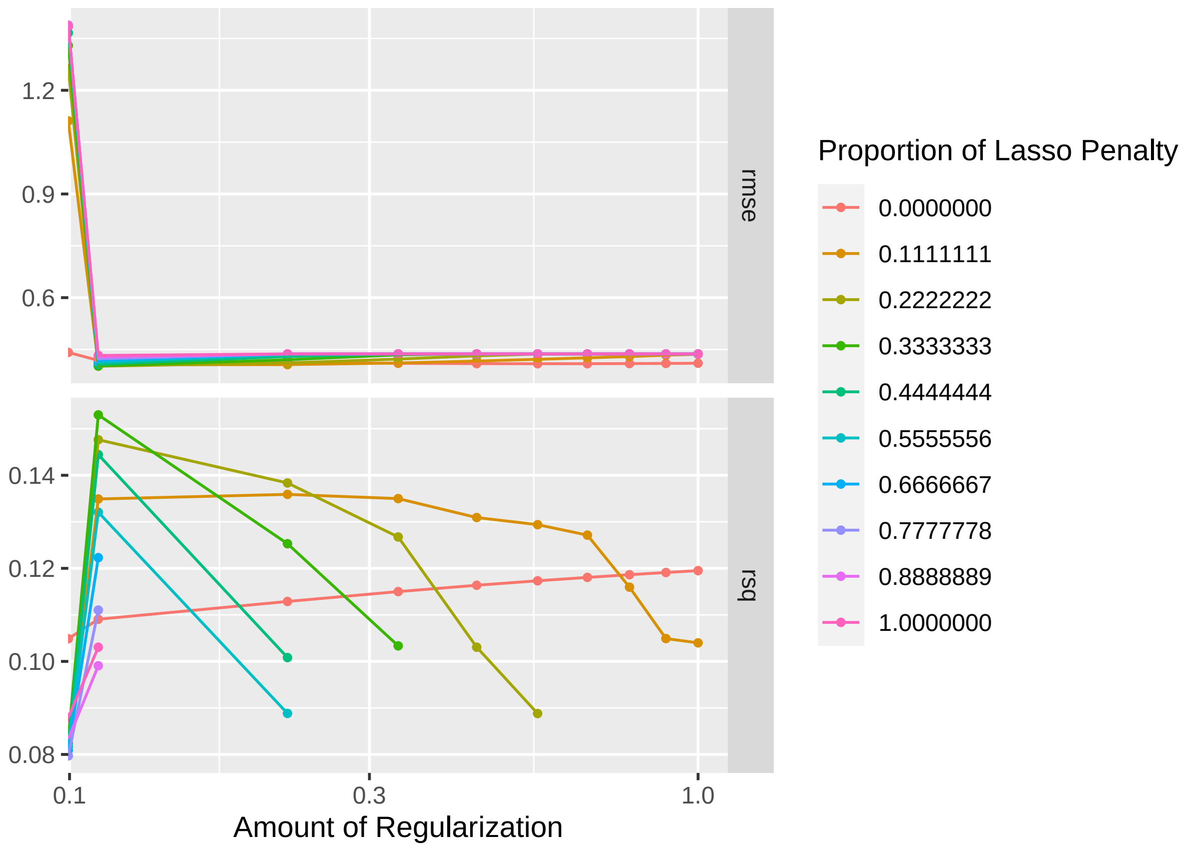
```
## # Tuning results
## # 5-fold cross-validation
## # A tibble: 5 x 4
##   splits           id    .metrics           .notes
##   <list>           <chr> <list>             <list>
## 1 <split [203/51]> Fold1 <tibble [250 x 7]> <tibble [1 x 3]>
## 2 <split [203/51]> Fold2 <tibble [250 x 7]> <tibble [26 x 3]>
## 3 <split [203/51]> Fold3 <tibble [250 x 7]> <tibble [26 x 3]>
## 4 <split [203/51]> Fold4 <tibble [250 x 7]> <tibble [1 x 3]>
## 5 <split [204/50]> Fold5 <tibble [250 x 7]> <tibble [26 x 3]>
##
## There were issues with some computations:
##
##   - Warning(s) x5: A correlation computation is required, but ‘estimate‘ is constant...
##   - Warning(s) x25: There are new levels in a factor: horror
##   - Warning(s) x25: There are new levels in a factor: sports
##   - Warning(s) x25: There are new levels in a factor: war
##
## Run ‘show_notes(.Last.tune.result)‘ for more information.
```

```
autoplot(movies_knn_tune)
```

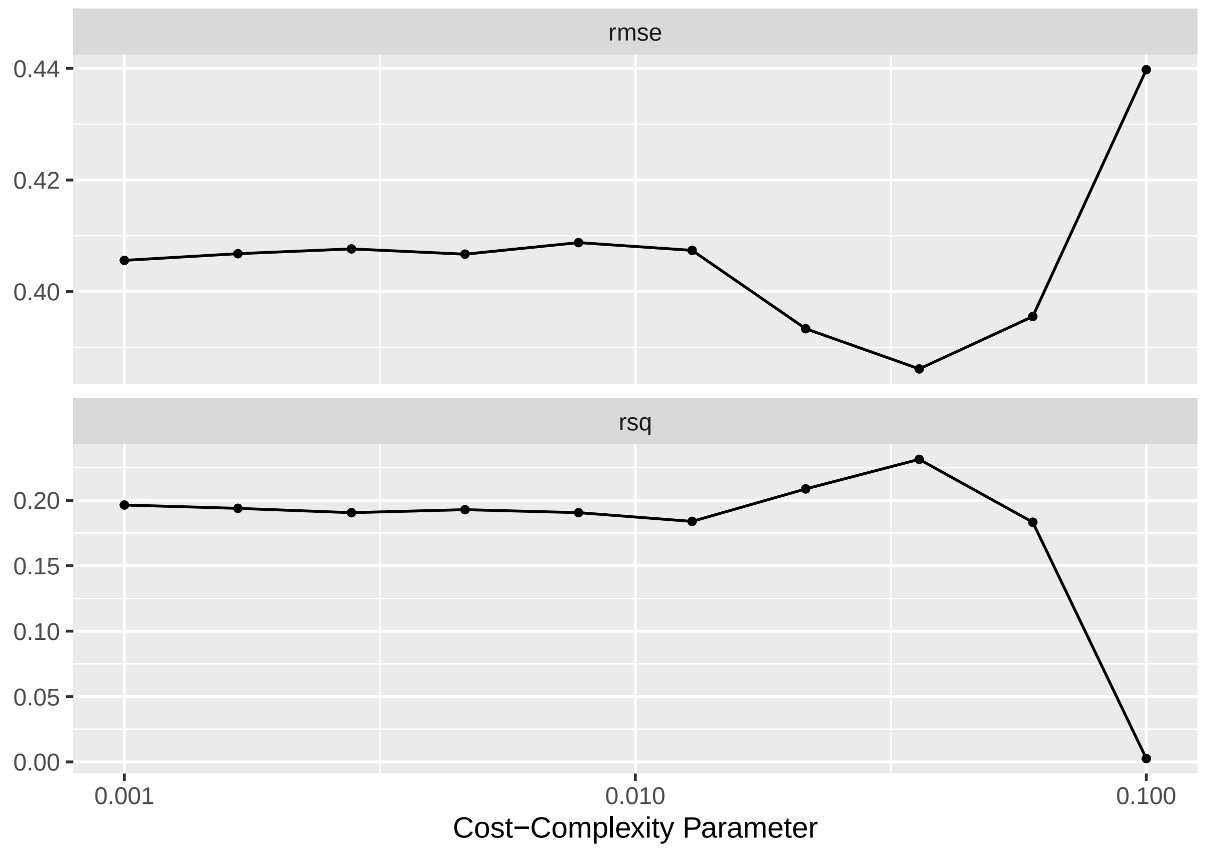


```
autoplot(movies_en_tune)
```

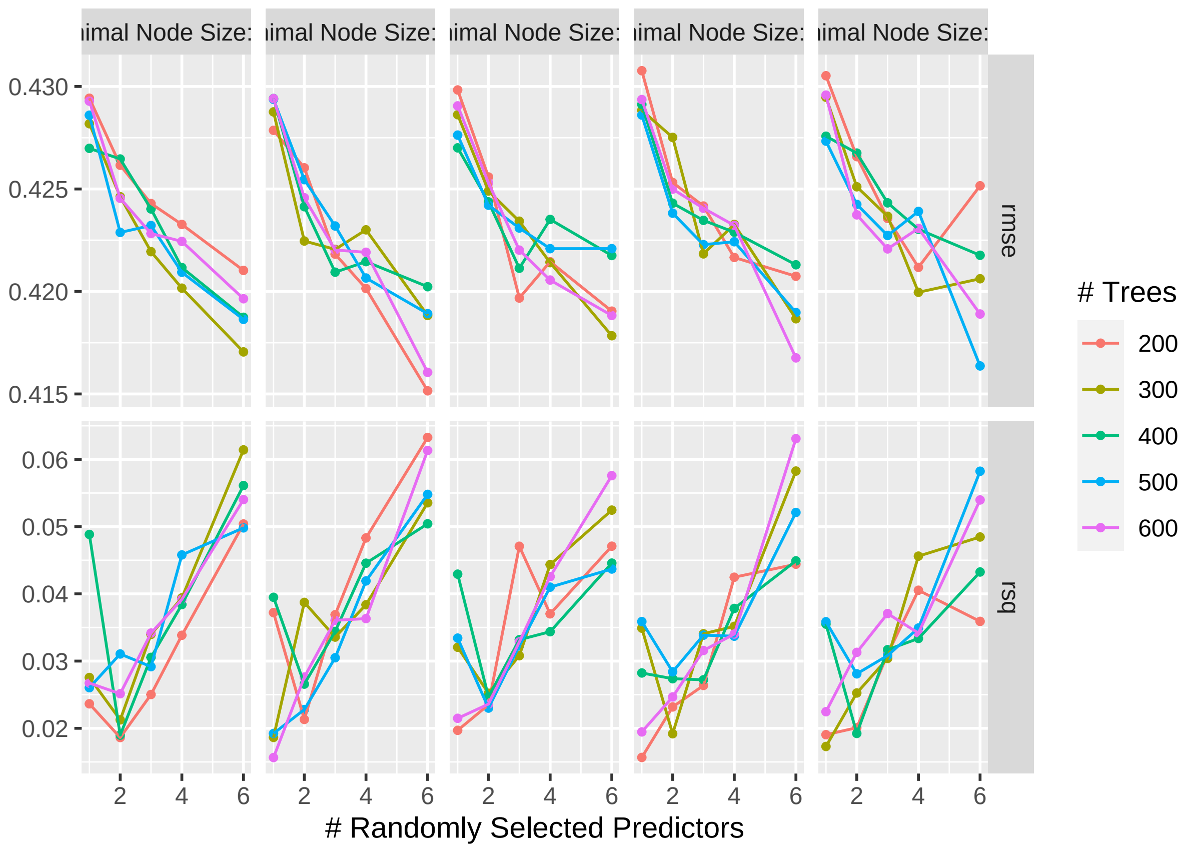
```
## Warning: Transformation introduced infinite values in continuous x-axis
## Transformation introduced infinite values in continuous x-axis
```



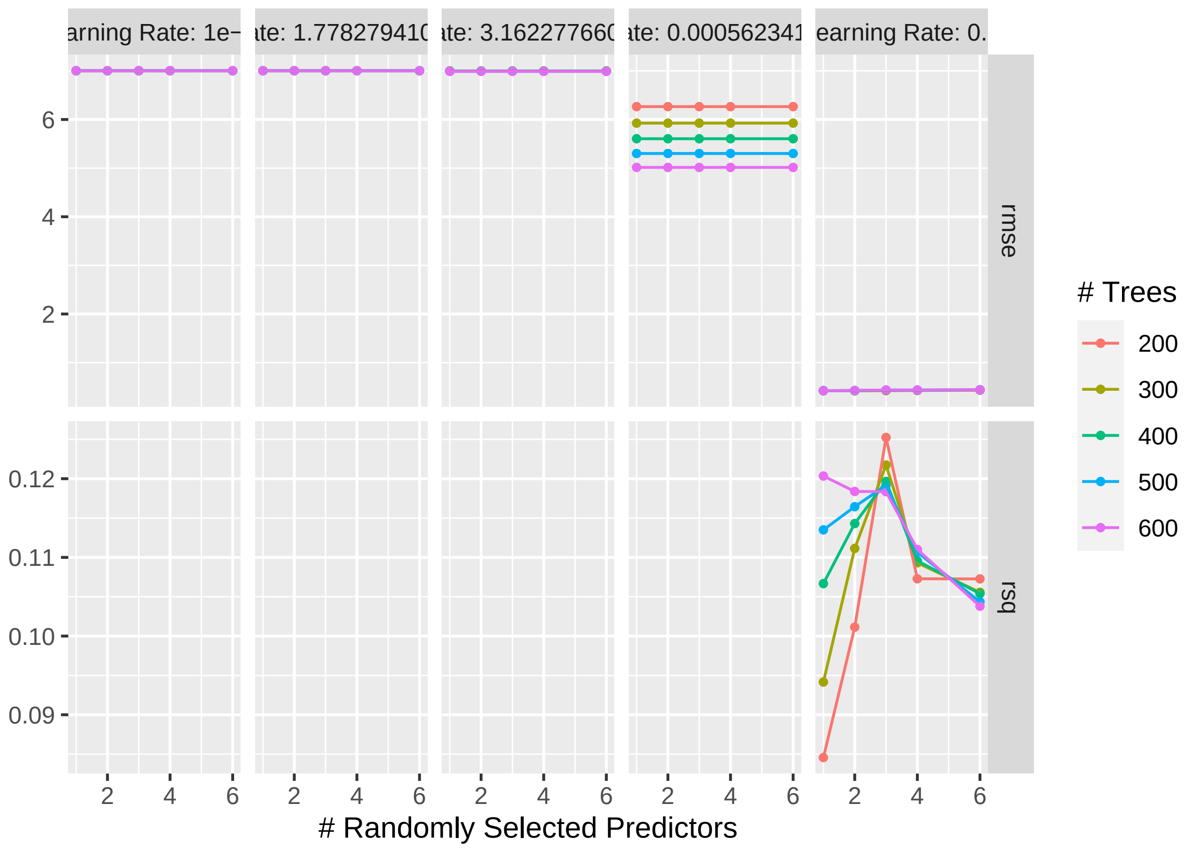
```
autoplot(movies_tree_tune)
```



```r
autoplot(movies_forest_tune)
```



```
autoplot(movies_bt_tune)
```



```r
#linear regression
shows_lm_tune <- tune_grid(
  object = shows_lm_workflow,
  resamples = shows_folds
)
```

```
## Warning: No tuning parameters have been detected, performance will be evaluated
## using the resamples with no tuning. Did you want to [tune()] parameters?

## > A | warning: prediction from a rank-deficient fit may be misleading

## There were issues with some computations   A: x1There were issues with some computations   A: x2
## There were issues with some computations   A: x2There were issues with some computations   A: x2   B
```

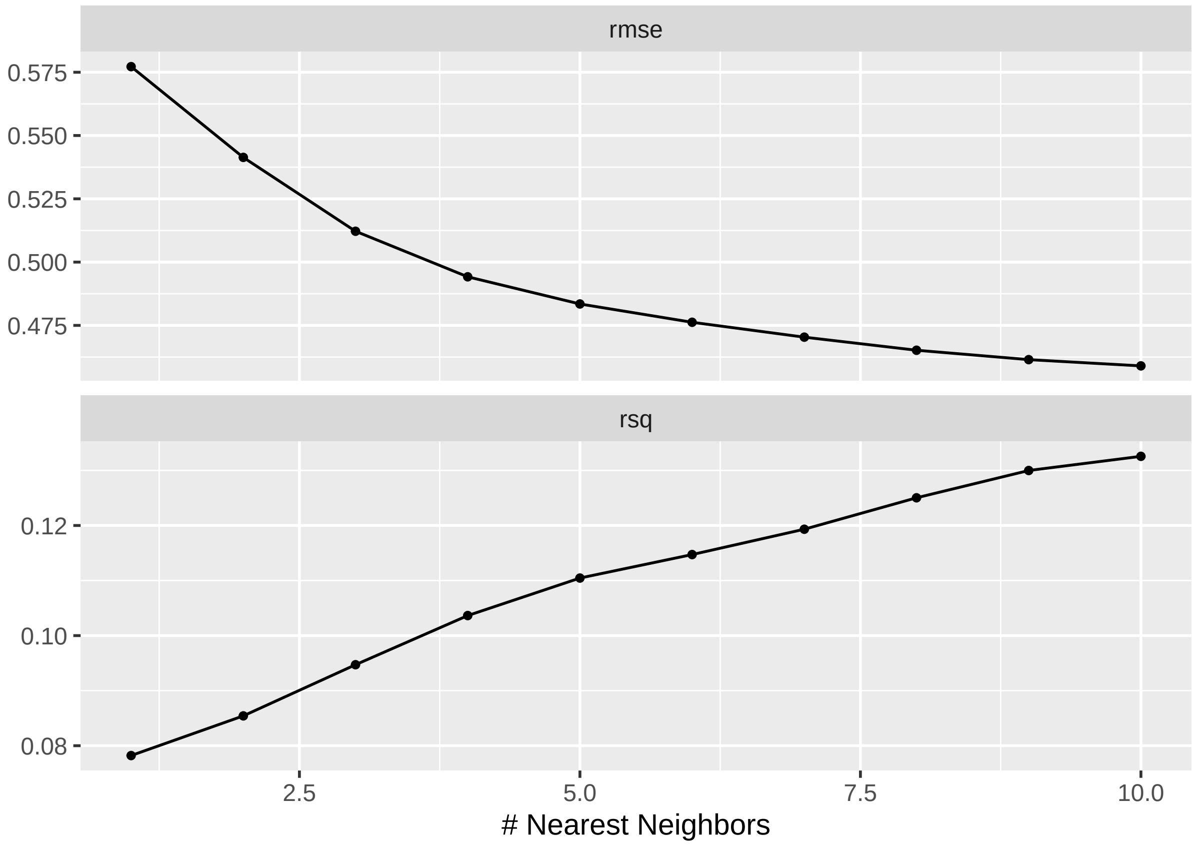
```r
# knn
shows_knn_tune <- tune_grid(
  object = shows_knn_workflow,
  resamples = shows_folds,
  grid = shows_knn_grid,
)
```

```
## > A | warning: There are new levels in a factor: romance, western
## There were issues with some computations   A: x1                                                   > B
##                 x Existing data has 33 rows.
##                 x Assigned data has 35 rows.
##                 i Only vectors of size 1 are recycled.
##                 Caused by error in `vectbl_recycle_rhs_rows()`:
##                 ! Can't recycle input of size 35 to size 33.
## There were issues with some computations   A: x1There were issues with some computations   A: x1   B
```

```
# elastic net linear regression
shows_en_tune <- tune_grid(
  object = shows_en_workflow,
  resamples = shows_folds,
  grid = shows_en_grid
)
```

```
## > A | warning: A correlation computation is required, but `estimate` is constant and has 0 standard
## There were issues with some computations   A: x1There were issues with some computations   A: x2
## There were issues with some computations   A: x2There were issues with some computations   A: x2   B
```

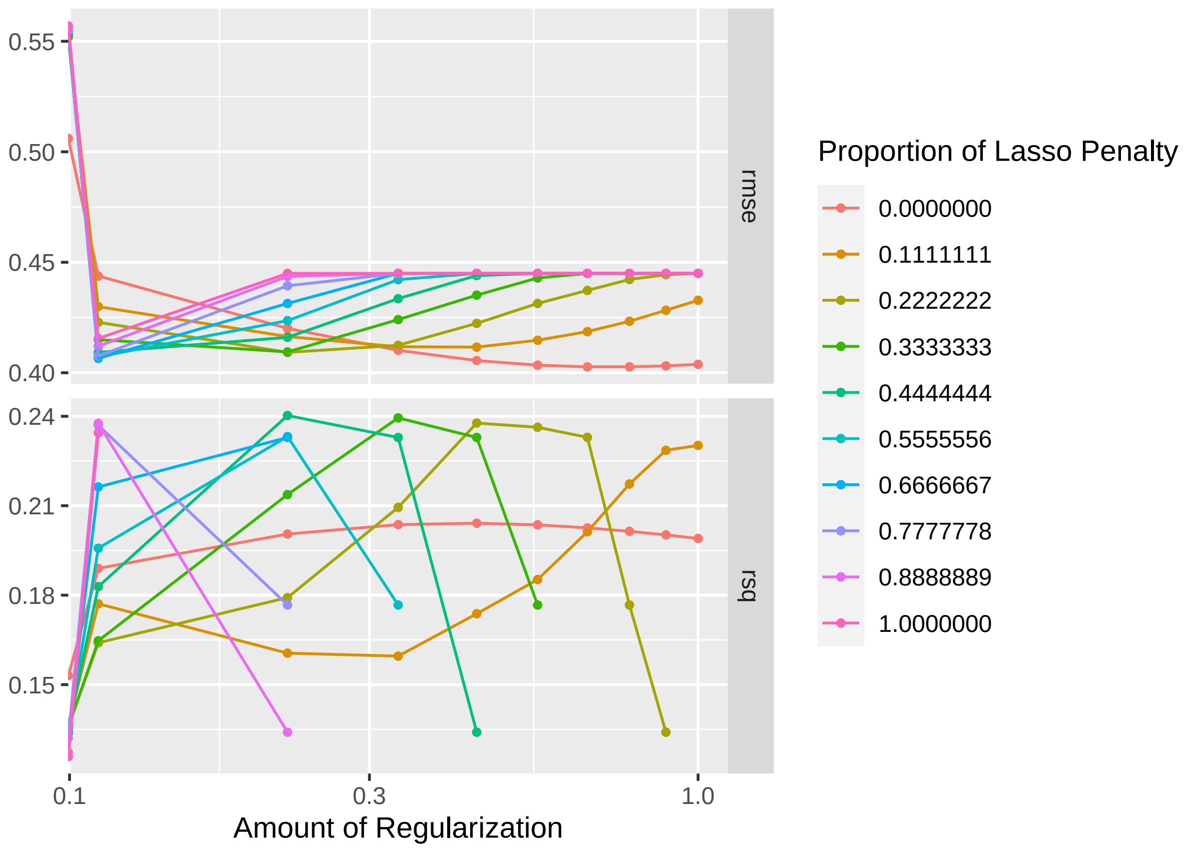
```
autoplot(shows_knn_tune)
```



```
autoplot(shows_en_tune)
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```



```
movies_knn_metrics <- collect_metrics(movies_knn_tune)
arrange(movies_knn_metrics[movies_knn_metrics$.metric=="rmse",c("neighbors", "mean", "std_err")], mean)
```

neighbors	mean	std_err
10	0.4438997	0.0202075
9	0.4480136	0.0198166
8	0.4532550	0.0191828
7	0.4599295	0.0179252
6	0.4680495	0.0152195
5	0.4779891	0.0116661
4	0.4917164	0.0083494
3	0.5102190	0.0047534
2	0.5414257	0.0033440
1	0.5888036	0.0131540

```
movies_lm_metrics <- collect_metrics(movies_lm_tune)
movies_lm_metrics[movies_lm_metrics$.metric=="rmse",c("mean", "std_err")] %>% kable()
```

mean	std_err
7385.771	7383.88

```
movies_en_metrics <- collect_metrics(movies_en_tune)
arrange(movies_en_metrics[movies_en_metrics$.metric=="rmse",c("penalty", "mixture", "mean", "std_err")]
```

penalty	mixture	mean	std_err
0.1111111	0.2222222	0.4017916	0.0166635
0.1111111	0.3333333	0.4022272	0.0180633
0.1111111	0.1111111	0.4057612	0.0138790
0.2222222	0.1111111	0.4062813	0.0166838
0.1111111	0.4444444	0.4069534	0.0189269
0.5555556	0.0000000	0.4087873	0.0153200

None of our models have a high R^2 value.