In-processing measures

Imports

```
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 0.2.0 --
```

```
## v broom        0.8.0     v recipes      0.2.0
## v dials        0.1.1     v rsample      0.1.1
## v dplyr        1.0.9     v tibble       3.1.7
## v ggplot2      3.3.6     v tidyr        1.2.0
## v infer        1.0.0     v tune         0.2.0
## v modeldata    0.1.1     v workflows    0.2.6
## v parsnip      0.2.1     v workflowsets 0.2.1
## v purrr        0.3.4     v yardstick    0.0.9
```

```
## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x recipes::step()  masks stats::step()
## * Learn how to get started at https://www.tidymodels.org/start/
```

```
library(discrim)
```

```
##
## Attaching package: 'discrim'
```

```
## The following object is masked from 'package:dials':
##
##     smoothness
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v readr   2.1.2     v forcats 0.5.1
## v stringr 1.4.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x readr::col_factor() masks scales::col_factor()
## x purrr::discard()    masks scales::discard()
## x dplyr::filter()     masks stats::filter()
## x stringr::fixed()    masks recipes::fixed()
## x dplyr::lag()        masks stats::lag()
## x readr::spec()       masks yardstick::spec()
```

```
source("../scripts/metrics_on_dataset.R")
```

Data

```
df <- read_rds("../data/selection.rds") %>%
  mutate(accepted = as.factor(accepted)) %>%
  select(-rating, -gender)
```

Modified Naive Bayes

Functions

```
# Modified Naive Bayes
df_disc <- function(df){
  fairness <- group_fairness(df, nationality, predicted)[[1]] %>%
    filter(predicted == "TRUE") %>%
    select(perc)
  max_val <- max(fairness)
  min_val <- min(fairness)
  max_val - min_val
}

adjust_fit <- function(cutoffs, direction){
  if (direction == "up"){
    cutoffs["Non_Dutch"] <- cutoffs["Non_Dutch"] - 0.01
  } else if (direction == "down"){
    cutoffs["Dutch"] <- cutoffs["Dutch"] + 0.01
  }
  cutoffs
}

nb_causal_discrimination <- function(df, fitted_model, cutoffs){
  # Determine the number of applicants who get a different outcome depending on their nationality
  pop_size <- nrow(df)

  # Flip nationalities
  inverted_df <- df %>%
    mutate(nationality = ifelse(nationality == "Dutch", "Non_Dutch", "Dutch"))

  predictions <- predict(fitted_model, df, type = "prob")[[".pred_TRUE"]]
  inverted_predictions <- predict(fitted_model, inverted_df, type = "prob")[[".pred_TRUE"]]

  # Add prediction column
  eval_df <- df %>%
    mutate(prediction = if_else(nationality == "Dutch",
                                predictions >= cutoffs["Dutch"],
                                predictions >= cutoffs["Non_Dutch"]),
           inv_prediction = if_else(nationality == "Dutch",
                                    inverted_predictions >= cutoffs["Non_Dutch"],
                                    inverted_predictions >= cutoffs["Dutch"]),
           different = prediction != inv_prediction)

  list(sum(eval_df$different)/pop_size, eval_df)
}
```

```
fitted_model <- naive_Bayes() %>%
  fit(accepted ~ nationality + test_score + english_cert + extracurricular, df)
df$predicted <- predict(fitted_model, df)
disc <- df_disc(df)
```

```
## ‘summarise()‘ has grouped output by ’nationality’. You can override using the
## ‘.groups‘ argument.
```

```
cutoffs <- c(Dutch = 0.5, Non_Dutch = 0.5)
print(disc)
```

```
## [1] 4.776572
```

```
while(disc > 1) {
  positive_label_count <- sum(df$accepted == "TRUE")
  predicted_positive_label_count <- sum(df$predicted == "TRUE")

  if (predicted_positive_label_count < positive_label_count) {
    cutoffs <- adjust_fit(cutoffs, "up")
  } else {
    cutoffs <- adjust_fit(cutoffs, "down")
  }

  predictions <- predict(fitted_model, df, type="prob")[[".pred_TRUE"]]

  df <- df %>%
    mutate(predicted = as.factor(if_else(nationality == "Dutch",
                                        predictions >= cutoffs["Dutch"],
                                        predictions >= cutoffs["Non_Dutch"])))

  new_disc <- df_disc(df)
  print(disc)
  # if (new_disc == disc) {
  #   print("no improvement")
  #   break
  # } else {
  disc <- new_disc
  # }

}
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 4.776572
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 4.18015
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 4.18015
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 3.981343
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 3.981343
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 3.981343
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 3.981343
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 3.981343
```

```
## `summarise()` has grouped output by 'nationality'. You can override using the
## `.groups` argument.
```

```
## [1] 3.981343
## ‘summarise()‘ has grouped output by ’nationality’. You can override using the
## ‘.groups‘ argument.
```

```
## [1] 3.186115
```

```
## ‘summarise()‘ has grouped output by ’nationality’. You can override using the
## ‘.groups‘ argument.
```

```
## [1] 3.186115
```

```
## ‘summarise()‘ has grouped output by ’nationality’. You can override using the
## ‘.groups‘ argument.
```

```
## [1] 3.186115
```

```
## ‘summarise()‘ has grouped output by ’nationality’. You can override using the
## ‘.groups‘ argument.
```

```
## [1] 2.390886
```

```
## ‘summarise()‘ has grouped output by ’nationality’. You can override using the
## ‘.groups‘ argument.
```

```
## [1] 2.390886
```

```
## ‘summarise()‘ has grouped output by ’nationality’. You can override using the
## ‘.groups‘ argument.
```

```
## [1] 2.390886
```

```
print(cutoffs)
```

```
##     Dutch Non_Dutch
##      0.50      0.35
```

```
print("Modified Naive Bayes")
```

```
## [1] "Modified Naive Bayes"
```

```
print("Group fairness")
```

```
## [1] "Group fairness"
```

```
print(group_fairness(df, nationality, predicted)[[1]])
```

```
## ‘summarise()‘ has grouped output by ’nationality’. You can override using the
## ‘.groups‘ argument.
## # A tibble: 4 x 4
##   nationality predicted total  perc
##   <fct>       <fct>     <int> <dbl>
## 1 Dutch       FALSE       412  82.9
## 2 Dutch       TRUE         85  17.1
## 3 Non_Dutch   FALSE       422  83.9
## 4 Non_Dutch   TRUE         81  16.1
```

```
print("Causal Discrimination")
```

```
## [1] "Causal Discrimination"
```

```
causal_disc <- nb_causal_discrimination(df, fitted_model, cutoffs)
print(causal_disc[[1]])
```

```
## [1] 0.014
```

Fairness classification

See the python folder in scripts