Assignment5

Yukta Chavan

12/04/2023

Data Cleaning

```
affairs <- read_csv("E:/dt/affairs.csv")
```

Q1

```
## Rows: 601 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): sex, child
## dbl (7): affair, age, ym, religious, education, occupation, rate
##
## 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.
```

```
head(affairs)
```

```
## # A tibble: 6 x 9
##   affair sex      age    ym child religious education occupation  rate
##    <dbl> <chr>  <dbl> <dbl> <chr>     <dbl>     <dbl>      <dbl> <dbl>
## 1      0 male      37 10    no            3        18          7     4
## 2      0 female    27  4    no            4        14          6     4
## 3      0 female    32 15    yes           1        12          1     4
## 4      0 male      57 15    yes           5        18          6     5
## 5      0 male      22  0.75 no            2        17          6     3
## 6      0 female    32  1.5  no            2        17          5     5
```

Q2:

The outcome variable is affair and the predictor variables are sex, age, ym, child, religious, education, occupation, and rate

```
skim(affairs)
```

Q3:

Table 1: Data summary

Name	affairs
Number of rows	601
Number of columns	9
Column type frequency:	
character	2
numeric	7
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
sex	0	1	4	6	0	2	0
child	0	1	2	3	0	2	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
affair	0	1	0.25	0.43	0.00	0	0	0	1	
age	0	1	32.49	9.29	17.50	27	32	37	57	
ym	0	1	8.18	5.57	0.12	4	7	15	15	
religious	0	1	3.12	1.17	1.00	2	3	4	5	
education	0	1	16.17	2.40	9.00	14	16	18	20	
occupation	0	1	4.19	1.82	1.00	3	5	6	7	
rate	0	1	3.93	1.10	1.00	3	4	5	5	

No missing values found. We have 601 observations and 9 variables. we found one incorrect variable that is affair because it shows 0 and 1 (numerical variable) instead of yes or no (categorical variable).

```
affairs <- affairs %>%
  mutate( affair = case_when( affair == 1 ~ "yes", TRUE ~ "no" ) ) %>%
  mutate_if( is.character, factor )
```

Q4

```
affairs %>% skim()
```

Q5

Table 4: Data summary

Name	Piped data
Number of rows	601
Number of columns	9
Column type frequency:	
factor	3
numeric	6
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
affair	0	1	FALSE	2	no: 451, yes: 150
sex	0	1	FALSE	2	fem: 315, mal: 286
child	0	1	FALSE	2	yes: 430, no: 171

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
age	0	1	32.49	9.29	17.50	27	32	37	57	
ym	0	1	8.18	5.57	0.12	4	7	15	15	
religious	0	1	3.12	1.17	1.00	2	3	4	5	
education	0	1	16.17	2.40	9.00	14	16	18	20	
occupation	0	1	4.19	1.82	1.00	3	5	6	7	
rate	0	1	3.93	1.10	1.00	3	4	5	5	

People responded as having had an affair: 150
People responded to having had children: 430

The mean age of respondents: 32.49 The mean response on the religious scale:3.12

Exploratory Analysis

```
affairs %>%
  count( affair, sex ) %>%
  pivot_wider( names_from = sex, values_from = n ) %>%
  mutate( prop = female / ( male + female ) )
```

```
## # A tibble: 2 x 4
##   affair female  male  prop
##   <fct>   <int> <int> <dbl>
## 1 no        243   208 0.539
## 2 yes        72    78 0.48
```

There is more proportion of female who said no to having an affair compared to yes.

```
affairs %>%
  count( affair,child ) %>%
  pivot_wider( names_from =child, values_from = n ) %>%
  mutate( prop = yes / ( no + yes) )
```

```
## # A tibble: 2 x 4
##   affair    no   yes  prop
##   <fct>  <int> <int> <dbl>
## 1 no       144   307 0.681
## 2 yes       27   123 0.82
```

Based on this, Yes, you are more likely to have children if you have an affair.

Split and preprocess

```
library(tidymodels)
```

Q1

```
## -- Attaching packages --------------------------------------- tidymodels 1.0.0 --

## v broom        1.0.4     v rsample      1.1.1
## v dials        1.1.0     v tune         1.0.1
## v infer        1.0.4     v workflows    1.1.3
## v modeldata    1.1.0     v workflowsets 1.0.0
## v parsnip      1.0.4     v yardstick    1.1.0
## v recipes      1.0.5

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

```
set.seed(1234)
affairs_split <- initial_split(affairs)
affairs_split
```

```
## <Training/Testing/Total>
## <450/151/601>
```

observations in Training set :450 observations in Testing set: 151

```
training_set <- training(affairs_split)
testing_set <- testing(affairs_split)

head(training_set)
```

Q2

```
## # A tibble: 6 x 9
##   affair sex      age     ym child religious education occupation  rate
##   <fct>  <fct>  <dbl>  <dbl> <fct>     <dbl>     <dbl>      <dbl> <dbl>
## 1 no     female    42 15     yes           3        14          1     3
## 2 no     female    27 10     yes           5        14          1     5
## 3 no     male      22  1.5   no            2        18          5     3
## 4 no     male      37 10     yes           1        16          6     4
## 5 no     female    22  0.125 no            4        12          4     5
## 6 no     male      32  4     no            1        20          6     5
```

Q3 step_downsample is a function from the themis package that performs downsampling on a data frame. Downsampling is a technique used to address class imbalance, where one class has many more observations than the other. step_downsample randomly samples observations from the majority class so that the two classes are more balanced.

```
library(themis)
library(recipes)
affairs_recipe <- recipe(affair ~ ., data = training_set) %>%
  step_downsample(affair) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
  step_normalize(all_predictors()) %>%
  prep()
```

Q4

```
preprocessed_train_data <- juice(affairs_recipe)
preprocessed_test_data <- affairs_recipe %>% bake(testing_set)
```

Q5

```
library(skimr)
skim(preprocessed_train_data)
```

Q6

Table 7: Data summary

Name	preprocessed_train_data
Number of rows	234
Number of columns	9
Column type frequency:	
factor	1
numeric	8
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
affair	0	1	FALSE	2	no: 117, yes: 117

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
age	0	1	0	1	-1.67	-0.57	0.01	0.58	2.89	
ym	0	1	0	1	-1.48	-0.77	-0.22	1.25	1.25	
religious	0	1	0	1	-1.72	-0.85	0.03	0.90	1.77	
education	0	1	0	1	-2.96	-0.89	-0.07	0.76	1.59	
occupation	0	1	0	1	-1.77	-0.67	0.43	0.98	1.53	
rate	0	1	0	1	-2.40	-0.66	0.20	1.07	1.07	
sex_male	0	1	0	1	-0.93	-0.93	-0.93	1.07	1.07	
child_yes	0	1	0	1	-1.68	-1.68	0.59	0.59	0.59	

we used downsample technique on the affair variable. Based on analysis, we can say that the sex and child variable can be seen as dummy variable and then have normalized all the predictors.

Tune and fit a model

```
library(tidymodels)
knn_spec <- nearest_neighbor(neighbors = tune()) %>%
             set_engine("kknn") %>%
             set_mode("classification")
knn_spec
```

Q1

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

```
set.seed(1234)
cv_splits <- vfold_cv(preprocessed_train_data, v = 5)
```

Q2

```
k_grid <- grid_regular(neighbors(range = c(5, 75)),levels = 25)
```

Q3

```
# Tune the model
library(kknn)
knn_tune <- tune_grid(object = knn_spec,
                      preprocessor = recipe(affair ~ ., data =  preprocessed_train_data),
                      resamples = cv_splits,
                      grid = k_grid)
```

Q4

```
best_knn <- select_best(knn_tune, "accuracy")

best_knn
```

Q5

```
## # A tibble: 1 x 2
##   neighbors .config
##       <int> <chr>
## 1        37 Preprocessor1_Model12
```

Value of k that gives the best accuracy based on our tuned mode: 37

```
final_knn <- finalize_model(knn_spec, best_knn)

final_knn
```

Q6

```
## K-Nearest Neighbor Model Specification (classification)
##
## Main Arguments:
##   neighbors = 37
##
## Computational engine: kknn
```

```
affairs_knn <- final_knn %>% fit(affair ~ ., data = preprocessed_train_data)
affairs_knn
```

Q7

```
## parsnip model object
##
##
## Call:
## kknn::train.kknn(formula = affair ~ ., data = data, ks = min_rows(37L,     data, 5))
##
## Type of response variable: nominal
## Minimal misclassification: 0.3974359
## Best kernel: optimal
## Best k: 37
```

Evaluation

```
predictions <- predict(affairs_knn, preprocessed_test_data, type = "class")
head(predictions)
```

Q1

```
## # A tibble: 6 x 1
##   .pred_class
##   <fct>
## 1 no
## 2 no
## 3 no
## 4 yes
## 5 yes
## 6 no
```

```
library(dplyr)
predictions <- bind_cols(predictions, select(preprocessed_test_data, affair = affair))
head(predictions)
```

Q2

```
## # A tibble: 6 x 2
##   .pred_class affair
##   <fct>       <fct>
## 1 no          no
## 2 no          no
## 3 no          no
## 4 yes         no
## 5 yes         no
## 6 no          no
```

```
predictions %>%
  conf_mat(truth = affair, estimate = .pred_class)
```

Q3

```
##           Truth
## Prediction no yes
##        no  81  11
##        yes 37  22
```

```
sensitivity_affair <- 81 / ( 81 + 37 )
sensitivity_affair
```

Q4

```
## [1] 0.6864407
```

```
specificity_affair <- 22 / ( 11 + 22 )
specificity_affair
```

```
## [1] 0.6666667
```

```
bono_info <- tibble(
  sex = "male",
  age = 47,
```

```
  ym = 15,
  child = "no",
  religious = 2,
  occupation = 6,
  education = 20,
  rate = 5)
bono_info
```

Q5

```
## # A tibble: 1 x 8
##   sex     age    ym child religious occupation education  rate
##   <chr> <dbl> <dbl> <chr>     <dbl>      <dbl>     <dbl> <dbl>
## 1 male     47    15 no            2          6        20     5
```

```
preprocessed_bono_info <- affairs_recipe %>% bake(new_data = bono_info)
```

```
## Warning:  There were 2 columns that were factors when the recipe was prepped:
##  'sex', 'child'.
##  This may cause errors when processing new data.
```

```
preprocessed_bono_info
```

```
## # A tibble: 1 x 8
##     age    ym religious education occupation  rate sex_male child_yes
##   <dbl> <dbl>     <dbl>     <dbl>      <dbl> <dbl>    <dbl>     <dbl>
## 1  1.74  1.25    -0.848      1.59      0.982  1.07     1.07     -1.68
```

```
predict(affairs_knn, new_data = preprocessed_bono_info, type = "prob")
```

```
## # A tibble: 1 x 2
##   .pred_no .pred_yes
##      <dbl>     <dbl>
## 1    0.375     0.625
```

Answer 5d- It is important to note that any predictive model can never be 100% accurate and there is always a possibility of error. Therefore, it would not be appropriate to make a prediction about someone's personal life without their explicit consent, and it is not ethical to share such information with anyone else without their consent. Therefore, it would not be appropriate to go to Bono's partner with the prediction of whether Bono will have an affair or not.