Assignment 17

Allison Tessman

2024-04-25

Question 1. In this question, we will work with the Data Analyst Jobs dataset

* Use the following code to create binary variable on rating, called Rating2
* Train and test a model to predict whether the rating of the job is high or low (target variable is Rating2) based on the job descriptions.

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.3 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ ggplot2 3.5.0 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.2 ✔ tidyr 1.3.0  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

df = read\_csv('https://bryantstats.github.io/math475/assignments/DataAnalyst3.csv')

## Rows: 2253 Columns: 16  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (13): Job.Title, Salary.Estimate, Job.Description, Company.Name, Locatio...  
## dbl (3): ...1, Rating, Founded  
##   
## 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.

# calculate the median rating  
m = median(df$Rating)  
  
# Create a binary rating variable  
df$Rating2 = if\_else(df$Rating > m, "high", "low")  
  
#Convert text variables to numeric  
library(caret)

## Loading required package: lattice  
##   
## Attaching package: 'caret'  
##   
## The following object is masked from 'package:purrr':  
##   
## lift

library(themis)

## Loading required package: recipes  
##   
## Attaching package: 'recipes'  
##   
## The following object is masked from 'package:stringr':  
##   
## fixed  
##   
## The following object is masked from 'package:stats':  
##   
## step

library(textrecipes)  
library(tidyverse)  
library(ranger)  
  
df <- df %>%   
 select(Rating2, Job.Description) %>%   
 rename(target = Rating2,  
 texts = Job.Description) %>%   
 drop\_na()  
  
a <- recipe(target ~.,  
 data = df) %>%   
 step\_tokenize(texts) %>%   
 step\_tokenfilter(texts, max\_tokens = 100) %>%   
 step\_tfidf(texts) %>%   
 step\_normalize(all\_numeric\_predictors()) %>%   
 step\_dummy(all\_nominal\_predictors()) %>%   
 step\_smote(target) %>%   
 prep()  
df <- juice(a)  
df

## # A tibble: 2,324 × 101  
## tfidf\_texts\_a tfidf\_texts\_ability tfidf\_texts\_across tfidf\_texts\_all  
## <dbl> <dbl> <dbl> <dbl>  
## 1 0.298 -0.313 0.407 -0.396   
## 2 -0.525 -0.409 -0.592 -0.676   
## 3 0.427 -0.799 -0.592 -0.313   
## 4 -0.302 -0.799 0.0534 0.0475  
## 5 0.354 -0.799 0.379 0.412   
## 6 1.16 0.533 2.15 -0.676   
## 7 -0.0612 -0.241 4.00 -0.676   
## 8 0.634 -0.799 -0.592 -0.676   
## 9 1.05 0.352 1.77 -0.676   
## 10 0.406 -0.314 0.404 1.00   
## # ℹ 2,314 more rows  
## # ℹ 97 more variables: tfidf\_texts\_an <dbl>, tfidf\_texts\_analysis <dbl>,  
## # tfidf\_texts\_analyst <dbl>, tfidf\_texts\_analytical <dbl>,  
## # tfidf\_texts\_analytics <dbl>, tfidf\_texts\_analyze <dbl>,  
## # tfidf\_texts\_and <dbl>, tfidf\_texts\_are <dbl>, tfidf\_texts\_as <dbl>,  
## # tfidf\_texts\_at <dbl>, tfidf\_texts\_be <dbl>, tfidf\_texts\_business <dbl>,  
## # tfidf\_texts\_by <dbl>, tfidf\_texts\_communication <dbl>, …

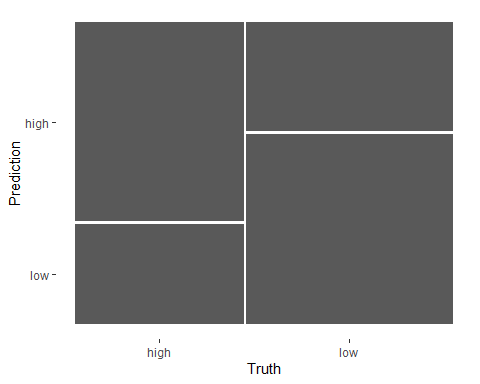
# Using Caret for modeling  
set.seed(2021)  
splitIndex <- createDataPartition(df$target, p = .7,   
 list = FALSE)  
df\_train <- df[ splitIndex,]  
df\_test <- df[-splitIndex,]  
  
model <- ranger(target ~ ., data = df\_train)  
  
# Testing the model  
pred <- predict(model, df\_test)$predictions  
  
cm <- confusionMatrix(data = pred, reference = df\_test$target)  
cm$overall[1]

## Accuracy   
## 0.6494253

#Accuracy = 0.6393678  
  
d = data.frame(pred = pred, obs = df\_test$target)  
library(yardstick)

##   
## Attaching package: 'yardstick'  
##   
## The following objects are masked from 'package:caret':  
##   
## precision, recall, sensitivity, specificity  
##   
## The following object is masked from 'package:readr':  
##   
## spec

d %>% conf\_mat(pred, obs) %>% autoplot



* Adding more predictors to the model to improve the testing accuracy of the model

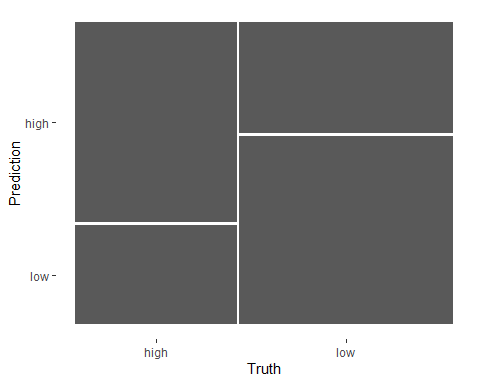
library(tidyverse)  
  
df = read\_csv('https://bryantstats.github.io/math475/assignments/DataAnalyst3.csv')

## Rows: 2253 Columns: 16  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (13): Job.Title, Salary.Estimate, Job.Description, Company.Name, Locatio...  
## dbl (3): ...1, Rating, Founded  
##   
## 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.

# calculate the median rating  
m = median(df$Rating)  
  
# Create a binary rating variable  
df$Rating2 = if\_else(df$Rating > m, "high", "low")  
  
#Convert text variables to numeric  
library(caret)  
library(themis)  
library(textrecipes)  
library(tidyverse)  
library(ranger)  
  
df <- df %>%   
 select(Rating2, Job.Description, Location, Industry) %>%   
 rename(target = Rating2,  
 texts = Job.Description) %>%   
 drop\_na()  
  
# Convert text data to numeric variables  
a <- recipe(target ~.,  
 data = df) %>%   
 step\_tokenize(texts) %>%   
 step\_tokenfilter(texts, max\_tokens = 100) %>%   
 step\_tfidf(texts) %>%   
 step\_normalize(all\_numeric\_predictors()) %>%   
 step\_dummy(all\_nominal\_predictors()) %>%   
 step\_smote(target) %>%   
 prep()  
df <- juice(a)  
  
set.seed(2021)  
splitIndex <- createDataPartition(df$target, p = .7,   
 list = FALSE)  
df\_train <- df[ splitIndex,]  
df\_test <- df[-splitIndex,]  
  
model <- ranger(target ~ ., data = df\_train)  
  
pred <- predict(model, df\_test)$predictions  
  
cm <- confusionMatrix(data = pred, reference = df\_test$target)  
cm$overall[1]

## Accuracy   
## 0.6465517

#Accuracy = 0.6465517  
#Accuracy got worse by adding more variables   
  
d = data.frame(pred = pred, obs = df\_test$target)  
library(yardstick)  
d %>% conf\_mat(pred, obs) %>% autoplot



Note: The accuracy got worse when more variables were added in (location and industry)

Question 2. Redo Question 1 on your own data (you can decide the target for your model)

library(tidyverse)  
  
df = read\_csv('https://bryantstats.github.io/math475/assignments/amazon\_reviews.csv')

## Rows: 34660 Columns: 21  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (12): id, name, asins, brand, categories, keys, manufacturer, reviews.d...  
## dbl (3): reviews.id, reviews.numHelpful, reviews.rating  
## lgl (4): reviews.didPurchase, reviews.doRecommend, reviews.userCity, revie...  
## dttm (2): reviews.date, reviews.dateAdded  
##   
## 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.

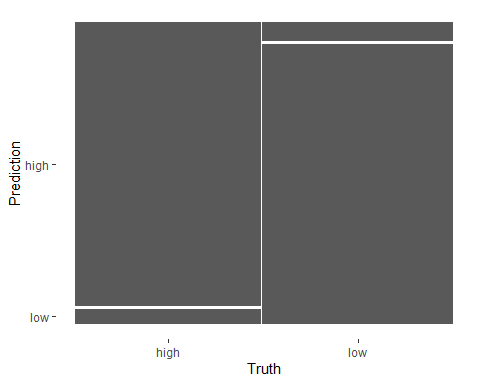
# Create a binary rating variable  
df$Rating2 = if\_else(df$reviews.rating > 3, "high", "low")  
  
#Convert text variables to numeric  
library(caret)  
library(themis)  
library(textrecipes)  
library(tidyverse)  
library(ranger)  
  
df <- df %>%   
 select(Rating2, reviews.text) %>%   
 rename(target = Rating2,  
 texts = reviews.text) %>%   
 drop\_na()  
  
a <- recipe(target ~.,  
 data = df) %>%   
 step\_tokenize(texts) %>%   
 step\_tokenfilter(texts, max\_tokens = 30) %>%   
 step\_tfidf(texts) %>%   
 step\_normalize(all\_numeric\_predictors()) %>%   
 step\_dummy(all\_nominal\_predictors()) %>%   
 step\_smote(target) %>%   
 prep()  
df <- juice(a)  
df

## # A tibble: 64,630 × 31  
## tfidf\_texts\_a tfidf\_texts\_amazon tfidf\_texts\_and tfidf\_texts\_as  
## <dbl> <dbl> <dbl> <dbl>  
## 1 -0.667 -0.299 -0.0531 -0.276  
## 2 1.39 -0.299 0.876 2.88   
## 3 -0.667 -0.299 -0.0531 -0.276  
## 4 -0.141 -0.299 0.472 0.530  
## 5 -0.130 0.234 0.501 0.136  
## 6 1.51 1.14 -0.849 -0.276  
## 7 -0.667 -0.299 1.45 -0.276  
## 8 0.283 -0.299 0.743 1.18   
## 9 0.568 -0.299 -0.332 0.671  
## 10 -0.667 -0.299 1.22 -0.276  
## # ℹ 64,620 more rows  
## # ℹ 27 more variables: tfidf\_texts\_but <dbl>, tfidf\_texts\_can <dbl>,  
## # tfidf\_texts\_easy <dbl>, tfidf\_texts\_for <dbl>, tfidf\_texts\_great <dbl>,  
## # tfidf\_texts\_have <dbl>, tfidf\_texts\_i <dbl>, tfidf\_texts\_in <dbl>,  
## # tfidf\_texts\_is <dbl>, tfidf\_texts\_it <dbl>, tfidf\_texts\_love <dbl>,  
## # tfidf\_texts\_my <dbl>, tfidf\_texts\_not <dbl>, tfidf\_texts\_of <dbl>,  
## # tfidf\_texts\_on <dbl>, tfidf\_texts\_so <dbl>, tfidf\_texts\_tablet <dbl>, …

# Using Caret for modeling  
set.seed(2021)  
splitIndex <- createDataPartition(df$target, p = .7,   
 list = FALSE)  
df\_train <- df[ splitIndex,]  
df\_test <- df[-splitIndex,]  
  
model <- ranger(target ~ ., data = df\_train)  
  
# Testing the model  
pred <- predict(model, df\_test)$predictions  
  
cm <- confusionMatrix(data = pred, reference = df\_test$target)  
cm$overall[1]

## Accuracy   
## 0.9431607

#Accuracy = 0.6393678  
  
d = data.frame(pred = pred, obs = df\_test$target)  
library(yardstick)  
d %>% conf\_mat(pred, obs) %>% autoplot



* Adding more predictors to the model to improve the testing accuracy of the model

library(tidyverse)  
  
df = read\_csv('https://bryantstats.github.io/math475/assignments/amazon\_reviews.csv')

## Rows: 34660 Columns: 21  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (12): id, name, asins, brand, categories, keys, manufacturer, reviews.d...  
## dbl (3): reviews.id, reviews.numHelpful, reviews.rating  
## lgl (4): reviews.didPurchase, reviews.doRecommend, reviews.userCity, revie...  
## dttm (2): reviews.date, reviews.dateAdded  
##   
## 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.

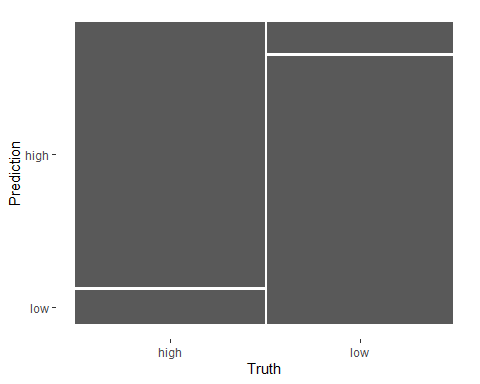
# Create a binary rating variable  
df$Rating2 = if\_else(df$reviews.rating > 3, "high", "low")  
  
df <- df[1:5000,]  
  
#Convert text variables to numeric  
library(caret)  
library(themis)  
library(textrecipes)  
library(tidyverse)  
library(ranger)  
  
df <- df %>%   
 select(Rating2, reviews.text, reviews.title) %>%   
 rename(target = Rating2,  
 texts = reviews.text) %>%   
 drop\_na()  
  
# Convert text data to numeric variables  
a <- recipe(target ~.,  
 data = df) %>%   
 step\_tokenize(texts) %>%   
 step\_tokenfilter(texts, max\_tokens = 30) %>%   
 step\_tfidf(texts) %>%   
 step\_normalize(all\_numeric\_predictors()) %>%   
 step\_dummy(all\_nominal\_predictors()) %>%   
 step\_smote(target) %>%   
 prep()

## New names:  
## • `reviews.title\_Awesome.tablet.` -> `reviews.title\_Awesome.tablet....255`  
## • `reviews.title\_Awesome.tablet.` -> `reviews.title\_Awesome.tablet....258`  
## • `reviews.title\_Excellent.tablet.for.the.price.` ->  
## `reviews.title\_Excellent.tablet.for.the.price....676`  
## • `reviews.title\_Excellent.tablet.for.the.price.` ->  
## `reviews.title\_Excellent.tablet.for.the.price....677`  
## • `reviews.title\_Good.product.` -> `reviews.title\_Good.product....1004`  
## • `reviews.title\_Good.product.` -> `reviews.title\_Good.product....1006`  
## • `reviews.title\_Good.tablet.` -> `reviews.title\_Good.tablet....1079`  
## • `reviews.title\_Good.tablet.` -> `reviews.title\_Good.tablet....1084`  
## • `reviews.title\_Great.Charger.` -> `reviews.title\_Great.Charger....1204`  
## • `reviews.title\_Great.Charger.` -> `reviews.title\_Great.Charger....1205`  
## • `reviews.title\_Great.e.reader` -> `reviews.title\_Great.e.reader...1246`  
## • `reviews.title\_Great.E.Reader` -> `reviews.title\_Great.E.Reader...1248`  
## • `reviews.title\_Great.e.reader.` -> `reviews.title\_Great.e.reader....1253`  
## • `reviews.title\_Great.e.reader.` -> `reviews.title\_Great.e.reader....1254`  
## • `reviews.title\_Great.e.reader` -> `reviews.title\_Great.e.reader...1255`  
## • `reviews.title\_Great.E.Reader` -> `reviews.title\_Great.E.Reader...1256`  
## • `reviews.title\_Great.for.anyone.` -> `reviews.title\_Great.for.anyone....1296`  
## • `reviews.title\_Great.for.anyone.` -> `reviews.title\_Great.for.anyone....1297`  
## • `reviews.title\_Great.for.my.kids.` ->  
## `reviews.title\_Great.for.my.kids....1360`  
## • `reviews.title\_Great.for.my.kids.` ->  
## `reviews.title\_Great.for.my.kids....1361`  
## • `reviews.title\_Great.for.reading.` ->  
## `reviews.title\_Great.for.reading....1388`  
## • `reviews.title\_Great.for.reading.` ->  
## `reviews.title\_Great.for.reading....1391`  
## • `reviews.title\_Great.for.the.price.` ->  
## `reviews.title\_Great.for.the.price....1411`  
## • `reviews.title\_Great.for.the.price.` ->  
## `reviews.title\_Great.for.the.price....1412`  
## • `reviews.title\_Great.gift.` -> `reviews.title\_Great.gift....1456`  
## • `reviews.title\_Great.gift.` -> `reviews.title\_Great.gift....1458`  
## • `reviews.title\_Great.Price.for.the.Amazon...Fire...7` ->  
## `reviews.title\_Great.Price.for.the.Amazon...Fire`  
## • `reviews.title\_Great.product.` -> `reviews.title\_Great.product....1619`  
## • `reviews.title\_Great.product.` -> `reviews.title\_Great.product....1628`  
## • `reviews.title\_Great.tablet.for.the.money.` ->  
## `reviews.title\_Great.tablet.for.the.money....1824`  
## • `reviews.title\_Great.tablet.for.the.money.` ->  
## `reviews.title\_Great.tablet.for.the.money....1826`  
## • `reviews.title\_Great.tablet.for.the.price.` ->  
## `reviews.title\_Great.tablet.for.the.price....1836`  
## • `reviews.title\_Great.tablet.for.the.price.` ->  
## `reviews.title\_Great.tablet.for.the.price....1837`  
## • `reviews.title\_Great.tablet.` -> `reviews.title\_Great.tablet....1867`  
## • `reviews.title\_Great.Tablet.` -> `reviews.title\_Great.Tablet....1868`  
## • `reviews.title\_Great.tablet.` -> `reviews.title\_Great.tablet....1882`  
## • `reviews.title\_Great.Tablet.` -> `reviews.title\_Great.Tablet....1883`  
## • `reviews.title\_Great.` -> `reviews.title\_Great....1934`  
## • `reviews.title\_Great.` -> `reviews.title\_Great....1942`  
## • `reviews.title\_I.love.it.` -> `reviews.title\_I.love.it....2028`  
## • `reviews.title\_I.love.it.` -> `reviews.title\_I.love.it....2033`  
## • `reviews.title\_Love.it.` -> `reviews.title\_Love.it....2287`  
## • `reviews.title\_Love.it.` -> `reviews.title\_Love.it....2296`  
## • `reviews.title\_Love.this.tablet.` -> `reviews.title\_Love.this.tablet....2354`  
## • `reviews.title\_Love.this.tablet.` -> `reviews.title\_Love.this.tablet....2355`  
## • `reviews.title\_My.daughter.loved.it.` ->  
## `reviews.title\_My.daughter.loved.it....2408`  
## • `reviews.title\_My.daughter.loved.it.` ->  
## `reviews.title\_My.daughter.loved.it....2409`  
## • `reviews.title\_My.wife.loves.it.` -> `reviews.title\_My.wife.loves.it....2447`  
## • `reviews.title\_My.wife.loves.it.` -> `reviews.title\_My.wife.loves.it....2449`  
## • `reviews.title\_Nice.fire.readers...2` -> `reviews.title\_Nice.fire.readers`  
## • `reviews.title\_Nice.tablet.` -> `reviews.title\_Nice.tablet....2553`  
## • `reviews.title\_Nice.tablet.` -> `reviews.title\_Nice.tablet....2558`  
## • `reviews.title\_Value.for.money.` -> `reviews.title\_Value.for.money....3091`  
## • `reviews.title\_Value.for.money.` -> `reviews.title\_Value.for.money....3092`  
## • `reviews.title\_Wife.loves.it.` -> `reviews.title\_Wife.loves.it....3193`  
## • `reviews.title\_Wife.loves.it.` -> `reviews.title\_Wife.loves.it....3194`  
## • `reviews.title\_Works.great.` -> `reviews.title\_Works.great....3247`  
## • `reviews.title\_Works.great.` -> `reviews.title\_Works.great....3250`

df <- juice(a)  
  
set.seed(2021)  
splitIndex <- createDataPartition(df$target, p = .7,   
 list = FALSE)  
df\_train <- df[ splitIndex,]  
df\_test <- df[-splitIndex,]  
  
model <- ranger(target ~ ., data = df\_train)  
  
pred <- predict(model, df\_test)$predictions  
  
cm <- confusionMatrix(data = pred, reference = df\_test$target)  
cm$overall[1]

## Accuracy   
## 0.8911786

#Accuracy = 0.6465517  
#Accuracy got worse by adding more variables   
  
d = data.frame(pred = pred, obs = df\_test$target)  
library(yardstick)  
d %>% conf\_mat(pred, obs) %>% autoplot



*NOTE: For the last chunk of code, I had issues with not even enough memory to do the full dataset with adding in more variables, so I took a subset of the data to get it to run!*