DSApps 2022 @ TAU: Assignment 6

The Price is Right!

Giora Simchoni

2022-05-22

Contents

Welcome

Welcome to Assignment 6 in R!

Remember:

- You can play with the assignment in Playground mode, but:
- Only your private Github repository assigned to you by the course admin will be cloned and graded (Submission mode, see instructions here)
- Like any other University assignment, your work should remain private
- You need to `git clone` your private Github repository locally as explained here
- You need to uncomment the starter code inside the chunk, replace the `### YOUR CODE HERE ###`, run the chunk and see that you're getting the expected result
- Pay attention to what you're asked to do and the required output
- For example, using a *different* function than the one you were specifically asked to use, will decrease your score (unless you amaze me)
- Your notebook should run smoothly from start to end if someone presses in the RStudio toolbar Run –> Restart R and Run All Chunks
- When you're done knit the entire notebook into a html file, this is the file that would be graded
- You can add other files but do not delete any files
- Commit your work and push to your private Github repository as explained here

This assignemtnt is due: 22/5 23:59

Packages

These are the packages you will need. If you don't have them, you need to uncomment the `install.packages()` line and install them first (you can also just copy this command to the R console and do it there if you don't want all the output printed in this notebook).

When you load the packages you may see different kinds of messages or warnings, skim them:

```
# install.packages(c("tidyverse", "tidymodels", "glmnet"))
library(tidyverse)
library(tidymodels)
library(glmnet)
```

The Price is Right Challenge

This assignment we're having our very own mini Kaggle-like challenge!

You! Are going! To predict the price of...

Women's Shoes!

That's right. I have scraped ebay for the price, title and some attributes of over 15K women's shoes (I did it with `rvest`, I doubt if you can call it ethical, so I won't share the script). But for ~1K of the shoes, the price is for me to know and for you to predict!

You will have 7 attempts at submitting the predicted price of the hidden shoes, and whoever reaches the lowest RMSE - wins!

Be sure to visit our Leaderboard to see the best scores.

Basic Exploration

There are two datasets in the data folder of this challenge:

- `ebay_women_shoes_train.rds`: contains 14K pairs of shoes with `id`, `price` (in USD), `title`, `condition`, `brand`, `seller_notes`, `location` and many more attributes
- `ebay_women_shoes_test.rds`: contains 1,044 pairs of shoes with all of the above except for `price` which is safely hidden with me

```
shoes_train <- read_rds("data/ebay_women_shoes_train.rds")
shoes_test <- read_rds("data/ebay_women_shoes_test.rds")

dim(shoes_train)
```

```
## [1] 14000    42
```

```
dim(shoes_test)
```

```
## [1] 1044   41
```

```
glimpse(shoes_train)
```

```
## Rows: 14,000
## Columns: 42
## $ id                              <int> 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, ~
## $ title                           <chr> "Manolo Blahnik New Runway Auth Pink Bla~
## $ price                           <dbl> 261.43500, 21.66735, 64.68305, 28.59980,~
## $ brand                           <chr> "Manolo Blahnik", "Unbranded", NA, "Unbr~
## $ style                           <chr> "Mule", "Pump", NA, "Platform, Slingback~
## $ heel_type                       <chr> NA, NA, NA, NA, NA, NA, NA, NA, "Stilett~
## $ heel_height                     <chr> "Mid (2-2.9 in)", "Low (1-1.9 in)", NA, ~
## $ width                           <chr> "Medium (B, M)", NA, NA, NA, NA, NA, NA,~
## $ shoe_width                      <chr> "B", "B", NA, "B", "3'", "B", "B", NA, N~
## $ material                        <chr> NA, NA, NA, NA, NA, NA, NA, "100% Leathe~
## $ occasion                        <chr> "Party/Cocktail", "Party/Cocktail", NA, ~
## $ country_region_of_manufacture   <chr> "Italy", NA, NA, NA, NA, NA, "China", "I~
## $ lining_material                 <chr> NA, "Fabric", NA, "PVC", NA, NA, "FAUX L~
## $ upper_material                  <chr> "Fabric", "Faux Suede", NA, "PVC", "Pate~
## $ shoe_size                       <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ toe_shape                       <chr> "Pointed Toe", "Round Toe", NA, "Peep To~
## $ model                           <chr> NA, NA, NA, NA, "Christian Louboutin Dec~
## $ year_of_manufacture             <chr> NA, "2010-2019", NA, "2020-2029", NA, NA~
## $ size                            <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ colour                          <chr> NA, NA, NA, NA, NA, NA, "Clear NUDE", "B~
## $ color                           <chr> "Pink Black", NA, NA, NA, "Black", "Gold~
## $ main_colour                     <chr> NA, NA, NA, NA, NA, NA, NA, "Black", NA,~
## $ lining                          <chr> NA, NA, NA, NA, NA, NA, NA, "Leather", N~
## $ sole                            <chr> NA, NA, NA, NA, NA, NA, NA, "Leather", N~
```

```
## $ vintage                      <chr> "No", "Yes", NA, NA, NA, "No", "No", NA,~
## $ closure                      <chr> NA, "Buckle", NA, "Buckle", NA, NA, "SLI~
## $ pattern                      <chr> NA, "Solid", NA, "Solid", NA, NA, NA, NA~
## $ theme                        <chr> NA, NA, NA, NA, NA, "Metal", NA, NA, NA,~
## $ fastening                    <chr> NA, NA, NA, NA, NA, NA, NA, "Slip On", "~
## $ platform_height              <chr> NA, NA, NA, NA, NA, NA, "0.39 in", NA, N~
## $ location                     <chr> "Newport Beach, California, United State~
## $ n_sold                       <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
## $ n_watchers                   <dbl> NA, NA, NA, 21, NA, NA, NA, NA, NA, NA, ~
## $ free_shipping                <int> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0~
## $ longtime_member              <int> 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1~
## $ same_day_shipping            <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1~
## $ fast_safe_shipping           <int> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0~
## $ returns                      <int> 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0~
## $ feedback                     <int> 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0~
## $ condition                    <chr> "New without box", "New without box", "N~
## $ seller_notes                 <chr> ""New, elegant & luxurious mules heels."~
## $ category                     <chr> "heels", "heels", "heels", "heels", "hee~
```

Three categories:

```
shoes_train %>% count(category)
```

```
## # A tibble: 3 x 2
##   category     n
##   <chr>    <int>
## 1 comfort   5130
## 2 flats     3829
## 3 heels     5041
```

Four conditions, some `NA`:

```
shoes_train %>% count(condition)
```

```
## # A tibble: 5 x 2
##   condition            n
##   <chr>            <int>
## 1 New with box      3011
## 2 New with defects   138
## 3 New without box   1803
## 4 Pre-owned         8412
## 5 <NA>               636
```

Top brands (of almost 3K...):

```
shoes_train %>% count(brand, sort = TRUE)
```

```
## # A tibble: 2,804 x 2
##    brand                n
##    <chr>            <int>
##  1 <NA>              2083
##  2 Dansko             647
##  3 Clarks             504
##  4 SAS                294
##  5 Chanel             265
##  6 Unbranded          261
##  7 Børn               223
##  8 Chloe              198
```

```
##  9 Sperry Top-Sider   139
## 10 Skechers           136
## # ... with 2,794 more rows
```

The most expensive shoes:

```
shoes_train %>% arrange(-price) %>% slice(1) %>% select(title, price)
```

```
## # A tibble: 1 x 2
##   title                                                    price
##   <chr>                                                    <dbl>
## 1 CHANEL Oxford, Lace up Shoes, NEW, Black, Leather, Size 38 1016.
```

The least expensive shoes:

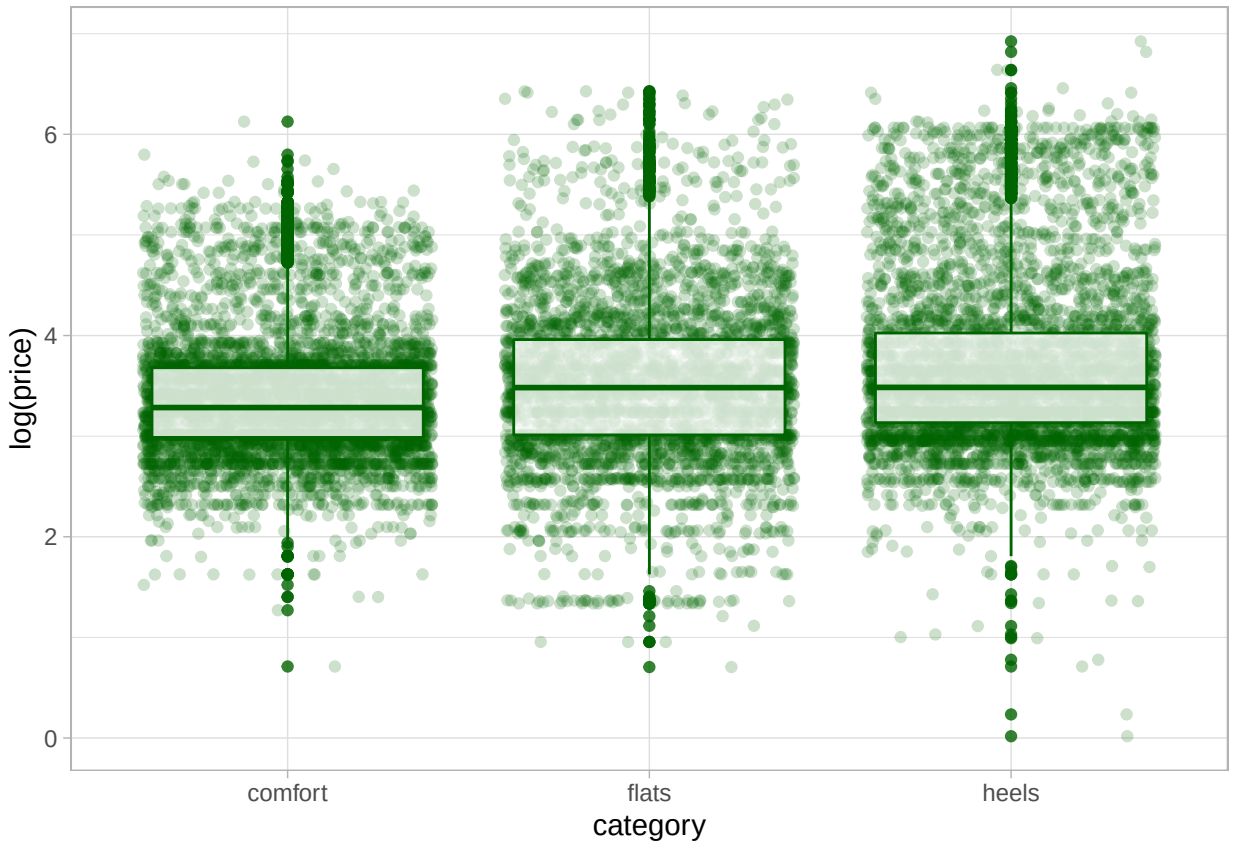
```
shoes_train %>% arrange(price) %>% slice(1) %>% select(title, price)
```

```
## # A tibble: 1 x 2
##   title                                                          price
##   <chr>                                                          <dbl>
## 1 Unisa Silver Glitter Ankle Strap Wedge Sandals Brown Cork Heel Womens 8~  1.02
```

This doesn't look like shoes, but that's ebay for you.

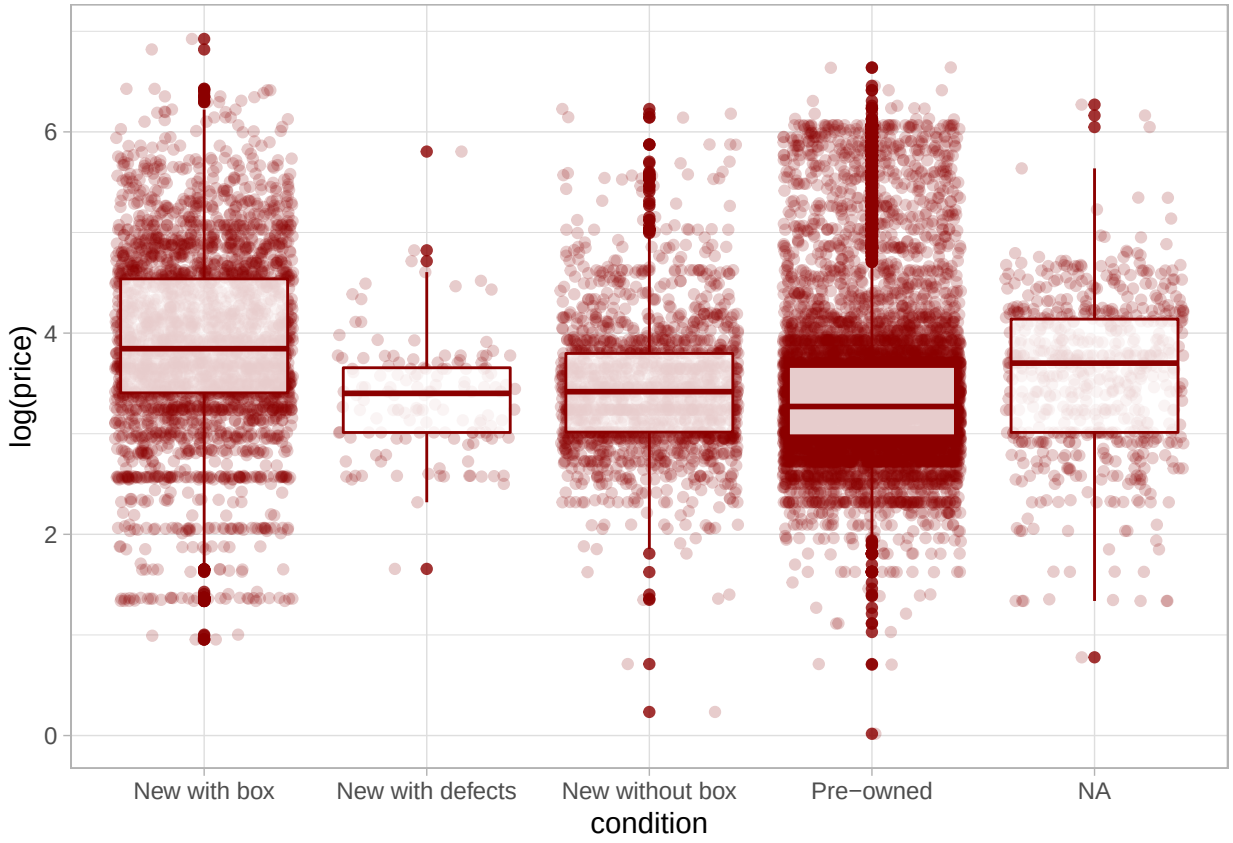
Let's see price by category. By the range of ~1,000 dollars, we'll need a log transformation:

```
shoes_train %>%
  ggplot(aes(category, log(price))) +
  geom_jitter(color = "darkgreen", alpha = 0.2) +
  geom_boxplot(color = "darkgreen", alpha = 0.8) +
  theme_light()
```



Let's see this by condition:

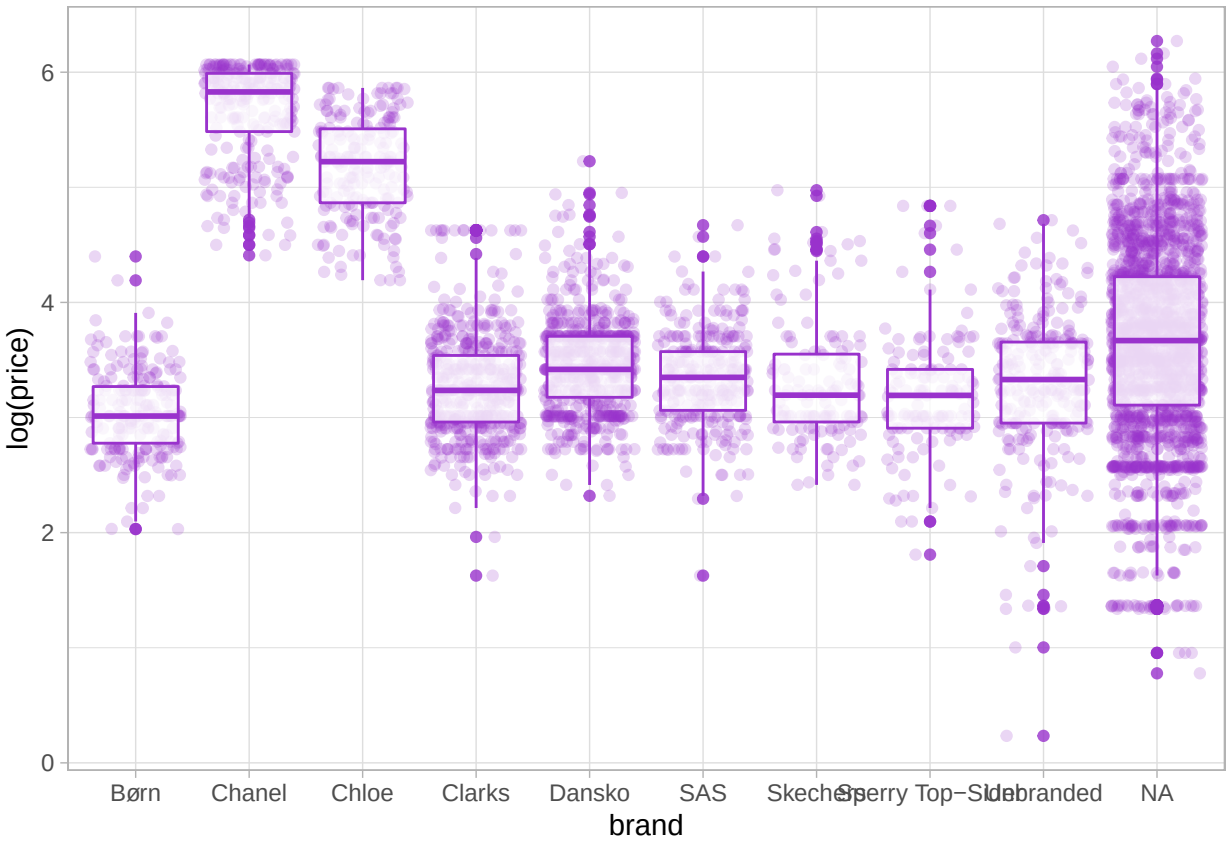
```
shoes_train %>%
  ggplot(aes(condition, log(price))) +
  geom_jitter(color = "darkred", alpha = 0.2) +
  geom_boxplot(color = "darkred", alpha = 0.8) +
  theme_light()
```



Finally see this by top brands:

```
top_brands <- shoes_train %>%
  count(brand, sort = TRUE) %>%
  slice(1:10) %>% pull(brand)

shoes_train %>%
  filter(brand %in% top_brands) %>%
  ggplot(aes(brand, log(price))) +
  geom_jitter(color = "darkorchid", alpha = 0.2) +
  geom_boxplot(color = "darkorchid", alpha = 0.8) +
  theme_light()
```



RMSE Baseline

Let's do a basic split (you can later re-split the data as you like):

```
library(tidymodels)

set.seed(42)
shoes_split <- initial_split(shoes_train, prop = 0.8)
shoes_train_tr <- training(shoes_split)
shoes_train_te <- testing(shoes_split)
```

If we simply predict the training set mean...

```
tr_price_mean <- mean(log(shoes_train_tr$price))

rmse_vec(log(shoes_train_te$price), rep(tr_price_mean, nrow(shoes_train_te)))
```

```
## [1] 0.79382
```

If we simply predict the mean of each category...

```
tr_price_mean_cat <- shoes_train_tr %>%
  group_by(category) %>%
  summarise(price_mean = mean(log(price)))

pred_price_cat <- shoes_train_te %>%
  inner_join(tr_price_mean_cat, by = "category") %>%
  pull(price_mean)
```

```
rmse_vec(log(shoes_train_te$price), pred_price_cat)
```

```
## [1] 0.7799116
```

If we add in condition, where we treat `NA` as another category...

```
shoes_train_tr <- shoes_train_tr %>%
  mutate(category = ifelse(is.na(category), "NA", category))

shoes_train_te <- shoes_train_te %>%
  mutate(category = ifelse(is.na(category), "NA", category))

tr_price_mean_cat_cond <- shoes_train_tr %>%
  group_by(category, condition) %>%
  summarise(price_mean = mean(log(price)))
```

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

```
pred_price_cat_cond <- shoes_train_te %>%
  inner_join(tr_price_mean_cat_cond, by = c("category", "condition")) %>%
  pull(price_mean)

rmse_vec(log(shoes_train_te$price), pred_price_cat_cond)
```

```
## [1] 0.7330369
```

Finally if we add the top brands, where `NA` is a brand and all other brands are "other"...

```
shoes_train_tr <- shoes_train_tr %>%
  mutate(brand = ifelse(is.na(brand), "NA",
                        ifelse(brand %in% top_brands, brand, "other")))

shoes_train_te <- shoes_train_te %>%
  mutate(brand = ifelse(is.na(brand), "NA",
                        ifelse(brand %in% top_brands, brand, "other")))

tr_price_mean_cat_cond_brand <- shoes_train_tr %>%
  group_by(category, condition, brand) %>%
  summarise(price_mean = mean(log(price)))
```

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

```
pred_price_cat_cond_brand <- shoes_train_te %>%
  left_join(tr_price_mean_cat_cond_brand, by = c("category", "condition", "brand")) %>%
  pull(price_mean)

rmse_vec(log(shoes_train_te$price), pred_price_cat_cond_brand)
```

```
## [1] 0.6287638
```

Throwing in interaction between category and condition and is the product sent with free shipping, though almost all coefficients are "significant", doesn't really help RMSE:

```
mod <- lm(log(price) ~ category*condition + brand + free_shipping, data = shoes_train_tr)

pred_lm <- predict(mod, shoes_train_te)
```

```
rmse_vec(log(shoes_train_te$price), pred_lm)
```

```
## [1] 0.6271918
```

What you need to do

(90 points) Build a sensible model, with your ML method of choice, to predict the `log(price)` of the 1,044 shoes in `shoes_test`.

```
shoes_test_processed <- shoes_test %>%
  mutate(brand = ifelse(is.na(brand), "NA",
                        ifelse(brand %in% top_brands, brand, "other")))

pred_model01 <- predict(mod, shoes_test_processed)
```

Once you do that, sink a CSV of your predictions titled e.g. `model01.csv`:

```
shoes_test$price_pred <- pred_model01

shoes_test %>%
  select(id, price_pred) %>%
  head()
```

```
## # A tibble: 6 x 2
##      id price_pred
##   <int>      <dbl>
## 1  3728       3.81
## 2 11185       3.44
## 3  4719       3.51
## 4  9326       3.32
## 5 10154       3.44
## 6  7230       3.32
```

```
shoes_test %>%
  select(id, price_pred) %>%
  write_csv("model01.csv")
```

Drop me a mail either by actually sending me a mail or opening an issue in your repo and assigning it to me, and wait to see your result on the Leaderboard!

WARNING: Be sure to name your models differently, otherwise your last result might run over your previous result, and you won't know which is which!

At the end of the period you should have a single pdf page with a short bulleted summary of what you did.

Further Dgeshim

- ALL MUST REPRODUCE (R or Python - you should have a notebook I can run)
- You may not under any circumstances overfit to the testing set (use your creativity for building a better model!)
- You may not search the price of the shoes in the testing set

Paper questions

(10 points) Read Sections 1-3 from Tree in Tree, an adorable paper from NeurIPS 2021 by Bingzhao Zhu, Mahsa Shoaran (of course, you're welcome to read the whole thing!).

Explain in your own up to 100 words what is "Tree in Tree" and how it improves on CART:

Our Answer: The proposed "Tree in Tree" algorithm is a new suggested way to construct large decision trees (DT). Unlike conventional DTs - it's done in a non-greedy way, with the main difference being that while DTs recursively split the leaf nodes in a greedy way, TnT fits new decision trees *in place* of the internal/leaf nodes, to optimize of the internal nodes, and basically replaces internal nodes with "micro" DTs. TnT is more efficient than CART, by sharing nodes between multiple paths (each node can have more than 1 parent), and it really helps to reduce the model's complexity. TnT also has a smaller model size and in general achieves better performance than CART models.

Look at Figure 2 example. The authors demonstrate very clearly how TnT can reach a more simple model than CART on these data. Please describe what would MARS do! You can either:

- hypothesize what it would do and explain shortly (but you better explain it well so I'm sure you got it)
- actually run `earth::earth()` on these data (I created it for you below in `df`) and show me the model (but you better use the right params for `earth()` otherwise you'd be left with a stump :()

```
df <- expand.grid(x1 = 1:3, x2 = 1:3)
df$y <- c(1, 0, 1, 0, 0, 0, 1, 0, 1) #1: triangle, 0: circle

library(earth)
```

```
## Loading required package: Formula
```

```
## Loading required package: plotmo
```

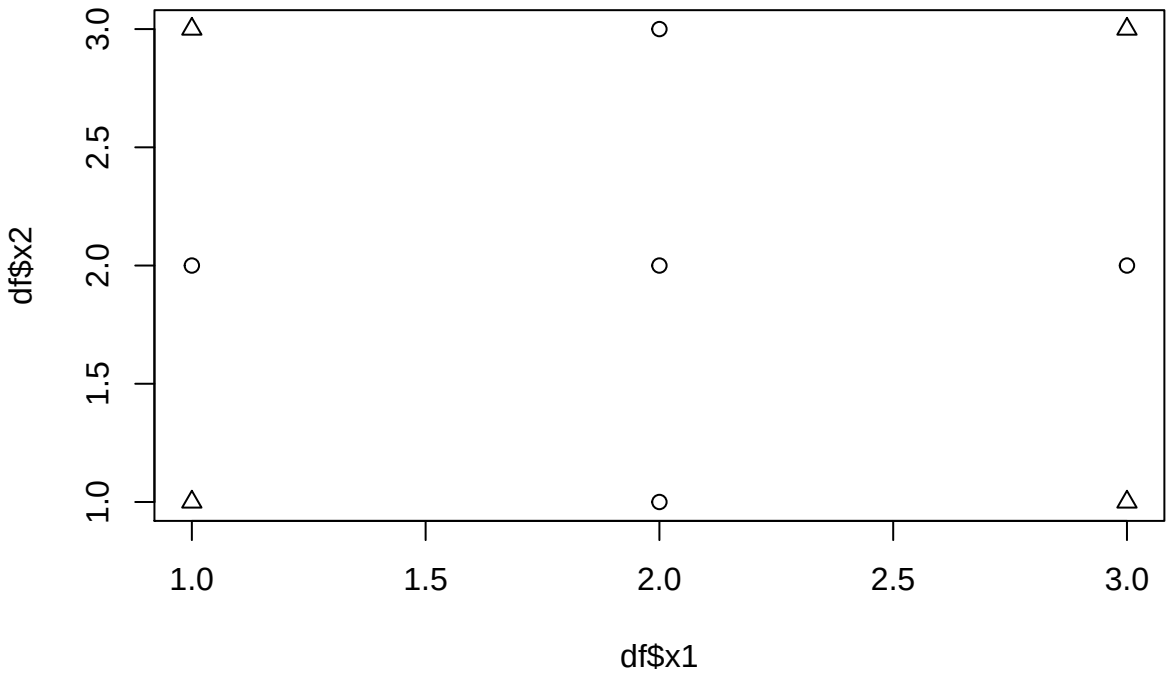
```
## Loading required package: plotrix
```

```
##
## Attaching package: 'plotrix'
```

```
## The following object is masked from 'package:scales':
##
##     rescale
```

```
## Loading required package: TeachingDemos
```

```
plot(df$x1, df$x2, pch = df$y + 1)
```



```
# We're not really sure if these are the right params, as in the example we've seen & in Google, this e
mod_mars <- earth(y ~ x1 + x2, data=df, degree= 1)
summary(mod_mars)
```

```
## Call: earth(formula=y~x1+x2, data=df, degree=1)
##
##             coefficients
## (Intercept)    0.4444444
##
## Selected 1 of 1 terms, and 0 of 2 predictors
## Termination condition: RSq changed by less than 0.001 at 1 term
## Importance: x1-unused, x2-unused
## Number of terms at each degree of interaction: 1 (intercept only model)
## GCV 0.3125    RSS 2.222222    GRSq 0    RSq 0
```

```
# Maybe like this, for binary classification?
mod_mars2 <- earth(y ~ x1 + x2, data=df, degree= 1, glm=list(family=binomial))

# Yep, we're confused. :(
```

Isn't MARS amazing?

Bonus 2 points: fit `rpart` on these data and show me that tree from Figure 2(b)!

Wrap up

And that's it, you have shown you can build a sensible, reproducible model, on big not-so-trivial data, to predict the price of women's shoes. Good luck with the rest of the course!