WRANGLING CATEGORICAL DATA

AMAZON EMPLOYEE ACCESS CHALLENGE

From Kaggle: “When an employee at any company starts work, they first need to obtain the computer access necessary to fulfill their role. This access may allow an employee to read/manipulate resources through various applications or web portals. It is assumed that employees fulfilling the functions of a given role will access the same or similar resources.

The objective of this competition is to build a model, learned using historical data, that will determine an employee’s access needs, such that manual access transactions (grants and revokes) are minimized as the employee’s attributes change over time. The model will take an employee’s role information and a resource code and will return whether or not access should be granted.”

AMAZON EMPLOYEE ACCESS CHALLENGE

Submission: For every line in the test set, submission files should contain two columns: id and ACTION. In the ground truth, ACTION is 1 if the resource should be allowed, 0 if the resource should not. Your predictions do not need to be binary. You may submit probabilities/predictions having any real value. The submission file should have a header.

Criteria: Submissions are judged on area under the ROC curve (so its probably best to submit predictions as probabilities).

AMAZON DATA

```
# A tibble: 6 × 10
  ACTION RESOURCE MGR_ID ROLE_ROLLUP_1 ROLE_ROLLUP_2 ROLE_DEPTNAME ROLE_TITLE
   <dbl>    <dbl>  <dbl>         <dbl>         <dbl>         <dbl>      <dbl>
1      1    39353  85475        117961        118300        123472     117905
2      1    17183   1540        117961        118343        123125     118536
3      1    36724  14457        118219        118220        117884     117879
4      1    36135   5396        117961        118343        119993     118321
5      1    42680   5905        117929        117930        119569     119323
6      0    45333  14561        117951        117952        118008     118568
# ℹ 3 more variables: ROLE_FAMILY_DESC <dbl>, ROLE_FAMILY <dbl>,
#   ROLE_CODE <dbl>
```

Notes:

1. Data: train.csv, test.csv, sampleSubmission.csv

AMAZON EMPLOYEE ACCESS CHALLENGE

What is different from the bike share data?

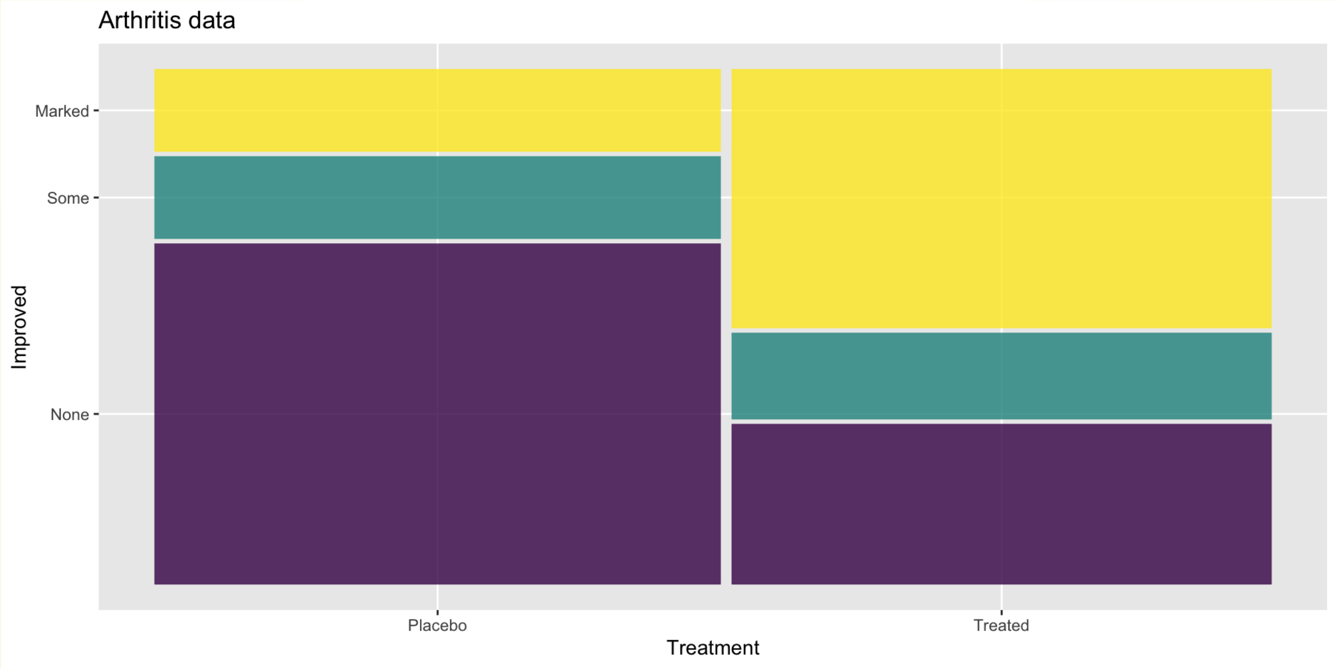
1. The response is categorical (nominal)
2. The explanatory variables are categorical even though they are read in as numeric.

VISUALIZING CATEGORICAL DATA

If we use scatterplots (primarily) when the response is quantitative, what do we use when the response to nominal?

1. Side-by-side boxplots `... + geom_box(aes(x=, y=Target))`
2. Mosaic Plots (a visual of a table)

```
library(ggmosaic)
ggplot(data=) + geom_mosaic(aes(x=product(CatVar), fill=Target))
```



POTENTIAL ISSUES WITH CATEGORICAL DATA

1. Categories with very few observations
    - Solution: If an explanatory variable, create an “other” category. If a target variable, balance it out (see later this unit)
2. Some ML algorithms require numeric features rather than categories
    - Solution: Encoding (the process of converting categories to numeric values)

ONE-HOT ENCODING

Each level of a categorical feature is mapped to a vector containing 0s and 1s

CompanyName	Categoricalvalue	Price
VW	1	20
Acura	2	10011
Honda	3	50000
Honda	3	10000

One hot encoding

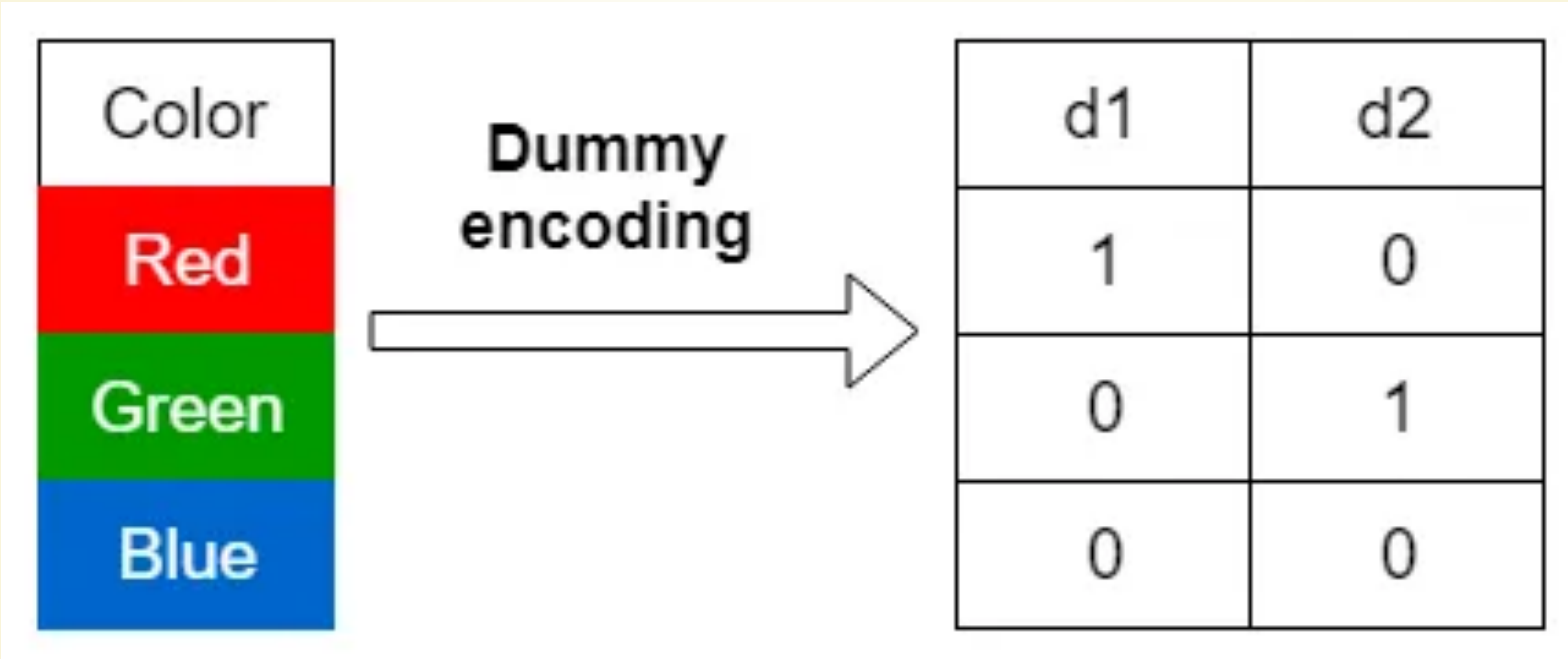
VW	Acura	Honda	Price
1	0	0	20000
0	1	0	10011
0	0	1	50000
0	0	1	10000

Advantage: categorical variables can be put it a numeric format

Disadvantage: large increase in dimensionality if there are a lot of levels in a categorical variable

DUMMY VARIABLE ENCODING

Drop one of the labels



Advantage: avoids the dummy variable trap (perfect collinearity)

Disadvantage: large increase in dimensionality

TARGET ENCODING

Replace a categorical value with the mean of the target for those observations.

(This is called target encoding because you use values of the *target* variable to encode the feature variables.)

workclass	target
State-gov	0
Self-emp-not-inc	1
Private	0
Private	0
Private	1

workclass	target mean
State-gov	0
Self-emp-not-inc	1
Private	1/3

workclass	target
0	0
1	1
1/3	0
1/3	0
1/3	1

Advantage: no change in dimensionality

Disadvantage: could result in data leakage or overfitting

TARGET ENCODING

There are more complicated versions of target encoding that involve weights, Bayesian methods, etc. Generally, for any type of target variable (response), we

1. Fit a model (usually a linear model) to predict y using just the categorical x_i
2. Replace x_i with $\hat{y}_i = \hat{f}(x_i)$.

HELPFUL STEP FUNCTIONS IN R

```r
library(tidymodels)
library(embed) # for target encoding

my_recipe <- recipe(rFormula, data=myDataset) %>%
  step_mutate_at(all_numeric_predictors(), fn = factor) %>% # turn all numeric features into factors
  step_other(var2, threshold = .05) %>% # combines categorical values that occur <5% into an "other" value
  step_dummy(all_nominal_predictors()) %>% # dummy variable encoding
  step_lencode_mixed(all_nominal_predictors(), outcome = vars(target_var)) #target encoding
  # also step_lencode_glm() and step_lencode_bayes()


# NOTE: some of these step functions are not appropriate to use together

# apply the recipe to your data
prep <- prep(my_recipe)
baked <- bake(prep, new_data = NULL)
```