Unit 08: Challenging (a.k.a. real) Data

Applied AI with R

Ferdinand Ferber and Wolfgang Trutschnig

Paris Lodron Universität Salzburg

6/1/24

Table of contents I

Section 1

Part I: Alastor Moody and the handling of missing data

Part I: Alastor Moody and the handling of missing data

AI generated image for the prompt “Alastor Moody with his magical eye inspecting a tablet in his Hogwards office.”

Setting

- In this unit we focus on missing data in the predictor variables, and not on missing values in the outcome variable (label).
- Missing values are ubiquitous in real-world datasets
  - Sensors sometimes just fail
  - Humans refuse to answer questions
  - Patients die before the end of the study
- On the other hand: Most ML algorithms can't digest `NA`s in the input
- How to handle this situation?

Setting

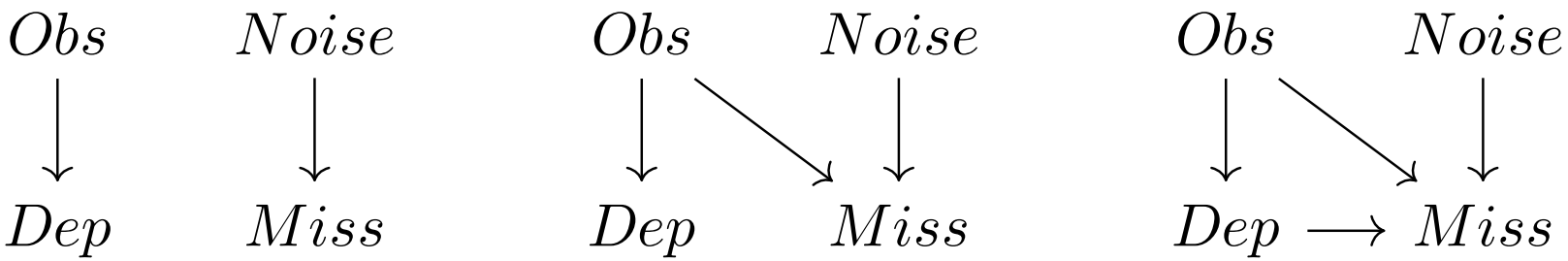
- In real-world datasets frequently predictor variables are interrelated.
- E.g. in a dataset on baseball players, there is a natural correlation between mass and height or between gender and mass, etc.
- Because of this, we might want to try (!) to restore (=impute) the missing values based on the non-missing values in the dataset.
- Nomenclature: In the following slides, the dependent variable is any predictor variable with missing values. The observed variables are all other predictor variables.

Types of missingness

- **Missing completely at random (MCAR):** If the events leading to the missingness are independent of both, the observed variables as well as the dependent variable.
- **Missing at random (MAR):** Missingness can be fully accounted for by observed variables, but is otherwise unrelated to the dependent variable.
- **Missing not at random (MNAR):** Not MCAR and MAR, i.e., the value of the variable that is missing is related to the reason it's missing.

Types of missingness

- `Obs` are some explanatory variables for the dependent variable `Dep`. Some values of `Dep` might be missing and the variable `Miss` indicates the missingness. The variable `Noise` is assumed to be completely unrelated to `X` and `Dep`.
- Arrows in the following diagrams denote statistical dependence (influence).



MCAR MAR MNAR

Types of missingness

Example: Variable `depression` (=dependent variable) and a variable `gender` (=explanatory variable). Some values of depression are missing in the dataset (but they still exist somewhere).

- **MCAR:** Any two individuals, regardless of their values of gender and depression, have the same probability of having a missing value for depression.
- **MAR:** Any two individuals with the same gender have the same chance of having a missing depression value (regardless of how large or small the depression value actually is)
- **MNAR:** Even among individuals of the same gender, the change of having a missing depression value depends (on the actual depression value). E.g. people with higher a depression score are more likely to not fill out the depression test.

Exercise

- Explain the difference between the response model and the imputation model.
- Consider measuring the mass of different objects with a scale. Come up with three different scenarios where missing data occurs, one for each of the types of missingness (MCAR, MAR, MNAR).

Determining and handling missingness

Type	Detection	Handling
MCAR	Little's test	Complete Analysis
MAR	Not possible	Model-based imputation
MNAR	Not possible	Challenging. Requires strong assumptions.

- Model-based imputation needs to assume MAR, because in that case a good model for $(\text{Dep}|\text{Miss} = 0, \text{Obs})$ will also be valid for $(\text{Dep}|\text{Miss} = 1, \text{Obs})$, because Dep and Miss are conditionally independent, given Obs.

Packages and data

- We'll work with the following packages

```
library(DescTools)
library(naniar)
```

- ...use the following dataset (with some more columns)

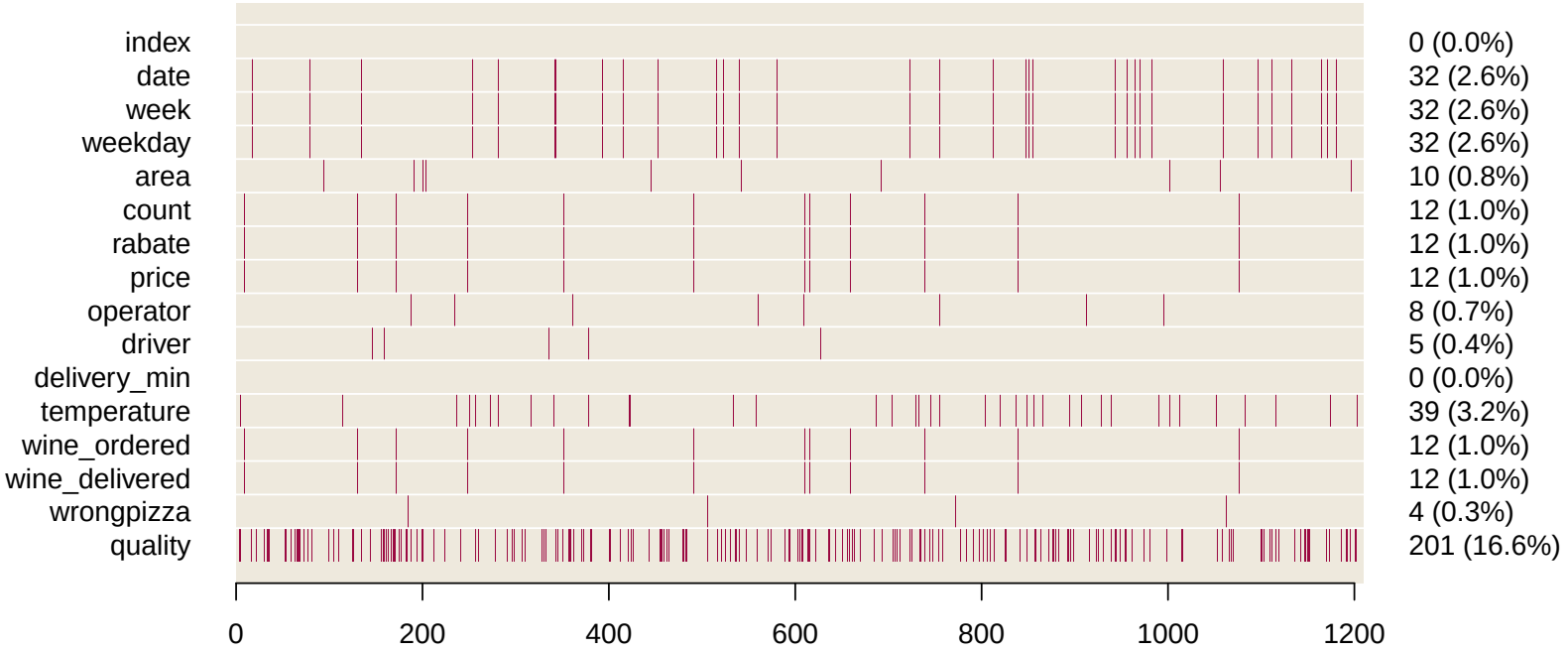
```
d.pizza
```

index	date	area	price	driver
1	2014-03-01	Camden	65.655	Taylor
2	2014-03-01	Westminster	26.980	Butcher
3	2014-03-01	Westminster	40.970	Butcher
4	2014-03-01	Brent	25.980	Taylor
5	2014-03-01	Brent	57.555	Carter

Overview plot

- In this plot the x axis is the row number and every red bar indicates an `NA` value

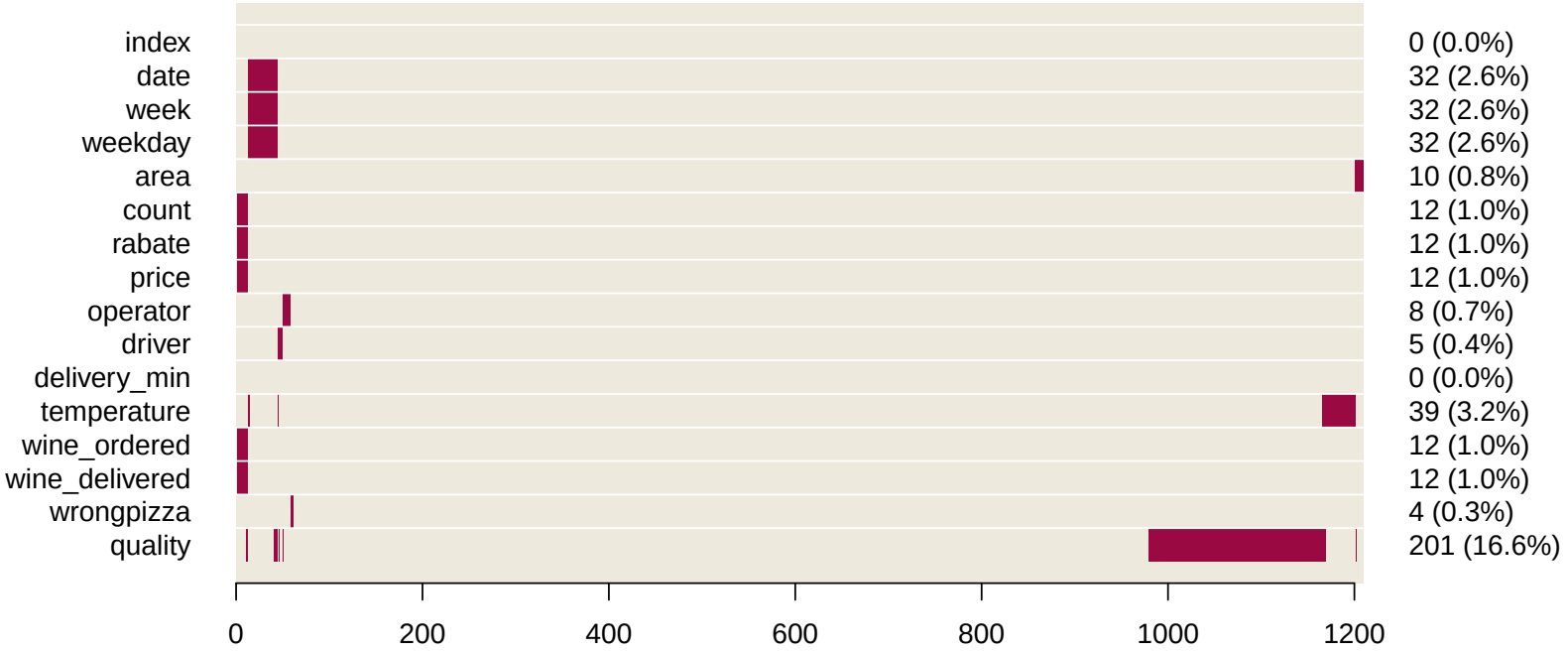
```
d.pizza |> DescTools::PlotMiss()
```



Overview plot

- One can use a clustering algorithm to cluster the missing values:

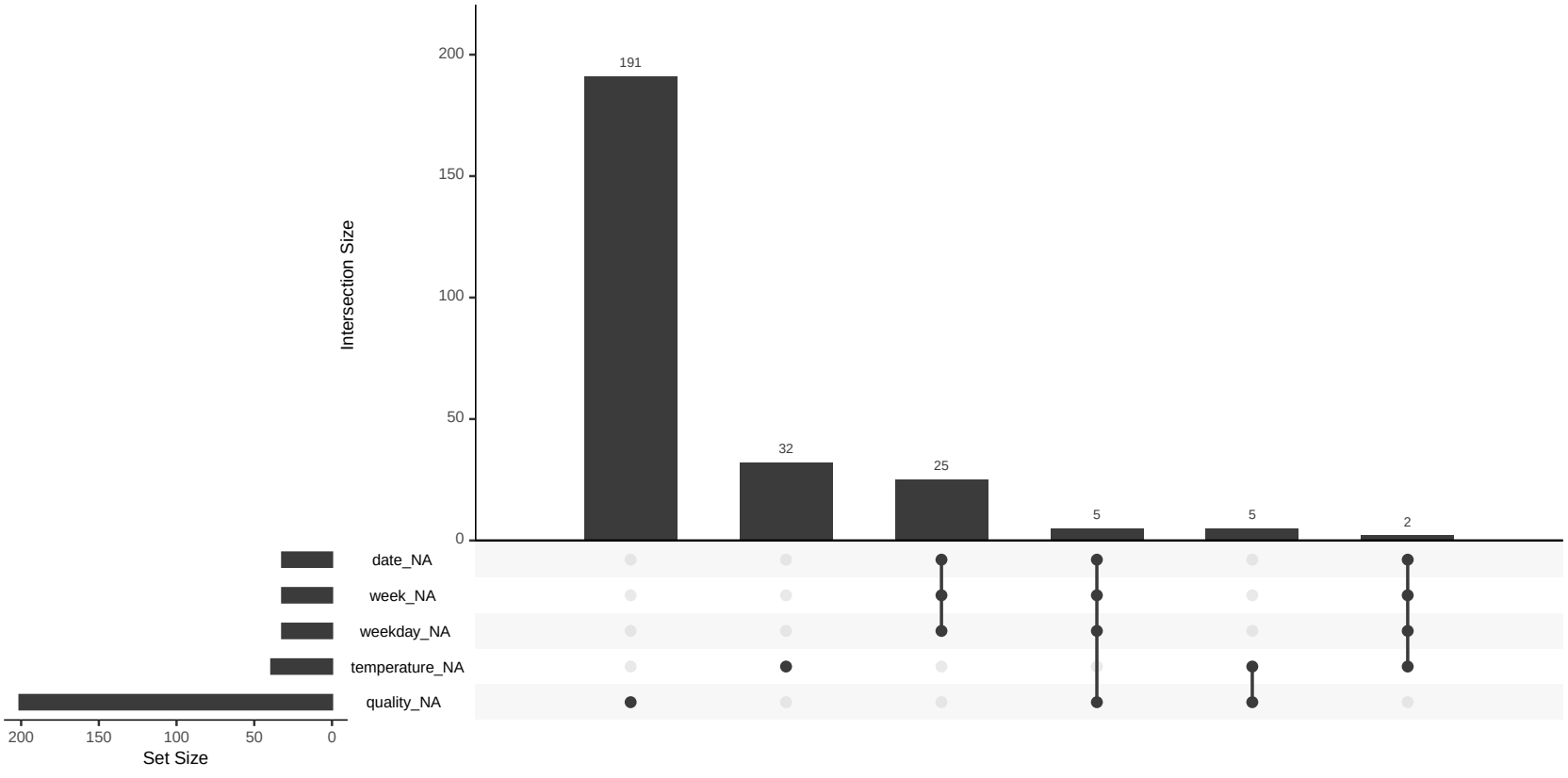
```
d.pizza |> DescTools::PlotMiss(clust = T)
```



Patterns

- We can inspect patterns of missingness:

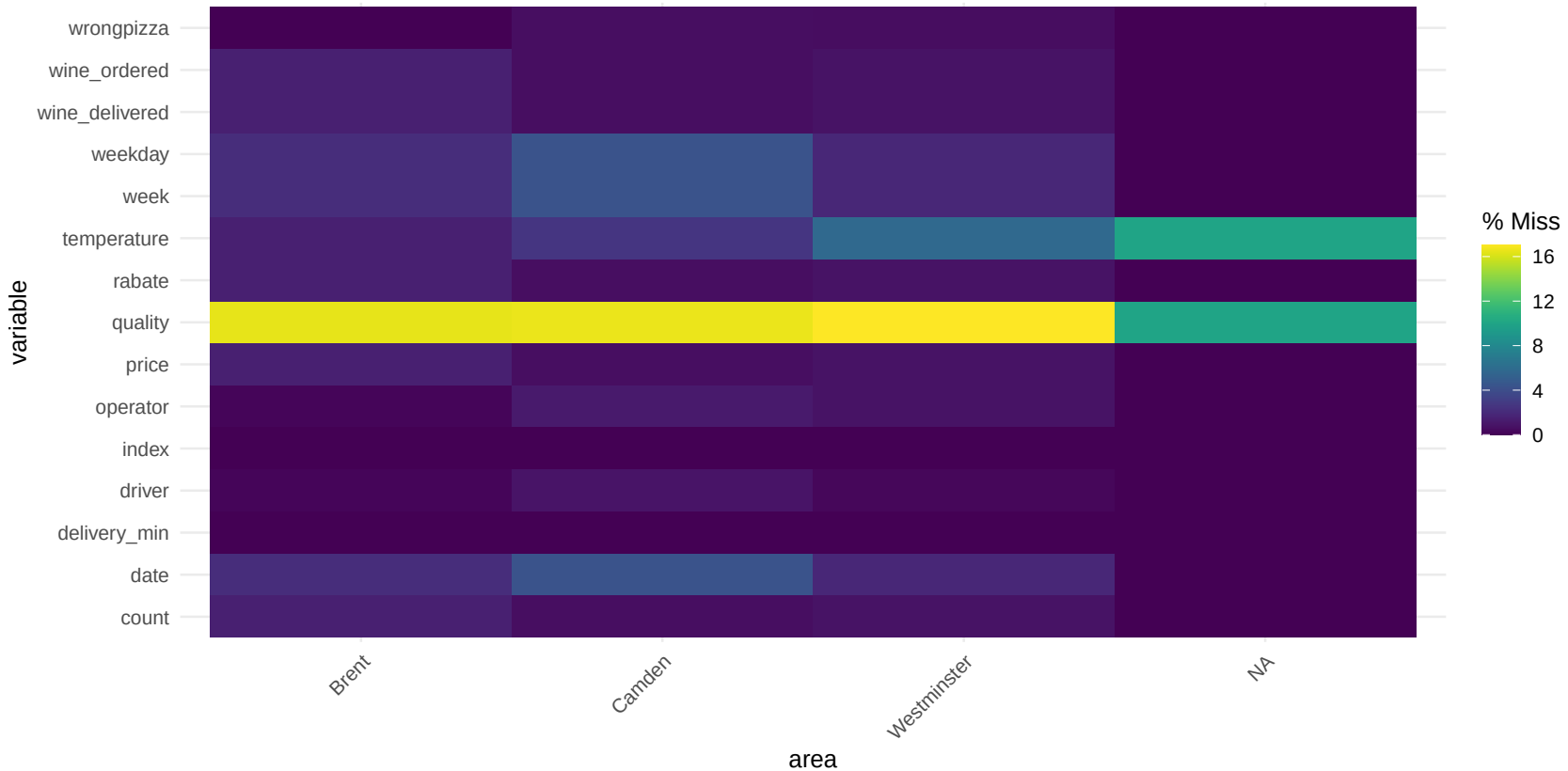
```
d.pizza |> naniar::gg_miss_upset()
```



Missingness across factors

- We can explore missingness across a factor

```
d.pizza |> naniar::gg_miss_fct(fct = area)
```



MCAR: Complete Case Analysis

- The *Complete Case Analysis* (CCA) works for datasets with MCAR, because the resulting dataset can be seen as a random subsample from the true distribution.
- In R this is done using

```
d.pizza |> drop_na()
```

MAR and MCAR: MICE

Multivariate imputation by chained equations (MICE) works as follows:

- For each column with missing data, specify and fit a model to predict the missing data from all other columns.
- Fix an ordering of the columns.
- In that order, apply the model to estimate the missing values ("placeholder values").
- Iterate, using placeholder values of last iteration for training the new models.
- **Notice that imputed values will influence the final model - all imputation errors will be part of the model.**

MICE in R

First, install the `mice` package. Then just call the `mice()` function. The `maxit` argument specifies how many iterations will be done. The `m` argument states how many rounds the whole algorithm will be repeated (results in multiple imputations).

```
mice_res <- d.pizza |>
  mice::mice(maxit = 10, m = 1, printFlag = F)
imputed_data <- mice_res |> pluck("data")
```

Mean/mode imputation

- In *mean/mode imputation* one replaces the missing value by the mean of the variable (in the continuous case) or by the mode of the variable (in the discrete case).
- If missingness is MCAR, this leads to unbiased point estimates, but artificially reduces the variance of the imputed variables.
- Implementation in `{tidymodels}`:

```
rec <- recipe(...) |>
  step_impute_mean(all_numeric_predictors()) |>
  step_impute_mode(all_factor_predictors())
```

Tree-based imputation

- A simpler version than MICE is imputation based on regression trees.
- For every variable with missing values a regression tree is trained and used for imputation, without multiple iterations.
- In the `{tidymodels}` framework this is called `step_impute_bag()`, because bagged trees (a sort of random forest) are used.

```
rec <- recipe(...) |>
  step_impute_bag()
```

Section 2

Part II: Severus Snape and the handling of outliers

Part II: Severus Snape and the handling of outliers

AI generated image for the prompt “Severus Snape looking angrily at a student in a hallway in Hogwards.”

Detecting outliers

Three main ideas on how to detect outliers:

- Reconstruction based:
  - Using unsupervised ML, one trains an encoder/decoder pair for the dataset
  - Points with high reconstruction error are declared as outliers
- Model based:
  - One trains one or several models on the data to predict one column given the others
  - Points with high residuals are declared as outliers
- Density based:
  - Points that lie in low-density regions of the input space are declared as outliers

The dataset

- We will again work with real data: reimbursements paid to deputies in the Brasilian Congress from 2009-2017.
- Here outliers are very interesting, because they might point towards instances of corruption.
- Load the `deputies.csv` file and parse it accordingly.
- deputies <- read_csv("http://trutschnig.net/deputies.csv")

Reconstruction-based outlier detection

- General idea: We train a model that learns to compress and decompress the dataset.
- The compressor should perform well on all "normal" points, but will have problems on outliers.
- By that logic, outliers can be identified as points with high reconstruction error.

Reconstruction-based outlier detection

- We will use the `dimRed` package that contains a lot of *dimensionality reductions*, i.e. compression models

```
library(dimRed)
```

- But the compression models only take numerical variables as input.
- Use `step_YeoJohnson()` and `step_range()` to normalize all numeric variables.
- Use `step_dummy()` to one-hot-encode all factorial variables.
- We omit the column `refund_date`, `deputy_name` and `company_name`.
- Use `prep()` and `bake(new_data = ...)` to apply the recipe to the data.

Reconstruction-based outlier detection

- Now we can encode/decode the dataset using one of the compressor models:

```
encoded <- dimRed::embed(
  deputies_preproc, "PCA", ndim = 3)
decoded <- encoded |>
  dimRed::inverse(dimRed::getDimRedData(encoded)) |>
  dimRed::getData()
```

Reconstruction-based outlier detection

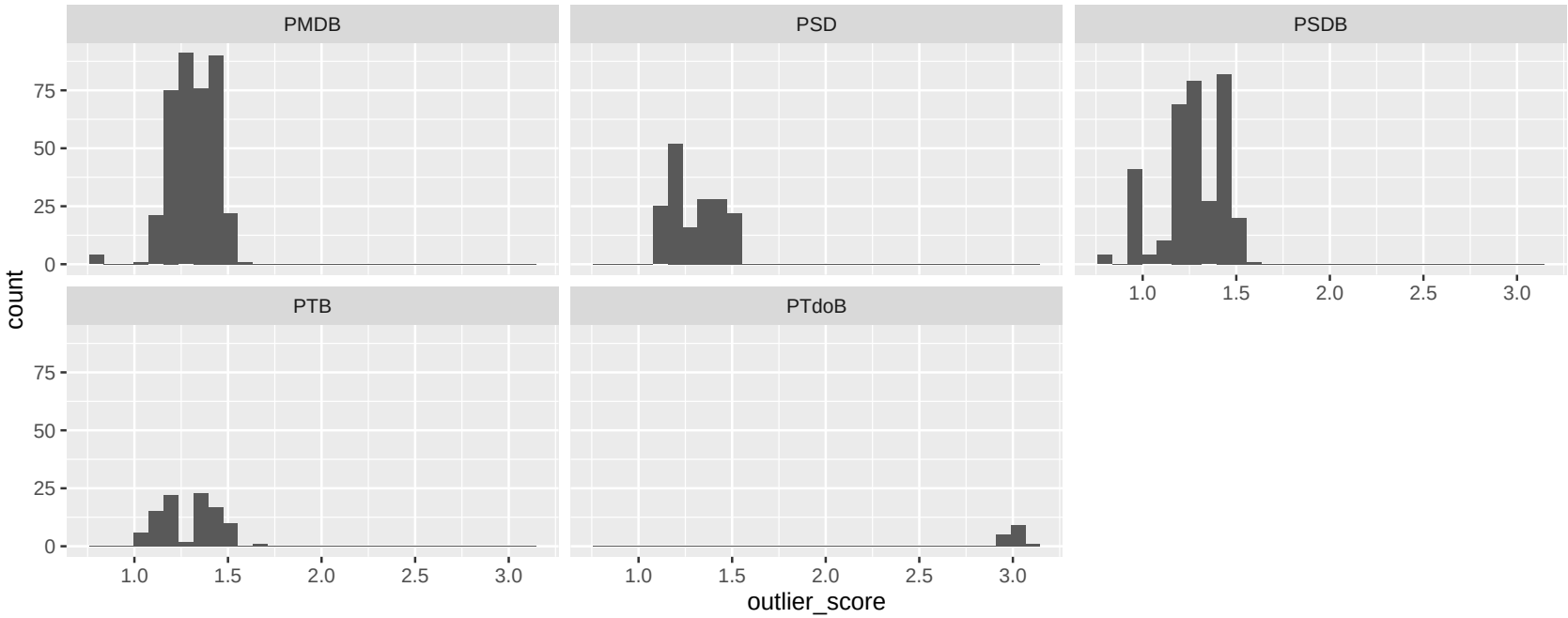
- Split the `deputies_preproc` dataframe into a list of vectors, each containing one observation/row. Do the same with the `decoded` matrix.
- Given two vectors `x` and `y` the statement `dist(rbind(x, y))` calculates the euclidean distance between `x` and `y`.
- Use one of the `purrr` verbs to calculate the distance between every row in `deputies_preproc` and `decoded`.
- Cast the resulting list of distances into a dataframe with one colum, named `outlier_score`.
- Plot a histogram of the outlier score.

Reconstruction-based outlier detection

- Use `bind_cols()` to add the outlier score to the original `deputies` dataframe.
- What have all outliers in common?
- Increase the `ndim = 3` argument in the `dimRed::embed()` function. What happens?

Reconstruction-based outlier detection

- Answer: All reimbursements for the smallest congress party (PTdoB) are flagged as outliers.



Model-based outlier detection

- The general idea is as follows:
- Partition the dataset into several *folds*.
- For each fold train a model to predict a chosen column from all other columns.
- For each data point, use all models to make a prediction.
- If the majority of the models is wrong (based on a given threshold), then the data point is declared an outlier.

Model-based outlier detection

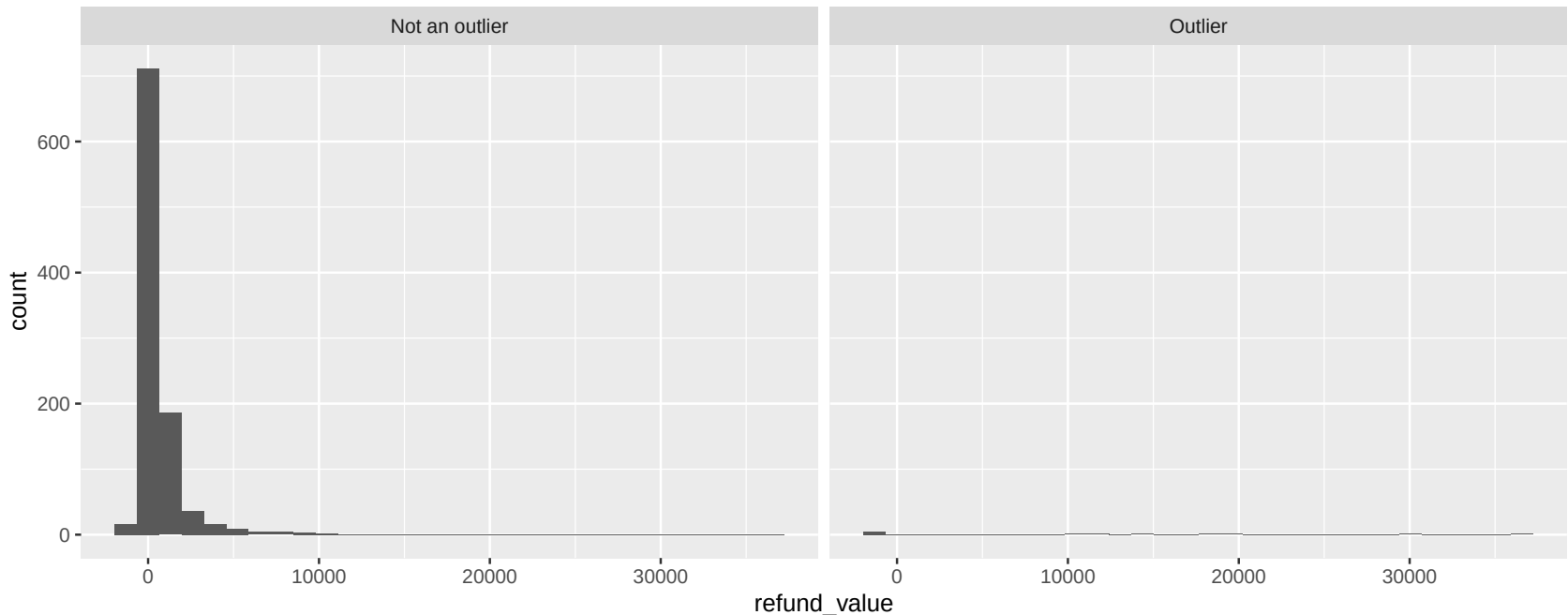
```
library(regfilter)

res <- regfilter::regIPF(refund_value ~ .,
                         data = deputies_preproc,
                         nfolds = 20,
                         t = 0.2)

deputies_with_outlier <- deputies |>
  mutate(outlier = row_number() %in% res$idnoise)
```

Model-based outlier detection

- Reproduce the outlier calculation.
- Plot `deputies_with_outlier`.
- What do all outliers have in common?

Model-based outlier detection

- Answer: Very small and very high refund values are flagged as outliers.



Density-based outlier detection

- Elementary idea: 'Normal' points lie in high-density regions and outliers lie in low-density regions.
- One common method is to flag all points as outliers that fall outside the $[Q_1 - 1.5 \cdot \text{IRQ}, Q_3 + 1.5 \cdot \text{IRQ}]$ interval for any column, where $\text{IRQ} := Q_3 - Q_1$ is the interquartile range.
- This corresponds to the default of boxplots in R (whisker length).
- A bit more nuanced: Local Outlier Factor (LOF).

Density-based outlier detection

- The *local outlier factor (LOF)* compares the local density of a point with the average local density of its k nearest neighbors.
- Here, the local density is based on the average *reachability distance* from its k nearest neighbors.
- The reachability distance between two points p and q is the normal euclidean distance, unless q is one of the k nearest neighbors of p. In that case it's set as the distance from p to its k-th nearest neighbor.
- If the local density of a point is low compared to its neighbors, it's declared as an outlier.

Density-based outlier detection

```
deputies_with_outlier <-
  DescTools::LOF(deputies_preproc, k = 10) |>
  data.frame(outlier_score = _) |>
  bind_cols(deputies)
```

Density-based outlier detection

- Reproduce the previous example.
- Can you spot any common patterns?

Handling outliers

- Keep outliers
- Remove outliers
  - Idea: Outliers "confuse" the model, removing them the model can better learn the actual structure of the data.
- Modify outliers
  - Correct outliers, e.g. in a survey, call the outlier-person again and ask if they understood the questions correctly.
  - Some researchers use *Winsorization*, clipping outliers to the 5th or 95th percentile of the data.
- Weight outliers based on outlier score (robust methods)
  - Some ML algorithms support *case weights*
  - Reducing weight of outliers (via their outlier score) implies a reduced effect on the loss.