THE UNIVERSITY OF TEXAS AT DALLAS

Feature & Target Engineering

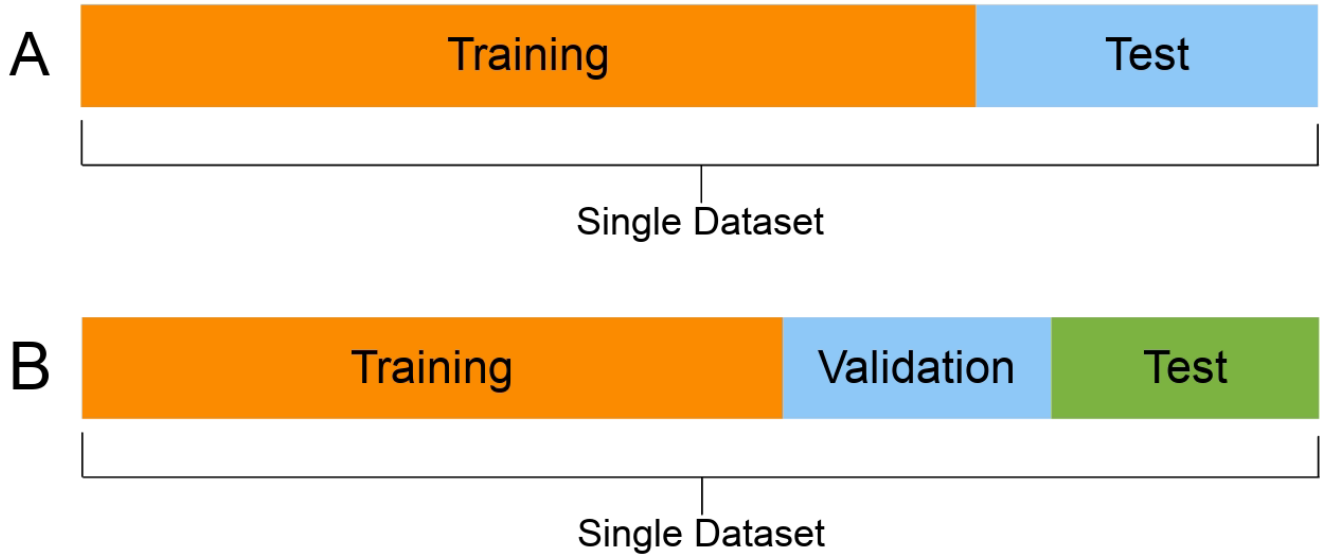
Yalin Yang
Doctoral student
Geographic information science

Population, Sampling distribution and unbiased estimator

- In **applied statistics** we are dealing with ***sampled data*** from the population and aim at estimating properties of the underlying population from which the random sample has been drawn

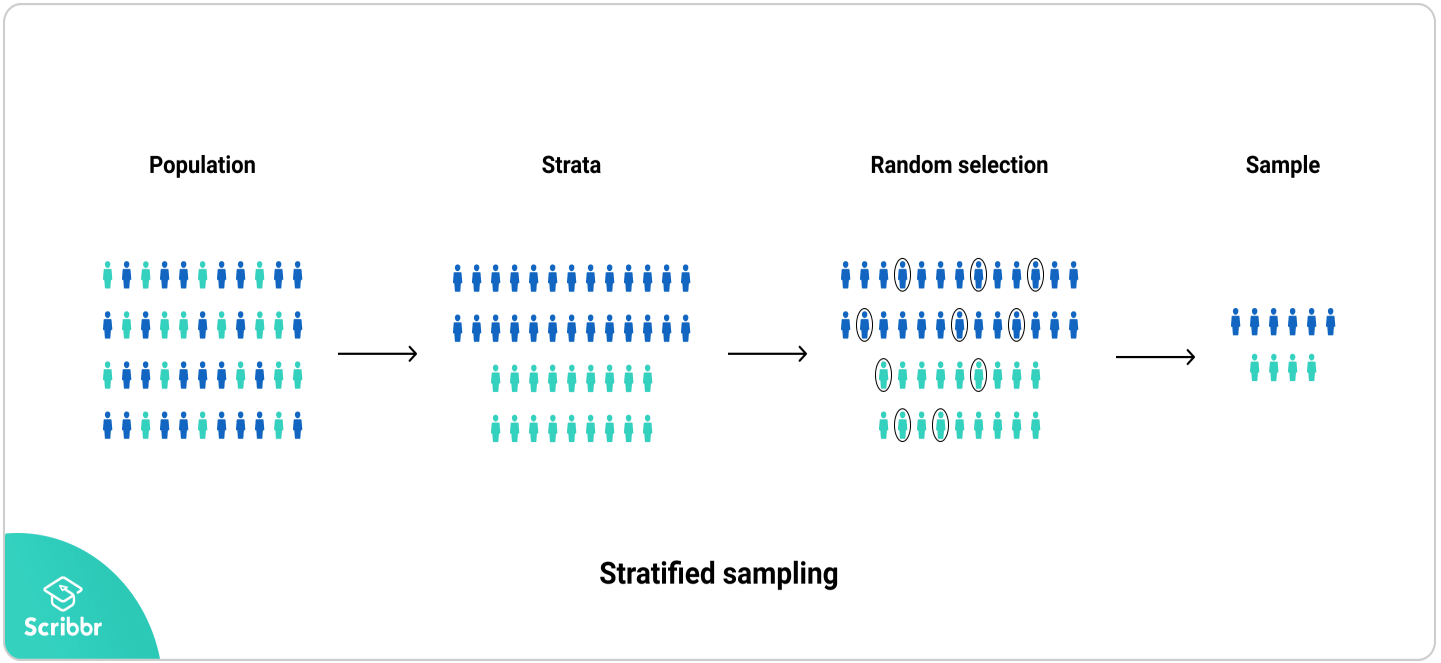
$$\mathrm{Bias}(\hat{\theta}, \theta) = \mathrm{Bias}_\theta[\,\hat{\theta}\,] = \mathrm{E}_{x|\theta}[\,\hat{\theta}\,] - \theta = \mathrm{E}_{x|\theta}[\,\hat{\theta} - \theta\,],$$

- In statistics, the **bias** of an estimator **is the difference between this estimator's expected value and the true value of the parameter** being estimated. An estimator or decision rule with zero bias is called unbiased.



Stratified Sampling

- Stratified sampling is a type of sampling method in which the total population is **divided into smaller groups** or strata to complete the sampling process.
- The strata is formed based on **some common characteristics** in the population data.
- After dividing the population into strata, the researcher randomly **selects the sample proportionally**.




```
# Stratified Random Sampling
amesSplit <- initial_split(ames, breaks=8, prop=0.7, strata="Sale_Price")
amesTrain <- training(amesSplit)
amesTest <- testing(amesSplit)
```

Box-Cox Transformation, log Transformation

Why we assume errors are normally distributed?

- The normal distribution often describes **the actual distribution of the random errors in real-world processes reasonably well**

- Due to the Central Limit Theorem, we may assume that there are lots of underlying facts affecting the process and the sum of these individual errors will tend to behave like in a **zero mean normal distribution**

Box-Cox Transformation, log Transformation

Box-Cox Transformation

- Causes for ***extreme observations***: [a] skewed distributions, [b] measurement or recoding errors, [c] extreme but feasible events (perhaps not belonging to the population under investigation).
- The Box-Cox transformation is a generalization of the power transformation: $Y = \frac{X^{\lambda}-1}{\lambda}$ and for $\lambda = 0$ we get $y = \ln(x)$.
- $\lambda > 1$ reduce negative skewness, whereas $\lambda < 1$ reduce positive skewness.
- For the interpreting purpose, of log transformation work well, we always take it as our first choice.

Missing data

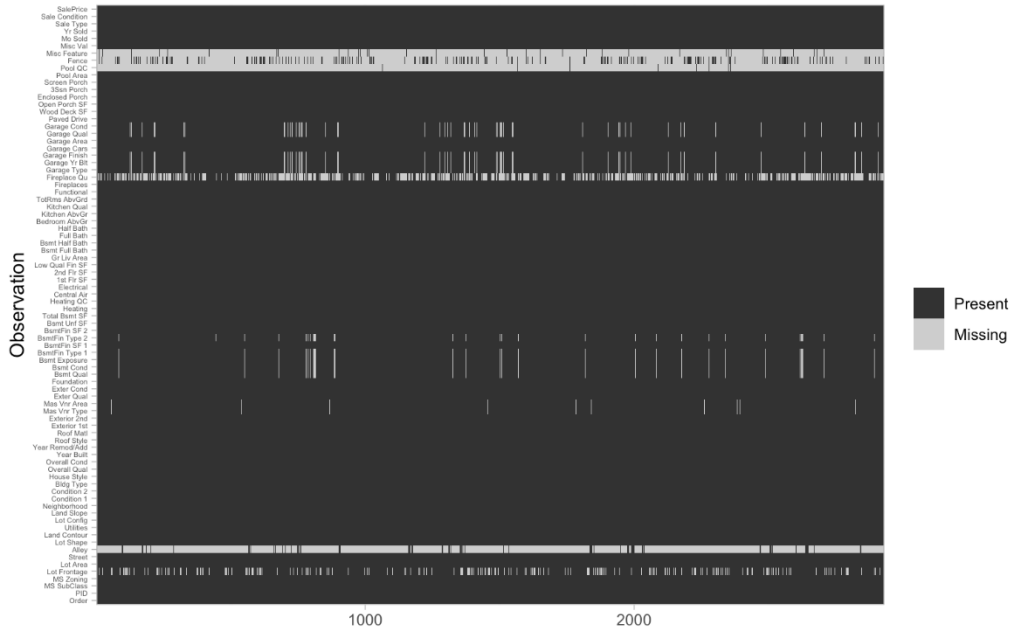
The reasons are usually lumped into two categories:

- **Informative missingness:** a structural cause for the missing value; whether this be deficiencies in how the data was collected or abnormalities in the observational environment
- **Missingness at random:** The missing values' occurrence is independent of the data collection process.

Visualizing missing values:

```
### Visualizing missing values
```{r fig.height=10, fig.width=15, message=FALSE, warning=FALSE}
library(ggplot2)
library(reshape2)

ames_raw %>% is.na() %>%
  melt() %>%
  ggplot(aes(x = Var2,y = Var1,fill=value)) +
  geom_raster() +
  coord_flip()+
  scale_y_continuous(NULL, expand = c(0,0)) +
  scale_fill_grey(name = '',labels = c("Present",'Missing'))+
  xlab('Observation') +
  theme(axis.text.y = element_text(size = 4))
```
```

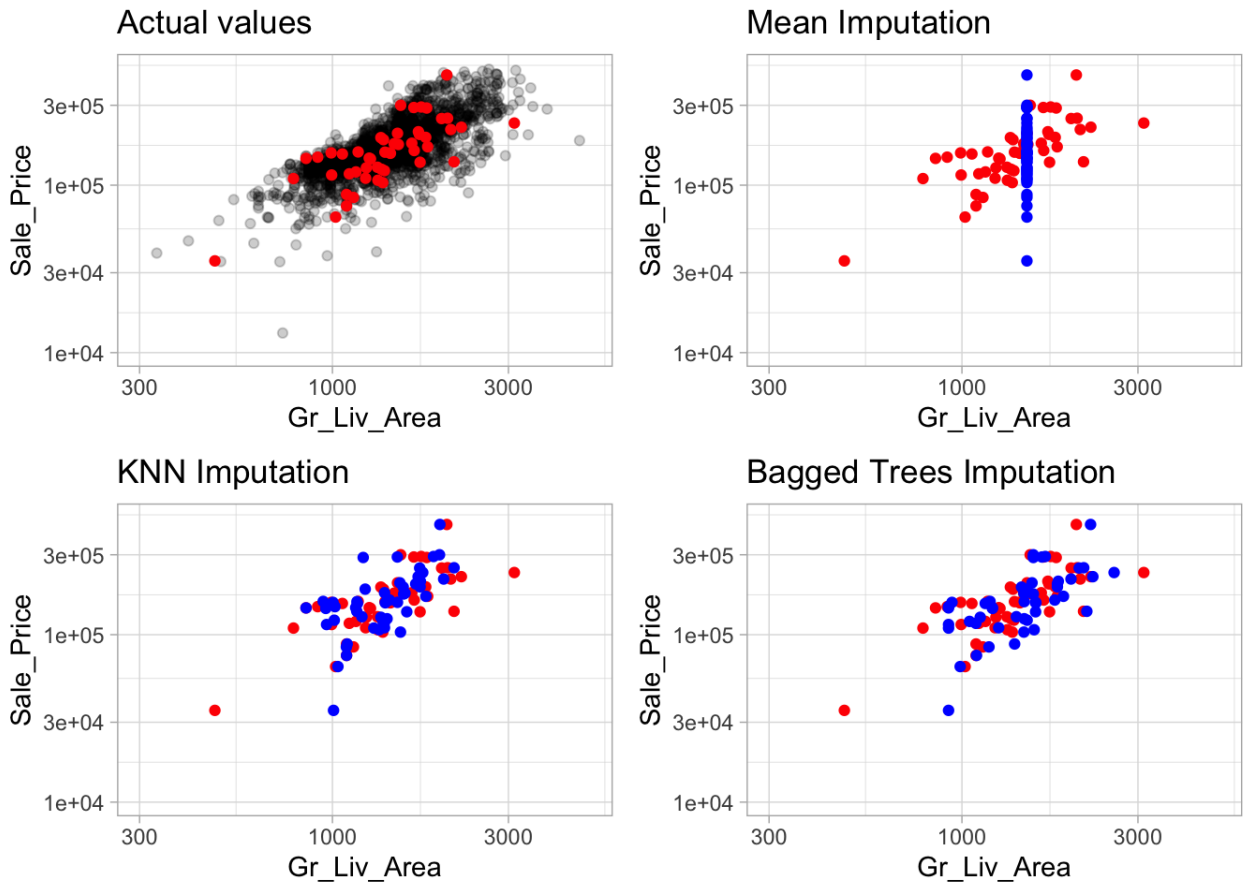


Imputation

imputation is the process replacing missing data with a **"best guess"** value.

Sometimes we **use mean, median, or other statistics to fill the missing data** , but this kind of process **do not consider any other attributes** for a given observation.

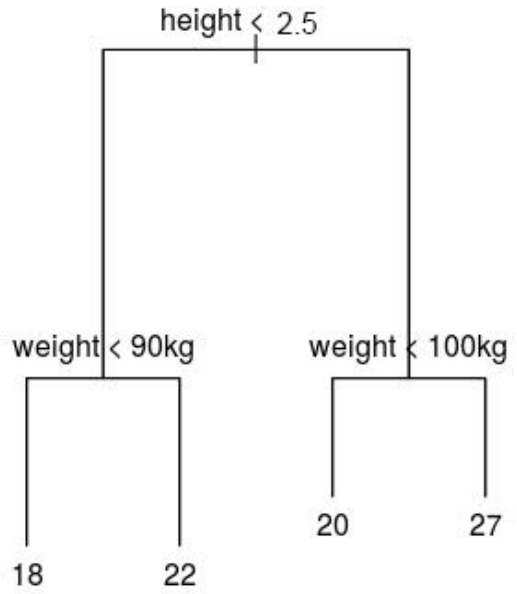
An alternative is to use grouped statistics to capture expected values for observations that **fall into similar groups**.(infeasible for larger datasets)

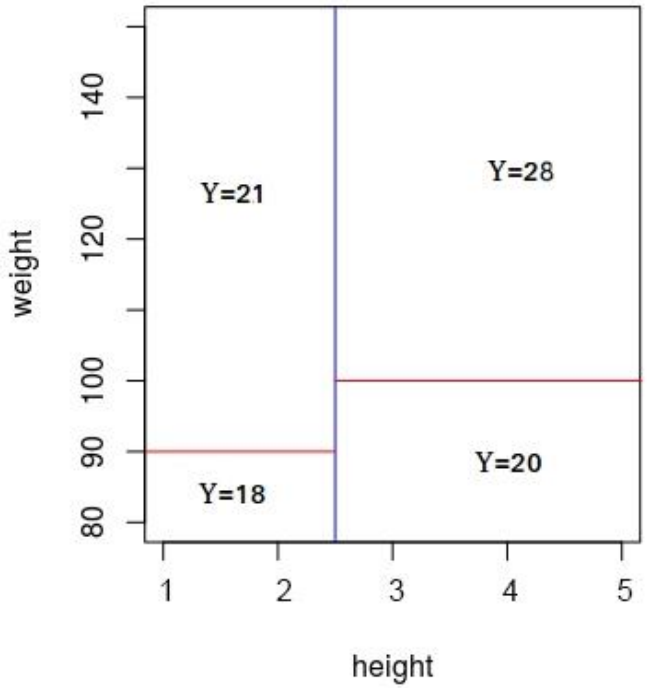


K-nearest neighbor and Tree-based method (Brief)

Tree based model

- Tree-based models use a series of **if-then rules** to generate predictions from one or more decision trees.
- Works for either regression or classification
  - **Classification tree** analysis is when the predicted outcome is the class (discrete) to which the data belongs.
  - **Regression tree** analysis is when the predicted outcome can be considered a real number (e.g. the price of a house, or a patient's length of stay in a hospital).





The objective is to find the regions $(R_1,R_2,...,R_J)$ for which the RSS is at its minimum:

$$RSS = \min_{R_1,R_2,...,R_J} \sum_{j=1}^{J} \sum_{i \in R_j} \left(y_i - \hat{y}_{R_j}\right)^2$$

Bagging Trees and Random Forest models (Brief)

Bagging Tree(Bootstrap aggregating)

- building many decision trees at a time by randomly sampling with replacement, or bootstrapping, from the original dataset.
- This **ensures variety** in the trees, which helps to **reduce the amount of overfitting**.

Random Forest

- Random forest models take this concept one step further. On top of **building many trees** from sampled datasets, each node is only allowed to split on a random selection of the model's features.

```
```{r}
ames_recipe %>%
  step_knnimpute(all_predictors(),neighbors = 6)
ames_recipe %>%
  step_bagimpute(all_predictors())
```
```

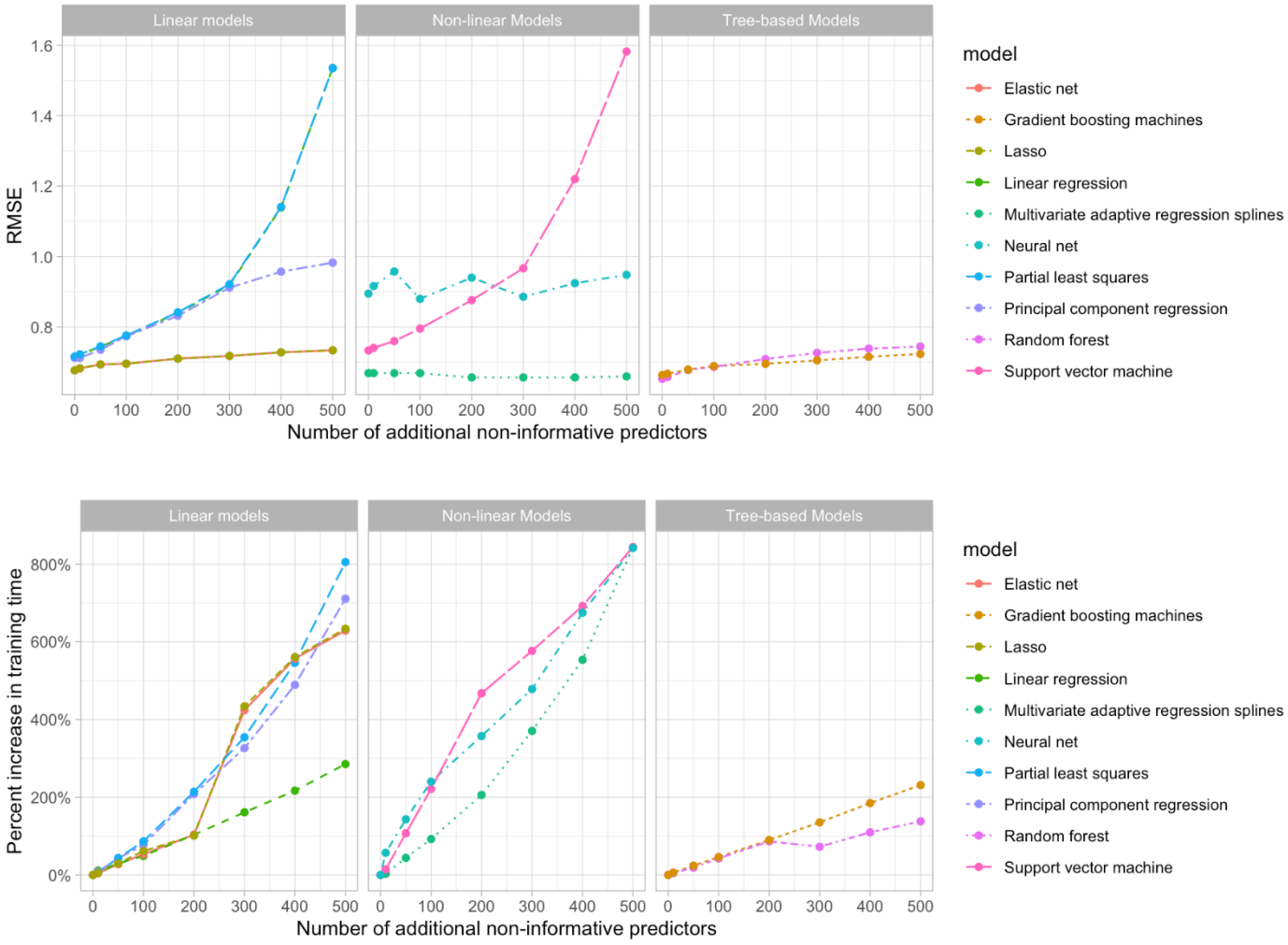
Feature Filtering

Reduce the number of features (input variables)

- Model becomes harder to interpret
- Costly to compute

What kind of features should be eliminated

- Zero and near-zero variance variables
  - The fraction of unique values over the sample size is low (say ≤10%).
  - The ratio of the frequency of the most prevalent value to the frequency of the second most prevalent value is large (say ≥20%).



Numeric feature engineering (Feature scaling)

Why we need Normalization and Standardization?

Distance-Based Algorithms

- Distance algorithms like KNN, K-means, and SVM are most affected by the range of features. This is because behind the scenes they are **using distances between data points to determine their similarity**.
- If all features have different scales, there is a chance that higher weightage is given to features with higher magnitude. This will impact the performance of the machine learning algorithm and obviously, **we do not want our algorithm to be biased towards one feature.**

Tree-Based Algorithms

Tree-based algorithms, on the other hand, are fairly insensitive to the scale of the features.

Numeric feature engineering (Feature scaling)

Normalization and Standardization

- Normalization is a scaling technique in which values are shifted and rescaled so that they end up **ranging between 0 and 1**. It is also known as Min-Max scaling.

$$X' = \frac{X - X_{min}}{X_{Max} - X_{min}}$$

- Standardization is another scaling technique where the values **are centered around the mean with a unit standard deviation.** This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.

$$X' = \frac{X - \mu}{\sigma}$$

Categorical feature engineering

Most models require that the predictors take numeric form. There are exceptions; for example, tree-based models naturally handle numeric or categorical features. However, even tree-based models can benefit from preprocessing categorical features.

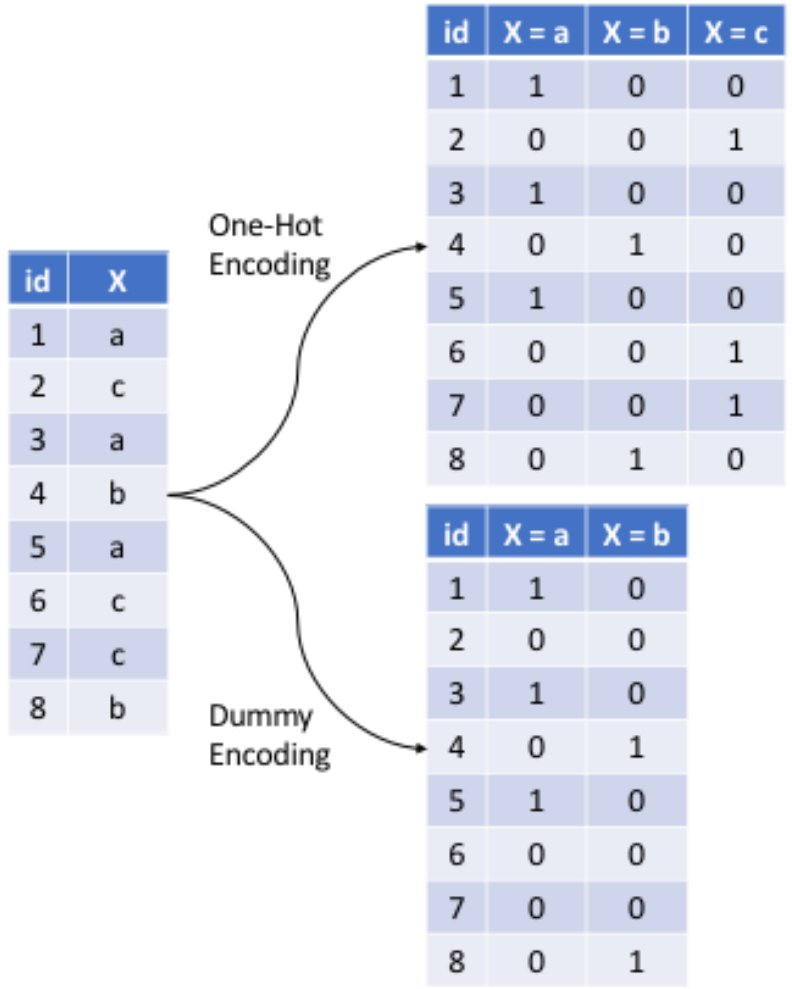
Lumping

- Similar to feature filtering, we want to find whether there is a dominant value (says >= 90%) in our variables.
- Sometimes we can benefit from collapsing, or “lumping” these into a lesser number of categories.Usually, we may want to collapse all levels that are observed in less than 10% of the training sample into another category. We can use `step_other()` to do so.
- However, lumping should be used sparingly as **there is often a loss** in model performance (Kuhn and Johnson 2013).

Categorical feature engineering

One-hot & dummy encoding

- Many models require that **all predictor variables be numeric**. Consequently, we need to intelligently **transform any categorical variables into numeric representations** so that these algorithms can compute.
  - The most common is referred to as **one-hot encoding**, where we transpose our categorical variables so that each level of the feature is represented as **a boolean value**.
  - Alternatively, we can create a full-rank encoding by **dropping one of the levels** (level $x = c$ has been dropped). This is referred to as dummy encoding.



Categorical feature engineering

Label encoding / Ordinal encoding

- Label encoding is a pure numeric conversion of the levels of a categorical variable.

- Notice that this result would be treated as ordered number, So usually we rank the feature before Label encoding

```
# Original categories
count(ames_train, Overall_Qual)
## # A tibble: 10 x 2
##    Overall_Qual       n
##    <fct>          <int>
##  1 Very_Poor          4
##  2 Poor               9
##  3 Fair              27
##  4 Below_Average    166
##  5 Average          565
##  6 Above_Average    513
##  7 Good             438
##  8 Very_Good        231
##  9 Excellent         77
## 10 Very_Excellent    23

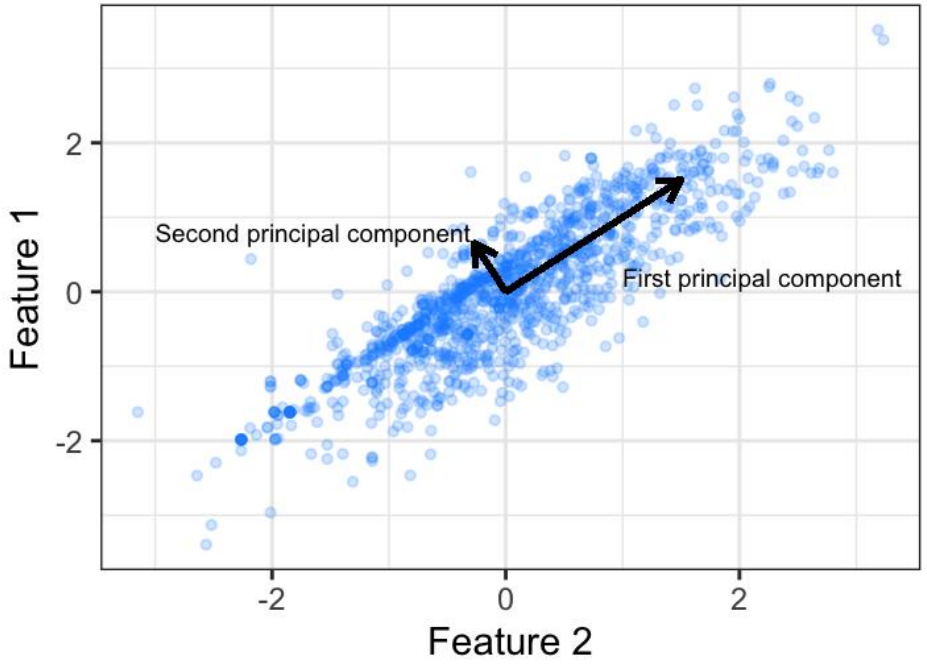
# Label encoded
recipe(Sale_Price ~ ., data = ames_train) %>%
  step_integer(Overall_Qual) %>%
  prep(ames_train) %>%
  bake(ames_train) %>%
  count(Overall_Qual)
## # A tibble: 10 x 2
##    Overall_Qual     n
##           <dbl> <int>
##  1            1     4
##  2            2     9
##  3            3    27
##  4            4   166
##  5            5   565
##  6            6   513
##  7            7   438
##  8            8   231
##  9            9    77
## 10           10    23
```

Dimension reduction

Dimension reduction is an alternative approach to **filter out non-informative features** without manually removing them. For example, we may wish to reduce the dimension of our features with principal components analysis.

Prerequisites

- Any missing values in the data must be removed or imputed;
- Typically, the data must all be numeric values
- Numeric data should be standardized



Data Preprocessing

1. Remove Near-zero variance features that are nominal
2. Ordinal encode our quality-based features
3. Center and scale all numeric features
4. Perform dimension reduction by using principal component analysis
5. one-hot encode remainning categorical features

Data leakage

Data leakage is when information from outside the training data set is used to create the model. Data leakage often occurs during the data preprocessing period. To minimize this, feature engineering should be done in isolation of each resampling iteration.