Applied Machine Learning

Feature Engineering: Categorical Encoding

Learning goals

- One-Hot Encoding
- Target/Impact Encoding

IMPORTANT TYPES OF FEATURE ENGINEERING

Feature engineering is on the intersection of **data cleaning**, **feature creation** and **feature selection**.

The goal is to solve common difficulties in data science projects, like

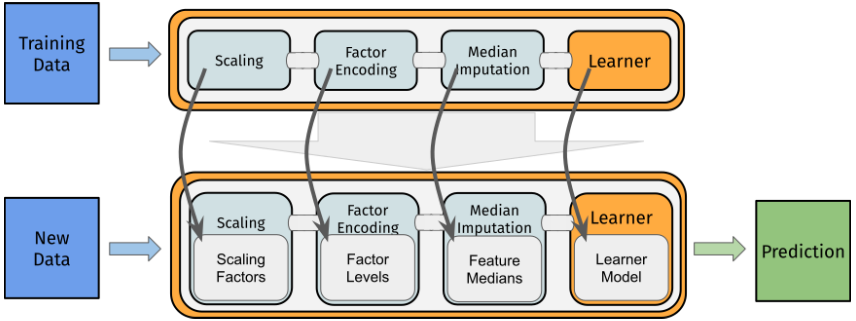
- skewed/*weird* feature distributions,
- (high cardinality) categorical features,
- functional (temporal) features,
- missing observations,
- high dimensional data,
- ...

and **improve model performance**.

TRAIN-TEST LEAKAGE

- Ultimately we are interested in predicting on future data!
- We can not use ’knowledge’ about future (test) data during training.
- Requires clearly separating train and test data in order to avoid overoptimistic conclusions:
- **Examples:**
  - Using knowledge about which features are relevant in test data
  - Scaling based on statistics of the test data
  - Missing patterns in the test data not present in training data

TRAIN-TEST LEAKAGE



AMES HOUSING DATA SET

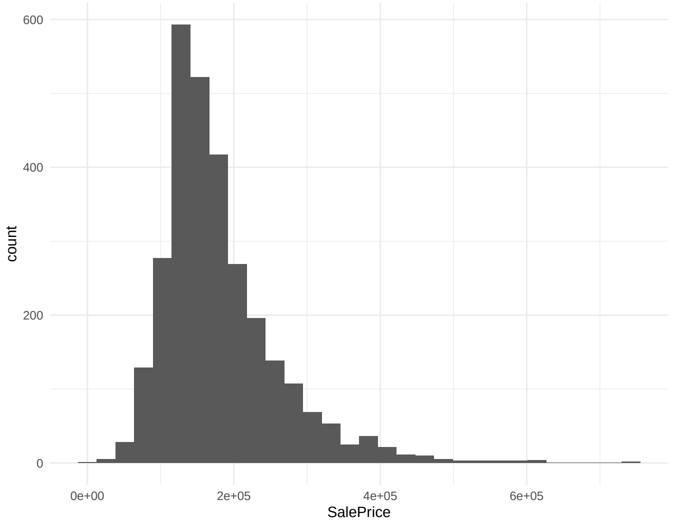
Ames is a small city in Iowa, USA.
It describes the sale of individual residential properties from 2006 to 2010.
The data was collected by Dean De Cock as an more complex alternative to the often used Boston Housing dataset.
It contains 2930 observations and

- 23 categorical features,
- 23 ordered categorical features,
- 12 integer features,
- 20 continuous features,
- 1 functional feature describing power usage of houses over a day.

Note: This is a slightly changed dataset from the original one.

AMES HOUSING DATA SET - TARGET

The goal is to predict the selling price based on these features



Handling of Categorical Features

CATEGORICAL FEATURES

A categorical feature is a feature with a finite number of discrete (unordered) *levels* $c_1, \ldots, c_k$, e.g., *House.Style=2Story* $\overset{?}{>}$ *SFoyer*.

- Categorical features are very common in practical applications.
- Except for few machine learning algorithms like tree-based methods, categorical features have to be encoded in a preprocessing step.

Encoding is the creation of a fully numeric representation from a categorical feature.

- Choosing the optimal encoding can be a challenge, especially when the number of levels k becomes very large.

ONE-HOT ENCODING

- Convert each categorical feature to k binary $(1/0)$ features, where k is the number of unique levels.
- One-Hot encoding does not loose any information of the feature and many models can correctly handle binary features.
- Given a categorical feature x_j with levels $c_1, \dots, c_k$, the new features are

$$\tilde{x}_{j,c} = \mathbb{I}(x_j)_c \quad c = c_1, \dots, c_k.$$

One-Hot encoding is often the go-to choice for the encoding of categorical features!

ONE-HOT ENCODING: EXAMPLE

Original slice of the dataset:

	SalePrice	Central.Air	Bldg.Type
1	189900	Y	1Fam
2	195500	Y	1Fam
3	213500	Y	TwnhsE
4	191500	Y	TwnhsE
5	236500	Y	TwnhsE

One-Hot Encoded:

	SalePrice	Central.Air.N	Central.Air.Y	Bldg.Type.1Fam	Bldg.Type.2fmCon	Bldg.Type.Duplex	Bldg.Type.Twnhs	Bldg.Type.TwnhsE
1	189900	0	1	1	0	0	0	0
2	195500	0	1	1	0	0	0	0
3	213500	0	1	0	0	0	0	1
4	191500	0	1	0	0	0	0	1
5	236500	0	1	0	0	0	0	1

DUMMY ENCODING

- Dummy encoding is very similar to one-hot encoding with the difference that only $k - 1$ binary features are created.
- A *reference* category is defined as all binary features being 0, i.e.,

$$\tilde{x}_{j,1} = 0, \dots, \tilde{x}_{j,k-1} = 0.$$

- Each feature $\tilde{x}_{j,1}$ represents the *deviation* from the reference category.
- While using a reference category is required for stability and interpretability in statistical models like (generalized) linear models, it is not necessary, rarely done in ML and can even have negative influence on the performance.

ONE-HOT ENCODING: LIMITATIONS

- One-Hot encoding can become extremely inefficient when number of levels becomes too large, as one additional feature is introduced for every level.
- Assume a categorical feature with $k = 4000$ levels, by using dummy encoding 4000 new features are added to the dataset.
- These additional features are very sparse.
- Handling such *high-cardinality categorical features* is a challenge, possible solutions are
    - specialized methods such as *factorization machines*,
    - **target/impact encoding**,
    - clustering feature levels or
    - feature hashing.

TARGET ENCODING

Developed to solve limitations of dummy encoding for high cardinality categorical features.

Goal: Each categorical feature x should be encoded in a single numeric feature $\tilde{x}$.

Basic definition for regression by Micci-Barreca (2001):

$$\tilde{x} = \frac{\sum_{i:x=l} y^{(i)}}{N_l}, \quad l = 1, \ldots, k,$$

where N_l is the number of observations of the l'th level of feature x.

TARGET ENCODING - EXAMPLE

	V1	V2	V3	V4	V5	V6
Foundation	BrkTil	CBlock	PConc	Slab	Stone	Wood
n	311	1244	1310	49	11	5

Encoding for wooden foundation:

	V1	V2	V3	V4	V5
house.id	17	893	986	2898	2899
SalePrice	164000	145500	143000	250000	202000
Foundation	Wood	Wood	Wood	Wood	Wood

$$\frac{164000 + 145500 + 143000 + 250000 + 202000}{5} = 180900$$

TARGET ENCODING - EXAMPLE

For all foundation types:

	V1	V2	V3	V4	V5	V6
Foundation	BrkTil	CBlock	PConc	Slab	Stone	Wood
Foundation(enc)	128107.3	148284.2	227069.5	110457.7	149786.8	180900.0

This mapping is calculated on training data and later applied to test data.

TARGET ENCODING FOR CLASSIFICATION

- Extending encoding to binary classification is straightforward, instead of the average target value the relative frequency of the positive class is used
- Multi-class classification extends this by creating one feature for each target class in the same way as binary classification.

TARGET ENCODING - ISSUES

Problem: Target encoding can assign extreme values to rarely occurring levels.
Solution: Encoding as weighted sum between global average target value and encoding value of level.

$$\tilde{x} = \lambda_l \frac{\sum_{i:x=l} y^{(i)}}{N_l} + (1 - \lambda_l) \frac{\sum_{i=1}^{n} y^{(i)}}{n}, \quad l = 1, \dots, k.$$

- λ_l can be parameterized and tuned, but optimally, tuning must be done for each feature and level separately (most likely infeasible!).
- Simple solution: Set $\lambda_l = \frac{N_l}{N_l + \epsilon}$ with regularization parameter ϵ.
- This shrinks small levels stronger to the global mean target value than large classes.

TARGET ENCODING - ISSUES

Problem: Label leakage! Information of $y^{(i)}$ is used to calculate $\tilde{x}$. This can cause overfitting issues, especially for rarely occurring classes.
Solution: Use internal cross-validation to calculate $\tilde{x}$.

- It is unclear how serious this problem is in practice.
- But: calculation of $\tilde{x}$ is very cheap, so it doesn't hurt.
- An alternative is to add some noise $\tilde{x}^{(i)} + N(0, \sigma_\epsilon)$ to the encoded samples.