Applied Machine Learning

Preprocessing and Encoding Features

UNIVERSITY OF
GOTHENBURG

CHALMERS

Richard Johansson

richard.johansson@cse.gu.se

easy case

sepal_length	sepal_width	petal_length	petal_width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
5.0	3.6	1.4	0.2
5.4	3.9	1.7	0.4
4.6	3.4	1.4	0.3
5.0	3.4	1.5	0.2
4.4	2.9	1.4	0.2
4.9	3.1	1.5	0.1

but what to do about this?

age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	target
27	Private	177119	Some-college	10	Divorced	Adm-clerical	Unmarried	White	Female	0	0	44	United-States	<=50K
27	Private	216481	Bachelors	13	Never-married	Prof-specialty	Not-in-family	White	Female	0	0	40	United-States	<=50K
25	Private	256263	Assoc-acdm	12	Married-civ-spouse	Sales	Husband	White	Male	0	0	40	United-States	<=50K
46	Private	147640	5th-6th	3	Married-civ-spouse	Transport-moving	Husband	Amer-Indian-Eskimo	Male	0	1902	40	United-States	<=50K
45	Private	172822	11th	7	Divorced	Transport-moving	Not-in-family	White	Male	0	2824	76	United-States	>50K

but what to do about this?

age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	target
27	Private	177119	Some-college	10	Divorced	Adm-clerical	Unmarried	White	Female	0	0	44	United-States	<=50K
27	Private	216481	Bachelors	13	Never-married	Prof-specialty	Not-in-family	White	Female	0	0	40	United-States	<=50K
25	Private	256263	Assoc-acdm	12	Married-civ-spouse	Sales	Husband	White	Male	0	0	40	United-States	<=50K
46	Private	147640	5th-6th	3	Married-civ-spouse	Transport-moving	Husband	Amer-Indian-Eskimo	Male	0	1902	40	United-States	<=50K
45	Private	172822	11th	7	Divorced	Transport-moving	Not-in-family	White	Male	0	2824	76	United-States	>50K

- **what we will discuss now:**
  - **encoding features** as numerical values
  - **transforming features** to make ML algorithms work better
  - dealing with **missing feature values**

how can we encode a dataset numerically?

various ways to view a tabular dataset

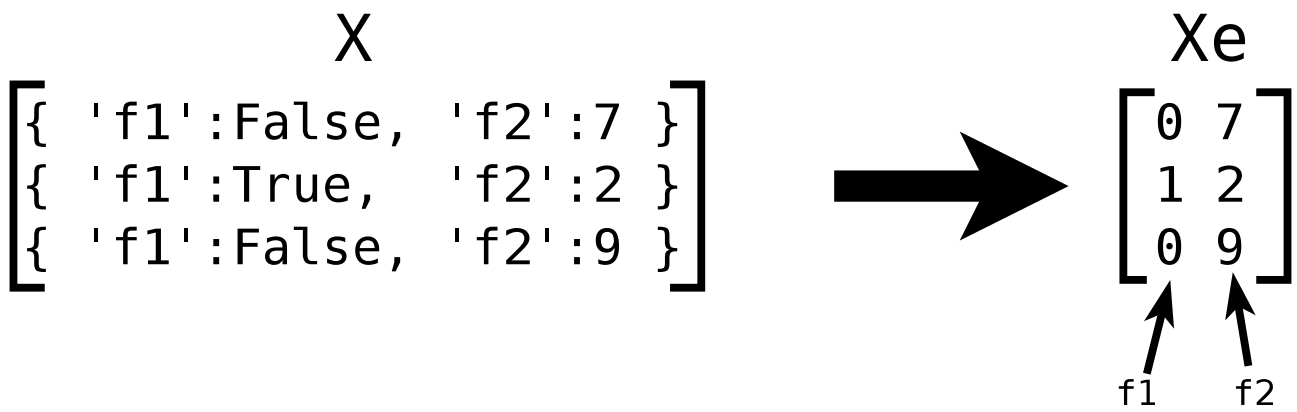
age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	target
27	Private	177119	Some-college	10	Divorced	Adm-clerical	Unmarried	White	Female	0	0	44	United-States	<=50K
27	Private	216481	Bachelors	13	Never-married	Prof-specialty	Not-in-family	White	Female	0	0	40	United-States	<=50K
25	Private	256263	Assoc-acdm	12	Married-civ-spouse	Sales	Husband	White	Male	0	0	40	United-States	<=50K
46	Private	147640	5th-6th	3	Married-civ-spouse	Transport-moving	Husband	Amer-Indian-Eskimo	Male	0	1902	40	United-States	<=50K
45	Private	172822	11th	7	Divorced	Transport-moving	Not-in-family	White	Male	0	2824	76	United-States	>50K

```
data_as_dicts = [
  { 'age':27, 'workclass':'Private', 'education':'Some-college', ... },
  { 'age':27, 'workclass':'Private', 'education':'Bachelors', ... },
  { 'age':25, 'workclass':'Private', 'education':'Assoc-acdm', ... },
]

data_as_lists = [
  [27, 'Private', 'Some-college', ... ],
  [27, 'Private', 'Bachelors', ... ],
  [25, 'Private', 'Assoc-acdm', ... ],
]
```

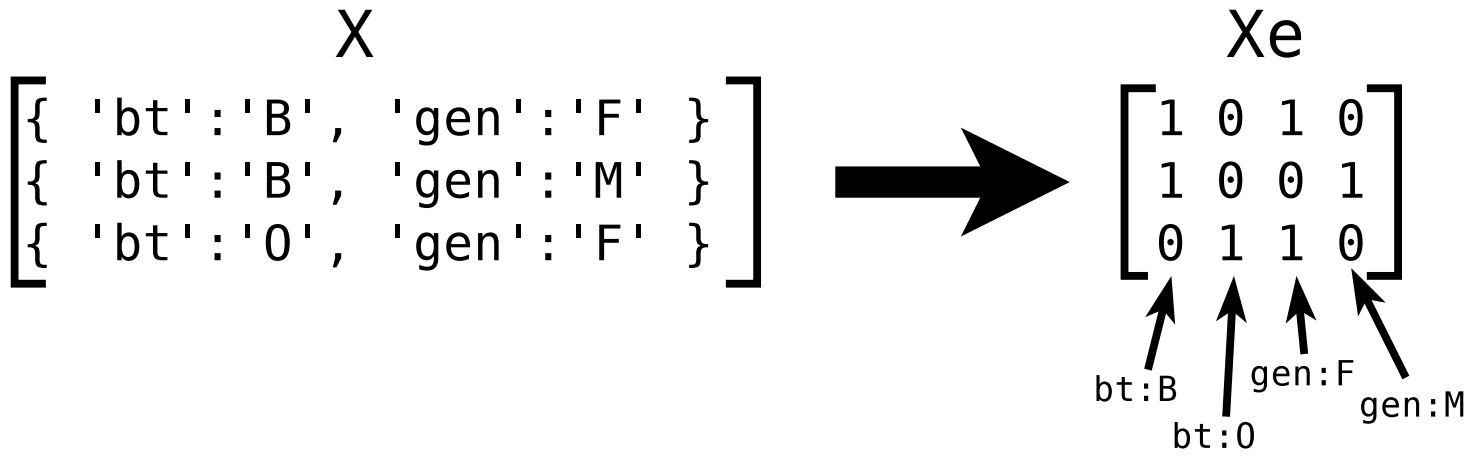
easy case: features are already numerical (or boolean)

- each feature is assigned its own column in the encoded matrix



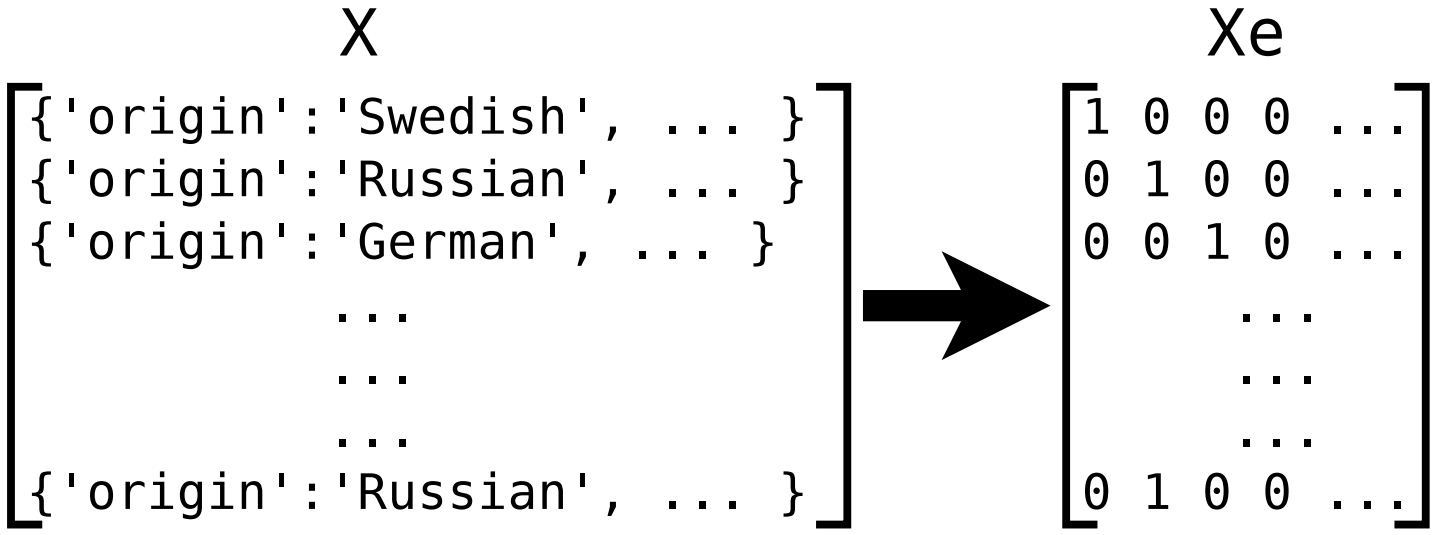
one-hot encoding of categorical features

- each **value** of a categorical feature gets its own column



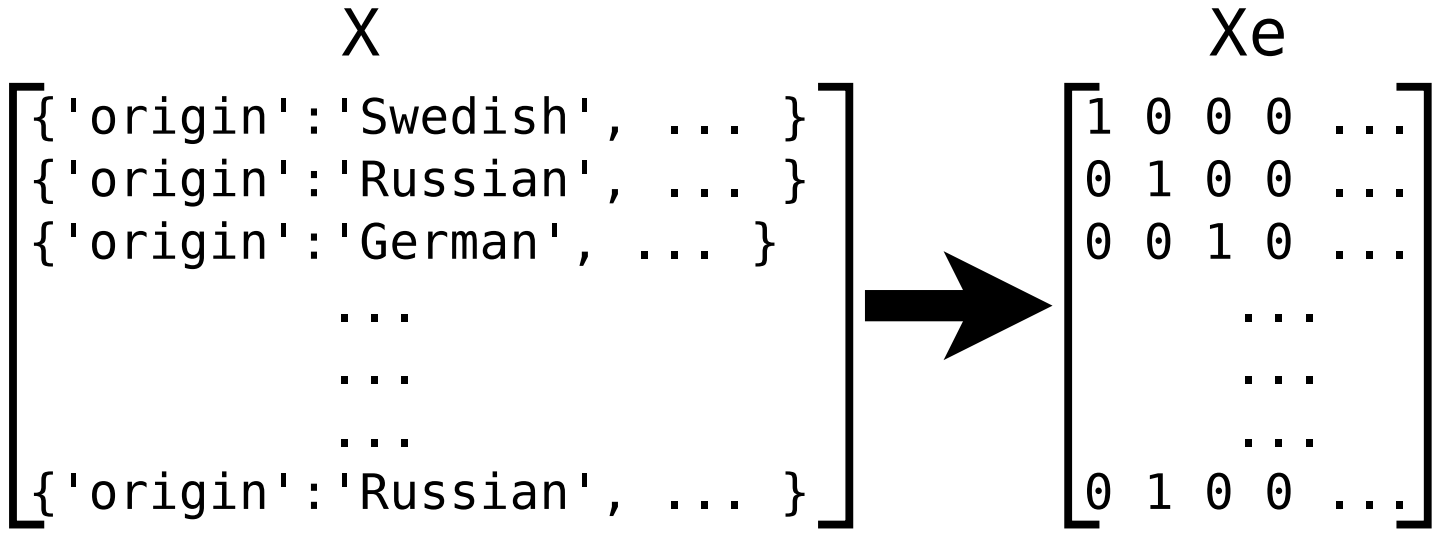
- this method is called **one-hot** or **one-of-k** encoding

isn't one-hot encoding overcomplicated and wasteful?



- why not simply encode each feature value as an integer? would make the encoded matrix more compact

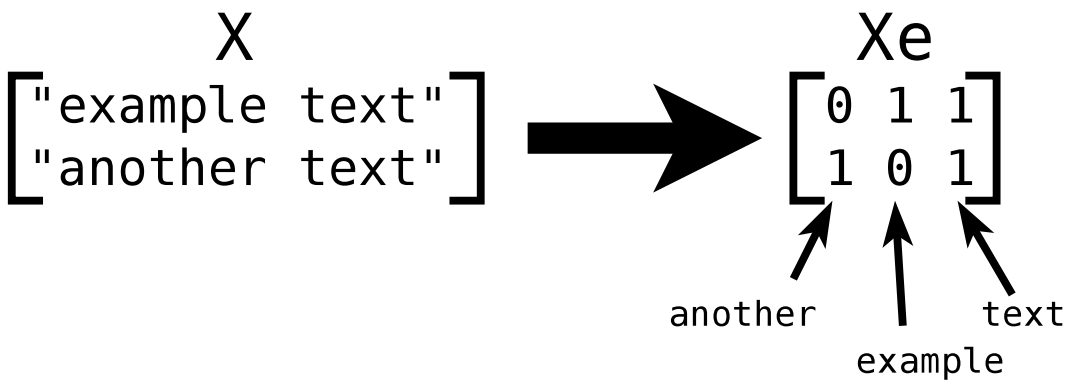
isn't one-hot encoding overcomplicated and wasteful?



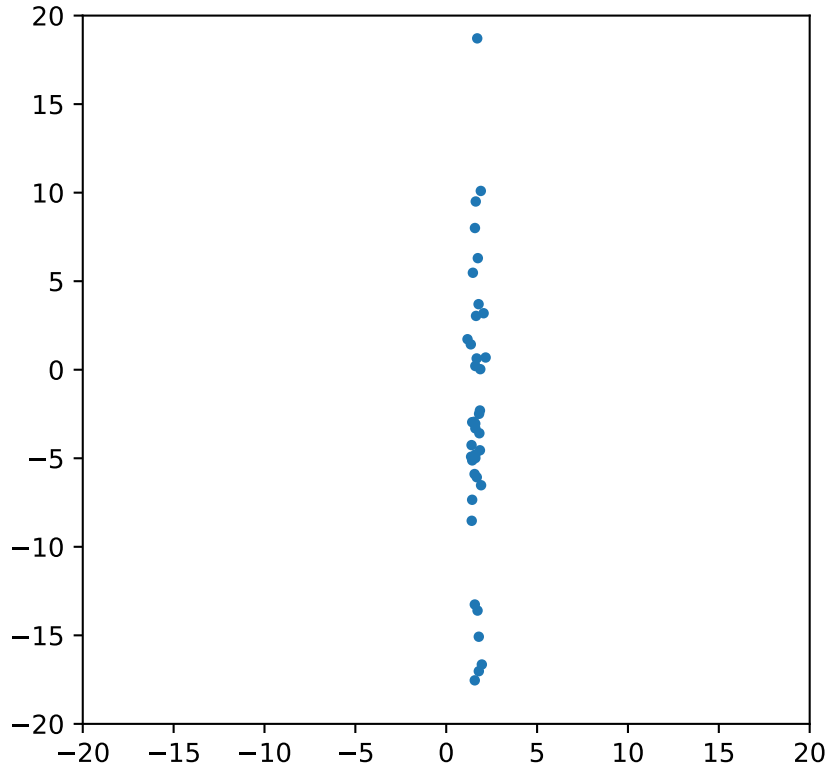
- why not simply encode each feature value as an integer? would make the encoded matrix more compact
- a naive integer encoding would create an **ordering** of the feature values that does not exist in the original data
- you can try direct integer encoding if a feature does have a natural ordering (e.g. ECTS grades **A**–**F**)

bag-of-words encoding of documents

- a **bag-of-words** representation encodes a document as a set of word counts
  - the document is a string or a list of tokens
- each observed word gets its own column



what if our features look like this?



- what if the features have different magnitudes?
- does it matter if a feature is represented as meters or millimeters?

why would "magnitude differences" be a problem?

- it **strongly affects** many models:
  - linear models (linear SVC, logistic regression, . . .)
  - neural networks
  - models based on distance or similarity (kNN, nonlin. SVC, . . .)
- it **does not matter** for most tree-based predictors
  - typically, they just consider thresholds of one feature at a time

scaling and normalization

- **min/max scaling**:

$$f_{new} = \frac{f - f_{min}}{f_{max} - f_{min}}$$

- **standard scaling**:

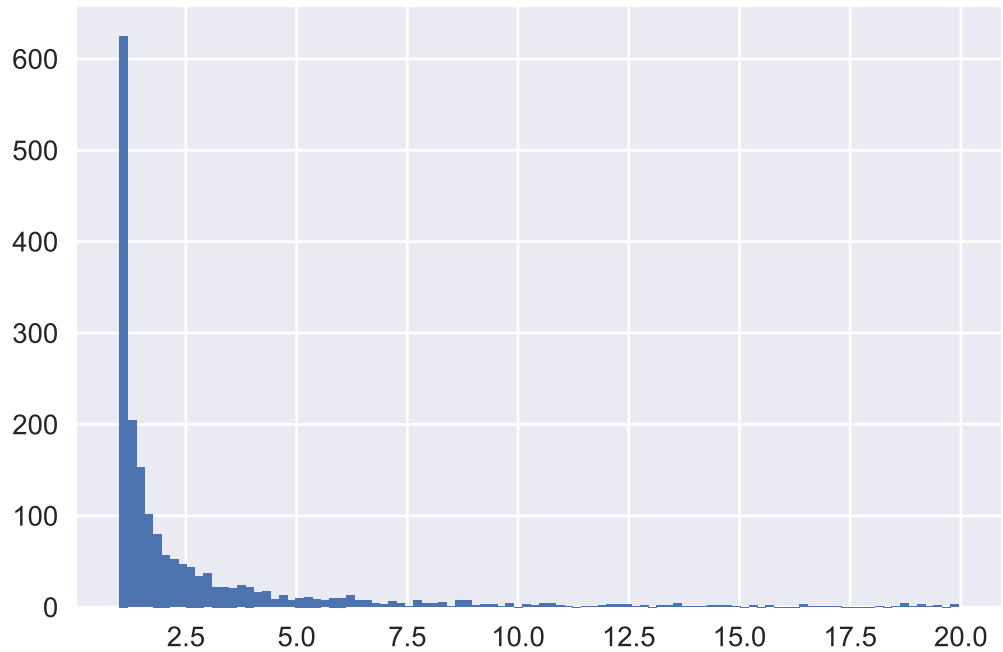
$$f_{new} = \frac{f - \bar{f}}{\sigma_f}$$

- **length normalization** (typically for documents):

$$\boldsymbol{x}_{new} = \frac{\boldsymbol{x}}{|\boldsymbol{x}|}$$

other feature transformations

- we may try to improve performance by trying other transformations
  - logarithm, square root, ...
- trial and error, exploration and your intuition



de-emphasizing common words in bag-of-words encoding

- **TF-IDF**: term frequency and inverse document frequency

$$\text{tf-idf}(t, D) = \text{tf}(t, D) \cdot \log \frac{1 + N}{1 + \text{df}(t)}$$

- where
  - $\text{tf}(t, D)$ is the count of word t in document D
  - $\text{df}(t)$ is the number of documents containing t

what can we do if some values are missing?

sepal_length	sepal_width	petal_length	petal_width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	NaN	1.3	0.2
4.6	3.1	1.5	0.2
5.0	NaN	1.4	0.2
5.4	3.9	1.7	0.4

- remove **instances**? (rows)
- remove **features**? (columns)

feature imputation

sepal_length	sepal_width	petal_length	petal_width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	NaN	1.3	0.2
4.6	3.1	1.5	0.2
5.0	NaN	1.4	0.2
5.4	3.9	1.7	0.4

- **feature imputation** methods try to "fill in the blanks"
- variants:
  - replacing with a constant (e.g. the mean feature value)
  - replacing with a random value
  - predicting the feature value from other features

conclusion

- we need to **encode** features numerically ("vectorize")
- features may need to be **scaled**, **normalized** or otherwise transformed
- we may need to **impute** missing values
- much of this is a black art!