DataSci 420

lesson 3: feature engineering

Seth Mottaghinejad

today's agenda

- **data pre-processing**
  - missing-value imputation
  - handling outliers
  - binning or not binning
- **feature engineering**
  - one-hot encoding
  - standarding or normalizing
  - risk factors / RMF

data pre-processing and feature engineering

- for some data scientists **pre-processing** can broadly refer to getting data ready for training, including **feature engineering**
- for others **pre-processing** and **feature engineering** are two separate steps:
  - **pre-processing** are the ordinary data-related tasks we perform, such as handling missing values and outliers
  - **feature engineering** is the more "advanced" ML-related tasks we perform, such as one-hot encoding, normalization, etc.

lab time

one way to think about FE is through the perspective of **transformations** whose inputs and outputs can be one or many columns, as such:

- **one to one**
- **one to many** (sometimes also called **feature extraction**)
- **many to one**

for each of the above cases, provide some examples (as you think of examples, ask yourself which one are **reversible**)

lab time

another way to think about FE is through the perspective of **transformations** that can either keep the column type or change the column type, as such:

- **numeric to numeric**
- **numeric to categorical**
- **categorical to numeric**
- **categorical to catogorical**

for each of the above cases, provide some examples

notebook time

we return to the lecture later

lab time

let's get more specific: find one FE example for each of these:

- one numeric to one numeric
- many numeric to one numeric
- one categorical to one categorical
- one categorical to one numeric
- one categorical to many numeric
- one numeric to one categorical
- many numeric to one categorical

FE examples: numeric to numeric

- **one numeric to one numeric:** normalization / standardization / log transform / trimming / flagging outliers
- **one numeric to many numeric:** extract day, month, and year from a date column
- **many numeric to one numeric:** extract driving distance from point A and B specified by their longitude latitude coordinates
- **many numeric to many numeric:** extract principal components from a set of features / RFM

FE examples: categorical and categorical

- **one categorical to one categorical:** turn a high-cardinality categorical features to a low-cardinality categorical feature, aka **recoding** or **remapping**
- **many categorical to one categorical:** create **interactions** of categorical variables
- **one categorical to many categorical:** not possible
- **many categorical to many categorical:** I can't think of a good example here, maybe some day...

FE examples: numeric to/from categorical

- **one categorical to one numeric:** risk factors (but this transformation **relies on the target**, otherwise it doesn't make sense, more on that in the next slide)
- **one categorical to many numeric:** one-hot encoding / word vectors (more on that when we talk about deep learning)
- **one numeric to one categorical:** binning (aka. bucketing or discretizing)
- **many numeric to one categorical:** k-means clustering (assuming you want to use them as features)

break time

FE specific examples: risk factors

- assumption: a **binary** (categorical) target, i.e. $Y = 1$ or $Y = 0$
- let X be a categorical feature, with x_i being its ith category
- R is a numeric feature, defined by

$$R_i = \log \frac{P(Y = 1|X = x_i)}{P(Y = 1|X = x_i)}$$

- $P(Y = 1|X = x_i)$ reads the probabilty of $Y = 1$ **given that** $X = x_i$, i.e. the **odds** of Y for the subset where $X = x_i$

FE specific examples: RFM

- assumption: we have **time series data**, i.e. we know **when** someone went to the store and **how much** they spent, so we can ask
  - **recency:** when was the last purchase they made
  - **frequency:** how often do they make a purchase in the last month (or any given window you choose)
  - **monetary:** how much money did they spend in the last month
- NOTE: in general, time series data can involve a lot of feature engineering steps, some trivial some not so much

date	customer	purchased
2020-01-01	XYZ	$50
2020-01-03	XYZ	$23
2020-02-11	XYZ	$35
2020-01-02	ABC	$50
2020-01-02	ABC	$23
2020-01-29	ABC	$35
⋮	⋮	⋮

date	customer	purchased	R	F	M
2020-01-01	XYZ	$50	NA	NA	NA
2020-01-03	XYZ	$23	2 days	2	$73
2020-01-11	XYZ	$35	8 days	1	$35
2020-01-02	ABC	$65	NA	NA	NA
2020-01-03	ABC	$25	1 day	2	$90
2020-01-29	ABC	$35	26 days	1	$35
⋮	⋮	⋮	⋮	⋮	⋮

FE more practical examples

- **retail:** RFM (recency, frequency, monetary)
- **economics:** elasticity of demand (measures sensitivity to price)
- **finance:** PE ratio (or many other financial ratios)
- **NLP:** TF-IDF (term frequency–inverse document frequency)
- **image processing:** any kind of filter
- **GIS:** measuring distance or driving time
- **mathematics:** change of coordinate (e.g. cartesian to polar)
- **graph-based:** number of connections, Google's PageRank

importance of feature engineering

- feature engineering can improve model performance much more than **model selection** or **hyper-parameter tuning** (topics of future lectures)
- **traditional ML** algorithms require us to do feature engineering manually, usually relying on some **domain knowledge**
- **deep learning** algorithms have feature engineering built in, meaning they engineer their own features during training without us needing to intervene

the end