Technische Hochschule Rosenheim

Modul - Unsupervised and Reinforcement Learning (URL)

Bachelor Programme AAI

03 - Data Preprocessing and Normalization

Prof. Dr. Marcel Tilly

Faculty of Computer Science, Cloud Computing

Agenda

Technische Hochschule Rosenheim

> Code can be found in DataProcessing.ipynb on GitHub or hosted on myBinder

- We will talk about more realistic data
- How to prepare and pre-process data

Datasets so far

- Very artificial and synthetic
- Limited on 2-dim (not by algorithm)
- It had the same scale on all features
- Does this reflect reality?

Use-Case: Retail Marketing

Technische Hochschule Rosenheim

- Retail companies often use clustering to identify groups of households that are similar to each other.
- For example, a retail company may collect the following information on households:
  - Household income
  - Household size
  - Head of household Occupation
  - Distance from nearest urban area
  - **What else???**
- They can then feed these variables into a clustering algorithm to perhaps identify the following clusters:-
  - Cluster 1: Small family, high spenders
  - Cluster 2: Larger family, high spenders
  - Cluster 3: Small family, low spenders
  - Cluster 4: Large family, low spenders

Use-Case: Streaming Services

Technische Hochschule Rosenheim

- Streaming services often use clustering analysis to identify viewers who have similar behavior.
- For example, a streaming service may collect the following data about individuals:
  - Minutes watched per day
  - Total viewing sessions per week
  - Number of unique shows viewed per month
  - **Is there more?**
- Using these metrics, a streaming service can perform cluster analysis to identify high usage and low usage users so that they can know who they should spend most of their advertising dollars on.

Use-Case: Sports Science

Technische Hochschule Rosenheim

- Data scientists for sports teams often use clustering to identify players that are similar to each other.
- For example, professional basketball teams may collect the following information about players:
  - Points per game
  - Rebounds per game
  - Assists per game
  - Steals per game
  - **Do you have an idea what is missing?**
- They can then feed these variables into a clustering algorithm to identify players that are similar to each other so that they can have these players practice with each other and perform specific drills based on their strengths and weaknesses.

Use-Case: Email Marketing

Technische Hochschule Rosenheim

- Many businesses use cluster analysis to identify consumers who are similar to each other so they can tailor their emails sent to consumers in such a way that maximizes their revenue.
- For example, a business may collect the following information about consumers:
  - Percentage of emails opened
  - Number of clicks per email
  - Time spent viewing email
  - **Mh, maybe more?**
- Using these metrics, a business can perform cluster analysis to identify consumers who use email in similar ways and tailor the types of emails and frequency of emails they send to different clusters of customers.

Use-Case: Health Insurance

Technische Hochschule Rosenheim

- Actuaries at health insurance companies often used cluster analysis to identify “clusters” of consumers that use their health insurance in specific ways.
- For example, an actuary may collect the following information about households:
  - Total number of doctor visits per year
  - Total household size
  - Total number of chronic conditions per household
  - Average age of household members
- An actuary can then feed these variables into a clustering algorithm to identify households that are similar. The health insurance company can then set monthly premiums based on how often they expect households in specific clusters to use their insurance.

Is the data comparable?

Technische Hochschule Rosenheim

Feature	Type	Range	Unit
Household size			
Head of household Occupation			
Distance from nearest urban area			
Time spent viewing email			
Points per game			
Rebounds per game			
Total number of doctor visits per year			
Total number of chronic conditions per household			
Average age of household members			

What can happen with data?

Technische Hochschule Rosenheim

- **Missing data**: Values are `0` or values are not set
  - Fill data

```
dataset_new[["Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI"]]
   .replace(0, np.NaN)
dataset_new["Glucose"].fillna(dataset_new["Glucose"].mean(), inplace = True)
```

- **Discrete types**: Semantic values

```
cat_cols=[ 'diabetes']
le=preprocessing.LabelEncoder()
data[cat_cols]=data[cat_cols].apply(le.fit_transform())
```

- **Continuous types**: Data has different scaling

Encoding

Technische Hochschule Rosenheim

- LabelEncoding
- One-hot Encoding

Label Encoding

Technische Hochschule Rosenheim

- The categorical value represents the numerical value of the entry in the dataset.
  - For example: if there were to be another company in the dataset, it would have been given categorical value as 4. As the number of unique entries increases, the categorical values also proportionally increases.

CompanyName	Categoricalvalue	Price
VW	1	20000
Acura	2	10011
Honda	3	30000
Honda	3	10000

- The categorical values start from 0 goes all the way up to N-1 categories.

> Why it can be problematic to use label encoding for model training?

One-hot Encoding

Technische Hochschule Rosenheim

- One hot encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction.

CompanyName	Categoricalvalue	Price
VW	[1,0,0]	20000
Acura	[0,1,0]	10011
Honda	[0,0,1]	30000
Honda	[0,0,1]	10000

Considerations for Clustering

Technische Hochschule Rosenheim

- **Case 1 - Incomparable units** (Height vs weight): You cannot compare, so the default decision is to standardize (equalize variances) => "every unique aspect of nature is assumed to have same, unit variability of observations"
- **Case 2 - Same units** (Height vs circumference): These are clearly independent (conceptually, not statistically) phenomena of reality. Their same-unitness seems a coincidence. The default decision is to standardize the features.
- **Case 3 - Same units, juxtaposed features** (Length of right arm vs of left arm): We could naturally compare the two lengths if we need so, they two are interchangeable, in a sense. The default decision is to leave variances as is (no matter how much they differ). Because "leave nature under study be how it is".
- **Case 4 - Undecided whether 2 or 3** (Length of arm vs length of leg): We could compare these but we are not interested in that, rather, we prefer to see the lengths as separate dimensions (albeit not irrelative phenomena).

What to do with different data?

Technische Hochschule Rosenheim

- **Scale** generally means to change the range of the values. The shape of the distribution doesn't change. Think about how a scale model of a building has the same proportions as the original, just smaller. That's why we say it is drawn to scale. The range is often set at 0 to 1.
- **Standardize** generally means changing the values so that the distribution's standard deviation equals one. Scaling is often implied.
- **Normalize** can be used to mean either of the above things (and more!). I suggest you avoid the term normalize, because it has many definitions and is prone to creating confusion.

Why Scale, Standardize, or Normalize?

Technische Hochschule Rosenheim

> Many machine learning algorithms perform better or converge faster when features are on a relatively similar scale and/or close to normally distributed.

Examples of such algorithm families include:

- linear and logistic regression
- **nearest neighbors**
- neural networks
- support vector machines with radial bias kernel functions
- **principal components analysis**
- linear discriminant analysis

- Scaling and standardizing can help features arrive in more digestible form for these algorithms.

MinMaxScaler

Technische Hochschule Rosenheim

For each value v in a feature F, *MinMaxScaler* subtracts the minimum value in the feature and then divides by the range and then normalize to v_n. The range is the difference between the original maximum and original minimum.

$$v = \frac{v - min(F)}{max(F) - min(F)}$$

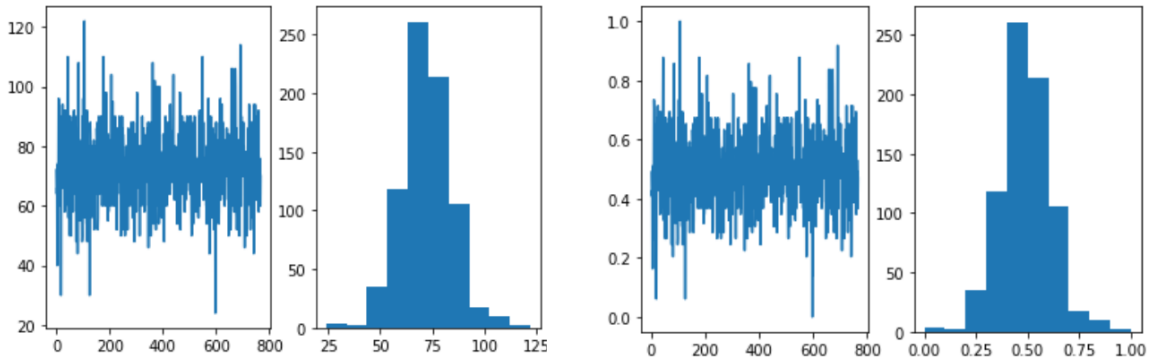
$$v_n = v * (max(V) - min(V)) - min(V)$$

Check: Documentation in scikit-learn:https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

MinMaxScaler

- *MinMaxScaler* preserves the shape of the original distribution. It doesn't meaningfully change the information embedded in the original data.
- The default range for the feature returned by MinMaxScaler is 0 to 1.

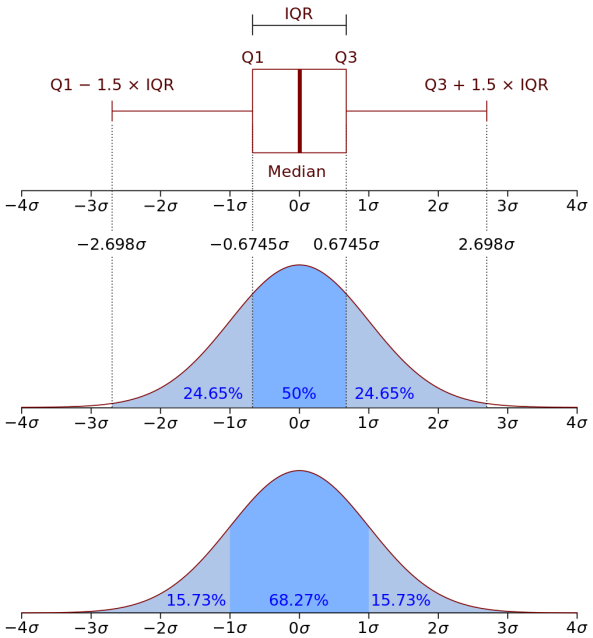
> Note that *MinMaxScaler* doesn't reduce the importance of outliers.
>
> *MinMaxScaler* isn't a bad place to start, unless you know you want your feature to have a normal distribution or you have outliers and you want them to have reduced influence.



Statistics

- A *quartile* is a type of quantile which divides the number of data points into four parts, or quarters
- The data must be ordered from smallest to largest to compute quartiles.

- The first quartile (Q1) is defined as the middle number between the smallest number (minimum) and the median of the data set. 25% of the data is below this point.
- The second quartile (Q2) is the median of a data set; thus 50% of the data lies below this point.
- The third quartile (Q3) is the middle value between the median and the highest value (maximum) of the data set. 75% of the data lies below this point



RobustScaler

Technische Hochschule Rosenheim

Robust Scaler algorithms scale features that are robust to outliers. The method it follows is almost similar to the MinMax Scaler but it uses the interquartile range (rather than the min-max used in MinMax Scaler). The median and scales of the data are removed by this scaling algorithm according to the quantile range.

For each value v in a feature F, *RobustScaler* subtracts the quartile value Q1 in the feature and then divides by the interqurtile range (Q3-Q1).

$$v_i = \frac{v_i - Q_1(F)}{Q_3(F) - Q_1(F)}$$

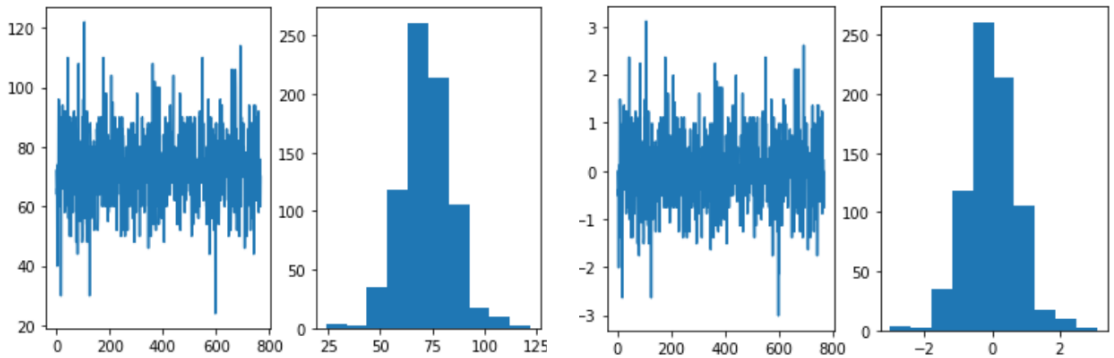
where Q1 is the 1st quartile, and Q3 is the third quartile.

> Check scikit-learn documentation: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html?highlight=robustscaler#sklearn.preprocessing.RobustScaler

RobustScaler

- The *RobustScaler* does not scale the data into a predetermined interval like MinMaxScaler.
- It does not meet the strict definition of scale.
- It reduces the effects of outliers.

> Note that the range for each feature after *RobustScaler* is applied is larger than it was for MinMaxScaler.
>
> Use *RobustScaler* if you want to reduce the effects of outliers, relative to MinMaxScaler.



StandardScaler

Technische Hochschule Rosenheim

The *StandardScaler* standardizes features by removing the mean and scaling to unit variance.

The standard score z of a value v in feature F is calculated as:

$$z = \frac{x - u}{s}$$

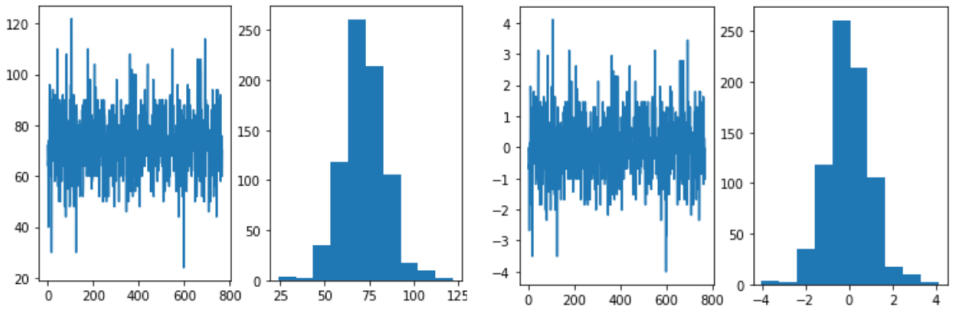
where u is the mean of the feature F or zero (if with_mean=False), and s is the standard deviation of the feature F or one if with_std=False.

> Check scikit-learn documentation: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html?highlight=standardscaler#sklearn.preprocessing.StandardScaler

StandardScaler

Technische Hochschule Rosenheim

- StandardScaler is the industry's go-to algorithm.
- StandardScaler standardizes a feature by subtracting the mean and then scaling to unit variance.
  - Unit variance means dividing all the values by the standard deviation.
  - StandardScaler does not meet the strict definition .
- StandardScaler results in a distribution with a standard deviation equal to 1. The variance is equal to 1 also, because variance = standard deviation squared. And 1 squared = 1.
- StandardScaler makes the mean of the distribution approximately 0.



Wrap-up on Data

Technische Hochschule Rosenheim

- No number data needs to be encoded!
- Scaling and standardizing your data is often a good idea.
- Tips:
  - Use **StandardScaler** if you want each feature to have zero-mean, unit standard-deviation.
  - Use MinMaxScaler if you want to have a light touch. It's non-distorting.
  - Use RobustScaler if you have outliers and want to reduce their influence.

> Also:
>
> - Check out scikit-learn's transformer/scaler:
> - e.g. QuantileTransformer(output_distribution='normal')

Summary

Technische Hochschule Rosenheim

- Today we discussed data pre-processing
  - concrete values into numbers
  - Normalization and scaling
- MinMaxScaler, RobustScaler, StandardScaler

> Code can be found in DataProcessing.ipynb on GitHub or hosted on myBinder

Exercise

Technische Hochschule Rosenheim

- On GitLab: https://inf-git.fh-rosenheim.de/aai-url/03_exercise
- Work on Python, again!
- Work on data and data pre-processing
- Try a more real dataset