BITS Pilani

Pilani | Dubai | Goa | Hyderabad

Introduction to Data Science

Module # 6 : Data Wrangling

IDS Course Team

BITS Pilani

The instructor is gratefully acknowledging the authors who made their course materials freely available online.

Data Quality Issues

- Noise and outliers;
- Missing data
- Inconsistent data
- Duplicate data
- Data that is biased
- Data that is unrepresentative of the phenomenon or population that the data is supposed to describe.

Pre-processing on Data

- Improve Data Quality
- To better fit a specified data mining or machine learning technique or tool.
- Improve the efficiency of the data mining or machine learning technique or tool.

Tasks in Data Preprocessing on Data

- Data Cleaning
  - applied to remove noise and correct inconsistencies in data.
- Data Integration
  - merges data from multiple sources into a coherent data store such as a data warehouse.
- Data Reduction
  - reduce data size by, for instance, aggregating, eliminating redundant features, or clustering.
- Data Transformations
  - scale the data to fall within a smaller range like 0.0 to 1.0 to improve the accuracy and efficiency of the algorithms involving distance measurements.

Table of Contents

1. Data Cleaning
2. Data Aggregation
3. Data Sampling
4. Handling Numeric Data
    - Discretization
5. Data Transformation
6. Dealing with Textual Data
7. Categorical Encoding
    - Binarization

Data Cleaning

- Data cleaning routines work to ”clean” the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies.
- Improve the efficiency of the data mining or machine learning technique or tool.

Missing Values

Techniques used to handle missing values:

- Ignore the tuple.
- Fill in the missing value manually.
- Use a global constant to fill in the missing value.
- Use a measure of central tendency for the attribute (e.g., the mean or median) to fill in the missing value.
- Use the attribute mean or median for all samples belonging to the same class as the given tuple.
- Use the most probable value to fill in the missing value.

Missing Values

- Ignore the tuple.
  - Used when the class label is missing in a classification task.
  - Not very effective, unless the tuple contains several attributes with missing values.
  - Poor technique when the percentage of missing values per attribute varies considerably.
  - By ignoring the tuple, we do not make use of the remaining attributes' values in the tuple. Such data could have been useful to the task at hand.
- Fill in the missing value manually.
  - Time consuming.
  - May not be feasible given a large data set with many missing values.

Missing Values

- Use a global constant to fill in the missing value.
  - Replace all missing attribute values by the same constant such as a label like ”Unknown” or -1.
  - If missing values are replaced by, say, ”Unknown,” then the mining program may mistakenly think that they form an interesting concept, since they all have a value in common—that of ”Unknown.” Hence, although this method is simple, it is not foolproof.
- Use a measure of central tendency for the attribute.
  - Central tendency indicates the ”middle” value of a data distribution. E.g., mean or median
  - For normal (symmetric) data distributions, the mean can be used.
  - Skewed data distribution should employ the median.

Missing Values

- Use the attribute mean or median for all samples belonging to the same class as the given tuple.
  - For example, if classifying customers according to credit risk, we may replace the missing value with the mean income value for customers in the same credit risk category as that of the given tuple.
  - If the data distribution for a given class is skewed, the median value is a better choice.
- Use the most probable value to fill in the missing value.
  - This may be determined with regression, inference-based tools using a Bayesian formalism, or decision tree induction.
  - For example, using the other customer attributes in the data set, we may construct a decision tree to predict the missing values for income.
  - Most popular strategy.

Missing Values - Python Examples

- In Pandas missing data is represented by two values:
  - None: None is a Python singleton object that is often used for missing data in Python code.
  - NaN : NaN (Not a Number), is a special floating-point value recognized by all systems that use the standard IEEE floating-point representation.
- Finding Missing Values
  - isnull()
  - notnull()
- Handing Missing Value
  - dropna()
  - fillna()
  - replace()
  - interpolate()

Noisy Data

- Noise is a random error or variance in a measured variable. Outliers may represent noise.
- Basic statistical description techniques (e.g., boxplots and scatter plots), and data visualization can be used to identify outliers.
- Noisy data can be removed by using smoothing techniques.
  - Binning
    - ★ Smoothing by bin means
    - ★ Smoothing by bin medians
  - Regression
  - Outlier Analysis

All techniques learned in Data Mining.

Noisy Data

Data Discretization (or transformation) and Data Reduction

- Many data smoothing methods are also used for data discretization (a form of data transformation) and data reduction.
  - For example, the binning techniques reduce the number of distinct values per attribute.
  - This acts as a form of data reduction for logic-based data mining methods, such as decision tree induction, which repeatedly makes value comparisons on sorted data,
- Concept hierarchies are a form of data discretization that can also be used for data smoothing.
  - For example: A concept hierarchy for price may map real price values into three categories: inexpensive, moderately priced, and expensive.
  - This mapping reduces the number of data values to be handled by the mining process.

Table of Contents

1. Data Cleaning
2. Data Aggregation
3. Data Sampling
4. Handling Numeric Data
   - Discretization
5. Data Transformation
6. Dealing with Textual Data
7. Categorical Encoding
   - Binarization

Data Aggregation

- Aggregation is combining two or more objects into a single object.
  - Consider a data set consisting of transactions (data objects) recording the daily sales of products in various store locations (Minneapolis, Chicago, Paris, ...) for different days over the course of a year.
  - One way to aggregate transactions for this data set is to replace all the transactions of a single store with a single store-wide transaction.
  - This reduces
    - ★ The hundreds or thousands of transactions that occur daily at a specific store to a single daily transaction.
    - ★ The number of data objects is reduced to the number of stores.

Data Aggregation

- To create the aggregate transaction that represents the sales of a single store or date.
- Quantitative attributes, such as price, are typically aggregated by taking a sum or an average.
- Qualitative attribute, such as item description, can either be omitted or summarized as the set of all the items that were sold at that location.
- The data in the table can also be viewed as a multidimensional array, where each attribute is a dimension
- From this viewpoint, aggregation is the process of eliminating attributes (such as the type of item) or reducing the number of values for a particular attribute (e.g., reducing the possible values for date from 365 days to 12 months).
- This type of aggregation is commonly used in Online Analytical Processing (OLAP).

Data Aggregation

- First, the smaller data sets resulting from data reduction require less memory and processing time.
  - This allows the use of more expensive data mining algorithms.
- Second, aggregation provides a high-level view of the data instead of a low-level view
  - E.g., aggregating over store locations and months gives us a monthly, per store view of the data instead of a daily, per item view.
- Finally, the behavior of groups of objects or attributes is often more stable than that of individual objects or attributes.
  - This statement reflects the statistical fact that aggregate quantities, such as averages or totals, have less variability than the individual objects being aggregated.
- Disadvantage of aggregation is the potential loss of interesting details.
  - In the store example, aggregating over months loses information about which day of the week has the highest sales.

Table of Contents

1. Data Cleaning
2. Data Aggregation
3. Data Sampling
4. Handling Numeric Data
   - Discretization
5. Data Transformation
6. Dealing with Textual Data
7. Categorical Encoding
   - Binarization

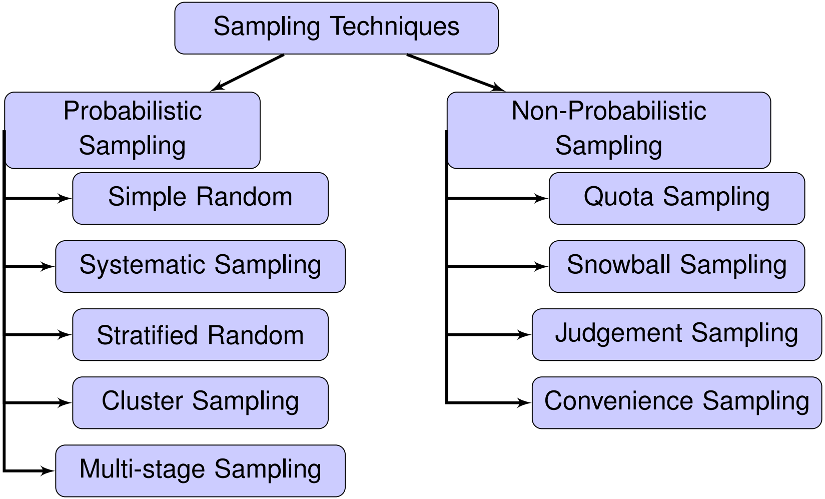
Data Sampling

- A process by which representative samples are selected from a well defined population is known as sampling.
- Sampling is a technique used for selecting a subset of the data objects to be analyzed.
- Sampling allows us to get information about the population based on the statistics from a subset of the population (sample), without having to investigate every individual.
- Why sampling?
  - Population may be unknown or infinite.
  - Resources (Money, Manpower, Material and Time (3MT) required may be huge.
  - Samples selected as representative of population gives consistent and reliable results (Estimates), what could have been obtained from Population (True value).

Data Sampling

- In statistics, sampling can be used for both preliminary investigation of the data and the final data analysis.
- Sampling can also be very useful in data mining.
- The motivations for sampling in statistics and data mining are often different.
  - Statisticians use sampling because obtaining the entire set of data of interest is too expensive or time consuming.
  - Data miners sample because it is too expensive or time consuming to process all the data.
- In some cases, using a sampling algorithm can reduce the data size to the point where a better, but more expensive algorithm can be used.

Sampling Techniques


Sampling Techniques

Probabilistic Sampling

- Simple Random
- Systematic Sampling
- Stratified Random
- Cluster Sampling
- Multi-stage Sampling

Non-Probabilistic Sampling

- Quota Sampling
- Snowball Sampling
- Judgement Sampling
- Convenience Sampling

Probabilistic Sampling Techniques

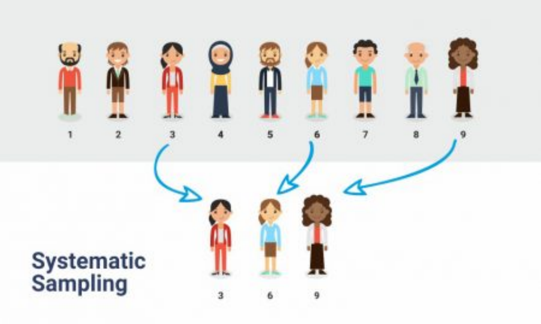
Probabilistic Sampling means that every item in the population has an equal chance of being included in sample.

Simple Random Sampling means that every case of the population has an equal probability of inclusion in sample.

- Eg: Randomly picking mango from a basket of fruits.
- Sampling without replacement
- Sampling with replacement

Probabilistic Sampling Techniques

Systematic Sampling is where every nth case after a random start is selected. Eg: Picking every 5th fruit from a basket of fruits.



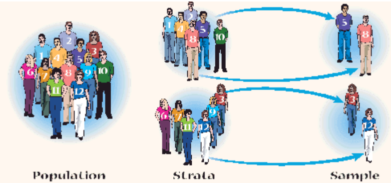
Probabilistic Sampling Techniques

Stratified Sampling is where the population is divided into strata and a random sample is taken from each strata.

Eg: One mango, one orange, one banana from a basket of fruits.

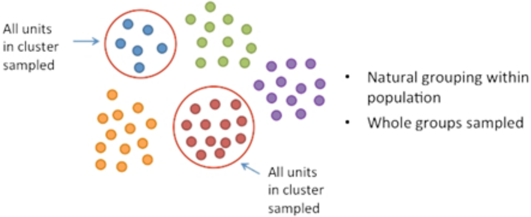
Two versions of stratified sampling:

- Equal numbers of objects are drawn from each group even though the groups are of different sizes.
- The number of objects drawn from each group is proportional to the size of that group.



Probabilistic Sampling Techniques

Cluster Sampling is where the whole population is divided into clusters or groups. A random sample is taken from these clusters, all of which are used in the final sample.
Eg: Divide the fruit basket into various clusters and choose a sample from each cluster.



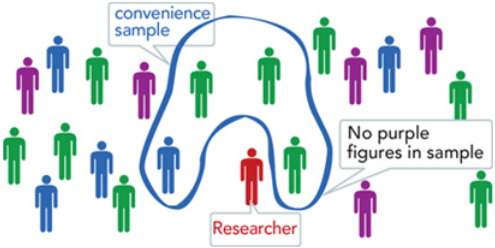
- Natural grouping within population
- Whole groups sampled

Non-Probabilistic Sampling Techniques

Non-Probabilistic Sampling is often associated with case study research design and qualitative research. A sample of participants or cases does not need to be representative. Clarity is required for the inclusion of cases.

Convenience Sampling is selecting participants because they are often readily and easily available.

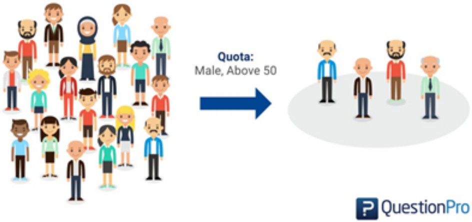
Eg: Using students or using friends or family as part of sample.



Non-Probabilistic Sampling Techniques

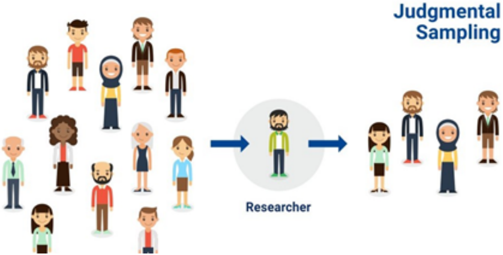
Quota Sampling Participants are chosen on the basis of predetermined characteristics so that the total sample will have the same distribution of characteristics as the wider population.

Eg: Selection of fruits based on colour.



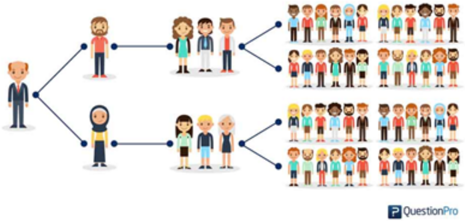
Sampling Techniques

Purposive or Judgmental Sampling particular events are selected deliberately in order to provide important information that cannot be obtained from other choices.



Non-Probabilistic Sampling Techniques

Snowball Sampling Uses a few cases to help encourage other cases to take part in the study, thereby increasing sample size.
Eg: secret societies and inaccessible professions



Sample Size

- Margin of error or Confidence interval *e*
  - A percentage that indicates how close the sample results will be to the true value of the overall population.
  - Smaller margin of errors will result in more accurate answers.
  - Smaller margin of error requires a larger sample.
  - Representation $\pm 5\%$
- Confidence level
  - Measures the degree of certainty regarding how well a sample represents the overall population within the chosen margin of error.
  - A larger confidence level indicates a greater degree of accuracy.
  - A larger confidence level requires a larger sample.
  - A confidence level of 95% allows you to claim that you are 95% certain that your results accurately fall within your chosen margin of error.

Sample Size

- Z-score *z*
  - Z-score is a constant value automatically set based on the confidence level.

Confidence (%)	z-score
99	2.58
95	1.96
90	1.65
85	1.44
80	1.28

- Standard Deviation *p*
- Population size *N*

Sample Size

- Sample Size n

$$n = \frac{\frac{z^2 p(1-p)}{e^2}}{1 + \frac{z^2 p(1-p)}{e^2 N}}$$

- For large populations Cochran's formula is used.

$$n = \frac{z^2 p(1-p)}{e^2}$$

Sample Size – Example

In study on the inhabitants of a large town, find out how many households serve breakfast in the mornings. Assume that we want 95% confidence, and at least $\pm 5\%$ percent precision.

- Assume that half of the families serve breakfast. This gives us maximum variability. $p = 0.5$
- Confidence level = 95 %. $z = 1.96$
- $e = 0.5$
- Cochran's formula

$$n = \frac{z^2 p(1-p)}{e^2}$$

- $n = 385$

Table of Contents

1. Data Cleaning
2. Data Aggregation
3. Data Sampling
4. Handling Numeric Data
   - Discretization
5. Data Transformation
6. Dealing with Textual Data
7. Categorical Encoding
   - Binarization

Handling Numeric Data

Techniques are

- Discretization
- Binarization
- Normalization
- Smoothing

Discretization

- Convert continuous attribute into a discrete attribute.
- Why?
  - Some data mining or ML algorithms (e.g., certain classification algorithms), require that the data be in the form of discrete or categorical attributes.
  - E.g., Naive Bayes, decision trees and their ensembles including Random forest, Minimum distance classifiers or KNN prefer discrete features.
- Issues
  - How to choose the number of intervals K ?
  - How to define the cut points which are relevant according to the studied problem?

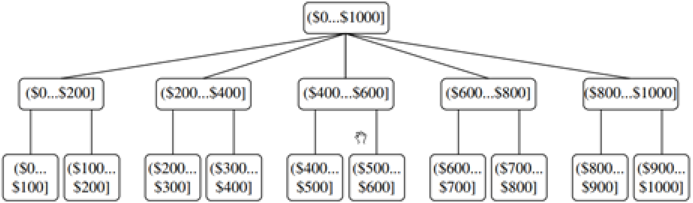
Age	Alzheimer
60	Yes
45	No
55	Yes
50	No

Discretization

- Discretization involves converting the raw values of a numeric attribute (e.g., age) into
  - interval labels (e.g., 0–10, 11–20, etc.)
  - conceptual labels (e.g., youth, adult, senior)
- Discretization Process
  - Data discretization and concept hierarchy generation are also forms of data reduction.
  - The raw data are replaced by a smaller number of interval or concept labels.
  - This simplifies the original data and makes the mining more efficient.
  - The resulting patterns mined are typically easier to understand.
  - Concept hierarchies are also useful for mining at multiple abstraction levels.

Concept Hierarchy

- Divide the range of a continuous attribute into intervals.
- Interval labels can then be used to replace actual data values.
- The labels, in turn, can be recursively organized into higher-level concepts.
- This results in a concept hierarchy for the numeric attribute.



A concept hierarchy for the attribute *price*, where an interval ($*X* ... $*Y*] denotes the range from $*X* (exclusive) to $*Y* (inclusive).

Discretization Techniques

Discretization techniques can be categorized based on how the discretization is performed.

- Supervised vs. Unsupervised discretization
  - If the discretization process uses class information, then we say it is supervised discretization. Otherwise, it is unsupervised.
- Top-down discretization or Splitting
  - The process starts by first finding one or a few points (called split points or cut points) to split the entire attribute range. Then the process repeats recursively on the resulting intervals.
- Bottom-up discretization or Merging
  - The process starts by considering all of the continuous values as potential split-points. Removes some by merging neighborhood values to form intervals. Then recursively applies this process to the resulting intervals.

Discretization Techniques

- Unsupervised discretization
  - Binning [Equal-interval, Equal-frequency] (Top-down split)
  - Histogram analysis (Top-down split)
  - Clustering analysis (Top-down split or Bottom-up merge)
  - Correlation analysis (Bottom-up merge)
- Supervised discretization
  - Entropy-based Decision Tree discretization (Top-down split)

Unsupervised Discretization

- Class labels are ignored.
- The best number of bins *k* is determined experimentally.
- User specifies the number of intervals and/or how many data points to be included in any given interval.
- Use Binning methods.

Unsupervised Discretization

- Heuristics used to choose intervals
  1. The number of intervals for each attribute should not be smaller than the number of classes (if known).

     $$K \geq C \text{ classes}$$

  2. Choose the number of intervals, n_{F_i}, for each attribute, $F_i(i = 1, \ldots, n)$ where n is the number of attributes)

     $$n_{F_i} = \frac{M}{3} \times C$$

     where M is the number of training examples and C is the number of known classes.

Discretization by Binning Methods

1. Equal Width (distance) binning
   - Each bin has equal width.

$$\textit{width} = \textit{interval} = \frac{\max - \min}{\#\textit{bins}}$$

   - Highly sensitive to outliers.
   - If outliers are present, the width of each bin is large, resulting in skewed data.
2. Equal Depth (frequency) binning
   - Specify the number of values that have to be stored in each bin.
   - Number of entries in each bin are equal.
   - Some values can be stored in different bins.

Binning Example

Discretize the following data into 3 discrete categories using binning technique.

70, 70, 72, 73, 75, 75, 76, 76, 78, 79, 80, 81, 53, 56, 57, 63, 66, 67, 67, 67, 68, 69, 70, 70.

Binning Example

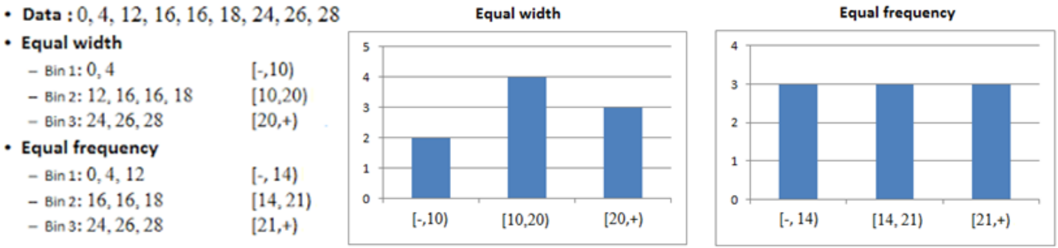
Original Data	53, 56, 57, 63, 66, 67, 67, 67, 68, 69, 70, 70, 70, 70, 72, 73, 75, 75, 76, 76, 78, 79, 80, 81			
Method		Bin1	Bin 2	Bin 3
Equal Width	width= 81-53 = 28 28/3 = 9.33	[53, 62) = 53, 56, 57	[62, 72) = 63, 66, 67, 67, 67, 68, 69, 70, 70, 70, 70	[72, 81] = 72, 73, 75, 75, 76, 76, 78, 79, 80, 81
Equal Depth	depth = 24 /3 = 8	53, 56, 57, 63, 66, 67, 67, 67	68, 69, 70, 70, 70, 70, 72, 73	75, 75, 76, 76, 78, 79, 80, 81

Discretization by Histogram Analysis

- Histogram analysis is an unsupervised discretization technique because it does not use class information.
- Histograms use binning to approximate data distributions and are a popular form of data reduction.
- A histogram for an attribute, X, partitions the data distribution of X into disjoint subsets, referred to as buckets or bins.
- If each bucket represents only a single attribute–value/frequency pair, the buckets are called singleton buckets.
- Often, buckets represent continuous ranges for the given attribute.
- The histogram analysis algorithm can be applied recursively to each partition in order to automatically generate a multilevel concept hierarchy.

Discretization by Histogram Analysis

1. Equal Width Histogram
   - The values are partitioned into equal size partitions or ranges.
2. Equal Frequency Histogram
   - The values are partitioned such that each partition contains the same number of data objects.



- **Data :** 0, 4, 12, 16, 16, 18, 24, 26, 28
- **Equal width**
  - Bin 1: 0, 4 [-,10)
  - Bin 2: 12, 16, 16, 18 [10,20)
  - Bin 3: 24, 26, 28 [20,+)
- **Equal frequency**
  - Bin 1: 0, 4, 12 [-, 14)
  - Bin 2: 16, 16, 18 [14, 21)
  - Bin 3: 24, 26, 28 [21,+)

Discretization by Clustering

- A clustering algorithm can be applied to discretize a numeric attribute, X, by partitioning the values of X into clusters or groups.
- Clustering takes the distribution of X into consideration, as well as the closeness of data points.
- Produces high-quality discretization results.

1. Top-down approach
   - Each initial cluster or partition may be further decomposed into several sub-clusters, forming a lower level of the hierarchy.
2. Bottom-up approach
   - Clusters are formed by repeatedly grouping neighboring clusters in order to form higher-level concepts

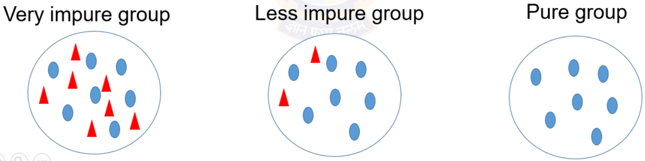
Hierarchical clustering - AGNES algorithm

Supervised Discretization

- Class labels are used.
- Entropy-based discretization
  - Entropy is the measure of the impurity /uncertainty in a group.

$$E(S) = \sum_{i=1}^{C} -p_i \log_2 p_i$$

  - Homogenous group has less entropy.
  - Heterogeneous group has more entropy.



Supervised Discretization

- Information Gain
  - Measures how much “information” a feature gives us about the class.
  - Features that perfectly partition should give maximal information.
  - Unrelated features should give no information.
  - It measures the reduction in entropy.

Supervised Discretization

- Entropy based Discretization
  1. Sort examples in increasing order.
  2. Choose a value that forms an interval. (There can be *m* intervals.)
  3. Calculate the entropy measure of this discretization. $E(S) = \sum_{i=1}^{C} -p_i \log_2 p_i$
  4. Calculate entropy for the target given a bin.

     $$E(S, F) = \sum_{\nu \in F} \frac{|S_\nu|}{|S|} E(S_\nu)$$

  5. Calculate Information Gain given a bin.

     $$I(F) = E(S) - E(S, F)$$

  6. Apply the process recursively until some stopping criterion is met.

Entropy based Discretization Example

For the given data, find out how to discretize Runs.

Runs	53	56	57	63	66	67	67	67	68	69	70	70
Won	Y	Y	Y	N	N	N	N	N	N	N	N	Y
Runs	70	70	72	73	75	75	76	76	78	79	80	81
Won	Y	Y	N	N	N	Y	N	N	N	N	N	N

Entropy based Discretization Example

- Discretize the feature Runs using 2 bins using the value 60. So two Bins ≤ 60 and > 60.
- Compute the entropy for the target.

Runs	
Y	N
7	17

$$
\begin{aligned}
E(Runs) &= E(7,17) \\
&= -\frac{7}{24}\log_2\frac{7}{24} - \frac{17}{24}\log_2\frac{17}{24} \\
&= 0.871
\end{aligned}
$$

Entropy based Discretization Example

- Compute the entropy for the target for the given bin.

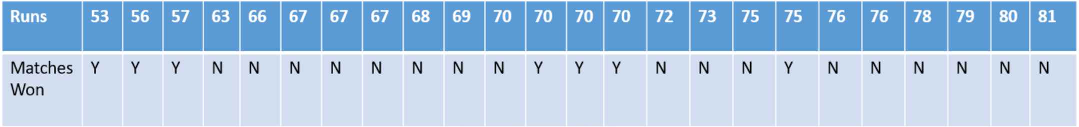
Runs	Won	
	Y	N
≤ 60	3	0
> 60	4	17

$$\begin{aligned} E(Won, Runs) &= P(\leq 60) * E(3,0) + P(> 60) * E(4,17) \\ &= \frac{3}{24} * \left(-\frac{3}{3}\log_2\frac{3}{3} - \frac{0}{3}\log_2\frac{0}{3}\right) + \frac{21}{24} * \left(-\frac{4}{21}\log_2\frac{4}{21} - \frac{17}{21}\log_2\frac{17}{21}\right) \\ &= 0.615 \end{aligned}$$

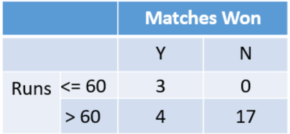
- Calculate the information gain.

$$IG(Won, Runs) = 0.87 - 0.615 = 0.256$$

Entropy based Discretization Example

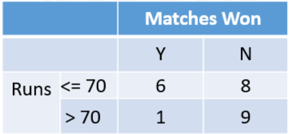


Runs	53	56	57	63	66	67	67	67	68	69	70	70	70	70	72	73	75	75	76	76	78	79	80	81
Matches Won	Y	Y	Y	N	N	N	N	N	N	N	N	Y	Y	Y	N	N	N	Y	N	N	N	N	N	N



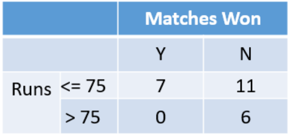
		Matches Won	
		Y	N
Runs	<= 60	3	0
	> 60	4	17

Information Gain = 0.256



		Matches Won	
		Y	N
Runs	<= 70	6	8
	> 70	1	9

Information Gain = 0.101



		Matches Won	
		Y	N
Runs	<= 75	7	11
	> 75	0	6

Information Gain = 0.148

Table of Contents

1. Data Cleaning
2. Data Aggregation
3. Data Sampling
4. Handling Numeric Data
   - Discretization
5. Data Transformation
6. Dealing with Textual Data
7. Categorical Encoding
   - Binarization

Variable Transformation

- Variable transformation involves changing the values of an attribute.
- For each object (tuple), a transformation is applied to the value of the variable for that object.
    1. Simple functional transformations
    2. Normalization

Simple Functional Transformation

- For this type of variable transformation, a simple mathematical function is applied to each value individually.
- For a variable x, simple transformations include
  - x^k, $\log x$, e^x, $\sqrt{x}$, $\frac{1}{x}$, $\sin x$, $\mid x \mid$
- In statistics, variable transformations, especially $\log x$, $\sqrt{x}$, $\frac{1}{x}$, are often used to transform data that does not have a Gaussian (normal) distribution into data that does. While this can be important, other reasons often take precedence in data mining.
- Eg: Transfer of data bytes may be represented in the the $\log_{10}$ transformation.

Simple Functional Transformation

- Variable transformations should be applied with caution since they change the nature of the data.
- For instance, the transformation $\frac{1}{x}$ reduces the magnitude of values that are 1 or larger, but increases the magnitude of values between 0 and 1.
- To understand the effect of a transformation, it is important to ask questions such as:
  - Does the order need to be maintained?
  - Does the transformation apply to all values, especially negative values and 0?
  - What is the effect of the transformation on the values between 0 and 1?

Normalization

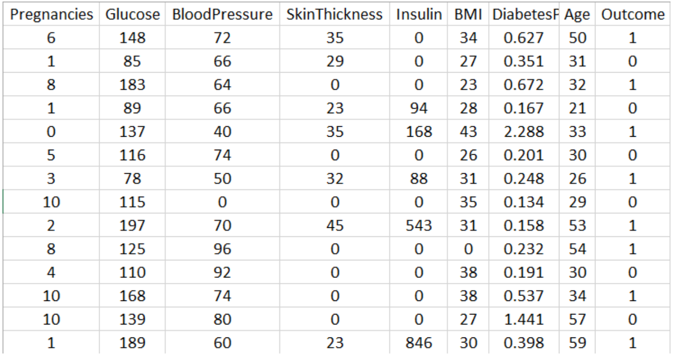
- Normalizing the data attempts to give all attributes an equal weight.
- The goal of standardization or normalization is to make an entire set of values have a particular property.
- Normalization is particularly useful for:
  - classification algorithms involving neural networks.
    - normalizing the input values for each attribute in the training tuples will help speed up the learning phase.
  - distance measurements such as nearest-neighbor classification and clustering.
    - normalization helps prevent attributes with initially large ranges (e.g., income) from outweighing attributes with initially smaller ranges (e.g., binary attributes).

Why Feature Scaling?

- Features with bigger magnitude dominate over the features with smaller magnitudes.
- Good practice to have all variables within a similar scale.
- Euclidean distances are sensitive to feature magnitude.
- Gradient descent converges faster when all the variables are in the similar scale.
- Feature scaling helps decrease the time of finding support vectors.

Why Feature Scaling?

- For distance-based methods, normalization helps prevent attributes with initially large ranges (e.g., income) from out-weighing attributes with initially smaller ranges (e.g., binary attributes).



Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesF	Age	Outcome
6	148	72	35	0	34	0.627	50	1
1	85	66	29	0	27	0.351	31	0
8	183	64	0	0	23	0.672	32	1
1	89	66	23	94	28	0.167	21	0
0	137	40	35	168	43	2.288	33	1
5	116	74	0	0	26	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35	0.134	29	0
2	197	70	45	543	31	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	38	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27	1.441	57	0
1	189	60	23	846	30	0.398	59	1

Algorithms Sensitive to Feature Magnitude

- Linear and Logistic Regression
- Neural Networks
- Support Vector Machines
- KNN
- K-Means Clustering
- Linear Discriminant Analysis (LDA)
- Principal Component Analysis (PCA)

Normalization

- Scale the feature magnitude to a standard range like $[0, 1]$ or $[-1, +1]$.
- Techniques
  - Min-Max normalization
  - z-score normalization
  - Normalization by decimal scaling
- Impact of outliers in the data ???

Min-Max Scaling

- Min-max scaling squeezes (or stretches) all feature values to be within the range of $[0, 1]$.
- Min-Max normalization preserves the relationships among the original data values.
- It will encounter an ”out-of-bounds” error if a future input case for normalization falls outside of the original data range for X.

$$\hat{x} = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad \text{for range}[0, 1]$$

$$\hat{x} = \frac{x - \min(x)}{\max(x) - \min(x)}(new_{max} - new_{min}) + new_{min} \qquad \text{for range}[new_{min}, new_{max}]$$

Min-Max Normalization

Suppose that the minimum and maximum values for the attribute income are \$12,000 and \$98,000, respectively. The new range is [0.0,1.0]. Apply min-max normalization to value of \$73,600.

$$
\begin{aligned}
\hat{x} &= \frac{x - \min(x)}{\max(x) - \min(x)}(new_{max} - new_{min}) + new_{min} \\
&= \frac{73600 - 12000}{98000 - 12000}(1.0 - 0.0) + 0.0 \\
&= 0.716
\end{aligned}
$$

z-score Normalization

- In z-score normalization (or zero-mean normalization), the values for an attribute, x, are normalized based on the mean $\mu(x)$ and standard deviation $\sigma(x)$ of x.
- The resulting scaled feature has a mean of 0 and a variance of 1.
- New range is $[-3\sigma, +3\sigma]$.

$$\hat{x} = \frac{x - \mu(x)}{\sigma(x)}$$

- z-score normalization is useful when the actual minimum and maximum of attribute X are unknown, or when there are outliers that dominate the min-max normalization.

z-score Normalization

Suppose that the mean and standard deviation of the values for the attribute income are \$54,000 and \$16,000, respectively. Apply z-score normalization to value of \$73,600.

$$\begin{aligned}\hat{x} &= \frac{x - \mu(x)}{\sigma(x)} \\ &= \frac{73600 - 54000}{16000} \\ &= 1.225\end{aligned}$$

Decimal Normalization

- Normalizes by moving the decimal point of values of attribute x.
- The number of decimal points moved depends on the maximum absolute value of x.
- New range is $[-1, +1]$.

$$\begin{aligned} j &= \text{smallest integer such that} \max(|\hat{x}|) < 1 \\ \hat{x} &= \frac{x}{10^j} \end{aligned}$$

Decimal Normalization

Example 1		
CGPA	Formula	Normalized CGPA
2	2/10	0.2
3	3/10	0.3
Example 2		
Bonus	Formula	Normalized Bonus
450	450/1000	0.45
310	310/1000	0.31
Example 3		
Salary	Formula	Normalized Salary
48000	48000/100000	0.48
67000	67000/100000	0.67

Table of Contents

1. Data Cleaning
2. Data Aggregation
3. Data Sampling
4. Handling Numeric Data
   - Discretization
5. Data Transformation
6. Dealing with Textual Data
7. Categorical Encoding
   - Binarization

Text Analysis

- Text analytics is driven by the need to process natural human language.
- Unlike numeric or categorical data, natural language does not exist in a structured format consisting of rows (of examples) and columns (of attributes).
- Text mining is, therefore, the domain of unstructured data science.
- Text mining refers to the process of separating valuable keywords from a mass of other words (or relevant documents) and use them to identify meaningful patterns or make predictions.

Text Analysis

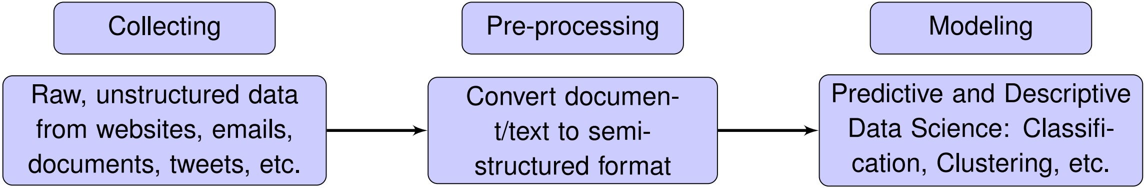
Text mining has many applications

- Entity identification
- Plagiarism detection
- Topic identification
- Text clustering
- Translation
- Automatic text summarization
- Fraud detection
- Spam filtering
- Sentiment analysis

Text Analysis

How Text mining works?

- The basic step in text mining involves converting unstructured text into semi-structured data.
- Models can be trained on the semi-structured data to detect patterns in the unseen text.



Collecting

Raw, unstructured data from websites, emails, documents, tweets, etc.

Pre-processing

Convert document/text to semi-structured format

Modeling

Predictive and Descriptive Data Science: Classification, Clustering, etc.

Term Frequency –Inverse Document Frequency

- Consider a web search problem where the user types in some keywords and the search engine extracts all the documents (essentially, web pages) that contain these keywords.
- How does the search engine know which web pages to serve up?
- In addition to using network rank or page rank, the search engine also runs some form of text mining to identify the most relevant web pages.
  - Example, the user types in the following keywords: ”RapidMiner books that describe text mining.”
- In this case, the search engines run on the following basic logic:
  - Give a high weight-age to those keywords that are relatively rare.
  - Give a high weight-age to those web pages that contain a large number of instances of the rare keywords.

Term Frequency –Inverse Document Frequency

- Term Frequency (TF)
  - Refers to the ratio of the number of times a keyword appears in a given document, n_k (where k is the keyword), to the total number of terms in the document, n.

$$TF = \frac{n_k}{n}$$

- Inverse Document Frequency (IDF)
  - Refers to the ratio of the total number of documents N, to the number of documents that contain the keyword k, N_k.

$$IDF = \log_2 \frac{N}{N_k}$$

- Term Frequency-Inverse Document Frequency (TF-IDF)

$$TF - IDF = \frac{n_k}{n} \times \log_2 \frac{N}{N_k}$$

Steps involved in the Textual Data Processing

- Removing special characters, changing the case (up-casing and down-casing).
- Tokenization — process of discretizing words within a document.
- Creating Document Vector or Term Document Matrix.
- Filtering Stop Words.
- Term Filtering.
- Stemming / Lemmatization.
- Forming n-grams and storing them in the document. vector

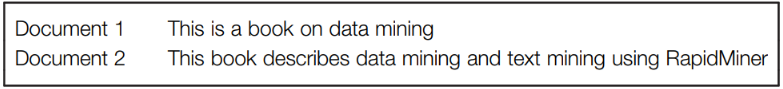
TOKENIZATION

- Document – In the text mining context, each sentence is considered a distinct document.
- Token – Each word is called a token.
- Tokenization – The process of discretizing words within a document is called tokenization.

Document Vector or Term Document Matrix

- Create a matrix where each column consists of a token and the cells show the counts of the number of times a token appears.
- Each token is now an attribute in standard data science parlance and each document is an example (record).
- Unstructured raw data is now transformed into a format that is recognized by machine learning algorithms for training.
- The matrix / table is referred to as Document Vector or Term Document Matrix (TDM)
- As more new statements are added that have little in common, we end up with a very sparse matrix.
- We could also choose to use the term frequencies (TF) for each token instead of simply counting the number of occurrences.

Term Document Matrix – Example



Document 1	This is a book on data mining
Document 2	This book describes data mining and text mining using RapidMiner

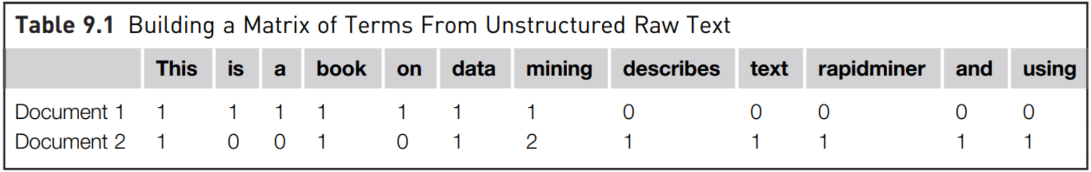


Table 9.1 Building a Matrix of Terms From Unstructured Raw Text

	This	is	a	book	on	data	mining	describes	text	rapidminer	and	using
Document 1	1	1	1	1	1	1	1	0	0	0	0	0
Document 2	1	0	0	1	0	1	2	1	1	1	1	1

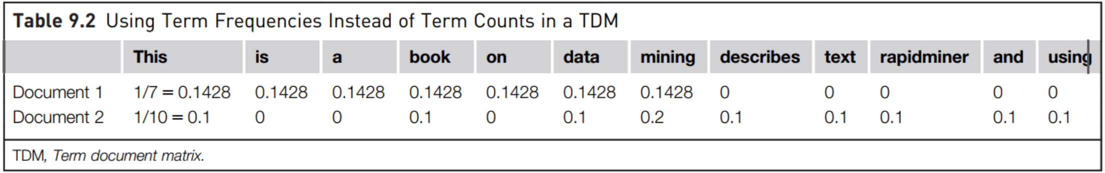


Table 9.2 Using Term Frequencies Instead of Term Counts in a TDM

	This	is	a	book	on	data	mining	describes	text	rapidminer	and	using
Document 1	1/7 = 0.1428	0.1428	0.1428	0.1428	0.1428	0.1428	0.1428	0	0	0	0	0
Document 2	1/10 = 0.1	0	0	0.1	0	0.1	0.2	0.1	0.1	0.1	0.1	0.1

TDM, *Term document matrix.*

Stop Words

- There are common words such as ”a,” ”this,” ”and,” and other similar terms.
- They do not really convey specific meaning.
- Most parts of speech such as articles, conjunctions, prepositions, and pronouns need to be filtered before additional analysis is performed.
- Such terms are called **stop words**.
- Stop word filtering is usually the second step that follows immediately after tokenization.
- The document vector gets reduced significantly after applying standard English stop word filtering.

Stop Words

- Domain specific terms might also need to be filtered out.
    - For example, if we are analyzing text related to the automotive industry, we may want to filter out terms common to this industry such as ”car,” ”automobile,” ”vehicle,” and so on.
- This is generally achieved by creating a separate dictionary where these context specific terms can be defined and then term filtering can be applied to remove them from the data.

Lexical Substitution

- **Lexical substitution** is the process of finding an alternative for a word in the context of a clause.
- It is used to align all the terms to the same term based on the field or subject which is being analyzed.
- This is especially important in areas with specific jargon, e.g., in clinical settings.

Stemming

- Stemming is usually the next process step following term filtering.
- Words such as ”recognized,” ”recognizable,” or ”recognition” may be encountered in different usages, but contextually they may all imply the same meaning.
- The root of all these highlighted words is ”recognize.”
- The conversion of unstructured text to structured data can be simplified by reducing terms in a document to their basic stems, because only the occurrence of the root terms has to be taken into account.
- This process is called stemming.

Stemming

- The most common stemming technique for text mining in English is the Porter method.
- Porter stemming works on a set of rules where the basic idea is to remove and/or replace the suffix of words.
  - Replace all terms which end in 'ies' by 'y,' such as replacing the term "anomalies" with "anomaly."
  - Stem all terms ending in "s" by removing the "s," as in "algorithms" to "algorithm."
- While the Porter stemmer is extremely efficient, it can make mistakes that could prove costly.
  - "arms" and "army" would both be stemmed to "arm," which would result in somewhat different contextual meanings.

LEMMATIZATION

- Lemmatization convert a word to its root form, in a more grammatically sensitive way.
    - While both stemming and lemmatization would reduce ”cars” to ”car,” lemmatization can also bring back conjugated verbs to their unconjugated forms such as ”are” to ”be.”
- Lemmatization uses POS Tagging (Part of Speech Tagging) heavily.
- POS Tagging is the process of attributing a grammatical label to every part of a sentence.
    - Eg: ”Game of Thrones is a television series.”
    - POS Tagging:
      ({”game”:”NN”},{”of”:”IN”},{”thrones”:”NNS”},{”is”:”VBZ”},{”a”:”DT”}, {”television”:”NN”},{”series”:”NN”})
      where: NN = noun, IN = preposition, NNS = noun in its plural form, VBZ = third-person singular verb, and DT = determiner.
- Combining POS Tagging and lemmatization is likely to give cleaner data than using only a stemmer.

N-GRAMS

- There are families of words in the spoken and written language that typically go together. Grouping such terms, called **n-grams**, and analyzing them statistically can present new insights.
- The final pre-processing step typically involves forming these n-grams and storing them in the document vector.
- Algorithms providing n-grams become computationally expensive and the results become huge so in practice the amount of ”n” will vary based on the size of the documents and the corpus.

Table of Contents

1. Data Cleaning
2. Data Aggregation
3. Data Sampling
4. Handling Numeric Data
   - Discretization
5. Data Transformation
6. Dealing with Textual Data
7. Categorical Encoding
   - Binarization

Binarization

- Binarization maps a continuous or categorical attribute into one or more binary attributes.
- Must maintain ordinal relationship.
- Algorithms that find association patterns require that the data be in the form of binary attributes.
  E.g., Apriori algorithm, Frequent Pattern (FP) Growth algorithm, Rapid Association Rule Mining (RARM)

Binarization Techniques

- One-hot encoding
- Label Encoding
- Ordinal Encoding
- Binary Encoding
- Count or Frequency Encoding
- Mean Encoding
- Hashing Encoding

One-Hot Encoding

- Binary attributes, where only the presence of 1 is important.
- Encode each categorical variable with a set of Boolean variables which take values 0 or 1, indicating if a category is present for each observation.
- One binary attribute for each categorical value.
- Advantages
  - Makes no assumption about the distribution or categories of the categorical variable .
  - Keeps all the information of the categorical variable .
  - Suitable for linear models.
- Disadvantages
  - Expands the feature space.
  - Does not add extra information while encoding.
  - Many dummy variables may be identical, introducing redundant information .
  - Number of resulting attributes may become too large.

One-Hot Encoding

- If a categorical variable can take on k values, it is tempting to define k dummy variables.
- A kth dummy variable is redundant; it carries no new information.
- **Using k dummy variables when only $k-1$ dummy variables are required is known as the dummy variable trap. Avoid this trap!**

One-Hot Encoding Example

- Assume an ordinal attribute for representing service of a restaurant: (*Awful* < *Poor* < *OK* < *Good* < *Great*) requires 5 bits to maintain the ordinal relationship.

Service Quality	X1	X2	X3	X4	X5
Awful	0	0	0	0	1
Poor	0	0	0	1	0
OK	0	0	1	0	0
Good	0	1	0	0	0
Great	1	0	0	0	0

Label Encoding

- Replace the categories by digits from 1 to *n* (or 0 to $n - 1$, depending the implementation), where *n* is the number of distinct categories of the variable.
- The numbers are assigned arbitrarily.
- Allows for quick benchmarking of machine learning models.
- Advantages
  - Straightforward to implement.
  - Does not expand the feature space.
  - Work well enough with tree based algorithms.
- Disadvantages
  - Does not add extra information while encoding.
  - Not suitable for linear models.
  - Does not handle new categories in test set automatically.

Label Encoding Example

- Assume an ordinal attribute for representing service of a restaurant: (Awful, Poor, OK, Good, Great)

Service Quality	Integer Value
Awful	0
Poor	1
OK	2
Good	3
Great	4

Frequency Encoding

- Categories are replaced by the count or percentage of observations of each category.
- Assumption: the number observations shown by each category is predictive of the target.
- Advantages
    - Straightforward to implement.
    - Does not expand the feature space.
    - Work well enough with tree based algorithms.
- Disadvantages
    - Not suitable for linear models.
    - Does not handle new categories in test set automatically.
    - If two different categories appear the same amount of times in the dataset, that is, they appear in the same number of observations, they will be replaced by the same number: may lose valuable information.

- Introduction to Data Mining, by Tan, Steinbach and Vipin Kumar (T3)
- The Art of Data Science by Roger D Peng and Elizabeth Matsui (R1)
- Introducing Data Science by Cielen, Meysman and Ali
- Data Science - Concepts and Practice by Vijay Kotuand BalaDeshpande
- Data mining: Concepts and techniques, by Han, J., Kamber, M., and Pei, J. (2012). (3rd ed.)

THANK YOU