Introduction to Data Science

Dr. Irfan Yousuf
Department of Computer Science (New Campus)
UET, Lahore

(Week 12: April 01 - 05, 2024)

Outline

- Spam Filtering
- Data Preprocessing

Spam Filtering

- Spam filters detect unsolicited, unwanted, and virus-infected email (called spam) and stop it from getting into email inboxes.

- Internet Service Providers (ISPs) use spam filters to make sure they aren’t distributing spam.

- Small- to medium- sized businesses (SMBs) also use spam filters to protect their employees and networks.

How do Spam Filters Work

- Spam filters use “heuristics” methods, which means that each email message is subjected to thousands of predefined rules (algorithms).

- Each rule assigns a numerical score to the probability of the message being spam, and if the score passes a certain threshold the email is flagged as spam and blocked from going further.

Types of Spam Filters

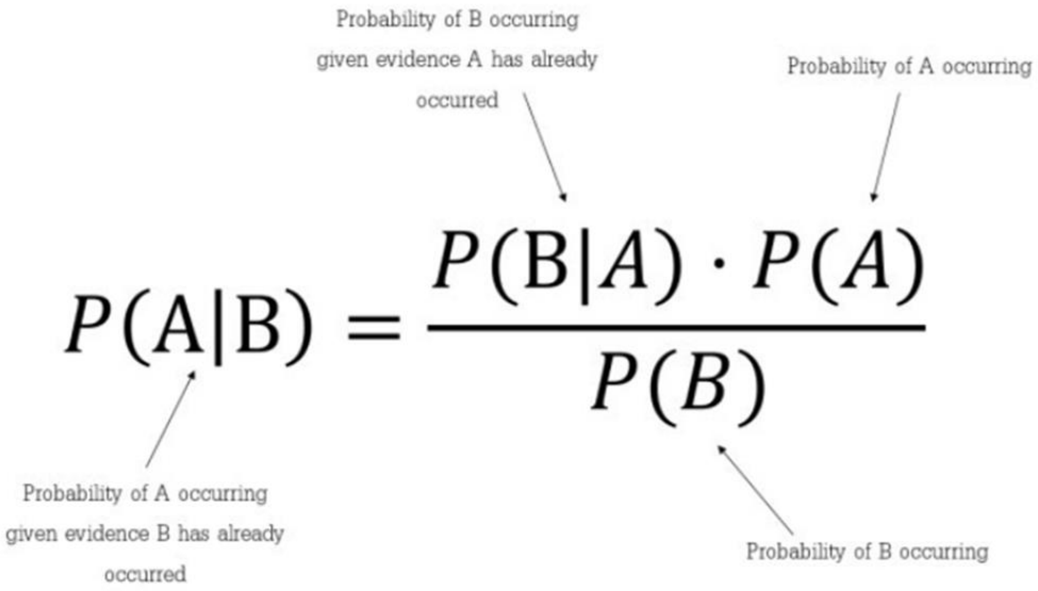
- **Content filters:** parse the content of messages, scanning for words that are commonly used in spam emails.
- **Header filters:** examine the email header source to look for suspicious information (such as spammer email addresses).
- **Blocklist filters:** stop emails that come from a blocklist of suspicious IP addresses. Some filters go further and check the IP reputation of the IP address.
- **Rules-based filters:** apply customized rules designed by the organization to exclude emails from specific senders, or emails containing specific words in their subject line or body.

Types of Spam Filters

- **Bayesian Filter:** A Bayesian filter is a filter that learns our spam preferences. When we mark emails as spam, the system will note the characteristics of the email and look for similar characteristics in incoming email, filtering anything that fits the formula directly in to spam for us.

- A Bayesian filter is one of the most intelligent types of spam filters because it is **able to learn** and adapt on its own.

How Bayesian Filter Works?



$$P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{B}|A) \cdot P(A)}{P(B)}$$

Probability of B occurring given evidence A has already occurred

Probability of A occurring

Probability of A occurring given evidence B has already occurred

Probability of B occurring

$$P_r(spam|word) = \frac{P_r(word|spam)P_r(spam)}{P_r(word)}$$

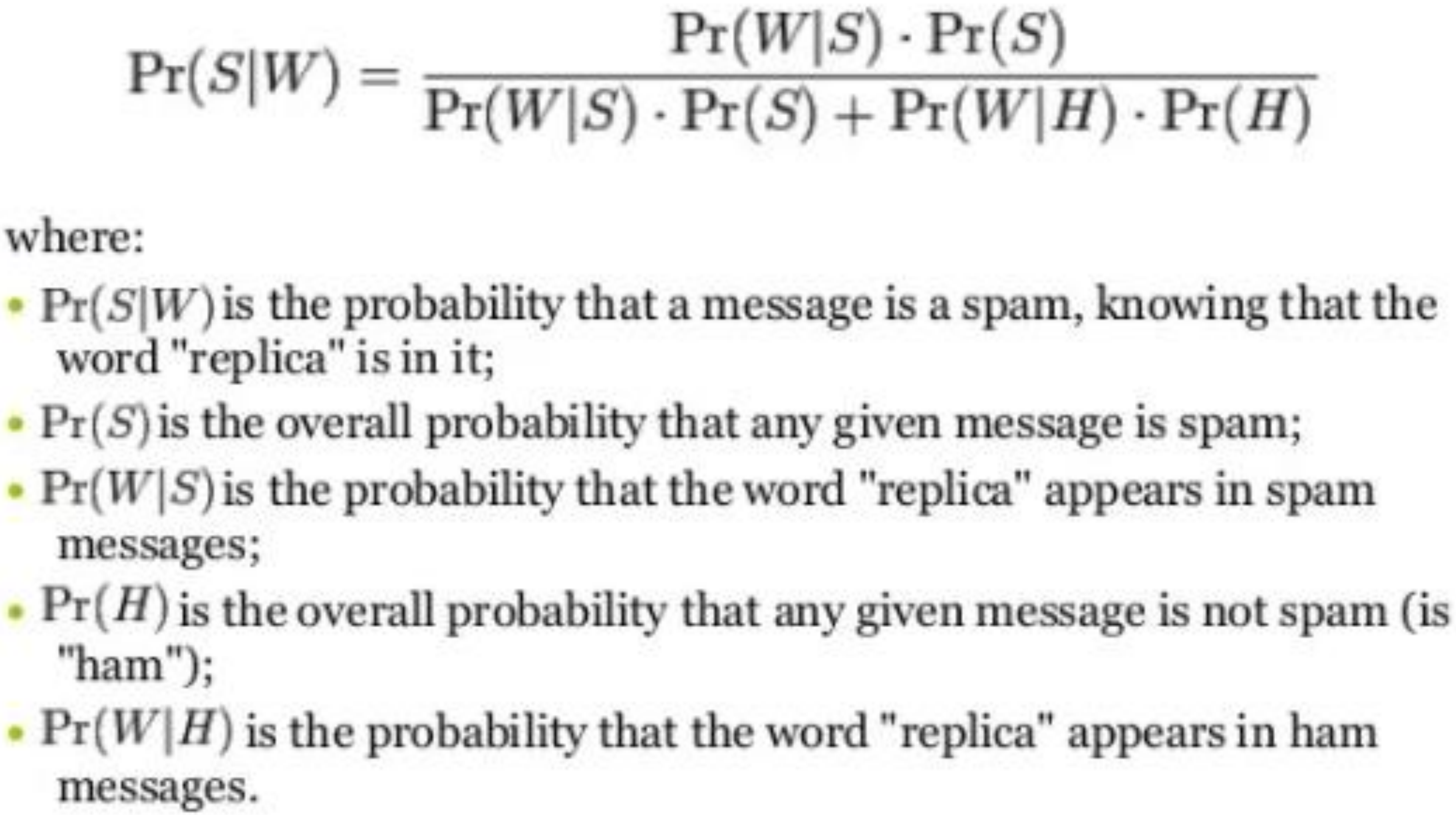
How Bayesian Filter Works?

Ham: E-mail that is generally desired and isn't considered spam. Better to use ‘non-Spam’.

$$P(s|w) = \frac{P(w|s)P(s)}{P(w|s \cup h)}$$

The probability that an email is a spam message if a word *w* occurs is defined by the probability that word *w* is in a spam message *s* multiplied by the general probability that the email is a spam message *s*. This gets divided by the probability of that word occurring in an e-mail (spam and ham combined)

How Bayesian Filter Works?



$$\Pr(S|W) = \frac{\Pr(W|S)\cdot\Pr(S)}{\Pr(W|S)\cdot\Pr(S) + \Pr(W|H)\cdot\Pr(H)}$$

where:

- $\Pr(S|W)$ is the probability that a message is a spam, knowing that the word "replica" is in it;
- $\Pr(S)$ is the overall probability that any given message is spam;
- $\Pr(W|S)$ is the probability that the word "replica" appears in spam messages;
- $\Pr(H)$ is the overall probability that any given message is not spam (is "ham");
- $\Pr(W|H)$ is the probability that the word "replica" appears in ham messages.

Spam vs. Ham Emails?

$$P = \frac{p_1 p_2 \ldots p_n}{p_1 p_2 \ldots p_n + (1 - p_1)(1 - p_2) \ldots (1 - p_n)}$$

We can achieve this by multiplying the probabilities for every word together and dividing by the **combined probability** of every word for being in a spam message plus the probability of every word for not being in a spam message.

Solved Example

Assume that we have the following set of email classified as spam or ham.

spam:	“send us your password”
ham:	“send us your review”
ham:	“password review”
spam:	“review us ”
spam:	“send your password”
spam:	“send us your account”

We are interested in classifying the following new email as spam or ham:

new email	“review us now”

Solved Example

spam:	“send us your password”
ham:	“send us your review”
ham:	“password review”
spam:	“review us ”
spam:	“send your password”
spam:	“send us your account”

Prior probabilities are:

$$\Pr(\text{spam}) = \frac{4}{6} \qquad \Pr(\text{ham}) = \frac{2}{6}$$

Solved Example

spam: “send us your password”

ham: “send us your review”

ham: “password review”

spam: “review us ”

spam: “send your password”

spam: “send us your account”

	$\Pr(\cdot \mid \text{spam})$	$\Pr(\cdot \mid \text{ham})$
review	1/4	2/2
send	3/4	1/2
us	3/4	1/2
your	3/4	1/2
password	2/4	1/2
account	1/4	0/2

Solved Example

spam:	“send us your password”
ham:	“send us your review”
ham:	“password review”
spam:	“review us ”
spam:	“send your password”
spam:	“send us your account”

	$\Pr(\cdot \mid \text{spam})$	$\Pr(\cdot \mid \text{ham})$
review	1/4	2/2
send	3/4	1/2
us	3/4	1/2
your	3/4	1/2
password	2/4	1/2
account	1/4	0/2

$$\Pr(\text{spam} \mid \text{review}) = \frac{\Pr(\text{review}|\text{spam})\Pr(\text{spam})}{\Pr(\text{review}|\text{spam})\Pr(\text{spam})+\Pr(\text{review}|\text{ham})\Pr(\text{ham})} = \frac{\frac{1}{4}\cdot\frac{4}{6}}{\frac{1}{4}\cdot\frac{4}{6}+\frac{2}{2}\cdot\frac{2}{6}} = \frac{1}{3}$$

Solved Example

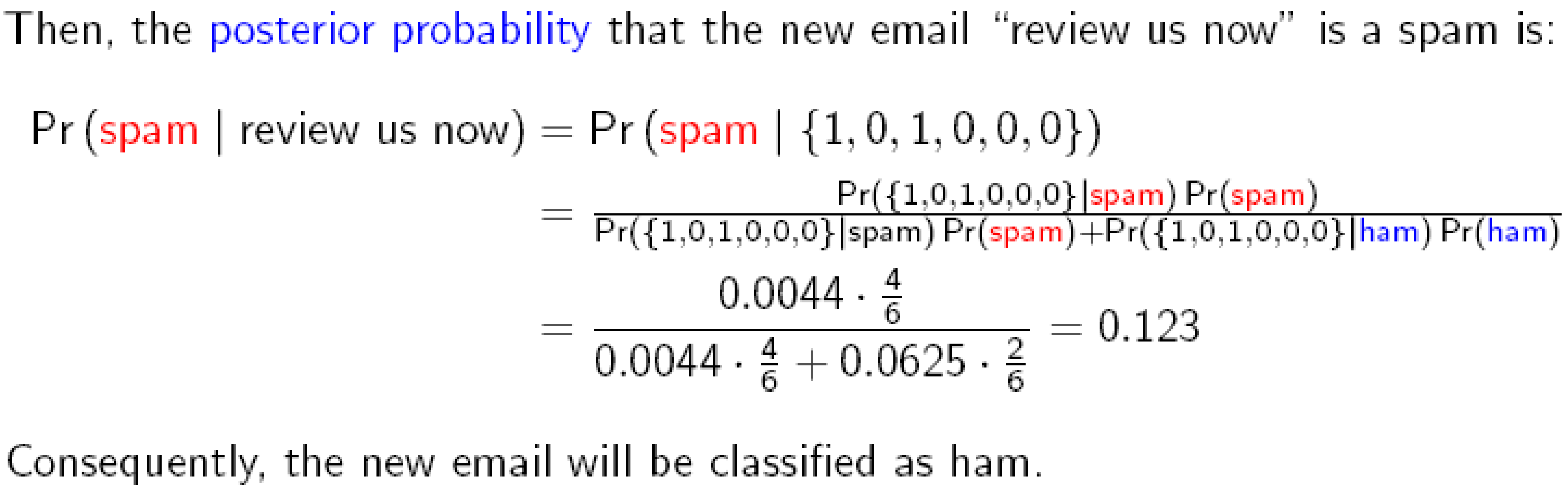
	$\Pr(\cdot \mid \text{spam})$	$\Pr(\cdot \mid \text{ham})$
review	$1/4$	$2/2$
send	$3/4$	$1/2$
us	$3/4$	$1/2$
your	$3/4$	$1/2$
password	$2/4$	$1/2$
account	$1/4$	$0/2$

- Assuming that the words in each message are independent events:

$$\begin{aligned}\Pr(\text{review us now} \mid \text{spam}) &= \Pr(\{1,0,1,0,0,0\} \mid \text{spam}) \\ &= \frac{1}{4}\left(1-\frac{3}{4}\right)\frac{3}{4}\left(1-\frac{3}{4}\right)\left(1-\frac{2}{4}\right)\left(1-\frac{1}{4}\right) = 0.0044\end{aligned}$$

$$\begin{aligned}\Pr(\text{review us now} \mid \text{ham}) &= \Pr(\{1,0,1,0,0,0\} \mid \text{ham}) \\ &= \frac{2}{2}\left(1-\frac{1}{2}\right)\frac{1}{2}\left(1-\frac{1}{2}\right)\left(1-\frac{1}{2}\right)\left(1-\frac{0}{4}\right) = 0.0625\end{aligned}$$

Solved Example



Then, the posterior probability that the new email "review us now" is a spam is:

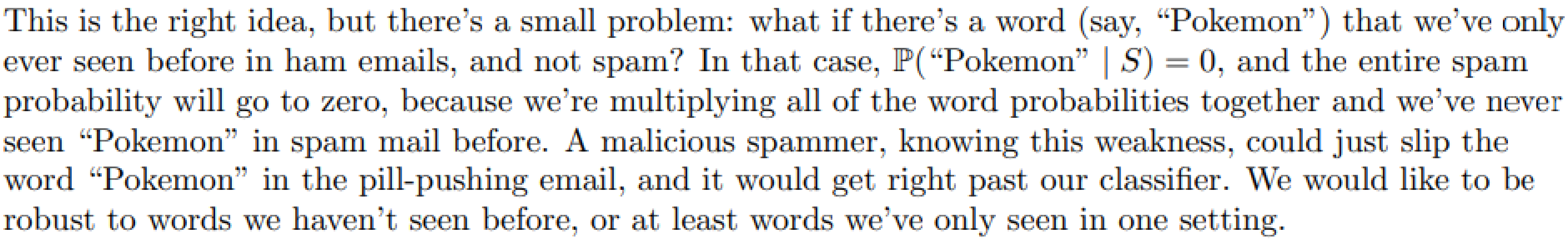
$$\begin{aligned}
\Pr(\text{spam} \mid \text{review us now}) &= \Pr(\text{spam} \mid \{1,0,1,0,0,0\}) \\
&= \frac{\Pr(\{1,0,1,0,0,0\}|\text{spam})\Pr(\text{spam})}{\Pr(\{1,0,1,0,0,0\}|\text{spam})\Pr(\text{spam})+\Pr(\{1,0,1,0,0,0\}|\text{ham})\Pr(\text{ham})} \\
&= \frac{0.0044 \cdot \frac{4}{6}}{0.0044 \cdot \frac{4}{6} + 0.0625 \cdot \frac{2}{6}} = 0.123
\end{aligned}$$

Consequently, the new email will be classified as ham.

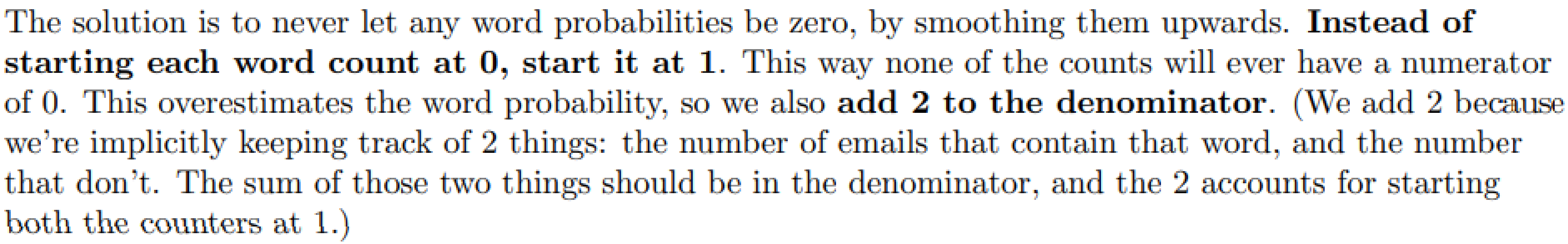
Working of a Spam Filter

- Filter filler words and special characters out of e-mail
- Split data into test and train set
- Calculate the total probability of an e-mail being spam or ham ($P(s)$ and $P(\mathrm{h})$)
- Calculate the conditional probability of a word occurring in a spam ($P(w|s)$)
- Calculate the total probability of P (spamminess) for every email in the train set.
- Search for best threshold
- Test on test set
- Evaluate results

Laplace smoothing



This is the right idea, but there's a small problem: what if there's a word (say, "Pokemon") that we've only ever seen before in ham emails, and not spam? In that case, $\mathbb{P}(\text{"Pokemon"} \mid S) = 0$, and the entire spam probability will go to zero, because we're multiplying all of the word probabilities together and we've never seen "Pokemon" in spam mail before. A malicious spammer, knowing this weakness, could just slip the word "Pokemon" in the pill-pushing email, and it would get right past our classifier. We would like to be robust to words we haven't seen before, or at least words we've only seen in one setting.



The solution is to never let any word probabilities be zero, by smoothing them upwards. **Instead of starting each word count at 0, start it at 1**. This way none of the counts will ever have a numerator of 0. This overestimates the word probability, so we also **add 2 to the denominator**. (We add 2 because we're implicitly keeping track of 2 things: the number of emails that contain that word, and the number that don't. The sum of those two things should be in the denominator, and the 2 accounts for starting both the counters at 1.)

Semester Project / Assignment

Spam Filter

Implement Spam Filter in Python

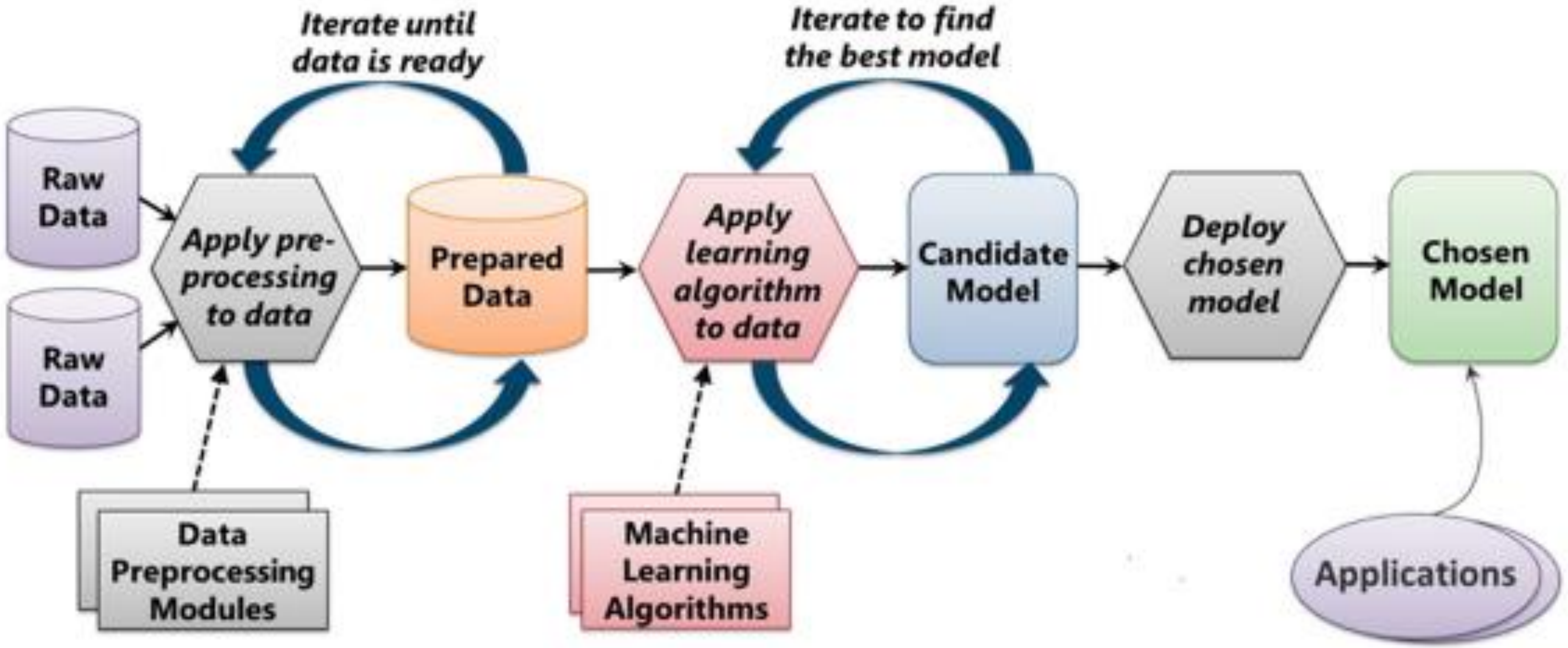
Data Preprocessing

Data Preprocessing

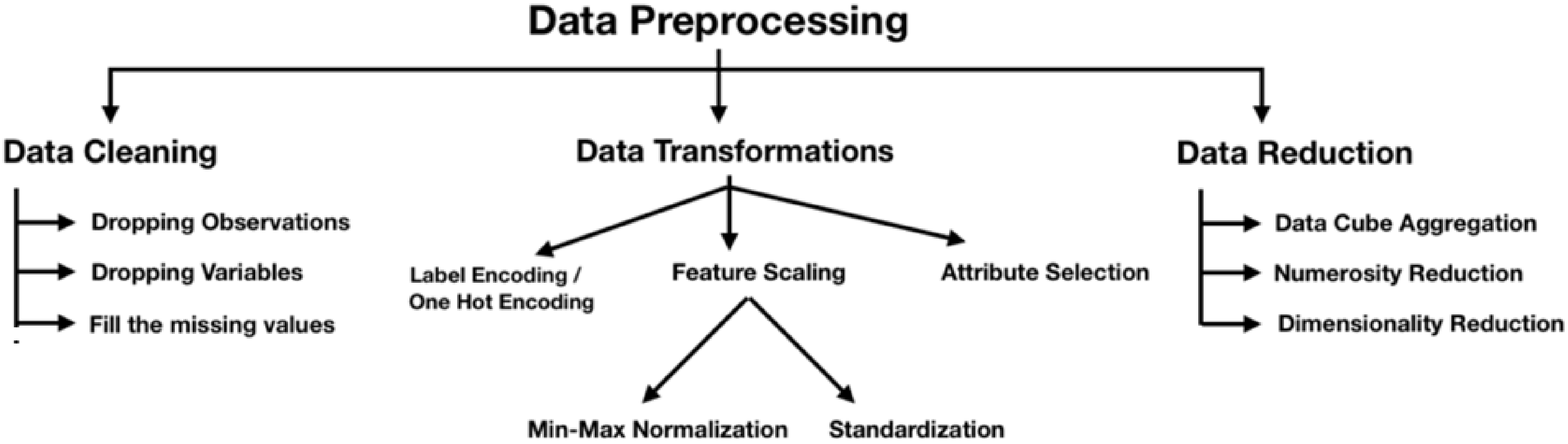
- Data preprocessing is a data mining technique that involves transforming raw data into an understandable format.

- Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors

Data Preprocessing


Iterate until data is ready
Iterate to find the best model
Raw Data
Raw Data
Apply pre-processing to data
Prepared Data
Apply learning algorithm to data
Candidate Model
Deploy chosen model
Chosen Model
Data Preprocessing Modules
Machine Learning Algorithms
Applications

Data Preprocessing



Data Cleaning

- Data cleaning, data cleansing, or data scrubbing is the act of first identifying any issues or bad data, then systematically correcting these issues.

- If the data is unfixable, you will need to remove the bad elements to properly clean your data.

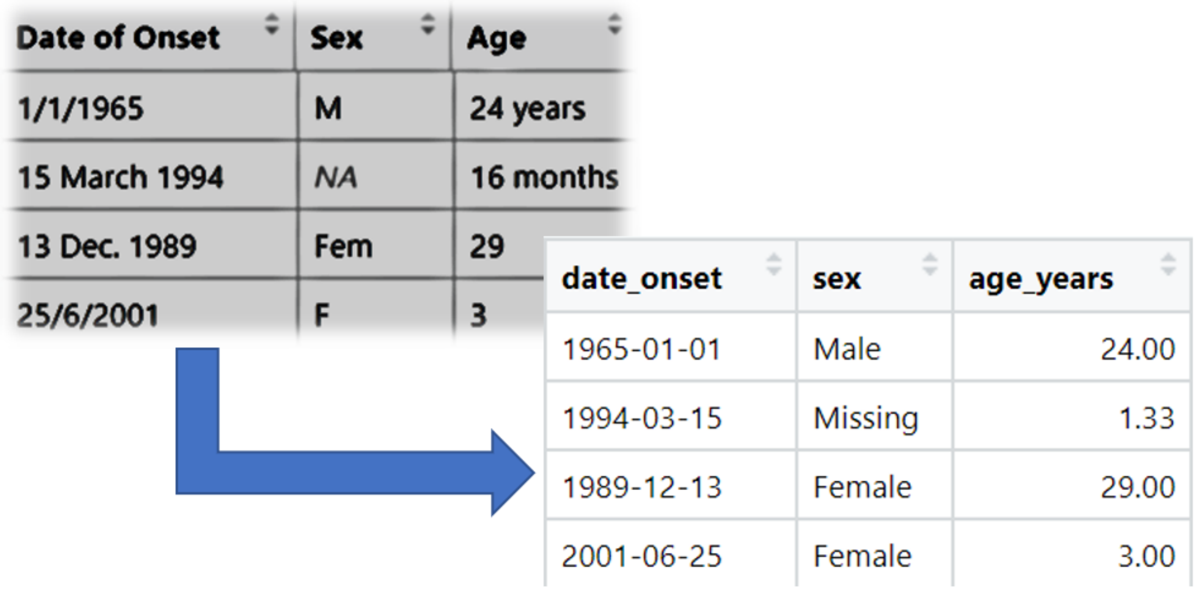
- Unclean data normally comes as a result of human error, scraping data, or combining data from multiple sources.

Data Cleaning Techniques

- Note that each case and data set will require different data cleaning methods.
  - Remove duplicates
  - Remove irrelevant data
  - Standardize capitalization
  - Convert data type
  - Handle missing values

Remove Duplicates

- When you collect your data from a range of different places, or scrape your data, it’s likely that you will have duplicated entries.
- Duplicates will inevitably skew your data and/or confuse your results.



Date of Onset	Sex	Age
1/1/1965	M	24 years
15 March 1994	*NA*	16 months
13 Dec. 1989	Fem	29
25/6/2001	F	3

date_onset	sex	age_years
1965-01-01	Male	24.00
1994-03-15	Missing	1.33
1989-12-13	Female	29.00
2001-06-25	Female	3.00

Remove Irrelevant Data

- Irrelevant data will slow down and confuse any analysis that you want to do.

- For instance, if you are analyzing the age range of your customers, you don’t need to include their email addresses.

Standardize capitalization

- Within your data, you need to make sure that the text is consistent.

- If you have a mixture of capitalization, this could lead to different erroneous categories being created.

Convert Data Types

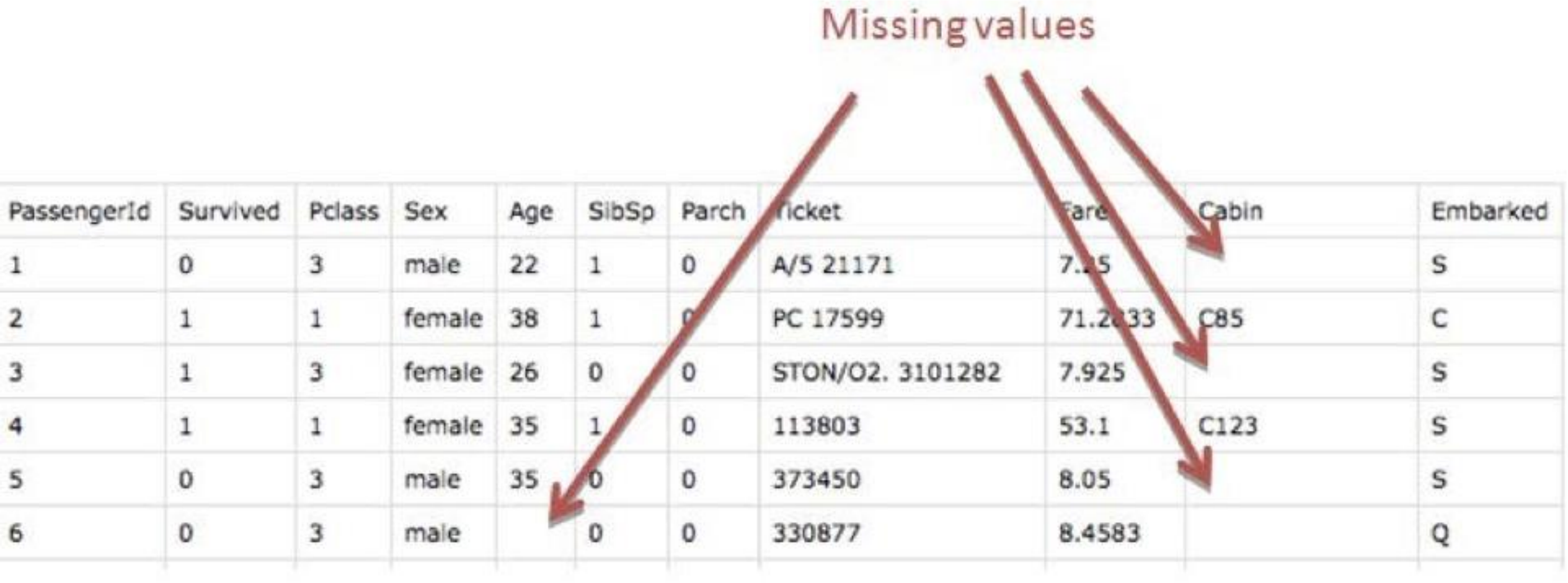
- Numbers are the most common data type that you will need to convert when cleaning your data.

- Often numbers are imputed as text, however, in order to be processed, they need to appear as numerals.

- For example, if you have an entry that reads **September 24th 2021**, you’ll need to change that to read **09/24/2021**.

Handle Missing Values

- Missing data is defined as the values or data that is not stored (or not present) for some variable/s in the given dataset.

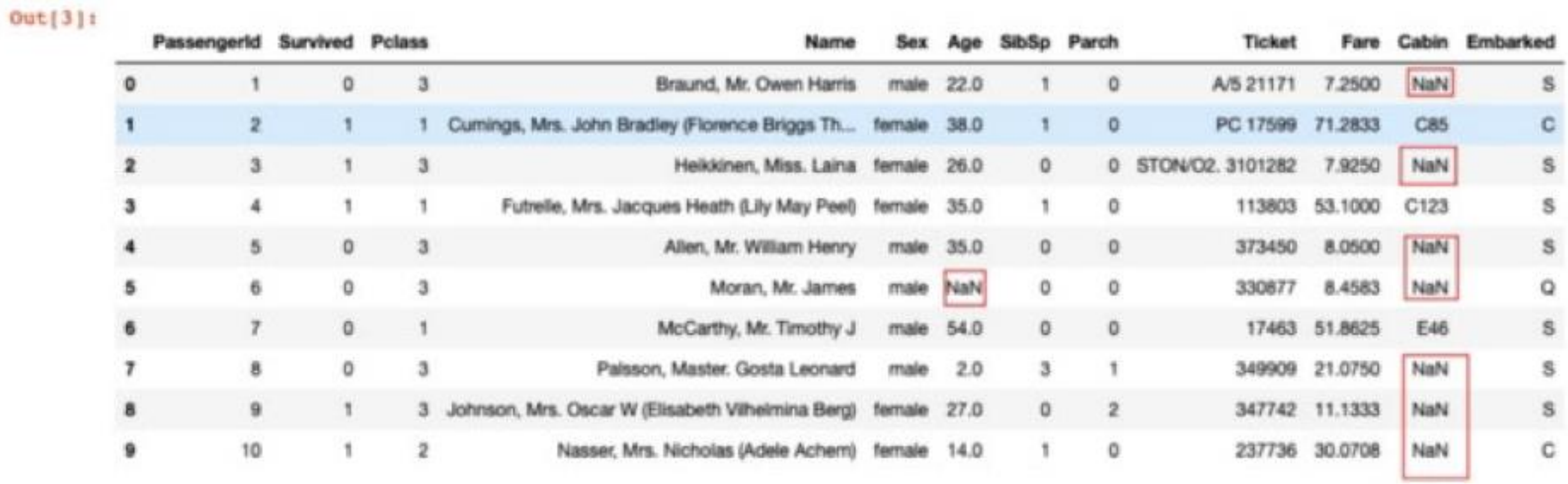


Missing values

PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	male	22	1	0	A/5 21171	7.25		S
2	1	1	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	female	35	1	0	113803	53.1	C123	S
5	0	3	male	35	0	0	373450	8.05		S
6	0	3	male		0	0	330877	8.4583		Q

Handle Missing Values

- In Pandas, usually, missing values are represented by NaN.



Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C

Types Of Missing Value

- Missing Completely At Random (MCAR).
- Missing At Random (MAR)
- Missing Not At Random (MNAR)

Missing Completely At Random (MCAR)

- In MCAR, the probability of **data being missing is the same for all the observations**.
- In this case, there is no relationship between the missing data and any other values observed or unobserved (the data which is not recorded) within the given dataset.
- That is, missing values are completely independent of other data. There is no pattern.
- In the case of MCAR, the data could be missing due to human error, some system/equipment failure, loss of sample, or some unsatisfactory technicalities while recording the values.

Missing At Random (MAR)

- Missing at random (MAR) means that the reason for missing values **can be explained by variables on which you have complete information** as there is some relationship between the missing data and other values/data.
- In this case, the data is not missing for all the observations. It is missing only within sub-samples of the data and there is some pattern in the missing values.
- For example, if you check the survey data, you may find that ‘Age’ values are mostly missing for a certain age group.

Missing Not At Random (MNAR)

- Missing values depend **on the unobserved data**.
- If there is some structure/pattern in missing data and other observed data **can not explain it**, then it is Missing Not At Random (MNAR).
- It can happen due to the reluctance of people in providing the required information. A specific group of people may not answer some questions in a survey.
- For example, suppose the name and the number of overdue books are asked in the poll for a library. So most of the people having no overdue books are likely to answer the poll. People having more overdue books are less likely to answer the poll.

Why Care About Handling Missing Value?

- Many machine learning algorithms fail if the dataset contains missing values. However, algorithms like K-nearest and Naive Bayes support data with missing values.

- You may end up building a biased machine learning model which will lead to incorrect results if the missing values are not handled properly.

- Missing data can lead to a lack of precision in the statistical analysis.

Handling Missing Value

- There are 2 primary ways of handling missing values:
    - Deleting the Missing values
    - Imputing the Missing Values

Deleting the Missing value

- Generally, this approach is not recommended. It is one of the quick but dirty techniques one can use to deal with missing values.
- If the missing value is of the type Missing Not At Random (MNAR), then it should not be deleted.
- If the missing value is of type Missing At Random (MAR) or Missing Completely At Random (MCAR) then it can be deleted.
- The disadvantage of this method is one might end up deleting some useful data from the dataset.
- Delete Entire Row
- Delete Entire Column

Imputing the Missing Value

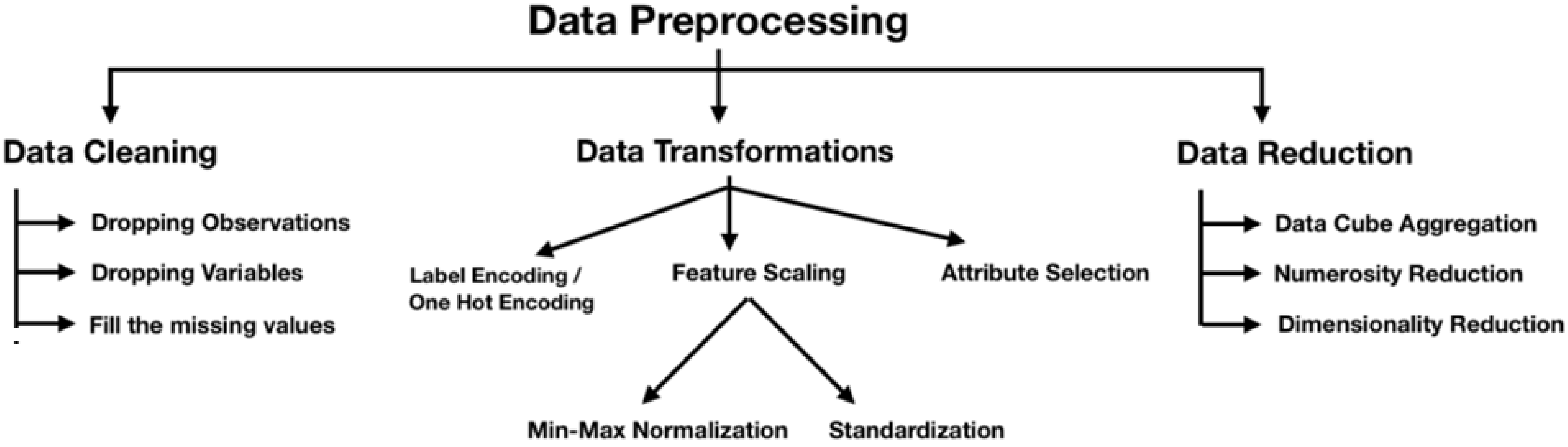
- Replacing With Arbitrary Value
- Replacing With Mean
- Replacing With Mode (for categorical data)
- Replacing With Median (data with outliers)
- Replacing with previous value – Forward fill (time series data.)
- Replacing with next value – Backward fill
- Interpolation

Adding missing indicator to encode “missingness” as a feature

- In some cases, while imputing missing values, you can preserve information about which values were missing and use that as a feature.
- Because sometimes there may be a relationship between the reason for missing values (also called the “missingness”) and the target variable you are trying to predict.

- Suppose you are predicting the presence of a disease and you can imagine a scenario in which a missing age is a good predictor of a disease because assume that we don’t have records for aged people.

Data Preprocessing



Data Preprocessing

Data Cleaning
- Dropping Observations
- Dropping Variables
- Fill the missing values

Data Transformations
- Label Encoding / One Hot Encoding
- Feature Scaling
  - Min-Max Normalization
  - Standardization
- Attribute Selection

Data Reduction
- Data Cube Aggregation
- Numerosity Reduction
- Dimensionality Reduction

Data Transformation

- It is a technique by which we can boost our model performance.

- Feature transformation is a mathematical transformation in which we apply a mathematical formula to a particular feature and transform the values which are useful for our further analysis.

Handling Categorical Data in Machine Learning

- Machine learning models require all input and output variables to be **numeric**.

- This means that if your data contains **categorical data (Text / String)**, you must **encode it to numbers** before you can fit and evaluate a model.

- The two most popular techniques are:
  - Label Encoding
  - One-Hot Encoding.

Handling Categorical Data in Machine Learning

- Categorical variables are often called nominal.

- **Nominal Variable (Categorical).** Variable comprises a finite set of discrete values with no relationship between values.

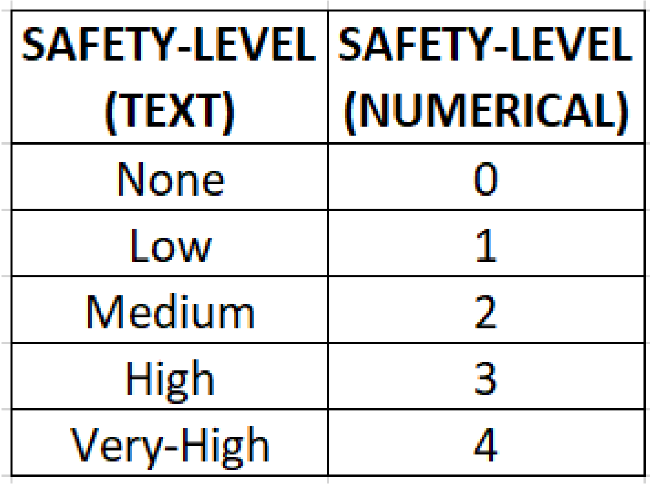
- **Ordinal Variable.** Variable comprises a finite set of discrete values with a ranked ordering between values.

- Many machine learning algorithms cannot operate on nominal data directly. They require all input variables and output variables to be numeric.

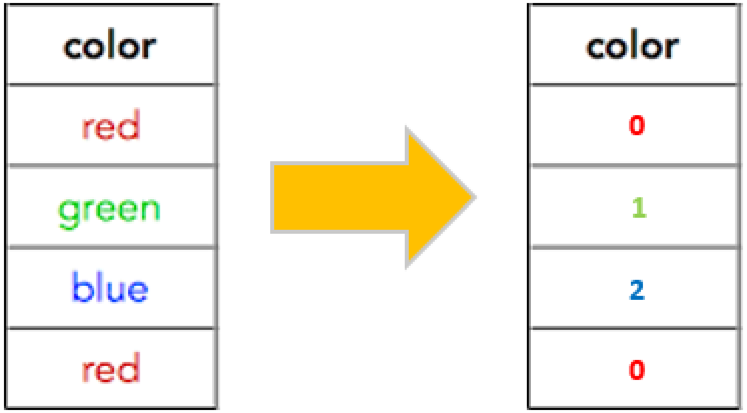
Label Encoding

- In label encoding, each unique category value is assigned an integer value.

- This is also called an ordinal encoding or an integer encoding and is easily reversible. Often, integer values starting at zero are used.



SAFETY-LEVEL (TEXT)	SAFETY-LEVEL (NUMERICAL)
None	0
Low	1
Medium	2
High	3
Very-High	4



color
red
green
blue
red

color
0
1
2
0

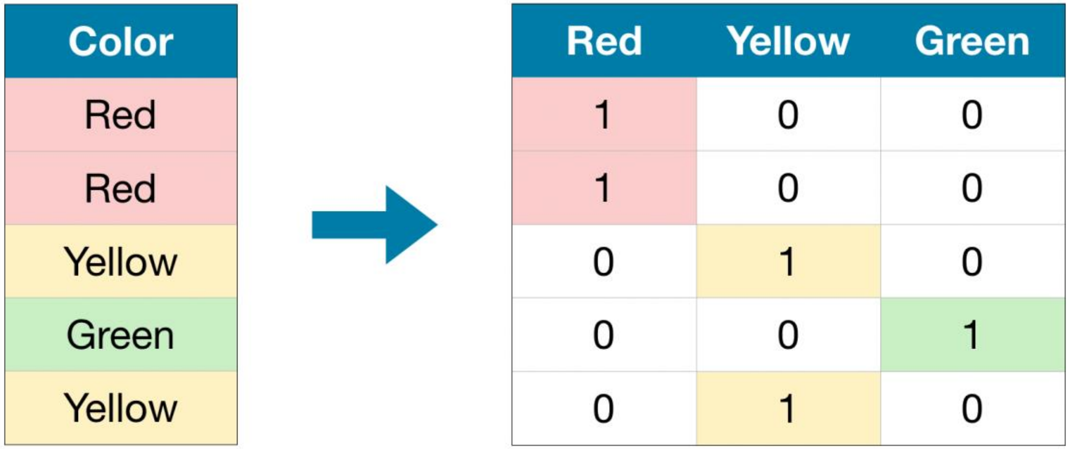
One Hot Encoding

- Forcing an ordinal relationship via label encoding and allowing the model to assume a natural ordering between categories may result in poor performance or unexpected results.

- One-Hot Encoding is the most common, correct way to deal with categorical data.

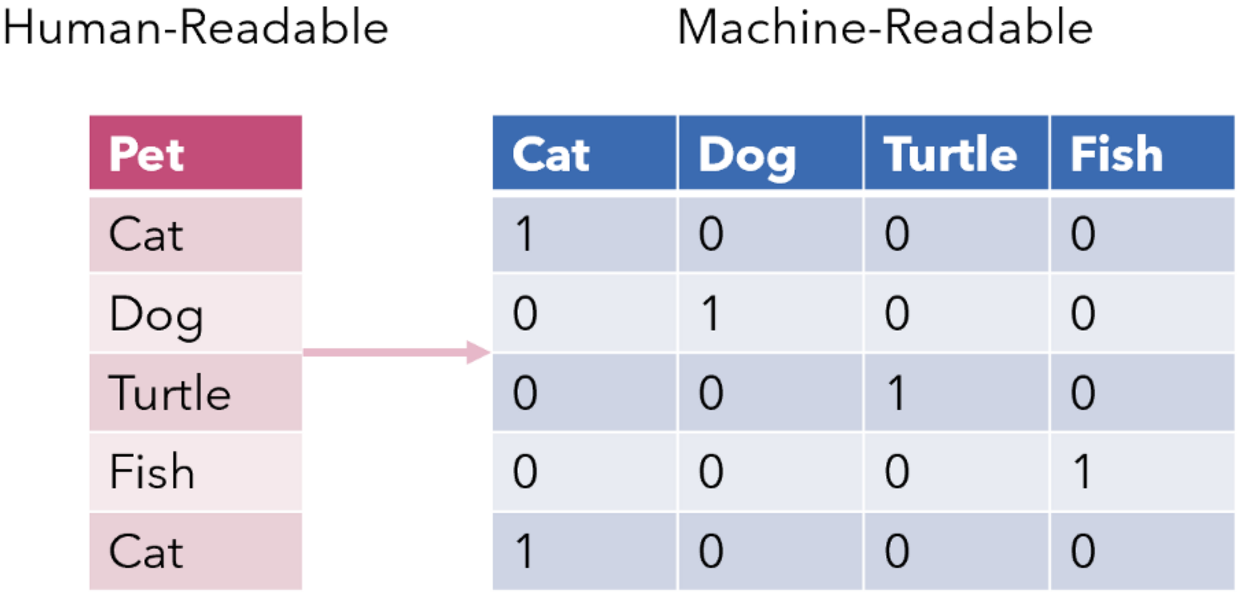
- It consists of creating an additional feature for each group of the categorical feature and mark each observation belonging (Value=1) or not (Value=0) to that group.

One Hot Encoding



Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1
0	1	0



Human-Readable

Pet
Cat
Dog
Turtle
Fish
Cat

Machine-Readable

Cat	Dog	Turtle	Fish
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1
1	0	0	0

One Hot Encoding Implementation

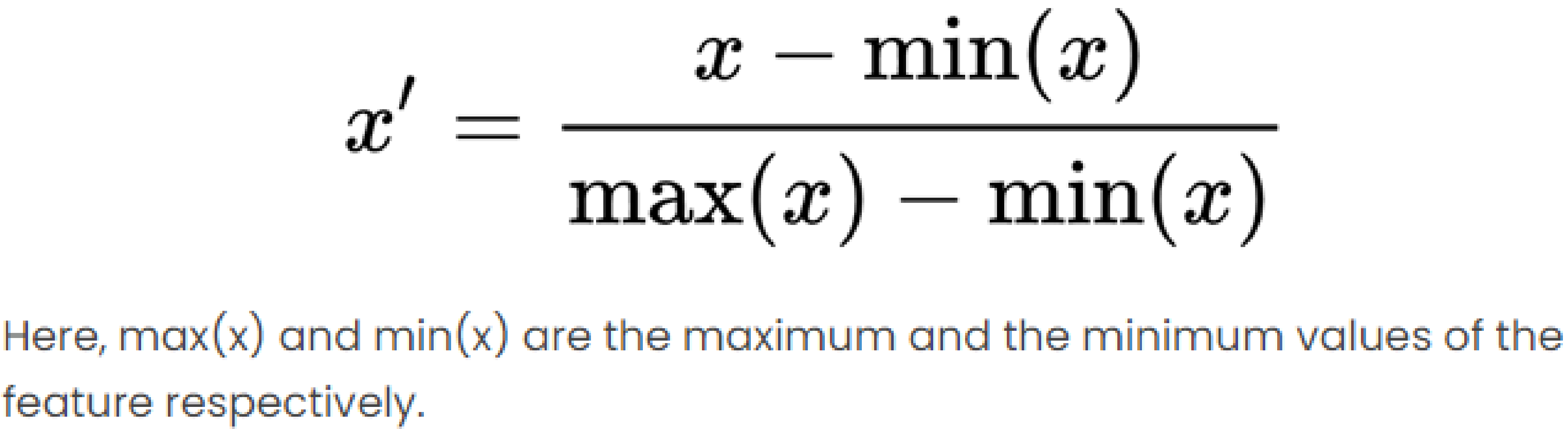
- Implementation of One Hot Encoding

Feature Scaling

- Feature scaling is a method used to normalize the range of independent variables or features of data.
- In data processing, it is also known as data normalization and is generally performed during the data preprocessing step.
- Example: If you have multiple independent variables like age, salary, and height; With their range as (18–100 Years), (25,000–75,000 Rs.), and (1–2 Meters) respectively, feature scaling would help them all to be in the same range, for example- centered around 0 or in the range (0,1) depending on the scaling technique.

Min-Max Normalization

- It is the simplest method and consists of rescaling the range of features to scale the range in [0, 1].



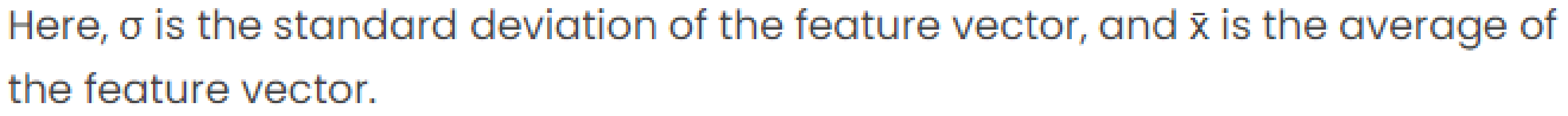
$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Here, max(x) and min(x) are the maximum and the minimum values of the feature respectively.

Standardization

- Feature standardization makes the values of each feature in the data have **zero mean and unit variance**.
- The general method of calculation is to determine the **distribution mean and standard deviation** for each feature and calculate the new data point by the following formula:

$$x' = \frac{x - \bar{x}}{\sigma}$$



Here, σ is the standard deviation of the feature vector, and x̄ is the average of the feature vector.

Normalization vs. Standardization

- **Normalization** is good to use when the distribution of data does not follow a Gaussian distribution. It can be useful in algorithms that do not assume any distribution of the data like K-Nearest Neighbors.

- In Neural Networks algorithm that require data on a 0–1 scale, normalization is an essential pre-processing step. Another popular example of data normalization is image processing, where pixel intensities have to be normalized to fit within a certain range (i.e., 0 to 255 for the RGB color range).

Normalization vs. Standardization

- **Standardization** can be helpful in cases where the data follows a Gaussian distribution. Though this does not have to be necessarily true. Since standardization does not have a bounding range, so, even if there are outliers in the data, they will not be affected by standardization.

- In clustering analyses, standardization comes in handy to compare similarities between features based on certain distance measures. Another prominent example is the Principal Component Analysis, where we usually prefer standardization over Min-Max scaling since we are interested in the components that maximize the variance.

Summary

- Spam Filtering
- Data Preprocessing