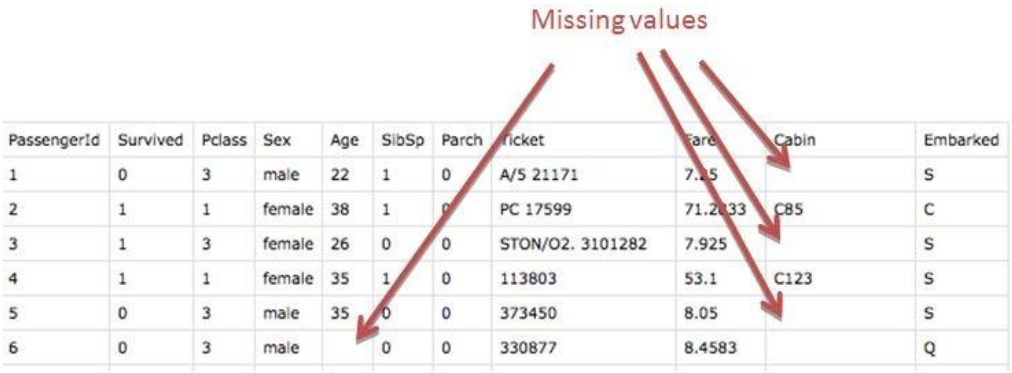
Chapter 5
Data Preprocessing

Data preprocessing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of the model to learn; therefore, it is extremely important that we preprocess our data before feeding it into our model.
The basic concepts used for data preprocessing are:

- Handling Null Values (Imputation)
- Standardization
- Handling Categorical Variables
- One-Hot Encoding
- Outlier Management

5.1 Handling Missing (Null) Values: In real world data, there are some instances where a particular element is absent because of various reasons, such as, corrupt data, failure to load the information, or incomplete extraction. Handling the missing values is one of the greatest challenges faced by analysts, because making the right decision on how to handle it generates robust data models.

The missing values are often encoded as NaNs, blanks or any other placeholders. Training a model with a dataset that has a lot of missing values can drastically impact the machine learning model's quality.



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

There are different ways of handling the missing values as shown in figure 5.1.

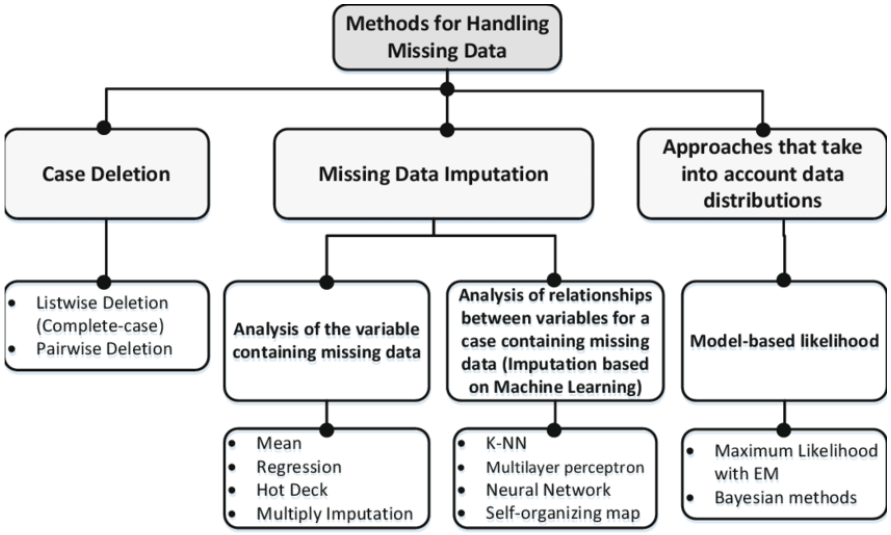


Fig. 5.1 Different ways of handling the missing values

5.1.1. Deleting Rows/Column

Delete a particular row if it has a null value for a particular feature and a particular column more than 70-75% of missing values. This method is advised only when there are enough samples in the data set. One has to make sure that after we have deleted the data, there is no addition of bias. Removing the data will lead to loss of information which will not give the expected results while predicting the output.

Pros:

- Complete removal of data with missing values results in robust and highly accurate model
- Deleting a particular row or a column with no specific information is better, since it does not have a high weightage

Cons:

- Loss of information and data

5.1.2. Replacing (Imputation) with Mean/Median/Mode

This works by calculating the mean/median/mode of the non-missing values in a column and then replacing the missing values within each column separately and independently from the others. This strategy can be applied on a feature which has numeric data like the age of a person or the ticket fare. Replacing with the above three approximations are a statistical approach of handling the missing values. This method is also called as leaking the data while training.

	col1	col2	col3	col4	col5
0	2	5.0	3.0	6	NaN
1	9	NaN	9.0	0	7.0
2	19	17.0	NaN	9	NaN

mean()

	col1	col2	col3	col4	col5
0	2.0	5.0	3.0	6.0	7.0
1	9.0	11.0	9.0	0.0	7.0
2	19.0	17.0	6.0	9.0	7.0

Pros:

- Easy and fast.
- Works well with small numerical datasets.

Cons:

- Doesn't factor the correlations between features. It only works on the column level.
- Will give poor results on encoded categorical features (do NOT use it on categorical features).
- Not very accurate.
- Doesn't account for the uncertainty in the imputations.

5.1.3. Imputation Using (Most Frequent) or (Zero/Constant) Values

This is another statistical strategy to impute missing values, which also works with categorical features (strings or numerical representations) by replacing missing data with the most frequent values within each column.

Pros:

- Works well with categorical features.

Cons:

- It also doesn't factor the correlations between features.
- It can introduce bias in the data.

	col1	col2	col3	col4	col5
0	2	5.0	3.0	6	NaN
1	9	NaN	9.0	0	7.0
2	19	17.0	NaN	9	NaN

df.fillna(0)

	col1	col2	col3	col4	col5
0	2	5.0	3.0	6	0.0
1	9	0.0	9.0	0	7.0
2	19	17.0	0.0	9	0.0

5.1.4. Imputation by Predicting the Missing Values

Using the features which do not have missing values, we can predict the NaN value with the help of a machine learning algorithm. This method may result in better accuracy, unless a missing value is expected to have a very high variance. We will be using linear regression to replace the nulls in the feature 'age', using other available features. One can experiment with different algorithms and check which gives the best accuracy instead of sticking to a single algorithm.

Pros:

- Imputing the missing variable is an improvement as long as the bias from the same is smaller than the omitted variable bias
- Yields unbiased estimates of the model parameters

Cons:

- Bias also arises when an incomplete conditioning set is used for a categorical variable
- Considered only as a proxy for the true values

5.2 Handling the Categorical Data

Categorical data are variables that contain label values rather than numeric values.

The number of possible values is often limited to a fixed set. Categorical variables are often called nominal.

Some examples include:

- A "pet" variable with the values: "dog" and "cat".
- A "color" variable with the values: "red", "green" and "blue".
- A "place" variable with the values: "first", "second" and "third".

Each value represents a different category. Some categories may have a natural relationship to each other, such as a natural ordering. The "place" variable above does have a natural ordering of values. This type of categorical variable is called an ordinal variable.

5.2.1 What is the Problem with Categorical Data?

Some algorithms can work with categorical data directly. For example, a decision tree can be learned directly from categorical data with no data transform required (this depends on the specific implementation). Many machine learning algorithms cannot operate on label data directly. They require all input variables and output variables to be numeric. In general, this is mostly a constraint of the efficient implementation of machine learning algorithms rather than hard limitations on the algorithms themselves. This means that categorical data must be converted to a numerical form. If the categorical variable is an output variable, you may also want to convert predictions by the model back into a categorical form in order to present them or use them in some application.

There are two steps to convert Categorical data to Numerical data:

1. Integer Encoding
2. One-Hot Encoding

5.2.2. Integer Encoding

As a first step, each unique category value is assigned an integer value. For example, "red" is 1, "green" is 2, and "blue" is 3. This is called a label encoding or an integer encoding and is easily reversible. For some variables, this may be enough. The integer values have a natural ordered relationship between each other and machine learning algorithms may be able to understand and harness this relationship. For example, ordinal variables like the "place" example above would be a good example where a label encoding would be sufficient.

5.2.3. One-Hot Encoding

For categorical variables where no such ordinal relationship exists, the integer encoding is not enough. In fact, using this encoding and allowing

the model to assume a natural ordering between categories may result in poor performance or unexpected results (predictions halfway between categories).

A *one hot encoding* is a representation of categorical variables as binary vectors. Then, each value is represented as a binary vector that is all zero values except the index of the value, which is marked with a 1.

In the "color" variable example, there are 3 categories and therefore 3 binary variables are needed. A "1" value is placed in the binary variable for the color and "0" values for the other colors. The binary variables are often called "dummy variables" in other fields, such as statistics.

For example:

1	Red	Green	Blue
2	1	0	0
3	0	1	0
4	0	0	1

town	area	price
monroe township	2600	550000
monroe township	3000	565000
monroe township	3200	610000
monroe township	3600	680000
monroe township	4000	725000
west windsor	2600	585000
west windsor	2800	615000
west windsor	3300	650000
west windsor	3600	710000
robbinsville	2600	575000
robbinsville	2900	600000

town	area	price
1	2600	550000
1	3000	565000
1	3200	610000
1	3600	680000
1	4000	725000
2	2600	585000
2	2800	615000
2	3300	650000
2	3600	710000
3	2600	575000
3	2900	600000

Example: Integer Encoding

town	area	price	monroe township	west windsor	robbinsville
monroe township	2600	550000	1	0	0
monroe township	3000	565000	1	0	0
monroe township	3200	610000	1	0	0
monroe township	3600	680000	1	0	0
monroe township	4000	725000	1	0	0
west windsor	2600	585000	0	1	0
west windsor	2800	615000	0	1	0
west windsor	3300	650000	0	1	0
west windsor	3600	710000	0	1	0
robbinsville	2600	575000	0	0	1
robbinsville	2900	600000	0	0	1

Example: One-Hot Encoding

5.3. Outlier Management

5.3.1 What is an Outlier?

An outlier is a data point that is distant from other similar points. Further simplifying an outlier is an observation that lies on abnormal observation amongst the normal observations in a sample set of population. In statistics, an outlier is an observation point that is distant from other observations.

"Outlier is an observation that appears far away and diverges from an overall pattern in a sample."

Examples:
1) Suppose you have a sample of 1000 people, and amongst them all have to choose one colour between Red and Blue.
If 999 choose Red and only one person chooses Blue, I would say that that person is an outlier for that sample.
2) Let's take an example, we do customer profiling and find out that the average annual income of customers is $0.8 million. But, there are two customers having annual income of $4 and $4.2 million. These two customer's annual income is much higher than rest of the population. These two observations will be seen as Outliers.

5.3.2. Types of outliers: Outliers can be of two kinds: univariate and multivariate.

- Univariate Outlier: A univariate outlier is a data point that consists of an extreme value on one variable.
- Multivariate Outlier: A multivariate outlier is a combination of unusual scores on at least two variables.

Most common causes of outliers on a data set:

- Data entry errors (human errors)
  - ✓ Example: Annual income of a customer is $100,000. Accidentally, the data entry operator puts an additional zero in the figure. Now the income becomes $1,000,000 which is 10 times higher. Evidently, this will be the outlier value when compared with rest of the population.
- Measurement errors (instrument errors)
  - ✓ Example: There are 10 weighing machines. 9 of them are correct, 1 is faulty. Weight measured by people on the faulty machine will be higher / lower than the rest of people in the group. The weights measured on faulty machine can lead to outliers.
- Experimental errors (data extraction or experiment planning/executing errors)

- ✓ Example: In a 100m sprint of 7 runners, one runner missed out on concentrating on the 'Go' call which caused him to start late. Hence, this caused the runner's run time to be more than other runners. His total run time can be an outlier.

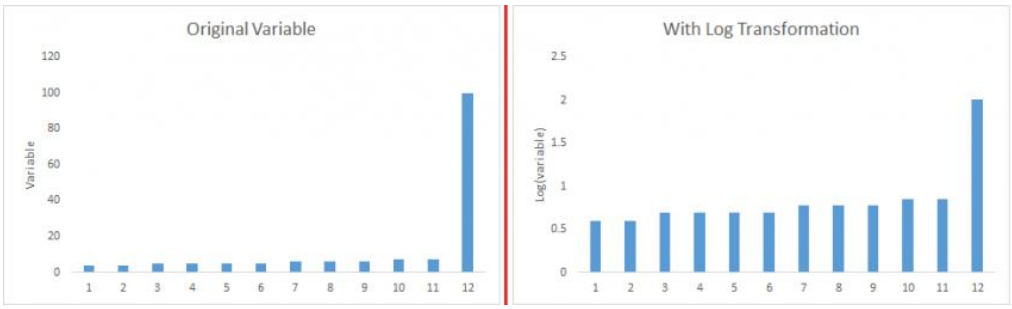
- Intentional (dummy outliers made to test detection methods)
  - ✓ Example: Teens would typically under report the amount of alcohol that they consume. Only a fraction of them would report actual value. Here actual values might look like outliers because rest of the teens are under reporting the consumption.
- Data processing errors (data manipulation or data set unintended mutations)
  - ✓ Example: While extracting data from multiple sources, it can be possible that some manipulation error on data can happen which may lead to cause outliers in the dataset.
- Sampling errors (extracting or mixing data from wrong or various sources)
  - ✓ Example: We have to measure the height of athletes. By mistake, we include a few basketball players in the sample. This inclusion is likely to cause outliers in the dataset.
- Natural (not an error, novelties in data)
  - ✓ In the process of producing, collecting, processing and analyzing data, outliers can come from many sources and hide in many dimensions. Those that are not a product of an error are called ***novelties***.

5.3.3. Methods of Dealing Outliers

The common techniques used to deal with outliers are:

1) Deleting observations: We delete outlier values if it is due to data entry error, data processing error or outlier observations are very small in numbers. We can also use trimming at both ends to remove outliers.

2) Transforming and binning values: Transforming variables can also eliminate outliers. Natural log of a value reduces the variation caused by extreme values. Binning is also a form of variable transformation. Decision Tree algorithm allows to deal with outliers well due to binning of variable. We can also use the process of assigning weights to different observations.



3) Imputing: We can also impute outliers. We can use mean, median, mode imputation methods. Before imputing values, we should analyse if it is natural outlier or artificial. If it is artificial, we can go with imputing values. We can also use statistical model to predict values of outlier observation and after that we can impute it with predicted values.

4) Treat Outliers separately: If there are significant number of outliers, we should treat them separately in the statistical model. One of the approach is to treat both groups as two different groups and build individual model for both groups and then combine the output.