Pantech e Learning
DIGITAL LEARNING SIMPLIFIED

Amazon Web Services

MLOps with AWS

Masterclass

aws

Machine Learning Operations with AWS

Day -11

Amazon Web Services

MLOps with AWS

Masterclass

Pantech e Learning
DIGITAL LEARNING SIMPLIFIED

Encoding

- Machine learning models can only work with numerical values.
- For this reason, it is necessary to transform the categorical values of the relevant features into numerical ones.
- This process is called feature encoding.

Pantech e Learning
DIGITAL LEARNING SIMPLIFIED

One-Hot Encoding

id	color
1	red
2	blue
3	green
4	blue

One Hot Encoding

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

```
from sklearn.preprocessing import OneHotEncoder

encoder = OneHotEncoder()


encoded_value = encoder.fit_transform(encoder)
```

Label Encoder

Original Data

Team	Points
A	25
A	12
B	15
B	14
B	19
B	23
C	25
C	29

Label Encoded Data

Team	Points
0	25
0	12
1	15
1	14
1	19
1	23
2	25
2	29

```
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()


encoded_value = encoder.fit_transform(encoder)
```

Missing/Null Values

- Missing values are common in real-world datasets
- Missing values can occur in datasets due to a variety of reasons such as data entry errors, equipment malfunctions, survey non-responses, and more.
- It is important to handle missing values properly to avoid biased or misleading results.

Sk-learn Imputers

- Simple Imputer
- KNN Imputer
- Iterative Imputer

Simple Imputer

Simple imputer follows a univariate approach to imputing missing values i.e. it only takes a single feature into consideration.

Some of the most common uses of simple imputer are:

- Mean
- Median
- Most frequent (mode)

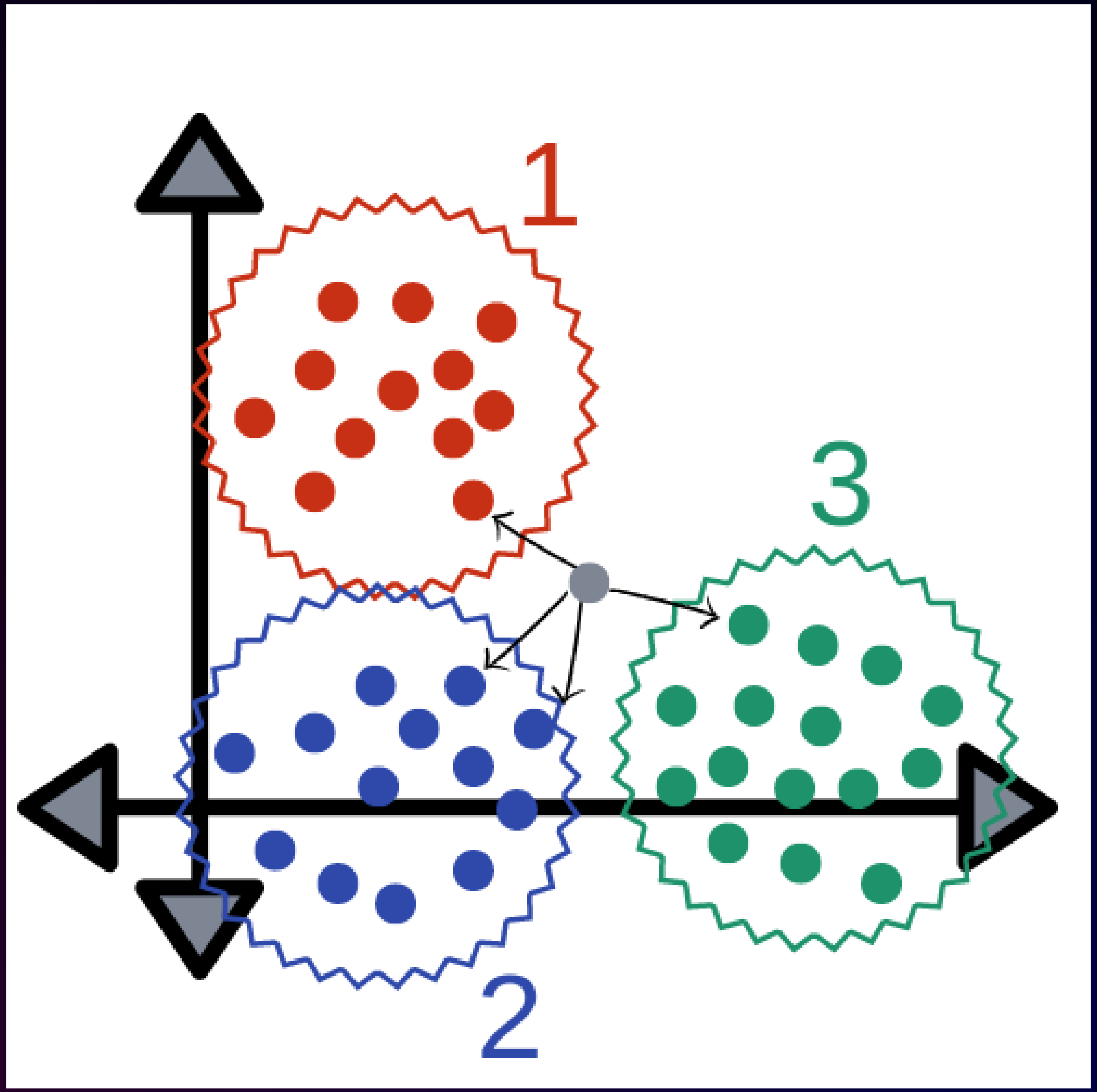
Simple Imputer

```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy = "mean")

imputer.fit_transform(data)
```

KNN Imputer

- KNN Imputer which is another multivariate imputation technique.
- This uses K-Nearest Neighbhors algorithm to impute the missing values.
- NN Imputer scans our dataframe for k nearest observations to the row with missing value.
- It will then proceed to fill the missing value with the average of those nearest observations.

KNN Imputer



KNN Imputer

```
from sklearn.impute import KNNImputer

imputer = KNNImputer(n_neighbors= )

imputer.fit_transform(data)
```

Iterative Imputer

- Iterative imputer is an example of a multivariate approach to imputation.
- It models the missing values in a column by using information from the other columns in a dataset.
- More specifically, it treats the column with missing values as a target variable while the remaining columns are used are predictor variables to predict the target variable.

Iterative Imputer

```
from sklearn.impute import IterativeImputer

imputer = IterativeImputer()

imputer.fit_transform(data)
```

Outlier Detection

- Outliers are data points that are significantly different from the other data points in a dataset.

- These can be observations that are unusually large or small compared to the rest of the data or data points that are far away from the main cluster of data points.

- In machine learning, outliers can have a significant impact on the accuracy of the model, as they can bias the model's predictions and lead to overfitting.

Outlier Detection Methods

- Z-Score method
- IQR method

Z-Score Method

- The Z-score of a data point is a measure of how many standard deviations away from the mean the data point is.
- A data point with a Z-score greater than a certain threshold is considered an outlier.
- A common threshold is a Z-score of 3 or greater, which corresponds to data points that are more than three standard deviations away from the mean.
- Identify the data points with Z-scores greater than the threshold as outliers.

$$z = \frac{x - \mu}{\sigma}$$

$\mu =$ Mean

$\sigma =$ Standard Deviation

Z-Score Method

```
def outlier(data):
  threshold = 3
  mean = np.mean(data)
  sd = np.std(data)
  for i in data:
    z_score = (i - mean)/sd
    if z_score > 3:
      print(i)
```

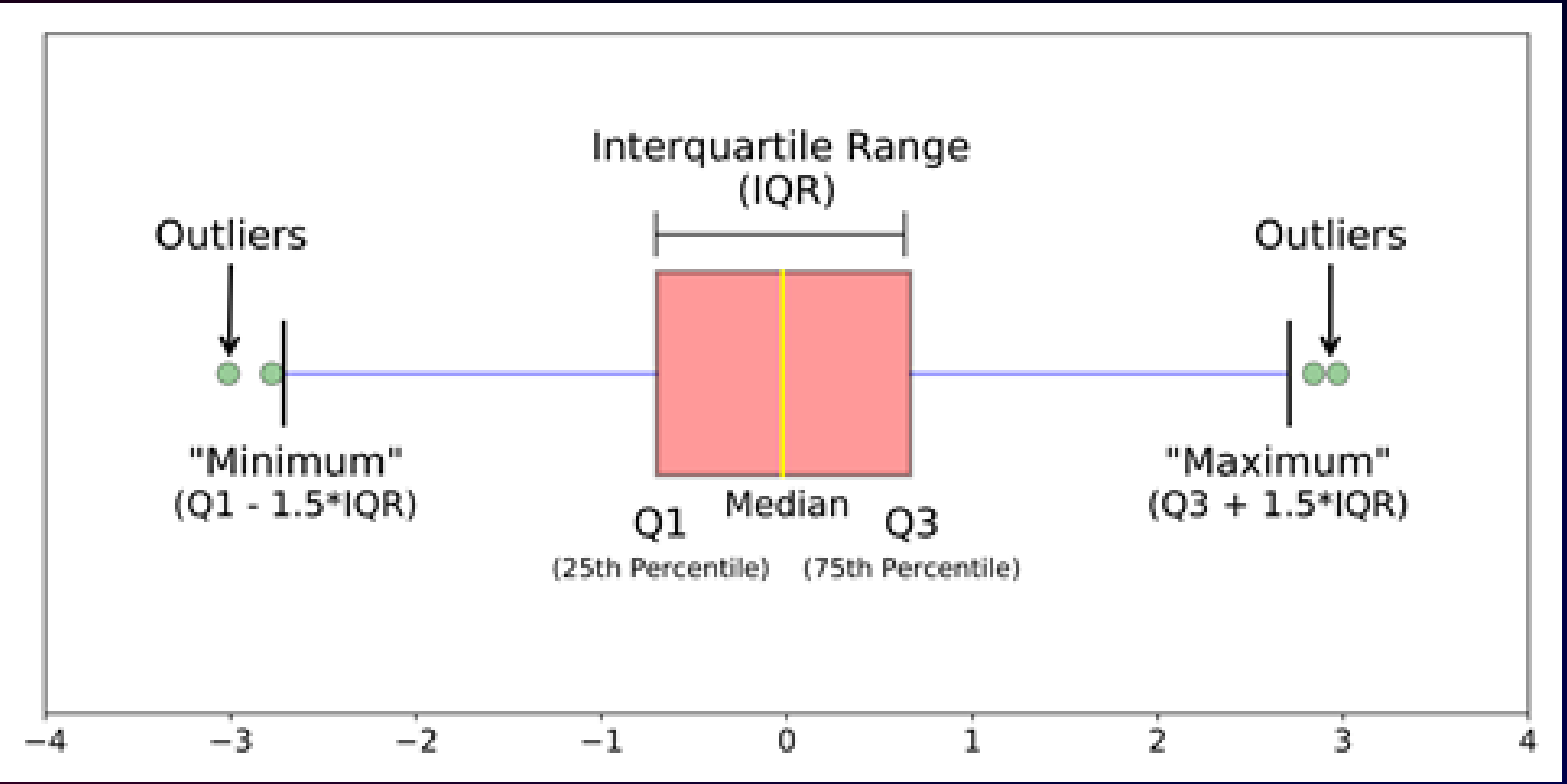
IQR Method

- The IQR is calculated as the difference between the 75th and the 25th percentiles of the data
- Calculate the IQR, which is the difference between the third quartile (Q3) and the first quartile (Q1) of the dataset.
- The first quartile, Q1, is the median of the lower half of the dataset, and the third quartile, Q3, is the median of the upper half of the dataset.

Calculate the lower and upper bounds for outliers using the following formulas:

- Lower Bound = Q1 - 1.5 x IQR , Upper Bound = Q3 + 1.5 x IQR
- Identify the data points hat are outside the lower and upper bounds as outliers.

IQR Method


Interquartile Range
(IQR)
Outliers
Outliers
"Minimum"
(Q1 - 1.5*IQR)
Q1
(25th Percentile)
Median
Q3
(75th Percentile)
"Maximum"
(Q3 + 1.5*IQR)
−4
−3
−2
−1
0
1
2
3
4

IQR Method

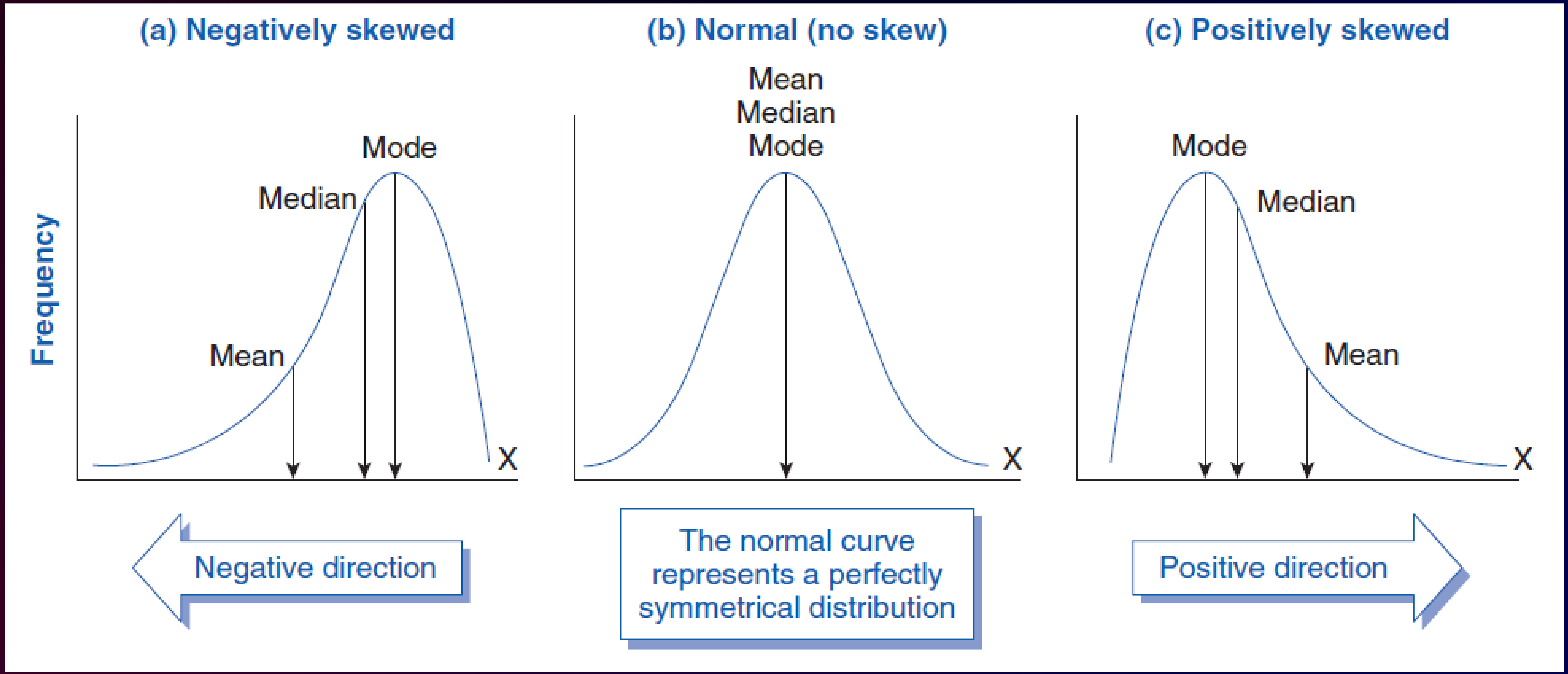
Pantech e Learning
DIGITAL LEARNING SIMPLIFIED

```
q1 = data.quantile(0.25)

q3 = data.quantile(0.75)

IQR = q3 - q1

lower_bound = q1 - 1.5*IQR

upper_bound = q3 + 1.5*IQR

data[(data<lower_bound) | (data > upper_bound)]
```

Use case

- Z-Score method works well for Gaussian(normal) distributed data

- IQR method is better for skewed distribution and for larger datasets with many outliers



Outlier Handling

- There are several methods for handling outliers, including:
- Trimming - Removing the outliers
- Capping - Setting a min and max value and replace the outliers
- Imputation - converting outliers as null values and then impute

Thank you

Pantech e Learning
DIGITAL LEARNING SIMPLIFIED

Pantech e Learning
DIGITAL LEARNING SIMPLIFIED