Applied Machine Learning

Dr. Soumen Das

Machine Learning

Text Books

1. Mitchell T. M. , Machine Learning , McGraw Hill
2. Duda R. O., Hart P. E., Strok D. G. , Pattern Classification, Wiley Interscience

Machine Learning

Course Outcomes (COs)

- 1.Understand the principles, advantages, limitations and possible applications of machine learning.
- 2. Identify the appropriate machine learning techniques for classification (for other task also).
- 3. Apply various pattern recognition, optimization and decision problems in Machine learning. **(Design of ML model to solve different applications in domains)**

Machine Learning

Evaluation scheme

Internal Assessment = 50 (test, quiz, assignment)
Mid Semester = 20
End Semester = 30

What is Learning?

Learning is a process to acquire new or partially new knowledge by improving its performance from experience or environment.

There are different kinds of learning methods.

Learning

Concept learning:

Concept learning is also known as **category learning**, **concept attainment**, and **concept.**

In a concept learning task, a human or machine learner is trained to classify objects by being shown a set of example objects along with their class labels.

Rote Learning (memorization):

Memorizing things without knowing the concept/ logic behind them. A chartered engineer could play the role of a project manager while students play the role of the engineers during a meeting.

Passive Learning (instructions):

Passive learning is a learning paradigm **where learners receive information from the instructor and adopt it**.
Learning from a teacher/expert. (Direct Instruction, Watching Television, Modeled Instruction, University Lectures).

Learning

Active learning:

Active learning is the subset of **machine learning** in which a learning algorithm can query a user interactively to label data with the desired outputs.

Examples: case studies, group projects, think-pair-share, peer teaching, debates

Analogy (experience):

Learning new things from our past experience. Analogy learning can be described the process of finding knowledge acquired in one domain and "using" it in a different domain by establishing similarities between "concepts" in the two domains and transferring relationships between concepts in one domain to the other.

Analogical learning typically involves:

(1) identifying a similarity between two entities (concepts), often referred to as a source entity and a target entity, and

(2) transferring properties or relationships from the source entity to the target entity.

Example: Case-Based Planning

Learning

Two kinds of analogy-based learning are here:

1. Transformational
2. Derivational

Transformational Analogy:
Look for a similar solution and copy it to the new situation making suitable substitutions wherever appropriate. Transformational analogy does not look at how the problem was solved it only looks at the final solution.
Suppose you are asked to prove a theorem in plane geometry. You might look for a previous solution of theorem which is very similar to current one

Derivational Analogy:
The history of the problem solution, the steps involved, is often relevant. We know how to find the area of a triangle and a square. Derivational analogy will help to solve the area/volume of a pyramid from this knowledge.

Learning

Inductive Learning (experience):

On the basis of past experience, formulating a generalized concept. Inductive reasoning makes broad generalizations from specific observations. Basically, there is data, then conclusions are drawn from the data. An example of inductive logic is, "The coin I pulled from the bag is a penny. Second coin from the bag is a penny. A third coin from the bag is a penny. Therefore, all the coins in the bag are pennies."

Deductive Learning:

Deriving new facts from past facts. Deductive reasoning, or deduction, starts out with a general statement, or hypothesis, and examines the possibilities to reach a specific, logical conclusion. For example, the premise "Every A is B" could be followed by another premise, "This C is A." Those statements would lead to the conclusion "This C is B.

For example, "All men are mortal. Harold is a man. Therefore, Harold is mortal."

Learning

Abductive reasoning: Abductive reasoning usually starts with an incomplete set of observations and proceeds to the likeliest possible explanation for the group of observations. Abductive reasoning is often used by doctors who make a diagnosis based on test results.

For example, a person walks into their living room and finds torn up papers all over the floor. The person's dog has been alone in the room all day. The person concludes that the dog tore up the papers because it is the most likely scenario

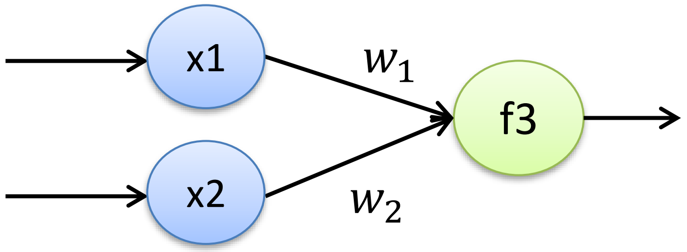
What is ML?

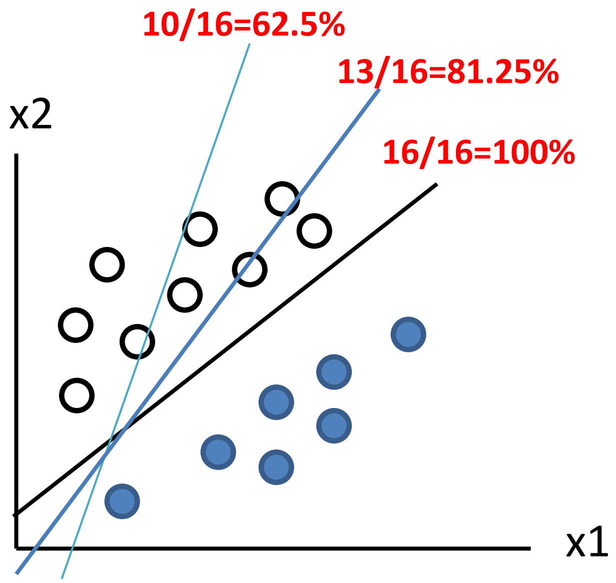
ML is programming computers to optimize a performance criterion using example data or past data. By E. Alpaydin

BP	Heart Beat	Class
120	70	Y
125	65	Y
130	59	N
150	78	N
135	66	N
125	75	N
120	76	Y

y= w1x1+ w2x2 + c

w1x1+ w2x2 + c = 0
x2 = w1/w2*x1 + c/w2





What is ML?

- **Tom Mitchell** provides a more modern **definition**: “A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, **if** its performance at tasks in **T**, as measured by **P**, improves with experience **E**.” ... To find that logic is called “**machine learning**”

Why Machine Learning ?

- **For some defined tasks, if algorithms are available then machine learning is not required**

- **For example, sorting numbers**

- **For some tasks, algorithms are not readily available, to solve those tasks machine learning algorithms are required.**

- **For example, to tell spam emails from legitimate emails.**

- **The problems where no human experts exist. Rainfall forecasting**
- **The problems where human experts exist, but they will be unable to explain their expertise. For example speech recognition.**
- **The problems where the environment changes frequently. For example share market predictions.**
- **The applications that need to be customised for individual users or for a group of users. For example, a program to filter unwanted electronic mail messages.**

- **ML is used to make quicker and reliable decision.**

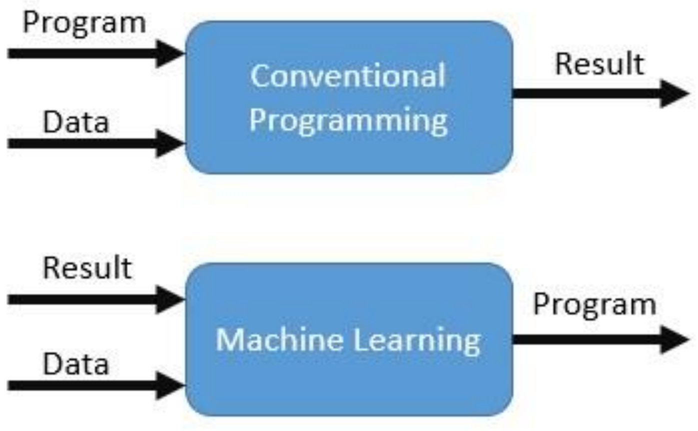
Some Facts about ML

- ML does a good and useful approximation.
- ML uses the theory of statistics in building mathematical models, because the core task is making inference from a sample.
- Application of machine learning methods to large databases is called data mining
- ML is a part of artificial intelligence.
- ML helps us to find solutions to many problems in vision, speech recognition and robotics.
- ML model may be predictive to make predictions in the future.
- ML model may also be descriptive to gain knowledge or it can be both

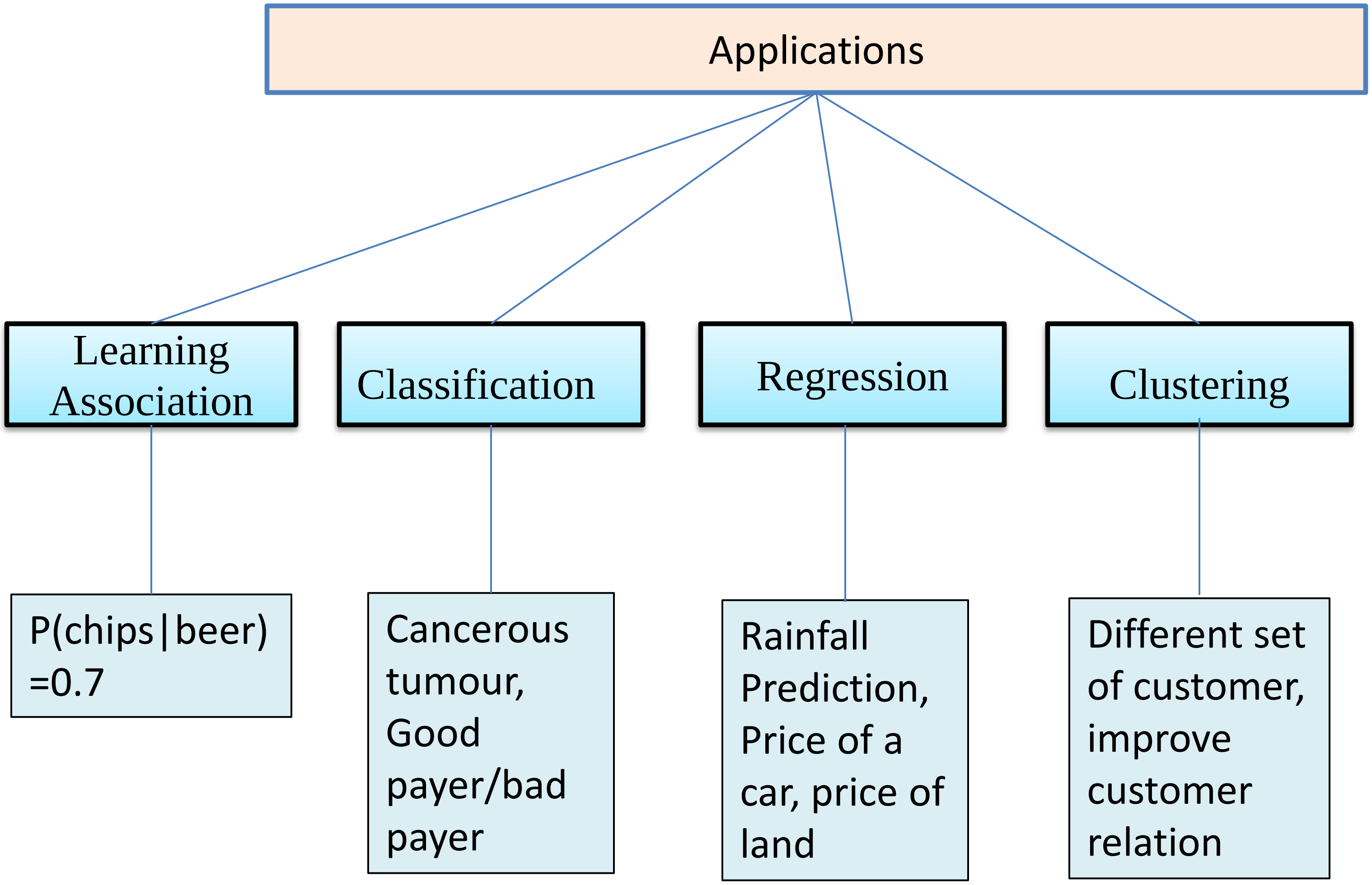
Conventional Programming vs ML Programming

In conventional programming, programs are created manually by providing input data and based on the programming logic, and the computer generates the output.

In machine learning programming, **the input and output data are fed to the algorithm, creating the program**



Kinds of ML Tasks


Applications
Learning Association
Classification
Regression
Clustering
P(chips|beer) =0.7
Cancerous tumour, Good payer/bad payer
Rainfall Prediction, Price of a car, price of land
Different set of customer, improve customer relation

Kinds of ML Tasks

- Learning Association
- Classification
    - Classification
    - Prediction
    - Pattern recognition
        - Optical Character Recognition (OCR)
        - Hand Writing Recognition
        - Face Recognition
        - Medical Diagnosis
        - Speech Recognition
        - Biometrics

    - Knowledge Extraction
    - Compression
    - Outlier Detection

- Regression

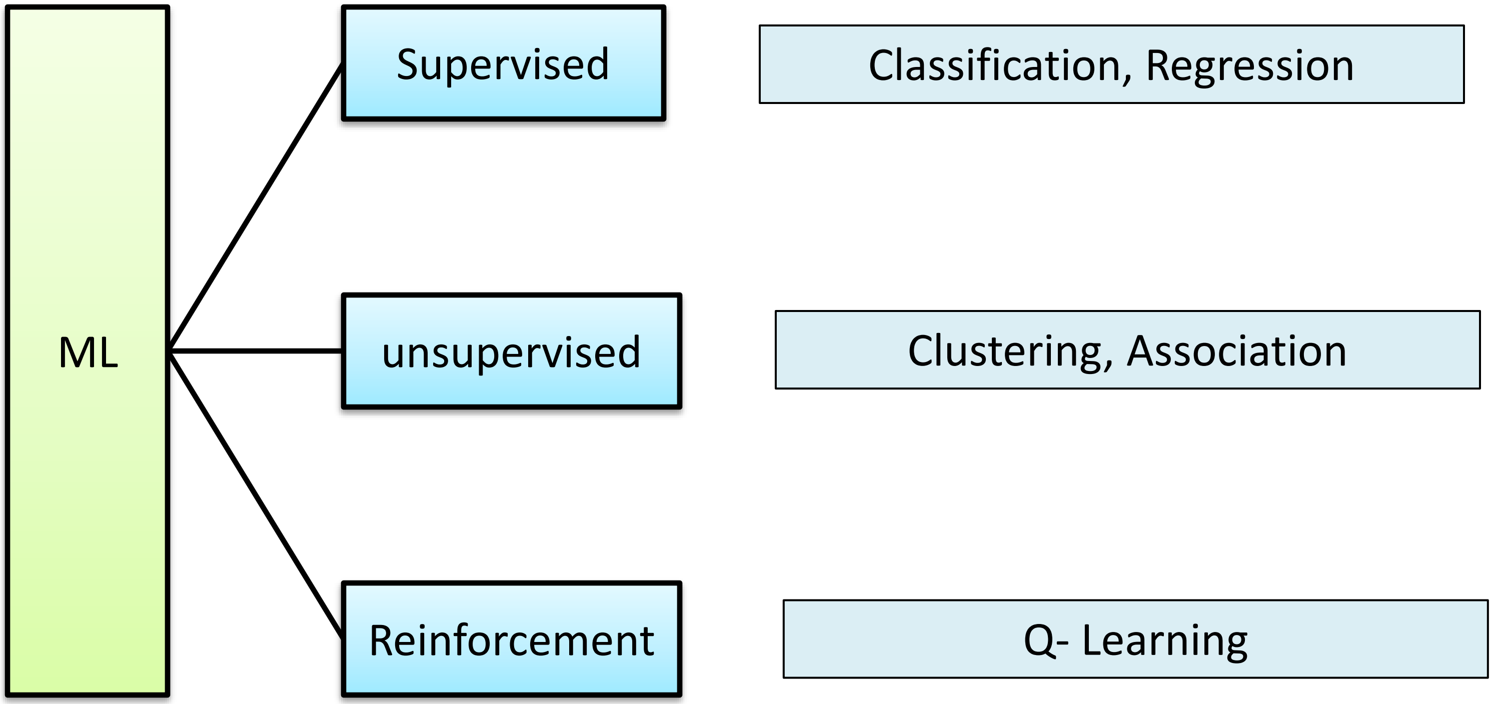
Examples of ML applications

Some most trending real-world applications of Machine Learning:

- Image Recognition (face detection)
- Speech Recognition (Speech to text)
- Traffic prediction (Google Map)
- Product recommendations (Amazon, Netflix, etc.)
- Self-driving cars (Deep learning is used)
- Email Spam and Malware Filtering (Multi-Layer Perceptron, Decision tree, and Naïve Bayes classifier)
- Online Fraud Detection (Feed Forward Neural network)
- Stock Market trading (Recurrent neural network like LSTM)
- Weather prediction (rain fall prediction etc. Recurrent neural network like LSTM)
- Medical Diagnosis (finding brain tumors; Deep learning)
- Automatic Language Translation (GNMT (Google Neural Machine Translation))

Classification of ML

Classification of ML Algorithm



Classification of ML

Supervised

Supervised Learning (SL) is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.

Advantage:

- Huge number of application
- Performance is good

Disadvantages:

- **Slow** (it requires human experts to manually label training examples one by one)
- **Costly** (a model should be trained on the large volumes of hand-labeled data to provide accurate predictions)

BP	Heart Beat	Weigh t	Class
120	70	50	Y
125	65	60	Y
130	59	52	N
150	78	70	N
135	66	85	N
125	75	82	N
120	76	90	Y

Roll. No	Age	Height	Weight
12	12	1.1	23
14	11	1.05	21.6
19	13	1.2	24.7
32	11	1.07	21.3

Classification of ML

Supervised

- Naïve Bayes
- Decision Tree (DT) [ID3, C4.5, C 5.0, CART]
- Support Vector Machine (SVM)
- **Artificial Neural Network (ANN)**
- K- Nearest Neighbour (K-NN)
- **Linear Regression**
- **Polynomial Regression**
- **Logistic Regression**

BP	Heart Beat	Weigh t	Class
120	70	50	Y
125	65	60	Y
130	59	52	N
150	78	70	N
135	66	85	N
125	75	82	N
120	76	90	Y

Roll. No	Age	Height	Weight
12	12	1.1	23
14	11	1.05	21.6
19	13	1.2	24.7
32	11	1.07	21.3

Classification of ML

Unsupervised

Unsupervised learning, also known as unsupervised machine learning, uses machine learning algorithms to analyze and cluster unlabeled datasets.

Advantages:

- solves the problem by learning the data without any labels.
- It is very helpful in finding patterns in data, which are not possible to find using normal methods.
- There is lesser complexity compared to the supervised learning task. Here, no one is required to interpret the associated labels and hence it holds lesser complexities.
- It is reasonably easier to obtain unlabeled data.

Patterns	**Value of attributes**		
	A1	A2	A3
X1	1	1	3
X2	2	3	6
X3	3	1	2
X4	4	4	2
X5	5	2	1

Classification of ML

Unsupervised

Unsupervised learning, also known as unsupervised machine learning, uses machine learning algorithms to analyze and cluster unlabeled datasets.

Disadvantages:

- has a *limited area of applications* (mostly for clustering purposes)
- provides *less accurate results*
- might require human intervention to understand the patterns and correlate them with the domain knowledge
- cannot get precise information regarding the output

Use:

Anomaly detection, Segmentation, Dimensionality reduction

Patterns	Value of attributes		
	A1	A2	A3
X1	1	1	3
X2	2	3	6
X3	3	1	2
X4	4	4	2
X5	5	2	1

Classification of ML

Unsupervised

- Clustering
  - K-Means
  - K-Mediod
  - CURE
  - BIRCH

- Association Rule Mining
  - Apriori Algorithm
  - Predictive Apriori Algorithm
  - Tertius Algorithm
  - Eclat
  - FP-Growth

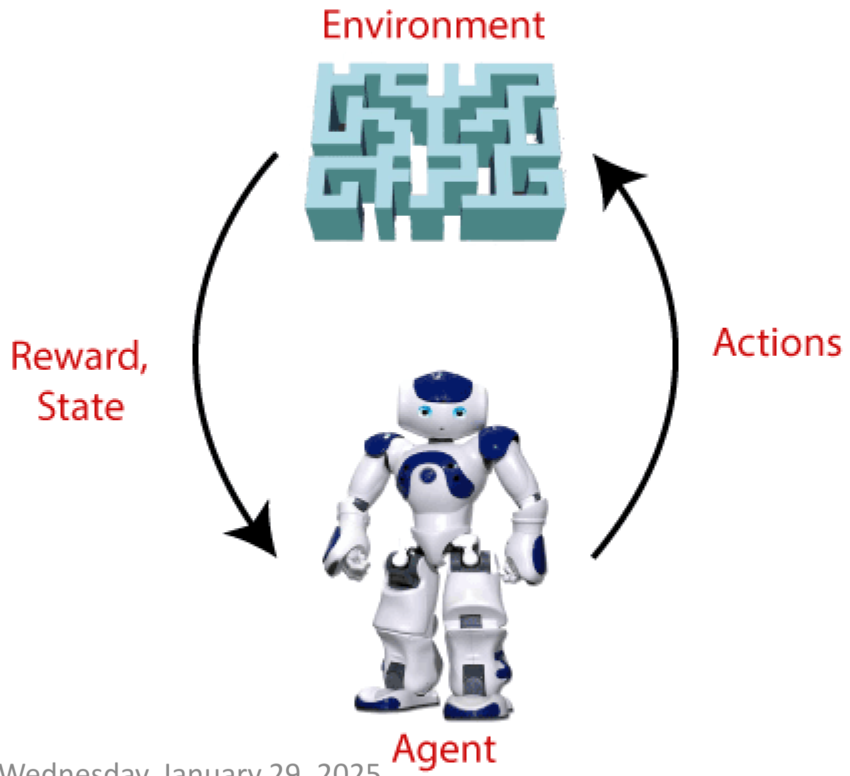
Patterns	Value of attributes		
	A1	A2	A3
X1	1	1	3
X2	2	3	6
X3	3	1	2
X4	4	4	2
X5	5	2	1

Classification of ML

Reinforcement

Reinforcement learning is a machine learning training method based on rewarding desired behaviors and/or punishing undesired ones.

Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. **For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty**



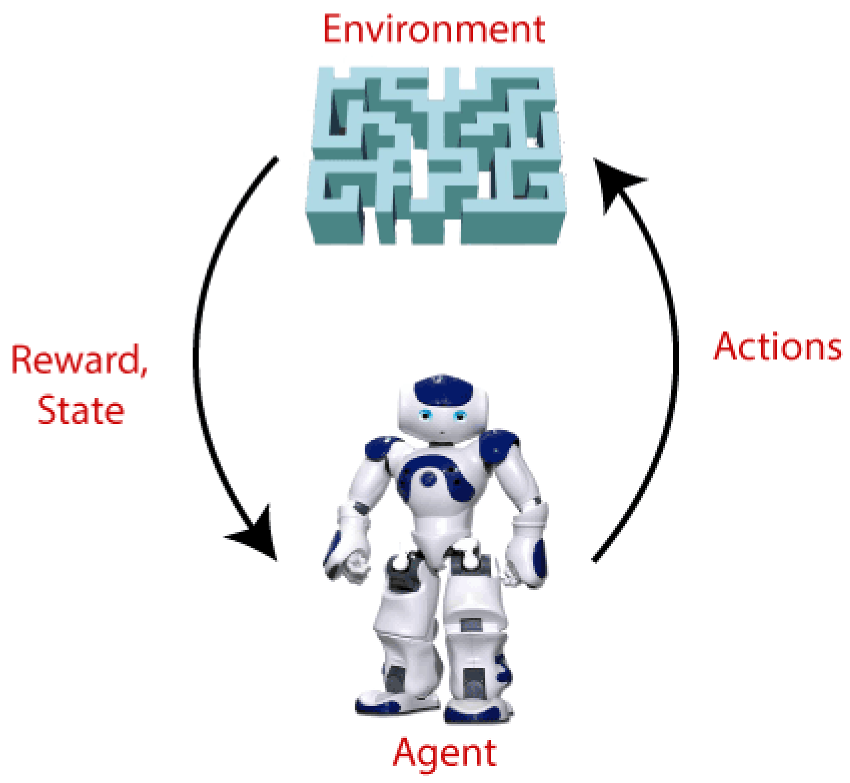
BP	Heart Beat	Weig ht	Feedba ck
120	70	50	reward
125	65	60	penalty
130	59	52	penalty
150	78	70	penalty
135	66	85	reward
125	75	82	reward
120	76	90	reward

Classification of ML

Reinforcement

Advantages:

- Reinforcement learning doesn't require large labeled datasets.
- It's **Innovative.**
- **Bias Resistance**
- **Goal-oriented**, Reinforcement learning can be used for sequences of actions.
- Reinforcement learning is **Adaptable.** Reinforcement learning doesn't require retraining because it adapts to new environments automatically on the fly.
- Reinforcement learning can be used to solve very complex problems that cannot be solved by conventional techniques.
- The model can correct the errors that occurred during the training process

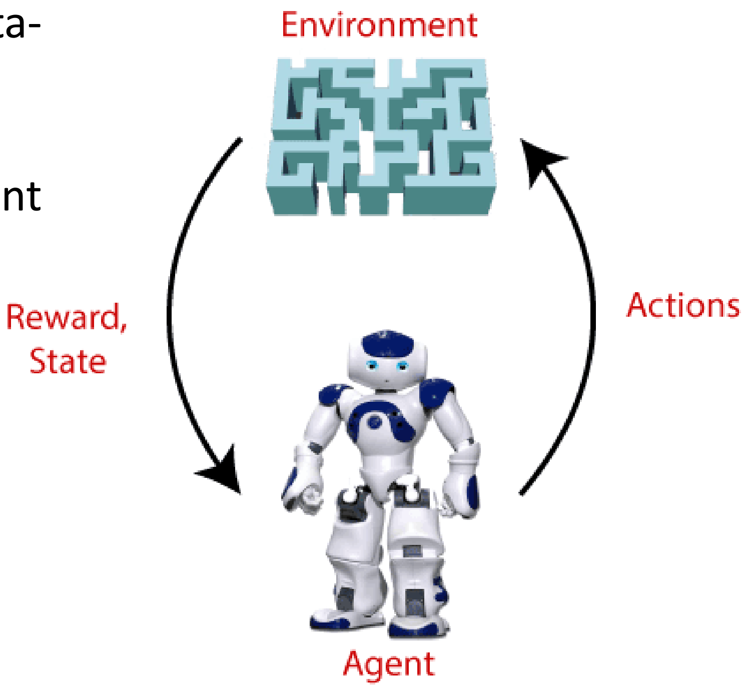


Classification of ML

Reinforcement

Disadvantages:

- Can diminish the results due to too much reinforcement learning
- Not preferable to use for solving simple problems.
- Needs a lot of data and a lot of computation. It is data-hungry
- Assumes the world is Markovian, which it is not
- The curse of dimensionality limits reinforcement learning heavily for real physical systems.



Applications:

Robotics: Robot navigation, walking etc

Control: Adaptive control such as Factory processes etc.

Game Playing: Game playing like chess, etc.

Chemistry: Optimizing the chemical reactions.

Business: business strategy planning

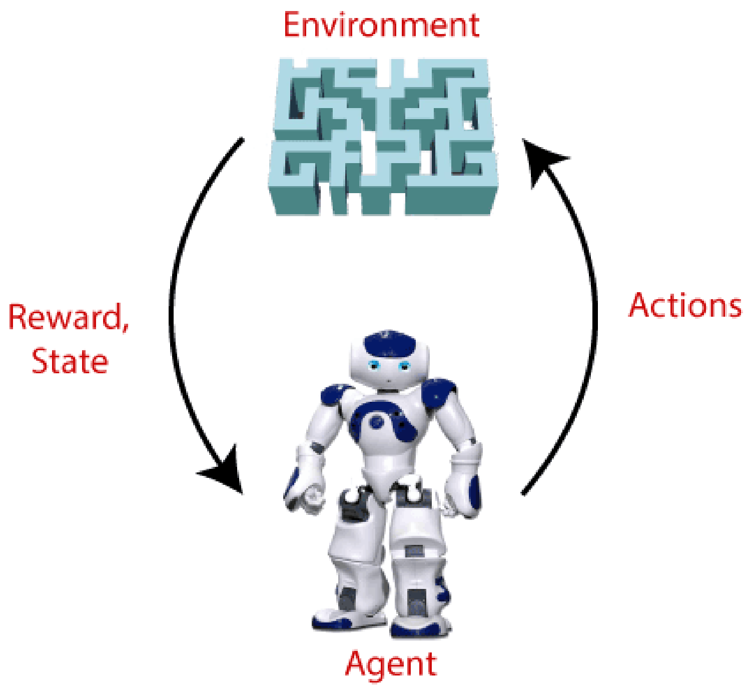
Manufacturing: automobile manufacturing companies

Finance Sector: finance sector for evaluating trading strategies

Classification of ML

Reinforcement

- Markov Decision Process (MDP)
- Q learning: Deep-Q-Neural Network (DQN)
- State Action Reward State Action (SARSA)



Semi-Supervised

Semi-Supervised Learning (SSL) is a learning problem that involves a small number of labeled examples and a large number of unlabeled examples.
Semi-supervised machine learning is **a combination of supervised and unsupervised machine learning methods**.

- Unlike unsupervised learning, SSL works for a variety of problems from classification and regression to clustering and association.

- Unlike supervised learning, the method uses small amounts of labeled data and also large amounts of unlabeled data, which reduces expenses on manual annotation and cuts data preparation time.

BP	Heart B	Weight	Class
120	70	50	Y
125	65	60	Y
130	59	52	N
150	78	70	?
135	66	85	?
125	75	82	?
120	76	90	?

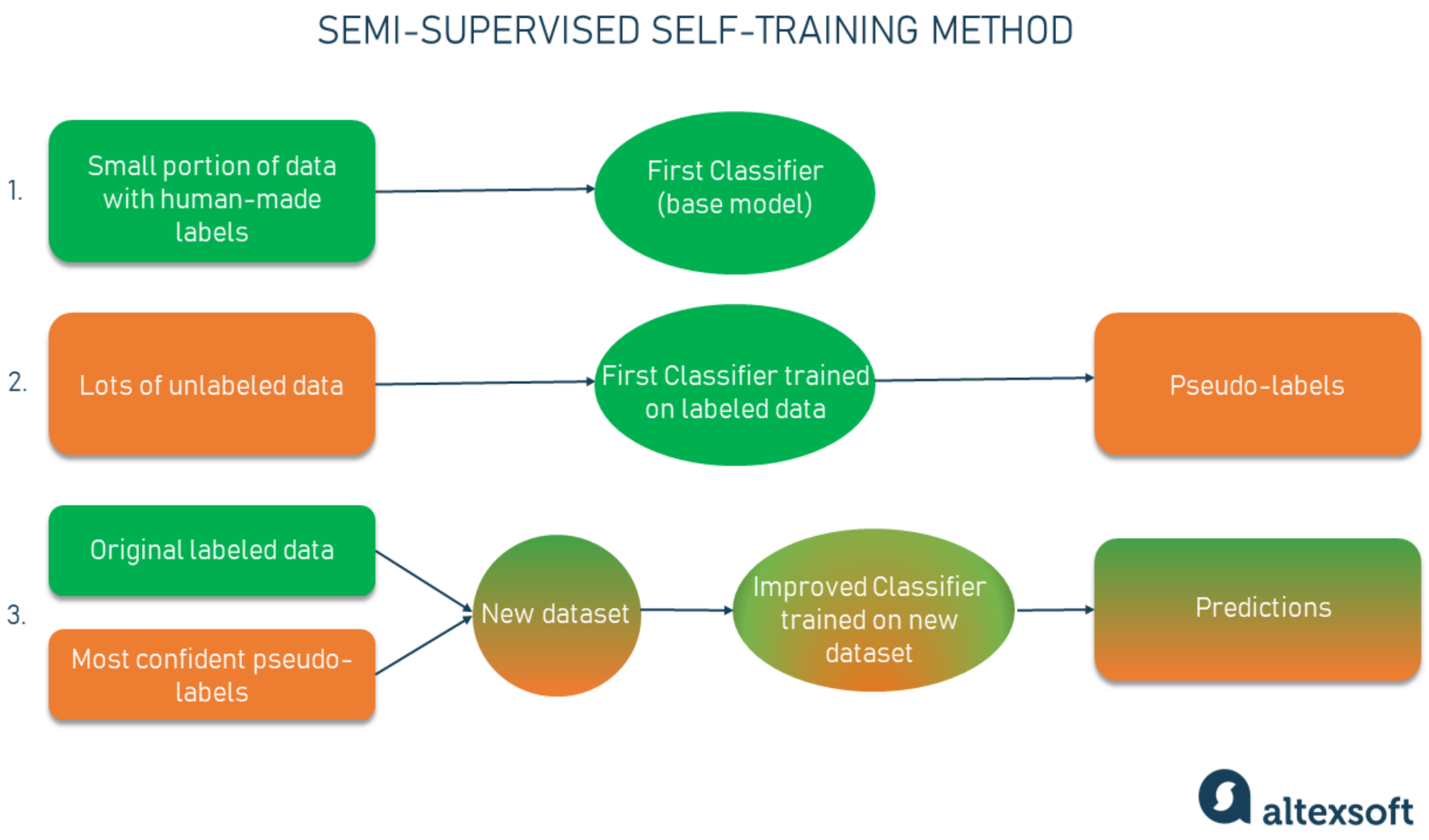
Advantages:

- Easy to understand.
- Reduces the amount of annotated data used.
- A stable algorithm
- High efficiency
- Large applications
- Work as domain expert

Semi-Supervised

Disadvantages:

- Low accuracy
- Results are not stable
- Not appropriate for complex problems


SEMI-SUPERVISED SELF-TRAINING METHOD
1.
Small portion of data with human-made labels
First Classifier (base model)
2.
Lots of unlabeled data
First Classifier trained on labeled data
Pseudo-labels
3.
Original labeled data
Most confident pseudo-labels
New dataset
Improved Classifier trained on new dataset
Predictions
altexsoft

Semi-Supervised

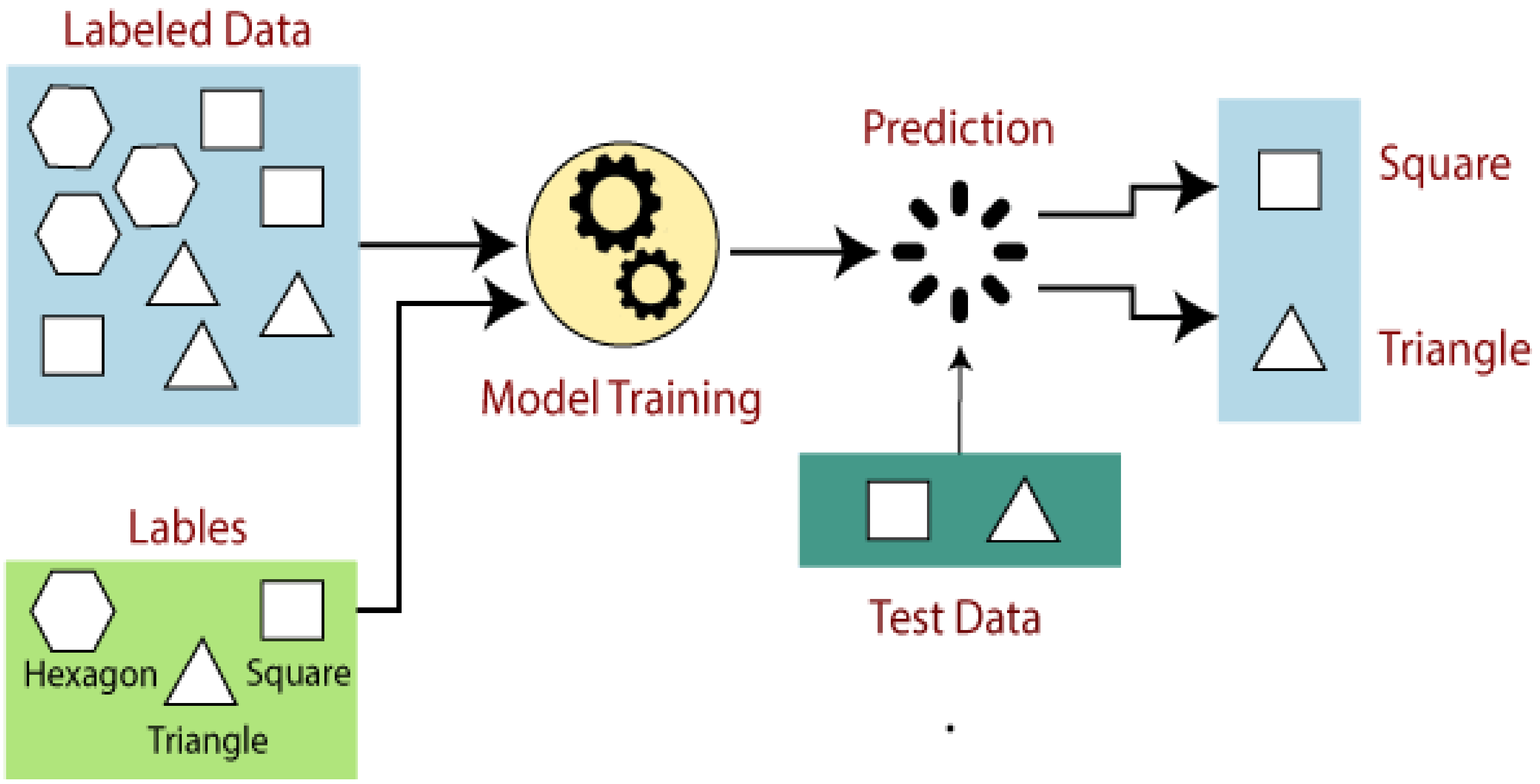
Applications:

- Text document classification
- Speech Analysis
- Protein Sequence Classification
- Internet Content Classification

Supervised Learning Algorithm

- Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output.
- The labelled data means some input data is already tagged with the correct output.
- In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly.
- Supervised learning is a process of providing input data as well as correct output data to the machine learning model.
- The aim of a supervised learning algorithm is to **find a mapping function to map the input variable(x) with the output variable(y)**.

Supervised Learning Algorithm

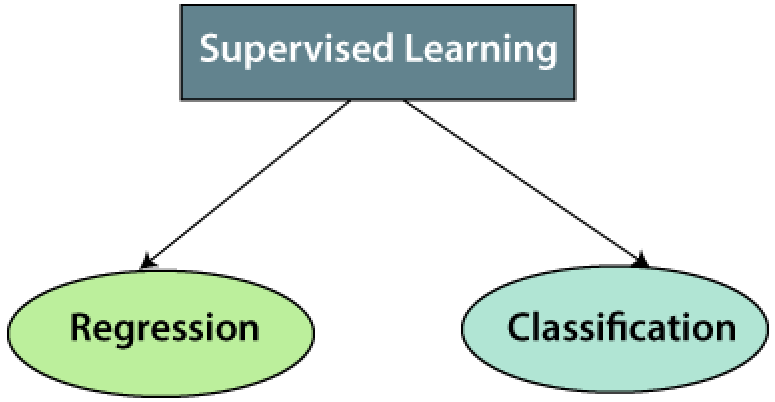


Supervised Learning Algorithm

Steps Involved in Supervised Learning:

- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training **dataset, test dataset, and validation dataset**.
- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set. If the model predicts the correct output, which means our model is accurate.

Supervised Learning Algorithm



Regression

Regression is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc. Below are some popular Regression algorithms which come under supervised learning

- Linear Regression
- Non-Linear Regression
- Polynomial Regression

Supervised Learning Algorithm

Classification

Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-false, etc.

- Random Forest
- Decision Trees
- Logistic Regression
- Support vector Machines

Supervised Learning Algorithm

Advantages of Supervised learning:

- With the help of supervised learning, the model can predict the output on the basis of prior experiences.
- In supervised learning, we can have an exact idea about the classes of objects.
- Supervised learning model helps us to solve various real-world problems such as **fraud detection, spam filtering**, etc.

Disadvantages of supervised learning

- Supervised learning models are not suitable for handling the complex tasks.
- Supervised learning cannot predict the correct output if the test data is different from the training dataset.
- Training required lots of computation times.
- In supervised learning, we need enough knowledge about the classes of object.

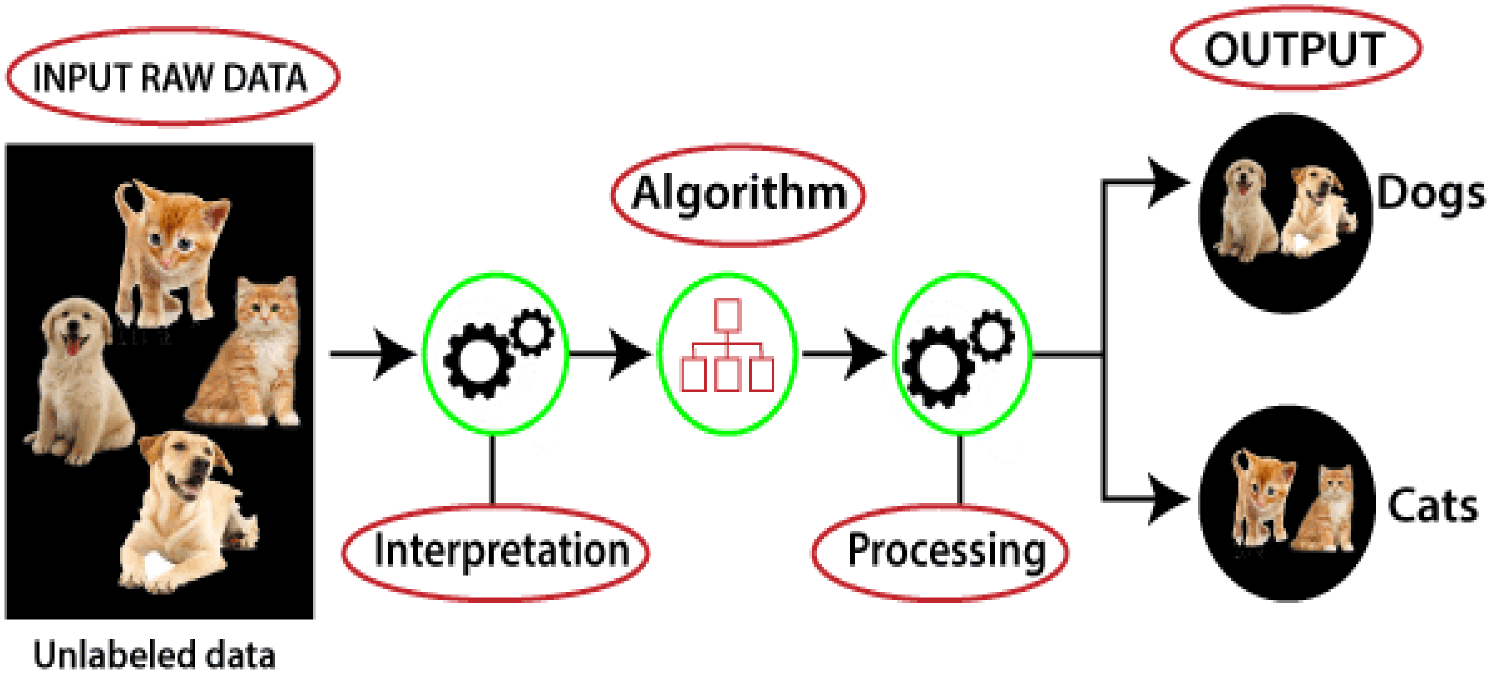
Unsupervised Learning Algorithm

- There may be many cases in which we do not have labeled data and need to find the hidden patterns from the given dataset. So, to solve such types of cases in machine learning, we need unsupervised learning techniques.

- Models itself find the hidden patterns and insights from the given data. It can be compared to learning which takes place in the human brain while learning new things

- Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to **find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format**.

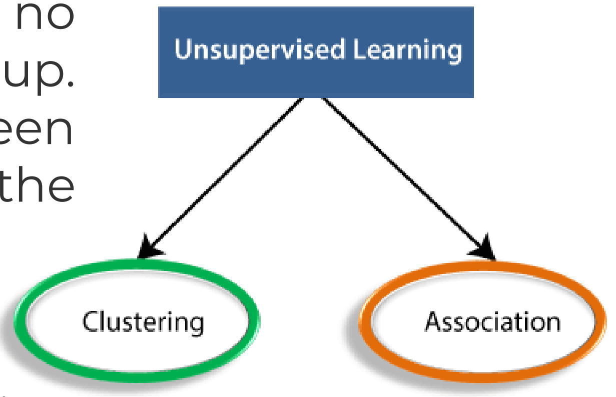
Unsupervised Learning Algorithm



Unsupervised Learning Algorithm

Clustering:

Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group. Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities.



Association:
An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is Market Basket Analysis.

Unsupervised Learning Algorithm

Advantages of Unsupervised learning

- It does not require training data to be labeled.
- Dimensionality reduction can be easily accomplished using unsupervised learning.
- Capable of finding previously unknown patterns in data.
- Unsupervised learning can help you gain insights from unlabeled data that you might not have been able to get otherwise.
- Unsupervised learning is good at finding patterns and relationships in data without being told what to look for. This can help you learn new things about your data.

Unsupervised Learning Algorithm

Disadvantages of Unsupervised learning

- Difficult to measure accuracy or effectiveness due to lack of predefined answers during training.
- The results often have lesser accuracy.
- The user needs to spend time interpreting and label the classes which follow that classification.
- Unsupervised learning can be sensitive to data quality, including missing values, outliers, and noisy data.
- Without labeled data, it can be difficult to evaluate the performance of unsupervised learning models, making it challenging to assess their effectiveness.

Reinforcement Learning Algorithm

- Reinforcement Learning (RL) is a branch of machine learning focused on making decisions to maximize cumulative rewards in a given situation.

- Unlike supervised learning, which relies on a training dataset with predefined answers, RL involves learning through experience.

- In RL, an agent learns to achieve a goal in an uncertain, potentially complex environment by performing actions and receiving feedback through rewards or penalties.

Reinforcement Learning Algorithm

How Reinforcement Learning Works

RL operates on the principle of learning optimal behavior through trial and error. The agent takes actions within the environment, receives rewards or penalties, and adjusts its behavior to maximize the cumulative reward. This learning process is characterized by the following elements:

- **Policy:** A strategy used by the agent to determine the next action based on the current state.
- **Reward Function:** A function that provides a scalar feedback signal based on the state and action.
- **Value Function:** A function that estimates the expected cumulative reward from a given state.
- **Model of the Environment:** A representation of the environment that helps in planning by predicting future states and rewards.

Reinforcement Learning Algorithm

- **Agent():** An entity that can perceive/explore the environment and act upon it.
- **Environment():** A situation in which an agent is present or surrounded by. In RL, we assume the stochastic environment, which means it is random in nature.
- **Action():** Actions are the moves taken by an agent within the environment.
- **State():** State is a situation returned by the environment after each action taken by the agent.
- **Reward():** A feedback returned to the agent from the environment to evaluate the action of the agent.

Reinforcement Learning Algorithm

Value-based:

The value-based approach is about to find the optimal value function, which is the maximum value at a state under any policy. Therefore, the agent expects the long-term return at any state(s) under policy π.

Policy-based:
Policy-based approach is to find the optimal policy for the maximum future rewards without using the value function. In this approach, the agent tries to apply such a policy that the action performed in each step helps to maximize the future reward.

The policy-based approach has mainly two types of policy:

1. **Deterministic:** The same action is produced by the policy (π) at any state.
2. **Stochastic:** In this policy, probability determines the produced action.

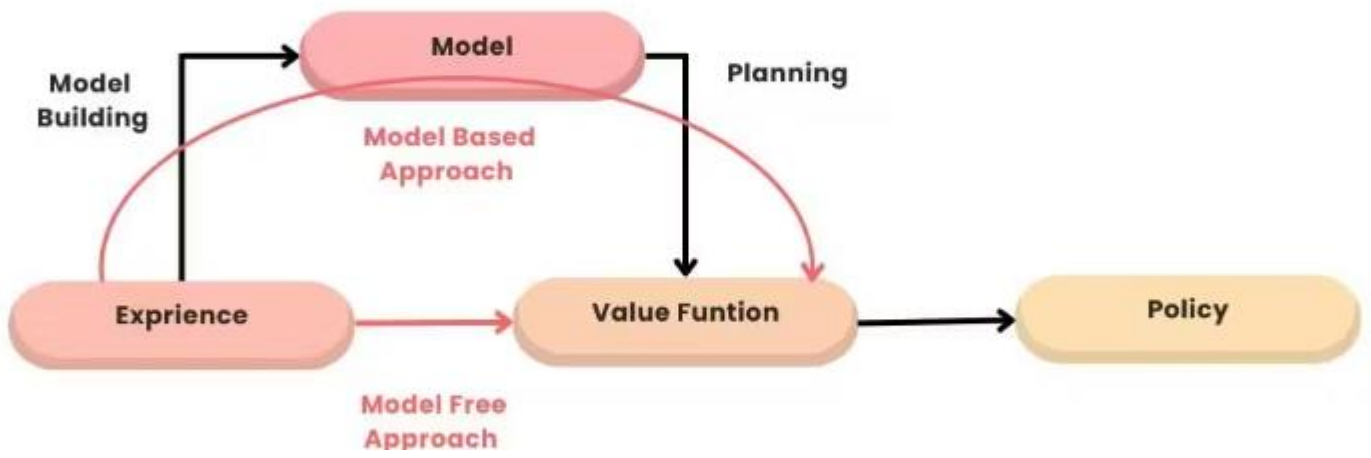
Model-based: In the model-based approach, a virtual model is created for the environment, and the agent explores that environment to learn it.

Reinforcement Learning Algorithm

Main points in Reinforcement learning

- **Input:** The input should be an initial state from which the model will start
- **Output:** There are many possible outputs as there are a variety of solutions to a particular problem
- **Training:** The training is based upon the input, The model will return a state and the user will decide to reward or punish the model based on its output.
- The model keeps continues to learn.
- The best solution is decided based on the maximum reward.

Reinforcement Learning Algorithm



Model-based algorithms

In model-based algorithms, the agent is able to predict the reward of an outcome and takes the action in order to maximize the reward. It is a greedy algorithm where the decision is entirely based on maximizing the rewards points.

It is used in situations where we have complete knowledge about an environment and the outcome of the actions in that environment. For environments that are fixed or static in nature, model-based algorithms are more suitable. They also allow agents to plan ahead.

Reinforcement Learning Algorithm

Model-free algorithms

In model-free algorithms, the agent carries out multiple actions multiple times and learns from the outcomes. Based on the learning experience, it tries to decide a policy or a strategy to carry out actions with an aim to get optimal reward points. This type of algorithm should be applied to environments with a dynamic nature and where we don't have complete knowledge about them.

A reinforcement learning example is autonomous driving cars that have a dynamic environment where there can be a lot of changes in traffic routes. Model-free algorithms are most suitable in such situations.

Reinforcement Learning Algorithm

Application of Reinforcement Learnings

i) Robotics: Automating tasks in structured environments like manufacturing.

ii) Game Playing: Developing strategies in complex games like chess.

iii) Industrial Control: Real-time adjustments in operations like refinery controls.

iv) Personalized Training Systems: Customizing instruction based on individual needs.

Reinforcement Learning Algorithm

Advantages and Disadvantages of Reinforcement Learning
Advantages:

- Reinforcement learning can be used to solve very complex problems that cannot be solved by conventional techniques.
- The model can correct the errors that occurred during the training process.
- In RL, training data is obtained via the direct interaction of the agent with the environment
- Reinforcement learning can handle environments that are non-deterministic, meaning that the outcomes of actions are not always predictable. This is useful in real-world applications where the environment may change over time or is uncertain.
- Reinforcement learning can be used to solve a wide range of problems, including those that involve decision making, control, and optimization.
- Reinforcement learning is a flexible approach that can be combined with other machine learning techniques, such as deep learning, to improve performance.

Reinforcement Learning Algorithm

Advantages and Disadvantages of Reinforcement Learning

Disadvantages:

- Reinforcement learning is not preferable to use for solving simple problems.
- Reinforcement learning needs a lot of data and a lot of computation
- Reinforcement learning is highly dependent on the quality of the reward function. If the reward function is poorly designed, the agent may not learn the desired behavior.
- Reinforcement learning can be difficult to debug and interpret. It is not always clear why the agent is behaving in a certain way, which can make it difficult to diagnose and fix problems.

Python and libraries for machine learning

What is Pandas?

- Pandas is a Python library used for working with data sets.
- It has functions for analyzing, cleaning, exploring, and manipulating data.
- The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.

Why Use Pandas?

- Pandas allows us to analyze big data and make conclusions based on statistical theories.
- Pandas can clean messy data sets, and make them readable and relevant.
- Relevant data is very important in data science.

What Can Pandas Do?

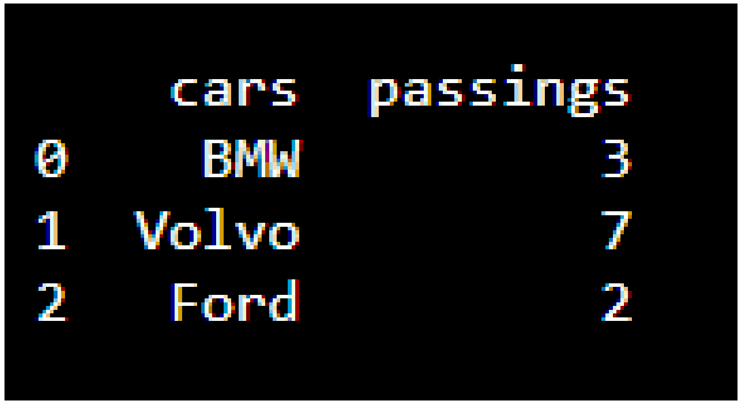
Pandas gives you answers about the data. Like:

- Is there a correlation between two or more columns?
- What is average value?
- Max value?
- Min value?

Pandas are also able to delete rows that are not relevant, or contains wrong values, like empty or NULL values. This is called *cleaning* the data.

Python and libraries for machine learning

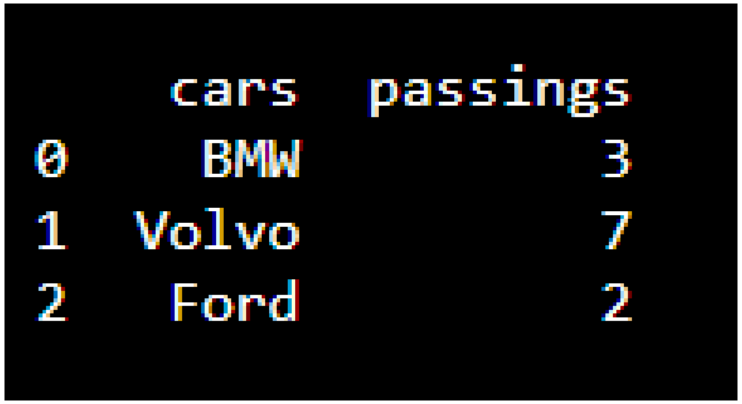
```
import pandas

import pandas

mydataset = {
  'cars': ["BMW", "Volvo", "Ford"],
  'passings': [3, 7, 2]
}

myvar = pandas.DataFrame(mydataset)

print(myvar)
```



```
    cars  passings
0    BMW         3
1  Volvo         7
2   Ford         2
```

Python and libraries for machine learning

```
import pandas

import pandas

mydataset = {
  'cars': ["BMW", "Volvo", "Ford"],
  'passings': [3, 7, 2]
}

myvar = pandas.DataFrame(mydataset)

print(myvar)
```



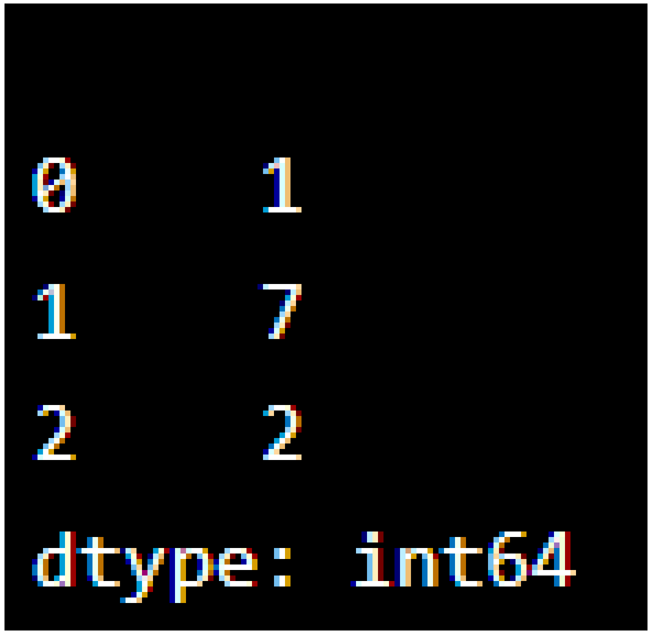
```
    cars  passings
0    BMW         3
1  Volvo         7
2   Ford         2
```

Python and libraries for machine learning

What is a Series?

- A Pandas Series is like a column in a table.
- It is a one-dimensional array holding data of any type.

```
import pandas as pd

a = [1, 7, 2]

myvar = pd.Series(a)

print(myvar)
```



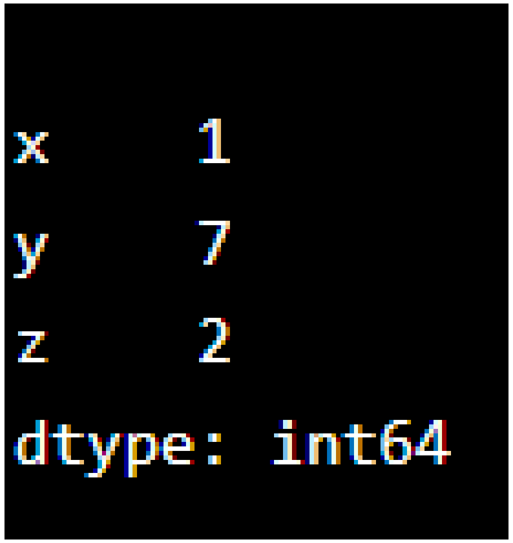
```
0     1
1     7
2     2
dtype: int64
```

Python and libraries for machine learning

Create Labels

With the index argument, you can name your own labels.

```
import pandas as pd

a = [1, 7, 2]

myvar = pd.Series(a, index = ["x", "y", "z"])

print(myvar)
```



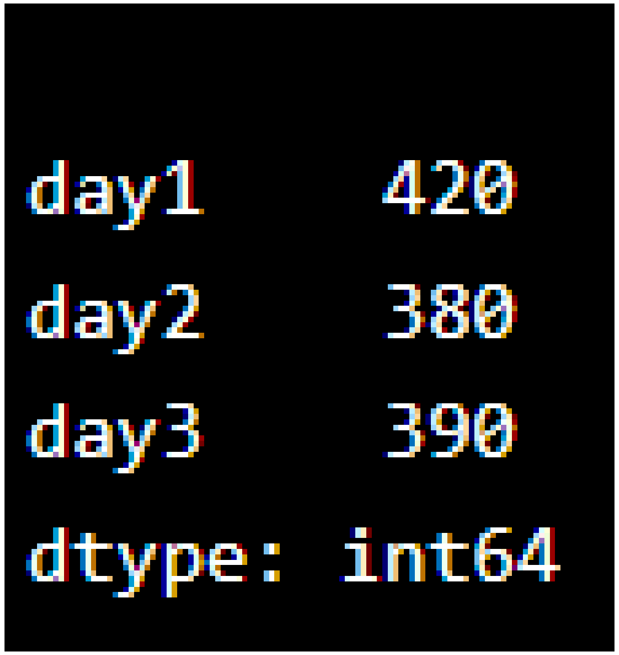
```
x     1
y     7
z     2
dtype: int64
```

Python and libraries for machine learning

Key/Value Objects as Series

- You can also use a key/value object, like a dictionary, when creating a Series.

```
import pandas as pd

calories = {"day1": 420, "day2": 380, "day3": 390}

myvar = pd.Series(calories)

print(myvar)
```



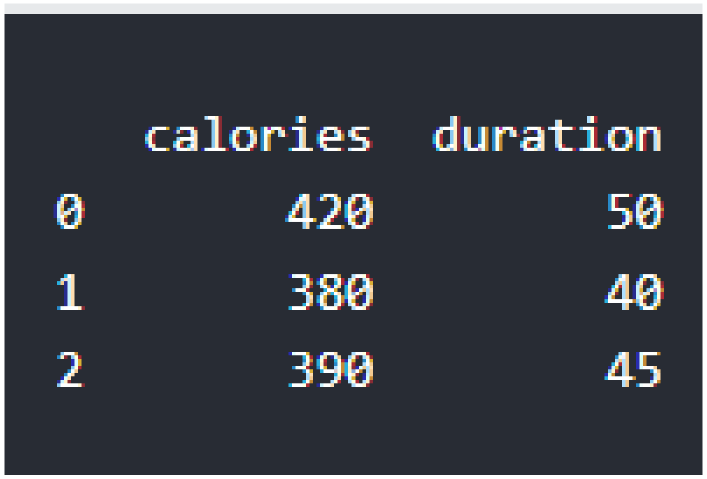
```
day1    420
day2    380
day3    390
dtype: int64
```

Python and libraries for machine learning

What is a DataFrame?

- A Pandas DataFrame is a 2 dimensional data structure, like a 2 dimensional array, or a table with rows and columns.

```
import pandas as pd

data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

#load data into a DataFrame object:
df = pd.DataFrame(data)

print(df)
```



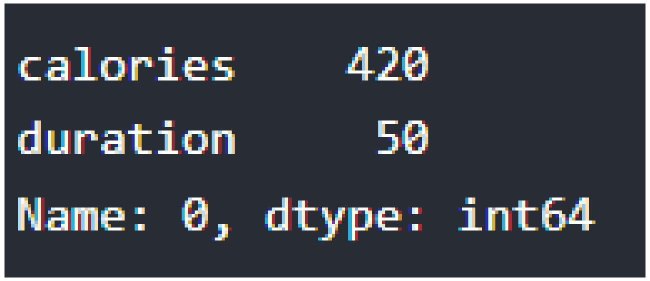
```
   calories  duration
0       420        50
1       380        40
2       390        45
```

Python and libraries for machine learning

Locate Row

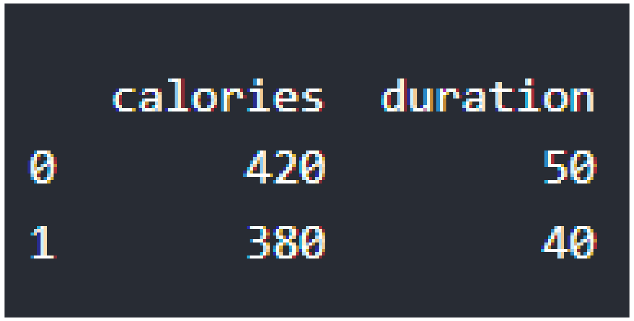
- As you can see from the result above, the DataFrame is like a table with rows and columns. Pandas use the loc attribute to return one or more specified row(s)

```
#refer to the row index:
print(df.loc[0]) #return row 0
```



```
calories    420
duration     50
Name: 0, dtype: int64
```

```
print(df.loc[[0, 1]]) #return row 0 and 1
```



```
   calories  duration
0       420        50
1       380        40
```

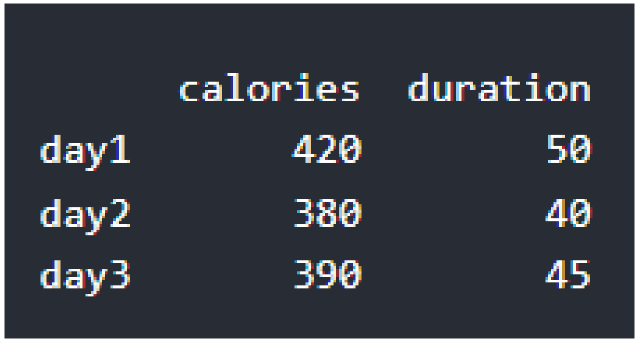
Python and libraries for machine learning

Named Indexes

- With the index argument, you can name your own indexes.

Add a list of names to give each row a name

```
import pandas as pd

data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

df = pd.DataFrame(data, index = ["day1", "day2", "day3"])

print(df)
```



```
      calories  duration
day1       420        50
day2       380        40
day3       390        45
```

Python and libraries for machine learning

Load Files Into a DataFrame

If your data sets are stored in a file, Pandas can load them into a DataFrame.

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df)
```

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.to_string()) # To read the entire dataframe
```

Python and libraries for machine learning

max_rows

- The number of rows returned is defined in Pandas option settings. You can check your system's maximum rows with the pd.options.display.max_rows statement.

```
import pandas as pd

print(pd.options.display.max_rows) #Check the number of maximum returned rows
```

Read JSON

- Big data sets are often stored, or extracted as JSON. JSON is plain text, but has the format of an object, and is well known in the world of programming, including Pandas. In our examples we will be using a JSON file called 'data.json'.

```
import pandas as pd

df = pd.read_json('data.json')

print(df.to_string())
```

Python and libraries for machine learning

Viewing the Data

- One of the most used method for getting a quick overview of the DataFrame, is the head() method. The head() method returns the headers and a specified number of rows, starting from the top.

Get a quick overview by printing the first 10 rows of the DataFrame

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.head(10))
```

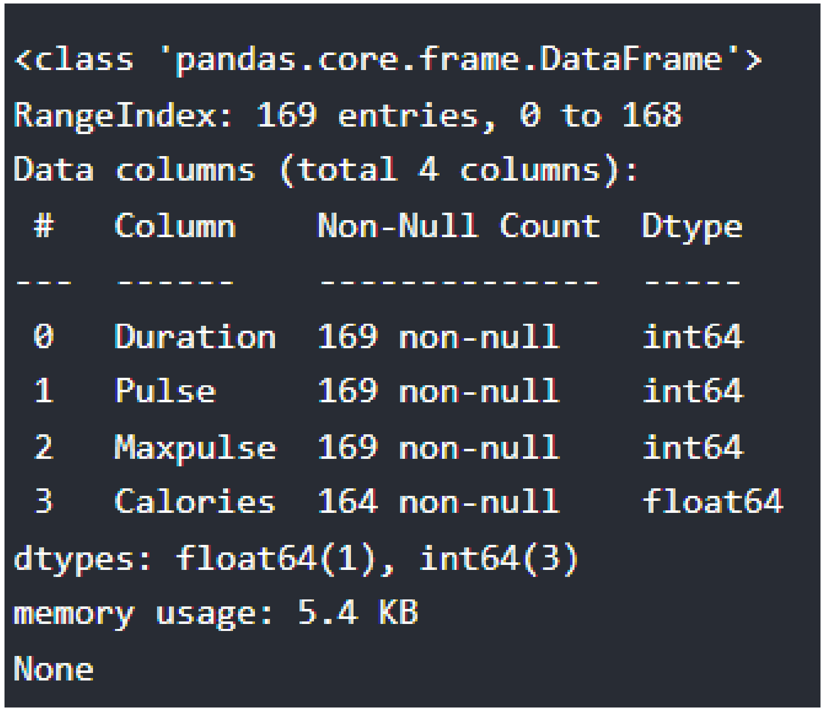
- There is also a tail() method for viewing the last rows of the DataFrame.
- The tail() method returns the headers and a specified number of rows, starting from the bottom.

Python and libraries for machine learning

Info About the Data

- The DataFrames object has a method called info(), that gives you more information about the data set.

```
print(df.info()) #print the information of the dataset
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 169 entries, 0 to 168
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Duration  169 non-null    int64  
 1   Pulse     169 non-null    int64  
 2   Maxpulse  169 non-null    int64  
 3   Calories  164 non-null    float64
dtypes: float64(1), int64(3)
memory usage: 5.4 KB
None
```

Python and libraries for machine learning

Null Values

- The info() method also tells us how many Non-Null values there are present in each column, and in our data set it seems like there are 164 of 169 Non-Null values in the "Calories" column. Which means that there are 5 rows with no value at all, in the "Calories" column, for whatever reason.
- Empty values, or Null values, can be bad when analyzing data, and you should consider removing rows with empty values. This is a step towards what is called cleaning data, and you will learn more about that in the next chapters.

Python and libraries for machine learning

Remove Rows

- One way to deal with empty cells is to remove rows that contain empty cells.
- This is usually OK, since data sets can be very big, and removing a few rows will not have a big impact on the result.

```
import pandas as pd

df = pd.read_csv('data.csv')

new_df = df.dropna()

print(new_df.to_string())
```

- By default, the dropna() method returns a new DataFrame, and will not change the original.

- If you want to change the original DataFrame, use the inplace = True argument:

Python and libraries for machine learning

```
import pandas as pd

df = pd.read_csv('data.csv')

df.dropna(inplace = True)

print(df.to_string())
```

- Now, the dropna(inplace = True) will NOT return a new DataFrame, but it will remove all rows containing NULL values from the original DataFrame.

Python and libraries for machine learning

Replace Empty Values

- Another way of dealing with empty cells is to insert a new value instead.
- This way you do not have to delete entire rows just because of some empty cells.
- The fillna() method allows us to replace empty cells with a value

```
import pandas as pd

df = pd.read_csv('data.csv')

df.fillna(130, inplace = True)

df["Calories"].fillna(130, inplace = True)
```
#Replace NULL values in the "Calories" columns with the number 130

Python and libraries for machine learning

Replace Using Mean, Median, or Mode

- A common way to replace empty cells, is to calculate the mean, median or mode value of the column.
- Pandas uses the mean() median() and mode() methods to calculate the respective values for a specified column:

```
import pandas as pd

df = pd.read_csv('data.csv')

x = df["Calories"].mean()

df["Calories"].fillna(x, inplace = True)
```

Python and libraries for machine learning

- To discover duplicates, we can use the duplicated() method.
- The duplicated() method returns a Boolean values for each row

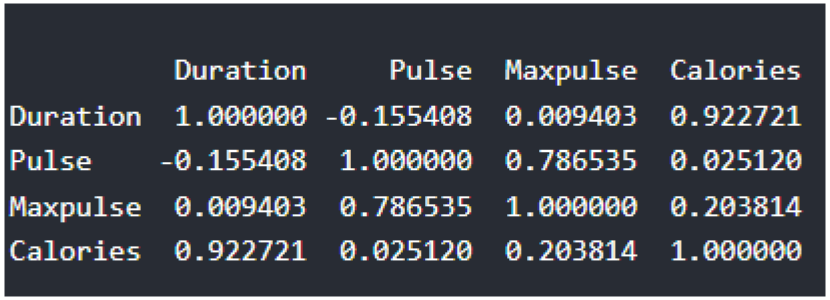
```
print(df.duplicated())

df.drop_duplicates(inplace = True) #drops duplicates
```

Finding Relationships

- A great aspect of the Pandas module is the corr() method.
- The corr() method calculates the relationship between each column in your data set.

```
df.corr()
```

- corr() method ignores "not numeric" columns.



```
          Duration     Pulse  Maxpulse  Calories
Duration  1.000000 -0.155408  0.009403  0.922721
Pulse    -0.155408  1.000000  0.786535  0.025120
Maxpulse  0.009403  0.786535  1.000000  0.203814
Calories  0.922721  0.025120  0.203814  1.000000
```

Python and libraries for machine learning

- The Result of the corr() method is a table with a lot of numbers that represents how well the relationship is between two columns.
- The number varies from -1 to 1.
- 1 means that there is a 1 to 1 relationship (a perfect correlation), and for this data set, each time a value went up in the first column, the other one went up as well.
- 0.9 is also a good relationship, and if you increase one value, the other will probably increase as well.
- -0.9 would be just as good relationship as 0.9, but if you increase one value, the other will probably go down.
- 0.2 means NOT a good relationship, meaning that if one value goes up does not mean that the other will.

Python and libraries for machine learning

Perfect Correlation:

We can see that "Duration" and "Duration" got the number 1.000000, which makes sense, each column always has a perfect relationship with itself.

Good Correlation:

"Duration" and "Calories" got a 0.922721 correlation, which is a very good correlation, and we can predict that the longer you work out, the more calories you burn, and the other way around: if you burned a lot of calories, you probably had a long work out.

Bad Correlation:

"Duration" and "Maxpulse" got a 0.009403 correlation, which is a very bad correlation, meaning that we can not predict the max pulse by just looking at the duration of the work out, and vice versa.

Python and libraries for machine learning

Plotting

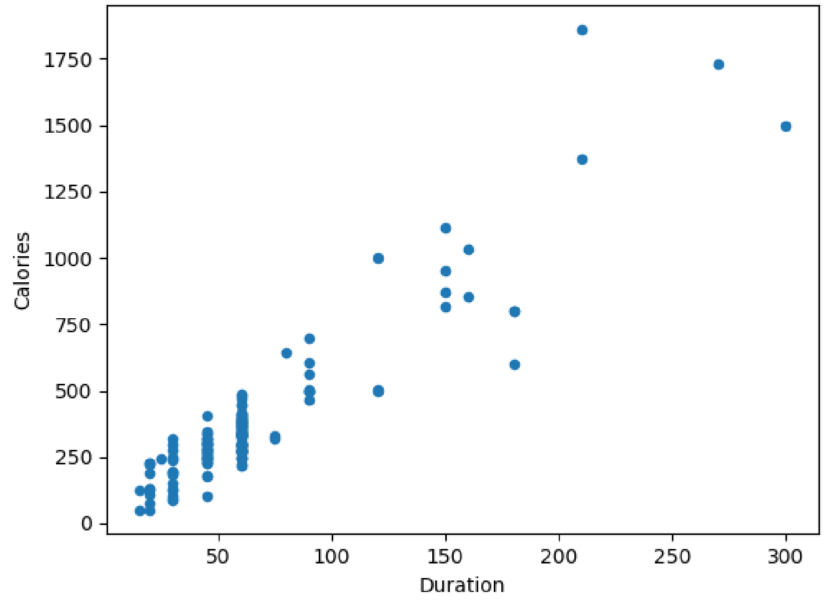
- Pandas uses the plot() method to create diagrams. We can use Pyplot, a submodule of the Matplotlib library to visualize the diagram on the screen.

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('data.csv')

df.plot()

plt.show()
```

Python and libraries for machine learning

Scatter Plot

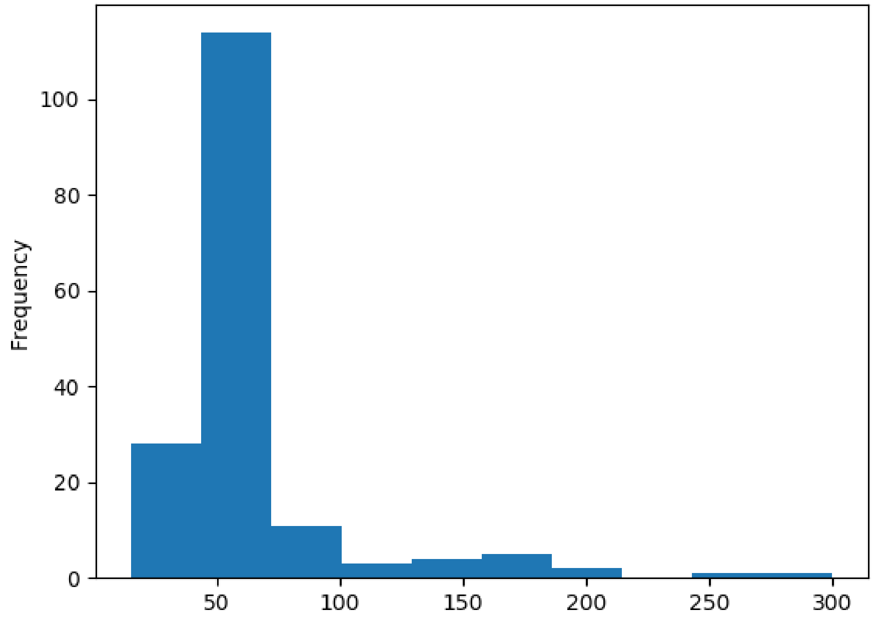
```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('data.csv')

df.plot(kind = 'scatter', x = 'Duration', y = 'Calories')

plt.show()
```



Python and libraries for machine learning

Histogram Plot

```
df["Duration"].plot(kind = 'hist')
```



Python and libraries for machine learning

What is NumPy?

- NumPy is a Python library used for working with arrays.
- It also has functions for working in domain of linear algebra, fourier transform, and matrices.
- NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python.

Python and libraries for machine learning

Why Use NumPy?

- In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.
- The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.
- Arrays are very frequently used in data science, where speed and resources are very important.

Why is NumPy Faster Than Lists?

- NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently.
- This behavior is called **locality of reference** in computer science. This is the main reason why NumPy is faster than lists. Also it is optimized to work with latest CPU architectures.

Python and libraries for machine learning

```
import numpy

arr = numpy.array([1, 2, 3, 4, 5])

print(arr)

# 0D array

import numpy as np

arr = np.array(42)

print(arr)

#1D array

import numpy as np

arr = np.array([1, 2, 3, 4, 5])

print(arr)
```

Python and libraries for machine learning

```
#2D array

import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]])

print(arr)

# checking the dimension of ndarray

import numpy as np

a = np.array(42)
b = np.array([1, 2, 3, 4, 5])
c = np.array([[1, 2, 3], [4, 5, 6]])
d = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])

print(a.ndim)
print(b.ndim)
print(c.ndim)
print(d.ndim)
```

Python and libraries for machine learning

```python
import numpy as np

arr = np.array([1, 2, 3, 4], ndmin=5)

print(arr)
print('number of dimensions :', arr.ndim)
```

#Create an array with 5 dimensions and verify that it has 5 dimensions

In this array the innermost dimension (5th dim) has 4 elements, the 4th dim has 1 element that is the vector, the 3rd dim has 1 element that is the matrix with the vector, the 2nd dim has 1 element that is 3D array and 1st dim has 1 element that is a 4D array.

Python and libraries for machine learning

Array operations

```
import numpy as np

arr = np.array([1, 2, 3, 4])

print(arr[2] + arr[3])
```

Accessing element in 2D array:

```
import numpy as np

arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])

print('2nd element on 1st row: ', arr[0, 1])
```

#Print the last element from the 2nd dim

```
import numpy as np

arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])

print('Last element from 2nd dim: ', arr[1, -1])
```

Python and libraries for machine learning

Slicing array:

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[-3:-1])
```

#Slice from the index 3 from the end to index 1 from the end

```
import numpy as np

arr = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])

print(arr[1, 1:4])
```

Python and libraries for machine learning

Data Types in NumPy

NumPy has some extra data types, and refer to data types with one character, like i for integers, u for unsigned integers etc.

Below is a list of all data types in NumPy and the characters used to represent them.

i - integer

b - boolean

u - unsigned integer

f - float

c - complex float

m - timedelta

Python and libraries for machine learning

M - datetime

O - object

S - string

U - unicode string

V - fixed chunk of memory for other type (void)

```
import numpy as np

arr = np.array(['apple', 'banana', 'cherry'])

print(arr.dtype)
```

The Difference Between Copy and View

- The main difference between a copy and a view of an array is that the copy is a new array, and the view is just a view of the original array.

- The copy *owns* the data and any changes made to the copy will not affect original array, and any changes made to the original array will not affect the copy.

- The view *does not own* the data and any changes made to the view will affect the original array, and any changes made to the original array will affect the view.

Python and libraries for machine learning

```
#Copy

import numpy as np

arr = np.array([1, 2, 3, 4, 5])
x = arr.copy()
arr[0] = 42

print(arr)
print(x)

#View

import numpy as np

arr = np.array([1, 2, 3, 4, 5])
x = arr.view()
arr[0] = 42

print(arr)
print(x)
```

Python and libraries for machine learning

Reshaping arrays

- Reshaping means changing the shape of an array.

- The shape of an array is the number of elements in each dimension.
- By reshaping we can add or remove dimensions or change number of elements in each dimension

- **Convert the following 1-D array with 12 elements into a 2-D array.**
- **The outermost dimension will have 4 arrays, each with 3 elements:**

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])

newarr = arr.reshape(4, 3)

print(newarr)
```

Loss Functions

Mean Squared Error (MSE)

- Fundamental concept in statistics and machine learning playing a crucial role in **assessing the accuracy** of the predictive models.
- It is a parameter to calculate the accuracy of the model.
- It measures the **average squared difference between predicted values and the actual values** in the dataset.
- **Mean squared error (MSE)** is a metric used to measure the average squared difference between the predicted values and the actual values in the dataset.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Loss Functions

- n is the number of observations in the dataset.
- y_i is the actual value of the observation.
- $y_i^{\wedge}$ is the predicted value of the ith observation.

Interpretation of Mean Squared Error

The Interpreting MSE involves understanding the magnitude of the error and its implications for model's performance.

- A lower MSE indicates that the model's predictions are closer to the actual values signifying better accuracy.
- Conversely, a higher MSE suggests that the model's predictions deviate further from true values indicating the poorer performance.

Loss Functions

Significance of Mean Squared Error

The Mean Squared Error is widely used in the various fields including the statistics, machine learning and econometrics due to its several important properties:

- It provides the quantitative measure of the accuracy of the predictive models.

- It penalizes large errors more heavily than small errors making it sensitive to the outliers.

- It is mathematically convenient and easy to the interpret making it a preferred choice for the evaluating model performance.

Loss Functions

Applications of Mean Squared Error

The Mean Squared Error is extensively used in the various applications including:

- **Regression analysis**: Assessing the goodness of fit of the regression models.
- **Model evaluation**: Comparing the performance of the different machine learning algorithms.
- **Optimization**: Minimizing MSE during the model training to the improve predictive accuracy.

Loss Functions

Advantages and Limitations of Mean Squared Error

The advantages and limitations of mean squared error is mentioned below:

Advantages

- Provides the comprehensive measure of the model accuracy.
- Sensitive to the both large and small errors.
- Easy to the calculate and interpret.

Limitations

- It can be heavily influenced by the outliers.
- It penalizes large errors disproportionately which may not always be desirable.

Loss Functions

Month	Actual	Predicted
January	42	46
February	51	48
March	53	55
April	68	73
May	74	77
June	81	83
July	88	87
August	85	85
September	79	75
October	67	70
November	58	55
December	43	41

$$\text{MSE} = \frac{\Sigma(y_i - p_i)^2}{n} = \frac{106}{12} = 8.83.$$

Loss Functions

y_i	y^*
49	47.48
63	62.6
58	53.15
60	62.6
58	64.49
61	60.71
60	60.71
63	55.04
60	55.04
52	49.37
62	64.49
30	39.92
59	64.49
49	49.37
68	62.6

$$\text{RMSE}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i-\hat{y}_i\right)^2}$$

Loss Functions

y_i	y^*	y_i-y^*	Square of (y_i-y^*)
49	47.48	1.52	**2.31**
63	62.6	0.4	**0.16**
58	53.15	4.2	**17.64**
60	62.6	-2.6	**6.76**
58	64.49	-6.49	**42.12**
61	60.71	0.29	**0.08**
60	60.71	-0.71	**0.50**
63	55.04	7.96	**63.36**
60	55.04	4.96	**24.60**
52	49.37	2.63	**6.92**
62	64.49	-2.49	**6.2**
30	39.92	-9.92	**98.41**
59	64.49	-5.49	**30.14**
49	49.37	-0.37	**0.14**
68	62.6	5.4	**29.16**
		SSE = 328.51 **MSE= 328.51/15=21.90**	

Loss Functions

What is Mean Absolute Error (MAE)?

Mean Absolute Error calculates the average difference between the calculated values and actual values. It is also known as scale-dependent accuracy as it calculates error in observations taken on the same scale used to predict the accuracy of the machine learning model

$$\text{MAE}= \quad \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|$$

Loss Functions

Mean Absolute Error (MAE)

Normalized MAE = MAE / (max value – min value)= 3.69/(9.92-0.29)= 0.38

Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the **RMSE should be more useful when large errors are particularly undesirable**. Both ranges from 0 to infinity.

y_i	y^*	$\lvert y_i - y^* \rvert$
49	47.48	1.52
63	62.6	0.4
58	53.15	4.2
60	62.6	2.6
58	64.49	6.49
61	60.71	0.29
60	60.71	0.71
63	55.04	7.96
60	55.04	4.96
52	49.37	2.63
62	64.49	2.49
30	39.92	9.92
59	64.49	5.49
49	49.37	0.37
68	62.6	5.4
		AE = 55.43 **MAE= 55.43/15=3.69**

Huber's Loss Functions

Huber Loss is a loss function used in regression problems. It is robust to outliers and combines the best properties of both Mean Squared Error (MSE) and Mean Absolute Error (MAE). The loss function is defined as:

Key Features:

- Quadratic for small errors (|a| ≤ δ): Acts like MSE, giving a smooth gradient and is sensitive to small errors.
- Linear for large errors (|a| > δ): Acts like MAE, reducing the impact of outliers by not letting the error grow quadratically.

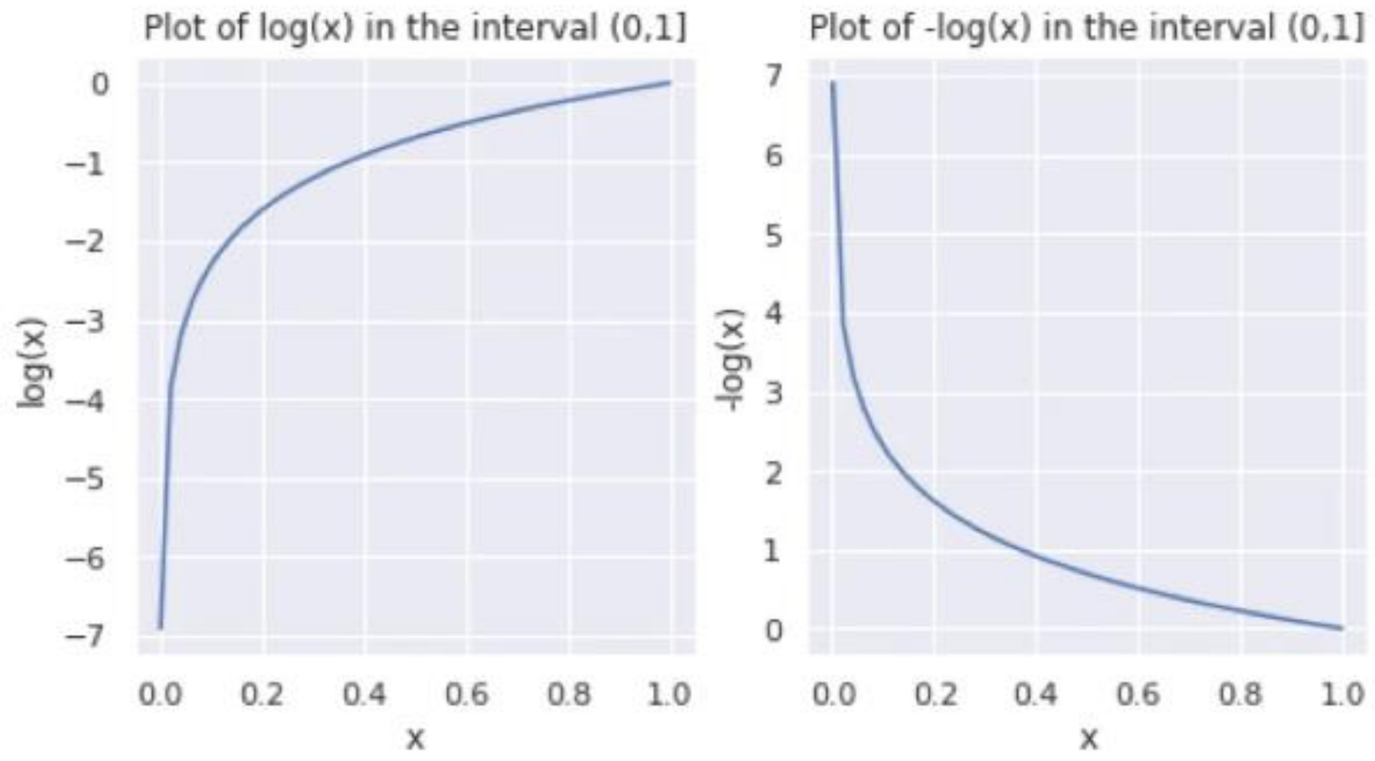
$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| \le \delta, \\ \delta(|a| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases}$$

Loss Functions

Categorical cross entropy:

```
import tensorflow as tf
import numpy as np

# True labels (one-hot encoded)
y_true = np.array([[0, 1, 0], [1, 0, 0], [0, 0, 1]])

# Predicted probabilities
y_pred = np.array([[0.1, 0.8, 0.1], [0.7, 0.2, 0.1], [0.2, 0.3, 0.5]])

# Categorical Cross-Entropy loss calculation
loss = tf.keras.losses.categorical_crossentropy(y_true, y_pred)

print("Loss:", loss.numpy())
```

Loss Functions



Why One Hot Encoding

Explicit Representation of Class Membership:

- **One-hot encoding** clearly shows which class a data point belongs to. Each class has a unique position in the vector, making the membership explicit.
- This representation is particularly useful in situations where the model or algorithm requires all classes to be represented equally in the input.

Compatibility with Some Algorithms:

- Some machine learning algorithms, like certain neural network architectures, expect input in a particular format, often as vectors. One-hot encoding ensures that the target labels meet these input requirements.
- For example, when computing dot products or performing matrix operations, one-hot encoded vectors work seamlessly.

Why One Hot Encoding

Interpretability:

- One-hot encoded vectors make it easier to interpret which class the model is predicting or working with because each dimension corresponds to a specific class.
- This can be beneficial in debugging or understanding the model's predictions.

Flexibility in Loss Functions

- While **Sparse Categorical Cross Entropy** simplifies the process by using integer labels, **Categorical Cross Entropy** with one-hot encoded labels provides flexibility for different scenarios, like when modifying or creating custom loss functions.

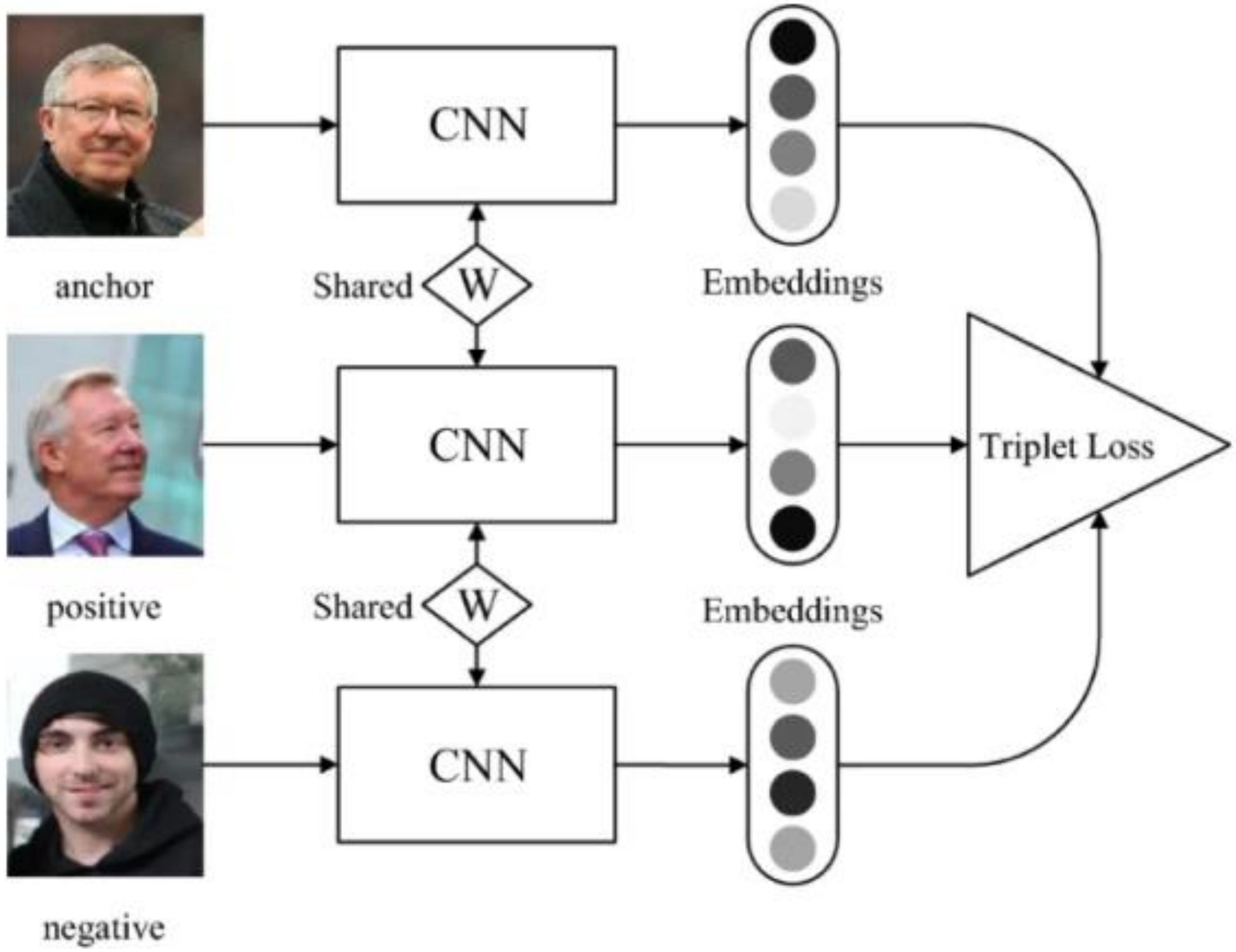
Triplet Loss

- Triplet loss is a way to teach a machine-learning model how to recognize the similarity or differences between items. It uses groups of three items, called triplets, which consist of an anchor item, a similar item (positive), and a dissimilar item (negative).

- The goal is to make the model understand that the anchor is closer to the positive than the negative item. This helps the model distinguish between similar and dissimilar items more effectively.

- In face recognition, for example, the model compares two unfamiliar faces and determines if they belong to the same person.

Triplet Loss



Triplet Loss

This scenario uses triplet loss to learn embeddings for every face. Faces from the same individual should be close together again and form well-separated clusters in the embedding space.

The objective of triplet loss is to build a representation space where the gap between similar samples is smaller than between different examples. By enforcing the order of distances, triplet loss models are embedded so that samples with identical labels appear nearer than those with other labels.

Hence, the triplet loss architecture helps us learn distributed embedding through the concept of similarity and dissimilarity. The mathematical depiction is shown below:

$$\sum_{i}^{N}[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \propto]$$

Triplet Loss

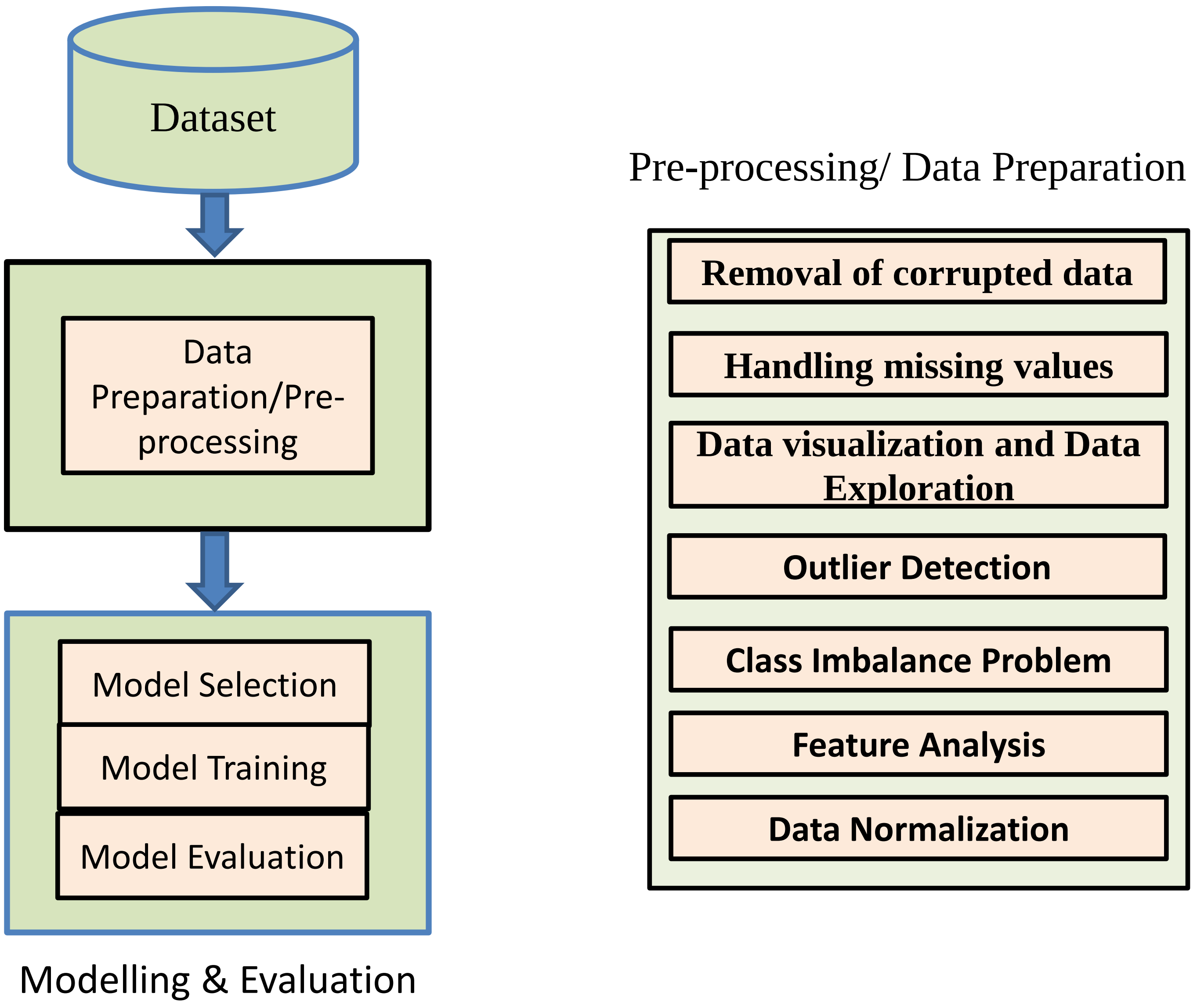
Where

- $f(x)$ accepts an input of x and generates a 128-dimensional vector w
- I represents the i'th input
- The subscript a denotes an anchor image, p is a positive image, and n is a negative image
- α refers to the bias

The goal is to minimize the above equation by minimizing the first term and maximizing the second term, and bias acts as a threshold.

An anchor (with fixed identity) negative is an image that doesn't share the class with the anchor—so, with a greater distance. In contrast, a **positive** is a point closer to the anchor, displaying a similar image. The model attempts to diminish the difference between similar classes while increasing the difference between different classes.

ML Model


Dataset
Data Preparation/Pre-processing
Model Selection
Model Training
Model Evaluation
Modelling & Evaluation
Pre-processing/ Data Preparation
Removal of corrupted data
Handling missing values
Data visualization and Data Exploration
Outlier Detection
Class Imbalance Problem
Feature Analysis
Data Normalization

ML Model

Pre-processing/ Data Preparation

- **Removal of corrupted data**
- **Handling missing values**
- **Data visualization and Data Exploration**
- **Outlier Detection**
- **Class Imbalance Problem**
- **Feature Analysis**
- **Data Normalization**

Improves quality of the training and reliability

Removal of erroneous data: not in format, not in range, outliers, missing values

A	B	C	D	E

$\rightarrow$

A	B	C	D

Age	weight	Height	pressure	Heart Beat	class
25	54	5.6	110	300	Y
30	64	5.7	130	400	N

Age	weight	Height	pressure	class
25	54	5.6	110	Y
30	64	5.7	130	N

ML Model

Pre-processing/ Data Preparation

- Removal of corrupted data
- Handling missing values
- Data visualization and Data Exploration
- Outlier Detection
- Class Imbalance Problem
- Feature Analysis
- Data Normalization

What is a Missing Value?

How is it created ?

BP	Heart Beat	Weight	Class
100	70	50	Y
101		60	Y
102	59		N
120		70	N
120	66	85	N
124	75	82	N
154	76		Y
124	56	81	Y
115	70	73	Y

ML Model

Pre-processing/ Data Preparation

- Removal of corrupted data
- Handling missing values
- Data visualization and Data Exploration
- Outlier Detection
- Class Imbalance Problem
- Feature Analysis
- Data Normalization

Type of Missing values:

- **MCAR: Missing Completely At random**
- **MAR: Missing At Random**
- **MNAR: Missing Not At Random**

MCAR: Missing Completely At random

- **If there is no relationship among the missing data and any other variable of the dataset**
- **Probability of missing is not related with any other variable.**

Roll. No	Due book
120	05
101	
102	03
112	09
105	02

ML Model

Pre-processing/ Data Preparation

- Removal of corrupted data
- Handling missing values
- Data visualization and Data Exploration
- Outlier Detection
- Class Imbalance Problem
- Feature Analysis
- Data Normalization

Type of Missing values:

- **MCAR: Missing Completely At random**
- **MAR: Missing At Random**
- **MNAR: Missing Not At Random**

MAR: Missing At Random

If there is a relationship among the missing data and any other variable of the dataset. Therefore need to analyze the relationship between the missing data and the variable on which it depends upon.

If the probability of being missing is the same only within groups defined by the *observed* data, then the data are missing at random (MAR). MAR is a much broader class than MCAR

Roll. No	Due book	Sex
120	05	M
101		F
102	03	F
112	09	M

Roll. No	Due book	Sex
100		M
103		M
115	03	F
111	09	F

ML Model

Pre-processing/ Data Preparation

- Removal of corrupted data
- Handling missing values
- Data visualization and Data Exploration
- Outlier Detection
- Class Imbalance Problem
- Feature Analysis
- Data Normalization

Type of Missing values:

- **MCAR: Missing Completely At random**
- **MAR: Missing At Random**
- **MNAR: Missing Not At Random**

MNAR: Missing Not At Random

There is a relationship between the missing data and the variable itself in which the data is missing. Required to proper understanding about the variable before making any imputation.

Year	No. Population
2005	1000
2006	1100
2007	1300
2008	
2009	1600

Year	No. Population
2010	1800
2011	2000
2012	2300
2013	
2014	2800

ML Model

Pre-processing/ Data Preparation

- Removal of corrupted data
- Handling missing values
- Data visualization and Data Exploration
- Outlier Detection
- Class Imbalance Problem
- Feature Analysis
- Data Normalization

Improves quality of the training

Handling Missing Values:

1. Deletion

2. Data Imputation

BP	Heart Beat	Weight	Class
100	70	50	Y
101		60	Y
102	59		N
120		70	N
120	66	85	N
124	75	82	N
154	76		Y
124	56	81	Y
115	70	73	Y

Handling Missing Values

BP	Heart Beat	Weight	Class
100	70	50	Y
101	65	60	Y
102	59	52	N
120		70	N
120	66	85	N
124	75	82	N
154	76		Y
124	56	81	Y
115	70	73	Y

Deletion:

Deletion methods are used when missing is occurred due to “missing completely at random” and “missing at random”

1. **Deleting Rows**
2. **Deleting Columns**
3. **Pairwise**

Handling Missing Values

Deletion:

1. **Deleting Rows**
2. **Deleting Columns**
3. **Pairwise**

BP	Heart Beat	Weight	Class
100	70	50	Y
101	65	60	Y
102	59	52	N
120		70	N
120	66	85	N
124	75	82	N
154	76	52	Y
124	56	81	Y
115	70	73	Y

BP	Heart Beat	Weight	Class
100	70	50	Y
101	65	60	Y
102	59	52	N
120	66	85	N
124	75	82	N
154	76	52	Y
124	56	81	Y
115	70	73	Y

Handling Missing Values

BP	Heart Beat	Weight	Class
100	70	50	Y
101		60	Y
102	59	52	N
120		70	N
120	66	85	N
124	75	82	N
154	76		Y
124		81	Y
115	70	73	Y

BP	Weight	Class
100	50	Y
101	60	Y
102	52	N
120	70	N
120	85	N
124	82	N
154	52	Y
124	81	Y
115	73	Y

Deletion:

1. **Deleting Rows**
2. **Deleting Columns**
3. **Pairwise**

Handling Missing Values

Deletion:

1. **Deleting Rows**
2. **Deleting Columns**
3. **Pairwise**

BP	Heart Beat	Weight	Class
100	70	50	Y
101		60	Y
102	59	52	N
120		70	N
120	66	85	N
124	75	82	N
154	76		Y
124		81	Y
115	70	73	Y

BP	Weigh t	Class
100	50	Y
101	60	Y
102	52	N
120	70	N
120	85	N
124	82	N
124	81	Y
115	73	Y

Pairwise correlation between predictor and target is found to help in deletion

BP--Class-----------------High
Heart Beat—Class-----Low
Weight-Class------------High

Handling Missing Values

BP	Heart Beat	Weigh t	Class
100	70	50	Y
101		60	Y
102	59	52	N
120		70	N
120	66	85	N
124	75	82	N
154	76		Y
124		81	Y
115	70	73	Y

BP	Heart Beat	Class
100	70	Y
102	59	N
120	66	N
124	75	N
154	76	Y
115	70	Y

Deletion:

1. Deleting Rows
2. Deleting Columns
3. Pairwise

Pairwise correlation between predictor and target is found to help in deletion

BP--Class-----------------High
Heart Beat—Class-----High
Weight-Class------------Low

Handling Missing Values

Deletion:

Pros: Trained model becomes robust as all the missing values are deleted.

Cons: 1. loss of information and 2. trained model works poorly if deletion is excessive

Handling Missing Values

BP	Heart Beat	Weight	Class
100	70	50	Y
101	65	60	Y
102	59	52	N
120	67	70	N
120	66	85	N
124	75	82	N
154	76	52	Y
124	56	81	Y
115	70	73	Y

Imputation:

1. **Mean**
2. **Median**
3. **Mode**
4. **Linear Interpolation**
5. **Linear Regression**
6. **K-NN**

Different ML techniques

Mean=(70+65+59+66+75+76+56+70)/8
= 67.125
= 67

Handling Missing Values

BP	Heart Beat	Weight	Class
100	Y	50	Y
101	Y	60	Y
102	N	52	N
120	Y	70	N
120	Y	85	N
124	Y	82	N
154	Y	52	Y
124	Y	81	Y
115	N	73	Y

Imputation:

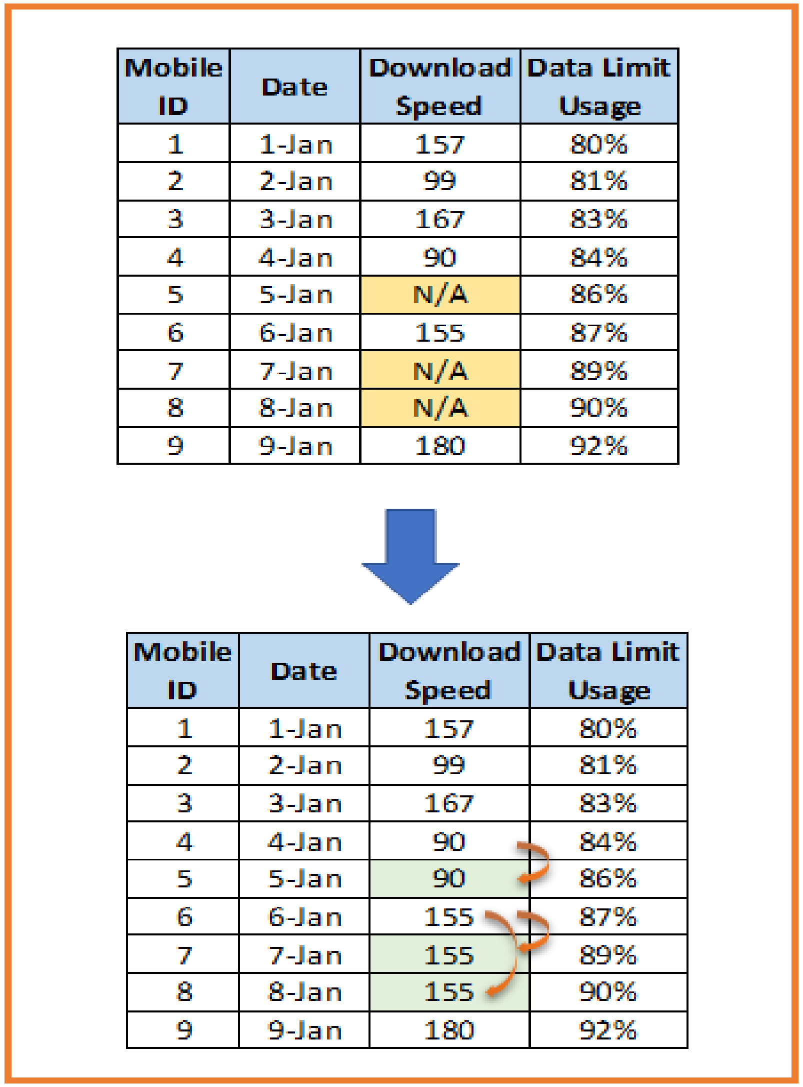
1. **Mean**
2. **Median**
3. **Mode**
4. **Linear Interpolation**
5. **Linear Regression (ML)**
6. **K-NN (ML)**
7. **SoftImpute (ML)**
8. **Mice (ML) (good)**
9. **MatrixFactorization (ML)**
10. **miss- Forest (ML)**
11. **Deductive Imputation (LR)**
12. Factor Analysis of Mixed Data (FAMD) (ML)

Categorical Data
K-NN is used value of k = 4

Handling Missing Values

Last Observation Carried Forward (LOCF)

If data is time-series data, one of the most widely used imputation methods is the last observation carried forward. Whenever a value is missing, it is replaced with the last observed value.



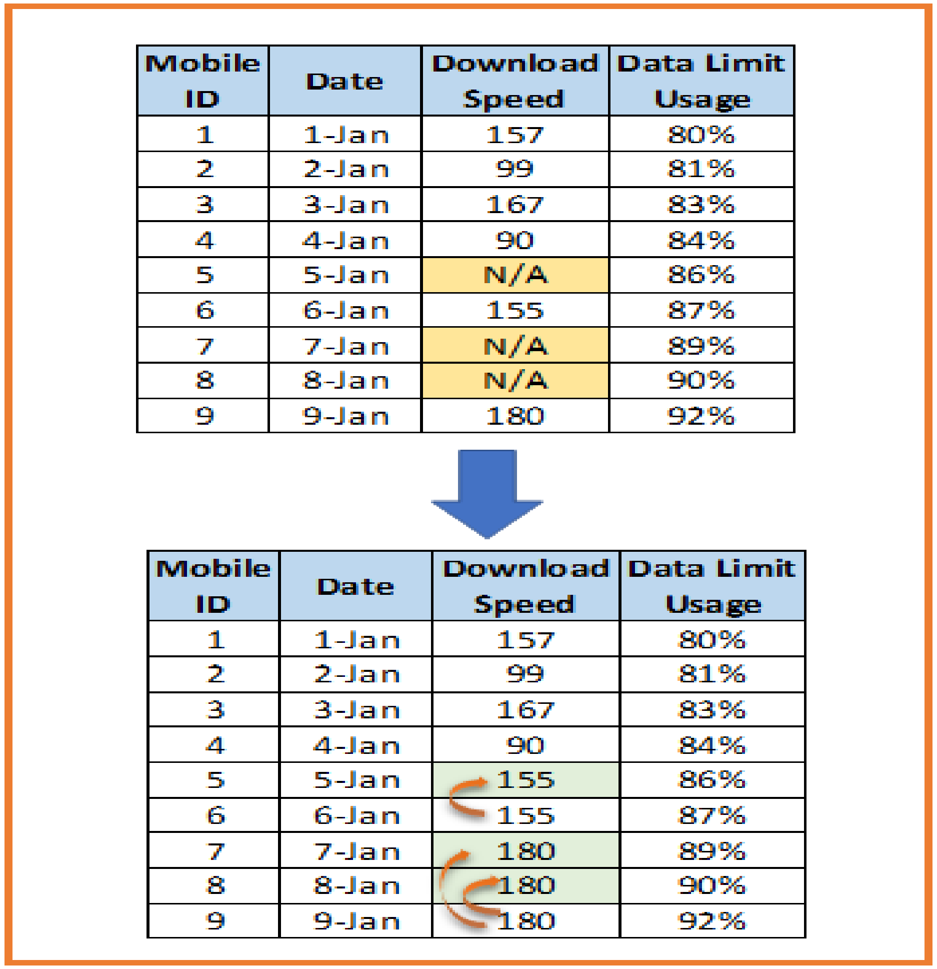
Mobile ID	Date	Download Speed	Data Limit Usage
1	1-Jan	157	80%
2	2-Jan	99	81%
3	3-Jan	167	83%
4	4-Jan	90	84%
5	5-Jan	N/A	86%
6	6-Jan	155	87%
7	7-Jan	N/A	89%
8	8-Jan	N/A	90%
9	9-Jan	180	92%

Mobile ID	Date	Download Speed	Data Limit Usage
1	1-Jan	157	80%
2	2-Jan	99	81%
3	3-Jan	167	83%
4	4-Jan	90	84%
5	5-Jan	90	86%
6	6-Jan	155	87%
7	7-Jan	155	89%
8	8-Jan	155	90%
9	9-Jan	180	92%

Handling Missing Values

Next Observation Carried Backward (NOCB)

It is a similar approach like LOCF which works oppositely by taking the first observation after the missing value and carrying it backward (“next observation carried backwards”, or NOCB).



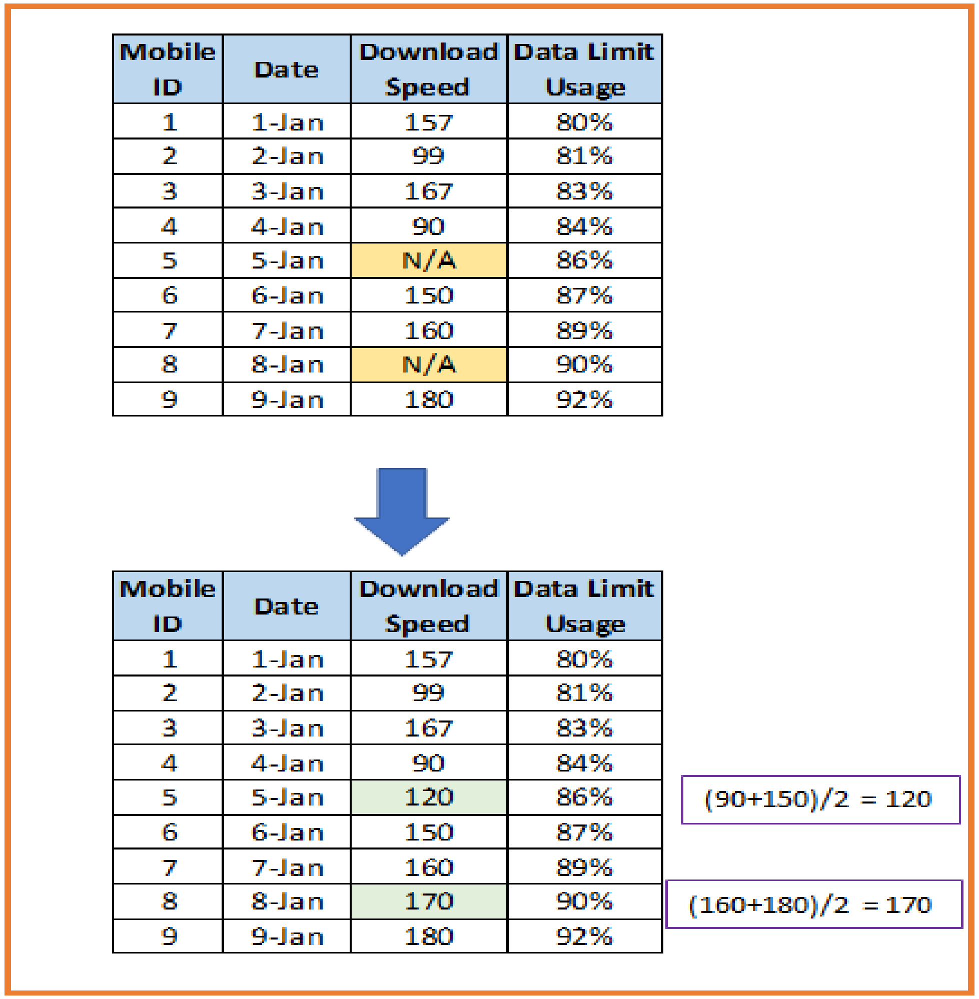
Mobile ID	Date	Download Speed	Data Limit Usage
1	1-Jan	157	80%
2	2-Jan	99	81%
3	3-Jan	167	83%
4	4-Jan	90	84%
5	5-Jan	N/A	86%
6	6-Jan	155	87%
7	7-Jan	N/A	89%
8	8-Jan	N/A	90%
9	9-Jan	180	92%

Mobile ID	Date	Download Speed	Data Limit Usage
1	1-Jan	157	80%
2	2-Jan	99	81%
3	3-Jan	167	83%
4	4-Jan	90	84%
5	5-Jan	155	86%
6	6-Jan	155	87%
7	7-Jan	180	89%
8	8-Jan	180	90%
9	9-Jan	180	92%

Handling Missing Values

Linear Interpolation

It is a mathematical method that adjusts a function to data and uses this function to extrapolate the missing data. **The simplest type of interpolation is linear interpolation, where the values before the missing data and after the same is used.**



Mobile ID	Date	Download Speed	Data Limit Usage
1	1-Jan	157	80%
2	2-Jan	99	81%
3	3-Jan	167	83%
4	4-Jan	90	84%
5	5-Jan	N/A	86%
6	6-Jan	150	87%
7	7-Jan	160	89%
8	8-Jan	N/A	90%
9	9-Jan	180	92%

Mobile ID	Date	Download Speed	Data Limit Usage
1	1-Jan	157	80%
2	2-Jan	99	81%
3	3-Jan	167	83%
4	4-Jan	90	84%
5	5-Jan	120	86%
6	6-Jan	150	87%
7	7-Jan	160	89%
8	8-Jan	170	90%
9	9-Jan	180	92%

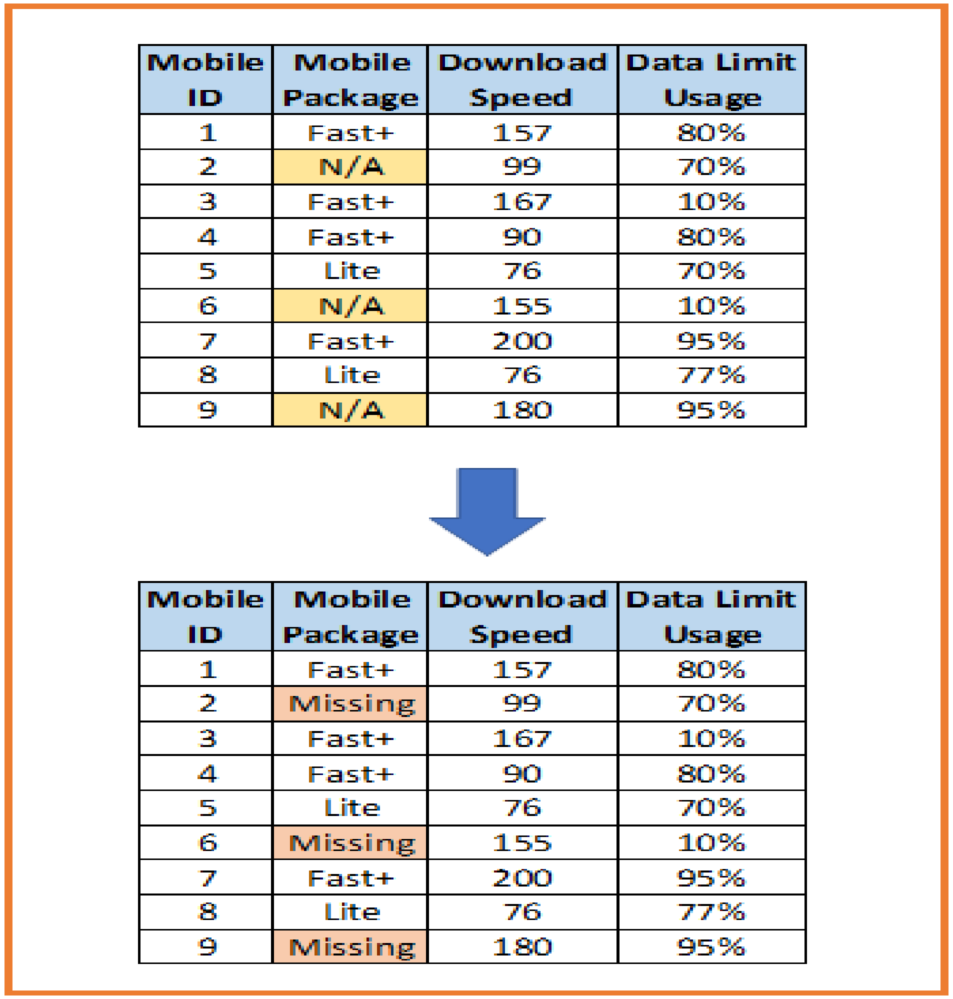
$(90+150)/2 = 120$

$(160+180)/2 = 170$

Handling Missing Values

Adding a category to capture NA

This is perhaps the most widely used method of missing data imputation for categorical variables. This method consists of treating missing data as an additional label or category of the variable.



Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	157	80%
2	N/A	99	70%
3	Fast+	167	10%
4	Fast+	90	80%
5	Lite	76	70%
6	N/A	155	10%
7	Fast+	200	95%
8	Lite	76	77%
9	N/A	180	95%

Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	157	80%
2	Missing	99	70%
3	Fast+	167	10%
4	Fast+	90	80%
5	Lite	76	70%
6	Missing	155	10%
7	Fast+	200	95%
8	Lite	76	77%
9	Missing	180	95%

Handling Missing Values

Imagine you have a categorical variable Education with the following categories:

•High School
•Bachelor's
•Master’s

Some values are missing (NaN). To handle this:

1.Replace the missing values with a new category, e.g., "Unknown".
2. Encode with level encoding or One-hot encoding

Handling Missing Values

Adding a category to capture NA

```
import pandas as pd

# Example dataset
data = {'Education': ['High School', 'Bachelor\'s', None,
'Master\'s', None]}
df = pd.DataFrame(data)

# Replace missing values with a new category
df['Education'] = df['Education'].fillna('Unknown')

print(df)
```

Handling Missing Values

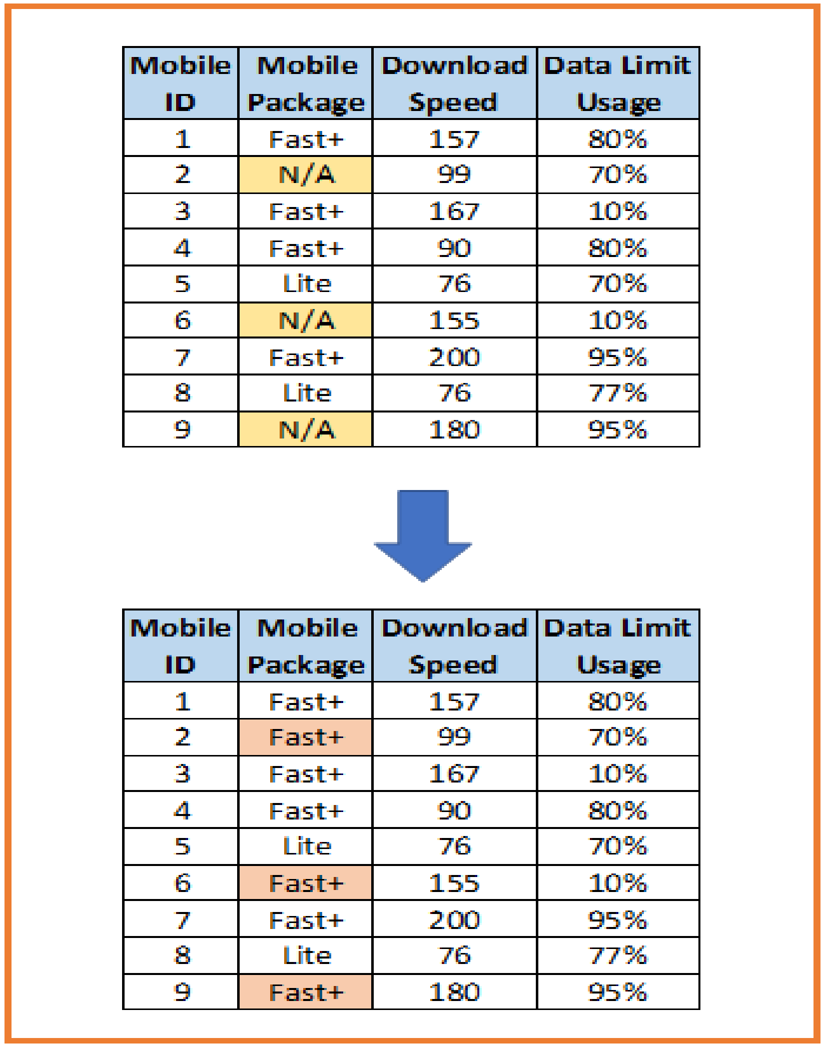
Adding a category to capture NA

```
from sklearn.preprocessing import LabelEncoder

# Label Encoding
label_encoder = LabelEncoder()
df['Education_Label'] =
label_encoder.fit_transform(df['Education'])
print(df)
```

Handling Missing Values

Frequent category imputation

Replacement of missing values by the most frequent category is the equivalent of mean/median imputation. It consists of replacing all occurrences of missing values within a variable with the variable's most frequent label or category.



Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	157	80%
2	N/A	99	70%
3	Fast+	167	10%
4	Fast+	90	80%
5	Lite	76	70%
6	N/A	155	10%
7	Fast+	200	95%
8	Lite	76	77%
9	N/A	180	95%

Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	157	80%
2	Fast+	99	70%
3	Fast+	167	10%
4	Fast+	90	80%
5	Lite	76	70%
6	Fast+	155	10%
7	Fast+	200	95%
8	Lite	76	77%
9	Fast+	180	95%

Missing Value Treatment using most recent data imputation techniques

MICE (Multiple Imputation by Chained Equation)

Handling Missing Values

Multiple Imputation

Multiple Imputation (MI) is a statistical technique for handling missing data.

The key concept of MI is to use the distribution of the observed data to estimate a set of plausible values for the missing data.

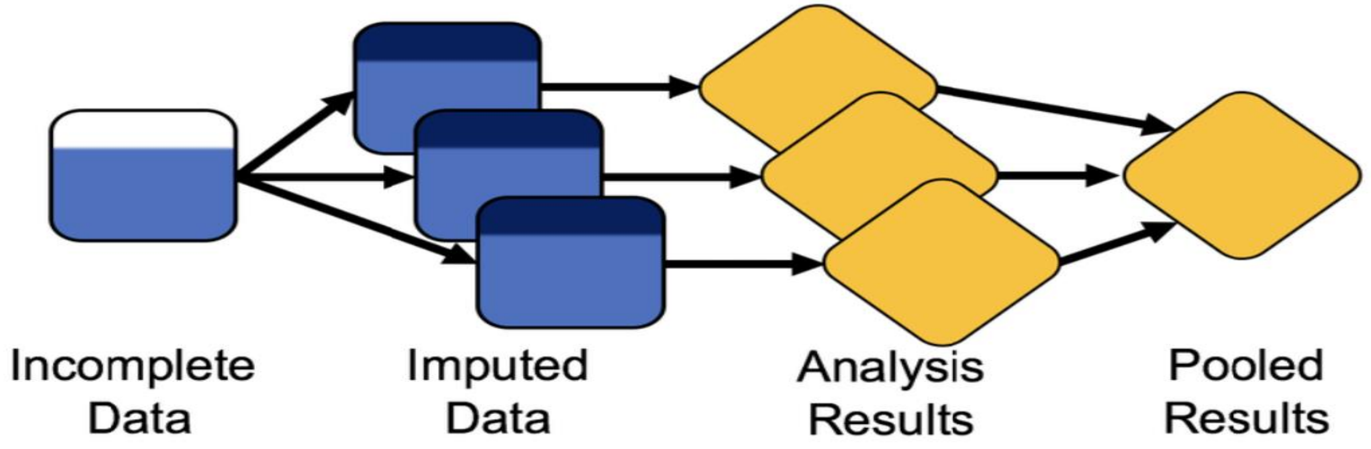
Estimates are combined to obtain a set of parameter estimates.

Multiple datasets are created and then analysed individually but identically to obtain a set of parameter estimates.

Multiple Imputation by Chained Equations (MICE) approach is a flexible way of handling more than one missing variable,

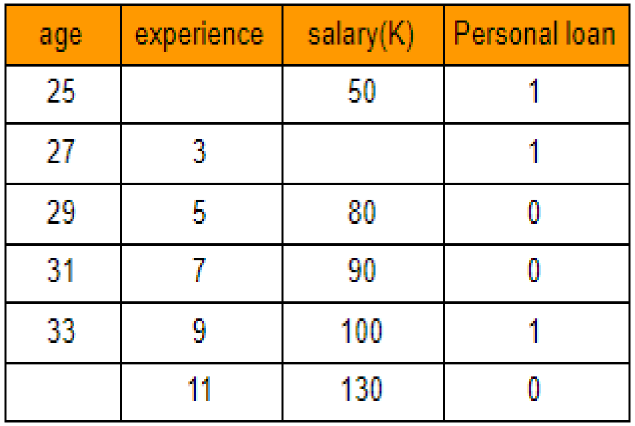
The benefit of the multiple imputations is to restore the natural variability of the missing values.

Multiple Imputation

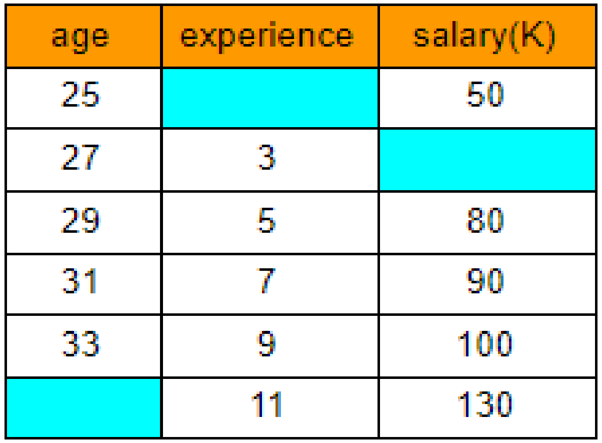


Multiple Imputation

First step would be to remove the "Personal Loan" column as it is the target column, we will not need this column for imputation.



age	experience	salary(K)	Personal loan
25		50	1
27	3		1
29	5	80	0
31	7	90	0
33	9	100	1
	11	130	0



age	experience	salary(K)
25	🟦	50
27	3	🟦
29	5	80
31	7	90
33	9	100
🟦	11	130

Multiple Imputation

Second step would be a simple imputation, such as imputing the mean, which is performed for every missing value in the dataset that leads to the formation of zeroth dataset.

age	experience	salary(K)
25		50
27	3	
29	5	80
31	7	90
33	9	100
	11	130

age	experience	salary(K)
25	7	50
27	3	90
29	5	80
31	7	90
33	9	100
29	11	130

zeroth dataset

Multiple Imputation

Third step would be to remove the "age" imputed values and keep the imputed values in other columns as shown here. Now, we will be imputing the columns from left to right.

age	experience	salary(K)
25	7	50
27	3	90
29	5	80
31	7	90
33	9	100
29	11	130

$\longrightarrow$

age	experience	salary(K)
25	7	50
27	3	90
29	5	80
31	7	90
33	9	100
	11	130

Multiple Imputation

In the fourth step, the remaining features and rows(top 5 rows of experience and salary) become the feature matrix (purple cells), "age" becomes the target variable(yellow cells).

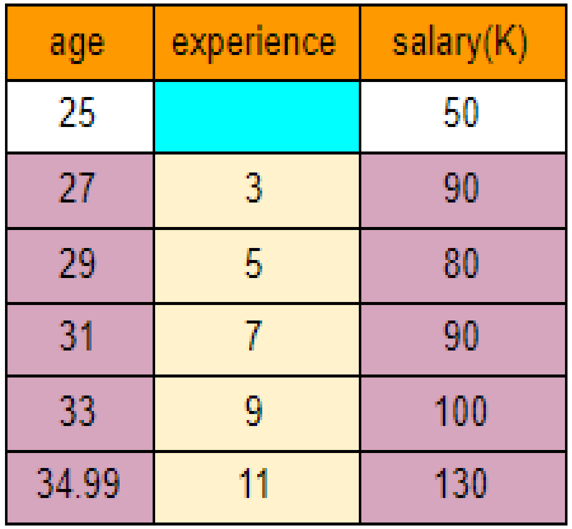
We will run the linear regression model on the fully filled rows with X= experience and salary and Y=age. To estimate the missing age, we will use the missing value row (white cells) as the test data. So, top 5 rows will be training data and the last row that has missing age will be test data. We will use (experience = 11 and salary = 130) to predict corresponding "age" value. When I did this, I found that my model predicted the age as 34.99.

age	experience	salary(K)
25	7	50
27	3	90
29	5	80
31	7	90
33	9	100
	11	130

Multiple Imputation

In the fifth step, we update the predicted age value in the missing cell in "age" column.

Now, remove "experience" imputed value. The remaining features and rows becomes the feature matrix(purple cells) and "experience" becomes the target variable(yellow cells). We will run the linear regression model on the fully filled rows with X= age and salary and Y=experience. To estimate the missing experience, we will use the missing value row (white cells) as the test data. The predicted value for experience is 0.98.



age	experience	salary(K)
25		50
27	3	90
29	5	80
31	7	90
33	9	100
34.99	11	130

Multiple Imputation

In the sixth step, we update the predicted experience value in the missing cell in "experience" column and remove "salary" imputed value.

The remaining features and rows becomes the feature matrix(purple cells) and "salary" becomes the target variable(yellow cells). We will run the linear regression model on the fully filled rows with X= age and experience and Y=salary. To estimate the missing salary, we will use the missing value row (white cells) as the test data. The predicted value for Salary is 70.

age	experience	salary(K)
25	0.98	50
27	3	
29	5	80
31	7	90
33	9	100
34.99	11	130

Multiple Imputation

Now we **impute the missing values in the original dataset** and the predicted values after 1st iteration is shown here.

Let's name this as **"First" dataset**.

age	experience	salary(K)
25	0.98	50
27	3	70
29	5	80
31	7	90
33	9	100
34.99	11	130

Multiple Imputation

In the seventh step, We will subtract the two datasets (zeroth and first). The resultant dataset is as below:

age	experience	salary(K)
25	7	50
27	3	90
29	5	80
31	7	90
33	9	100
29	11	130

minus

age	experience	salary(K)
25	0.98	50
27	3	70
29	5	80
31	7	90
33	9	100
34.99	11	130

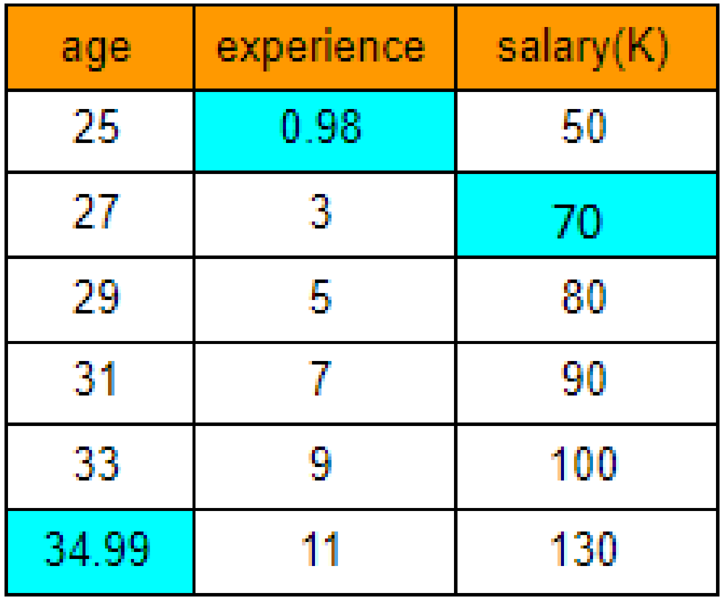
$\rightarrow$

age	experience	salary(K)
0	6.02	0
0	0	20
0	0	0
0	0	0
0	0	0
-5.99	0	0

If we observe, the absolute difference between 2 datasets are higher in few imputed values. Our aim is to reduce these differences close to 0. To achieve this we have to do many iterations. So, now we repeat the steps 2-6 with the new dataset (first), until we get a stable model. i.e. until the difference between the 2 latest imputed datasets becomes very small, close to 0. Technically, we stop the iterations when a pre-defined threshold is reached or a pre defined maximum number of iterations gets completed.

Multiple Imputation

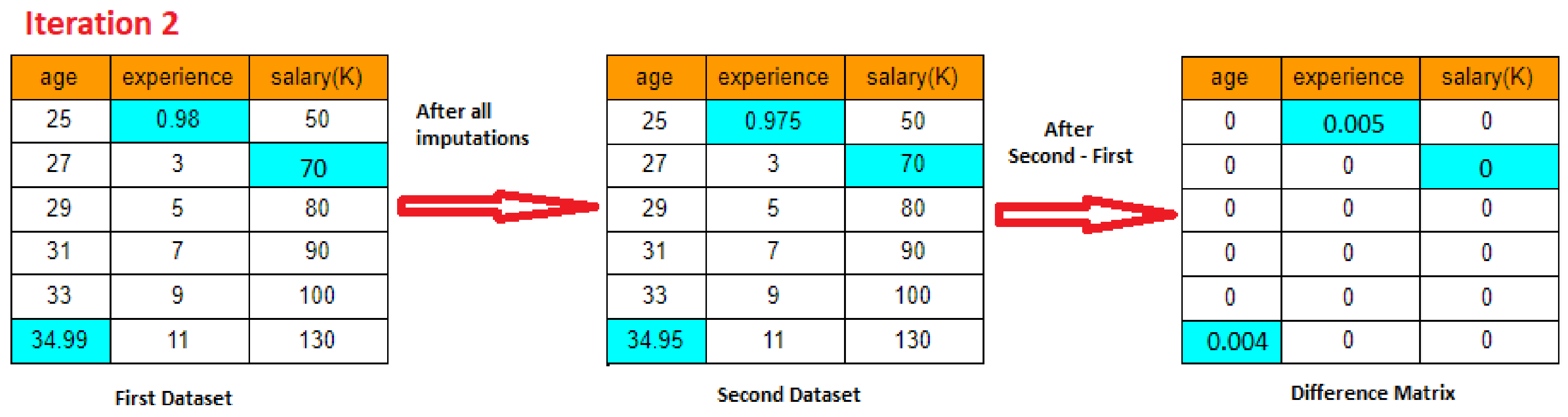
Now we will use the "first" dataset as our base dataset to do imputations, and discard the "Zeroth" dataset which had the mean imputations. With "first" dataset as base, let's perform all the previous steps and again predict the imputed values for the initial 3 missing values.



age	experience	salary(K)
25	0.98	50
27	3	70
29	5	80
31	7	90
33	9	100
34.99	11	130

Multiple Imputation

Here's is the iteration 2 values and the new dataset values are subtracted from the first dataset and got the difference matrix as below:



Iteration 2

age	experience	salary(K)
25	0.98	50
27	3	70
29	5	80
31	7	90
33	9	100
34.99	11	130

First Dataset

After all imputations

age	experience	salary(K)
25	0.975	50
27	3	70
29	5	80
31	7	90
33	9	100
34.95	11	130

Second Dataset

After Second - First

age	experience	salary(K)
0	0.005	0
0	0	0
0	0	0
0	0	0
0	0	0
0.004	0	0

Difference Matrix

Now, after second iteration, we can see that the difference is very negligible. We can either stop here as we almost got the same numbers, or proceed with next iteration until we get 0 difference.