

Artificial Intelligence

Dr. Panem Charanarur

1

Thinking Machines

- Are there tasks which cannot easily be automated? If so, what are the

limitations?

- How do computers abilities compare to that of humans?

What is an AI?



Computers versus humans

- A computer can do some things better than a human can
 - Adding a thousand four-digit numbers
 - Drawing complex, 3D images
 - Store and retrieve massive amounts of data
- However, there are things humans can do much better.

Thinking Machines



A computer would have difficulty identifying the cat, or matching it to another picture of a cat.

Computer or human?

- Which of the following occupations could (or should) be performed by computers? –

Postman

- Bookstore clerk
- Librarian
- Doctor
- Lawyer
- Judge
- Professor

Thinking Machines

Artificial intelligence (AI)

The study of computer systems that attempt to model and apply the intelligence of the human mind

For example, writing a program to pick out objects in a picture

7

First things first...

- Of course, first we have to understand why we use the term “intelligence” in regard to

humans.

- What defines “intelligence”?
- Why is it that we assume humans are intelligent? – Are monkeys intelligent? Dogs? Ants? Pine trees?

Early History

- In 1950 English mathematician Alan Turing wrote a landmark paper titled “Computing Machinery and Intelligence” that asked the question: “Can machines think?”
- Further work came out of a 1956 workshop at Dartmouth sponsored by John McCarthy. In the

proposal for that workshop, he coined the phrase a “study of artificial intelligence”

Can Machines Think?

- So Turing asked: “Can machines think?” He felt that such machines would eventually be constructed.
- But he also realized a bigger problem. How would we know if we’ve succeeded?

The Turing Test

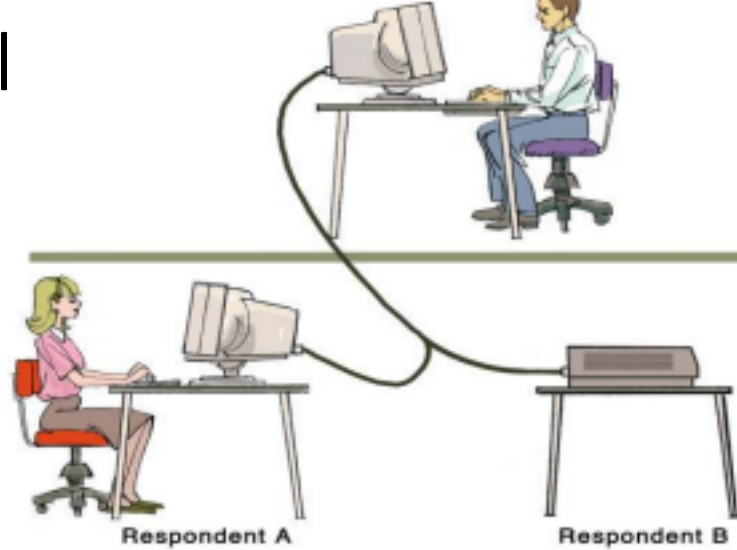
Turing test

A test to empirically determine whether a

computer has acl

Figure 13.2

In a Turing test, the interrogator must determine which respondent is the computer and which is the human



The Turing Test

- Passing the Turing Test does not truly show that the machine was thinking. It simply shows that it generated behavior consistent with thinking.

- weak equivalence:** the two systems (human and computer) are equivalent in **results** (output), but they do not necessarily arrive at those results in the same way
- Strong equivalence:** the two systems use the same internal processes to produce results

12

The Turing Test

Loebner prize

The first formal instantiation



of the Turing test, held
annually

Chatbots

A program designed to carry on a conversation
with a human user

13

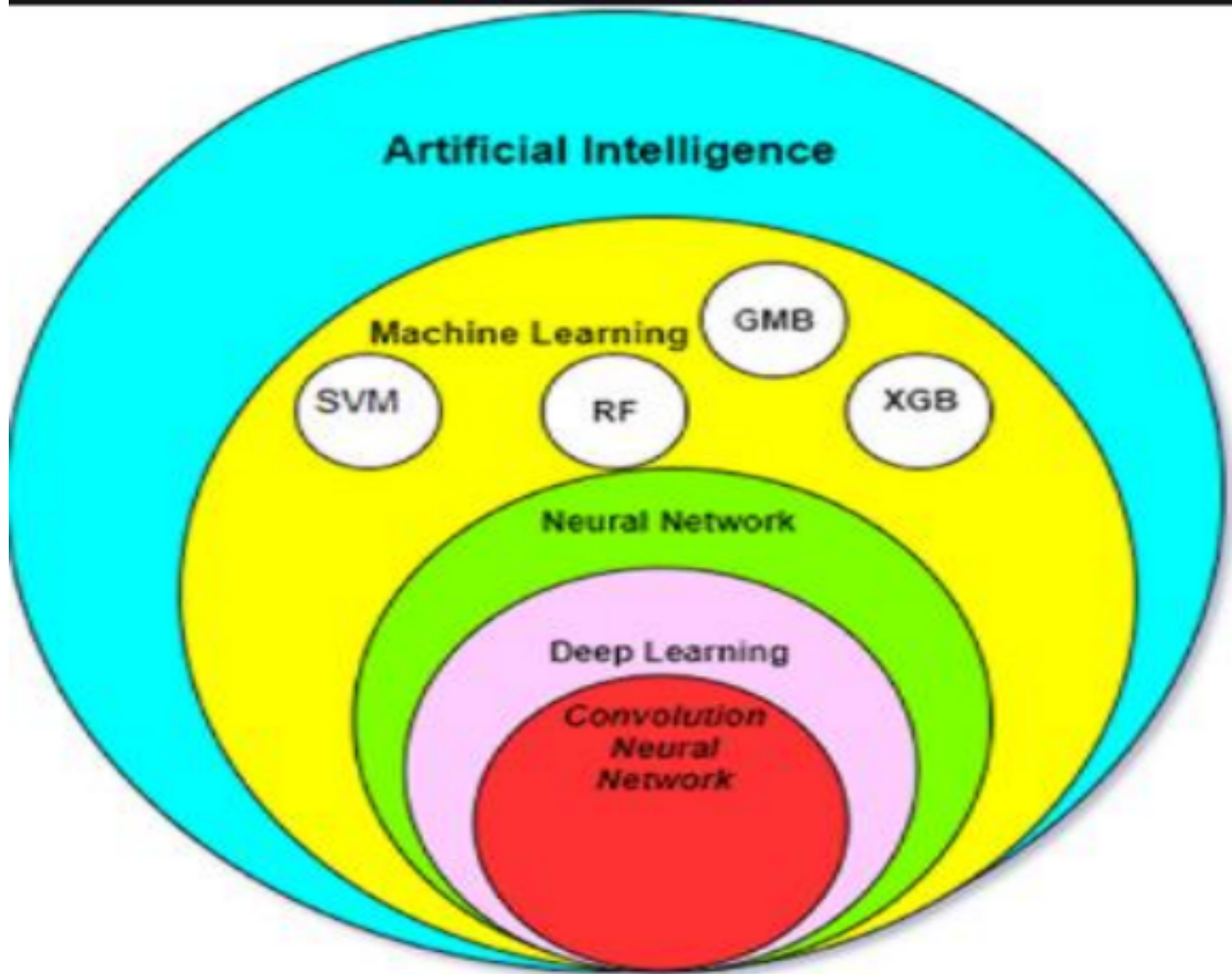
Knowledge Representation

- We want to compare the way that computers and humans work to see if we can better understand why each have their (computational)

strengths.

- Processing Models
- Knowledge Representation
- Reasoning

INTRODUCTION TO Machine Learning



Machine Learning (ML)

- ML is a branch of artificial intelligence: – Uses computing based systems to make sense out of data
 - Extracting patterns, fitting data to functions, classifying data, etc
- ML systems can learn and improve
 - With historical data, time and experience
- Bridges theoretical computer science and real noise data.

ML in real-life




10 active competitions

Sort By: Price

Active All Entered

Main Site All Eval Metrics Q




Predicting Red Hat Business Value

Classify customer potential

A month to go - **Featured**

1,292 teams
1,082 kernels
\$10,000




Bosch Production Line Performance

Reduce manufacturing failures

3 months to go - **Featured**

84 teams
\$10,000




TalkingData Mobile User Demographics

Get to know millions of mobile device users

12 days to go - **Featured**

1,479 teams
2,446 kernels
\$25,000




Grupo Bimbo Inventory Demand

Maximize sales and minimize returns of bakery goods

7 days to go - **Featured**

1,995 teams
2,714 kernels
\$25,000



Digit Recognizer

Classify handwritten digits using the famous MNIST data

4 months to go - **Getting Started**

1,028 teams
5,718 kernels
Knowledge

Big Data

- Widespread use of personal computers and wireless communication leads to “big data” •
- We are both producers and consumers of data •

Data is not random, it has structure, e.g., customer behavior

- We need “big theory” to extract that structure from data for
 - (a) Understanding the process
 - (b) Making predictions for the future

19

Why “Learn” ?

- Machine learning is programming computers to optimize a performance criterion using example data or past experience.
- There is no need to “learn” to calculate payroll

- Learning is used when:
 - Human expertise does not exist (navigating on Mars),
 - Humans are unable to explain their expertise (speech recognition)
 - Solution changes in time (routing on a computer network)
 - Solution needs to be adapted to particular cases (user biometrics)

20

What We Talk About When We Talk About “Learning”

- Learning general models from a data of particular examples

- Data is cheap and abundant (data warehouses, data marts); knowledge is expensive and scarce. • Example in retail: Customer transactions to consumer behavior:

*People who bought “Blink” also bought “Outliers”
(www.amazon.com)*

- Build a model that is *a good and useful approximation* to the data.

21

Data Mining

- Retail: Market basket analysis, Customer relationship management (CRM)

- **Finance:** Credit scoring, fraud detection
- **Manufacturing:** Control, robotics, troubleshooting
- **Medicine:** Medical diagnosis
- **Telecommunications:** Spam filters, intrusion detection
- **Bioinformatics:** Motifs, alignment
- **Web mining:** Search engines
- ...

22

What is Machine Learning?

- Optimize a performance criterion using

example data or past experience.

- Role of Statistics: Inference from a sample
- Role of Computer science: Efficient algorithms to
 - Solve the optimization problem
 - Representing and evaluating the model for inference 23

Machine Learning as a Process

Define Objectives

- Define measurable and quantifiable goals - Use this stage to learn about the problem

Data preparation

- Work better than the naïve approach or previous system
- Do the results make sense in the context of the problem



- Normalization - Transformation - Missing Values - Outliers

- Data Splitting
- Features Engineering - Estimating Performance - Evaluation and Model Selection

- Study models accuracy



Applications

- Association
- Supervised Learning
 - Classification
 - Regression
- Unsupervised Learning
- Reinforcement Learning

ISSUES:

Asking the wrong question
Trying to solve the wrong problem
Not having enough data
Not having the right data
Having too much data
Hiring the wrong people
Using the wrong tools
Not having the right model
Not having the right yardstick

Challenges



Not enough training data.
Poor Quality of data.
Irrelevant features.
Nonrepresentative training data.
Overfitting and Underfitting.

ML Vector

What is a Vector?

A vector is a tuple of one or more values called scalars.

Vectors are often represented using a lowercase character such as “v”; for example:

$$v = (v_1, v_2, v_3)$$

Where v_1 , v_2 , v_3 are scalar values, often real values.

Vectors are also shown using a vertical representation or a It is common to represent the target variable as a vector with the lowercase “y” when describing the training of a machine learning algorithm.

It is common to introduce vectors using a geometric analogy, where a vector represents a point or coordinate in an n-dimensional space, where n is the number of dimensions.

Defining a Vector

We can represent a vector in Python as a [NumPy array](#).

A NumPy array can be created from a list of numbers. For example, below we define a vector with the length of 3 and the integer values 1, 2 and 3.

```
# create a vector
from numpy import array
v = array([1, 2, 3])
print(v)
```

Vector Addition

Two vectors of equal length can be added together to create a new third vector. $c = a + b$

The new vector has the same length as the other two vectors. Each element of the new vector is calculated as the addition of the elements of the other vectors at the same index; for example:

$a + b = (a_1 + b_1, a_2 + b_2, a_3 + b_3)$ Or, put another way:

```
c[0] = a[0] + b[0]
c[1] = a[1] + b[1]
c[2] = a[2] + b[2]
```

We can add vectors directly in Python by adding NumPy arrays. # add vectors

```
from numpy import array
a = array([1, 2, 3])
print(a)
b = array([1, 2, 3])
print(b)
```

```
c = a + b print(c)
```

29

Vector Subtraction

One vector can be subtracted

from another vector of equal length to create a new third vector.

$$c = a - b$$

```
# subtract vectors
from numpy import array
a = array([1, 2, 3])
print(a)
b = array([0.5, 0.5, 0.5])
print(b)
c = a - b
print(c)
```

Vector Multiplication

Two vectors of equal length can be multiplied together.

$$c = a * b$$

```
# multiply vectors
from numpy import array
a = array([1, 2, 3])
print(a)
b = array([1, 2, 3])
print(b)
c = a * b
print(c)
```

Vector Division

Two vectors of equal length can be divided.

$$c = a / b$$

```
# divide vectors
```

```
from numpy import
```

```
array a = array([1, 2, 3])
```

```
print(a)
```

```
b = array([1, 2, 3])
```

```
print(b)
```

```
c = a / b
```

```
print(c)
```

Vector Dot Product

We can calculate the sum of the multiplied elements of two vectors

of the same length to give a scalar. This is called the dot product, named because of the dot operator used when describing the operation.

```
# dot product vectors
```

```
from numpy import array
```

```
a = array([1, 2, 3])
```

```
print(a)
```

```
b = array([1, 2, 3])
```

```
print(b)
```

```
c = a.dot(b)
```

```
print(c)
```

```
d = a @ b
```

```
print(d)
```

What is a Matrix?

A matrix is a [two-dimensional array](#) of scalars with one or more columns and one or more rows.

We can represent a matrix in Python using a two-dimensional NumPy array. A NumPy array can be constructed given a list of lists. For example, below is a 2 row, 3 column matrix.

```
# add matrices
```

```
from numpy import array A = array([[1, 2, 3], [4, 5, 6]]) print(A)
```

```
B = array([[1, 2, 3], [4, 5, 6]]) print(B)
```

```
C = A + B
```

```
print(C)
```

```
# subtract matrices
```

```
from numpy import array A = array([[1, 2, 3], [4, 5, 6]]) print(A)
```

```
B = array([[0.5, 0.5, 0.5], [0.5, 0.5, 0.5]])
```

```
print(B)
```

```
C = A - B
```

```
print(C)
```

```
# element-wise multiply matrices from numpy import array A = array([[1, 2, 3], [4, 5, 6]]) print(A)
```

```
B = array([[1, 2, 3], [4, 5, 6]]) print(B)
```

```
C = A * B
```

```
print(C)
```

```
# divide matrices
```

```
from numpy import array A = array([[1, 2, 3], [4, 5, 6]]) print(A)
B = array([[1, 2, 3], [4, 5, 6]]) print(B)
C = A / B
print(C)
# matrix-vector
multiplication
from numpy import array
A = array([[1, 2], [3, 4], [5, 6]])
print(A)
B = array([0.5, 0.5]) print(B)
C = A.dot(B)
print(C)
D = A @ B
print(D)
```

32

Linear Equation

The linear equation is the central part of linear algebra by which many problems are formulated and solved. It is an equation for a straight line.

Linear Equations in Linear Regression

Regression is a process that gives the equation for the straight line. It tries to find a best-fitting line with a specific set of data. The equation of the straight line bases on the linear equation:

$$Y = bX + a$$

Where,

a = It is a *Y-intercept* and determines the point where the line crosses the Y-axis.

b = It is a *slope* and determines the direction and degree to which the line is tilted.

33

Mean, Median, and Mode

Mean - The average value To calculate the mean, find the sum of all values, and divide the sum by the number of values.

Use the NumPy mean() method to find the average

```
speed: import numpy
```

```
speed =  
[99,86,87,88,111,86,103,87,94  
, 78,77,85,86]
```

```
x = numpy.mean(speed)
```

```
print(x)
```

Median-The mid point value.

The median value is the value in the middle, after you have sorted all the values.

Use the NumPy median() method

to find the middle value:	print(x)	=
import numpy		[99,86,87,88,111,86,103,8
		7,94,78,77,85,86]
speed =	Mode - The most common	
[99,86,87,88,111,86,103,8	value	
7 ,94,78,77,85,86]	The Mode value is the value	x = stats.mode(speed)
	that appears the most	
	number of times	print(x)
x = numpy.median(speed)	from scipy import stats	
	speed	

34

Standard Deviation

Standard deviation is a number that describes

Standard Deviation, σ :

4.8989794855664 Count, N:8

Sum, Σx :144

Mean, μ :18

Variance, σ^2 : 24

Steps

$$\sigma^2 = \Sigma(x_i - \mu)^2 / N$$

$$= (10 - 18)^2 + \dots + (16 - 18)^2 / 8$$

$$= 192/8 = 24$$

$$\Sigma \sqrt{24} = 4.898979485566 \quad 4$$

how spread out the values are.

A low standard deviation means that most of the numbers are close to the mean (average) value.

A high standard deviation means that the

values are spread out over a wider range.

```
import numpy
```

```
speed = [86,87,88,86,87,85,86]
```

```
x = numpy.std(speed)
```

```
print(x)
```

Variance

Variance is another number that indicates how spread out the values are. In fact, if you take the square root of the variance, you get the standard deviation! Or the other way around, if you multiply the standard deviation by itself, you get the variance!

1. Find the mean:

$$(32+111+138+28+59+77+97) / 7 = 77.4$$

2. For each value: find the difference from the

$$\text{mean: } 32 - 77.4 = -45.4$$

$$111 - 77.4 = 33.6$$

$$138 - 77.4 = 60.6$$

$$28 - 77.4 = -49.4$$

$$59 - 77.4 = -18.4$$

$$77 - 77.4 = -0.4$$

$$97 - 77.4 = 19.6$$

3. For each difference: find the square value:

$$(-45.4)^2 = 2061.16$$

$$(33.6)^2 = 1128.96$$

$$(60.6)^2 = 3672.36$$

$$(-49.4)^2 = 2440.36$$

$$(-18.4)^2 = 338.56$$

$$(-0.4)^2 = 0.16$$

$$(19.6)^2 = 384.16$$

4. The variance is the average number of these squared differences:

Use the NumPy var() method to

find the variance:

import numpy

speed =

[32,111,138,28,59,77,97] x =

numpy.var(speed)

print(x)

$$(2061.16 + 1128.96 + 3672.36 + 2440.36 + 338.56 + 0.16 + 384.16) / 7 = 1432.2$$

36

What is Probability?

- ✓ Uncertainty involves making decisions with incomplete information, and this is the way we generally operate in the world
- ✓ Handling uncertainty is typically described using everyday words like chance, luck, and risk.

- ✓Probability is a field of mathematics that gives us the language and tools to quantify the uncertainty of events and reason in a principled manner.
- ✓We can assign and quantify the likelihood of things we care about, such as outcomes, events, or numerical values.

Why is Probability Important to Machine Learning?

It would be fair to say that probability is required to effectively work through a machine learning predictive modeling project.

Machine learning is about developing predictive models from uncertain data. Uncertainty means working with imperfect or incomplete information.

Uncertainty is fundamental to the field of machine learning, yet it is one of the aspects that causes the most difficulty for beginners, especially those coming from a developer background.

There are three main sources of uncertainty in machine learning, they are: noisy data, incomplete coverage of the problem domain and imperfect models.

Nevertheless, we can manage uncertainty using the tools of probability. As machine learning practitioners, we must have an understanding of probability in order to manage the uncertainty we see in each project.

37

Probability is Importance of Machine Learning

- ✓Classification models must predict a probability of class membership.
- ✓Algorithms are designed using probability (e.g. Naive Bayes).
- ✓Learning algorithms will make decisions using probability (e.g. information gain).

- ✓ Sub-fields of study are built on probability (e.g. Bayesian networks).
- ✓ Algorithms are trained under probability frameworks (e.g. maximum likelihood).
- ✓ Models are fit using probabilistic loss functions (e.g. log loss and cross entropy).
- ✓ Model hyperparameters are configured with probability (e.g. Bayesian optimization).
- ✓ Probabilistic measures are used to evaluate model skill (e.g. brier score, ROC).

```
from scipy.stats import norm
import numpy as np

data_start = -5
data_end = 5
data_points = 11
data = np.linspace(data_start, data_end, data_points)

mean = np.mean(data)
std = np.std(data)

probability_pdf = norm.pdf(3, loc=mean, scale=std)

print(probability_pdf)
```

38

Correlation

Statistics and data science are often concerned about the relationships between two or more variables (or features) of a dataset. Each data point in the dataset is an **observation**, and the **features** are the properties or attributes of those observations.

Every dataset you work with uses variables and observations. For example, you might be interested in understanding the following:

- How the height of basketball players is correlated to their shooting accuracy
- Whether there's a relationship between employee work experience and salary
- What mathematical dependence exists between the population density and the gross domestic product of different countries.

```
import numpy as np
x = np.arange(10, 20)
y = np.array([2, 1, 4, 5, 8, 12, 18,
25, 96, 48])
r = np.corrcoef(x, y)
print(r)
```

39

Regression

- ✓The term regression is used when you try to find the relationship between variables.
- ✓In Machine Learning, and in statistical modeling, that relationship is used to predict the outcome of future events.

Linear Regression

Linear regression uses the relationship between the data-points to draw a straight line through all them.

This line can be used to predict future values.

40

How Does it Work?

Python has methods for finding a relationship between data-points and to draw a line of linear regression. We will show you how to use these methods instead of going through the mathematic formula.

In the example below, the x-axis represents age, and the y-axis represents

speed. We have registered the age and speed of 13 cars as they were passing a tollbooth. Let us see if the data we collected could be used in a linear regression:

```
import matplotlib.pyplot as plt
```

```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
```

```
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]
```

```
plt.scatter(x, y)
plt.show()
```

```
import matplotlib.pyplot as plt
```

```
from scipy import stats
```

```
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
```

```
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]
```

```
slope, intercept, r, p, std_err = stats.linregress(x, y)
```

```
def myfunc(x):
    return slope * x + intercept
```

```
mymodel = list(map(myfunc, x))
```

```
plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()
```

41

Handling and Representing Data:

✓ The definition of Data handling is in the title itself, that is, Handling the data in such a way that it becomes easier for people to understand and comprehend the given information.

✓ Hence, The **process of collecting, Recording, and representing data**

in some form of graph or chart to make it easy for people to understand is called **Data handling**.

- ✓ sometimes to analyze this data for certain trends, patterns may become difficult if the data is in its raw format. To overcome this data visualization comes into play.
- ✓ Data visualization provides a good, organized pictorial representation of the data which makes it easier to understand, observe, analyze.

Scatter Plot

Scatter plots are used to observe relationships between variables and uses dots to represent the relationship between them.

The [scatter\(\)](#) method in the matplotlib library is used to draw a scatter plot.

Example:

```
Python3
import pandas as pd
import matplotlib.pyplot as plt
# reading the database
data =
pd.read_csv("tips.csv")
# Scatter plot with day against
tip plt.scatter(data['day'],
```

```
data['tip'])
# Adding Title to the Plot
plt.title("Scatter Plot")

# Setting the X and Y
labels plt.xlabel('Day')
plt.ylabel('Tip')
```

Line Chart

[Line Chart](#) is used to

represent a relationship between two data X and Y on a different axis. It is plotted using the **plot()** function. Let's see the below example.

Example:

Python3

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

```
# reading the database
data = pd.read_csv("tips.csv")
# Scatter plot with day against tip
plt.plot(data['tip'])
plt.plot(data['size'])
```

```
plt.show()
```

```
plt.show()
```

```
# Adding Title to the Plot
plt.title("Scatter Plot")
```

```
# Setting the X and Y labels
plt.xlabel('Day')
plt.ylabel('Tip')
```

Bar Chart

A [bar plot](#) or bar chart is a graph that represents the category of data with rectangular bars with lengths and heights that is proportional to the values which they represent. It can be created using the **bar()** method.

```
# Adding the legends
plt.show()
```

Example:

Python3

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

```
# reading the database
data =
pd.read_csv("tips.csv")
# Bar chart with day against tip
plt.bar(data['day'], data['tip'])
plt.title("Bar Chart")
```

```
# Setting the X and Y labels
plt.xlabel('Day')
plt.ylabel('Tip')
```

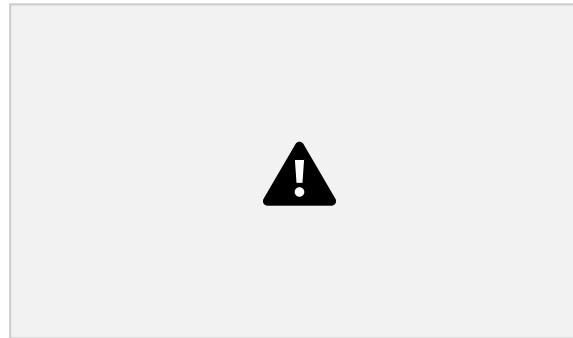
43

Supervised learning

Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. Basically supervised learning is when we teach or train the machine using data that is well labeled. Which means some data is already tagged with the correct answer.

After that, the machine is provided with a new set of examples(data) so that the supervised learning algorithm analyses the training data(set of training examples) and produces a correct outcome from labeled data.

For instance, suppose you are given a basket filled with different kinds of fruits. Now the first step is to train the machine with all different fruits one by one like this:



If the shape of the object is rounded and has a depression at the top, is red in color, then it will be labeled as –**Apple**.

If the shape of the object is a long curving cylinder having Green-Yellow color, then it will be labeled as –**Banana**.

44

Now suppose after training the data, you have given a new separate fruit, say Banana from the basket, and asked to identify it.



Since the machine has already learned the things from previous data and this time has to use it wisely. It will first classify the fruit with its shape and color and would confirm the fruit name as BANANA and put it in the Banana category. Thus the machine learns the things from training data(basket containing fruits) and then applies the knowledge to test data(new fruit).

Supervised learning is classified into two categories of algorithms:

Classification: A classification problem is when the output variable is a category, such as “Red” or “blue” or “disease” and “no disease”.

Regression: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Supervised learning deals with or learns with “labeled” data. This implies that some data is already tagged with the correct answer.

Types:-

Regression

Logistic Regression

Classification

Naive Bayes Classifiers

K-NN (k nearest neighbors)

Decision Trees

Support Vector Machine

Advantages:-

Supervised learning allows collecting data and produces data output from previous experiences.

Helps to optimize performance criteria with the help of experience.

Supervised machine learning helps to solve various types of real-world computation problems.

Disadvantages:-

Classifying big data can be challenging.

Training for supervised learning needs a lot of computation time. So, it requires a

Steps



47

Prediction

Understanding the `predict()` function in Python

In the domain of **data science**, we need to apply different machine learning models on the data sets in order to train the data. Further which we try to predict the values for the untrained data.

This is when the `predict()` function comes into the picture.

Python predict() function enables us to **predict the labels of the data values** on the basis of the trained model.

```
model.predict(data)
```

48

Classification

In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data.

Examples of classification problems include:

Given an example, classify if it is spam or not.

Given a handwritten character, classify it as one of the known characters. Given recent user behavior, classify as churn or not.

Binary Classification

Multi-Class Classification

1. Binary classification refers to those classification tasks that have two class labels.

Examples include:

Email spam detection (spam or not). Churn prediction (churn or not). Conversion prediction (buy or not).

Popular algorithms that can

be used for binary classification include:

Logistic Regression

k-Nearest Neighbors

Decision Trees

Support Vector

Machine Naive Bayes


```
# example of binary classification
task from numpy import where
from collections import Counter
from sklearn.datasets import
make_blobs from matplotlib import
pyplot
# define dataset
X, y = make_blobs(n_samples=1000, centers=2,
random_state=1)
# summarize dataset shape
print(X.shape, y.shape)
# summarize observations by class
```

```
label counter = Counter(y)
print(counter)
# summarize first few examples
for i in range(10):
    print(X[i], y[i])
# plot the dataset and color the by class
label for label, _ in counter.items():
    row_ix = where(y == label)[0]
    pyplot.scatter(X[row_ix, 0], X[row_ix,
1], label=str(label))
pyplot.legend()
pyplot.show()
```

50

2. Multi-class classification refers to those classification tasks that have more than two class labels. Examples include:

Face classification.

Plant species classification.

Optical character
recognition.

```
# example of multi-class classification
task from numpy import where
from collections import Counter
from sklearn.datasets import
make_blobs from matplotlib import
```

```
pyplot
# define dataset
X, y = make_blobs(n_samples=1000, centers=3,
random_state=1)
# summarize dataset shape
print(X.shape, y.shape)
# summarize observations by class
label counter = Counter(y)
print(counter)
# summarize first few examples
for i in range(10):
    print(X[i], y[i])
```

```
# plot the dataset and color the by class
label for label, _ in counter.items():
    row_ix = where(y == label)[0]
    pyplot.scatter(X[row_ix, 0], X[row_ix, 1],
                    label=str(label))
pyplot.legend()
pyplot.show()
```

51

[3. Multi-label classification](#) refers to those classification tasks that have two or more class labels, where one or more class labels may be predicted for each example.

Consider the example of [photo classification](#), where a given photo may have multiple objects in the scene and a model may predict the presence of multiple known objects in the photo, such as “*bicycle*,” “*apple*,” “*person*,” etc.

```
# example of a multi-label classification task
from sklearn.datasets import
make_multilabel_classification
# define dataset
X, y =
make_multilabel_classification(n_samples=1000,
n_features=2, n_classes=3, n_labels=2,
random_state=1)
# summarize dataset shape
print(X.shape, y.shape)
```

```
# summarize first few examples
```

```
for i in range(10):
```

```
    print(X[i], y[i])
```

52

4. Imbalanced classification refers to classification tasks where the number of examples in each class is unequally distributed. Examples include:

Fraud detection.

Outlier detection.

Medical diagnostic tests.

```
# example of an imbalanced binary classification task
from numpy import where
from collections import Counter
from sklearn.datasets import make_classification
from matplotlib import pyplot
# define dataset
X, y = make_classification(n_samples=1000, n_features=2, n_informative=2,
    n_redundant=0, n_classes=2, n_clusters_per_class=1, weights=[0.99,0.01],
    random_state=1)
# summarize dataset shape
print(X.shape, y.shape)
# summarize observations by class label
counter = Counter(y)
print(counter)
# summarize first few examples
for i in range(10):
    print(X[i], y[i])
# plot the dataset and color the by class label
for label, _ in counter.items():
    row_ix = where(y == label)[0]
    pyplot.scatter(X[row_ix, 0], X[row_ix, 1], label=str(label))
pyplot.legend()
```

Understanding Data

Being data scientist, we all have to work on a lot of different datasets. To fit any predictive model on a dataset, we need to understand the complexity of the dataset before deciding which predictive model to use to get optimal performance.

But, datasets are generally huge, and manually understanding each data point in the dataset is impossible. Thus, we need some metrics and visualizations to help us understand the nature of data and give us a brief overview.

Exploratory Data Analysis (EDA) analyzes and visualizes data to extract insights from it. It can be described as a process of summarizing important characteristics of data to have a better understanding. To learn about the process of EDA.

We will see in detail each necessary step for EDA, why it is required, and how to implement it.

Feature Selection For Machine Learning in Python

The data features that you use to train your machine learning models have a huge influence on the performance you can achieve.

Irrelevant or partially relevant features can negatively impact model performance.

Feature selection is a process where you automatically select those features in your data that contribute most to the prediction variable or output in which you are interested.

Having irrelevant features in your data can decrease the accuracy of many models, especially linear algorithms like linear and logistic regression. Three benefits of performing feature selection before modeling your data are:

Reduces Overfitting: Less redundant data means less opportunity to make decisions based on noise.

Improves Accuracy: Less misleading data means modeling accuracy improves.

Reduces Training Time: Less data means that algorithms train faster.⁵⁵

```
# Feature Selection with Univariate Statistical
Tests from pandas import read_csv
from numpy import set_printoptions
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
# load data
filename = 'pima-indians-diabetes.data.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age',
'class'] dataframe = read_csv(filename, names=names)
array = dataframe.values
X = array[:,0:8]
Y = array[:,8]
# feature extraction
test = SelectKBest(score_func=f_classif, k=4)
fit = test.fit(X, Y)
# summarize scores
set_printoptions(precision=3)
print(fit.scores_)
features = fit.transform(X)
# summarize selected features
print(features[0:5,:])
```

Feature Normalization

Normalisation is another important concept needed to change all features to the same scale. This allows for faster convergence on learning, and more uniform influence for all weights.

Scaling

Standard Scaler

It standardize features by removing the mean and scaling to unit variance The standard score of a sample x is calculated as:

$$z = (x - \mu) / s$$

Min Max Scale

Another way to normalise is to use the Min Max Scaler, which changes all features to be between 0 and 1, as defined below:



RobustScaler

Works similarly to standard scaler except that it uses median and quartiles, instead of mean and variance. Good as it ignores data points that are outliers.

Normalizer

Scales each data point such that the feature vector has a Euclidean length of 1. Often used when the direction of the data matters, not the length of the feature vector.

Pipeline

Scaling have a chance of leaking the part of the test data in train-test split into the training data. This is especially inevitable when using cross-validation.

Persistence

To save the fitted scaler to normalize new datasets, we can save it using pickle or joblib for reusing in the future.

58

Data Cleaning

When working with multiple data sources, there are many chances for data to be incorrect, duplicated, or mislabeled. If data is wrong, outcomes and algorithms are unreliable, even though they may look correct.

Data cleaning is the process of changing or eliminating garbage, incorrect, duplicate, corrupted, or incomplete data in a dataset.



59

Training Data

Machine Learning algorithms learn from data. They find relationships, develop understanding, make decisions, and evaluate their confidence from the training data they're given. And the better the training data is, the better the model performs.



Neural networks and other artificial intelligence programs require an initial set of data, called training data, to act as a baseline for further application and utilization. This data is the foundation for the program's growing library of information.

60

Validation Dataset

The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.

The validation set is used to evaluate a given model, but this is for frequent

evaluation. We, as machine learning engineers, use this data to fine-tune the model hyperparameters.

Hence the model occasionally sees this data, but never does it “*Learn*” from this. We use the validation set results, and update higher level hyperparameters.

So the validation set affects a model, but only indirectly. The validation set is also known as the Dev set or the Development set. This makes sense since this dataset helps during the “development” stage of the model.

61

Test Dataset

The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

The Test dataset provides the gold standard used to evaluate the model. It is only used once a model is completely trained(using the train and validation sets).

The test set is generally what is used to evaluate competing models (For example on many Kaggle competitions, the validation set is released initially along with the

training set and the actual test set is only released when the competition is about to close, and it is the result of the the model on the Test set that decides the winner).

Many a times the validation set is used as the test set, but it is not good practice. The test set is generally well curated. It contains carefully sampled data that spans the various classes that the model would face, when used in the real world.



62

Different Models of Supervised Learning

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.



63

1. Regression

Regression algorithms are used if there is a relationship between the input variable and the output variable. It 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:

1. Linear Regression-Keerthan reddy

monday

2. Regression Trees-sagar puri
tuesday

3. Non-Linear Regression-sumith
rathi-wed

4. Bayesian Linear Regression
shivam-friday

5. Polynomial Regression-seema
staruday

2. 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.Spam

1. Filtering-Vinayabagawat-mon
2. Random Forest-kavita-Tue
3. Decision Trees-kavita-wed
- 4.

Logistic Regression-ashes
cakrava-fridy

5. Support vector Machines
puspendar-sat

Hyper parameters



Model Parameter

A model parameter is a configuration variable that is internal to the model and whose value can be estimated from data.

- ✓ They are required by the model when making predictions.
- ✓ Their values define the skill of the model on your problem.
- ✓ They are estimated or learned from data.
- ✓ They are often not set manually by the practitioner.
- ✓ They are often saved as part of the learned model.

Hyperparameter

A hyperparameter is a configuration that is external to the model and whose value cannot be estimated from data.

They are often used in processes to help estimate model parameters.

They are often specified by the practitioner.

They can often be set using heuristics. They are often tuned for a given predictive modeling problem.

- ✓ The learning rate for training a neural network.
- ✓ The C and sigma hyperparameters for

support vector machines.

- ✓ The k in k-nearest neighbors.

Unsupervised

What is Unsupervised Learning?

Unsupervised Learning is a machine learning technique in which the users do not need to supervise the model. Instead, it allows the model to work on its own to discover patterns and information that was previously undetected. It mainly deals with the unlabelled data.

Unsupervised Learning Algorithms allow users to perform more complex processing tasks compared to supervised learning. Although, unsupervised learning can be more unpredictable compared with other natural learning methods. Unsupervised learning algorithms include clustering, anomaly detection, neural networks, etc.

Why Unsupervised Learning?

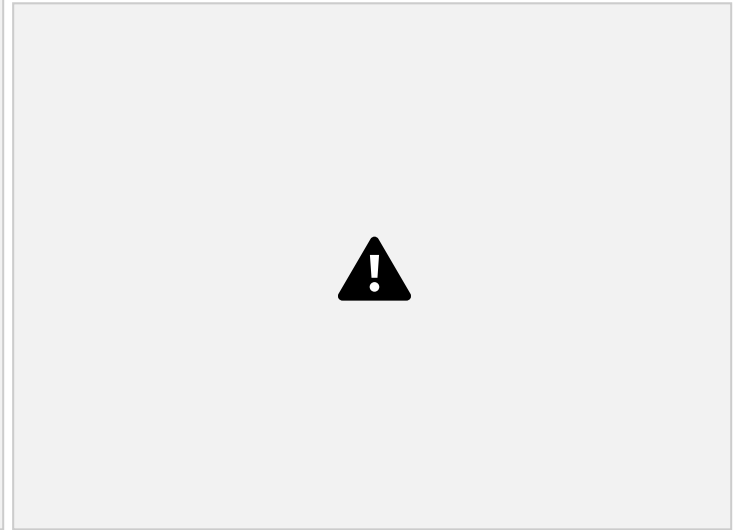
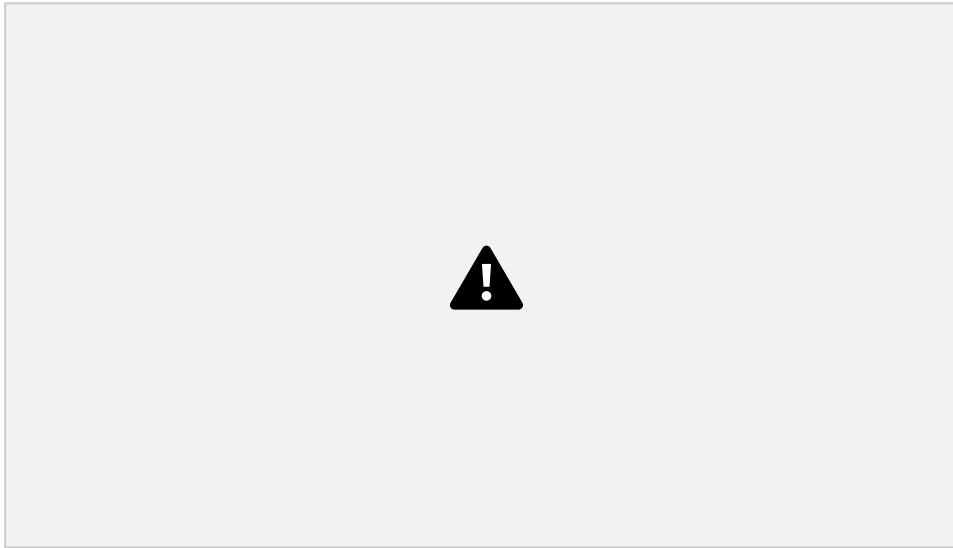
Here, are prime reasons for using Unsupervised Learning in [Machine Learning](#):

- ✓ Unsupervised machine learning finds all kind of unknown patterns in data.
- ✓ Unsupervised methods help you to find features which can be useful for categorization.
- ✓ It is taken place in real time, so all the input data to be analyzed and labeled in the presence of learners.
- ✓ It is easier to get unlabeled data from a computer than labeled data, which needs manual intervention.

Unsupervised

Example of Unsupervised Machine Learning

Let's, take an example of Unsupervised Learning for a baby and her family dog. She knows and identifies this dog. Few weeks later a family friend brings along a dog and tries to play with the baby.



Baby

has not seen this dog earlier. But it recognizes many features (2 ears, eyes, walking on 4 legs) are like her pet dog. She identifies the new animal as a dog. This is unsupervised learning, where you are not taught but you learn from the data (in this case data about a dog.) Had this been supervised learning, the family friend would have told the baby that it's a dog as shown in the above Unsupervised Learning example.

Other Examples:

A subgroup of cancer patients grouped by their gene expression measurements

Groups of shopper based on their browsing and purchasing histories

Movie group by the rating given by movies viewers

Clustering Types of Unsupervised Learning

Algorithms Below are the clustering types of

Unsupervised Machine Learning algorithms:

Unsupervised learning problems further grouped into clustering and association problems.

Clustering

Clustering Types of Unsupervised Learning Algorithms

Below are the clustering types of Unsupervised Machine

Learning algorithms:

Unsupervised learning problems further grouped into clustering and association problems.

Clustering



68

Types of USL:

Hierarchical clustering-Adnan CV

K-means clustering-Mansi Gaonkar

K-NN (k nearest neighbors)-Mansi Gaonkar

Principal Component Analysis-Kartik Saraswat

Independent Component Analysis-Kartik Saraswat

Singular Value Decomposition-Gautham Krishna

Hierarchical Clustering

Hierarchical clustering is an algorithm which builds a hierarchy of clusters. It begins with all the data which is assigned to a cluster of their own. Here, two close cluster are going to be in the same cluster. This algorithm ends when there is only one cluster left.

K-means Clustering

K means it is an iterative clustering algorithm which helps you to find the highest value for every iteration. Initially, the desired number of clusters are selected. In this clustering method, you need to cluster the data points into k groups. A larger k means smaller groups with more granularity in the same way. A lower k means larger groups with less granularity.

Agglomerative clustering

This type of K-means clustering starts with a fixed number of clusters. It allocates all data into the exact number of clusters. This clustering method does not require the number of clusters K as an input. Agglomeration process starts by forming each data as a single cluster.

Dendrogram

In the Dendrogram clustering method, each level will represent a possible cluster. The height of dendrogram shows the level of similarity between two join clusters. The closer to the bottom of the process they are more similar cluster which is finding of the group

from dendrogram which is not natural and mostly subjective.

K- Nearest neighbors

K- nearest neighbour is the simplest of all machine learning classifiers. It differs from other machine learning techniques, in that it doesn't produce a model. It is a simple algorithm which stores all available cases and classifies new instances based on a similarity measure.

Principal Components Analysis

In case you want a higher-dimensional space. You need to select a basis for that space and only the 200 most important scores of that basis. This base is known as a principal component. The subset you select constitute is a new space which is small in size compared to original space. It maintains as much of the complexity of data as possible.

Association

Association rules allow you to establish associations amongst data objects inside large databases. This unsupervised technique is

69

about discovering interesting relationships between variables in large databases. For example, people that buy a new home most

Supervised vs. Unsupervised Machine Learning

Here is the main difference between Supervised vs. Unsupervised Learning:

Parameters		Supervised machine learning technique	Applications of Unsupervised Machine Learning ✓ Some application of Unsupervised Learning Techniques are:
Input Data		Algorithms are trained using labeled data.	Unsupervised machine learning technique
Computational Complexity		Supervised learning is a simpler method.	Algorithms are used against data which is not labelled
Accuracy		Highly accurate and trustworthy method.	Unsupervised learning is computationally complex Less accurate and trustworthy method.

- ✓Clustering automatically split the dataset into groups base on their similarities
- ✓Anomaly detection can discover unusual data points in your dataset. It is useful for finding fraudulent transactions

✓ Association mining identifies sets of items which often occur together in your dataset ✓ Latent variable models are widely used for data preprocessing. Like reducing the number of features in a dataset or decomposing the dataset into multiple components

Disadvantages of Unsupervised Learning

You cannot get precise information regarding data sorting, and the output as data used in unsupervised learning is labeled and not known

Less accuracy of the results is because the input data is not known and not labeled by people in advance. This means that the machine requires to do this itself.

The spectral classes do not always correspond to informational classes.

The user needs to spend time interpreting and label the classes which follow that classification. Spectral properties of classes can also change over time so you can't have the same class information while moving from one image to another.