Day 1 : 18th December 2023

Group - 17, 21, 25, 29

Names :
Garvitaa Agarwal
Vaidika Padigela
Vishnu Priya (25) — Cohort 5
D. Greethika

Q. How is Machine Learning different from traditional programming ?

Feature	**Traditional Programming**	**Machine Learning**
Approach	Rule-based	Data-driven
Output	Deterministic	Probabilistic
Adaptability	Limited to programmed scenarios	Adaptable to new scenarios
Learning	No self-learning	Self-learning

- Traditional programming tells the computer what to do, while Machine Learning shows the computer what to learn.
- Traditional programming is rule-based and requires human intervention to update the rules or logic when circumstances change. In contrast, ML algorithms are data-driven and can adapt and learn from new data without explicitly changing the algorithm itself.
- Traditional Programming: Relies less on data. The quality of the output depends mainly on the logic defined by the programmer.Machine Learning: Heavily reliant on data. The quality and quantity of the training data significantly impact the performance and accuracy of the model.

- Machine Learning enables the development of systems that can learn and adapt from data rather than relying solely on predefined rules.
- In traditional programming, the programmer writes specific instructions for the computer to follow. These instructions tell the computer exactly what to do in order to solve a problem.Machine Learning, on the other hand, is a different approach. Instead of telling the computer exactly what to do, we give it lots of examples and let it learn from those examples. It's like showing the computer many pictures of cats and dogs and letting it figure out how to tell them apart. The computer uses these examples to find patterns and make predictions or decisions based on those patterns. It learns from the data and can improve its performance over time. So instead of programming the computer with specific rules, we let it learn on its own from the examples we provide.
- Traditional programming uses explicit instructions, while Machine Learning uses examples and patterns to make predictions or decisions.

Example :

1. Imagine you want to teach a child how to distinguish between cats and dogs.In traditional programming, you would give the child a set of rules, like "if it has four legs and barks, it's a dog."In machine learning, you would show the child many pictures of cats and dogs, and let them identify the common features of each animal. Over time, the child would develop their own understanding of how to tell the difference.

2. Let's say we want to create a program that detects spam emails. In traditional programming, we would need to define specific rules to identify spam emails. For example, we could define rules such as "If an email contains the word 'free' and is from an unknown sender, classify it as spam." However, with Machine Learning, we can take a different approach. Instead of explicitly defining rules, we can train a machine learning algorithm to recognize patterns in spam emails. We would provide the algorithm with a large dataset of emails, labeling them as either spam or not spam. The algorithm would then learn the patterns that differentiate spam from non-spam emails.Once trained, the machine learning algorithm can make predictions on new, unseen emails using the patterns it learned. It doesn't rely on explicit rules but rather on the patterns it detects in the data. This allows the algorithm to adapt to new types of spam and improve its accuracy over time, even without the programmer updating any rules.

3. If you’re building a program which makes a game, in traditional programming you need to first write down the rules of the game or the code that defines how to play “legally”, playing well is a different thing. For a game to be perfectly understood, master gamers have a body of knowledge. For eg in chess, we have a set of moves and then we have their evaluations and we need to decide which one is the best based on the resultant positions of that move. A traditional chess program simply generates possible moves and then evaluates and it is a deterministic approach. But we need the smartness of the programmer in implementing them efficiently.Traditional programmers need a function which evaluates a path/move and

chooses the optimal move. Such programs are also very efficient today and work very well. Meanwhile a ML algorithm will say given these positions in the state space and the possible moves in the state space, it will determine which evaluation function will give me the best move, it determines which function is the best.

4. Trivial example - To convert a distance from miles to kms, in the traditional way, you are given the distance in miles and you multiply it by the fixed number to get the answer in kms. In ML, you are given a bunch of distances in miles and corresponding kms, and you have to determine how to convert it so that in future for other input, you can convert it, but there's no guarantee that the answer will be correct in this case.

"Giving the wrong answers for questions which did not have an answer." - good description for ML

Important -

The ML algo says this is the input and this is the output so what function will give this output for the given input.
In ML, you know the input and the output, the purpose is to produce the function which will produce the corresponding output for the given input. This is why ML is data driven.

In traditional programming, you are given an input and the function to perform and you have to find out the output.
While in ML, you are given a bunch of input and their corresponding output and the goal is to determine the best function which will provide the correct outputs for the given inputs.

Overfitting :
Latching on to incidental data.

Curse of dimensionality:
The number of features (dimensions) increases , you are going to need a humongous amount of data which any real world problem usually doesn't have.
The data you have for training a model, is truly representing the underlying problem.

Blessing of Sparseness:
sow pods txts

Given a program that randomly generates letters, what is the probability that it actually generates an actual word?

Write a program which looks at frequency distribution of letters in English and generates a word according to that, not completely randomly.
Mayzner
ETAOIN SHRDLU

Matrix multiplication : composition of linear transformations

$$\begin{bmatrix} p & q \\ r & s \end{bmatrix} \times \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

M1 = p(ax1 + bx2) + q(cx1 + dx2)
M1 = pa x1 + qc x1 + pb x2 + qd x2
M1 = (pa + qc) x1 + (pb + qd) x2

M2 = r(ax1 + bx2) + s(cx1 + dx2)
M2 = ra x1 + sc x1 + rb x2 + ds x2
M2 = (ar + cs) x1 + (br + ds) x2

If the determinant of a matrix is one then the scale is not switching

xkcd 184

Complex and imaginary numbers :
Multiplying by i is rotating it by 90 degrees

Functional programming: reducing computation to its essence
I can replace a variable/an expression by a value, my program will remain correct in syntax and logic
Function is a named piece of computation.
The whole point of functional programming is to construct computation in piece and any piece can be replaced.

Monoid : strings

Traditional Programming

Machine Learning Programming

Traditional Programming:

Rule Based Logic:

Here, the developers explicitly mention rules and logic to instruct the computer on how to perform a task.

Clear Instructions:

The programmer provides a set of step-by-step instructions for the computer to follow, specifying the exact actions to be taken for different scenarios.

Task-Specific Code:

Each possible scenario must be anticipated and addressed through predefined code, making the program task-specific.

Limited Adaptability:

Traditional programs are static and do not adapt to new data or changing conditions without manual modification of the code

Accuracy :

Can be 100% as it works according to the pre defined rules but ML prediction is never 100% accurate.

Machine Learning:

Learning from Data:

Machine learning algorithms learn from data by identifying patterns, allowing the system to make decisions.

Decision Making:

Rather than following explicit rules,ML models generalize from the examples and then it makes decisions based on patterns observed in the training data.

Adaptability:

ML models can adapt to new data and evolving conditions,improving their performance over time.

Probabilistic Outputs:

Machine learning models often provide probabilistic outputs, indicating their confidence in a particular prediction, unlike traditional programs that provide deterministic outputs.

NOTE:

In Software engineering (traditional programming), you have to start the program from scratch and re-write the code. In machine learning, you may not need to do everything from scratch.

Eg:

Write down the rules of the game, and write down the code to play the game “legally”. In traditional programming, a master player sits with a programmer who tries to decode the thought process of the player and implement it into the game. A traditional chess program depends on the programmer to implement it efficiently. You need a function that evaluates and optimises the path.

In a machine learning program, for every move, it analyses the possible moves for the future. It evaluates complex criteria. There is no need for a grand master to dictate their thought process. It would rather ask for the data of the moves made by different players to choose the next best move. It is essentially playing by the statistics.

Traditional program figures out the top three moves explained by the grand master. A machine learning program already has the best three moves and it evaluates the best move according to the success of the move. It does not just evaluate the most successful move at the certain position, instead it checks for the function that would result in those moves so that it can make a move for the future. There is no way a lookup table can be constructed with the finite memory of the moves made.

Model:

Given the inputs and corresponding outputs, what is the transformational model that can be made?

Prediction:

Output for the input which was not a part of the initial dataset/input.

In a typical machine learning program, data comes from real world measurements. The problem with real world measurements is that there is a probability of errors. There is also a problem of wrong predictions when you show data that was not shown before.

The equivalent of clever hans (it is a horse that can do arithmetic operations but was instead found to be only working on the owner's expressions) in machine learning is an owner-fitted model. It only checks if the user is happy with the output.

Eg:

The US defense had a machine learning model that differentiates between their tanks and enemy tanks. The data had clear images of US tanks and not-so-clear images of the enemy tanks. So the ML model was trained on the basis of whether the image was clear or not in order to know if the tank belongs to the US or not.

Alpha Zero:

AlphaZero was a chess model of the Google deep-mind labs that was trained by giving the rules of the game and made the model play against itself for the datasets. Within four hours, it managed to play 100 games against a traditional programming model (stockfish), winning 72 games and losing 38. I created totally new moves in a few situations. A typical chess playing program measures the tactics, whereas the strategic moves made by the grand masters can easily defeat them. But with a machine learning model, it can also strategize the moves to be made in the future and analyze the moves made by the grandmaster and defeat them.

Curse of Dimensionality:

1) Population = $-100 < x < 100$

   Sample = 150 points (x,y)

   Q)What percentage of the population is the sample?

   Sol) Around 75 % (150/199)*100

2) Population = -100 < x1< =100

=-100 < x2 <= 100

Data points = 150 points(x1, x2, y)

Q)What percentage of the population is the sample?

Sol) 0.375% (150/(40000))

3) Population = -100 < x1< =100

=-100 < x2 <= 100

=-100 < x3 <= 100

Data points = 1000 points(x1, x2, x3,y)

Q)What percentage of the population is the sample?

Sol) 0.0125% (1000/(40000 *200))

4) Population = 10 dimensions

Data points = 1 billion

Q)What percentage of the population is the sample?

Sol)10^9/(200^10)

= 10 ^ 9 / ((2 ^ 10) * (100 ^ 10))

= 10 ^ -14

Q) I have a program which will generate words randomly. What is the probability that it generates an actual word?

0.25 million/Summation of 26 from 2 to 15

Approx 10^6/(4*(10^21))

Q) Write a program which actually looks at the frequency distribution of the letters

Most frequent = ETAOIN SHRDLU

Look at mayzer research paper

https://norvig.com/mayzner.html

Q) Why is matrix multiplication done in row x column? Why is it defined in that way?

A) Matrices are nothing but linear transformations. Matrix multiplications are nothing but compositions of linear transformations.

[a b [p q
C d] x r s]

M1 = p(ax1 + bx2) + q(cx1 + dx2)
M1 = pa x1 + qc x1 + pb x2 + qd x2
M1 = (pa + qc) x1 + (pb + qd) x2

M2 = r(ax1 + bx2) + s(cx1 + dx2)
M2 = ra x1 + sc x1 + rb x2 + ds x2
M2 = (ar + cs) x1 + (br + ds) x2

Rotational Matrix :

Determinant is 1 in rotational matrices.

NOTE 2:

- Matrix multiplication is associative but not commutative.
- Multiple dimensions in ML so we use subscripts.
- Notation is very important
- Makes generalization very easy

Generalize linear equation

y = wx + b

W = omega = weights

b = beta = bias = w0 = one convenient weight

= sigma (wi xi)

Blessing of Sparsity:

If data is sparse, in some view there is some relation between some axes.

If we can somehow rotate our axis so that some values can be discarded, the no. of dimensions come down. It's possible when there is some relation between the axes.

To do this, transformation is needed.

How to find the rotation?

Go back to the matrix, is there some way to rotate the matrix and clearly say this thing is most important.

Find that rotation in which we can reduce the no. of dimensions to eight.Transform in such a way that some of the omegas become zero, i.e. reduce dimensions.

Eigen values and Eigen vectors give information about the matrix.Higher eigen values are more important dimensions. We can pretend the last few eigen values do not exist. Corresponding Eigen vectors are the directions of axes in the new system.

Eigen vector consists of the directions of the new axis.

We can just take the two most important dimensions and work with them.

This is easier to visualise.

Cayley- Hamilton Theorem:

This theorem states that every square matrix satisfies its own characteristic equation.

In other words, the scalar polynomial $p(\lambda) = \det(\lambda I - \sigma)$ also holds for the stress polynomial $p(\sigma)$.

Complex Numbers:

On a number line

Multiply by -1 = rotate by 180 degrees

Multiply by i = rotate 90 Degrees

To generalize square roots,

Think of planes

Square roots do not exist on number lines

They exist on planes.

Nth degree equation has n roots(this if restricted to number line, cannot be meaningful)

Functional Programming:

You can replace any expression by a value and it will still remain correct.

One can compose functions in maths, but can the same be done with programming?

A Function is a NAMED piece of computation.

Rules for such computations:

Strings are monoids(associative + identity).

Properties such as associativity, identity[there exists an i such that for all

x, i (some op) x == x]

Monoid + inverse → group structure

So, we need to consider these structures in functional programming.

Q. How to use ziro studio (ztutor)? All good features that we must use?

Instead of opening someone else's project, just create your own dummy project to talk to the AI, you don't even need to write any code.
https://zirostudio-dev.web.app/user-view

Q) What is Keras?

Keras is a neural networks API used to help build deep learning models.

Fashion MNIST:

- It is a dataset that includes 70k images and 10 categories(obviously related to fashion).
- Images are 28 x 28.
- Used for training neural networks.
- Already built into tensorflow

Q) What is numpy used for?

Meeting ID = 88098709700
Passcode = 1537659305