**Name : Manav Butani**

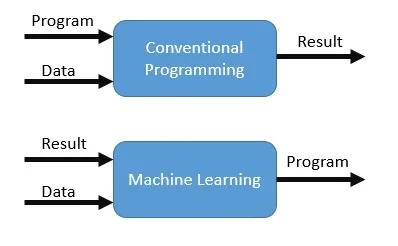
**Id : 19IT016**

1. **Basics of machine learning**
   1. **What are the differences between machine learning and conventional computer programs?**

**Answer:**

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

On the contrary, in machine learning programming, the input and output data are fed to the algorithm, creating the program.



* 1. **Explain three types of machine learning with examples and applications.**
     1. **Supervised machine learning.**
     2. **Unsupervised machine learning.**
     3. **Reinforcement learning.**

**Answer:**

***Supervised Machine Learning:***

Supervised learning is a machine learning method in which models are trained using labeled data. In supervised learning, models need to find the mapping function to map the input variable (X) with the output variable (Y).

Supervised Machine learning

Supervised learning needs supervision to train the model, which is similar to how a student learns things in the presence of a teacher. Supervised learning can be used for two types of problems: Classification and Regression.

Example: Suppose we have an image of different types of fruits. The task of our supervised learning model is to identify the fruits and classify them accordingly. So to identify the image in supervised learning, we will give the input data as well as output for that, which means we will train the model by the shape, size, color, and taste of each fruit. Once the training is completed, we will test the model by giving the new set of fruit. The model will identify the fruit and predict the output using a suitable algorithm.

***Unsupervised Machine Learning:***

Unsupervised learning is another machine learning method in which patterns are inferred from the unlabeled input data. The goal of unsupervised learning is to find the structure and patterns from the input data. Unsupervised learning does not need any supervision. Instead, it finds patterns from the data by its own.

unsupervised learning can be used for two types of problems: Clustering and Association.

Example: To understand unsupervised learning, we will use the example given above. So unlike supervised learning, here we will not provide any supervision to the model. We will just provide the input dataset to the model and allow the model to find the patterns from the data. With the help of a suitable algorithm, the model will train itself and divide the fruits into different groups according to the most similar features between them.

***Reinforcement Machine learning:***

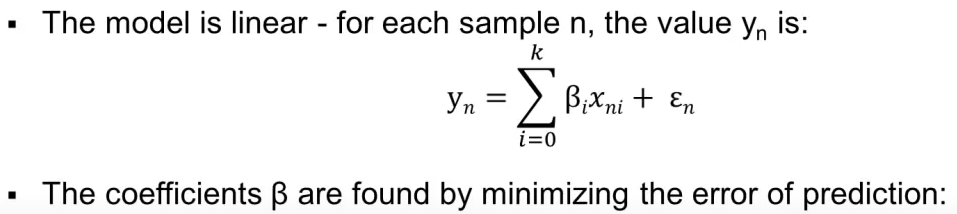
Reinforcement learning is a machine learning training method based on rewarding desired behaviors and/or punishing undesired ones. In general, a reinforcement learning agent is able to perceive and interpret its environment, take actions and learn through trial and error.

Example: Chess game

1. **Linear Regression**
   1. **Explain linear regression OLS method with example and application.**
   2. **Explain linear regression Gradient Descent method with example and application.**
   3. **Explain Gradient descent with an example.**
   4. **Explain Multivariate Linear regression with example and application.**
   5. **Explain Polynomial regression with example and application.**
   6. **Explain how to measure error in regression and application.**

**Answer: (a)**

Ordinary Least Squares regression (OLS) is a common technique for estimating coefficients of linear regression equations which describe the relationship between one or more independent quantitative variables and a dependent variable (simple or multiple linear regression).



Example: Predict the height of plants based on the days they spent in sunlight. Before exposure the height of plants is 30 cm. They grow 0.1 cm after exposure to sunlight for a day.

y = height of plants

x = number of days spent in sunlight

e = 30 cm, when x=0

B = 0.1

Estimate the height of plants after 20 days.

y = 0.1\*20 + 30 = 32 cm

**Answer: (b)**

Gradient descent is an optimization algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient.

When there are one or more inputs you can use a process of optimizing the values of the coefficients by

iteratively minimizing the error of the model on your training data. This operation is called Gradient Descent and works by starting with random values for each coefficient.

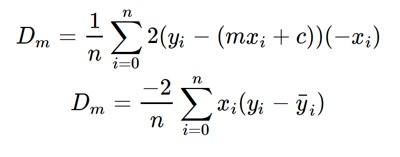
Gradient descent is a very important algorithm to find minima. If you want to understand neutral networks, you have to understand gradient descent.

**Answer: (c)**

Imagine a valley and a person with no sense of direction who wants to get to the bottom of the valley. He goes down the slope and takes large steps when the slope is steep and small steps when the slope is less steep. He decides his next position based on his current position and stops when he gets to the bottom of the valley which was his goal.

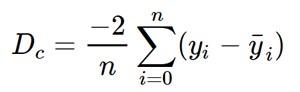
Let’s try applying gradient descent to m and c and approach it step by step:

1. Initially let m = 0 and c = 0. Let L be our learning rate. This controls how much the value of m changes with each step. L could be a small value like 0.0001 for good accuracy.
2. Calculate the partial derivative of the loss function with respect to m, and plug in the current values of x, y, m and c in it to obtain the derivative value D.



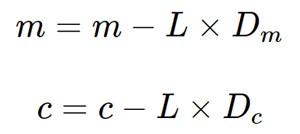
Derivative with respect to m

Dₘ is the value of the partial derivative with respect to m. Similarly lets find the partial derivative with respect to c, Dc :



Derivative with respect to c

3. Now we update the current value of m and c using the following equation:



4. We repeat this process until our loss function is a very small value or ideally 0 (which means 0 error or 100% accuracy). The value of m and c that we are left with now will be the optimum values.

Now going back to our analogy, m can be considered the current position of the person. D is equivalent to the steepness of the slope and L can be the speed with which he moves. Now the new value of m that we calculate using the above equation will be his next position, and L×D will be the size of the steps he will take. When the slope is more steep (D is more) he takes longer steps and when it is less steep (D is less), he takes smaller steps. Finally he arrives at the bottom of the valley which corresponds to our loss = 0.

Now with the optimum value of m and c our model is ready to make predictions !

**Answer: (d)**

Multivariate Regression is a method used to measure the degree at which more than one independent variable (predictors) and more than one dependent variable (responses), are linearly related.

Multiple regression, we mean only one dependent variable with a single distribution or variance. The predictor variables are more than one. To summarize multiple refers to more than one predictor variables but multivariate refers to more than one dependent variables

Example: An agriculture scientist wants to predict the total crop yield expected for the summer. He collected details of the expected amount of rainfall, fertilizers to be used, and soil conditions. By building a Multivariate regression model scientists can predict his crop yield. With the crop yield, the scientist also tries to understand the relationship among the variables.

**Answer: (e)**

• Polynomial Regression is a regression algorithm that models the relationship between a dependent(y) and independent variable(x) as nth degree polynomial. The Polynomial Regression equation is given below:

y= b0+ b1x^1 + b2x^2 + b2x^3 +...... bnx^n

• The dataset used in Polynomial regression for training is of non-linear nature.

• It is also called the special case of Multiple Linear Regression in ML. Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression.

A Polynomial Regression algorithm is also called Polynomial Linear Regression because it does not depend on the variables, instead, it depends on the coefficients, which are arranged in a linear fashion.

Example: Polynomial regression widely applied to predict spread of covid 19 and other.

1. **Logistic Regression**
   1. **Explain logistic regression with example and application.**
   2. **Explain Gradient descent optimization in logistic regression with examples.**
   3. **Explain accuracy, confusion matrix, Precision, recall and f1 score.**
   4. **Explain log loss function and its role in logistic regression.**
   5. **Explain sigmoid function and its role in logistic regression.**

**Answer: (a)**

Logistic regression is used for solving classification problems.

Examples and Applications:

On the basis of the categories, Logistic Regression can be classified into three types:

* Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
* Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
* Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

**Answer: (b)**

Gradient Descent is an iterative optimization algorithm, used to find the minimum value for a function. The general idea is to initialize the parameters to random values, and then take small steps in the direction of the “slope” at each iteration.

Gradient descent is a way to minimize an objective function J(θ) parameterized by a model's parameters θ∈Rd by updating the parameters in the opposite direction of the gradient of the objective function ∇θJ(θ) w.r.t. to the parameters. The learning rate η determines the size of the steps we take to reach a (local) minimum.

example:

In other words, we follow the direction of the slope of the surface created by the objective function downhill until we reach a valley.

**Answer: (c)**

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.



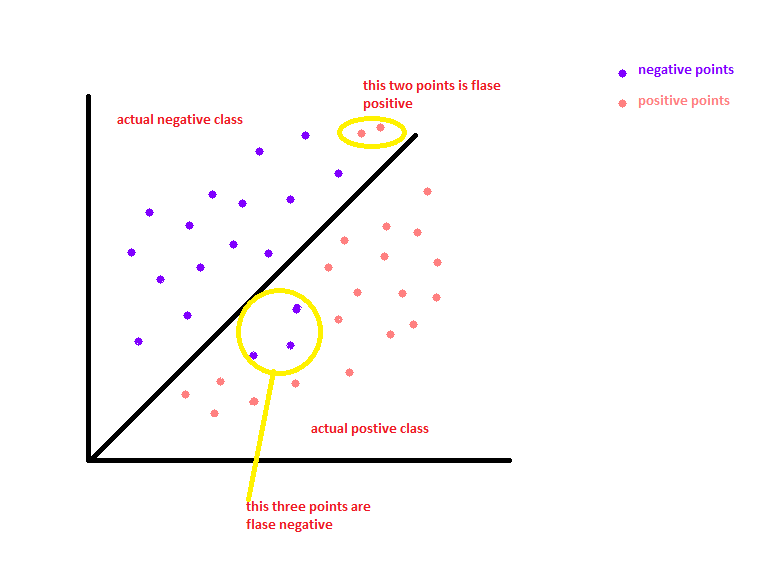
True Positives (TP) - These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes. E.g. if actual class value indicates that this passenger survived and predicted class tells you the same thing.

True Negatives (TN) - These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no. E.g. if actual class says this passenger did not survive and predicted class tells you the same thing.

False positives and false negatives, these values occur when your actual class contradicts with the predicted class.

False Positives (FP) – When actual class is no and predicted class is yes. E.g. if actual class says this passenger did not survive but predicted class tells you that this passenger will survive.

False Negatives (FN) – When actual class is yes but predicted class in no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.



Once you understand these four parameters then we can calculate Accuracy, Precision, Recall and F1 score.

Accuracy - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positives and false negatives are almost the same. Therefore, you have to look at other parameters to evaluate the performance of your model. For our model, we have got 0.803 which means our model is approx. 80% accurate.

Accuracy = TP+TN/TP+FP+FN+TN

Precision - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labeled as survived, how many actually survived? High precision relates to the low false positive rate. We have got 0.788 precision which is pretty good.

Precision = TP/TP+FP

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers is: Of all the passengers that truly survived, how many did we label? We have got recall of 0.631 which is good for this model as it’s above 0.5.

Recall = TP/TP+FN

F1 score - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall. In our case, F1 score is 0.701.

F1 Score = 2\*(Recall \* Precision) / (Recall + Precision) // harmonic mean

**Answer: (d)**

Log Loss is the most important classification metric based on probabilities. It's hard to interpret raw log-loss values, but log-loss is still a good metric for comparing models. For any given problem, a lower log loss value means better predictions.

As we know that Goodness of model in logistic regression can not be defined by

R2\_socre, using the log loss function we get an idea about that particular point and also for the overall model.

**Answer: (e)**

In order to map predicted values to probabilities, we use the Sigmoid function. The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities.

The dataset value which is outlined can cause a large prediction error if we are using simple linear regression but with the help of sigmoid function the outlier problem is solved easily and also improves the accuracy of the model.

**4. Perceptron**

**What is a perceptron?**

**Answer:**

Perceptron is a Machine Learning algorithm for supervised learning of various binary classification tasks. Further, Perceptron is also understood as a neural network unit that helps to detect certain input data computations in business intelligence.

The Perceptron model is also treated as one of the best and simplest types of Artificial Neural networks. However, it is a supervised learning algorithm of binary classifiers. Hence, we can consider it as a single-layer neural network with four main parameters, i.e., input values, weights and Bias, net sum, and an activation function.

**How AND and OR logic gates can be simulated using perceptron learning?**

**Answer:**

The purpose of this answer is NOT to mathematically explain how the neural network updates the weights, but to explain the logic behind how the values are being changed in simple terms.

step 1) Initialize weight values and bias

step 2) Forward Propagate

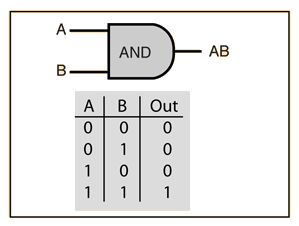
step 3) Check the error

step 4) Backpropagate and Adjust weights and bias

step 5) Repeat for all training examples

AND Gate

From our knowledge of logic gates, we know that an AND logic table is given by the diagram below



The question is, what are the weights and bias for the AND perceptron?

First, we need to understand that the output of an AND gate is 1 only if both inputs (in this case, x1 and x2) are 1. So, following the steps listed above;

case 1)

From w1\*x1+w2\*x2+b, initializing w1, w2, as 1 and b as –1, we get;

x1(1)+x2(1)–1

Passing the first row of the AND logic table (x1=0, x2=0), we get;

0+0–1 = –1

From the Perceptron rule, if Wx+b≤0, then y`=0. Therefore, this case is correct, and no need for Backpropagation.

case 2)

Passing (x1=0 and x2=1), we get;

0+1–1 = 0

From the Perceptron rule, if Wx+b≤0, then y`=0. This row is correct, as the output is 0 for the AND gate.

From the Perceptron rule, this works (for both case 1, case 2 case 3).

case 4)

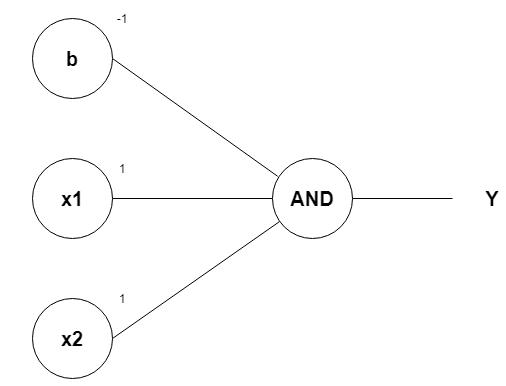
Passing (x1=1 and x2=1), we get;

1+1–1 = 1

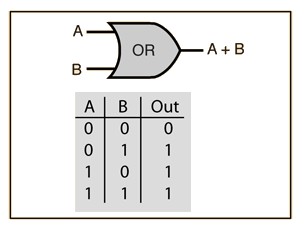
Again, from the perceptron rule, this is still valid.

Therefore, we can conclude that the model to achieve an AND gate, using the Perceptron algorithm is;

x1+x2–1



OR Gate



From the diagram, the OR gate is 0 only if both inputs are 0.

case 1)

From w1x1+w2x2+b, initializing w1, w2, as 1 and b as –1, we get;

x1(1)+x2(1)–1

Passing the first row of the OR logic table (x1=0, x2=0), we get;

0+0–1 = –1

From the Perceptron rule, if Wx+b≤0, then y`=0. Therefore, this row is correct.

case 2)

Passing (x1=0 and x2=1), we get;

0+1–1 = 0

From the Perceptron rule, if Wx+b <= 0, then y`=0. Therefore, this case is incorrect.

So we want values that will make inputs x1=0 and x2=1 give y` a value of 1. If we change w2 to 2, we have;

0+2–1 = 1

From the Perceptron rule, this is correct for both the row 1 and 2.

case 3)

Passing (x1=1 and x2=0), we get;

1+0–1 = 0

From the Perceptron rule, if Wx+b <= 0, then y`=0. Therefore, this case is incorrect.

Since it is similar to that of row 2, we can just change w1 to 2, we have;

2+0–1 = 1

From the Perceptron rule, this is correct for both the row 1, 2 and 3.

case 4)

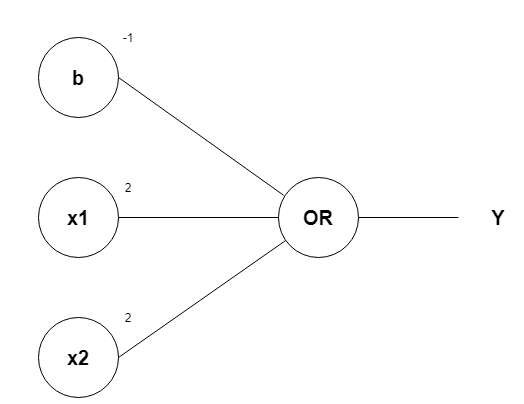
Passing (x1=1 and x2=1), we get;

2+2–1 = 3

Again, from the perceptron rule, this is still valid. Quite Easy!

Therefore, we can conclude that the model to achieve an OR gate, using the Perceptron algorithm is;

2x1+2x2–1



1. **Neural Network**
   1. **What is a neural network? How is it able to learn any function(or Explain piece by piece linear)?**
   2. **Demonstrate working of neural networks with calculations of forward-propagation and back-propagation.**
   3. **Discuss strategies for optimizing deep neural networks.**
   4. **What are applications of neural networks? Explain with references and case studies.**
   5. **Calculate forward and backward propagation on following values for one pass. Draw a neural network diagram for the following values.**

**Input Values: [2, 3], Output value: [1] Initial waits:**

**Input layer waits: w1 = 0.11, w2 = 0.21, w3 = 0.12, w4 = 0.08, Hidden layer waits: w5 = 0.14, w6 = 0.15**

**Ref:**

[**https://www.anotsorandomwalk.com/backpropagation-example-with-numbers-step-by-step/**](https://www.anotsorandomwalk.com/backpropagation-example-with-numbers-step-by-step/)

**Answer: (a)**

Neural networks reflect the behavior of the human brain, allowing computer programs to recognize patterns and solve common problems in the fields of AI, machine learning, and deep learning.

Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

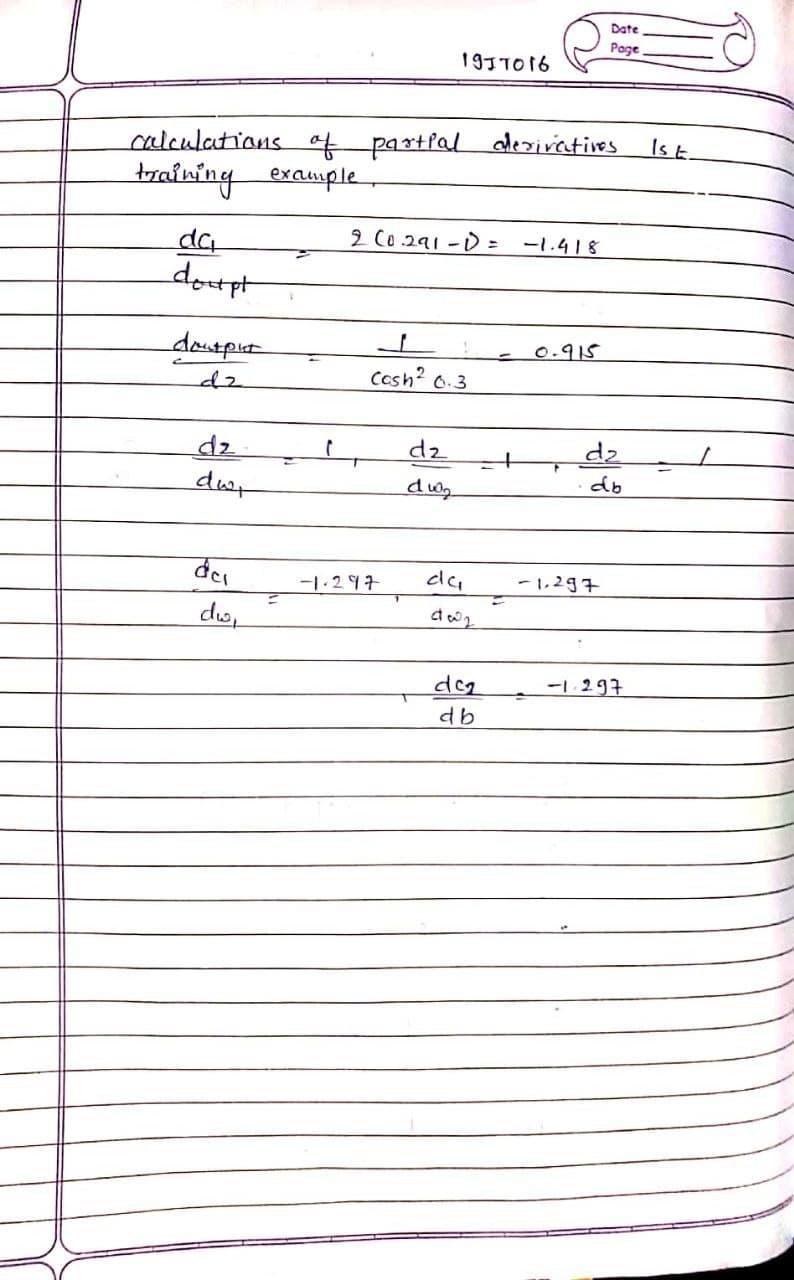
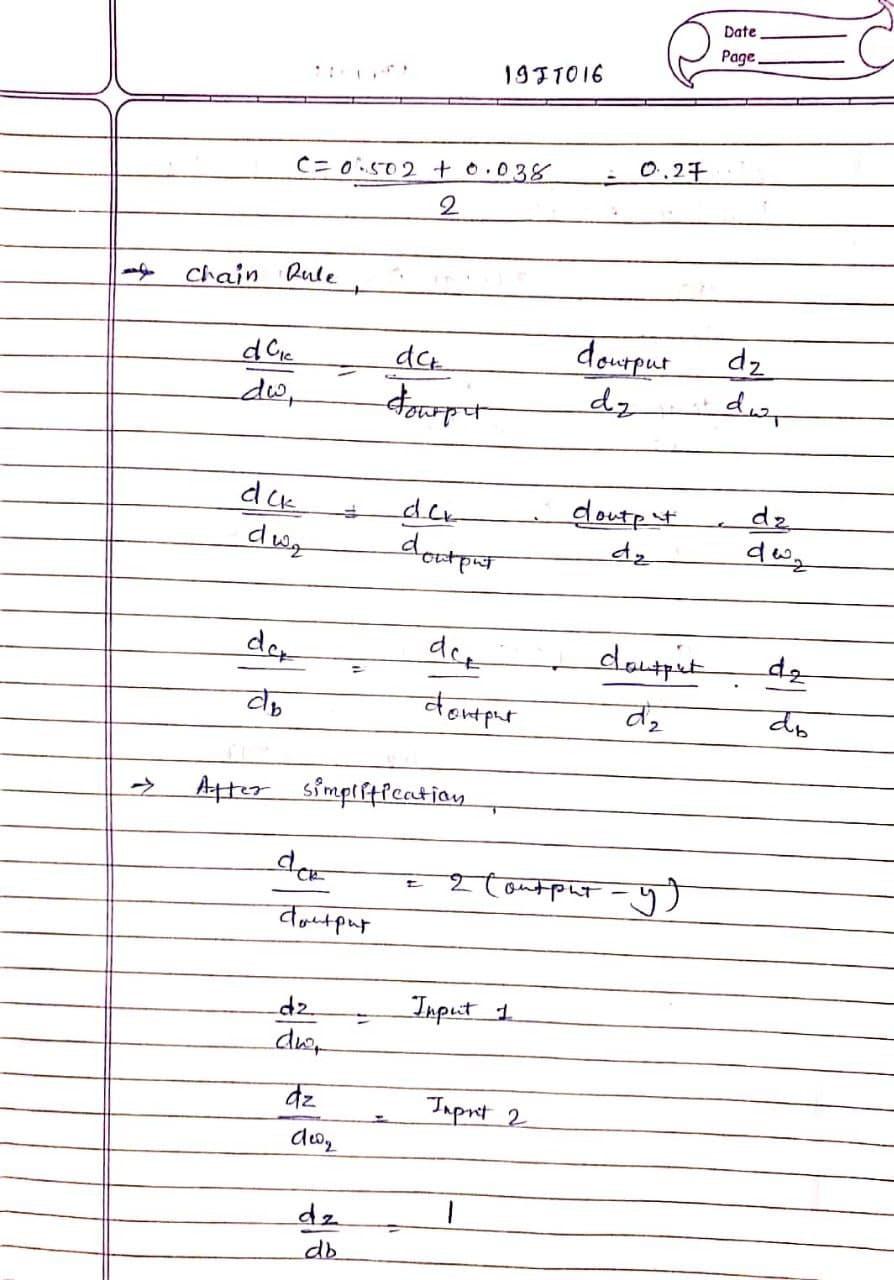
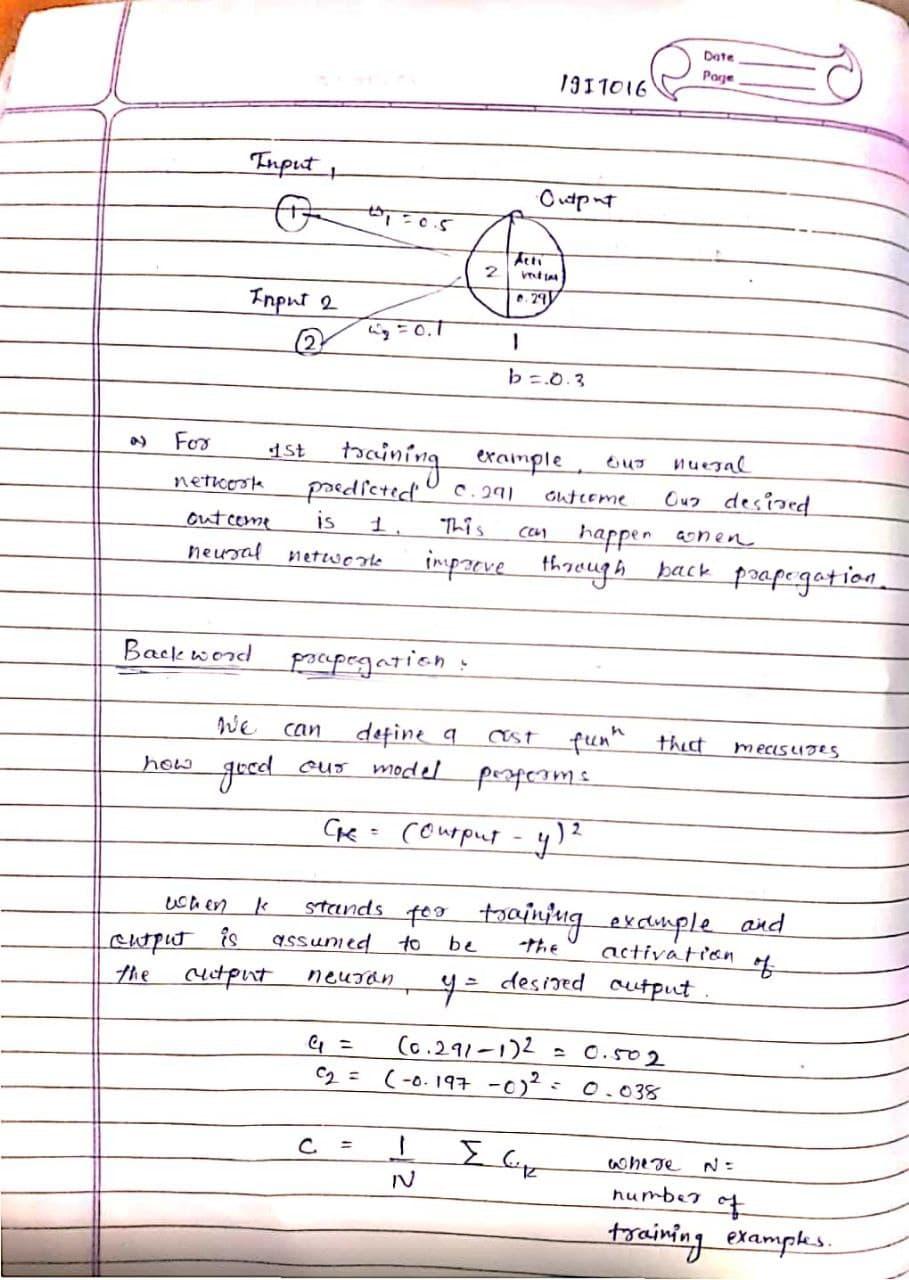
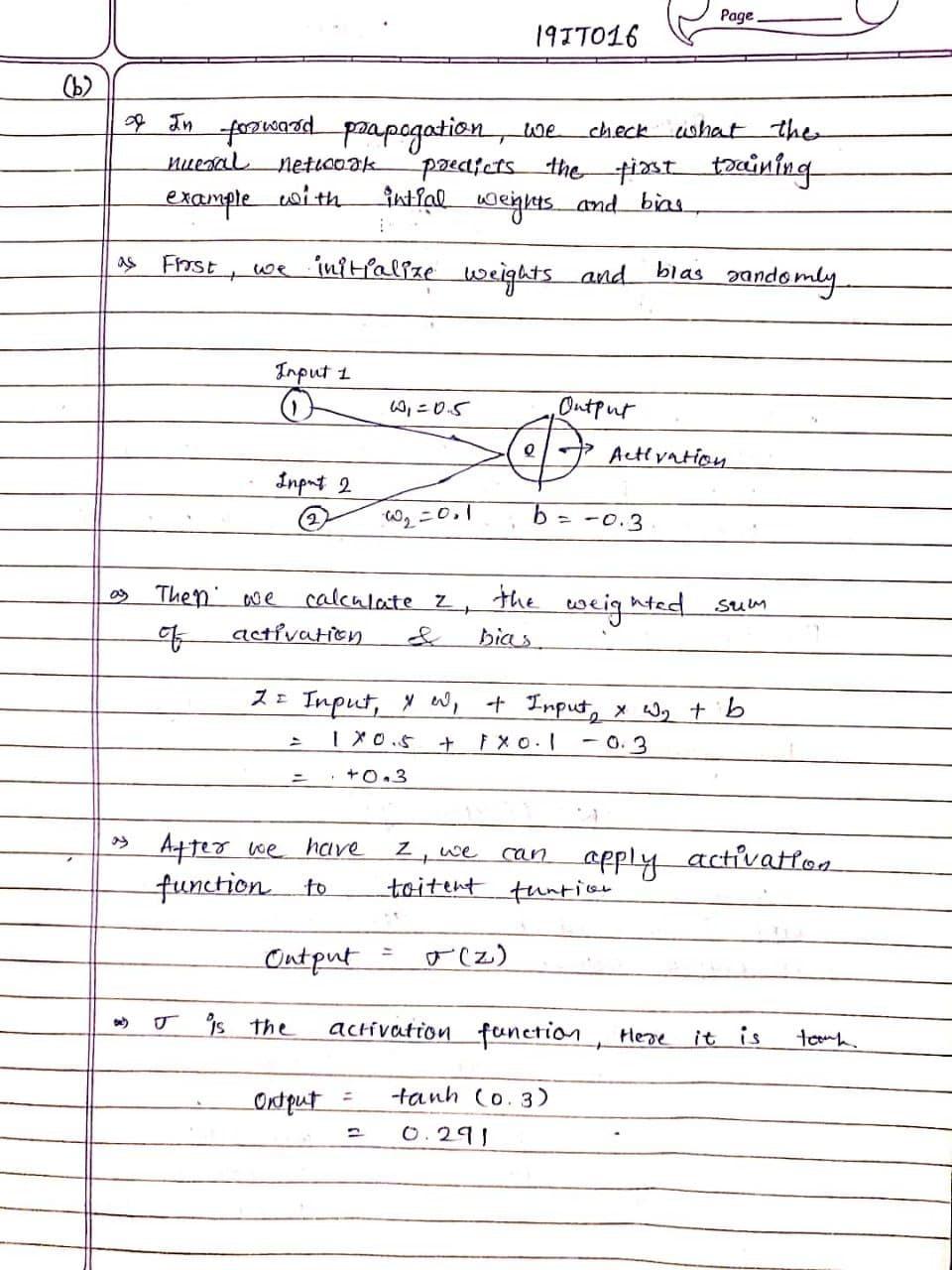
Think of each individual node as its own linear regression model, composed of input data, weights, a bias (or threshold), and an output. The formula would look something like this:

∑wixi + bias = w1x1 + w2x2 + w3x3 + bias

output = f(x) = 1 if ∑w1x1 + b>= 0; 0 if ∑w1x1 + b < 0

Once an input layer is determined, weights are assigned. These weights help determine the importance of any given variable, with larger ones contributing more significantly to the output compared to other inputs. All inputs are then multiplied by their respective weights and then summed. Afterward, the output is passed through an activation function, which determines the output. If that output exceeds a given threshold, it “fires” (or activates) the node, passing data to the next layer in the network. This results in the output of one node becoming in the input of the next node. This process of passing data from one layer to the next layer defines this neural network as a feedforward network.

**Answer: (b)**



**Answer: (c)**

Feature normalization is perhaps the most effective technique in accelerating the optimization of deep networks. Batch Normalization is a clear winner in this paradigm! Batch Normalization also works at a wide selection of learning rates and alleviates the need for explicit network regularization.

Optimization Techniques popularly used in Deep Learning

1. Gradient Descent

Gradient Descent algorithm

A Gradient Descent is an iterative algorithm, that starts from a random point on the function and traverses down its slope in steps until it reaches lowest point of that function. This algorithm is apt for cases where optimal points cannot be found by equating the slope of the function to 0. For the function to reach minimum value, the weights should be altered. With the help of back propagation, loss is transferred from one layer to another and “weights” parameter are also modified depending on loss so that loss can be minimized.

Cost function: θ=θ−α⋅∇J(θ)

2. Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent is an extension of Gradient Descent, where it overcomes some of the disadvantages of Gradient Descent algorithm. SGD tries to overcome the disadvantage of computationally intensive by computing the derivative of one point at a time. Due to this fact, SGD takes more number of iterations compared to GD to reach minimum and also contains some noise when compared to Gradient Descent.

3. Mini Batch — Stochastic Gradient Descent

MB-SGD is an extension of SGD algorithm. It overcomes the time-consuming complexity of SGD by taking a batch of points / subset of points from dataset to compute derivative.

**Answer: (d)**

Speech Recognition

ANNs have been used for speech recognition −

• Multilayer networks

• Multilayer networks with recurrent connections

Character Recognition

• Multilayer neural networks such as Backpropagation neural networks.

Human Face Recognition

• Fully-connected multilayer feed-forward neural network trained with the help of back-propagation algorithm.

• For dimensionality reduction, Principal Component Analysis PCA is used.

Signature Verification Application